Universidad de la República
Facultad de Ingeniería

Procesamiento Multimodal de Señales en la Interpretación de Música

Memoria de proyecto presentada a la Facultad de Ingeniería de la Universidad de la República por

Bernardo Marenco, Magdalena Fuentes, Florencia Lanzaro

en cumplimiento parcial de los requerimientos para la obtención del título de Ingeniero Electricista.

Tutor

Ing. Martín Rocamora ..... Universidad de la República
Ing. Álvaro Gómez ..... Universidad de la República

Tribunal

Ing. Guillermo Carbajal ..... Universidad de la República
Ing. Ignacio Irigaray ..... Universidad de la República
Ing. Juan Pechiar ..... Universidad de la República

Montevideo
jueves 2 julio, 2015

Procesamiento Multimodal de Señales en la Interpretación de Música, Bernardo Marenco, Magdalena Fuentes, Florencia Lanzaro.

Esta tesis fue preparada en LaTeX usando la clase iietesis (v1.1).
Contiene un total de 181 páginas.
Compilada el jueves 2 julio, 2015.
http://iie.fing.edu.uy/

Agradecimientos

A los tutores Martín Rocamora y Álvaro Gómez por su disposición y constante colaboración.

A Pablo Cancela, Ignacio Irigaray y Ernesto López por permitirnos invadir su espacio de trabajo.

A Guillermo Carbajal y Andrés Vallejo por su cooperación y paciencia.

A Luis Jure, Haldo Spontón, Juan Martín López, Rafael Grompone, Daniel Argente, Tomás Laurenzo, Matías Tailanián, Juan Cardelino, Guillermo Rocamora y su equipo, Roberto Rodríguez y Sergio Beheregaray por aportar cada uno un poco de su tiempo para hacer mejor a este proyecto.

A nuestras familias y amigos, por el apoyo incondicional brindado durante todo este tiempo.

Abstract En el presente proyecto se propone un enfoque multimodal para la transcripción de la música de percusión a partir de grabaciones de audio y video. Se utilizaron varias técnicas de procesamiento de señales de manera de derivar información útil de cada uno de los modos. Esto incluyó la detección automática de ciertos objetos de interés en el video y la determinación del instante en el que ocurre un golpe en el audio. Para el desarrollo del sistema multimodal se resolvió utilizar el enfoque de *Feature-Level Fusion*, en el cual la integración de la información proveniente de cada modo se realiza a nivel de características. Una vez fusionada la información de los distintos modos, se utilizaron técnicas de reconocimiento de patrones para diseñar un sistema de clasificación multimodal. Se realizaron pruebas usando cada modo por separado para evaluar las ventajas de usar un enfoque mutlimodal respecto a utilizar un único modo. Dichos experimentos reflejan que este enfoque es capaz de mejorar el desempeño alcanzado con cada fuente de información por separado, mostrando las ventajas del método propuesto.

Tabla de contenidos

Agradecimientos I

1. Introducción 1
1.1. Fundamentos y antecedentes . . . 1
1.2. Marco del proyecto . . . 2
1.3. Objetivos . . . 2
1.3.1. Objetivos generales . . . 2
1.3.2. Objetivos específicos . . . 2
1.4. Motivación . . . 3
1.5. Resumen del Proyecto . . . 4
1.5.1. Datasets . . . 5
1.5.2. Modo sensor . . . 6
1.5.3. Modo audio . . . 7
1.5.4. Modo video . . . 8
1.5.5. Sistema multimodal . . . 12
1.6. Estructura de la documentación . . . 14

2. Conjuntos de datos 17
2.1. Dataset eMe . . . 17
2.2. Dataset Zavala . . . 18
2.2.1. Pre-producción . . . 19
2.2.2. Registro de video . . . 22
2.3. Etiquetado y sincronización . . . 23

I Procesamiento de señales 25

3. Procesamiento de audio 27
3.1. Detección de eventos y etiquetado . . . 27
3.2. Extracción de características . . . 29
3.2.1. Determinación de características a utilizar . . . 29
3.2.2. Primer conjunto de características derivadas del audio . . . 30
3.2.3. Segundo conjunto de características derivadas del audio . . . 33
3.2.4. Tercer conjunto de características derivadas del audio . . . 35

4. Procesamiento de video **37**
4.1. Segmentación de la lonja . . . 38
4.1.1. Primer algoritmo: detección utilizando un acumulador unidimensional . . . 38
4.1.2. Segundo algoritmo: detección por mínimos cuadrados . . . 40
4.2. Segmentación del palo . . . 42
4.2.1. Primer algoritmo: filtro de color y detección de segmentos . 42
4.2.2. Segundo algoritmo: mejoras de la detección en cada base de datos . . . 44
4.3. Segmentación de la mano . . . 51
4.3.1. Primer algoritmo: segmentación por color . . . 51
4.3.2. Segundo algoritmo: clasificación automática . . . 54
4.4. Extracción de características . . . 60
4.4.1. Conjunto geométrico . . . 61
4.4.2. Conjunto DCT . . . 66

II Clasificación **69**

5. Marco teórico **71**
5.1. Introducción al Reconocimiento de Patrones . . . 71
5.1.1. Árboles de decisión . . . 71
5.1.2. Vecinos más cercanos(k-NN) . . . 73
5.1.3. Máquinas de vectores de soporte (SVM) . . . 73
5.1.4. Selección de características . . . 74
5.1.5. Evaluación de desempeño . . . 76
5.2. Procesamiento multimodal . . . 77

6. Selección de características **79**
6.1. Audio . . . 80
6.1.1. Determinación de parámetros óptimos . . . 81
6.2. Video . . . 82
6.2.1. Selección de características en el conjunto geométrico . . . 83
6.2.2. Selección de características del conjunto DCT . . . 84
6.2.3. Selección final de características del modo video . . . 86
6.2.4. Determinación de parámetros óptimos . . . 88
6.3. Enfoque multimodal . . . 89
6.3.1. Combinación de las características extraídas de cada modo 90

7. Evaluación de desempeño **95**
7.1. Evaluación considerando tres tipos de golpes . . . 95
7.1.1. Audio . . . 95
7.1.2. Video . . . 96
7.1.3. Enfoque multimodal . . . 97
7.2. Evaluaciones considerando seis tipos de golpe . . . 98
7.2.1. Audio . . . 98

7.2.2. Video . . . 100
7.2.3. Enfoque multimodal . . . 101
7.3. Evaluación sobre un registro de la base Zavala . . . 103

8. Discusión, conclusiones y trabajo futuro **107**

A. Sensores **111**
A.1. Introducción . . . 111
A.2. Sensores Básicos . . . 111
A.2.1. Sensores Piezoeléctricos . . . 112
A.2.2. Sensores Resistivos de Fuerza . . . 113
A.2.3. Sensores de Fibra Optica . . . 113
A.2.4. Sensores Capacitivos . . . 113
A.2.5. Acelerómetros . . . 114
A.3. Sistemas de Captura de Movimiento . . . 116
A.3.1. Leap Motion Controller . . . 117
A.3.2. Kinect . . . 119
A.4. Evaluación de la utilización de sensores en el proyecto . . . 120

B. Filtro de Color **123**
B.1. Espacio YUV . . . 123
B.2. Segmentación en el plano UV . . . 123

C. Filtro de Kalman **129**

D. Detección y seguimiento de marcadores **133**
D.1. Algoritmo de seguimiento de marcadores . . . 133
D.1.1. Oclusiones y diferencias de iluminación . . . 135
D.1.2. Corrector . . . 136
D.2. Visualizador . . . 137

E. Reconstrucción 3D de la escena **139**
E.1. Descripción de la escena . . . 139
E.2. Calibración de cámaras . . . 140
E.2.1. Parámetros intrínsecos y extrínsecos . . . 140
E.2.2. Calibración de una cámara . . . 142
E.3. Geometría de un par estéreo . . . 143
E.4. Calibración estéreo . . . 144
E.5. Triangulación . . . 146
E.6. Aplicación de este enfoque al problema . . . 149

F. Software **151**

Referencias **157**

Índice de tablas **165**

Tabla de contenidos

Índice de figuras **167**

Capítulo 1

Introducción

El presente capítulo presenta un panorama general del proyecto, exponiendo los objetivos y un breve resumen de su desarrollo y ejecución. En los capítulos siguientes se desarrollará en profundidad cada tema aquí mencionado.

1.1. Fundamentos y antecedentes

Las interacciones humanas raramente se dan involucrando únicamente un sentido. En cambio, resulta más común que los intercambios se den sobre varias vías, como ser idioma escrito o hablado, contacto visual, expresiones, posturas, etc., tanto del emisor como del receptor. En el campo del procesamiento de señales, cada una de estas vías es comúnmente denominada *modo*. Por lo tanto, el término *análisis multimodal* hace referencia a una disciplina que busca analizar, modelar y entender cómo extraer la información necesaria de múltiples de estas vías. Recientemente, este tipo de análisis ha cobrado un gran impulso en el procesamiento de señales [40], especialmente aplicado a la interacción humano-máquina. Dentro de sus aplicaciones típicas se encuentran el reconocimiento de voz (*speech-to-text*) [49, 51, 68] y la identificación de personas usando varias cualidades, como timbre de voz y rasgos faciales [34].

Los avances tecnológicos de los últimos años, en particular el aumento de la capacidad de cómputo necesaria para procesar varias fuentes de información en simultáneo, han posibilitado la incorporación del análisis multimodal en aplicaciones multimedia. Uno de los grandes exponentes dentro de este mundo multimedia es la música, la cual es un fenómeno naturalmente multimodal [40]. Además de su representación en notación musical o como una grabación de audio, la música está asociada a varios otros modos de información, ya sea en forma de texto (e.g. letra, género, reseñas), imagen (e.g. arte de tapa, fotos) o video (e.g. video-clips).

Trabajar con información musical multimodal requiere el desarrollo de métodos para establecer de forma automática relaciones semánticas entre diferentes representaciones y formatos, por ejemplo, sincronización de audio y partitura, o alineamiento de audio y letra en una canción. Esto tiene diversas aplicaciones, como la interacción multimodal para la búsqueda de música [73], la identificación de ins-

trumentos musicales usando información audiovisual [61], la correlación visual de videos de música [46,60], la transcripción automática audiovisual de batería [47,48], el análisis de escenas de danza [37, 39] y el análisis musical interactivo [86], entre otras.

En este proyecto se buscó estudiar y hacer uso de las técnicas de procesamiento multimodal en la interpretación de música, tomando como caso de estudio la interpretación de Candombe. Se pretendió explotar la información complementaria que puede extraerse de diferentes modos, tales como registros de audio y video, apuntando a facilitar mediante herramientas automáticas tareas como la transcripción a notación musical o el estudio de gestualidad y técnica interpretativa. Por ejemplo, para la transcripción automática, el instante de ocurrencia de un evento puede determinarse con suficiente precisión sólo a partir de la señal de audio, pero para reconocer automáticamente un cierto tipo de golpe puede ser más efectivo usar además de características sonoras información visual (como en [47]).

1.2. Marco del proyecto

El proyecto fue apoyado por el Programa de Iniciación a la Investigación de la *Comisión Sectorial de Investigación Científica* (CSIC). Se contó además con el apoyo de los dos principales grupos de investigación del Departamento de Procesamiento de Señales del Instituto de Ingeniería Eléctrica, Facultad de Ingeniería, UdelaR: el Grupo de Procesamiento de Audio (GPA[1]) y el Grupo de Tratamiento de Imágenes (GTI[2]).

1.3. Objetivos

1.3.1. Objetivos generales

Uno de los principales intereses del proyecto fue generar experiencia sobre técnicas de procesamiento multimodal de señales, un campo de investigación en pleno desarrollo y el cual no ha sido explorado en profundidad en el ámbito local. Se persiguió también el objetivo de contribuir al estudio de músicas tradicionales, en particular el Candombe afro-uruguayo, por medio de la generación de datos y el desarrollo de herramientas de software que fomenten el uso de la tecnología en estudios musicológicos.

1.3.2. Objetivos específicos

Los objetivos específicos que este proyecto intentó alcanzar fueron:

[1]Página web del grupo: `http://iie.fing.edu.uy/investigacion/grupos/gpa/`, página del grupo en CSIC: `http://darwin.csic.edu.uy/grupos/grupos?tipo=unover&id=1616`.

[2]Página web del grupo: `http://webiie.fing.edu.uy/node/4`, página del grupo en CSIC: `http://darwin.csic.edu.uy/grupos/grupos?tipo=unover&id=45`.

- Explorar diferentes tecnologías de adquisición de datos (micrófonos, cámaras de video y otros sensores) evaluando la utilidad de la información generada para el estudio de interpretación de Candombe.
- Contribuir con registros multimodales de alta calidad de la ejecución de Candombe a cargo de intérpretes reconocidos, para ser utilizados en este proyecto y en investigaciones posteriores.
- Estudiar algunas de las técnicas existentes en el campo del procesamiento multimodal de señales y evaluar la utilidad de este enfoque en el estudio de la interpretación de música.
- Desarrollar aplicaciones que permitan realizar una clasificación automática básica de una interpretación de repique. Este tipo de aplicaciones pueden facilitar o complementar la transcripción manual en estudios musicológicos (como [55]).

1.4. Motivación

Tal como se mencionó anteriormente, las técnicas basadas en procesamiento multimodal han tenido un importante auge en los últimos años dando lugar a nuevas líneas de investigación en el campo del procesamiento de señales. Si bien la experiencia en procesamiento de señales a nivel local es muy considerable, el paradigma multimodal no ha sido explorado aún en profundidad. Por tanto, la motivación principal del presente proyecto fue dar los primeros pasos en un área de investigación en pleno desarrollo, combinando el conocimiento existente en los grupos de investigación del Departamento de Procesamiento de Señales (IIE, FING, UdelaR).

Considerar al Candombe como caso de estudio representó por sí mismo una motivación adicional. La práctica del Candombe constituye uno de los más valiosos aportes de la comunidad afro-uruguaya a nuestro patrimonio y es a la vez uno de las rasgos más característicos de la cultura popular [54]. El ritmo ha sido integrado en distinto grado en varios géneros de la música uruguaya, como el tango o el canto popular, y ha dado lugar al candombe-beat y otras formas musicales posteriores. Desde los estudios pioneros de hace más de cincuenta años [29] diversos trabajos musicológicos han abordado el análisis del ritmo y la técnica de los tambores [27,41, 42,50,54,55]. Sin embargo, es tan solo recientemente que se propone la aplicación de herramientas automáticas de procesamiento de audio con ese propósito, en el contexto de un trabajo de investigación incipiente dentro del GPA [67,74,75].

En relación a esto, el presente proyecto pretendió complementar y extender el trabajo que se lleva adelante actualmente incorporando el enfoque del procesamiento multimodal de señales. A través de la generación de registros multimodales de interpretaciones de Candombe y el desarrollo de herramientas de software para su procesamiento y análisis, se buscó aportar a la aplicación de la tecnología para fortalecer la generación de conocimiento, la protección y la revalorización de este fenómeno cultural. Además de la Escuela Universitaria de Música (que participa

de este proyecto), otras instituciones como el Centro de Documentación Musical Lauro Ayestarán, del Ministerio de Educación y Cultura, podrán beneficiarse de este tipo de trabajo.

Asimismo, en sintonía con otros esfuerzos en curso en la comunidad científica se espera que, como se señala en [79][3], el estudio cuidadoso de una tradición musical particular fuera del paradigma de música comercial occidental pueda contribuir a la construcción de modelos más generales y ricos que los que actualmente dominan la investigación en tecnologías de la información aplicadas a la música.

Por último, habida cuenta del importante desarrollo de la industria del software en el país, cabe señalar que la incorporación de tecnologías de la información como las planteadas en este proyecto puede significar un valor agregado de la industria nacional y ampliar el espectro de aplicaciones que actualmente se desarrollan.

1.5. Resumen del Proyecto

El objetivo principal de este proyecto fue desarrollar un sistema multimodal para la clasificación de golpes en la interpretación de repique. Para ello, se planteó al inicio un escenario que involucraba el uso de tres modos de información diferentes: un modo derivado de las grabaciones de video, uno procedente de grabaciones de audio y otro obtenido de sensores de posición, movimiento y/o presión. Llamaremos a dichos modos de información *modo video*, *modo audio* y *modo sensor* respectivamente.

Como primer acercamiento al problema se consideró una clasificación básica, en la que se distinguía entre tres tipos de golpes: golpes de *mano*, *palo* o *madera*. Luego se aumentó la complejidad introduciendo tres tipos de golpes adicionales: *rebotado* (en el que el palo da golpes sucesivos en un intervalo corto de tiempo), *borde* (golpe de palo sobre el borde de la lonja) y *flam* (golpe casi simultáneo de mano y palo). La diferenciación entre seis tipos de golpes fue realizada por considerarse que el análisis de este problema es más cercano a la realidad. Además, el estudio de las diferencias entre cada uno de estos golpes puede brindar información útil sobre técnicas interpretativas y gestualidad.

Para el desarrollo del sistema multimodal se resolvió utilizar el enfoque de *Feature-Level Fusion*, en el cual la integración de la información proveniente de cada modo se realiza a nivel de características [40]. Cabe aclarar que el enfoque utilizado no es estrictamente *Feature-Level Fusion*, en el sentido de que las características de cada modo no fueron calculadas independientemente una de otra, sino que se usó información del audio para guiar la extracción de características. La idea detrás de esto es que la ubicación temporal de los golpes de percusión puede establecerse fácilmente a partir de la señal de audio, como se hace en [38]. En este sentido, el primer paso del sistema multimodal desarrollado es detectar la ubicación temporal de los golpes a partir del audio, lo que se conoce usualmente como

[3]El proyecto que allí se describe, CompMusic, es probablemente uno de los más grandes que se lleva adelante actualmente en Recuperación de Información Musical (MIR). Está financiado por el European Research Council con 2.5 millones de Euros y durará 5 años (2011-2016). Ver `http://compmusic.upf.edu`.

detección de onsets. Como se verá más adelante, para la detección de onsets se siguió un proceso semiautomático basado en el procesamiento de la señal de audio y la posterior verificación manual de los eventos detectados. Luego, cada evento se etiquetó de forma manual, para indicar a cuál de las clases definidas corresponde. La información sobre la ubicación temporal de los golpes se utilizó para calcular las características de cada modo en el entorno de un golpe. En la Figura 1.1 se ejemplifica con un golpe de palo algunas de las señales utilizadas. La señal indicada como spectral flux está calculada a partir de la señal de audio y se usa para determinar la ubicación del evento, mientras que la última señal mostrada en el esquema se calcula a partir del video e indica la posición vertical del palo, lo que resulta útil para definir el tipo de golpe.

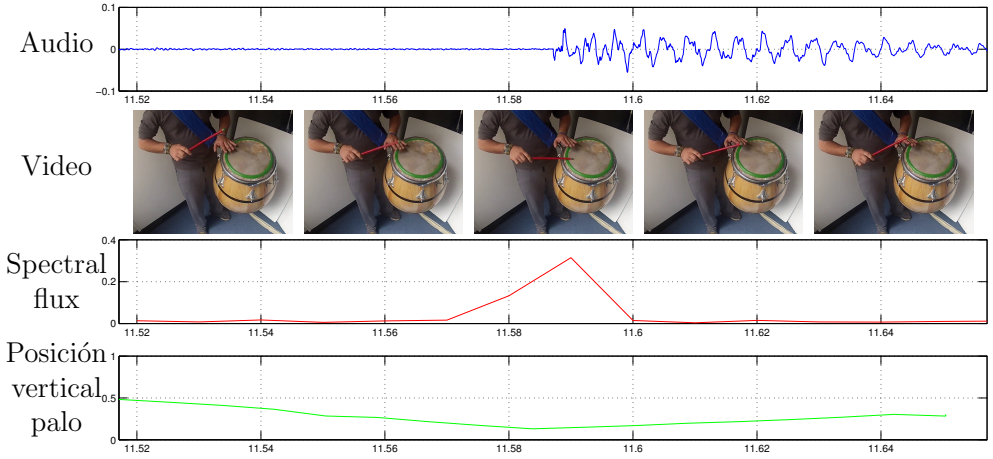


Figura 1.1: Ejemplo de procesamiento multimodal para un golpe de palo. Todas las cantidades graficadas están normalizadas.

Una vez determinadas las características de cada modo, se utilizaron técnicas de reconocimiento de patrones para diseñar un sistema de clasificación multimodal. Dado que la principal motivación de combinar distintos modos fue obtener una mejor solución para el problema frente a la utilización de cada modo como fuente única de información, se realizaron pruebas empleando cada modo por separado de manera de evaluar esta hipótesis.

A continuación se presenta una breve explicación de cada ítem del proyecto.

1.5.1. Datasets

Se trabajó con dos bases de datos distintas. La primera consiste en registros de audio de buena calidad y video a alta tasa de cuadros por segundo, y fue realizada en el Estudio de Música Electroacústica (eMe) de la Escuela Universitaria de Música [19] previo al comienzo de este proyecto. Participaron en este registro cuatro intérpretes, quienes realizaron dos tomas cada uno de improvisación de repique. Se referirá a esta base de aquí en adelante como *base eMe*.

El segundo conjunto de datos fue planificado y producido en el marco del proyecto. Se realizó un rodaje en la sala Zavala Muniz del Teatro Solís en el que participaron cinco intérpretes de Candombe de reconocida trayectoria. En esta oportunidad, se grabó audio de buena calidad y video a alta tasa de cuadros por segundo en configuración estéreo para tener la posibilidad de realizar reconstrucción 3D de la escena. Esta base se denominará *base Zavala*.

En ambos registros se acondicionó la escena para facilitar el procesamiento de imágenes, pintando el palo de color verde o rojo, y de verde el contorno de la lonja. Además, en el segundo registro se colocaron marcadores en el cuerpo de los intérpretes de manera de tener puntos correspondientes en las grabaciones de video estéreo. En la figura 1.2 se muestran los cuadros (o *frames*) izquierdo y derecho del video estéreo obtenido durante el registro.

Figura 1.2: Imagen estéreo de un intérprete (Sergio Ortuño) durante el registro.

El etiquetado de los datos estuvo a cargo de Luis Jure [20] y Martín Rocamora [21]. Esto significa determinar, a partir del audio y video capturados, qué tipo de golpe fue ejecutado por el intérprete y en qué momento. Esta información es escencial para el desarrollo de un sistema automático de transcripción (o de reconocimiento de patrones), porque son las etiquetas las que permiten ajustar o entrenar el sistema y evaluar su desempeño.

1.5.2. Modo sensor

El primer modo estudiado fue el modo que involucra sensores de posición, movimiento o presión, entre otros. Para ello se investigó qué tipo de sensores existían en el momento a disposición en el mercado, bajo la premisa de que el sensor elegido debía ser una solución acabada y no implicar desarrollo de hardware o software específico. A su vez, se planteó que no sería deseable que la solución

fuese invasiva, lo que fue determinante a la hora de descartar varios sensores (como por ejemplo los piezoeléctricos). También se tuvieron en cuenta requisitos propios del problema, como la velocidad de respuesta requerida.

Una vez realizada la investigación primaria y definidos los requisitos de funcionamiento, se hicieron pruebas con algunos sensores. Todas ellas determinaron la no inclusión de este modo de información en la solución final del proyecto por no cumplir con las especificaciones mencionadas anteriormente. Por lo tanto se centró el análisis en los otros dos modos: audio y video.

1.5.3. Modo audio

En este caso se realizó una búsqueda bibliográfica de las características usualmente utilizadas en procesamiento de audio para la detección de eventos y el reconocimiento de distintos tipos de sonidos de percusión.

Se trabajó con tres grupos de características, todos derivados del espectro de la señal de audio. El primer grupo consistió en características que consideran a la envolvente del espectro como una distribución de probabilidad y calculan medidas que describen su forma. En particular, el conjunto considerado estuvo formado por los cuatro primeros momentos estadísticos (denominados *spectral centroid*, *spectral spread*, *spectral skewness* y *spectral kurtosis* en la bibliografía), medidas de cómo crece y decrece la envolvente (*spectral slope* y *spectral decrease*, respectivamente) y el máximo valor que toma respecto a la media (*spectral crest*).

Como segundo grupo de características se consideraron los coeficientes cepstrales de frecuencias mel (*Mel Frequency Cepstral Coefficients*, MFCCs), que también intentan describir la envolvente del espectro pero con un enfoque distinto. El término *cepstral* hace referencia al *cepstrum* de una señal, definido como la transformada inversa de Fourier del logaritmo de $|X(f)|$, siendo $X(f)$ la transformada de Fourier de la señal de audio [57]. En el caso que la señal sea discreta, también lo será su cepstrum, por lo que éste puede describirse por los *coeficientes cepstrales*. Éstos son computados a través de un banco de filtros mel, el cual tiene la particularidad de que las frecuencias centrales de cada filtro están equiespaciadas en la escala mel de frecuencias. Además, cada filtro tiene forma triangular, con ganancia unidad en su frecuencia central y con sus otros dos vértices ubicados en la frecuencia central de los filtros adyacentes.

El tercer conjunto consistió en dos características adicionales derivadas del *spectral flux* de audio. El mismo es una medida de la variación local del espectro de la señal [57]. Dado que un golpe de percusión representa una gran concentración de energía en un período corto de tiempo, éste se manifestará en el spectral flux como un máximo local. Como ciertos golpes (por ejemplo el flam o el rebotado) son en realidad una sucesión de golpes en un pequeño intervalo de tiempo, se caracterizan por presentar una sucesión de máximos locales en el spectral flux, donde el primer máximo es el de mayor amplitud (ver Figura 1.3). Por lo tanto, se calcularon dos caracteríticas derivadas del spectral flux para intentar reflejar esta realidad: la cantidad de máximos en una ventana de tiempo centrada en el evento de audio y la diferencia de alturas entre el primer y segundo máximo (en caso de

que éste existiese).

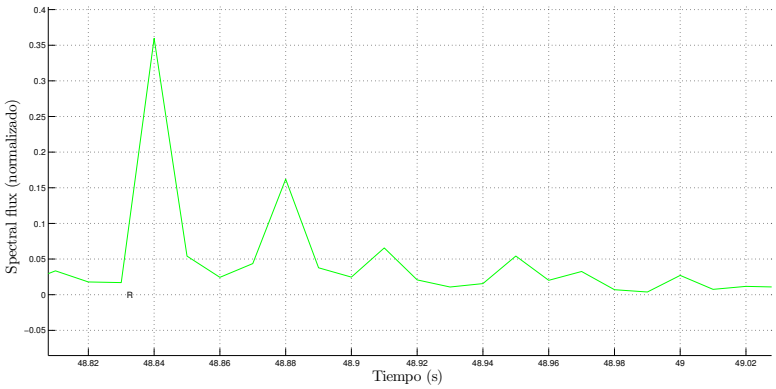


Figura 1.3: Spectral flux (normalizado) de un golpe rebotado. El máximo de mayor amplitud indica el comienzo del golpe, mientras que los máximos siguientes son causados por los rebotes en la lonja.

Sobre la unión de los conjuntos se aplicaron dos métodos de selección de características, uno basado en correlación y otro por encapsulado, obteniendo como selección final un conjunto formado por algunos MFCCs, los cuatro momentos, la pendiente negativa de la envolvente y ambas características extraídas del spectral flux.

Con las características seleccionadas se comparó el desempeño de distintos clasificadores realizando una búsqueda exhaustiva de sus mejores parámetros. Los algoritmos de k-NN (considerando 5 vecinos) y SVM (con un kernel de base radial gaussiana) fueron los que alcanzaron mayor desempeño, obteniéndose promedios en el entorno de 93 % y 94 % respectivamente (ver Sección 6.1.1). Estos promedios fueron obtenidos para el problema con seis clases, realizando 10 repeticiones con cada clasificador, utilizando validación cruzada en 10 particiones sorteadas aleatoriamente.

1.5.4. Modo video

El tercer modo estudiado fue el correspondiente al video. El trabajo puede separarse en tres grandes etapas.

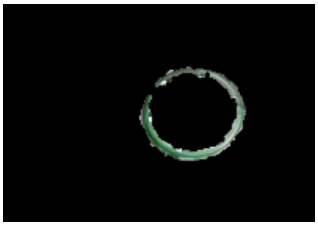
Primera etapa: detecciones y segmentación

La primera etapa consistió en la detección y segmentación de tres elementos de la escena: la mano izquierda del intérprete (ya que todos eran diestros), el palo y la lonja.

Para la lonja se utilizó un filtro de color verde de manera de separar su contorno del resto de la imagen. Se ajustó una elipse al contorno obtenido con el objetivo de caracterizar numéricamente su posición. Este ajuste se realizó utilizando [15]. En

la Figura 1.4 se muestran los pasos seguidos para detectar la lonja en un cuadro dado.

(a) Frame original



(b) Filtro de color verde sobre 1.4(a)

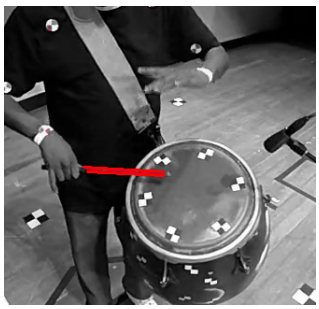
(c) Ajuste de una elipse a los puntos de primer plano de 1.4(b), usando [15]

Figura 1.4: Pasos para la detección de la lonja.

Para la detección de palo también se utilizó un filtro de color (rojo o verde dependiendo del video). Las condiciones de iluminación de algunos de los registros de palo rojo provocaron que el color fuese muy variable en una misma toma, por lo que en estos casos se utilizó un algoritmo de extracción de fondo [92]. Dicho algoritmo permite separar en el video los elementos que presentan más movimiento de aquellos que están fijos, como se muestra en la Figura 1.5(b), por lo que se utilizó para separar el palo del resto de los elementos de la escena. Una vez segmentado se aplicó un algoritmo de detección de segmentos de recta [52] para describir numéricamente su ubicación (Figura 1.5(c)). El procedimiento se presenta en la Figura 1.5.

(a) Imagen original

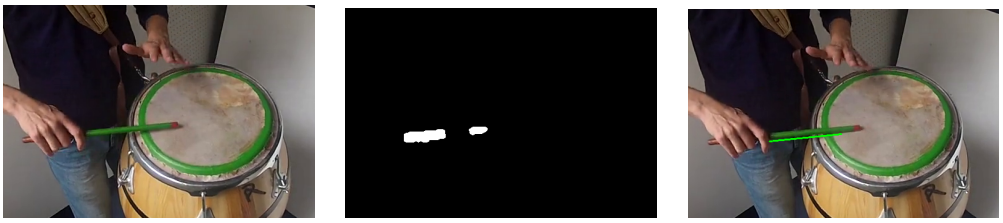
(b) Máscara obtenida a partir de aplicar el algoritmo de extracción de fondo sobre la Figura 1.5(a).



(c) Detección de segmentos con el algoritmo LSD.

Figura 1.5: Procedimiento de un algoritmo de detección de palo implementado.

Para los casos en que el palo está pintado de verde se utilizó otro enfoque. Dado que la lonja está pintada del mismo color y confundía al detector de segmentos, se la eliminó de la imagen, como se muestra en la Figura 1.6(a). Aplicando nuevamente el detector de segmentos se detectó el palo y se corrigió su largo (Figura 1.6(b))



(a) Frame original

(b) Filtrado por color sobre 1.6(a) y eliminado de la lonja.

(c) Detección del palo verde sobre la Figura 1.6(b)

Figura 1.6: Procedimiento de detección de palo verde

Para detectar la mano izquierda se comenzó por segmentar la piel del intérprete. Primero se intentó modificar el filtro de color utilizado en las segmentaciones anteriores de manera de separar la piel del resto de la escena [83]. Dado que existen otros elementos con colores similares, como ser el piso de la sala o el tambor, esta solución no fue suficiente, como se muestra en la Figura 1.7(b). Se resolvió entonces utilizar el filtro de color para separar la piel y los elementos similares del resto de la escena. La segmentación final de piel se realizó utilizando un clasificador a la salida del filtro de color. Se probaron dos clasificadores distintos: un Árbol de Decisión y un Random Forest. Las características usadas para la clasificación fueron los valores YUV de cada pixel. En la Figura 1.7 se muestra el resultado de cada paso de esta detección para un frame en particular.

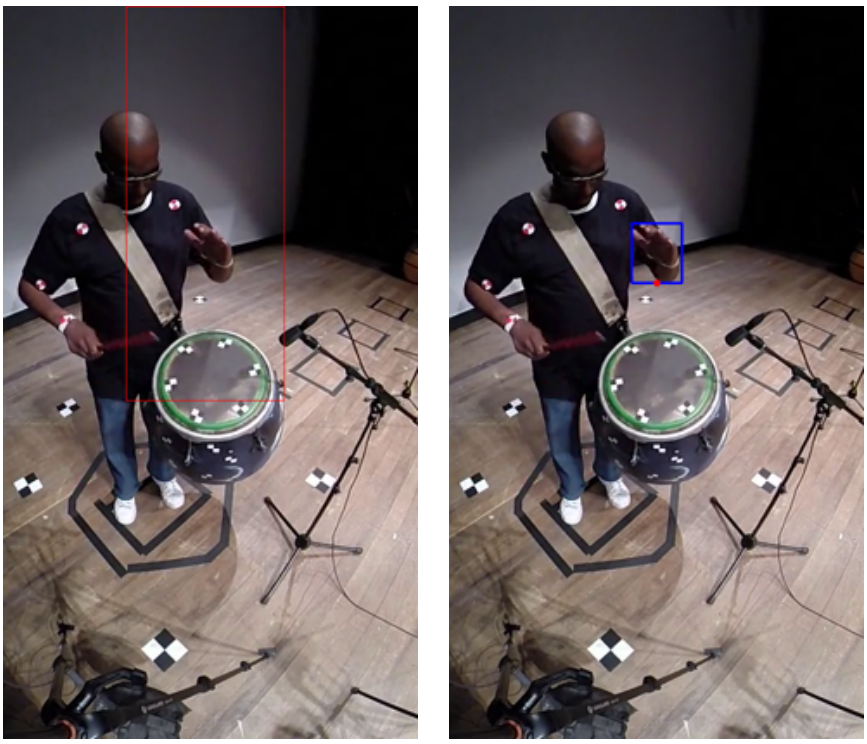
(a) Frame original

(b) Filtro de color sobre 1.7(a)

(c) Clasificador aplicado a 1.7(b), utilizando como características los valores YUV de cada píxel

Figura 1.7: Pasos para la detección de piel.

Una vez segmentada la piel, se utilizó como referencia la ubicación de la lonja previamente detectada para efectivamente determinar la posición de la mano izquierda, ya que se asumió que durante la mayor parte del video la mano izquierda se encuentra en una zona por encima de la lonja (Figura 1.8(a)). Asumiendo que la mano siempre se encuentra dentro de esta zona de interés, se buscó el contorno



(a) Zona de búsqueda de la mano izquierda.

(b) Bounding box del contorno más grande detectado y punto utilizado como estimador de la posición de la mano izquierda.

Figura 1.8: Zona de búsqueda de la mano izquierda y bounding box del contorno más grande.

más grande de la detección de piel dentro de la máscara. Una vez separado, se intentó obtener una medida numérica que fuese representativa de la posición de la mano. Una primera aproximación fue hallar el *bounding box* del contorno más grande, que se define como el menor rectángulo (con lados paralelos a los lados de la imagen) que contiene a dicho contorno. Como un indicador de la posición de la mano se tomó al punto medio del segmento inferior del bounding box. La Figura 1.8(b) muestra el bounding box detectado y el punto que se toma como indicador de la posición de la misma. Finalmente se realizó un filtrado de Kalman [56] sobre el punto detectado para eliminar ruido proveniente de la segmentación.

Segunda etapa: cálculo de características

La segunda etapa de trabajo sobre el video se centró en derivar de la segmentación medidas númericas que contengan información útil para la clasificación de distintos tipos de golpes. Usualmente estas medidas son llamadas *características*. Por ejemplo, si en una imagen se tiene detectada la posición de la mano y la de la lonja, se puede derivar una característica que mida la distancia entre ellas. Esto

contiene información útil para determinar la existencia de un golpe de mano.

Se calcularon dos conjuntos de características. El primero intentó describir el movimiento de mano y palo a lo largo del tiempo. Las características calculadas tanto para la mano como para el palo fueron: la distancia normalizada a la lonja, los máximos y mínimos de la velocidad vertical, y la cantidad de cruces por cero de dichas derivadas.

El segundo conjunto elegido estuvo motivado porque ciertos golpes se diferencian de otros por su comportamiento oscilatorio. Un claro ejemplo es la distinción entre un golpe de palo simple y uno rebotado: si bien en ambos casos el palo se encuentra cercano a la lonja, la diferencia radica en que en el segundo el palo golpea repetidas veces, por lo que es de esperarse que la posición de la punta oscile mientras dure el golpe. Fue así que se calcularon los primeros diez coeficientes de la Transformada Discreta de Coseno (DCT) de las posiciones verticales de la mano y la punta del palo.

Tercera etapa: selección de características y clasificación unimodal

La última etapa de trabajo sobre el modo video consistió en la selección de características. Para ello se utilizaron las mismas herramientas de selección que se usaron en el modo audio. Las características seleccionadas en esta etapa fueron: la distancia vertical y horizontal de la punta del palo al punto más bajo de la lonja, la distancia vertical del palo respecto al mismo punto de la lonja, mínimo de la velocidad vertical de la mano, cantidad de cruces por cero de la velocidad vertical de la punta del palo, los coeficientes número 2 y 3 de la DCT de la posición vertical de la mano y los primeros siete coeficientes de la DCT de la posición vertical del palo.

De igual forma que para el modo audio, se probaron distintos algoritmos de clasificación realizando una búsqueda exhaustiva de sus mejores parámetros. Los algoritmos de k-NN (considerando 3 vecinos) y SVM (con un kernel de base radial Gaussiana) fueron los que alcanzaron mayor desempeño, obteniéndose promedios en el entorno de 93 % y 90 % respectivamente (ver Sección 6.2.4). Nuevamente, estos promedios fueron obtenidos para el problema de seis clases, realizando 10 iteraciones con cada clasificador, utilizando validación cruzada en 10 particiones sorteadas aleatoriamente.

1.5.5. Sistema multimodal

Como se mencionó anteriormente, para el desarrollo del sistema multimodal se utilizó el enfoque de *Feature-Level Fusion*. Se implementaron dos métodos distintos para la combinación de las características de cada modo. El primero consistió en unir las características previamente seleccionadas de cada modo en un único conjunto de características, como se muestra en la Figura 1.9. De aquí en adelante se referirá a este método como Multimodal 1.

En el segundo método de combinación se unieron todas las características de cada modo en un único conjunto y se realizó selección sobre el mismo, de la misma

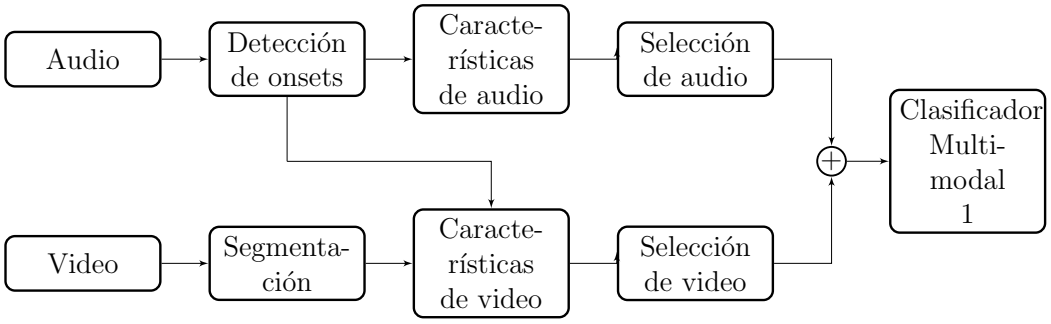


Figura 1.9: Diagrama de bloques del método Multimodal 1. El símbolo $\bigoplus$ representa la unión de conjuntos.

manera que se hizo para cada modo por separado. La Figura 1.10 muestra el diagrama de bloques para este método, que será referido como Multimodal 2.

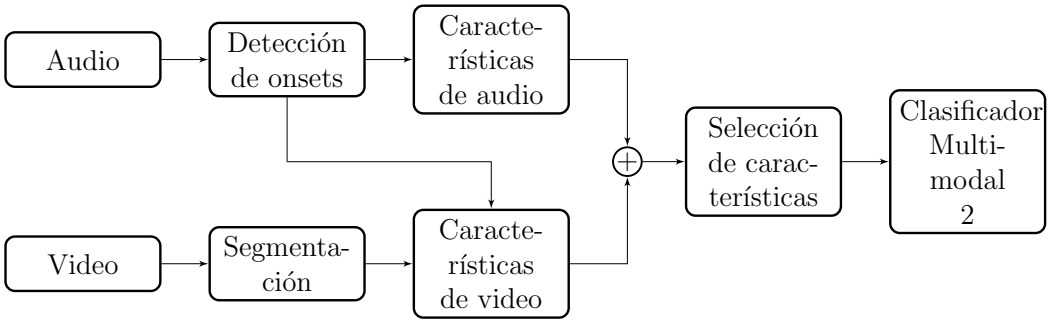


Figura 1.10: Diagrama de bloques del método Multimodal 2. El símbolo $\bigoplus$ representa la unión de conjuntos.

De manera de decidir cuál método resulta más útil para la solución del problema, se realizó la evaluación del desempeño de clasificación, utilizando como conjunto de entrenamiento los videos de palo rojo de la base eMe. Como conjuntos de prueba se usaron los videos de palo verde de la base eMe y el único video etiquetado de la base Zavala. Como clasificador se utilizó SVM con un kernel de base radial Gaussiana, ya que fue el que obtuvo mejores resultados para cada modo por separado. Se realizó una búsqueda exhaustiva de sus mejores parámetros, de manera análoga a lo que se hizo para los clasificadores unimodales. En base a los resultados obtenidos, se decidió utilizar el método Multimodal 1 como método final de fusión de características.

Para comparar las diferencias que existen entre utilizar la información de cada modo por separado y combinarla para formar un clasificador multimodal, se realizaron pruebas de desempeño sobre los videos de la base eMe. Se utilizaron los registros de tres intérpretes para entrenar, mientras que los registros del restante se usaron como conjunto de prueba. Se consideraron todas las variantes posibles de conjuntos entrenamiento/prueba.

Las pruebas se dividieron en dos grupos. En primera instancia se consideró la clasificación de tres tipos de golpes: madera, mano y palo. Luego, se agregaron otros

tres: borde, rebotado y flam. A partir de ellas pudo constatarse que la fusión de la información proveniente de cada modo produce mejores resultados que considerar los modos de forma aislada. Por ejemplo, estudiando el porcentaje de acierto en la clasificación de cada golpe (Tabla 1.1) se observa que el enfoque multimodal presenta en todos los casos un desempeño mayor.

Golpe	*Audio (%)*	*Video (%)*	*Multimodal (%)*
Madera	97,18	91,18	98,23
Mano	86,63	98,92	99,19
Palo	71,34	74,78	89,86
Rebotado	71,75	26,95	76,19
Borde	83,90	57,78	93,74
Flam	6,25	23,53	45,88

Tabla 1.1: Comparación del porcentaje de clasificación correcta de los modos audio y video frente al enfoque multimodal para cada tipo de golpe.

Finalmente, para estudiar el poder de generalización del sistema multimodal, una última prueba consistió en evaluar el desempeño del sistema frente a nuevos datos. Para eso se usó un registro de la base Zavala. A partir de dicha prueba, se constató que dos de las características seleccionadas no resultaban adecuadas para hacer el algoritmo más general, por tratarse de medidas dependientes de las condiciones de grabación. Además, se observó que el porcentaje de acierto en la clasificaciñon aumenta significativamente si esas características no son consideradas, por lo que se decidió no incluirlas del conjunto final.

1.6. Estructura de la documentación

Las características de cada dataset se desarrollan en el Capítulo 2, donde se incluye además la información de cómo se llevó a cabo el rodaje para la obtención del segundo conjunto de datos. También en dicha sección se encuentra una breve descripción del proceso de etiquetado de los golpes.

Los siguientes dos Capítulos presentan toda la información relacionada al procesamiento llevado a cabo sobre los dos modos de información utilizados: el 3 se refiere al audio y el 4 al video.

El Capítulo 5 presenta el marco teórico sobre el que se desarrollará la segunda parte del proyecto. Allí pueden encontrarse los conceptos básicos de reconocimiento de patrones que se utilizarán a lo largo de toda la documentación, como por ejemplo las definiciones de los distintos algoritmos de clasificación y los enfoques de selección de características utilizados. Además, se hace una breve introducción al procesamiento multimodal, brindando una reseña de la metodología usualmente empleada.

En el Capítulo 6 se desarrollan las pruebas llevadas a cabo para la selección de características en cada modo. También se presentan los enfoques de combinación

estudiados.

El Capítulo 7 presenta una evaluación del desempeño de la clasificación unimodal frente a la multimodal. Además se presenta una medida del poder de generalización del sistema multimodal, al estudiar su desempeño frente a un registro grabado en condiciones totalmente diferentes a los utilizados para su desarrollo.

El Capítulo 8 presenta las conclusiones extraídas del trabajo llevado a cabo durante el proyecto, además de discusiones sobre los resultados obtenidos y sobre el posible trabajo futuro que surge a partir de esta investigación.

En el Apéndice A se desarrolla toda la información respecto a los sensores estudiados, brindando además una fundamentaión de por qué no se consideró este modo de información en el proyecto.

Los dos Apéndices siguientes presentan dos algoritmos utilizados para el procesamiento de video. El B explica todo lo referente al funcionamiento del filtro de color. El C realiza una introducción básica al filtro de Kalman usado en la detección de la mano, explicando cómo fue configurado para la detección en este problema particular.

Los Apéndices D y E presentan el trabajo que se llevó a cabo en relación a la reconstrucción 3D. El primero versa sobre el algoritmo desarrollado para detección y seguimiento de ciertos puntos en el video. Fue desarrollado pensando en tener puntos correspondientes en las imágenes estéreo, lo que es un requisito indispensable para realizar la reconstrucción. El segundo trata sobre el proceso de reconstrucción 3D y brinda los fundamentos de por qué esta información no fue utilizada en el sistema desarrollado.

Finalmente, el Apéndice F brinda un panorama general del software entregado junto con el proyecto y cómo pretende ser utilizado, además de los requerimientos mínimos necesarios para su funcionamiento.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 2

Conjuntos de datos

Para la realización de este proyecto fue imprescindible contar con registros de audio y video sobre los cuales realizar el procesamiento de señales y posterior clasificación. Dado que hasta ese momento no existían registros disponibles con los requerimientos necesarios (por ejemplo, videos a suficiente tasa de muestreo), los datos fueron generados específicamente para el proyecto.

Se utilizaron dos conjuntos de datos (o *datasets*): uno cuya fecha de grabación es anterior al comienzo del presente proyecto (pero que fue grabado pensando en generar material para el mismo y que se llamará *base eMe*) y otro cuya grabación se realizó durante el marco del proyecto (*base Zavala*).

El hecho de llevar a cabo una instancia de grabación durante el proyecto implicó un importante trabajo previo. Se realizó una etapa de pre-producción en la cual se determinaron cambios respecto al registro anterior. Por ejemplo, se decidió utilizar cámaras que trabajaran con mayor tasa de cuadros o frame por segundo (*fps*) y cambiar la orientación de las mismas a una posición vertical. Dada la oportunidad única que se presentaba, se debió pensar en hacer el registro lo más completo posible, lo que llevó a que se obtuvieran más datos de los que aquí se utilizan. Por ejemplo, se utilizaron cámaras en configuración estéreo, lo cual puede ser de utilidad para el análisis de gestualidad o para realizar una reconstrucción 3D de la escena[1]. También se registró el toque de otros tambores: chico, piano y distintas configuraciones de cuerdas. La generación de estos datos significa una contribución directa del proyecto a investigaciones futuras, tanto en el área del procesamiento multimodal como en el área de la musicología.

2.1. Dataset eMe

El primer conjunto de datos utilizado fue el resultado de una sesión de grabación llevada a cabo en el Estudio de Música Electroacústica de la Escuela Universitaria de Música por Martín Rocamora y Haldo Spontón el Sábado 12 de Octubre

[1]También durante el rodaje se escanearon los tambores utilizando un Kinect de manera de tener más información para ajustar el modelo 3D. Álvaro Gómez fue el principal encargado de esta tarea.

de 2013. En dicha sesión se contó con la presencia de cuatro intérpretes diferentes: Ricardo Gómez, Juanita Fernández, Victoria Bonanata y Nestor Ferreira. De esta base se cuentan con dos interpretaciones de repique de cada intérprete etiquetadas.

El registro de audio se realizó utilizando micrófonos Neumann TLM 103 y un grabador Tascam HD-P2 Portable Stereo. Todo el audio se grabó a una frecuencia de muestreo de 44100 Hz, con 16 bits.

El registro de video fue realizado con tres cámaras distintas: una Canon 7D con resolución de 1280×720 grabando a 50 *fps*, una GoPro Hero2 (848×480, 120 *fps*) y una Panasonic FZ100 (320×240, 240 *fps*). El procesamiento de video en este proyecto se llevó a cabo con las tomas obtenidas de la GoPro, por considerarse que proporcionaba un buen compromiso entre resolución espacial y temporal.

2.2. Dataset Zavala

El segundo conjunto de datos proviene del registro realizado en el marco del proyecto, llevado a cabo el Sábado 6 de Setiembre de 2014 en la Sala Zavala Muniz del Teatro Solís. Se contó con la presencia de cinco de los intérpretes más destacados del género: Luis Giménez, Sergio Ortuño, Gustavo Oviedo, Fernando "Hurón"Silva y Héctor Manuel Suárez.

Este registro consistió en la grabación, tanto de audio como de video, de los distintos intérpretes. Se realizaron tomas de cada intérprete en solitario, así como en varias configuraciones de cuerdas (de a tres, cuatro y cinco tambores). Si bien en el proyecto solo se trató con el toque de repique, por las razones expuestas en el comienzo del presente capítulo, también se grabaron interpretaciones de los restantes tambores (chico y piano). Además, existió un registro de video con un objetivo documental, realizado por un equipo de cuatro personas con tres cámaras simultáneas, y un equipo de dos personas (cámara y sonido directo) que documentó el proceso de producción del registro, incluyendo entrevistas a los involucrados. Esto se fundamenta ya que la grabación se realizó en el marco de una línea de investigación que va más allá del proyecto de grado, y que involucra tanto a docentes del Instituto de Ingeniería Eléctrica como de la Escuela Universitaria de Música. Este último es el caso de Luis Jure, quien fue el responsable de la dirección musical del registro.

El registro de audio estuvo a cargo del equipo del Estudio de Música Electroacústica de la Escuela Universitaria de Música. Martín Rocamora y Juan Martín López fueron los que llevaron a cabo esta tarea.

El registro de video a alta tasa fue realizado por el grupo de trabajo responsable del presente proyecto. Un objetivo de esta nueva instancia de grabación fue obtener registros que permitan crear un modelo 3D tanto del tambor como de los movimientos del intérprete. Por lo tanto el registro fue realizado utilizando dos cámaras formando un par estéreo. Este punto se desarrolla en el Apéndice E.

2.2.1. Pre-producción

El registro presentó varios desafíos técnicos que hubo que resolver previo al día del rodaje, por lo que se llevó a cabo una etapa de pre-producción en donde se realizaron pruebas piloto y ensayos.

En primer lugar, para la grabación de video se necesitaron cámaras con una alta tasa de cuadros por segundo, por lo que se decidió emplear una versión mejorada de la utilizada para la base eMe (*GoPro Hero Black 3+*). Ésta es capaz de capturar video a una tasa de 240 cuadros por segundo con una resolución de 848×480 píxeles, pero presenta la desventaja de ser de óptica fija y de corta longitud focal, lo que no permite acercamientos y además introduce cierta distorsión en la imagen. Contar con un par estéreo significó buscar una forma de sincronizar la grabación en ambas cámaras. Esto pudo lograrse utilizando el control remoto que el fabricante provee con las mismas, que permite sincronizar dos o más cámaras vía WiFi, pero presenta como inconveniente que el consumo de energía por parte de cada cámara aumenta sustancialmente. Así, se decidió que éstas debían mantenerse conectadas a una fuente de energía durante todo el rodaje.

Se realizaron tomas de prueba de manera de determinar a qué distancia debían estar separadas las cámaras para asegurar suficientes puntos en común entre las tomas y tener alta precisión en profundidad. Con los objetos de interés ubicados a 1 metro de distancia, se concluyó que las cámaras debían estar separadas una distancia de 30 cm entre sí, de manera de obtener una resolución en profundidad menor a 1 cm en la reconstrucción 3D. Dicho cálculo se explica en el Anexo E.

En dichas pruebas se pretendió además evaluar el tamaño medio de los archivos que se generarían en la grabación y estimar la tasa de transferencia de archivos entre las cámaras y una computadora. A modo de ejemplo, una toma de 2:07 minutos resultó en un archivo de 520 MB. Por lo tanto, transferir directamente los archivos desde las cámaras a una computadora luego de cada toma consumiría una cantidad excesiva de tiempo. Se decidió entonces realizar la transferencia durante las pausas del rodaje, extrayendo manualmente las tarjetas SD de cada cámara. Esto implicó dos cosas: en primer lugar, contar con capacidad suficiente para almacenar varias tomas, lo cual se solucionó utilizando tarjetas de memoria de 32 GB. En segundo lugar, poder extraerlas de las cámaras alterando lo menos posible al par estéreo. Si bien existían soluciones comerciales de distinto tipo para el soporte de las cámaras, ninguna de ellas cumplía con todos los requisitos, por lo que fue necesario diseñar y construir un soporte a medida. Fue así que se recurrió al Taller del IIE solicitando la construcción de un soporte que permitiese regular la separación entre las cámaras, acceder a los puertos USB para la alimentación y extraer las tarjetas SD sin alterar el par estéreo. Además, se requirió un fácil acceso a los controles de cada cámara. Vale la pena destacar la labor realizada por los integrantes del Taller quienes realizaron este trabajo de forma expeditiva pero precisa. El soporte puede verse en la Figura 2.1.

También como consecuencia de estas pruebas se constató la necesidad de poder ver en tiempo real lo que graba cada cámara para asegurar un encuadre correcto, por lo que se decidió aprovechar la salida HDMI de las mismas para visualizar en un par de monitores la ubicación del intérprete en la toma.

Figura 2.1: Soporte utilizado para la configuración estéreo de las cámaras.

Analizando las tomas de prueba, se verificó también que la grabación del par estéreo no estaba perfectamente sincronizada: mediante la utilización de un cronómetro en una de las pruebas, se constató una diferencia de 25 frames aproximadamente entre las tomas de cada cámara. Así, se decidió que para el registro sería necesario utilizar una claqueta de manera de saber exactamente el desfasaje existente entre las tomas derecha e izquierda.

Para simplificar la reconstrucción 3D es necesario tener puntos fácilmente ubicables en las imágenes estéreo. Es por ello que se marcaron los tambores con marcadores de referencia o fiduciarios, es decir, pequeños tableros en blanco y negro que permiten ubicar el punto central con buena precisión. Las tomas de prueba fueron utilizadas para determinar la cantidad y ubicación de los marcadores a utilizar. En la figura 2.2 se ilustra la ubicación de estos marcadores en uno de los repiques utilizados en la grabación.

Figura 2.2: Ubicación de los marcadores en uno de los repiques utilizados en el registro.

Adicionalmente, se marcaron las zonas de interés del cuerpo del intérprete. Las zonas marcadas fueron el hombro, la articulación que une el brazo con el antebrazo y la parte anterior de la muñeca de ambos brazos. Además, en el brazo izquierdo también se ubicó un marcador sobre la parte posterior de la muñeca. Algunos frames de los videos grabados en estas pruebas piloto se utilizaron luego como referencia a la hora de colocar los marcadores a cada intérprete.

Al igual que lo realizado en la base eMe, se pintó de verde el borde de la lonja para facilitar su detección. En este caso, la pintura sirve para obtener fácilmente una estimación de la ubicación de la lonja, simplemente con un filtrado de color sobre la imagen. Sobre el resultado de este filtrado se aplica luego el algoritmo de detección de elipses.

En la figura 2.3 se muestra un ejemplo de imagen estéreo obtenida en el registro, donde puede verse la ubicación de los marcadores utilizados.

Figura 2.3: Imagen estéreo de un intérprete (Sergio Ortuño) durante el registro, con los marcadores utilizados.

En dicha imagen se ve que también se agregaron marcadores en el piso de la sala. Esto fue hecho de manera de tener puntos adicionales en los videos a partir de los cuales extraer información extra de profundidad.

Los palos con los que tocaron los intérpretes también se modificaron para la ocasión. Dado que el color de la lonja ya se encontraba marcada de color verde,

se decidió pintar el palo totalmente de rojo, de forma de facilitar su detección. De esta manera se puede filtrar la imagen con el color correspondiente al elemento que se desea detectar.

Otra variable a considerar fue la iluminación de la sala: grabando a 240 cuadros por segundo, se observó que los videos resultantes de las pruebas preliminares (grabados en el eMe) presentaban, cuadro a cuadro, una alta variación en la iluminación debido a las fluctuaciones en la tensión que alimentaba los equipos de alumbrado, fenómeno comúnmente denominado *flicker*. Es por ello que se llevaron a cabo algunas pruebas para establecer experimentalmente el grado de *flicker* generado por las luces de la sala. Dichas pruebas consistieron en la grabación de video a 240 cuadros por segundo, con distintas configuraciones de iluminación en la sala. Observando los videos obtenidos se pudo descartar que hubieran problemas entre el equipamiento del teatro y la tasa de adquisición seleccionada para el registro. A su vez, se determinó el nivel de iluminación con un medidor de lúmenes. Las pruebas determinaron un nivel de aproximadamente 100 lúmenes, con lámparas incandescentes, que fueron las utilizadas luego en la grabación.

La etapa de pre-producción finalizó con la escritura de un protocolo de tareas para el registro, definiendo los roles de cada integrante del grupo y delineando las estrategias de respaldo (cámaras y baterías de repuesto, suficientes tarjetas) y procedimientos en caso de falla.

2.2.2. Registro de video

Para la grabación de audio se utilizó una grabadora SoundDevices 788 y micrófonos en diferentes configuraciones, a una frecuencia de muestreo de 44.100 Hz con 16 bits. Se situaron micrófonos Sennheiser direccionales cercanos a los tambores, de forma tener un registro independiente de cada tambor, que fueron las grabaciones de audio utilizadas en este proyecto para la generación de características. Además, se colocaron a mayor distancia dos micrófonos Schoeps (de tipo cardioide y figura 8) en configuración centro-laterales (*mid-side*) y dos AKG omnidireccionales, con el objetivo de obtener un sonido que aproveche la acústica de la sala para fines documentales.

Se registraron 51 interpretaciones de candombe, comenzando por tomas con un tambor, para luego realizar grabaciones de cuerdas de tres, cuatro y hasta cinco integrantes. En la Tabla 2.1 se muestra en detalle la cantidad de tomas realizadas en cada modalidad.

Tambores Solos			*Cuerdas*		
Piano	*Chico*	*Repique*	*De 3*	*De 4*	*De 5*
11	12	14	9	3	2

Tabla 2.1: Desglose de las tomas realizadas en el registro del dataset Zavala.

Dentro de las tomas con un solo tambor, además de registrar toques usuales, se

grabaron algunas en las que los intérpretes daban solamente golpes aislados. Éstas tomas fueron pensadas para varios objetivos. Por ejemplo, pueden ser de utilidad para analizar los golpes separadamente y así entrenar un clasificador. Se podría también estudiar más detenidamente la gestualidad de cada tocador y ver cómo las distintas técnicas afectan al sonido de un mismo golpe. Además, pueden servir como material de trabajo para síntesis digital.

Una de las grabaciones de la cuerda integrada por los cinco intérpretes se registró utilizando tambores sin marcar, que fueron llevados por los mismos intérpretes. Esta toma se realizó para tener una grabación inalterada que complementase el registro documental, pero también puede ser de utilidad para desarrollos posteriores que no necesiten del uso de marcadores. Además, posibilita tener un registro de audio con los tambores propios de los intérpretes, que son los que utilizan regularmente.

En la Figura 2.4 se muestra la ubicación de las cámaras y los micrófonos para el rodaje, según lo planificado en las pruebas anteriores.

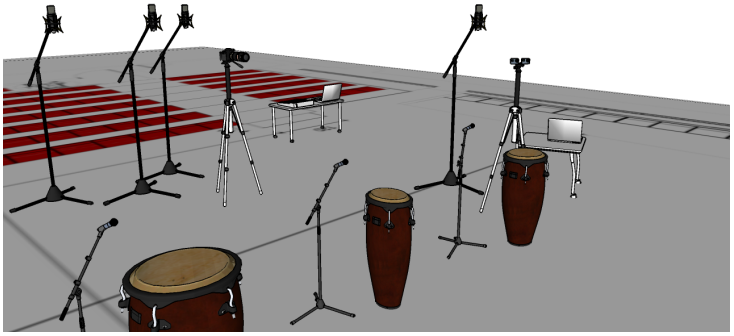


Figura 2.4: Ubicación de las cámaras y los micrófonos para el rodaje, según lo planificado en las pruebas anteriores al mismo.

Respecto a las decisiones tomadas en la pre-producción, es de destacar el hecho de que la previa asignación de roles dentro del grupo contribuyó a que la grabación se pudiera realizar de manera fluida y sin mayores contratiempos. Analizando las tomas realizadas, se vió que se logró un encuadre correcto para los propósitos del proyecto, ya que el intérprete siempre se mantiene en cuadro y los marcadores pueden distinguirse razonablemente bien. Cabe aclarar que existen momentos en los que el marcador ubicado en el brazo izquierdo queda oculto por el propio brazo, como muestra la Figura 2.5.

2.3. Etiquetado y sincronización

Para realizar un sistema de clasificación automática supervisada, deben contarse con datos *etiquetados*. Esto es, deben tenerse datos en los que se sabe a ciencia

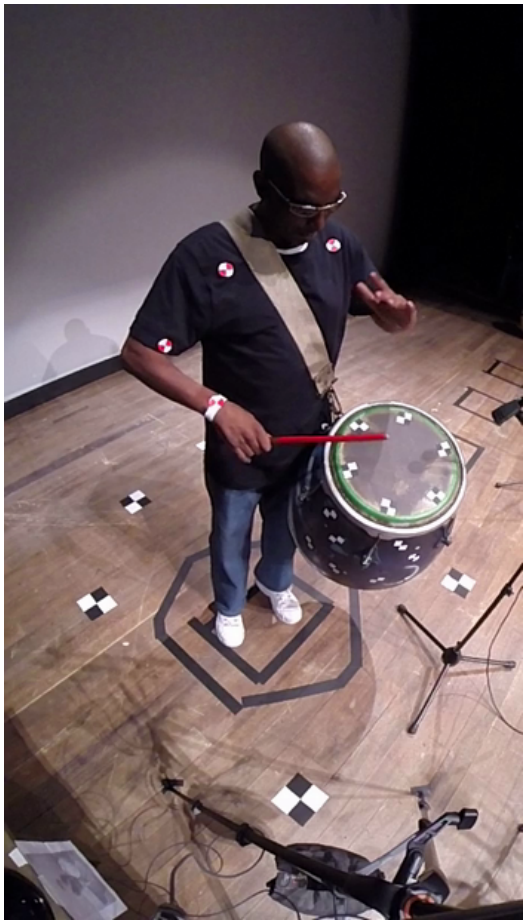


Figura 2.5: Oclusión del marcador en el brazo izquierdo durante una de las tomas del registro.

cierta a qué clase pertenecen. Para este proyecto en particular, esto implica dos cosas: conocer la ubicación temporal de los golpes en el audio y determinar qué clase de golpe son (mano, palo, madera, flam, rebotado o borde).

Para ello, a partir del audio, se extrajeron los tiempos de los golpes de forma automática mediante una detección con el *spectral flux* [38], para luego etiquetarlos a mano. Dicho etiquetado fue llevado a cabo por Luis Jure y Martín Rocamora. Por más información sobre la detección de eventos en el audio, ver Capítulo 3, Sección 3.1.

La sincronización entre el audio y el video fue realizada con la ayuda de la claqueta utilizada en la grabación. Se ubicó en cada registro la posición del golpe de claqueta, simplemente escuchando el audio u observando el video. Usando esta información, se cortaron las grabaciones de manera tal que, en cada toma, los registros de audio y video correspondientes estuviesen en la misma base de tiempos. Para ello se utilizaron las etiquetas: las tomas se cortaron de forma que comenzaran 2 segundos antes del primer golpe etiquetado y finalizaran 2 segundos después del último golpe.

Parte I

Procesamiento de señales

Capítulo 3

Procesamiento de audio

Las señales de audio de instrumentos de percusión presentan la dificultad de no tener información tonal bien definida, con lo cual resulta difícil determinar qué tipo de golpe fue realizado ya que se debe discriminar según el timbre del mismo. Se entiende por timbre el atributo de un sonido que permite diferenciarlo respecto a otro con igual intensidad, altura (o frecuencia) y duración [6]. Identificar un sonido de altura definida como el producido por un piano o una guitarra es un problema más sencillo que la identificación de timbre, fundamentalmente cuando los timbres son similares. Un golpe de mano sobre la lonja del tambor o un golpe de palo en lonja o madera tienen características tímbricas diferentes, pero dependiendo de la realización particular del golpe o la duración del mismo puede ser más o menos complicado lograr una buena determinación de sus diferencias.

En este capítulo se expone el proceso que se llevó a cabo con el fin de encontrar descriptores de la señal de audio que resultasen útiles para discriminar entre distintos tipos de golpes, independientemente de las realizaciones particulares de los mismos.

3.1. Detección de eventos y etiquetado

Como se mencionó en la Sección 2, los golpes se ubicaron temporalmente utilizando la señal de audio. En la bibliografía, este procedimiento se conoce como detección de eventos u *onsets*. En realidad, el término *onset* hace referencia al comienzo de un evento musical de audio [59], pero dada la corta duración de los eventos en una grabación de percusión, la ubicación temporal de un golpe coincidirá con el onset más cercano.

El enfoque aquí implementado utiliza el flujo espectral (*spectral flux*) de la señal de audio. El mismo puede pensarse como una medida de los cambios que se producen localmente en el espectro de la señal [57]. Su definición matemática es:

$$\mathrm{SF}(n) = \frac{\sqrt{\sum_{k=0}^{N/2-1} \mathrm{HWR}(|X(k,n)| - |X(k,n-1)|)^2}}{N/2},$$

donde $X(k, n)$ representa la k-ésima componente de la Transformada Discreta de Fourier de la n-ésima trama de audio, N es la cantidad de muestras en la trama y $\mathrm{HWR}(x)$ es la rectificación de media onda de la señal:

$$\mathrm{HWR}(x) = \frac{x + |x|}{2}.$$

En este caso, como se usaron tramas de 20 ms (con un salto de 10 ms) se tiene que $N = 20\,\mathrm{ms} * f_s$, siendo f_s la frecuencia de muestreo de la señal de audio. Por ejemplo, una f_s de 44.100 Hz corresponde a 882 muestras.

La idea de aplicar rectificación de media onda es considerar sólo las diferencias de magnitud del espectro cuando sean positivas, es decir, sólo cuando haya un aumento en la energía espectral. Dado que un golpe de percusión representa un gran aumento de energía en un período corto de tiempo, se manifestará en el spectral flux como un máximo local. Por lo tanto, la detección de eventos en el audio consistió en hallar todos los máximos locales del spectral flux. Se consideraron como candidatos a golpes sólo aquellos máximos locales que alcanzaran una amplitud de al menos el 10 % de la amplitud máxima, dado que se asume que todo golpe debe causar un aumento de energía considerable en el espectro de la señal.

Sin embargo, no todos los máximos locales que cumplan esta condición indican necesariamente la presencia de un golpe. Por ejemplo, un rebotado es en realidad una sucesión de golpes de palo ejecutados en un pequeño intervalo de tiempo. Cada uno de estos golpes generará un máximo local en el spectral flux, como se ve en la Figura 3.1. En ese caso, la ubicación temporal del golpe deberá coincidir

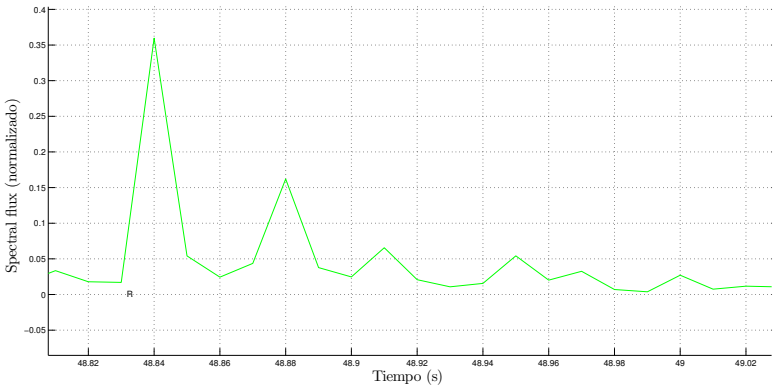


Figura 3.1: Spectral flux (normalizado) de un golpe rebotado. El máximo de mayor amplitud indica el comienzo del golpe, mientras que los máximos siguientes son causados por los rebotes en la lonja.

con el comienzo de la ejecución del mismo, es decir, con la ubicación del primer máximo local, debiéndose descartar todos los máximos siguientes correspondientes a los rebotes.

El etiquetado de los golpes se hizo manualmente, lo que consiste en establecer el tipo de golpe y su ubicación. El proceso manual estuvo guiado por la detección

automática de onsets. En este proceso fue necesario ignorar los diferentes máximos que corresponden a un mismo golpe, conservando sólo uno, además de agregar la posición de aquellos golpes que no se detectan. Martín Rocamora y Luis Jure fueron los que llevaron a cabo dicha tarea, basándose en la escucha de los audios registrados.

Las estadísticas del etiquetado se muestran en la Tabla 3.1. Allí puede verse qué porcentaje de eventos detectados corresponden efectivamente a golpes y qué porcentaje de los golpes no son detectados por este método. Estos porcentajes surgen de comparar la cantidad de golpes en las etiquetas contra la cantidad de máximos detectados en el spectral flux.

TruePositives	*FalseNegatives*
96,20 %	1,28 %

Tabla 3.1: Estadísticas del proceso de etiquetado. *TruePositives* indica el porcentaje de eventos detectados en el flujo espectral que efectivamente corresponden a golpes, mientras que *FalseNegatives* es el porcentaje de golpes que no son detectados por este método.

3.2. Extracción de características

El timbre de un instrumento está determinado en gran medida por la distribución de energía en el espectro. Por esta razón, en la clasificación de instrumentos musicales es habitual el uso de características que describen el contenido espectral de un sonido [65,70]. Por lo tanto, se decidió centrar el estudio de las características de audio en aquellas derivadas del espectro.

3.2.1. Determinación de características a utilizar

Para la determinación de las características a utilizar se eligieron tres conjuntos. Los primeros dos son extensamente utilizados en el estudio de señales de audio.

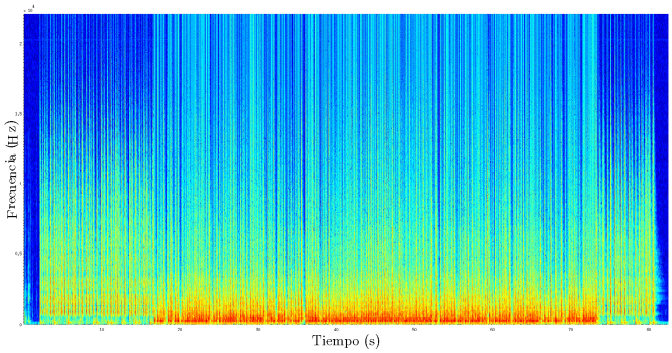
Por un lado, se calcularon características que consideran a la envolvente del espectro de la señal como una distribución de probabilidad y calculan medidas que describen su forma, por ejemplo los primeros momentos estadísticos.

El segundo conjunto consistió en el cálculo de los coeficientes cepstrales de frecuencias mel de la señal (MFCCs por sus siglas en inglés). Los MFCCs son tal vez las características más usadas para la descripción de timbre [32, 44, 63].

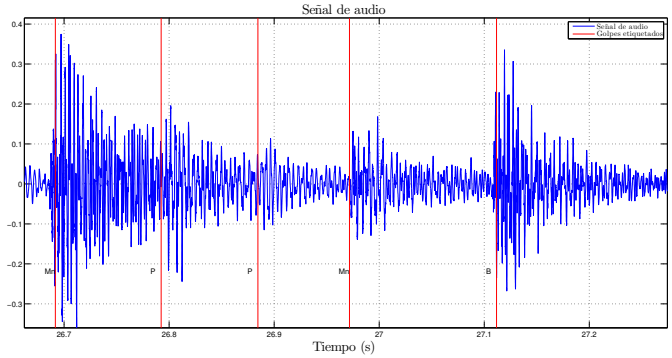
El tercer conjunto elegido estuvo motivado porque un golpe de percusión representa una gran concentración de energía en un período corto de tiempo. Así, se manifestará en el spectral flux como un máximo local. Dado que algunos golpes son en realidad una sucesión de golpes en un pequeño intervalo de tiempo, resultarán en una sucesión de máximos locales en el spectral flux. Las características calculadas intentan reflejar esta realidad.

Para la generación de características del audio se debieron determinar parámetros tales como la frecuencia máxima de observacion f_{max}, el ancho de ventana con

el que se calculó el espectro y el salto entre ventanas. Los valores elegidos fueron una $f_{max} = 5\,\text{kHz}$ y un ancho de ventana de 90 ms. La elección de dichos valores puede justificarse observando la Figura 3.2(a), donde se puede ver que el mayor contenido de energía de la señal se encuentra por debajo de los 5 kHz. Fijar una f_{max} de 5 kHz asegura que no se está computando información poco aprovechable (lo cual reduce los costos computacionales) y se está descartando información como ruido ambiente, entre otros. Por otro lado, al observar la señal de audio con golpes de palo y mano (Figura 3.2(b)), puede verse que la duración de los mismas está en el orden de los 100 ms, con lo cual tomar una ventana de 90 ms de duración asegura un buen compromiso entre resolución espectral y temporal para este caso.



(a) Espectrograma de una señal típica de toque de repique.



(b) Espaciamiento temporal de los eventos en el audio. **Mn** representa un golpe de mano, **P** uno de palo y **B** uno de borde.

Figura 3.2: Señal de audio típica de toque de repique

3.2.2. Primer conjunto de características derivadas del audio

El primer conjunto estuvo compuesto por características que consideran al espectro de la señal como una distribución de probabilidad y calculan medidas

que describen su forma. En todos los casos las definiciones fueron extraídas de [59] y se presentan a continuación. La extracción de características en este caso se realizó centrando la ventana de 90 ms en el instante del golpe.

Centroide espectral (*spectral centroid*)

El *spectral centroid* representa el centro de masa (primer momento) de la energía espectral. Se define de la siguiente manera:

$$v_{SC}(n) = \frac{\sum_{k=0}^{N/2-1} k.|X(k,n)|^2}{\sum_{k=0}^{N/2-1} |X(k,n)|^2}.$$

El resultado es un índice correspondiente a un bin de frecuencia, en el rango $0 \leq v_{SC} \leq \frac{N}{2} - 1$. Valores bajos indican que predominan los componentes de bajas frecuencias, poca cantidad en alta frecuencia, y poco brillo [59].

Extensión espectral (*spectral spread*)

El *sprectral spread* se puede interpretar como la desviación estándar de la potencia del espectro alrededor del spectral centroid. Su definición matemática es:

$$v_{SS}(n) = \sqrt{\frac{\sum_{k=0}^{N/2-1} (k - v_{SC}(n))^2.|X(k,n)|^2}{\sum_{k=0}^{N/2-1} |X(k,n)|^2}}.$$

El resultado es un bin de frecuencia en el rango de $0 \leq v_{SS} \leq \frac{N}{4}$. Valores bajos indican una concentración de la energía en una región específica.

Oblicuidad espectral (*spectral skewness*)

El *skewness* de una variable se define como el tercer momento central de la misma dividida entre el cubo de su desviación estándar. Es una medida de la simetría que tiene la función de densidad de probabilidad de las muestras [59]. Es cero para distribuciones simétricas, negativa para distribuciones con su masa centrada a la derecha y positiva para distribuciones con su masa centrada a la izquierda. En el caso que la variable considerada sea el espectro de la señal, se define el *spectral skewness* de la siguiente manera:

$$v_{SSk}(n) = \frac{2 \sum_{k=0}^{N/2-1} (|X(k,n)| - \mu_{|X|})^3}{N\sigma_{|X|}^3}$$

donde $\mu_{|X|}$ es la media aritmética del módulo de la DFT de la señal y $\sigma_{|X|}^2$ su varianza.

Curtosis espectral (*spectral kurtosis*)

El *kurtosis* se define como el cuarto momento de una variable, dividida entre su desviación estándar elevada a la 4. Es una medida de lo alejada que está una densidad de probabilidad de ser una distribución Gaussiana [59]. Puede interpretarse como una medida de qué tan plana es una señal en un entorno de su media. Nuevamente, si la variable considerada es el espectro de la señal, el *spectral kurtosis* se define como:

$$v_k(n) = \frac{2 \sum_{k=0}^{N/2-1} (|X(k,n)| - \mu_{|X|})^4}{N\sigma_{|X|}^4} - 3\,.$$

Disminución espectral (*spectral decrease*)

El *spectral decrease* es un estimador de la pendiente de la parte decreciente de la envolvente del espectro de una señal. Se define como:

$$v_{SD}(n) = \frac{\sum_{k=1}^{N/2-1} \frac{1}{k}.(|X(k,n)| - |X(0,n)|)}{\sum_{k=1}^{N/2-1} |X(k,n)|}\,.$$

El resultado es siempre menor o igual a 1.

Bajada espectral (*spectral slope*)

El *spectral slope* es una característica similar al spectral decrease en el sentido de que mide la pendiente del espectro. Se calcula utilizando una aproximación lineal del módulo del mismo. La pendiente es estimada mediante la siguiente ecuación:

$$v_{SSI}(n) = \frac{N \sum_{k=0}^{N/2-1} k|X(k,n)| - \left(\sum_{k=0}^{N/2-1} k\right)\left(\sum_{k=0}^{N/2-1} |X(k,n)|\right)}{N \sum_{k=0}^{N/2-1} k^2 - \left(\sum_{k=0}^{N/2-1} k\right)^2}\,.$$

Cresta espectral (*spectral crest*)

El *spectral crest* mide la amplitud máxima del espectro de la señal respecto de su media. Su ecuación matemática es la siguiente:

$$v_{SCr}(n) = \frac{\max_{0 \le k \le N/2-1} |X(k,n)|}{\sum_{k=0}^{N/2-1} |X(k,n)|}\,.$$

Es una medida de la cantidad de componentes tonales que tiene la señal [59], e indica si la misma presenta o no crestas significativas.

3.2.3. Segundo conjunto de características derivadas del audio

Con respecto al segundo conjunto, una forma de describir la distribución de energía en el espectro podría ser tomando directamente la DFT de la señal como característica. Esto implicaría un costo computacional muy grande (pues se deberían manejar vectores de características de tanta cantidad de elementos como puntos tenga la DFT) y asimismo una mayor complejidad de entrenamiento. Además, cualquier cambio mínimo en las condiciones en las cuales se realizaron las grabaciones (por ejemplo un cambio en la afinación de la lonja) significaría que información relevante que se encuentra a cierta frecuencia varíe de posición para cada toma. Esto implicaría que las características que computen dicha información varíen de una toma a otra, provocando que el clasificador deba aprender un intervalo de características óptimas en lugar de identificar que siempre se trata de la misma. Sin embargo, existen otros métodos que sustituyen a la DFT de manera de concentrar la información en menos bins y de esta manera evitar este tipo de confusiones.

En este proyecto se utilizaron los *coeficientes cepstrales de frecuencias mel (Mel Frequency Cepstral Coefficients, MFCCs)*, ya que es un método ampliamente utilizado para la descripción del timbre y reconocido por dar buenos resultados [32, 44, 63]. El mismo se basa en caracterizar el espectro de la señal mediante su envolvente, lo que resulta beneficioso, ya que da suficiente información acerca de su timbre en una menor cantidad de información.

Para el cálculo de los MFCCs se utilizó una ventana de 90 ms con un salto de 45 ms. En este caso se procesó la señal completa. A cada evento se le asignan los MFCCs calculados en la ventana de centro más próximo a la ocurrencia del mismo.

Mel Frecuency Cepstral Coefficients (MFCCs)

El término *cepstral* hace referencia al *cepstrum* de una señal, definido como la transformada inversa de Fourier del logaritmo de $|X(f)|$ siendo $X(f)$ la transformada de Fourier de la señal de audio:

$$\mathrm{Cep}(\tau) = \int_{-\infty}^{+\infty} \log(|X(f)|) e^{j2\pi f \tau} df.$$

El filtrado de una señal equivale a una convolución en el dominio del tiempo (de la señal con la respuesta al impulso del filtro) y a una multiplicación en el dominio de la frecuencia. En este sentido, la motivación del cepstrum es transformar dicha multipicación en una suma, al aplicar logaritmo a la Transformada de Fourier [57].

En el caso que la señal sea discreta, el cepstrum de la señal también será discreto, por lo que estará determinado por los coeficientes de la inversa de la Transformada Discreta de Fourier (DFT). En la práctica se utilizan los *coeficientes cepstrales de predicción lineal (LPCCs)* o los *coeficientes cepstrales de frecuencias mel (MFCCs)*. Los primeros son calculados utilizando un modelo de predicción lineal, mientras que los segundos son computados a través de un banco de filtros mel.

El cálculo de los coeficientes cepstrales a través de los filtros mel tiene la particularidad de que las frecuencias centrales de cada filtro están equiespaciadas en la escala mel de frecuencias. Esta se relaciona con la escala lineal estándar de frecuencias mediante la siguiente relación [80]:

$$m = 2595 \log_{10} \left(\frac{f}{700} + 1 \right)$$

siendo f la frecuencia en la escala usual y m la frecuencia en escala mel. La escala mel fue diseñada con el objetivo de emular la forma en que el oído humano percibe el tono de una señal de audio. El American National Standards Institute define tono como el atributo perceptual que permite el ordenamiento de sonidos desde bajos a altos en una escala de frecuencias [81]. El espaciamiento de tonos en la escala mel es tal que coincide con la percepción auditiva. Por ejemplo, el tono de un sonido de 500 mels es percibido como la mitad del tono de uno de 1000 mels, mientras que el tono de uno de 2000 mels se percibe como el doble.

Cada filtro del banco tiene forma triangular, con ganancia unidad en su frecuencia central y con sus otros dos vértices ubicados en la frecuencia central de los filtros adyacentes.

Para computar los MFCCs, la señal es enventanada en tramas. Cada trama es filtrada con un banco de K filtros mel. Se toma el módulo de la salida de cada filtro, se eleva al cuadrado y se suma sobre todo el rango de frecuencias del filtro. Las sumas de cada filtro son unidas en un único vector de dimensión K. Luego, se toma el logaritmo de dicho vector y se transforma de nuevo al dominio temporal utilizando la Transformada Discreta de Coseno (DCT). La misma se define, para una señal discreta $x[k]$ de largo N, como:

$$\mathrm{DCT}_x[i] = \sum_{k=1}^{N} x[k] \cos \left[\frac{\pi}{N} i \left(n - \frac{1}{2} \right) \right].$$

Además de volver al dominio temporal, la DCT tiene la ventaja que decorrelaciona los coeficientes en una manera similar a PCA [62], logrando obtener distinta información en cada coeficiente MFCC. Las diferentes etapas del cálculo de los MFCCs se muestran en la Figura 3.3.

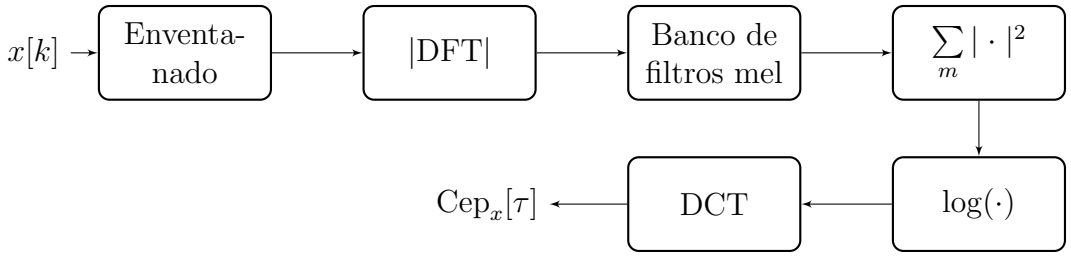


Figura 3.3: Diagrama de bloques del proceso de cálculo de los MFCCs

En este caso se calcularon los MFCCs dividiendo la señal en tramas de 90 ms, con un salto de 45 ms. La cantidad de filtros mel en el banco se fijó en 160. Se utilizaron como características los 40 primeros MFFCs de la señal enventanada, ya que concentran la mayor cantidad de información de la señal.

3.2.4. Tercer conjunto de características derivadas del audio

Dado que un golpe de percusión representa una gran concentración de energía en un período corto de tiempo, éste se manifestará en el spectral flux como un máximo local. Como ciertos golpes (por ejemplo el flam o el rebotado) son en realidad una sucesión de golpes en un pequeño intervalo de tiempo, se caracterizan por presentar una sucesión de máximos locales en el spectral flux, donde el primer máximo es el de mayor amplitud (ver Figura 3.1). Por lo tanto, se propusieron dos características que intentan reflejar esta realidad: la cantidad de máximos del spectral flux en una ventana de tiempo centrada en el evento de audio (que se notará como sf_1) y la diferencia de alturas entre el primer y segundo máximo (cuando éste no existe, la característica vale cero). Esta característica será referida de ahora en más como sf_2. Ambas características se calcularon con una ventana de 90 ms a partir del instante del golpe.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 4

Procesamiento de video

Para poder analizar la escena y extraer características que permitan un estudio profundo de la misma, es necesario poder detectar automáticamente los elementos más importantes que en ella aparecen. Éstos son la lonja, el palo y la mano del intérprete. Para simplificar su detección, se realizaron algunas intervenciones mínimas a la hora de realizar los registros, teniendo el cuidado de no modificar la sonoridad de los tambores ni incomodar al intérprete. Se pintaron el palo y la circunferencia de la lonja, para asistir mediante un filtrado de color a los distintos algoritmos de segmentación. A su vez, como se dijo en el Capítulo 2, se agregaron marcadores al tambor, los brazos del intérprete y el piso. Esto fue pensado para tener puntos fácilmente reconocibles entre ambas cámaras y utilizar dicha información para realizar una reconstrucción 3D, enfoque que finalmente no se incluyó en la solución. Este punto se desarrolla en el Apéndice E.

A continuación se explican las herramientas desarrolladas para realizar la segmentación en las distintas bases de datos y el posterior procesado de estos datos para la determinación de características de video. En la Figura 4.1 puede verse el diagrama de bloques del procesamiento realizado en el video.

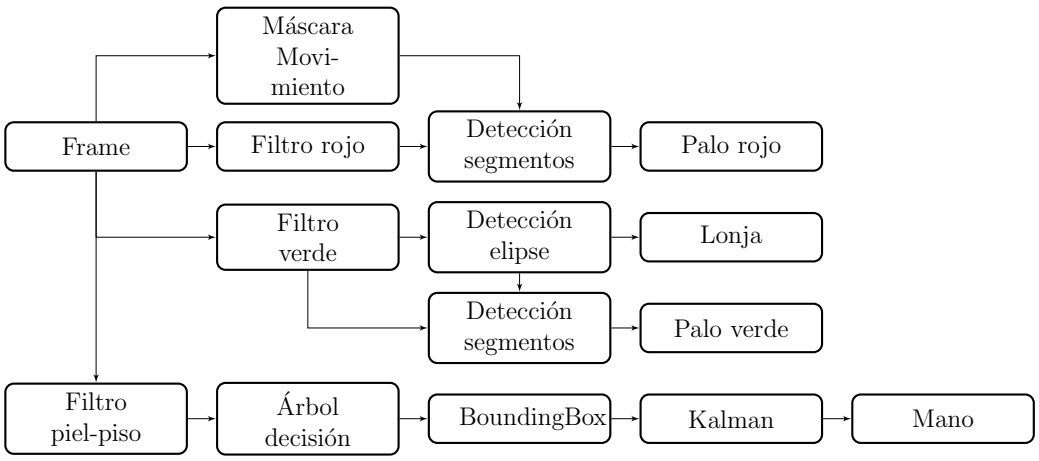


Figura 4.1: Diagrama de bloques de la segmentación de objetos en el video.

4.1. Segmentación de la lonja

Si bien la lonja del tambor es aproximadamente un círculo, ésta se transforma en una elipse debido a la perspectiva de la cámara. El problema de detección de elipses en imágenes es un problema clásico [36,43,85,90]. En general, este proceso se divide en dos etapas: una de detección de bordes en la imagen y otra de estimación de elipses sobre la imagen obtenida en la primera etapa.

El objetivo de la primera etapa es obtener una imagen binaria en la que sólo sean puntos de primer plano aquellos que representen un borde en la imagen. Esto se debe a que, en general, las elipses que aparecen en una imagen son el borde de algún objeto en ella presente.

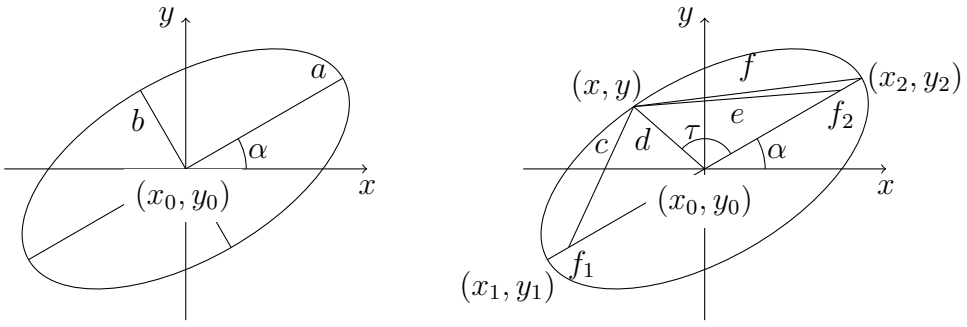
En cuanto a la segunda etapa, la mayoría de los algoritmos existentes se basan en la transformada de Hough [36, 85, 90]. Si bien este tipo de aproximación ha sido muy estudiada, es intensiva computacionalmente, ya que se requiere un acumulador que tenga tantas dimensiones como parámetros tenga la curva a detectar (en el caso de la elipse, cinco).

En primer lugar se implementó un algoritmo que utiliza el enfoque planteado en [89]. El mismo utiliza un acumulador unidimensional, reduciendo así el costo computacional involucrado. Luego se implementó un segundo algoritmo utilizando funciones de OpenCV [15].

4.1.1. Primer algoritmo: detección utilizando un acumulador unidimensional

Dado un conjunto de puntos de un plano (puntos de borde), el problema consiste en encontrar la elipse que mejor ajusta dichos puntos. Esto requiere determinar los cinco parámetros de dicha elipse: eje mayor a, eje menor b, coordenadas del centro (x_0, y_0) y ángulo de rotación α. En la Figura 4.2(a) se muestra una representación gráfica de una elipse y sus parámetros.

En [89], los autores asumen que cada par de píxeles de borde (x_1, y_1) y (x_2, y_2)



(a) Parámetros de una elipse.

(b) Geometría de la elipse.

Figura 4.2: Características de la elipse

son los vértices del eje mayor de una elipse. Luego se calculan los parámetros correspondientes a esa elipse:

$$\begin{aligned} x_0 &= \frac{x_1 + x_2}{2} \\ y_0 &= \frac{y_1 + y_2}{2} \\ a &= \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2} \\ \alpha &= \arctan\left(\frac{y_2 - y_1}{x_2 - x_1}\right). \end{aligned}$$

Si se analiza la geometría de la elipse (Figura 4.2(b)) puede verse que, de contar con un tercer punto -de coordenadas (x, y)- perteneciente a la elipse, puede calcularse el eje menor:

$$\left(\frac{b}{2}\right)^2 = \frac{\left(\frac{a}{2}\right)^2 d^2 \sin^2(\tau)}{\left(\frac{a}{2}\right)^2 - d^2\cos^2(\tau)} \Rightarrow b = 2\sqrt{\frac{\left(\frac{a}{2}\right)^2 d^2 \sin^2(\tau)}{\left(\frac{a}{2}\right)^2 - d^2\cos^2(\tau)}}$$

con

$$\begin{aligned} d &= \sqrt{(x - x_0)^2 + (y - y_0)^2} \\ \cos(\tau) &= \frac{a^2 + d^2 - f^2}{2ad}. \end{aligned}$$

Así, una vez fijado un par de píxeles (x_1, y_1) y (x_2, y_2), para el resto de los píxeles se realiza una votación, donde cada uno vota sobre un candidato a eje menor, obteniendo de esta manera un acumulador unidimensional. Este procedimiento se repite para cada par de puntos de toda la imagen.

Este algoritmo no sólo disminuye la dimensionalidad de la matriz de acumulación respecto a aquellos que utilizan la Transformada de Hough, sino que también permite imponer restricciones geométricas para hacerlo aún más eficiente. Por ejemplo, una vez determinado el centro y el eje mayor a de una elipse, sólo los puntos que se encuentren a una distancia menor que $\frac{a}{2}$ del centro podrán pertenecer a la misma. Imponiendo esta restricción, el proceso de votación se lleva a cabo sobre un subconjunto de píxeles, reduciendo así el tiempo de cómputo. Si además se tiene alguna estimación de las dimensiones de la elipse a detectar, puede definirse un rango de variación para a de manera tal que los pares de puntos cuya distancia caiga fuera de este rango no sean considerados como candidatos a vértices del eje mayor. De igual manera, puede imponerse un rango de variación para el ángulo de rotación α a considerar, o para la relación de aspecto de la elipse[1].

Implementación

Como fue mencionado anteriormente, para detectar una elipse en la imagen (en este caso la correspondiente a la lonja) debe contarse con una imagen binaria.

[1] Cociente entre la longitud del eje menor y la del eje mayor de la elipse, $\frac{b}{a}$.

Para ello, usualmente se aplica sobre la imagen donde se quiere ubicar la elipse una detección de bordes. En este proyecto, antes de la detección de bordes se aplicó el filtro de color explicado en el Apéndice B para simplificar la búsqueda, ya que en ambas bases de datos los intérpretes fueron grabados tocando con un repique en el cual el borde de la lonja se encontraba pintado de verde.

Finalmente, se aplicó el algoritmo de detección de elipses. La implementación del mismo se obtuvo de Matlab Central [7].

Se determinaron las consideraciones geométricas a las que se hizo referencia en la sección anterior midiendo las dimensiones de la lonja en distintas imágenes, como se muestra en la Figura 4.3. Así, se decidió considerar un rango de variación para a de entre 100 y 200 píxeles. Para el ángulo α, se impuso que el algoritmo tomase en cuenta sólo aquellas elipses que verificasen $\alpha \in [-20^o, 20^o]$. En este caso no se impusieron restricciones sobre la relación de aspecto.

Figura 4.3: Medidas realizadas para estimar la longitud del eje mayor de la elipse. Las distancias están medidas en pixeles.

Además, para que el algoritmo sea aún menos exhaustivo computacionalmente, es posible sortear al azar un subconjunto de los puntos de la imagen de borde y luego aplicar a éstos la detección de elipses. En este caso, se sortearon $N/2$ puntos, siendo N la cantidad de puntos de primer plano en la imagen de borde.

En la Tabla 4.1 se muestra cómo disminuye la cantidad de ejes mayores considerados cuando se aplican estas restricciones. Los resultados son un promedio de los datos obtenidos sobre 15 imágenes de prueba.

	Original	*Distancia*	*Angulares*	*Submuestreo*
Ejes mayores	2.684.733	649.656	143.324	2.658

Tabla 4.1: Tabla comparativa de los posibles ejes mayores con las distintas restricciones.

Un ejemplo del resultado de la detección se muestra en la Figura 4.4.

4.1.2. Segundo algoritmo: detección por mínimos cuadrados

La función FitEllipse de OpenCV [15] es utilizada para detectar elipses en una imagen. Para ello recibe como entrada un conjunto de puntos 2D y calcula la elipse

Figura 4.4: Resultado de la detección de la lonja por el primer algoritmo.

que mejor se les aproxima en el sentido de mínimos cuadrados. La función devuelve el rectángulo en el que está inscripta la elipse. En este caso se utilizó para ajustar una elipse a una selección de puntos obtenida mediante el filtrado de la imagen. Imponiendo luego una condición de relación de aspecto se obtuvo el resultado de este algoritmo.

Implementación

En primer lugar se utilizó el filtro de color sobre la imagen, detectando previamente el tono de verde que tiene el borde de la lonja del tambor. A esta imagen filtrada se le detectaron los bordes de forma de simplificar el problema aún más, y luego se utilizó la función FitEllipse para obtener los rectángulos en donde están inscriptas las elipses. Sobre dichos rectángulos se impuso luego una condición de relación de aspecto para descartar todas aquellas elipses que estuvieran muy estiradas o que no se aproximaran a la forma del tambor (que es conocida). Un ejemplo del resultado del proceso puede verse en la Figura 4.5:

(a) Elipse detectada sobre la imagen original

(b) Elipse detectada

Figura 4.5: Resultado de la detección de la elipse

Los dos algoritmos propuestos presentan un desempeño similar. Sin embargo, se resolvió utilizar el segundo por ser más sencillo y requerir un costo computacional menor, además ser ser más eficiente al estar implementado en C++. Este aspecto es

importante teniendo en cuenta que se van a procesar videos con una gran cantidad de frames. Además, este algoritmo es conveniente ya que permite integrarlo a otros algoritmos que fueron implementados haciendo uso de la biblioteca de OpenCV, como se verá más adelante.

4.2. Segmentación del palo

Al igual que la lonja, el palo es uno de los objetos de interés de la escena. Conociendo la posición de la punta del palo en cada frame, se tiene una idea de si es posible que ocurra un golpe de palo, flam, rebotado o borde. Por lo tanto, su detección es importante para la determinación de características derivadas del video. A continuación se presentan dos alternativas planteadas para resolver este problema. La primera fue creada para detectar el palo en los videos en que está pintado de rojo y la segunda para los que está de color verde.

4.2.1. Primer algoritmo: filtro de color y detección de segmentos

Como primera aproximación se utilizó un enfoque simple basado en la segmentación de color y detección de segmentos. Para ello, se intentó primero separar el palo de la escena utilizando el filtro de color explicado en el Apéndice B. Luego se detectaron los segmentos correspondientes mediante un algoritmo del estado del arte en detección de segmentos en imágenes digitales: *Line Segment Detector* (LSD) [52].

Algoritmo de detección de segmentos (LSD)

El algoritmo *Line Segment Detector* se basa en la búsqueda de contornos rectos dentro de la imagen, es decir, regiones en donde el nivel de gris cambia notoriamente entre píxeles vecinos. Los límites de dichas regiones son llamados *líneas de nivel* (del inglés *level-line*), y pueden ser detectados mediante el módulo del gradiente de la imagen (Figura 4.6).

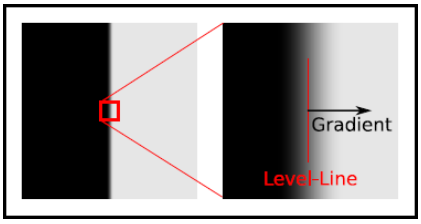


Figura 4.6: Concepto de gradiente y level-line - Imagen extraída de [52]

El algoritmo comienza generando un campo de orientaciones asociadas a cada uno de los píxeles, como se muestra en la Figura 4.7. Dicho campo se obtiene calculando el ángulo de la línea de nivel en cada píxel (Figura 4.6). Luego, se separan grupos conexos de píxeles cuyas líneas de nivel llevan el mismo ángulo a

menos de cierta tolerancia τ (regiones denominadas *regiones de líneas de soporte* o *line-support regions*), como puede verse en la Figura 4.7.

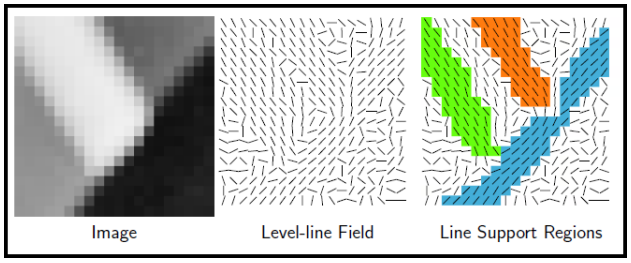


Figura 4.7: Imagen, campo de orientaciones y las regiones determinadas - Imagen extraída de [52]

Una vez determinadas las regiones de líneas de soporte, se busca el rectángulo que mejor aproxime cada región. Cada rectángulo queda determinado por su centro, dirección, ancho y longitud. La magnitud del gradiente asociado a cada píxel hace las veces de masa. Se toma la dirección del rectángulo igual a la dirección del eje de inercia principal de la región, y el centro igual a su centro de masa. El ancho y la longitud del segmento son elegidos de manera de cubrir el 99 % de la masa de la región.

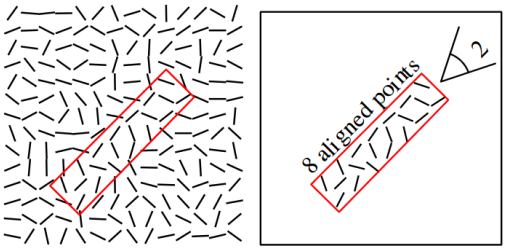


Figura 4.8: Ejemplo de puntos alineados - Imagen extraída de [52]

Cada rectángulo es sujeto luego a un procedimiento de validación basado en el método *a-contrario*. Para ello se realiza un conteo de los píxeles de cada rectángulo (n), y del número de píxeles alineados (k). Los píxeles alineados son aquellos en los que el ángulo de la línea de nivel coincide con el ángulo del rectángulo, a menos de una tolerancia τ (Figura 4.8). Los valores de n y k son luego utilizados para validar o no el rectángulo como un segmento detectado. El método de validación define un modelo 'ruidoso' o *a contrario* (H_o) en el cual la estructura buscada (en este caso un segmento) no está presente. Luego, se valida un rectangulo si la ocurrencia del segmento correspondiente al mismo en el modelo H_o es suficientemente baja, es decir, si este segmento no podría ocurrir 'por casualidad'.

Utilización del LSD en el primer algoritmo de detección

Una vez segmentado el palo mediante el filtro de color (Figura 4.9(a)), se hizo uso del algoritmo LSD para detectar los segmentos que lo caracterizan (Figura

4.9(b)). Se utilizó una tolerancia de $\tau = 100$ grados y se aceptó una densidad mínima de puntos alineados por región de 90 %. Dado que en algunas ocasiones el LSD reconoce varios segmentos de la escena de longitud menor a la del palo, se realizó un proceso de validación en el cual se discriminaron aquellos muy pequeños, conservando los segmentos más grandes.

(a) Segmentación del palo rojo con filtro de color en la base eMe

(b) Detección de segmentos en la imagen segmentada

Figura 4.9: Resultado de la detección del palo con LSD, primer algoritmo (imágenes con zoom)

Este enfoque no dió buenos resultados para todos los videos de las bases de datos. Este es el caso de algunos videos de la base eMe, donde el color del palo coincide con el color del aro de la lonja (Figura 4.10(a)) produciendo que el algoritmo no detectara los segmentos correctos. Además, en la base Zavala hay cambios importantes de iluminación en el palo a lo largo de un mismo video (ver Figura 4.10(b)), provocando que cambie de color y que por lo tanto el filtro no funcione como se espera. Para resolver estos dos problemas se probaron varios enfoques distintos, presentándose a continuación las soluciones definitivas.

4.2.2. Segundo algoritmo: mejoras de la detección en cada base de datos

A continuación se explican las mejoras introducidas en el algoritmo anterior para lograr la segmentación del palo en la totalidad de los videos de las bases de datos. Dado que se tenían problemas distintos, los enfoques para mejorar el algoritmo en cada base de datos fueron también diferentes.

Base eMe

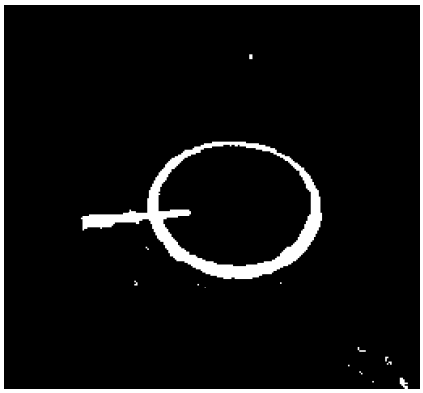
Este segundo algoritmo fue implementado con el objetivo de mejorar la detección en las tomas en las que el palo y la lonja están pintadas del mismo color, ya que como se muestra en la Figura 4.11(a), el filtro de color segmenta ambos elementos y esto confunde al algoritmo que detecta segmentos. Dado que se contaba

(a) Ejemplo de imagen con palo verde, base de datos eMe

(b) Ejemplo de cambio de iluminación que afecta el color del palo rojo, base de datos Zavala

Figura 4.10: Ejemplos donde el primer algoritmo para la detección del palo no fue suficiente (imágenes con zoom)

con una buena detección de la lonja (Figura 4.11(b)) se optó por eliminarla de la escena mediante el uso de una máscara binaria. Como se muestra en la Figura 4.12(a), se dilató la detección de manera que cubriese el espacio que ocupa el aro de la lonja. Utilizando la máscara sobre el resultado del filtro de color se obtuvo una imagen binaria en la que el único elemento presente es el palo.



(a) Resultado del filtro de color cuando la lonja y el palo tienen el mismo color (imagen con zoom)

(b) Segmentación de la lonja cuando el palo es verde (imagen con zoom)

Figura 4.11: Resultados intemedios segundo algoritmo

Este procedimiento tiene el problema de que en aquellos frames en los que el palo golpea la lonja y se le superpone, al restar la elipse dilatada se fragmenta el palo, como se muestra en la Figura 4.12(b). Por lo tanto, al aplicar el algoritmo LSD se obtienen cuatro segmentos correspondientes a los dos fragmentos, en lugar

de dos. Se decidió conservar los segmentos más grandes ya que en aquellos casos en los que el palo no se superpone a la lonja el LSD obtiene los dos segmentos correspondientes al mismo sin la necesidad de realizar ajustes. En el caso en que esto no sucede, en general los segmentos más largos corresponden al fragmento del palo sujeto por la mano. También en este caso, dichos segmentos son en general de largo menor al real, por lo que se procedió a alargarlos hasta alcanzar un largo fijo, que se seteó arbitrariamente en 100 píxeles. El umbral en el largo fue estimado en base al cálculo de la longitud del palo en varios frames del video.

En la Figura 4.13 puede verse el resultado de segmentar una imagen utilizando el primer y el segundo algoritmo propuestos (Figuras 4.13(a) y 4.13(b) respectivamente). Como allí se refleja, la detección mejora notoriamente utilizando este algoritmo.

(a) Máscara utilizada para restar la lonja

(b) Detección del palo verde en imagen sin lonja

Figura 4.12: Máscaras utilizadas para la detección del palo verde en el segundo algoritmo para la base eMe (imágenes con zoom)

(a) Detección del palo verde en imagen con lonja (primer algoritmo)

(b) Detección del palo verde en imagen sin lonja (segundo algoritmo)

Figura 4.13: Comparación de la detección para palo verde con el primer y el segundo algoritmo propuestos (imágenes con zoom)

Base Zavala

Para la detección del palo en la base Zavala se utilizó un enfoque diferente. El problema en este caso fue que el palo no es detectado por el filtro de color en varios frames debido a diferencias de iluminación, como se ve en la Figura 4.14. Una alternativa posible consiste en muestrear el nuevo color del palo y utilizarlo para filtrar la imagen cuando el filtrado por color no funcione. Sin embargo, esto presenta la desventaja de que el color no es homogéneo en todos los frames y en algunos casos no tiene un contraste significativo respecto del fondo, como se ve en la Figura 4.10(b). Otra alternativa para mejorar la detección se basa en que el palo es el objeto que se mueve más rápido en la escena, por lo que puede utilizarse un algoritmo de extracción de fondo [92] para lograr separarlo del resto de los objetos. A continuación se describe el algoritmo de extracción de fondo utilizado en la solución final.

(a) Ejemplo donde el palo es rojo

(b) Ejemplo donde el palo cambia de color

Figura 4.14: Ejemplo de cambios de color en el palo debido a la iluminación en la base Zavala (imágenes con zoom)

Algoritmo de extracción de fondo (MOG2)

El algoritmo de extracción de fondo utilizado (de ahora en más denominado *MOG2*[2]) es el descrito en [92]. En particular, se utilizó su implementación en *OpenCV* [2]. El MOG2 es un algoritmo adaptativo que utiliza mezcla de densidades de probabilidad Gaussianas para describir un modelo del fondo de la escena a nivel de píxel. Dicho algoritmo asume que existen objetos en la escena con un comportamiento estático y desarrolla un modelo de esta situación. Cuando un objeto está en movimiento, es posible detectarlo ya que no puede ser descrito en dicho modelo (Figura 4.15). El algoritmo *MOG2* tiene la particularidad de que

[2]La denominación se eligió por ser éste el nombre de la implementación en OpenCV utilizada. La sigla MOG responde a que el algoritmo es de la familia *Mixture of Gaussians*.

tanto los parámetros de dichas Gaussianas como la cantidad de densidades que se usan para describir cada píxel son actualizados a lo largo del video.

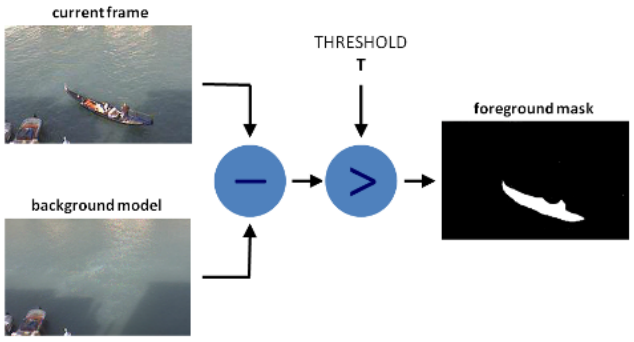


Figura 4.15: Esquema funcionamiento algoritmos de extracción de fondo (imagen extraída de [3])

El modelo de fondo se construye de la siguiente forma. Primero se determina la condición que debe cumplir un píxel para ser fondo. Dado un píxel con ciertos valores RGB (que representaremos como $\vec{x}^{(t)}$), la decisión de si pertenece o no al fondo estará dada por el siguiente umbral de decisión Bayesiana:

$$R = \frac{p(BG|\vec{x}^{(t)})}{p(FG|\vec{x}^{(t)})} = \frac{p(\vec{x}^{(t)}|BG)p(BG)}{p(\vec{x}^{(t)}|FG)p(FG)}\,, \tag{4.1}$$

donde BG refiere a los píxeles del fondo (*background*), FG a los píxeles de objetos en movimiento (*foreground* o primer plano) y R es el valor de dicho umbral. Dado que en general no se tiene información sobre los objetos que se mueven en la escena ni su frecuencia de aparición, se toma por simplicidad $p(BG) = p(FG)$ y se asume una distribución uniforme para su probabilidad de aparición $p(\vec{x}^{(t)}|FG) = c_{FG}$. De esta manera la decisión de si un píxel pertenece o no al fondo queda determinada por la siguiente relación:

$$p(\vec{x}^{(t)}|BG) > c_{FG}R. \tag{4.2}$$

El modelo de fondo dado por $p(\vec{x}^{(t)}|BG)$ es estimado a partir de un conjunto de entrenamiento χ, por lo tanto se anotará como $p(\vec{x}^{(t)}|\chi, BG)$.

La actualización de parámetros de cada densidad Gaussiana se realiza de la siguiente manera. Dado un período de tiempo T y dado un instante de tiempo t se tiene un conjunto de entrenamiento $\chi_T = \{\vec{x}^{(t)}, ..., \vec{x}^{(t+T)}\}$. Para cada nuevo frame se tendrá por lo tanto un nuevo conjunto de entrenamiento para poder estimar el modelo. Cabe señalar que dado un cierto conjunto χ_T, puede ocurrir que existan píxeles que no pertenezcan al fondo debido a la aparición de un objeto en movimiento en los frames que se usan para entrenar. Por lo tanto, el modelo que se está estimando en realidad es uno en que se incluyen ambas cosas. Dicho modelo se construye con la suma de M Gaussianas de la siguiente forma:

$$p(\vec{x}^{(t)}|\chi_T, BG + FG) = \sum_{m=1}^{M} \hat{\pi}_m \mathcal{N}(\vec{x}^{(t)}, \hat{\vec{\mu}}_m, \hat{\vec{\sigma}}_m^2 I)\,,$$

donde $\hat{\vec{\mu}}_1, ..., \hat{\vec{\mu}}_M$ son las estimaciones de las medias y $\hat{\vec{\sigma}}_1, ..., \hat{\vec{\sigma}}_M$ son las estimaciones de las varianzas de las densidades Gaussianas. Los valores $\hat{\pi}_1, ..., \hat{\pi}_M$ son los distintos pesos de cada densidad en la suma final (comprendidos entre 0 y 1). Los autores proponen las siguientes ecuaciones para actualizar dichos parámetros:

$$\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha(o_m^{(t)} - \hat{\pi}_m) \tag{4.3}$$

$$\hat{\vec{\mu}}_m \leftarrow \hat{\vec{\mu}}_m + o_m^{(t)}(\alpha/\hat{\pi}_m)\vec{\delta}_m$$

$$\hat{\vec{\sigma}}_m^2 \leftarrow \hat{\vec{\sigma}}_m^2 + o_m^{(t)}(\alpha/\hat{\pi}_m)(\vec{\delta}_m^T\vec{\delta}_m - \hat{\vec{\sigma}}_m^2)\,,$$

donde $\vec{\delta}_m = \vec{x}^{(t)} - \hat{\vec{\mu}}_m$. La constante α describe una envolvente exponencial que decae y limita la influencia de los datos anteriores. Por último, $o_m^{(t)}$ vale 1 para aquella componente 'más cercana' con $\hat{\pi}_m$ más alto y 0 para el resto.

Para tener un modelo de los píxeles del fondo y disminuir la influencia de aquellos que no los son, se agrupan los grupos de píxeles del conjunto de entrenamiento y se calculan los parámetros de las densidades Gaussianas sobre los B grupos más grandes de píxeles (que se asume corresponderían al fondo), con B dado por:

$$B = arg\ \min_b \left(\sum_{m=1}^{b} \hat{\pi}_m > (1 - c_f) \right)\,,$$

donde c_f es una medida de la máxima cantidad de píxeles que pueden pertenecer a un objeto en movimiento y no alterar el modelo de fondo. Modificando dicho parámetro se puede determinar cuántos frames se debe quedar quieto un objeto para considerarlo como parte del modelo. En particular, dados los valores de c_f y α, para ser considerado parte del fondo un objeto debe quedarse quieto un número de frames dado por $log(1 - c_f)/log(1 - \alpha)$.

Para determinar la cantidad de números de componentes a utilizar a medida que se va actualizando el modelo, el algoritmo de [92] realiza la siguiente modificación a la ecuación 4.3:

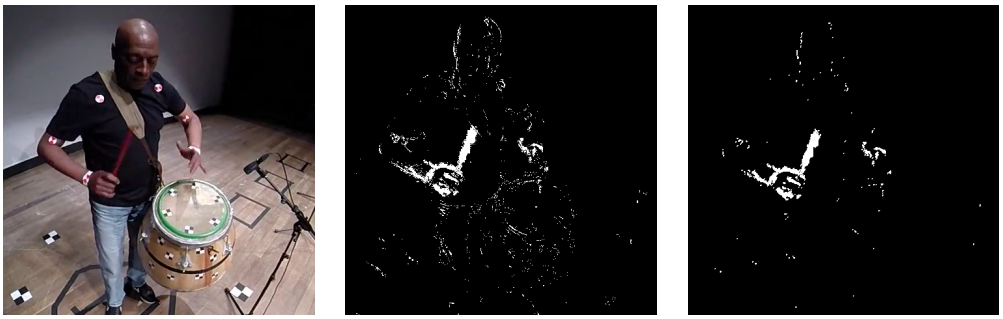
$$\hat{\pi}_m \leftarrow \hat{\pi}_m + \alpha(o_m^{(t)} - \hat{\pi}_m) - \alpha c_T \tag{4.4}$$

donde $c_T = c/T$. El valor de c se explica con detalle en [93]. La ecuación 4.4 permite que existan valores de $\hat{\pi}_m$ negativos. Cuando dicho valor es negativo, el componente m relativo al mismo es descartado del modelo. Los valores de $\hat{\pi}_m$ obtenidos son normalizados para que sigan comprendidos entre 0 y 1.

Utilización del MOG2 en el segundo algoritmo de detección

Para la utilización de dicho algoritmo, se debieron modificar los valores por defecto de algunos de sus parámetros. Dado que no habían tomas del fondo de la escena sin los intérpretes, los videos de las bases de datos utilizadas no eran una aplicación típica de este tipo de algoritmos y se debió tener esto en cuenta para ajustarlos a las condiciones del problema. En la implementación de OpenCV, los parámetros *BackgroundRatio* y *setHistory* determinan la cantidad de frames que debe quedarse quieto un objeto para formar parte del modelo de fondo. Para

ser parte del modelo, el objeto debe permanecer sin moverse durante *BackgroundRatio*setHistory* frames. Para este problema, se utilizaron valores de *BackgroundRatio*= 0,9 y *setHistory*= 50, lo que implica un total de 45 frames. Este valor es razonable si se tiene en cuenta que se filmó con una tasa de 240 fps y que los objetos demoran varios frames en moverse. Por otro lado, se utilizaron 5 Gaussianas para construir dicho modelo. En la Figura 4.16 puede verse un ejemplo del resultado de aplicar este extractor de fondo.



(a) Imagen original

(b) Máscara obtenida a partir del algoritmo *MOG2*

(c) Máscara obtenida a partir del algoritmo *MOG2* con erosión y dilatación

Figura 4.16: Máscara de movimiento obtenida a partir del algoritmo *MOG2* (imágenes con zoom)

Una vez determinados los parámetros del MOG2, se lo incluyó en la detección del palo de la siguiente forma. Dado que el video fue grabado a 240 fps, los objetos no pueden presentar un cambio de posición significativo de un frame al otro, por lo que el palo en el frame nuevo debería estar en un entorno del último palo encontrado. Por lo tanto, en aquellos frames en los que no se pudo utilizar el filtro de color rojo se creó una máscara en torno a la posición del palo detectado en el frame anterior. Aplicándola luego al resultado del MOG2 se obtuvo una segmentación del palo que se utilizó para detectar el segmento mediante el algoritmo LSD. Es necesario realizar un enmascarado dado a que el palo no es el único objeto que se mueve en la escena, como se ve en la Figura 4.16(c). Un diagrama del algoritmo se presenta a en la Figura 4.17.

Se realizaron correcciones sobre los segmentos detectados, de manera de dar continuidad a las detecciones. En particular, se impusieron restricciones de distancia entre el palo detectado en un frame y el anterior, teniendo en cuenta que la misma no puede variar mucho de frame a frame. Se utilizó también el filtro de movimiento obtenido con *MOG2* para corregir el largo del palo, que en algunos casos varía mucho cuadro a cuadro debido a variaciones de iluminación. Dentro de las correcciones que se hicieron en esta etapa se determinó cuál de los dos extremos que devuelve el algoritmo LSD correspondía con la punta, ya que es de importancia en etapas posteriores del procesamiento. Esta solución funciona también en los videos de palo rojo de la base eMe, por lo que se la adoptó como solución final para la segmentación también en ese caso.

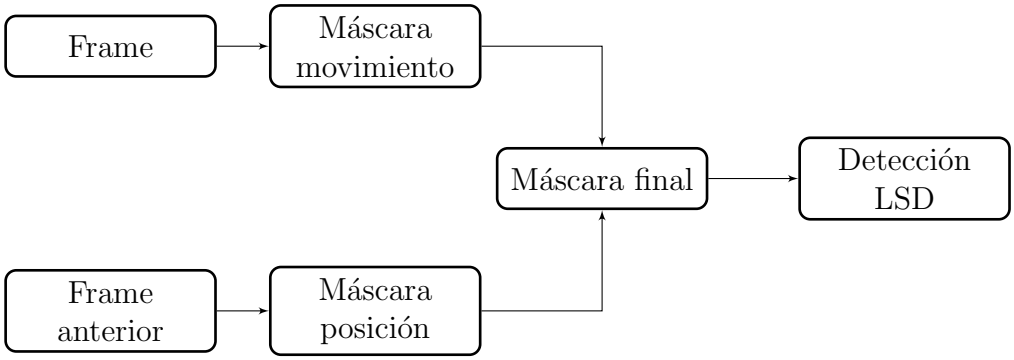


Figura 4.17: Diagrama de bloques seguimiento local del palo.

4.3. Segmentación de la mano

Dado que el golpe de mano era uno de los tipos de golpe que interesaba clasificar, la posición de la mano del intérprete fue otro de los elementos que se intentó segmentar en el video. Conocer la posición de la mano no sólo permite determinar si es posible que un evento sea o no de un golpe de mano, sino que también puede ayudar a desambiguar otros tipos de golpes, como por ejemplo el flam. Al ser este un golpe simultáneo de mano y palo, si se tienen detectados ambos en la escena, existe un flam cuando ambos elementos se acercan juntos hacia la lonja.

En este caso se buscó detectar la mano izquierda de los intérpretes, dado que todos ellos son diestros. El primer paso para su detección consistió en detectar todas las zonas pertenecientes a la piel del intérprete. Una vez segmentada la piel, se utilizó la información de la detección de la lonja para determinar la posición de la mano, dado que, la mayor parte del tiempo, la mano izquierda se encuentra por encima de la lonja. Los buenos resultados obtenidos mediante el algoritmo de detección de la lonja permitieron tomarla como referencia para la detección de la mano.

4.3.1. Primer algoritmo: segmentación por color

En primera instancia, se pensó en utilizar el filtro de color usado para la detección del palo y de la lonja para segmentar la piel de los intérpretes.

Para determinar el color de referencia del filtro, se muestrearon manualmente puntos sobre la piel de cada intérprete. Se implementó el filtrado de manera análoga a lo hecho para el palo y la lonja, para lo cual se probaron varios umbrales de similitud y saturación hasta determinar una pareja aceptable.

Al intentar segmentar la piel con este enfoque sobre la base Zavala, las condiciones de la grabación jugaron un papel preponderante. Debido a la colocación de marcadores en la escena, fue necesario ubicar las cámaras de manera tal que todos los marcadores fuesen visibles en el registro. Como consecuencia, el piso de la sala aparece en gran medida en todos los videos obtenidos, como se muestra en la Figura 4.18(a). El hecho de que el piso de la sala también fuera de madera

dificultó la segmentación de piel, ya que existen elementos de la escena que entran dentro del rango de U y V considerado como piel y, sin embargo, no lo son.

(a) Frame obtenido de la base Zavala.

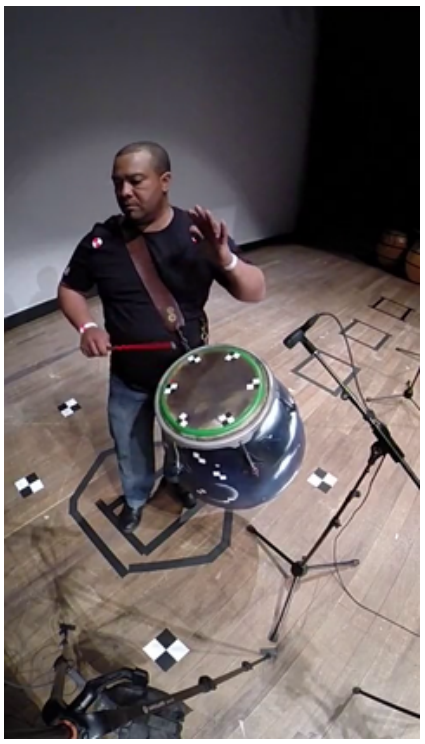
(b) Resultado de aplicar un filtro de color con rango de filtrado amplio (similitud = 3, saturación = 5) a la Figura 4.18(a).

Figura 4.18: Frame obtenido de la base Zavala y un ejemplo de filtrado en esta base, en el que no se logra distinguir la piel del piso.

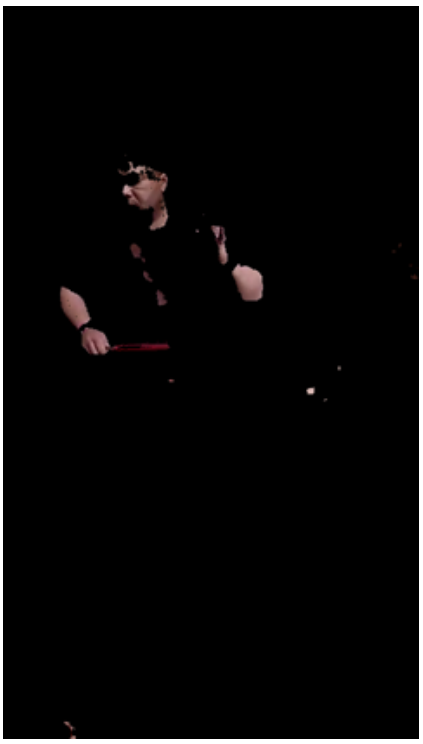
Podría pensarse en variar los umbrales de saturación y similitud hasta obtener una segmentación satisfactoria, pero las pruebas realizadas evidencian que esto no es posible. En primer lugar, utilizando un rango amplio de filtrado (esto es, valores bajos para los umbrales de similitud y saturación), no fue posible separar la piel de los intérpretes del piso de la sala. En la Figura 4.18(b) se muestra un ejemplo de filtrado poco restrictivo sobre la Figura 4.18(a).

En segundo lugar, utilizando un filtro más restrictivo (valores altos de saturación y similitud) surgió otro inconveniente asociado a las condiciones de grabación: la iluminación de la sala, sumada a la ubicación de las cámaras y de los intérpretes, hace que, por momentos, objetos de la escena se hagan sombra a sí mismos. Un ejemplo puede verse en la Figura 4.19(a), donde la palma de la mano izquierda queda oscurecida por su posición respecto a las fuentes de luz.

El hecho de no contar con una iluminación uniforme redundó entonces en que,



(a) La posición del intérprete es tal que la palma de la mano izquierda queda oscurecida por su posición respecto a las fuentes de luz.



(b) Resultado de aplicar un filtro de color restrictivo (similitud = 14, saturación = 20) a la figura 4.19(a).

Figura 4.19: Frame extraido de la base Zavala. Problemas relativos a la posición del intérprete respecto a las fuentes de luz y resultado de un filtrado restrictivo en ese caso.

al usar un filtro restrictivo, no se obtuviese una segmentación completa de la mano en los momentos en los que esta queda oscurecida. En la Figura 4.19(b) se muestra el resultado de aplicar un filtro de color altamente restrictivo sobre la Figura 4.19(a).

Cabe aclarar que con los videos de la base eMe se tuvo un problema similar, con la diferencia que el elemento que se confundía con la piel de los intérpretes no era el piso de la sala sino el tambor. En el caso de la base eMe el filtro de color es más eficiente que para la base Zavala, pero sigue sin ser suficiente, como se ve en la Figura 4.20. Si bien de aquí en adelante las imágenes utilizadas son de la base Zavala, el comportamiento es totalmente análogo para la base eMe.

(a) Imagen extraída de la base eMe.

(b) Resultado de aplicar un filtro de color poco restrictivo (similitud = 6, saturación = 5) a la figura 4.20(a).

Figura 4.20: Frame extraido de la base eMe y resultado de aplicar un filtrado poco restrictivo.

4.3.2. Segundo algoritmo: clasificación automática

Un primer intento para separar la piel del resto de la escena fue utilizar un filtrado de color poco restrictivo, para luego separar la piel del piso (o del tambor, dependiendo de la base) mediante un algoritmo de segmentación. Dado que las manos del intérprete se mueven más que el piso o el tambor, se realizó la segmentación en base a la misma estimación de movimiento utilizada para el seguimiento local del palo [92].

Este enfoque evidenció nuevas falencias en el registro ya que si bien en un primer momento se pensó que el *flicker* de la sala no tendría influencia, la estimación de movimiento contradijo esta hipótesis. Al observar el resultado de la extracción, existen cuadros aislados en los que ciertos píxeles del piso son clasificados como frente (esto es, el algoritmo detecta movimiento en esas ubicaciones) y sin embargo no se aprecia movimiento alguno en el video. En la Figura 4.21 se muestra un ejemplo de este fenómeno. La Figura 4.21(a) muestra la estimación de movimiento para un determinado frame, mientras que 4.21(b) es la estimación para el frame siguiente.

Un segundo enfoque posible es utilizar la información de luminancia de los píxeles. Ésta no es utilizada por el filtro de color, ya que sólo trabaja sobre los canales U y V. Para determinar si el canal Y aporta información relevante al problema, se muestrearon valores de la componentes YUV tanto de la mano como del piso. Graficando los valores obtenidos (Figura 4.22) pueden extraerse algunas conclusiones.

Como puede verse en las Figuras 4.22(a) y 4.22(b), utilizando la información del canal Y y sólo el canal V o el canal Y y sólo el canal U, no es posible obtener una segmentación razonable. Sin embargo, como se observa en 4.22(c), utilizando los tres canales sí parecería ser posible realizar la separación entre piel y piso. Por lo tanto se implementó un algoritmo de clasificación automática de manera de segmentar la piel. Como el objetivo de este clasificador es separar los píxeles correspondientes a piel y piso, se aplica primero un filtro de color poco restrictivo, de manera de obtener una imagen con píxeles que correspondan o bien a la piel del intérprete o bien al piso de la sala. Luego se probó segmentar con dos clasificadores

(a) Estimación de movimiento para un frame

(b) Estimación de movimiento para el frame siguiente

Figura 4.21: Ejemplo del fenómeno de flicker detectado por el estimador de movimiento [92].

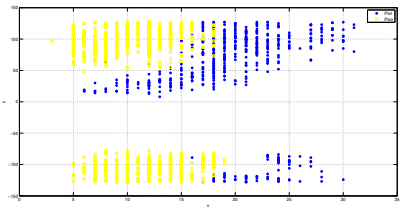
usando solamente datos muestreados sobre estos conjuntos resultantes.

Como primera aproximación, se utilizó como clasificador un Árbol de Decisión. La elección estuvo basada en que es una de las opciones más simples dentro de los algoritmos de clasificación, además de requerir un bajo costo computacional. *OpenCV* cuenta con una implementación propia de un árbol de decisión, basada en [31].

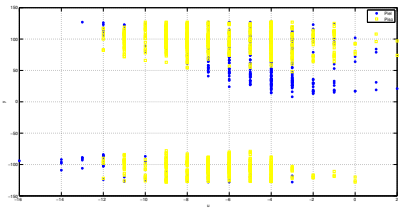
Como segunda aproximación se utilizó un clasificador del tipo Random Forest [30] (explicado en detalle en la Sección 5.1). El clasificador fue entrenado usando un total de 100 árboles.

Para ambos clasificadores, se dividió el conjunto de datos muestreados de manera de utilizar el 80 % de los datos para entrenamiento, destinando el 20 % restante para estimar desempeño. Con este procedimiento, se obtuvieron los resultados presentados en la Tabla 4.2. Los resultados para cada intérprete son el resultado del promedio del error de clasificación sobre 4 videos de la base Zavala para cada uno.

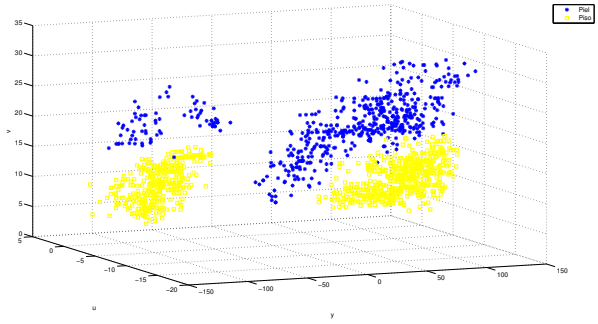
Como puede observarse, el mejor desempeño se obtuvo con el *Random Forest*. Sin embargo, el desempeño de un sólo Árbol es muy similar. Esto puede ser explicado observando la Figura 4.22(c). En ella puede constatarse que las clases son linealmente separables, por lo que es razonable que un clasificador tan simple como un Árbol de como resultado un bajo error de clasificación. Se decidió utilizar el



(a) Y vs. V



(b) Y vs. U



(c) Y vs. U vs. V

Figura 4.22: Muestreo de valores Y,U y V para la piel (azul) y el piso (amarillo)

	Árbol		*Random Forest*	
Intérprete	*Entrenamiento*	*Validación*	*Entrenamiento*	*Validación*
Hurón	1.71 %	2.46 %	1.69 %	2.41 %
Sergio	2.00 %	2.97 %	1.91 %	2.75 %
Luis	3.27 %	4.48 %	3.21 %	4.40 %
Héctor	2.10 %	2.64 %	2.09 %	2.68 %
Promedio	2.27 %	3.14 %	2.23 %	3.06 %

Tabla 4.2: Porcentaje de errores de clasificación de los conjuntos de entrenamiento y validación, para el Árbol y el *Random Forest*, sobre los intérpretes de la base Zavala.

clasificador *Random Forest* como solución de la segmentación de la mano izquierda del intérprete por presentar mejor desempeño.

Seguimiento de la mano izquierda

Una vez segmentada la piel, se utilizó la detección previa de la lonja en el video para efectivamente segmentar la posición de la mano izquierda. Esto se hizo asumiendo que durante la mayor parte del video la mano izquierda se encuentra por encima de la lonja.

Por lo tanto, el primer paso de la segmentación de la mano consistió en generar

una máscara, definida como toda la región de la imagen que está por encima del centro de la lonja, como se muestra en la figura 4.23.

Figura 4.23: Zona de búsqueda de la mano izquierda.

Asumiendo que la mano siempre se encuentra dentro de esta zona de interés, se buscó el contorno más grande de la detección de piel dentro de la máscara. Dado que en algunas detecciones de piel la mano no era una región conectada al brazo del intérprete (ya sea por los marcadores utilizados en la base Zavala o porque algunos intérpretes utilizaron pulseras en la base eMe), esta no necesariamente representaba el contorno más grande en la máscara. Por dicha razón, se dilató la detección de piel dentro de la máscara, de manera de conectar artificialmente estas dos regiones. Un ejemplo de la detección sin dilatar se muestra en la Figura 4.24, mientras que 4.25 muestra la detección con dilatación sobre la misma imagen.

Una vez separado el contorno más grande se intentó obtener una medida numérica que fuese representativa de la posición de la mano. Una primera aproximación fue hallar el *bounding box* del contorno más grande. Éste se define como el menor rectángulo (con lados paralelos a los lados de la imagen) que contiene a dicho contorno. Como un indicador de la posición de la mano se tomó al punto medio del segmento inferior del bounding box. Esto se puede observar en la Figura 4.24(b), donde se muestra el bounding box detectado y el punto que se toma como indicador de la posición de la mano.

El hecho de que existan elementos con componentes YUV similares a los de la piel hace que la detección de piel no sea exacta. Esto repercutió en la segmen-

(a) Estimación de movimiento dentro de la máscara de búsqueda de la mano izquierda

(b) Bounding box del contorno más grande de 4.24(a).

Figura 4.24: Detección de piel dentro de la máscara de búsqueda de la mano izquierda y bounding box del contorno más grande.

tación de la mano izquierda: existen momentos del video en los que el contorno más grande segmentado incluye otros elementos además de la mano (por ejemplo, existen frames en los que los bordes del palo se detectan como piel y la dilatación hace que queden conectadas al brazo del intérprete). Esto causa que el bounding box se desajuste y por lo tanto el punto medio del segmento inferior ya no sea una buena estimación de la posición de la mano, como se ve en la Figura 4.26.

Sin embargo, observando la evolución temporal de dicho punto se puede constatar que estos casos son los menos frecuentes, siendo estas variaciones esporádicas y por un período corto de tiempo. Por lo tanto se decidió utilizar un filtro de Kalman [56] de manera de suavizar las variaciones debidas a este fenómeno. Éste se explica con detalle en el Apéndice C.

(a) Dilatación de 4.24(a)

(b) Bounding box del contorno más grande de 4.25(a).

Figura 4.25: Dilatación de 4.24(a) y detección del bounding box sobre la imagen resultante.

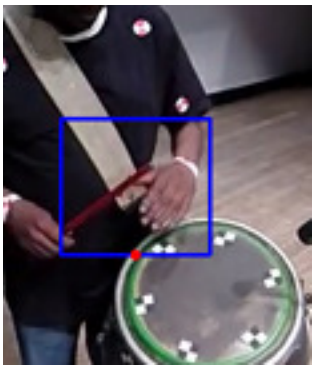


Figura 4.26: Bounding box del contorno más grande detectado cuando la segmentación de piel no es buena.

4.4. Extracción de características

Una vez detectados y caracterizados numéricamente el palo, la lonja y la mano del intérprete a lo largo del video, se exploró el comportamiento de dichos datos en los distintos tipos de golpe con el fin de determinar un conjunto de características que logre distinguirlos.

Se utilizó la ubicación temporal de cada golpe (obtenida del etiquetado) para estimar una ventana en la que se espera se encuentre la información del mismo en las detecciones del video. Por ejemplo, para un golpe de mano, se espera que el punto inferior de la misma esté cercano a la lonja dentro de la ventana considerada. De forma análoga, para un golpe de palo se espera que su punta esté cercana al tambor. Un comportamiento típico de estos tipos de golpes es que la mano o el palo baje y vuelva a subir entorno al evento, lo que implica que se tengan mínimos locales en las posiciones verticales de los mismos. En la Figura 4.27 puede verse este comportamiento.

Para determinar el tamaño de la ventana de trabajo se debió tener el cuidado de incluir los eventos pero evitar introducir información correspondiente a otros golpes cercanos en el tiempo.

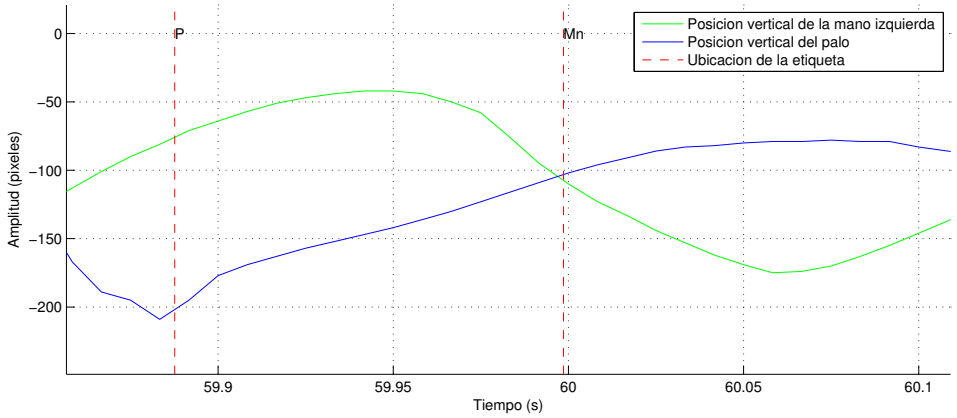


Figura 4.27: Ejemplo del comportamiento de las detecciones de la mano y la punta del palo para un golpe de palo y otro de mano.

Analizando los desfasajes existentes entre las detecciones del video y la ubicación de los eventos para distintos golpes, se observó que los golpes sucesivos de palo, rebotado y borde pueden ocurrir en intervalos de tiempo menores que los golpes de palo seguidos de golpes de mano o madera. Considerando además que los mínimos en la posición de la punta del palo no presentaron un desfasaje considerable con la ubicación de las etiquetas (como se refleja en la Figura 4.27), estos tiempos determinaron un máximo en la ventana de trabajo. El valor de dicha ventana se estableció en 140 ms centrada en la ubicación del evento etiquetado. Sin embargo, los mínimos de la detección de mano presentan un atraso de casi 70 ms con la ubicación del golpe, debido al suavizado con el filtro de Kalman. Se decidió entonces utilizar un ventana para determinar las características del palo

(que anotaremos W_P) y otra para las de la mano (W_M). Ésta última se tomó de forma asimétrica: 30ms antes del evento, y 90ms después.

Se trabajó con dos conjuntos de características. El primero, al que llamaremos *conjunto geométrico*, fue pensado para discriminar entre las clases palo, mano y madera. El segundo, *conjunto DCT*, para discriminar las clases rebotado, borde y flam de la clase palo.

4.4.1. Conjunto geométrico

A continuación se define la nomenclatura utilizada a lo largo de esta sección de manera de simplificar la notación.

- posPy: coordenada y de la punta del palo.
- posPx: coordenada x de la punta del palo.
- posMy: coordenada y de la mano.
- posMx: coordenada x de la mano.
- posEy: coordenada y del punto inferior de la lonja.
- posEx: coordenada x del punto más a la izquierda de la lonja .

Para referirse a estas cantidades consideradas dentro de la ventana de trabajo se utilizarán los subíndices correspondientes. Así, posEy_{W_P} refiere a la coordenada y del punto inferior de la lonja dentro de la ventana W_P usada para el cálculo de las características relacionadas con el palo.

Además, dado un vector de números reales v, se denominará $\bar{v}$ a la mediana de v.

Características derivadas de la posición

En primer lugar se propusieron como características geométricas las posiciones verticales y horizontales de las detecciones de mano y palo referenciados a la posición de la lonja. Estas distancias fueron normalizadas para independizarse de los valores relativos a cada video, ya que existen diferencias de posición en videos de intérpretes diferentes y particularmente entre las dos bases de datos. Fueron pensadas para informar si en el entorno de la ocurrencia de un evento el palo o la mano están cerca de la lonja, condición necesaria para que exista un golpe de palo o mano. La expresión numérica de dichas características se muestra a continuación.

- Posición vertical de la punta del palo relativa a la lonja:

$$\text{posYpalo}_{\text{norm}} = \frac{\overline{\text{posPy}}_{W_P} - \overline{\text{posEy}}_{W_P}}{\max_{10}(\text{posPy}_{W_P} - \text{posEy}_{W_P})}. \tag{4.5}$$

  Dicho cálculo se realizó tomando la mediana de los valores pertenecientes a la ventana de trabajo W_P. Previendo que las señales presentan ruido y que

por lo tanto no es conveniente utilizar valores puntuales para los cálculos, se normalizó calculando la mediana de los diez valores más grandes de la diferencia posPy − posEy dentro de la ventana W_P. Procediendo de forma similar se calcularon las demás características derivadas de la posición.

- Posición vertical de la mano relativa a la lonja:

$$\text{posYmano}_{\text{norm}} = \frac{\overline{\text{posMy}}_{W_M} - \overline{\text{posEy}}_{W_M}}{\max_{10}(\text{posMy}_{W_M} - \text{posEy}_{W_M})}.$$

- Posición horizontal de la punta del palo relativa a la lonja:

$$\text{posXpalo}_{\text{norm}} = \frac{\overline{\text{posPx}}_{W_P} - \overline{\text{posEx}}_{W_P}}{\max_{10}(\text{posPx}_{W_P} - \text{posEx}_{W_P})}.$$

- Posición horizontal de la mano relativo a la lonja:

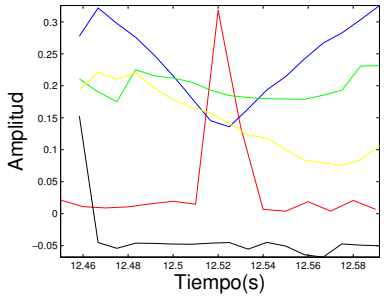
$$\text{posXmano}_{\text{norm}} = \frac{\overline{\text{posMx}}_{W_M} - \overline{\text{posEx}}_{W_M}}{\max_{10}(\text{posMx}_{W_M} - \text{posEx}_{W_M})}.$$

En las Figuras 4.28 y 4.29 se muestra el comportamiento de las distancias relativas de la mano y el palo respecto a la lonja, para los distintos tipos de golpe. Dichas distancias se encuentran normalizadas. La ventana utilizada para las características de palo fue W_P, mientras que para las características de mano se utilizó la ventana W_M.

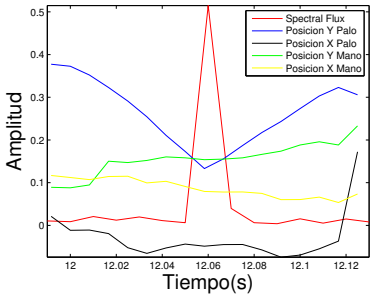
Como se observa en la Figura 4.28, los golpes de palo presentan un mínimo local en la distancia del palo a la lonja cercano al instante del evento (el cual coincide con un evento detectado en el spectral flux del audio). Análogamente, la posición vertical de la mano respecto a la de la lonja en un golpe de mano también presenta un mínimo dentro de la ventana, pero como se dijo antes está retrasado aproximadamente 70 ms de la posición del máximo local correspondiente del spectral flux (Figura 4.28).

Un golpe de tipo flam se debería observar como un golpe de palo seguido de uno de mano (o uno de mano seguido por uno de palo) separados una distancia menor a la de un golpe común. Dado que la etiqueta esta colocada en el instante de tiempo correspondiente el primer golpe y que además la mano presenta el desfasaje descrito anteriormente, cuando el primer golpe es de palo, el golpe de mano queda fuera de la ventana, como se observa en la Figura 4.29.

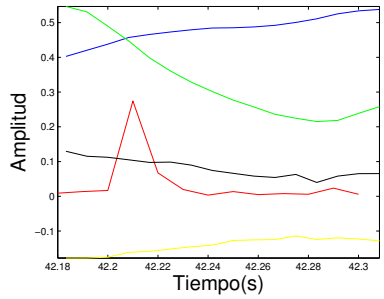
Los golpes de madera se diferencian del resto de los golpes de palo principalmente porque la punta del palo se encuentra por debajo de la lonja, dando lugar a valores negativos a la expresión 4.5, como se observa en la Figura 4.29. Esto no sucede con los golpes rebotados y de borde, en los cuales la punta tiene posiciones similares a los golpes de palo. A diferencia del resto, los golpes rebotados están formados por un mínimo local de la posición de la palo (de forma similar al de un golpe de palo) y por mínimos subsiguientes de menor amplitud. Los golpes de borde tienen la particularidad de presentar en su mayoría una pendiente de magnitud mayor antes de ocurrido el golpe que después, lo que puede ser de utilidad para diferenciarlo del golpe de palo.



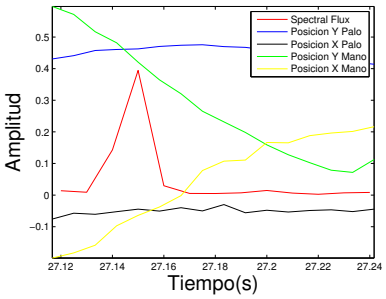
(a) Ejemplo de golpe de palo en la ventana W_P.



(b) Ejemplo de golpe de palo en la ventana W_P.

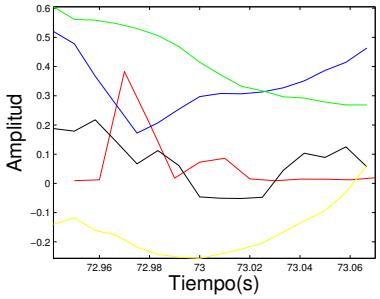


(c) Ejemplo de golpe de mano en la ventana W_M.

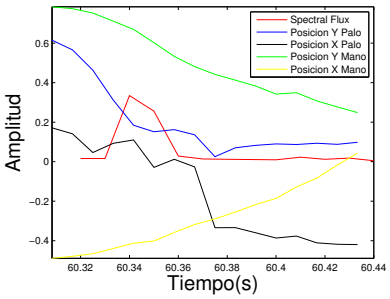


(d) Ejemplo de golpe de mano en la ventana W_M.

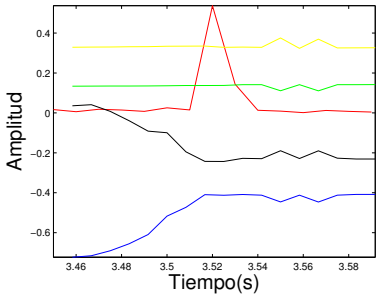
Figura 4.28: Posiciones relativas a la lonja de las detecciones en la ventana de trabajo para golpes de palo y mano.



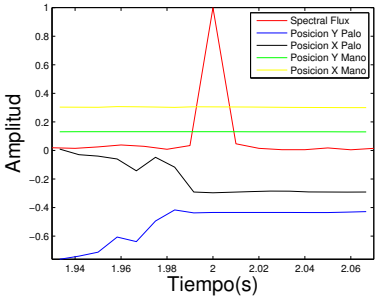
(a) Ejemplo de golpe de flam en la ventana W_M.



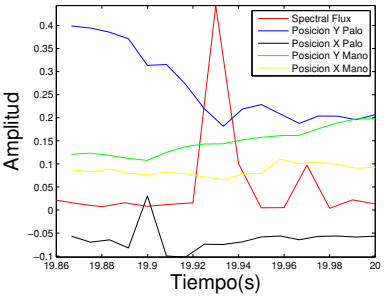
(b) Ejemplo de golpe de flam en la ventana W_M.



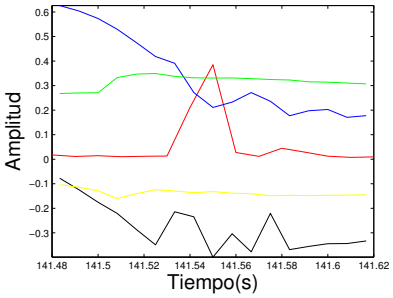
(c) Ejemplo de golpe de madera en la ventana W_P.



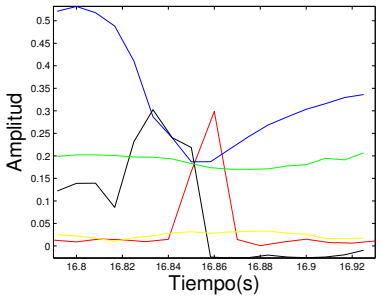
(d) Ejemplo de golpe de madera en la ventana W_P.



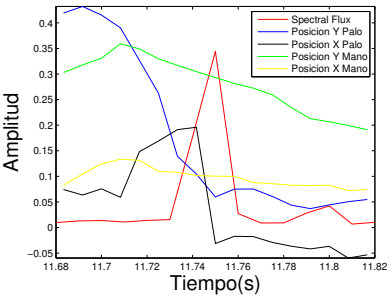
(e) Ejemplo de golpe de rebotado en la ventana W_P.



(f) Ejemplo de golpe de rebotado en la ventana W_P.



(g) Ejemplo de golpe de borde en la ventana W_P.



(h) Ejemplo de golpe de borde en la ventana W_P.

Figura 4.29: Posiciones relativas a la lonja en la ventana de trabajo para golpes de madera, flam, rebotado y borde.

Características derivadas de la velocidad

El segundo subconjunto de características geométricas propuesto fueron los máximos y mínimos de la velocidad de la mano y del palo, es decir, los máximos y mínimos de las derivadas primeras de la posición de cada uno de estos elementos. De aquí en más se denotarán como $\max_{\mathrm{M}}\mathrm{d1}$, $\min_{\mathrm{M}}\mathrm{d1}$, $\max_{\mathrm{P}}\mathrm{d1}$ y $\min_{\mathrm{P}}\mathrm{d1}$.

Dichas características fueron pensadas para brindar información complementaria de los distintos tipos de golpe. Por ejemplo, en un golpe rebotado el palo generalmente golpea la lonja con menor velocidad que un golpe de palo. Luego se aleja de la misma con una velocidad menor a la que lo haría en el caso del otro golpe. Por lo tanto, estas características podrían ser útiles para distinguir entre esos tipos de golpes.

Debido a que posPy y posMy presentan ruido proveniente de las detecciones que dificultan el cálculo de derivada, se aproximaron dichas curvas de posición por polinomios de quinto grado, como se muestra en la Figura 4.30.

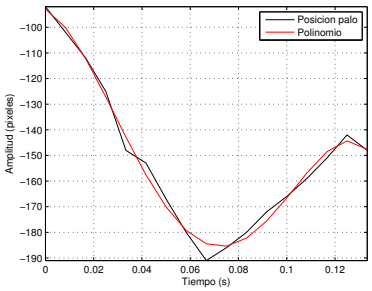
Se puede observar como el suavizado debido a este polinomio ayuda a disminuir el ruido existente en las detecciones. Sin embargo, tiene la desventaja de recortar mínimos que sería importante mantener para poder discriminar ciertas clases. Ese es el caso del golpe de rebotado que se muestra en la Figura 4.30, en el cual los mínimos correspondientes los golpes secundarios se ven amortiguados. Se decidió realizar la aproximación polinomial de todas maneras ya que permitía realizar el cálculo de la derivada de manera más simple, además de reducir el ruido proveniente de las detecciones (sobre todo en relación al palo, el cual no recibió un suavizado previo).

Características de cruces por cero

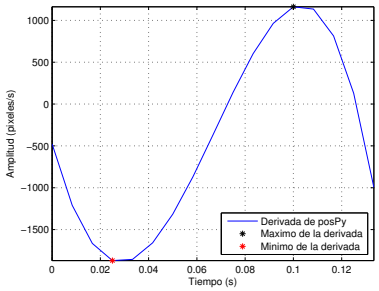
La característica de cruces por cero es la última del conjunto geométrico y fue pensada con el objetivo de ayudar a clasificar golpes de tipo rebotado. Para ello se computó la cantidad de mínimos locales de la función posPy en la ventana de trabajo W_P, mediante el conteo de los cruces por cero de su derivada. El conteo se realizó sólo para aquellos cruces que ocurrieran con una pendiente positiva de la derivada, de manera tal que correspondieran sólo a mínimos locales.

La característica de cruces por cero fue ideada teniendo en cuenta que el rebotado es el único golpe que es seguido por golpes de menor intensidad, y que por lo tanto normalmente presenta varios mínimos locales de menor amplitud subsiguientes al mínimo principal. Aunque se vio que dicho cálculo podría estar afectado tanto por el suavizado realizado con la aproximación de polinomio como por la presencia de algún otro máximo correspondiente a otro golpe o a ruido, en muchos casos es de utilidad y se decidió incluirlo (Ver Figura 4.30). De forma análoga se incluyó también como característica a la cantidad de mínimos de posMy en la ventana de trabajo W_M, ya que podría ayudar a clasificar otros tipos de golpes, aunque esta hipótesis fue descartada en la etapa de selección de características.

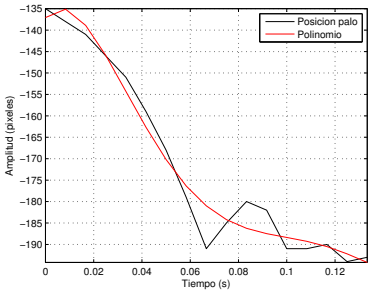
Estas características se notarán como $\mathrm{ceros}_{\mathrm{M}}\mathrm{d1}$ para los cruces por cero de la velocidad de la mano y $\mathrm{ceros}_{\mathrm{P}}\mathrm{d1}$ para los de la velocidad del palo.



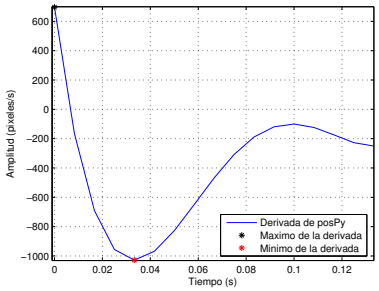
(a) Comparación de posPy vs polinomio en un golpe de palo



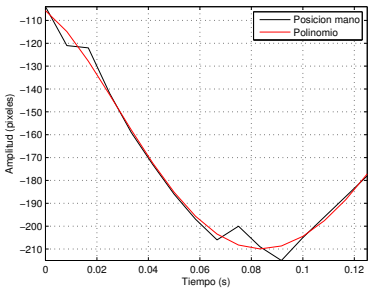
(b) Derivada del polinomio, utilizada como estimación de la velocidad del palo en un golpe de palo.



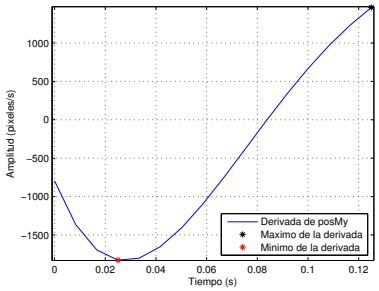
(c) Comparación de posPy vs polinomio en un golpe rebotado



(d) Derivada del polinomio, utilizada como estimación de la velocidad del palo en un golpe rebotado.



(e) Comparación de posMy vs polinomio



(f) Derivada del polinomio, utilizada como estimación de la velocidad de la mano.

Figura 4.30: Aproximación polinomial de las detecciones y su derivada para distintos tipos de golpe.

4.4.2. Conjunto DCT

Las características anteriores requirieron de cierto acondicionamiento, como ser el suavizado realizado mediante el ajuste con polinomios o el cálculo de la mediana de la señal. Esto tiene la ventaja de eliminar ruido no deseado pero la desventaja de

recortar ciertos mínimos de amplitud pequeña útiles para clasificar los golpes que se pretenden discriminar con este conjunto. Buscando mantener esa información, se utilizaron algunos coeficientes de la Transformada Discreta del Doseno (DCT) de posPy y posMy como características.

Dada una señal discreta $x[k]$ de largo N, los coeficientes $c[k]$ de la DCT de x se calculan como:

$$c[k] = w[k] \sum_{n=1}^{N} x[n] \cos\left(\frac{\pi(2n+1)(k-1)}{2N}\right),$$

donde:

$$w[k] = \begin{cases} \frac{1}{\sqrt{N}}, k = 1 \\ \sqrt{\frac{2}{N}}, 2 < k \leq N \end{cases}$$

Elección de los coeficientes de la DCT

Observando la señal reconstruida en el tiempo a partir de la DCT (Figura 4.31) se determinó la cantidad de coeficientes necesarios para obtener en la señal reconstruida los mínimos antes mencionados, decidiendo incluir los primeros diez de posPy y posMy.

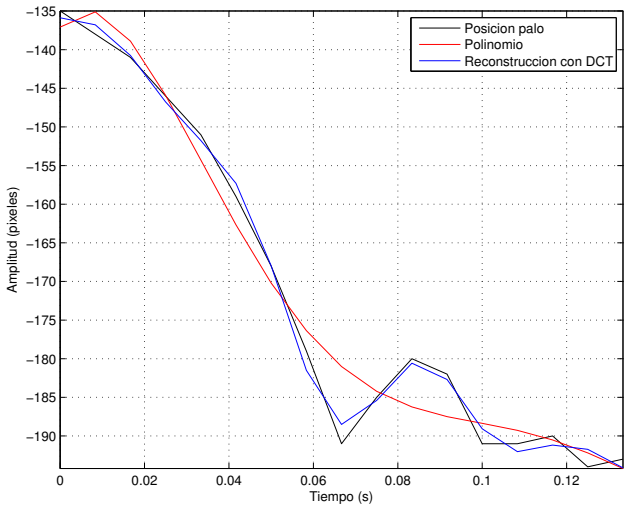


Figura 4.31: Posición vertical de la punta del palo en un golpe rebotado, aproximación por un polinomio de grado 5 y reconstrucción usando los primeros 10 coeficientes de la DCT.

Esta página ha sido intencionalmente dejada en blanco.

Parte II

Clasificación

Capítulo 5

Marco teórico

En este capítulo se presenta una breve descripción de algunos conceptos importantes referidos al reconocimiento de patrones abordados en el transcurso de este proyecto. Se describirán brevemente los distintos tipos de clasificadores utilizados y la forma en que se determinaron sus parámetros óptimos. Además, se presentarán los fundamentos de las diferentes técnicas de *selección de características* utilizadas [87] y se describirán brevemente las técnicas usadas para comparar el desempeño de diferentes clasificadores o conjuntos de características [88].

En el presente capítulo se mencionan también las técnicas de integración multimodal utilizadas en este proyecto.

5.1. Introducción al Reconocimiento de Patrones

5.1.1. Árboles de decisión

Los árboles de decisión son clasificadores con una estructura de árbol. Para construirlos se parte de un nodo raíz en donde se encuentran todas las muestras de entrenamiento. Luego se procede a hacer la primera ramificación. El objetivo de una ramificación es lograr un subconjunto más homogéneo que el que se tiene, y la creación de una nueva rama se realiza en términos del valor de impureza de la misma. La impureza de un nodo es una medida de qué tan homogéneo es. Cuanto más homogéneo es el nodo más baja es la impureza. Para realizar una ramificación, se selecciona la característica que maximiza el decremento de impureza, y se elige una decisión asociada a ella. De esta forma se crean nodos intermedios, cada uno asociado a una partición de los datos, de manera que repitiendo el procedimiento anterior para cada uno de ellos, se logra construir el árbol. Una vez construido el árbol se clasifican los datos empezando en el nodo raíz y siguiendo el camino determinado por las decisiones tomadas en cada nodo, hasta llegar a una hoja. La etiqueta asignada a ésta es la que se asignará a dicho patrón.

Este método de clasificación puede lograr (dependiendo del problema) un desempeño similar a otros como redes neuronales o k-NN, y requiere un bajo costo computacional. Además, tiene la ventaja de poder ser interpretado fácilmente,

indicando cuáles son las características más apropiadas y los valores que permiten diferenciar mejor las clases. Sin embargo, hay que tener especial cuidado al elegir el tamaño del árbol, ya que hacer crecer el árbol hasta el mínimo de impureza sobreajustará los datos, mientras que un árbol muy pequeño puede no aportar información suficiente para lograr una buena discriminación. Es por esto que existen técnicas de *podado del árbol*. Éstas son básicamente dos: la primera posibilidad es hacer crecer el árbol hasta el límite y luego podarlo (pos-podado), con la desventaja de un mayor costo computacional. La segunda alternativa (pre-podado) consiste en decidir durante el proceso de construcción hasta donde se quiere hacer crecer al árbol, logrando un menor costo computacional pero con la desventaja de perder ramificaciones posteriores eventualmente beneficiosas.

En este proyecto se utilizó un árbol del tipo C4.5 ajustando sus parámetros de forma de obtener el mejor desempeño. Para ello se realizó una búsqueda exhaustiva en una grilla de valores posibles de sus dos parámetros principales, usando la implementación *GridSearch* de WEKA. El primero es el nivel de confianza C usado en el podado, y el segundo la cantidad mínima de patrones de entrenamiento en cada rama. El parámetro C se define para medir si la reducción de impureza debida a una ramificación es estadísticamente significativa o no. Esto se hace determinando si la ramificación tiene la misma distribución que su nodo antecesor, lo que implicaría que se hizo de forma aleatoria y su aporte no es valioso. Para ello, se realiza un test de hipótesis el cual acepta o rechaza una hipótesis nula, que en este caso es que la distribución de la ramificación y el nodo padre son la misma. En caso que el valor del test sea mayor a la hipótesis nula, la misma es rechazada y se realiza la ramificación. En caso contrario se detiene el crecimiento. Por otro lado, la cantidad mínima de patrones de entrenamiento se utiliza para eliminar nodos del árbol que tienen como salida un número de patrones menor al mínimo en cada una de sus ramas.

Una variante que se utilizó fue el *Random Forest* [30], en la Sección 4.3. Dicho clasificador consiste en un conjunto de árboles, donde cada uno da una clasificación o voto. El *Random Forest* elige la clasificación que tiene la mayor cantidad de votos sobre todos los árboles que lo componen. El término *Random* responde a cómo se construye este clasificador. Si el número de datos de entrenamiento es N, para cada árbol se toman N muestras al azar del conjunto de entrenamiento, con reposición. Esto implica que algunas muestras serán sorteadas más de una vez, y algunas no serán elegidas nunca. Las muestras sorteadas serán el conjunto de entrenamiento para construir el árbol. Luego, si cada muestra tiene M características, se especifica un número $m << M$ de manera tal que, en cada nodo, m características son seleccionadas al azar. La mejor bifurcación sobre estas m características es usada para bifurcar el nodo. Si no se aplica este procedimiento y unas pocas características son predictores de mucho peso para la respuesta, serán seleccionadas en la mayoría de los árboles, causando que estos estén muy correlacionados. En este caso, se tomó $m = \sqrt{M}$.

5.1.2. Vecinos más cercanos(k-NN)

Este método de clasificación consiste en tomar, a partir de cierta métrica definida, una esfera centrada en el patrón a clasificar con un radio tal que encierre k patrones, para luego clasificarlo según la clase más ocurrente dentro de la misma. Tiene la ventaja de ser simple y dar buenos resultados. Sin embargo, posee un alto costo computacional ya que requiere explorar todo el conjunto de datos, por lo que se hace muy costoso computacionalmente si se tiene un conjunto grande y de alta dimensión. En este contexto, la dimensionalidad de los datos está dada por la cantidad de características o descriptores. Otra desventaja que presenta este clasificador es que su desempeño se ve disminuido cuando se tienen patrones de entrenamiento con ruido, por lo que en este caso se debe elegir un k suficientemente grande para evitar errores de clasificación. Además, no toma en cuenta que existen características más relevantes que otras ya que todas las características influyen de igual manera a la hora de calcular la distancia a los patrones más cercanos.

Se utilizó un clasificador por k-NN implementado en WEKA, estimando además el número óptimo de vecinos a considerar. La métrica utilizada en todos los casos fue la Euclídea.

5.1.3. Máquinas de vectores de soporte (SVM)

Las máquinas de vectores de soporte (*Support Vector Machines*, SVM) son uno de los ejemplos más conocidos de los llamados *métodos de kernel*. La idea es representar los patrones en un espacio de altas dimensiones y allí utilizar el producto interno como medida de distancia. Se busca que un problema no separable linealmente en el espacio de características original lo sea en el nuevo espacio. La potencia del enfoque radica en que el producto interno en el espacio de altas dimensiones puede calcularse a partir de operaciones simples sobre los patrones de entrada sin necesidad de hacer explícitamente el mapeo entre ambos espacios. Esto permite formulaciones no lineales de cualquier algoritmo que pueda describirse en términos de productos internos (por ejemplo, kernel PCA).

Un kernel k puede considerarse como una función que dados dos patrones devuelve un número real que caracteriza su similitud [78]. Un tipo usual de medida de similitud es el producto interno $k(x, x') = \langle x, x' \rangle$. Un enfoque más general para definir la medida de similitud consiste en efectuar un mapeo ϕ (típicamente no lineal) de forma de representar los patrones en un espacio Y que admita un producto interno. De esta forma la medida de similitud puede definirse a partir del producto interno en Y como

$$k(x, x') = \langle \phi(x), \phi(x') \rangle.$$

Seleccionar el kernel apropiado para un determinado problema es la cuestión más relevante en los métodos de kernel. Dentro de los kernels más utilizados se encuentran:

- Kernels polinómicos: $k(x, x') = \langle x, x' \rangle^d$
- Kernels de base radial gaussiana (RBF): $k(x, x') = e^{-\gamma \langle x - x', x - x' \rangle^2}$

La idea de realizar el mapeo ϕ es encontrar una superficie de decisión S en el espacio Y que logre una mayor separabilidad entre las clases. Es decir, dado $\phi : X \to Y$ tal que $\dim(Y) > \dim(X)$, la superficie S debe satisfacer $S = \{\mathbf{x} : g(\mathbf{x}) = 0\}$, siendo:

$$g(\mathbf{x}) = \mathbf{w}^T \phi(\mathbf{x}) + b.$$

De todos los hiperplanos que dividen los patrones de entrenamiento correctamente, se determina el hiperplano separador de margen máximo, ya que así se minimiza el error de generalización del clasificador. Este hiperplano se halla determinando los parámetros $\mathbf{w}$ y b de la solución de máximo margen:

$$\begin{cases} \min_{\mathbf{w},b} \frac{1}{2}||\mathbf{w}||^2 \\ \text{sujeto a: } t_n(\mathbf{w}^T\phi(\mathbf{x}) + b) \geq 1 \ , \ n = 1, \ldots, N \end{cases}$$

siendo t_n las etiquetas de las clases. Para clasificar un patrón resta entonces determinar de qué lado está de la hipersuperficie S.

De manera de abarcar el caso en que no exista un hiperplano que separe correctamente las clases (por ejemplo porque éstas están solapadas) se modifica el algoritmo de SVM para autorizar algunos puntos mal clasificados en el entrenamiento, algoritmo conocido usualmente como C-SVM. Para esto se define una penalización ξ_n para cada patrón de entrenamiento definida como:

$$\xi_n = \begin{cases} 0, \text{ si } \mathbf{x}_n \text{ bien clasificada} \\ |t_n - g(\mathbf{x}_n)|, \text{ si no} \end{cases}$$

Así, sustituyendo la clasificación perfecta (*hard margin*) por la condición relajada (*soft margin*) se obtiene:

$$\begin{cases} \min_{\mathbf{w},b} \left\{ \frac{1}{2}||\mathbf{w}||^2 + C \sum_{n=1}^{N} \xi_n \right\} \\ \text{sujeto a: } t_n(\mathbf{w}^T\phi(\mathbf{x}) + b) \geq 1 - \xi_n \text{ y } \xi_n \geq 0 \ , \ n = 1, \ldots, N \end{cases}$$

donde C es una constante que penaliza el crecimiento de los ξ_n en la función a minimizar. De esta forma, C determina el compromiso existente entre mayor generalidad y una mayor cantidad de errores en el entrenamiento.

En este caso se utilizó un kernel RBF. Para determinar el factor γ asociado a dicho kernel y el parámetro C, se realizó una búsqueda exhaustiva sobre una grilla de valores predeterminados.

5.1.4. Selección de características

En la práctica se ha observado que el desempeño de un clasificador puede disminuir si el número de patrones de entrenamiento n no es suficiente, problema comúnmente denominado *Maldición de la dimensionalidad* [11]. Sea d el número de características presentes para la clasificación, se considera como buena práctica que la relación n/d sea mayor a diez. Dado que en un problema real puede ser difícil aumentar la cantidad de instancias para la clasificación, se deben utilizar técnicas de *selección de características* para disminuir d.

Dichas técnicas ayudan a determinar cuáles son las características más relevantes para la clasificación, descartando aquellas que introducen ruido o confunden al clasificador. De esta forma se mejora la performance del clasificador a través de una representación más estable, disminuyendo además la posibilidad de un sobreajuste a los datos. A su vez, no solamente se facilita la visualización y la comprensión del problema, sino que se ahorra tiempo y memoria de procesamiento.

Existen varios enfoques para realizar selección de características [28]. Los dos más básicos son *selección por filtrado* y *selección por encapsulado*. El primero realiza la selección antes de entrenar el clasificador, por lo que es independiente del mismo. Este método es llamado por *filtrado* debido a que el conjunto de características es filtrado realizando una valoración basada en información general de los datos de entrenamiento. En el proyecto se utilizó un filtrado *basado en correlación*, el cual evalúa conjuntos de características buscando obtener una buena correlación con las clases y penalizando la correlación entre características, de manera de tener menor redundancia. El enfoque de *selección por encapsulado*, considera el desempeño de un algoritmo de clasificación como forma de valorar los conjuntos de características. Este método presenta la desventaja de requerir un alto costo computacional.

La aplicación de estos métodos implica recorrer todo el conjunto de características y determinar el subconjunto más apropiado según la valoración que se está considerando. La cantidad de conjuntos de características posibles se incrementa exponencialemente con la cantidad de características existentes, haciendo computacionalmente imposible evaluar todos las posibles combinaciones de características. Es importante entonces aplicar estrategias de búsqueda para recorrer este espacio. Existen varios métodos, dos de los más básicos son: *selección incremental (forward selection)* y *selección decremental (backward selection)*. El primero parte de un conjunto vacío, agregando características de a una a medida que se recorre el espacio. Para determinar la característica que debe ser agregada se evalúa el conjunto incluyendo, de a una, todas las características del espacio que no pertenezcan al conjunto. Luego se agrega solo aquella que presente un mejor desempeño (utilizando, por ejemplo, validación cruzada). El procedimiento se repite hasta que ninguna de las características evaluadas produzcan un incremento en el desempeño del conjunto seleccionado. Este método garantiza encontrar una selección óptima local, pero no necesariamente en forma global. El método de selección decremental realiza el procedimento de forma análoga, pero comenzando con el conjunto de características completa y descartándolas de a una.

En este proyecto se utilizó un método un poco más sofisticado llamado *Best-first*, el cual puede funcionar de forma incremental, decremental, o con una combinación de ellos. Best-first no termina cuando el desempeño deja de aumentar sino que utiliza un configuración anterior guardada en una lista ordenada por desempeño de los conjuntos evaluados anteriormente, para luego retomar la búsqueda. Si no se detiene la búsqueda, el algoritmo puede explorar todo el espacio de características. En general se determina entre sus parámetros un criterio de parada para que esto no suceda.

5.1.5. Evaluación de desempeño

En los problemas de clasificación de patrones es natural medir el desempeño de un clasificador en términos del porcentaje de datos bien o mal clasificados. Si la clasificación obtenida para una determinada instancia es correcta, entonces se la cuenta dentro de las exitosas. De lo contrario será contada como errónea. La tasa de error es entonces la proporción de datos clasificadas erróneamente, mientras que el desempeño del clasificador estará determinando por las que fueron correctamente clasificadas.

Estimar el desempeño utilizando los datos de entrenamiento resulta demasiado optimista, ya que el clasificador fue construido a partir de ese mismo conjunto. Si por el contrario la validación se realiza utilizando un conjunto diferente al usado para el entrenamiento, se tiene una medida más realista de cómo se desempeñará el sistema frente a nuevos datos. Es por esto que se suele reservar una porción de los datos para el entrenamiento y el resto para la validación. Dado que no se dispone de infinitos datos para la clasificación, se presenta un compromiso entre utilizar la mayor cantidad de datos posibles para el entrenamiento o para la validación. En el primer caso se obtendría un clasificador más confiable, mientras que en el segundo una mejor estimación de desempeño. Se establece como buena práctica tomar 2/3 de los datos para entrenamiento y el resto para validación.

Para que la estimación sea confiable, se debe tener el cuidado de no quedarse con un único conjunto de entrenamiento/test. Usar una única partición de los datos es vulnerable frente a un mal sorteo de los datos. Este sorteo puede ser optimista u pesimista respecto al desempeño del clasificador. Tomar varias particiones y promediar los desempeños para cada partición es una forma de atenuar este efecto. Una opción es utilizar la técnica de *validación cruzada*, en la que se divide el conjunto de instancias en m particiones iguales o *folds*. Se utilizan $m-1$ particiones para el entrenamiento y una para la validación, y se repite el procedimiento m veces, de manera que cada instancia es usada una vez para validación. Promediando el error obtenido en las m validaciones se determina la estimación de desempeño. La técnica de validación cruzada utilizando 10 particiones - *10-fold-CV* - es ampliamente utilizada.

Si bien la validación cruzada presenta las ventajas mencionadas anteriormente, tiene el problema de que los resultados dependen de la división en particiones que se haga. Ese particionado puede ser particular y generar estimaciones optimistas u pesimistas. Es por eso que para atenuar el efecto del particionado se recurre a repetirlo varias veces (y promediar resultados o compararlos estadísticamente). La interfaz de usuario de WEKA *Experimenter* [1] permite realizar experimentos y analizarlos de una forma práctica. Dado un conjunto de datos de entrenamiento, un experimento permite realizar varias pruebas de desempeño de forma automática y comparar estadísticamente distintas soluciones a partir de un test de Student con cierto nivel de significación. Además, el Experimenter utiliza un test de Student modificado que tiene en cuenta que los diferentes resultados vienen de un mismo conjunto de datos y no son corridas completamente independientes.

[1] http://www.inf.ed.ac.uk/teaching/courses/dme/experimenter-tutorial.pdf

Este tipo de práctica resulta interesante por ejemplo para elegir entre clasificadores diferentes. En este caso, dado un conjunto de datos y sus características, se consideran estos clasificadores y se repite una prueba de desempeño para cada uno k veces. Considerando que la diferencia de desempeños tiene una distribución de Student y dado un nivel de confianza, se puede determinar si la media de las diferencias es significativamente diferente a cero verificando si excede los intervalos de confianza. De esta forma es posible comparar el desempeño entre clasificadores y determinar si uno de ellos resulta más conveniente.

En este proyecto se separó el total de instancias de la base eMe en dos subconjuntos. Uno estuvo formado por los registros de la base eMe de palo rojo (ver capítulo 2) y se utilizó para seleccionar las características y elegir un clasificador (en las etapas de entrenamiento y validación). El otro, compuesto por los registros de palo verde de la base eMe más el único registro etiquetado de la base Zavala, se reservó para la clasificación (etapa de test). La selección de características se realizó de las dos formas explicadas anteriormente, mientras que el clasificador con mejor desempeño se determinó en cada caso mediante un experimento repitiendo 10 veces *10-fold-CV*, utilizando un intervalo de confianza del 95 %.

5.2. Procesamiento multimodal

Como se hizo referencia en el Capítulo 1, el término *análisis multimodal* refiere a una disciplina que busca analizar, modelar y entender cómo extraer e integrar información de múltiples vías. En el contexto del procesamiento de señales, estas vías son llamadas *modos*.

Existen varios criterios para clasificar las técnicas del análisis multimodal [40]. Uno de ellos las discrimina según cómo se integra la información de los distintos modos, distinguiendo entre técnicas de integración temprana (*early integration techniques*) y técnicas de integración tardía (*late integration techniques*). Las técnicas de integración temprana se basan en el uso de los datos crudos, es decir, sin ninguna transformación previa (salvo operaciones básicas de preprocesamiento como disminución de ruido, normalización, remuestreo). Por el contrario, las técnicas de integración tardía tratan de explotar la información conjunta en los modos a nivel de decisión, combinando la salida de varios clasificadores monomodales. Otro criterio de clasificación para las técnicas de análisis multimodal se basa en la diferencia entre caracterizar las relaciones entre distintos modos, lo que se denomina *Cross-Modal Processing*, frente a combinar eficientemente la información extraída de cada modo, lo que es denominado *Multimodal Fusion*.

Dentro de los métodos de integración temprana, el *Cross-Modal Processing* propone varias maneras de expresar la relación entre los distintos modos. En este conjunto se destacan técnicas como *Canonical Correlation Analysis* (CCA), *Co-Inertia Analysis* (CoIA) y *Cross-Modal Factor Analysis* (CFA). También existen técnicas que buscan combinar distintas características extraídas a partir de los diferentes modos para lograr una representación común. Esto se denomina *Feature-Level Fusion* y presenta la desventaja de que se suelen obtener representaciones

de dimensión muy alta. Para reducir la dimensionalidad de estas representaciones pueden aplicarse técnicas de transformación como *Principal Component Analysis* (PCA), *Independent Component Analysis* (ICA) ó *Linear Discrimination Analysis* (LDA).

Las técnicas de integración tardía proponen formas de combinar la salida de varios clasificadores individuales. Muchos trabajos están basados en que cada clasificador toma una decisión sobre un modo y luego se combinan las salidas ponderadas de cada clasificador. Dicha ponderación en general está ligada a heurísticas o a procesos de "ensayo y error". Sin embargo, estas ideas pueden formalizarse en un marco Bayesiano, lo que permite además considerar ciertas imprecisiones en los diferentes clasificadores. Otra estrategia posible es usar como características las propias salidas de los clasificadores, para luego implementar un nuevo clasificador basado en esas características que resuelva de manera óptima la fusión multimodal.

En este proyecto se utilizó el método de *Feature-Level Fusion*. En dicho método, los conjuntos de características provenientes de distintos modos de información son consolidados en uno sólo, aplicando previamente técnicas de normalización, transformación y reducción de características [64]. La gran ventaja de este método es que permite detectar conjuntos de características provenientes de diferentes modos que se complementan aportando mayor información en conjunto que cada una de ellas por separado. Sin embargo, se obtienen generalmente conjuntos con alto número de características, por lo que se vuelve necesario disponer de gran cantidad de datos de entrenamiento o aplicar técnicas para reducir la dimensionalidad del conjunto de características [53]. Como se mencionó, una posible solución podría ser la de aplicar técnicas de transformación de características. Otra posibilidad es utilizar técnicas de *selección de características*, que fue el enfoque utilizado en este proyecto.

Capítulo 6

Selección de características

En este capítulo se presenta el proceso de selección de características llevado a cabo para cada modo. Además se explican los enfoques de combinación multimodal de las características extraídas de cada uno de ellos.

Tanto en los modos audio y video como en el enfoque multimodal se utilizaron dos técnicas diferentes de selección de características. La primera técnica utilizada fue *selección basada en correlación*, que, como se dijo en el Capítulo 5 requiere un bajo costo computacional y no depende del clasificador. Luego se utilizó un enfoque de *selección por encapsulado*, evaluando subconjuntos de características en base al desempeño de un árbol C4.5 con los parámetros por defecto de WEKA ($C = 0{,}25$ y $m = 2$). Para recorrer el espacio de características se usó el algoritmo *Best First* usando selección incremental (forward-selection). Además, se adoptó un enfoque de validación cruzada con 10 particiones para disminuir el impacto de la selección de datos de entrenamiento y validación.

En cuanto al conjunto de instancias utilizado para realizar la selección, se decidió trabajar con los cuatro videos de la base eMe con palo rojo. Esto fue así porque fueron los primeros para los cuales se tuvieron las etiquetas y detecciones de audio y video concluídas, mientras que en las tomas restantes se seguían realizando ajustes. También se decidió determinar los parámetros óptimos de los clasificadores con dicho conjunto. En el Capítulo 7 se incluirán las tomas que no participaron en dicho proceso para determinar el desempeño y generalidad de la solución.

Para comparar las selecciones obtenidas mediante ambos métodos de selección se podrían contrastar simplemente los desempeños obtenidos al clasificar. Sin embargo, éstos desempeños pueden deberse al conjunto particular de datos que se utiliza para realizar el cálculo, y no ser por lo tanto una medida representativa. Se resolvió entonces realizar un test de Student, con un nivel de confianza de 95 % (explicado en la Sección 5.1). De esta forma se puede establecer si existe evidencia estadística significativa para considerar que los desempeño de los algoritmos comparados son diferentes. Dado que se cuenta con poca cantidad de instancias para el entrenamiento de los datos, se repitió 10 veces el proceso de validación cruzada con 10 particiones.

6.1. Audio

Dado que se tienen 49 características derivadas del audio, es fundamental aplicar métodos de selección como los detallados en la Sección 5.1 para reducir su dimensionalidad, descartando aquellas que no aporten al problema y puedan confundir a los clasificadores. Las selecciones obtenidas utilizando el método basado en correlación y por encapsulado pueden observarse en la Tabla 6.1, donde se describe con un número del 1 al 10 la cantidad de veces que una característica fue seleccionada en la validación cruzada. El método por correlación seleccionó 37 características mientras que el método por encapsulado seleccionó 5, de un total de 49.

Característica	*Corr*	*Enc*	*Característica*	*Corr*	*Enc*	*Característica*	*Corr*	*Enc*
$\text{spec}_{\text{centroid}}$	10	0	mfcc_8	10	1	mfcc_{24}	10	2
$\text{spec}_{\text{spread}}$	10	9	mfcc_9	10	1	mfcc_{25}	10	0
$\text{spec}_{\text{skewness}}$	10	3	mfcc_{10}	1	4	mfcc_{26}	5	1
$\text{spec}_{\text{kurtosis}}$	10	6	mfcc_{11}	1	1	mfcc_{27}	6	1
$\text{spec}_{\text{decrease}}$	10	9	mfcc_{12}	0	1	mfcc_{28}	10	10
$\text{spec}_{\text{slope}}$	6	0	mfcc_{13}	9	3	mfcc_{29}	10	1
$\text{spec}_{\text{crest}}$	0	0	mfcc_{14}	10	2	mfcc_{30}	3	2
sf_1	10	3	mfcc_{15}	10	3	mfcc_{31}	10	1
sf_2	10	10	mfcc_{16}	10	6	mfcc_{32}	0	0
mfccs_1	10	10	mfcc_{17}	10	3	mfcc_{33}	0	0
mfcc_2	10	3	mfcc_{18}	10	0	mfcc_{34}	10	4
mfcc_3	10	10	mfcc_{19}	9	0	mfcc_{35}	10	0
mfcc_4	10	2	mfcc_{20}	9	0	mfcc_{36}	10	0
mfcc_5	8	0	mfcc_{21}	10	1	mfcc_{37}	10	1
mfcc_6	0	0	mfcc_{22}	0	1	mfcc_{38}	10	0
mfcc_7	10	0	mfcc_{23}	10	2	mfcc_{39}	10	0
						mfcc_{40}	10	0

Tabla 6.1: Selección por correlación y por encapsulado de las características de audio. La columna titulada *Corr* indica la cantidad de veces que se seleccionó la característica correspondiente en el experimiento de selección, usando el método de correlación. La columna *Enc* indica lo análogo para el método por encapsulado.

Dado que los métodos seleccionaron distintas características, se realizó un test de Student para determinar cuál de las dos selecciones era más conveniente, con un nivel de confianza de 95 %. En la Figura 6.1 se muestra el resultado del experimento realizado.

Se puede observar que las selecciones obtenidas son estadísticamente equivalentes al conjunto original. Sin embargo, el encapsulado selecciona unos pocos MFCCs. Dado que no se contaba con una gran cantidad de datos, la selección por encapsulado podría sobreajustar la solución a ese conjunto de datos. Es deseable para tener mayor generalidad contar con rangos de MFCCs en lugar de coeficientes aislados. Por lo tanto, se decidió tomar la selección por correlación como conjunto de características final del modo audio. Este conjunto incluye al conjunto de selección por encapsulado y se resume en la Tabla 6.2.

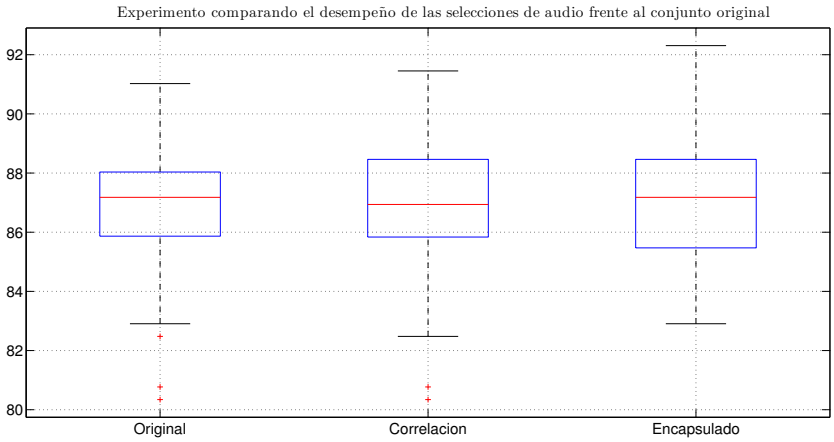


Figura 6.1: Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, comparando las distintas selecciones de las características de audio. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

$\text{spec}_{\text{centroid}}$	mfcc_3	mfcc_{16}	mfcc_{28}
$\text{spec}_{\text{spread}}$	mfcc_4	mfcc_{17}	mfcc_{29}
$\text{spec}_{\text{skewness}}$	mfcc_5	mfcc_{18}	mfcc_{31}
$\text{spec}_{\text{kurtosis}}$	mfcc_7	mfcc_{19}	mfcc_{34}
$\text{spec}_{\text{decrease}}$	mfcc_8	mfcc_{20}	mfcc_{35}
sf_1	mfcc_9	mfcc_{21}	mfcc_{36}
sf_2	mfcc_{13}	mfcc_{23}	mfcc_{37}
mfcc_1	mfcc_{14}	mfcc_{24}	mfcc_{38}
mfcc_2	mfcc_{15}	mfcc_{25}	mfcc_{39}
			mfcc_{40}

Tabla 6.2: Selección final de características de audio.

6.1.1. Determinación de parámetros óptimos

Una vez seleccionadas las características del modo audio se procedió a determinar los parámetros óptimos de tres clasificadores: un árbol de decisión C4.5, SVM y K-NN. Para esto se realizó una búsqueda exhaustiva restringida a una grilla de valores, eligiendo los parámetros que presenten un mejor desempeño, estimado mediante validación cruzada sobre un subconjunto de los datos compuesto por los Intérpretes 2 y 3 (correspondientes a las cuatro tomas en las que los intérpretes tocaron con el palo rojo). Para el árbol de decisión se determinaron dos parámetros: el nivel de confianza C usado en el podado y la cantidad mínima m de patrones admitida en cada rama. Para el algoritmo de vecinos más cercanos se determinó la

cantidad de vecinos k para la clasificación. Para SVM se utilizó un kernel RBF, buscando la mejor combinación de la constante de complejidad C y el parámetro γ del kernel. Los resultados obtenidos se muestran en la Tabla 6.3.

Clasificador	*Parámetros*	
C4.5	$C = 0{,}1$	$m = 4$
SVM	$C = 4$	$\gamma = 1$
K-NN	$K = 5$	- -

Tabla 6.3: Parámetros óptimos de la selección de audio.

A los efectos de determinar qué clasificador presenta mejor desempeño se repitió 10 veces la validación cruzada en 10 particiones y se efectuó un test de Student. Los resultados se presentan en la Figura 6.2.

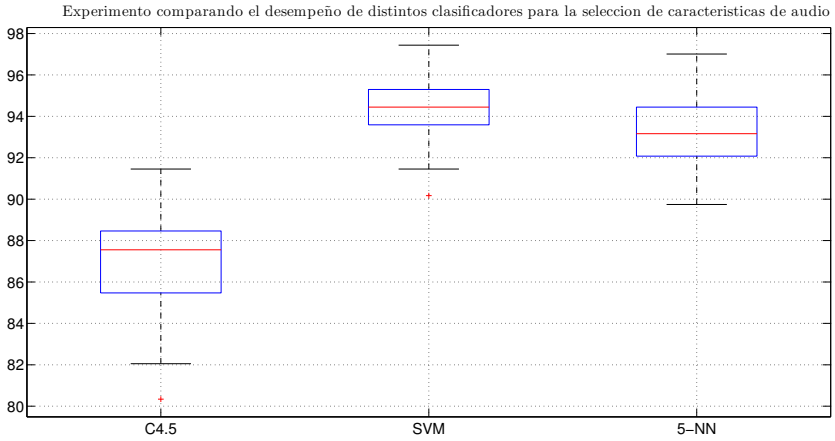


Figura 6.2: Test de Student con un nivel de significancia de 95 % realizado utilizando tres clasificadores distintos para la selección de audio. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

Los resultados anteriores reflejan que el árbol de decisión presenta un desempeño inferior a los otros. Los algoritmos de vecinos más cercanos y SVM son similares entre sí, con una leve mejora del último. Es por esto que se decidió utilizar este algoritmo para las pruebas de evaluación de desempeño.

6.2. Video

Debido a que se tenían dos conjuntos de características de video diseñados para detectar tipos de golpes distintos, la selección de características en este caso

se realizó en tres etapas. En primer lugar se utilizó el *conjunto geométrico* para obtener características que lograran una buena clasificación de las clases principales: madera, mano y palo. Luego se trabajó con el *conjunto DCT* para intentar clasificar adecuadamente las clases secundarias: rebotado, borde y flam. Por último se unieron las características seleccionadas de estos dos conjuntos en uno nuevo, buscando una buena clasificación global.

Para determinar las mejores características de cada conjunto se utilizaron las mismas dos técnicas de selección de características que para el audio.

En cuanto al conjunto de instancias utilizado para realizar la selección, se decidió trabajar con los cuatro videos de la base eMe con palo rojo al igual que en el audio. De esta manera es posible comparar el desempeño sobre los mismos datos. A continuación se presenta un resumen de las pruebas realizadas.

6.2.1. Selección de características en el conjunto geométrico

Se realizó selección de características mediante los métodos antes descritos, utilizando validación cruzada con 10 particiones. En la Tabla 6.4 se muestran estos resultados.

Características	*Corr*	*Enc*
$\text{posYPalo}_{\text{norm}}$	10	10
$\text{posYMano}_{\text{norm}}$	10	7
$\text{posXPalo}_{\text{norm}}$	10	10
$\text{posXMano}_{\text{norm}}$	0	6
$\text{min}_{\text{M}}\text{d1}$	10	7
$\text{max}_{\text{M}}\text{d1}$	0	9
$\text{min}_{\text{P}}\text{d1}$	10	10
$\text{max}_{\text{P}}\text{d1}$	10	4
$\text{ceros}_{\text{M}}\text{d1}$	10	9
$\text{ceros}_{\text{P}}\text{d1}$	10	5

Tabla 6.4: Selección por correlación y por encapsulado del conjunto geométrico. se describe con un número del 1 al 10 la cantidad de veces que una característica fue seleccionada en la validación cruzada.

Como se observa en en la Tabla 6.4, ambas técnicas de selección descartaron la característica $\text{posXmano}_{\text{norm}}$. Esto resulta razonable, ya que dicha característica no parece aportar mayor información por sí misma. Salvando esta coincidencia, los métodos presentaron selecciones distintas.

Para determinar cuál de las dos selecciones era más conveniente, se realizó un test de Student con un nivel de confianza de 95 % en WEKA. Se decidió además incluir en el experimento un tercer conjunto de características que fuera la unión de las selecciones (al que se llamó $\text{S}_{\text{U}}^{\text{geo}}$). Por último, se incorporó al conjunto geométrico como referencia. El test se realizó aplicando la técnica de validación

cruzada con 10 particiones, haciendo 10 iteraciones y clasificando con un árbol con parámetros por defecto.

En la Figura 6.3 se muestran los resultados obtenidos. Se observó que los tres conjuntos de características mencionados anteriormente tienen un desempeño estadísticamente igual al original. Dado el resultado del experimento, se decidió conservar el conjunto unión de ambas selecciones de manera de suprimir solamente aquellas características que fueron descartadas por ambos métodos.

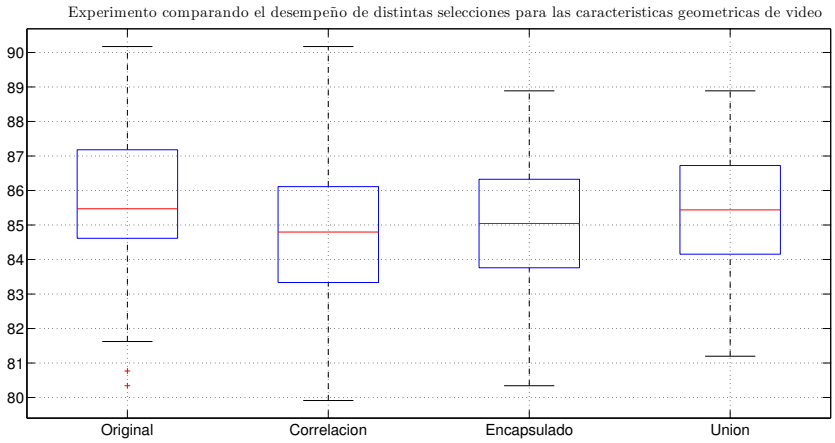


Figura 6.3: Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado sobre las selecciones del conjunto geométrico. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

6.2.2. Selección de características del conjunto DCT

Al igual que en el caso anterior, se realizó selección de características. En la Tabla 6.5 se presentan los resultados de la selección.

Analizando la Tabla 6.5 resulta razonable que se seleccionen más coeficientes de la DCT del palo que de la mano. De dichos coeficientes depende fuertemente la discriminación entre palo, borde y rebotado, golpes que presentan información importante en un rango más amplio de frecuencia. Sin embargo, para la detección de mano no se necesita la información en alta frecuencia, sino que alcanza con los primeros coeficientes de la DCT.

En este caso también se presentaron diferencias entre los conjuntos obtenidos utilizando el método por correlación y por encapsulado. De igual manera que con el conjunto geométrico, se realizó un test de Student para determinar el conjunto más conveniente. La notación utilizada es análoga a la selección del conjunto geométrico, y también en este caso se consideró el conjunto unión de las seleccio-

nes por correlación y por encapsulado. El experimento realizado se presenta en la Figura 6.4.

Características	*Corr*	*Enc*	*Características*	*Corr*	*Enc*
DCT_M1	0	7	DCT_P1	10	10
DCT_M2	10	10	DCT_P2	10	10
DCT_M3	10	6	DCT_P3	10	10
DCT_M4	0	8	DCT_P4	10	4
DCT_M5	0	4	DCT_P5	10	10
DCT_M6	0	1	DCT_P6	10	6
DCT_M7	0	2	DCT_P7	10	2
DCT_M8	0	5	DCT_P8	0	2
DCT_M9	0	0	DCT_P9	0	3
DCT_M10	0	1	DCT_P10	0	2

Tabla 6.5: Selección por correlación y por encapsulado del conjunto DCT

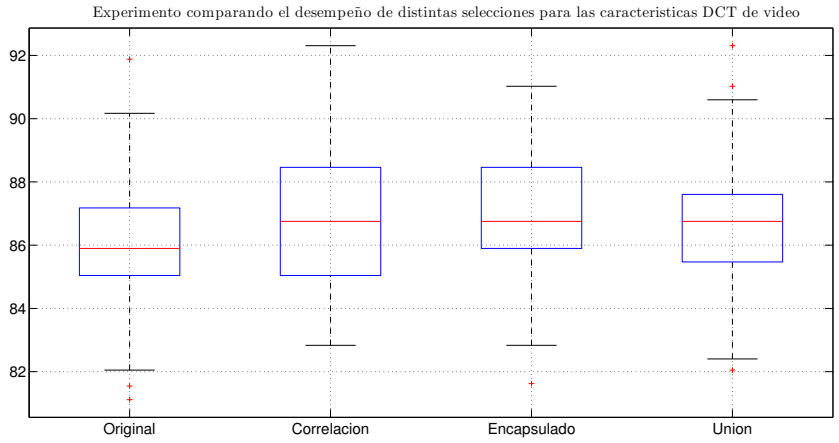


Figura 6.4: Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado sobre la selección del conjunto DCT. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

Al igual que en el caso anterior, los conjuntos presentan estadísticamente el mismo desempeño. Por lo tanto, utilizando el mismo criterio que en la selección geométrica se consideró a S_U^{DCT} (la unión de ambas selecciones) como conjunto final de características. Éste está formado por los primeros cuatro coeficientes de la DCT correspondientes a la mano y los primeros siete del palo.

6.2.3. Selección final de características del modo video

Para terminar con la selección de características del modo video, se realizó una última prueba considerando la unión de los conjuntos seleccionados. Esto es, un nuevo conjunto (al cual denominaremos $S_{geo+DCT}$) compuesto por S_U^{geo} y S_U^{DCT}. Dado que cada uno de los conjuntos fue pensado con un fin particular, es de esperar que la unión de ambos funcione mejor que cada uno por separado. A su vez, al haber realizado selección sobre las características geométricas y sobre los coeficientes de la DCT por separado, también es de esperar que existan características que sean redundantes cuando se considera la unión de las selecciones. Se realizó entonces selección por correlación y por encapsulado sobre las 20 características de $S_{geo+DCT}$. En la Tabla 6.6 puede verse el resultado de dicha selección.

Características	*Corr*	*Enc*	*Características*	*Corr*	*Enc*
$posYPalo_{norm}$	10	9	DCT_M2	10	10
$posYMano_{norm}$	9	4	DCT_M3	10	7
$posXPalo_{norm}$	10	10	DCT_M4	0	5
min_Md1	10	1	DCT_P1	10	7
max_Md1	0	4	DCT_P2	10	10
min_Pd1	2	5	DCT_P3	10	10
max_Pd1	0	4	DCT_P4	8	4
$ceros_Md1$	0	4	DCT_P5	10	7
$ceros_Pd1$	10	0	DCT_P6	10	5
DCT_M1	0	3	DCT_P7	10	2

Tabla 6.6: Selección por correlación y por encapsulado del conjunto $Seleccion_{geo+DCT}$.

Las características descartadas por ambas selecciones fueron: max_Pd1, min_Pd1, max_Md1, $ceros_Md1$, DCT_M1 y DCT_M4. En contraste a lo que se consideró en un principio, el máximo y el mínimo de la velocidad del palo no contribuyeron a distinguir entre los tipos de golpe. Debido a que existen varios tipos de golpes en los que el participa el palo (palo, rebotado, borde y flam), los valores máximos y mínimos de la velocidad del palo no aportan mayor información a la discriminación entre estas clases.

Lo mismo sucedió con el máximo de la velocidad de la mano. Dicho valor busca describir la velocidad con la que se aleja la mano de la lonja luego de un golpe, pero no se tiene ningún tipo de comportamiento particular luego del mismo. En algunas realizaciones la mano queda apoyada sobre la lonja, pero en otras se aleja de la misma para ejecutar otro golpe.

Una excepción se tuvo con el mínimo de la velocidad de la mano, ya que cuando se produce un golpe de mano el valor absoluto de este mínimo debe ser más grande que cuando no lo hay.

En cuanto a la DCT_M1, sucede lo mismo que con el máximo de la velocidad de la mano. Dado que esta característica equivale a la media de la posición de la

mano en la ventana de trabajo, debería existir un comportamiento típico de los golpes de mano para que fuera de utilidad.

Finalmente, es razonable que se descarte la $\mathrm{DCT_M}4$ debido a que se precisan pocos coeficientes para caracterizar el comportamiento de la mano, ya que ésta no presenta oscilaciones en el entorno de un golpe.

Nuevamente y tal como se muestra en la Figura 6.5, se realizó un test de Student para comparar el desempeño de $\mathrm{S_{geo+DCT}}$ con los obtenidos mediante su selección utilizando los métodos por correlación ($\mathrm{S_C^{geo+DCT}}$) y por encapsulado ($\mathrm{S_E^{geo+DCT}}$). Se comprobó que los tres conjuntos presentan desempeños estadísticamente equivalentes.

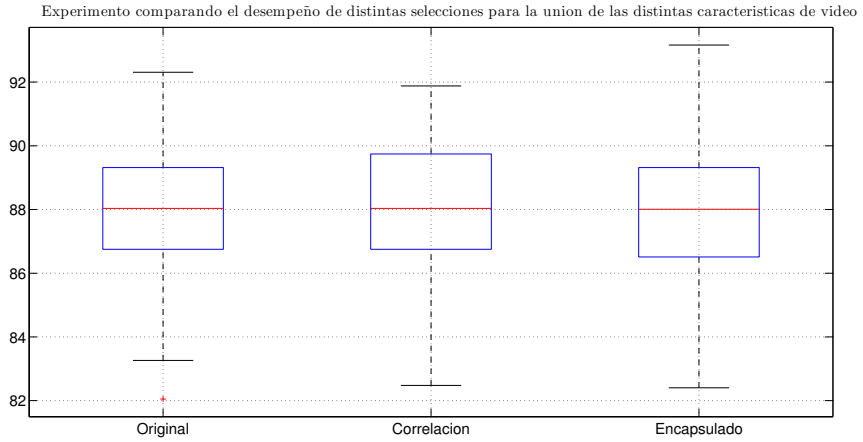


Figura 6.5: Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado sobre la unión de características de video. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

Dados los resultados obtenidos en las sucesivas etapas, se decidió elegir al conjunto $\mathrm{S_C^{geo+DCT}}$ como la selección final de características del modo video, dado que esta representa la unión de los conjuntos de selección. Dicho conjunto se llamará de ahora en más $\mathrm{S_{video}}$, por tratarse de la selección final de este modo. Un resumen de las características seleccionadas se presenta en la Tabla 6.7.

De forma de comparar los distintos conjuntos que se consideraron en el proceso de selección de video, se realizó un último experimento utilizando un test de Student (con iguales parámetros que los anteriores). Estas pruebas se presentan en la Figura 6.6 y resumen el total de las pruebas realizadas para este modo.

$\text{posYPalo}_{\text{norm}}$	$\text{DCT}_{\text{P}}1$
$\text{posYMano}_{\text{norm}}$	$\text{DCT}_{\text{P}}2$
$\text{posXPalo}_{\text{norm}}$	$\text{DCT}_{\text{P}}3$
$\text{min}_{\text{M}}\text{d}1$	$\text{DCT}_{\text{P}}4$
$\text{ceros}_{\text{P}}\text{d}1$	$\text{DCT}_{\text{P}}5$
$\text{DCT}_{\text{M}}2$	$\text{DCT}_{\text{P}}6$
$\text{DCT}_{\text{M}}3$	$\text{DCT}_{\text{P}}7$

Tabla 6.7: Selección de video .

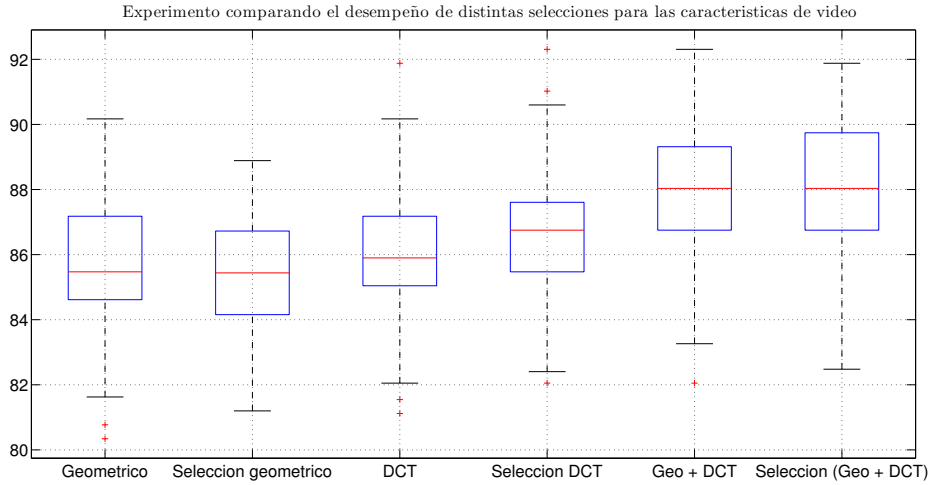


Figura 6.6: Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado utilizando 6 conjuntos de características. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

6.2.4. Determinación de parámetros óptimos

Una vez determinada la selección de características del modo video se realizó una búsqueda exhaustiva de los mejores parámetros de tres algoritmos, de manera análoga a lo realizado en el modo audio. Los resultados obtenidos se resumen en la Tabla 6.8.

En la Figura 6.7 se muestra el experimento realizado comparando los tres algoritmos de clasificación. Se determinó la utilización del algoritmo SVM para realizar la estimación de desempeño por presentar un mejor desempeño.

Clasificador	*Parámetros*	
C4.5	$C = 0{,}1$	$m = 2$
SVM	$C = 4$	$\gamma = 6$
K-NN	$K = 3$	- -

Tabla 6.8: Parámetros óptimos de la selección de video.

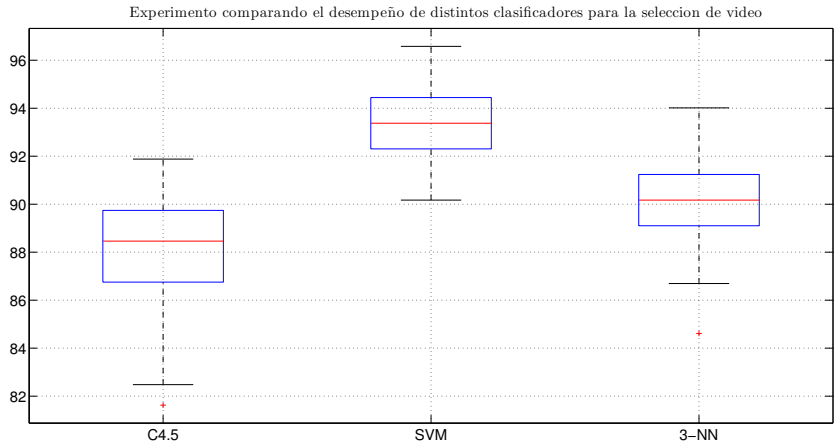


Figura 6.7: Test de Student con un nivel de significancia de 95 %, realizado para la selección final de video, utilizando tres clasificadores. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

6.3. Enfoque multimodal

El desarrollo del sistema multimodal requirió el estudio de las posibles técnicas de combinación de la información extraída de cada modo. Se resolvió utilizar el enfoque de *feature level fusion*, es decir, se combinó la información proveniente de cada modo a nivel de características [40].

Cabe aclarar que el enfoque utilizado no es estrictamente *feature level fusion*, en el sentido que las características de cada modo no fueron calculadas independientemente una de otra, sino que se usa información del audio para guiar el cálculo de las características de video. La idea es que la ubicación temporal de los golpes de percusión puede establecerse fácilmente a partir de la señal de audio, como se hace en [38]. Como se explicó anteriormente, esta información se utilizó para etiquetar los videos y las etiquetas fueron utilizadas para calcular las características de los modos en un entorno de cada golpe. La ventaja de este método es que se reduce el costo computacional, ya que el cálculo no se realiza sobre las señales enteras, sino que se procesan algunas tramas de la señal en el entorno de cada evento detectado.

Se implementaron dos enfoques diferentes para combinar las características de

los distintos modos. El primero consistió en unir las características seleccionadas en cada modo en un único conjunto, como se muestra en la Figura 6.8. De aquí en adelante se referirá a este método como Multimodal 1.

Por otro lado, en el segundo enfoque se unieron el total de las características extraídas de cada modo en un único conjunto, para luego realizar selección por el mismo procedimiento utilizado para cada modo por separado. Este método será referido de aquí en más como Multimodal 2. La Figura 6.9 muestra el diagrama de bloques para el método Multimodal 2.

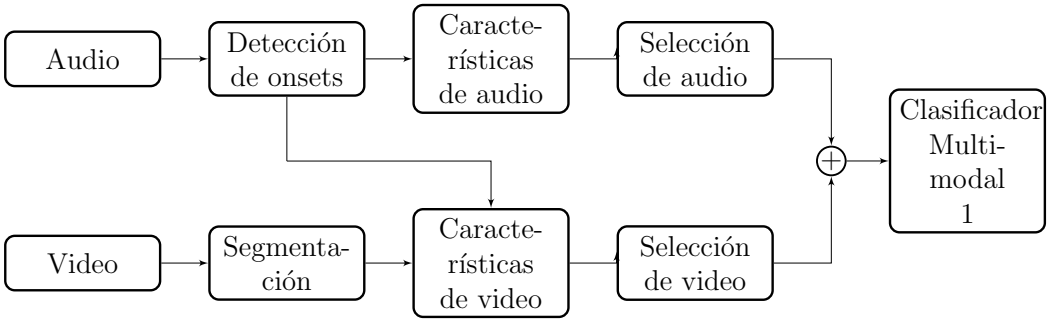


Figura 6.8: Diagrama de bloques del método Multimodal 1. El símbolo ⊕ representa la unión de conjuntos.

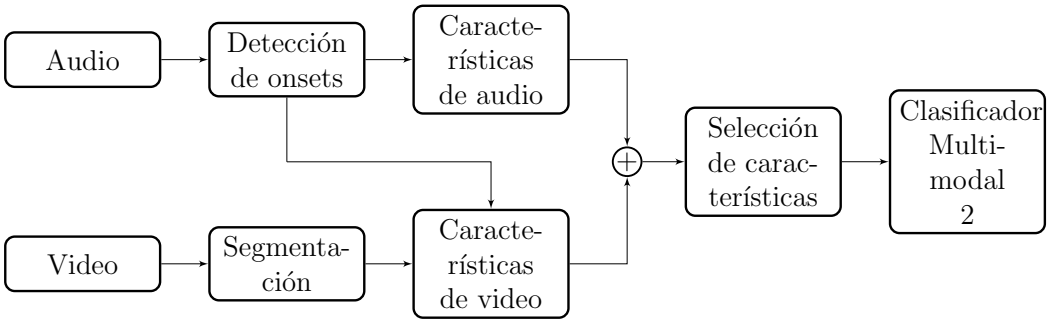


Figura 6.9: Diagrama de bloques del método Multimodal 2. El símbolo ⊕ representa la unión de conjuntos.

6.3.1. Combinación de las características extraídas de cada modo

Los métodos de combinación de las características extraídas de cada modo se resumen en las Figuras 6.8 y 6.9.

El método Multimodal 1 consistió en unir las selecciones de audio y video, por lo que no se volvió a aplicar selección. Las características consideradas se resumen en la Tabla 6.9.

En el segundo caso (Multimodal 2), el total de las características calculadas en cada modo se unió en un solo conjunto, sobre el cual se aplicó selección de características. Los métodos de selección fueron los mismos que los utilizados en

$spec_{centroid}$	$mfcc_8$	$mfcc_{25}$	$posXPalo_{norm}$
$spec_{spread}$	$mfcc_9$	$mfcc_{28}$	min_Md1
$spec_{skewness}$	$mfcc_{13}$	$mfcc_{29}$	$ceros_Pd1$
$spec_{kurtosis}$	$mfcc_{14}$	$mfcc_{31}$	DCT_M2
$spec_{decrease}$	$mfcc_{15}$	$mfcc_{34}$	DCT_M3
sf1	$mfcc_{16}$	$mfcc_{35}$	DCT_P1
sf_2	$mfcc_{17}$	$mfcc_{36}$	DCT_P2
$mfcc_1$	$mfcc_{18}$	$mfcc_{37}$	DCT_P3
$mfcc_2$	$mfcc_{19}$	$mfcc_{38}$	DCT_P4
$mfcc_3$	$mfcc_{20}$	$mfcc_{39}$	DCT_P5
$mfcc_4$	$mfcc_{21}$	$mfcc_{40}$	DCT_P6
$mfcc_5$	$mfcc_{23}$	$posYPalo_{norm}$	DCT_P7
$mfcc_7$	$mfcc_{24}$	$posYMano_{norm}$	

Tabla 6.9: Conjunto Multimodal 1.

cada modo por separado (uno basado en correlación y otro por encapsulado). Nuevamente para este último se utilizó un árbol C4.5 con parámetros por defecto. En la Tabla 6.10 se muestran los resultados de la selección para este caso.

Para tener evidencia estadística significativa de las diferencias de desempeño entre las selecciones, se realizó un experimento utilizando 10 veces *10-fold-CV*, de forma análoga a las selecciones de cada modo. En este experimento se compararon los conjuntos S_C^{M2}, S_E^{M2}, su unión (S_U^{M2}) y el conjunto original de las 79 características provenientes de los modos audio y video. Para ello se utilizó un árbol del tipo C4.5 con parámetros por defecto y un test de Student con un nivel de significancia de 95 % para la estimación. Los resultados se presentan en la Figura 6.10.

Se resolvió entonces considerar al conjunto formado por la unión de ambos métodos (correlación y encapsulado) como la selección de Multimodal 2. El conjunto se presenta en la Tabla 6.11.

Como se observa en las Tablas 6.9 y 6.11, las selecciones por ambos enfoques son muy similares. En particular, el conjunto Multimodal 1 tiene 51 características, mientras que el Multimodal 2 tiene 47. Los coeficientes MFCCs seleccionados en cada caso son casi los mismos, y algo similar sucede con la DCT. La principal diferencia está en las características del conjunto de audio denotadas como *spec* y en el conjunto geométrico de video.

Con el objetivo de decidir cuál método resultaba más útil para la solución del problema, se evaluó el desempeño de la clasificación utilizando como conjunto de entrenamiento los registros de palo rojo de la base eMe y tres registros de prueba: los registros de la base eMe de palo verde y el único registro etiquetado de la base Zavala. Como clasificador se utilizó el algoritmo SVM con un kernel de base radial Gaussiana, ya que fue con el que se obtuvieron mejores resultados para cada modo por separado. Se realizó una búsqueda exhaustiva de sus mejores parámetros, de manera análoga a lo que se hizo para los clasificadores unimodales. Los mismos se resumen en la Tabla 6.12.

Características	*Corr*	*Enc*	*Características*	*Corr*	*Enc*	*Características*	*Corr*	*Enc*
$posYPalo_{norm}$	10	2	DCT_P7	10	1	$mfcc_{14}$	10	0
$posYMano_{norm}$	10	8	DCT_P8	0	0	$mfcc_{15}$	10	1
$posXPalo_{norm}$	10	1	DCT_P9	0	4	$mfcc_{16}$	3	1
$posXMano_{norm}$	5	1	DCT_P10	1	1	$mfcc_{17}$	10	0
min_Md1	10	0	$spec_{centroid}$	10	1	$mfcc_{18}$	10	1
max_Md1	0	3	$spec_{spread}$	10	1	$mfcc_{19}$	10	1
min_Pd1	10	1	$spec_{skewness}$	3	1	$mfcc_{20}$	9	0
max_Pd1	0	0	$spec_{kurtosis}$	7	0	$mfcc_{21}$	10	2
$ceros_Md1$	10	3	$spec_{decrease}$	7	9	$mfcc_{22}$	0	1
$ceros_Pd1$	10	0	$spec_{slope}$	1	0	$mfcc_{23}$	10	2
DCT_M1	0	1	$spec_{crest}$	0	0	$mfcc_{24}$	10	2
DCT_M2	10	7	sf_1	10	2	$mfcc_{25}$	10	0
DCT_M3	10	1	sf_2	10	6	$mfcc_{26}$	5	0
DCT_M4	0	2	$mfccs_1$	10	1	$mfcc_{27}$	6	0
DCT_M5	0	2	$mfcc_2$	10	1	$mfcc_{28}$	10	0
DCT_M6	0	1	$mfcc_3$	10	0	$mfcc_{29}$	10	0
DCT_M7	0	1	$mfcc_4$	10	2	$mfcc_{30}$	2	0
DCT_M8	0	2	$mfcc_5$	1	0	$mfcc_{31}$	10	0
DCT_M9	0	0	$mfcc_6$	1	4	$mfcc_{32}$	0	0
DCT_M10	0	0	$mfcc_7$	5	2	$mfcc_{33}$	0	0
DCT_P1	10	9	$mfcc_8$	8	1	$mfcc_{34}$	10	0
DCT_P2	10	10	$mfcc_9$	10	1	$mfcc_{35}$	10	0
DCT_P3	10	10	$mfcc_{10}$	3	0	$mfcc_{36}$	9	0
DCT_P4	0	0	$mfcc_{11}$	2	0	$mfcc_{37}$	10	0
DCT_P5	10	1	$mfcc_{12}$	0	0	$mfcc_{38}$	10	0
DCT_P6	10	3	$mfcc_{13}$	9	3	$mfcc_{39}$	9	1
						$mfcc_{40}$	10	1

Tabla 6.10: Selección por correlación y por encapsulado de todas las características extraídas de los modos audio y video.

Los resultados de la clasificación se muestran en la Tabla 6.13. Como puede observarse, los desempeños de cada conjunto son similares para los registros de la base eMe. Sin embargo, Multimodal 1 presenta un incremento de 5.65 % sobre Multimodal 2 cuando se valida el clasificador con el registro de la base Zavala. Por esta razón se resolvió utilizar como selección al conjunto Multimodal 1, que será referido de ahora en más como $S_{multimodal}$.

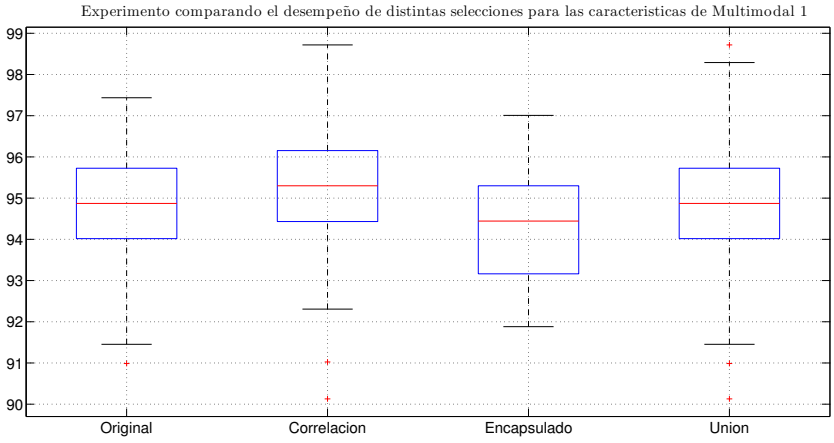


Figura 6.10: ETest de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, comparando las distintas selecciones de las características de Multimodal 2. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers.

$\text{spec}_{\text{centroid}}$	mfcc_{14}	mfcc_{31}	$\text{min}_{\text{M}}\text{d1}$
$\text{spec}_{\text{spread}}$	mfcc_{15}	mfcc_{34}	$\text{min}_{\text{P}}\text{d1}$
$\text{spec}_{\text{decrease}}$	mfcc_{17}	mfcc_{35}	$\text{ceros}_{\text{M}}\text{d1}$
sf1	mfcc_{18}	mfcc_{36}	$\text{ceros}_{\text{P}}\text{d1}$
sf_2	mfcc_{19}	mfcc_{37}	$\text{DCT}_{\text{M}}2$
mfcc_1	mfcc_{20}	mfcc_{38}	$\text{DCT}_{\text{M}}3$
mfcc_2	mfcc_{21}	mfcc_{39}	$\text{DCT}_{\text{P}}1$
mfcc_3	mfcc_{23}	mfcc_{40}	$\text{DCT}_{\text{P}}2$
mfcc_4	mfcc_{24}	$\text{posYPalo}_{\text{norm}}$	$\text{DCT}_{\text{P}}3$
mfcc_8	mfcc_{25}	$\text{posYMano}_{\text{norm}}$	$\text{DCT}_{\text{P}}5$
mfcc_9	mfcc_{28}	$\text{posXPalo}_{\text{norm}}$	$\text{DCT}_{\text{P}}6$
mfcc_{13}	mfcc_{29}		$\text{DCT}_{\text{P}}7$

Tabla 6.11: Conjunto de características obtenidas por el método Multimodal 2.

Conjuntos	C	γ
Multimodal 1	6	0,4
Multimodal 2	7,4	0,3

Tabla 6.12: Parámetros óptimos de las selecciones Multimodal 1 y Multimodal 2, para SVM usando un kernel RBF.

Entrenamiento	*Validación*	*Multimodal 1*	*Multimodal 2*
Intérpretes 2 y 3	Intérprete 1	88,68 %	88,79 %
Intérpretes 2 y 3	Intérprete 4	87,54 %	88,46 %
Intérpretes 2 y 3	Intérprete Zavala	67,80 %	62,15 %

Tabla 6.13: Resultados de la clasificación usando las características de Multimodal 1 y Multimodal 2.

Capítulo 7

Evaluación de desempeño

La estimación de desempeño se realizó incluyendo los registros con palo verde, los cuales no se habían tenido en cuenta en el proceso de selección y determinación de parámetros. Para estas pruebas no se incluyeron datos del dataset Zavala. Ya que se cuenta únicamente con una toma etiquetada de ese registro, se decidió reservarla para validar los datos cuando el entrenamiento se realiza con los registros de palo rojo.

Se decidió dividir los datasets por intérpretes, utilizando 3 de ellos para entrenamiento y el restante para validar. Variando qué conjunto de intérpretes se considera para entrenar y cuál para validar, pueden extraerse conclusiones significativas respecto a cuáles tipos de golpes son los más fácilmente clasificables y cuáles los menos. Además, es posible analizar cómo afecta al desempeño la inclusión o no de algún intérprete en el conjunto de entrenamiento.

La evaluación se realizó de dos formas distintas: en primer instancia se clasificó entre palo, mano y madera y luego se distinguió además entre golpes de borde, rebotados o flam. Cabe aclarar que los golpes de borde y rebotado se consideraron como golpes de palo en la primer instancia, mientras que el flam no se consideró.

7.1. Evaluación considerando tres tipos de golpes

Una primera aproximación a la solución del problema se llevó a cabo considerando únicamente tres tipos de golpes: palo, mano y madera.

7.1.1. Audio

Los resultados de la clasificación para diferentes conjuntos de entrenamiento y test se presentan en Tabla 7.1. Analizando las matrices de confusión puede verse que la clase madera resulta ser la que tiene menor error de clasificación. Esto puede atribuirse a que el golpe de madera es similar entre las distintas interpretaciones, mientras que los golpes de mano y palo varían significativamente intérprete a intérprete. Además, el golpe de madera es el que más se diferencia de los otros golpes, hecho que también explica el resultado anterior. En los casos en los que

la madera es mal clasificada, la mayoría de las veces se confunde con un golpe de palo, lo que es esperable ya que el sonido de madera se asemeja más a uno de golpe de palo que a uno de mano. Esto tiene sentido además teniendo en cuenta que los golpes de borde en la clasificación con 3 clases están contados como palo, y que los golpes de madera y borde son los que tienen un sonido más similar.

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*			
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 2 y 3	Intérprete 4	89,49 %	68	0	0	$\|a =$ madera
			0	339	29	$\|b =$ mano
			1	59	351	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 3 y 4	Intérprete 2	90,19 %	173	0	11	$\|a =$ madera
			0	297	82	$\|b =$ mano
			1	7	459	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 2 y 4	Intérprete 3	96,82 %	182	0	0	$\|a =$ madera
			0	267	9	$\|b =$ mano
			0	19	404	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 2, 3 y 4	Intérprete 1	68,90 %	178	0	4	$\|a =$ madera
			2	114	160	$\|b =$ mano
			7	101	315	$\|c =$ palo

Tabla 7.1: Resultados de la clasificación unimodal con el audio, considerando tres tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 1$.

También en las matrices de confusión se puede ver que las clases que más se confunden son palo y mano. Esto resulta coherente con lo que se espera de la clasificación, ya que hasta para un oído entrenado resulta difícil desambiguar estos golpes sólo a partir del audio.

Es destacable el hecho de que los mayores errores de clasificación se obtienen cuando se utiliza al Intérprete 1 como conjunto de prueba. Puede observarse que en estos casos la mayoría de los golpes de mano se clasifican incorrectamente como golpes de palo. Esto puede explicarse teniendo en cuenta que el Intérprete 1 tocó con una afinación un poco más baja que el resto y que por lo general, un golpe de mano tiene un sonido más grave que uno de palo.

7.1.2. Video

La evaluación de desempeño obtenida para el modo video se presenta en la Tabla 7.2.

Como puede observarse, se obtienen porcentajes de acierto muy altos, lo que resulta coherente ya que en el video las clases madera, mano y palo son fácilmente distinguibles. Además, al igual que en el audio, el hecho de agrupar varias clases

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*			
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 2 y 3	Intérprete 4	95,87 %	68	0	0	$\|a =$ madera
			0	351	17	$\|b =$ mano
			11	7	393	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 3 y 4	Intérprete 2	94,07 %	131	21	32	$\|a =$ madera
			0	378	1	$\|b =$ mano
			0	7	460	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 2 y 4	Intérprete 3	98,83 %	132	0	1	$\|a =$ madera
			0	454	4	$\|b =$ mano
			0	10	689	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 2, 3 y 4	Intérprete 1	99,32 %	182	0	0	$\|a =$ madera
			1	273	2	$\|b =$ mano
			0	3	420	$\|c =$ palo

Tabla 7.2: Resultados de la clasificación unimodal con el video, considerando tres tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 6$

bajo la categoría palo reduce significativamente la dificultad del problema, haciendo que la información de un único modo sea suficiente para lograr porcentajes de clasificación muy cercanos al 100 %.

Es de destacar lo que sucede cuando se toma al Intérprete 2 como conjunto de test: la madera se confunde con las otras clases más frecuentemente que para los otros intérpretes y que para la misma intérprete considerando sólo el audio. Estas diferencias pueden ser atribuidas a problemas con las detecciones en el video, tanto en la ubicación de la punta del palo como en la segmentación de la mano izquierda.

7.1.3. Enfoque multimodal

En la Tabla 7.3 se presentan los resultados obtenidos utilizando tres clases para la clasificación. Nuevamente, se consideraron los golpes de borde y rebotado como pertenecientes a la clase palo y se descartaron los golpes de flam.

En la Tabla 7.4 se muestra la comparación de los desempeños obtenidos en los modos audio y video, y en la combinación multimodal, considerando tres clases. El desempeño del conjunto de características multimodal es levemente mejor que los unimodales, salvo para el caso en que se clasifica al Intérprete 1. En este caso, el desempeño utilizando solamente el video fue mejor, lo que tiene sentido teniendo en cuenta que en el audio se tenía un desempeño pobre en relación a las pruebas con los demás intérpretes. Como fue mencionado, la afinación de las tomas del Intérprete 1 era muy diferente al resto. Lo interesante de este resultado es que aunque se tiene un registro cuyo porcentaje de clasificación es malo en uno de los modos por separado, el clasificador multimodal tiene igualmente un buen

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*			
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 2 y 3	Intérprete 4	97,40 %	68	0	0	$\|a =$ madera
			0	359	9	$\|b =$ mano
			0	13	398	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 3 y 4	Intérprete 2	98,35 %	174	0	10	$\|a =$ madera
			0	377	2	$\|b =$ mano
			0	5	462	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 1, 2 y 4	Intérprete 3	99,22 %	133	0	0	$\|a =$ madera
			0	456	2	$\|b =$ mano
			1	7	691	$\|c =$ palo
			a	b	c	$\leftarrow$ clasificación
Intérpretes 2, 3 y 4	Intérprete 1	96,93 %	182	0	0	$\|a =$ madera
			0	267	9	$\|b =$ mano
			0	18	405	$\|c =$ palo

Tabla 7.3: Resultados de la clasificación Multimodal 1 considerando tres tipos de golpes. Se utilizó un clasificador SVM con $C = 6$ y un kernel RBF de $\gamma = 0{,}4$

desempeño.

Entrenamiento	*Validación*	*Audio (%)*	*Video (%)*	*Multimodal (%)*
Intérpretes 1, 2 y 3	Intérprete 4	89,49	95,87	97,40
Intérpretes 1, 3 y 4	Intérprete 2	96,82	98,83	99,22
Intérpretes 1, 2 y 4	Intérprete 3	90,19	94,07	98,35
Intérpretes 2, 3 y 4	Intérprete 1	68,90	99,32	96,93

Tabla 7.4: Comparación de los porcentajes de clasificación obtenidos en los modos audio y video, y en la combinación multimodal, utilizando 3 clases.

7.2. Evaluaciones considerando seis tipos de golpe

Se agregaron tres clases o tipos de golpes a considerar: rebotado, borde y flam.

Nuevamente, se realizó el entrenamiento con los datos de tres de los intérpretes, validando con el restante. Se variaron estos conjuntos de manera de abarcar todas las combinaciones posibles de parejas entrenamiento/prueba.

7.2.1. Audio

Los resultados de la evaluación utilizando seis clases para el modo audio se muestran en la Tabla 7.5.

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*						
			a	b	c	d	e	f	$\leftarrow$ clasificación
			68	0	0	0	0	0	$\|a =$ madera
			0	348	17	1	2	0	$\|b =$ mano
Intérpretes 1, 2 y 3	Intérprete 4	83,66 %	0	55	226	21	0	0	$\|c =$ palo
			0	1	6	43	0	4	$\|d =$ rebotado
			1	8	2	0	44	0	$\|e =$ borde
			0	4	2	17	2	3	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			173	0	0	0	11	0	$\|a =$ madera
			0	329	32	7	11	0	$\|b =$ mano
Intérpretes 1, 3 y 4	Intérprete 2	87,64 %	0	7	205	14	17	0	$\|c =$ palo
			0	0	15	57	0	0	$\|d =$ rebotado
			1	0	7	1	143	0	$\|e =$ borde
			0	0	0	4	1	1	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			128	0	0	0	5	0	$\|a =$ madera
			0	421	36	0	0	1	$\|b =$ mano
Intérpretes 1, 2 y 4	Intérprete 3	88,18 %	0	32	214	2	1	0	$\|c =$ palo
			0	11	26	114	1	16	$\|d =$ rebotado
			7	1	2	0	272	0	$\|e =$ borde
			0	11	0	1	1	0	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			182	0	0	0	0	0	$\|a =$ madera
			5	188	44	38	0	1	$\|b =$ mano
Intérpretes 2, 3 y 4	Intérprete 1	60,06 %	8	142	164	26	0	0	$\|c =$ palo
			0	2	4	7	0	0	$\|d =$ rebotado
			15	42	3	0	10	0	$\|e =$ borde
			7	11	2	17	0	1	$\|f =$ flam

Tabla 7.5: Resultados de la clasificación unimodal con el audio, considerando seis tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 1$

Al igual que en la sección anterior, la clase madera resulta ser la que tiene menor error de clasificación. La diferencia es que en este caso los golpes mal clasificados, son asignados a la clase borde. Esto es esperable debido a que, de todos los golpes, el de borde es el que tiene un sonido más agudo, similar al de la madera.

La inclusión de tres clases adicionales causa además que los golpes de mano se clasifiquen de mejor manera: mientras que para tres tipos de golpes el promedio de golpes de mano bien clasificados fue de 78,41 %, agregando los tres golpes adicionales este valor crece a 87,19 %. La explicación puede deberse a que, al dividir la clase palo en sub clases, se generan límites de decisión más simples y que son más facilmente aproximables por el clasificador.

Otra discusión interesante es la que se plantea a partir del análisis de desempeño cambiando los datos de entrenamiento y test rotando los diferentes intérpretes. Al considerar los 6 golpes los resultados son consistentes con el caso anterior de 3 golpes, es decir, el Intérprete 1 presenta una diferencia con el resto. Probablemente esto se deba a la diferencia en la afinación.

Este hecho plantea además una cuestión interesante y es que dada una cierta base de datos, diferencias en la afinacion de la lonja o en las condiciones acústi-

cas del lugar donde se realizó el registro pueden generar gran variabilidad en los resultados. Por lo tanto si se quiere un sistema con mayor generalidad debería validarse el algoritmo con un conjunto de datos aún menos correlacionado con los datos de entrenamiento, por ejemplo, provenientes de otra instancia de grabación. En la sección 7.3 se analizará el desempeño del sistema cuando se lo enfrenta a un registro no utilizado para su desarrollo, proveniente de la base Zavala.

7.2.2. Video

La evaluación de desempeño en este caso se presenta en la Tabla 7.6.

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*						
			a	b	c	d	e	f	$\leftarrow$ clasificación
			68	0	0	0	0	0	$\|a =$ madera
			0	356	2	0	3	7	$\|b =$ mano
Intérpretes 1, 2 y 3	Intérprete 4	74,28 %	9	34	174	46	39	0	$\|c =$ palo
			11	7	14	16	6	0	$\|d =$ rebotado
			0	1	18	3	33	0	$\|e =$ borde
			0	16	3	0	6	3	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			135	23	14	10	1	1	$\|a =$ madera
			0	378	1	0	0	0	$\|b =$ mano
Intérpretes 1, 3 y 4	Intérprete 2	75,77 %	0	2	172	8	61	0	$\|c =$ palo
			0	4	53	7	8	0	$\|d =$ rebotado
			0	6	47	1	93	5	$\|e =$ borde
			0	2	2	0	2	0	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			132	0	0	0	0	1	$\|a =$ madera
			0	457	0	0	0	1	$\|b =$ mano
Intérpretes 1, 2 y 4	Intérprete 3	77,67 %	0	3	232	2	8	4	$\|c =$ palo
			0	14	101	52	1	0	$\|d =$ rebotado
			0	1	120	12	132	17	$\|e =$ borde
			0	3	0	0	3	7	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			182	0	0	0	0	0	$\|a =$ madera
			1	274	1	0	0	0	$\|b =$ mano
Intérpretes 2, 3 y 4	Intérprete 1	88,03 %	0	3	270	5	60	2	$\|c =$ palo
			0	1	2	8	3	0	$\|d =$ rebotado
			0	1	2	0	65	2	$\|e =$ borde
			0	15	0	0	13	10	$\|f =$ flam

Tabla 7.6: Resultados de la clasificación unimodal con el video, considerando seis tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 6$

Como se observa en la Tabla 7.6, el desempeño de la clasificación con seis clases disminuye respecto al de tres clases. Observando las matrices de confusión, el golpe que presenta mayor error de clasificación es el rebotado. Desde el punto de vista del video, la discriminación de este tipo de golpe es muy exigente ya que para distinguirlo de un golpe de palo se requiere información precisa sobre las oscilaciones de la punta. El hecho de que las detecciones presenten ruido y el haber realizado un suavizado para la determinación de algunas de las características

puede explicar que dicho golpe se clasifique correctamente sólo en un 27,95 % de los casos. Contrastando este resultado, en el audio se tiene un porcentaje de clasificación del 71,99 %. Esto responde a que el golpe rebotado presenta cualidades sonoras fácilmente distinguibles del resto de los golpes de palo.

Otra clase que presenta un alto error de clasificación es el borde, el cual es mal clasificado en un 42,22 % de los casos. La ejecución de este tipo de golpe varía significativamente entre intérpretes, lo cual dificulta la detección en el video. Nuevamente, en el audio no se tiene este problema ya que el golpe de borde es tímbricamente similar entre los distintos intérpretes, teniéndose un error de clasificación de tan solo 15,95 %.

Como era de esperar, la clasificación del golpe de madera en el video presenta buenos resultados. La madera es el golpe con mayor diferencia respecto al resto de los golpes de palo, ya que es el único en el que la punta del mismo está por debajo de la lonja. Los peores resultados para este golpe se obtienen al clasificar los ejecutados por el Intérprete 2, al igual que en la clasificación con tres clases.

En contraste con el audio, el intérprete con mayor porcentaje de aciertos en la clasificación de video es el Intérprete 1. En este caso, no se presenta el problema de los cambios de afinación entre las tomas, lo que ocasionaba problemas en el audio. A su vez, los videos del Intérprete 1 tienen pocos golpes rebotados y de borde respecto al resto de los videos, lo que explica que la clasificación sea buena.

Una diferencia significativa respecto al audio es que la mano tiene alto porcentaje de clasificación. En el audio se tenía un porcentaje de aciertos del 87,13 %, mientras que en el video se logra un 98,99 %.

Al igual que en el audio, los golpes de flam en general no son bien clasificados. Dicho resultado puede explicarse teniendo en cuenta que son pocas las muestras de este tipo de golpe. A su vez, como se explicó en la Sección 4.4 existe un retardo significativo en las detecciones de mano, y en el caso de los golpes de flam esto es crítico.

7.2.3. Enfoque multimodal

La Tabla 7.7 muestra los resultados de la clasificación utilizando seis clases, mientras que en la Tabla 7.8 se muestra la comparación de los desempeños obtenidos en los modos audio y video, y en la combinación multimodal.

Como se observa, el desempeño utilizando el conjunto de características multimodal en todos los casos mejora respecto a los desempeños unimodales. En el caso del audio y el video, se tiene un promedio de desempeño de 79,89 % y 78,94 % respectivamente, mientras que con el enfoque multimodal se tiene un promedio de 92,34 %.

El hecho de que exista una mejora en el desempeño de clasificación cuando se consideran ambos modos en conjunto implica que se logran compensar las carencias de cada modo por separado. En la Tabla 7.9 se muestra el porcentaje promedio de acierto para cada golpe, diferenciando por modo de información y comparando contra el enfoque multimodal. Allí puede verse que el uso de un enfoque multimodal hace que se clasifiquen de mejor manera todos los tipos de golpes, obteniendo

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*						
			a	b	c	d	e	f	← clasificación
			68	0	0	0	0	0	$\|a =$ madera
			0	361	0	2	3	2	$\|b =$ mano
Intérpretes 1, 2 y 3	Intérprete 4	89,48 %	0	10	258	33	1	0	$\|c =$ palo
			0	0	13	41	0	0	$\|d =$ rebotado
			0	1	5	0	49	0	$\|e =$ borde
			0	5	1	13	3	6	$\|f =$ flam
			a	b	c	d	e	f	← clasificación
			132	0	0	0	0	0	$\|a =$ madera
			0	457	1	0	0	0	$\|b =$ mano
Intérpretes 1, 3 y 4	Intérprete 2	91,22 %	0	9	236	3	1	0	$\|c =$ palo
			0	0	24	139	1	4	$\|d =$ rebotado
			1	0	3	0	278	0	$\|e =$ borde
			0	4	1	0	0	8	$\|f =$ flam
			a	b	c	d	e	f	← clasificación
			174	0	0	0	10	0	$\|a =$ madera
			0	378	1	0	0	0	$\|b =$ mano
Intérpretes 1, 2 y 4	Intérprete 3	95,93 %	0	3	215	9	15	1	$\|c =$ palo
			0	1	36	35	0	0	$\|d =$ rebotado
			0	2	10	1	138	1	$\|e =$ borde
			0	0	0	0	1	5	$\|f =$ flam
			a	b	c	d	e	f	← clasificación
			182	0	0	0	0	0	$\|a =$ madera
			1	272	0	3	0	0	$\|b =$ mano
Intérpretes 2, 3 y 4	Inérprete 1	92,71 %	0	22	310	4	3	1	$\|c =$ palo
			0	1	3	9	0	0	$\|d =$ rebotado
			0	1	5	3	59	2	$\|e =$ borde
			0	4	3	11	0	20	$\|f =$ flam

Tabla 7.7: Resultados de la clasificación Multimodal 1 con 6 tipos de golpes. Se utilizó un clasificador SVM con $C = 6$ y un kernel RBF de $\gamma = 0{,}4$.

Entrenamiento	*Validación*	*Audio (%)*	*Video (%)*	*Multimodal (%)*
Intérpretes 1, 2 y 3	Intérprete 4	83.66	74.28	89.48
Intérpretes 1, 3 y 4	Intérprete 2	88.18	77.67	95.93
Intérpretes 1, 2 y 4	Intérprete 3	87.64	75.77	91.22
Intérpretes 2, 3 y 4	Intérprete 1	60.06	88.03	92.71

Tabla 7.8: Comparación de los porcentajes de clasificación obtenidos en los modos audio y video, y en la combinación multimodal, utilizando 6 clases.

siempre un desempeño superior a cualquiera de los modos por separado.

Es destacable el caso del flam. Si bien el promedio de acierto en cada modo resultó extremadamente bajo (6,25 % para el audio y 23,53 % para el video), mejora notablemente cuando se combina la información procedente de cada uno. Sin embargo, el desempeño para el caso multimodal no llega a superar el 50 %. Esto puede deberse a dos cosas. Por un lado, el flam es el golpe del cual se tienen menos realizaciones, lo que implica que se tengan pocos datos para entrenar el algoritmo. Por otro lado, la ejecución del golpe varía entre intérpretes: en algunos casos el

golpe de palo que acompaña a la mano es un golpe simple, mientras que en otros es rebotado.

Golpe	*Audio(%)*	*Video(%)*	*Multimodal(%)*
Madera	97.18	91.18	98.23
Mano	86.63	98.92	99.19
Palo	71.34	74.78	89.86
Rebotado	71.75	26.95	76.19
Borde	83.90	57.78	93.74
Flam	6.25	23.53	45.88

Tabla 7.9: Comparación del porcentaje de clasificación correcta de los modos audio y video frente al enfoque multimodal para cada tipo de golpe.

7.3. Evaluación sobre un registro de la base Zavala

Las pruebas anteriores determinaron qué grado de generalización tienen las características propuestas en la base eMe, obteniendo porcentajes promedios de 97.97 % considerando tres clases y 92.33 % considerando seis. Sin embargo, es interesante estimar qué grado de generalidad presenta este sistema al entrenar con una base de datos y validar con otra. Como se evidenció en la Tabla 6.13, al realizar la clasificación sobre el registro perteneciente a la base Zavala se obtuvo un desempeño considerablemente menor.

El bajo desempeño en este caso motivó que se estudiaron nuevamente las características del conjunto $S_{\text{multimodal}}$ determinando cuáles de ellas eran fuertemente dependientes de las condiciones de grabación. Se constató entonces que dos de ellas no eran apropiadas si se deseaba obtener un conjunto invariante a distintas condiciones de grabación: DCT_P1 y $mfcc_1$. La primera se refiere a la posición media de la coordenada y de la punta del palo, por lo que al no estar referida a ningún elemento de la escena, resulta dependiente del video utilizado. La segunda es el valor de continua del espectro de la señal de audio, por lo que en este caso la característica depende de las condiciones de grabación del audio de cada toma. Dado que los registros de la base eMe fueron grabados en un mismo rodaje, éstos presentan cualidades similares (excepto por una de las tomas del Intérprete 1, como se explicó en la sección 7.1.1). El registro de la base Zavala fue grabado con otro equipamiento y en un ambiente con acústica diferente, ocasionando que las condiciones de grabación sean inevitablemente diferentes.

Fue así que se estudió el desempeño del sistema multimodal cuando no se consideran ambas características. Los resultados fueron obtenidos clasificando con SVM (con los parámetros óptimos determinados) y se presentan en la Figura 7.10.

Se puede observar que la exclusión de las dos características genera un aumento del 15,8 % en el desempeño y una mejor discriminación de la clase palo. Resulta razonable, ya que una diferencia importante es que los videos de la base eMe fueron

Características	*Desempeño*	*Matriz de confusión*						
		a	b	c	d	e	f	← clasificación
		36	0	0	0	0	0	$\|a =$ madera
		0	62	0	0	0	0	$\|b =$ mano
$S_{multimodal}$	66,67 %	1	11	19	25	11	0	$\|c =$ palo
		1	0	0	1	7	0	$\|d =$ rebotado
		0	0	0	0	0	0	$\|e =$ borde
		0	3	0	0	0	0	$\|f =$ flam
		a	b	c	d	e	f	← clasificación
		36	0	0	0	10	0	$\|a =$ madera
		0	61	0	1	0	0	$\|b =$ mano
$S_{multimodal}$ sin DCT_P1 y $mfcc_1$	82,48 %	0	1	47	12	7	0	$\|c =$ palo
		0	0	0	1	8	0	$\|d =$ rebotado
		0	0	0	0	0	0	$\|e =$ borde
		0	1	1	0	1	0	$\|f =$ flam

Tabla 7.10: Resultado de entrenar con registros de palo rojo de la base eMe y validar con el registro de base la Zavala. En el primer caso se considera el conjunto $S_{multimodal}$ completo. En el segundo, las características DCT_P1 y $mfcc_1$ se quitaron del conjunto. Ambos resultados se obtuvieron utilizando SVM con los parámetros óptimos determinados en la sección 6.3.1.

grabados con la cámara en posición horizontal mientras que los de la Zavala se filmaron con las cámaras en posición vertical. Por lo tanto, la característica DCT_P1 toma valores diferentes en ambos casos. Además, los audios correspondientes a las dos bases fueron grabadas con tambores diferentes y con distintas afinaciones. Esto puede explicar que ocurra una confusión de un 71,64 % al clasificar golpes de palo con todo el conjunto $S_{multimodal}$ y una de 29,85 % cuando se descartan las dos características en cuestión.

Por lo tanto, se decidió descartar las características DCT_P1 y $mfccs_1$ de la selección multimodal de forma de tener un conjunto de características menos dependiente de las condiciones de grabación.

En la Tabla 7.11 se pueden observar las pruebas con el nuevo conjunto de características, utilizando tres intérpretes para entrenar y el restante para la validación. Allí puede verse que el desempeño es similar al obtenido con el conjunto que sí consideraba estas características, por lo que parece razonable no considerar estas características en la solución final.

Entrenamiento	*Validación*	*Desempeño*	*Matriz de confusión*						
			a	b	c	d	e	f	$\leftarrow$ clasificación
			68	0	0	0	0	0	$\|a =$ madera
			0	361	2	2	2	1	$\|b =$ mano
Intérpretes 1, 2 y 3	Intérprete 4	89,82 %	0	9	262	31	0	0	$\|c =$ palo
			0	0	14	40	0	0	$\|d =$ rebotado
			0	2	5	0	48	0	$\|e =$ borde
			0	4	1	13	3	7	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			133	0	0	0	0	0	$\|a =$ madera
			0	457	1	0	0	0	$\|b =$ mano
Intérpretes 1, 3 y 4	Intérprete 2	91,12 %	0	9	236	3	1	0	$\|c =$ palo
			0	0	27	133	1	7	$\|d =$ rebotado
			3	0	3	0	276	0	$\|e =$ borde
			0	4	1	0	0	8	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			172	0	0	0	12	0	$\|a =$ madera
			0	378	1	0	0	0	$\|b =$ mano
Intérpretes 1, 2 y 4	Intérprete 3	95,39 %	0	3	216	9	14	1	$\|c =$ palo
			0	1	35	36	0	0	$\|d =$ rebotado
			1	2	10	1	137	1	$\|e =$ borde
			0	0	0	0	1	5	$\|f =$ flam
			a	b	c	d	e	f	$\leftarrow$ clasificación
			182	0	0	0	0	0	$\|a =$ madera
			1	274	0	2	0	0	$\|b =$ mano
Intérpretes 2, 3 y 4	Inérprete 1	92,06 %	0	26	298	5	4	7	$\|c =$ palo
			0	1	3	9	0	0	$\|d =$ rebotado
			0	1	6	3	58	2	$\|e =$ borde
			0	7	1	5	0	25	$\|f =$ flam

Tabla 7.11: Resultados de la clasificación con la base eMe, con 6 tipos de golpes, sacando $\mathrm{DCT_P}1$ y mfcc_1, y utilizando SVM.

Esta página ha sido intencionalmente dejada en blanco.

Capítulo 8

Discusión, conclusiones y trabajo futuro

En este proyecto se estudiaron las aplicaciones del análisis multimodal a la interpretación de Candombe. En particular, se analizó el impacto que tiene la combinación de distintos modos de información en un problema de clasificación de golpes en el toque de repique.

Este tipo de análisis ha ganado popularidad en los últimos tiempos, aunque no se registraban hasta la fecha trabajos en el ámbito local aplicados al análisis musical usando técnicas multimodales. Por lo tanto, este trabajo pretendió ser una primera aproximación a este tipo de técnicas, de manera de generar experiencia y sentar las bases para investigaciones futuras en el área.

El proyecto también pretendió contribuir al estudio de músicas tradicionales, en particular el Candombe afro-uruguayo, por medio de la generación de datos y el desarrollo de herramientas de software que fomenten el uso de la tecnología en estudios musicológicos. En ese sentido, se llevó a cabo un registro de interpretación de candombe en la sala Zavala Muníz del Teatro Solís, del cual participaron cinco de los intérpretes más destacados del género. En la actualidad son escasos los registros de buena calidad de audio y video del toque de Candombe. Por lo tanto, la generación de estos datos significa una contribución directa del proyecto a investigaciones futuras, tanto en el área del procesamiento multimodal como en el área de la musicología. De aquí en más se cuenta con una base de datos etiquetada del toque no sólo de repique sino también de chico y piano, además de distintas configuraciones de cuerdas, con registros de audio y video de buena calidad.

Además, dado que se grabó con un par de cámaras de video en configuración estéreo, el registro puede ser de utilidad para realizar una reconstrucción 3D, lo que puede ser aplicado a análisis de gestualidad. En ese sentido, en el proyecto se desarrolló un algoritmo de seguimiento de ciertos puntos de la escena de manera de tener puntos correspondientes en ambas tomas, que, si bien no fue utilizado para la clasificación de golpes, es otro subproducto destacable del registro y del proyecto en general.

En un principio se evaluó la utilización de tres modos distintos: audio, video y sensores. Se decidió centrar el análisis en dos de ellos (audio y video), ya que no se logró hallar un sensor que se ajustara a los requerimientos del problema. Sin embargo, la tecnología y soluciones disponibles en sensores avanzan rápidamente,

por lo que es posible que en un futuro cercano cumplan los requerimientos.

De manera de cuantificar la ganancia de información obtenida al combinar los distintos modos, se comparó el desempeño que se tiene al considerar cada modo como fuente de información única frente al obtenido cuando se combina esta información. Los resultados de las distintas pruebas son claros en este aspecto: al combinar la información de los distintos modos se obtiene siempre un mejor desempeño que al considerar cada modo por separado.

Una primera aproximación a la solución del problema consistió en considerar una clasificación básica distinguiendo entre tres tipos de golpes: madera, mano y palo. Los resultados obtenidos para este esquema de clases revelan que la clasificación puede llevarse a cabo considerando únicamente las características extraídas del video. Cabe aclarar que esto sigue siendo un enfoque multimodal, ya que las características de video se calculan usando la información de la ubicación de los golpes extraída del audio.

Al aumentar la cantidad de golpes a clasificar (agregando los golpes de borde, flam y rebotado) pudo constatarse que la información de un único modo ya no resulta suficiente para obtener un alto porcentaje de acierto en la clasificación. Fue con este esquema de clases que resultó más evidente la ventaja que representa contar con información derivada de más de un modo: para todos los tipos de golpes, el enfoque multimodal es el que obtiene un mejor desempeño en la clasificación.

Dado que el desarrollo del sistema multimodal se hizo utilizando solamente los registros de la base eMe, otra conclusión interesante se extrae al estudiar el desempeño del sistema cuando se lo utiliza para clasificar un video de la base Zavala. Las características de la grabación de esta última diferían significativamente de las de la base eMe, por lo que la prueba permitió medir la capacidad de generalización del algoritmo. En la primera instancia de clasificación sobre este registro se obtuvo un bajo porcentaje de acierto. Al analizar las características utilizadas para la clasificación, se observó que dos de ellas no resultaban adecuadas si se quería que el algoritmo fuese lo más general posible. Éstas eran el primer coeficiente de la DCT del palo (que representa la posición media de la punta del mismo y por lo tanto depende de la posición relativa de la cámara respecto al intérprete) y el primer MFCC de audio (relacionado con la potencia total de la señal, lo que implica que depende de las condiciones de la grabación). Al removerlas del conjunto de características y volver a realizar la clasificación se verificó que efectivamente conspiraban contra la capacidad de generalización del algoritmo, ya que se obtuvo un porcentaje de acierto $15,8\,\%$ superior.

De este punto se desprende un hecho a tener en cuenta para trabajos futuros y es que, de contar con más tomas etiquetadas de la base Zavala, se podría entrenar el clasificador usando una combinación de tomas de las distintas bases. De esta forma se espera que el sistema se ajuste menos a un tipo de toma particular y que por lo tanto se obtenga un mejor desempeño frente a registros más disímiles.

Contar con más etiquetas de la base Zavala permitiría a su vez incorporar características derivadas de la reconstrucción 3D de la escena ya que esta base es la única que se grabó con cámaras de video en configuración estéreo. Como se tiene un único video etiquetado de menos de un minuto de duración, no se cuenta con

una cantidad suficiente de golpes para entrenar un clasificador, pero esto podría cambiar si se etiquetan todos los videos de la base Zavala.

El uso de otros enfoques para la combinación de la información proveniente de cada modo también es un aspecto a tener en cuenta en un futuro. Dado que aquí se usó un enfoque de integración temprana, podría pensarse en explorar aquellos de integración tardía. Contrastando los resultados obtenidos con las distintas técnicas de combinación puede surgir una discusión interesante respecto a cuál resulta más útil para los distintos problemas de clasificación multimodal.

La evaluación rigurosa de cada etapa de la detección de objetos en el video es otra tarea que quedó pendiente. La principal dificultad consistió en generar las etiquetas de video contra las cuales evaluar las detecciones, ya que es un proceso manual que insume mucho tiempo. Contar con etiquetas de detección permitiría mejorar la segmentación y eso tendría un impacto positivo en las etapas posteriores del sistema. A su vez, realizar una validación de las segmentaciones permitiría concluir de forma más precisa si los errores de clasificación son producto de un error en las detecciones o si es un problema del poder de discriminación de las características.

También queda pendiente modificar la detección de eventos en el audio para que sea totalmente automática. Para ello, una posibilidad es agregar a las etiquetas de los registros una clase que represente los falsos positivos, y asignarla a todos aquellos máximos del spectral flux que son detectados como un golpe pero no lo son. A su vez, podría ser de utilidad agregar algún tipo de restricción temporal de manera de que en la detección de eventos no se consideren aquellos máximos que se encuentren muy cercanos a un máximo anteriormente detectado. Esta restricción podria ayudar a la detección del evento correspondiente a un golpe de flam, ya que en este tipo de golpe se presentan dos máximos de similar amplitud, pero se debe considerar únicamente el primero de ellos.

Finalmente, de desarrollarse un sensor que cumpliese con los requisitos planteados, podría ser incorporado como un modo de información más. En el futuro deberían realizarse relevamientos periódicos del estado del arte en esta materia, siguiendo de cerca el desarrollo de sensores que se adecúen a las necesidades particulares del problema.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice A

Sensores

A.1. Introducción

Dada una aplicación que involucra transcripción de movimiento, es razonable cuestionarse si la utilización de uno o más sensores en la adquisición de datos agrega información relevante al problema (además de la que aportan las cámaras y los micrófonos). En este sentido, se realizó una búsqueda para determinar qué tipo de sensores podrían ser convenientes para este problema en particular y se estudió la viabilidad de su aplicación en el proyecto.

En una primera etapa se realizó una revisión de los sensores disponibles en el mercado y sus principios de funcionamiento, para luego tomar la decisión de si incluirlos o no en el proyecto. Dicha búsqueda se realizó bajo el supuesto de que un sensor útil para este proyecto sería uno que en sí mismo planteara una solución terminada, es decir, que no precisara desarrollar hardware o software adicional, ya que esto excedería los objetivos del proyecto. A su vez, se planteó como otro requisito que el sensor seleccionado no debía ser invasivo para el intérprete y además debía ser práctico de utilizar. Esto se debió a que dichos sensores serían utilizados en el contexto de un registro en el cual no se dispondría de mucho tiempo y donde los intérpretes involucrados no estarían familiarizados con este tipo de procedimientos.

A continuación se presenta un rápido análisis de los sensores distinguiéndolos en dos clases: los sensores básicos y los sensores de captura de movimiento.

A.2. Sensores Básicos

En primera instancia se realizó una revisión de los sensores que son utilizados generalmente en aplicaciones de detección de movimiento, tales como acelerómetros, sensores infrarrojos, etc. Dichos sensores se caracterizan por ser soluciones simples, que precisan como máximo un circuito o una interfaz sensor-computadora sencilla para su utilización. En esta sección se presentan breves comentarios sobre los mismos, distinguiéndolos según su principio de funcionamiento.

A.2.1. Sensores Piezoeléctricos

Un sensor piezoeléctrico es un dispositivo capaz de traducir tensiones mecánicas en eléctricas. Basa su funcionamiento en la propiedad que presentan determinados materiales de polarizarse eléctricamente cuando son deformados por la acción de una fuerza. Dicha polarización genera un campo eléctrico que provoca una diferencia de potencial en la superficie del material. Más especificamente, la piezoelectricidad resulta del efecto combinado del comportamiento eléctrico:

$$D = \epsilon E \tag{A.1}$$

donde D es el desplazamiento eléctrico o densidad de flujo eléctrico, ϵ es la permitividad eléctrica del material y E es el campo eléctrico; y la ley de Hooke

$$S = sT \tag{A.2}$$

donde S es la deformación, s es el inverso del módulo de Young (parámetro que caracteriza la elasticidad del material) y T es la tensión. Estas igualdades se pueden combinar para formar dos ecuaciones acopladas, quedando la relación entre carga y deformación dada por:

$$S = [s^E]T + [d]^t E \tag{A.3}$$

$$D = [d]T + [\epsilon^T]E \tag{A.4}$$

donde $[d]$ es la matriz que representa el efecto piezoeléctrico directo y $[d]^t$ el inverso (matriz traspuesta). El superíndice E señala que la magnitud está medida bajo campo eléctrico constante o nulo. Asimismo, el superíndice T indica lo propio pero referido a un campo de fuerza.

Dicho funcionamiento presenta la ventaja de ser simple (en el sentido de que el modelo es simple), además de que son baratos y fáciles de conseguir. En la mayoría de las aplicaciones que utilizan este tipo de sensores [35, 72], se coloca el piezoeléctrico en la zona donde se desea detectar si hay contacto y mediante los cambios de tensión en el mismo se determina cuando se produce un golpe.

En el caso particular de la detección de golpes en la lonja de un tambor, se podría pensar en colocar un sensor piezoeléctrico sobre una zona de la misma (por ejemplo la zona donde "debería" golpear el palo) y así determinar si el intérprete realizó un golpe de palo o no. Esto presenta varias desventajas. Para empezar, el hecho de colocar un sensor sobre la lonja modifica el sonido de la misma, ya que limita su vibración. Además, si el tamaño de dicho sensor fuera muy pequeño y se aceptara como despreciable este efecto, el sensor sería de todas formas inadecuado, debido a que el supuesto de que el intérprete golpea en una zona del tambor con el palo y en otra con la mano no se cumple estrictamente y menos en el caso del repique. Se podría solucionar colocando por lo menos cuatro sensores para determinar con mayor precisión la ubicación del golpe, pero sería una configuración mucho más invasiva, y a su vez no aseguraría que el golpe esté ubicado con precisión. Podría presentarse además el problema del *crosstalk*, el cual refiere a la activación del sensor debido vibraciones secundarias producto de impactos en otros sensores. Eso sin tener en cuenta que la utilización de un piezoeléctrico implica la necesidad de cierto cableado, que limita e incomoda el movimiento del intérprete.

A.2.2. Sensores Resistivos de Fuerza

El sensor de fuerza resistivo (FSR) es un dispositivo de película de polímero (PTF) que presenta una disminución de la resistencia cuando aumenta la fuerza aplicada a la superficie activa. Su sensibilidad está optimizada para el uso por toque humano de dispositivos electrónicos. Con respecto a este proyecto, un sensor de este tipo se encuentra en la misma línea que los piezoeléctricos, ya que al ser de contacto es necesario colocarlos sobre la lonja para detectar deformaciones. Esto ya presenta una desventaja, pero además, como dichos sensores se activan solamente cuando se produce el contacto sobre ellos, se necesitan sensores muy grandes o superficies chicas para que la detección tenga éxito. Aunque son robustos frente a los problemas de crosstalk, por las desventajas presentadas anteriormente no son una buena opción para la aplicación en cuestión.

A.2.3. Sensores de Fibra Optica

Un sensor de fibra óptica es un sensor que utiliza fibra óptica ya sea en el elemento a sensar o como medio para transportar las señales desde un sensor remoto hasta un procesador o elemento electrónico que procese la señal. Dichos sensores presentan varias ventajas en cuanto a su desempeño en entornos industriales y debido a su inmunidad a la interferencia electromagnética. Otra ventaja es que pueden utilizarse en configuraciones no invasivas (sensando a distancia). Sin embargo, requieren equipamientos específicos que no son fáciles de obtener en el mercado y son muy caros. A su vez, utilizar este tipo de sensores implicaría desarrollar un software que interprete las señales detectadas por ellos y un hardware que los adapte a esta aplicación en particular. A su vez, no hay garantía de que esto implique un aporte de información significativo, por lo que no son una opción a considerar.

A.2.4. Sensores Capacitivos

Los sensores capacitivos basan su funcionamiento en detectar los cambios de capacidad que se generan entre una cierta superficie de interés y la superficie activa del sensor. Una configuración posible para esta aplicación utilizando este tipo de sensor sería colocar su superficie activa fija sobre la lonja y tomar como superficies de interés la mano y el palo, con lo cual se detectaría la proximidad de dichos objetos al superar cierto umbral de capacidad. Dado que la capacidad depende de la constante capacitiva del material, se tendrían valores diferentes frente a un golpe con la mano y con el palo, permitiendo distinguirlos. El principal problema que plantea este tipo de sensor es que tienen poca precisión, con lo que si el elemento a seguir es pequeño, se pueden tener inconvenientes. A su vez, los ejemplos de desarrollo de aplicaciones con este tipo de sensores son poco exigentes en cuanto a la tasa a la que procesan los datos y a la precisión con que se detectan los objetos (como puede verse en [77]), lo cual resulta un problema para esta aplicación.

A.2.5. Acelerómetros

Se denomina acelerómetro a cualquier instrumento destinado a medir aceleraciones. Un acelerómetro también es usado para determinar la posición de un cuerpo, pues al conocerse su aceleración en todo momento, es posible calcular los desplazamientos que tuvo. Considerando que se conocen la posición y velocidad original del cuerpo bajo análisis, sumando los desplazamientos medidos se determina la posición. Varias aplicaciones que intentan determinar movimiento utilizan acelerómetros como sensores. Además, son dispositivos ampliamente utilizados en el mundo audiovisual [82, 84].

Existen una serie de factores que son característicos de todos los sensores y fueron particularmente importantes a la hora de elegir cuáles acelerómetros serían adecuados para esta aplicación, ya que existe una gran variedad de este tipo de sensor. Dichos factores son:

1. Rango dinámico: El rango dinámico es el valor máximo de la señal de entrada que el acelerómetro puede medir antes de distorsionar o saturar la señal de salida.

2. Sensibilidad: La sensibilidad es el factor de escala de un sensor o sistema, medida en términos de cambio en la señal de salida por cambio de la entrada medida. Refiere a la habilidad del sensor de capturar movimientos pequeños.

3. Respuesta de frecuencia: Es el rango de frecuencia en la que el sensor detecta el movimiento y muestra una salida sin distorsionar.

4. Eje sensible: Los acelerómetros están diseñados para detectar entradas en referencia a un eje; acelerómetros de eje único sólo pueden detectar entradas a lo largo de un plano. Acelerómetros de tri-eje pueden detectar entradas en cualquier plano y son necesarios para la mayoría de las aplicaciones.

5. Tamaño y masa: El tamaño y la masa de un acelerómetro puede cambiar las características del objeto que está siendo probado. La masa de los acelerómetros debe ser insignificante respecto de la masa del sistema a supervisar.

Teniendo en cuenta que la sensibilidad, el tamaño y la masa del sensor serían parámetros críticos para esta aplicación, se realizó una breve revisión de los tipos de acelerómetros más utilizados para determinar si alguno se adecuaba a estos requerimientos.

Acelerómetro piezoeléctrico

Este acelerómetro se basa en que, cuando se comprime un retículo cristalino piezoeléctrico, se produce una carga eléctrica proporcional a la fuerza aplicada. Los elementos piezoeléctricos se encuentran comprimidos por una masa, sujeta al otro lado por un muelle y todo el conjunto dentro de una caja metálica. Cuando el conjunto es sometido a vibración, el disco piezoeléctrico se ve sometido a una fuerza variable, proporcional a la aceleración de la masa. Debido al efecto piezoeléctrico

se desarrolla un potencial variable que será proporcional a la aceleración. Dicho potencial variable se puede registrar sobre un osciloscopio o voltímetro.

Este dispositivo junto con los circuitos eléctricos asociados se puede usar para la medida de velocidad y desplazamiento además de la determinación de formas de onda y frecuencia. Una de las ventajas principales de este tipo de transductor es que se puede hacer tan pequeño que su influencia sea despreciable sobre el dispositivo vibrador. El intervalo de frecuencia típica es de 2 Hz a 10 KHz. Los acelerómetros electrónicos basados en el efecto piezoeléctrico permiten medir la aceleración en una, dos o tres dimensiones. Esta característica permite medir la inclinación de un cuerpo, ya que es posible determinar con el acelerómetro la componente de la aceleración provocada por la gravedad que actúa sobre el cuerpo. Actualmente este tipo de acelerómetros se pueden construir en un sólo chip de silicio, incluyendo en el mismo la parte electrónica que se encarga de procesar las señales.

Acelerómetro de condensador

Estos acelerómetros miden el cambio de capacidad eléctrica de un condensador generado a partir del movimiento de una masa sísmica situada entre las placas del mismo. El capacitor se constituye por tres placas dispuestas en planos paralelos y alineadas por sus ejes. Las dos externas están fijas y la del medio está mecánicamente acoplada al dispositivo cuyo desplazamiento se desea medir. El sistema forma entonces un circuito de dos condensadores variables conectados en serie con valores que vienen dados por:

$$C_1 = \frac{\epsilon\epsilon_0 A}{d+x} \qquad\qquad C_2 = \frac{\epsilon\epsilon_0 A}{d-x} \tag{A.5}$$

Cuando la distancia entre la placa móvil y una de las fijas se incrementa en una cantidad x, la distancia de la placa móvil a la otra placa se reduce en la misma cantidad. Por lo que, midiendo los respectivos voltajes de los capacitores a una corriente conocida es posible determinar el valor de la capacitancia, lo cual da una idea de la aceleración del objeto.

Este sensor es una buena alternativa para la medición de pequeños desplazamientos con gran precisión y son utilizados para corregir la orientación de la pantalla de los smartphones cuando se los gira en sentido horizontal o vertical.

Como ya se dijo antes, una configuración con cables como en [84] no se puede implementar en esta aplicación porque es demasiado invasivo para los intérpretes. Lo adecuado sería una solución inalámbrica. Un ejemplo de un sensor capacitivo inalámbrico es el sensor LilyPad XBee [1]. Éste consiste en un acelerómetro que mide la aceleración en tres ejes (basado en un ADXL335[2]) implementado en un parche para ropa de 50mm de diámetro en el cual también se incluye un microprocesador (ATmega328 [3]) y un módulo XBee que permite conectividad inalámbrica con otro

[1] `https://www.sparkfun.com/products/12921`

[2] `https://www.sparkfun.com/datasheets/DevTools/LilyPad/ADXL335.pdf`

[3] `http://www.atmel.com/devices/atmega328.aspx`

módulo incluido en un Arduino [4]. Este tipo de solución podría funcionar para poder determinar el movimiento de las articulaciones del intérprete y obtener información de la gestualidad del mismo. Sin embargo, existe la posibilidad de extraer dicha información mediante procesamiento de imágenes utilizando marcadores en lugar de dichos sensores, lo cual simplifica la configuración ya que no es necesario realizar la implementación del sistema Arduino-LilyPad. Un sensor con estas posibilidades sería de gran utilidad si brindara información sobre la gestualidad de la mano (por ejemplo, cada dedo por separado), lo cual no es fácil de conseguir mediante el procesamiento de imágenes. Sin embargo, no hay implementaciones de menor tamaño que permitan utilizar este sensor de esta manera.

Acelerometro de efecto Hall

Utilizan una masa sísmica donde se coloca un imán y un sensor de efecto Hall que detecta cambios en el campo magnético. Dichos cambios tendrán relación con la aceleración de la masa sísmica, ya que estarán determinados por la distancia del imán al sensor Hall en cada instante. Si fluye corriente por el sensor y se lo expone a un campo magnético que fluye en dirección perpendicular a la corriente, entonces se generará por efecto Hall un voltaje en el mismo, que será proporcional al producto del campo magnético y de la corriente. Si se conoce el valor de la corriente, entonces se puede vincular los cambios de voltaje en el sensor con aceleración del cuerpo en estudio. En equilibrio se tiene:

$$F_e = F_m \rightarrow q.E = q.v.B \rightarrow E = v.B \rightarrow \frac{V_H}{d} = v.B \rightarrow V_H = v.B.d \qquad (A.6)$$

Las ecuaciones anteriores reflejan la dependencia del voltaje respecto al campo magnético, que a su vez varía en función de la aceleración. Estos sensores son más difíciles de conseguir que los mencionados anteriormente y aunque parecen ser poco invasivos, presentan el mismo problema que los sensores capacitivos en el sentido de que implican implementación de un sistema para poder utilizarlos. En contraparte no aportan información que no se pueda conseguir por otro medio.

A.3. Sistemas de Captura de Movimiento

Los sistemas de captura de movimiento son sistemas que involucran uno o varios tipos de sensores integrados mediante un software que interactúa con el usuario. Dichos sistemas permiten realizar aplicaciones de detección de movimiento de alto nivel, sin tener que ocuparse de implementaciones de hardware. Es por ello que este tipo de dispositivos fueron de particular interés en la revisión de sensores realizada, en particular aquellos que realizaban seguimiento de manos (información de gran utilidad para la transcripción). Se presentan a continuación comentarios

[4] Un ejemplo puede verse en `http://blog.arduino.cc/2014/06/24/wireless-controlled-robotic-hand-made-with-arduino-lilypad/comment-page-1/`

sobre dos sensores de captura de movimiento con los cuales se realizaron varias pruebas de concepto.

A.3.1. Leap Motion Controller

Introducción

El Leap Motion Controller [5] es un sensor que sigue el movimiento de manos, dedos y herramientas, por ejemplo lápices u objetos similares, con gran precisión y velocidad. Fue pensado como un periférico para la computadora, complementario al teclado y al mouse, que permite manejar varias aplicaciones simplemente moviendo las manos en el aire. Se conecta a la computadora a través de un puerto USB y dispone de un software multiplataforma simple que permite utilizarlo sin tener que realizar ninguna adaptación del mismo. Dicho sensor se constituye de tres sensores infrarrojos que implementan un campo de visión del sensor en forma de pirámide invertida de 150° de amplitud.

Accesibilidad al dispositivo

Los datos accesibles del Leap están estructurados en clases, pudiéndose acceder a los objetos por separado pero no a datos de más bajo nivel adquiridos por los sensores infrarrojos. La estructura de las clases más importantes está detallada a continuación:

`Listener`

Define funciones para manejar los eventos (paquetes de información de los sensores) que envía el Leap.

Funciones importantes:

- void onConnect(controller arg0) - es llamado cuando se conecta el Leap, imprime “Connected”

`Finger y Tool`

Son las clases que contienen la información del tracking de los dedos y herramientas que detecta el sensor. La diferenciación entre un objeto dedo y un objeto tool (herramienta) se realiza de acuerdo a la forma. Por ejemplo, los dedos son menos rígidos y más gruesos que un lápiz común. Son objetos del tipo Pointable (la clase Pointable es una clase la cual administra las características físicas de dedos y tools) y son descritos mediante objetos de la clase vector, que tienen un módulo, una dirección y sentido. Al igual que las manos y los frames, a cada dedo y tool le corresponde un ID (que se mantiene mientras está visible).

[5] `http://www.leapmotion.com`

`Hands`

Contiene toda la informacion sobre la posición, características, movimiento, lista de dedos y herramientas asociadas a cada mano que identifica el sensor. Esta clase impone algunas restricciones al sensor, por ejemplo, se recomienda que hayan hasta dos manos como máximo (tracking óptimo). Cada mano tiene un ID (se mantiene mientras la mano este visible) y una lista de dedos y tools.

Algunos atributos importantes de esta clase (importantes para el tracking de la misma) son:

- Vector normal a la mano
- Vector en dirección de los dedos
- Propiedades de la esfera que queda determinada por la curvatura de la mano

`Frame`

Es la clase que contiene la información extraída por el sensor en cada instante de muestreo. Un objeto frame tiene: un id o número identificador, un timestamp, una lista con todas las manos existentes en ese frame con sus respectivos dedos y herramientas.

Funciones importantes:

- Finger finger(int id), Hand hand(int id), Tool tool(int id), Pointable pointables(int id) - Devuelven el objeto correspondiente a ese id.

`Controller`

Es la interfaz que comunica al sensor con el software. Permite manejar la información extraída de cada frame de dos maneras principalmente:

- Realizando Pulling - se crea un objeto controller y se llama a la función controller.frame() que devuelve el último frame (se puede hacer que devuelva hasta los últimos 60 frames). Esto se puede realizar constantemente para no perder eventos)
- Creando una instancia de Listener - cuando hay un frame disponible el Controller llama a listener.OnFrame() y ejecuta las sentencias que estén escritas por el usuario en dicho método

Funciones importantes:

- Bool addListener(Listener listener) - Agrega un objeto de la clase Listener
- Devicelist devices() - Devuelve una lista de devices del Leap (field o view, id,calibrated positions)

`Gestures`

Reconoce ciertos patrones de movimiento como gestos específicos (por ejemplo un círculo). Los gestos se determinan para cada dedo o tool individualmente.

Sobre la aplicación de este sensor en el proyecto

Se realizaron varias pruebas de concepto con el Leap Motion para evaluar su aplicabilidad en el proyecto.

En primera instancia, se realizó una configuración que simulaba el uso del sensor captando una improvisación de repique. Para ello se colocó el Leap en un soporte que permitió probarlo en varias posiciones relativas a la lonja y que lo mantuvo seguro de posibles vibraciones debidas al toque del tambor. Se utilizó una computadora con gran poder de procesamiento (para descartar el mal funcionamiento de la misma como un factor atenuante) y mediante un mousepad sobre una mesa (que ofició de tambor) y un palo de repique típico se simularon los movimientos de un intérprete de candombe. Esta prueba se realizó con el sensor en diferentes posiciones:

1. Perpendicular a la lonja
2. Con una pequeña inclinación con respecto a la normal a la superficie
3. Paralela a la superficie de prueba mirando hacia la misma

Las conclusiones de esta primera prueba fueron que en todos los casos el sensor no es apropiado para este proyecto. Debido a que los movimientos son muy rápidos, el Leap detecta manos que no están, y el tracking se pierde constantemente. Se comprobó además que el campo de visibilidad es muy reducido para esta aplicación.

En una segunda instancia, se probó cambiar el modo de operación del Leap Motion a High-Speed para obtener una mayor tasa de fps y comprobar si en ese caso el sensor podía trackear mejor las manos. Según el fabricante, en este modo se debería obtener una tasa de 214 fps usando conexión USB 2.0, y 295 fps usando conexión USB 3.0, con la desventaja de obtener una menor precisión. Sin embargo, el sensor se ajusta por defecto y de forma automática al modo "Robusto" y sólo se habilita un cambio de modo cuando se encuentra en una zona oscura, obteniéndose resultados levemente mejores que en las condiciones antes descritas. A su vez, si el Leap se encuentra en el modo Rápido pero se lo lleva a una zona con demasiada luz, éste vuelve automáticamente a modo Robusto. Esto es una gran desventaja para la aplicación ya que la inclusión de este sensor en la configuración comprometería la iluminación necesaria para una buena filmación.

A.3.2. Kinect

Lanzado por Microsoft en 2010, el Kinect es una interfaz de juego primeramente pensada para ser utilizada con la plataforma de juego XBox 360. Esta plataforma permite al usuario controlar e interactuar con la consola sin la necesidad de tocar

un controlador. Para llevar a cabo esta tarea, el sistema se basa en una interfaz que reconoce gestos, comandos de voz y objetos e imágenes. Debido a esta característica es que múltiples desarrolladores han puesto la mira en la adaptación de este dispositivo para otras aplicaciones más allá de la industria del videojuego.

Ejemplos como [91] muestran adaptaciones del Kinect para reconocimiento de gestos y seguimiento en general. Las referencias [76] [71] y [45] muestran que es posible obtener un sistema robusto de reconocimiento de dedos y gestualidad manual utilizando esta interfaz.

En cuanto a la utilización de Kinect para resolver este problema en particular, existen ventajas y desventajas. Como principal ventaja puede mencionarse que parece haber gran cantidad de trabajo preexistente en el campo de seguimiento y análisis de gestualidad manual, con aparentemente buenos resultados. Esto, sumado a referencias como [76] en las cuales el sistema implementado está muy bien descripto, hacen del Kinect una buena opción. Otra ventaja adicional es que el Kinect no representa un método invasivo a la hora de implementar un software de análisis de gestualidad manual, lo que en principio permitiría generar un sistema que sea a la vez portable y permita al ejecutante del instrumento la libertad usual a la hora de tocar.

Las grandes desventajas que tiene este sistema son principalmente dos. La primera, es que al ser una herramienta diseñada por Microsoft, está regida por su habitual práctica de software privativo. Sin embargo, existen alternativas diseñadas por desarrolladores independientes que permiten utilizar el Kinect separado de la consola de juego y bajo los sistemas operativos más comunes. Ejemplos son las plataformas CLUNI [5] y OpenNI [18] que permiten utilizar Kinect bajo Windows.

La segunda desventaja viene dada por las restricciones técnicas del Kinect. Éste está compuesto por una cámara, un sensor de profundidad y un sistema de micrófonos. El fabricante declara que la cámara puede dar como salida un video a 30 cuadros por segundo, con una resolución VGA de 8-bits (640 x 480 pixels). Este frame rate es insuficiente para lograr una captura de gestualidad manual satisfactoria, debido a que es de esperar que en una improvisación de repique se llegue a tener golpes de gran velocidad y de ancho de banda no acotado, como por ejemplo golpes rebotados de palo.

A.4. Evaluación de la utilización de sensores en el proyecto

Luego de una amplia revisión de los tipos de sensores y sistemas de captura de movimiento disponibles en el mercado, se concluyó que no es viable la inclusión de los mismos en este proyecto. Esta decisión se basa en que como se vio anteriormente, al momento del desarrollo de esta aplicación, ninguno de los sensores estudiados se adaptan a los requerimientos.

En el caso de los sensores básicos (acelerómetros, infrarrojos, etc), son soluciones muy invasivas y en todos los casos hay que implementar hardware, sin que ello garantice un aporte significativo de información. Por otro lado, los sistemas

de captura de movimiento disponibles al momento del desarrollo de este proyecto no satisfacen los requerimientos del mismo, ya sea porque presentan limitaciones técnicas o porque fueron diseñados para funcionar de forma óptima bajo condiciones controladas, y no es posible adaptarlas a los requerimientos de este proyecto.

Finalmente se decidió que se trabajará con dos modos de información (audio y video) para realizar la transcripción.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice B

Filtro de Color

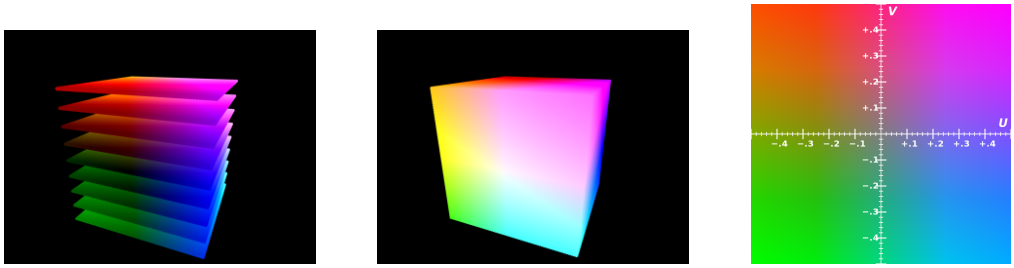
La segmentación de una imagen consiste en su descomposición en grupos reducidos de píxeles con un cierto significado semántico. El objetivo de esto puede ser la simplificación de la imagen, su reducción de tamaño, la transformación de la misma para facilitar su posterior procesado, entre otros [9]. En este sentido, la segmentación de color es un procesamiento relativamente simple que permite destacar objetos de un determinado color del resto de la escena. Para la simplificación de un problema de detección de objetos, suele pintarse el objeto de interés de un color facilmente diferenciable, de manera de usar esta información para facilitar el procesamiento. En este caso, se utilizó dicho enfoque para la detección del palo, lonja y mano izquierda. Para ello, se implementó un filtro de color en el espacio YUV basado en el filtro que utiliza el software VLC [25], en particular, se adaptó a C++ el código de la versión 2.1.5.

B.1. Espacio YUV

YUV es un espacio de color que codifica los colores según valores de luminancia (Y) y croma (U, V). Varios formatos de compresión con pérdida utilizan la codificación en el espacio YUV asignando un mayor ancho de banda para el canal Y que para las componentes de U y V, debido a que el ojo humano es más sensible a las diferencias de brillo que a las diferencias de color. De esta forma se logra que los errores de transmisión o las imperfecciones de compresión sean menos notorias que utilizando una representación RGB directa. Esto es particularmente útil para la compresión de video y televisión. En la Figura B.1 se ven distintas representaciones de este espacio de color para distintos valores de Y, U y V.

B.2. Segmentación en el plano UV

La segmentación de color en el espacio UV presenta la ventaja de que permite distinguir entre dos colores independientemente de su nivel de luminancia. Dado un cierto color con valores de crominancia u_1 y v_1, el mismo se representa como un



(a) Cubo YUV en capas para distintos valores de Y

(b) Cubo del espacio YUV

(c) Plano UV para Y=0.5

Figura B.1: Representaciones del espacio YUV - Imágenes extraídas de [22], [26]

punto en el plano UV. La representación de dicho plano puede verse en la Figura B.1(c).

Filtrar un cierto color implica determinar todos aquellos colores que se parecen a un color de referencia dado un cierto criterio. El algoritmo utilizado se explica a continuación.

Se considera un color de referencia (C_{ref}) caracterizado a partir de ahora por los valores de croma u_{ref} y v_{ref}. Se quiere determinar un conjunto de colores que sean parecidos al color de referencia, es decir, determinar los puntos del plano UV que estén cerca de (u_{ref}, v_{ref}) dada una cierta distancia que se determinará. Un punto cualquiera del plano (C) se representa con sus respectivos valores de u y v, como se ve en la Figura B.2.

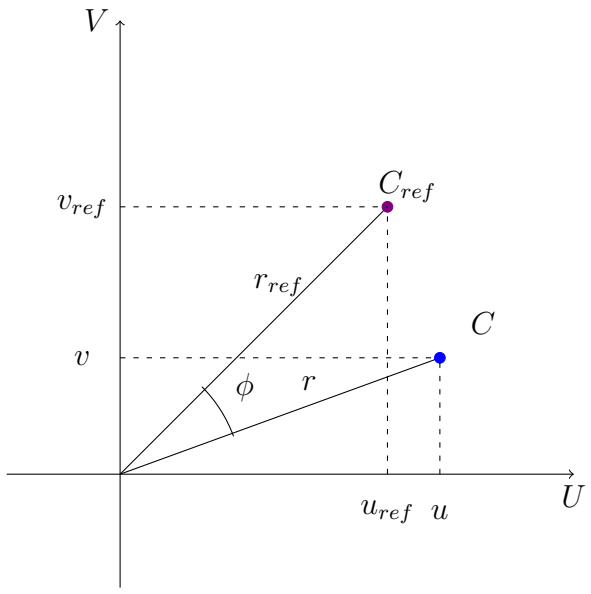


Figura B.2: Representación de los colores en el plano UV.

Se definen ahora las distancias al origen de C_{ref} y C respectivamente:

$$r_{ref} = \sqrt{u_{ref}^2 + v_{ref}^2}$$

$$r = \sqrt{u^2 + v^2}.$$

Observando la Figura B.1(c) puede verse que esta distancia representa la saturación del color, y que a mayor distancia se tiene un valor de saturación mayor. En la teoría de color, la saturación o pureza de un color está determinada por una combinación de su intensidad luminosa y de la distribución de sus diferentes longitudes de onda en el espectro de colores [24]. Un color más saturado o puro será aquel que esté constituído de una sola longitud de onda con alta intensidad.

Se define la distancia dada por:

$$d_{aux} = \sqrt{u_{aux}^2 + v_{aux}^2} \tag{B.1}$$

donde u_{aux} y v_{aux} se definen como:

$$u_{aux} = r_{ref}u - ru_{ref} \tag{B.2}$$

$$v_{aux} = r_{ref}v - rv_{ref}. \tag{B.3}$$

Para ver con mayor claridad el significado de esta distancia, se opera con las coordenadas del punto auxiliar (u_{aux}, v_{aux}). Partiendo de las ecuaciones B.2 y B.3 y elevando al cuadrado de ambos lados se obtiene:

$$u_{aux}^2 = r_{ref}^2u^2 + r^2u_{ref}^2 - 2r_{ref}ru_{ref}u$$

$$v_{aux}^2 = r_{ref}^2v^2 + r^2v_{ref}^2 - 2r_{ref}rv_{ref}v.$$

A partir de ello se calcula d_{aux}^2 de la siguiente forma:

$$d_{aux}^2 = r_{ref}^2u^2 + r^2u_{ref}^2 - 2r_{ref}ru_{ref}u + r_{ref}^2v^2 + r^2v_{ref}^2 - 2r_{ref}rv_{ref}v.$$

Agrupando:

$$d_{aux}^2 = r_{ref}^2(u^2 + v^2) + r^2(u_{ref}^2 + v_{ref}^2) - 2r_{ref}r(u_{ref}u + v_{ref}v). \tag{B.4}$$

De la ecuación B.4 pueden identificarse las siguientes expresiones:

$$r^2 = u^2 + v^2 \tag{B.5}$$

$$r_{ref}^2 = u_{ref}^2 + v_{ref}^2 \tag{B.6}$$

$$< (u, v), (u_{ref}, v_{ref}) >= u_{ref}u + v_{ref}v. \tag{B.7}$$

A su vez, sabiendo que el producto escalar de B.7 puede escribirse como:

$$< (u, v), (u_{ref}, v_{ref}) >= r_{ref}r\cos(\phi) \tag{B.8}$$

con ϕ el ángulo entre r_{ref} y r.

Con lo cual, sustituyendo las expresiones B.5, B.6 y B.8 en B.4 se tiene:

$$\begin{aligned} d_{aux}^2 &= r_{ref}^2 r^2 + r^2 r_{ref}^2 - 2r_{ref}^2 r^2 \cos(\phi) \\ &= 2(r_{ref} r)^2 (1 - \cos(\phi)). \end{aligned}$$

Finalmente, considerando la identidad trigonométrica:

$$\sin(\frac{\phi}{2})^2 = 1 - \cos(\phi)$$

se obtiene la siguiente expresión:

$$\boxed{d_{aux} = \sqrt{2} r_{ref} r \sin(\frac{\phi}{2})} \tag{B.9}$$

Analizando la distancia de la ecuación B.9, se puede ver que vale cero solamente cuando ϕ vale cero o es múltiplo de 2π. Observando la Figura B.2, esto sucede para aquellos puntos que se encuentran sobre la misma recta. Dichas rectas del plano UV representan un tono particular (Figura B.1(c)). Por lo tanto, dos colores cuya distancia d_{aux} valga cero tienen el mismo tono, pudiendo tener diferentes valores de saturación. La distancia máxima está dada para $\phi = \pi$, e implica tonos opuestos en el plano UV. Según el modelo de la CIECAM02 [66], un tono se define técnicamente como el grado de similitud entre el estímulo percibido y el estímulo descrito como rojo, el verde, el azul o el amarillo [8].

Teniendo en cuenta todo lo anterior, se consideran para la segmentación de color en este espacio dos tipos de umbrales, uno para la saturación y otro para la similitud de tono. Dichos umbrales determinan un rango de valores entre los cuales se considera que el color es parecido al que se tiene por referencia.

El umbral de saturación define una distancia al origen mínima y otra máxima en el plano UV, las cuales determinan un anillo de colores posibles, como se muestra en la Figura B.3(a).

Este umbral depende del valor de saturación que tenga el color de referencia y de lo restrictivo que se quiera ser respecto a las diferencias en saturación. El filtro del software VLC versión 2.1.5 determina sólo un valor mínimo de saturación, sin embargo, estudiando la distribución de los colores de interés de los distintos videos se decidió agregrar un umbral superior que determine un valor máximo para la saturación. Esto se hizo con el objetivo de distinguir entre colores con valores muy cercanos en tono pero con diferencias de saturación, por ejemplo los colores de piel y piso, o palo y piel para los videos de la base Zavala (ver Figura B.4).

La formulación matemática para este umbral se muestra a continuación:

$$\boxed{SAT\_TH_{min} < r < SAT\_TH_{max}}$$

con SAT_TH_{min} y SAT_TH_{max} los umbrales que se dan como entrada del algoritmo.

Por otro lado, el umbral de similitud propone restricciones respecto al ángulo ϕ y determina qué tan restrictivo se es respecto a las diferencias de tono. El umbral en este caso está dado por:

$$d_{aux}^2.SIM\_TH < (r_{ref}r)^2 \tag{B.10}$$

Sustituyendo la expresión B.9 en B.10 se tiene:

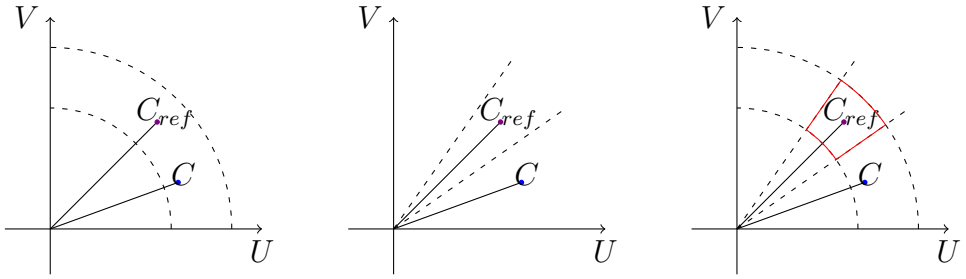
$$2(r_{ref}r)^2 sin(\frac{\phi}{2})^2.SIM\_TH < (r_{ref}r)^2$$

$$2sin(\frac{\phi}{2})^2.SIM\_TH < 1$$

Con lo cual la expresión para el umbral en similitud queda:

$$\boxed{sin(\frac{\phi}{2})^2 < \frac{1}{2.SIM\_TH}} \tag{B.11}$$

Este umbral determina un cono plano en torno al color de referencia, como puede verse en la Figura B.3(b).



(a) Restricciones de saturación en el plano UV

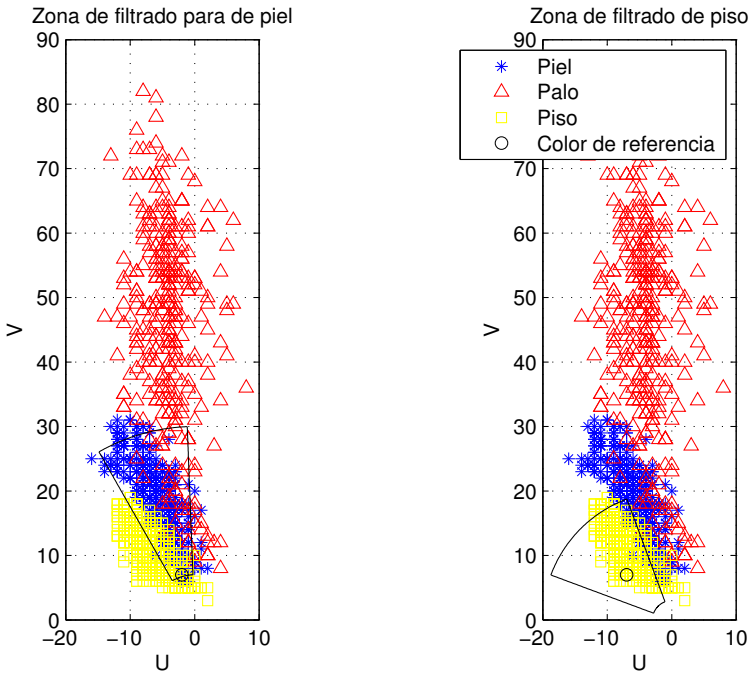
(b) Restricciones de similitud en el plano UV

(c) El interior de la zona delimitada por rojo representa la zona de filtrado.

Figura B.3: Filtro de color implementado en el plano UV.

Finalmente, uniendo ambos umbrales se tiene el filtro de la Figura B.3(c). Un ejemplo de este filtrado puede verse en la Figura B.4, donde se tienen muestras de palo, piso y piel de la base Zavala y el filtrado mediante un color de referencia.

En la Figura B.4 puede verse la diferencia entre un filtro convencional y el filtro implementado.



(a) Ejemplo del uso del filtro de color para segmentar piel

(b) Ejemplo del uso del filtro de color para segmentar piso

Figura B.4: Ejempo del uso del filtro de color en la base Zavala

(a) Filtro de color basado en umbrales en los canales YUV

(b) Filtro de color con umbrales por saturación y similitud

Figura B.5: Comparación de distintos filtros de color en el espacio UV para segmentar piel en la base eMe (imágenes con zoom)

Apéndice C

Filtro de Kalman

El filtro de Kalman [56] es un estimador *óptimo*, esto es, un estimador que infiere parámetros de interés a partir de una serie de medidas ruidosas observadas a lo largo del tiempo. Éste se utilizó en la Sección 4.3 para suavizar el ruido proveniente de la detección de la mano izquierda del intérprete.

Matemáticamente, se asume que la señal que se quiere estimar $X[k]$ (llamada comúnmente variable de estado) depende de las observaciones en tiempos anteriores mediante la siguiente ecuación:

$$X[k+1] = TX[k] + GU[k] + w[k],$$

donde T es la llamada matriz de transición, G la matriz de control, $U[k]$ el vector de control y $w[k]$ ruido blanco gaussiano de media 0 y varianza σ_w (que se asume conocida). La matriz de transición T modela la dinámica del sistema en ausencia de ruido y debe ser definida de antemano.

En este caso, el modelo utilizado fue un modelo simple de mecánica: se consideraron como variables de estado la posición, la velocidad y la aceleración del punto medio del bounding box, en ambas componentes (horizontal y vertical). Dada la alta tasa de registro de video, se asumió que la aceleración se mantenía constante de un frame al siguiente. Así, si $x[k]$ e $y[k]$ son las posiciones horizontal y vertical del punto en el frame k, $v_x[k]$ y $v_y[k]$ las velocidades respectivas y $a_x[k]$ y $a_y[k]$ las aceleraciones, el vector de variables de estado $X[k]$ resulta:

$$X[k] = \begin{pmatrix} x[k] \\ y[k] \\ v_x[k] \\ v_y[k] \\ a_x[k] \\ a_y[k] \end{pmatrix}.$$

Para determinar la matriz de transición se utiliza el modelo mecánico de una partícula con aceleración constante, por lo que:

$$a_x[k+1] = a_x[k],$$

siendo análogo para la aceleración en y. Dado que la aceleración es la derivada primera de la velocidad, si se conoce la aceleración en el frame k, la velocidad en el frame $k+1$ puede estimarse como

$$v_x[k+1] = v_x[k] + a_x[k],$$

cumpliéndose lo análogo para la dirección y. Razonando de la misma manera, se tiene una estimación para la posición en el frame $k+1$ en función de la posición, la velocidad y la aceleración en el frame k:

$$x[k+1] = x[k] + v_x[k] + \frac{1}{2}a_x[k],$$

siendo análogo para la posición según y.

Así, el vector de variables de estado en el frame $k+1$ se relaciona con el del frame k de la siguiente manera:

$$X[k+1] = \begin{pmatrix} 1 & 0 & 1 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1/2 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} X[k],$$

y la matriz de transición T resulta entonces

$$T = \begin{pmatrix} 1 & 0 & 1 & 0 & 1/2 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1/2 \\ 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}.$$

No se utilizaron de variables de control en este caso, por lo que se fijó $G = 0$.

El filtro además compara el estado predicho en tiempo k con observaciones hechas en ese instante. Para ello, dado un vector de observaciones $Z[k]$, halla un estimador del mismo $\hat{Z}[k]$ según la ecuación:

$$\hat{Z}[k] = HX[k] + v[k],$$

$v[k]$ ruido blanco gaussiano de media 0 y varianza σ_v que se asume conocida. H es la llamada matriz de observación. Ésta determina la observación que se obtendría si el estado fuese totalmente conocido.

En este caso, como variables de observación se utilizaron la posición y la velocidad del punto (en ambas direcciones). La posición se obtiene directamente como el punto medio del segmento inferior del bounding box, mientras que la velocidad se estima como la diferencia entre la posición actual y la posición en el frame

anterior (en x e y). Por lo tanto, el vector $Z[k]$ de las variables de observación está compuesto por:

$$Z[k] = \begin{pmatrix} x[k] \\ y[k] \\ v_x[k] \\ v_y[k] \end{pmatrix}.$$

La matriz de observación resulta entonces:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

Los valores de las varianzas utilizadas se definieron variando las mismas y analizando el comportamiento del estimador en función del comportamiento esperado, resultando finalmente $\sigma_w = 0{,}0001$ y $\sigma_v = 100$.

Definidas estas cantidades, el proceso del filtrado es el siguiente:

1. Se predice el estado según la ecuación $X[k+1] = TX[k]$.
2. Se estima la observación según $\hat{Z}[k+1] = HX[k+1]$.
3. Se mide la observación $Z[k+1]$ y se computa la diferencia con el estado estimado $D[k+1] = Z[k+1] - \hat{Z}[k+1]$.
4. Se actualiza el estado estimado según la ecuación $X[k+1] = X[k+1] + W[k+1]D[k+1]$, donde $W[k]$ es la llamada *ganancia de Kalman*. Esta ganancia es calculada de manera tal que se minimiza el error cuadrático medio de la diferencia $D[k+1]$ [56].

Al igual que todas las detecciones de video, esto fue implementado en OpenCV, dado que cuenta con una implementación propia de un filtro de Kalman.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice D

Detección y seguimiento de marcadores

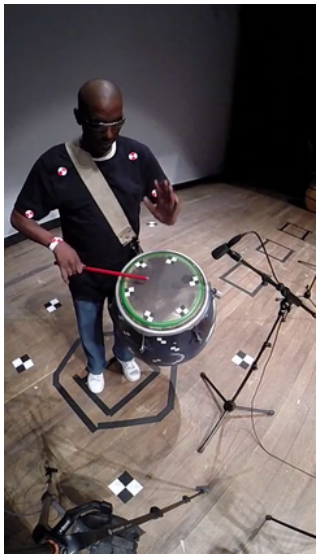
En la sesión de grabación del seis de Setiembre de 2014 se colocaron marcadores en la escena con el fin de determinar con precisión la ubicación de ciertos puntos en el espacio. Se entiende por marcador a una cuadrícula de cuatro espacios cuyos cuadrados opuestos son del mismo color. Dichos marcadores se ubicaron en el suelo, en el tambor y en los brazos del intérprete, como se muestra en la figura D.1(a).En este caso se utilizaron marcadores blancos y rojos para el cuerpo del intérprete, y blancos y negros para el resto de la escena.

Su utilización fue pensada para, al igual que las cámaras dispuestas como par estéreo, poder triangular dichos puntos con facilidad y así obtener un modelo 3D de la escena. Sin embargo, también resultan útiles para otras aplicaciones. Pueden utilizarse para obtener con precisión la zona en la que se encuentran los brazos del intérprete para ayudar a realizar el filtro de piel. También se pueden utilizar para validar los algoritmos de seguimiento de la piel, el palo y la lonja. Resulta de interés por lo tanto, tener la ubicación de los marcadores en la escena durante todo el video. Para ello se realizó un algoritmo de seguimiento de la posición de los mismos.

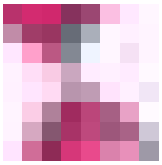
D.1. Algoritmo de seguimiento de marcadores

Primeramente se marcan en el primer frame los puntos a seguir manualmente. Para cada punto marcado se crea un template o imagen de referencia -llamado desde ahora t_{ref}- recortando una pequeña área en torno al punto. Un ejemplo de los templates utilizados se muestra en la Figura D.1(b). A partir de estos puntos se procede a determinar cuáles serán los puntos correspondientes en los frames siguientes.

Para obtener la posición del marcador en el frame n se comienza por obtener un



(a) Disposición de los marcadores en la escena.



(b) Ejemplo de t_{ref}.

Figura D.1: Ubicación de los marcadores y un ejemplo de los templates utilizados.

nuevo template de referencia. Para ello, se utiliza la técnica de template matching[1] sobre el frame $n-1$ buscando en éste el template que mejor se aproxime a t_{ref}. Esta búsqueda se restringe a una zona rectangular cuyo centro es el punto previamente detectado en el frame $n-1$. En la figura D.2 se muestra un diagrama de esta etapa, donde t'_{ref} es la salida de la misma.

Luego, se vuelve a utilizar template matching para comparar t'_{ref} con el frame n. Esta búsqueda también se restringe a una zona rectangular cuyo centro coincide con el punto detectado en el frame $n-1$. El template determinado en este caso -t_{final}- junto con la zona de búsqueda se muestran en la Figura D.3. El centro de t_{final} será la estimación de la ubicación del marcador para el frame n. Dicho template se utiliza para determinar los casos en los que hubo oclusión, como se verá más adelante.

Luego de procesados todos los cuadros, el resultado se imprime en un archivo de texto conteniendo el valor de las coordenadas de los puntos para cada frame.

[1]La técnica de template matching admite varios métodos para realizar la comparación entre el template y la imagen. En este caso se resolvió utilizar la distancia definida como:

$$R(x,y) = \sum_{x',y'} (T(x',y') - I(x+x',y+y'))^2 \tag{D.1}$$

donde I es la imagen, T el template, y R el resultado.

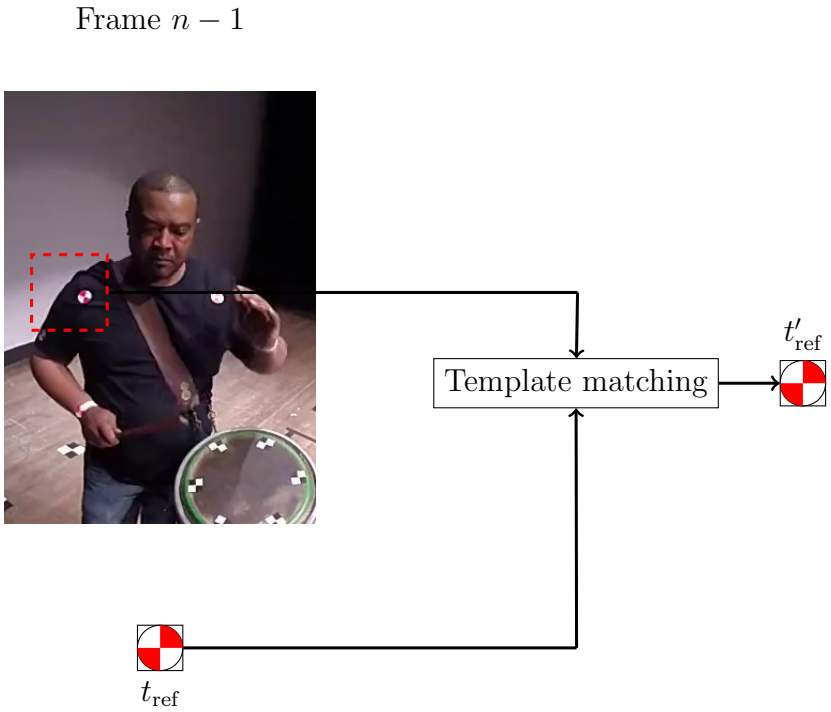


Figura D.2: Obtención del nuevo template de referencia.

D.1.1. Oclusiones y diferencias de iluminación

Se tuvo en consideración para el desarrollo del algoritmo los casos de puntos que presentan diferencias de iluminación y oclusiones durante el video. Para contemplar los casos en que esto ocurre, al determinarse el punto correspondiente en el nuevo frame, se estudia si existe para ese punto una oclusión o cambios en la iluminación. Se decidió tomar como medida de similitud la norma euclídia entre los templates t_{ref} y t_{final} para determinar si éstos son suficientemente diferentes. Además, se determinó previamente el valor de la norma típica para el caso de no oclusión analizando los videos de la base de datos. Por lo tanto imponiendo que la norma de la diferencia entre ambos templates sea mayor a la norma para el caso de no oclusión, se determina cuando ocurre una oclusión. En ese caso, se descarta el punto hallado y se toma como punto válido el punto correspondiente al frame anterior.

Se tomaron además ciertas medidas con el fin de hacer que el valor de la norma sea más robusto frente a ruido proveniente, por ejemplo, de cambios en la iluminación. En primer lugar se introdujo un filtro pasabajos de manera que t_{ref} se actualice en cada frame en el caso de no ocurrir una oclusión. La implementación se realizó de la siguiente forma:

$$t_{\text{ref}} = t_{\text{ref}} \times \alpha + t'_{\text{ref}} \times (1 - \alpha) \tag{D.2}$$

El valor de α se fijó en 0.95 en base a pruebas realizadas en la base de datos. Por

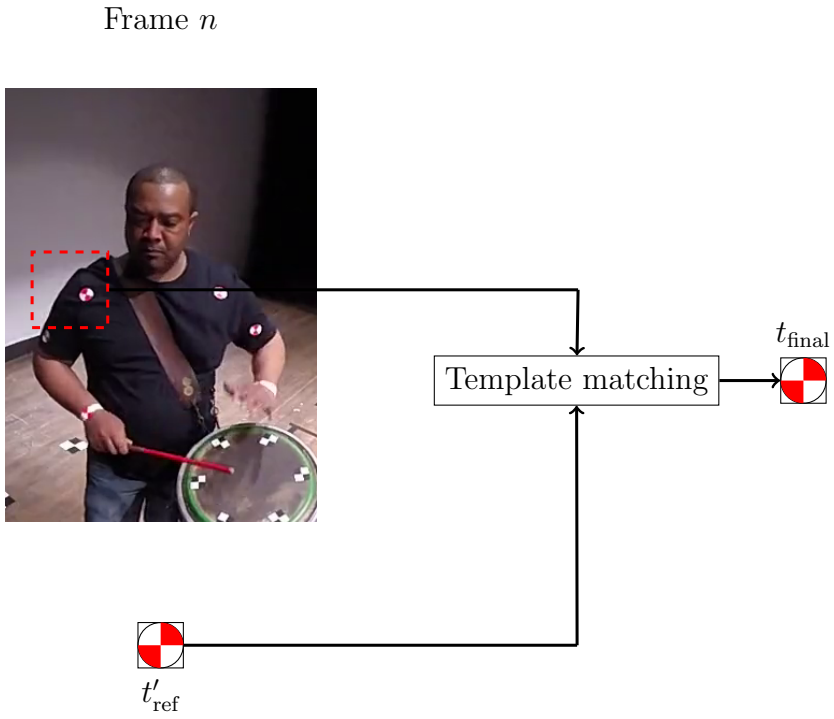


Figura D.3: Estimación final de la posición del marcador en el frame actual.

lo tanto se obtiene un template que se va actualizando a medida que transcurre el video, pesando con mayor importancia al t_{ref} ya que se sabe con certeza que es un template bueno. De esta forma se obtiene no solo un mejor resultado en el valor de la norma a la hora de detectar una oclusión, si no también una mejor detección de t'_{ref} en el primer template matching del algoritmo.

Otra medida que se tuvo en cuenta fue la de comparar los canales Cr normalizados correspondientes al espacio de color YCbCr de los templates t_{ref} y t_{final}, en vez de compararlos directamente. De esta forma, al no considerar el canal Y el cual contiene información sobre la luminancia, se obtienen templates más robustos frente a cambios en la iluminación.

D.1.2. Corrector

El algoritmo también brinda la posibilidad de corregir un punto que esté mal detectado, siempre que la distancia entre éste y el punto correcto sea de menos de veinte píxeles. Para esto se debe pausar la ejecución mediante el uso de la tecla 'r' y utilizar el mouse para marcar el punto deseado. Desde ese momento el programa sigue ejecutandose normalmente pero con el punto corregido para ese frame en particular y también para los siguientes.

D.2. Visualizador

Para poder visualizar el resultado del algoritmo anterior se implementó un programa que permite cargar el archivo de puntos correspondientes al mismo y un video para luego mostrarlo en pantalla. Dicho visualizador realiza muy poco procesamiento por lo que se visualiza el resultado en un tiempo similar al real. Además se incorporó la posibilidad de corregir un punto mal detectado de igual forma que en el algoritmo de detección de marcadores pero para un frame en particular. Para finalizar, el algoritmo crea un archivo de texto igual al que recibe como entrada pero con los puntos corregidos. De esta forma se tienen dos instancias en las que se puede corregir los puntos mal detectados: la primera con la intención de corregir el algoritmo desde un frame en adelante, y la segunda para corregir un punto mal detectado en un frame en particular independiente del resto.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice E

Reconstrucción 3D de la escena

Para poder analizar una improvisación de repique en detalle, puede resultar interesante disponer de un modelo 3D de la escena. A partir de esta información se pueden extraer características que permitan describir con precisión aspectos de la técnica interpretativa (por ejemplo, el ángulo de incidencia del palo sobre la lonja). Existen varios algoritmos que permiten encontrar a partir de dos imágenes estéreo de la escena y los parámetros intrínsecos de las cámaras la transformación que lleva la imagen 2D a la escena 3D. Con esto es posible saber la posición de un objeto respecto a las cámaras, y repitiendo este procedimiento, la posición relativa de los objetos entre sí. A continuación se presenta un resumen de los aspectos que se investigaron para la inclusión de este enfoque en el cálculo de las características del video. Dado que se contaba con un tiempo reducido para la ejecución del proyecto y se tenía sólo un video de la base Zavala etiquetado, se decidió no incluir este enfoque en la solución final.

E.1. Descripción de la escena

Como se explicó en la Sección 2.2, se realizó un registro en la sala Zavala Muniz del Teatro Solís en el que se utilizaron dos cámaras GoPro Hero Black 3+ formando un par estéreo. En la transformación 3D a 2D que se da al adquirir imágenes, se pierde la información de profundidad de la escena, por lo que para recuperarla es necesario tener dos cámaras en esta configuración [33]. Para simplificar la reconstrucción, se incluyeron marcadores en la escena, de manera de tener puntos fácilmente ubicables en las imágenes estéreo. Es por ello que se marcaron los tambores con pequeños tableros en blanco y negro, ya que de esta forma el punto central es determinado con buena exactitud en las imágenes adquiridas. Adicionalmente, se marcaron las zonas de interés del cuerpo del intérprete y el piso, como se observa en la Figura E.2. En el caso de los intérpretes se marcaron el hombro, la articulación que une el brazo con el antebrazo y la parte anterior y posterior de la muñeca de ambos brazos. Dado que las cámaras iban a ser manipuladas durante el registro y su posición es una variable crítica del problema de triangulación, para asegurarse de que la configuración permaneciera intocada se construyó un soporte,

como se muestra en la Figura E.1. La construcción del mismo estuvo a cargo de Roberto Rodríguez y Sergio Beheregaray del taller del IIE.

Figura E.1: Soporte utilizado para la configuración estéreo de las cámaras.

E.2. Calibración de cámaras

El término calibración de cámaras refiere al proceso de encontrar el conjunto de parámetros intrínsecos y extrínsecos que modelan el proceso de formación de la imagen a través de la óptica de las mismas [33]. Esto consiste en asociar un rayo que pasa por el centro óptico de la cámara con un punto del plano de la imagen (asumiendo un modelo pinhole simple [14]). Para ello, se adquieren imágenes de un patrón de calibración de estructura espacial conocida y se ponen en correspondencia ciertos puntos de su estructura con su proyección en las imágenes. A continuación se presenta un breve desarrollo de estos puntos.

E.2.1. Parámetros intrínsecos y extrínsecos

Parámetros intrínsecos

Los parámetros intrínsecos son aquellos que definen la geometría interna y la óptica de la cámara. Es decir que determinan un modelo de cómo la cámara proyecta los puntos del mundo 3D al plano de la imagen en 2D. Dicho modelo es válido siempre y cuando no varíen las características y posiciones relativas de la óptica y el sensor imagen [16]. Los parámetros se detallan a continuación:

Punto principal: El punto principal (punto C, Figura E.5) es el punto intersección entre el plano de la imagen y el eje óptico. Este último se define como la recta perpendicular al plano de la imagen que pasa por el centro de cámara O. Las coordenadas de este punto suelen especificarse en píxeles, y son expresadas respecto al sistema solidario al plano de la imagen (O',x',y').

Figura E.2: Imagen estéreo de un intérprete (Sergio Ortuño) durante el registro de la base *Zavala* con los marcadores utilizados.

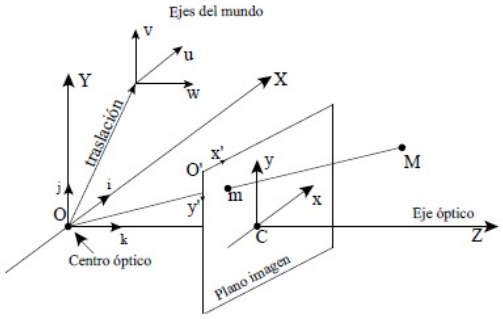


Figura E.3: Parámetros intrínsecos de una cámara - Imagen extraída de [16].

Distancia focal: La distancia focal de una cámara es la distancia existente entre su centro (punto O) y el punto principal. Las coordenadas de este punto suelen especificarse en píxeles horizontales y verticales.

Parámetros extrínsecos

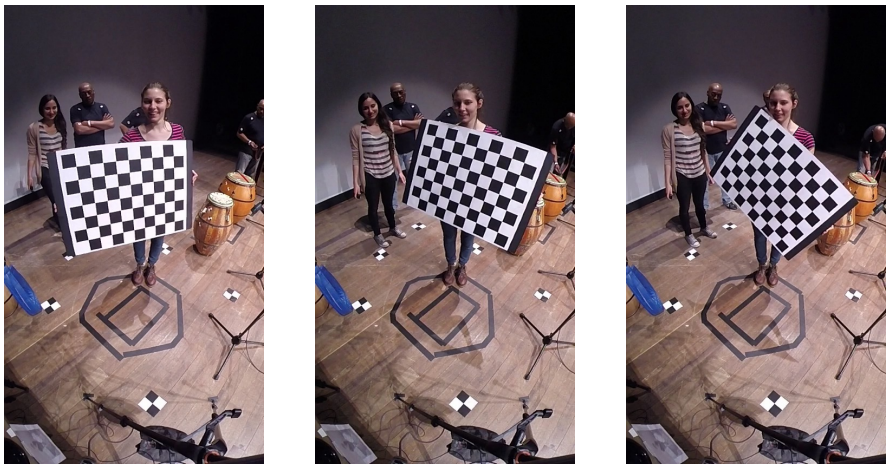
Éstos relacionan los sistemas de referencia del mundo real y la cámara, describiendo la posición y orientación de la cámara en el sistema de coordenadas del mundo real.

Vector de traslacion T: Vector que determina la ubicación del centro óptico de la cámara (O) con respecto a los ejes del mundo real (v,u,w).

Matriz de rotación R: Matriz que relaciona la rotación de la posición de la cámara (O,X,Y,Z) con respecto a los ejes del mundo real (v,u,w).

E.2.2. Calibración de una cámara

Para realizar la reconstrucción 3D, es necesario primero caracterizar cada cámara por separado. Para ello se utilizó como patrón de calibración un tablero de dimensiones 70x56cm, donde cada cuadrado medía 7cm de lado. Se realizó la calibración de cada cámara utilizando la herramienta *Camera calibration Toolbox for Matlab* [4] ya que ha sido utilizada en varios trabajos a nivel académico y es reconocida por sus buenos resultados. Las imágenes usadas para realizar las pruebas provienen de fotos tomadas durante el rodaje con las cámaras mencionadas anteriormente, con una resolución de 848x480 píxeles (ver Figura E.4). Las fotos fueron obtenidas entre toma y toma a lo largo del rodaje para considerar el caso en que las posiciones de las cámaras sufrieran algún cambio. Se supuso igualmente que en caso de ser así, dichos cambios serían mínimos porque las cámaras se mantuvieron en el soporte durante todo el registro.



(a) (b) (c)

Figura E.4: Algunas de las imágenes utilizadas para realizar la calibración de cada cámara

Para computar la calibración el algoritmo recibe las medidas del tablero y las posiciones de las esquinas de cada imagen. Luego devuelve los parámetros

intrínsecos de la cámara. Los resultados se presentan en las Tablas E.1 y E.2.

Largo Focal:	f = [371.87 ; 378.06] ± [1.94 ; 2.07]
Punto principal:	cc = [423.00 ; 231.43] ± [2.52 ; 2.44]
Distorsión:	kc = [-0.2930 ; 0.1256 ; -0.0003 ; -0.0038 ; 0.0000]

Tabla E.1: Parámetros intrínsecos cámara Izquierda.

Largo Focal:	f = [372.73 ; 379.55] ± [2.17 ; 2.58]
Punto principal:	cc = [416.44 ; 228.54] ± [2.86 ; 2.62]
Distorsión:	kc = [-0.2700 ; 0.0939 ; -0.0017 ; -0.0013 ; 0.0000]

Tabla E.2: Parámetros intrínsecos cámara derecha.

Como puede observarse en dichas tablas, los parámetros intrínsecos de ambas cámaras son muy similares entre sí. Esto era de esperarse debido a que son cámaras iguales (GoPro Hero Black 3+ en ambos casos).

E.3. Geometría de un par estéreo

De manera de entender cómo se configuró el par estéreo para el rodaje, se explican a continuación los principales parámetros de un par estéreo (según [33]). Se anotó con el subíndice l y r a los parámetros correspondientes a las cámaras izquierda y derecha respectivamente.

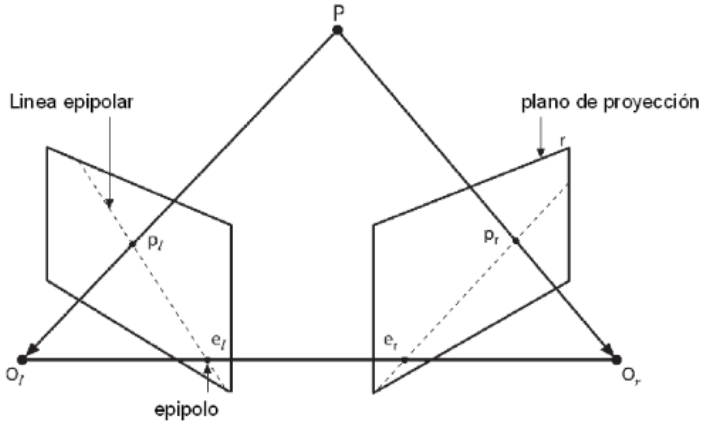


Figura E.5: Parámetros de un par estéreo - Imagen extraída de [33].

- O_l y O_r: centro de la cámara.

- Π_l y Pi_r: plano de la imagen.
- P: un punto del espacio
- p_l y p_r: proyección del punto P en el plano imagen
- e_l y e_r: epipolo

Se define como epipolo de la cámara izquierda (e_l) a la proyección del centro de la cámara derecha (O_r) sobre el plano de la imagen izquierda (Π_l). La definición es análoga en el caso de e_r.

E.4. Calibración estéreo

Una vez obtenidos los parámetros de cada cámara, se caracterizó el par estéreo en su conjunto. Ello consistió en determinar los vectores de traslación y rotación entre las cámaras ($R_{camaras}$ y $T_{camaras}$) [33]. Para el registro se colocaron las cámaras en paralelo, de manera de que el desplazamiento de una respecto a la otra se de solamente en una dirección (como se muestra en la Figura E.6). Para determinar la separación entre ellas, se asumió que las cámaras eran constructivamente iguales y que por lo tanto era razonable suponer que tenían igual distancia focal f. A su vez, al colocarlas en paralelo se supuso también que los centros de las cámaras estaban relacionados simplemente por un vector de traslación $T_{camaras}$ de dirección x (indicada en el dibujo) y módulo B, y que por tanto la matriz de rotación entre cámaras era la identidad. Dicha matriz se relaciona con el vector de rotación $R_{camaras}$ mediante *Rodrigues* [23], y en este caso una matriz de rotación igual a la identidad implica un vector de rotación nulo. Dada esta configuración, la proyección de un punto del espacio P tiene igual coordenada y en los planos de ambas cámaras, por lo que esta dimensión no se consideró. Sin embargo, la coordenada x de dicho punto presenta valores distintos en cada proyección, cuya diferencia se le conoce como disparidad (d). Dado un punto P a una distancia Z del plano de las cámaras, la resolución en profundidad h del par estéreo estará dada por la relación [69]:

$$h = \frac{dZ^2}{fB}$$

Dada una disparidad de 1px y teniendo en cuenta que los intérpretes se encontrarían aproximadamente a 1m de las cámaras, se deberían colocar las cámaras con una separación de $B = 26{,}4cm$ para obtener una resolución de 1cm. Sin embargo, éstas se colocaron finalmente con una separación de $B = 30cm$ para lograr una resolución mayor y mantener cierto margen de error respecto a la ubicación de los intérpretes. Dicho cálculo fue realizado en la etapa de pre-producción del registro.

Finalmente, para caracterizar el par estéreo se utilizó el *Stereo Toolbox Calibration* de Matlab, en particular, el algoritmo *calib_stereo.m*. Este algoritmo utiliza los parámetros intrínsecos previamente calculados de cada cámara e imágenes simultáneas del damero para determinar los vectores $R_{camaras}$ y $T_{camaras}$. Los

resultados de la calibración se presentan en la Tabla E.3. Se puede observar que el vector $R_{camaras}$ es aproximadamente nulo, como era de esperarse dada la configuración de las cámaras (ver Figura E.6). A su vez, se ve que el vector de traslación $T_{camaras}$ presenta valores pequeños en las coordenadas x y z, y un valor aproximadamente de 300mm en la coordenada y, que corresponde al corrimiento horizontal de las cámaras de 30cm. Éste aparece en la coordenada y debido a que las cámaras se dispusieron de manera vertical para el rodaje (ver Figura E.1).

$R_{camaras}$:	om = [-0.017 0.034 0.012] ± [0.001 0.001 0.000]
$T_{camaras}$:	T = [-5.30 300.48 -4.450] ± [0.77 0.87 0.38]

Tabla E.3: Parámetros de calibración estéreo.

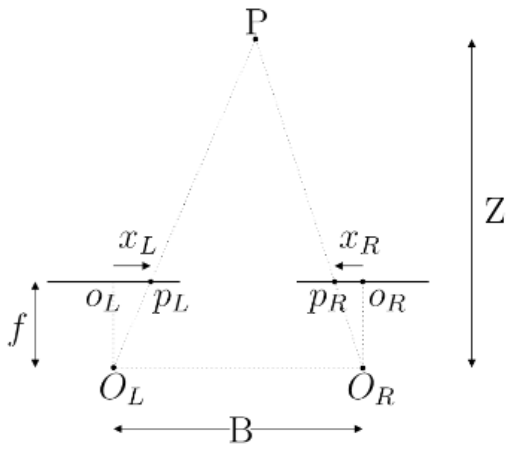
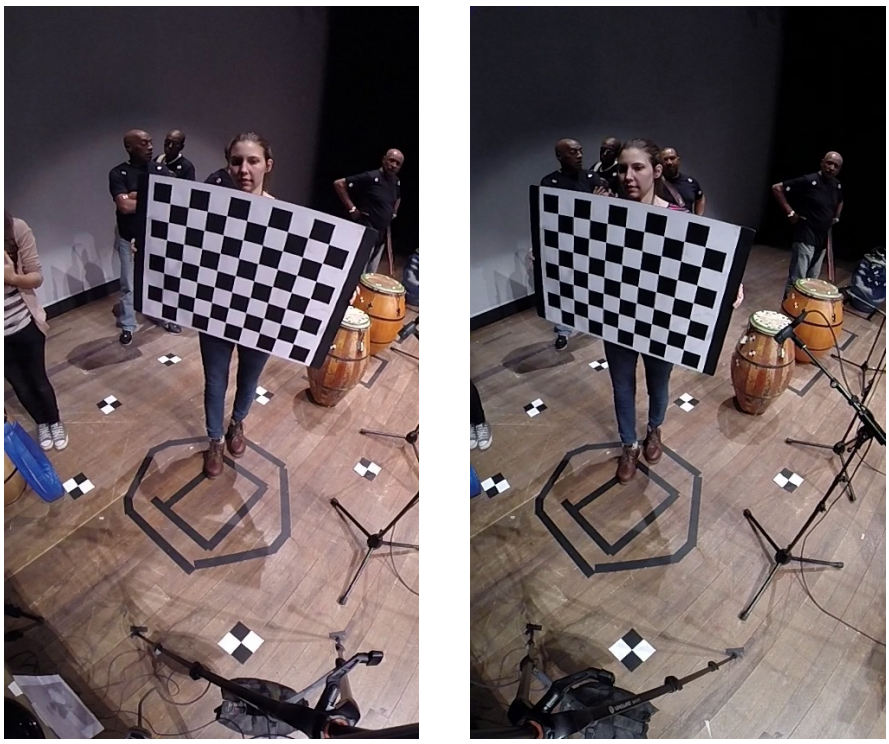


Figura E.6: Geometría de la configuración utilizada en el par estéreo - Imagen extraída de [58]



(a) (b)

Figura E.7: Imágenes para calibración estéreo.

E.5. Triangulación

El proceso de triangulación consiste en encontrar la posición de un punto del espacio a partir de la proyección del mismo en dos imágenes (de cámaras izquierda y derecha). Observando la Figura E.6, puede verse que conociendo la distancia focal, la disparidad y la posición relativa entre las cámaras es simple determinar la profundidad Z del objeto en la escena a partir de la expresión:

$$Z = \frac{Bf}{d}$$

Una vez determinados los parámetros de las cámaras y del par estéreo, se realizó una prueba triangulando los puntos para dos frames dados (uno correspondiente a cada cámara) de uno de los videos de la base Zavala. Para ello, primero se determinó el desfasaje en frames entre las cámaras para ese video, y así se obtuvieron frames correspondientes. Para realizar la triangulación se debieron encontrar puntos comunes entre ambos frames. Éstos se hallaron mediante el algoritmo ASIFT [1], determinando las correspondencias que se muestran en la Figura E.9. Para aumentar la eficiencia computacional de este algoritmo se realizó una previa rectificación de las imágenes (ver Figura E.8), como se recomienda en [69]. Dos imágenes se consideran rectificadas si sus líneas epipolares se intersectan en el infinito, como es el caso de la Figura E.6.

(a) (b)

Figura E.8: Imágenes rectificadas

Finalmente se computó la triangulación estéreo utilizando el *Camera Calibration Toolbox* de Matlab, en particular el algoritmo *stereo_triangulation.m*. Luego se graficó la nube de puntos triangulados mediante el algoritmo *pcl_viewer* [17]. En la Figura E.10 puede verse el resultado de la triangulación.

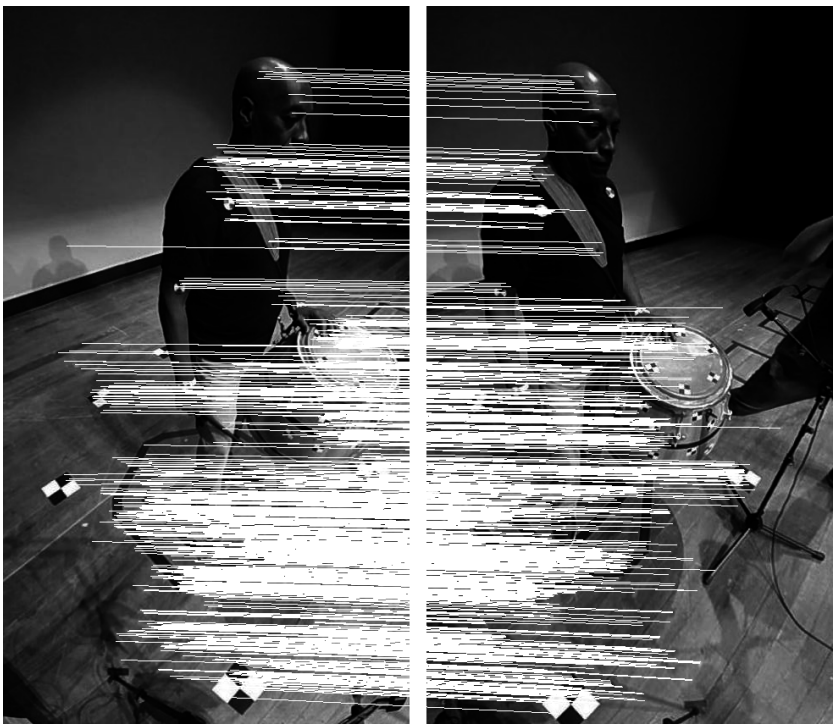
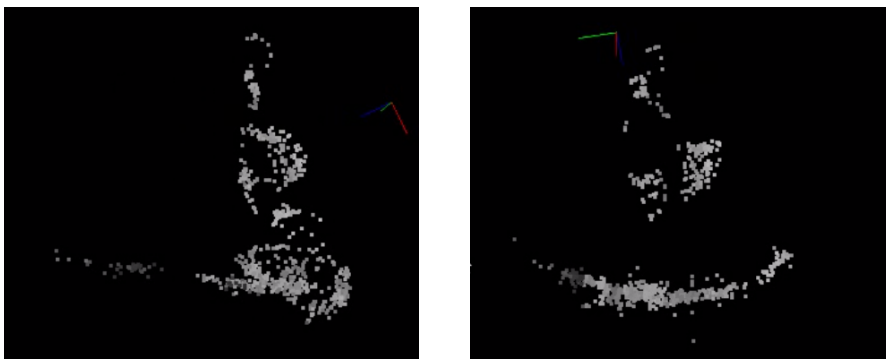


Figura E.9: Correspondencias entre puntos de imagen derecha e izquierda utilizando el algoritmo ASIFT.



(a) Reconstrucción 3D - Vista de perfil

(b) Reconstrucción 3D - Vista de frente

Figura E.10: Reconstrucción 3D de la escena.

E.6. Aplicación de este enfoque al problema

Dado que no se contaba con tiempo suficiente para realizar pruebas adicionales y se tenía un sólo video etiquetado de la base Zavala, se decidió no incluir este enfoque en la solución final. De todas maneras, es interesante discutir cómo podría haber sido la incorporación de la reconstrucción 3D a la determinación de características de video. Una posibilidad hubiera sido utilizar el algoritmo explicado en el Apéndice D para obtener la posición de los marcadores a lo largo del video. Luego, de forma similar a lo explicado en este capítulo, se podría haber triangulado cada pareja de puntos para determinar su ubicación en el espacio. A partir de ellos, se inferiría la posición del tambor y de los brazos del intérprete. Luego, con un procedimiento similar al cálculo de las características geométricas de la Sección 4.4, se podrían haber determinado características geométricas en el espacio. Este enfoque queda como trabajo futuro una vez que se tengan etiquetados los demás videos de la base Zavala.

Esta página ha sido intencionalmente dejada en blanco.

Apéndice F

Software

En esta sección se pretende hacer un repaso del software entregado con el presente proyecto. Para su correcta utilización se debe tener instalado:

1. Ubuntu 14.04 LTS
2. CMake 2.6 o superior
3. GCC 4.4.x o superior
4. GTK+2.x o superior, incluyendo encabezados (libgtk2.0-dev)
5. Git
6. Python 2.6 o superior y Numpy 1.5 o superior, con paquetes de desarrollador (python-dev, python-numpy)
7. ffmpeg o libav development packages: libavcodec-dev, libavformat-dev, libswscale-dev
8. OpenCV 3.0.0 alpha o superior
9. Matlab 2011 o superior
10. Matpy (extensión de Matlab para llamar funciones de Python, ver [13])
11. WEKA 2.7.12 Developer Version

Por una guía de instalación de los puntos 2 al 8, ver por ejemplo [10]. Para la implementación de la segmentación de color se adaptó a C++ el filtro de color del software VLC versión 2.1.5. Dicho filtro fue utilizado en el software del proyecto como una biblioteca bajo la licencia GPL [12].

A continuación se presenta una reseña de cada módulo del software entregado así como las instrucciones para su utilización. El orden de presentación es el orden en el que debería ser ejecutado cada módulo para el correcto funcionamiento del sistema:

- **crearEstructuraDeDirectorios.sh**: El software entregado requiere una estructura de directorios específica para poder funcionar. De manera de asegurar esto, lo primero que se debe hacer es correr este script en un terminal, ejecutando:

  `./crearEstructuraDeDirectorios.sh`.

- **compileAll.sh**: Script para compilar todos los archivos fuente .cpp. Crea, dentro de cada directorio correspondiente, una carpeta llamada `bin` donde se guardan los ejecutables de cada módulo. Se ejecuta de manera análoga a crearEstructuraDeDirectorios.sh. Una vez ejecutado, puede consultarse la utilización de cada módulo yendo a la carpeta `bin` correspondiente y corriendo:

  `./nombreDelMódulo`.

  Por ejemplo, si se quiere saber cómo utilizar `detectarElipse`, se debe ir a `detectarElipse/bin` y correr

  `./detectarElipse`.

- **detectarElipse**: Implementación en C++ de la detección de la elipse, utilizando la librería OpenCV. Dentro del directorio se encuentra el archivo fuente `detectarElipse.cpp` y un archivo `CMakeLists.txt` necesario para su compilación. Esta estructura se repite para todos los módulos implementados en C++. Para utilizarlo, se debe ejecutar:

  `./detectarElipse <video_path>`.

  donde `<video_path>` es la ubicación del video que se quiere procesar. En `/Datos/Elipses/` se guardan, en formato .txt, los 5 parámetros de la elipse detectada en cada frame.

- **detectarPiel**: Módulo de detección de piel, también implementado en C++ utilizando la librería OpenCV. Existen dos implementaciones distintas, una para cada base de datos, ya que en cada una se modifica el color utilizado para filtrar en base al color de piel de los intérpretes.

  Se ejectua desde el `bin` correspondiente de la siguiente manera:

  `./detectarPiel [-f <data_path>] <video_path>`.

  El argumento opcional `<data_path>` indica la ubicación de un archivo .csv que tiene los datos de las componentes YUV utilizadas para la clasificación. En caso de que este no se especifique, antes de la detección se ejecuta una etapa de muestreo, en la que el usuario muestrea manualmente puntos de la piel de los intérprete y de los elementos que se confunden con ésta en el

filtro de color (el tambor en el caso de la base eMe, el piso de la sala en el caso de la Zavala). Los datos de este muestreo se guardan en formato .csv en `/Datos/MuestreoColor/`.

Una vez corrido el algoritmo, las detecciones en cada frame se guardan en `/Datos/Piel/<video_name>/`, siendo `<video_name>` el nombre del video procesado. Dentro de cada uno de estos directorios, se generan dos carpetas, donde el nombre de cada una responde al clasificador utilizado: `Arbol` y `RandomForest`.

- **boundingBox**: Segmentación de la mano izquierda en base a la detección de piel y la segmentación de la elipse, por lo que dichos módulos deben ejecutarse antes de éste. También está implementado en C++ usando OpenCV, por lo que se utiliza de manera análoga a los anteriores:

  `./boundingBox <video_path>`.

  El resultado de este módulo es la ubicación de un punto (que es el producto de aplicar un filtro de Kalman al punto medio del segmento inferior del bounding box de la mano). Se guardan, para cada frame, las coordenadas x e y del punto, en formato .txt. Estos datos se ubican en `/Datos/BoundingBox/`.

- **detectarPaloVerde** y **detectarPaloRojo**: Implementaciones en C++ usando OpenCV de la detección del palo en videos en los que el palo utilizado es de color verde o rojo respectivamente. Dado que el primero utiliza la ubicación de la elipse, dicho módulo debe ser ejecutado antes. Se usa de manera análoga a los anteriores y los resultados de la detección se guardan en `/Datos/Palo/`, en formato .txt. Las primeras dos columnas de este archivo representan las coordenadas x e y de la punta del palo detectada y las dos restantes las coordenadas x e y del otro extremo del palo. Cada fila representa corresponde a un frame.

- **detectarMarcadores**: Detección y seguimiento de marcadores, también implementado en C++ usando OpenCV. Se utiliza de manera análoga a los anteriores:

  `./deteccionMarcadores <video_path>`.

  Al ejecutarse, el usuario debe seleccionar en el primer frame los marcadores a seguir. Si esto no fuese posible (por ejemplo porque los marcadores que se quieren seguir no se ven), apretando la tecla '0' se avanza en el video. Cuando se tenga un frame adecuado, presionado 'Space' se habilita la selección de marcadores. Para ello, se puede hacer zoom con el scroll del mouse de manera de marcarlos de manera más precisa. Cuando se hayan seleccionado todos los marcadores, se presiona 'ESC' y comienza el seguimiento. Este módulo cuenta con la posibilidad de corregir las detecciones durante la ejecución. Para ello, basta con hacer click cerca del marcador que se quiere corregir.

Además, la ejecución puede pausarse y reanudarse en cualquier momento presionando la tecla 'R'.

Los resultados se guardan en `/Datos/Marcadores/`, en formato .txt. Este archivo de salida tiene tantas filas como frames tenga el video, y $2 \times N$ columnas, siendo N la cantidad de marcadores sobre los que se hizo seguimiento.

- **visualizador**: Implementación en C++ para visualizar las detecciones de palo, mano, lonja y marcadores que estén disponibles en el directorio `/Datos/`. Utiliza la librería OpenCV y se usa de esta esta forma:

  `./visualizador <video_path>`.

  El código permite corregir la ubicación de los marcadores para un determinado frame, con el mismo procedimiento que el utilizado en **detectarMarcadores**. Las nuevas ubicaciones son guardadas en `/Datos/Marcadores/<video_name>/`, en formato .txt. Además, la ejecución puede pausarse y reanudarse en cualquier momento presionando la tecla 'R'. En la Figura F.1 se muestra una captura de esta interfaz.

- **audioComputation**: Módulo implementado en Matlab para computar el spectral flux de la señal de audio y sus MFCCs. El cálculo del spectral flux (`spectralFluxComputation.m`) se realiza utilizando la librería de procesamiento de audio `ra` de Python, la cual se incluye en la carpeta `ra`. Esta es una librería desarrollada por Leonardo Nunes, en colaboración con Martín Rocamora. El código no está disponible de manera libre por el momento, pero está prevista su liberación en un futuro cercano (existe un artículo actualmente en proceso de evaluación en el cual se libera el código).

  Para llamar funciones de Python en Matlab se usa el mex matpy [13]. El spectral flux calculado se guarda en `/Datos/spectralFlux/`, donde la primer columna contiene los centros de cada ventana utilizada en el cálculo y la segunda son los datos del spectral flux propiamente dichos.

  Esta etapa implementa además la detección de eventos que luego es utilizada para el etiquetado. Los onsets detectados se guardan en `/Datos/onsets/`.

  El cálculo de los MFCCs se realiza en `mfccComputation.m`. En este caso, además de seleccionarse el audio a procesar, se debe seleccionar el archivo de etiquetas correspondiente. Los 40 coeficientes de los MFCCs calculados para cada golpe se guardan en `/Datos/MFCCs/`.

- **featureComputation**: Módulo implementado en Matlab para computar el total de características propuestas en el modo audio y video para todos los registros etiquetados. El archivo `main.m` usa la función `caracteristicas_multimodal.m` para realizar el cálculo de las características. Dicho cálculo se realiza utilizando los datos de las detecciones que se encuentran en la carpeta `/Datos/`.

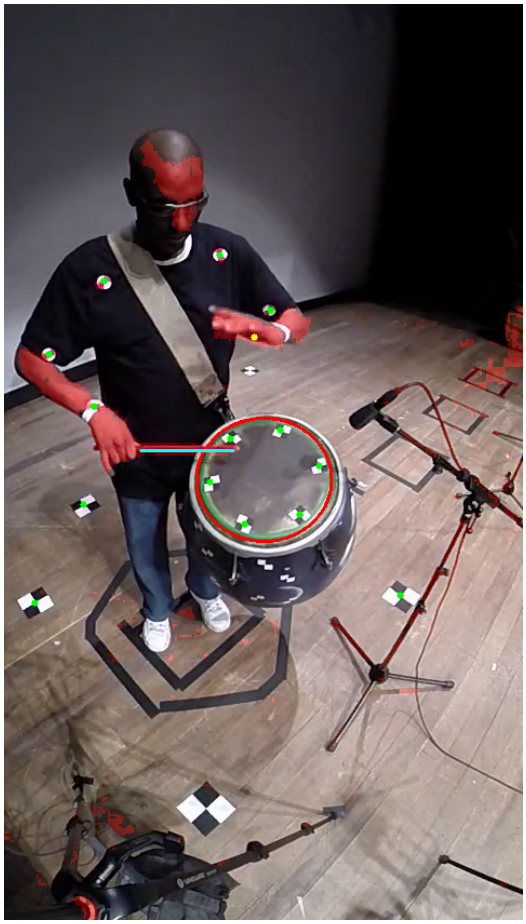


Figura F.1: Ejemplo de la salida del visualizador. Los puntos verdes indican la posición de los marcadores, el punto amarillo el estimador de la posición de la mano, el segmento celeste la posición del palo y la elipse roja la posición de la lonja. Superpuesto a la imagen se muestra, en rojo, la piel del intérprete detectada.

Además imprime los archivos de extensión `.arff` reconocidos por la plataforma WEKA en el directorio `/featureComputation/features/`. En dicha carpeta se imprime un archivo `.arff` por video.

- **concatenarTodos.sh**: Script para generar los `.arff` para las pruebas cruzadas entre intérpretes. Se crea dentro del directorio correspondiente, un archivo en el cual se concatenan los `.arff` que deberían usarse para realizar cada prueba. Estos archivos se encuentran en subcarpetas dentro de `/featureComputation/features/3_train_1_test/` organizadas según el intérprete. Para ejecutar el script se debe correr:

  `./concatenarTodos.sh`.

  Una vez generados los archivos, si se quieren reproducir las pruebas de la Sección 7 en el software WEKA con el intérprete *fulano*, se debe utilizar

para entrenar el archivo que se encuentra en la carpeta

`/featureComputation/features/3_train_1_test/datos_train_`*fulano*

, mientras que para clasificar se debe usar el archivo que se ubica en

`/featureComputation/features/3_train_1_test/datos_test_`*fulano*

Referencias

[1] ASIFT. http://www.ipol.im/pub/art/2011/my-asift/. Acceso: 14-05-2015.

[2] Background Substractor. http://docs.opencv.org/3.0-beta/modules/video/doc/motion_analysis_and_object_tracking.html?highlight=background#BackgroundSubtractorMOG2%20:%20public%20BackgroundSubtractor. Acceso: 11-05-2015.

[3] Background Substractors Methods - OpenCV. http://docs.opencv.org/3.0-beta/doc/tutorials/video/background_subtraction/background_subtraction.html?highlight=background. Acceso: 11-05-2015.

[4] Camera calibration toolbox for matlab, Jean-Yves Bougue, November 2012. http://www.vision.caltech.edu/bouguetj/calib_doc/. Acceso: 11-05-2015.

[5] Code Laboratories. Clnui sdk and drivers. http://www.codelaboratories.com. Acceso: 11-05-2015.

[6] Definición de timbre - Escuela Universitaria de Música. http://www.eumus.edu.uy/docentes/maggiolo/acuapu/tbr.html. Acceso: 23-05-2015.

[7] Ellipse Detection Using 1D Hough Transform. http://www.mathworks.com/matlabcentral/fileexchange/33970-ellipse-detection-using-1d-hough-transform. Acceso: 11-05-2015.

[8] Hue - en.wikipedia. http://en.wikipedia.org/wiki/Hue. Acceso: 11-05-2015.

[9] Image segmentation - en.wikipedia. http://en.wikipedia.org/wiki/Image_segmentation. Acceso: 11-05-2015.

[10] Installing OpenCV 3.0.0 on Ubuntu 14.04. http://rodrigoberriel.com/2014/10/installing-opencv-3-0-0-on-ubuntu-14-04/. Acceso: 11-05-2015.

[11] Introducción - Reconocimiento de Patrones - Curso de Grado y Posgrado, IIE, Fing. http://eva.fing.edu.uy/pluginfile.php/63323/mod_resource/content/5/Introducci%C3%B3n_2013.pdf. Acceso: 20-05-2015.

[12] Licencia GPL. http://es.wikipedia.org/wiki/GNU_General_Public_License. Acceso: 11-05-2015.

[13] Matpy – call Python from MATLAB. http://algoholic.eu/matpy/. Acceso: 11-05-2015.

[14] Modelo Pinhole. http://en.wikipedia.org/wiki/Pinhole_camera_model. Acceso: 14-05-2015.

[15] OpenCV fitEllipse. http://docs.opencv.org/modules/imgproc/doc/structural_analysis_and_shape_descriptors.html?highlight=fitellipse#fitellipse. Acceso: 11-05-2015.

[16] Parámetros del modelo pinhole. http://iie.fing.edu.uy/investigacion/grupos/gti/timag/trabajos/2013/laser/Procesamiento.html. Acceso: 14-05-2015.

[17] Pcl viewer. http://pointclouds.org/. Acceso: 14-05-2015.

[18] Prime Sense. Openni platform. http://www.openni.org. Acceso: 11-05-2015.

[19] Página web del estudio de música electroacústica. http://www.eumus.edu.uy/eme/. Acceso: 11-05-2015.

[20] Página web personal de luis jure en la escuela universitaria de música. http://www.eumus.edu.uy/docentes/jure/. Acceso: 11-05-2015.

[21] Página web personal de Martín Rocamora en la Facultad de Ingeniería. http://iie.fing.edu.uy/~rocamora/. Acceso: 11-05-2015.

[22] Página web YUV Colorspace.

[23] Rodrigues. http://en.wikipedia.org/wiki/Rodrigues%27_rotation_formula. Acceso: 22-05-2015.

[24] Saturation - en.wikipedia. http://en.wikipedia.org/wiki/Colorfulness#Saturation. Acceso: 11-05-2015.

[25] VLC Media Player. http://www.videolan.org/vlc/. Acceso: 11-05-2015.

[26] Wikipedia - YUV.

[27] Coriún Aharonian. *Músicas populares del Uruguay*. Universidad de la República, 2007.

[28] R.P.W. Duin A.K. Jain and J. Mao. Statistical Pattern Recognition: A Review. *IIE Transactions on Pattern Analysis and Machine Intelligence*, 22(1), 2000.

[29] Lauro Ayestarán. La música negra. La Música en el Uruguay, Montevideo, 1953.

[30] L. Breiman and A. Cutler. Random Forests. `http://www.stat.berkeley.edu/~breiman/RandomForests/`. Acceso: 11-05-2015.

[31] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[32] Juan José Burred and Alexander Lerch. A hierarchical approach to automatic musical genre classification. In *Proceedings of the 6th international conference on digital audio effects*, pages 8–11, 2003.

[33] Guillermo Carbajal, Gastón Marín, and María Clara Pérez. Mira : Microscopio con realidad aumentada y reconstrucción de superficies estéreo - capítulo 8. `http://iie.fing.edu.uy/publicaciones/2010/CMP10`, feb 2010.

[34] C. C. Chibelushi, J. S D Mason, and F. Deravi. Integrated person identification using voice and facial features. In *Image Processing for Security Applications (Digest No.: 1997/074), IEE Colloquium on*, pages 4/1–4/5, 1997.

[35] Darryl PJ Cotton, Paul H Chappell, Andy Cranny, Neil M White, and Steve P Beeby. A novel thick-film piezoelectric slip sensor for a prosthetic hand. *Sensors Journal, IEEE*, 7(5):752–761, 2007.

[36] ER Davies. Finding ellipses using the generalised Hough transform. *Pattern Recognition Letters*, 9(2):87–96, 1989.

[37] Y Demir, E Erzin, Y Yemez, and AM Tekalp. Evaluation of audio features for audio-visual analysis of dance figures. In *European Signal Processing Conference (EUSIPCO)*, 2008.

[38] Simon Dixon. Onset detection revisited. In *Proc. of the Int. Conf. on Digital Audio Effects (DAFx-06)*, pages 133–137, 2006.

[39] Slim Essid, Xinyu Lin, Marc Gowing, Georgios Kordelas, Anil Aksay, Philip Kelly, Thomas Fillon, Qianni Zhang, Alfred Dielmann, Vlado Kitanovski, et al. A multimodal dance corpus for research into real-time interaction between humans in online virtual environments. In *ICMI Workshop On Multimodal Corpora For Machine Learning*, November 2011.

[40] Slim Essid and Gaël Richard. Fusion of Multimodal Information in Music Content Analysis. In Meinard Müller, Masataka Goto, and Markus Schedl, editors, *Multimodal Music Processing*, volume 3 of *Dagstuhl Follow-Ups*, pages 37–52. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, Dagstuhl, Germany, 2012.

[41] Luis Ferreira. El repicado del Candombe. In *V Jornadas Argentinas de Musicología, Instituto Nacional de Musicología "Carlos Vega", Bs.As.*, 1990.

[42] Luis Ferreira. *Los Tambores del Candombe*. Colihue-Sepé Ediciones, 1997.

[43] Andrew W Fitzgibbon, Robert B Fisher, et al. A buyer's guide to conic fitting. *DAI Research paper*, 1996.

[44] Jonathan T Foote. Content-based retrieval of music and audio. In *Voice, Video, and Data Communications*, pages 138–147. International Society for Optics and Photonics, 1997.

[45] Valentino Frati and Domenico Prattichizzo. Using Kinect for hand tracking and rendering in wearable haptics. In *World Haptics Conference (WHC), 2011 IEEE*, pages 317–321. IEEE, 2011.

[46] O. Gillet, S. Essid, and G. Richard. On the Correlation of Automatic Audio and Visual Segmentations of Music Videos. *IEEE Trans. Cir. and Sys. for Video Technol.*, 17(3):347–355, March 2007.

[47] O. Gillet and G. Richard. Automatic transcription of drum sequences using audiovisual features. In *Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP '05). IEEE International Conference on*, volume 3, pages iii/205–iii/208 Vol. 3, 2005.

[48] Olivier Gillet and Gaël Richard. Enst-drums: an extensive audio-visual database for drum signals processing. In *In Proceedings of the 7th International Symposium on Music Information Retrieval (ISMIR 2006*, pages 156–159, 2006.

[49] Roland Goecke and Bruce Millar. Statistical Analysis of the Relationship between Audio and Video Speech Parameters for Australian English. In *ISCA Tutorial and Research Workshop on Auditory-Visual Speech Processing AVSP 2003*, 2003.

[50] Gustavo Goldman. *Candombe. ¡Salve Baltasar! La fiesta de reyes en el barrio sur de Montevideo.* Perro Andaluz Ediciones, 2003.

[51] Guillaume Gravier, Gerasimos Potamianos, and Chalapathy Neti. Asynchrony modeling for audio-visual speech recognition. In *Proceedings of the second international conference on Human Language Technology Research*, HLT '02, pages 1–6, San Francisco, CA, USA, 2002. Morgan Kaufmann Publishers Inc.

[52] Rafael Grompone von Gioi, Jérémie Jakubowicz, Jean-Michel Morel, and Gregory Randall. LSD: a Line Segment Detector. *Image Processing On Line*, 2:35–55, 2012.

[53] Mihai Gurban and Jean-Philippe Thiran. *Multimodal Signal Processing.* Academic Press, Oxford, 2010.

[54] L. Jure. Principios generativos del toque de repique del candombe. In *Coloquio internacional sobre las culturas afroamericanas y la música*, Montevideo, Uruguay, 2011.

[55] Luis Jure. ¡Perico, suba ahí! Pautación y análisis de un solo de repique de Pedro 'Perico' Gularte. In *VII Jornadas Argentinas de Musicología, Instituto Nacional de Musicología "Carlos Vega", Bs.As.*, 1992.

[56] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. *Journal of Fluids Engineering*, 82(1):35–45, 1960.

[57] Anssi Klapuri and Manuel Davy. *Signal processing methods for music transcription*. Springer, 2006.

[58] Federico Lecumberry. Cálculo de disparidad y segmentación de objetos en secuencias de video. Master's thesis, Facultad de Ingeniería - UDELAR, 2005.

[59] Alexander Lerch. *An Introduction to Audio Content Analysis: Applications in Signal Processing and Music Informatics*. John Wiley & Sons, 2012.

[60] Dongge Li, Nevenka Dimitrova, Mingkun Li, and Ishwar K. Sethi. Multimedia content processing through cross-modal association. In *Proceedings of the eleventh ACM international conference on Multimedia*, MULTIMEDIA '03, pages 604–611, New York, NY, USA, 2003. ACM.

[61] A. Lim, K. Nakamura, K. Nakadai, T. Ogata, and H. Okuno. Audio-Visual Musical Instrument Recognition. In *National Convention of Audio-Visual Information Processing Society*, March 2011.

[62] Beth Logan and Stephen Chu. Music summarization using key phrases. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*, volume 2, pages II749–II752. IEEE, 2000.

[63] Beth Logan et al. Mel Frequency Cepstral Coefficients for Music Modeling. In *ISMIR*, 2000.

[64] Petros Maragos, Alexandros Potamianos, and Patrick Gros. *Multimodal Processing and Interaction: Audio, Video, Text*. Multimedia Systems and Applications. Springer, Dordrecht, 2008.

[65] K. D. Martin. Sound-Source Recognition: A Theory and Computational Model. In *Acoustics, Speech, and Signal Processing, 2000. ICASSP'00. Proceedings. 2000 IEEE International Conference on*. PhD thesis, MIT. Cambridge, MA., 1999.

[66] Nathan Moroney, Mark D Fairchild, Robert WG Hunt, Changjun Li, M Ronnier Luo, and Todd Newman. The CIECAM02 color appearance model. In *Color and Imaging Conference*, volume 2002, pages 23–27. Society for Imaging Science and Technology, 2002.

[67] L. Nunes, M. Rocamora, L. Jure, and L. W. P. Biscainho. Beat and Downbeat Tracking Based on Rhythmic Patterns Applied to the Uruguayan Candombe Drumming. In *Proceedings of the 16th International Society for Music Information Retrieval Conference (ISMIR), Málaga, España*, en evaluación, Octubre 2015.

[68] E. Petajan, B. Bischoff, D. Bodoff, and N. M. Brooke. An improved automatic lipreading system to enhance speech recognition. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '88, pages 19–25, New York, NY, USA, 1988. ACM.

[69] J. Preciozzi. Dense urban elevation models from stereo images by an affine region merging approach. Master's Thesis, Universidad de la República, Montevideo, Uruguay, 2006.

[70] L.R. Rabiner and R.W. Schafer. Digital Processing of Speech Signals. Prentice Hall, New Jersey, 1978.

[71] Jagdish L Raheja, Ankit Chaudhary, and Kunal Singal. Tracking of fingertips and centers of palm using Kinect. In *Computational Intelligence, Modelling and Simulation (CIMSiM), 2011 Third International Conference on*, pages 248–252. IEEE, 2011.

[72] Christian Rendl, Patrick Greindl, Michael Haller, Martin Zirkl, Barbara Stadlober, and Paul Hartmann. PyzoFlex: printed piezoelectric pressure sensing foil. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, pages 509–518. ACM, 2012.

[73] M. Rocamora, P. Cancela, and A. Pardo. Query by humming: Automatically building the database from music recordings. *Pattern Recognition Letters, Special Issue on Robust Recognition Methods for Multimodal Interaction*, 2013.

[74] M. Rocamora, L. Jure, and L. W. P. Biscainho. Tools for detection and classification of piano drum patterns from candombe recordings. In *Proceedings of the 9th Conference on Interdisciplinary Musicology (CIM14), Berlin, Germany*, page 382–387, December 2014.

[75] Martín Rocamora. *Transcripción y análisis automático de música de percusión: El Candombe afro-uruguayo como caso de estudio.* PhD thesis, Universidad de la República, Facultad de Ingeniería, en curso, inicio 2012.

[76] Daniel James Ryan. Finger and gesture recognition with Microsoft Kinect. 2012.

[77] Munehiko Sato, Ivan Poupyrev, and Chris Harrison. Touché: enhancing touch interaction on humans, screens, liquids, and everyday objects. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 483–492. ACM, 2012.

[78] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

[79] X. Serra. A Multicultural Approach in Music Information Research. In *International Society for Music Information Retrieval Conference (ISMIR)*, Miami, Florida, USA, 2011.

[80] Stanley Smith Stevens, John Volkmann, and Edwin B Newman. A scale for the measurement of the psychological magnitude pitch. *The Journal of the Acoustical Society of America*, 8(3):185–190, 1937.

[81] ANSI Psychoacoustical Terminology. S3. 20. *New York, NY: American National Standards Institute*, 1973.

[82] Todor Todoroff. Wireless digital/analog sensors for music and dance performances. In *Proc. NIME*, volume 11, pages 515–518, 2011.

[83] Filipe Tomaz, Tiago C, and Hamid Shahbazkia. Improved automatic skin detection in color images. In *Proceeding of VIIth Digital Computing: Techniques and Applications*, pages 419–427, 2003.

[84] See-Ho Tsang, Abdul Haseeb Ma, Karim S Karim, Ash Parameswaran, and Albert M Leung. Monolithically fabricated polymermems 3-axis thermal accelerometers designed for automated wirebonder assembly. In *Micro Electro Mechanical Systems, 2008. MEMS 2008. IEEE 21st International Conference on*, pages 880–883. IEEE, 2008.

[85] Saburo Tsuji and Fumio Matsumoto. Detection of ellipses by a modified Hough transformation. *IEEE Transactions on Computers*, 27(8):777–781, 1978.

[86] Barry Vercoe. The Synthetic Performer in the Context of Live Performance. In *Proc.*
of the International Computer Music Conference (ICMC), pages 199–200, Paris, 1984.

[87] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. 2ndEdition, Morgan Kaufmann, San Francisco, 2005.

[88] P. E. Hart Witten, R. O. Duda and D. G. Stork. *Pattern Classification*. 2ndEdition, Wiley Interscience, 2001.

[89] Yonghong Xie and Qiang Ji. A new efficient ellipse detection method. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 2, pages 957–960. IEEE, 2002.

[90] Lei Xu, Erkki Oja, and Pekka Kultanen. A new curve detection method: randomized Hough transform (RHT). *Pattern recognition letters*, 11(5):331–338, 1990.

[91] Zhengyou Zhang. Microsoft Kinect sensor and its effect. *MultiMedia, IEEE*, 19(2):4–10, 2012.

[92] Zoran Zivkovic. Improved adaptive Gaussian mixture model for background subtraction. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 2, pages 28–31. IEEE, 2004.

[93] Z.Zivkovic and F.van der Heijden. Recursive Unsupervised Learning of Finite Mixture Models. volume 26, 2004.

Índice de tablas

1.1. Comparación del porcentaje de clasificación correcta de los modos audio y video frente al enfoque multimodal para cada tipo de golpe. 14

2.1. Desglose de las tomas realizadas en el registro del dataset Zavala. . 22

3.1. Estadísticas del proceso de etiquetado. *TruePositives* indica el porcentaje de eventos detectados en el flujo espectral que efectivamente corresponden a golpes, mientras que *FalseNegatives* es el porcentaje de golpes que no son detectados por este método. 29

4.1. Tabla comparativa de los posibles ejes mayores con las distintas restricciones. 40
4.2. Porcentaje de errores de clasificación de los conjuntos de entrenamiento y validación, para el Árbol y el *Random Forest*, sobre los intérpretes de la base Zavala. 56

6.1. Selección por correlación y por encapsulado de las características de audio. La columna titulada *Corr* indica la cantidad de veces que se seleccionó la característica correspondiente en el experimiento de selección, usando el método de correlación. La columna *Enc* indica lo análogo para el método por encapsulado. 80
6.2. Selección final de características de audio. 81
6.3. Parámetros óptimos de la selección de audio. 82
6.4. Selección por correlación y por encapsulado del conjunto geométrico. se describe con un número del 1 al 10 la cantidad de veces que una característica fue seleccionada en la validación cruzada. 83
6.5. Selección por correlación y por encapsulado del conjunto DCT . . 85
6.6. Selección por correlación y por encapsulado del conjunto $Seleccion_{geo+DCT}$. 86
6.7. Selección de video 88
6.8. Parámetros óptimos de la selección de video. 89
6.9. Conjunto Multimodal 1. 91
6.10. Selección por correlación y por encapsulado de todas las características extraídas de los modos audio y video. 92
6.11. Conjunto de características obtenidas por el método Multimodal 2. 93
6.12. Parámetros óptimos de las selecciones Multimodal 1 y Multimodal 2, para SVM usando un kernel RBF. 93

6.13. Resultados de la clasificación usando las características de Multimodal 1 y Multimodal 2. 94

7.1. Resultados de la clasificación unimodal con el audio, considerando tres tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 1$. 96
7.2. Resultados de la clasificación unimodal con el video, considerando tres tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 6$ 97
7.3. Resultados de la clasificación Multimodal 1 considerando tres tipos de golpes. Se utilizó un clasificador SVM con $C = 6$ y un kernel RBF de $\gamma = 0{,}4$ 98
7.4. Comparación de los porcentajes de clasificación obtenidos en los modos audio y video, y en la combinación multimodal, utilizando 3 clases. 98
7.5. Resultados de la clasificación unimodal con el audio, considerando seis tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 1$ 99
7.6. Resultados de la clasificación unimodal con el video, considerando seis tipos de golpes. Se utilizó un clasificador SVM con $C = 4$ y un kernel RBF de $\gamma = 6$ 100
7.7. Resultados de la clasificación Multimodal 1 con 6 tipos de golpes. Se utilizó un clasificador SVM con $C = 6$ y un kernel RBF de $\gamma = 0{,}4$. 102
7.8. Comparación de los porcentajes de clasificación obtenidos en los modos audio y video, y en la combinación multimodal, utilizando 6 clases. 102
7.9. Comparación del porcentaje de clasificación correcta de los modos audio y video frente al enfoque multimodal para cada tipo de golpe. 103
7.10. Resultado de entrenar con registros de palo rojo de la base eMe y validar con el registro de base la Zavala. En el primer caso se considera el conjunto $S_{\text{multimodal}}$ completo. En el segundo, las características DCT_P1 y $mfcc_1$ se quitaron del conjunto. Ambos resultados se obtuvieron utilizando SVM con los parámetros óptimos determinados en la sección 6.3.1. 104
7.11. Resultados de la clasificación con la base eMe, con 6 tipos de golpes, sacando DCT_P1 y $mfcc_1$, y utilizando SVM. 105

E.1. Parámetros intrínsecos cámara Izquierda. 143
E.2. Parámetros intrínsecos cámara derecha. 143
E.3. Parámetros de calibración estéreo. 145

Índice de figuras

1.1. Ejemplo de procesamiento multimodal para un golpe de palo. Todas las cantidades graficadas están normalizadas. 5
1.2. Imagen estéreo de un intérprete (Sergio Ortuño) durante el registro. 6
1.3. Spectral flux (normalizado) de un golpe rebotado. El máximo de mayor amplitud indica el comienzo del golpe, mientras que los máximos siguientes son causados por los rebotes en la lonja. 8
1.4. Pasos para la detección de la lonja. 9
1.5. Procedimiento de un algoritmo de detección de palo implementado. 9
1.6. Procedimiento de detección de palo verde 10
1.7. Pasos para la detección de piel. 10
1.8. Zona de búsqueda de la mano izquierda y bounding box del contorno más grande. 11
1.9. Diagrama de bloques del método Multimodal 1. El símbolo $\bigoplus$ representa la unión de conjuntos. 13
1.10. Diagrama de bloques del método Multimodal 2. El símbolo $\bigoplus$ representa la unión de conjuntos. 13

2.1. Soporte utilizado para la configuración estéreo de las cámaras. . . 20
2.2. Ubicación de los marcadores en uno de los repiques utilizados en el registro. 20
2.3. Imagen estéreo de un intérprete (Sergio Ortuño) durante el registro, con los marcadores utilizados. 21
2.4. Ubicación de las cámaras y los micrófonos para el rodaje, según lo planificado en las pruebas anteriores al mismo. 23
2.5. Oclusión del marcador en el brazo izquierdo durante una de las tomas del registro. 24

3.1. Spectral flux (normalizado) de un golpe rebotado. El máximo de mayor amplitud indica el comienzo del golpe, mientras que los máximos siguientes son causados por los rebotes en la lonja. 28
3.2. Señal de audio típica de toque de repique 30
3.3. Diagrama de bloques del proceso de cálculo de los MFCCs 34

4.1. Diagrama de bloques de la segmentación de objetos en el video. . . 37
4.2. Características de la elipse 38

4.3. Medidas realizadas para estimar la longitud del eje mayor de la elipse. Las distancias están medidas en pixeles. 40
4.4. Resultado de la detección de la lonja por el primer algoritmo. . . . 41
4.5. Resultado de la detección de la elipse 41
4.6. Concepto de gradiente y level-line - Imagen extraída de [52] 42
4.7. Imagen, campo de orientaciones y las regiones determinadas - Imagen extraída de [52] 43
4.8. Ejemplo de puntos alineados - Imagen extraída de [52] 43
4.9. Resultado de la detección del palo con LSD, primer algoritmo (imágenes con zoom) 44
4.10. Ejemplos donde el primer algoritmo para la detección del palo no fue suficiente (imágenes con zoom) 45
4.11. Resultados intemedios segundo algoritmo 45
4.12. Máscaras utilizadas para la detección del palo verde en el segundo algoritmo para la base eMe (imágenes con zoom) 46
4.13. Comparación de la detección para palo verde con el primer y el segundo algoritmo propuestos (imágenes con zoom) 46
4.14. Ejemplo de cambios de color en el palo debido a la iluminación en la base Zavala (imágenes con zoom) 47
4.15. Esquema funcionamiento algoritmos de extracción de fondo (imagen extraída de [3]) 48
4.16. Máscara de movimiento obtenida a partir del algoritmo *MOG2* (imágenes con zoom) 50
4.17. Diagrama de bloques seguimiento local del palo. 51
4.18. Frame obtenido de la base Zavala y un ejemplo de filtrado en esta base, en el que no se logra distinguir la piel del piso. 52
4.19. Frame extraido de la base Zavala. Problemas relativos a la posición del intérprete respecto a las fuentes de luz y resultado de un filtrado restrictivo en ese caso. 53
4.20. Frame extraido de la base eMe y resultado de aplicar un filtrado poco restrictivo. 54
4.21. Ejemplo del fenómeno de flicker detectado por el estimador de movimiento [92]. 55
4.22. Muestreo de valores Y,U y V para la piel (azul) y el piso (amarillo) 56
4.23. Zona de búsqueda de la mano izquierda. 57
4.24. Detección de piel dentro de la máscara de búsqueda de la mano izquierda y bounding box del contorno más grande. 58
4.25. Dilatación de 4.24(a) y detección del bounding box sobre la imagen resultante. 59
4.26. Bounding box del contorno más grande detectado cuando la segmentación de piel no es buena. 59
4.27. Ejemplo del comportamiento de las detecciones de la mano y la punta del palo para un golpe de palo y otro de mano. 60
4.28. Posiciones relativas a la lonja de las detecciones en la ventana de trabajo para golpes de palo y mano. 63

4.29. Posiciones relativas a la lonja en la ventana de trabajo para golpes de madera, flam, rebotado y borde. 64

4.30. Aproximación polinomial de las detecciones y su derivada para distintos tipos de golpe. 66

4.31. Posición vertical de la punta del palo en un golpe rebotado, aproximación por un polinomio de grado 5 y reconstrucción usando los primeros 10 coeficientes de la DCT. 67

6.1. Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, comparando las distintas selecciones de las características de audio. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 81

6.2. Test de Student con un nivel de significancia de 95 % realizado utilizando tres clasificadores distintos para la selección de audio. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 82

6.3. Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado sobre las selecciones del conjunto geométrico. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 84

6.4. Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado sobre la selección del conjunto DCT. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 85

6.5. Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado sobre la unión de características de video. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 87

6.6. Test de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, realizado utilizando 6 conjuntos de características. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 88
6.7. Test de Student con un nivel de significancia de 95 %, realizado para la selección final de video, utilizando tres clasificadores. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 89
6.8. Diagrama de bloques del método Multimodal 1. El símbolo $\bigoplus$ representa la unión de conjuntos. 90
6.9. Diagrama de bloques del método Multimodal 2. El símbolo $\bigoplus$ representa la unión de conjuntos. 90
6.10. ETest de Student con un nivel de significancia de 95 % y utilizando un árbol de parámetros $C = 0{,}25$ y $m = 2$, comparando las distintas selecciones de las características de Multimodal 2. La línea dentro de cada rectángulo denota la mediana, mientras que los extremos superior e inferior son los percentiles 75 y 25, respectivamente. Las barras verticales denotan los valores máximo y mínimo obtenidos y los puntos que quedan por fuera de éstas son outliers. 93

B.1. Representaciones del espacio YUV - Imágenes extraídas de [22], [26] 124
B.2. Representación de los colores en el plano UV. 124
B.3. Filtro de color implementado en el plano UV. 127
B.4. Ejempo del uso del filtro de color en la base Zavala 128
B.5. Comparación de distintos filtros de color en el espacio UV para segmentar piel en la base eMe (imágenes con zoom) 128

D.1. Ubicación de los marcadores y un ejemplo de los templates utilizados. 134
D.2. Obtención del nuevo template de referencia. 135
D.3. Estimación final de la posición del marcador en el frame actual. . . 136

E.1. Soporte utilizado para la configuración estéreo de las cámaras. . . 140
E.2. Imagen estéreo de un intérprete (Sergio Ortuño) durante el registro de la base *Zavala* con los marcadores utilizados. 141
E.3. Parámetros intrínsecos de una cámara - Imagen extraída de [16]. . 141
E.4. Algunas de las imágenes utilizadas para realizar la calibración de cada cámara 142
E.5. Parámetros de un par estéreo - Imagen extraída de [33]. 143
E.6. Geometría de la configuración utilizada en el par estéreo - Imagen extraída de [58] 145
E.7. Imágenes para calibración estéreo. 146

E.8. Imágenes rectificadas 147
E.9. Correspondencias entre puntos de imagen derecha e izquierda utilizando el algoritmo ASIFT. 148
E.10. Reconstrucción 3D de la escena. 148

F.1. Ejemplo de la salida del visualizador. Los puntos verdes indican la posición de los marcadores, el punto amarillo el estimador de la posición de la mano, el segmento celeste la posición del palo y la elipse roja la posición de la lonja. Superpuesto a la imagen se muestra, en rojo, la piel del intérprete detectada. 155

Esta es la última página.
Compilado el jueves 2 julio, 2015.
http://iie.fing.edu.uy/