UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Análisis de interacciones y actividades en entornos controlados

-PROYECTO FIN DE CARRERA-

Sergio Suja Garrido
Diciembre 2012

Análisis de interacciones y actividades en entornos controlados

Autor: Sergio Suja Garrido
Tutor: Juan Carlos San Miguel Avedillo
Ponente: José María Martínez Sánchez
email: Sergio.Suja@estudiante.uam.es, Juancarlos.Sanmiguel@uam.es, JoseM.Martinez@uam.es

Video Processing and Understanding Lab
Departamento de Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Diciembre 2012

Trabajo parcialmente financiado por el gobierno español bajo el proyecto TEC2011-25995 (EventVideo)

Resumen

En este trabajo, se propone contribuir al estado del arte en el campo del reconocimiento automático de eventos en entornos muy controlados tales como salas de reuniones o espacios interiores. El objetivo es proponer nuevos métodos para mejorar el prototipo desarrollado por el Grupo de Tratamiento e Interpretación de Vídeo de la Universidad Autónoma de Madrid (VPU-UAM) en el marco de la competición internacional de detección de eventos relacionados con personas ICPR - HARL 2012.

En primer lugar, se realiza un estudio exhaustivo del estado del arte relacionado y del prototipo disponible. Después, el trabajo se centra en la etapa de extracción de características, y se proponen e implementan diversos métodos para detección de piel humana en imágenes. Posteriormente, se desarrolla un método de detección de eventos determinista basado en máquinas de estados finitos (FSMs) con el que se modelan los eventos de interés (interacciones persona-objeto y actividades entre dos personas) basándose, entre otras casacterísticas, en información de piel. Finalmente, se evalua el sistema modificado sobre los datasets disponibles y se presentan los resultados a la competición internacional ICPR - HARL 2012.

Palabras Clave

Vídeo análisis, reconocimiento de eventos, actividades, interacciones, extracción de características, detección de piel, espacios de color, modelado de eventos, entornos controlados.

Abstract

In this work, we propose to contribute to the state of art in the field of automatic event recognition focusing on the analysis of interactions and activities of persons in highly controlled environments such as meeting rooms or indoor video-surveillance. The goal is to improve the prototype developed by the Video Processing and Understanding Lab of the Universidad Autónoma de Madrid (VPU-UAM) considering the framework of the international competition ICPR - HARL 2012 for human-related events.

First, the related state of art and the available prototype are studied. Then, feature extraction is explored for the task of skin detection where different methods are proposed and implemented. Later, a deterministic method for event detection based on finite-state-machines (FSMs) is developed for the competition events (person-object interactions and activities for two persons) by analyzing, among others, skin regions of persons. Finally, such improvements are included in the initial prototype, which is evaluated on available datasets and the results are presented to the international competition ICPR - HARL 2012.

Keywords

Video analysis, event recognition, activity, interaction, feature extraction, skin detection, color spaces, event modeling, controlled environments.

Agradecimientos

En primer lugar, agradecer a mi tutor Juan Carlos todo el apoyo que me ha dado y la dedicación que ha demostrado hacia mí a lo largo del proyecto, así como por su paciencia y por todo lo que me ha ayudado, aconsejado y enseñado.

A mis profesores, Jesús Bescós y José María Martínez, por haberme brindado la oportunidad de realizar este proyecto en el VPU-Lab, así como la ayuda prestada durante la realización del mismo. También a los miembros del laboratorio por crear tan buen ambiente y amenizar las horas de trabajo. Así mismo, gracias a todos los profesores que he tenido a lo largo de la carrera por la formación que me han dado.

A mis compañeros de la carrera, a todos ellos, por estar siempre ahí y por compartir conmigo todo el camino. Sois muchos, pero todos formáis de una forma u otra parte de esto. Entre ellos, quiero dar las gracias a parte a ciertas personas que han sido muy especiales para mí. A Pascu, mi eterno compañero, por estar siempre a mi lado en todo y durante toda la carrera, apoyarme siempre y aguantarme, que no es poco. A Marta, por ser un constante apoyo durante todo el camino y compartir conmigo tantas cosas tanto dentro como fuera de la universidad. A Javi, por su generosidad y por no dejar que me durmiera solo en todas las clases. A Casas y a Brande, por ser como son y hacerme pasar siempre tan buenos ratos, así como animarme día a día a seguir adelante.

A mis amigos, porque es imposible tener unos mejores. Por haber compartido conmigo grandes momentos en mi vida y no fallarme nunca en los malos momentos.

A mi hermano Jorge, por su ayuda en todo y su enorme paciencia, y a mi hermana Irene, por animarme siempre cada día y hacerme sentir orgulloso.

Y finalmente a mis padres, por hacerme ser quien soy, por confiar siempre en mí, por su cariño, sus consejos y su apoyo incondicional.

Sergio Suja Garrido
Diciembre 2012

Índice general

Índice de figuras

Índice de tablas

Acrónimos

BBOX *Bounding Box*: Área definida por un punto de referencia, una altura y una anchura.

BLOB *Binary Large OBject:* Elementos utilizados en las bases de datos para almacenar datos de gran tamaño que cambian de forma dinámica. En este proyecto, conjunto de píxeles de una imágen que conforman una región.

BN *Bayesian Networks:* Modelo probabilístico gráfico que representa un conjunto de variables aleatorias y sus dependencias.

DBN *Dynamic Bayesian Networks:* BN que representa secuencias de variables.

FSM *Finite State Machine:* Modelo matemático de computación usado para diseñar programas y circuitos lógicos secuenciales.

HMM *Hidden Markov Model:* Tipo de modelo estadístico de Markov con estados ocultos (sin observar).

HSV *Hue Saturation Value Color Model:* Modelo de color obtenido de una transformación no lineal del espacio de color RGB.

LDS *Linear Dynamical Systems:* Tipo especial de sistemas dinámicos dondo la ecuación que obierna el sistema es lineal.

PN *Petri Net:* Lenguaje de modelado matemático para descripción de sistemas distribuidos.

RGB *Red Green Blue Color Model:* Modelo que representa un color mediante la mezcla por adición de los tres colores primarios: rojo, verde y azul.

ROI *Region Of Interest:* Subconjunto de muestras de una imagen (área) para un propósito en particular.

YCbCr *YCbCr Color Space:* Modelo para representar el color basado en tres componentes. *Y* es la luminancia, *Cb* y *Cr* son las componentes cromáticas de diferencia de azul y diferencia de rojo.

Capítulo 1

Introducción

1.1. Motivación

En la actualidad, dentro del campo del análisis de secuencias de video, está ganando gran importancia el reconocimiento automático de eventos [1]. Esto se debe a la gran cantidad de usos que tiene en vídeo-vigilancia, anotación automática de vídeo e interfaces persona-ordenador, entre otras aplicaciones. De especial importancia es su uso en vídeo-vigilancia, ya que permite dotar a los sistemas de análisis de secuencias de vídeo-vigilancia de automatismos capaces de detectar dichos eventos sin necesidad de personal de monitorización. La figura 1.1 muestra algunos ejemplos de aplicaciones.

Es importante aclarar desde el comienzo los dos tipos principales de eventos bajo estudio, que son: interacciones y actividades. El primero se refiere a las acciones llevadas a cabo entre varias personas o entre personas y objetos del escenario (e.g., darse la mano, coger un objeto). El segundo tipo de eventos se refiere a aquellas acciones que no involucran ningún tipo de interacción física entre varias personas u objetos (e.g., andar, correr, entrar en una habitación).

También es importante en este punto explicar como se clasifican los escenarios habitualmente,

Figura 1.1: Ejemplo de aplicaciones del análisis de secuencias de vídeo
(Figuras extraídas de [2], [1] y [3])

Figura 1.2: Ejemplos de escenario

de manera que quede claro que tipo de escenarios se tratan en este proyecto. Se pueden clasificar de dos formas:

- Controlados vs No controlados: En los primeros tenemos información adicional respecto a la que pueda darnos el propio escenario (objetos fijos en la escena, posibles eventos, etc), mientras que en los segundos no.

- Concurridos vs No concurridos: Depende exclusivamente del número de objetos en movimiento en escena (personas, coches, etc). Un gran número dificulta las tareas de análisis.

En la figura 1.2 aparecen dos ejemplos: el primero es un escenario no controlado y concurrido, el segundo es un escenario controlado y no concurrido.

Las técnicas desarrolladas hasta ahora para detección de eventos, aunque resuelven en cierta medida algunos problemas que van surgiendo, no consiguen funcionar para todos los casos que pueden darse en un entorno real. Además, se centran principalmente en escenarios donde ocurran pocos eventos, haya pocas personas u objetos implicados y los eventos se ejecuten de manera similar, es decir, escenarios controlados, ya que para entornos altamente concurridos ninguna técnica consigue resultados aceptables actualmente.

La mayoría de los sistemas desarrollados para la detección de eventos, así como el prototipo del VPU-Lab [3][4], parten de la misma base y siguen una serie de etapas de análisis de vídeo comunes:

1) Segmentación de regiones de interés: Esta etapa busca extraer de la imagen aquellas regiones que se consideran claves para detectar el evento en la secuencia de vídeo (e.g., segmentación de primer plano (frente) de aquello considerado como fondo [3][4]).

2) Seguimiento de regiones de interés: Una vez obtenidas las regiones de interés, en esta etapa se realizará un seguimiento de las mismas, ya que cualquier evento resultará de alguna actividad o interacción relacionada con el movimiento de dichas regiones.

3) Extracción de características: En esta etapa se calcularán dichas características para todas las regiones de interés detectadas en la etapa anterior (e.g., velocidad [3], detección de personas [4]). Posteriormente, las características extraídas son utilizadas en el reconocimiento de eventos.

4) Reconocimiento de eventos: Por último, se utilizarán distintas técnicas para combinar las características extraídas con el objetivo de buscar similitudes entre los datos observados en la escena (posibles eventos) y los modelos definidos para los eventos de interés.

Los problemas surgen principalmente en la primera etapa, debido a la propia naturaleza de los vídeos, donde se dan infinitas situaciones que dificultan enormemente la correcta detección de los objetos de interés en escena [2]. Por tanto, esta es la etapa en la que más se centra la investigación en este campo. También el reconocimiento final de eventos presenta grandes dificultades debido a múltiples problemas, como son la efectividad de las técnicas de análisis que componen el sistema, la disponibilidad limitada de datos de entrenamiento, la apariencia similar de diferentes eventos, el modelado de eventos complejos y el punto de vista.

Todo esto ha generado un gran número de técnicas [1][2][5] que pretenden solucionar los problemas que van surgiendo en dichas etapas, pero debido a la gran variedad y a lo específicas que son, aún es difícil incorporar dichas técnicas a cualquier sistema de detección de eventos. En concreto, se ha observado que muchas de las aproximaciones existentes están limitadas a eventos con baja complejidad (e.g., datos precisos, vídeos con una sola persona realizando la acción) y que además requieren grandes cantidades de datos de entrenamiento. En esta situación, su uso no es posible en escenarios con alta densidad de personas donde la cantidad de datos de entrenamiento es escasa y no es generalizable. Adicionalmente, la alta complejidad computacional de muchas aproximaciones existentes limita su uso en aplicaciones en tiempo real.

Una vez explicado todo esto, se puede decir que la principal motivación de este proyecto reside en la necesidad de mejorar el estado actual del arte en el campo del reconocimiento automático de eventos y contribuir en el análisis de interacciones y actividades en entornos muy controlados tales como salas de reuniones o espacios interiores.

1.2. Objetivos

El objetivo del proyecto consistirá en mejorar el prototipo para detección automática de eventos existente en el Grupo de Tratamiento e Interpretación de Vídeo de la Universidad Autónoma de Madrid (VPU-UAM) [3][4]. Se realizará un estudio de las técnicas actuales en el campo del reconocimiento de eventos, y posteriormente se propondrán mejoras al prototipo y se implementarán aquellas que sean más interesantes. El objetivo principal perseguido es obtener un sistema que opere en tiempo real y que necesite la menor cantidad posible de datos de entrenamiento.

Adicionalmente, este proyecto se realizará en el marco de la competición internacional ICPR - HARL 2012 [6]. Su objetivo es la localización y reconocimiento de actividades humanas (e.g., coger/dejar objetos, escribir con un teclado, abrir una puerta) en entornos controlados con un número variable de personas (1-5). El objetivo de la participación es doble. Por un lado, se obtendrá material adecuado para la detección de eventos seleccionados por la organización. De esta manera, se realizarán las mejoras convenientes sobre el prototipo con el fin de obtener los mejores resultados sobre el dataset otorgado por la organización. Por otro lado, la participación en la competición proporcionará una evaluación directa de las mejoras obtenidas con respecto a otros organismos internacionales involucrados en este área.

Por tanto, el objetivo mencionado puede ser desglosado en varios objetivos generales:

1) Estudio del estado del arte

Antes de cualquier proyecto hay que realizar una revisión del estado del arte en el campo tratado. En este caso, habrá que centrar dicha revisión en el campo de reconocimiento automático de eventos. En especial, se realizará un estudio de los eventos propuestos por la competición HARL 2012 en la literatura, para ver cómo afrontan otros investigadores los problemas que acarrea cada evento. Además, se realizará un estudio exhaustivo del prototipo actual del VPU-Lab, base sobre la que desarrollar cualquier mejora, para conocer todas las limitaciones de este sistema antes de implementar nuevas funciones.

2) Extracción de características

Tras la revisión del estado del arte, se propondrán mejoras en la extracción de características. En especial, el objetivo principal será mejorar el sistema de detección de piel.

3) Modelado e implementación de nuevos eventos

Se realiza un estudio de los eventos propuestos por la competición, se describen y se modelan. Una vez hecho esto, se realiza la implementación de estos eventos sobre el prototipo utilizando las mejoras realizadas sobre la extracción de características.

4) Evaluación de mejoras

En primer lugar se realizará un análisis de todo el material disponible para este proyecto, es decir, de los datasets públicos disponibles y del dataset proporcionado por la organización de la competición HARL 2012. Además, se propone generar un nuevo dataset para este proyecto. A partir de todo ese material, se compararán los resultados obtenidos con los referentes al estado del arte, de manera que se pueda comprobar si las mejoras realizadas proporcionan mejores resultados.

1.3. Organización de la memoria

El documento consta de los siguientes capítulos:

- Capítulo 1: Introducción, motivación del proyecto y objetivos.
- Capítulo 2: Estado del arte en sistemas de reconocimiento automático de eventos, tipos de aproximaciones, aproximaciones existentes y datasets utilizados.
- Capítulo 3: Estudio del prototipo del VPU-Lab para la detección de eventos y sus limitaciones.
- Capítulo 4: Estudio de los espacios de colores más usados y de varios algoritmos para la detección de piel.
- Capítulo 5: Descripción de los eventos propuestos y modelado de los mismos mediante FSMs. Problemas surgidos.
- Capítulo 6: Evaluación y comparativa de los detectores de piel propuestos y evaluación de las mejoras propuestas en reconocimiento de eventos. Incluye una descripcion de los datos utilizados, resultados sobre los mismos y resultados proporcionados por HARL 2012.
- Capítulo 7: Resumen, conclusiones finales y trabajo futuro.

Al final del documento varios apéndices describen algunos aspectos del proyecto más en detalle:

- Apéndice A: Pruebas con un algoritmo de detección de puertas en entornos controlados.
- Apéndice B: Breve descripción del dataset SSG.
- Apéndice C: Implementación de eventos mediante FSMs.
- Apéndice D: Tablas de resultados de los algoritmos probados para detección de piel.
- Apéndice E: Participantes, métricas utilizadas en HARL 2012 y resultados completos de la evaluación de los sistemas desarrollados.

Capítulo 2

Estado del arte

2.1. Introducción

Durante estos últimos años, la investigación en el campo del análisis de secuencias de vídeo para reconocimiento de eventos se ha incrementado mucho, generándose así gran cantidad de proyectos destinados a modelar y reconocer eventos y a solucionar los numerosos problemas asociados. En consecuencia, todos estos trabajos pueden clasificarse de distintas maneras en función de las técnicas que utilizan.

En este capítulo se dará un breve resumen de los principales tipos de aproximaciones existentes hoy en día, para después centrarse en los eventos dedicados a las interacciones persona-objeto y persona-persona: que características extraen de los vídeos, como modelan los eventos a partir de estas características, etc. Por último, describiremos los principales datasets relacionados con este tipo de eventos.

Este capítulo se estructura de la siguiente forma: aproximaciones genéricas de modelado y reconocimiento de eventos (sección 2.2), reconocimiento de interacciones persona-objeto y persona-persona (sección 2.3) y datasets (sección 2.4).

2.2. Aproximaciones genéricas de modelado y reconocimiento de eventos

Todas las aproximaciones desarrolladas en la actualidad son muy diferentes unas de otras, pero en general, presentan tambien varias similitudes, ya que la mayoría provienen de técnicas ampliamente conocidas en reconocimiento de patrones. La estructura de las mismas y la forma de operar hace que todas ellas puedan ser organizadas en grupos. Así, los tres criterios principales de clasificación son: según la existencia o no de entrenamiento previo (probabilistas vs deterministas), según si están destinadas a tratar acciones simples o actividades complejas, y

según su estructura y complejidad (monocapa vs jerárquicas), relacionada normalmente con la complejidad de los eventos que se desea tratar.

En este apartado vamos a definir muy brevemente los distintos tipos de aproximaciones para tener una idea de que puntos de vista se utilizan a la hora se afrontan los retos que presenta la detección de eventos.

2.2.1. Aproximaciones probabilistas vs deterministas

Aunque esta clasificación no aparece explícitamente en ningún documento que trate el tema de los eventos, todos los estudios realizados sobre aproximaciones siempre especifican para cada una si depende de entrenamiento o no ([1], [2]). En [7], aunque utilizan otra clasificación, sí utilizan los términos "determinista" y "probabilista" al describir algunas aproximaciones. Por tanto, todas las aproximaciones existentes para modelado y reconocimiento de eventos se pueden reunir, en función de esto, en dos grandes grupos: probabilistas y deterministas.

- Las aproximaciones probabilistas tienen en cuenta la incertidumbre del análisis a bajo nivel y su estructura puede ser aprendida de datos de entrenamiento o definida explícitamente. Según los datos de entrenamiento, la estructura modelará de una forma u otra cada evento. Estos métodos no aseguran que para una misma entrada siempre se proporcione la misma salida, debido a la incertidumbre antes comentada. Algunos ejemplos son las *Bayesian Networks* (BNs), los *Hidden Markov Models* (HMMs) las *Dynamic Bayesian Networks* (DBNs) [7], que requieren grandes cantidades de datos de entrenamiento para generar los modelos de los eventos automáticamente.

- Las aproximaciones deterministas no tienen en cuenta la incertidumbre del análisis a bajo nivel y su estructura es especificada completamente por el desarrollador. Estos métodos definen una serie de reglas generales para detectar el evento de interés y no realizan ninguna sugerencia acerca de las técnicas de análisis requeridas para poder aplicar dichas reglas. Por tanto, estos métodos aseguran que para una misma entrada siempre se proporcione la misma salida, ya que no tienen en cuenta la incertidumbre antes mencionada. Algunos ejemplos son las *Finite-State-Machines* (FSMs), que se suelen usar para modelar las actividades más simples como secuencias de estados ordenados y completamente observables, y las *Petri Nets* (PNs), que permiten coordinar actividades múltiples y modelar relaciones como secuencias, concurrencia y sincronización.

Normalmente, las aproximaciones deterministas se aplican para detectar acciones simples (e.g. detectar posturas), mientras que las probabilistas se utilizan mayoritáriamente para acciones complejas que requieran una duración para su realización (e.g. pasos de un baile consecutivos). Aun así, aproximaciones probabilistas aplicadas sobre acciones simples dan bastantes buenos resultados, generalmente mejores que las deterministas.

Ahora bien, es importante destacar que las aproximaciones probabilistas dependen de una etapa de entrenamiento previa, y que debe ir actualizándose. Esta etapa de entrenamiento debe ser lo suficientemente genérica para que los modelos sean lo más fiables posibles y representen lo mejor posible aquello que describen, por lo tanto, requieren de una gran cantidad de datos para su entrenamiento. Además, son computacionalmente mucho mas costosas que las deterministas, y por tanto, mucho más lentas, lo cual no es favorable a la hora de aplicar dichos algoritmos en aplicaciones en tiempo real.

2.2.2. Aproximaciones para acciones simples y actividades complejas

En [2] describen numerosas aproximaciones para la detección de eventos, utilizando una clasificación basada en si están destinadas a detectar acciones simples o actividades complejas.

2.2.2.1. Aproximaciones para acciones simples

Para el grupo de aproximaciones para acciones simples, hablan de tres tipos de aproximaciones: no-paramétricas, volumétricas y paramétricas. Las primeras las definen como aproximaciones que extraen características de cada *frame* del vídeo, de manera que si dichas características concuerdan con las especificadas para un evento (reglas), éste será detectado. Por otro lado, las aproximaciones volumétricas no extraen características *frame* a *frame*, sino que consideran el vídeo como un volumen 3D de píxeles (figura 2.1), extrayendo características de dicho volumen y analizando su carácter temporal. Por último, las aproximaciones paramétricas buscan concordancias en el tiempo entre la acción realizada en el vídeo y un modelo de dicha acción entrenado mediante datos de entrenamiento obtenidos de otros vídeos en los que se realice dicha acción. Algunos ejemplos de estas técnicas son los *Hidden Markov Models* (HMMs) y *Linear Dynamical Systems* (LDSs) [2].

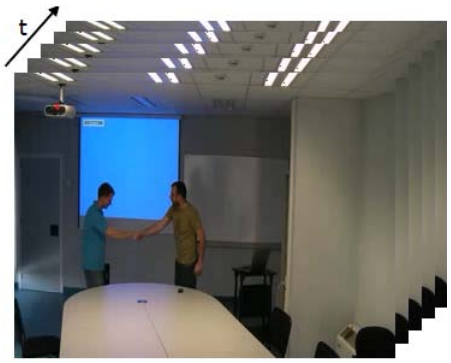


Figura 2.1: Ejemplo de volumen 3D (XYT)

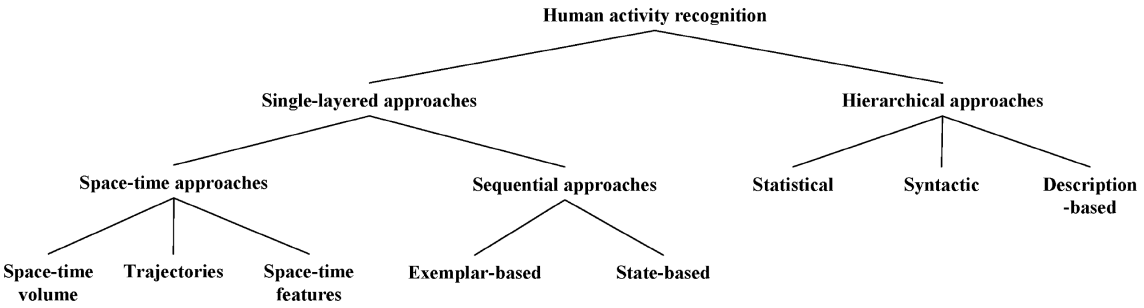


Figura 2.2: Clasificación de aproximaciones para reconocimiento de actividades humanas (Figura extraída de [1])

2.2.2.2. Aproximaciones para actividades complejas

Las aproximaciones anteriores se preocupan en su mayoría por el modelado y reconocimiento de acciones simples de un solo actor. Las aproximaciones englobadas en este grupo buscan modelar actividades de mayor complejidad, que requieren un nivel más alto de representación y razonamiento.

Este grupo de aproximaciones para actividades complejas se divide en tres tipos: aproximaciones por modelos gráficos, sintácticas y basadas en conocimiento. Sin entrar al detalle en este tipo de aproximaciones, hay que destacar que la mayoría de ellas resultan de la combinación o modificación de muchas de las aproximaciones desarrolladas para acciones simples. Algunos ejemplos de estas aproximaciones son las *Dynamic Bayesian Networks* (DBNs) y *Petri Nets* (PNs) [2], las cuales se engloban en el subconjunto de aproximaciones por modelos gráficos. Así mismo, algunas aproximaciones sintácticas utilizan en sus niveles más bajos HMMs y *Bayesian Networks* (BNs) [2].

2.2.3. Aproximaciones monocapa vs jerárquicas

Otra forma de clasificar las aproximaciones para detección de eventos es en función de su complejidad. En [1] desarrollan esta idea para clasificar las aproximaciones, creando dos grandes grupos: aproximaciones monocapa y aproximaciones jerárquicas. En la figura 2.2 se puede observar como clasifican todas las aproximaciones, en este caso sin hacer distinción entre si usan entenamiento o no, sino más bien a nivel de complejidad en la primera división, y luego en función de las técnicas utilizadas en el resto de niveles. Esta clasificación, más actual que la presentada en [2], presenta aún así grandes similitudes con ella.

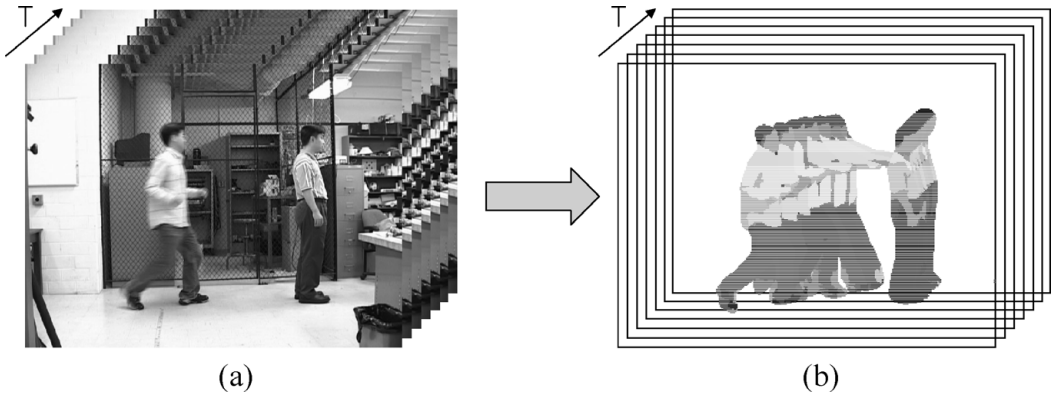


Figura 2.3: Ejemplo de un modelo construido como volumen XYT
El modelo de la interacción se contruye a partir de (a) imagenes originales y (b) blobs del frente (Figura extraída de [1])

2.2.3.1. Aproximaciones monocapa

Las aproximaciones monocapa se denominan así debido a que el evento se detecta en una sóla etapa, directamente considerando los datos del vídeo como un todo. Estas aproximaciones consideran un evento como una secuencia de imágenes particular con ciertas características, de manera que si una secuencia cualquiera de un vídeo cumple con dichas características, el evento será detectado. Se pueden dividir en dos tipos: espacio-temporales y secuenciales.

Las aproximaciones espacio-temporales consideran cada vídeo como un volumen 3D (XYT) y establecen una serie de condiciones que debe cumplir cada uno a lo largo del tiempo (e.g. presencia de ciertas características, flujos de movimiento, gradientes). Una típica aproximación espacio-temporal consiste en construir un modelo como un volumen 3D (XYT) de cada actividad humana, a partir de los vídeos de entrenamiento [8]. Para un vídeo dado, se construye un volumen 3D, se compara con el modelo anterior y se mide la similitud en contorno y apariencia. Al final, el sistema deduce de esa comparación si el nuevo vídeo presenta dicha actividad. Un ejemplo de un modelo construido a partir de una secuencia es el mostrado en la figura 2.3.

En cambio, las aproximaciones secuenciales interpretan los vídeos como una secuencia de observaciones, de características. Primero convierten las secuencias de vídeo en vectores de características, extrayendo éstas a partir del estado de los blobs en cada *frame*. A partir de datos de entrenamiento, pero no necesariamente entrenando un modelo, se estudian como son los vectores de características para cada actividad. Si un vector de características en otro vídeo es similar a un vector típico de alguna actividad, el sistema decidirá que la actividad se ha producido. Estas aproximaciones se pueden realizar mediante plantillas de las actividades obtenidas a partir de ejemplos o mediante modelos de estados (e.g. FSMs, HMMs, DBNs, PNs). Estas aproximaciones se utilizan normalmente para analizar movimientos humanos secuenciales simples, e.g *andar*, *saltar*, *saludar*. La figura 2.4 muestra un ejemplo de HMM que modela la acción de *estirar brazo*.

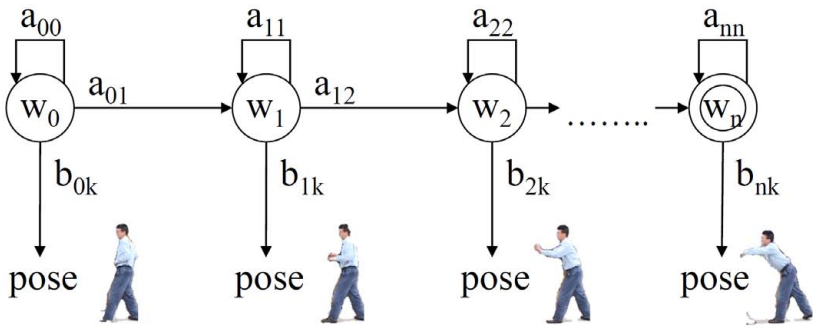


Figura 2.4: Ejemplo de HMM para modelar una actividad
(Figura extraída de [1])

2.2.3.2. Aproximaciones jerárquicas

Las aproximaciones jerárquicas buscan reconocer actividades complejas, estableciendo que estas siempre se pueden subdividir en actividades más simples. Si es capaz de reconocer estas subactividades (o subeventos), debería ser relativamente sencillo reconocer las actividades más complejas. Comparándolas con las monocapa, se podría decir que son aproximaciones multicapa. Por tanto, la mayor ventaja de este tipo de aproximaciones es su habilidad para reconocer actividades con estructuras complejas a partir de actividades más simples. Además, son perfectamente compatibles con un análisis a nivel semántico de las interacciones entre humanos y/o objetos. Todo esto permite, en primer lugar, reducir el entrenamiento necesario, ya que no se entrenarán modelos para las actividades complejas, sino sólo para las simples. Además, permiten incorporar conocimientos humanos previos sobre dicha interacción en su representación. Existen tres tipos principales de aproximaciones jerárquicas: sintácticas, estadísticas y descriptivas.

Las primeras desarrollan una grámatica especial con una serie de reglas, donde cada símbolo representa una accion simple (atómica) y se establece que las actividades complejas son cadenas de símbolos. Así, si un conjunto de símbolos forman un cadena conocida cumpliendo las reglas, se detectará el evento correspondiente a dicha cadena. En la figura 2.5 se presenta un ejemplo de producción de reglas para representar y reconocer una interacción.

```
Fighting  ->  Punching                 : 0.3
          |   Punching Fighting        : 0.7

Punching  ->  stretch withdraw         : 0.8
          |   stretch stay_withdrawn    : 0.1
          |   stay_stretched withdraw   : 0.1
```

Figura 2.5: Ejemplo de reglas
(Figura extraída de [1])

Las estadísticas simplemente utilizan modelos secuenciales, como en las aproximaciones monocapa (e.g. HMMs, DBNs), pero con varios niveles, convirtiendo así acciones complejas en un conjunto de acciones atómicas. Un ejemplo de esto se muestra en la figura 2.6, donde la actividad

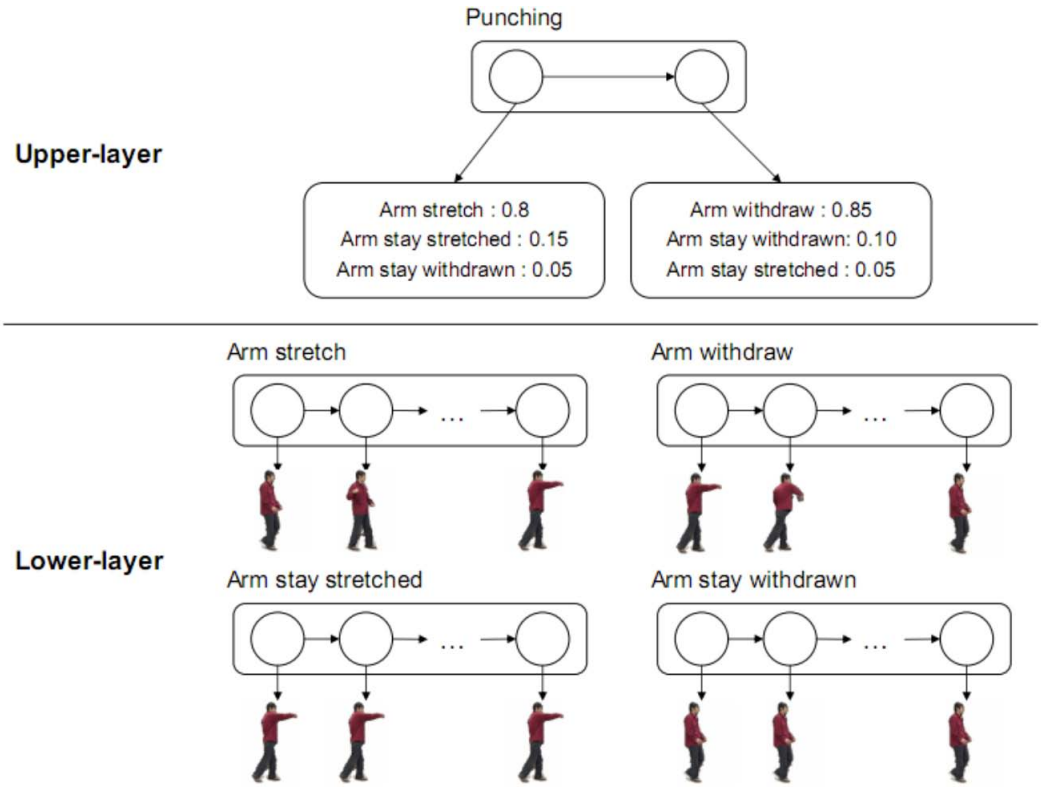


Figura 2.6: Ejemplo de modelo mediante dos capas de HMMs
(Figura extraída de [1])

golpear se modela a partir de dos capas de HMMs.

Por último, las descriptivas representan una actividad compleja mediante actividades simples, describiendo sus relaciones temporales, espaciales y lógicas. Si una actividad cumple ciertas relaciones, el evento será detectado.

2.3. Reconocimiento de actividades e interacciones persona-objeto y persona-persona

Uno de los objetivos de este proyecto es mejorar el prototipo existente para reconocer los eventos propuestos por la organización de la competición ICPR - HARL 2012, eventos correspondientes a actividades o interacciones como pueden ser coger un objeto, dar un objeto a otra persona, teclear o entrar por una puerta. En la literatura existen numerosos trabajos donde tratan estos eventos y otros muchos similares a los propuestos. Por tanto, antes de volcarse en el modelado y desarrollo de los eventos propuestos, conviene repasar dichos trabajos y estudiar cuáles son las técnicas más populares para abordar esta tarea, así como para descubrir cuáles son los problemas más comunes a día de hoy y buscar soluciones viables.

2.3.1. Extracción de características e información contextual

En primer lugar, y cómo ya se ha explicado en apartados anteriores (sección 1.1), una etapa que siempre está presente en todo sistema de reconocimiento de eventos es la extracción de características, entendiendo por característica cualquier tipo de información que podamos extraer del vídeo, pudiendo ser tanto información de imagen como de audio. Antes de continuar, conviene destacar que este proyecto pretende realizar el reconocimiento de eventos a partir de información visual únicamente, dejando de lado la información de audio.

Esta etapa de extracción de características varía mucho de un sistema a otro, y al igual que existen muchas características que suelen ser comunes a todos los eventos, también hay algunas que son muy específicas para cada evento. Por tanto, si damos un repaso a varios trabajos de investigación descubriremos que existen muchas maneras de obtener características útiles para el reconocimiento.

Es importante aclarar antes que las características se pueden obtener a partir de puntos de cada *frame* del vídeo o de regiones más grandes:

- Puntos de interés: Son puntos específicos que interesa seguir en cada *frame* a lo largo de un vídeo para estudiar su evolución a lo largo del mismo, ya que pueden aportar información clave para la detección de eventos. Estos puntos pueden ser, por ejemplo, puntos característicos del cuerpo de una persona del vídeo (manos, codos, hombros, cabeza), que pueden aportar información valiosa sobre el movimiento de dicha persona.

- Regiones de interes (ROI, *Region of Interest*) y BLOBs (*Binary Large OBject*): A diferencia de los puntos de interés, en este caso se siguen regiones enteras de píxeles en cada *frame* de un vídeo. Estas regiones pueden ser regiones estáticas en un vídeo, pero también suelen asociarse a objetos en movimiento y se corresponden con los llamados BLOBs . Un blob es una región de píxeles que aparece diferenciada del resto en la máscara binaria correspondiente al frente del vídeo, es decir, a la máscara que representa a los píxeles que han cambiado de un *frames* a otro *frame* consecutivo.

A continuación se presentan aquellas características más relevantes para la detección de eventos:

- Tamaño de blob: Es una de las características más típicas en todos los trabajos, ya que permite estimar cosas como la distancia de objetos en profundidad, distinguir entre objetos y personas, etc.

- Velocidad: Es muy importante para temas de seguimiento de objetos, aunque también es una característica útil para reconocer ciertos eventos. Se puede aplicar tanto a puntos aislados como a blobs enteros.

Figura 2.7: Ejemplo de puntos de interés (a) y blobs (b)
En (a) aparecen en rojo los puntos de interés. En (b), la imagen de la derecha corresponde a la máscara de frente, con los blobs marcados en blanco
(Figura (a) extraída de [9])

- Trayectoria: Igual que con la velocidad, la trayectoria es de gran utilidad para el seguimiento de objetos, ya que permite realizar una estimación de la posición futura de un objeto o persona. Suele usarse mucho aplicado a puntos de interés de un objeto, para detectar movimientos en ciertas zonas del objeto, como por ejemplo, el movimiento de los brazos de una persona [9].

- Probabilidad de ser persona: Es una característica muy importante para extraer de un blob, pues permite difereciar una persona de un objeto. En [10] utilizan un algoritmo basado en identificar los bordes característicos de cada parte del cuerpo y generar cuatro modelos de bordes para cada una de las partes (cuerpo, cabeza, torso y piernas). Los cuatro modelos son generados mediante entrenamiento, y luego mediante un detector para cada parte que utiliza como referencia dichos modelos, se detectan las cuatro partes del cuerpo. Esta característica es muy importante para poder distinguir entre objetos y personas, algo generalmente vital en detección de eventos.

- Probabilidad de ser grupo: Esta característica se extrae para un conjunto de blobs cercanos entre sí. Un ejemplo es [11], donde explican las condiciones que debe cumplir un conjunto de blobs detectados como personas para ser reconocido como grupo, como son el tamaño relativo de los blobs participantes, la distancia entre ellos, la velocidad de movimiento de los blobs o la cantidad de blobs en un área. Estas son características muy comunes y siempre resultan útiles, ya que permiten saber cosas como el tamaño del objeto o persona, si esta cerca o lejos, como se mueve, etc. Otro ejemplo de esto es [12], donde utilizan el área de los blobs y la distancia entre ellos.

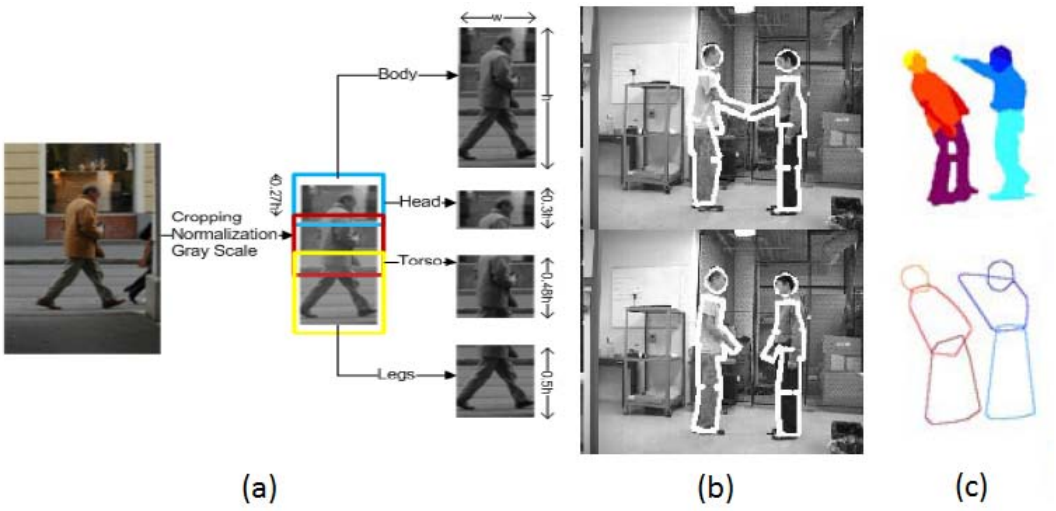


Figura 2.8: Ejemplos de detección de partes del cuerpo
Figura (a) extraída de [10], (b) extraída de [14], (c) extraída de [15]

- Simetría: La simetría puede ser útil para detectar cierto tipo de blobs. En [13] comprueban si la silueta de un blob, antes detectado como persona, es simétrica, y si no es así interpretan que esa persona carga un objeto.

- Estaticidad: El tiempo también juega un papel importante en la detección de eventos. En este caso, si un punto de interés o un blob no se mueve ni cambia su forma durante varios *frames*, se considerará estático.

- Regiones del cuerpo: También se puede dividir el cuerpo de una persona en partes, como en [14], donde definen un modelo de autómata y comparan este modelo con los blobs de las personas, pudiendo así aproximar las regiones del cuerpo al autómata. Incluso se puede estimar la pose de las personas, como en [15], donde tras dividir el cuerpo humano en tres partes (cabeza, torso y brazos, y cintura y piernas), buscan estimar la pose de la cabeza, los brazos y las piernas. Otra forma de detectar partes del cuerpo es la ya vista antes, presentada en [10]. Algunos de estos ejemplos pueden verse en la figura 2.8.

- Detección de piel: Se pueden extraer regiones de piel a partir de la cromaticidad de la imagen, ya que la piel presenta un color muy característico. Un ejemplo es [16], donde utilizan un detector de piel basado en umbralización adaptativa de los canales de color.

  La umbralización consiste simplemente en elegir un rango de valores para cada canal de color (según el espacio de colores), y establecer que todos aquellos píxeles que presenten unos valores dentro de dichos rangos para los canales umbralizados serán detectados como piel.

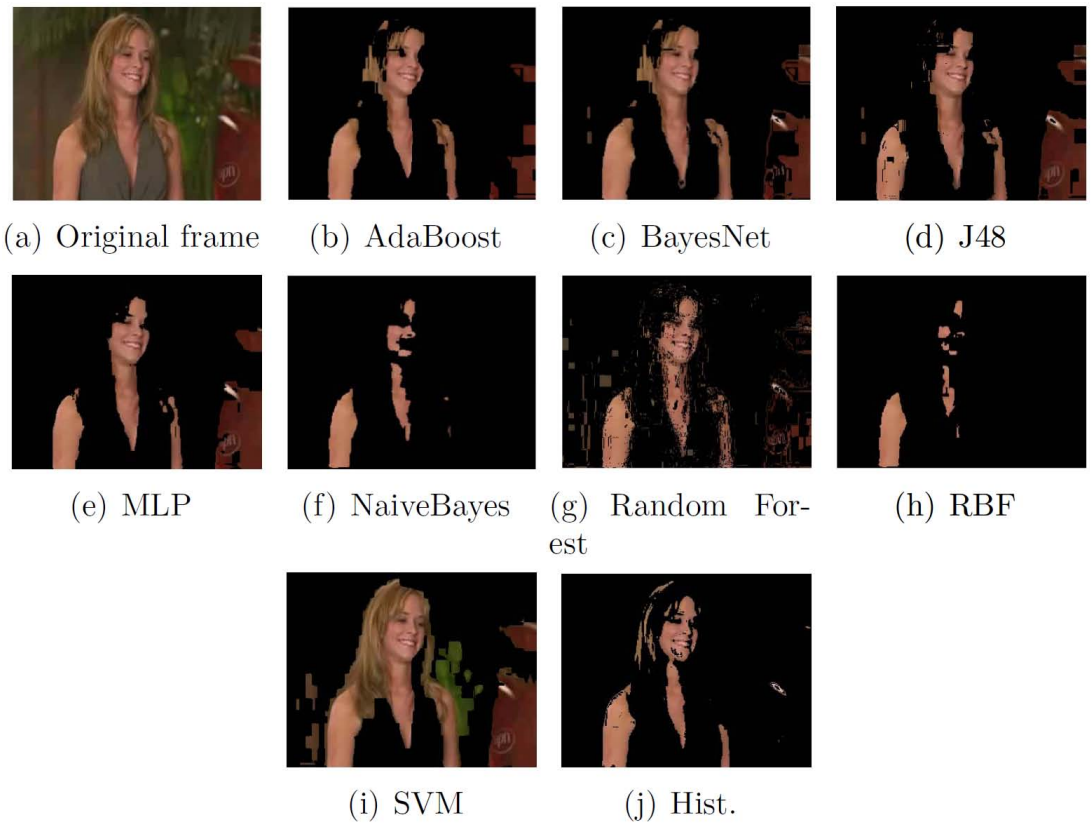


Figura 2.9: Comparativa de algoritmos de detección de piel
Figura extraída de [18]

Existen numerosas formas de detectar la piel: desde la más simple umbralización de los canales de color, hasta utilizar algoritmos de entrenamiento de lo más complejos. En [17] se resume gran parte del estado del arte relacionado con este tema: cuales son los espacios de colores más usados (e.g. RGB, HSV, YCbCr) y técnicas más utilizadas (e.g. modelos de histogramas, GMMs, BNs, Random Forest), de las cuales algunas utilizan entrenamiento previo a la detección.

Además existen varios trabajos donde realizan comparaciones entre espacios de colores y técnicas para detección de piel, como es [18], donde realizan una evaluación de varios de los algoritmos más actuales para varios espacios de colores (ver figura 2.9).

- Detección de objetos de interés: También se pueden detectar ciertas regiones del escenario como objetos interesantes para la detección de ciertos eventos. Algunos ejemplos típicos son las puertas, portátiles, libros, etc. En el apéndice A se presenta algunos ejemplos de detección de puertas, junto con pruebas realizadas con un algoritmo diseñado para este fin, los resultados obtenidos y un breve resumen del algoritmo usado.

Figura 2.10: Ejemplo de anotación contextual
(Figura extraída de [19])

- Información contextual: En contraposición a la detección de objetos de interés, también se pueden anotar previamente dichos objetos de interés en el escenario. Anotando estos objetos se le da al sistema información de su localización, del espacio que ocupan en escena y de su categoría (tipo de objeto), información que se denomina contextual (figura 2.10). Ésto es perfectamente realista, ya que normalmente las cámaras están fijas en el escenario, por tanto, es válido que se conozca la localización de los objetos en escena desde el arranque del sistema en cuestión. En [19, 20, 21] introducen información de este tipo acerca del escenario (objetos fijos o portables). Además, la información contextual también se refiere a las relaciones que puedan existir entre eventos (exclusión mutua, orden de ocurrencia, etc.).

2.3.2. Aproximaciones existentes

Una vez conocidas algunas de las características más importantes que se pueden extraer para poder realizar la detección de eventos y de definir el concepto de información contextual, vamos a presentar algunos de los trabajos en los cuales se tratan interacciones persona-objeto y actividades, e interacciones persona-persona. Se han analizado sólo aquellos trabajos que presentan interacciones y actividades similares a los eventos propuestos en la competición HARL. Muchas de las características que extraen para hacer funcionar sus técnicas de detección son las descritas en el subapartado anterior.

2.3.2.1. Interacciones persona-objeto y actividades

- Dejar o coger un objeto (en/de una caja, mesa, escritorio, etc.)

  Las técnicas utilizadas para esta clase de eventos son las basadas en seguimiento de objetos [22], pero a veces el objeto no se encuentra en escena al comienzo del vídeo, o desaparece de escena cuando es guardado en algún recipiente del escenario, por tanto, no siempre son fiables.

En [19, 20] introducen información contextual acerca del escenario. De esta manera, ya se conoce previamente la posición de los objetos que se usarán en el evento o de aquellos recipientes o mesas donde podrían estar o dejarse, facilitando así mucho el problema, ya que si una persona acude en el vídeo a zonas ya clasificadas previamente, sabremos que algo está ocurriendo (figura 2.10). Ahora bien, esta técnica tiene algunas limitaciones; por ejemplo, la información se define para cada escenario y punto de vista de la cámara, es decir, si trabajamos con un dataset en el cual cada vídeo tiene un escenario y punto de vista de cámara distinto, habrá que definir dicha información para cada uno de los vídeos.

- Entrar/Salir de una habitación

  Este tipo de eventos ocurren siempre en presencia de una puerta, pudiendo estar abierta o cerrada, tanto al principio como al final del evento. Además, siempre aparecerá o desaparecerá una persona en dicha puerta. Ahora bien, aunque la detección de la puerta no siempre es necesaria, si facilita las cosas. Hay varias formas de conocer la puerta: se puede utilizar información previa [19, 20] o se pueden utilizar técnicas de detección de puertas a partir del marco [23]. En [24] tratan varios eventos similares a los propuestos aquí, entrenando su sistema con varias secuencias de vídeo en las que sucedan dichos eventos, para hacer que éste aprenda los patrones que conllevan dichas acciones y pueda reconocerlas en otros vídeos de test.

- Interacción con una puerta

  En este caso se quiere especificar que la persona interactúa de alguna manera con la puerta, y que no se limita sólo a atravesar su marco, como es el caso del evento anterior. Aquí, es necesario obligatoriamente conocer previamente la localización de la puerta cerrada [19, 20] o detectarla [23]. Conociendo la puerta, habrá que detectar cualquier tipo de interacción o acceso a ella, parecido al caso de coger/dejar un objeto de un recipiente. También es posible utilizar la propuesta de [25], que utiliza sustracción de fondo para detectar grandes regiones en movimiento en los casos en los que la puerta comienze o acabe cerrada. Cualquier puerta al comienzo del vídeo será detectada como fondo, pero cuando ésta se abra o se cierre, aparecerá como una región rectangular del frente (figura 2.11). Adicionalmente, en dicha región aparecerá o desaparecerá un blob correspondiente a una persona. El principal problema se da cuando la puerta está en un lateral respecto al punto de vista de la cámara.

- Abandono de equipaje (dejar e irse)

  Este evento ha sido ampliamente estudiado debido a que es una de las aplicaciones que más interesa actualmente desarrollar, el abandono de objetos. Por tanto, podemos encontrar numerosas técnicas. Por ejemplo, en [26] utilizan la fusión de algoritmos simples para

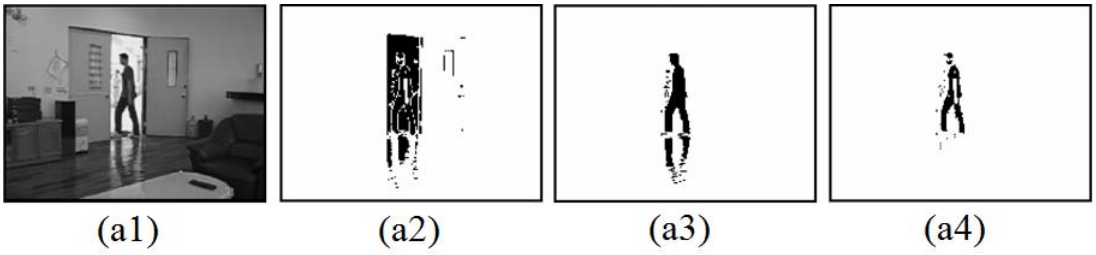
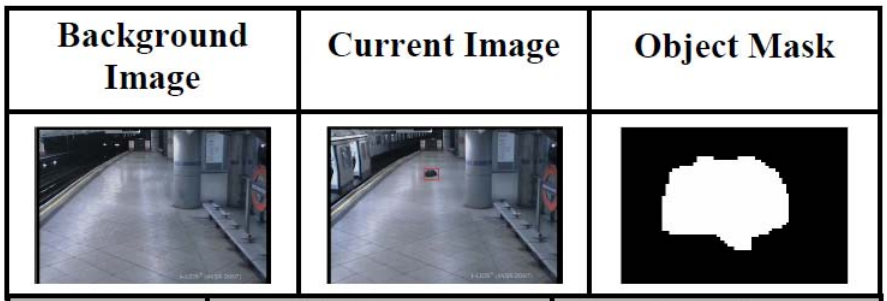


Figura 2.11: Ejemplo de detección de *apertura de puerta*
(Figura extraída de [25])

detectar el objeto y clasificarlo como objeto abandonado, usando información de contornos, de estaticidad, detectándolo previamente como objeto, buscando en *frames* anteriores a su poseedor, etc (figura 2.12). En [27] utilizan también distintos parámetros calculados a partir del objeto para identificarlo como objeto abandonado.



Detector	Evidence		Decision	
	Unattended (U)	Stolen (S)	Method	Ground Truth
Color Histogram	0.17	0.22	S	U
High-Gradient	0.14	0.35	S	U
Low-Gradient	0.85	0.0	U	U
Fusion	0.85	0.0	U	U

Figura 2.12: Ejemplo de detección de *equipaje desatendido* o *robo de equipaje*
(Figura extraída de [26])

- Escribir en un teclado

  Existen numerosas propuestas para detectar este evento, pero la mayoría de ellas utilizan audio, buscando detectar el sonido producido al teclear. Aun así, existen algunas propuestas que sólo utilizan información visual. Por ejemplo, en [28], en primer lugar buscan localizar portátiles, comparando el objeto detectado con modelos de portátil para varios puntos de vista. Una vez localizado el portátil, buscan que las manos de la persona estén sobre el portátil, y que en esa zona se produzcan cambios en el frente (figura 2.13). El problema de este método es que se necesita tener una cámara en un plano cenital para poder detectar

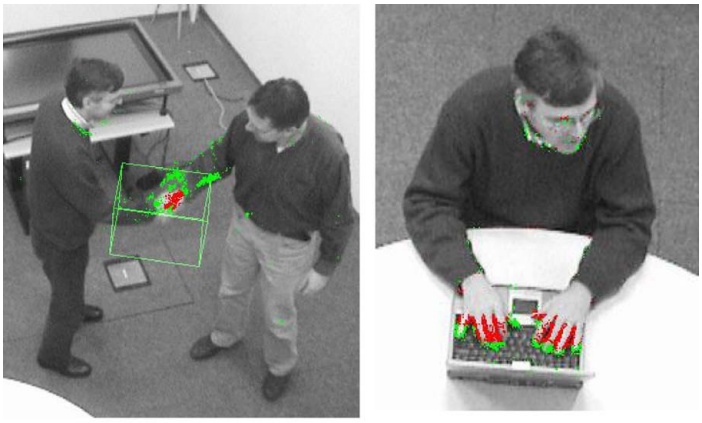


Figura 2.13: Ejemplo de detección de *apretón de manos* (a) y *teclear* (b)
(Figura extraída de [28])

las manos sobre el teclado, además de que el evento también puede producirse con un teclado de un ordenador de sobremesa, que es mucho más difícil de detectar.

Otra posibilidad es la inserción de información previa sobre el escenario [19, 20], donde conste la localización del teclado o portátil, suponiendo que antes y durante el evento dicho teclado no se moverá de la posición establecida en la información previa.

- Hablar por teléfono

  Este evento involucra un teléfono, por tanto, es necesario conocer la localización del mismo, ya sea mediante una detección de objetos similares a un teléfono o mediante introducción previa de información sobre el escenario, y buscando una interacción entre una persona y un objeto ya clasificado como teléfono. El principal problema está en que el teléfono no siempre es fijo, sino que en ocasiones la persona utilizará un móvil escondido durante toda la secuencia hasta el momento del evento.

  Otra posibilidad es la realizada en [29], donde buscan detectar interacciones entre la cabeza y las manos. Utilizan cadenas de Markov para modelar las acciones e inferencia Bayesiana a partir de los parámetros extraídos para diferenciar entre eventos que involucren cabeza y manos. El problema de este método es que el teléfono se sitúe detrás de la cabeza de la persona, como mencionan en el propio documento.

2.3.2.2. Interacciones persona-persona

- Una persona da un objeto a otra

  Para reconocer este evento es imprescindible detectar y seguir el objeto involucrado. Para

Figura 2.14: Ejemplo de detección de objeto por asimetría
(Figura extraída de [13])

ello existen numerosos estudios sobre la detección y el seguimiento de objetos, donde se presentan gran cantidad de técnicas [22]. El problema está en detectar dichos objetos. En muchos casos el objeto que pasa de una persona a otra no ha sido reconocido previamente a la acción, y a veces su tamaño dificulta mucho su detección, pareciendo que las dos personas involucradas en la acción se dan la mano, lo que provoca una confusión de eventos. Por tanto, en este caso el problema reside en la segmentación del frente, que puede no ser válida para detectar el objeto.

Otra posibilidad es la propuesta de [13], donde detectan la silueta de una persona y suponen que la silueta debe ser simétrica. Cualquier elemento de dicha silueta que rompa la simetría es un objeto que carga dicha persona (figura 2.14). Por tanto, establecen que si la asimetría cambia de persona, se ha producido un intercambio del objeto, y el evento será reconocido. El problema se da cuando el objeto es pequeño, ya que no altera la silueta, o cuando la pose de la persona no la haga simétrica. También puede ocurrir que el objeto sea ocluído por la propia persona. Esto también es válido para la interacción persona-objeto de abandonar equipaje, o de coger/dejar objeto.

- Discusión entre dos o más personas

  Existen numerosos estudios para este tipo de eventos, pero la mayoría de ellos utiliza información de audio, ya que la voz es la principal característica de este evento. Resulta muy difícil definirlo teniendo sólo información de vídeo, que es, desafortunadamente, el caso estudiado en este proyecto. Por tanto, es muy difícil encontrar soluciones que se centren en este evento y no usen audio.

  Una de las soluciones que aparecen en varios trabajos, al margen de la técnica usada, es la propuesta en [30], donde se define este evento como un conjunto de sub-eventos más simples que se supone que forman parte de una conversación normal, como son el movimiento de brazos de las personas que discuten, los cambios de pose o la distancia entre cuerpos. El problema se da cuando ambas personas se encuentran completamente inmóviles, haciéndose indetectable con este método. Además, estas técnicas basadas en

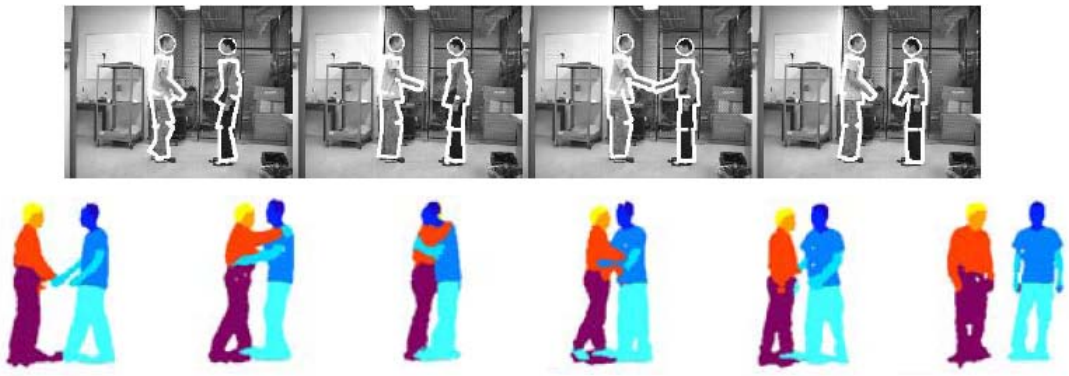


Figura 2.15: Ejemplos de interacción (*handshaking* y *hug*)
(Figuras extraídas de [14] y [15])

división de eventos en sub-eventos suelen ser bastante complejas de implementar.

En [11] explican las condiciones que debe cumplir un conjunto de blobs detectados como personas para ser reconocido como grupo. Esto es similar para una discusión entre personas, si suponemos que las personas involucradas en este evento no pueden estar involucradas a la vez en otro evento y no están agrupados sin hacer nada. El problema es que las suposiciones sólo son válidas si conocemos el marco en el que trabajará el sistema. Además, es importante tener en cuenta el tiempo que deben estar juntos los blobs para ser considerados grupo, pues si no cualquier cruce de personas podría provocar el evento.

- Apretón de manos

  Este evento es una de las interacciones entre personas más estudiadas en la literatura, habiendo así muchos métodos distintos para detectarla.

  En [15], utilizan DBN (*Dynamic Bayesian Networks*) con una jerarquía de tres niveles. El primero se fija en la evolución de la pose de las partes del cuerpo, el segundo en la pose general del cuerpo, y el tercero en la interacción entre las dos personas, añadiendo además características de carácter temporal y espacial (figura 2.15). El problema es que las DBN tienen un alto coste computacional y requieren datos de entrenamiento que en muchas situaciones no está fácilmente disponible.

  En [31, 32], mediante la transformada de Hough son capaces también de detectar este evento, entre muchos otros. El principal problema de este método es que necesita una etapa de entrenamiento previa.

  En [28], mediante una detector de gestos, buscan la zona cercana al torso donde se conecten ambas personas, fijándose además en la distancia de los cuerpos (figura 2.13).

2.4. Datasets

Además del dataset proporcionado por la organización de la competición ICPR - HARL 2012, existen muchos dataset públicos disponibles destinados a la investigación en el campo de la detección automática de eventos, entre otros. Adicionalmente, proporcionan una forma de testear cualquier módulo de un sistema de detección automática de eventos (e.g., segmentación del frente, seguimiento, extracción de características). Estos son algunos de los dataset disponibles que presentan vídeos útiles para este proyecto:

- **The LIRIS human activities dataset (ICPR 2012 - HARL)** [33]

  URL: http://liris.cnrs.fr/voir/activities-dataset/

  Este será el dataset principal en este proyecto. Contiene los 10 eventos propuestos y es un claro ejemplo de un escenario interior controlado, donde los vídeos son realizados con cámaras estáticas en una habitación o un pasillo, y generalmente no aparecen en escena más de tres personas.

  Consta de dos sets de vídeos, pero en este proyecto sólo se utilizará el set 2, que presenta 367 acciones en 167 vídeos a color y a resolución DVD (720x576). El set 2 está dividido en dos subsets: *training-validation* y *test*. El primero es el entregado al comienzo de la competición, y el segundo es sobre el cual hay que obtener resultados para enviarlos a la organización.

- **Event Detection dataset (EDds) (Video Processing and Understanding Lab - UAM)**

  URL: http://www-vpu.ii.uam.es/EDds/index.html

  Este dataset contiene 17 secuencias tomadas con una cámara estática en una sala de reuniones, con una resolución de 320x240 píxeles a 12 fps. El dataset está enfocado a dos tipos de eventos: interacciones y actividades, en particular, dos actividades (*Hand Up* y *Walking*) y tres interacciones persona-objeto (*Leave*, *Get* and *Use object*).

- **UT-Interaction dataset (ICPR 2010 - SDHA)**

  URL: http://cvrc.ece.utexas.edu/SDHA2010/Human_Interaction.html

  Este dataset contiene vídeos en los cuales se ejecutan 6 tipos de interacciones persona-persona en un escenario exterior (*Handshaking, Hugging*, *Kicking*, *Pointing*, *Punching*, *Pushing*). Hay un total de 20 secuencias de vídeo de 1 minuto de duración aproximadamente. Estos vídeos han sido tomados con una resolucion de 720x480 píxeles, a 30 fps, y la altura de una persona ronda los 200 píxeles.

- **AMI Meeting Corpus**

URL: http://corpus.amiproject.org/

Este dataset esta compuesto por 100 horas de vídeos grabados en salas de reuniones con cámaras estáticas en distintas posiciones de la sala. En estas secuencias hay anotados eventos del tipo movimientos de cabeza (asentir, negar), de movimiento de brazos (apuntar a la pantalla), interacciones con objetos sobre la mesa (tomar apuntes) o movimientos generales (sentarse, levantarse). Los vídeos presentan una resolución alta, de 720x576 píxeles.

- **ISE Lab dataset (Hermes)**

  URL: http://iselab.cvc.uab.es/tools-and-resources

  El dataset presenta una sola secuencia grabada desde distintas posiciones del escenario. They act in a discussion sequence sitting around a table, where bags are carried, left and picked from the floor, and bottles are carried, left and picked from the vending machine and from the table. Consiste en una grabación de tres personas en una sala, donde se producen diversas acciones entre ellos y con algunos objetos (coger/dejar objeto en una mesa, coger/dejar objeto en el suelo, coger objeto de máquina expendedora, discusión), acciones que en ocasiones presentan oclusiones o interrupciones. La secuenca de vídeo consta de 2003 frames a una resolución de 1392x1040 píxeles a 15fps.

- **AVSS 2007 (iLids dataset)**

  URL: http://www.eecs.qmul.ac.uk/~andrea/avss2007_d.html

  Consta de tres secuencias grabadas en el andén de una estación de metro donde el principal evento es el abandono de objeto. Cada secuencia presenta un nivel de dificultad mayor que la anterior. Los vídeos presentan una resolución de 720x576 píxele y duran aproximadamente 3.5 minutos.

- **CANDELA project**

  URL: http://www.multitel.be/~va/candela/abandon.html

  Este dataset contiene 16 secuencias de vídeo grabadas en una habitación, en las cuales suceden interacciones entre humano y objeto principalmente, como el abandono de objetos, coger/dejar un objeto o dar un objeto a otra persona. Los vídeos tienen una resolución de 352x288 píxeles y una duración aproximada de 30 segundos.

- **PETS 2006**

  URL: http://www.cvg.rdg.ac.uk/PETS2006/data.html

  Este dataset consta de 7 datasets distintos sobre el mismo escenario. Esta formado por secuencias multi-sensor que contienen escenarios cuyo principal evento es el abandono de equipaje, cuya complejidad es creciente.

La resolución de todas las secuencias es el estándar PAL (768 x 576 píxeles, 25 *frames* por segundo) y están comprimidas como secuencias de imagenes JPEG (aprox. 90 % de calidad).

- **PETS 2007**

  URL: http://www.cvg.rdg.ac.uk/PETS2007/data.html

  Este dataset consta de 8 datasets distintos, con secuencias multi-sensor sobre 3 escenarios distintos, cuya complejidad es creciente y que presentan tres eventos: merodear, robo de equipaje atendido, desatender equipaje.

  La resolución de todas las secuencias es el estándar PAL (768 x 576 píxeles, 25 *frames* por segundo) y están comprimidas como secuencias de imagenes JPEG (aprox. 90 % de calidad).

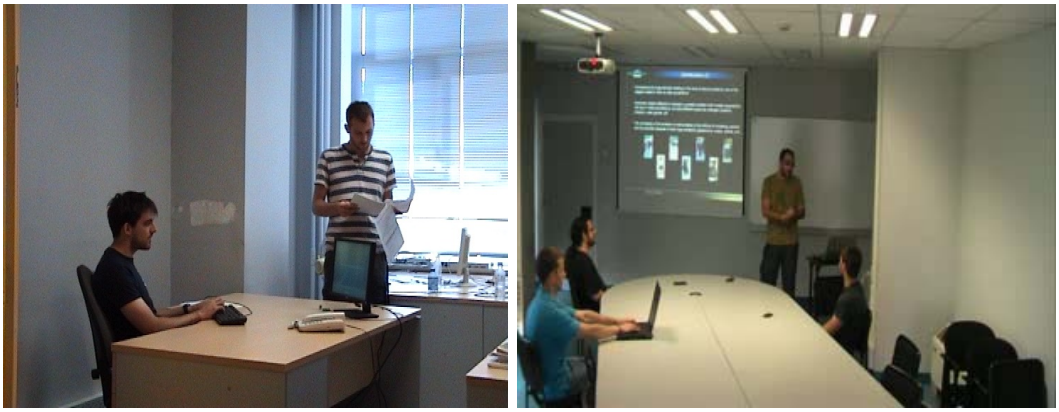


Figura 2.16: Ejemplos de los dataset LIRIS, ED

Capítulo 3

Estudio del prototipo del VPU-Lab

3.1. Introducción

Tras haber analizado los trabajos relacionados con el reconocimiento automático de eventos, en este capítulo se realizará un estudio del prototipo realizado por el VPU-Lab para detección de eventos. Este será el sistema en el cual se desarrollarán todos los cambios o mejoras propuestos en este proyecto, razón por la que conviene explicar brevemente su funcionamiento, para así entender futuros cambios en el mismo.

El prototipo del VPU-Lab fue diseñado para afrontar el reconocimiento de eventos en vídeo en tiempo real utilizando un solo punto de vista. Se basa en los siguientes principios:

- El sistema debe ser capaz de trabajar en escenarios con varias personas en escena.
- No estará basado en un sistema dependiente de una fase de entrenamiento previa.
- La complejidad computacional deberá ser lo suficientemente baja para que su uso sea válido en aplicaciones en tiempo real.
- Usará información contextual sobre el escenario, dada previamente al sistema.

Este capítulo describe el prototipo en el siguiente orden: arquitectura del sistema (sección 3.2), detección de eventos (actividades e interacciones) (sección 3.3), información contextual (sección 3.4) y limitaciones del prototipo (sección 3.5).

3.2. Arquitectura del sistema

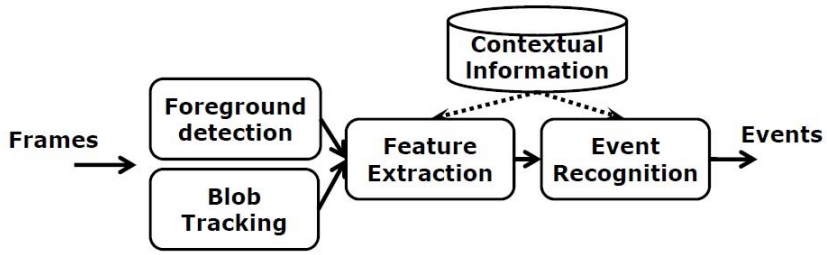


Figura 3.1: Diagrama de bloques del sistema propuesto

El diagrama de bloques de la figura 3.1 representa de manera simple la arquitectura del sistema. En primer lugar, a partir de los frames de la secuencia de vídeo, las etapas *foreground detection* (detección de frente) y *blob tracking* (seguimiento de blobs) realizan la detección de objetos de interés. Después, la etapa *feature extraction* (extracción de características) extrae las características de dichos objetos. Por último, la etapa *event recognition* (reconocimiento de eventos), encargada de entregar a la salida los eventos detectados a partir de las características obtenidas y la información contextual (*contextual information*). A continuación se explicarán con más detalle cada una de las etapas:

- *Foreground Detection:* Esta etapa internamente está subdividida en varias subetapas, que son: *background substraction* (sustracción de fondo), *shadow and highlight filter* (eliminación de sombras y reflejos) y *connected component analysis* (generación de blobs).

  - *Background substraction:* Aplica un algoritmo de sustracción del fondo entre dos frames: el inicial (*background model*), el cual define el fondo de la escena, y el actual que está siendo tratado. Además, al resultado se le aplica un umbral para eliminar el ruido de la cámara, es decir, si la resta de ambas imágenes para un mismo píxel no supera cierto umbral, se considerará fondo. El resultado final es lo que llamamos el frente (*foreground*) de la imagen (objetos móviles en escena). En la ecuación 3.1, I representa la imagen actual, B es la imagen de fondo y β es el umbral de decisión. El resultado será F, que es la máscara de frente.

$$F(I[x,y]) \Longleftrightarrow \sum_{i=-W}^{W} \sum_{j=-W}^{W} (|I[x+i,y+j] - B[x+i,y+j]|)^2 > \beta \qquad (3.1)$$

  - *Shadow and highlight filter:* Se aplica una eliminación de sombras y reflejos basada en la reducción de la crominancia y la intensidad entre la imagen actual y el fondo de escena. Esto se realiza ya que la aparición de sombras y reflejos puede hacer que aparezcan nuevas regiones en el frente erróneas.

- *Connected component analysis:* Los píxeles detectados como frente en etapas anteriores se agrupan en regiones conectadas y etiquetadas llamadas blobs. Sólo se detectan los blobs de un determinado tamaño, filtrando así blobs demasiado pequeños.

Hay que destacar que ésta es la etapa más crítica de todo el proceso realizado por el sistema, ya que cualquier fallo en esta etapa será arrastrado a etapas posteriores generando detecciones de eventos erróneas.

- *Blob Tracking:* El objetivo de esta etapa es realizar un seguimiento correcto de los objetos móviles escena a escena. El seguimiento se realiza sobre los blobs que identifican estos objetos.

  El algoritmo utilizado en este prototipo identifica los blobs a partir de cinco características: posición (x, y), máscara del blob obtenida en la etapa de extracción de frente, centroide del blob, ancho y alto del blob (w, h) e histograma de color.

  La finalidad es asignar un ID a cada blob. Para ello, el sistema genera una matriz de coste V_{ij} entre los blobs previos (O_i) y los nuevos blobs (N_j). El coste V_{ij} entre dos blobs i y j se define con el valor *score metric* (SM), que es un valor obtenido de una ecuación que relaciona las características antes descritas. Valores de SM cercanos a 0 indican que el blob N_j es un buen candidato para ser asociado con O_i. De este modo, el algoritmo es capaz de predecir la nueva posición del blob y relacionar blobs entre escenas consecutivas, es decir, reasignar las ID a los blobs correspondientes.

  Por tanto, la salida de esta etapa junto con la anterior será un listado de blobs, cada uno con su ID, a los cuales se les aplicará la extracción de características en la siguiente etapa.

- *Feature extraction:* Para poder reconocer eventos, el sistema debe extraer una serie de características de cada blob detectado en la escena. Según que tipo de eventos se quieran detectar y según el tipo de escenario, se extraerán unos u otros. A continuación, pasamos a explicar algunas características que este sistema puede extraer:

  - *BlobVelocity:* A partir del centro de masa del cuarto inferior (*Lq*, *lower quarter*, figura 3.2) de los blobs previo y nuevo, relacionados en la etapa de seguimiento, se puede obtener la velocidad. El centro de masa en *Lq* se estimará utilizando momentos.

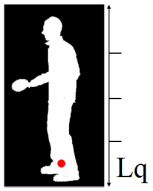


Figura 3.2: Cuarto inferior de un blob humano

Después, utilizando la siguiente fórmula, donde vx y vy son la diferencia entre dichos centros de masa en las direcciones x e y, se obtiene la velocidad.

$$BlobVelocity = \sqrt{v_x^2 + v_y^2} \tag{3.2}$$

- *BlobTrajectory:* Trayectoria del blob, usada para determinar ciertos movimientos.
- *PeopleLikelihood:* Clasifica el blob como persona. Se puede obtener con distintos algoritmos. Por ejemplo, el algoritmo puede detectar una persona como la unión de cuatro partes dentro del blob (cuerpo, cabeza, torso y piernas), o también puede fijarse en las proporciones del blob y la silueta. Cualquier método dará una probabilidad que, si supera cierto umbral, identificará al blob como persona.
- *GroupLikelihood:* Mediante distancias entre blobs detectados como personas, el sistema puede clasificar un conjunto de blobs como grupo.
- *PeopleSkin:* Estableciendo ciertos valores de cromaticidad como color de la piel, se pueden detectar las regiones de piel descubiertas. Esto se realizará sólo para blobs clasificados como personas ($PeopleLikelihood > \tau_1$), La técnica utilizada en el prototipo es la umbralización adaptativa de tinte propuesta en [16].
- *EdgeEnergy:* Permite detectar objetos del frente que han sido insertados o sustraídos mediante el método Bhattacharyya de distancias entre histogramas de color entre las regiones externa e interna del blob.
- *ObjectOwner:* Para blobs clasificados como objeto ($PeopleLikelihood < \tau_2$), en el momento de su aparición se determina que el blob clasificado como persona más cercano al objeto será su poseedor.
- *ObjectCompactness:* Se define mediante el valor de los píxeles, el ancho y el alto del blob para aquellos clasificados como objeto.

Al final de esta etapa obtendremos otro listado de blobs que relaciona cada uno con sus características y que pasará a la última etapa, el reconocimiento de eventos.

3.3. Detección de eventos (actividades e interacciones)

La etapa de reconocimiento de eventos (*event recognition*) puede ser desarrollada de diversas formas en función del escenario de estudio y de los eventos que queramos reconocer en él. Los eventos que detecta este sistema se pueden clasificar de dos formas:

- *Interacciones humano-objeto:* Eventos que involucren una persona y un objeto
- *Actividades humanas:* Eventos que implican movimiento de una persona.

Recordar que en etapa no conlleva una etapa de entrenamiento previa. La detección de eventos está basada en reglas "semánticas" (definidas explícitamente por el diseñador). El opuesto son aproximaciones basadas en aprendizaje, en las cuales las reglas se aprenden de manera implícita (desconocidas por diseñador en la mayoría de los casos).

En este sistema, la mayoría de los eventos se detectan combinando las características extraídas de los blobs del frente en un marco de inferencia Bayesiano para segmentos cortos de tiempo (1-25 frames). Este método sigue la siguiente fórmula:

$$P(H/E_{1...N}) = \frac{\prod_{i=1}^{N} P(E_i/H)P(H)}{\prod_{i=1}^{N} P(E_i/H)P(H) + \prod_{i=1}^{N} P(E_i/\widetilde{H})P(\widetilde{H})} \tag{3.3}$$

donde H es el evento a detectar, E_i son todas las características o evidencias que definen dicho evento y N es el número de características o evidencias. Para la fórmula anterior se asignan previamente los siguientes valores:

- $P(H) = P(\widetilde{H}) = 0{,}5$ (No hay información previa de los eventos)
- $P(E_i/\widetilde{H}) = 0{,}5$ (Probabilidades difíciles de estimar)
- $P(E_i/H)$ (Diferente para cada evidencia del evento)

A continuación, en las siguientes tablas 3.1 y 3.2 se muestran los eventos que el sistema detecta mediante este método. Para cada evidencia de cada evento se puede calcular la probabilidad de ocurrencia.

Evento	Modelado de evento
Leave Object (LEA)	(1) El blob no apareció unos pocos frames antes. (2) El blob aparece ahora como blob de fondo. (3) El blob es clasificado como objeto. (4) El blob tiene un poseedor. (5) La distancia objeto-persona es baja.
Get Object (GET)	(1) El blob no apareció unos pocos frames antes. (2) El blob aparece ahora como blob de fondo. (3) El blob tiene un poseedor. (4) La distancia objeto-persona es baja. (5) Un *Obj. Contextual* existe en el mismo lugar.
Use Object (USE)	(1) El blob de persona solapa el *Obj. Portable.* (2) *PeopleSkin* solapa el *Obj. Portable* durante τ frames.

Tabla 3.1: Modelado de interacciones humano-objeto

Evento	Modelado de evento
Walking (WLK)	(1) El blob de frente es clasificado como persona (y no como grupo) al menos τ frames. (2) El *BlobVelocity* es significativa durante τ frames. (3) *HandUp* no esta siendo realizado
Hand Up (HUP)	(1) El blob de frente es clasificado como persona (y no como grupo) al menos τ frames. (2) La distribución de *PeopleSkin* varía. (3) *Walking* no esta siendo realizado

Tabla 3.2: Modelado de actividades humanas

Para algunos eventos no es necesario utilizar inferencia bayesiana, sino que con alguna de las características extraídas ya es posible realizar la detección. Los siguientes dos eventos que puede detectar el sistema muestran este caso:

- *Opposing Flow (OFE):* Para un flujo de personas a través de un área limitada se establece un sentido de movimiento como natural y se detecta cualquier movimiento de personas en sentido contrario al establecido. Se utiliza una operación con los vectores que definen el movimiento natural y el opuesto. Si el resultado supera cierto valor umbral, se detecta el evento.

- *Person Runs (PRE):* Si la *BlobVelocity* supera un cierto valor umbral, se considera que la persona correspondiente al blob está corriendo, detectándose el evento.

3.4. Información contextual

Una de las características más relevantes de este sistema es el uso de información contextual, información sobre aquellos objetos que pueda haber ya en escena y sea interesante tener en cuenta para la futura detección de eventos, así como información adicional sobre los eventos. La información contextual se puede dividir en tres tipos:

- *Espacial:* Distribución de los objetos interesantes en el escenario y categorización de los mismos (móvil o contextual, fijo o portátil).

  MobileObject o *ContextualObject* (*Fixed* o *Portable*) (figura 3.3)

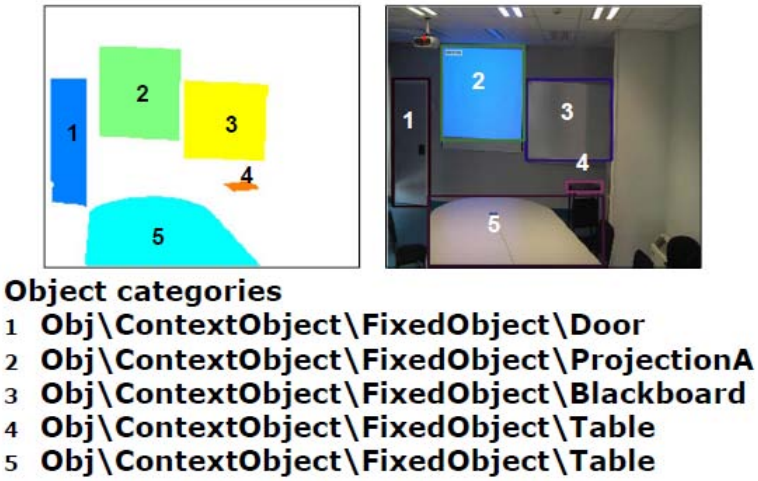


Figura 3.3: Información contextual espacial

- *De objetos:* Relaciones entre objetos (relación de tamaño entre persona y objeto...).
- *De eventos:* Relaciones entre eventos (exclusión mutua, orden de ocurrencia...).

Esta información será utilizada en las etapas de extracción de características y de reconocimiento de eventos, tal y como se presenta en la figura 3.1.

La anotación de información contextual espacial se realiza manualmente con un programa en el entorno MATLAB, a partir del primer frame de cada secuencia de vídeo. Dicho programa permite seleccionar manualmente mediante polígonos en la imágen aquellos objetos deseados para después clasificarlos como se ha descrito anteriormente.

3.5. Limitaciones del prototipo

Lo primero a tener en cuenta a la hora de hablar de limitaciones es saber en que escenario estamos trabajando con este sistema.

- *Escenarios controlados:* Podemos obtener información contextual de manera sencilla y podemos condicionar más o menos que objetos y personas aparecerán en escena, además de limitar en cierta medida los eventos que queremos detectar. Ahora bien, todo funciona muy bien para escenarios con una sola persona, e incluso con dos personas. Con un número mayor de personas, el número de oclusiones es mucho mayor, y esto dificulta mucho la etapa *Foreground Detection* (3.2). Aun así, se consiguen resultados bastante buenos.
- *Escenarios altamente concurridos:* Este tipo de escenarios presentan innumerables dificultades para este sistema de detección de eventos. El número de objetos y personas que

pueden aparecer en escena es muy alto y variable, y por tanto, la probabilidad de oclusiones es muy elevada. Consecuentemente, existe una gran dificultad para extraer blobs del vídeo. Puesto que el prototipo está diseñado para escenarios controlados, en este tipo de escenarios se hace casi imposible detectar cualquier evento.

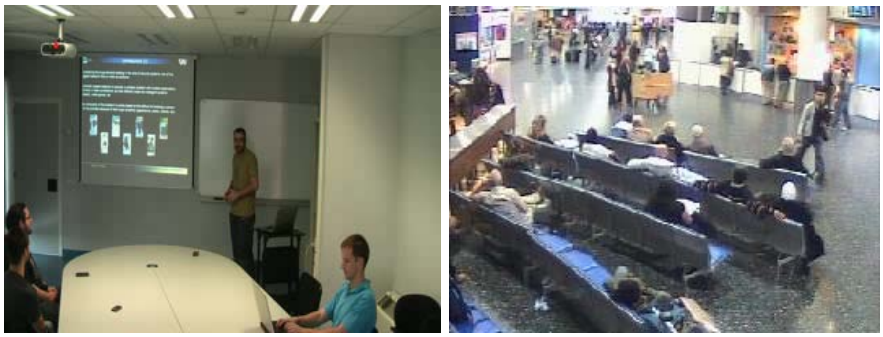


Figura 3.4: Escenario controlado y escenario altamente concurrido

A continuación se presentan las limitaciones y problemas que se dan en escenarios controlados, que es la clase de escenarios que se estudiarán en este proyecto.

La detección del frente es el principal problema de este sistema. Como se ha explicado en el punto 3.2, es quizás la etapa más crítica. Si hay regiones del frente que no se detectan, muy difícilmente podremos detectar eventos provocados por esas regiones que no hemos detectado. Esto suele ocurrir con regiones en movimiento muy similares al fondo sobre el que se mueven, razón por la que el sistema no es capaz de distinguir el frente del fondo en dichas regiones (figura 3.5).

Otra limitación derivada de la extracción del frente es la necesidad de un fondo. El sistema extrae (frame a frame) el frente a partir de una imagen de fondo, que es el primer frame del vídeo. Esto limita al sistema, ya que sólo podrá trabajar correctamente con vídeos en los que al comienzo del mismo no haya ningúna persona en escena. Si por el contrario, aparece una persona, ésta comenzará como parte del fondo, y cuando se mueva, dejará en el frente un blob erróneo llamado "fantasma", ya que no hay realmente ningún objeto en el frente (figura 3.5).

Como ya se ha comentado, otro gran problema son las oclusiones entre objetos o personas. Esto hace que en la etapa de detección del frente, el sistema englobe en un mismo blob varios objetos. Debido a esto, al sistema se le pueden escapar muchos eventos que puedan estar relacionados con estos objetos. Además, tras una breve fusión de blobs, los blobs involucrados aparecen con nuevas ID, lo que hace que la ID sea una característica del blob que no podemos usar en muchas ocasiones para seguir un objeto, y afecte a todos aquellos eventos que dependan del seguimiento de la misma.

Otro problema está en el cambio de iluminación que pueda producirse en el escenario, ya que puede cambiar la cromaticidad de todo, generar nuevas sombras y reflejos, etc. Esto puede

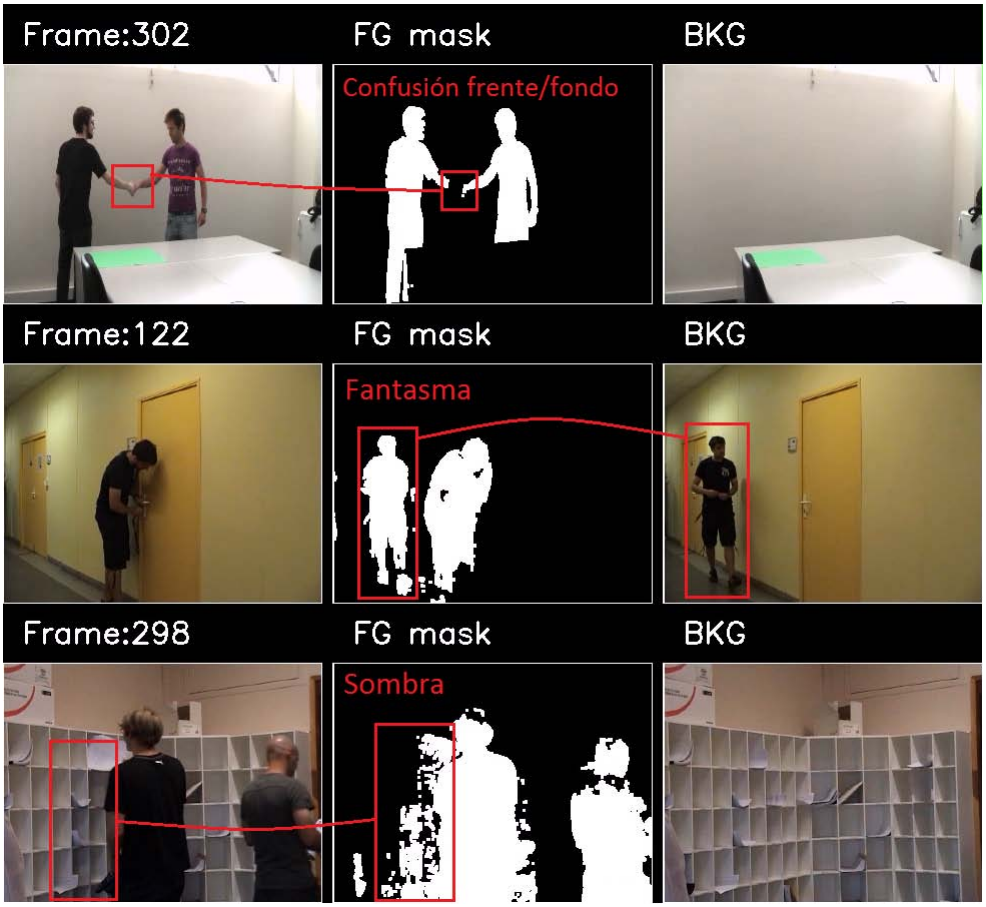


Figura 3.5: Ejemplos de fallos típicos

afectar a la extracción de características como *PeopleSkin.* Si esto ocurre, no podremos fiarnos de la piel para detectar ciertos eventos, ya que si justo la región que nos interesa en un momento dado no tiene piel detectada, no se detectará el evento de interés.

También hay que saber escoger los valores umbrales de todas las operaciones que requieran de ellos. Además, hay que tener en cuenta los cambios del escenario que se puedan producir, ya que pueden hacer que algún valor umbral pierda sentido, y requiera un cambio de valor. Por tanto, la dependencia del sistema con el escenario también suele ser muy grande, y para escenarios aparentemente similares, los resultados pueden ser muy distintos.

El módulo de detecctión de piel es mejorable, ya que para algunas secuencias de vídeo no es capaz de extraer correctamente la piel de las personas. En la figura 3.6 hay tres ejemplos de situaciones donde no reconoce la piel perfectamente. Aunque el algoritmo utilizado para la umbralización adaptativa en este prototipo reconoce bastante piel como tal, también reconoce como piel muchas regiones que no debería, como se puede observar en las zonas marcadas en rojo en los ejemplos de la imagen.

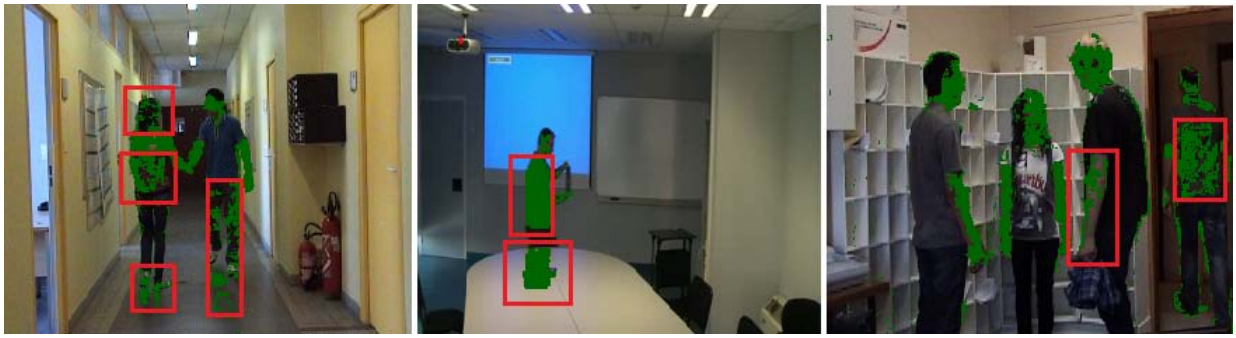


Figura 3.6: Ejemplos del detector de piel del prototipo

Por último, el prototipo está diseñado para los eventos específicados en el apartado 3.3, y no es capaz de detectar la mayoría de los eventos propuestos en la competición. Los eventos relacionados con coger/dejar objeto son muy similares a los de la competición, y el evento de usar objeto puede servir de base a otros eventos, pero el prototipo no es capaz de reconocer eventos que impliquen una interaccion entre dos personas, como es un apretón de manos o una discusión. Tampoco reconoce eventos en los que una persona interaccione con objetos estáticos del entorno, como por ejemplo, la puerta. Cada evento nuevo que se quiera incorporar al sistema, hay que implementarlo.

Por todo esto, será necesario realizar un estudio y planificación de mejoras para el prototipo, de forma que podamos mejorar los resultados del mismo y conseguir que sea capaz de detectar todos los eventos posibles de los propuestos en la competición.

Capítulo 4

Extracción de características: piel

4.1. Introducción

Para el ser humano es fácil reconocer las regiones de piel de una persona en una imagen, pero lo que ve una máquina no es más que un conjunto de píxeles con ciertos valores RGB (figura 4.1) que es, en definitiva, una región de colores. Está caracteristica es de gran interés en el modelado y deteccion de eventos basados en personas. Por tanto, será interesante buscar formas de diferenciar regiones de piel de regiones de no-piel a partir de dichos valores RGB.

En este capítulo se relatará el estudio realizado de los distintos espacios de colores que se suelen usar para la detección de piel en imágenes y de algunos métodos propuestos para dicho fin. El objetivo es realizar un estudio exhaustivo de dichos espacios de colores analizando sus canales de manera independiente, viendo las relaciones entre ellos, observando como se comportan en distintas imágenes y estudianto si son útiles para la detección de piel. Después, las conclusiones obtenidas en dicho estudio se utilizarán a la hora de comparar varios algoritmos para la deteción

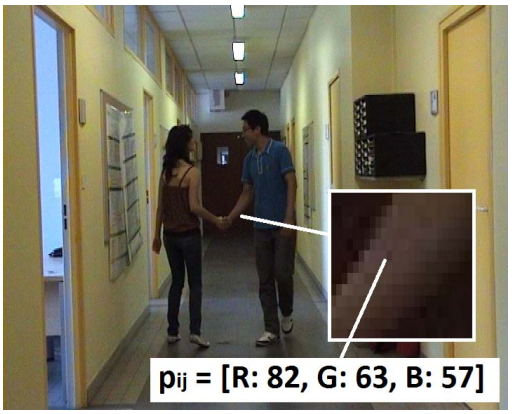


Figura 4.1: Ejemplo de piel a nivel de píxel

de piel, buscando así uno que funcione de manera correcta para los dataset usados. La meta final de este estudio es encontrar un algoritmo de detección de piel que pueda sustituir al actual en el protopipo del VPU-Lab, siempre y cuando éste mejore los resultados obtenidos.

Este capítulo sigue la siguiente estructura: estudio de espacios de color (sección 4.2) y detectores de piel: umbralización, SDMIM y Random Forest (sección 4.3).

4.2. Estudio de espacios de color

Los algoritmos destinados a reconocer piel en imágenes suelen depender del espacio de color en el cual se represente la imágen. Hay espacios de colores en los cuales el color de la piel se diferencia más del resto de colores. Así mismo, hay canales de color dentro de un mismo espacio de colores que discriminan más la piel que otros. La motivación de este estudio es descubrir que espacio de color y que canales son los más adecuados para reconocer la piel con un algoritmo destinado a dicho fin.

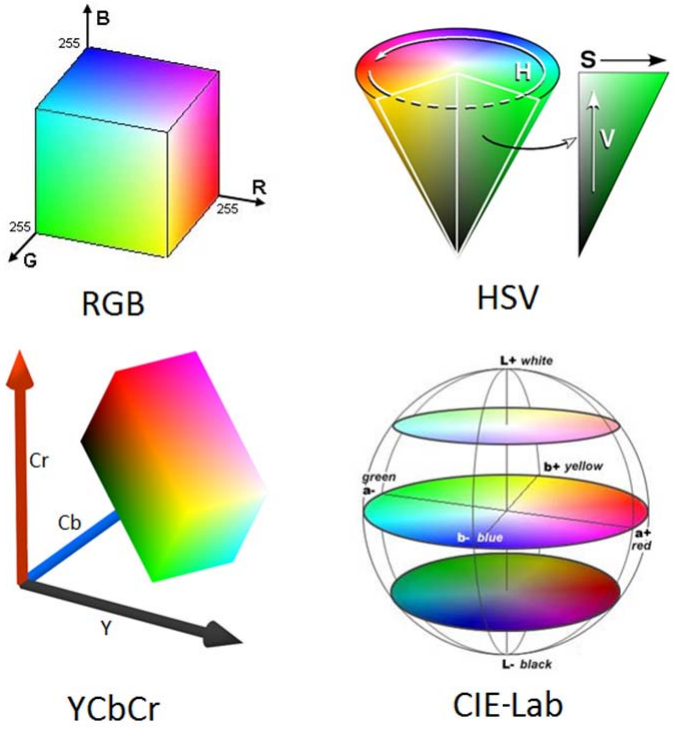


Figura 4.2: Espacios de color utilizados

El espacio de color básico es el RGB (colores primarios *Red, Green, Blue*), y a partir de este, han ido apareciendo numerosos espacios de colores, que no son más que transformaciones del primero. Los espacios de colores elegidos para el estudio son: RGB, HSV (*Hue, Saturation,*

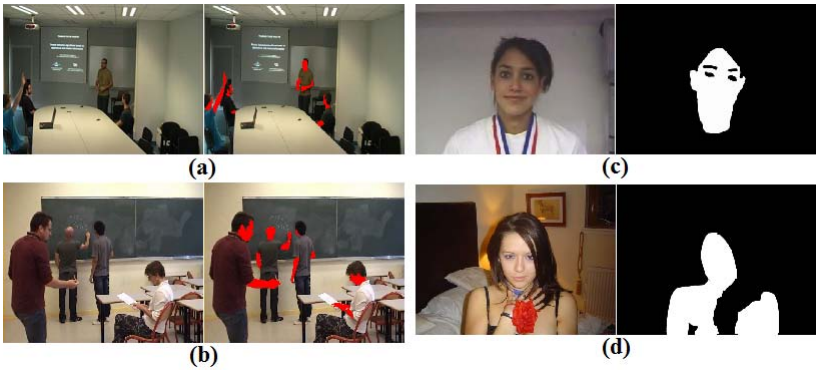


Figura 4.3: Ejemplos dataset para detección de piel
Datasets: (a) ED (b) LIRIS (c) SKIN (d) MCG

Value), YCbCr (*Y luminance, Chrominance Blue, Chrominance Red*) y CIE-Lab (*Lightness, a and b color-opponent dimensions*) (figura 4.2). Se han utilizado estos cuatro espacios por ser los más comunes en detección de piel así como en muchos otros estudios [17], aunque existen muchas más transformaciones de espacios de color que también podrían ser usadas. Las transformaciones de espacios de colores se asume que reducen el solape entre piel y no- piel y proveen de parametros más robustos ante variaciones en la iluminación.

Se utilizaron cuatro datasets disponibles: ED (VPU-Lab) [34], LIRIS (HARL) [6][33], SKIN (vídeos) [35] y MCG (imágenes) [36][37]. La mayor parte del esfuerzo se centró en estudiar los dos primeros dataset, ya que los otros dos no guardan ningún parecido con el escenario que nos interesa. Aun así, fueron también utilizados para obtener más información. Para los dos primeros dataset, compuestos de vídeos, se capturaron entre 25-30 imágenes con diversas situaciones y escenarios y se anotó la piel.

4.2.1. Análisis de canales

En este subapartado se explicarán todos los pasos seguidos en este estudio. Su buscó principalmente analizar cada canal de cada espacio de colores independientemente del resto. Para ello, se obtuvieron los histogramas de cada canal de cada espacio de color, de cada dataset en conjunto y del conjunto global de todos los dataset. A partir de ellos, se extrajeron datos e información para el estudio, a parte de la información visual que ya dan los histogramas.

4.2.1.1. Obtención de histogramas

- **Histogramas de piel para cada imagen de cada datasets, cada canal y cada espacio de color**

  Esto sirvió como primer paso del estudio de canales, ya que es una forma de observar como

queda definida la piel en cada espacio de color en numerosos casos. En las figuras 4.4 y 4.5, ya se puede apreciar como el espacio RGB no presenta ningún canal que represente la piel en un rango pequeño de valores. Esta es la razón por la que no se suele usar este espacio de colores para detección de piel [17].

Estos histogramas sirven además para detectar las imágenes problemáticas en cada dataset, pues si alguna imagen presenta unos histogramas de piel muy distintos al resto de imágenes del dataset, significará que la piel en esa imagen será muy difícil de detectar, por ser tan distinta en color al resto. Las figuras 4.4 y 4.5 muestran las imágenes original y anotada con la piel en rojo, y debajo las cuatro transformaciones de espacios de color con los tres histogramas para los canales de color de cada una.

- **Histogramas de piel para cada datasets, para cada canal de cada espacio de color**

  Esto sirve para obtener una aproximación de los valores que toma la piel en cada dataset y poder compararla con la de otros dataset, y así poder comprobar si un algoritmo de detección de piel podría detectarla en distintos dataset sin tener que adaptarlo a cada uno. El resultado es que para los canales que representan la cromaticidad existen pequeñas diferencias, pero en general la piel suele ser similar en todos los datasets. Para los canales que representan la iluminación, las diferencias son mucho mayores entre datasets. Un ejemplo de esto queda representado en la figura 4.6, referente al espacio HSV, donde se puede apreciar que el canal C1 (H) presenta unos histogramas muy parecidos en los cuatro datasets, mientras que los otros 2 canales no, en especial el canal C3 (V), correspondiente a la iluminación. La línea negra discontinua es la gaussiana ajustada sobre el histograma, que más adelante se explicará su uso. Las figuras 4.7 y 4.8 representan los histogramas para los espacios YCbCr y CIE-Lab.

- **Histogramas globales**

  Estos histogramas han sido obtenidos a partir de todos los dataset juntos para cada canal de cada espacio de color. Con estos histogramas podemos observar cuales son los canales de color que menos varían de un dataset a otro. Si un canal esta claramente acotado en un rango reducido de valores, significará que representa la piel de todos los dataset de la misma manera y además será más fácil umbralizar, lo cual hará mucho más fácil la detección de la misma. En la figura 4.9 se puede observar que los canales que aparentemente mejor se comportan son el canal “H” de HSV, los canales “Cb” y “Cr” de YCbCr y los canales “a” y “b” de CIE-Lab.

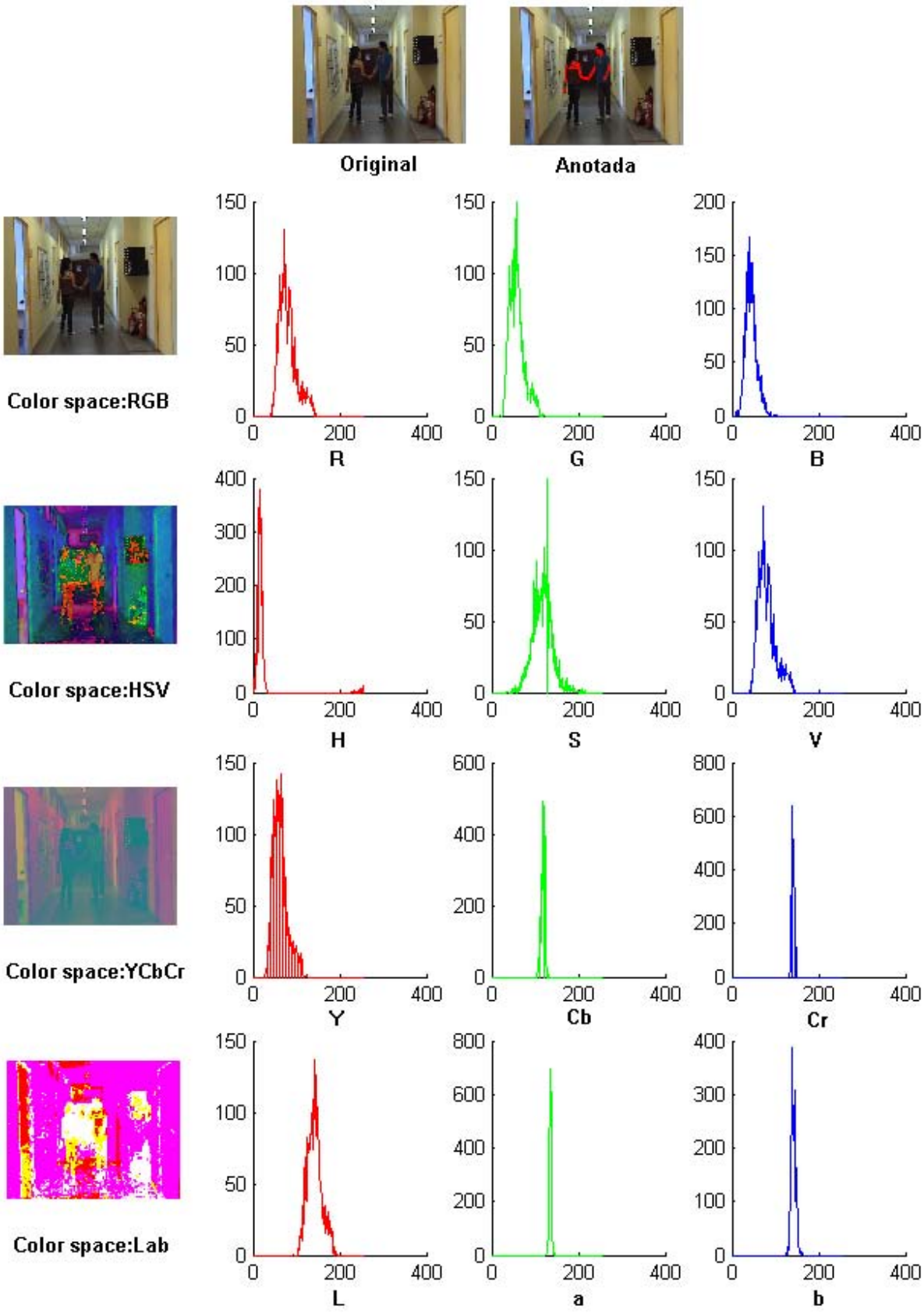


Figura 4.4: Ejemplos de histogramas para una imagen del dataset LIRIS
(para una imagen, histogramas de los 3 canales de los 4 espacios de colores)

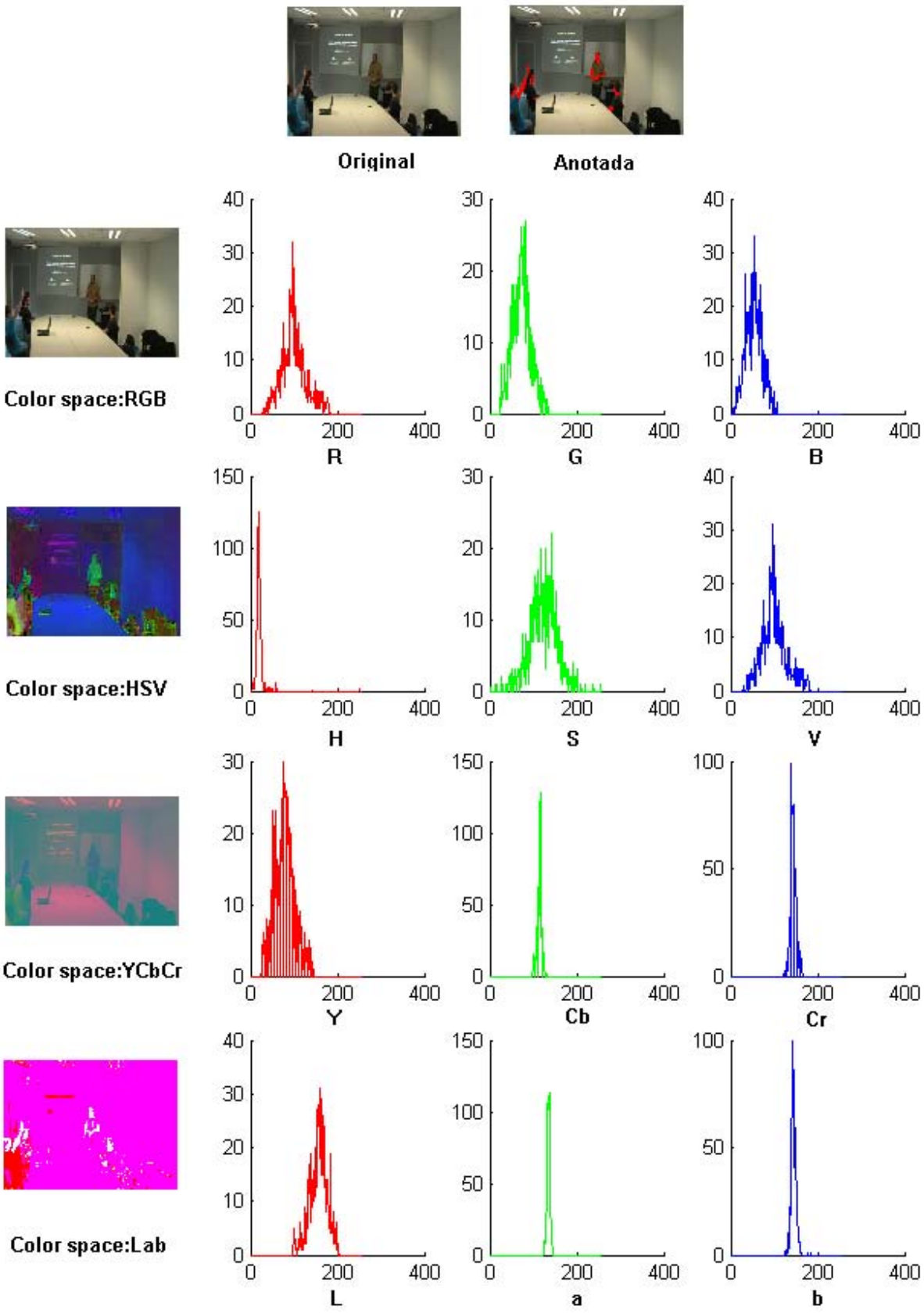


Figura 4.5: Ejemplos de histogramas para una imagen del dataset ED
(para una imagen, histogramas de los 3 canales de los 4 espacios de colores)

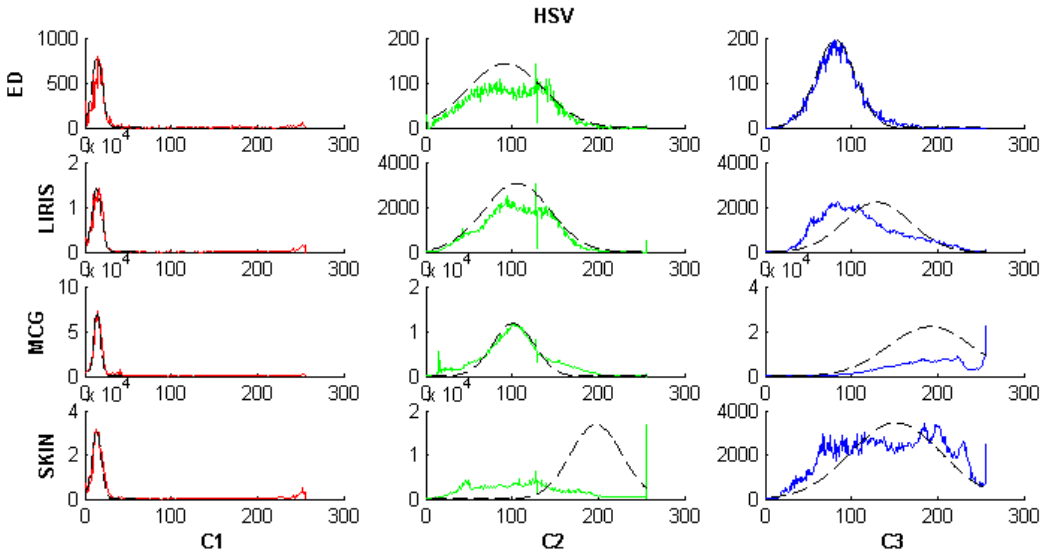


Figura 4.6: Ejemplos de histogramas HSV para cada dataset

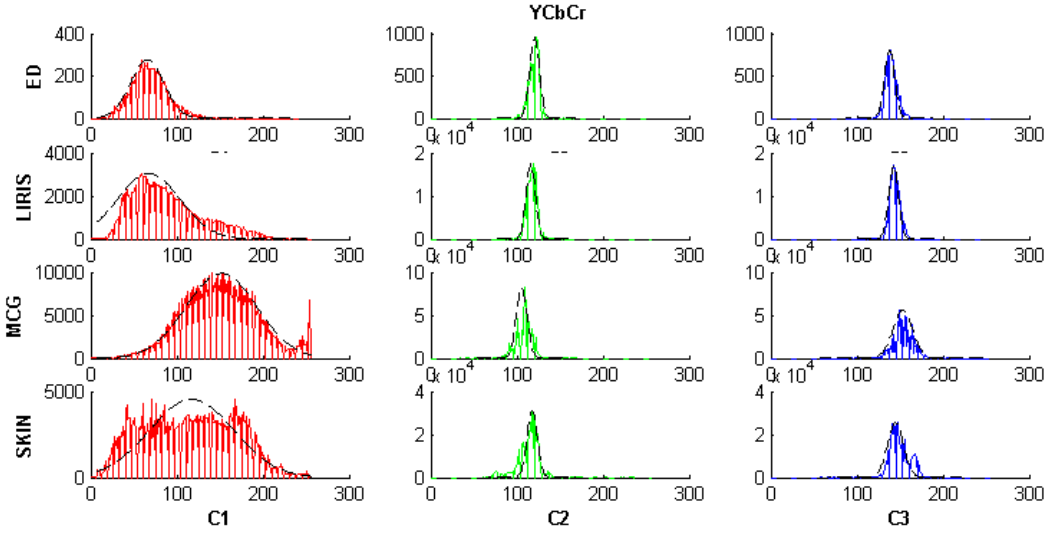


Figura 4.7: Ejemplos de histogramas YCbCr para cada dataset

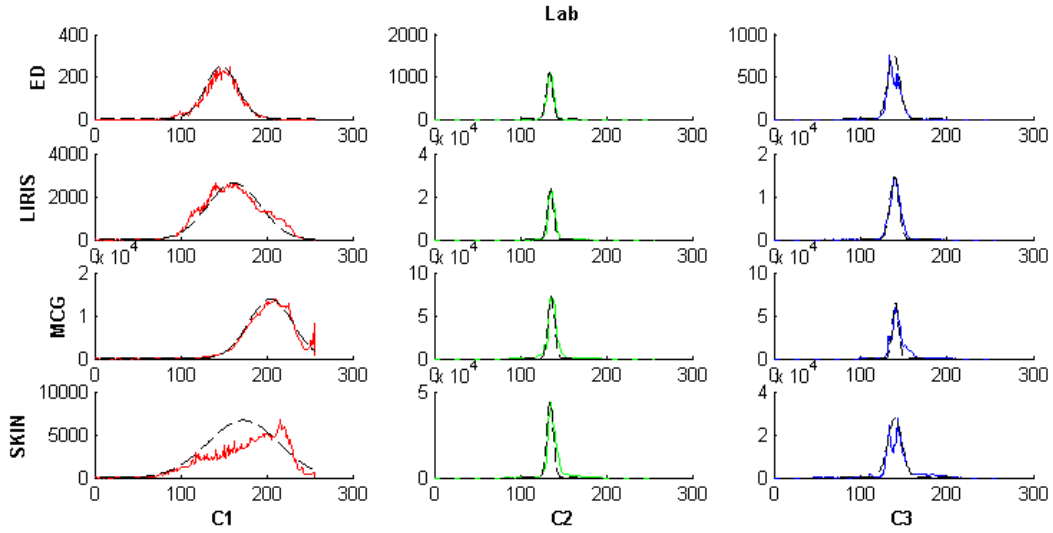


Figura 4.8: Ejemplos de histogramas CIE-Lab para cada dataset

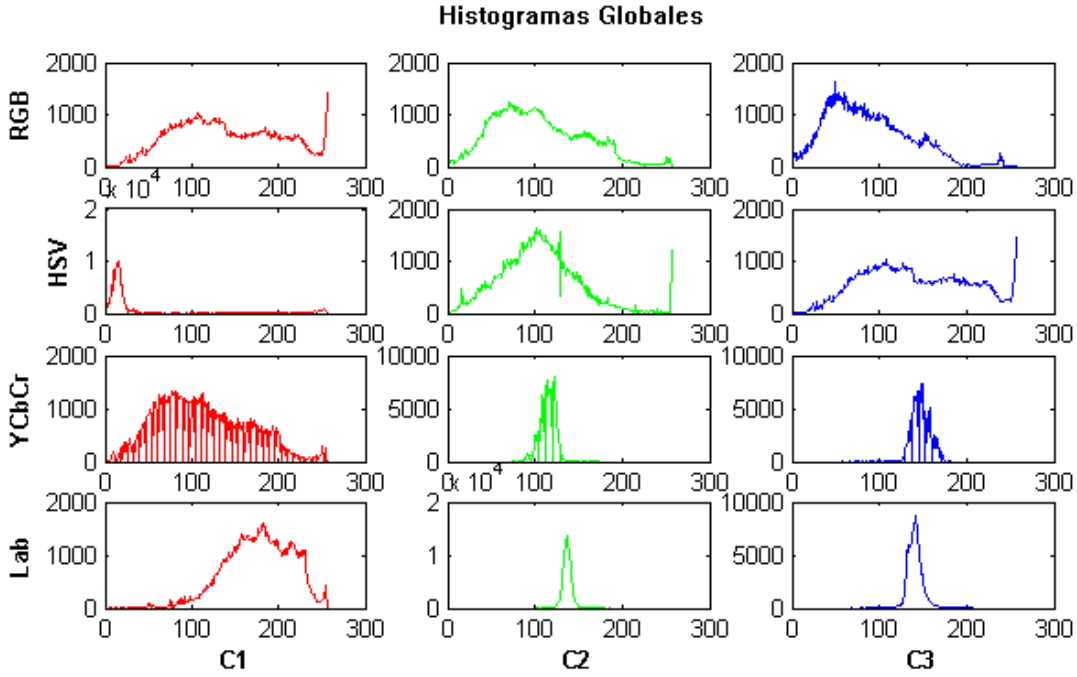


Figura 4.9: Histogramas globales de piel

4.2.1.2. Análisis de los histogramas

A partir de los histogramas, que nos aportan mucha información visual, se obtuvieron los valores de media y desviación de la gaussiana ajustada al histograma. Para obtener estos valores se utilizó la técnica llamada *gaussian fitting*, que busca una gaussiana que se ajuste lo mejor posible a un histograma dado y devuelve los valores de media μ y desviación σ de dicha gaussiana. El valor μ se calcula como la media de las muestras del histograma, y el valor σ se calcula como la variación del $\backsim 70\,\%$ de los elementos alrededor de la media. En las figuras 4.6, 4.7 y 4.8 se pueden obervar con una línea discontinua negra las gaussianas sobre los histogramas.

El objetivo de obtener estos valores es la búsqueda de los canales aparentemente más discriminativos entre piel y no-piel, que serán aquellos que cumplan los dos siguientes requisitos:

- *Media constante entre los distintos dataset:* El canal que cumpla esto es capaz de representar la piel de distintos datasets de la misma manera, permitiendo así establecer unos umbrales para reconocer la piel comunes a cualquier dataset.

- *Desviación muy pequeña:* Si se cumple esto, significará que la gaussiana es muy estrecha y que la piel queda muy bien diferenciada del resto en dicho canal de color. Ese canal reunirá los valores correspondientes a la piel en un pequeño rango de valores, dejando el resto del espectro de color para todo aquello que no sea piel. Por tanto, con unos umbrales muy cercanos entre sí se puede detectar la piel sabiendo que no producirá demasiados falsos positivos.

En la tabla 4.1 se muestran marcados los cinco canales de colores que presentan una media μ más constante entre los cuatro dataset. Esta constancia de la media se mide con la desviación estándar de la media entre los cuatro dataset, presentada en la última fila de la tabla, y cuanto menor sea, mejor será el canal. Por otro lado, en la tabla 4.2 se muestra la media realizada entre los cuatro dataset de la desviación σ para todos los canales de color, donde se destacan en negrita los que presentan una media de la desviación más pequeña. Si la media de la desviación entre los cuatro dataset es pequeña, indicará que dicho canal discrimina la piel muy bien para cualquier dataset.

A partir de estos datos, se puede observar que los mismos cinco canales que presentan una media muy constante son los mismos que presentan una media de la desviación más pequeña. Esto significa que dichos canales son muy buenos para detectar piel en todos los dataset.

Resultado: “H”, “a”, “b”, “Cb”, “Cr” son los canales que mejor cumplen las condiciones descritas anteriormente, y por tanto, serán en los que más nos centremos.

Una medida adicional que se obtuvo fue la correlación entre canales. Para esto fue utilizada la correlación de Pearson para matrices bidimensionales. Este valor sirve como medida de la información (similitud) que aporta cada canal, para poder buscar aquellos canales que den

Media μ									
	H	S	V	Y	Cb	Cr	L	a	b
ED	**14,160**	92,844	81,726	65,395	**120,278**	**137,540**	147,546	**133,825**	**138,538**
LIRIS	**13,782**	105,788	130,070	67,169	**115,915**	**142,302**	161,882	**135,058**	**139,197**
MCG	**14,173**	101,470	191,787	151,594	**105,415**	**153,048**	204,523	**136,023**	**140,537**
SKIN	**13,692**	197,702	151,833	117,236	**117,530**	**144,516**	172,988	**134,609**	**139,769**
Desv	**0,217**	42,547	39,721	36,173	**5,630**	**5,618**	20,969	**0,794**	**0,736**

Tabla 4.1: Medias μ del ajuste gaussiano de los canales de color de interés para cada dataset y desviación (desv) entre datasets

Media de la desviación σ			
	C1	C2	C3
HSV	**5,269**	34,051	40,664
YCbCr	36,543	**5,846**	**8,088**
CIE-Lab	29,022	**3,734**	**6,093**

Tabla 4.2: Media entre los cuatro dataset para la desviación σ del ajuste gaussiano de cada canal de color

información de piel diferente. Si el valor de esta función es 0, será totalmente diferente, si es 1, será igual.

La medida fue realizada para los canales "H" y "S" de HSV (usamos "S" ya que es el canal típicamente usado con "H" para detectar piel), los canales "Cb" y "Cr" de YCbCr, y los canales "a" y "b" de CIE-Lab, que son los canales de color de interés según el estudio anterior de la media y la desviación de la gaussiana ajustada.

Los resultados obtenidos (tabla 4.3) han sido calculados a partir de la acumulación de píxeles de piel previa a la creación de histogramas, primero para cada dataset y después haciendo la media entre todos los dataset. También se obtuvieron las correlaciones de manera global, acumulando los píxeles de todos los dataset, y el resultado fue similar.

Resultado: "H" y "a" parecen ser los canales más distintos, además de ser de los más discriminativos.

Correlación entre canales						
	H	S	Cb	Cr	a	b
H	1	0,266	0,397	0,250	**0,094**	0,386
S	0,266	1	0,595	0,572	0,569	0,848
Cb	0,397	0,595	1	0,748	0,147	0,830
Cr	0,250	0,572	0,748	1	0,638	0,572
a	**0,094**	0,569	0,147	0,638	1	0,237
b	0,386	0,848	0,830	0,572	0,237	1

Tabla 4.3: Correlación entre canales

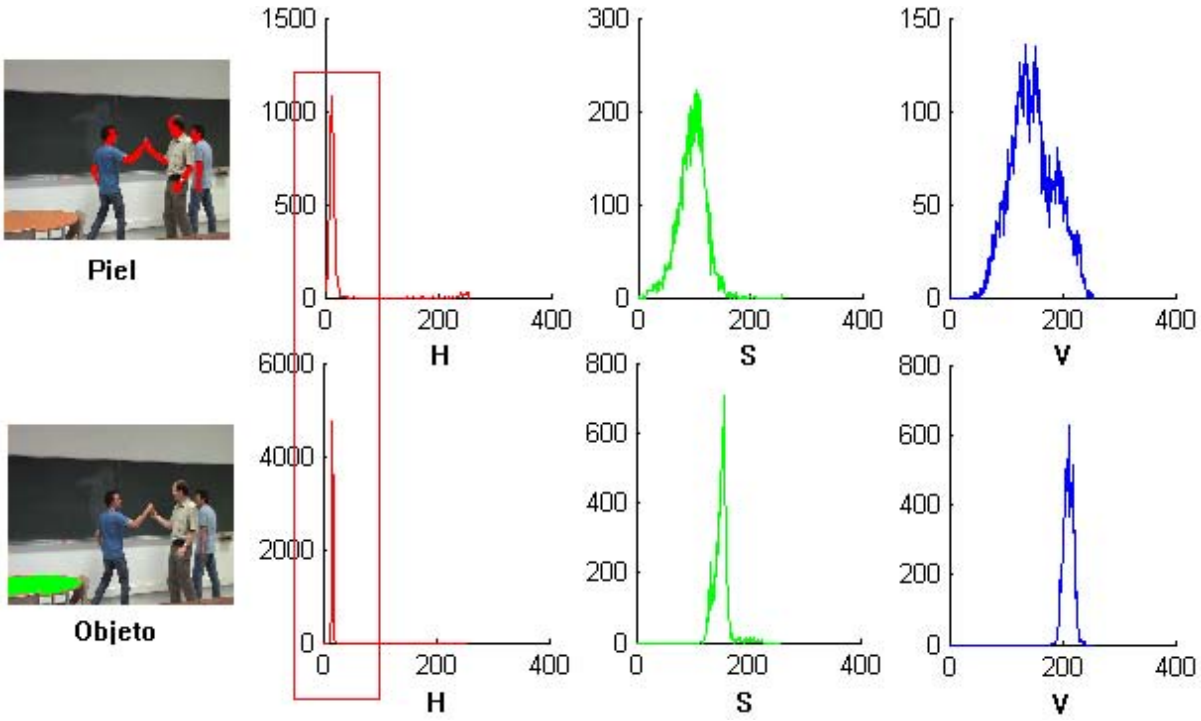


Figura 4.10: Ejemplo de histogramas HSV de piel vs objeto

4.2.2. Piel vs No-Piel

Por último, se dió un último paso en este estudio para demostrar que, a pesar de haber comprobado que hay canales muy discriminativos, aún hay ciertas regiones de no-piel en los vídeos que son imposibles de distinguir de la piel. Además, se realizó una comparativa entre los histogramas de piel y los de no-piel para los dos canales que resultaron ser más interesantes: "H" y "a".

- **Histogramas de piel vs objetos**

  Se obtuvieron los histogramas de color de algunos objetos con color similar a la piel, para saber hasta que punto se parecen a la piel. Esto se realizó principalmente sobre el dataset LIRIS, por ser el que más problemas daba al respecto. Un ejemplo de esto son los histogramas presentados en la figura 4.10, 4.11 y 4.12. En la parte superior de las imágenes aparecen los histogramas de los canales de color usados para la piel (marcada en rojo en la imagen anotada) y en la parte inferior los histogramas de color del objeto anotado (marcada en verde en la imagen anotada). Si comparamos los histogramas marcados con un cuadro rojo, que son precisamente los relacionados con el color para esos espacios de color, vemos que los objetos podrían ser confundidos con piel, ya que presentan unos histogramas muy similares.

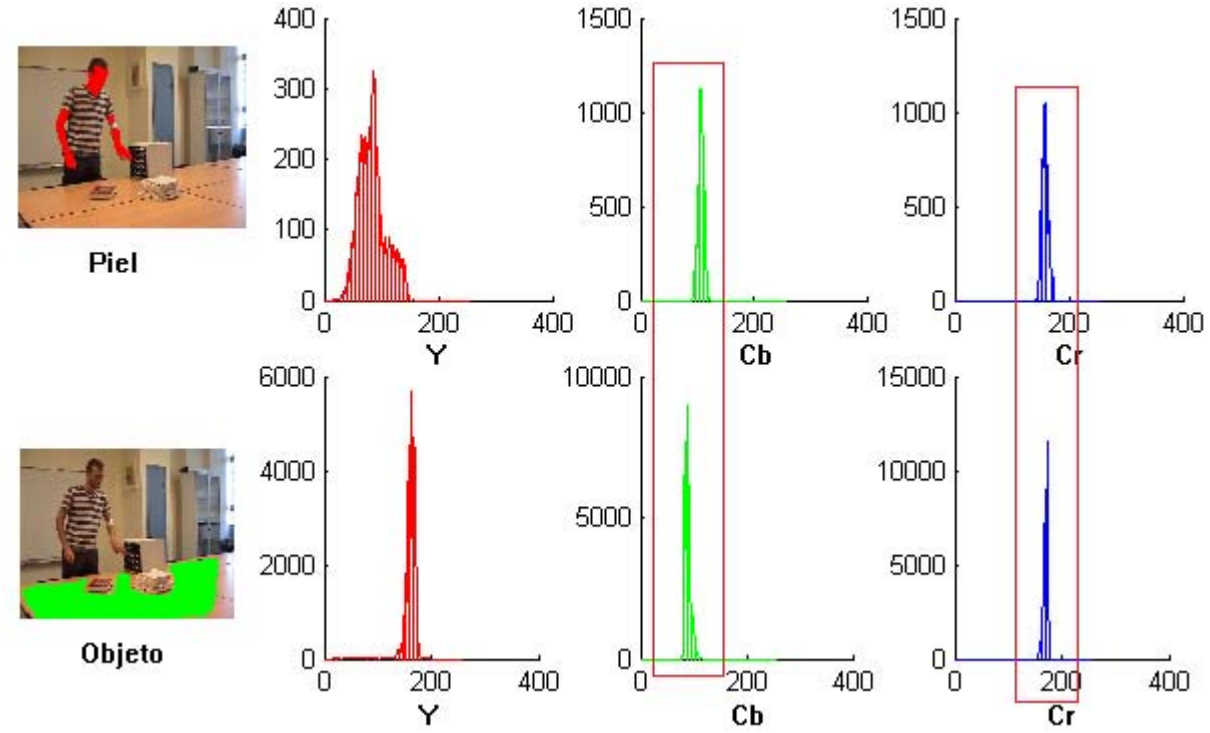


Figura 4.11: Ejemplo de histogramas YCbCr de piel vs objeto

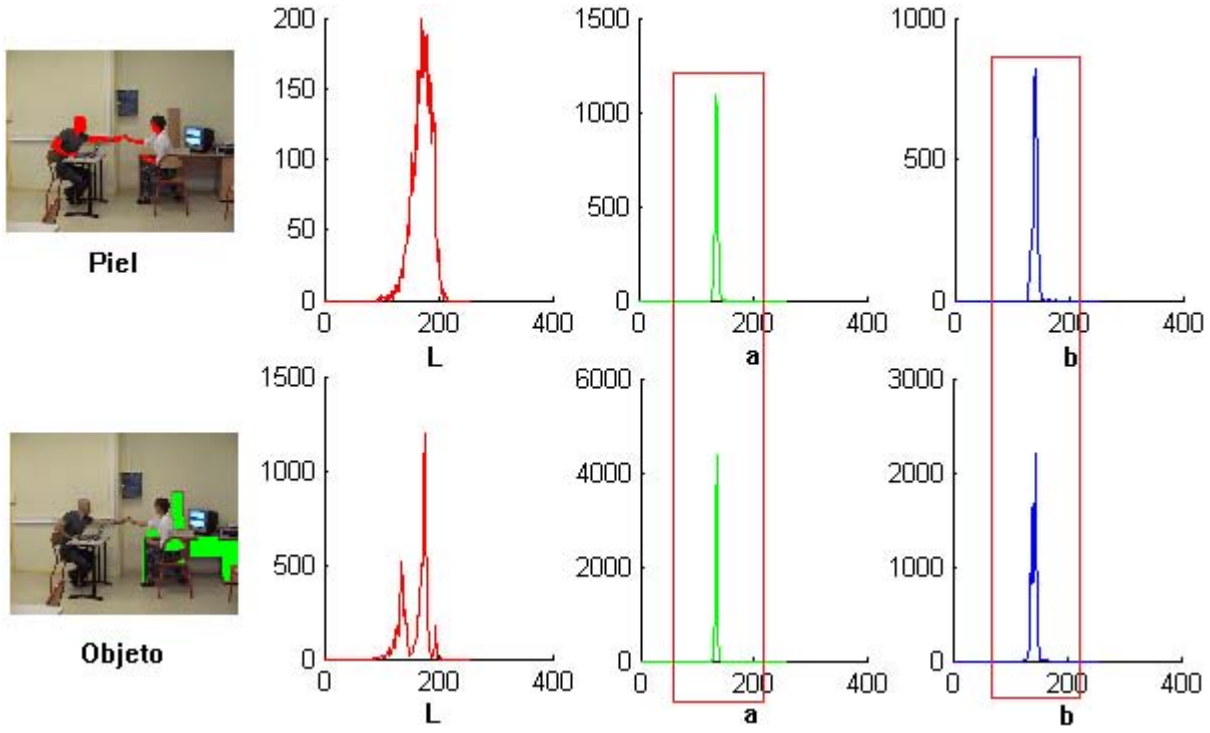


Figura 4.12: Ejemplo de histogramas CIE-Lab de piel vs objeto

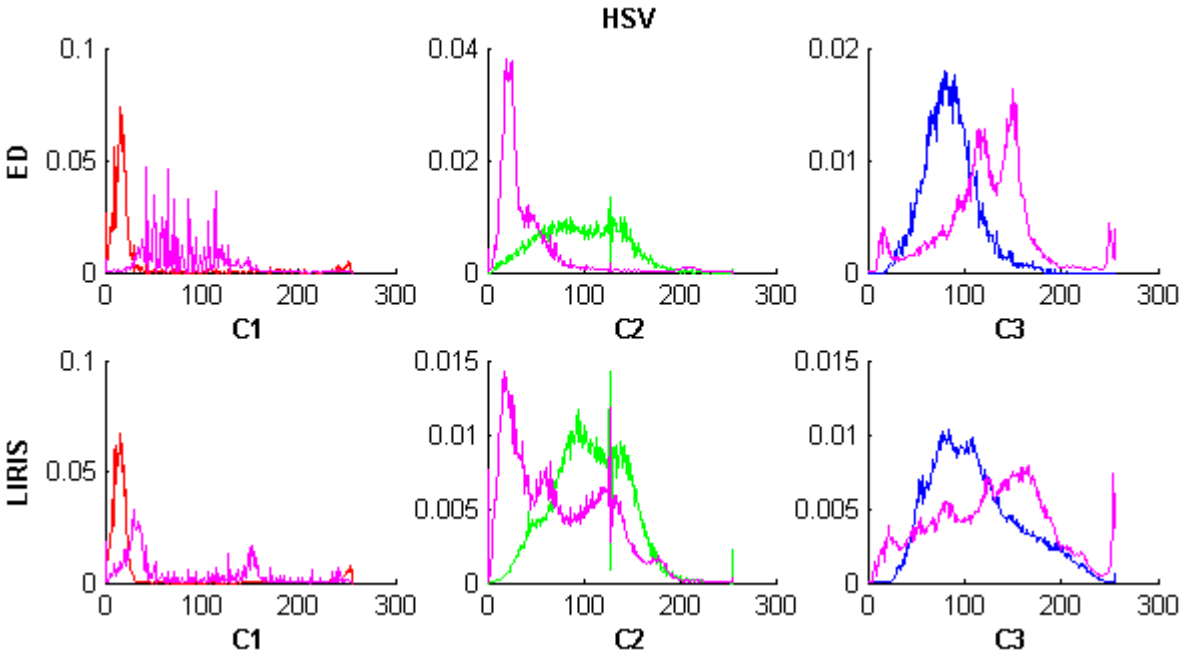


Figura 4.13: Histogramas piel vs no-piel para HSV

- **Histogramas de piel vs no-piel normalizados**

  Se obtienen calculando los histogramas de todo lo que no sea piel y comparando estos con los anteriores extraídos para la piel. Resulta útil para determinar si los canales escogidos discriminan bien piel de no-piel. Si ambos histogramas solapan, significará que ese canal no es útil para detectar piel, ya que detectará mucha no-piel como piel. En las figuras 4.13, 4.14 y 4.15 se presentan ejemplos de estos histogramas, donde en violeta aparecen los histogramas de no-piel.

  Tras observar las gráficas, se puede afirmar que el canal "H" de HSV y el canal "a" de CIE-Lab son los canales para los cuales aparentemente menos solapan los histogramas de piel y no-piel.

  Por último, se calculó la distancia de Bhattacharyya entre histogramas de piel y no-piel. Esta complementa la visualización de los histogramas de piel vs no-piel, aportando información del parecido de dichos histogramas entre sí.

  El resultado no es muy concluyente, ya que todos los histogramas solapan algo, aunque algunos aparentemente no solapen apenas (figura 4.13). En la tabla 4.4 se puede observar que los canales "H", "a" y "Cr" son los que presentan menor distancia (menor sopale) dentro de cada dataset, en especial para los dataset ED y LIRIS, que son los que más nos interesan.

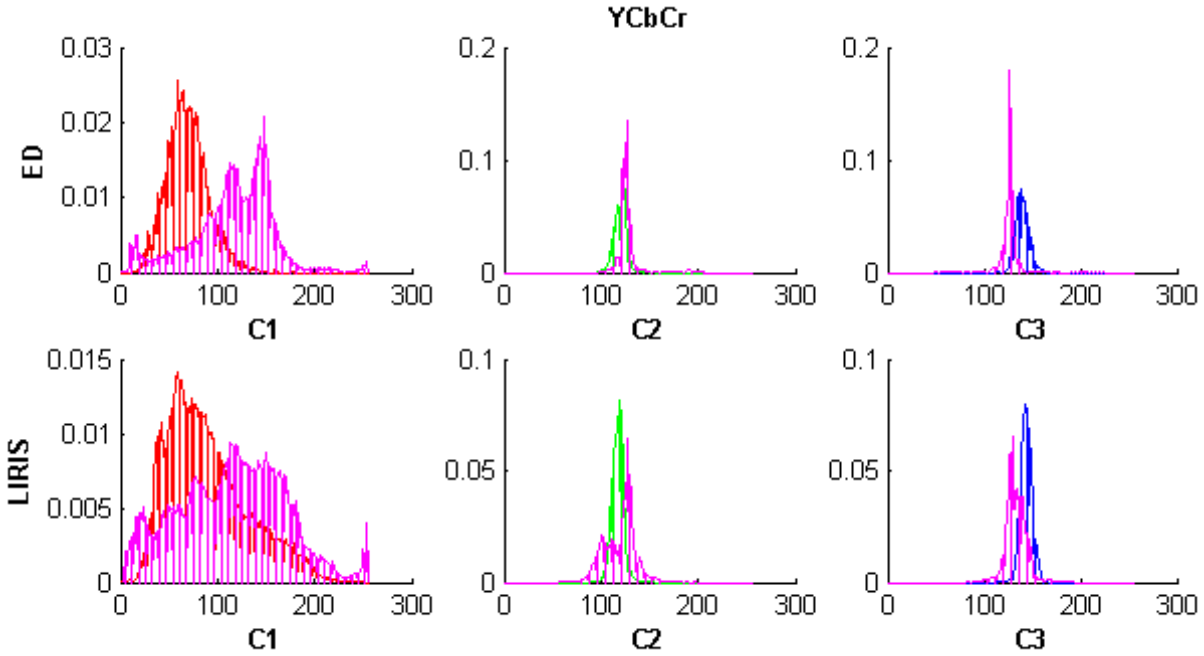


Figura 4.14: Histogramas piel vs no-piel para YCbCr

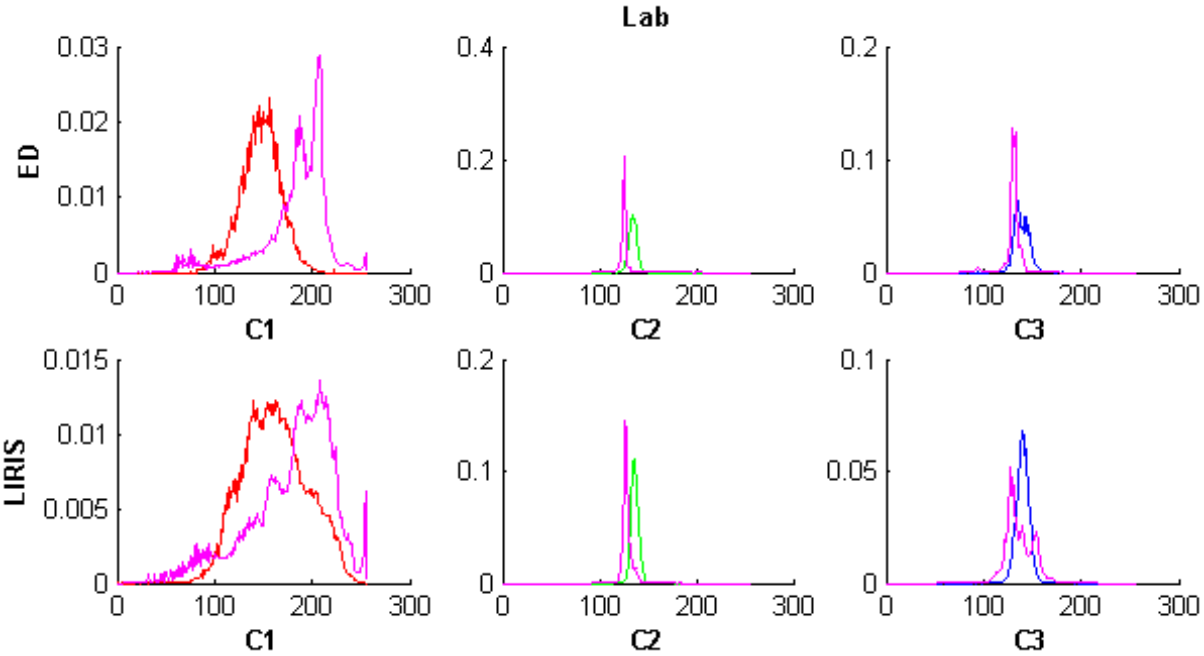


Figura 4.15: Histogramas piel vs no-piel para CIE-Lab

D. Bhattacharyya Piel/Fondo									
	H	S	V	Y	Cb	Cr	L	a	b
ED	**0,346**	0,6674	0,771	0,634	0,846	**0,450**	0,616	**0,359**	0,770
LIRIS	**0,543**	0,871	0,936	0,921	0,747	**0,706**	0,915	**0,559**	0,838
MCG	**0,701**	0,871	0,849	0,865	0,740	**0,652**	0,868	**0,773**	0,806
SKIN	**0,809**	0,921	0,968	0,976	0,892	**0,755**	0,978	**0,780**	0,934

Tabla 4.4: Tablas de Distancia Bhattacharyya para HSV, YCbCr y CIE-Lab

Consideraciones: Todos los histogramas están representados de 0 a 255, pero en realidad cada espacio de color tiene un rango distinto de valores. Por ejemplo, “a” y “b” se mueven en un rango de -110 a 110. Ahora bien, en los histogramas estos canales agrupan todo lo que es no-piel muy cerca de la piel, y aunque van de 0 a 255, ambos grupos siempre están entre 100 y 200.

4.2.3. Conclusión

Los canales que han resultado ser más discriminativos según los datos han sido el canal “H” de HSV y el canal “a” de CIE-Lab. Por tanto, aunque para las pruebas que se realizaron se usaron más canales de colores, hay que poner especial atención en estos dos.

4.3. Detectores

A partir del estudio anterior, se desarrollaron tres detectores distintos: uno basado en la clásica umbralización buscando el mejor umbral, otro desarrollado a partir de un detector de sombras, y un último desarrollado a partir del algoritmo *Random Forest.*

4.3.1. Umbralización

La umbralización se puede usar sobre imágenes con la finalidad de segmentar, es decir, separar los objetos de una imagen que nos interesen del resto. Se trata de asignar cada píxel a un cierto grupo, llamado comúnmente "segmento". El segmento se establece mediante dos valores umbrales, inferior y superior. Si el valor de un píxel cae dentro de ese segmento, será asignado al grupo de interés. La ecuación 4.1 representaría un segmento para el canal C:

$$TH_{inf} < C_i < TH_{sup} \tag{4.1}$$

donde C_i representa el valor del píxel i para el canal C y TH_{inf} y TH_{sup} los umbrales que definen el segmento. Si ese píxel cumple la condición de la ecuación, será reconocido como parte del segmento.

Este método se puede usar en detección de piel, buscando aquellos umbrales que definan el mejor segmento que represente la piel en cada canal del espacio de color usado. De esta

manera, si los valores de un píxel para todos los canales de colores establecidos caen dentro de los segmentos de dichos canales, este píxel será reconocido como piel.

Para encontrar estos umbrales, se ha realizado un estudio a partir de los histogramas obtenidos para cada dataset. A partir del máximo en el histograma, se va descendiendo hacia ambos lados y sumando el número de píxeles en cada punto, hasta reunir un 80 % de los píxeles del dataset. De esta manera, se obtienen dos valores umbrales, uno a cada lado del máximo del histograma. Se realizó esto para cada histograma de cada canal de color. En la figura 4.16 se puede ver un ejemplo de los umbrales obtenidos para los canales de HSV en los 4 dataset.

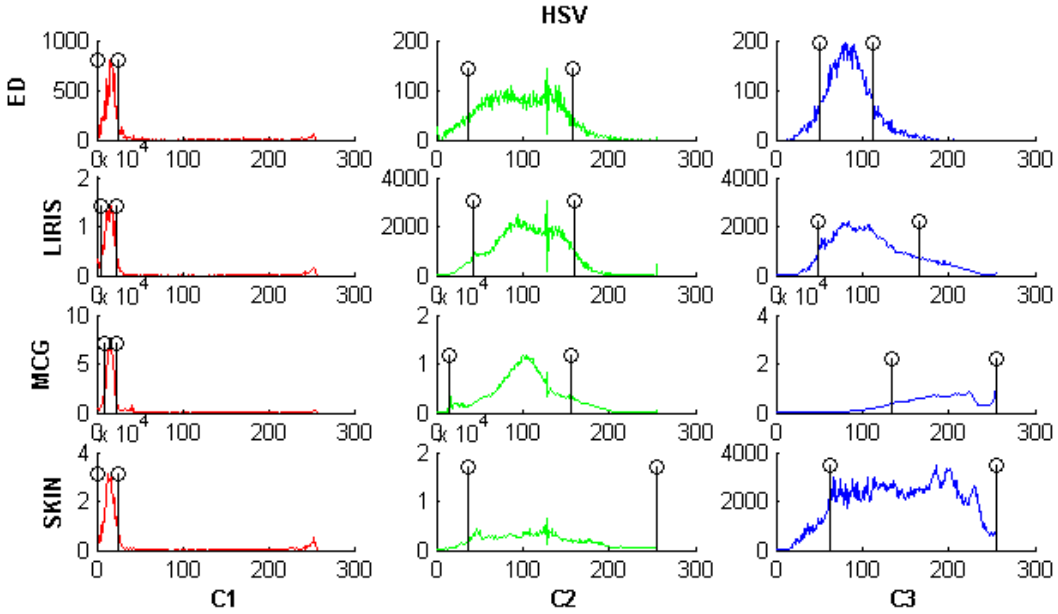


Figura 4.16: Ejemplo de histogramas con umbrales para HSV

En las tablas 4.5 y 4.6 se muestran la media y desviación para los 4 dataset de los umbrales para cada canal. En la primera tabla están marcados los canales que tienen unos umbrales medios más cercanos entre sí, y en la segunda tabla los que presentan una menor desviación. Ambas características son las más deseables para los umbrales, ya que discriminarán mejor la piel.

Media umbrales						
	C1		C2		C3	
	low	high	low	high	low	high
HSV	**4**	**23**	33	182	74	198
YCbCr	50	171	**104**	**125**	**136**	**159**
CIE-Lab	130	216	**131**	**143**	**131**	**151**

Tabla 4.5: Media de umbrales de los 4 dataset

Desviación umbrales						
	C1		C2		C3	
	low	high	low	high	low	high
HSV	**2,947**	**1,299**	10,592	42,605	35,686	61,321
YCbCr	25,955	59,180	**8,584**	**1,920**	**2,872**	**9,808**
CIE-Lab	24,753	30,605	**1,000**	**2,062**	**1,581**	**2,693**

Tabla 4.6: Desviación de umbrales de los 4 dataset

A partir de los valores obtenidos para los umbrales, se realizó un detector de piel por umbralización. Como umbrales se escogieron la media de los umbrales obtenidos en el estudio, y se utilizaron sólo aquellos canales que se mantienen más constantes para los cuatro dataset, es decir, aquellos con una desviación menor.

4.3.2. Algoritmo SDMIM

4.3.2.1. Introducción

El algoritmo SDMIM (*Skin Detection by Mutual Information Maximization*) es una adaptación de la técnica para la detección de sombras desarrollado en el VPU-Lab [38], modificada para que detecte piel en imágenes mediante umbralización. Ahora bien, el método de búsqueda de umbrales del prototipo también ha sido modificado.

4.3.2.2. Descripción del algoritmo original

El algoritmo original está compuesto por tres detectores distintos, que trabajan sobre el espacio de color HSV, cuyo objetivo es detectar sombras en imágenes. El primero está basado en la intensidad (V), y es controlado por dos umbrales α y β. El segundo utiliza la crominancia, teniendo solo en cuenta la saturacion (S), pues se asume que las sombras sólo reducen la saturación del color de los píxeles, y está controlado por un solo umbral τ_S. Por último, el último detector es un algoritmo de textura controlado por un solo umbral d.

Este algoritmo realiza una búsqueda de los cuatro umbrales que den un mejor valor de *agreement.* Este valor se utiliza como valor de referencia para determinar si el sistema acierta o no a la hora de reconocer piel. La ecuación 4.2 describe el *agreement* usado en este algoritmo:

$$Agreement = F(SM_I(\alpha,\ \beta),\ SM_S(\tau_S),\ SM_T(d)) \tag{4.2}$$

donde F es la función que calcula el *agreement* a partir de los mapas obtenidos con cada umbral: SM_I para la intensidad, SM_S para la saturación, y SM_I para la textura. El cálculo de *agreement* se calcula mediante correlación de los mapas. A mayor correlación, mayor parecido entre mapas, lo que supone que la suma de los tres mapas dará un mapa con las sombras

detectadas correctamente.

Para obtener el mejor valor de *agreement* posible se realiza una búsqueda dinámica de los mejores umbrales. Para ello se usa un algoritmo de gradiente ascendente. Si el valor inicial de los parámetros está bien escogido, debería resultar un buen *agreement* para los tres detectores independientes. El proceso de maximización de *agreement* se define de la siguiente forma:

$$S_i = S_{i-1} + \eta \nabla F(S_{i-1}) \tag{4.3}$$

donde $S_i = \alpha_i, \beta_i, \tau_{i,S}, d_i$ son los parámetros escogidos en cada iteración i, $\nabla F(S_i)$ es el gradiente de la función particularizado a Si y $\eta > 0$ es una constante. La optimización de los umbrales se realiza en dos etapas con distinta precisión en la búsqueda. Esta precisión se controla con η. En la primera etapa se buscan los umbrales óptimos variándolos en saltos más gruesos que en la segunda etapa, donde se buscarán umbrales alrededor del umbral establecido en la primera etapa, pero con variaciones más finas.

4.3.2.3. Modificaciones

La modificación propuesta tiene como objetivo adaptar este algoritmo a la tarea de detección de piel. Esto implica la utilización de los dos canales de color relativos a la crominancia en el espacio de color usado, probando diferentes combinaciones de estos. Los canales de color utilizados son: H, S, a, b, Cb, Cr.

En primer lugar, se utilizan las medias de la media y desviación de las gaussianas ajustadas al histograma de los cuatro dataset obtenidas en el estudio anterior (véase 4.2.1.2) para generar gaussianas que representen a los histogramas de piel de los canales que se estén usando. Dichas gaussianas se normalizan, de manera que su valor máximo sea 1 y el mínimo 0.

Después, se establece un umbral referente a la probabilidad de que el valor de un píxel esté contenido en la gaussiana, umbral que puede variar de 0 a 1, descrito mediante la siguiente ecuación:

$$piel = \left\{ \begin{array}{lll} 1 & si & f(x) > Th \\ 0 & resto & \end{array} \right\} \tag{4.4}$$

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{4.5}$$

donde $f(x)$ es la función de la gaussiana, μ y σ son la media y la desviación obtenidas previamente, y Th es el umbral establecido.

Dicho umbral controla la desviación que puede tener el valor de un píxel respecto de la media, por tanto, establece dos umbrales alrededor de la media. Por ejemplo, si el umbral se sitúa en 1, significará que sólo se reconocerán como piel aquellos píxeles cuyo valor tenga una probabilidad

del 100 % de caer dentro de la gaussiana, es decir, aquellos con una desviación de 0 respecto de la media. Esto implica que el rango de valores que se reconocerán como piel será sólo un punto, la media.

En la figura 4.17 se puede ver como escogiendo un umbral TH_{prob} se pueden regular a la vez los dos umbrales TH_1 y TH_2, que son los que limitan el rango de valores que se escogen como piel alrededor de la media. Los píxeles cuyo valor caiga dentro de ese rango serán reconocidos como piel.

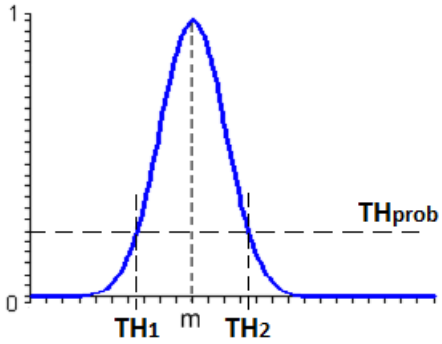


Figura 4.17: Ejemplo de la búsqueda de umbrales

De esta manera, los umbrales se reducen de 4 a 2 respecto al algoritmo original, uno para cada canal de color usado. Esto aumenta mucho la velocidad del programa a costa de que la precisión para reconocer piel baje un poco, ya que realmente es una forma de mover los umbrales de dos en dos. En la figura 4.18 se muestra una comparación entre el algoritmo con 4 umbrales y con 2 umbrales, en la cual se puede ver como con 4 umbrales reconoce un poco mejor la piel, aunque también tiene más falsos positivos.

La búsqueda de los umbrales se realiza igual que en el algoritmo original, partiendo de un valor umbral de 0.5 y buscando en los alrededores de este para ambos umbrales, primero con un barrido grueso, y luego un barrido fino. En ningún caso se permite que los umbrales sean 0 ó 1, pues como se ha explicado, estos valores suponen coger todo el rango de valores de un canal o coger un rango de valores nulo, respectivamente, lo cual no tiene sentido.

En la figura 4.19 se muestra un ejemplo de la variación de una máscara resultante del barrido que realiza el algoritmo sobre un canal de color. En cada imagen se puede ver como sucesivamente las regiones detectadas como piel, representadas en blanco, van reduciéndose debido a que el umbral va aumentando, lo que hace que el rango de valores que representa la piel se reduzca.

Como *agreement* se utiliza el resultado de la correlación entre las máscaras de piel obtenidas para cada canal de color (C_n):

$$Agreement = |F(SM_1, SM_2)| \tag{4.6}$$

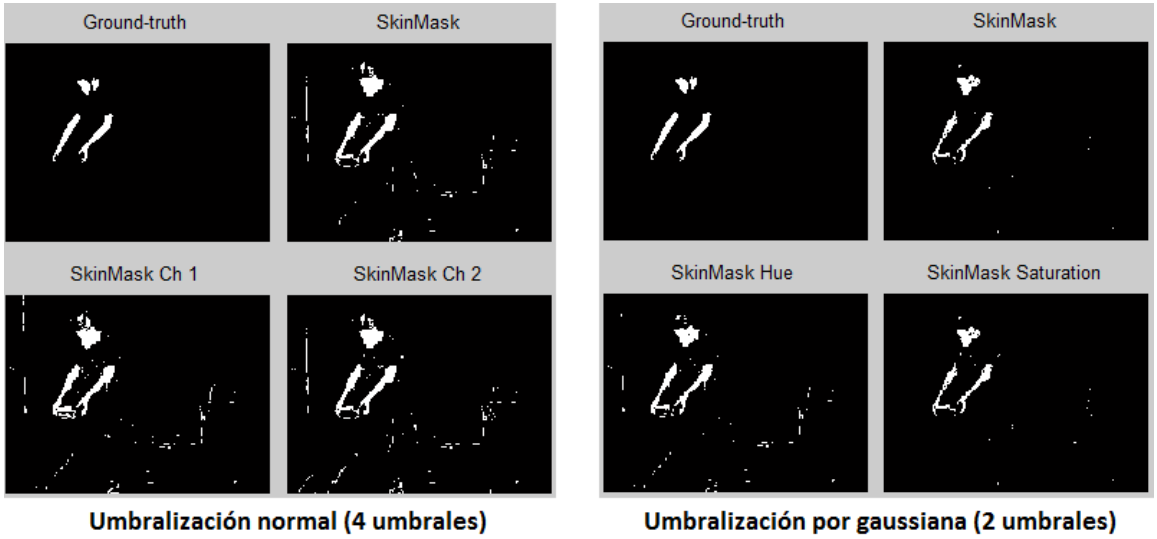


Figura 4.18: Ejemplo del algoritmo con 4 y 2 umbrales

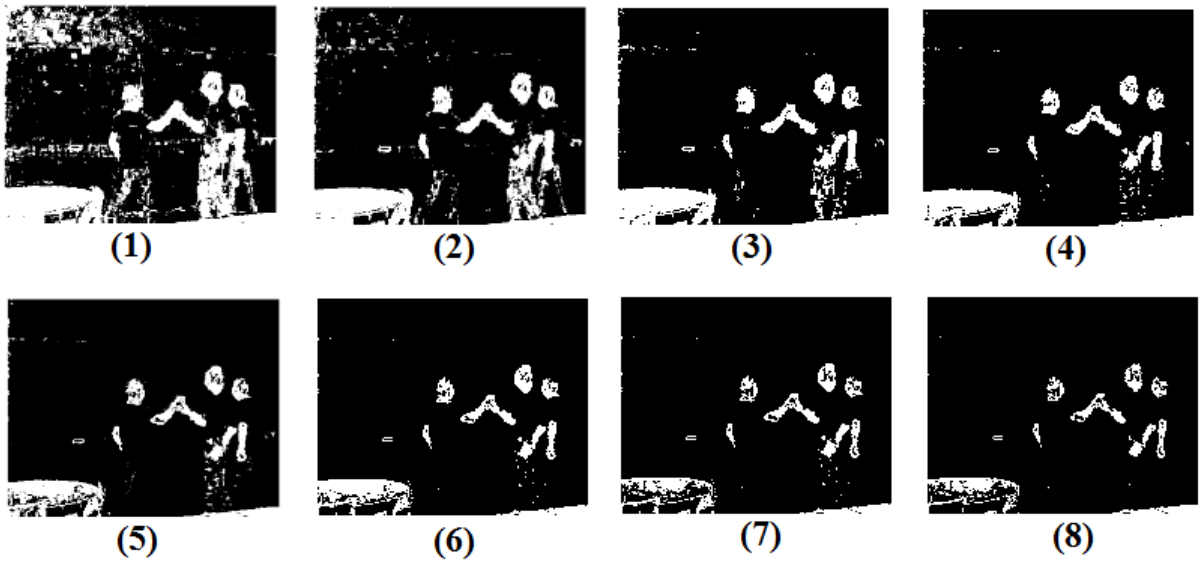


Figura 4.19: Ejemplo de la evolución de una máscara de piel para un canal de color
Cada imagen representa la máscara de piel obtenida de un sólo canal con un umbral cada vez más cercano a 1, de manera que el detector cada vez es más discriminativo.

donde F es la función que calcula el *agreement* mediante correlación de Pearson, que mide el parecido entre dos máscaras, y SM_1 y SM_2 son las máscaras correspondientes a la umbralización de 2 canales.

De esta manera, si comparamos las máscaras de piel que nos da este *agreement* con el *groundtruth*, obtenemos valores de TPR (*True Positive Rate*) y FPR (*False Positive Rate*).

4.3.2.4. Comparativa con umbrales óptimos

Para estudiar el funcionamiento óptimo del algoritmo SDMIM, se propuso obtener los umbrales óptimos mediante una búsqueda de umbrales mucho más fina y exhaustiva, y utilizando el *groundtruth* (mapas binarios indicando las regiones de piel de la imagen).

En primer lugar, se realizó un barrido de 0 a 1 para la probabilidad umbral ejecutando el programa para cada valor. En este punto se podría comenzar en los valores obtenidos del estudio de canales de color, pero puesto que el objetivo es buscar los umbrales óptimos para una futura comparativa de los resultados del algoritmo, se realiza el barrido completo para asegurarse el resultado óptimo, ya que en este paso no importa el tiempo de ejecución.

Como *agreement* (A) se utilizó una combinación del porcentaje de positivos verdaderos (TP) y del de falsos positivos (FP), que se extraen comparando las máscaras de piel obtenidas en cada ejecución del barrido con el *groundtruth* (anotación de piel) (GT) de las imágenes. Se busca que la combinación de los dos valores sea lo mayor posible, es decir, que el primero sea lo mayor posible y el segundo lo menor posible. Las siguientes ecuaciones definen el *agreement*:

$$\begin{aligned} A &= (A(1) + (1 - A(2)) \\ A(1) &= \sum_i (\sum_j TPmap(i,j)) / \sum_i (\sum_j skinGTmap(i,j)) \\ A(2) &= \sum_i (\sum_j FPmap(i,j)) / \sum_i (\sum_j \overline{skinGTmap(i,j)}) \end{aligned} \tag{4.7}$$

donde $TPmap$ es el mapa de *True Positives,* $FPmap$ es el mapa de *False Positives* y *skinGTmap* es la máscara de piel del *groundtruth.*

Así, se obtienen los umbrales "óptimos" y una medida de hasta dónde puede llegar la precisión del programa para los canales de color que se prueben.

4.3.3. Algoritmo *Random Forest*

La idea de probar un sistema de reconocimiento de piel basado en el algoritmo *Random Forest* surge del estudio presentado en [18], donde realizan una comparativa de varios algoritmos de entrenamiento y concluyen que el mejor algoritmo es el *Random Forest.* Además, hay que destacar que en dicho trabajo realizan una comparativa usando 2 y 3 canales de color de cada espacio de colores, a pesar de que muchos estudios concuerdan en que no tiene sentido utilizar

el canal relacionado con la iluminación. A partir de aquí y a lo largo del resto del documento, se dirá que un espacio de colores es 2D si sólo se usan 2 canales, y 3D si se usan los 3 canales.

Según [39], un *random forest* es un conjunto de árboles de predicción, que no son más que sistemas independientes de predicción. Cada árbol toma como entrada un vector de características, en nuestro caso, un vector con los valores de los canales de color utilizados del píxel que queremos reconocer. Cada árbol devuelve una clasificación del vector. El algoritmo decidirá como se clasifica dicho vector en función de la cantidad de votos que ha recibido cada clasificación para el vector, escogiendo aquella más votada.

Todos los árboles son entrenados a partir del mismo set de parámetros de entrenamiento, una matriz de N vectores con M parámetros, en el caso de píxeles, M canales de color. Ahora bien, cada árbol es entrenado con un subconjunto de n vectores distinto generado a partir del set de entrenamiento mediante el procedimiento *bootstrap*. Esto consiste en seleccionar aleatoriamente un número de vectores n igual al número de vectores del set global N, pero permitiendo reemplazamiento, es decir, un vector puede aparecer varias veces o ninguna. Además, en cada nodo de cada árbol no todos los parámetros son usados para calcular la mejor ramificación, sino que se escoge un conjunto m aleatorio que debe ser mucho menor que M (en nuestro caso, puesto que M será 2 ó 3, m será probablemente 1). Cada árbol crece completamente y sin límite (como se puede hacer en la construcción de un clasificador con arboles normales).

En la figura 4.20 se muestra un ejemplo de un árbol de decisión. El vector x es un vector de parámetros p, que en este proyecto serán sólo 2 ó 3, según los canales utilizados. El número de árboles se puede escoger, y cada uno de ellos nos dará una decisión, un voto k para una de las posibles clases. En este proyecto sólo hay dos clases: piel y no-piel, y cada árbol sólo podrá votar si es una u otra. Según cual obtenga más votos, será reconocido como tal.

El prototipo *Random Forest* cuenta con dos funciones principales: *train* y *predict*. La función *train* recibe todos los datos de entrenamiento en forma de matriz, así como todos los parámetros especificados para configurar el sistema (número de modelos a desarrollar, número de árboles, profundidad de los árboles, iteraciones máximas, etc.). Esta función devolverá tantos modelos como se le hayan especificado (en este caso dos, piel y no-piel). La función *predict* es la encargada de clasificar cada muestra comparándola con los modelos creados anteriormente. Esta recibe sólo una muestra en forma de vector, y devuelve un valor de decisión que se utilizará para diferenciar predicciones.

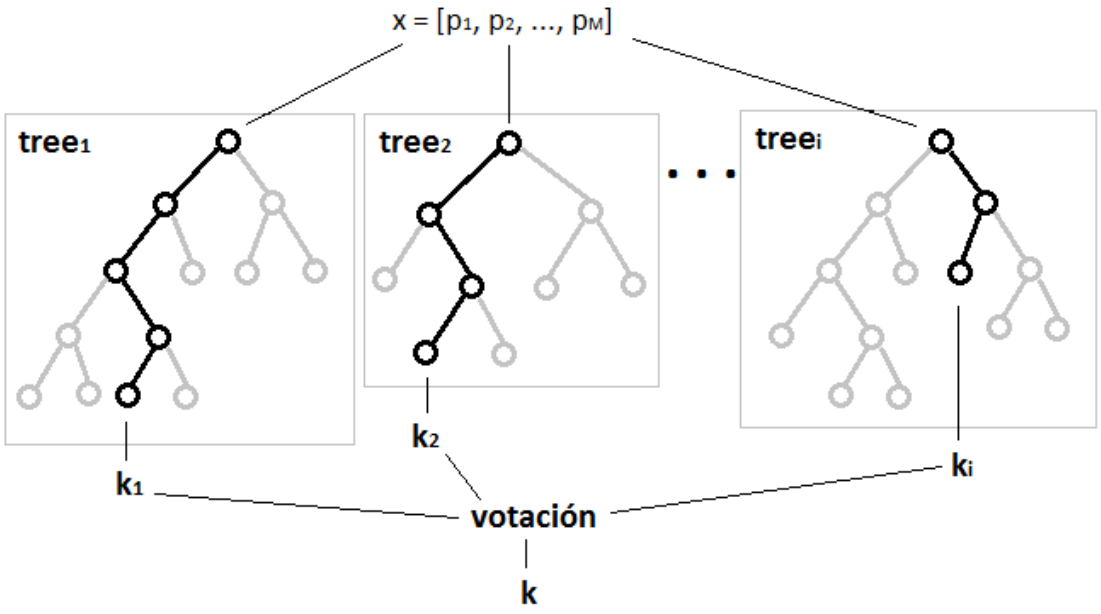


Figura 4.20: Ejemplo de *Random Forest*

Capítulo 5

Reconocimiento de Eventos

5.1. Introducción

En este capítulo se aborda la tarea del modelado y reconocimiento de eventos. Se realizará un estudio individual de cada uno de los eventos propuestos por la competición ICPR-HARL 2012 para averiguar en que consisten, que subeventos implican, que características se podrían extraer para hacer más sencilla su detección y que técnicas podrían ser usadas para este fin. Siguiendo estos pasos, algunos eventos serán descartados debido a su complejidad, dejando así un subconjunto de eventos sobre el cual se centrará todo el esfuerzo de este trabajo. Por último, se describirán detalles de la implementación realizada y sus problemas derivados.

Este capítulo sigue la siguiente estructura: esquema del sistema global (sección 5.2), descripción de los eventos propuestos (HARL) 5.3, fase 1: implementación genérica y problemas (sección 5.4) y fase 2: ajuste para la competición HARL (sección 5.5).

5.2. Esquema del sistema global

En el capítulo 3 se describió la estructura del prototipo diseñado por el VPU-Lab para la detección de eventos. La figura 5.1 resume las etapas del sistema. Las etapas *Foreground Detection* (detección de frente) y *Blob Tracking* (seguimiento de blob) son las encargadas de obtener los *blobs* del frente de la escena y realizar un seguimiento de los mismos a lo largo de la secuencia de vídeo. La siguiente etapa, *Feature Extraction* (extracción de características) depende completamente de las etapas anteriores y del bloque *Contextual Information* (información contextual). Esta etapa es la encargada de extraer de los *blobs* todas las características en las que se basará el sistema para detectar los eventos. El bloque *Contextual Information* introduce al sistema información sobre el escenario dada por el usuario. Por último, la etapa *Event Recognition* (reconocimiento de eventos) se encarga de reunir toda la información obtenida de las características extraídas y aportada por el usuario para detectar los eventos en cada *frame* del

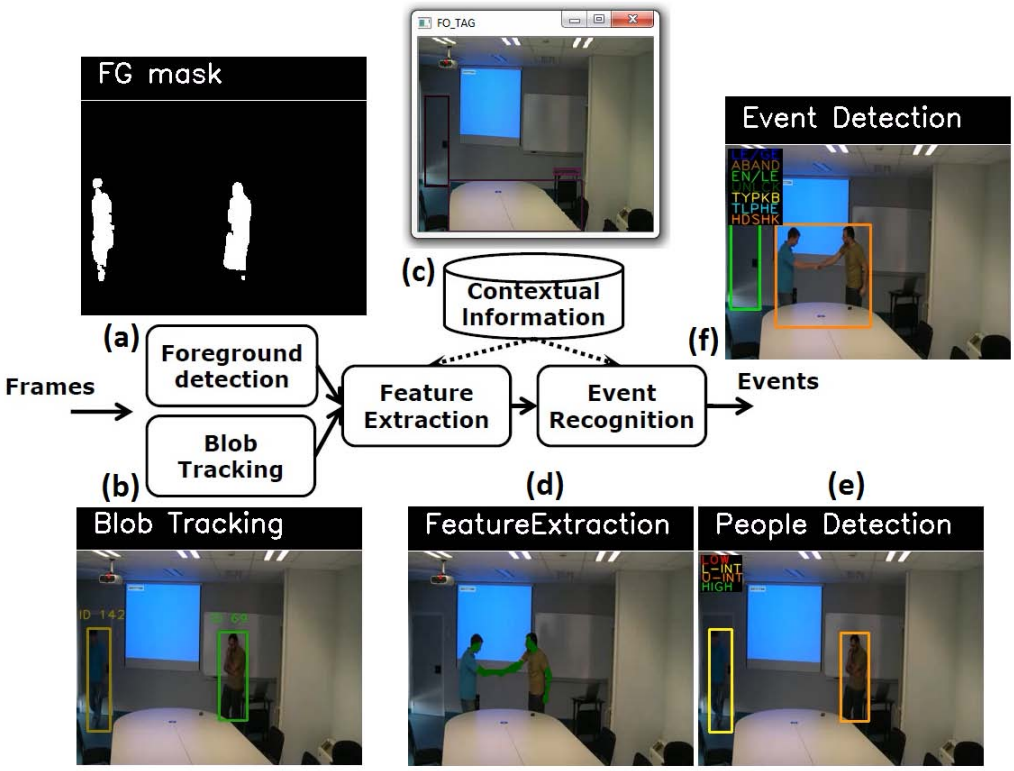


Figura 5.1: Diagrama de bloques del sistema

vídeo. El objetivo final es mostrar al usuario algo como lo mostrado en la captura (f) de la figura 5.1, donde se marcan con *bounding box* con distintos colores los distintos eventos que suceden en la secuencia de vídeo.

A partir de este prototipo base, se decidió usar modelos sencillos sin entrenamiento basados en *Finite State Machines* (FSMs) para modelar cada evento, en lugar del marco de inferencia bayesiano utilizado en el prototipo. Además, se propuso implementar los nuevos eventos propuestos en la competición ICPR - HARL 2012.

5.3. Descripción y análisis de los eventos propuestos (HARL)

La organización de la competición propone diez eventos que se engloban dentro del escenario de interés en este proyecto. Se trata de eventos que típicamente podrían encontrarse en un escenario de interior, como es una planta de despachos, oficinas o aulas. Además, el dataset entregado para la competición está desarrollado en un escenario claramente controlado, donde el número de personas y de eventos que ocurren es reducido, y dichos eventos son relativamente sencillos de detectar, o al menos, es su intención.

En la tabla 5.1 se recogen los diez eventos propuestos. Cada uno tiene un código de dos letras

Nº	Código	Evento	Tipo
1	DI	Discusión entre dos o más personas	HH
2	GI	Dar objeto a otra persona	HH, HO
3	BO	Coger/Dejar objeto (de/en algún sitio)	HO
4	EN	Entrar/Salir de una habitación (sin desbloquear)	-
5	ET	Intentar entrar en una habitación (sin éxito)	-
6	LO	Desbloquear puerta y entrar en una habitación	-
7	UB	Abandono de equipaje (dejarlo e irse)	HO
8	HS	Apretón de manos	HH
9	KB	Escribir en un teclado	HO
10	TE	Hablar por teléfono	HO

Tabla 5.1: Eventos

como identificador. Además, también se indica si el evento implica una interacción entre dos personas o entre persona y objeto mediante las siglas HH (Humano-Humano) y HO (Humano-Objecto). En la figura 5.2 se muestran dos ejemplos de cada evento en el dataset LIRIS.

Como se puede apreciar, los eventos en sí son muy sencillos, fáciles de reconocer visualmente por cualquier persona, pero hay que recordar que el objetivo de este proyecto es realizar una deteción automática, lo cual no es una tarea sencilla. Por ello, para entender mejor los eventos propuestos se realizó un estudio de todos los vídeos del dataset entregado en la competición. El principal objetivo de esta revisión de los vídeos fue descubrir que características son las más comunes y evidentes en un evento, cuales habría que obtener para reconocer cada evento y comprobar si con las técnicas disponibles se podrían extraer dichas características. Todo esto sirvió para descartar la implementación de algunos eventos y centrar todo el esfuerzo en aquellos que parecía factible abordar.

A partir de este estudio de los eventos se diseñaron los modelos que servirían como aproximación a la solución final desarrollada en este proyecto. Estos modelos sirvieron para implementar los eventos mediante máquinas de estados, en las que cada estado depende de una serie de condiciones o reglas que, en caso de cumplirse, supondrán que está sucediendo la acción a la que representa dicho estado. En caso de recorrerse toda la máquina de estados de un evento, el sistema podrá concluir que el evento ha sido detectado.

A continuación se presenta una breve descripción de cada evento, de las características más importantes a extraer para cada uno y de los problemas observados únicamente a partir de la visualización de los vídeos.

1) Discusión entre dos o más personas (DI)

Dos o más personas discutiendo, frente a frente o de cara a una pizarra. Los protagonistas no entran en contacto en ningún momento y se mantienen en posición moviendo la boca, la cabeza, los brazos o cambiando de pose.

Figura 5.2: Ejemplos en el dataset LIRIS de los eventos propuestos

Se requiere detección de personas, comprobar que se encuentren a una distancia pequeña o nula y que tengan un tamaño similar (para tener en cuenta la profundidad del escenario también). Hay que detectar que las personas implicadas no se desplacen (alejen) en un tiempo establecido T. Dos características útiles son la pose, para comprobar que se es-

tén mirando o estén mirándo en la misma dirección, y la detección de movimiento en la boca/cara.

La principal dificultad de este evento reside en cómo diferenciar una situación de discusión entre dos personas de una situación en la que las dos personas simplemente permanezcan cerca sin hacer nada.

2) Dar objeto a otra persona (GI)

El evento implica a dos personas y un objeto. La persona poseedora del objeto al inicio (persona A) da dicho objeto a otra (persona B). Las dos personas entrarán en contacto sólo para intercambiar el objeto.

Se requiere detección de personas y detección de piel de las manos para detectar el contacto entre ellas mediante otra característica, el solapamiento de regiones. También se requiere detección de objetos en posesión de una persona. Por último, habría que realizar un seguimiento del objeto, desde de la persona A hasta la persona B.

La principal dificultad de este evento reside en detectar el objeto, que suele ser pequeño y estar desde el inicio en posesión de la persona A. Tampoco es fácil seguir un objeto en posesión de una persona, y menos aún cuando este objeto cambia de persona, debido a las múltiples oclusiones sobre el mismo.

3) Coger/Dejar objeto (de/en algún sitio) (BO)

Este evento implica cuatro eventos que se podrían considerar distintos, pero que la competición HARL considera como uno sólo. En primer lugar, reúne los eventos de *Coger* y *Dejar Objeto*. Luego, estos dos eventos pueden suceder en dos lugares distintos: en un sitio visible (mesa) o en un contenedor. El primer caso implica que el objeto es cogido o dejado en una zona en la cual estaba o quedará visible. En el segundo caso, nunca veremos el objeto nada más que cuando la persona lo porte. Este evento siempre implica una persona y un objeto (visible o no). Para todos los casos, el evento se considera el mismo.

Se requiere detección de personas, detección de piel y detectar contacto de la piel con el contenedor o con la zona para depositar el objeto (previamente anotados) mediante solapamiento. Si el objeto es dejado y está visible, hay que detectar un cambio en el frente. Si lo ha cogido, además de detectar el área en el frente, hay que seguir el objeto en la persona. Si se da el caso del contenedor, si se trata de coger, hay que detectar el nuevo objeto y seguirlo en la persona, y si se trata de dejar, el objeto desaparecerá de la persona y de escena.

La dificultad de este evento varía según el caso. Si el objeto no es visible, es muy difícil detectarlo cuando está en posesión de una persona, y saber que una persona accede a un contenedor con el propósito de coger o dejar algo es sencillo, pero hay que suponer que esta

cogiendo o dejando algo. Si el objeto está visible, es bastante más sencillo, ya que incluso cuando el objeto sea cogido, esté pertenecería antes al fondo y por tanto dejará una figura en el frente, al igual que cuando se deja el objeto.

4) Entrar/Salir de una habitación (sin desbloquear) (EN)

Este evento implica una persona atravesando una puerta. La puerta puede estar abierta o cerrada, pero nunca bloqueada. "Entrar" es salir de escena y "salir" es entrar en escena, independientemente de que la escena suceda dentro o fuera de una sala.

Es necesario detectar a la persona y realizar un buen seguimiento de la misma. También es necesario tener la anotación de las puertas en escena. La persona habrá sido detectada antes de entrar dentro del marco si se trata de "entrar", mientras que la persona aparecerá por primera vez dentro del marco de la puerta si se trata de "salir". Tras esto, si la persona "entra", desaparecerá por la zona del marco de la puerta, y si la persona "sale", saldrá completamente del marco de la puerta.

El evento no presenta dificultades a priori, aunque las puertas laterales respecto de la cámara pueden presentar cierta dificultad.

5) Intentar entrar en una habitación (sin éxito) (ET)

Este evento implica una persona intentando abrir una puerta sin éxito.

Es necesario detectar a la persona delante de la puerta (previamente anotada). Hay que comprobar que la persona no se desplaza durante un tiempo T (estaticidad). Finalmente, hay que comprobar que la persona se aparta de la puerta y que la distancia entre persona y puerta es suficiente. Una característica útil es detectar o anotar el pomo de la puerta.

La dificultad del evento reside en que es imposible en muchos casos saber si la persona está intentando abrir la puerta, ya que normalmente ocluye el pomo. Por tanto, puesto que no se puede especular nunca con lo que pasa si no se tiene información suficiente, esté será en muchas ocasiones un evento imposible de tratar.

6) Desbloquear puerta y entrar en una habitación (LO)

Este evento consiste en una persona atravesando una puerta previamente bloqueada. En este caso, la persona siempre sale de escena.

Hay que detectar a la persona delante de la puerta (previamente anotada), comprobando que permanece estática un tiempo establecido T. Tras esto, hay que detectar un gran cambio en el frente dentro del marco de la puerta. Por último, dicho frente desaparecerá junto con la persona. Una característica útil es detectar o anotar el pomo de la puerta.

La única dificultad aparente es la reconocer que la persona está desbloqueando la puerta.

7) Abandono de equipaje (dejarlo e irse) (UB)

Este evento consiste en una persona abandonando un objeto del tipo mochila, bolsa, maleta, etc. El evento no tiene lugar hasta que la persona se aleje lo suficiente.

Hay que detectar a la persona y al objeto que deja en el suelo. La persona debe ser identificada como la poseedora del objeto. Tras esto, hay que detectar que la persona se aleja del objeto utilizando la distancia entre ellos o detectando la desaparición de la persona de escena.

Este evento no presenta dificultades a priori.

8) Apretón de manos (HS)

Este evento consiste en dos personas que se dan un apretón de manos frente a frente.

Hay que detectar a las dos personas implicadas y comprobar que sean de un tamaño similar (por temas de profundidad en la escena), además de comprobar que la distancia entre ellos es pequeña o nula. También hay que detectar la región piel de las manos, hacer un seguimiento de las mismas y detectar que una entra en contacto con la otra mediante solapamiento.

La principal dificultad de este evento es la oclusión del apretón. Además, hay que encontrar la forma de saber que esas regiones de piel pertenecen a los brazos, ya que no vale el contacto de cualquier región de piel. También hay que diferenciar el apretón de un simple cruce de regiones de piel.

9) Escribir en un teclado (KB)

Este evento consiste en una persona tecleando en un teclado de ordenador de sobremesa o en un portátil.

Hay que detectar a la persona implicada en el evento al igual que el teclado (previamente anotado, a menos que se trate de un portátil recién introducido en escena). La distancia entre el teclado y la persona debe ser pequeña o nula. Además, hay que detectar las regiones de piel de la persona y que una de ellas esté en contacto u ocluyendo al teclado durante un tiempo determinado.

La dificultad de este evento consiste en detectar las manos sobre el teclado, que en ocasiones, queda ocluído por la pantalla del portátil o por la propia persona.

10) Hablar por teléfono (TE)

Este evento consiste en una persona hablando por teléfono (fijo o móvil), que implica coger el teléfono, llevárselo a la cabeza y hablar.

Hay que detectar a la persona implicada y detectar la piel de las manos. Después, hay que detectar solapamiento entre la piel y el teléfono. El teléfono será anotado previamente en caso de ser fijo. Tras esto, hay que detectar un cambio en el frente sobre el teléfono. Esto último no ocurrirá si el teléfono es móvil y está en posesión de la persona. Por último, hay que seguir el teléfono en posesión de la persona y detectar solapamiento entre la piel de la cara y el teléfono.

Este evento es difícil de detectar, a priori, por varios motivos. En primer lugar, si el teléfono es móvil y la persona lo saca de su bolsillo, será casi imposible detectarlo. Además, detectar que una region de piel está ocluyendo parcialmente la piel de la cara también es difícil de detectar. Pero más difícil aún será detectar este evento si la persona coloca el móvil detrás de su cabeza respecto a la visión de la cámara.

5.3.1. Consideraciones generales de los eventos

A parte de lo descrito en el apartado anterior para cada evento, hay que realizar una serie de consideraciones generales aplicables a todos ellos.

Todos los eventos descritos pueden ser realizados de muchas maneras distintas dentro del mismo dataset, y cada realización de un evento supone un reto a la hora de implementarlo, ya que puede requerir características distintas a las comunes o carecer de ellas.

Otro problema típico son las oclusiones. Un ejemplo de ello viene provocado por la posición de las personas a la hora de realizar el evento respecto del punto de vista de la cámara. Muchas veces la propia persona que realiza el evento se sitúa de espaldas a la cámara, ocluyendo al propio evento. También las oclusiones pueden ser provocadas por otros objetos al moverse delante de los implicados en el evento, justo en el momento de realización de este.

Es importante tener en cuenta el *framerate* de los vídeos de cada dataset, ya que para muchos eventos se utiliza el número de *frames* como medida de tiempo. Si se establece un número determinado de *frames* para que suceda un evento, hay que tener en cuenta que no es generalizable a todos los datasets. Un evento puede durar los mismos segundos en dos datasets distintos, pero ese tiempo traducido a *frames* será distinto en función del *framerate* del dataset.

Al igual que ocurre con el *framerate*, la resolución de los vídeos influye a la hora de utilizar las distancias en una imagen. Hay que tener en cuenta que las distancias entre objetos en una imágen se miden en píxeles, por tanto, no se pueden usar valores que no sean relativos a la resolución de la propia imagen. Lo que en un dataset puede ser una distancia grande, en otro con mayor resolución puede ser una distancia despreciable.

5.4. Fase 1: Modelado genérico y problemas

5.4.1. Listado de los objetos contextuales

Una vez estudiados todos los eventos, hay que incluir en el sistema un listado de los nuevos objetos contextuales, que posteriormente habrá que anotar en cada vídeo que aparezcan. En la tabla 5.2 se muestran los objetos contextuales y los eventos en los que son usados. Aunque el sistema base es capaz de leer muchos más objetos contextuales de los aquí recogidos, no tiene sentido incluir en las anotaciones aquellos que no son usados en ningún evento.

Tipo	Objeto	Evento
Fijo	Puerta	EN, ET, LO
	Mesa	BO
	Caja	BO
Portátil	Teléfono fijo	TE
	Teléfono móvil	TE
	Teclado fijo	KB
	Ordenador portátil	KB

Tabla 5.2: Objetos contextuales
NOTA: Puede ser confuso que los objetos Telefóno fijo y Teclado fijo sean objetos portátiles, pero son objetos que realmente pueden ser desplazados en algunos vídeos. En cualquier caso, a efectos prácticos, en este sistema no importa nada que un objeto sea fijo o portátil, ninguna condición de evento depende de ello.

5.4.2. Eventos descartados

En el apartado 5.3 se realizó una descripción y análisis de todos los eventos de la competicion HARL 2012. También se han descrito las principales dificultades que se pueden observar directamente de la visualización de los vídeos. En este proyecto se propuso inicialmente descartar aquellos eventos con una dificultad demasiado elevada y centrar todo el esfuerzo en aquellos que sí parecen ser posibles de detectar. Los eventos descartados inicialmente fueron:

- *Discusión*: La distancia entre la cámara y las personas es normalmente grande, resultando casi imposible distinguir movimiento en la cara mediante extracción de frente. Además, sin audio, existen pocos estudios donde intenten si quiera tratar este evento. Sin estas características, resulta muy difícil distinguir este evento de dos personas paradas una en frente de la otra.

- *Dar Objeto*: Es muy difícil detectar un objeto en manos de una persona, más aún si dicho objeto tiene un tamaño muy pequeño. Además, dicho objeto debe seguirse de una persona a otra. También es muy difícil diferenciarlo de un apretón de manos.

- *Intentar Entrar*: No se pueden hacer suposiciones de que una persona está intentando abrir una puerta si no se ve ni siquera el pomo de ésta.

El resto de eventos en principio no quedaron descartados, aunque más adelante surgirían problemas que llevarían en algunos casos a implementar solamente parte de ellos.

5.4.3. Modelado

Los eventos escogidos fueron inicialmente modelados a partir de lo visto en el apartado 5.3 mediante máquinas de estados sencillas, para después poder realizar su implementación directamente a partir de esos modelos mediante FSMs (Anexo C). Ahora bien, algunas de las características propuestas no han sido posibles de extraer y se han buscado otras soluciones.

Para este apartado conviene recordar las principales características que extrae el prototipo del VPU-Lab, presentadas en el apartado 3.2. Además, se han añadido algunas funciones más, como son: *CalculateBlobHistogram*, para calcular los histogramas de cada canal de color RGB de un blob, *CalculateBoundingBoxOverlap*, para calcular el porcentaje de solape de una *bbox* sobre otra, y *EuclideanBBoxDistance*, para calcular la distancia euclídea entre los centros de dos blobs. También, se estableció un umbral de *PeopleLikelihood* más bajo que el predeterminado en el prototipo, tras realizar algunas pruebas. Siempre que se hable de una persona en líneas posteriores, se supondrá que el blob tiene un umbral por encima de ese valor, y viceversa para un objeto. Por último, se han usado varios contadores de tiempo en frames, especialmente el *StaticCount*, que indica la cantidad de frames que un blob permanece estático.

A continuación se describirá como ha sido modelado cada evento y como se han usado las características extraídas. Recordar que las condiciones que se describen para cada estado son las que deben cumplirse para que la FSM pase a dicho estado.

- **Coger/Dejar objeto:** El sistema prototipo ya contaba con una forma de detectar este evento para objetos visibles dejados sobre una mesa, por lo tanto, se ha modificado ligeramente el código existente para adecuarlo a más casos, prescindiendo de algunas de las sugerencias generadas en el estudio del evento. Además, el evento estaba diseñado igual para "dejar" y "coger", lo que repercute en menos estados en la FSM. La figura 5.3 muestra el modelo de la máquina de estados utilizada.

  - S1: Aparece un nuevo blob correspondiente a un objeto, que será el candidato. Se busca su poseedor, que debe ser persona, y se establece que el objeto debe tener unas limitaciones de tamaño respecto al poseedor. Si todo esto se cumple, se comprueba que el candidato haya permanecido varios frames estático.
  - S2: Se buscan todas las mesas anotadas en escena. Para cada mesa, se calcula el solapamiento del *bounding box* del candidato sobre ella. Si el solapamiento es suficiente, se

calcularán los histogramas del candidato. Tras esto, se calculan los histogramas de la región de fondo solapada por el candidato y se calcula la distancia de Bhattacharyya entre los histogramas. Esto se hace para comprobar si el candidato tiene unos histogramas muy similares al fondo, ya que si se diera el caso, indicaría que posiblemente no existe tal evento, y el blob ha sido provocado por algún cambio de iluminación en la zona. Si no fuera así, la FSM saltaría a este estado y detectaría el evento.

- S3, S4: Estos estados fueron añadidos para realizar el caso en el que el evento se produjera en un contendor, pero no llegó a ser desarrollado del todo en esta etapa del proyecto. Como se comentó en el apartado 5.3, este caso es bastante complejo de tratar y, después de realizar varias pruebas, no se lograron resultados. Más adelante, en una etapa posterior del proyecto, se incluiría esta opción.

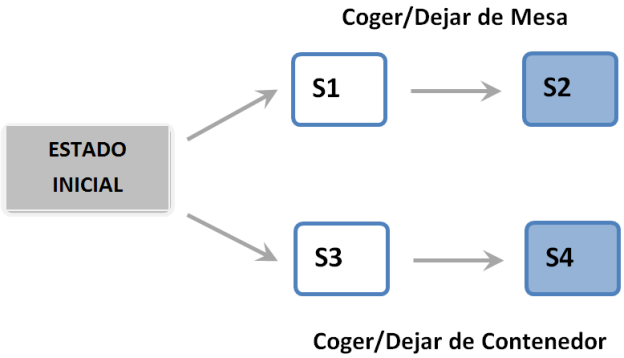


Figura 5.3: Diagrama BO

Una vez detectado el evento, el candidato se integra en el fondo. Si el objeto es dejado, aparecerá en el fondo sobre la mesa, si es cogido, desaparecerá. De esta forma, el evento puede producirse de nuevo con el mismo objeto, ya que cualquier acción sobre él producirá de nuevo un blob en escena. En la figura 5.4 se puede ver un ejemplo de esto.

Problemas

- Una mala eliminación de sombras y reflejos puede provocar la aparición de muchos blobs en el frente que no deberían aparecer. Si estos aparecen cerca de un blob detectado como persona, pueden ser detectados como objetos dejados erróneamente y, además, integrarse en el fondo, empeorando éste.
- Debido al límite del tamaño de blob impuesto en el sistema, los objetos pequeños, como puede ser un teléfono móvil, generan un blob en el frente muy pequeño que no siempre es detectado como objeto.

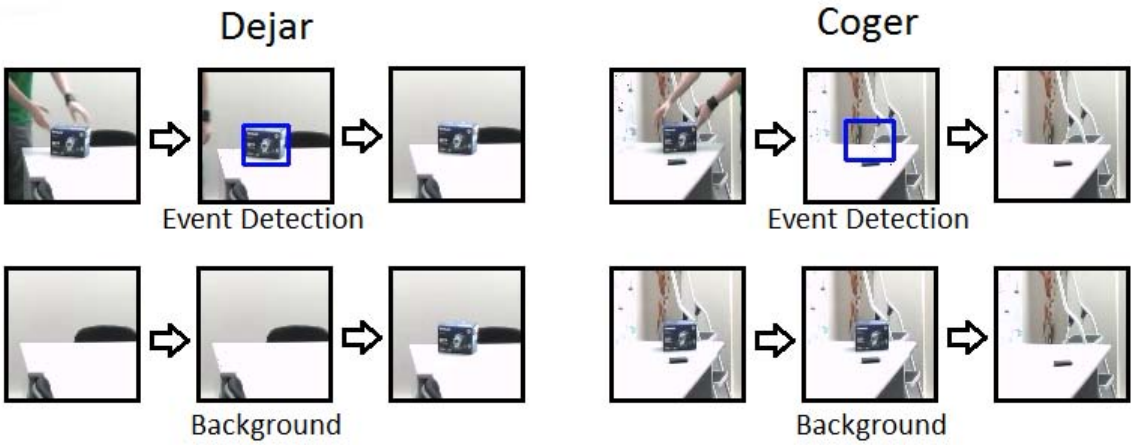


Figura 5.4: Ejemplo de BO (Dejar y Coger)

- La integración en el fondo de un objeto dejado o la sustracción de uno cogido no es perfecta, y puede ocasionar errores a la hora de detectarlo.
- Cada portátil dejado en escena generan un *bounding box* clasificado como portátil que no es mostrado en pantalla. Cuando se recogen, a veces no se eliminan dichos *bounding box*, provocando detecciones del evento *Escribir en teclado* erróneas en sitios donde hubo un portátil en frames anteriores.

- **Entrar/Salir:** Para este evento ha sido necesario realizar un modelo más complejo que para otros eventos, además de relaciones también más complejas entre estados. En la figura 5.5 se muestra dicho modelo. Los tres primeros estados corresponden al la acción de "entrar", los tres siguientes a la acción de "salir", y el estado 7 es un estado de seguridad. Los estados 3 y 6 son los que devolverán el *score* igual a 1. Hay que recordar también que en este modelo el candidato es la puerta, por tanto, estará presente siempre en escena.

  - S1: Se buscan todos aquellos blobs clasificados como persona y se comprueba su solapamiento con la puerta. Si no hay solapamiento, indicará que la persona se encuentra fuera del área limitada por el marco de la puerta. En ese caso, se calcula su distancia euclídea respecto a la puerta y si se encuentra cerca, se guardará su ID.
  - S2: Se buscan de nuevo todos los blobs persona y entre ellos se busca aquel con la misma ID que se guardó en el estado anterior. En caso de encontrarla, se calcula el solapamiento de ella con la puerta. Si hay un solapamiento alto, se considera que la persona está bajo el marco de la puerta.
  - S3: Se busca de nuevo la persona con el ID guardado. Si se encuentra otra persona que también está dentro o parcialmente dentro del marco de la puerta y la persona

buscada lleva entre 10 y 20 frames sin aparecer, se supone que la nueva persona es la buscada, pero que ha cambiado de ID. En este caso, se actualiza el ID guardado. Si en cambio pasan 20 frames y no hay ninguna persona bajo el marco de la puerta, se considera que la persona buscada ha entrado. En este caso, la FSM saltará a este estado y se detectará el evento.

- S4: Recorre todos los blob persona, y si no hay ningún ID guardado o ninguna persona tiene un ID igual al guardado, calcula el solapamiento con la puerta de cada persona. Si alguno es mayor que el umbral establecido, se considerará que el blob está bajo el marco de la puerta. Después se busca ese mismo blob en frames anteriores, y si no existía, se considerará que es la primera vez que está en escena.
- S5: Se busca de nuevo una persona, en este caso, que esté ocluyendo parcialmente la puerta. Si es así, se guardará su ID.
- S6: Se busca la persona que tenga el mismo ID que el guardado. Si existe, se mide la distancia de ésta con la puerta. Si se considera que está fuera del marco de la puerta, se saltará a este estado y se detectará el evento.
- S7: La FSM saltará a este estado desde los estados 3 y 6 una vez que hayan pasado 100 frames desde la detección del evento. Desde este estado, se saltará inmediatamente al estado inicial, de tal forma que el evento pueda producirse de nuevo.

Las relaciones que se muestran en la figura 5.5 han sido establecidas de tal manera para paliar varios problemas. En primer lugar, el estado 1 se autocomprueba cada frame para guardar siempre el último ID de la persona más cercana a la puerta. En el estado 2 se comprueba el 1, ya que puede darse el caso de que la persona pase por delante de la puerta sin llegar a cruzarla. El estado 1 comprueba siempre el estado 4, pues puede aparecer alguien bajo el marco de la puerta y pasar de largo, y si no estuvieran conectados, impediría en este estado que se produjera el evento. Lo mismo ocurre con el estado 1, el cual comprueba el estado 4 por si ocurriera que una persona sólo pasa cerca de la puerta.

Problemas

- En ocasiones falla debido a la presencia de varias personas en las cercanías de la puerta, sobre todo para las salidas.
- No funciona para puertas que aparezcan de lado en escena, en una pared lateral o similares. Esto se debe a que, en ocasiones, es mucho suponer que se trate de una puerta y no un pasillo. Además, el área bajo el marco de la puerta es muy reducido o nulo, haciendo imposible aplicar el método usado. Aun así, en una etapa más avanzada del proyecto, se incluiría una solución para esto.

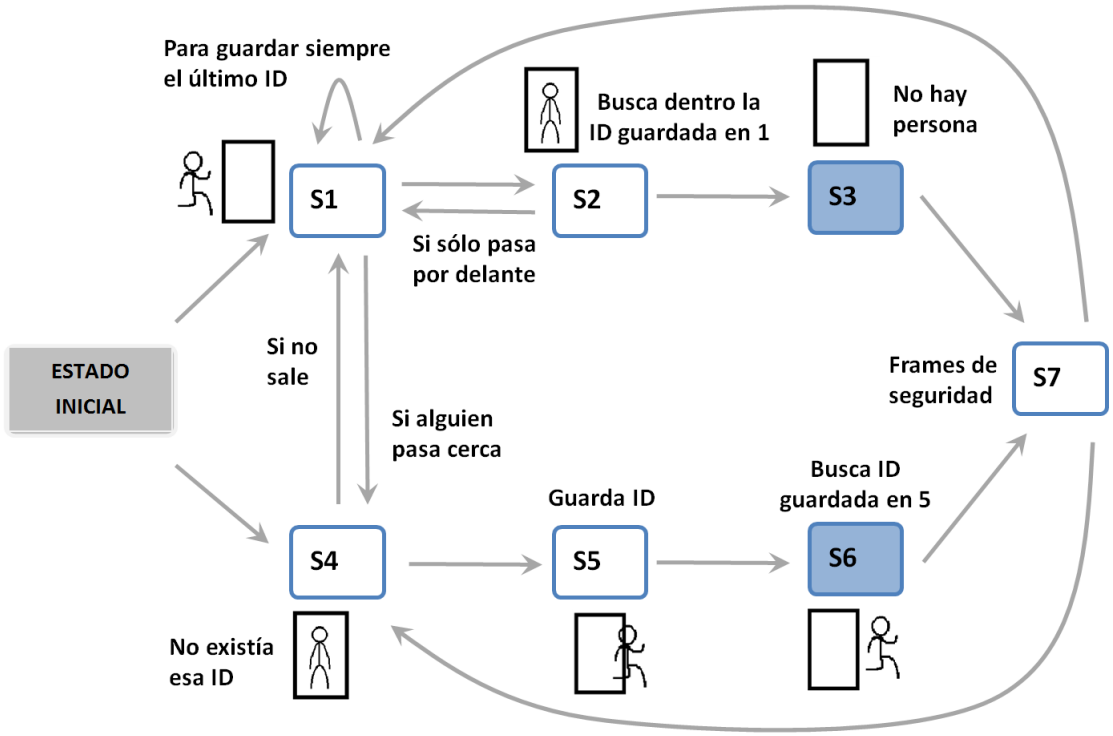


Figura 5.5: Modelo del evento EN

- A veces falla simplemente porque la persona pasar por delante, aunque este caso se ha solucionado casi por completo con las relaciones descritas anteriormente.
- Si la puerta ocupa un lateral del vídeo y una persona aparece por ahí, lo detecta (en principio sería correcto dada la información que posee el sistema).

- **Desbloquear puerta:** En esta etapa del proyecto se trató de buscar una solución para este evento, sin éxito. La idea inicial era diseñarlo como el caso de "entrar" del evento *Entrar/Salir*, añadiendo en ambos eventos algo que diferenciara cuando la puerta se desbloquea y cuando no. El problema principal es que en muchos casos el pomo de la puerta no está visible, y el tiempo que la persona actúa sobre la puerta es bastante variable. Debido a esto, inicialmente se acabó descartando el evento, aunque en una etapa posterior del proyecto se diseñaría una solución que sería aceptable para muchos de los casos existentes en los vídeos.

- **Abandono de equipaje:** Para este evento se han añadido algunas características nuevas en sustitución de algunas de las descritas en el estudio del evento. En la figura 5.5 se muestra el modelo utilizado.

  - S1: Se comprueba que el candidato es un objeto. Después se busca su poseedor, que

debe ser una persona. El candidato debe cumplir unas condiciones de tamaño respecto a su poseedor. Si todo esto se cumple, se guarda el ID del poseedor y se calculan sus histogramas. Por último, si el candidato permanece estático unos pocos frames, se comprueba que se trata de dejar un objeto y no de cogerlo.

- S2: Se busca entre todos los blobs aquel que tenga el mismo ID que el guardado, que será el poseedor. Si se encuentra, se calcula la distancia entre el poseedor y el candidato. Si ésta es suficientemente grande, se saltará a este estado. Si por el contrario no se encontró ningún blob con el ID guardado, se recorren de nuevo todos los blobs comparando sus histogramas con los histogramas guardados del poseedor mediante la distancia de Bhattacharyya. Si la distancia de Bhattacharyya entre ellos no es muy alta, se considera que ese blob es el poseedor, y se comprueba la distancia euclídea entre él y el candidato. Al igual que antes, si la distancia es suficientemente grande, se salta a este estado. Por último, si no se encuentra al poseedor ni por su ID ni por sus histogramas, también se salta a este estado.
- S3: Si el candidato permanece estático 30 frames, se salta a este estado y se detecta el evento.

Este evento tiene una relación más entre estados. El estado 2 se comprueba a sí mismo, ya que puede darse el caso de que el poseedor desaparezca unos pocos frames y reaparezca después. Si esto ocurre y aún no se han cumplido los 30 frames sin poseedor, se reseteará el contador y no se detectará el evento.

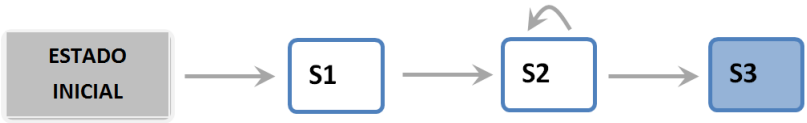


Figura 5.6: Diagrama UB

Problemas

- Una mala segmentación de frente puede producir que un blob se divida en dos repentinamente. Si uno de ellos se mantiene estático, puede interpretar que el otro es el poseedor y detectar el evento.
- Si el blob del objeto no es capaz de permanecer estático, aunque realmente lo esté, nunca detectará el evento.
- Un mal seguimiento de objetos puede producir que el sistema no sea capaz de mantener el ID, y un cambio de ID puede ser interpretado como desaparición del poseedor, provocando la detección errónea del evento. En principio, esto está solucionado con

la autocomprobación del estado 2, pero si supera los 30 frames sin reaparecer, lo detectará erróneamente.

- **Apretón de manos:** Este evento fue planteado con un modelo de 2 estados como el de la figura 5.7. Debido a las dificultades descritas en el apartado 5.3 y a que no se dispone de un algoritmo de deteción de partes del cuerpo, tuvieron que buscarse otras alternativas. El candidato de este evento será el blob formado por las dos personas dándose la mano.

  - S1: Se extraen las regiones de piel del candidato y se busca una que se sitúe aproximadamente en el centro de la *bounding box*. Se busca que esa región de piel tenga un blob más ancho que alto (brazos extendidos y unidos).
  - S2: Para cada región de piel, busca que haya estado estática 10 frames (tiempo razonable para un apretón). A partir del frame anterior a la aparición del candidato (comienzo del apretón), el sistema se fija en 25 frames anteriores y se fija en las zonas que ocupan en el frame actual las regiones de piel estáticas. Busca los blobs que ocluirían dichas regiones. Si una región de piel sería ocluída en el pasado por dos blobs distintos clasificados como persona, será la región de unión de los brazos. Por último, si esos dos blobs persona en el pasado estaban separados y tenían tamaños similares, se tratarán de las dos personas que realizan el apretón. Por tanto, si todo esto se cumple, saltará a este estado y se detectará el evento.

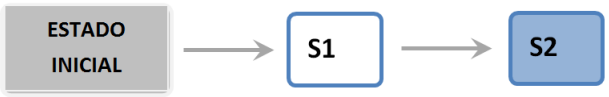


Figura 5.7: Diagrama HS

Problemas

- Una mala segmentación del frente en ocasiones provoca que los brazos no aparezcan correctamente en el frente. Si esto sucede, el candidato formado por ambas personas no aparecerá y el evento no será detectado.
- Una mala extracción de piel de los brazos puede provocar que el evento no se detecte.
- Si una persona es mucho más grande que la otra por interferencia de un tercer blob, el centro del *bounding box* final no se corresponderá al apretón.
- Si el apretón no se ve porque una persona esté en frente de la otra, evidentemente, no se detectará.

- **Escribir en teclado:** El evento tiene dos variantes, teclear en portátil o en teclado, aunque ambas se tratan prácticamente igual. El evento ha sido implementado para ambos casos mediante la misma máquina de estados, mostrada en la figura 5.8.

    - S1: En primer lugar, se extraen las regiones de piel del candidato. Después, se buscan todos los portátiles (*laptop*) en escena, se extraen las regiones de piel del candidato y se comprueba el solapamiento de todas ellas con cada portátil. De la misma manera se realiza con los teclados (*keyboard*). Las diferencias entre ambos son el umbral de solapamiento establecido para considerar positivo el evento y que la *bounding box* de los portátiles se expande un poco antes de comprobar el solapamiento.
    - S2: En este estado se vuelve a realizar la comprobación de solapamiento con el portátil o teclado del estado anterior, pero se añade la condición de que debe estar ocluyéndose durante 250 frames. Si en algun momento se deja de ocluir, se resetea a 0 el contador. Si se cumplen los 250 frames, se detectará el evento.

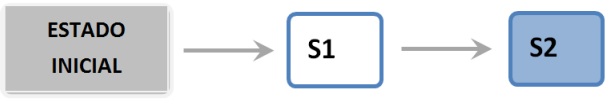


Figura 5.8: Diagrama KB

Problemas

- El único problema que tiene este evento es que salta con demasiada facilidad. Si una persona está cerca del objeto y alguna región de piel está en contacto con este durante un periodo largo de tiempo, lo detecta como evento, en especial para portátiles.

- **Hablar por teléfono:** Este evento tiene dos variantes, con teléfono fijo y con móvil. En la figura 5.9 se muestran las dos ramas de la máquina de estados referentes a dichas variantes. Ahora bien, en el apartado 5.3 se dieron motivos por los que este evento para móviles se consideraba muy difícil de detectar. Tras estudiar los vídeos del dataset de la competición, se decidió implementar el evento sólo para teléfonos fijos. Aún así, debido a la dificultad de detectar el auricular en la cara del candidato, a veces imposible de detectar, este evento fue implementado de forma que sólo se detecte cuando se accede al teléfono.

    - S1: Busca todos los blobs clasificados como persona en escena y busca que su solapamiento con el teléfono candidato sea mayor que un umbral establecido. Si es así, guarda el ID de ese blob.
    - S2: Busca al blob con el ID guardado y calcula su solapamiento de nuevo. Si no está solapando ya, se considera que ha descolgado el auricular. Si no encuentra el blob con

el ID anterior, aun así considera que se ha accedido al teléfono y que el blob puede haber cambiado de ID. Tras esto, busca otro blob, esta vez con distinto ID y que solape con el teléfono. Se busca en frames anteriores que ese blob no existiera. Si no existía, es el producido al retirar el auricular, y se guardará su ID.

- S3: Por último, se busca mediante el ID el blob anterior. Si no se encuentra, se salta a este último estado y se detecta el evento.
- S4: Este estado fue añadido para realizar el caso del teléfono móvil, pero no llegó a ser desarrollado.

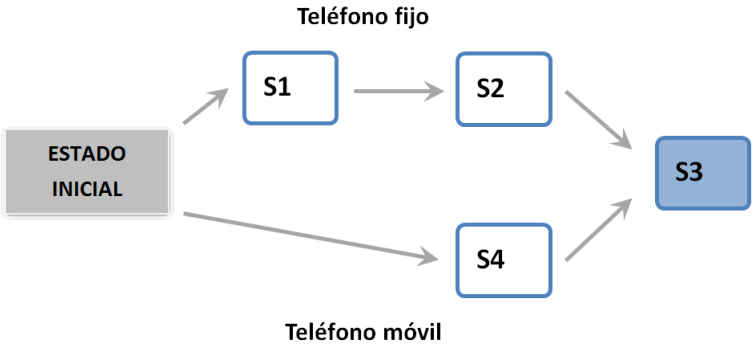


Figura 5.9: Diagrama TE

Problemas

- Si aparece un blob sobre el teléfono, puede dificultar la detección del evento.
- Este evento generalmente va seguido de varias detecciones erróneas del evento *Coger/Dejar objeto*, debidas al cable del teléfono, a retirar el auricular, etc. Si se detecta dicho evento sobre el teléfono antes que el propio evento *Hablar por teléfono*, este último no se detectará, aunque es algo que no suele ocurrir. Normalmente se detecta este evento y luego el evento *Coger/Dejar objeto*, lo cual es correcto, ya que se ha retirado un objeto.
- Habría que añadir algo que fuera capaz de detectar el micrófono sobre la cara de la persona, o algo similar. Como está diseñado ahora se asemeja más al evento *Coger/Dejar objeto* que al evento *Hablar por teléfono*.

Una vez diseñado, se realizaron pruebas sobre los datasets ED, SSG (apéndice B) y LIRIS-train. Los resultados se incluyen en el capítulo 6.

5.5. Fase 2: Ajuste para la competición HARL

En la última etapa de este proyecto, todo el esfuerzo se centró en obtener los mejores resultados posibles sobre el dataset LIRIS-test. Dichos resultados serían los que se enviarían a la organización de la competición en forma de ficheros XML. Estos ficheros llevarían la información sobre los eventos detectados en cada vídeo del dataset. Por tanto, se realizaron pruebas con el sistema descrito en el apartado 5.4 sobre los dataset LIRIS-train y LIRIS-test, y se comprobó que el sistema presentaba una baja eficacia. De esta manera, se decidió optar por rediseñar algunos eventos y buscar soluciones para casos que se habían descartado, teniendo por objetivo mejorar los resultados únicamente para este dataset, a costa de disminuir la generalidad del sistema.

5.5.1. Cambios realizados y adición de nuevos módulos

A continuación se describen los cambios realizados para mejorar los resultados del sistema:

- Nueva función *CalculatePixelOverlap*: Función que devuelve el porcentaje de solape entre dos blobs a nivel de píxel.
- Módulo de detección de personas independiente del fondo: Debido a la mala segmentación del frente a partir del fondo, muchos blobs no eran obtenidos correctamente. Esto es especialmente crítico para blobs persona implicados en algún tipo de evento. Este nuevo módulo basado en el detector de objetos HOG (*Histogram of Oriented Gradients*) es, en teoría, capaz de localizar personas sin necesidad de un fondo. Ahora bien, tras realizar varias pruebas con el dataset, se comprobó que no resultaría eficaz utilizarlo en sustitución del actual extractor de blobs. Por tanto, este módulo se utiliza como un elemento más de decisión para aquellos casos en los que un blob persona no es extraído correctamente.
- Módulo de detección de caras de perfil: Módulo basado en el algoritmo *Haar Like Features* de OpenCV. Utilizado para detectar caras de perfil en blobs persona. Esto módulo resulta muy útil para poder diseñar el evento *Discusión*.
- Nuevos objetos contextuales: Se añadieron nuevos objetos contextuales: suelo, caja y puertas laterales como objetos fijos, y pomo como objeto portátil. El primero se utiliza para el evento *Abandono de equipaje*, el segundo para el evento *Coger/Dejar objeto*, el tercero para el caso *Entrar/Salir* de puertas laterales, y el cuarto para los eventos *Desbloquear puerta* e *Intentar entrar*.

5.5.2. Nuevos eventos

A partir de las propuestas de mejora anteriores se decidió buscar soluciones para eventos que anteriormente se habían descartado, como el evento *Discusión*, el evento *Intentar entrar* y el caso

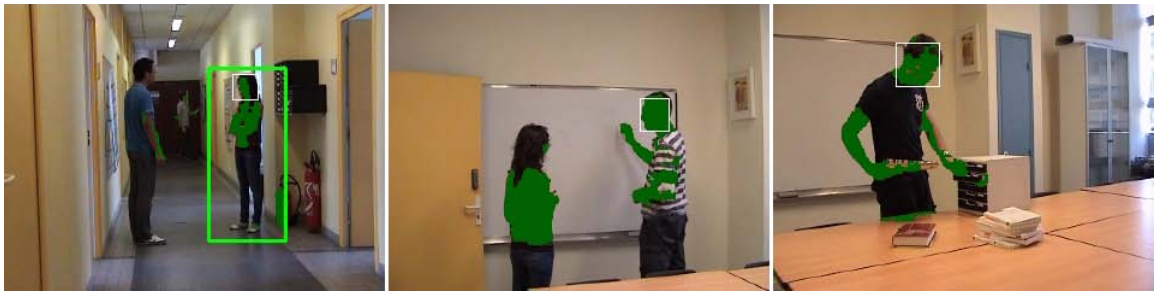


Figura 5.10: Ejemplos de detección de caras de perfil
Las caras de perfil aparecen marcadas con un cuadro blanco

Coger/Dejar objeto en contenedor. Además, se decidió mejorar otros eventos ya implementados, como *Abandono de equipaje* y *Entrar/Salir*, y desarrollar correctamente el evento *Desbloquear puerta*. A continuación se describen los cambios y adiciones más importantes en los eventos:

- **Discusión:** Tras la adición del nuevo módulo de detección de caras de perfil, se decidió implementar este evento, a pesar de no solventar la dificultad descrita en el apartado 5.3. La figura 5.11 representa el modelo del evento.

  - S1: Se buscan dos personas que tengan una altura similar, y que se encuentren a una distancia razonable. Estas condiciones han de cumplirse durante varios frames.
  - S2: Se busca que las personas implicadas sigan cumpliendo las condiciones anteriores. Mientras se cumplan, se buscan caras de perfil para cada blob. Se busca que en ambas personas se reconozca al menos 1 vez su cara de perfil y más veces en una de ellas.

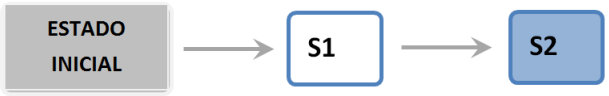


Figura 5.11: Diagrama GI

Problemas

  - El módulo no da resultados demasiado buenos y casi siempre deja a una de las dos personas sin detección de cara.

- **Coger/Dejar objeto en contenedor**: Esta configuración del evento se diseñó para el caso en que se cogiera o dejara un objeto dentro de un contenedor, anotado con el objeto contextual "caja". Utiliza los estados S3 y S4 que no se usaron, estados a los que sólo puede acceder un blob candidato que sea un objeto contextual "caja".

- S3: Se busca una persona y se extraen sus regiones de piel. Se calculará que porcentaje de solape a nivel de píxel existe entre cada región de piel y la caja. Se busca que el solape sea suficiente, sin llegar a cubrir toda la caja, y se cumpla durante algunos frames consecutivos. En cambio, si el solape de la persona a nivel de píxel es demasiado alto, se considerará que la persona está pasando por delante de la caja pero no está accediendo a ella.
- S4: Se busca de nuevo a la persona del primer estado, se extraen sus regiones de piel y se comprueba que no haya solape alguno durante unos pocos frames. De nuevo, se busca que no se dé en ningún caso un solape excesivo.

Problemas

- En general funciona bien, pero pueden surgir problemas cuando una persona cruza por delante de la caja y después accede a ella.

- **Intentar entrar:** El evento fue implementado tras la adición del nuevo objeto contextual "pomo" y de la nueva función que calcula el movimiento a nivel de píxel. La figura 5.12 representa el modelo del evento.

  - S1: Primero se busca una persona, un pomo en la puerta y se extraen las regiones de piel de la persona. Para cada región de piel, se busca que esté solapada a nivel de píxel con el pomo, pero que el blob de la persona no solape completamente el pomo. Se comprueba que se cumplan estas condiciones durante un número razonable de frames.
  - S2: Se comprueba que la puerta no presente demasiado movimiento durante unos pocos frames. Si esto es así, se supondrá que la puerta se está abriendo y no se producirá este evento. Si no ocurre esto, y ya no existe solape con el pomo, se supondrá que la persona ha desistido en su intento y se detectará el evento.

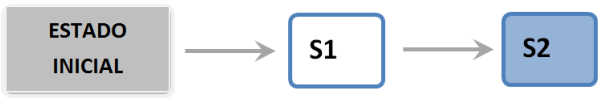


Figura 5.12: Diagrama ET

Problemas

- Hay casos en los que la persona no intenta abrir la puerta durante mucho tiempo. Estos casos pueden no ser detectados.

- **Desbloquear puerta:** Al igual que para el evento anterior, la nueva función y el nuevo objeto contextual hacen posible diseñar este evento mas fácilmente. La figura 5.13 representa el modelo del evento.

  - S1: Primero se busca una persona y un pomo en la puerta, y se extraen las regiones de piel de la persona. Se comprueba para cada región de piel que exista solape con el pomo, pero que el blob de la persona no solape completamente el pomo. Se comprueba que se cumplan estas condiciones durante un tiempo. Si es así, se considera que está tratando de abrir la puerta.
  - S2: Se calculará la cantidad de movimiento sobre la puerta mediante el porcentaje de solape a nivel de píxel sobre la puerta. Si este es muy alto durante unos pocos frames, se considerará que la puerta se está abriendo.
  - S3: Se busca la ID de la persona de los anteriores estados. Si no se encuentra, se busca otro blob persona que se encuentre en la misma posición pocos frames después. Si aún después de esta búsqueda no hay ninguna persona pocos frames después, se considera que la persona ha entrado y, por tanto, se detectará el evento.

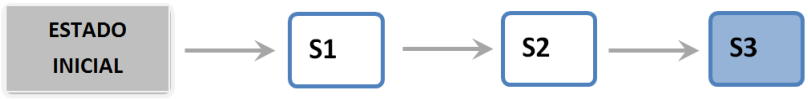


Figura 5.13: Diagrama LO

Problemas

- Puesto que el evento depende de que la puerta se abra casi en su totalidad, si esto no ocurre, el evento puede ser confundido con el evento *Intentar entrar*.

- **Cambios en Entrar/Salir y Abandono de equipaje**: El evento *Entrar/Salir* fue retocado para funcionar correctamente en presencia de los nuevos eventos *Intentar entrar* y *Desbloquear puerta*. Por otra parte, al evento *Abandono de equipaje* se le incluyó la búsqueda de solape del objeto abandonado con el nuevo objeto contextual "suelo".

Una vez realizados todos los cambios en esta segunda fase del sistema, se realizaron pruebas y se enviaron los resultados y el sistema completo a la competición HARL. Los resultados se incluyen en el capítulo 6.

Capítulo 6

Resultados

6.1. Introducción

En este capítulo se presentarán todos los resultados obtenidos durante el proyecto, tanto de detección de piel (Capítulo 4) como de reconocimiento de eventos (Capítulo 5). Para los relacionados con el reconocimiento de eventos, se realizará un repaso de los datasets utilizados y de las dificultades generales que presentan dichos datasets, para posteriormente poder entender mejor los resultados obtenidos.

Este capítulo sigue la siguiente estructura: datos, algoritmos y métrica usada, resultados e interpretación de los mismos en detección de piel (sección 6.2), datasets utilizados, métrica, resultados e interpretación de los mismos en reconocimiento de eventos (sección 6.3).

6.2. Detección de piel

El estudio descrito en el Capítulo 4 dió lugar a numerosas pruebas para determinar en que medida se mejoraba el estado del arte. Para ello, se realizó una comparativa entre los métodos del SoA (*State of Art*) y los nuevos métodos que se propusieron (umbralización con valores escogidos del estudio, SDMIM y RF), a partir de tres conjuntos de imágenes con la piel anotada de los principales datasets. En este apartado primero se describirán los datos usados para la comparativa, los métodos de detección de piel utilizados y la métrica utilizada para su evaluación, para después presentar los resultados obtenidos y conclusiones sobre ellos.

6.2.1. Datos

Se utilizaron tres datasets para esta comparativa: ED (VPU), LIRIS (HARL) y SSG (apéndice B). Para probar los distintos algoritmos, se escogieron entre 25 y 30 imágenes lo más distintas posible dentro de cada dataset y se anotó la piel en ellas. De esta manera, la comparativa se

haría comparando a nivel de píxel las máscaras de piel obtenidas con dichos algoritmos con las imágenes anotadas, midiendo así la eficiencia de estos. La tabla 6.1 muestra el número total de muestras de piel utilizadas para la comparativa.

Dataset	ED	LIRIS	SSG
# imágenes	25	30	25
# total píxeles	27041	680548	203963
# píxeles anotados piel	10816	272219	81585

Tabla 6.1: Datos de piel utilizados para test

Para los algoritmos que requieren entrenamiento, se anotaron más imágenes de cada dataset, con el fin de entrenar al sistema con imágenes distintas a las usadas en el testeo. En este caso, para el dataset ED, debido a que la imágenes no presentan un gran número de píxeles de piel, se anotaron más imágenes que del resto de datasets para el entrenamiento. La tabla 6.2 muestra el número total de muestras de piel de cada dataset de entrenamiento.

Dataset	ED	LIRIS	SSG
# imágenes	60	25	25
# total píxeles	58441	478988	212591
# píxeles anotados piel	23376	191595	85036

Tabla 6.2: Datos de piel utilizados para entrenamiento

Para cada dataset se toma un número de muestras igual al número de muestras tomadas del dataset que menos muestras tenga, así el sistema es entrenado con cada dataset en la misma proporción. También hay que tener en cuenta que la proporción de piel y no-piel no está equilibrada nunca, ya que normalmente hay más no-piel que piel en una imágen. Lo normal es entrenar con más muestras aquel modelo que aparezca con más frecuencia, por tanto, la proporción de piel y no-piel tomada de cada imágen para entrenar el sistema es de 40 % y 60 % respectivamente.

6.2.2. Algoritmos

El SoA está formado por aquellos algoritmos que ya estaban desarrollados a la hora de iniciar este proyecto y son los siguientes:

- Umbralización con valores por defecto (U-SoA): sistema de detección por umbralización clásica, disponible desde el comienzo del proyecto y configurado con unos umbrales predeterminados para las combinaciones “H-S” y “Cb-Cr”.
- *Adaptive Skin Detector* de OpenCV (ASD-SoA): sistema usado inicialmente en el prototipo del VPU-Lab, basado en el código disponible en las librerías de OpenCV 2.2 C++ [1].

[1] http://opencv.willowgarage.com/

- *Random Forest* (RF-SoA): sistema desarrollado a partir del código para *Random Trees* disponible en OpenCV 2.2, basado en el estudio [39]. En este proyecto se implementó el algoritmo, el ajuste de parámetros y el código necesario para la lectura de vídeos e imágenes de entrenamiento. Así mismo, se diseñó el sistema de muestra de resultados, comparando resultados con *groundtruth*. El SoA para este algoritmo es el propuesto por [18] para detección de piel, que trabaja sobre los tres canales (3D) de HSV.

Los nuevos algoritmos utilizados, descritos en el apartado 4.3, son aquellos con los que se pretenden mejorar los resultados obtenidos por el SoA, y son los siguientes:

- Umbralización con umbrales escogidos (U): Sistema de detección por umbralización clásica con umbrales escogidos del estudio del apartado 4.3.

- *Random Forest* (RF): Algoritmo RF trabajando sobre 2 canales (2D) de los estudiados en el apartado 4.2.

- *Skin Detection by Mutual Information Maximization* (SDMIM): Versión del prototipo para la detección de sombras desarrollado en el VPU-Lab [38] en el entorno MATLAB. Se obtuvieron resultados para las dos configuraciones descritas en el apartado 4.3.2.

6.2.3. Métrica

La comparativa se hizo anotando los resultados de TPR (*True Positives Rate)* y FPR (*False Positives Rate*), los cuales se definen como:

$$TPR = \frac{\#PíxelesPielDetectadosCorrectamente}{\#PíxelesPielAnotados} \cdot 100 \tag{6.1}$$

$$FPR = \frac{\#PíxelesPielDetectadosErróneos}{\#PíxelesNoPielAnotados} \cdot 100 \tag{6.2}$$

Hay que tener en cuenta que cobra más importancia el TPR, puesto que el FPR se supone que se reducirá mucho si la detección de piel se realiza sólo sobre el blob de la persona. Además, también hay que tener en cuenta que el FPR normalmente será muy bajo debido a la manera en que se obtiene dicho valor, pero cualquier aumento pequeño en realidad es un empeoramiento del algoritmo bastante grande, en lo que se refiere a la no-piel detectada como piel.

Aun así, ambos valores se combinaron para obtener un valor único con el que poder comparar directamente dos algoritmos distintos. Dicho valor se calcula como se expresa en la fórmula 6.3, pues se busca que el valor de TPR sea lo más alto posible y el valor de FPR lo más bajo posible:

$$Rend = (TPR + (100 - FPR))/2 \tag{6.3}$$

6.2.4. Resultados

En este apartado se presentarán y discutirán los principales resultados obtenidos para cada algoritmo (la versión completa de los resultados está disponible en el apéndice D) y finalmente se compararán los mejores resultados con el estado del arte actual.

6.2.4.1. Observaciones generales

Estas observaciones han sido realizadas a partir de la visualización directa de las numerosas máscaras de piel obtenidas para cada canal de color individualmente, comparándolas con el *groundtruth*. Estas observaciones corroboran las conclusiones del capítulo 4.

- "H" y "a" son los canales que individualmente mejor reconocen la piel. Prácticamente por si solos son capaces de reconocerla bastante bien, así como reconocer lo que no es piel. Ahora bien, normalmente uno de los dos reconoce más piel que el otro, pero también más no-piel. Por tanto, el resultado de unirlos da una máscara de piel casi igual a aquel que reconoce menos piel, pero con pérdida de algún píxel de piel debido al que reconoce más, por no reconocer alguno de los píxeles que reconoce el otro (figura 6.1).

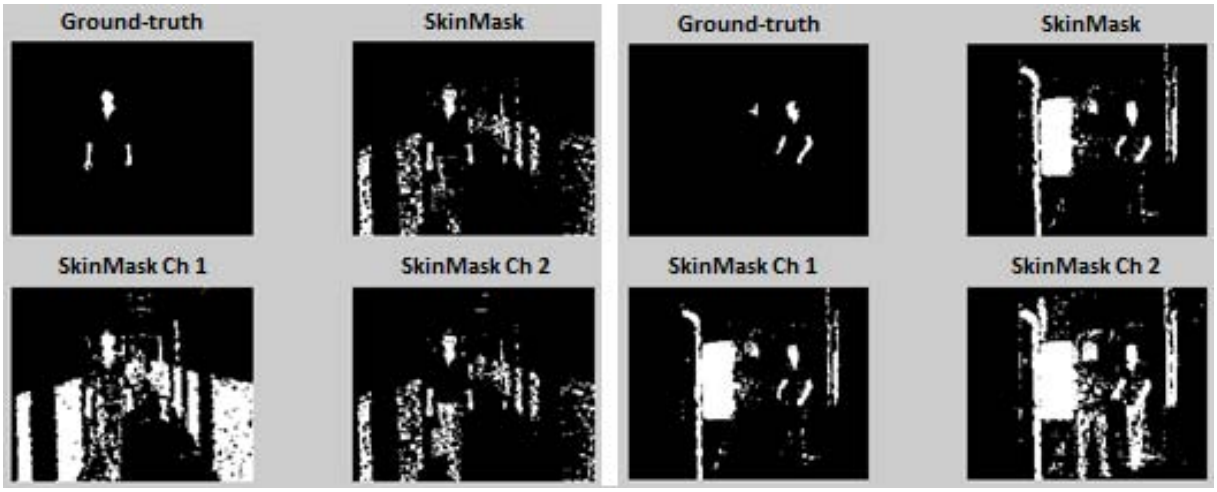


Figura 6.1: Ejemplo de "H-a" con umbrales óptimos (SDMIM)
Las imágenes de arriba representan el groundtruth y la máscara de piel obtenida. Las de abajo representan la máscara obtenida con cada canal independiente.

- "S" y "b" son canales que reconocen casi toda la piel, pero también reconocen como tal muchísima no-piel. Ahora bien, generalmente la piel que reconocen concuerda con "H" o "a", y además no reconocen como piel zonas de no-piel que los canales "H" y "a" si reconocían como tal erróneamente (figura 6.2).

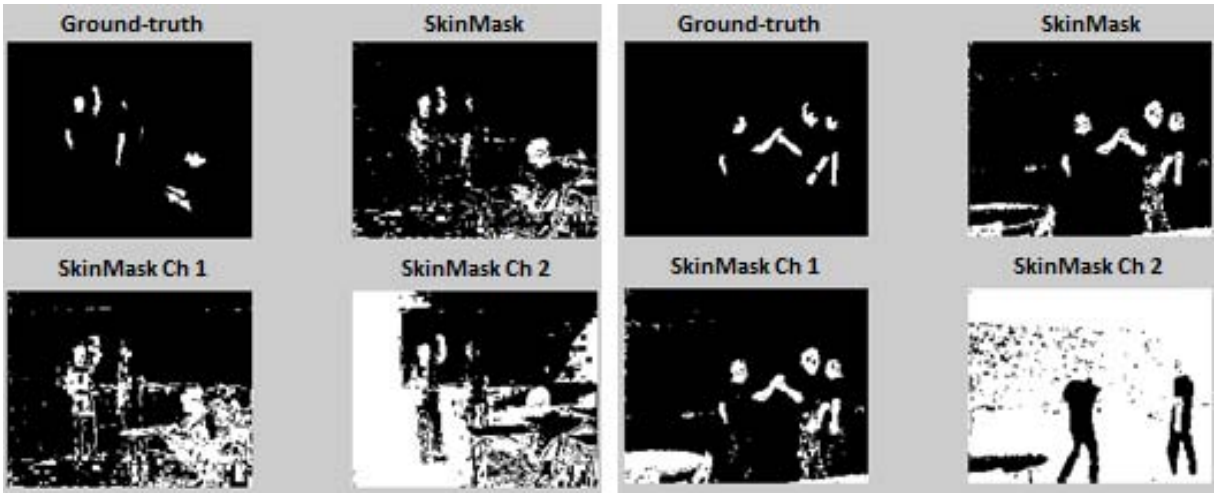


Figura 6.2: Ejemplo de "H-S" y de "a-b" con umbrales óptimos (SDMIM)

- "Cb" y "Cr" pertenecen al mismo espacio de colores, y dan máscaras muy distintas entre ellos, pero se complementan muy bien (figura 6.3).

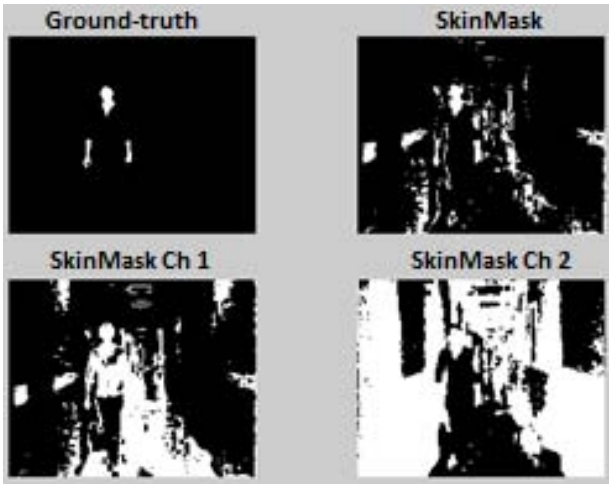


Figura 6.3: Ejemplo de "Cb-Cr" con umbrales óptimos (SDMIM)

6.2.4.2. Algoritmo de umbralización

La umbralización con valores por defecto tenía umbrales no muy bien escogidos, como muestra la tabla 6.3, donde se observa que TPR no es muy alto con "H-S", y si lo es con "Cb-Cr" pero presenta un FPR muy elevado. Por tanto, se han usado los valores obtenidos en el estudio de canales de colores cuyos resultados se presentan en la tabla 6.4. Aún escogiendo los umbrales del estudio, la umbralización directa no deja de ser un método demasiado simple y que no proporciona generalmente buenos resultados, como se pueden ver en los resultados de TPR y FPR. En

Figura 6.4: Ejemplos de umbralización con umbrales escogidos para LIRIS, ED y SSG

la figura 6.4 se muestran ejemplos del detector por umbralización. En la primera imagen la piel se detecta bien, aunque detecta mucho fondo como piel. Las otras tres imágenes son ejemplos de que no funciona correctamente.

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	30	1	65	61	7	77	50	3	74	72
Cb-Cr	72	10	81	91	50	71	89	16	87	80

Tabla 6.3: Umbralización con valores por defecto (U-SoA)

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	56	1	78	72	12	80	80	5	88	82
H-a	47	1	73	55	8	74	75	4	86	78

Tabla 6.4: Umbralización con umbrales escogidos del estudio (U)

6.2.4.3. Algoritmo *Adaptive Skin Detector*

Este algoritmo detecta la piel más o menos bien para los dataset ED y SSG, aunque no tanto para el dataset LIRIS, pero también detecta erróneamente como piel mucha no-piel, como se

puede ver en la tabla 6.5 y la figura 6.5.

ED			LIRIS			SSG			
TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
73	21	76	77	58	60	90	22	84	73

Tabla 6.5: *Adaptive Skin Detector* de OpenCV (ASD-SoA)

Figura 6.5: Ejemplos de *Adaptive Skin Detector* (ASD-SoA) para los dataset LIRIS y ED

6.2.4.4. Algoritmo SDMIM

Las dos primeras observaciones generales son útiles para entender la búsqueda óptima de umbrales de SDMIM. Por su forma de búsqueda, "H-a" no llega a destacar por encima de otras elecciones de espacios de colores como "H-S" o "a-b", que utilizan uno de esos dos canales (figuras 6.1 y 6.2), lo cual resulta extraño según lo visto en el estudio de canales de colores. Pero si uno se basa en lo explicado en la observación general de cada canal de color, tiene sentido. Como se explicó en 4.3.2, la búsqueda de umbrales óptimos se realiza utilizando como *agreement* el TPR y FPR obtenido a partir del *groundtruth*. Por tanto, si usamos "H" o "a" en la combinación de canales, el resultado final será una máscara igual a la máscara obtenida con uno de esos dos, independientemente del segundo canal que se haya usado. En la tabla 6.6 se puede ver que en general, las combinaciones usadas se comportan de manera similar, y no detectan demasiado fondo como piel. Las figuras 6.1, 6.2 y 6.3 muestran ejemplos del algoritmo.

En cambio, SDMIM con búsqueda normal de umbrales que utiliza la correlación como medida de *agreement*, "H-a" sí que destaca un poco respecto al resto ya que, como se vió en las tablas de

correlaciones, son los canales que mejor resultado obtienen, y por tanto, la búsqueda de umbrales se realizará con mayor acierto para estos canales. Si la correlación entre dos canales de por sí es baja, las máscaras de piel de ambos canales serán normalmente muy distintas, y por tanto, aunque el sistema busque umbrales que mejoren esa correlación, la mejor máscara resultante de dos canales será peor que la resultante de dos canales con alta correlación. En este caso, para "H-S" busca que "S" se parezca más a "H", cuando son dos canales muy distintos, de ahí que falle algo más que "H-a". Lo mismo ocurre con "a-b". Como se puede ver en los resultados de la la tabla 6.7, "H-a" es la combinación que más se acerca al punto óptimo obtenido en la búsqueda óptima (utilizando *groundtruth*), y es el que mejor se comporta en media. Y en general, se puede ver que las combinaciones "H-S", "a-b" y "a-S" tambien dan buenos resultados.

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	74	1	87	83	12	86	83	4	90	88
H-a	74	1	87	82	11	86	83	4	90	88
a-b	89	3	93	86	12	87	93	6	94	91
a-S	92	3	95	88	15	87	91	6	93	92
H-b	73	2	86	83	13	85	83	4	90	87
Cb-Cr	82	5	89	88	16	86	91	5	93	89

Tabla 6.6: SDMIM con búsqueda de umbrales óptimos

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	55	1	77	76	19	79	64	3	81	79
H-a	72	2	85	81	19	81	83	6	89	85
a-b	70	5	83	77	17	80	83	8	88	84
a-S	84	4	90	76	23	77	77	6	86	84
H-b	66	2	82	75	16	80	78	6	86	83
Cb-Cr	88	59	65	88	39	75	79	20	80	73

Tabla 6.7: SDMIM con búsqueda normal de umbrales

6.2.4.5. Algoritmo RF (SoA y otros canales)

El estado del arte del *Random Forest* (RF-SoA) es el presentado en el estudio [18], del cual se obtuvo la idea: RF con HSV y entrenamiento-testeo con el mismo dataset. En este caso, el resultado es el mejor posible, tanto en TPR como FPR, lo cual es lógico si se entrena con el mismo dataset y se usa la información de luminancia contenida en el canal "V".

La afirmación de [18] sobre la potencial mejora al utilizar el canal V es cierta, pero sólo si se dan ciertas condiciones. La luminancia aporta más información de la piel en el caso de que siempre sea similar en todo el entrenamiento, aunque no es una característica de la piel, sino del

escenario. Por eso, si se añadieran otros datasets donde la luminancia variase considerablemente, añadirla como característica de la piel no tendría sentido pues empeoraría los resultados. Si se quiere un sistema que pueda trabajar en muchos escenarios distintos y con iluminaciones distintas, "V" no es una característica adecuada ya que el color de piel no depende de la iluminación. Incluso en un mismo escenario, si la iluminación cambia, el entrenamiento del sistema ya no será el adecuado para ese cambio, pues el sistema no se adapta una vez entrenado.

Lo anterior se puede demostrar con los resultados obtenidos al utilizar varios dataset para entrenar el modelo de piel con RF-SoA. La tabla 6.8 muestra los resultados obtenidos para HSV con distinto número de datasets para entrenamiento, donde se observa que empeoran ligeramente a medida que se añaden datasets al entrenamiento. Esto se debe a la incorporación de la luminancia en el entrenamiento. Se han incluído también en la tabla 6.8 los resultados con el mismo entrenamiento y testeo, aunque realmente no tiene sentido entrenar con lo mismo que se pretende probar.

En cualquier caso, el empeoramiento es reducido debido a que cuanta más información se añade al algoritmo *Random Forest*, más bueno es, pero hay que tener en cuenta que cuantos más datos se le dan para entrenar, menos peso tendrán individualmente. De ahí que añadir la "V" no provoque una disminución tan pronunciada del rendimiento del sistema, teniendo ya dos canales que discriminan bien la piel.

	ED			LIRIS			SSG			
Dataset train	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
1 (=imgs test)	78	1	89	91	11	90	89	7	91	90
1 (≠imgs test)	60	1	80	88	10	89	89	7	91	87
2 (DS test + otro)*	61	1	80	89	14	88	86	7	90	86
3 (ED+LIRIS+SSG)	61	1	80	89	18	86	85	6	90	85
4 (3+SKIN)	57	1	78	89	16	87	82	5	89	85
5 (4+MCG)	53	1	76	86	12	87	75	4	86	83

Tabla 6.8: RF utilizando HSV (RF-SoA)
**Train* con mismo dataset para *test* (distintas imágenes) y otro distinto (2 dataset)

Se realizaron pruebas similares al RF-SoA con varias combinaciones de 2 canales de color (2D), omitiendo siempre el canal referente a la luminancia. En la tabla 6.9 se muestran los resultados para 5 combinaciones de canales de color y entrenamiento con 1 y 3 datasets. Se observa que, en general para todas las combinaciones, los resultados también empeoran al introducir más datasets al entrenamiento, a pesar de no usar luminancia. Esto se debe a que la piel para el entrenamiento es más generalista y no tan específica para un dataset, por tanto, el rango de valores posibles para la piel se expande y la probabilidad de que un píxel cualquiera sea piel aumenta. Los píxeles considerados como fondo en un dataset puede que incluyan píxeles que en otro dataset se consideran piel, aumentando más la confusión del sistema.

Figura 6.6: Ejemplos de RF con "H-a" para los dataset LIRIS, ED y SSG

		ED			LIRIS			SSG			
		TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	a)	67	1	83	86	12	87	87	6	91	87
	b)	61	1	80	91	16	88	77	5	86	85
H-a	a)	60	1	80	87	9	89	86	7	90	86
	b)	57	1	78	94	15	90	77	5	86	85
a-b	a)	59	1	79	88	10	89	85	6	90	86
	b)	57	1	78	94	16	89	76	5	86	84
a-S	a)	59	1	79	85	12	87	86	6	90	85
	b)	56	1	78	92	17	88	76	5	86	84
Cb-Cr	a)	58	1	79	88	9	90	86	6	90	86
	b)	57	1	78	94	17	89	80	5	88	85

Tabla 6.9: RF utilizando dos canales
a) *Train* con mismo dataset para *test* (distintas imágenes)
b) *Train* mezcla de 3 datasets (ED, LIRIS, SSG)

Por último, se probó también alguna combinación de 3 canales sin luminancia, como son "H-a-Cr", y "a-S-Cr". Ambas dan resultados que no son malos comparándolas con otras combinaciones, pero tampoco destacan respecto a combinaciones de 2 canales.

6.2.4.6. Comparativa de algoritmos y conclusiones

A continuación, se comparan los mejores resultados de cada algoritmo en la tabla 6.10.

En la figura 6.7 se puede observar el rendimiento del algoritmo RF con las combinaciones de canales "HSV", "HS" y "Ha" frente al algoritmo SDMIM con la combinación "Ha" para un número variable de datasets para entrenamiento. Esta gráfica ha sido obtenida a partir de los resultados presentados en el Anexo D. Se puede observar que, salvo alguna excepción, en general la detección de piel con RF empeora a medida que se utiliza un dataset más para entrenarlo. Así mismo, se corrobora que los resultados del algoritmo RF con información de luminancia también decaen a medida que se van utilizando más datasets de entrenamiento. El SDMIM se mantiene más o menos constante en torno al mismo valor. El SDMIM no tiene un rendimiento

	ED	LIRIS	SSG	MEDIA
U-SoA (HS)	65	77	74	72
ASD-SoA (HS)	76	60	84	73
RF-SoA (HSV) *	80	89	91	87
U (HS)	78	80	88	82
SDMIM (Ha) **	85	81	89	85
RF (HS) *	83	87	91	87
RF (Ha) *	80	89	90	86

Tabla 6.10: Comparativa rendimiento
* *Train* con mismo dataset para *test* (distintas imágenes)
** Obtención de media y desviación con 5 datasets

completamente constante ya que, en cierto modo, si depende de entrenamiento, pues depende de cuántos datasets se hayan usado para estimar la media y desviación de las gaussianas utilizadas.

Los puntos marcados con círculos en la figura 6.7 representan los valores medios máximos de rendimiento, presentados en la tabla 6.10 (valores redondeados).

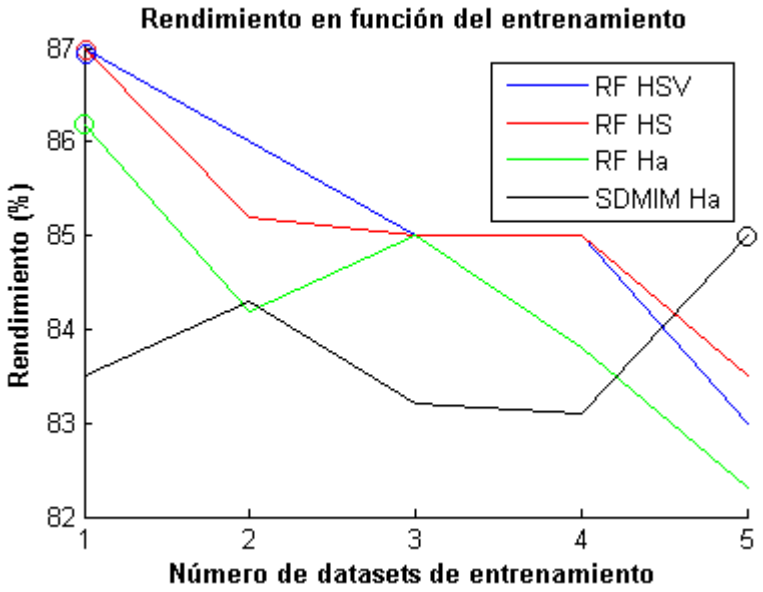


Figura 6.7: Rendimiento frente a número de datasets para entrenamiento

Si hubiera que elegir un algoritmo de los propuestos, para un sistema general destinado a su uso en cualquier escenario convendría usar el SDMIM, que no depende apenas del entrenamiento y presenta una precisión similar al estado del arte. En cambio, si se quisiera un sistema especializado en un escenario concreto, el RF es mucho más fiable, entrenado sólo con datos de dicho escenario.

6.3. Reconocimiento de eventos

6.3.1. Datasets usados

El sistema propuesto es evaluado con los tres dataset principales de este proyecto: ED, LIRIS (descritos en el apartado 2.4) y SSG (apéndice B). A continuación, primero se describen brevemente las dificultades que presentan estos tres dataset relacionados con la detección de eventos y las ocurrencias de los eventos en cada uno (tablas 6.11 y 6.12), para después presentar los resultados obtenidos con ellos.

	Dificultades
SSG	Eliminación de reflejos y sombras incorrecta. Muchas regiones indebidas en el frente. Luminosidad muy alta dificulta la extracción del frente y empeora detección de piel. Baja distancia entre cámara y personas. Plano demasiado cercano a las personas.
ED	Objetos demasiado pequeños no generan blob. Varias personas en escena generan numerosas oclusiones. Distancia entre cámara y personas e iluminación baja dificulta segmentación de frente.
LIRIS	Fondo difícil de extraer, presencia de personas al comienzo de las secuencias. Eliminación de reflejos y sombras incorrecta. Varias personas en escena genera numerosas oclusiones. Perspectiva muy dispar, cámara situada en lugares no realistas y complicadas de tratar. Eventos realizados de manera compleja y en situaciones difíciles de tratar.

Tabla 6.11: Dificultades de los dataset

Dataset	Ocurrencia de Eventos							Complejidad
	BO	EN	LO	UB	HS	KB	TE	
SSG	32	3	9	13	3	8	10	L
ED	46	44	-	-	9	20	-	M
LIRIS-train	9	20	-	6	11	12	5	H

Tabla 6.12: Descripción de datasets

6.3.2. Métrica

Teniendo en cuenta la localización, un evento se detecta correctamente en función de la cantidad de solape espacial y temporal entre el evento definido en el *groundtruth* y la detección:

$$Deteccion = \begin{cases} 1 & si\ SER > 0\ y\ STR > 0 \\ 0 & resto \end{cases} \tag{6.4}$$

donde SER y STR son, respectivamente, el Solape Espacial Relativo y el Solape Temporal Relativo entre el evento detectado y el definido en el *groundtruth*. Por tanto, con que el evento

Dataset	BO		EN		LO		UB		HS		KB		TE		Total		
	P	R	P	R	P	R	P	R	P	R	P	R	P	R	P	R	F
SSG	0.45	0.94	0.75	1	1	0.44	0.69	0.81	1	0.33	0.88	0.88	0.90	0.90	0.60	0.81	0.69
ED	0.56	0.74	0.89	0.77	-	-	-	-	0.88	0.78	0.95	0.95	-	-	0.74	0.79	0.76
LIRIS-train	0.56	0.56	0.40	0.10	-	-	0.50	0.67	0.67	0.36	1	0.08	1	0.60	0.59	0.30	0.40
MEDIA	0.52	0.75	0.68	0.62	1	0.44	0.60	0.74	0.85	0.49	0.94	0.64	0.95	0.75	0.64	0.63	0.62

Tabla 6.13: Rendimiento general del sistema

sea detectado mínimamente dentro del *groundtruth*, la detección será correcta.

Para evaluar el rendimiento del sistema, se utilizaron las medidas de *Precision* (P), *Recall* (R) y una combinación de las mismas llamada *F-Score* (F), definidas a continuación:

- *Precision* (P): Indica la proporción de eventos detectados correctamente frente a todos los eventos detectados errónea y correctamente.

$$P = \frac{\#TP}{\#TP + \#FP} \tag{6.5}$$

- *Recall* (R): Indica la proporción de eventos detectados correctamente frente al número de eventos que suceden en escena y que debería detectar.

$$R = \frac{\#TP}{\#TP + \#FN} \tag{6.6}$$

- *F-Score* (F): Combina *Recall* y *Precision* en una sola puntuación cuya ventaja es que el mínimo de los dos valores de rendimiento se enfatiza:

$$F = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{6.7}$$

Aunque los resultados finales para la fase 1 y la fase 2 se midieron con los mismas métricas, las métricas utilizadas por la organización de la competición son más complejas. Para una descripción detallada de las métricas de utilizadas por la organización, véase el apéndice E.

6.3.3. Resultados Fase 1 y su interpretación

Los resultados para cada dataset se muestran en la tabla 6.13. Recordar que estos resultados fueron obtenidos con la primera versión del sistema, diseñada para funcionar correctamente en todos los casos posibles de los tres datasets.

Aunque los resultados para SSG y ED son bastante buenos, se puede observar como el rendimiento para el dataset LIRIS-train baja considerablemente. La figura 6.8 presenta algunos de los fallos del sistema. En la primera fila de imágenes se puede ver que la máscara de frente no se corresponde con lo visto en la imágen original debido a una mala sustracción de fondo, además

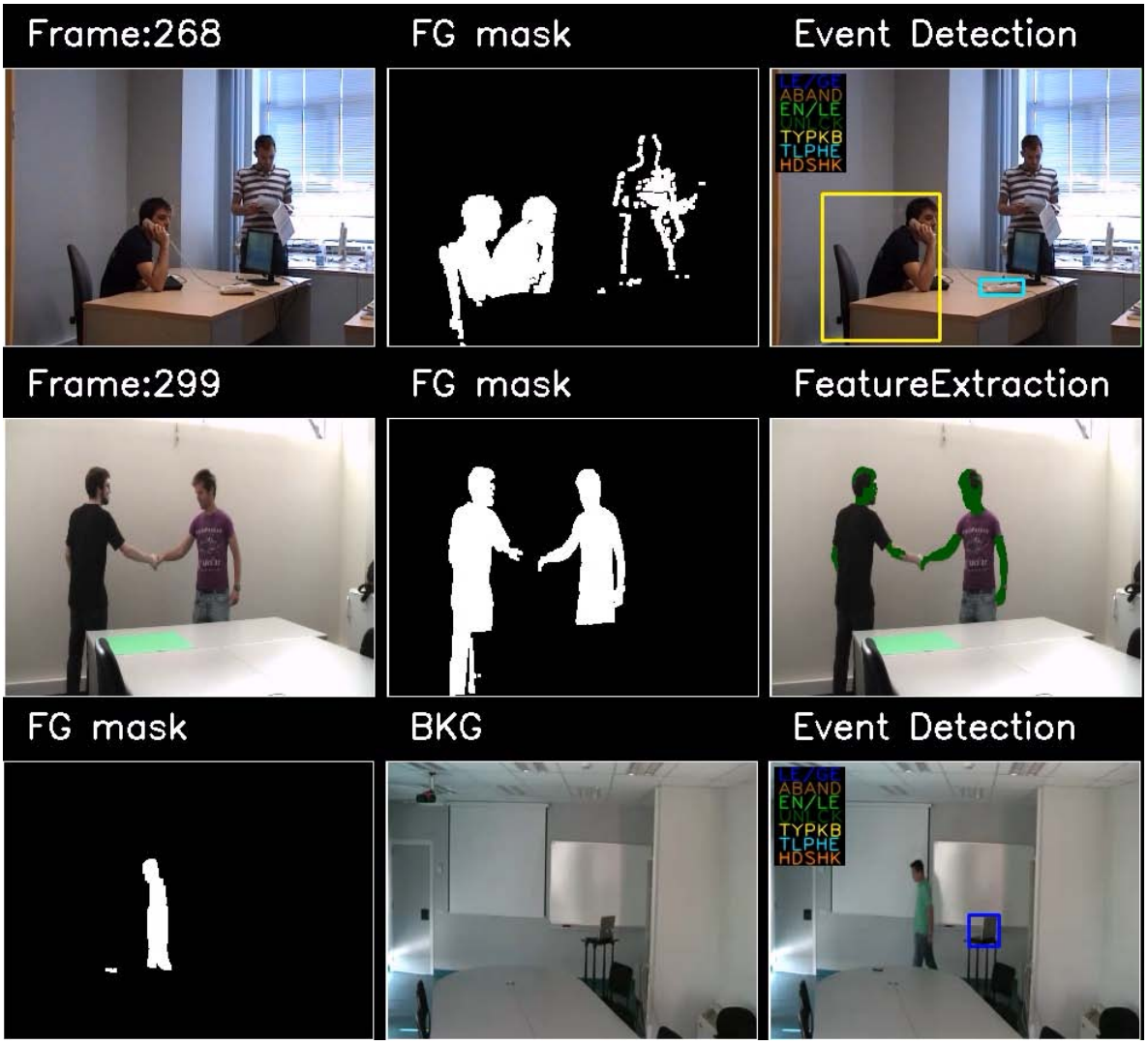


Figura 6.8: Ejemplos de fallos

de que el evento KB no está bien diseñado para ese ejemplo, ya que la persona no está tecleando, sino hablando por teléfono. El evento se detecta erróneamente porque la piel del brazo solapa con el teclado. En la segunda fila se puede ver que de nuevo falla la sustracción de fondo justo en el punto de interés del apretón, haciendo que no se detecte la piel y, por tanto, no se detecte el evento. En la tercera fila se observa que, aunque la máscara de frente está bien, el teléfono móvil (objeto pequeño sobre la mesa) no se ha detectado cono BO, mientras que el portátil si (integrado en la imágen de fondo y marcado en la de detección de eventos). Los tres fallos aqui descritos son causa de las dificultades de cada dataset explicadas en el apartado 6.3.1.

También se realizaron pruebas con los vídeos del dataset LIRIS-test, pero los resultados fueron peores que con el dataset LIRIS-train. De hecho, como se explicó en anteriores apartados,

más adelante se decidió realizar todos los cambios pertinentes para adaptar el sistema al dataset LIRIS, en especial, al LIRIS-test. Los resultados del sistema modificado específicamente para el dataset LIRIS-test son los presentados en el siguiente apartado, y fueron calculados por la propia organización de la competición HARL 2012.

6.3.4. Resultados Fase 2 y su interpretación

La organización de la competicion HARL 2012 realizó su propia evaluación de los sistemas presentados por los participantes (de 70 participantes inscritos, solamente 4 enviaron resultados). El primer participante (ADSC-NUS-UIUC) fue una colaboración entre las siguientes instituciones: Advanced Digital Sciences Center (Singapur), National University of Singapore (Singapur) y University of Illinois at Urbana-Champaign (EEUU). El segundo participante (TATA-ISI) comprendia dos instituciones de India: Innovation Lab, Tata Consultancy Services e Indian Statistical Institute. El tercer participante (IACAS) era Chinese Academy of Sciences (Beijing, China). El sistema propuesto en este proyecto (VPULAB-UAM) fue el cuarto participante. Además, existían dos datasets de la competición: D1 (información de color y profundidad) y D2 (información de color, utilizado en este proyecto). La evaluación se realizó de dos formas distintas: teniendo en cuenta sólo la detección pura del evento (es decir, si el video contiene el evento detectado) y considerando la localización espacial y temporal. Para más información acerca de la evaluación, acudir al apéndice E.

El participante ADSC-NUS-UIUC utilizó el dataset D1. Su sistema realiza detección de objetos y personas por separado, basando esta última en un detector de poses humanas entrenado con las poses extraídas del *groundtruth*. Después, extraen atributos de las interacciones entre personas y objetos. Por último, clasifican los escenarios según la orientación de la escena. Con toda esta información, localizan y detectan los eventos.

El participante TATA-ISI utilizó el dataset D1. Su sistema se basa en dos etapas. Primero realiza la segmentación de los objetos móviles en escena a partir de la variación del valor de cada píxel y mediante información de profundidad. Después, evalúa las características extraídas de las imágenes segmentadas y utiliza las técnicas de reconocimiento de acciones por poses descritas en [40].

Por último, el participante IACAS utilizó el dataset D2. Su sistema se entrena a partir del *groundtruth* extrayendo STIPs (*Space-Time Interest Points*) con el método descrito en [41]. Con los STIPs entrena una SVM (*Support vector machine*) para cada actividad. Después, para el testeo extrae de nuevo los STIPs y los clasifica con las SVMs entrenadas. La localización de los eventos se realiza usando una variante de la técnica descrita en [42].

6.3.4.1. Rendimiento de detección pura y reconocimiento - sin localización

La primera medida del rendimiento ignora la información de localización espacial y temporal del resultado (número de *frame* y *bounding box*) y sólo da información del rendimiento de la detección pura y reconocimiento a través de las medidas. Los resultados se muestran en la tabla 6.14, con el resultado obtenido para nuestro sistema marcado en negrita.

Equipo	Dataset	Recall	Precision	F-Score
ADSC-NUS-UIUC	D1	0.74	0.41	0.53
TATA-ISI	D1	0.08	0.17	0.11
VPULABUAM	**D2**	**0.36**	**0.66**	**0.46**
IACAS	D2	0.30	0.46	0.36

Tabla 6.14: Rendimiento sin localización

Podemos afirmar que el sistema desarrollado funciona bastante bien en comparación a los otros participantes. Aunque presenta un *recall* bajo, es decir, sólo detecta un 36 % de los eventos que debería, es el segundo en este valor, y es el sistema más preciso de todos los presentados con un 66 % de *precision*. Si se comparan estos valores con los obtenidos en el apartado 6.3 para el dataset LIRIS-train, se observa como el sistema ha perdido en *precision*, pero ha ganado en *recall*. Ahora bien, hay que tener en cuenta que el dataset LIRIS-test demostró ser más difícil que el LIRIS-train, y además se realizaron numerosos cambios y adición de nuevos módulos, lo cual añade variabilidad a los resultados respecto a los obtenidos en la fase 1 del proyecto.

Si comparamos nuestros resultados con los del resto de participantes, vemos que sin usar información de profundidad pudimos obtener unos resultados que se acercan bastante a los del primer participante, que sí utiliza dicha información. Del mismo modo ocurre con los resultados del cuarto participante. La información de profundidad puede mejorar mucho la segmentación, lo cual repercute en el *recall*, pues una mala segmentación puede provocar que se pierdan muchos eventos. Ésta posiblemente sea la causa de que nuestro sistema y el del cuarto participante presenten un *recall* mucho más bajo que el del primer participante. Ahora bien, comparando el valor de *precision*, se puede observar que los dos sistemas que utilizan la pose humana para el reconocimiento presentan un valor más bajo que los que no. Esto puede significar que la pose no es una característica suficientemente discriminativa para diferenciar unos eventos de otros.

6.3.4.2. Rendimiento de detección, reconocimiento y localización a un nivel de calidad del 10 %

Teniendo en cuenta la localización, determinar si una acción se detecta correctamente requiere establecer umbrales referentes a la cantidad de solape entre la acción definida en el *groundtruth* y la detectada. Dichos umbrales establecen que al menos un 10 % del resultado solape con el groundtruth. En apéndice E se puede encontrar más información acerca de este protocolo de

evaluación, así como gráficas del rendimiento del sistema variando dichos umbrales. En la tabla 6.15 aparecen los valores de *Precision* y *Recall* para este caso, con el resultado obtenido para nuestro sistema marcado en negrita.

Equipo	Dataset	Recall	Precision	F-Score
ADSC-NUS-UIUC	D1	0.63	0.33	0.44
TATA-ISI	D1	N/A	N/A	N/A
VPULABUAM	**D2**	**0.04**	**0.08**	**0.05**
IACAS	D2	0.03	0.04	0.03

Tabla 6.15: Rendimiento con localización

Si se observan los resultados obtenidos en este caso, queda claro que el sistema no es, a priori, fiable. Ahora bien, hay que tener en cuenta que las anotaciones de los eventos han sido realizadas a mano y según el criterio de la organización. Ese criterio comprende cuándo comienza y cuándo acaba el evento y cuánto espacio ocupa en la imagen, lo cual depende completamente del juicio del que lo realiza, y que no tiene porque coincidir con el resultado obtenido por el sistema debido a que la organización no proporcionó reglas para los eventos anotados. En la figura 6.9 se puede observar como el evento de abandono (UB) es marcado de manera completamente distinta en nuestro sistema que en la anotación, tanto espacial como temporalmente. De hecho es probable que cuando se marca el evento en el sistema, en la anotación ya no esté marcado. Lo mismo ocurre con el ejemplo del evento de hablar por teléfono (TE), donde en el sistema se marca el teléfono, mientras que la anotación marca a la persona.

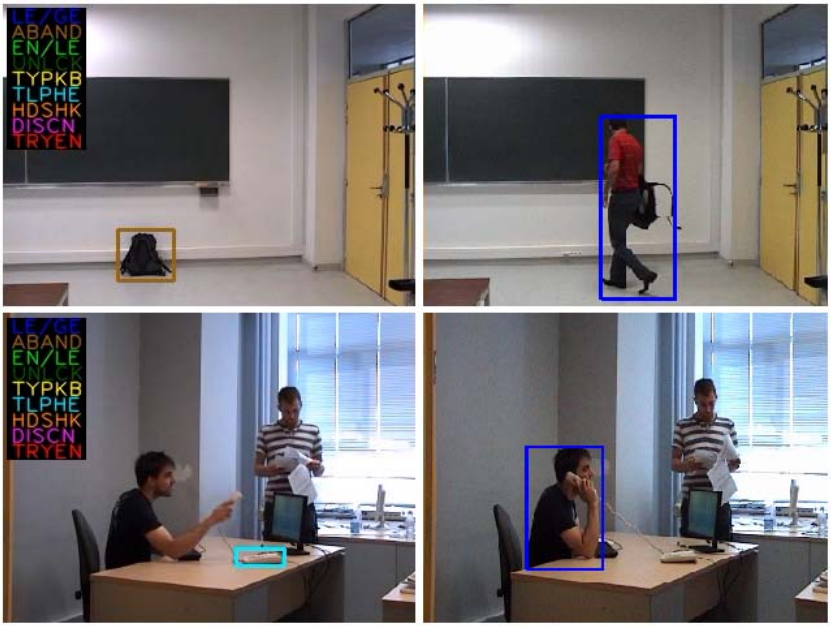


Figura 6.9: Comparativa de la detección de nuestro sistema con la anotación dada en LIRIS

Por tanto, sabiendo como funciona el sistema desarrollado en este proyecto y conociendo como han sido realizadas las anotaciones, no es de extrañar este resultado, y no aporta información definitiva del rendimiento del sistema. De hecho, sólo un sistema de los 4 presentados ha obtenido buenos resultados en este apartado, y no es comparable con el trabajo desarrollado en este proyecto pues dicho sistema utiliza un dataset distinto que posee información de profundidad.

6.3.4.3. Matrices de confusión

En la figura 6.10 se muestra la matriz de confusión desarrollada en esta evaluación para el sistema diseñado en este proyecto. Las matrices de confusión del resto de participantes se encuentran en el apéndice E. La matriz de confusión representa sólo parejas de acciones del groundtruth y detectadas. Además, hay que indicar que las acciones del groundtruth no detectadas NO están incluídas y las acciones detectadas sin equivalente en el groundtruth NO están incluídas.

detection / groumd truth	1 DI	2 GI	3 BO	4 EN	5 ET	6 LO	7 UB	8 HS	9 KB	10 TE
1 DI	50.0	0.0	0.0	0.0	0.0	0.0	0.0	50.0	0.0	0.0
2 GI	50.0	0.0	50.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3 BO	0.0	0.0	80.0	0.0	0.0	0.0	20.0	0.0	0.0	0.0
4 EN	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
5 ET	33.3	0.0	0.0	0.0	66.7	0.0	0.0	0.0	0.0	0.0
6 LO	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
7 UB	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8 HS	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9 KB	0.0	0.0	66.7	33.3	0.0	0.0	0.0	0.0	0.0	0.0
10 TE	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0

Figura 6.10: Matriz de confusión de VPULABUAM

Figura 6.11: Ejemplos de detecciones correctas

Observando la matriz de confusión, hay ciertos datos que demuestran el error en la localización espacio-temporal de eventos. Por ejemplo, de los LO del *grountruth* asociados con detecciones, que el 100 % se correspondan con ET no tiene sentido, cuando en las pruebas realizadas en este proyecto se detectaban bastantes eventos LO correctamente. Otro caso es el de UB, que en las pruebas realizadas en este proyecto se veía claramente que era un evento que se detectaba en bastantes ocasiones también. En cambio, la matriz refleja que solamente un 20 % se asocian con BO mientras que un 80 % de UB permanece sin asociar. En la figura 6.11 se muestra un caso de cada uno de estos eventos detectado correctamente, a diferencia de lo que muestra la matriz. En la primera imagen aparece una bolsa abandonada y detectada correctamente y la segunda imagen pertenece a una secuencia en la que una persona desbloquea la puerta y la atraviesa cerrándola tras de sí, tras lo cual el sistema detecta correctamente el evento.

6.3.4.4. Conclusión

El sistema presentado a la competición ha quedado segundo en todas las clasificaciones, lo cual se puede considerar bueno en vista de los resultados obtenidos por otros participantes y la baja participación en la competición, probablemente producto de la dificultad de la misma. Ahora bien, en el fondo no ha logrado detectar muchos de los eventos presentes en el dataset LIRIS, lo cual indica que el sistema es aún muy mejorable.

La figura 6.12 muestra varios ejemplos de algunos de los eventos que han sido detectados correctamente, mientras que la figura 6.13 muestra ejemplos de algunos casos en los que no se han detectado. En esta última figura se muestran además las máscaras del frente y los *bounding box* de los blob en cada caso, para entender mejor las causas del error. Por ejemplo, en el primer caso, la mala segmentación de la persona de la izquierda hace que su blob no sea considerado por la etapa de seguimiento de blobs, y se necesitan dos blobs para detectar el evento *Apretón de manos*. En el segundo caso, debido a que la persona se encuentra en escena desde el comienzo

Figura 6.12: Ejemplos de eventos reconocidos

de la secuencia y forma parte del fondo, no hay blob detectado que realize el evento *Escribir en teclado*. En el tercer caso, la mala segmentación sobre el teléfono provoca que no se detecte el evento *Hablar por teléfono*, y además se puede ver que la persona estaba ya en escena al comienzo de la secuencia, pues hay un fantasma en la máscara de frente. Finalmente, en el último caso no se detecta el evento *Abandono de equipaje* debido a que la persona no llega a desaparecer de escena y además el sistema no considera que se haya alejado lo suficiente, aunque realmente si lo ha hecho, pero hacia el fondo de la escena.

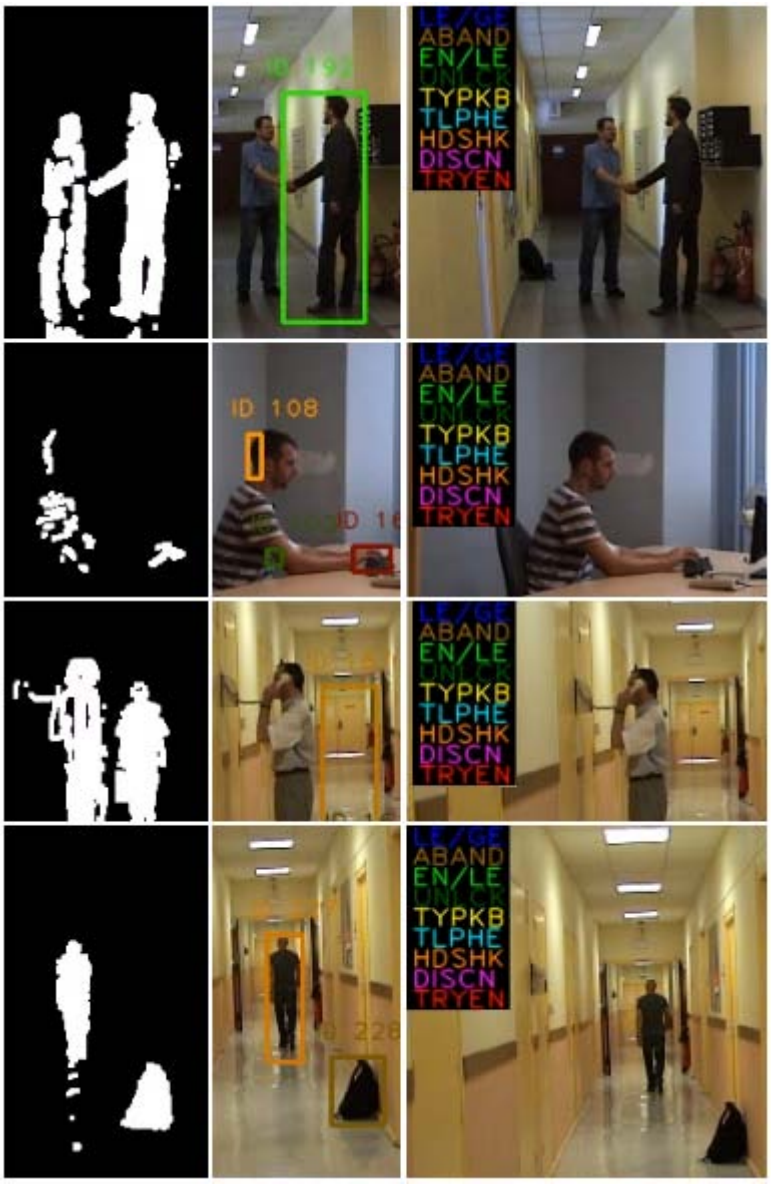


Figura 6.13: Ejemplos de eventos no reconocidos

Capítulo 7

Conclusiones y trabajo futuro

7.1. Resumen del trabajo

En este proyecto se ha llevado a cabo un estudio sobre la detección automática de eventos en entornos controlados. El objetivo principal es la mejora del prototipo actual del VPU-Lab destinado a dicha tarea, buscando en todo momento aumentar su efectividad y extender el rango de eventos que el sistema es capaz de detectar, todo ello sin dejar de ser un sistema capaz de operar en tiempo real y necesitando la menor cantidad de datos de entrenamiento posible.

Las contribuciones de este proyecto han sido las siguientes:

- **Estudio de espacios de colores típicamente usados en detección de piel y evaluación de combinaciones de espacios de colores para dicho fin.** Se ha realizado un estudio exhaustivo de los espacios de color más típicos en detección de piel, que son RGB, HSV, YCbCr y CIE-Lab. Se han extraído histogramas de cada canal de cada espacio de color para numerosas imágenes, tanto para piel anotada como para no-piel, y se han comparado entre sí. Se han buscado umbrales óptimos para la discriminación de piel sobre no-piel. Se han comparado canales de colores mediante datos como la media y la desviación de la gaussiana ajustada al histograma, la correlación entre histogramas o la distancia de Bhattacharyya. Se ha comprobado su funcionalidad sobre numerosas imágenes mediante porcentajes de falsos positivos y verdaderos positivos. Por último, se ha escogido la combinación que aparentemente mejores resultados otogra.

- **Integración de un nuevo módulo para detección de piel y comparación del mismo con otras propuestas.** Se ha integrado un nuevo módulo para detección de piel basado en el algoritmo *Random Forest* de OpenCV y se ha comparado con el método utilizado hasta el momento en el prototipo y otro algoritmo destinado a detección de sombras modificado para detección de piel. El resultado ha sido una ligera mejora en la detección de piel con el nuevo módulo *Random Forest*, a pesar de requerir un entrenamiento

previo.

- **Elaboración de un dataset de prueba para la detección de eventos en entornos muy controlados.** Un nuevo dataset ha sido elaborado representando los eventos propuestos en la competición HARL 2012. Los eventos han sido realizados de la manera más sencilla posible, en un entorno muy controlado y con una sola persona en escena, excepto en aquellos casos en los que el evento implique a dos personas.
- **Modelado de eventos a partir de máquinas de estados sencillas e implementación en el sistema.** Para cada evento propuesto se ha desarrollado un modelo basado en máquinas de estados. Se ha realizado un cambio completo del sistema en la forma de detectar los eventos integrando dichas máquinas de estados sencillas.
- **Integración de nuevos módulos al sistema para extracción de características.** Dos nuevos módulos se han integrado en el sistema: un módulo de detección de personas independiente del fondo de la secuencia y un módulo de detección de caras de perfil. La finalidad de esto es la mejora de resultados para la detección de los eventos propuestos
- **Evaluación del sistema completo sobre tres conjuntos distintos de datos.** Se ha realizado una evaluación del sistema para los tres dataset disponibles: SSG, ED y LIRIS. Los tres dataset comprenden secuencias de vídeo en entornos controlados, pero con distinta dificultad, siendo el primero el más sencillo y el último el más díficil. Los resultados para los dos primeros fueron relativamente buenos. Los resultados con el dataset de la competición inicialmente no fueron satisfactorios. Tras una serie de cambios, los resultados enviados a la organización fueron aceptables.

7.2. Conclusiones

Gracias a la evaluación del sistema sobre tres dataset distintos se han podido identificar claramente numerosos problemas en el sistema, principalmente en etapas anteriores a la detección de eventos, etapa que depende completamente del resto del sistema.

En primer lugar, en lo referente a la detección de piel, todos los métodos probados detectan relativamente bien la piel de una imagen. Ahora bien, les cuesta mucho discriminar aquellos objetos que presentan colores similares a los de la piel. Se ha comprobado como en general funcionan mejor los algoritmos basados en entrenamiento que los puramente deterministas. Aún así, queda mucho trabajo por delante hasta lograr un detector de piel completamente fiable que dependa sólo de información a nivel de píxel. La mejor manera de utilizar actualmente un detector de piel es combinarlo con un detector de personas, de manera que se extraiga solamente la piel de las personas y así evitar hacerlo de la imagen completa. Ésta es la opción elegida en este sistema, y con ella se obtienen resultados bastante aceptables.

Observando los resultados obtenidos de la comparativa de algoritmos y espacios de colores, no queda claro cuales son mejores. Esto se debe a que todos los resultados dependen de muchos factores: como estén hechas las anotaciones, que imágenes de los dataset se hayan escogido para el estudio, la alta variabilidad de los datos, la suposición de que los datos obtenidos en el estudio sean los correctos para probar los distintos métodos, etc. Por tanto, este estudio debería ser meramente orientativo, para poder basarse en algo a la hora de elegir algún método para detección de piel, pero siempre teniendo en cuenta que seguramente existan métodos que puedan resultar mejores que los propuestos.

Por otro lado, la detección de eventos no ha resultado muy satisfactoria. En primer lugar, esta etapa depende completamente de las etapas anteriores a ella. La etapa de detección del frente no funciona correctamente, y esto provoca que no se extraigan blobs precisos para tratar después en la etapa de detección de eventos. Además, esta etapa a su vez depende del módulo de eliminación de sombras y ruido, el cual a veces elimina partes que no debería de regiones en movimiento (interior de blobs principalmente). Estos problemas se arrastran a etapas posteriores, como es el seguimiento de blobs y la extracción de características.

El método propuesto basado en máquinas de estados sencillas ha resultado ser un método bastante dependiente del dataset que se esté tratando. Se consiguieron resultados buenos para los dos dataset SSG y ED con una configuración relativamente genérica del sistema, pero dicha configuración no proporcionó buenos resultados para el dataset LIRIS, que es el único que presenta un carácter algo más realista. Hubo que realizar cambios en muchos parámetros, cambios también estructurales en las máquinas de estados e incluir nuevos módulos para obtener nuevas características, todo para poder lograr un buen funcionamiento del sistema sobre este último dataset. Por tanto, se puede afirmar que el método de FSMs es sencillo de implementar y sin entrenamiento, y proporciona resultados relativamente buenos, aunque es bastante dependiente del dataset (o escenario) seleccionado.

Los resultados obtenidos en la competición HARL 2012 son, en cierto modo, los esperados, aunque algo confusos en algunos aspectos. El sistema ha quedado segundo en el ranking con unos resultados no demasiado buenos, pero que en comparación con otros participantes son aceptables. El único sistema que ha quedado por encima del sistema desarrollado en este proyecto fue uno que utiliza información de profundidad. Por tanto, como ya se sabía, la información de profundidad puede ayudar enormemente para lograr el objetivo de estos sistemas (principalmente por la mejora introducida en la etapa de segmentación de primer plano).

En conclusion, se puede decir que el sistema necesita aún numerosas mejoras en etapas anteriores a la detección de eventos. En definitiva, la dificultad de esta última etapa ha residido en crear un sistema que conviva con los posibles fallos que pueden llegar de etapas anteriores y que los subsane en la medida de lo posible, ya que si todas las etapas previas del sistema funcionasen correctamente, se trataría de una etapa relativamente sencilla de desarrollar. Aun

así, los resultados obtenidos para el sistema completo, pero aún en desarrollo, son bastante buenos respecto al estado del arte. A pesar de no haber logrado grandes mejoras, este proyecto ha servido principalmente para detectar numerosos problemas sobre los que trabajar.

7.3. Trabajo Futuro

Este proyecto ha servido enormemente para detectar los principales problemas del sistema al completo. Por tanto, se proponen las siguientes líneas de trabajo futuro:

- **Mejora de la etapa *Foreground Detection*:** En esta etapa se propone trabajar dos módulos: sustracción de fondo y eliminación de sombras/reflejos. Para el primero, se deberá incrementar la robustez frente a inicializacion y cambios de iluminación graduales. Para el segundo, se debe desarrollar un nuevo algoritmo o adaptar de modo dinámico los parámetros de funcionamiento.

- **Integración de nuevas técnicas para extracción de características en la etapa *Feature extraction*:** En esta parte se propone tres líneas de trabajo futuro: estudio de otras propuestas para detección de piel basadas en entrenamiento (y otras combinaciones de canales de color), mejora del módulo de detección de personas, detección de partes del cuerpo de una persona (y extracción de sus trayectorias) y detección de objetos determinados (teclado, teléfono, puertas, etc.).

- **Integración de aproximaciones probabilísticas para reconocimiento de eventos:** Utilizar aproximaciones basadas en entrenamiento previo al tratamiento de un vídeo. El entrenamiento permite generar modelos de los eventos propuestos para después comparar las realizaciones de cada evento presentes en un vídeo con los modelos generados, y de esta manera, discernir si realmente el evento se ha producido o no en función de la similitud de la realización del evento con el modelo.

- **Extensión del sistema a nuevos eventos:** Aunque en este proyecto se han tratado los eventos típicos en un ambiente de oficina, existen otros tambien interesantes como sentarse en una silla o escribir en una pizarra. Además, muchos otros eventos basados en interacciones con objetos pueden ser realizados a partir de los ya modelados, realizando pequeñas modificaciones sobre ellos.

- **Utilización de información de profundidad:** Muchos proyectos de investigación están comenzando a utilizar información de profundidad mediante cámaras con sensores capaces de obtener dicha información (Kinect). Con esta información, cualquier problema de confusión entre una persona y el fondo debería quedar solucionado, entre otros muchos problemas que derivan de trabajar con imágenes en dos dimensiones.

- **Utilización de información de audio:** El uso de audio puede proporcionar información muy valiosa para cierto tipo de eventos, como *Discusión* y *Escribir en teclado*, donde podría detectarse el habla de las personas implicadas en el evento o el sonido del teclado, respectivamente.

Bibliografía

[1] J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Comp. Surveys (CSUR)*, 43(3):1–43, April 2011. 1, 3, 8, 10, 11, 12, 13

[2] P. Turaga, R. Chellappa, V.S. Subrahmanian, and O. Udrea. Machine recognition of human activities: A survey. *Circuits and Systems for Video Technology, IEEE Transactions on*, 18(11):1473–1488, November 2008. 1, 3, 8, 9, 10

[3] Juan C. SanMiguel and J.M. Martinez. A semantic-based probabilistic approach for real-time video event recognition. *Computer Vision and Image Understanding*, 116(9):937–952, Sept. 2012. 1, 2, 3

[4] A. Garcia, J.C. SanMiguel, V. Fernandez-Carbajales, M.A. Garcia, and J.M. Martinez. Event detection for the trecvid 2009 video surveillance dataset: the vpulab-uam contribution. In *Technical Report, SemanticVideo.TR.2009.02*, 2009. 2, 3

[5] G. Lavee, E. Rivlin, and M. Rudzsky. Understanding video events: a survey of methods for automatic interpretation of semantic occurrences in video. *Trans. Sys. Man Cyber Part C*, 39(5):489–504, September 2009. 3

[6] ICPR. Harl, human activities recognition and localization competition (http://liris.cnrs.fr/harl2012/index.html), 2012. 4, 39

[7] M. Ryoo and J. Aggarwal. Semantic representation and recognition of continued and recursive human activities. *International Journal of Computer Vision*, 82(1):1–24, January 2009. 8

[8] A. F. Bobick, J. W. Davis, Ieee Computer Society, and Ieee Computer Society. The recognition of human movement using temporal templates. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:257–267, 2001. 11

[9] Y. Sheikh, M. Sheikh, and M. Shah. Exploring the space of a human action. In *Proceedings of the Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1 - Volume 01*, ICCV '05, pages 144–149, Washington, DC, USA, 2005. IEEE Computer Society. 15

[10] A. Garcia-Martin and J.M. Martinez. Robust real time moving people detection in surveillance scenarios. In *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*, pages 241–247, September 2010. 15, 16

[11] F. Cupillard, F. Bremond, and M. Thonnat. Tracking groups of people for video surveillance. In *2nd European Workshop on Advanced Video-based Surveillance Systems*, pages 88–100, Kingston (UK), September 4 2001. 15, 23

[12] P. Kilambi, E. Ribnick, A.J. Joshi, O. Masoud, and N.Papanikolopoulos. Estimating pedestrian counts in groups. *Computer Vision and Image Understanding*, 110(1):43–59, 2008. 15

[13] I. Haritaoglu, R. Cutler, D. Harwood, and L.S. Davis. Backpack: detection of people carrying objects using silhouettes. In *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, volume 1, pages 102–107, 1999. 16, 22

[14] S. Park and J. K. Aggarwal. Video retrieval of human interactions using model-based motion tracking and multi-layer finite state automata. In *In Lecture Notes in Computer Science: Image and Video Retrieval*, CIVR'03, pages 394–403, Berlin, Heidelberg, 2003. Springer-Verlag. 16, 23

[15] S. Park and J. Aggarwal. A hierarchical bayesian network for event recognition of human actions and interactions. In *Association For Computing Machinery Multimedia Systems Journal*, volume 10, pages 164–179, August 2004. 16, 23

[16] F. Dadgostar and A. Sarrafzadeh. An adaptive real-time skin detector based on hue thresholding: A comparison on two motion tracking methods. *Pattern Recognition Letters*, 27(12):1342–1352, September 2006. 16, 30

[17] P. Kakumanu, S. Makrogiannis, and N. Bourbakis. A survey of skin-color modeling and detection methods. *Pattern Recognition*, 40(3):1106–1122, March 2007. 17, 39, 40

[18] R. Khan, A. Hanbury, J. Stöttinger, and A. Bais. Color based skin classification. *Pattern Recognition Letters*, 33(2):157–163, January 2012. 17, 57, 85, 90

[19] D. Ayers and M. Shah. Monitoring human behavior from video taken in an office environment. *Image and Vision Computing*, 19(12):833–846, December 2001. 18, 19, 21

[20] J.C. SanMiguel, J.M. Martinez, and A. Garcia. An ontology for event detection and its application in surveillance video. In *Proc. of the IEEE Int. Conf. on Advanced Video and Signal based Surveillance*, pages 220–225, Genova (Italy), September 2009. 18, 19, 21

[21] J.C. SanMiguel, M. Escudero-Vinolo, J.M. Martinez, and J. Bescos. Real-time single-view video event recognition in controlled environments. In *Content-Based Multimedia Indexing (CBMI), 2011 9th International Workshop on*, pages 91–96, June 2011. 18

[22] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Comput. Surv.*, 38(4), December 2006. 18, 22

[23] R. Munoz-Salinas, E. Aguirre, M. Garcia-Silvente, and A. Gonzales. Door detection using computer vision and fuzzy logic. In *WSEAS Transactions on Systems*, volume 10, pages 3047–3052, December 2004. 19, 115

[24] T. Xiang and S. Gong. Video behavior profiling for anomaly detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 30(5):893–908, May 2008. 19

[25] D. Tsai and S. Lai. Independent component analysis-based background subtraction for indoor surveillance. *Image Processing, IEEE Transactions on*, 18(1):158–167, January 2009. 19, 20

[26] J.C. SanMiguel and J.M. Martinez. Robust unattended and stolen object detection by fusing simple algorithms. In *Proceedings of the 2008 IEEE Fifth International Conference on Advanced Video and Signal Based Surveillance*, pages 18–25, Washington, DC, USA, 2008. IEEE Computer Society. 19, 20

[27] S. Ferrando, G. Gera, and C. Regazzoni. Classification of unattended and stolen objects in video-surveillance system. In *Video and Signal Based Surveillance, 2006. AVSS '06. IEEE International Conference on*, page 21, november 2006. 20

[28] P. Chippendale and O. Lanz. Optimised meeting recording and annotation using real-time video analysis. In *Proceedings of the 5th international workshop on Machine Learning for Multimodal Interaction*, MLMI '08, pages 50–61, Berlin, Heidelberg, 2008. Springer-Verlag. 20, 21, 23

[29] J.-W. Hsieh, C.-L. Lin, J.-C. Cheng, P. Wu, and D.-Y. Chen. Handhold object detection and event analysis using visual interaction clues. In *17th International Conference on Distributed Multimedia Systems*, August 2011. 21

[30] N. Ghosh, B. Bhanu, and G. Denina. Continuously evolvable bayesian nets for human action analysis in videos. In *Distributed Smart Cameras, 2009. ICDSC 2009. Third ACM/IEEE International Conference*, pages 1–8, September 2009. 22

[31] A. Yao, J. Gall, and L. Van Gool. A hough transform-based voting framework for action recognition, June 2010. 23

[32] W. Waltisberg, A. Yao, J. Gall, and L. Van Gool. Variations of a hough-voting action recognition system, 2010. 23

[33] C. Wolf, J. Mille, L.E. Lombardi, O. Celiktutan, M. Jiu, M. Baccouche, E. Dellandrea, C.-E. Bichot, C. Garcia, and B. Sankur. The liris human activities dataset and the icpr 2012 human activities recognition and localization competition. Technical Report Technical Report RR-LIRIS-2012-004, LIRIS Laboratory, March 2012. 24, 39

[34] UAM) VPU-Lab (EPS. Event detection dataset (edds) (http://www-vpu.eps.uam.es/ds/edds/index.html). 39

[35] Christian Liensberger. Skin video database with ground truth (http://www.feeval.org/data-sets.html). 39

[36] Multimedia Computing Group (MCG). A benchmark human skin database (http://mcg.ict.ac.cn/mcg-skin.htm). 39

[37] L. Huang, T. Xia, Y. Zhang, and S. Lin. Human skin detection in images by mser analysis. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 1257 –1260, sept. 2011. 39

[38] J.C SanMiguel and J.M. Martinez. Shadow detection in video surveillance by maximizing agreement between independent detectors. In *Image Processing (ICIP), 2009 16th IEEE International Conference on*, pages 1141–1144, November 2009. 53, 85

[39] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, October 2001. 58, 85

[40] S. Mukherjee, S.K. Biswas, and D.P. Mukherjee. Recognizing human action at a distance in video by key poses. *Circuits and Systems for Video Technology, IEEE Transactions on*, 21(9):1228 –1241, September 2011. 97

[41] I. Laptev. On space-time interest points. *Int. J. Comput. Vision*, 64(2-3):107–123, September 2005. 97

[42] J. Yuan, Z. Liu, and Y. Wu. Discriminative subvolume search for efficient action detection. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2442 –2449, June 2009. 97

[43] Visual door detection integrating appearance and shape cues. *Robotics and Autonomous Systems*, 56(6):512–521, June 2008. 115

[44] Z. Chen and S. T. Birchfield. Visual detection of lintel-occluded doors from a single image. In *Computer Vision and Pattern Recognition Workshops, 2008. CVPRW '08. IEEE Computer Society Conference on*, pages 1–8, June 2008. 115

[45] X. Yang and Y. Tian. Robust door detection in unfamiliar environments by combining edge and corner features. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 57–64, June 2010. 115, 116

[46] C. Juenemann, A. Corbin, and J. Li. Robust door detection, from, 2010. 115, 116, 117

Apéndice A

Detección de puertas

La detección de puertas es uno de los campos más tratados en el área de la detección automática de objetos, pudiéndose encontrar muchos documentos que tratan este tema. En [23] buscan en primer lugar todas las líneas rectas del escenario mediante extracción de bordes con el algoritmo de Canny, y con esas rectas buscan formas similares a una puerta con un mínimo de tres segmentos. Hay técnicas más complejas, como las utilizadas en [43, 44], que además de los bordes, utilizan también información de color, de silueta, de textura, además de entrenamiento previo a la detección. Además de buscar rectas, también se pueden extraer las esquinas del escenario, como en [45], donde buscan cuatro esquinas unidas por rectas. Un ejemplo de detección de puertas perteneciente al último estudio mencionado se presenta en la figura A.1.

Estas técnicas de extracción de puertas aún no son muy efectivas. Detectar una puerta de un escenario real es muy complicado y depende de muchos factores del escenario. De hecho, en este proyecto se han realizado algunas pruebas con un algoritmo disponible en la red, correspondiente al trabajo presentado en [46]. Este algoritmo se desarrolla principalmente en tres etapas: obtención de la región de interés (ROI), extracción de las líneas de la puerta y clasificación de una región como puerta.

En la imagen A.2 se puede observar que los resultados del algoritmo son aparentemente muy buenos. Según se describe en el documento, el algoritmo logra una tasa media de detección del 80 % con un retardo medio de 7 segundos para imágenes de 750x550 píxeles. En la imágen A.3 hay algunos ejemplos de estas detecciones de puerta, donde éstas aparecen en azul dentro de un marco rojo.

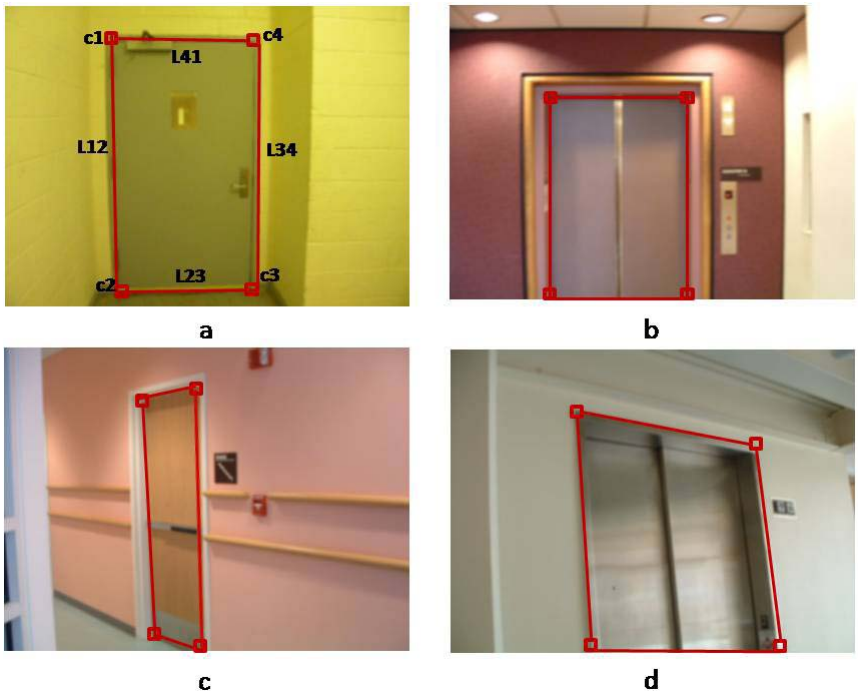


Figura A.1: Ejemplo de detección de puertas
Figura extraída de [45]

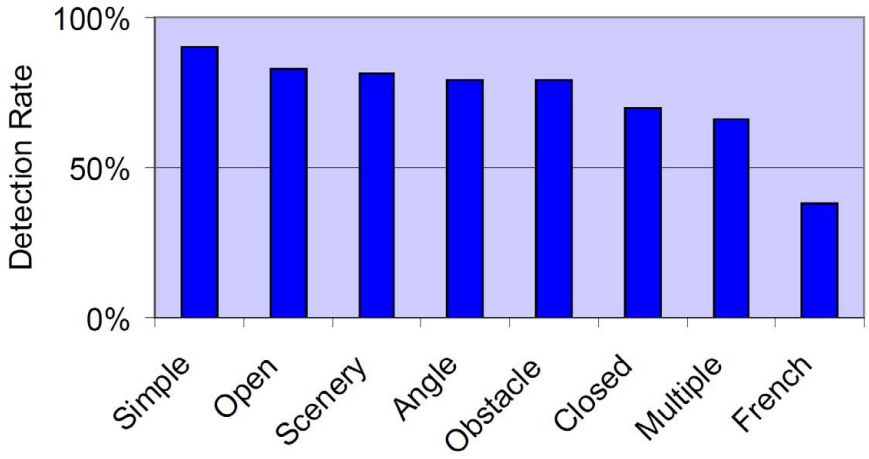


Figura A.2: Resultados
Figura extraída de [46]

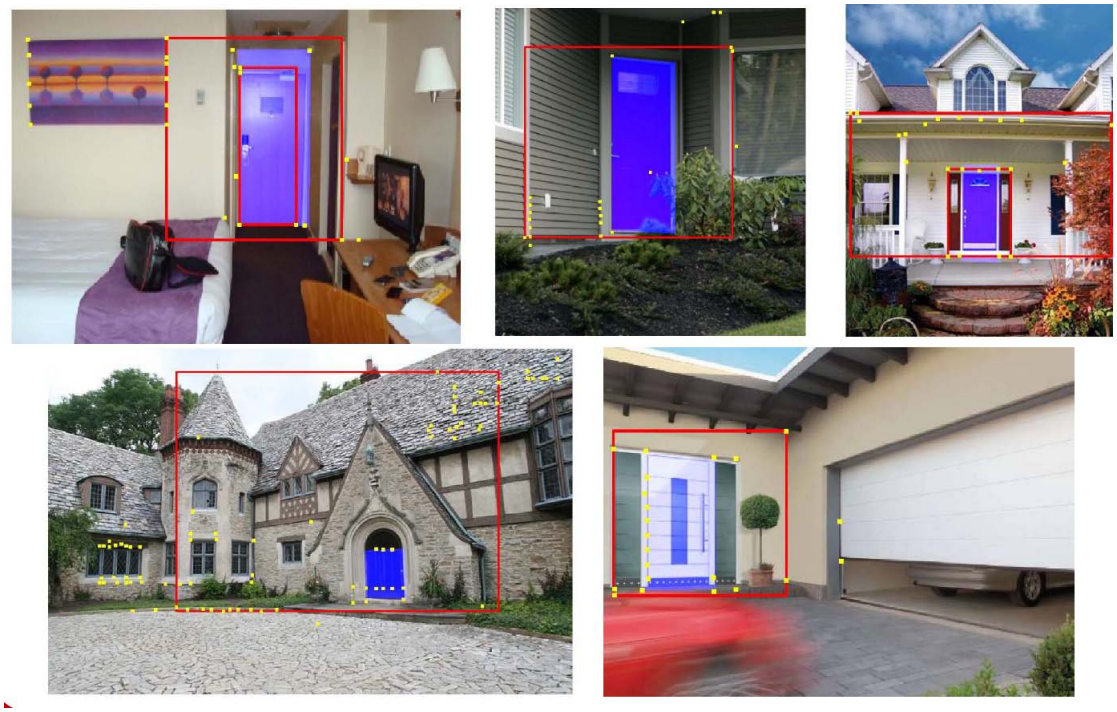


Figura A.3: Ejemplos de detecciones de puerta
Figura extraída de [46]

A la vista de estos resultados, se decidió probar el algoritmo con varias imágenes de los dataset SSG, ED y LIRIS. Para cada dataset, se trataron de escoger imágenes que representaran todos los distintos tipos de puertas presentes en los tres dataset. Los resultados obtenidos se presentan en las siguientes imágenes: A.4, A.5, A.6. Como se puede observar, para el dataset SSG, se detectan bien 2 de las 3 puertas diferentes que hay. Para el dataset ED, sólo hay 2 puertas diferentes, y no se detecta ninguna. Por último, en el dataset LIRIS se pueden encontrar 10 tipos diferentes de puertas, de las cuales, sólo 3 se detecta medianamente bien.

Como se puede observar por los resultados, el algoritmo no funciona para los dataset utilizados en este proyecto, lo cual es sorprendente a la vista de los resultados obtenidos en el documento [46]. Además, no detecta nada en algunos casos en los que la puerta se ve completamente y de manera muy clara.

Figura A.4: Ejemplos de detección de puertas sobre el dataset SSG

Figura A.5: Ejemplos de detección de puertas sobre el dataset ED

Figura A.6: Ejemplos de detección de puertas sobre el dataset LIRIS

Apéndice B

Dataset SSG

El dataset SSG fue grabado especialmente para este proyecto. La finalidad de este dataset es representar de la manera más simple posible los eventos propuestos por la competición HARL 2012, intentando evitar todos aquellos detalles que dificultan enormemente la detección de eventos en el dataset LIRIS.

Este dataset fue grabado en la sala de reuniones del laboratorio del Grupo de Tratamiento e Interpretación de Vídeo de la Universidad Autónoma de Madrid (VPU-UAM). Los vídeos están protagonizados por un sólo actor, excepto para aquellos eventos que requieren interacción entre dos personas, pero siempre buscando la mayor simplicidad posible. Todos los eventos se realizan de cara a la cámara, tratando de evitar siempre cualquier tipo de oclusión del evento. En cada secuencia se realizan como máximo 2 eventos distintos y nunca a la vez, y puede haber varias realizaciones del mismo evento consecutivas.

El dataset fue realizado durante la primera fase del proyecto, de ahí que los eventos incluídos son los escogidos en dicha fase del proyecto, sin incluir los eventos DI y ET (véase la tabla B.1)

El dataset cuenta con 51 secuencias de vídeos grabados con una cámara estática, agrupadas para cada uno de los eventos. Las secuencias tienen una resolución de 1920x1080 píxeles a 25 fps. En la figura B.1 se muestran algunas capturas de este dataset representando varios eventos.

Ocurrencia de Eventos						
BO	EN	LO	UB	HS	KB	TE
32	3	9	13	3	8	10

Tabla B.1: Ocurrencias de eventos en SSG

Figura B.1: Ejemplos del dataset SSG

Apéndice C

Implementación mediante *Finite State Machines* (FSMs)

La técnica escogida para realizar la implementación de todos los eventos en el prototipo del VPU-Lab fueron las *Finite State Machines* (FSMs), por su simplicidad a la hora de implementarlas y de trabajar con ellas. En la figura C.1 se muestra un diagrama del funcionamiento de las máquinas de estados implementadas en el programa.

Una FSM se crea al aparecer un nuevo blob en el frente y dura el tiempo que dicho blob permanezca identificado con un ID. Si el blob cambia de ID por cualquier motivo, se creará una nueva FSM para él. Para cada blob se creará una FSMs para cada uno de los eventos en los que dicho blob pueda estar implicado, lo cual depende de la naturaleza del blob. La naturaleza del blob puede ser la de un blob aparecido en el frente repentinamente o la de un objeto contextual, que a su vez puede ser de varios tipos, lo cual condiciona los eventos que puede realizar.

Por tanto, una FSM tiene dos parámetros esenciales que la definen: *candidateFSM* (candidato) y *eventName* (nombre del evento). El parámetro *candidateFSM* indica la ID del blob que realiza el evento, llamado candidato, para el cual se comprobará el evento indicado por *eventName,* que define dicha FSM. En cada frame, el sistema comprobará a partir de las FSMs cada uno de los eventos para cada candidato. Como se ha introducido antes, un candidato no puede realizar todos los eventos existentes, sino que depende de su naturaleza. Para ciertos eventos, por simplicidad, se consideró que el candidato debía ser el objeto contextual implicado, como sucede, por ejemplo, con los eventos que implican puertas, donde el blob principal sobre el que se extraen las características es el relativo a la puerta, que no aparece en el frente y por tanto no se visualiza nunca, lo cual no quiere decir que no esté ahí. Por tanto, algunos eventos sólo pueden ser realizados por un objeto contextual, mientras que otros, sólo pueden realizarlos blos aparecidos en el frente, sean objetos o personas.

Otros dos parámetros muy importantes en una FSM son el *currentState*, que indica en que

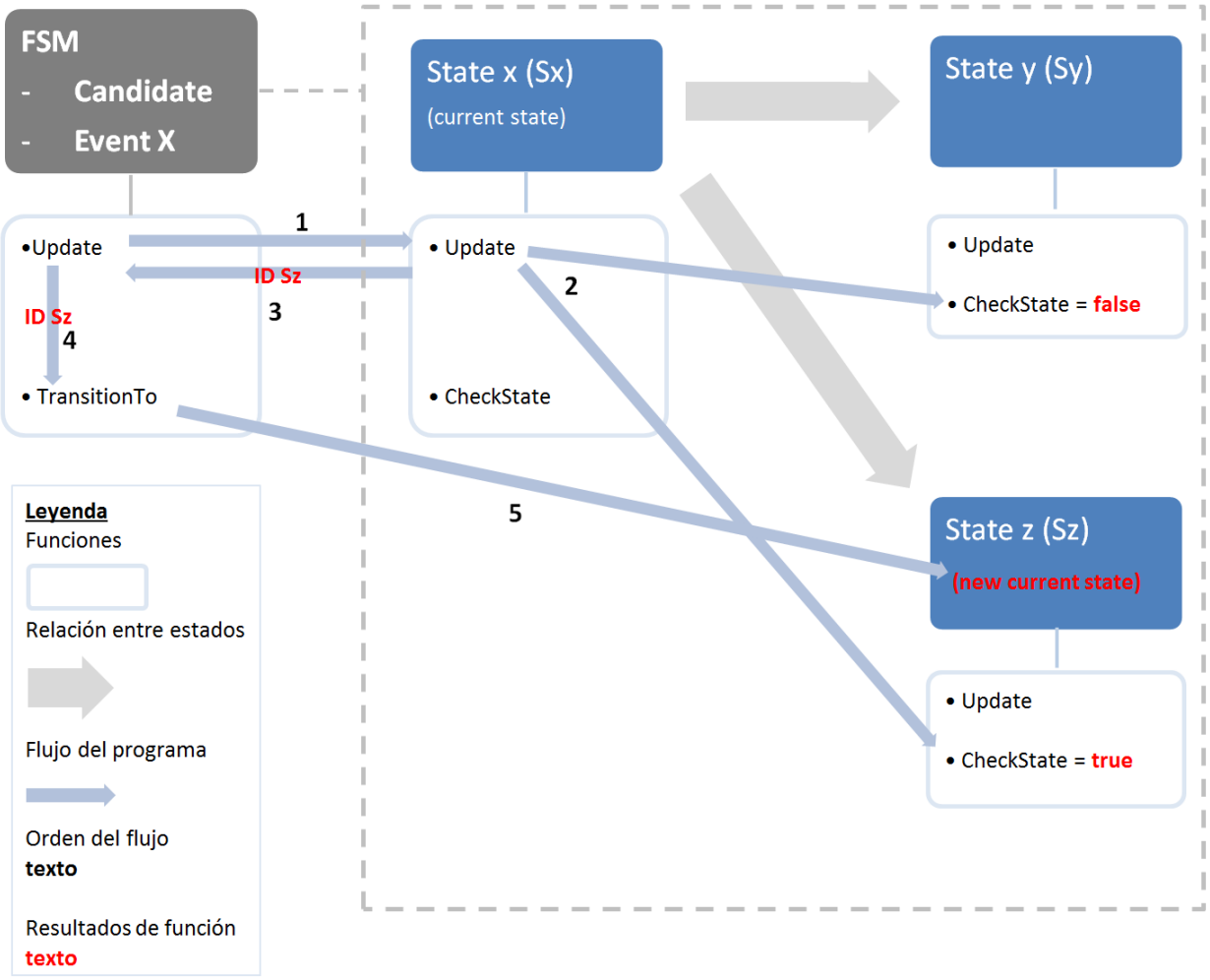


Figura C.1: Esquema de funcionamiento de una FSMs en el prototipo VPU-Lab

estado se encuentra dicha FSM, es decir, el estado actual, y *stateRelations*, que es una matriz que indica las relaciones entre estados de la FSM (ejemplo en figura C.2).

En la creación de una FSM, para cada tipo de evento se definirán unos estados diferentes. Mediante la función *AddState* se añade un estado a una FSM, y mediante la función *DefineRelation* se establece una relación entre dos estados, rellenando el campo correspondiente de la matriz *stateRelations*. Además, se actualizará por primera vez el parámetro *currentState*, tomando como valor un puntero al primer estado, que es común a todas las FSM y se llama *Init* (de *Initial State*, estado inicial). De esta manera, se puede construir automáticamente y de manera sencilla una FSM para cada candidato y evento.

Cada estado tiene un nombre, formado por el nombre del evento y el número de estado (ejemplo: UB_S1), y un número que lo identifica (ID). Las principales funciones de un estado son: *Update* y *CheckEvent* (véase la figura C.1). La funcion *Update* recorre la fila de la matriz

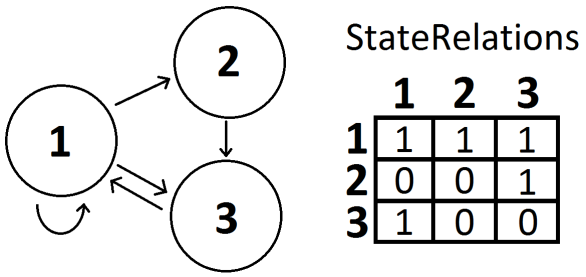


Figura C.2: Ejemplo de matriz *stateRelations*

stateRelations correspondiente al estado actual de la FSM. Para todos los estados con los que este relacionado, llama a la función *checkState* de dichos estados. La función *checkState*, que siempre será llamada desde un estado anterior al que la realiza, es la encargada de comprobar las condiciones que se tienen que dar para que la FSM salte a ese estado. De esta forma, si algún estado relacionado con el actual cumple las condiciones de salto, devolverá un valor positivo (*true*) desde su función *checkState*, y la función *Update* del estado actual devolverá a la FSM el identificador ID del estado que le devolvió *true*.

Volviendo a las FSM, su funcionamiento parte de dos funciones: *Update* y *TransitionTo*. En cada frame, para cada candidato y para cada FSM el sistema llamará a la funcion *Update* de dicha FSM, que es la encargada de llamar a su vez a la función *Update* del *currentState*. Como se ha explicado antes, puede darse el caso de que la función *Update* el estado actual devuelva un ID de un estado al que está relacionado. Si esto ocurre, la función *Update* de la FSM recogerá dicha ID y llamará a la función *TransitionTo*. Esta función simplemente actualiza el *currentState* y lo sitúa en el nuevo estado indicado por el ID que se ha obtenido de la función *Update*, produciéndose así el salto de estado.

Por último, los estados tienen una función más, llamada *DoEXIT*, que sólo puede ser llamada durante la ejecución de *TransitionTo* de la FSM. Esta función hace que el estado actual, al cual se habrá transicionado, compruebe si se trata de un estado final de la FSM de algún evento. Según lo sea o no, devolverá un valor llamado *score* como 1 ó 0 respectivamente. Este *score* es el único valor que devuelve la función *Update* de la FSM, y será el valor que indique si el evento ha sido reconocido o no.

Si el evento se ha reconocido, otro bloque del sistema se encargará de crear una nueva *bounding box* para el candidato que provocó dicho evento, con un color característico de dicho evento, de manera que el usuario pueda ver exactamente donde se ha producido el evento. En la figura C.3 se muestra un ejemplo de esto.

Figura C.3: Ejemplo de reconocimiento de eventos
(Amarillo = KB, Verde = EN, Azul = BO, Cyan = TE)

Apéndice D

Resultados completos de detección de piel

Para cada método se presentan los resultados obtenidos para cada dataset y para cada combinación de canales de color que se han utilizado. En cada caso, se muestran los valores de *TPR* y *FPR*, el valor combinación de ellos *Rend* y la media del rendimiento de todos los dataset.

D.1. Estado del Arte (SoA, *State of Art*)

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	30	1	65	61	7	77	50	3	74	72
Cb-Cr	72	10	81	91	50	71	89	16	87	80

Tabla D.1: Umbralización con valores por defecto (U-SoA)

ED			LIRIS			SSG			
TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
73	21	76	77	58	60	90	22	84	73

Tabla D.2: *Adaptive Skin Detector* de OpenCV (ASD-SoA)

	ED			LIRIS			SSG			
Dataset train	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
1 (=imgs test)	78	1	89	91	11	90	89	7	91	90
1 (≠imgs test)	60	1	80	88	10	89	89	7	91	87
2 (DS test + otro)*	61	1	80	89	14	88	86	7	90	86
3 (ED+LIRIS+SSG)	61	1	80	89	18	86	85	6	90	85
4 (3+SKIN)	57	1	78	89	16	87	82	5	89	85
5 (4+MCG)	53	1	76	86	12	87	75	4	86	83

Tabla D.3: RF con HSV (RF-SoA)
**Train* con mismo dataset para *test* (distintas imágenes) y otro distinto (2 dataset)

D.2. Otros algoritmos

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	56	1	78	72	12	80	80	5	88	82
H-a	47	1	73	55	8	74	75	4	86	78

Tabla D.4: Umbralización con umbrales escogidos del estudio (U)

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	73	2	86	89	13	88	87	6	91	88
H-a	73	2	86	88	11	89	85	6	90	88
a-b	72	2	85	89	11	89	84	5	90	88
a-S	72	2	85	8	10	88	86	6	90	88
Cb-Cr	72	2	85	88	10	89	84	6	89	88
H-a-Cr	70	2	84	90	11	90	86	6	90	88
a-S-Cr	76	2	87	91	12	90	87	6	91	89
H-a-Cb	78	2	88	90	11	90	88	7	91	90

Tabla D.5: RF train/test iguales

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	67	1	83	86	12	87	87	6	91	87
H-a	60	1	80	87	9	89	86	7	90	86
a-b	59	1	79	88	10	89	85	6	90	86
a-S	59	1	79	85	12	87	86	6	90	85
Cb-Cr	58	1	79	88	9	90	86	6	90	86
H-a-Cr	58	1	79	86	9	89	85	6	90	86
a-S-Cr	62	1	81	88	10	89	88	6	91	87
H-a-Cb	61	2	80	87	9	89	86	6	90	86

Tabla D.6: RF train/test distintos del mismo dataset

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	58	0	79	90	14	88	83	6	89	85
H-a	50	1	75	93	13	90	81	5	88	84
a-b	49	1	74	92	13	90	82	6	88	84
a-S	51	1	75	90	16	87	81	6	88	83
Cb-Cr	49	1	74	92	14	89	82	5	89	84
H-a-Cr	50	1	75	91	13	89	-	-	-	82
a-S-Cr	61	1	80	90	15	88	-	-	-	84
H-a-Cb	53	1	76	92	14	89	-	-	-	83

Tabla D.7: RF en 2D y *train* mezcla de 2 datasets

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	61	1	80	91	16	88	77	5	86	85
H-a	57	1	78	94	15	90	77	5	86	85
a-b	57	1	78	94	16	89	76	5	86	84
a-S	56	1	78	92	17	88	76	5	86	84
Cb-Cr	57	1	78	94	17	89	80	5	88	85
H-a-Cr	52	1	76	90	15	88	79	5	87	84
a-S-Cr	61	1	80	90	18	86	82	5	89	85
H-a-Cb	57	1	78	93	18	88	81	5	88	85

Tabla D.8: RF en 2D y *train* mezcla de 3 datasets

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	63	1	81	92	18	87	77	5	86	85
H-a	54	1	77	93	14	90	75	4	86	84
a-b	55	1	77	94	15	90	76	5	86	84
a-S	51	1	77	91	16	88	72	5	84	83
Cb-Cr	53	1	76	93	16	89	77	5	86	84
H-a-Cr	-	-	-	-	-	-	-	-	-	-
a-S-Cr	-	-	-	-	-	-	-	-	-	-
H-a-Cb	-	-	-	-	-	-	-	-	-	-

Tabla D.9: RF en 2D y *train* mezcla de 4 datasets

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	58	1	79	89	15	87	73	4	85	84
H-a	50	1	75	91	12	90	69	4	83	83
a-b	51	6	73	92	13	90	71	4	84	82
a-S	45	7	69	88	13	88	67	4	82	80
Cb-Cr	46	1	73	91	14	89	71	4	84	82
H-a-Cr	-	-	-	-	-	-	-	-	-	-
a-S-Cr	-	-	-	-	-	-	-	-	-	-
H-a-Cb	-	-	-	-	-	-	-	-	-	-

Tabla D.10: RF en 2D y *train* mezcla de 5 datasets

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	74	1	87	83	12	86	83	4	90	88
H-a	74	1	87	82	11	86	83	4	90	88
a-b	89	3	93	86	12	87	93	6	94	91
a-S	92	3	95	88	15	87	91	6	93	92
H-b	73	2	86	83	13	85	83	4	90	87
Cb-Cr	82	5	89	88	16	86	91	5	93	89

Tabla D.11: SDMIM con búsqueda de umbrales óptimos

	ED			LIRIS			SSG			
	TPR	FPR	Rend	TPR	FPR	Rend	TPR	FPR	Rend	MEDIA
H-S	55	1	77	76	19	79	64	3	81	79
H-a	72	2	85	81	19	81	83	6	89	85
a-b	70	5	83	77	17	80	83	8	88	84
a-S	84	4	90	76	23	77	77	6	86	84
H-b	66	2	82	75	16	80	78	6	86	83
Cb-Cr	88	59	65	88	39	75	79	20	80	73

Tabla D.12: SDMIM búsqueda normal

Apéndice E

Competición ICPR - HARL 2012

E.1. Participantes

70 equipos fueron inscritos en la competición, de los cuales, sólo 4 participantes (tabla E.1) presentaron resultados.

Autores y equipo afiliado	Siglas	Dataset	Loc. provista
Bingbing Ni and Yong Pei Advanced Digital Sciences Center, Singapore Jun Tan, Jian Dong and Shuicheng Yan National University of Singapore, Singapore Pierre Moulin University of Illinois at Urbana-Champaign	ADSC-NUS-UIUC	D1	Yes
Dr. Tanushyam Chattopadhyay, Sangheeta Roy and Aniruddha Sinha Innovation Lab, Tata Consultancy Services, Kolkata Prof. Dipti Prasad Mukherjee and Apurbaa Mallik Indian Statistical Institute, Kolkata	TATA-ISI	D1	No
Juan C. SanMiguel and Sergio Suja **Video Processing and Understanding Lab** **Universidad Autonoma of Madrid, Spain**	**VPULABUAM**	**D2**	**Yes**
Yonghao He, Hao Liu, Wei Sui, Shiming Xiang and Chunhong Pan Institute of Automation Chinese Academy of Sciences, Beijing	IACAS	D2	Yes

Tabla E.1: Participantes de HARL 2012

E.2. Evaluación y métricas

E.2.1. Métrica utilizada

El objetivo de la evaluación es medir el emparejamiento entre el *groundtruth* anotado y el resultado obtenido del sistema evaluado, es decir, entre:

- Una lista G de acciones anotadas (*groundtruth*) $G^{v,a}$, donde $G^{v,a}$ corresponde a la a-ésima acción en el v-ésimo vídeo y donde cada acción queda representada mediante un *bounding box* $G_b^{v,a}$, marcado con la misma clase de la acción.
- Una lista G de acciones detectadas (resultado) $D^{v,a}$, donde $D^{v,a}$ corresponde a la a-ésima acción en el v-ésimo vídeo y donde cada acción queda representada mediante un *bounding box* $D_b^{v,a}$, marcado con la misma clase de la acción.

El objetivo es medir el grado de similitud entre las dos listas. La medida debería penalizar la pérdida de información, que ocurre si una acción o partes (espaciales o temporales) de una acción no han sido detectadas, y debería penalizar la aparición de información confusa, como falsas detecciones o detecciones demasiado largas (espacial o temporalmente). Esta métrica pretende cumplir los siguientes objetivos:

1) La métrica debe proveer una evaluación cuantitativa: debe representar de manera intuitiva cuantas acciones han sido detectadas correctamente y cuantas falsas alarmas han aparecido.

2) La métrica debe proveer una evaluación cualitativa: debe dar una interpretación fácil de la calidad de las detecciones.

Ambos objetivos están relacionados: el número de acciones que se consideran detectadas depende de la los requerimientos de calidad que se imponen para considerar una acción como detectada. Debido a esto, se propone una manera de combinar estos dos objetivos:

1) Proveer valores tradicionales de *precision* y *recall* que den una evaluación cuantitativa. Una acción se considera que es detectada correctamente o no mediante dos umbrales fijados a partir de la cantidad de solapamiento entre la acción anotada (*groundtruth*) y la acción detectada.

2) Completar la métrica con gráficas que ilustren la dependencia entre cantidad y calidad. Estas gráficas describirán el comportamiento del algoritmo de detección usado.

La primera medida, *Recall*, describe cuántas ocurrencias de un evento han sido detectadas correctamente, respecto al número total de ocurrencias en el dataset. La segunda medida, *Precision*, describe cuantos falsos positivos produce el sistema, o dicho de otra forma, cuantas acciones

han sido detectadas correctamente respecto al total de acciones detectadas, tanto buenas como malas.

$$Recall(G, D) = \frac{Number\ of\ correctly\ found\ actions}{Number\ of\ actions\ in\ the\ groundtruth} \tag{E.1}$$

$$Precision(G, D) = \frac{Number\ of\ correctly\ found\ actions}{Number\ of\ found\ actions} \tag{E.2}$$

Ahora bien, esta definicion depende del criterio impuesto para considerar una acción como correctamente detectada. ¿Cómo de parecidas deben ser las *bounding boxes* detectadas a las del *groundtruth*? ¿Cómo de parecida debe ser la duración de la acción detectada respecto a la del *groundthruth*? ¿Como tratar una acción multiple detectada respecto a una acción simple en el *groundtruth* y vice versa? La manera más intuitiva de decidir estas cosas se presentan en la siguiente definición:

$$\begin{aligned} Recall(G, D) &= \frac{\sum_v \sum_a IsMatched(G^{v,a}, BestMatch(G^{v,a}, D^v))}{\sum_v |G^v|} \\ Precision(G, D) &= \frac{\sum_v \sum_a IsMatched(BestMatch(D^{v,a}, G^v), D^{v,a})}{\sum_v |D^v|} \end{aligned} \tag{E.3}$$

Ambas medidas confían en encontrar la acción de una lista que mejor encaje con la acción dada; *Recall* empareja cada acción del *groundtruth* con una acción de la lista de detectadas, mientras que *Precision* empareja cada acción de la lista de detectadas con una de las acciones de la lista de *groundtruth*. Esto se hace en dos pasos:

- Para una acción simple, la función $BestMatch$ encuentra el mejor emparejamiento con una acción de la otra lista (detecciones o *groundtruth*).
- Para un par de acciones (detección o *groundtruth*), $IsMatched$ decide si esta correspondencia satisface el criterio de solape espacial y temporal.

La función $BestMatch$ encuentra el mejor emparejamiento para una acción en una lista de potenciales candidatos para el emparejamiento. Maximiza el solape de dos acciones normalizado sobre todos los frames:

$$BestMatch(X^{v,a}, Y^v) = arg \max_{a'=1...\|Y^v\|} \frac{2 \cdot Area(X^{v,a} \cap Y^{v,a'})}{Area(X^{v,a}) + Area(Y^{v,a'})} \tag{E.4}$$

La funcion $IsMatched$ decide si una acción emparejada está lo suficientemente emparejada basandose en cuatro criterios: dos de carácter espacial y dos de carácter temporal. Los criterios que una acción detectada debe cumplir para ser emparejada con una acción del *groundtruth* son:

- Ambas deben ser el mismo tipo de acción, y

- El tamaño del solape entre las *bounding boxes* debe ser suficientemente grande respecto al tamaño de los bounding boxes del set de *groundtruth*, es decir, debe encontrarse una porción suficientemente grande del rectángulo correspondiente al *groundtruth*. Para ignorar las diferencias temporales, el cálculo de solape se realiza sólo entre frames que son parte de ambas acciones, detectada y *groundtruth*.

- El tamaño del solape entre las *bounding boxes* debe ser suficientemente grande respecto al tamaño de los bounding boxes del set de detectadas, es decir, el espacio detectado en exceso es suficientemente pequeño. Para ignorar las diferencias temporales, el cálculo de solape se realiza sólo entre frames que son parte de ambas acciones, detectada y *groundtruth*.

- El número de frames que forman parte de ambas acciones es suficientemente grande respecto al numero de frames del set de *groundtruth*, es decir, se encuentra la accion un tiempo suficientemente largo.

- El número de frames que forman parte de ambas acciones es suficientemente grande respecto al numero de frames del set de detectadas, es decir, el tiempo en exceso que es detectada la acción es suficientemente pequeño.

Para dar una expresión formal de esto, se abreviarán los términos de la siguiente forma. Una acción del *groundtruth* será $g = G^{v,a}$ y una acción detectadas será $d = D^{v,a'}$. Además, se denota como $g\mid_d$ al set de bounding boxes de la acción del *groundtruth* g restringido únicamente a los frames que también forman parte de la accion detectada d. Del mismo modo, se denota como $d\mid_g$ al set de bounding boxes de la acción detectada d restringido únicamente a los frames que también forman parte de la accion del *groundruth* g. Con esto, la expresión queda así:

$$IsMatched(g,d) = \begin{cases} 1 & if \quad \begin{array}{rl} \frac{Area(g\cap d)}{Area(g|_d)} > t_{rs} & and \\ \frac{Area(g\cap d)}{Area(d|_g)} > t_{ps} & and \\ \frac{NoFrames(g\cap d)}{NoFrames(g)} > t_{rt} & and \\ \frac{NoFrames(g\cap d)}{NoFrames(d)} > t_{pt} & and \\ Class(g) = Class(d) & \end{array} \\ 0 & else \end{cases} \tag{E.5}$$

donde $Area(X)$ is la suma de las áreas de las *bounding boxes* del set X y $\cap$ es la operación de intersección que retorna el solape entre dos *bounding box*. $NoFrames(X)$ es el número de frames en el set X.

E.2.2. Ranking

Para obtener una sóla medida, *Recall* y *Precision* se combinan en el tradicional *F-Score* (o media armónica). La ventaja de esto es que el mínimo de los dos valores de rendimiento se enfatiza:

$$F = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall} \tag{E.6}$$

Sin embargo, esta medida sigue dependiendo de los criterios de calidad escogidos, es decir, de los umbrales $t_{rs}, t_{ps}, t_{rt}, t_{pt}$. La medida final que se utiliza integra el rendimiento en todo el rango del criterio de emparejamiento, variando esas constantes. Se crean cuatro medidas, cada una midiendo el rendimiento mientras uno de los umbrales varía y el resto se mantienen fijos en un valor muy bajo ($\epsilon = 0{,}1$). Denotando el *F-Score* de la ecuación E.6 como $F(t_{rs}, t_{ps}, t_{rt}, t_{pt})$, tenemos:

$$\begin{aligned} I_{rs} &= \tfrac{1}{N} F(t_{rs}, \epsilon, \epsilon, \epsilon) \\ I_{ps} &= \tfrac{1}{N} F(\epsilon, t_{ps}, \epsilon, \epsilon) \\ I_{rt} &= \tfrac{1}{N} F(\epsilon, \epsilon, t_{rt}, \epsilon) \\ I_{pt} &= \tfrac{1}{N} F(\epsilon, \epsilon, \epsilon, t_{pt}) \end{aligned} \tag{E.7}$$

donde N es el número de muestras de la integración. El valor usado para el ranking es la media de esos cuatro valores:

$$IntegratedPerfomance = \frac{1}{4}(I_{rs}, I_{ps}, I_{rt}, I_{pt}) \tag{E.8}$$

E.2.3. Rendimiento vs. curvas de calidad

Las medidas se complementan con gráficos visuales que ilustran la dependiencia entre cantidad (*precision* y *recall*) y calidad (requerimientos establecidos de calidad). Los gráficos se obtienen de las medidas integrales obtenidas con la ecuación E.8 y describen el comportamiento del algoritmo de detección.

Cada diagrama representa los valores de *Precision*, *Recall* y *F-Score* sobre un umbral variable. Uno de los umbrales rt, pt, rs, ps varía sobre el eje x, mientras que el resto permanecen fijos en 10 %.

E.2.4. Matrices de confusión

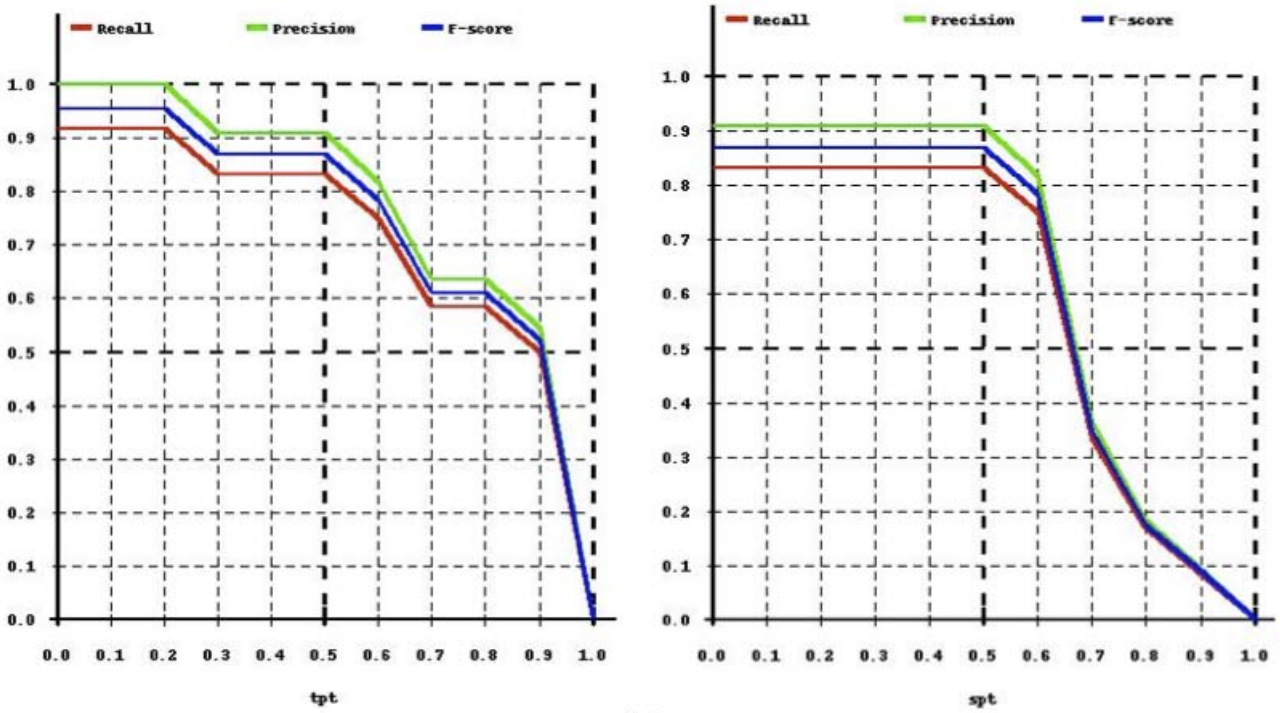
Las matrices de confusión ilustran el rendimiento de clasificación pura. Puede ser obtenido fácilmente asociando una acción detectada al rectángulo de cada acción del *groundtruth* usando las ecuaciones E.3 y E.4, al retirar la restricción de igualdad de clase de E.4.

Performance at fixed values	
measure	value
recall	0.833
precision	0.909
fscore	0.869
Performance on Fscore (with single threshold variation)	
threshold	value
trt	0.696
tpt	0.727
srt	0.791
spt	0.601
Combined performance	
0.70375	

(a)

	1	2	3	4	5	6	7	8	9	10
1	**66**	1	1	1	2	5	19	3	1	1
2	0	**92**	1	0	1	0	2	1	3	0
3	2	0	**78**	0	15	0	1	0	4	0
4	0	0	0	**100**	0	0	0	0	0	0
5	2	1	11	1	**64**	0	11	2	8	0
6	9	0	0	0	1	**88**	1	0	0	1
7	3	3	7	0	11	1	**63**	3	2	7
8	3	13	2	0	7	0	7	**64**	2	2
9	5	9	6	2	14	0	14	13	**34**	3
10	11	0	0	1	9	4	14	1	3	**57**

(b)



(c)

Figura E.1: Ejemplo de los gráficos obtenidos por la organización de HARL 2012

E.3. Resultados completos

E.3.1. Rendimiento de detección pura y reconocimiento - sin localización

La primera medida del rendimiento ignora la información de localización espacial y temporal del resultado (*bounding box* y número de *frame*) y sólo da información del rendimiento de la detección pura y reconocimiento a través de las medidas. Los resultados se muestran en la tabla E.2.

Equipo	Dataset	Recall	Precision	F-Score
ADSC-NUS-UIUC	D1	0.74	0.41	0.53
TATA-ISI	D1	0.08	0.17	0.11
VPULABUAM	**D2**	**0.36**	**0.66**	**0.46**
IACAS	D2	0.30	0.46	0.36

Tabla E.2: Rendimiento sin localización

E.3.2. Rendimiento de detección, reconocimiento y localización a un nivel de calidad del 10 %

Teniendo en cuenta la localización, determinar si una acción se detecta correctamente requiere establecer umbrales referentes a la cantidad de solape entre la acción definida en el *groundtruth* y la detectada. En la siguiente tabla aparecen los valores de *Precision* y *Recall* para unos umbrales del 10 %, lo cual quiere decir que una acción detectada se corresponde a una acción del *groundtruth* si:

- al menos un 10 % del tiempo que dura la acción del *groundtruth* se encuentra correctamente (umbral rt);
- al menos un 10 % del tiempo de la acción detectada se cubre con el *groundtruth* (umbral pt);
- al menos un 10 % del bounding box del *groundtruth* se cubre con el de la acción detectada (umbral rs);
- al menos un 10 % del bounding box de la acción detectada se cubre con el del *groundtruth* (umbral ps);

Los resultados se muestran en la tabla E.3.

Equipo	Dataset	Recall	Precision	F-Score
ADSC-NUS-UIUC	D1	0.63	0.33	0.44
TATA-ISI	D1	N/A	N/A	N/A
VPULABUAM	**D2**	**0.04**	**0.08**	**0.05**
IACAS	D2	0.03	0.04	0.03

Tabla E.3: Rendimiento con localización

E.3.3. Rendimiento de detección, reconocimiento y localización: rendimiento integrado

Esta última medida integra el rendimiento obtenido anteriormente en el rango completo de valores de los umbrales. Los cuatro umbrales de calidad rt, pt, rs, ps se cambian y el *F-Score* es integrado en el intervalo de posibles valores. De esta manera, este valor no dependerá de los umbrales escogidos anteriormente. Los resultados se muestran en la tabla E.4.

Equipo	Dataset	Rec-T	Pre-T	Rec-S	Pre-S	Total
ADSC-NUS-UIUC	D1	0.27	0.37	0.29	0.37	0.33
TATA-ISI	D1	N/A	N/A	N/A	N/A	N/A
VPULABUAM	**D2**	**0.03**	**0.03**	**0.02**	**0.03**	**0.03**
IACAS	D2	0.03	0.00	0.01	0.01	0.02

Tabla E.4: Rendimiento integrado

E.3.4. Rendimiento vs. curvas de calidad

Aquí se presenta el rendimiento frente a las curvas de calidad. Cada diagrama representa los valores de *Precision*, *Recall* y *F-Score* sobre un umbral variable. Uno de los umbrales rt, pt, rs, ps varía sobre el eje x, mientras que el resto permanecen fijos en 10 %.

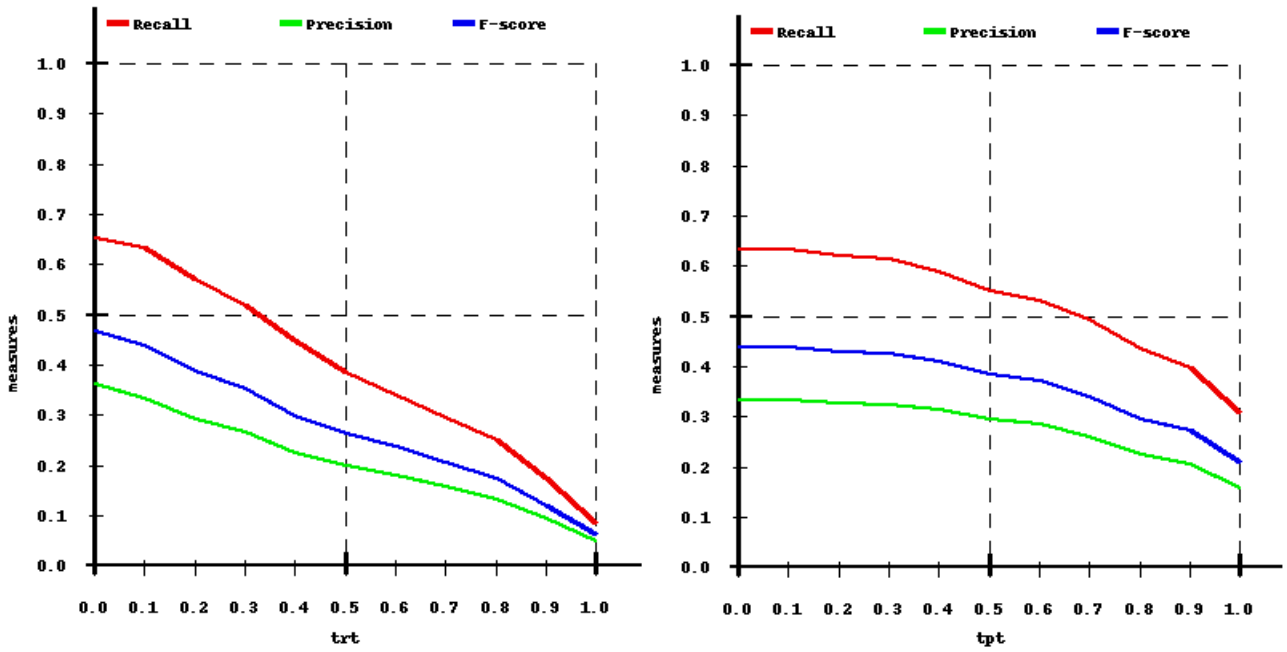


Figura E.2: ADSC-NUS-UIUC variando rt y pt

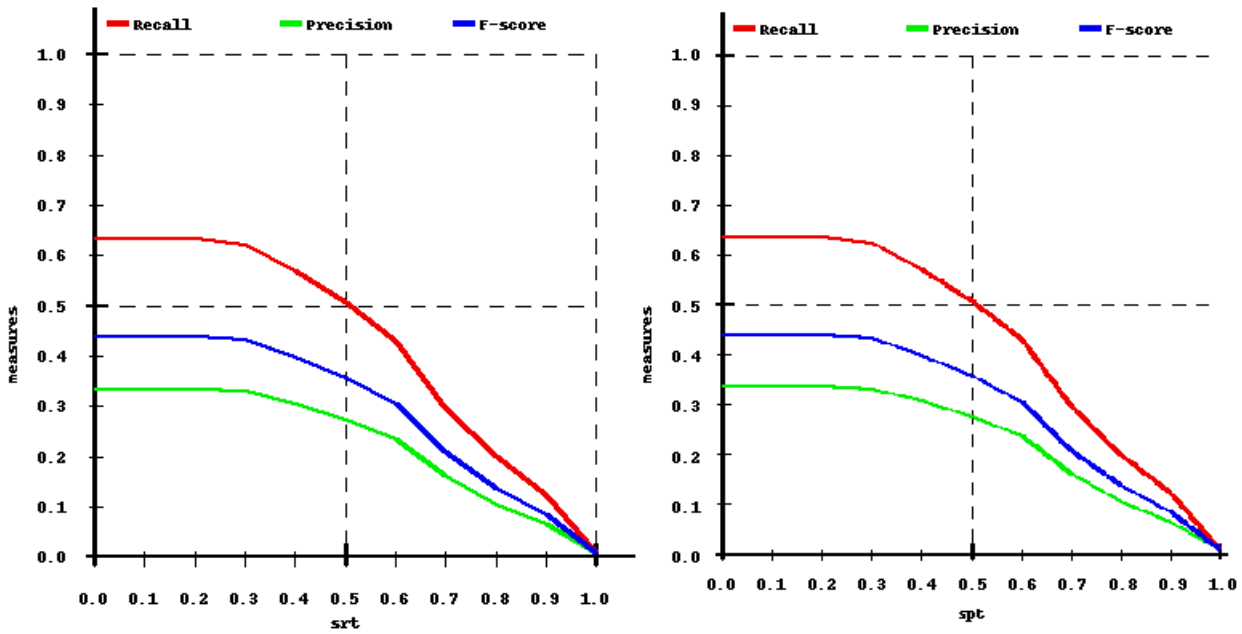


Figura E.3: ADSC-NUS-UIUC variando rs y ps

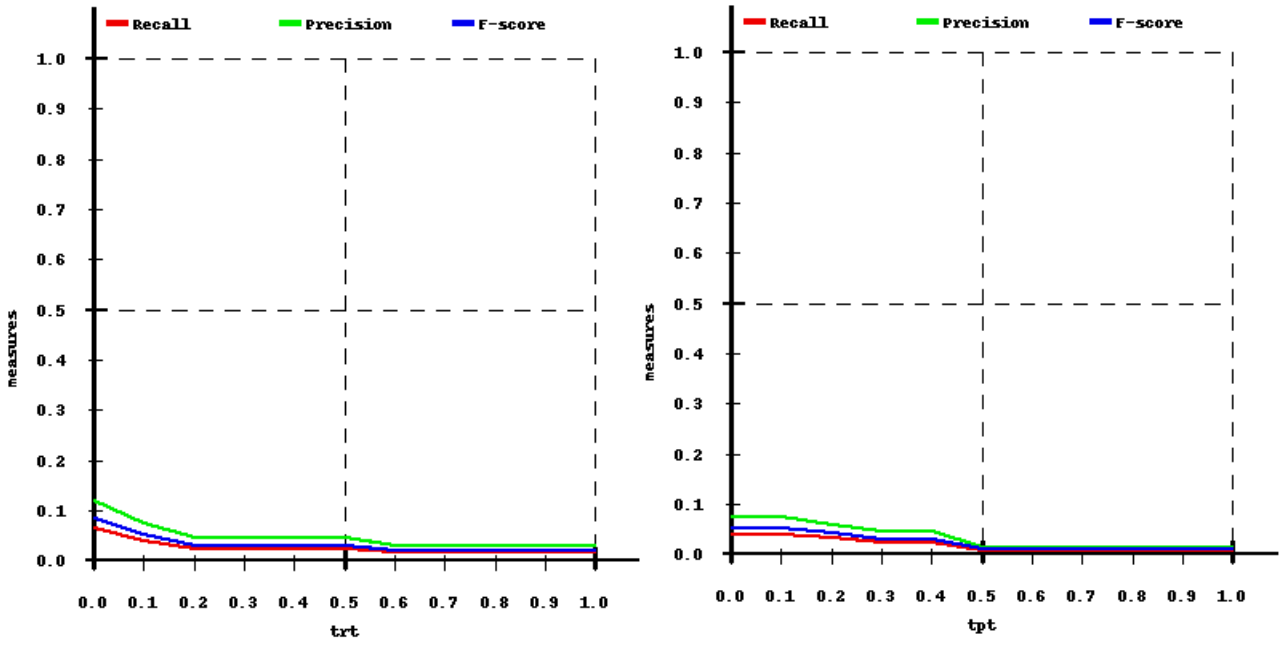


Figura E.4: VPULABUAM variando rt y pt

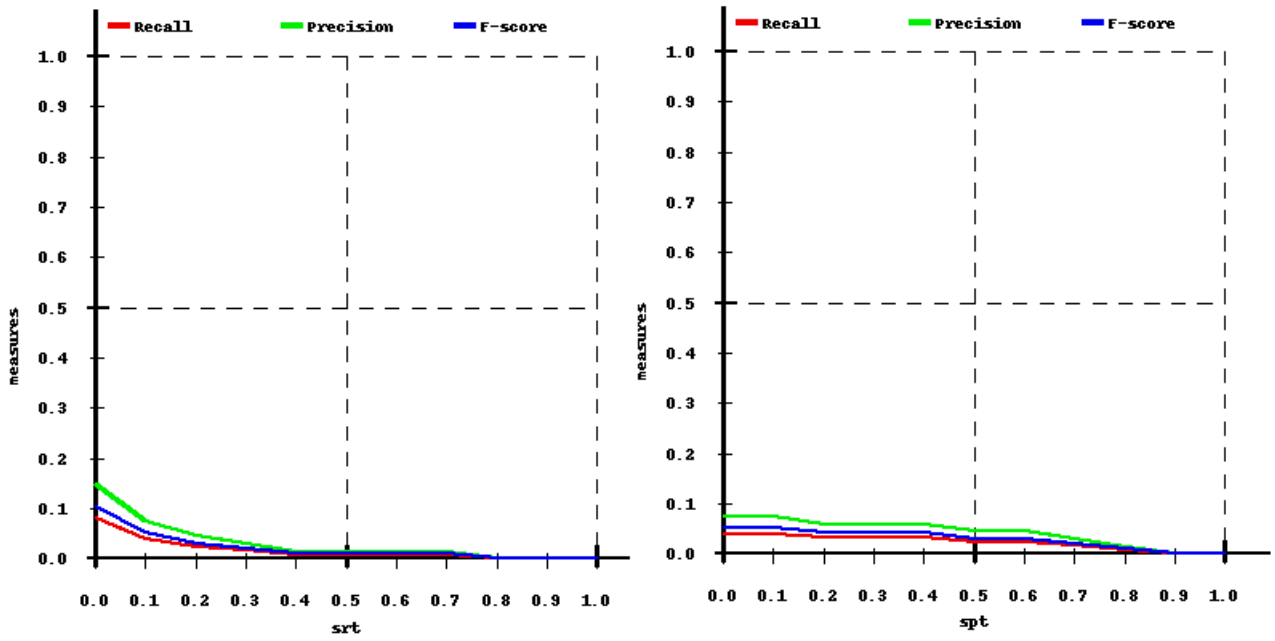


Figura E.5: VPULABUAM variando rs y ps

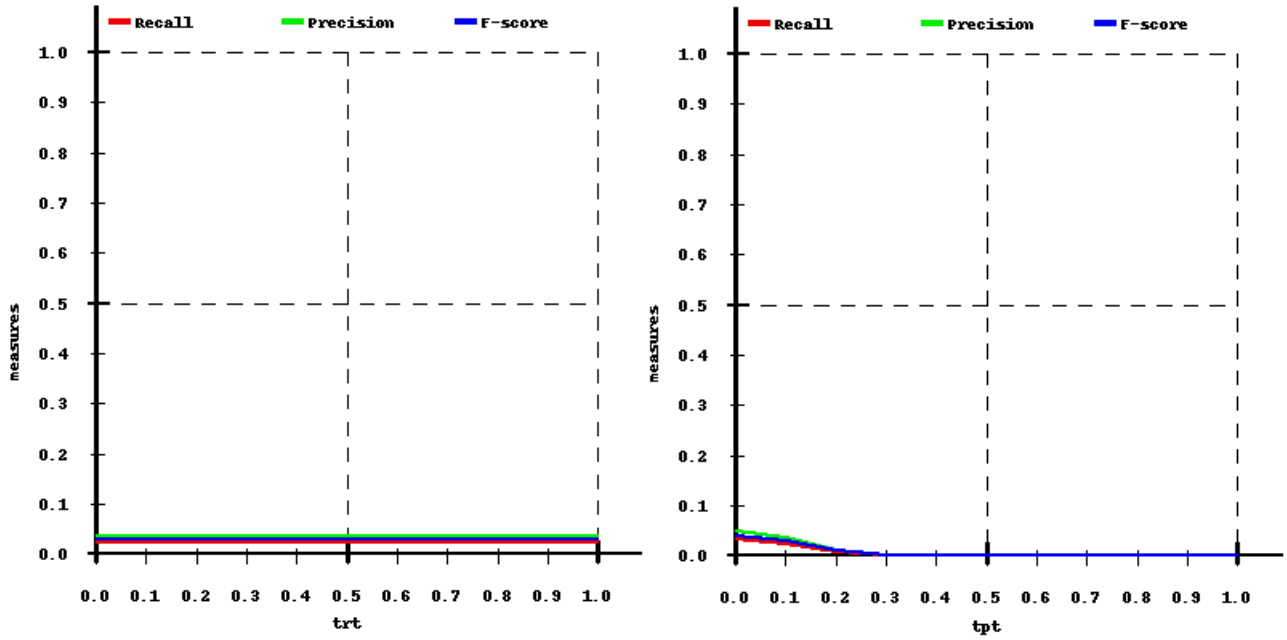


Figura E.6: IACAS variando rt y pt

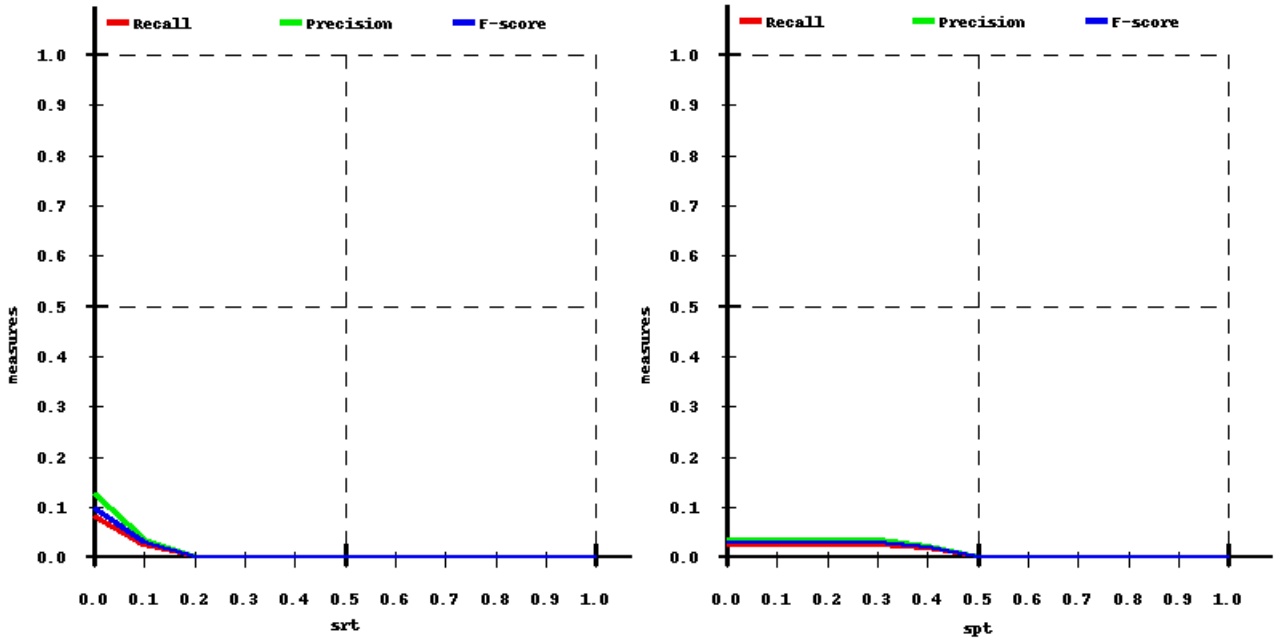


Figura E.7: IACAS variando rs y ps

E.3.5. Matrices de confusión

Las siguientes matrices de confusión representan sólo parejas de acciones del groundtruth y detectadas:

- Las acciones del groundtruth no detectadas NO están incluídas;
- Las acciones detectadas sin equivalente en el groundtruth NO están incluídas.

detection / groumd truth	1 DI	2 GI	3 BO	4 EN	5 ET	6 LO	7 UB	8 HS	9 KB	10 TE
1 DI	50.0	0.0	0.0	0.0	0.0	0.0	0.0	50.0	0.0	0.0
2 GI	50.0	0.0	50.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3 BO	0.0	0.0	80.0	0.0	0.0	0.0	20.0	0.0	0.0	0.0
4 EN	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
5 ET	33.3	0.0	0.0	0.0	66.7	0.0	0.0	0.0	0.0	0.0
6 LO	0.0	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0
7 UB	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8 HS	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
9 KB	0.0	0.0	66.7	33.3	0.0	0.0	0.0	0.0	0.0	0.0
10 TE	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	100.0

Figura E.8: Matriz de confusión de VPULABUAM

ground truth \ detection	1 DI	2 GI	3 BO	4 EN	5 ET	6 LO	7 UB	8 HS	9 KB	10 TE
1 DI	33.3	6.7	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2 GI	33.3	50.0	0.0	0.0	0.0	16.7	0.0	0.0	0.0	0.0
3 BO	0.0	7.7	69.2	0.0	0.0	0.0	0.0	0.0	15.4	7.7
4 EN	0.0	0.0	0.0	95.8	4.2	0.0	0.0	0.0	0.0	0.0
5 ET	0.0	0.0	0.0	0.0	88.9	11.1	0.0	0.0	0.0	0.0
6 LO	0.0	0.0	0.0	11.1	0.0	88.9	0.0	0.0	0.0	0.0
7 UB	20.0	0.0	40.0	0.0	0.0	0.0	40.0	0.0	0.0	0.0
8 HS	6.3	56.3	6.3	6.3	0.0	0.0	0.0	25.0	0.0	0.0
9 KB	0.0	20.0	10.0	0.0	0.0	0.0	0.0	0.0	70.0	0.0
10 TE	0.0	0.0	57.1	0.0	0.0	0.0	0.0	0.0	14.3	28.6

Figura E.9: Matriz de confusión de ADSC-NUS-UIUC

ground truth \ detection	1 DI	2 GI	3 BO	4 EN	5 ET	6 LO	7 UB	8 HS	9 KB	10 TE
1 DI	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2 GI	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3 BO	50.0	0.0	0.0	0.0	0.0	25.0	0.0	25.0	0.0	0.0
4 EN	0.0	0.0	0.0	88.9	0.0	11.1	0.0	0.0	0.0	0.0
5 ET	0.0	0.0	0.0	25.0	25.0	25.0	0.0	25.0	0.0	0.0
6 LO	0.0	0.0	0.0	100.0	0.0	0.0	0.0	0.0	0.0	0.0
7 UB	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
8 HS	0.0	0.0	14.3	57.1	0.0	14.3	0.0	14.3	0.0	0.0
9 KB	100.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
10 TE	33.3	0.0	0.0	33.3	0.0	33.3	0.0	0.0	0.0	0.0

Figura E.10: Matriz de confusión de IACAS

Apéndice F

Presupuesto

1) Ejecucion Material

- Compra de ordenador personal (Software incluido) ..... 2.000 €
- Alquiler de impresora láser durante 6 meses ..... 260 €
- Material de oficina ..... 150 €
- Total de ejecución material ..... 2.400 €

2) Gastos generales

- 16 % sobre Ejecucion Material ..... 352 €

3) Beneficio Industrial

- 6 % sobre Ejecucion Material ..... 132 €

4) Honorarios Proyecto

- 1800 horas a 15 €/ hora ..... 27.000 €

5) Material fungible

- Gastos de impresión ..... 280 €
- Encuadernación ..... 200 €

6) Subtotal del presupuesto

- Subtotal Presupuesto ..... 32.774 €

7) I.V.A. aplicable

- 21 % Subtotal Presupuesto ..... 6.882,5 €

8) Total presupuesto

- Total Presupuesto ..... 39.656,5 €

Madrid, Diciembre de 2012

El Ingeniero Jefe de Proyecto

Fdo.: Sergio Suja Garrido

Ingeniero Superior de Telecomunicación

Apéndice G

Pliego de condiciones

Este documento contiene las condiciones legales que guiarán la realización, en este proyecto, de un "Sistema de análisis y reconocimiento de interacciones y actividades en entornos controlados" para ser visto en pantallas de baja resolución. En lo que sigue, se supondrá que el proyecto ha sido encargado por una empresa cliente a una empresa consultora con la finalidad de realizar dicho sistema. Dicha empresa ha debido desarrollar una línea de investigación con objeto de elaborar el proyecto. Esta línea de investigación, junto con el posterior desarrollo de los programas está amparada por las condiciones particulares del siguiente pliego.

Supuesto que la utilización industrial de los métodos recogidos en el presente proyecto ha sido decidida por parte de la empresa cliente o de otras, la obra a realizar se regulará por las siguientes:

Condiciones generales

1) La modalidad de contratación será el concurso. La adjudicación se hará, por tanto, a la proposición más favorable sin atender exclusivamente al valor económico, dependiendo de las mayores garantías ofrecidas. La empresa que somete el proyecto a concurso se reserva el derecho a declararlo desierto.

2) El montaje y mecanización completa de los equipos que intervengan será realizado totalmente por la empresa licitadora.

3) En la oferta, se hará constar el precio total por el que se compromete a realizar la obra y el tanto por ciento de baja que supone este precio en relación con un importe límite si este se hubiera fijado.

4) La obra se realizará bajo la dirección técnica de un Ingeniero Superior de Telecomunicación, auxiliado por el número de Ingenieros Técnicos y Programadores que se estime preciso para el desarrollo de la misma.

5) Aparte del Ingeniero Director, el contratista tendrá derecho a contratar al resto del personal, pudiendo ceder esta prerrogativa a favor del Ingeniero Director, quien no estará obligado a aceptarla.

6) El contratista tiene derecho a sacar copias a su costa de los planos, pliego de condiciones y presupuestos. El Ingeniero autor del proyecto autorizará con su firma las copias solicitadas por el contratista después de confrontarlas.

7) Se abonará al contratista la obra que realmente ejecute con sujeción al proyecto que sirvió de base para la contratación, a las modificaciones autorizadas por la superioridad o a las órdenes que con arreglo a sus facultades le hayan comunicado por escrito al Ingeniero Director de obras siempre que dicha obra se haya ajustado a los preceptos de los pliegos de condiciones, con arreglo a los cuales, se harán las modificaciones y la valoración de las diversas unidades sin que el importe total pueda exceder de los presupuestos aprobados. Por consiguiente, el número de unidades que se consignan en el proyecto o en el presupuesto, no podrá servirle de fundamento para entablar reclamaciones de ninguna clase, salvo en los casos de rescisión.

8) Tanto en las certificaciones de obras como en la liquidación final, se abonarán los trabajos realizados por el contratista a los precios de ejecución material que figuran en el presupuesto para cada unidad de la obra.

9) Si excepcionalmente se hubiera ejecutado algún trabajo que no se ajustase a las condiciones de la contrata pero que sin embargo es admisible a juicio del Ingeniero Director de obras, se dará conocimiento a la Dirección, proponiendo a la vez la rebaja de precios que el Ingeniero estime justa y si la Dirección resolviera aceptar la obra, quedará el contratista obligado a conformarse con la rebaja acordada.

10) Cuando se juzgue necesario emplear materiales o ejecutar obras que no figuren en el presupuesto de la contrata, se evaluará su importe a los precios asignados a otras obras o materiales análogos si los hubiere y cuando no, se discutirán entre el Ingeniero Director y el contratista, sometiéndolos a la aprobación de la Dirección. Los nuevos precios convenidos por uno u otro procedimiento, se sujetarán siempre al establecido en el punto anterior.

11) Cuando el contratista, con autorización del Ingeniero Director de obras, emplee materiales de calidad más elevada o de mayores dimensiones de lo estipulado en el proyecto, o sustituya una clase de fabricación por otra que tenga asignado mayor precio o ejecute con mayores dimensiones cualquier otra parte de las obras, o en general, introduzca en ellas cualquier modificación que sea beneficiosa a juicio del Ingeniero Director de obras, no tendrá derecho sin embargo, sino a lo que le correspondería si hubiera realizado la obra con estricta sujeción a lo proyectado y contratado.

12) Las cantidades calculadas para obras accesorias, aunque figuren por partida alzada en el presupuesto final (general), no serán abonadas sino a los precios de la contrata, según las condiciones de la misma y los proyectos particulares que para ellas se formen, o en su defecto, por lo que resulte de su medición final.

13) El contratista queda obligado a abonar al Ingeniero autor del proyecto y director de obras así como a los Ingenieros Técnicos, el importe de sus respectivos honorarios facultativos por formación del proyecto, dirección técnica y administración en su caso, con arreglo a las tarifas y honorarios vigentes.

14) Concluida la ejecución de la obra, será reconocida por el Ingeniero Director que a tal efecto designe la empresa.

15) La garantía definitiva será del 4 % del presupuesto y la provisional del 2 %.

16) La forma de pago será por certificaciones mensuales de la obra ejecutada, de acuerdo con los precios del presupuesto, deducida la baja si la hubiera.

17) La fecha de comienzo de las obras será a partir de los 15 días naturales del replanteo oficial de las mismas y la definitiva, al año de haber ejecutado la provisional, procediéndose si no existe reclamación alguna, a la reclamación de la fianza.

18) Si el contratista al efectuar el replanteo, observase algún error en el proyecto, deberá comunicarlo en el plazo de quince días al Ingeniero Director de obras, pues transcurrido ese plazo será responsable de la exactitud del proyecto.

19) El contratista está obligado a designar una persona responsable que se entenderá con el Ingeniero Director de obras, o con el delegado que éste designe, para todo relacionado con ella. Al ser el Ingeniero Director de obras el que interpreta el proyecto, el contratista deberá consultarle cualquier duda que surja en su realización.

20) Durante la realización de la obra, se girarán visitas de inspección por personal facultativo de la empresa cliente, para hacer las comprobaciones que se crean oportunas. Es obligación del contratista, la conservación de la obra ya ejecutada hasta la recepción de la misma, por lo que el deterioro parcial o total de ella, aunque sea por agentes atmosféricos u otras causas, deberá ser reparado o reconstruido por su cuenta.

21) El contratista, deberá realizar la obra en el plazo mencionado a partir de la fecha del contrato, incurriendo en multa, por retraso de la ejecución siempre que éste no sea debido a causas de fuerza mayor. A la terminación de la obra, se hará una recepción provisional previo reconocimiento y examen por la dirección técnica, el depositario de efectos, el interventor y el jefe de servicio o un representante, estampando su conformidad el contratista.

22) Hecha la recepción provisional, se certificará al contratista el resto de la obra, reservándose la administración el importe de los gastos de conservación de la misma hasta su recepción definitiva y la fianza durante el tiempo señalado como plazo de garantía. La recepción definitiva se hará en las mismas condiciones que la provisional, extendiéndose el acta correspondiente. El Director Técnico propondrá a la Junta Económica la devolución de la fianza al contratista de acuerdo con las condiciones económicas legales establecidas.

23) Las tarifas para la determinación de honorarios, reguladas por orden de la Presidencia del Gobierno el 19 de Octubre de 1961, se aplicarán sobre el denominado en la actualidad "Presupuesto de Ejecución de Contrata" y anteriormente llamado "Presupuesto de Ejecución Material" que hoy designa otro concepto.

Condiciones particulares

La empresa consultora, que ha desarrollado el presente proyecto, lo entregará a la empresa cliente bajo las condiciones generales ya formuladas, debiendo añadirse las siguientes condiciones particulares:

1) La propiedad intelectual de los procesos descritos y analizados en el presente trabajo, pertenece por entero a la empresa consultora representada por el Ingeniero Director del Proyecto.

2) La empresa consultora se reserva el derecho a la utilización total o parcial de los resultados de la investigación realizada para desarrollar el siguiente proyecto, bien para su publicación o bien para su uso en trabajos o proyectos posteriores, para la misma empresa cliente o para otra.

3) Cualquier tipo de reproducción aparte de las reseñadas en las condiciones generales, bien sea para uso particular de la empresa cliente, o para cualquier otra aplicación, contará con autorización expresa y por escrito del Ingeniero Director del Proyecto, que actuará en representación de la empresa consultora.

4) En la autorización se ha de hacer constar la aplicación a que se destinan sus reproducciones así como su cantidad.

5) En todas las reproducciones se indicará su procedencia, explicitando el nombre del proyecto, nombre del Ingeniero Director y de la empresa consultora.

6) Si el proyecto pasa la etapa de desarrollo, cualquier modificación que se realice sobre él, deberá ser notificada al Ingeniero Director del Proyecto y a criterio de éste, la empresa consultora decidirá aceptar o no la modificación propuesta.

7) Si la modificación se acepta, la empresa consultora se hará responsable al mismo nivel que el proyecto inicial del que resulta el añadirla.

8) Si la modificación no es aceptada, por el contrario, la empresa consultora declinará toda responsabilidad que se derive de la aplicación o influencia de la misma.

9) Si la empresa cliente decide desarrollar industrialmente uno o varios productos en los que resulte parcial o totalmente aplicable el estudio de este proyecto, deberá comunicarlo a la empresa consultora.

10) La empresa consultora no se responsabiliza de los efectos laterales que se puedan producir en el momento en que se utilice la herramienta objeto del presente proyecto para la realización de otras aplicaciones.

11) La empresa consultora tendrá prioridad respecto a otras en la elaboración de los proyectos auxiliares que fuese necesario desarrollar para dicha aplicación industrial, siempre que no haga explícita renuncia a este hecho. En este caso, deberá autorizar expresamente los proyectos presentados por otros.

12) El Ingeniero Director del presente proyecto, será el responsable de la dirección de la aplicación industrial siempre que la empresa consultora lo estime oportuno. En caso contrario, la persona designada deberá contar con la autorización del mismo, quien delegará en él las responsabilidades que ostente.