



Universidad Nacional del Litoral

**Facultad de Ingeniería y Ciencias Hídricas
Ingeniería en Informática**

Proyecto Final de Carrera

**Método para detección y seguimiento de
objetos con aplicaciones en Realidad
Aumentada**

Autor: Pfarher, Christian Nicolás

Director: Dr. Albornoz, Enrique Marcelo

Co-Director: Dr. Martínez, Cesar

19 de junio de 2013

Resumen

El reconocimiento de patrones es un área de estudio creciente y ha dado lugar a aplicaciones muy relevantes, sin embargo aún queda mucho por explorar. En este trabajo, se presenta un método para la detección de objetos planos en cualquier perspectiva sin emplear marcadores, con procesamiento en tiempo real sobre flujo de video. Luego, se aplica en un prototipo de realidad aumentada.

El método desarrollado utiliza un detector y descriptor de características del estado del arte denominado SURF. Para encontrar la posición del objeto se usa la homografía que se genera mediante la búsqueda de correspondencias de características entre una imagen patrón y el flujo de video, obtenido en tiempo real utilizando una cámara web de una computadora portátil. A los pasos generales propuestos en diferentes trabajos, aquí se le agregan etapas para mejorar la detección aumentando el rendimiento en cuanto a la velocidad y una serie de validaciones para evitar detecciones incorrectas y homografías que produzcan transformaciones defectuosas. También se proponen algunas técnicas sencillas para incrementar los detalles y mejorar la iluminación, a partir de experimentos y análisis de su comportamiento.

Finalmente se presentan dos prototipos: el primero localiza una imagen patrón en la escena, la cual puede presentarse escalada, rotada, y/o en perspectiva, y luego se sobrepone una fotografía en la región donde se detectó el patrón. El segundo prototipo fue desarrollado para una aplicación publicitaria, utilizando como imagen de patrón el packaging de un producto alimenticio y aplicando realidad aumentada para brindar información adicional del mismo.

Prefacio

El reconocimiento de objetos en imágenes ha sido y es un área de investigación que se ha mantenido en desarrollo e investigación constante. La habilidad de reconocer e identificar objetos en una imagen resulta esencial en áreas como vigilancia por video, imágenes médicas, realidad aumentada, etc.

El desarrollo de un motor propio de reconocimiento de objetos con aplicaciones en realidad aumentada, que sea fácilmente adaptable a aplicaciones específicas resulta de gran interés propio y para el grupo. Si bien se puede encontrar software desarrollado, diferentes características de los mismos no lo hacen apropiado para su uso. Además, en el contexto local la realidad aumentada se ha convertido en un tema que constituye un “nicho tecnológico y de mercado” que resulta ampliamente atractivo para su desarrollo, investigación y explotación.

El término de realidad aumentada se usa para definir una visión de un entorno físico del mundo real, cuyos elementos se combinan con elementos virtuales (imágenes, videos, sonidos, etc.), logrando crear una realidad mixta en tiempo real.

En este trabajo se presenta un método para la detección y seguimiento de objetos planos con aplicaciones en realidad aumentada, sin patrones artificiales y en un ambiente controlado. El método desarrollado es aplicado sobre flujo de video en tiempo real capturado con una cámara web estándar. El procedimiento tiene dos etapas principales: en una primera instancia se captura una imagen del objeto patrón, en una segunda etapa se captura flujo de video (imágenes objetivo) en tiempo real, para llevar a cabo la detección

del objeto y el cálculo de su posición en la imagen, para luego “enriquecer la realidad”.

Para llevar a cabo el procedimiento descripto anteriormente, se diseña un método que utiliza un detector rápido de características robustas denominado SURF [5]. El resultado de la detección y descripción de características es usado posteriormente en una etapa de búsqueda de coincidencias entre las características de la imagen patrón y la imagen objetivo. Tras encontrar los potenciales pares coincidentes, se procede a eliminar valores espurios y se calcula la homografía que resulta ser un mapeo perspectivo entre la imagen patrón y la imagen objetivo. Dado que es común que se obtengan falsos positivos al buscar la homografía, se propone un criterio para rechazar transformaciones erróneas provocadas por homografías mal estimadas. Todo este procedimiento es combinado con técnicas heurísticas para lograr un prototipo final optimizado para aplicaciones de realidad aumentada.

En lo que respecta a herramientas utilizadas para la codificación, existe variedad de software y librerías para la manipulación de videos e imágenes. OpenCV (Open Source Computer Vision)¹ es una librería open source² de visión computacional, escrita en C y C++, la cual puede ser ejecutada sobre diferentes sistemas operativos (Linux, Windows y Mac OS X). La misma, ha sido ampliamente adoptada como la herramienta para el desarrollo en la comunidad de investigadores y desarrolladores en el campo de visión computacional [33], debido a que fue diseñada para lograr gran eficiencia computacional, haciendo foco en el desarrollo de aplicaciones en tiempo real. Es por ello que, como base para el desarrollo del código fuente de nuestro método, se utiliza la citada librería.

La tesis se encuentra organizada en cinco capítulos, como se explica a continuación.

En el Capítulo 1, se expone la motivación central del desarrollo de este proyecto, seguido de una reseña del estado del arte del reconocimiento de objetos y algunas de sus aplicaciones, haciendo hincapié en realidad aumentada. Posteriormente, se describen los objetivos generales y específicos, seguidos del los alcances del presente trabajo.

En el Capítulo 2, se tratan conceptos y métodos del marco teórico del procesamiento de imágenes, haciendo énfasis en la detección de características invariantes a escala y rotación.

El Capítulo 3 presenta el diseño del método propuesto basado en los fundamentos teóricos del capítulo anterior y utilizando el método de detección

¹<http://SourceForge.net/projects/opencvllibrary>

²<http://opensource.org>

de características propuesto allí. También, se describen detalladamente las etapas del método propuesto, planteando mejoras y optimizaciones para las mismas con el fin de obtener un mejor desempeño en el tiempo de procesamiento, como así también, en la correcta detección del objeto en la escena.

En el Capítulo 4, se detallan los experimentos y los resultados obtenidos, donde se consideran variaciones de la iluminación y diferentes técnicas de realce de detalles sobre la imagen. Luego, se presentan dos experimentos y se propone un prototipo publicitario como resultado final del método desarrollado.

Finalmente, en el Capítulo 5, se exponen las conclusiones finales y los desarrollos futuros para corto, mediano y largo plazo.

Christian Nicolás Pfarher
Santa Fe, Argentina.
19 de junio de 2013

Índice general

Resumen	III
Prefacio	v
Índice de figuras	xiii
1. Introducción	1
1.1. Motivación	1
1.2. Estado del arte	4
1.2.1. Conceptos en realidad aumentada	4
1.2.2. Puntos característicos y descriptores	6
1.2.3. Detección y correspondencia de puntos de interés	10
1.2.4. Búsqueda de correspondencias	12
1.2.5. Transformación proyectiva	12
1.3. Objetivos del Proyecto Final	13
1.3.1. Objetivos generales	13
1.3.2. Objetivos específicos	13
1.4. Alcances del Proyecto Final	14
2. Fundamentos teóricos	15
2.1. Operaciones morfológicas en imágenes	15

2.1.1. Dilatación	16
2.1.2. Erosión	16
2.2. Realce de imágenes en el dominio espacial	18
2.2.1. Umbral	18
2.2.2. Transformación logarítmica	19
2.2.3. Ecualización de histograma	19
2.2.4. Filtrado pasa altos	20
2.2.5. Filtrado de alta potencia	21
2.3. Detección de puntos claves	21
2.3.1. El problema de cambio de escala	22
2.3.2. Puntos claves basados en la matriz hessiana	24
2.3.3. Determinación de la localización de puntos claves	27
2.4. Descripción de puntos claves	29
2.4.1. Asignación de la orientación	29
2.4.2. Creación del descriptor	30
2.5. Correspondencia entre puntos claves	33
2.5.1. El algoritmo de árboles KD aleatorio	34
2.5.2. Remoción de correspondencias no válidas	35
2.6. Formación de la imagen y transformación proyectiva	36
2.6.1. Introducción	36
2.6.2. El modelo de cámara oscura	36
2.6.3. Transformación proyectiva y estimación de la homografía	40
3. Método propuesto	49
3.1. Diseño del método	49
3.2. Conversión a escala de grises	52
3.3. Mejoras en la iluminación y realce de detalles	53
3.4. Detección de la región de interés	53
3.4.1. Área del rectángulo delimitador mínimo	57
3.5. Extracción y descripción de características	58
3.6. Correspondencia entre puntos claves	59
3.7. Detección de homografía	61
3.8. Detección de homografías mal estimadas	62

3.9. Condición de presencia previa	63
3.10. Realidad aumentada en el flujo de video	66
3.10.1. Transformación Perspectiva	66
4. Experimentos y Resultados	69
4.1. Definición de imágenes	69
4.2. Experimentos	71
4.2.1. Experimento 1	72
4.2.2. Experimento 2	80
4.3. Implementación de prototipos	83
5. Conclusiones y trabajos futuros	87
5.1. Conclusiones	87
5.2. Trabajos futuros	88
Bibliografía	91

Índice de figuras

1.1.	Diagrama continuo de Realidad-Virtualidad	5
2.1.	Efecto de la operación de erosión y dilatación sobre una imagen	17
2.2.	Umbral binario	19
2.3.	Curva de la transformación logarítmica	19
2.4.	Fotografía tomada a diferentes escalas para la misma escena .	24
2.5.	Pirámide de escala de imágenes para SIFT y SURF	26
2.6.	Derivadas parciales gaussianas discretas y aproximadas	26
2.7.	Longitudes de los filtros para 3 diferentes octavas	27
2.8.	Representación de una octava compuesta por tres niveles de escala	28
2.9.	Suma de intensidades usando imágenes integrales	29
2.10.	Filtros Wavelets Haar	30
2.11.	Asignación de la orientación para un punto clave	30
2.12.	Ejemplo de ventanas del descriptor a diferentes escalas sobre una imagen	31
2.13.	Interpretación gráfica del descriptor SURF	32
2.14.	Descriptores resultantes para tres subregiones con patrones diferentes	32
2.15.	Modelo de cámara oscura y su interpretación geométrica . .	38
2.16.	Transformación entre diferentes espacios de coordenadas . .	40

2.17. Mapeo de puntos de un plano a otro	42
2.18. Estimación robusta de una línea	44
3.1. Diagrama de flujo de la etapa de configuración	50
3.2. Diagrama de flujo de la etapa de ejecución	51
3.3. Detalle del proceso de detección de movimiento	54
3.4. Detección de movimiento	55
3.5. Ejemplo de extracción de características en una imagen	60
3.6. Correspondencias de puntos entre imágenes patrón y objetivo	61
3.7. Polígonos resultantes de la transformación con la matriz de homografía	64
3.8. Esquema de restauración de la última transformación válida .	65
3.9. Esquema de transformación perspectiva	67
4.1. Esquema del ambiente en el que se realizaron las pruebas . . .	71
4.2. Imágenes obtenidas para condiciones de iluminación diferentes	72
4.3. Resultados de aplicar las técnicas para la mejora en iluminación y detalles en condición B_N	74
4.4. Resultados de aplicar las técnicas para la mejora en iluminación y detalles en condición B_H	75
4.5. Resultados de aplicar las técnicas para la mejora en iluminación y detalles en condición B_L	76
4.6. Cantidad de puntos detectados para la imagen con luminosidad normal, alta y baja	78
4.7. Tiempos promedios de operación para Prueba 1	82
4.8. Capturas de imágenes con enriquecimiento de la realidad . . .	84
4.9. Capturas en la etapa de ejecución para el prototipo publicitario	86

CAPÍTULO 1

Introducción

En este capítulo se presenta la motivación del desarrollo de este proyecto, seguido de una descripción del estado del arte del reconocimiento de objetos y algunas de sus aplicaciones en realidad aumentada. Tras ello, se exponen los objetivos generales y específicos de esta tesis, como así también los alcances de la misma.

1.1 Motivación

Las imágenes y videos presentes en el mundo digital, combinados con la potencialidad que ofrecen los ordenadores actuales, han hecho más fácil la creación de aplicaciones sofisticadas en el área de procesamiento de imágenes y visión computacional [33].

Los seres humanos reconocemos fácilmente gran cantidad de objetos, independientemente del punto de vista en que se mire, las condiciones de iluminación, su tamaño, escala o incluso si se encuentran rotados, trasladados u obstruidos. Sin embargo, esta tarea no resulta trivial en el área de visión computacional. Así, el reconocimiento de objetos se puede resumir en el objetivo de buscar un objeto particular dado en una imagen o un video (secuencia de imágenes).

La tarea de buscar puntos correspondientes entre dos imágenes de una

misma escena u objeto es usada en una gran variedad de sistemas de visión computacional. La búsqueda de estos puntos, se realiza mediante detectores de puntos claves cuyo desafío es poder encontrar en una imagen puntos que sean fácilmente identificables y a la vez robustos a diferentes transformaciones. Así, estos detectores se convierten en la base de gran cantidad de herramientas basadas en visión computacional como ser el reconocimiento de objetos, vigilancia por video, imágenes médicas, la realidad aumentada y la restauración de imágenes entre otros.

Un sistema de realidad aumentada (RA) reemplaza parte del mundo real con objetos virtuales (generados por computadoras), los cuales parecen coexistir en el mismo espacio que el ambiente real cuando se lo es visto a través de un dispositivo de visualización. Si bien existen ampliaciones a esta definición, se puede definir a un sistema de realidad aumentada como aquel que posee las siguientes propiedades [3]:

- combina objetos virtuales y reales en un mismo ambiente,
- trabaja interactivamente y en tiempo real y
- detecta y realiza un “seguimiento” de objetos reales y virtuales entre sí.

Desde hace tiempo, los sistemas de realidad aumentada (del inglés, Augmented Reality) y reconocimiento de objetos, han pasado a ser un tema pujante en el área de visión computacional, tomando gran interés en la comunidad científica y viéndose esto reflejado en diversidad de aplicaciones: en el área del e-commerce - publicidad [40], robótica [14], juegos [3, 29], libros interactivos [30], industria [8]; en diferentes ambientes [31, 53, 63], aplicable sobre diferentes objetos [35, 36, 39] y llevada a cabo mediante el uso de distintos dispositivos [66].

La habilidad de reconocer e identificar objetos como imágenes, texto, logos, etc. en una imagen o video para luego ser usados como “patrones”, es esencial en aplicaciones de RA, además de ser usada en control de calidad, vigilancia e identificación visual [34].

Existe una distinción en RA entre dos tipos de sistemas [11]:

- los que usan marcadores (**Marker based**), en los cuales se identifica un “marcador artificial” colocado en el ambiente, y por ende alterando el mismo y,
- los que no usan marcadores (**Markerless**), que se sustentan en usar características naturales para identificar objetos del mundo real, es de-

cir, que no usan un “marcador artificial” para asistir al reconocimiento del objeto.

En ambos casos, el objeto 2D o 3D virtual da la apariencia de aparecer “pegado” al marcador o a la característica natural cuando se ve a través del área de visualización de RA.

De la misma forma para la detección y seguimiento del objetos, existen diferentes métodos [11], entre los cuales se pueden nombrar:

- seguimiento basado en localización: involucra sensores como GPS, acelerómetros, giroscopios, etc. para identificar el objeto y su posición (por lo general menos precisos).
- seguimiento óptico: se basa en el análisis e identificación de características a partir de la imagen de forma tal que permitan identificar y obtener la posición del objeto.
- una combinación de los dos anteriores.

En este trabajo nos enmarcaremos en sistemas de RA sin marcadores y con detección y seguimiento de objetos que realicen un procesamiento de las características de la imagen, es decir, el método: seguimiento óptico.

Así, se propone construir un método con las características anteriormente nombradas, que permita reconocer objetos planos e identificar su posición en la imagen, en un ambiente controlado, para la posterior aplicación de realidad aumentada.

El desarrollo de un motor propio de RA que sea fácilmente adaptable a aplicaciones específicas (comerciales, educativas, lúdicas, etc.) resulta de gran interés propio y para el grupo del *SINC*, ya que si bien existen software similares desarrollados, no todos trabajan en tiempo real, otros son distribuidos bajo licencias privativas o utilizan marcadores artificiales, entre otras características que no lo hacen de interés nuestro. Además, se trabaja en una tecnología de punta que se encuentra en auge en estos tiempos y que presenta un futuro prospero. Una de las piezas fundamentales del motor citado, son los métodos de identificación y reconocimiento de objetos los cuales serán expuestos en el presente trabajo. Para ello, será necesario aplicar conocimientos de visión computacional [9, 26, 33] e investigar métodos de extracción de características en imágenes [56]. Existen gran cantidad de investigaciones en el área, tales como:

- el estudio de descriptores visuales [20] y locales [32, 48, 64], entre los cuales se encuentran el detector rápido de características robustas

SURF (del inglés, Speeded-Up Robust Features) [5] y la transformación de características invariante a la escala SIFT (del inglés, Scale Invariant Feature Transform) [43], etc.

- el reconocimiento de puntos claves [57],
- la detección en tiempo real de objetos [25],
- el seguimiento sin marcadores [13],
- el análisis de características invariantes en imágenes [44, 62] y
- la clasificación de puntos característicos [38], entre otros.

Dichos temas, se tratan de abordar en menor o mayor medida para el correcto desarrollo del presente proyecto.

1.2 Estado del arte

La realidad aumentada es una tecnología que utiliza la generación por computadora de objetos virtuales, para mezclarlos en una secuencia de imágenes reales. De modo diferente al concepto de Realidad Virtual, un sistema de RA permite al usuario ver el ambiente real con los objetos virtuales inter-actuando en él.

Cuando se tienen imágenes de una misma escena, las correspondencias 2D pueden ser extraídas en cada fotograma para estimar la posición del objeto buscado mediante la identificación de puntos característicos naturales (cuando se trata de RA sin marcadores) o marcadores artificiales que actúan como patrones alterando la naturalidad de la escena mediante una imagen artificial.

1.2.1 Conceptos en realidad aumentada

La RA permite enriquecer la perspectiva sobre imponiendo objetos virtuales (2D o 3D) en el mundo real, con el objetivo de lograr persuadir al observador y haciéndole creer que el objeto virtual forma parte del ambiente real. De esta forma, se puede interpretar la RA, como una mezcla entre un



Figura 1.1: Diagrama continuo de Realidad-Virtualidad (Figura adaptada de [52]).

mundo real y uno virtual tal como se visualiza en el Diagrama continuo de Realidad-Virtualidad propuesto en [52] y mostrado en la Fig. 1.1.

Algunos de los conceptos que se manejan en realidad aumentada son los siguientes:

Vista real Se refiere a la secuencia o flujo de video producido por la cámara.

La aplicación de RA captura imágenes desde esta secuencia de video, aumentándolas con objetos virtuales para crear una “vista aumentada”. [11]

Identificación y seguimiento (registration & tracking) Describe los métodos disponibles para alinear un objeto virtual con un sistema de coordenadas en tres dimensiones en una vista real. En el seguimiento, se pueden distinguir entre diferentes métodos: *seguimiento basado en localización* y *seguimiento óptico* o una combinación de ambos [11, 35].

Punto de interés (POI) Es un elemento de información asociado con una ubicación geográfica (longitud, latitud, altura) o un patrón visual (marcador, tapa de libro (sin marcadores), etc.) que se puede representar de alguna forma por la aplicación de RA [3, 11].

Objeto virtual Es algún tipo de contenido digital que es dibujado por la aplicación de RA y es superpuesto en la vista real. Por lo general, incluye modelos 3D, imágenes 2D, iconos, textos, o incluso elementos no visuales como audio, entre otros [35], [11].

RA basadas/no basadas en marcadores Cuando es utilizado el reconocimiento de imágenes para alinear un objeto virtual (seguimiento óptico), hay una distinción entre sistemas: los que identifican un *marcador artificial* como un código matricial 2D colocado en un lugar determinado o sobre algún objeto del mundo real; los que utilizan la *detección de características naturales de los objetos*, para identificar objetos del

mundo real inalterados (como tapas de libros, posters, etc.), los cuales no poseen marcas artificiales que asistan en el reconocimiento del objeto. En ambos casos, el objeto virtual 2D o 3D, aparece “pegado” al marcador o característica natural cuando se ve a través de la ventana gráfica de RA [35], [11].

Reconocimiento basados en localización Se refiere al reconocimiento basado en información de geo-localización, obtenida a partir de dispositivos o sensores de localización. El término, es usado para hacer una distinción entre sistemas que usan sensores de localización, en contraste con los que hacen seguimiento óptico del objeto usando técnicas de reconocimiento de imagen. El reconocimiento basado en localización, es aplicado sobre todo en ambientes exteriores [3, 11].

Seis grados de libertad (del inglés, Six Degrees of Freedom -6DoF)
Se refiere a la capacidad del sistema de seguimiento para mantener la alineación de un objeto del mundo real en un espacio tridimensional. Esto es, la capacidad de moverse hacia delante/atrás, arriba/abajo, izquierda/derecha (traslación en tres ejes perpendiculares), combinados con la rotación sobre tres ejes perpendiculares.

Reconstruir la posición de la cámara con 6DoF es el objetivo más importante en RA, ya que de esta manera se determina la posición de la cámara y, en consecuencia se obtiene la posibilidad de poder dibujar los objetos virtuales en una perspectiva correcta [3, 11, 35].

1.2.2 Puntos característicos y descriptores

La detección de puntos característicos y los descriptores de imágenes, son una característica ampliamente usada en problemas de reconocimiento de objetos, detección de imágenes, seguimiento visual, reconstrucción 3D entre otras aplicaciones. El objetivo es seleccionar ciertos puntos claves en una imagen, de forma que estos permitan distinguirla de otras sin la necesidad de tener que analizar la totalidad de la imagen. Esta aproximación funciona correctamente en la medida que se detecten la cantidad suficiente de puntos de interés que sean “distinguibles” y, a su vez, estables para que puedan ser nuevamente localizados en otra escena.

Características locales

Una característica local puede ser interpretada como un patrón de imagen que difiere de su vecindad y puede venir representada por ejemplo por un punto, un borde o pequeños trozos de la imagen. Es común la realización de ciertos cálculos sobre una región alrededor de la característica local detectada y su resultado se convierte en un vector numérico el cual recibe el nombre de descriptor. Estos últimos pueden ser utilizados para variedad de aplicaciones.

Se puede hacer una distinción de los detectores de características, dependiendo del uso que se le de a los mismos:

- *Características locales específicas que otorgan una interpretación semántica de un contexto limitado específico*, por ejemplo: de bordes detectados en imágenes aéreas, se puede decir que comúnmente corresponden a caminos o rutas.
- *Características locales que proveen un conjunto limitado de puntos bien localizados e individualmente identificables*. En este caso, lo que representa la característica no es realmente relevante. Lo importante es que su ubicación pueda ser determinada con exactitud y de manera estable en el tiempo, por ejemplo situaciones de correspondencia de puntos y seguimiento de los mismos, calibración de la cámara y reconstrucción 3D.
- *Características locales que pueden ser usadas como una representación robusta de la imagen permitiendo reconocer objetos o escenas sin la necesidad de segmentación*. Aquí no es relevante lo que la característica represente, ni tampoco es necesario que pueda ser precisamente localizada, sino que lo que importa es un análisis del tipo estadístico. Aplicaciones como la clasificación de escenas y análisis de texturas se encuentran dentro de esta categorización.

Los tres escenarios aquí planteados no resultan ser los únicos posibles, pero se exponen con el objetivo de enfatizar las diferentes propiedades requeridas para diversas situaciones. Claramente cada uno de los escenarios, impone restricciones sobre los puntos característicos a utilizar. En este trabajo, nos enfocaremos en el segundo (correspondencia de puntos y estabilidad de los mismos) y tercer caso.

Propiedades de características locales

Las características locales comúnmente vienen representadas por una extensión espacial, por ejemplo una vecindad de píxeles de un punto de interés dado. Los detectores seleccionan características locales basadas en la intensidad de los patrones subyacentes.

Las características son consideradas útiles cuando poseen las siguientes propiedades:

- **Repetibilidad:** dadas dos imágenes tomadas bajo diferentes condiciones del mismo objeto o escena, un alto porcentaje de las características detectadas debe poder ser encontrada en ambas imágenes.
- **Distintividad:** los patrones de intensidad subyacentes a las características detectadas deberían mostrar variaciones significativas, de forma tal que las características sean distinguibles y se puedan encontrar posteriormente pares de correspondencias.
- **Localización:** Las características deberían ser locales para reducir la posibilidad de oclusión y permitir un modelo de aproximación a las deformaciones geométricas y fotométricas entre dos imágenes tomadas bajo diferentes condiciones visuales.
- **Cantidad:** El número de características detectadas debe ser lo suficientemente grande para que aún en objetos pequeños, una cantidad razonable de éstos puedan ser detectada. Esta cantidad depende de la aplicación y, por lo general, puede ser adaptada mediante un umbral. La densidad de las características debe reflejar el contenido de información de la imagen de forma de proveer una representación compacta de la misma.
- **Precisión:** Las características detectadas deberían ser precisamente localizadas respecto a la posición en la imagen, a la escala y en lo posible a la forma.
- **Eficiencia:** la detección de características en una imagen nueva, debería poseer tiempos lo más acotados posibles para una aplicación.

La Repetibilidad es comúnmente una de las propiedades de mayor importancia y se puede lograr alcanzando dos propiedades, invariancia y robustez:

- **Invariancia:** debido a que se esperan grandes deformaciones, la idea consiste en modelar matemáticamente las mismas de forma de tener

un método de detección de características que no se vea afectado por dichas transformaciones matemáticas.

- **Robustez:** en el caso de deformaciones relativamente pequeñas, se deben usar métodos de detección menos sensitivos a dichas deformaciones, por ejemplo, la precisión de la detección puede decrecer pero no debería hacerlo de forma drástica. Algunas deformaciones típicas atacadas en estos casos son: el ruido en la imagen, efectos de discretización, artefactos de compresión, borroneado, etc. También, las desviaciones geométricas y fotométricas del modelo matemático utilizado para obtener invariancia, comúnmente se superan mediante la inclusión de mayor robustez.

La importancia de las propiedades mencionadas, dependen de la aplicación sobre la cual se trabajará. El ajuste de éstas, conllevan un compromiso y se fortalecerán unas u otras dependiendo de cada caso en particular.

La *localización* y *distintividad* son propiedades interdependientes que no pueden ser maximizadas simultáneamente: cuanto más local es una característica, menos información está disponible sobre el patrón de intensidad subyacente y se hace más difícil de encontrar la correspondencia correcta (*distintividad*). Por otro lado, en el caso de objetos planos o rotación de cámaras, las imágenes están relacionadas por la homografía global, y no existen problemas con las occlusiones o discontinuidad de profundidades. Bajo estas condiciones, la cantidad de características locales puede ser incrementada sin problemas, resultando ser altamente distintivas, pero impactando negativamente en la *eficiencia*.

Similarmente, al incrementar el nivel de *invariancia*, se reduce la *distintividad*. Un análisis similar se puede hacer entre *robustez* y *distintividad*, típicamente cierta información (considerada ruido) es descartada con el objetivo de lograr *robustez*. Como resultado, es importante tener una idea clara del grado requerido de invariancia o robustez para una aplicación específica. Es difícil alcanzar alta invariancia y robustez al mismo tiempo.

La *precisión* es especialmente importante cuando se precisan usar correspondencias, por ejemplo, para estimar la geometría epipolar o calibrar una cámara.

La *cantidad* es particularmente útil en métodos de reconocimiento de objetos o escenas, donde es importante cubrir densamente el objeto de interés. Sin embargo y como se ha mencionado, una gran cantidad de características impacta negativamente en el tiempo de computo.

1.2.3 Detección y correspondencia de puntos de interés

En visión computacional, los conceptos de puntos de interés (puntos claves) y puntos característicos, como la correspondencia entre los mismos resultan ampliamente usados en diversas tareas. La idea consiste en seleccionar algunos puntos especiales de la imagen (una cantidad suficiente) que sean distinguibles, estables, posean repetibilidad y puedan localizarse como se describió en la Sec. 1.2.2.

La búsqueda de correspondencia de puntos discretos en imágenes, puede ser dividida en tres grandes pasos:

- Primeramente, los puntos de interés (esquinas, uniones de tipo T, etc.) son seleccionados como características distintivas de la imagen. La propiedad más sobresaliente de un detector de puntos de interés es su repetibilidad que expresa la robustez del detector en buscar los mismos puntos de interés bajo diferentes condiciones de visualización.
- Luego, se representa la vecindad de cada punto de interés detectado mediante un vector de características (vector descriptor). Idealmente, este descriptor debe ser distintivo, robusto al ruido, a la detección de desplazamientos y a las deformaciones geométricas o fotométricas de la imagen.
- Finalmente, se buscan las correspondencias entre los vectores descriptors de las imágenes. Dicha correspondencia, generalmente se encuentra basada en la distancia entre los vectores (por ejemplo: la euclídea). La dimensión del descriptor impacta directamente en el tiempo de cálculo. Así, buscar correspondencias entre vectores de bajas dimensiones resulta más rápido, pero a su vez los vectores se hacen menos distintivos. Como contraparte vectores de altas dimensiones involucran altos tiempos de procesamiento logrando mayor distintividad.

Cuando se trabaja con características locales uno de los primeros inconvenientes a resolver es obtener un alto nivel de invarianza. Claramente, esto depende de las deformaciones geométricas y fotométricas que pueda sufrir la imagen. En nuestro caso, nos centraremos en los cambios de escala y la rotación en el plano, asumiendo que los efectos de segundo orden: inclinación, perspectiva y anisotropía son cubiertos en cierto grado por la robustez global del descriptor utilizado. En cuanto a las deformaciones fotométricas, se asume un modelo lineal simple con un desplazamiento y un cambio de contraste (factor de escala).

En este trabajo, se tratará con características locales, las mismas describen una parte de la imagen; en contraposición con las características globales que describen la imagen como un todo.

Existen diferentes tipos de características locales: aquellas que se basan en líneas o curvas (detectadas por ejemplo con la transformada de Hough [15]) y otras como las que utilizaremos en este trabajo que se valen de puntos característicos, como esquinas, bordes, etc. Por ejemplo, para obtener esquinas, uno de los más usados es el detector de esquinas de Harris [9, 22], el cual está basado en los eigenvalores de la matriz de segundo orden para determinarlas, aunque no resulta invariante a escala. Otros detectores de características como el detector de regiones extremas más estables [46] y el de características por test de segmento acelerado [33, 59] se encuentra también en gran parte de la bibliografía.

El concepto de selección de escala automática [41] permitió detectar puntos asignándole una escala a cada uno de ellos. Para la detección de los mismos, se experimentó tanto con el determinante de la matriz hessiana como con el laplaciano (traza de la matriz hessiana). Otros autores [49] refinaron este método, creando un detector de características robusto e invariante a escala, logrando una alta repetibilidad. De esta manera, surgió el detector *Harris-Laplace* y *Hessian-Laplace*. Para el primero de ellos, se utiliza una medida de harris para seleccionar la ubicación de la característica y el laplaciano para seleccionar la escala. Para el caso del detector *Hessian-Laplace* se utiliza el determinante de la matriz hessiana para seleccionar la ubicación de la característica y el laplaciano para seleccionar la escala.

Enfocado en la velocidad, Lowe [45] propuso aproximar el laplaciano del gaussiano (LoG) mediante filtros diferencia de gaussianos (DoG). Otros detectores de puntos de interés invariantes a escala fueron propuestos, el detector de regiones salientes [28] que utiliza la maximización de la entropía en la región y el detector de regiones basado en bordes [27] que detecta regiones locales convexas sobre contornos de imagen, entre otros.

Del estudio de los detectores existentes y de las comparaciones en diferentes publicaciones [48, 50], se ha concluido que los detectores basados en el hessiano son los más estables y poseen mayor repetibilidad que los basados en esquinas de harris. Incluso, se han obtenido mejores resultados mediante el uso del determinante de la matriz hessiana en lugar de la traza de la misma [50]. También, se ha observado que las aproximaciones como el DoG pueden aumentar la velocidad a un bajo costo en términos de precisión. Así, surgió SIFT [44] que calcula un histograma de los gradientes orientados localmente alrededor del punto de interés y genera un vector característico de 128-dimensiones. Luego, se propusieron varios refinamientos a este esque-

ma básico como el análisis de componentes principales - transformación de características invariante a la escala (PCA-SIFT: del inglés, Principal Components Analysis - Scale Invariant Feature Transform) [9], [48] que construye un descriptor de 36 dimensiones. De esta forma es más rápido el proceso de buscar coincidencias, aunque pierde distintividad y velocidad en el cálculo de características respecto a SIFT [51]. De esta manera, el método SIFT se consolidó como el más adecuado para usos prácticos, resultando distintivo y relativamente rápido lo cual resulta crucial en aplicaciones “on-line”.

Más tarde y basados en las ideas de SIFT, se propuso un nuevo detector y descriptor de características denominado SURF [5, 6]. El mismo, se basa en ideas del SIFT e introduce algunas modificaciones respecto al uso de imágenes integrales, filtros caja y un método de indexación rápida para búsqueda de correspondencias, que permite generar un descriptor de 64 dimensiones que se calcula y compara más rápidamente, manteniendo una precisión y distintividad valorable.

1.2.4 Búsqueda de correspondencias

Existe una extensa variedad de publicaciones [2, 7, 42, 54] que abordan la búsqueda de correspondencias mediante el algoritmo de búsqueda del vecino más cercano, en los que su rendimiento varía dependiendo de las propiedades del conjunto de datos, tales como: la dimensionalidad, correlación, características de agrupación y tamaño. Uno de los algoritmos utilizados es KD-tree [7, 18] (básicamente se trata de un árbol binario balanceado de búsqueda), el cual ha dado buenos resultados para la búsqueda exacta en datos de bajas dimensiones. Este algoritmo, ha sido modificado en diversos trabajos [1, 2, 7, 10, 19, 37, 42, 47, 55] para lograr reducir los tiempos de ejecución con datos de grandes dimensiones, a costa de obtener resultados aproximados. En el trabajo de Slipa-Anan y Hartley [1, 24], se propuso el uso de múltiples árboles KD aleatorios conocidos por su término en inglés como **Randomized KD-Tree**, que obtiene resultados satisfactorios en un amplio rango de problemas [54] reduciendo los tiempos de cálculo.

1.2.5 Transformación proyectiva

Las imágenes son producidas usando una cámara, que captura la escena mediante la proyección de luz que pasa por la lente e impacta en un sensor. El hecho de que una imagen se forma a través de la proyección de una escena

3D en un plano 2D, impone la existencia de una importante relación entre la escena, la imagen y diferentes imágenes de la misma escena. La geometría proyectiva es la herramienta usada para describir y caracterizar, en términos matemáticos, el proceso de formación de la imagen.

Algunos de los conceptos fundamentales en esta temática involucran temas como: el modelo pin-hole de la cámara [9, 33], la calibración de la cámara que arroja como resultado los parámetros de distorsión y la matriz que la caracteriza [3, 9, 33], la correspondencia de imágenes [9, 33], el cómputo de la homografía para estimar la posición del objeto [9, 14, 33], entre otros.

1.3 Objetivos del Proyecto Final

Una vez introducida la tarea de interés y el marco general de trabajo, se enuncian a continuación los objetivos generales y particulares del presente Proyecto Final.

1.3.1 Objetivos generales

- Desarrollar un método para reconocer y seguir objetos en una secuencia de video digital y desarrollar un prototipo que haga uso del mismo aplicándolo a realidad aumentada.
- Afianzar y extender los conocimientos adquiridos en el cursado de la carrera Ingeniería en Informática.

1.3.2 Objetivos específicos

- Realizar el relevamiento del estado del arte en métodos utilizados para la detección y seguimiento de objetos en el Procesamiento Digital de imágenes.
- Diseñar y desarrollar un método reconocedor y seguidor de objetos planos en el flujo de video tomado por una cámara web estándar, sobre un ambiente controlado.

- Implementar el método en un algoritmo computacional que pueda ser utilizado en diferentes sistemas operativos.
- Optimizar el procesamiento llevado a cabo para aplicarlo en tiempo real.
- Implementar una aplicación prototipo específica (en el área de turismo, educación, publicidad, juegos u otros) aplicando el método desarrollado.

1.4 Alcances del Proyecto Final

En este proyecto se presenta un método para la detección de objetos planos en la escena y determinando su ubicación se superpone un objeto virtual para lograr “enriquecer la realidad”.

Entre los alcances del proyecto se pueden enumerar los siguientes:

- El trabajo se enmarcará en un sistema del tipo sin marcadores con método de reconocimiento del tipo seguimiento óptico.
- El proyecto involucra el desarrollo de un prototipo para ser utilizado en una computadora con una cámara web estándar. Cabe aclarar, que no se pretende realizar una aplicación final específica y completa (estudio y diseño de interfaz amigable al usuario, introducción amigable de parámetros, etc.) orientada al uso de un usuario final.
- Aunque no es un objetivo la implementación de varios métodos y su comparación, en una etapa previa se revisarán las características de algunos métodos para poder seleccionar alguno que se adecue a los requerimientos.

CAPÍTULO 2

Fundamentos teóricos

En este capítulo, se presentan los fundamentos teóricos sobre los cuales se sustenta el proyecto y particularmente el método luego utilizado. Primero se describen algunas operaciones morfológicas sobre imágenes y algunas técnicas de realce sobre las mismas. Tras ello, se explica un método de detección y descripción de características en imágenes. También, se presenta una técnica para búsqueda de correspondencias entre puntos característicos de imágenes, que sirve como base para encontrar la relación entre dos imágenes de un mismo objeto fotografiado desde diferentes perspectivas.

2.1 Operaciones morfológicas en imágenes

El procesamiento morfológico en imágenes es un tipo de procesamiento en el cual la forma espacial o estructura de los objetos en la imagen es modificada. La erosión y dilatación son dos operaciones morfológicas fundamentales que pueden ser aplicadas a imágenes en escala de grises o imágenes binarias. En esta sección se describen tanto la erosión como la dilatación enfocándose en imágenes binarias que es el tipo utilizado en este trabajo.

El lenguaje de la morfología matemática es el de la teoría de conjuntos y los mismos representan objetos en las imágenes. Por ejemplo, el conjunto de todos los píxeles negros en una imagen binaria es una descripción morfológica

completa de la imagen. En imágenes binarias, el conjunto en cuestión es miembro de un espacio entero 2-D Z^2 , donde cada elemento del conjunto es una tupla (vector 2-D) cuyas coordenadas son las coordenadas (x, y) de un píxel negro (o blanco, dependiendo de la convención) de la imagen.

A continuación, definiremos algunas notaciones que serán utilizadas para definir posteriormente la erosión y la dilatación: nos referiremos con A a un conjunto en Z^2 , $a = (a_1, a_2)$ un elemento de A , \emptyset un conjunto vacío, $A \subseteq B$ para denotar que A es un subconjunto de B , $A \cap B$ para indicar la intersección entre A y B (el conjunto de elementos que corresponde tanto a A como a B), $\hat{B} = \{w|w = -b, \text{ para } b \in B\}$ para indicar la reflexión del conjunto B , $(A)_z = \{c|c = a + z, \text{ para } a \in A\}$ para indicar la traslación del conjunto A por $z = (z_1, z_2)$ y $\{\cdot\}$ la notación de un conjunto.

2.1.1 Dilatación

Sea A y B un conjunto en Z^2 , la dilatación entre A y B denotada como $A \oplus B$ queda definida como

$$A \oplus B = \{z|(\hat{B})_z \cap A \neq \emptyset\}, \quad (2.1)$$

donde B comúnmente es conocido como máscara o kernel.

Mediante la dilatación, los objetos crecen en su tamaño y algunos de los “espacios” dentro de ellos son rellenados. Un ejemplo de dilatación sobre la imagen de la Fig. 2.1a utilizando el kernel de la Fig. 2.1b, se puede observar en la Fig. 2.1c (se realizaron 2 dilataciones para resaltar el efecto de la operación).

2.1.2 Erosión

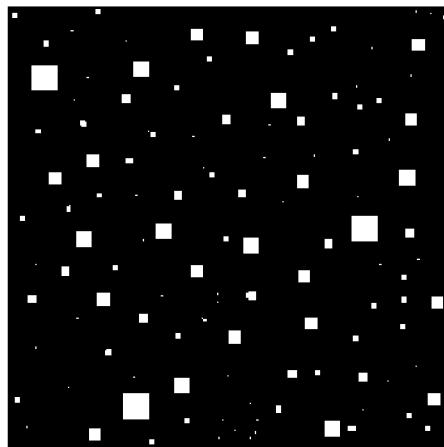
Sea un conjunto A y B de Z^2 , la erosión de A por B , denotada como $A \ominus B$ queda definida como

$$A \ominus B = \{z|(B)_z \subseteq A\}. \quad (2.2)$$

De la ecuación (2.2) se puede interpretar que la erosión de A por B es el conjunto de todos los puntos z tales que B , trasladados por z , están contenidos en A .

Uno de los usos más simples de la erosión es para eliminar detalles irrelevantes (en términos de tamaño) de una imagen binaria. En una imagen

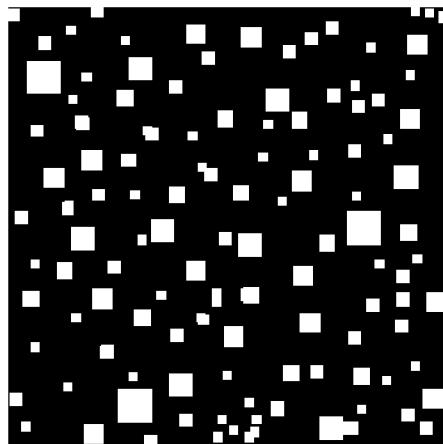
erosionada, el tamaño de los objetos se ve reducido y el ruido o detalles irrelevantes (aislados) es eliminado. Un ejemplo de erosión sobre la imagen de la Fig. 2.1a utilizando el kernel de la Fig. 2.1b, se puede observar en la Fig. 2.1d (se realizaron 2 erosiones para resaltar el efecto de la operación).



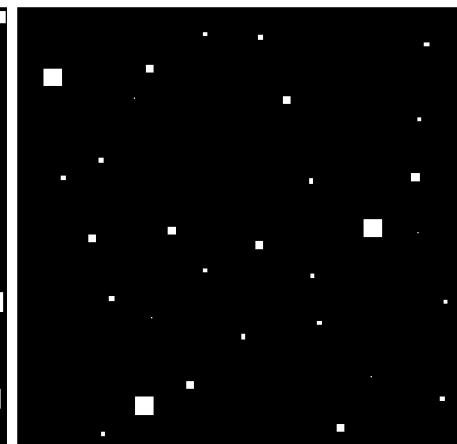
(a) Imagen sobre la que se realizan las operaciones (Figura tomada de [21]).

B		
1	1	1
1	1	1
1	1	1

(b) Kernel o máscara de 3×3 .



(c) Dilatación ($\times 2$).



(d) Erosión ($\times 2$).

Figura 2.1: Efecto de la operación de erosión y dilatación sobre la imagen (a). (b) Máscara o kernel utilizado para cada una de las operaciones. (c) Resultado de aplicar la dilatación ($\times 2$). (d) Resultado de aplicar la erosión ($\times 2$).

2.2 Realce de imágenes en el dominio espacial

En esta sección se presentan brevemente algunas técnicas en el dominio espacial. Las mismas, fueron seleccionadas debido a que presentan tiempos de cálculo pequeños y son orientadas a la tarea de realce de detalles o mejora de la iluminación en las imágenes.

Los métodos en el dominio espacial son procedimientos que operan directamente sobre los píxeles de la imagen y pueden denotarse con la expresión $g(x, y) = T[f(x, y)]$, donde $f(x, y)$ es la imagen de entrada, $g(x, y)$ es la imagen resultado de la operación y T es un operador sobre f definido sobre alguna vecindad alrededor de (x, y) .

El enfoque principal en la definición de la vecindad alrededor de un punto (x, y) , es el uso de una zona de sub imagen cuadrada o rectangular centrada en (x, y) . Cuando esta vecindad se establece a un valor de 1×1 (un píxel), g depende únicamente del valor de f en (x, y) y T se convierte en una función de transformación de intensidad de la forma $s = T(r)$ donde r y s son variables que denotan el nivel de gris de $f(x, y)$ y $g(x, y)$ en el punto (x, y) respectivamente.

2.2.1 Umbral

Podemos definir una función de umbral como una transformación con la forma de la expresión (2.3) donde r_{max} y s_{max} son los valores máximos de intensidad de la imagen $f(x, y)$ y $g(x, y)$ respectivamente y u una constante que define el valor de umbral de la transformación.

La función de umbral particular presentada en la Fig. 2.2 produce una imagen binaria asignando el valor s_{max} a los valores que superan u y 0 a los demás.

$$s = \begin{cases} 0 & \text{para } r < u, \\ s_{max} & \text{para } r \geq u \end{cases} \quad \text{con } 0 \leq r \leq r_{max} \text{ y } 0 \leq u \leq r_{max}. \quad (2.3)$$

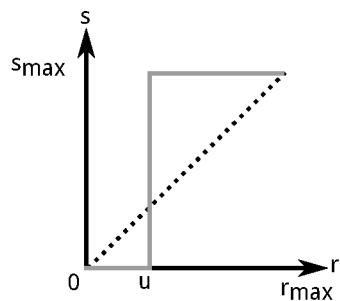


Figura 2.2: Umbral binario.

2.2.2 Transformación logarítmica

Esta transformación viene expresada por la ecuación (2.4), donde s es el valor resultante de aplicar la transformación sobre el valor de entrada r (con $r \geq 0$) y c es una constante, que en la práctica comúnmente se establece como $c = 1$. Utilizada cuando la imagen de entrada tiene un rango dinámico grande, expande las intensidades oscuras y comprime las intensidades claras como se puede observar en la curva de la Fig. 2.3.

$$s = c \log(1 + r). \quad (2.4)$$

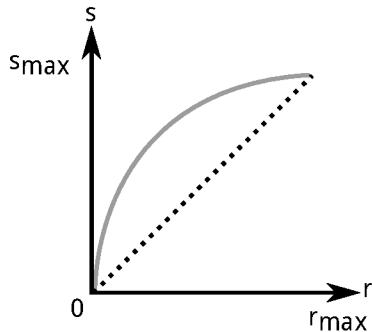


Figura 2.3: Curva de la transformación logarítmica.

2.2.3 Ecualización de histograma

El histograma es una función discreta que describe de manera global la apariencia de una imagen y puede ser interpretada como el número de píxeles

en función de las intensidades de grises.

La ecualización del histograma de una imagen de grises es una transformación que pretende obtener el mismo número de píxeles para cada nivel de gris de la imagen, logrando obtener así un histograma con una distribución que se aproxima a la uniforme. Como resultado de esta transformación global que redistribuye los grises de la imagen original para que abarque una gama más completa de los grises disponibles, se obtiene una expansión en el rango dinámico de la imagen.

Si se tienen imágenes digitales (valores discretos) con intensidades entre 0 y 1, y sea:

- n la cantidad de píxeles de la imagen,
- p el número de bits de resolución de la imagen,
- L el máximo nivel de gris, dado por el valor de p ,
- k el nivel de gris,
- r_k el nivel de gris normalizado a 1, con valores $r_k = 0, \frac{1}{2^p-1}, \frac{2}{2^p-1}, \dots, 1$.
- $p_r(r_k)$ la probabilidad de un nivel de gris en la imagen calculado como:

$$p_r(r_k) = \frac{n_k}{n}, \text{ con } \begin{cases} 0 \leq r_k \leq 1 \\ k = 0, 1, \dots, L-1 \end{cases}, \quad (2.5)$$

la función de transformación utilizada es la Función de Distribución Acumulada:

$$s_k(r_k) = T(r_k) = \sum_{j=0}^k \frac{n_j}{n} = \sum_{j=0}^k p_r(r_j), \text{ con } \begin{cases} 0 \leq r_k \leq 1 \\ k = 0, 1, \dots, L-1 \end{cases}. \quad (2.6)$$

2.2.4 Filtrado pasa altos

Los filtros pasa altos realzan las altas frecuencias que son los detalles de una imagen y pueden ser realizados mediante una diferenciación espacial. La aplicación de este tipo de filtros, realza ejes, bordes y diversas discontinuidades (incluido el ruido) y atenúa áreas cuyos niveles de grises varían lentamente (bajas frecuencias).

Un filtro pasa altos puede ser aplicado mediante la convolución de la imagen con un kernel pasa altos que puede ser de diferentes tamaños y que según

esto, actuará de diferentes maneras (realzando en mayor o menor medida los detalles). Si definimos a $f(x + s, y + t)$ como el valor de los píxeles del bloque seleccionado de la imagen y $w(s, t)$ los coeficientes del kernel, se puede expresar el filtrado lineal con la expresión definida en la ecuación (2.7).

$$g(x, y) = \sum_{s=-a}^a \sum_{t=-b}^b w(s, t) f(x + s, y + t). \quad (2.7)$$

Si la suma de los coeficientes del kernel es 1, se realzan las altas frecuencias conservándose el brillo medio de la imagen sobre la cual se aplica el filtro.

2.2.5 Filtrado de alta potencia

El filtrado de alta potencia es una generalización del filtro de máscara difusa y se obtiene mediante la diferencia entre una versión amplificada de la imagen original y una versión suavizada de la misma.

Sea una constante $A \geq 1$, $f(x, y)$ la imagen sobre la cual aplicaremos el filtrado de alta potencia, $PB(f(x, y))$ el resultado de aplicar un filtro pasa bajos a la imagen $f(x, y)$ y $g(x, y)$ el resultado de la operación, el filtrado de alta potencia queda definido como en la ecuación (2.8). En el caso particular de establecer el valor $A = 1$, se obtiene un filtrado pasa altos.

$$g(x, y) = Af(x, y) - PB(f(x, y)). \quad (2.8)$$

2.3 Detección de puntos claves

En visión computacional, el concepto de puntos de interés es ampliamente usado como se ha descripto en la Sec. 1.2.2. El método SURF fue desarrollado por Herbert Bay et. al [5] como un detector de puntos claves y descriptor robusto. El mismo, se basa en las ideas planteadas en el método denominado SIFT, pero con algunas diferencias que se presentan resumidas a continuación:

- SURF posee una velocidad de cálculo superior sin perder rendimiento de manera considerable, ésto es logrado mediante la reducción en la dimensión de los vectores descriptores y reduciendo la complejidad de cálculo en los mismos mediante el uso de imágenes integrales [65] y filtros tipo caja (del inglés, Box filters)[61],

- la gran velocidad de cálculo resulta ser un factor positivo, a pesar de una tolerable pérdida de robustez,
- el cálculo de la posición y la escala de puntos claves se realiza utilizando una aproximación a la matriz hessiana (que se detallará en la Sec. 2.3.2),
- utiliza siempre la imagen original para el análisis del espacio escala, a diferencia de SIFT que realiza un redimensionamiento de la imagen en cada paso,
- la longitud de los vectores característicos para SURF es la mitad que para SIFT (64 y 128 valores, respectivamente).

La selección del detector SURF, se ve fundamentada por ser un detector y descriptor de puntos de interés rápido, invariante a escala y rotación que provee una ganancia en velocidad debido al uso de imágenes integrales [5]. Este tipo de imágenes permite reducir drásticamente el número de operaciones para convoluciones con los filtros tipo caja, independientemente del tamaño de escala. Además, SURF utiliza una aproximación a la matriz hessiana para la detección de puntos de interés la cual resulta comparable con los detectores de puntos de interés del estado del arte, o aún incluso, se obtiene mejores resultados en ciertos casos [5]. En lo que respecta al descriptor, al estar basado en la suma de las componentes de las Wavelets Haar, la naturalidad de descripción del patrón de intensidad subyacente a la imagen resulta ser más distintiva que los enfoques basados en histogramas.

En las secciones siguientes, se presenta en detalle los pasos mediante los que se lleva a cabo la detección y descripción de características mediante SURF, introduciendo brevemente los conceptos de invarianza a escala y de imágenes integrales.

2.3.1 El problema de cambio de escala en correspondencias de puntos

Cuando se trata de hacer coincidir características entre imágenes (por ejemplo para el reconocimiento de las mismas), el problema del cambio de escala se hace presente. Al ser analizadas distintas imágenes, éstas pueden estar tomadas a diferentes distancias respecto del objeto de interés, de forma que los objetos aparecen de diferentes tamaños en la imagen. Luego, si se trata de hacer coincidir las mismas características entre dos imágenes usando un tamaño fijo de píxeles vecinos, la intensidad de los patrones no coincidirá debido al cambio de escala presente en las mismas y por lo tanto, el reconocimiento fallará.

Para solucionar este problema, el concepto de características invariantes a la escala es introducido en visión computacional, en el que la idea central radica en tener un factor de escala asociada con cada punto clave detectado.

Para comprender mejor esta situación, en la Fig. 2.4 se presenta un ejemplo de puntos claves detectados con el método SURF. Si se considera la parte inferior de la ventana superior derecha de la fotografía, tanto en la Fig. 2.4a como en la Fig. 2.4b, la característica SURF ha sido detectada en la misma ubicación y los círculos correspondientes (de diferentes tamaños) contienen los mismos elementos visuales. En la Fig. 2.4a la cámara se encuentra más cerca de la escena y el círculo marcado con la flecha roja posee la misma información visual que en la Fig. 2.4b donde la escena fue capturada a diferente escala (alejándose la cámara de la escena). Si bien, en este caso no se da para todas las todas las características, la razón de repetición es lo suficientemente alta para permitir buenas coincidencias entre las dos imágenes. Además, como información adicional, se puede observar una línea radial dentro de cada círculo que indica una orientación asignada a cada punto.

Invarianza a escala

Es sabido que los filtros gaussianos son comúnmente usados para detección de puntos [5]. Si la escala del filtro es muy pequeña, el resultado incluye muchos puntos redundantes de detalles innecesarios. Contrariamente, si la escala es muy grande, los puntos de regiones con soporte pequeño tienen a desaparecer con el borroneado. Luego, para solucionar los problemas existentes en el filtrado gaussiano con escalas fijas, se han propuesto procedimientos espacio-escala basados en la representación de la curvatura discreta multi escalar. El esquema está basado en el criterio de estabilidad que establece que la presencia de una esquina debe ocurrir como un máximo de curvatura observable en la mayoría de las escalas [41].

La derivada de una imagen puede ser estimada mediante el uso de filtros gaussianos. Los mismos utilizan un parámetro σ que representa la varianza de la función gaussiana usada para construir el filtro y que define implícitamente la escala en que la derivada es evaluada.

Si se calcula el laplaciano de un punto en una imagen usando filtros gaussianos a diferentes escalas, se obtienen diferentes valores en los que se puede observar la evolución de las respuestas para diferentes factores de escala. Así, se obtiene una curva que alcanza un valor máximo en algún valor de σ específico. Si se extrae este valor para dos imágenes del mismo objeto (tomadas a escalas diferentes), la relación que existe entre los σ máximos se corresponderán en relación con las escalas en que fueron tomadas cada una

de las fotografías. Esta observación es el núcleo del proceso de extracción de características invariantes a la escala que es aplicado tanto en SIFT como en SURF.

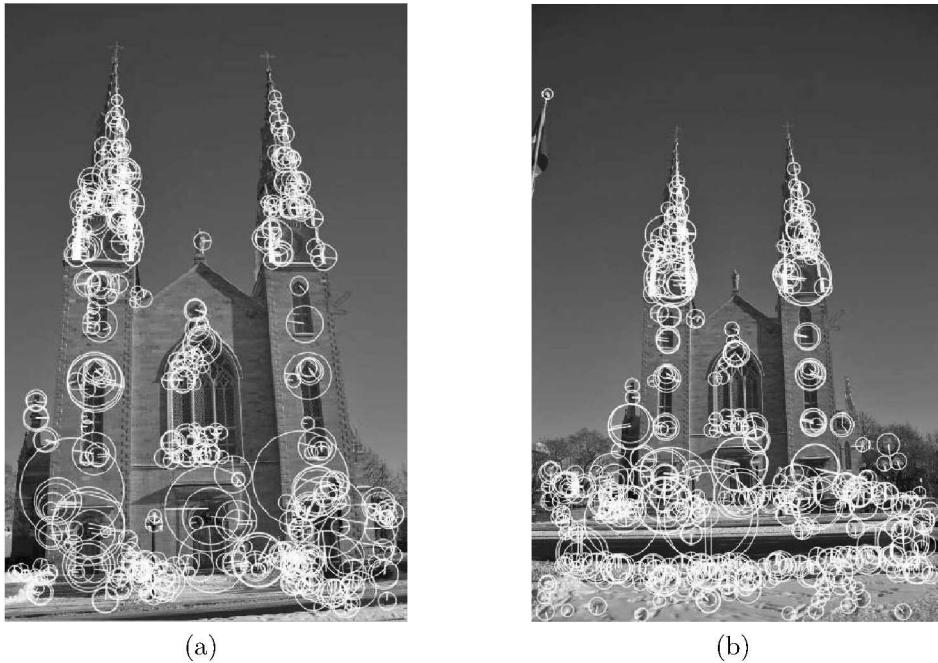


Figura 2.4: Fotografía tomada a diferentes escalas de la misma escena. El tamaño de los círculos de cada punto clave detectado es proporcional a la escala calculada para cada uno de ellos. (Figuras tomadas de [33]).

2.3.2 Puntos claves basados en la matriz hessiana

Con la idea que se introdujo en la sección 2.3.1, el descriptor SURF hace uso de la matriz hessiana (2.9) y describe cómo las intensidades de los píxeles se distribuyen dentro de una vecindad, que es dependiente de la escala de cada punto de interés detectado por el hessiano. Más específicamente, utiliza el determinante de la matriz para la determinación de la localización y escala de los puntos. Así, cuando el determinante del hessiano es un máximo local (previa eliminación de mínimos mediante un umbral), se determina un punto

clave. El uso del hessiano es alentado por su velocidad de cálculo y precisión.

$$H(x, y) = \begin{bmatrix} \frac{\partial^2 I}{\partial x^2} & \frac{\partial^2 I}{\partial x \partial y} \\ \frac{\partial^2 I}{\partial y \partial x} & \frac{\partial^2 I}{\partial y^2} \end{bmatrix}; \text{ con } \frac{\partial^2 I}{\partial x \partial y} = \frac{\partial^2 I}{\partial y \partial x}. \quad (2.9)$$

Sea $\mathbf{p} = (x, y)$ un punto en la imagen I , la matriz hessiana en \mathbf{p} en la escala σ viene definida como:

$$\mathcal{H}(\mathbf{p}, \sigma) = \begin{bmatrix} L_{xx}(\mathbf{p}, \sigma) & L_{xy}(\mathbf{p}, \sigma) \\ L_{xy}(\mathbf{p}, \sigma) & L_{yy}(\mathbf{p}, \sigma) \end{bmatrix}, \quad (2.10)$$

donde $L_{xx}(\mathbf{p}, \sigma)$ es la convolución de la derivada segunda de una gaussiana $\frac{\partial^2}{\partial x^2}g(\sigma)$ con la imagen I en el punto \mathbf{p} ; $L_{xy}(\mathbf{p}, \sigma)$ es la convolución de la derivada segunda de una gaussiana $\frac{\partial^2}{\partial x \partial y}g(\sigma)$ con la imagen I en el punto \mathbf{p} y $L_{yy}(\mathbf{p}, \sigma)$ es la convolución de la derivada segunda de una gaussiana $\frac{\partial^2}{\partial y^2}g(\sigma)$ con la imagen I en el punto \mathbf{p} .

A pesar de que los filtros gaussianos pueden ser utilizados para el análisis del espacio escala como se introdujo en la Sec. 2.3.1, SURF hace uso de los filtros tipo caja [60]. Éstos son una aproximación a las derivadas gaussianas segundas y pueden ser aplicados rápidamente cuando se utilizan con las imágenes integrales (véase más adelante la Sec. 2.3.3). Aquí es donde se encuentra presente una de las diferencias que contribuye a una mejora en velocidad del método SURF respecto a SIFT, pues en este último el espacio escala es creado a partir de imágenes suavizadas repetidamente mediante un filtro gaussiano que luego, son submuestreadas (produciéndose un efecto de aliasing indeseado) para alcanzar escalas mayores. En el caso de SURF, el espacio escala es analizado mediante el incremento en el tamaño del filtro. Un esquema del espacio escala SIFT puede verse en la Fig. 2.5 (izquierda) en el que se redimensiona la imagen para obtener las escalas sucesivas; mientras que en el espacio escala SURF en la Fig. 2.5 (derecha) el redimensionamiento se produce sobre el filtro tipo caja y la imagen es siempre la misma.

Así el determinante de la matriz hessiana (aproximado) usado por SURF queda definido como:

$$\det(\mathcal{H}_{\text{aproximado}}) = D_{xx}D_{yy} - (wD_{xy})^2 \quad (2.11)$$

con $w = 0.9$ y donde D_{xx} , D_{yy} y D_{xy} son las derivadas parciales gaussianas de segundo orden aproximadas. La ponderación relativa w de la respuesta del filtro es usada para balancear la expresión del determinante hessiano. Si bien el mismo varía para diferentes escalas, en la práctica se puede establecer este

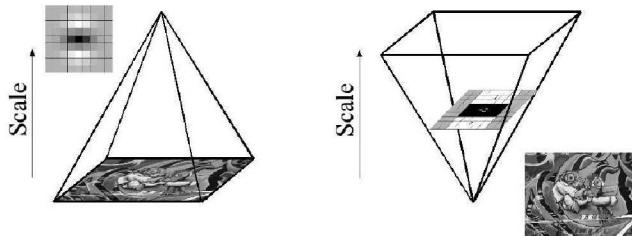


Figura 2.5: Pirámide de escala de imágenes para el método SIFT (izquierdo) y el método SURF (derecha). (Figura tomada de [5]).

factor como constante con $w = 0.9$, ya que el mismo no tiene un impacto significante en los resultados [5].

La derivada parcial de segundo orden gaussiana en la dirección “y” L_{yy} puede ser observada en la Fig. 2.6a y su homóloga aproximada D_{yy} en la Fig. 2.6b; en tanto que en la Fig. 2.6c se encuentra representada la derivada parcial de segundo orden gaussiana en la dirección “x-y” L_{xy} y su homóloga aproximada D_{xy} (Fig. 2.6d). Como se ha mencionado, en el método SURF

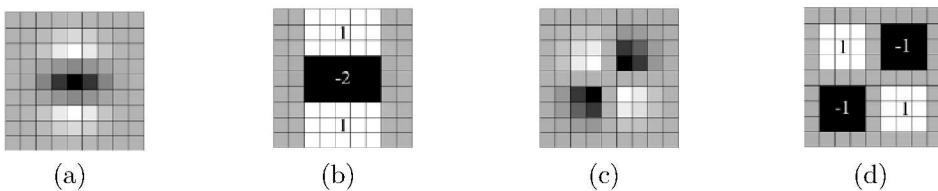


Figura 2.6: Derivadas parciales gaussianas de segundo orden discretas y sus homólogas aproximadas (también referenciadas como filtros tipo caja). Las regiones grises de la imagen son iguales a cero (Figuras tomadas de [5]).

se aplican sucesivamente filtros de tamaño creciente sobre la imagen original. El filtro más pequeño utilizado tiene un tamaño de 9×9 y corresponde a la derivada parcial de segundo orden de una gaussiana con $\sigma = 1.2$ siendo este el nivel de escala inicial que da la máxima resolución espacial ($s = 1.2$). De la misma forma, convolucionando la imagen original con filtros de mayores dimensiones se van obteniendo los niveles siguientes.

El espacio escala para el descriptor SURF está dividido en octavas. Una octava representa una serie de respuestas obtenidas mediante la convolución de la imagen original con filtros de tamaños cada vez mayores. En total, una octava comprende un factor de escalado de 2 y cada una de ellas es subdividida en un número constante de niveles de escala (véase la Fig. 2.8). En la Fig. 2.7 se puede observar un esquema gráfico del análisis de 3 octavas en donde

el eje horizontal, expresado logarítmicamente, representa las escalas. Notar que para cada nueva octava, el tamaño del filtro es incrementado al doble (en la primer octava el paso es 6, en la segunda de 12, en la tercera 24, etc.) y cada una de ellas empieza con un tamaño de filtro igual al segundo de la octava anterior. También, se puede observar que las octavas están solapadas, esto, con el objetivo de cubrir todas las posibles escalas. Por ejemplo, para la primer octava el espacio escala comienza con filtros de 9×9 continuando con los de 15×15 , 21×21 , y 27×27 . Debido a que se realiza una supresión de no-máximos¹ tanto espacialmente como con las escalas vecinas, las respuestas del hessiano en el primer y último nivel son usadas solo para comparación. Por ello, luego de realizar una interpolación, la escala posible más pequeña es $\sigma = 1.6$ correspondiente a un filtro de 12×12 , y la más grande a $\sigma = 3.2$ correspondiente a un filtro de 24×24 . Consideraciones similares se aplican para las demás octavas. Para más detalles se puede consultar [4].

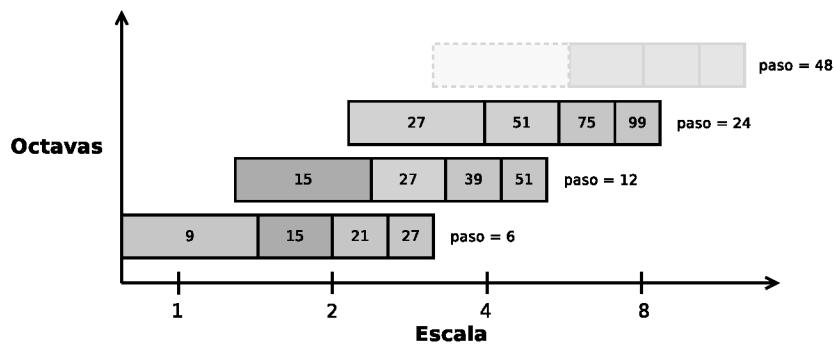


Figura 2.7: Representación gráfica de las longitudes de los filtros para 3 diferentes octavas (Figura adaptada de [5]).

2.3.3 Determinación de la localización de puntos claves

Como se ha visto en la sección anterior, se construye una “pirámide de respuestas” con diferentes niveles de escalas dentro de las octavas. La localización de los puntos claves como su escala, se determinan hallando los extremos entre los 8 vecinos en el nivel evaluado y los 2×9 vecinos en el nivel inmediatamente superior e inferior mediante la supresión de no-máximos

¹La supresión de no-máximos fija en cero a todos los píxeles de una vecindad que son menores al máximo valor presente en la misma. También se utiliza el término filtro de máxima para referirse a este concepto.

en la vecindad de $3 \times 3 \times 3$. Un esquema puede ser observado en la Fig. 2.8 donde se demarca dicha vecindad.

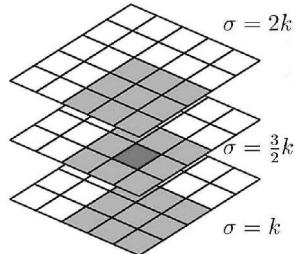


Figura 2.8: Representación de una octava compuesta por tres niveles de escala $\sigma = k$, $\sigma = \frac{3}{2}k$, $\sigma = 2k$.

Una vez que el máximo local es identificado, la posición precisa de cada punto clave es obtenida a través de interpolación y el resultado es un conjunto de puntos claves localizados con precisión sub-píxel, los cuales están asociados a un valor de escala [4].

Imágenes integrales

Las imágenes integrales [65] permiten calcular más rápidamente la convolución con filtros tipo caja. La imagen de entrada $I_{\sum}(\mathbf{x})$ en la posición $\mathbf{x} = (x, y)^T$ representa la suma de todos los píxeles de la imagen de entrada I dentro de una región rectangular formada por el origen y \mathbf{x} .

$$I_{\sum}(\mathbf{x}) = \sum_{i=0}^{i \leq x} \sum_{j=0}^{j \leq y} I(i, j). \quad (2.12)$$

Una vez que la imagen integral se ha calculado, son necesarias tres sumas y cuatro operaciones de acceso a memoria para calcular la suma de las intensidades sobre cualquier área rectangular vertical independientemente de su tamaño (véase la Fig. 2.9). El tiempo de cálculo resulta independiente del tamaño, lo cual favorece en la aproximación de SURF en la que se utilizan filtros de grandes tamaños.

La utilización de esta técnica, resulta en un algoritmo invariante a tiempos de cálculo ante cambios del tamaño de la imagen, lo cual lo hace particularmente útil cuando se presentan imágenes de grandes dimensiones.

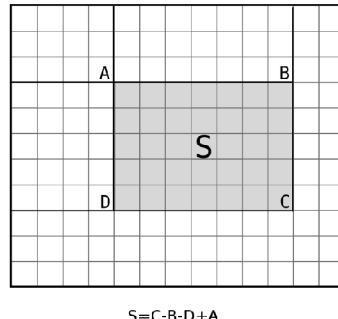


Figura 2.9: Suma de intensidades en un rectángulo usando imágenes integrales.

2.4 Descripción de puntos claves

Tras la localización y la asociación de escala a los puntos claves, la siguiente etapa corresponde en asignar una orientación a cada uno de ellos para otorgarles invarianza ante la rotación mediante la orientación del mismo.

2.4.1 Asignación de la orientación

Para definir la orientación, primeramente se calculan las respuestas con los filtros presentados en la Fig. 2.10 sobre un área circular de radio $6s$ alrededor del punto clave, siendo s la escala en que se detectó el punto. Tanto el muestreo como el tamaño de los filtros son dependientes de la escala y se establecen a s y $4s$ respectivamente (a mayor escala, mayor es la dimensión de las wavelets). La evaluación de los filtros realizada aquí, se vale nuevamente de las imágenes integrales para realizar los cálculos más rápidamente. Tras el cálculo de las respuestas, las mismas son ponderadas con una gaussiana de valor $\sigma = 2s$ centrada en el punto clave.

Luego, las respuestas son representadas como puntos en el espacio (respuestas horizontales a lo largo del eje de abscisas dx y verticales dy en el de ordenadas) y la orientación dominante se determina mediante la suma de todas las respuestas en una ventana deslizante orientada de tamaño $\pi/3$, como puede observarse en la Fig. 2.11. Así, las respuestas horizontales y verticales dentro de la ventana son sumadas constituyendo un vector de orientación

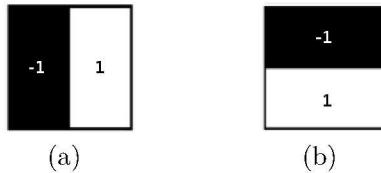


Figura 2.10: Filtros Wavelets Haar para calcular las respuestas en la dirección x (a) e y (b).

local. La orientación final para el punto clave, es aquella en la que el vector resulta ser el más largo entre todas las ventanas.

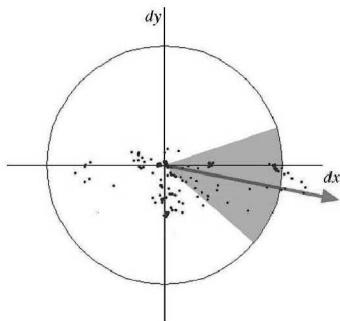


Figura 2.11: Asignación de la orientación para un punto clave con una ventana deslizante orientada de tamaño $\frac{\pi}{3}$. (Figura tomada de [5]).

2.4.2 Creación del descriptor

En esta última parte, se termina la creación del descriptor SURF. El primer paso consiste en construir una región cuadrada de tamaño $20s$ centrada en el punto de interés y orientada de acuerdo al resultado que se obtuvo en la sección 2.4.1. Ejemplos de esas regiones cuadradas son ilustradas en la Fig. 2.12. Seguidamente, cada región es dividida en subregiones de 4×4 como se puede observar en la grilla cuadrada orientada sobre el punto de interés en la Fig. 2.13a. Luego para cada subregión se calculan las respuestas de las waveletes Haar con una separación de muestreo de 5×5 . Las subdivisiones de 2×2 de cada cuadrado corresponden a los campos reales del descriptor que son las sumas dx , $|dx|$, dy y $|dy|$ calculadas relativas a la orientación de la grilla como se observa en la Fig. 2.13b donde se denota con d_x y d_y a la respuesta de la wavelet Haar en la dirección horizontal y vertical respectivamente (relativas a la orientación del punto clave, con tamaño del filtro



Figura 2.12: Detalle de una escena mostrando el tamaño de las ventanas orientadas del descriptor a diferentes escalas. (Figura tomada de [5]).

igual a $2s$). Se debe tener en cuenta que para incrementar la robustez ante deformaciones geométricas y errores de localización, las respuestas d_x y d_y son primero ponderadas con un gaussiano con $\sigma = 3.3s$ centrado en el punto de interés. Luego, las respuestas d_x y d_y se suman en cada subregión, como así también los valores absolutos de las mismas $|d_x|$ y $|d_y|$ con el objetivo de brindar información de la polaridad sobre los cambios de intensidad. De esta forma, cada subregión queda representada por un vector de 4 dimensiones \mathbf{v} , que caracteriza su intensidad localmente $\mathbf{v} = (\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|)$. Debido a que se tienen 16 subregiones (4×4) con vectores de 4 dimensiones por subregión, se obtiene un vector descriptor de 64 dimensiones (16×4) por cada punto clave detectado, por lo que se puede decir que el descriptor SURF consiste en las respuestas de la wavelet Haar en una región de 4×4 alrededor del punto clave [16]. Las repuestas wavelet son invariantes a un sesgo en la iluminación, mientras que la invarianza al contraste es alcanzado mediante la normalización del descriptor.

En la Fig. 2.14 se pueden observar los descriptores para tres subregiones con patrones de imágenes diferentes. En el caso de una región homogénea se puede observar que todos los valores son relativamente bajos; en presencia de frecuencias en la dirección de x , los valores de $\sum |d_x|$ son altos, mientras los otros son bajos y si la intensidad se incrementa gradualmente en la dirección x , ambos valores $\sum d_x$ y $\sum |d_x|$ son altos.

Si bien existen muchos parámetros del método que pueden variarse, en este capítulo se han expuestos aquellos que el autor [5] recomienda según sus resultados en la publicación original.

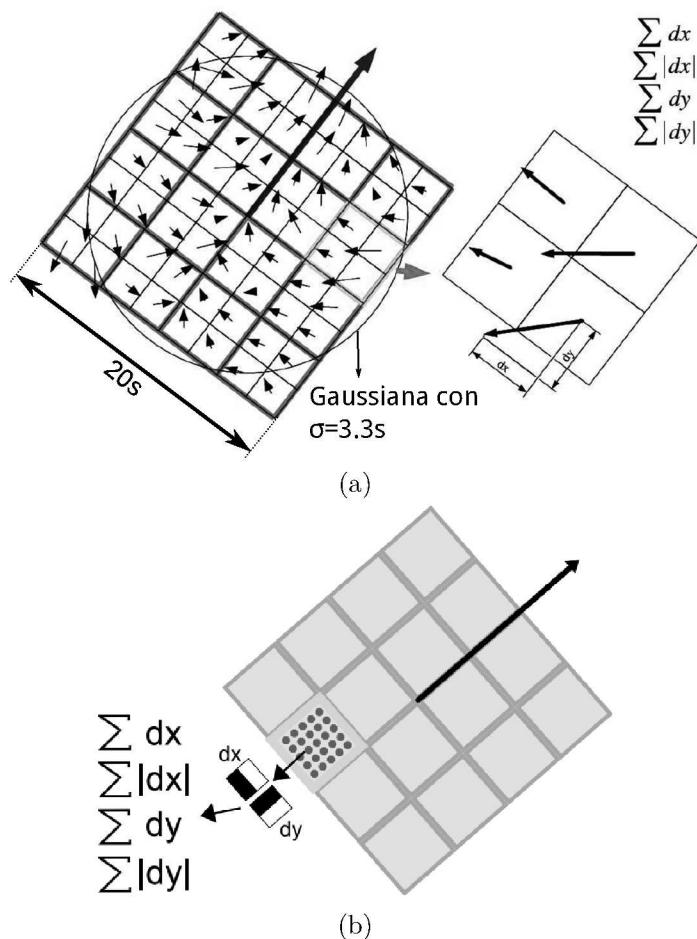


Figura 2.13: Interpretación gráfica del descriptor SURF. (Figuras tomadas de [5]).



Figura 2.14: Descriptores resultantes para tres subregiones con patrones diferentes (Figura tomada de [5]).

2.5 Correspondencia entre puntos claves

El término correspondencia entre puntos o entre imágenes se puede interpretar como el cálculo de un valor que represente el grado de similitud entre dos imágenes. Para ello, la correspondencia entre los puntos claves de dos imágenes es buscada mediante el cálculo de la distancia euclídea entre los vectores característicos asociados a los puntos claves detectados en cada una de las imágenes. Es decir que el problema que se plantea aquí, es el de comparar los descriptores de los puntos claves de las imágenes para determinar coincidencias de puntos y así afirmar o no la presencia del objeto buscado.

El método de búsqueda del vecino más cercano (NNS: del inglés, Nearest Neighbor Search) [2] es un método de gran utilidad. Se ha aplicado a gran variedad de aplicaciones como el reconocimiento de imágenes, la compresión de datos, los sistemas de recuperación de documentos, estadísticas y análisis de datos, entre otros. NNS es un problema de optimización que intenta buscar los puntos más cercanos en un espacio métrico: dado un conjunto de puntos $P = \{p_1, \dots, p_n\}$ en un espacio métrico M y un punto de consulta $q \in M$, encontrar el punto más cercano a q en P de forma eficiente, donde M es un espacio euclídeo d-dimensional y la distancia es medida por ejemplo mediante la distancia euclídea.

Existe una variante al algoritmo NNS denominada k-NN (k-Nearest Neighbor) que a diferencia del anterior, se evalúa a qué clase pertenecen los K vecinos más cercanos para decidir la clase. Así, en el caso $K = 1$ se está en presencia del algoritmo NNS que se describió anteriormente.

Resolver problemas de búsqueda de vecinos más cercanos, no resulta trivial en espacios de grandes dimensiones [54]. No es usual encontrar algoritmos que posean un rendimiento mayor al de la búsqueda lineal (también conocida como “búsqueda por fuerza bruta”), la cual resulta costosa computacionalmente y hasta a veces, imposible de usar en muchas aplicaciones. Es por esto, que se ha generado un gran interés en algoritmos que puedan realizar la búsqueda del vecino más cercano de forma aproximada, con lo cual es posible lograr mejoras significativas en tiempo de ejecución con errores de precisión relativamente pequeños y aceptables [7, 54].

En el trabajo de Slipa-Anan y Hartley [1, 24], se propuso la creación de una estructura de múltiples árboles KD o K-dimensionales aleatorios conocidos por su término en inglés como **Randomized KD-Tree**, que brinda la posibilidad de obtener resultados satisfactorios en un amplio rango de problemas

[54]. Más específicamente, para casos en los que se trata con vectores similares a los presentados en la Sec. 2.4 resulta una alternativa aceptable. Por ello, se utilizará el método descripto en [1, 24] con los parámetros estudiados que se describen en [54].

2.5.1 El algoritmo de árboles KD aleatorio

El algoritmo KD-tree clásico resulta eficiente con datos de bajas dimensiones [18], pero su rendimiento se ve afectado rápidamente al aumentar la dimensionalidad de los datos. Para obtener una velocidad mayor a la de la búsqueda lineal, se hace necesario establecer una búsqueda aproximada del vecino más cercano. Esto mejora el tiempo de búsqueda pero, como contrapartida, el algoritmo no siempre da como resultado el vecino más cercano. Para realizar esta búsqueda aproximada de forma rápida, se crea una estructura de árbol que contribuye a la reducción en los tiempos de procesamiento.

Los elementos guardados en el árbol KD-tree, son vectores de altas dimensiones. En la raíz del árbol (primer nivel), los datos son divididos en dos mitades por un hiper plano ortogonal para una dimensión elegida y con un valor de umbral. Generalmente, esta división se realiza con la media, en la dimensión con la mayor varianza del conjunto de datos. En características visuales provistas por SIFT o SURF, utilizar la media en la dimensión con mayor varianza es la que presenta el mejor rendimiento [54]. Para construir el árbol, se compara el vector de entrada con el “valor de partición” para determinar a qué mitad del árbol pertenece dicho vector. Cada una de las dos mitades de los datos es dividida de igual manera y en forma recursiva, para lograr crear un árbol binario completamente balanceado.

A diferencia del algoritmo KD-tree clásico, los árboles aleatorios son construidos seleccionando la dimensión de división de forma aleatoria sobre las primeras D dimensiones en las que los datos poseen mayor varianza. Se usa el valor fijo $D = 5$ que resulta el más adecuado para diferentes datos [54].

Cuando se realiza la búsqueda en el árbol, una cola con prioridad es mantenida a través de todos los arboles aleatorios, por lo que la búsqueda queda ordenada mediante el incremento de la distancia a cada nodo del borde. El grado de aproximación, se determina mediante el examen de un número fijo de nodos hoja. Cuando es alcanzado este número, se termina la búsqueda y se obtienen los candidatos. Se debe tener en cuenta que la cantidad de memoria utilizada aumenta linealmente con el número de árboles aleatorios, una característica negativa cuya importancia no resulta menor en la sobrecarga

del sistema.

2.5.2 Remoción de correspondencias no válidas

Usar la forma de búsqueda de coincidencias descripta en la sección anterior no resulta del todo adecuada, ya que para un punto clave de consulta siempre se encontrarán valores de correspondencias que no necesariamente son válidos. Por ello, se hace necesario la aplicación de un umbral para determinar las potenciales correspondencias válidas. Usar un valor de umbral global que se compare con la distancia a la característica más cercana no funciona correctamente según lo afirma Lowe [44]. Por ello, se propone considerar una medida más efectiva obtenida mediante la comparación de la distancia del vecino más cercano respecto del segundo vecino más cercano. Así la estrategia que se propone en [44] es la utilizada en este trabajo y es la que se describe a continuación.

Sean A y B dos imágenes sobre las cuales se quieren buscar correspondencias. Consideremos a_i con $i = 0 \dots n$ un punto del conjunto de puntos claves detectados en A , n el total de puntos claves detectados en A , av_i el vector característico asociado al punto a_i y de forma similar para la imagen B . Para cada a_i , se seleccionan los dos mejores puntos claves candidatos $p_1 \in b_i$ y $p_2 \in b_i$ cuyos vectores de características asociados a cada uno representan las distancias euclídeas mínimas d_1 y d_2 respectivamente respecto a av_i . Luego, si se cumple la proporción $\frac{d_1}{d_2} > \varepsilon$ ($\varepsilon = 0.8$ [44]) la coincidencia es rechazada. El valor de $\varepsilon = 0.8$ fue seleccionado de acuerdo al estudio llevado a cabo por Lowe [44] que afirma que se alcanzan a eliminar un 90 % de falsas coincidencias mientras se descartan sólo un 5 % de buenas coincidencias, resultando en el valor más apropiado.

Esta remoción de pares de correspondencias, que resultan presuntamente pares inadecuados, reduce el número de pares disponibles para buscar la correspondencia pero realza la habilidad de buscar la homografía correcta mediante la reducción de correspondencias incorrectas.

2.6 Conceptos de formación de la imagen y transformación proyectiva

Hasta el momento, se ha descripto la forma de obtener puntos claves y vectores descriptores sobre imágenes y mediante la correspondencia de características, determinar la similitud entre dos imágenes. Sin embargo, aún no se tiene suficiente información para determinar la posición del objeto buscado. Para ello, es necesario introducir algunos conceptos básicos de formación de la imagen que se explicarán a continuación.

2.6.1 Introducción

El principal componente para la visión en una escena es la luz. La misma proviene como un rayo que sale de una fuente (por ejemplo: el sol o una lámpara) y viaja a través del espacio hasta intersectar un objeto donde parte de la luz es absorbida y otra reflejada (percibida como el color) la cual es captada por los ojos (o cámara) y recogida en la retina (o imagen).

Existen diversos dispositivos para obtener imágenes, uno de los más comunes, es la cámara digital. Ésta captura la escena mediante la proyección de luz en un sensor a través del lente de la cámara. El hecho de que la imagen se forma a través de la proyección de la escena 3D en un plano 2D, implica la existencia de importantes relaciones entre la escena y su imagen, y es la geometría proyectiva la herramienta usada para describir y caracterizar en términos matemáticos este proceso de formación de la imagen [9, 33]. Existe un modelo teórico útil denominado **modelo de cámara oscura** (del inglés: “**pinhole camera model**”) [9, 23] con el cual se puede entender la geometría básica en la proyección de rayos.

2.6.2 El modelo de cámara oscura

La cámara oscura es un instrumento óptico, que permite obtener una proyección plana de una imagen externa en la pared interna de una caja. La simplificación presentada en el modelo de cámara oscura es el fundamento del funcionamiento de las cámaras fotográficas. A continuación, se presentan

diferentes pasos a través de los cuales se generalizará progresivamente el modelo básico de cámara oscura para entender la geometría básica en la proyección de rayos.

Consideremos la proyección central de puntos en el espacio sobre un plano. Sea \mathbf{C} el centro de proyección (*centro de la cámara* o *centro óptico*) como origen de un sistema de coordenadas euclídeo, $Z = f$ el *plano de la imagen* o *plano focal*, p el *punto principal* y considerando que el plano de la imagen está posicionado enfrente del centro de la cámara. Bajo el modelo de cámara oscura, el punto $\mathbf{X} = (X, Y, Z)^T$ en el espacio, es mapeado a un punto en el plano de la imagen dado por la intersección del mismo, con la línea que une \mathbf{X} con el centro de proyección C como se puede observar en la Fig. 2.15a. Mediante el análisis geométrico de la Fig. 2.15b, se puede calcular que el punto \mathbf{X} de la escena es mapeado al plano de la imagen en el punto $\mathbf{x} = (fX/Z, fY/Z, f)^T$. Ignorando la última componente, el mapeo del sistema de coordenadas global al de coordenadas de la imagen (un mapeo de un espacio Euclídeo \mathbb{R}^3 a uno en \mathbb{R}^2) puede expresarse mediante la notación (2.13), asumiéndose que el origen de coordenadas en el plano de la imagen se encuentra en el punto principal.

$$(\mathbf{X}, \mathbf{Y}, \mathbf{Z}) \mapsto (fX/Z, fY/Z)^T. \quad (2.13)$$

Si los puntos son representados mediante una representación homogénea, la proyección central puede expresarse como un mapeo lineal entre coordenadas homogéneas y la notación (2.13) puede re-expresarse como en la ecuación (2.14).

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} f & 0 & 0 \\ f & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \quad (2.14)$$

Si se denota con \mathbf{X} al punto de la escena representado por el vector homogéneo $(X, Y, Z, 1)^T$, \mathbf{x} al punto en el plano de la imagen representado por un vector de 3 dimensiones homogéneo y P a la *matriz de proyección de la cámara* homogénea de 3×4 , la expresión 2.14 puede ser escrita como

$$\mathbf{x} = P\mathbf{X}. \quad (2.15)$$

Si bien anteriormente se asumió que el origen de coordenadas en el plano de la imagen se encuentra en el punto principal, en la práctica generalmente esto no es así, por lo que la expresión 2.13 se puede convertir a una forma

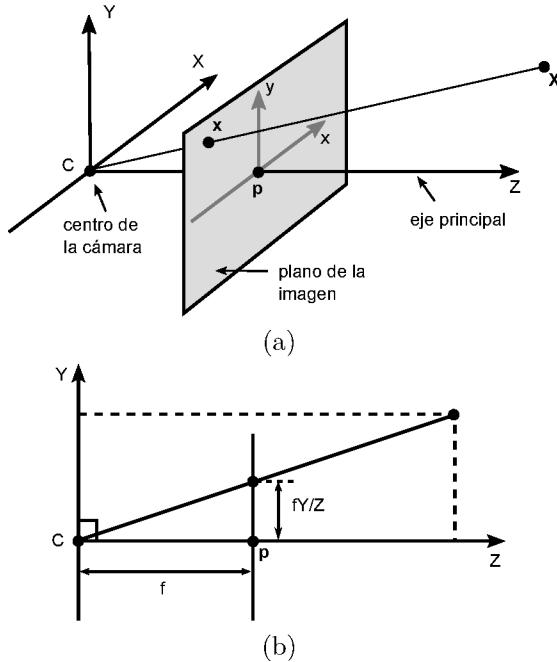


Figura 2.15: Modelo de cámara oscura (a) y su interpretación geométrica (b). (Figuras adaptadas de [23]).

más general como se establece en 2.16, donde $(p_x, p_y)^T$, son las coordenadas del punto principal. La expresión:

$$(X, Y, Z)^T \mapsto (fX/Z + p_x, fY/Z + p_y)^T, \quad (2.16)$$

puede ser expresada en coordenadas homogéneas:

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} f & p_x & 0 \\ f & p_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}. \quad (2.17)$$

Si denotamos con $[I | \mathbf{0}]$ la representación de una matriz dividida en un bloque de 3×3 (la matriz identidad I) al que se le concatena un vector columna (un vector de ceros de $[3 \times 1] \mathbf{0}$) y sea la *Matriz de calibración de la cámara*:

$$K = \begin{bmatrix} f & p_x \\ f & p_y \\ 0 & 0 \end{bmatrix}, \quad (2.18)$$

combinando esto en la expresión 2.17 se puede llegar a una nueva expresión:

$$\mathbf{x} = K[I | \mathbf{0}]\mathbf{X}_{cam}, \quad (2.19)$$

donde \mathbf{X}_{cam} representa al vector $(X, Y, Z, 1)^T$, asumiéndose que la cámara se encuentra posicionada en el origen del sistema de coordenadas euclídeo con el eje principal de la cámara apuntando en la dirección negativa del eje Z y donde el punto \mathbf{X}_{cam} está expresado en dicho sistema de coordenadas. Tal sistema de puede denominar como *sistema de coordenadas de la cámara*.

Parámetros extrínsecos e intrínsecos

Generalmente los puntos del espacio están expresados en un sistema de coordenadas denominado “Global”. Este sistema y el *sistema de coordenadas de la cámara* están relacionados mediante una rotación y una traslación como se puede observar en el esquema de la Fig. 2.16. Sea:

- $\tilde{\mathbf{X}}$ un vector de 3 dimensiones no homogéneo que representa las coordenadas de un punto en el sistema de coordenadas global,
- $\tilde{\mathbf{X}}_{cam}$ el mismo punto, pero en el sistema de coordenadas de la cámara,
- $\tilde{\mathbf{C}}$ las coordenadas del centro de la cámara en el sistema global y,
- R la matriz de rotación de 3×3 que representa la orientación del sistema de coordenadas de la cámara,

se puede escribir: $\tilde{\mathbf{X}}_{cam} = R(\tilde{\mathbf{X}} - \tilde{\mathbf{C}})$ que en coordenadas homogéneas se puede expresar como:

$$\mathbf{X}_{cam} = \begin{bmatrix} R & -R\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\tilde{\mathbf{C}} \\ 0 & 1 \end{bmatrix} \mathbf{X} \quad (2.20)$$

y combinado con la expresión 2.19, resulta en

$$\mathbf{x} = KR[I|-\tilde{\mathbf{C}}]\mathbf{X}, \quad (2.21)$$

que es la ecuación general de mapeo para el modelo de cámara oscura, donde \mathbf{X} está expresado en coordenadas globales. La ecuación (2.21) tiene 9 grados de libertad (9DoF): 3 para K (f, p_x, p_y), 3 para R y 3 para $\tilde{\mathbf{C}}$ donde los parámetros de K son conocidos como *parámetros internos de la cámara o intrínsecos* que resultan constantes para un sistema lente-cámara, mientras que los de R y $\tilde{\mathbf{C}}$ son denominados *externos o extrínsecos* y relacionan la orientación de la cámara y la posición del sistema de coordenadas global y resultan ser diferentes para diferentes “puntos de vista”.

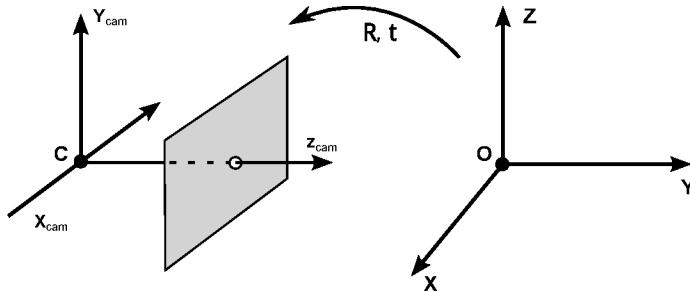


Figura 2.16: Esquema de transformación euclídea entre el sistema de coordenadas globales y el de la cámara. (Figura adaptada de [23]).

Si no se explicita el centro de la cámara, se puede representar la transformación del sistema global al de la imagen como $\tilde{\mathbf{X}}_{cam} = R\tilde{\mathbf{X}} + \mathbf{t}$ y la matriz de la cámara puede simplificarse a la expresión:

$$P = K[R|\mathbf{t}] \text{ con } \mathbf{t} = -R\tilde{\mathbf{C}}. \quad (2.22)$$

El modelo de la cámara derivado hasta el momento, asume que las coordenadas de las imágenes son coordenadas euclídeas, que tienen igual escala en ambos ejes direccionales. Sin embargo, esto no siempre es así, ya que los píxeles de las cámaras pueden no ser perfectamente cuadrados. Denotando m_x y m_y el número de píxeles por unidad de distancia en las direcciones x e y respectivamente (en coordenadas del plano imagen), se puede obtener una forma general de la matriz de calibración de la cámara mostrada en 2.23 donde $\alpha_x = fm_x$ y $\alpha_y = fm_y$ representan la distancia focal de la cámara en términos de dimensiones de píxeles en la dirección x e y respectivamente, $\tilde{\mathbf{x}}_0 = (x_0, y_0)$ es el punto principal con coordenadas $x_0 = m_x p_x$ y $y_0 = m_y p_y$ y el parámetro s representa la distorsión de la cámara.

$$K = \begin{bmatrix} \alpha_x & s & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.23)$$

2.6.3 Transformación proyectiva y estimación de la homografía

Al tomar dos imágenes de una misma escena desde diferentes puntos de vista, existe una importante relación proyectiva entre éstas y la escena. Estas imágenes, pueden haber sido obtenidas mediante la misma cámara (tomando la fotografía desde dos puntos de vistas diferentes), o mediante dos cámaras posicionadas en diferentes lugares observando el mismo punto.

Una transformación proyectiva (también conocida con los términos equivalentes: colineación u homografía) es un mapeo invertible h de \mathbb{P}^2 a sí mismo, de tal manera que tres puntos x_1, x_2, x_3 están sobre la misma línea si y solo si $h(x_1), h(x_2), h(x_3)$ también lo están.

Definición 1. *Un mapeo $h: \mathbb{P}^2 \rightarrow \mathbb{P}^2$ es una transformación proyectiva si y solo si existe una matriz de 3×3 (H) tal que para cada punto en \mathbb{P}^2 representado por un vector \mathbf{x} se cumple que $h(\mathbf{x}) = H\mathbf{x}$.*

Una definición algebraica se expresa en la Def. 1, de la que se puede interpretar que:

- cualquier punto en \mathbb{P}^2 es representado por un vector homogéneo \mathbf{x} de 3 componentes y $H\mathbf{x}$ es un mapeo lineal en coordenadas homogéneas,
- cualquier transformación lineal invertible de coordenadas homogéneas es una transformación proyectiva.

Si consideramos el conjunto de puntos (x_i, y_i, z_i) perteneciente a una primera imagen y sabemos que mapean a un conjunto de puntos (x'_i, y'_i, z'_i) en la segunda imagen (ambos en coordenadas homogéneas), la relación entre las dos imágenes es una homografía si se cumple la ecuación

$$\begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} = H \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} \implies \begin{bmatrix} x'_i \\ y'_i \\ z'_i \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix}. \quad (2.24)$$

Se puede interpretar de la ecuación (2.24) que la homografía H mapea coordenadas x en una imagen a coordenadas x' en otra. Es decir, la proyección a través de rayos que pasan a través de un punto en común (el centro de proyección), definen un mapeo de un plano a otro. Este mapeo punto a punto, preserva el mapeo de una línea en un plano a otra línea en otro plano, pero el paralelismo no es necesariamente preservado. Si se considera un plano que pase a través del centro de proyección (Fig. 2.17), el mismo intersectaría los planos π y π' mapeando líneas de un plano a otro. Dado este mapeo, la proyección central es una transformación proyectiva y puede ser representado por un mapeo lineal en coordenadas homogéneas de la forma $\mathbf{x}' = H\mathbf{x}$.

La matriz H contiene nueve elementos y resulta ambigua al escalarla (el uso de coordenadas homogéneas significa que cualquier múltiplo de la homografía, tendría el mismo efecto). Agregando un factor de escala s , la expresión (2.24) queda como

$$\begin{bmatrix} sx'_i \\ sy'_i \\ s \end{bmatrix} = H \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix}. \quad (2.25)$$

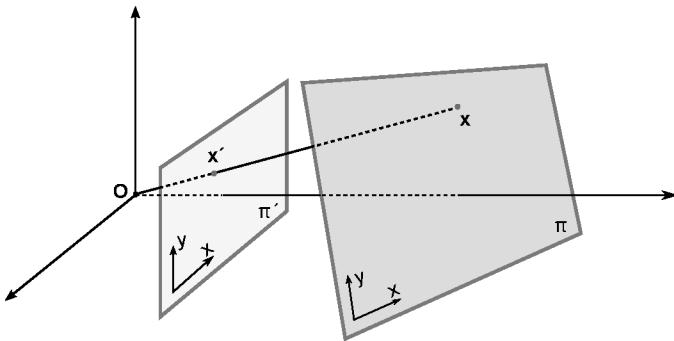


Figura 2.17: La proyección central mapea puntos de un plano a puntos en otro plano. (Figura adaptada de [23]).

Una vez que se a logrado obtener H , todos los puntos en una vista pueden ser convertidos a la segunda vista usando la relación 2.25.

No es común conocer las correspondencias entre puntos de forma exacta, incluso se pueden tener pares de correspondencias que no son válidos o tener más de 4 pares de correspondencias en cuyo caso la matriz H puede resultar incorrecta. Por ello, para estos casos, se procede mediante una *estimación* de la homografía.

Estimación de la homografía

Si se conocen exactamente cuatro correspondencias (tres de ellas no colineales), se puede encontrar una solución exacta de la matriz H . Esta solución es denominada “solución mínima” y resulta importante ya que define el subconjunto mínimo requerido para los algoritmos de estimación robusta como RANSAC. En la práctica, generalmente se conocen más de cuatro correspondencias (válidas o no) y estas correspondencias no resultan totalmente compatibles con una transformación proyectiva. Por ello, nos encontramos con la tarea de determinar la “mejor” transformación a partir de los datos, lo cual se convierte en la búsqueda de la transformación H que minimice una función de costo.

Transformación lineal directa. La transformación lineal directa o DLT (del inglés, Direct Linear Transform) es un algoritmo lineal para determinar H dado un conjunto de 4 puntos 2D correspondientes $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$. La transformación viene dada por la ecuación $\mathbf{x}'_i = H\mathbf{x}_i$ donde los vectores están dados en coordenadas homogéneas, es decir que \mathbf{x}'_i y $H\mathbf{x}_i$ no son iguales (tienen la misma dirección pero pueden diferir en magnitud por un escalar que no sea

cero). Denotando $\mathbf{x}_i = (x_i, y_i, w_i)^\top$, $\mathbf{x}'_i = (x'_i, y'_i, w'_i)^\top$ y $H = \begin{pmatrix} \mathbf{h}^{1\top} \\ \mathbf{h}^{2\top} \\ \mathbf{h}^{3\top} \end{pmatrix}$, donde $\mathbf{h}^{i\top}$ es un vector fila que denota la fila i de H , mediante algunos cálculos [23, p. 89] y teniendo presente que el sistema $\mathbf{x}'_i = H\mathbf{x}_i$ posee tres ecuaciones, de las cuales solo dos son independientes (la tercera fila es obtenida para un factor de escala) se obtiene

$$\begin{bmatrix} \mathbf{0}^\top & -w'_i \mathbf{x}_i^\top & y'_i \mathbf{x}_i^\top \\ w'_i \mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i \mathbf{x}_i^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}. \quad (2.26)$$

El sistema 2.26 se puede escribir como $A_i \mathbf{h} = \mathbf{0}$ donde A_i es la matriz de 2×9 y \mathbf{h} el vector de 9×1 de la ecuación (2.26). Para cada correspondencia de puntos se obtienen 2 ecuaciones, por lo que 4 correspondencias (forman un sistema de 8×8) resultan suficientes para resolver los 8 grados de libertad de H (recordar que se determina para un factor de escala), siempre que se tengan al menos 3 puntos no colineales.

Si se tienen más de cuatro correspondencias $\mathbf{x}_i \leftrightarrow \mathbf{x}'_i$, el conjunto de ecuaciones $A\mathbf{h} = \mathbf{0}$ tiene muchas soluciones. Además, si estas correspondencias no son exactas, la solución encontrada no será adecuada, por lo que el problema se convierte en estimar la homografía minimizando una función de error [23].

Existen algoritmos como *least median of squares* (LMEDS) [58] o el de *random sample consensus* (RANSAC)[17, 23] que pueden hallar la mejor solución aproximada minimizando el error y a su vez, tratando de detectar cuáles son los supuestos valores de coincidencias válidos (del inglés, inliers) y los espurios (del inglés, outliers). Además son capaces de utilizar más de cuatro correspondencias para obtener una solución más exacta. El método LMEDS, a diferencia del RANSAC, no necesita un umbral para distinguir entre las correspondencias válidas y las espurias, sin embargo, LMEDS sólo funciona correctamente cuando hay más de un 50 % de valores válidos [23, 32].

Homografía con RANSAC. El objetivo que se plantea aquí, es el de determinar un conjunto de correspondencias (eliminando valores espurios) de forma que la homografía pueda ser estimada de manera óptima a partir de las mismas mediante la transformación lineal directa descripta.

Para explicar y entender el algoritmo RANSAC, primero se presenta un problema que puede ser fácilmente visualizado el cual consiste en estimar una línea recta a partir de un conjunto de puntos 2D. Este problema, también se puede pensar como estimar una transformación afín 1D del tipo $x' = ax + b$ entre puntos correspondientes que están sobre dos líneas.

El problema, se encuentra ilustrado en la Fig. 2.18(a) (los puntos negros son válidos y los blancos son espurios) donde se puede ver que el ajuste por mínimos cuadrados de los puntos (regresión ortogonal), es afectado de forma severa por los valores espurios. Así, dado un conjunto de puntos 2D, se debe buscar la línea que minimiza la suma de los cuadrados de las distancias perpendiculares, de tal forma que ninguno de los puntos válidos se desvíe de la línea por más de t unidades. Aquí, se presentan dos inconvenientes: la línea se debe ajustar a los datos y se deben clasificar los puntos en válidos o espurios.

Existen muchos tipos de algoritmos robustos y la selección de uno u otro depende de la proporción de los valores espurios [23]. Aquí se describe el estimador robusto RANSAC que es capaz de hacer frente a una gran proporción de valores atípicos.

Se empieza mediante la selección aleatoria de dos puntos; estos puntos definen una línea. El *soporte* para esta línea, es medido por la cantidad de puntos que se encuentran bajo un umbral de distancia. Luego, la selección aleatoria es repetida varias veces y la línea con mayor soporte es considerada como el mejor ajuste. Los puntos que están por debajo del umbral de distancia son considerados válidos y constituyen el *conjunto consensuado*. Como se observa en la Fig. 2.18(b), si un punto no es válido, la línea posee menos soporte lo cual favorece a un mejor ajuste. Por ejemplo, el soporte para la línea $\langle \mathbf{a}, \mathbf{b} \rangle$ en la Fig. 2.18(b) es 10, mientras que para la línea $\langle \mathbf{a}, \mathbf{d} \rangle$ en el que los puntos de ejemplo son vecinos, es 4. Consecuentemente y a pesar que ambas líneas contienen valores válidos, se selecciona la línea $\langle \mathbf{a}, \mathbf{b} \rangle$ por tener mayor soporte.

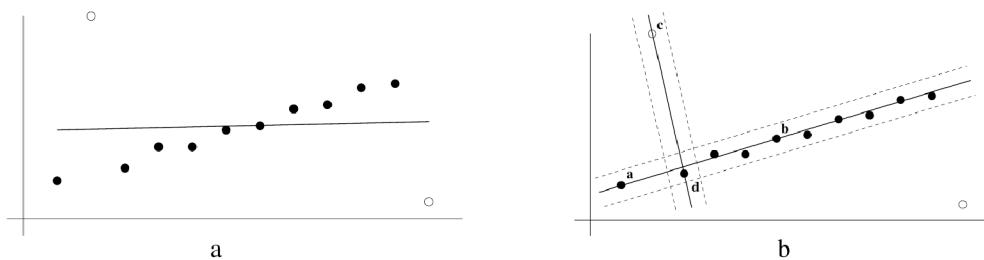


Figura 2.18: Estimación robusta de una línea. (a) Ajuste por mínimos cuadrados. (b) Algoritmo RANSAC (las líneas punteadas denotan el umbral de distancia). (Figura tomada de [23]).

Generalizando lo anteriormente descripto, podemos decir que deseamos ajustar un *modelo* (en el ejemplo, una línea) a los datos y la muestra aleatoria consiste en un subconjunto mínimo de datos (2 puntos en el ejemplo)

suficientes para determinar el modelo. En el caso en el que el modelo es una homografía plana y los datos son un conjunto de correspondencias 2D, el subconjunto mínimo está compuesto por cuatro correspondencias.

El algoritmo RANSAC tiene como objetivo el ajuste robusto de un modelo para un conjunto de datos S que contiene valores espurios. RANSAC, requiere de una cantidad mínima de puntos s para instanciar los parámetros libres del modelo y sigue los siguientes pasos:

1. Se seleccionan aleatoriamente un conjunto de puntos $s \in S$ y se instancia el modelo para este subconjunto,
2. Se determina el conjunto de puntos S_i que se encuentran dentro de un umbral de distancia t respecto al modelo. El conjunto S_i es el conjunto consensuado de muestras y define los valores válidos de S .
3. Si la cantidad de elementos válidos en S_i es mayor que un umbral T , se estima nuevamente el modelo usando todos los puntos de S_i y se termina el algoritmo.
4. Si la cantidad de elementos válidos en S_i es menor que T , se selecciona un nuevo subconjunto y se repiten los pasos arriba mencionados.
5. Luego de N iteraciones, el conjunto consensuado S_i con mayor cantidad de elementos es seleccionado y se estima el modelo usando este conjunto de datos.

donde t , N y T son:

- **Umbral de distancia t :** Existen diferentes medidas de distancias, como la de transferencia del error simétrico, la del error de Sampson y la del error de reproyección que resulta la más adecuada en el caso de la estimación de la homografía entre dos imágenes [23]. La formula del error de reproyección, viene dada por la ecuación $d_{\perp}^2 = d(\mathbf{x}, \hat{\mathbf{x}})^2 + d(\mathbf{x}', \hat{\mathbf{x}'})^2$ donde $\mathbf{x} \leftrightarrow \mathbf{x}'$ es la correspondencia de puntos y $\hat{\mathbf{x}'} = H\hat{\mathbf{x}}$ es la correspondencia exacta. Aquellos puntos que cumplan la condición $d_{\perp}^2 > t^2$ son considerados como espurios y los que cumplan con $d_{\perp}^2 \leq t^2$ son considerados válidos. Es usual que en la práctica el valor de t sea seleccionado empíricamente, sin embargo si se asume que la medida del error tiene una distribución gaussiana con media 0 y desviación estándar σ el valor de t puede calcularse. Por ejemplo, para el caso de la homografía, con una probabilidad del 95 % de que la correspondencia sea válida se utiliza $t^2 = 5.99\sigma^2$. Para más destalles se puede consultar [23, p. 119].
- **Tamaño del Conjunto consensuado T :** la regla es terminar el al-

goritmo si el tamaño del conjunto consensuado es similar a la cantidad de valores válidos que se cree que está presente en el conjunto de datos, dada la premisa de la proporción de valores espurios que, por ejemplo, para n puntos es $T = (1 - \epsilon)n$.

- **Cantidad de muestras N :** es computacionalmente innecesario e ineficiente probar cada muestra posible. Por eso, se selecciona una cantidad de muestras N lo suficientemente alta para asegurar con una probabilidad p , que por lo menos una de las muestras aleatorias de s puntos, no contiene puntos espurios. Usualmente se establece a $p = 0.99$ [23]. Si suponemos que w es la probabilidad que cualquier punto seleccionado de los datos es un punto válido, implica que $\epsilon = 1 - w$ representa la probabilidad que sea un punto espurio. Luego, se necesitan N selecciones, cada una de s puntos, donde $(1 - w^s)^N = 1 - p$. Así, N queda definida como:

$$N = \log(1 - p) / \log(1 - (1 - \epsilon)^s). \quad (2.27)$$

Para $s = 4$ muestras, con una proporción de valores espurios del $\epsilon = 50\%$, son necesarias 72 muestras para asegurarse con una probabilidad de $p = 0.99$, que al menos una de las muestras no contiene valores espurios. Como se observa en la ecuación (2.27), la cantidad de muestras está relacionada con la proporción de valores espurios, de forma que la cantidad de muestras requeridas deben ser menor que la cantidad de valores espurios. Consecuentemente, el costo computacional de las muestras es aceptable aún cuando la cantidad de valores espurios resulta elevado. Por otro lado, la cantidad de muestras se incrementa con la cantidad mínima del subconjunto (para un ϵ y p dado). De aquí, se puede decir que usar más del mínimo que se requiere (4 o más puntos en el caso de la homografía), contribuirán a una mejor estimación y el soporte determinado reflejará con mayor precisión al verdadero soporte. Pero se debe tener en cuenta que esta ventaja, incrementa el costo computacional debido al incremento del número de muestras.

Determinación adaptativa de la cantidad de muestras Por lo general ϵ es desconocido por lo que en dicho caso, el algoritmo es inicializado usando el peor caso de estimación de ϵ , y esta estimación es actualizada a medida que se encuentran más conjuntos consistentes. Por ejemplo, si se supone que el peor caso es $\epsilon = 0.5$ y del conjunto consensuado se encuentra un 80 % de datos como válidos, la estimación actualizada es $\epsilon = 0.2$.

La idea de “probar” los datos mediante el conjunto consensuado puede ser aplicada repetidamente de forma de determinar adaptativamente la cantidad de muestras N . Si tenemos en cuenta el ejemplo mencionado en el párrafo anterior, la peor estimación $\epsilon = 0.5$ determina el valor inicial de N en base a la ecuación (2.27). Cuando el conjunto consensuado contiene más del 50 % de datos encontrados, sabemos que hay por lo menos esa misma cantidad de valores válidos. Esta estimación actualizada de ϵ determina un N reducido de acuerdo a la ecuación (2.27). Esta actualización es repetida para cada muestra y cada vez que es encontrado un conjunto consensuado con ϵ menor que el ya estimado, la cantidad N también disminuye. El algoritmo termina tan pronto como N muestras han sido analizadas.

Los pasos que se siguen para la determinación adaptativa de la cantidad de muestras como así también la proporción de valores espurios de cada conjunto consensuado, son los siguientes:

1. $N = \infty$, $contador_muestras = 0$.
2. Mientras $N > contador_muestras$ repetir:
 - Seleccionar una muestra y contar el numero de valores válidos.
 - Establecer $\epsilon = 1 - \frac{\text{cantidad de valores válidos}}{\text{cantidad total de puntos}}$
 - Establecer N con ϵ según la ecuación (2.27) para $p = 0.99$
 - Incrementar $contador_muestras$ por 1
3. Fin.

El algoritmo RANSAC, es aplicado a las potenciales correspondencias para estimar la homografía como así también las correspondencias válidas consistentes con dicha estimación. Como se ha mencionado, cuatro correspondencias son suficientes para determinar la homografía (siempre que no hayan tres colineales), sin embargo, se puede obtener una mejor estimación usando todos los valores válidos (en vez de sólo cuatro), además de obtener un conjunto con mayor cantidad de correspondencias válidas de forma que la homografía resulte más precisa. Esto, se logra con la minimización de la función del error de reproyección mediante el algoritmo Levenberg-Marquardt (método iterativo con una variación al método iterativo de Gauss-Newton [23, p. 600]). Todos los pasos descriptos para la estimación de la homografía se describen a continuación:

Calculo de la homografía 2D entre dos imágenes.

1. Se asume que se tiene un conjunto de potenciales correspondencias entre las imágenes.
2. Se repite para N muestras (N determinado con el algoritmo adaptativo)
 - Se seleccionan aleatoriamente cuatro correspondencias y se calcula la homografía H .
 - Se calcula la distancia d_{\perp} para cada correspondencia.
 - Se calcula la cantidad de valores válidos consistentes con H entre la cantidad de correspondencias para las cuales $d_{\perp} < t = \sqrt{5.99}\sigma$ píxeles.
3. Re-estimar H a partir de todas las correspondencias clasificadas como válidas, mediante la minimización de la función acumulada de error de reproyección: $\sum_i d(\mathbf{x}_i, \hat{\mathbf{x}}_i)^2 + d(\mathbf{x}'_i, \hat{\mathbf{x}}'_i)^2$ sujeto a $\hat{\mathbf{x}}'_i = \hat{H}\hat{\mathbf{x}}_i \forall i$ usando el algoritmo de Levenberg-Marquardt.

CAPÍTULO 3

Método propuesto

En este capítulo se presenta el diseño del método propuesto sustentándose en el marco teórico introducido en el capítulo anterior. Se describen cada una de las etapas involucradas en el método, planteándose algunas técnicas para el realce de detalles y mejora de la iluminación de la imagen. Además, se presentan estrategias utilizadas para mejorar el desempeño en el tiempo de procesamiento, como así también para obtener una correcta detección del objeto en la escena.

3.1 Diseño del método

El método para el reconocimiento y seguimiento de un objeto en una secuencia de video que se propone en este trabajo consta de dos fases, cada una de ellas compuesta por varios procesos internos. Las fases pueden observarse en los diagramas de las Figs. 3.1 y 3.2 y las hemos definido como:

- **Configuración:** la configuración se realiza una sola vez al inicio del algoritmo y es en la que se registra la imagen que posteriormente se detectará en el flujo de video. Esta etapa se encuentra representada en la Fig. 3.1, la cual consta de varios procesos: primeramente, una conversión a escala de grises con una fórmula perceptualmente ponderada, luego se realiza un pre-procesamiento de iluminación y realce

de detalles y finalmente, se procede con la extracción y descripción de características sobre la imagen.

- **Ejecución:** esta etapa contempla la captura de un frame del flujo de video proporcionado por la cámara web, el procesamiento para detectar el objeto registrado en la etapa anterior y la posterior superposición de un objeto virtual para enriquecer la realidad. Todo esto está representado en la Fig. 3.2, la cual esta compuesta por diferentes procesos, entre los cuales se encuentran: la detección de movimiento, la extracción y descripción de características, la búsqueda de correspondencias (que usa como entrada la salida del proceso de configuración), la detección de la homografía y las validaciones posteriores para finalmente, en caso que la imagen patrón haya sido detectada, obtener un frame con el objeto virtual superpuesto. Estos procesos y otros que por cuestiones de brevedad no se mencionan aquí, serán descriptos en las secciones que componen este capítulo.

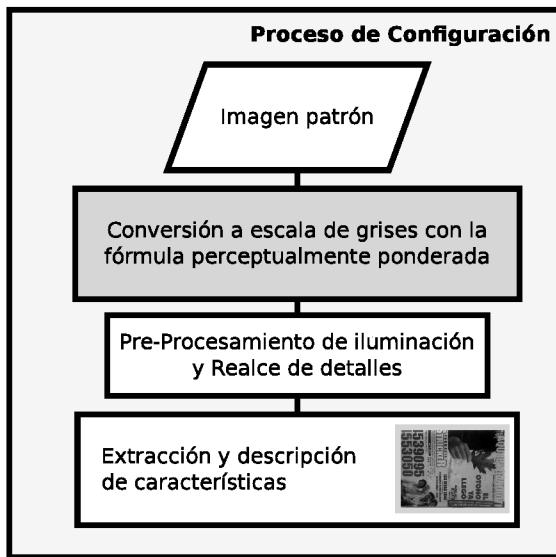


Figura 3.1: Diagrama de flujo de la etapa de Configuración.

En la fase de configuración, la imagen de un objeto conocido es convertida a escala de grises mediante una fórmula perceptualmente ponderada de la forma que se expresa en la Sec. 3.2. Tras ello, se aplica un proceso para mejorar la iluminación y resaltar detalles con el objetivo de mejorar la cantidad de puntos detectados, sobre todo en los casos en que la iluminación es baja. Luego, se procede con el método para la detección de características para obtener los puntos junto con sus descriptores asociados. Este proceso, tiene

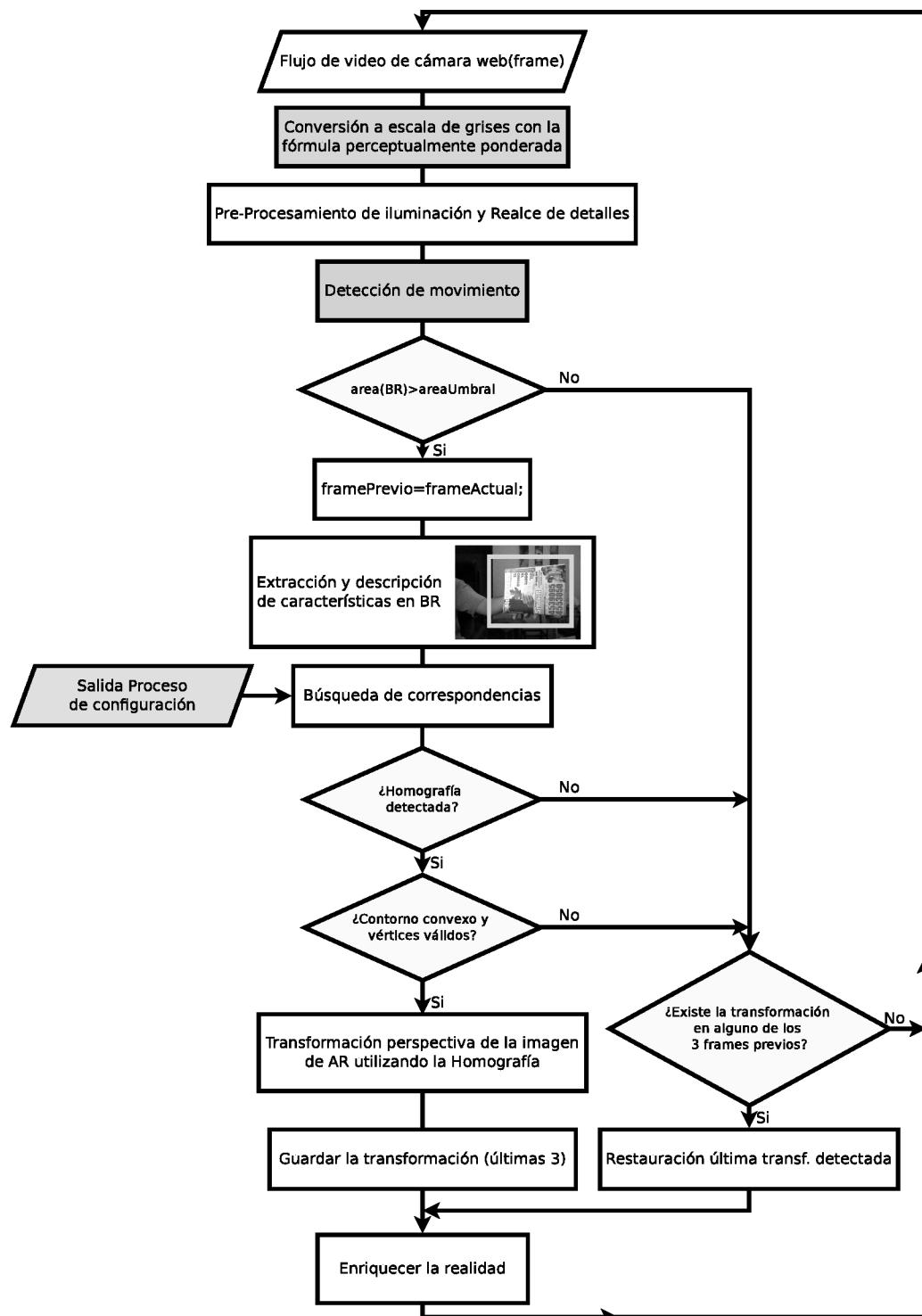


Figura 3.2: Diagrama de flujo de la etapa de ejecución del método propuesto.

como salida un vector característico asociado a cada punto clave detectado, y es utilizado en la fase de ejecución para la detección del objeto.

En lo que respecta a la fase de ejecución, se aplica un proceso similar al anteriormente mencionado y se lleva a cabo sobre la imagen del flujo de video donde se busca el objeto. Un segundo paso consiste en utilizar los puntos claves y sus descriptores para la búsqueda de correspondencias de características. El proceso consiste en crear pares entre los descriptores de las imágenes pertenecientes a las dos fases mencionadas (configuración y ejecución) de acuerdo a sus similitudes, utilizando una búsqueda del vecino más cercano. Además, con el objetivo de aumentar la confianza respecto a los pares de correspondencias, se aplica un paso de eliminación de pares irrelevantes o espurios. Una vez que se obtiene un conjunto de correspondencias filtradas y potencialmente válidas, se estima la homografía con la ayuda de un método estadístico que le provee mayor confiabilidad en los resultados. Mediante la homografía, se obtiene un mapeo perspectivo entre la imagen patrón y la imagen del flujo de video (imagen objetivo). La existencia de la misma, establece la localización de la imagen patrón en la imagen objetivo y también especifica la transformación perspectiva que existe entre ellas. A pesar de haber aplicado un proceso para eliminar valores espurios, se presentan casos en que siguen existiendo valores inválidos de correspondencias que llevan a una mala estimación de la homografía. Por ello, se establece un criterio para el rechazo de homografías mal estimadas basado en la forma convexa del polígono formado por la proyección de las esquinas de la imagen patrón de forma similar a como se realiza en [32].

Cabe aclarar que la extracción y descripción de características mediante SURF, la búsqueda de correspondencias mediante árboles KD aleatorios y la estimación de la homografía mediante RANSAC fueron integrados en el diseño del método de forma similar a como se realizó en [32].

A continuación se presentan en detalle los procesos representados en los bloques de la Fig. 3.2.

3.2 Conversión a escala de grises con una fórmula perceptualmente ponderada

Una imagen puede poseer varios canales de colores. En nuestro caso, la imagen que proviene del flujo de video es una imagen de tres canales (RGB).

Debido a que el algoritmo utilizado para extracción de características utiliza imágenes en tono de grises de 8 bits, se aplica una conversión antes de procesar la imagen.

Para el cálculo del valor de gris se usa la fórmula perceptualmente ponderada definida en la ecuación (3.1) donde R , G y B representan los canales rojo, verde y azul de la imagen de entrada respectivamente, e I representa la imagen resultado convertida a tonos de grises.

$$I = 0.299R + 0.587G + 0.114B. \quad (3.1)$$

3.3 Mejoras en la iluminación y realce de detalles

Como se ha mencionado el algoritmo SURF utilizado para la extracción y descripción de características, resulta variante a cambios de iluminación. Debido a ello, se plantean alternativas para aumentar el contraste de la imagen. Además, se ha notado un efecto de borroneado y/o detección de puntos insuficientes sobre todo en situaciones con imágenes en condición de iluminación baja, por lo que se han planteado algunas posibles estrategias para tratar de contrarrestar dicho problema.

Para esta etapa (“Pre-Procesamiento iluminación y detalles” en la Fig. 3.2) se plantearon algunas alternativas simples de procesamiento en el dominio espacial: transformación logarítmica, ecualización de histograma, filtrado con pasa altos, filtrado de alta potencia y una combinación de ecualización y posteriormente filtrado de alta potencia. La elección de éstas técnicas se fundamentó principalmente en su velocidad de cómputo. La discusión de los resultados será presentada en la sección 4.2.1.

3.4 Detección de la región de interés

En esta sección, se proponen algunos procesos orientados a obtener un menor tiempo de procesamiento y consecuentemente un menor tiempo de ejecución.

Como es sabido, tanto el uso del método SURF para la extracción de características como la búsqueda de coincidencias entre imágenes, conllevan un

gran tiempo de procesamiento, el cual se convierte en un factor clave a la hora de lograr fluidez en la reproducción del flujo de video [7, 18, 33, 42]. Para atacar este problema, se propone detectar la región de interés (parte cambiante del flujo de video) y así aplicar el procesamiento sólo a una región de la imagen capturada. Para ello, se utilizan diferentes técnicas de procesamiento de imágenes en el siguiente orden: diferencia de imágenes, umbral, erosión, dilatación y detección del rectángulo delimitador mínimo. Estas herramientas, fueron elegidas debido a su simplicidad y al bajo tiempo de procesamiento que requieren, de forma de no incrementar significativamente el tiempo de proceso total del algoritmo.

El objetivo entonces, es detectar la parte cambiante de la imagen entre un ciclo y el siguiente para realizar la extracción de características en la zona de la escena que ha cambiado, asumiéndose un sistema en el que la cámara web se encuentra fija y lo que se mueve es el objeto en la escena. Para esto, se llevan a cabo una serie de pasos que se observan en la Fig. 3.3, a fin de determinar un área que contenga el cambio de la imagen. A continuación, se describen cada una de las operaciones realizadas, cuyos resultados parciales pueden verse en la Fig. 3.4.

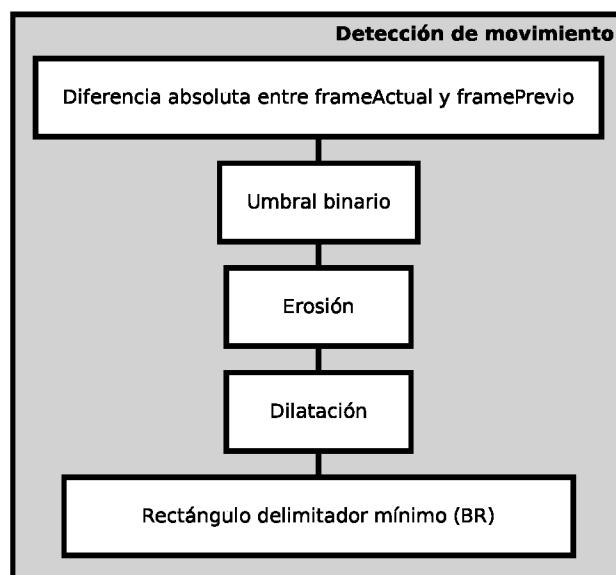


Figura 3.3: Detalle del proceso de detección de movimiento introducido en el diagrama 3.2.

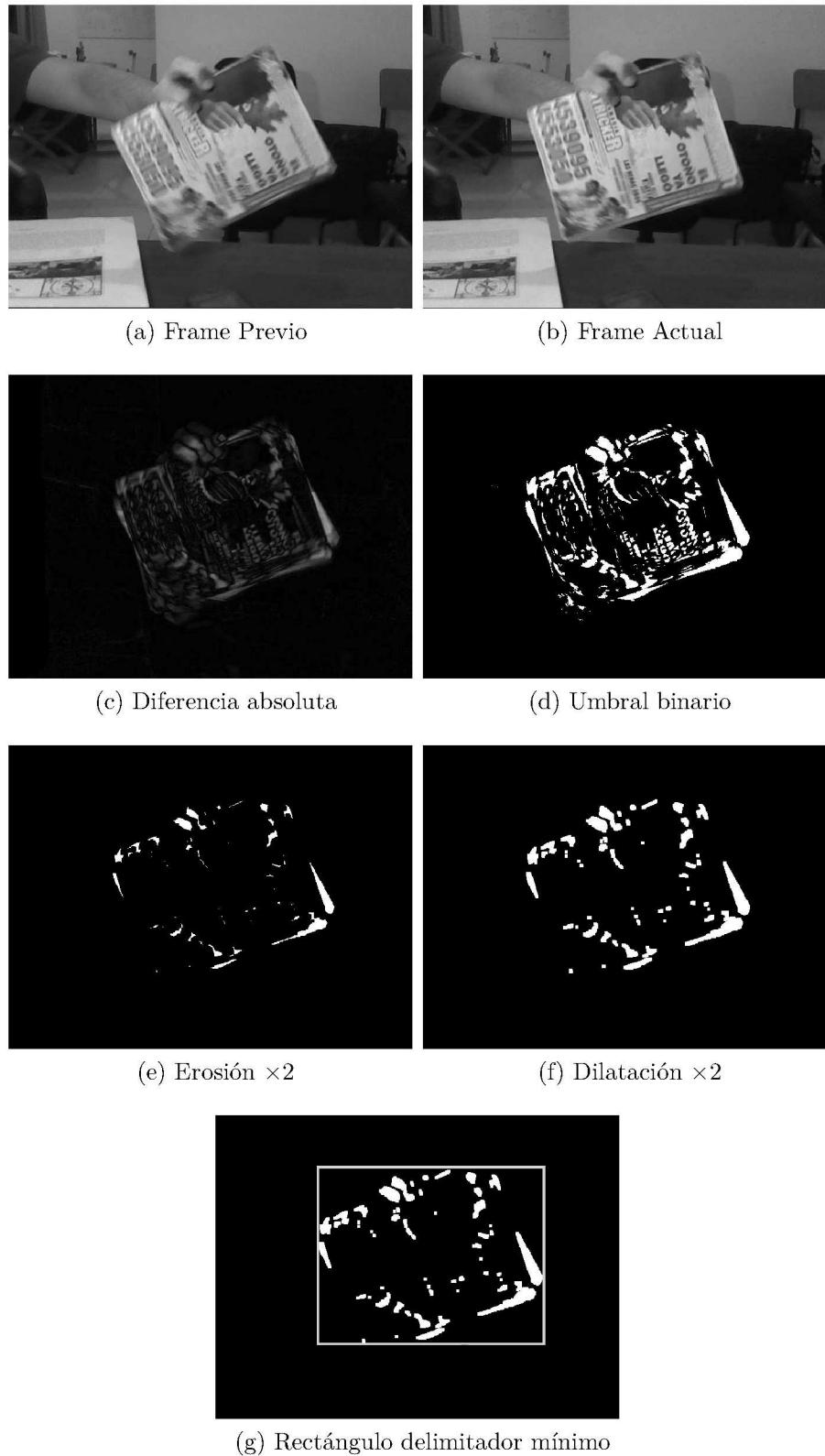


Figura 3.4: Resultado de los procesos aplicados para la detección de movimiento.

Diferencia absoluta entre frame actual y frame previo

La diferencia de imágenes es utilizada en el estudio del movimiento para detectar el cambio producido entre imágenes captadas en instantes de tiempo diferentes. Aquí, se aplica con el objetivo de distinguir el cambio en la imagen cuadro a cuadro y consiste en calcular el valor absoluto de la diferencia entre el cuadro F del flujo de video capturado en el tiempo t (actual) Fig. 3.4b y el capturado en $t - 1$ (anterior) Fig. 3.4a:

$$D(x, y) = |F_t(x, y) - F_{t-1}(x, y)| \quad \forall x, y \in F. \quad (3.2)$$

El resultado de la operación es una imagen D (Fig. 3.4c) en la que se observa la parte que ha cambiado de una imagen a otra. Cabe aclarar que se utiliza el valor absoluto de la diferencia, ya que usar la resta produce resultados fuera del rango de valores que se utilizan en las imágenes.

Si bien se utiliza la ecuación (3.2) para la diferencia de imágenes, el “frame anterior” no representa siempre el previo al actual procesado (nótese la asignación $framePrevio = frameActual$ dentro del condicional $area(BR) > areaUmbral$ en la Fig. 3.2). Es decir que la diferencia, se hace entre el fotograma actual y el último en el que se detectó movimiento. Esta decisión, se ve fundamentada en el hecho que la diferencia entre frames sucesivos, donde el desplazamiento es muy pequeño, genera información espuria. Cabe agregar que ésto también beneficia al tiempo promedio de operación del sistema, ya que se ignora el cómputo de frames irrelevantes para la aplicación.

Como se puede observar en la Fig. 3.4c, existe presencia de ruido y “patrones speckle” en partes de la imagen donde no hay movimiento. Estos efectos indeseados, se tratarán de eliminar mediante una combinación de técnicas que se describen a continuación.

Umbral Binario

A partir de la imagen diferencia D obtenida anteriormente, se procede con la aplicación de un umbral binario de la forma que se expresó en la ecuación (2.3) para remover valores de intensidad bajos (ruido y “patrones speckle” pueden ser observados debajo de la esquina inferior izquierda de la revista en la Fig. 3.4c). Se asigna el valor $s_{max} = 255$ a los píxeles que superen el umbral con un valor definido empíricamente ($u = 50$) y 0 a los demás píxeles, obteniéndose como resultado una nueva imagen s , ver Fig. 3.4d.

A pesar de haber eliminado gran parte del ruido en la imagen, en la práctica se dan casos en los que puntos aislados continúan apareciendo en

la imagen. Variar el límite (u) establecido para la umbralización no brinda una solución a este problema, ya que al aumentar u se elimina información importante y al disminuirlo, aumenta la información espuria. Es por esto que fue necesario proponer dos procesos que se mencionan a continuación.

Erosión y dilatación

Para eliminar los puntos aislados que no hayan sido removidos anteriormente, se propone aplicar una operación de erosión. Posteriormente, se aplica una operación de dilatación para recuperar el tamaño aproximado de los objetos de interés.

Para llevar a cabo la erosión y la dilatación, se utilizó un kernel de 3×3 píxeles y cada una de las operaciones fue aplicada dos veces (dos erosiones y luego 2 dilataciones). En el caso que haya habido movimiento, el resultado final es una imagen negra con un sector blanco, que representa la zona cambiante de la imagen. El resultado se puede observar en la Fig. 3.4e y 3.4f.

Rectángulo delimitador mínimo

En este paso se busca detectar el rectángulo más pequeño que encierra a todos los puntos cuyo valor no es negro. Al utilizar un rectángulo delimitador mínimo (del inglés, bounding rect o BR), se pueden descartar regiones de la imagen que no necesitamos procesar en el algoritmo. El resultado final de la operación se puede observar con un rectángulo de color verde en la Fig. 3.4g.

3.4.1 Área del rectángulo delimitador mínimo

La detección de la parte cambiante de la imagen en el flujo de video, determina un BR sobre el que se ejecuta el procesamiento. En la práctica, se pueden identificar situaciones particulares que involucran la determinación del BR:

1. ¿Qué sucede si no se detecta movimiento?
2. ¿Qué sucede si el BR del área detectada es demasiado pequeña?

Para dar respuesta a estos interrogantes, se ha propuesto un proceso en el algoritmo que censa el tamaño del BR mediante un bloque condicional presentado en el diagrama de la Fig. 3.2, cuya expresión es: $area(BR) >$

areaUmbral, donde *area*(\cdot) es una función que calcula el área en píxeles sobre el argumento y *areaUmbral* el valor en píxeles que el área debe superar.

Respecto al primer interrogante, si no existe movimiento tampoco existe *BR* y por ende la condición de área no se satisface. Entonces, el flujo del procesamiento se deriva al procedimiento establecido en la Sec. 3.9.

Para la segunda pregunta, se hizo una búsqueda del valor óptimo de *areaUmbral* para la configuración general de la escena que se utiliza. El mismo fue establecido de manera empírica en 10000 píxeles que fue el valor con el que se obtuvieron mejores resultados. Ésto limita la detección cuando las esquinas proyectadas de la imagen patrón resultantes de aplicar la homografía, forman un área relativamente pequeña (se puede pensar en un cuadrilátero de 100×100 sobre la ventana de 640×480). Obtener la cantidad suficiente de características en objetos de estos tamaños es una limitante, que presenta el dispositivo de adquisición de imágenes utilizado. De esta manera, si el área de *BR* no supera el umbral establecido, se considera a la región como “demasiado pequeña” y el procesamiento al igual que en el caso anterior, se deriva a la acción que se describe en la Sec. 3.9.

3.5 Extracción y descripción de características

En la etapa de extracción y descripción de características, se detectan puntos claves o de interés en la imagen que se está analizando con el método que se describió en la Sec. 2.3. Dicha detección se realiza en la subimagen determinada por *BR*. Esta etapa involucra básicamente dos pasos:

1. Detectar los puntos claves en la imagen.
2. Extraer un vector de características (64 elementos) para cada punto detectado en la imagen.

Para llevar a cabo dichos pasos, se ha utilizado un algoritmo descripto en [6] para el cual se especifican los siguientes parámetros:

- **nOctaves:** identifica el número de octavas que se utilizan en la búsqueda de puntos claves. Se estableció en el valor 4 en el que se obtuvieron resultados satisfactorios [6].

- **nOctaveLayers:** es el número de filtros o capas utilizado dentro de cada octava. Se estableció en el valor 2 el cual resultó adecuado y es recomendado en [6].
- **hessianThreshold:** es un valor de umbral que se utiliza para eliminar máximos locales detectados con el determinante del hessiano descripto en la Sec. 2.3. Aquellas características cuyo determinante del hessiano superan dicho umbral son extraídas. Los valores para este parámetro, dependen del promedio local del contraste, la nitidez y los detalles de la imagen. Estos valores serán tratados en el capítulo 4. Un ejemplo de los efectos de este parámetro sobre una misma imagen puede ser observado en las Fig. 3.5a y 3.5b, donde la línea verde representa la orientación para el punto clave detectado (marcado en color azul) y la escala viene representada por el círculo rojo centrado en cada punto clave. En la Fig. 3.5a se fijó el umbral en 800 (más puntos detectados), mientras que en la Fig. 3.5b se utilizó el valor 3000 con lo cual se puede ver una disminución en la cantidad de los puntos detectados.

3.6 Correspondencia entre puntos claves

Luego de detectar los puntos claves y sus descriptores tanto en la imagen patrón como en la imagen objetivo, se procede con la búsqueda de coincidencias entre ambas imágenes. Para ello, se utiliza la búsqueda del vecino más cercano, aquí se utilizan dos vecinos al punto clave detectado como se describe en la sección 2.5. Además, se utiliza la técnica de árboles KD para acelerar la búsqueda y el filtrado de correspondencias no válidas, que se describió en la sección 2.5.2.

De esta forma se obtiene un conjunto de pares de puntos entre la imagen patrón y la imagen objetivo. Un ejemplo puede ser observado en la Fig. 3.6, donde los parámetros descriptos en la Sec. 2.5.2 se establecieron a $hessianThreshold = 2000$ y $\varepsilon = 0.8$. Las líneas entre las imágenes representan las correspondencias detectadas entre los puntos claves de la imagen patrón (izquierda) y la imagen objetivo (derecha), correspondiente al flujo de video en un instante de tiempo. Las correspondencias espurias que se pueden observar (por ejemplo, la letra “o” en la palabra “otoño”), pueden ser luego descartadas mediante el uso de RANSAC en la estimación de la homografía.



(a)



(b)

Figura 3.5: Extracción de características para la imagen patrón con dos valores de umbral aplicado sobre el Hessiano.



Figura 3.6: Correspondencias de puntos entre imágenes patrón y objetivo antes de estimar la homografía con RANSAC.

Las funcionalidades de búsqueda del vecino más cercano, junto con el armando de árboles para una indexación más rápida fueron realizadas mediante la interfase FLANN¹ provista por OpenCV.

3.7 Detección de homografía

La transformación proyectiva u homografía es la proyección perspectiva entre dos imágenes y es derivada de las posiciones de los puntos claves en la imagen patrón y los puntos coincidentes detectados en la imagen objetivo, de acuerdo a como se ha explicado en la Sec. 2.6.3. En este trabajo, la existencia de la homografía entre los puntos en la imagen patrón y los de la imagen objetivo significa que el objeto (representado por la imagen patrón) existe en la imagen objetivo.

Como se ha mencionado en la Sec. 2.6.3, existen diferentes opciones para llevar a cabo la estimación. En el presente trabajo, se usará el algoritmo RANSAC, ya que este brinda mejores resultados cuando hay varios puntos de correspondencia espurios, lo que es común en el caso de localizar objetos planos en la escena [32]. La desventaja de usar el algoritmo RANSAC recae en que, dados más de cuatro puntos, siempre se encuentra la homografía independientemente si ésta tiene sentido. Es decir, que si el objeto no se halla en la escena, pero se encuentran coincidencias que son espurias, la homografía

¹http://opencv.willowgarage.com/documentation/cpp/flann_fast_approximate_nearest_neighbor_search.html

será detectada y producirá resultados indeseables.

3.8 Detección de homografías mal estimadas

Para salvar los casos en que no se detecta el objeto o la transformación con la homografía hallada presenta una morfología inválida, se aplican algunas validaciones. Las alternativas que se proponen, han permitido obtener un mejor resultado en la detección del objeto en la imagen objetivo, eliminando los falsos positivos. Para ello se utilizan dos criterios:

- Convexidad
- Distancia entre vértices

Convexidad

Este criterio es usado para rechazar transformaciones deformes que forman una figura cóncava incorrecta (forma de mariposa, véase la Fig. 3.7a). Consiste en comprobar la convexidad sobre el polígono formado por las esquinas proyectadas de la imagen patrón, luego de aplicar la transformación con la matriz de homografía. En el caso de polígonos no convexos, se está en presencia de una detección inválida y se concluye que el objeto buscado no está presente en la escena.

Distancia entre vértices

Si bien con el criterio de convexidad se han salvado algunas situaciones, se presentan casos en los que los polígonos tienen una forma convexa y aun así son inválidos, y superponer el objeto virtual con esta forma carece de sentido. Un ejemplo de esto puede apreciarse en la Fig. 3.7b.

Para salvar esta situación, se planteó una estrategia en las que las coordenadas del polígono resultante de la transformación obtenida con la matriz homográfica debían estar presentes dentro del área de visualización de la ventana o viewport. Esta idea, no resultó ser una solución adecuada, ya que se detectan casos en que el polígono es correcto, pero alguno de sus vértices se sale del viewport. Una ilustración de esto se puede apreciar en la Fig. 3.7c, donde el polígono marcado con trazo verde es correcto, pero dos vértices del

mismo (círculos rojos) se salen del viewport (marco celeste). Para detectar éstos casos, se propuso un criterio de validación de vértices del polígono:

$$(|\alpha_x - \gamma_x| < \Delta X) \vee (|\beta_x - \lambda_x| < \Delta X) \vee (|\alpha_y - \gamma_y| < \Delta Y) \vee (|\beta_y - \lambda_y| < \Delta Y), \quad (3.3)$$

donde:

- $\alpha(x, y)$, $\beta(x, y)$, $\gamma(x, y)$ y $\lambda(x, y)$ son las coordenadas de los vértices del polígono resultante de la transformación homográfica, con α opuesto a γ y β opuesto a λ ,
- x y y son las coordenadas del píxel en la imagen para el eje de las abscisas y ordenadas respectivamente,
- ΔX es la cantidad de píxeles a comprobar entre vértices opuestos en el eje de las abscisas, y
- ΔY es la cantidad de píxeles a comprobar entre vértices opuestos en el eje de las ordenadas.

Si se satisface esta condición para $\Delta X = \Delta Y = 20$ píxeles el polígono es descartado. Los valores de los delta han sido establecidos empíricamente, considerando los mejores resultados obtenidos. Cabe acotar que es posible aplicar este criterio con los valores de deltas mencionados, ya que la detección de características en un área tan pequeña está limitada por el dispositivo de adquisición de imágenes utilizado.

3.9 Condición de presencia previa

Se puede pensar que en el caso de no cumplirse la “condición de área” establecida anteriormente, es lógico reiniciar el proceso capturando una nueva imagen del flujo de video. Sin embargo, esta acción no resulta del todo adecuada por lo que se explicará a continuación. Se pueden analizar diferentes situaciones:

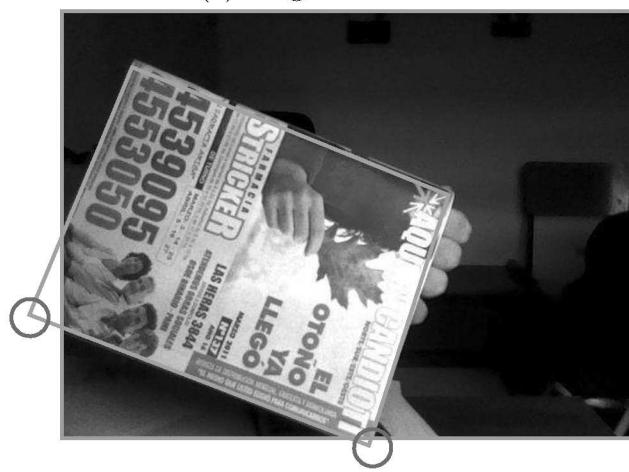
- no hay movimiento o el área del *BR* detectado es demasiado pequeña,
- la homografía no ha sido detectada,
- la homografía ha sido detectada pero la condición de la Sec. 3.8 no se satisface.



(a) Polígono cóncavo



(b) Polígono convexo



(c) Polígono convexo fuera del área de visualización de la ventana.

Figura 3.7: Polígonos resultantes de la transformación con la matriz de homografía.

Si ante estas situaciones, se reinicia el proceso de detección volviendo al inicio del diagrama de la Fig. 3.2, se deja de superponer el objeto virtual aunque el objeto aún permanezca en la escena. Aún más, si se pierde la detección del objeto entre un par de frames sucesivos, se produce un efecto de “parpadeo” del objeto virtual dibujado. Ambas situaciones claramente son indeseadas y para dar solución a las mismas, teniendo como fin brindar la mayor fluidez en el flujo de video resultante, se propone un proceso que restaura la última transformación válida sobre las últimas tres imágenes procesadas (Fig. 3.2). De lo mencionado, se deduce que ante la no detección de una transformación durante tres frames consecutivos, se deja de superponer el objeto.

En la Fig. 3.8 se esquematiza una secuencia de imágenes capturadas del flujo de video en diferentes instantes de tiempo t . El frame actual capturado se representa por f_t y los previos mediante f_{t-i} con $i = 1, 2, 3 \dots n$. El buffer de transformaciones para la restauración está compuesto por los últimos tres frames como se indica en el gráfico. En este se puede observar que en el frame actual (f_t) y en el antepenúltimo (f_{t-2}), no se han detectado las transformaciones y como resultado se restaura la transformación detectada en f_{t-1} sobre el frame actual f_t , ya que se ha detectado al menos una transformación en el buffer de transformaciones.

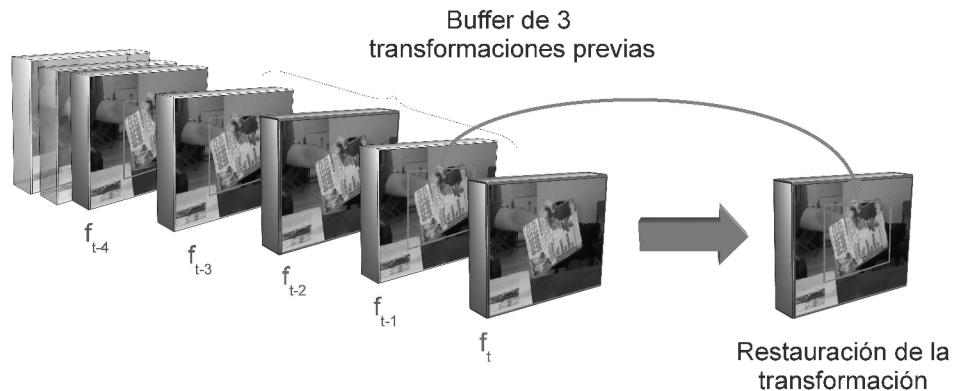


Figura 3.8: Esquema de restauración de la última transformación válida.

Para el caso donde no se detecta la transformación en ninguno de los últimos tres frames procesados, no se altera el flujo de video y se reinicia el procedimiento capturando una nueva imagen del flujo de video y comenzando así un nuevo ciclo en el método.

3.10 Realidad aumentada en el flujo de video

Esta etapa comprende la presentación del resultado final al proceso completo descripto en este capítulo. Como se observa en el diagrama de la Fig. 3.2, el último paso antes de recomenzar el proceso es “enriquecer la realidad”, que se traduce en actualizar el frame de video sobreponiendo el objeto virtual. El proceso que es llevado a cabo aquí, dependerá de la detección de la imagen patrón en el flujo de video y de la acción que se quiera ejecutar en base a ello.

Para el caso donde no se detecta la imagen patrón en el flujo de video, simplemente se debe dibujar el fotograma del flujo de video para actualizar la escena. Por el contrario, si la imagen fue detectada, se aplica la misma acción establecida anteriormente, pero además se debe superponer el objeto virtual. La realidad puede ser modificada utilizando un sinfín de opciones, como ser texto, audio, fotografías, etc., o incluso una combinación de las mismas. Para nuestra aplicación, se propone la modificación del objeto patrón presente en la imagen mediante la reimpresión de otro en su lugar.

Para superponer el “objeto de realidad aumentada” en la posición y perspectiva correcta, se utiliza una Transformación Perspectiva.

3.10.1 Transformación Perspectiva

La transformación perspectiva es un tipo de transformación geométrica, que nos permite realizar un dimensionamiento no uniforme y rotación al objeto de RA que vamos a incluir en la escena.

La transformación perspectiva está estrechamente relacionada con la proyección perspectiva. Recordemos que esta última, mapea puntos del sistema físico de coordenadas tridimensionales globales en puntos del plano de la imagen en dos dimensiones, a través de un conjunto de líneas de proyección que se intersectan en un punto en común llamado centro de proyección. La transformación perspectiva es un tipo específico de homografía (homografía plana), que relaciona dos imágenes diferentes, las cuales son proyecciones alternativas del mismo objeto tridimensional sobre dos planos proyectivos diferentes.

El objetivo aquí es transformar la imagen “objeto de realidad aumentada”

para luego sobreimprimir en la escena y lograr enriquecer la realidad. La homografía H calculada anteriormente, mapea puntos de una primer imagen a puntos en una segunda imagen. Lo que se necesita para transferir los puntos de la primer imagen a la segunda es la homografía inversa. Sabemos que a partir de la detección del objeto en el flujo de video se obtiene una estimación de H , luego para superponer la imagen virtual en la perspectiva correcta en el flujo de video se utiliza la inversa (H^{-1}). Un esquema del proceso puede ser observado en la Fig. 3.9. Como es necesario transformar los píxeles de una imagen en una posición a otra, se debe tener en cuenta una interpolación para dibujar todos los píxeles en la nueva posición. Para realizar la trans-

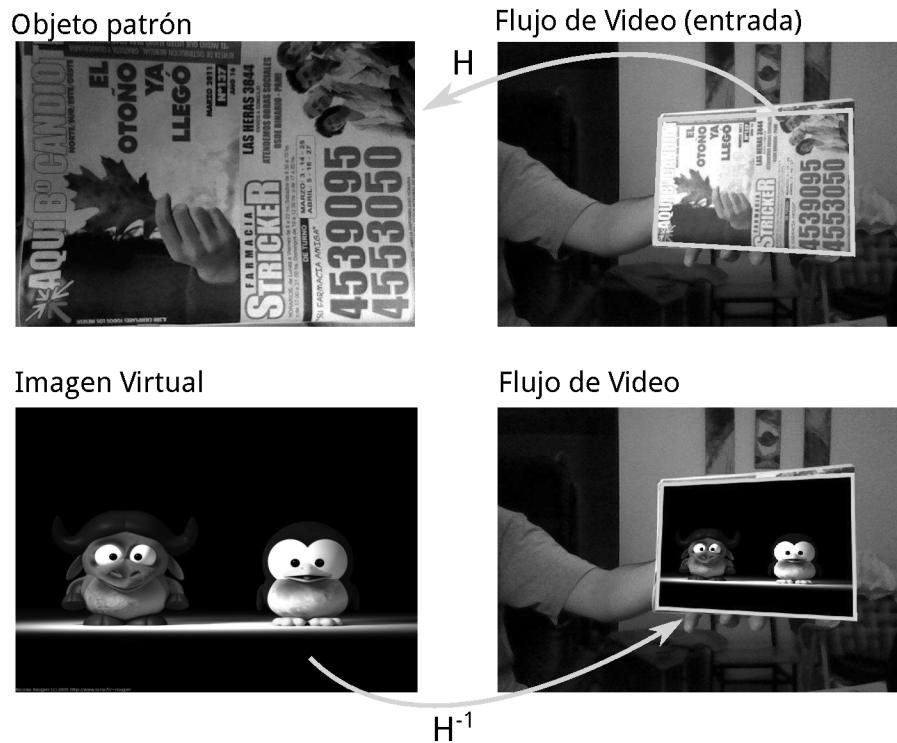


Figura 3.9: Esquema de transformación perspectiva utilizando la homografía H .

formación perspectiva se utilizó *cvWarpPerspective*² (librería OpenCV) que permite transformar los puntos dada la matriz de homografía, realizando la interpolación mencionada.

²http://opencv.willowgarage.com/documentation/cpp/imgproc_geometric_image_transformations.html#cv-warpperspective

CAPÍTULO 4

Experimentos y Resultados

En este capítulo, se presentan las pruebas y se discuten los resultados de cada etapa del método propuesto. Primeramente, se detallan y discuten los experimentos realizados con respecto a las mejoras de iluminación y realce de las imágenes utilizadas. Luego, se presentan distintas pruebas discutiendo los tiempos de procesamiento para las diferentes etapas del método propuesto. Finalmente, se expone un prototipo publicitario que incorpora el método desarrollado.

4.1 Definición de imágenes

Se pueden identificar tres diferentes tipos de imágenes utilizadas por el método y que denominamos: patrón, objetivo y objeto de realidad aumentada. Cabe aclarar que el objeto de RA puede ser obtenido de una imagen 2D, una vista de un modelo 3D, etc.

1. **Imagen patrón:** es la imagen usada como patrón que se pretende detectar y seguir en cada fotograma del flujo de video. Luego, en su lugar se superpone el *objeto de realidad aumentada*. La adquisición de la imagen patrón cumple con ciertas restricciones prácticas:

- el tamaño de la imagen debe ser de 640×480 píxeles,

- las condiciones de iluminación deben ser adecuadas para detectar características,
- la imagen debe ser rica en detalles, es decir que debe poseer bordes o características identificables y
- el plano de la imagen debe estar aproximadamente perpendicular al lente de la cámara al momento de la captura.

2. **Imagen objetivo:** es un fotograma del flujo de video, adquirido con la cámara web en tiempo real. Sobre este fotograma es donde se detecta la imagen patrón. Para esta imagen también se establecen algunas restricciones prácticas:

- el frame capturado con la cámara web posee un tamaño de 640×480 píxeles,
- las condiciones de iluminación deben ser adecuadas para detectar características y
- la imagen debe ser rica en detalles.

3. **Objeto de realidad aumentada:** es la definición de una imagen (ej: foto, tapa de libro, revista, texto etc.) que es superpuesta en el flujo de video.

La captura de la imagen patrón y la imagen objetivo, fue realizada con una cámara web con una resolución de 640×480 píxeles de una computadora portátil Toshiba Satellite A505-S69803.

Para las pruebas se ha utilizado como imagen patrón la tapa de una revista de $22\text{cm.} \times 17\text{cm}$. También, se han propuesto tres condiciones de iluminación diferentes para evaluar mejoras orientadas a la robustez del método y que denominamos:

- **Iluminación normal (B_N):** se simuló un ambiente con iluminación adecuado para la lectura. La habitación consta de: dos ventanas tras cortinas semitraslúcidas ubicadas a 5 metros (d_4) de la cámara web, iluminación mediante una lámpara de bajo consumo de 18 W. (equivalente a 90 W. de una lámpara incandescente) situada en el techo de la habitación a 1.7 metros (d_2) de la escena y a 0.4 metros (d_3) por detrás del objeto a detectar. Éste último, se situó a una distancia de 0.5 metros (d_1) de la cámara web (en la práctica esta distancia puede variarse entre 0.4 y 0.6 metros aproximadamente). Un esquema de las distancias descriptas puede observarse en la Fig. 4.1, donde las proporciones de

las líneas que identifican las cotas respecto a la medidas reales no han sido tenidas en cuenta, ya que sólo fue pensado para esquematizar las distancias en el entorno de pruebas. Para esta escena de iluminación, la lámpara direccional no está encendida.

- **Iluminación alta (B_H):** la imagen se capturó en la misma situación que la iluminación normal, con la diferencia que se estableció una iluminación direccional a través de una lámpara incandescente de 60 W., que apunte directamente a la escena situada frente a la cámara, como se observa en el esquema de la Fig. 4.1.
- **Iluminación baja (B_L):** la imagen se capturó en la misma habitación que las pruebas anteriores, pero tanto la lámpara de bajo consumo como la lámpara direccional están apagadas. Para dar una idea más acabada de la condición de iluminación, se puede pensar en una escena en la que se dificulta la lectura normal de un documento.

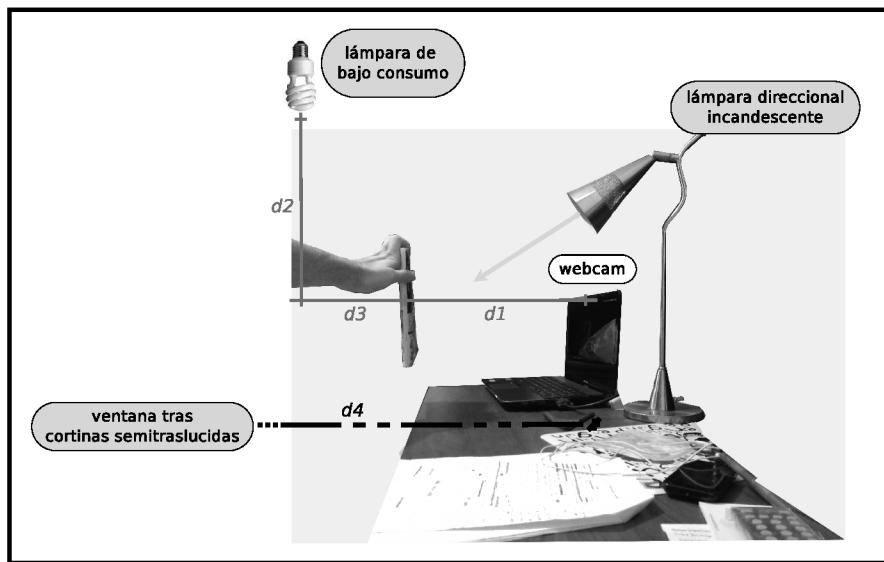


Figura 4.1: Esquema del ambiente en el que se realizaron las pruebas.

4.2 Experimentos

Para analizar el comportamiento del método se proponen pruebas diferentes:

- **Experimento 1:** evaluación del costo computacional y detección de puntos claves bajo las condiciones de iluminación B_N , B_H y B_L .
- **Experimento 2:** evaluación detallada del costo computacional en etapa de ejecución, considerando los pasos del algoritmo propuesto.

4.2.1 Experimento 1

En este experimento, se propone medir el comportamiento del método para diferentes condiciones de iluminación: B_N (Fig. 4.2a), B_H (Fig. 4.2b) y B_L (Fig. 4.2c). Para ello, se capturó la imagen patrón en las tres condiciones de iluminación y luego se aplicó a cada una de las imágenes las técnicas descriptas en la Sec.2.2, evaluándose el costo computacional y la cantidad de puntos característicos detectados.



(a) Imagen patrón con condición B_N .



(b) Imagen patrón con condición B_H .



(c) Imagen patrón con condición B_L .

Figura 4.2: Imágenes obtenidas para condiciones de iluminación diferentes.

Los resultados de las técnicas aplicadas con los parámetros que se mencionarán a continuación, pueden observarse en la Fig. 4.3, Fig. 4.4 y Fig. 4.5 para la condiciones de iluminación B_N , B_H y B_L , respectivamente. Por cuestiones de brevedad, haremos referencia a las imágenes de la Fig. 4.3 para la condición B_N , pero una interpretación similar se puede realizar para las demás condiciones.

La imagen patrón que se utilizó en la condición B_N se encuentra ilustrada en la Fig. 4.3a y es sobre ésta que se aplican las diferentes técnicas mencionadas. Para la *transformación logarítmica*, el valor c se estableció en 1 y el resultado de aplicar esta transformación, se puede observar en la Fig. 4.3b. El resultado la *ecualización* puede apreciarse en la Fig. 4.3c, mientras que en la Fig. 4.3d, se puede visualizar el resultado de un *filtrado pasa altos* con el kernel:

$$\begin{bmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{bmatrix} \quad (4.1)$$

En lo que respecta al *filtrado de alta potencia*, se estableció el valor $A = 2$ para la expresión (2.8) y se utilizó un kernel de un filtro pasa altos de la forma:

$$\begin{bmatrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{bmatrix} \quad (4.2)$$

El resultado de su aplicación se puede ver en la Fig. 4.3e.

Finalmente, se propuso una alternativa mixta para aumentar el contraste de la imagen realzando a su vez los detalles. Para ello, se aplicó la *ecualización del histograma* y posteriormente, sobre la imagen resultante, el *filtrado de alta potencia* con los parámetros mencionados anteriormente (Fig. 4.3f).

Como es de esperar, al realizar un análisis visual subjetivo de los resultados obtenidos al aplicar las diversas técnicas, se puede observar que para las tres condiciones de iluminación, el *filtrado pasa altos* y el *filtro de alta potencia* realzan los detalles, mientras que la *ecualización* mejora el contraste. En el caso particular de la condición B_L , se puede decir que la *ecualización* y la combinación de *ecualización* y *filtrado de alta potencia* son las que logran los mejores resultados respecto de las demás técnicas, esto como consecuencia de la maximización del contraste en la imagen resultante de la operación de ecualización.

Los resultados del tiempo de procesamiento (expresado en milisegundos) y la cantidad de puntos claves detectados para cada técnica en las tres condiciones de iluminación, se presentan resumidos en el Cuadro 4.1. Para la

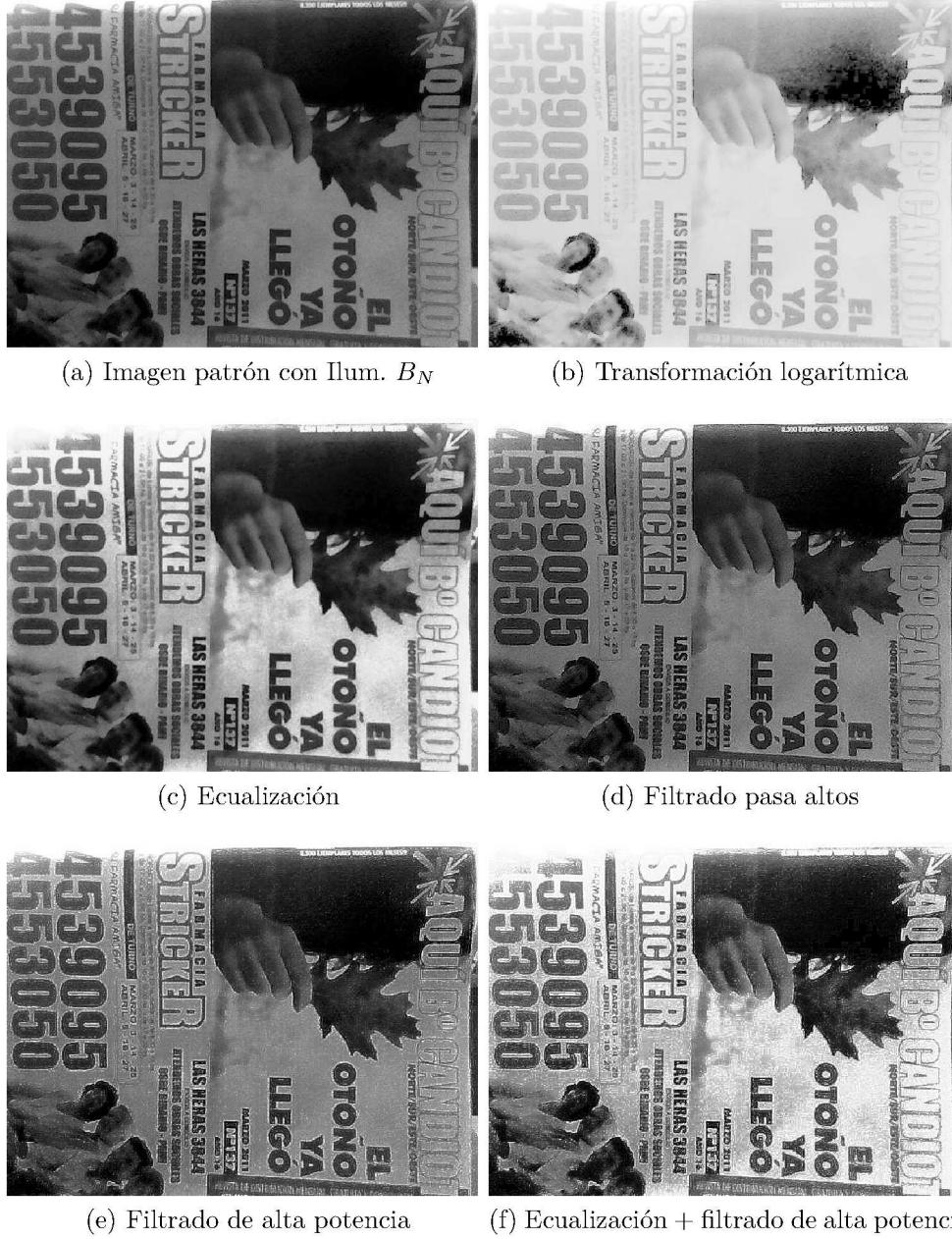


Figura 4.3: Resultados de aplicar las técnicas para la mejora en iluminación y detalles en condición B_N .

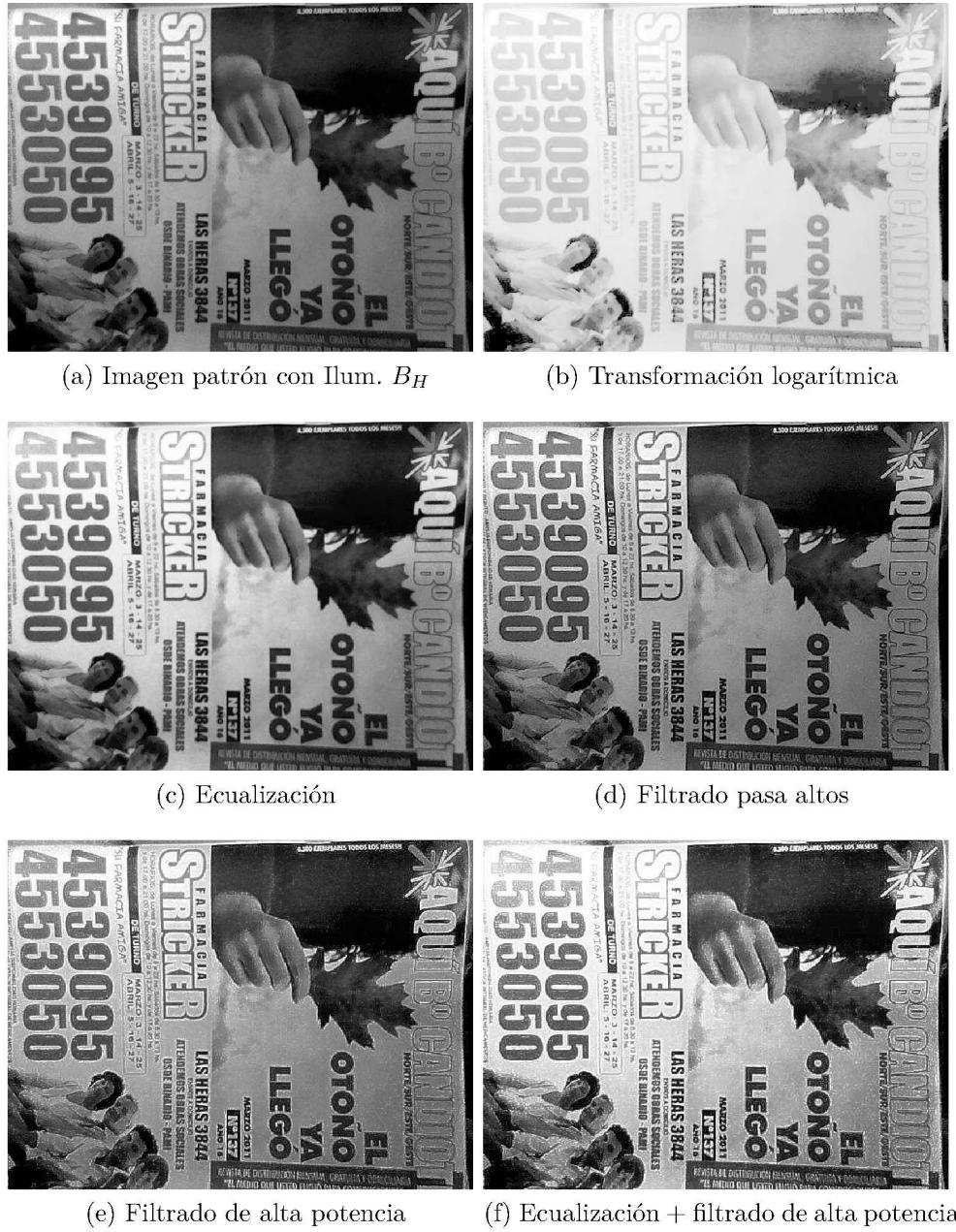


Figura 4.4: Resultados de aplicar las técnicas para la mejora en iluminación y detalles en condición B_H

(a) Imagen patrón con Ilum. B_L 

(b) Transformación logarítmica



(c) Ecualización



(d) Filtrado pasa altos



(e) Filtrado de alta potencia



(f) Ecualización + filtrado de alta potencia

Figura 4.5: Resultados de aplicar las técnicas para la mejora en iluminación y detalles en condición B_L

detección de puntos claves en la imagen, se utilizó un umbral hessiano que se estableció empíricamente en 700 y fue el mismo para las tres condiciones de iluminación.

	t [ms]	B_N	B_H	B_L
Sin Procesamiento	0.00	958	1297	154
Logaritmo	7.15	515	622	493
Ecualización	0.70	1546	1472	1233
F. Pasa Altos	1.26	1633	2006	269
F. Alta Potencia	3.10	1952	2216	353
Ec.+F. Alta Potencia	4.31	2704	2443	2002

Cuadro 4.1: Tiempo de procesamiento en milisegundos y cantidad de características sobre la imagen patrón en condiciones de iluminación B_N , B_H y B_L .

En el Cuadro 4.1, se puede observar que el tiempo de procesamiento consumido para la aplicación del *logaritmo* es superior a los demás valores, lo cual puede resultar incomprendible por tratarse de una operación sencilla. Se cree que el tiempo de cálculo se ve afectado por la forma en que fue implementado en el código de la aplicación: por un lado se debe tener en cuenta que la operación de logaritmo de OpenCV implementa el logaritmo neperiano y no el logaritmo en base 10, por lo cual se hizo necesario realizar varias operaciones para obtener el resultado correcto; por otro lado, existen casos en los que se describe que la instalación y compilación de OpenCV con soporte para Intel IPP¹ podría ser una solución al inconveniente, lográndose tiempos de cálculo más acotados. Sin embargo y como la cantidad de puntos característicos detectados es menor que cuando no se aplica procesamiento alguno (exceptuando la imagen en condiciones B_L), no se le dio mayor importancia.

Las técnicas que involucran una operación de convolución (*filtro pasa altos*, *filtro de alta potencia*) obtienen tiempos de procesamiento mayores que la *ecualización de la imagen*. Además y como es de esperar, la combinación de *ecualización y filtrado de alta potencia* resulta mayor que los anteriormente mencionados.

Si observamos en el Cuadro 4.1 la cantidad de características detectadas para cada técnica, se aprecia que la combinación de *ecualización + filtro de alta potencia* es la que presenta mayor cantidad de detecciones de puntos. Ésto es producto de una iluminación más uniforme lograda con la ecualización

¹<http://opencv.willowgarage.com/wiki/IPP>

y del realce de los detalles como resultado de la aplicación del filtro de alta potencia.

Para el *filtro pasa altos* y el de *alta potencia* el incremento en la cantidad de puntos detectados se debe a que dichos filtros realzan las altas frecuencias y discontinuidades. También se puede notar que en el caso del *filtrado de alta potencia*, se detectan más características que con el *filtro pasa altos* y ésto puede deberse a que el primero no afecta las bajas frecuencias. Mediante la *ecualización*, se obtienen más puntos detectados que cuando no se procesa la imagen y si bien no supera a los otros procesamientos, ésta puede considerarse la más útil si la prioridad es el tiempo.

Del análisis realizado, se observa que todos los procesamientos aplicados incrementan la cantidad de puntos detectados, exceptuando el logaritmo (sólo incrementa la cantidad de puntos en la imagen en condiciones B_L). Creemos que la obtención de menor cantidad de puntos detectados como resultado de aplicar la transformación logarítmica en condiciones B_N (Fig. 4.3b) y B_H (Fig. 4.4b), se ve justificada en que las imágenes resultantes poseen intensidades más homogéneas obteniéndose menor distinción en los detalles. Para la condición B_L , la cantidad de puntos detectados se incrementa respecto al de la imagen sin procesamiento, como resultado de un aumento de los detalles (obsérvese el borde superior de la mano de la imagen en la Fig 4.5b comparándose con la imagen de la Fig. 4.5a).

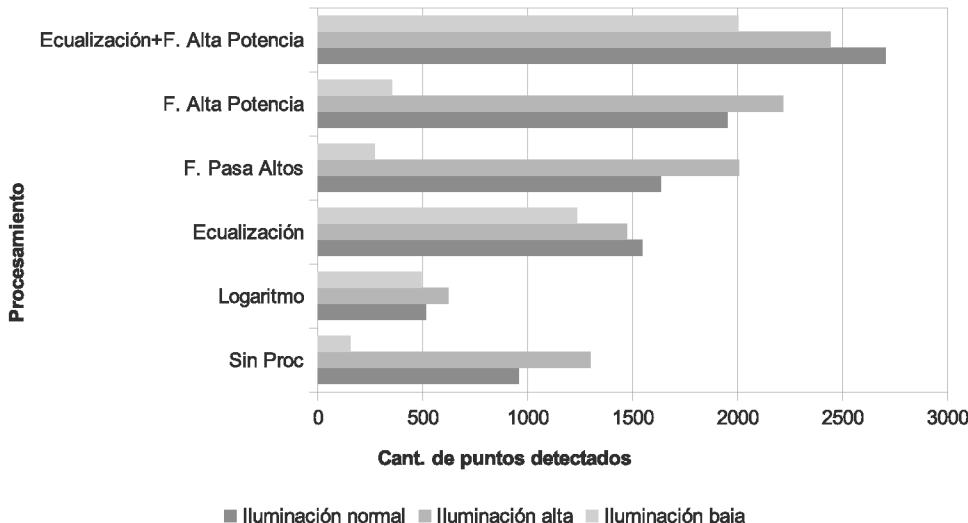


Figura 4.6: Cantidad de puntos detectados para la imagen con luminosidad normal, alta y baja.

En el gráfico de la Fig. 4.6 se presenta una comparación de la cantidad de puntos detectados para las condiciones B_N , B_H y B_L establecidas. Claramente se advierte que cuando se realiza la combinación de *ecualización y filtrado de alta potencia* en el caso de un ambiente en condición B_N , la cantidad de puntos detectados casi llega a triplicarse, respecto de cuando no se aplica técnica alguna. En tanto que en una condición B_H , la detección se duplica. Si analizamos lo que pasa en la condición de iluminación B_L , la cantidad de características posee valores bajos respecto de las otras condiciones de iluminación, ésta diferencia se evidencia aún más cuando no se aplica la ecualización, ya que los *filtros pasa altos y de alta potencia* actúan sobre una imagen oscura sobre la que realzan los detalles, pero la diferencia de grises entre píxeles vecinos resulta insuficiente para una correcta detección.

Hasta aquí se han planteado tres condiciones ambientales diferentes. Además de éstas, se han realizado algunas pruebas adicionales tratando de observar el comportamiento del método en condiciones de iluminación intermedias (entre B_N , B_H y B_L). Las mismas, no serán expuestas en este trabajo en detalle, sin embargo se realizará una reseña de las conclusiones obtenidas.

Si bien la obtención de una mayor cantidad de características no presenta una garantía de lograr una mejor detección del objeto buscado (pueden ser características espurias), en la práctica y sobre todo en situaciones de iluminación baja o en una condición intermedia entre B_L y B_N , donde la cantidad de características resulta insuficiente para lograr la detección del objeto, se notó que al aplicar *ecualización* o *ecualización + filtro de alta potencia*, se pudo detectar el objeto que previamente y sin procesamiento, no resultaba detectable. A pesar de obtener una correcta detección, se presentaron casos en que la detección fue “intermitente”. Se varió el umbral hessiano de forma de discriminar menos puntos característicos, pero no se obtuvieron resultados apreciablemente superiores.

En lo que respecta a las condiciones B_N y B_H , los resultados obtenidos sin la aplicación de un procesamiento de mejora del realce de iluminación y detalles, resultaron satisfactorios en las mayorías de las pruebas. Si bien la aplicación de las técnicas de *ecualización*, *filtrado pasa altos*, *filtrado de alta potencia* y la combinación de *ecualización y filtrado de alta potencia* mejoran la detección incrementando las características halladas y la robustez general del método, esto repercute negativamente en el tiempo de procesamiento, sobre todo en la aplicación de la *ecualización* y la combinación con el *filtro de alta potencia*. Sin embargo, la aplicación de estas técnicas cuando se está en presencia de un ambiente de iluminación intermedio entre B_L y B_N , contribuyen a incrementar la calidad en la detección. En la práctica las decisiones dependerán de lo que se quiera priorizar (calidad en la detección o

velocidad de ejecución) y del ambiente en que se use el método. En base a ello, se puede tomar la decisión de la técnica a utilizar, como así también del umbral hessiano que permita la mejor relación entre velocidad y calidad de detección.

4.2.2 Experimento 2

En el capítulo 3, se ha descripto el diagrama de flujo del método desarrollado, el cual presenta condicionales que permiten mejorar la detección y minimizar los tiempos de procesamiento. En esta sección se analizan y discuten los tiempos que insumen los procesos observables en el diagrama de la Fig. 3.2 para obtener una conclusión más detallada sobre la efectividad de las alternativas planteadas.

Cuando se realiza el procesamiento en condiciones de baja iluminación (como se ha descripto en la Sec. 4.2.1), la detección no resulta del todo estable (se produce efecto de “parapadeo” seguido y repetitivo del objeto de RA). Dado que aquí se desea obtener conclusiones cuando el método es “estable” en la detección, se realizaron dos pruebas en condiciones de iluminación B_N y B_H , estableciéndose empíricamente el umbral hessiano para lograr un equilibrio entre el tiempo de procesamiento y la correcta detección del objeto. La aplicación de alguna de las técnicas mencionadas en el “Pre-Procesamiento de Iluminación y Realce de detalles” no fue necesaria, ya que con las condiciones ambientales se obtuvieron resultados aceptables. Si bien aplicar alguna de estas técnicas mejora la calidad de detección, también incrementa el tiempo de procesamiento y es por ello que se deja a criterio del usuario la decisión de aplicarlas, según lo considere necesario. Las pruebas se definieron como:

- **Prueba 1 (P1):** duración: 1:50 minutos, umbral hessiano: 3500. Imagen con condición B_N Fig. 4.2a²
- **Prueba 2 (P2):** duración: 1:35 minutos, umbral hessiano: 5000. Imagen con condición B_H Fig. 4.2b³

Cabe aclarar que **P1** y **P2** son dos pruebas independientes, en las que el objeto presenta movimientos parecidos aunque diferentes en las escenas.

Para cada ciclo de la fase de ejecución de cada prueba se computaron los siguientes datos:

²Video de la prueba: <http://youtu.be/E8uGptzdEiw>

³Video de la prueba: <http://youtu.be/Pkiub9ZXP4k>

- Tiempo de *detección de movimiento*,
- Tiempo de *extracción y descripción de las características en BR* *,
- Tiempo de *búsqueda de correspondencias* *,
- Tiempo de *estimación de la homografía* *,
- Tiempo de comprobación *contorno convexo y vértices válidos* *,
- Tiempo de *transformación perspectiva de la imagen de RA utilizando la homografía* *,
- Cuadros por segundo (FPS),
- Total de puntos detectados en la imagen patrón,
- Total de puntos detectados en la imagen objeto,
- Total de coincidencias potencialmente válidas.

Si se observa el diagrama de la Fig. 3.2, se notará que las operaciones marcadas con (*) pueden ignorarse si se cumple una determinada condición en el ciclo de ejecución, sin impedir que se realice el enriquecimiento de la realidad. Esto como consecuencia del buffer de transformaciones descripto en la sección 3.9. Así, por ejemplo si la comprobación de contornos convexo y vértices válidos (CCVV) no se cumple, entonces no se calcula la transformación perspectiva (TP); o si la homografía no es detectada, luego, no se realiza la CCVV y por ende tampoco la TP; aún más, si no se detecta movimiento en al escena, entonces ninguno de los procesos anteriormente mencionados es ejecutado. Así, todos estos condicionales producen que el tiempo promedio de procesamiento disminuya cuando alguna (o todas) las operaciones son ignoradas. Para identificar esta situación la denominaremos con el término: “Proceso con bifurcación en condicionales (PCBC)”; en caso contrario se denominará “Proceso sin bifurcación en condicionales (PSBC)” y sería equivalente al diagrama del método propuesto en la Fig. 3.2, realizando las operaciones independientemente de lo que suceda en la escena.

En el gráfico de la Fig. 4.7, se pueden observar los tiempos promedios en la fase de ejecución para el PCBC y PSBC en P1. Para analizar los datos, se calcularon los FPS promedio (FPS_{prom}) para el proceso de ejecución. De ello, se pudo advertir que el valor de FPS_{prom} cuando se da el PSBC fue de 6.93 FPS, mientras que para el PCBC fue de 28.35 FPS, lo cual constituye una mejora notable en el tiempo de procesamiento. Dicho de otro modo, la lógica de los condicionales planteados para mejorar el tiempo de procesamiento y

la detección del objeto evidencia un gran incremento en los FPS_{prom} y esto se debe en gran parte a que evita el proceso de extracción y descripción de características (observar que la barra verde en el gráfico representa casi la mitad de tiempo sobre la barra violeta).

En lo que respecta a los tiempos en los diferentes procesos, se puede visualizar en la Fig. 4.7 que la extracción y descripción de características es la que más tiempo requiere, a pesar de que se minimizó el procesamiento en un área de la imagen (BR) como se explicó en la Sec. 3.4. Experimentalmente se comprobó que si el área BR en la que se extraen características es del 50 % de la imagen patrón de 640×480 píxeles, el tiempo de procesamiento representa aproximadamente un 42 % del tiempo total que se requiere para extraer las características en la totalidad de la imagen.

El segundo proceso que más tiempo consume es la estimación de la homografía por tratarse de un método iterativo que estima la matriz H y refina las coincidencias. La transformación perspectiva que involucra una interpolación, ocupa el tercer lugar, seguida de la búsqueda de correspondencias, la detección de movimiento y finalmente, la CCVV que está compuesta de operaciones triviales.

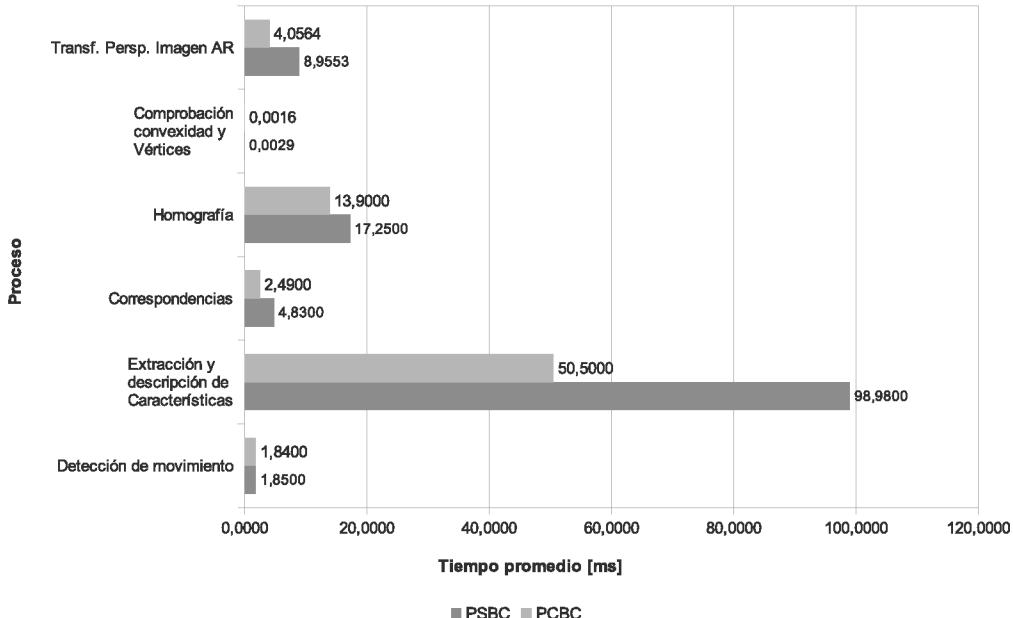


Figura 4.7: Tiempos promedios de operación para Prueba 1

En lo que respecta a total de puntos detectados y las potenciales corres-

pondencias válidas, para P1 se detectaron 192 puntos claves en la imagen patrón y un promedio de 130 puntos en la imagen del flujo de video. De los mismos se obtuvo (en promedio) 38 pares de coincidencias potencialmente válidas, observándose un máximo de 81 pares.

En la Fig. 4.8c a 4.8f se presentan diferentes capturas del resultado final del método, utilizando como imagen patrón la Fig. 4.8a y como objeto de realidad aumentada la imagen de la Fig. 4.8b. El umbral hessiano se estableció a 3500 y la escena se encuentra en condición B_N de iluminación.

En lo que respecta a la prueba P2, los detalles de los resultados no serán expuestos aquí para no extender este informe. Sin embargo, se debe mencionar que de su análisis se obtuvieron resultados similares en los órdenes de magnitud de tiempos por procesos. La cantidad de puntos claves detectados, no puede ser comparada con P1, ya que se trata de un umbral e imagen diferente, en condiciones diferentes de iluminación. De todas formas, podemos decir que la cantidad de puntos detectados en la imagen patrón, se vio incrementada a pesar de aumentar el umbral hessiano, debido a que la imagen sobre la cual se aplicó poseía mayor luminosidad y, por lo tanto mayores detalles apreciables.

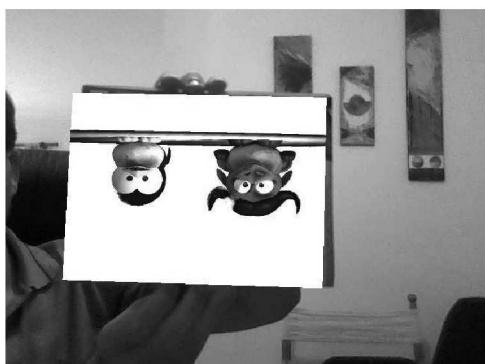
4.3 Implementación de prototipos

Como se ha mencionado en los objetivos específicos del presente trabajo, se ha propuesto implementar un prototipo que otorgue una visión del resultado final del método desarrollado. Para ello, se plantearon dos alternativas: en la primera de ellas, se enriquece la realidad mediante una imagen sobre la tapa de una revista y en la Fig. 4.8 se pueden ver algunas imágenes de esta implementación.

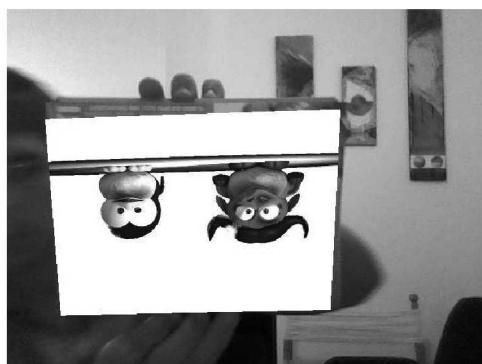
Como segunda alternativa, se planteó proveer información inherente a un producto comestible de forma publicitaria, en el que se brinda su precio y una imagen de un plato preparado con el mismo. El resultado del prototipo se puede visualizar en la secuencia de imágenes de la Fig. 4.9, donde: la Fig. 4.9a es la imagen patrón, la Fig. 4.9b es el objeto de realidad aumentada a sobreimprimir (el fondo blanco en la imagen es transparente) y las imágenes de la Fig. 4.9c a la Fig. 4.9f, representan diferentes capturas de la secuencia de video en la etapa de ejecución. Cabe aclarar que se presentan sólo algunas de las detecciones exitosas. Un video de este prototipo en condición B_N de iluminación puede ser visto en la url: <http://youtu.be/j1xPZkg1JHs>.

(a) Imagen patrón con Ilum. B_N 

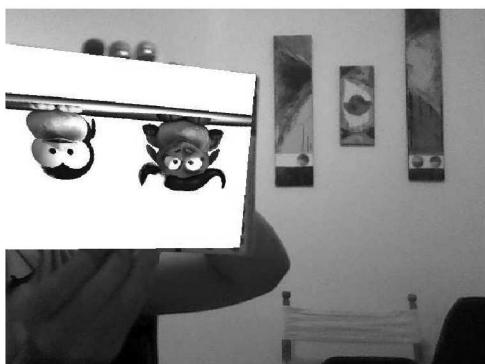
(b) Objeto de RA



(c)



(d)



(e)



(f)

Figura 4.8: Imagen patrón, objeto de RA y capturas de imágenes con enriquecimiento de la realidad.

Los prototipos de realidad aumentada desarrollados aquí son sólo algunas de las aplicaciones posibles, y se pueden pensar en aplicaciones más avanzadas, como por ejemplo que provean una interactividad mayor con el usuario, a partir de la introducción de audio, videos, animaciones, etc.



Figura 4.9: Imagen patrón, objeto de realidad aumentada a sobreimprimir en el flujo de video y diferentes capturas en la etapa de ejecución para el prototipo publicitario.

CAPÍTULO 5

Conclusiones y trabajos futuros

En este último capítulo, se describen las conclusiones obtenidas tras realizar el Proyecto Final, junto a un conjunto de trabajos futuros que permitan seguir líneas de investigación sobre el mismo, para mejorarlo o adaptarlo a diferentes necesidades.

5.1 Conclusiones

En el presente proyecto, se ha propuesto un método para detectar objetos planos en un ambiente controlado, mediante la detección de características sobre imágenes obtenidas con una cámara web de computadora. Se introdujo el concepto de realidad aumentada, como así también los tipos de sistemas más conocidos (con y sin marcadores) y los métodos para la detección y seguimiento de objetos basados en localización y/o visualización. Se han mencionado detectores de características comunes reportados en la bibliografía y diferentes herramientas que son válidas para la detección y seguimiento de objetos. Particularmente, se ha descripto en detalle el método SURF (sobre el cual se sustenta el método propuesto) contrastándolo con SIFT, como así también, los pasos para llegar a obtener la homografía para la detección y ubicación de la posición de la imagen patrón.

El proceso general anteriormente mencionado se ha modificado median-

te diferentes procesamientos y validaciones para detectar al objeto correc-tamente y producir transformaciones coherentes, sin afectar al tiempo de procesamiento de forma considerable. Adicionalmente, se han planteado téc-nicas simples para realce de detalles e iluminación sobre las cuales se han realizado algunos experimentos para evaluar su comportamiento, de forma que el usuario pueda considerar la aplicación de alguna de ellas, si la si-tuación lo requiriese. También, se han propuesto estrategias heurísticas para evitar cálculos innecesarios o efectos de “parpadeos” del objeto de realidad aumentada, logrando mayor naturalidad en el enriquecimiento de la realidad y considerando operaciones de tiempos acotados de procesamiento. Tras ello, se ha concluido con dos prototipos, uno de los cuales es un prototipo publici-tario sobre el cual se brinda información adicional inherente a un producto.

La codificación del método se realizó mediante el uso de lenguaje C++ y la librería OpenCV que permiten la portabilidad a diferentes sistemas ope-rativos.

Finalmente, el desarrollo de este proyecto final permitió afianzar los cono-cimientos en el área de visión computacional. El reconocimiento de objetos y la realidad aumentada son temáticas que brindan la posibilidad de ser utili-zadas en diversidad de aplicaciones, siendo interesante encarar desarrollos e investigaciones futuras, que esperemos puedan surgir a partir de este trabajo.

5.2 Trabajos futuros

A fin de mejorar el desempeño general, en esta sección se proponen algunas modificaciones que podrían ser aplicadas al método desarrollado o nuevas investigaciones a realizar. Se han identificado algunas de ellas, clasificándolas en trabajos futuros a corto, mediano y largo plazo, los cuales son mencionados a continuación.

Corto Plazo:

- Implementar el método sobre GPU [12] y con múltiples hilos de pro-cesamiento [67] aplicado al proceso de extracción y descripción de ca-racterísticas. Existen algunas implementaciones en el tema: <http://asrl.utias.utoronto.ca/code/gpusurf/index.html>,
<http://www.d2.mpi-inf.mpg.de/surf>.
- Utilizar para la búsqueda de correspondencias una variante de la in-terfase FLANN implementada por OpenCV que se denomina “nano

flann". En la página oficial del proyecto de la interfase, se hacen notar bondades de eficiencia en tiempo y consumo de memoria <http://code.google.com/p/nanoflann/>.

- Implementar una selección automática del umbral hessiano para la discriminación de puntos.

Mediano Plazo:

- Usar otros detectores en el proceso de extracción y descripción de características. Una comparación y pruebas sobre algunas imágenes con diversos detectores pueden observarse en <http://bit.ly/hox3gW>.
- Investigar e implementar soluciones para el manejo de la oclusión entre objetos virtuales y el mundo real.

Largo Plazo:

- Implementar una aplicación con interfaz gráfica orientada a un usuario final almacenando en una base de datos diferentes imágenes patrón, luego, identificar en el flujo de video cual de las imágenes almacenadas se encuentra presente y llevar a cabo una acción particular para el enriquecimiento de la realidad.
- Implementar el sistema en dispositivos móviles (tablets, celulares, etc.).

Bibliografía

- [1] C. Silpa Anan and R. I. Hartley. Optimised KD-trees for fast image descriptor matching. In *CVPR*, pages 1–8, 2008.
- [2] Sunil Arya, David M. Mount, Nathan S. Netanyahu, Ruth Silverman, and Angela Y. Wu. An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *J. ACM*, 45(6):891–923, November 1998.
- [3] Ronald Azuma, Yohan Baillot, Reinhold Behringer, Steven Feiner, Simon Julier, and Blair MacIntyre. Recent advances in augmented reality. *IEEE Comput. Graph. Appl.*, 21(6):34–47, November 2001.
- [4] Herbert Bay. *From wide-baseline point and line correspondences to 3D*. PhD thesis, 2009.
- [5] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up Robust Features (SURF). *Computer Vision and Image Understanding: CVIU*, 110(3):346–359, June 2008.
- [6] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *In ECCV*, pages 404–417, 2006.
- [7] Jeffrey S. Beis and David G. Lowe. Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *Proceedings of the 1997 Conference on Computer Vision and Pattern Recognition (CVPR '97)*, CVPR '97, pages 1000–1006, Washington, DC, USA, 1997. IEEE Computer Society.

- [8] Selim Benhimane, Hesam Najafi, Matthias Grundmann, Yakup Genc, Nassir Navab, and Ezio Malis. Real-time object detection and tracking for industrial applications. In Alpesh Ranchordas and Helder Araújo, editors, *VISAPP (2)*, pages 337–345. INSTICC - Institute for Systems and Technologies of Information, Control and Communication, 2008.
- [9] Gary Bradski and Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. O'Reilly Media, 1st edition, October 2008.
- [10] S. Brin. Near neighbor search in large metric spaces. In *Proceedings of the 21st International Conference on Very Large Data Bases (VLDB '95)*, pages 574–584, San Francisco, Ca., USA, sep 1995. Morgan Kaufmann Publishers, Inc.
- [11] Ben Butchart. Augmented reality for smartphones - a guide for developers and content publishers. Technical report, 2011.
- [12] Jian Cao, Xiao-fang Xie, Jie Liang, and De-dong Li. GPU Accelerated Target Tracking Method. In David Jin and Sally Lin, editors, *Advances in Multimedia, Software Engineering and Computing Vol.1*, volume 128 of *Advances in Intelligent and Soft Computing*, pages 251–257. Springer Berlin Heidelberg, 2012.
- [13] Zhuo Chen and Xinyu Li. Markless tracking based on natural feature for augmented reality. In *Educational and Information Technology (ICEIT), 2010 International Conference on*, volume 2, pages 126–129, sept. 2010.
- [14] Daniel Conrad and Guilherme N. DeSouza. Homography-based ground plane detection for mobile robot navigation using a modified EM algorithm. In *ICRA*, pages 910–915. IEEE, 2010.
- [15] Richard O. Duda and Peter E. Hart. Use of the hough transformation to detect lines and curves in pictures. *Commun. ACM*, 15(1):11–15, January 1972.
- [16] Christopher Evans. Notes on the opensurf library. Technical Report CSTR-09-001, University of Bristol, January 2009.
- [17] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.
- [18] Jerome H. Friedman, Jon Louis Bentley, and Raphael Ari Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3(3):209–226, September 1977.

- [19] Keinosuke Fukunaga and Patrenahalli M. Narendra. A branch and bound algorithm for computing k-nearest neighbors. *IEEE Transactions on Computers (TOC)*, C-24(7):750–753, jul 1975.
- [20] Óscar Boullosa García. *Estudio comparativo de descriptores visuales para la detección de escenas cuasi-duplicadas*. PhD thesis, 2011.
- [21] R.C. González and R.E. Woods. *Digital image processing*. Prentice Hall, 2002.
- [22] Chris Harris and Mike Stephens. A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151, 1988.
- [23] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge books online. Cambridge University Press, 2003.
- [24] Richard Hartley and Chanop Slipa Anan. Localisation using an image-map. 2004.
- [25] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Pascal Fua, and Nassir Navab. Dominant orientation templates for real-time detection of texture-less objects. In *CVPR*, pages 2257–2264. IEEE, 2010.
- [26] B. Jähne and H. Haussecker. *Computer Vision and Applications: A Guide for Students and Practitioners*. Electronics & Electrical. Academic Press [Imprint], 2000.
- [27] Frédéric Jurie and Cordelia Schmid. Scale-invariant shape features for recognition of object categories. In *CVPR (2)*, pages 90–96, 2004.
- [28] Timor Kadir and Michael Brady. Saliency, scale and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.
- [29] Jong-Hyoun Kim and T. Cho. The initiative experiments for utilizing real cards in online trading card game. In *Computer Sciences and Convergence Information Technology (ICCIT), 2010 5th International Conference on*, pages 182–185, 2010.
- [30] Kiyoung Kim, Jonghee Park, and Woontack Woo. Marker-less tracking for multi-layer authoring in AR books. In Stéphane Natkin and Jérôme Dupire, editors, *ICEC*, volume 5709 of *Lecture Notes in Computer Science*, pages 48–59. Springer, 2009.
- [31] Georg Klein and David W. Murray. Parallel tracking and mapping for small AR workspaces. In *ISMAR* [36], pages 225–234.

- [32] Michal Kottman. Planar object detection using local feature descriptors. In *Information Sciences and Technologies Bulletin of the ACM Slovakia, Special Section on Student Research in Informatics and Information Technologies*, pages 59–63, 2011.
- [33] Robert Laganière. *OpenCV 2 Computer Vision Application Programming Cookbook*. Packt Publishing, June 2011.
- [34] Ahyun Lee, Jae-Young Lee, Seok-Han Lee, and Jong-Soo Choi. Markerless augmented reality system based on planar object tracking. In *Frontiers of Computer Vision (FCV), 2011 17th Korea-Japan Joint Workshop on*, pages 1–4, feb. 2011.
- [35] Taehee Lee and Tobias Höllerer. Handy AR: Markerless inspection of augmented reality objects using fingertip tracking. In *ISWC*, pages 83–90. IEEE, 2007.
- [36] Taehee Lee and Tobias Höllerer. Initializing markerless tracking using a simple hand gesture. In *ISMAR*, pages 259–260. IEEE, 2007.
- [37] Bastian Leibe, Krystian Mikolajczyk, and Bernt Schiele. Efficient Clustering and Matching for Object Class Recognition. In *Proceedings of the British Machine Vision Conference*, pages 81.1–81.10. BMVA Press, BMVA Press, 2006. doi:10.5244/C.20.81.
- [38] Vincent Lepetit and Pascal Fua. Towards Recognizing Feature Points using Classification Trees. EPFL Technical Report IC/2004/74, Ecole Polytechnique Fédérale de Lausanne, September 2004.
- [39] Vincent Lepetit and Pascal Fua. Monocular model-based 3d tracking of rigid objects. *Found. Trends. Comput. Graph. Vis.*, 1(1):1–89, January 2005.
- [40] Xinyu Li and Dongyi Chen. Augmented reality in e-commerce with markerless tracking. In *Information Management and Engineering (ICIME), 2010 The 2nd IEEE International Conference on*, pages 609–613, april 2010.
- [41] Tony Lindeberg. Feature detection with automatic scale selection. *International Journal of Computer Vision*, 30:79–116, 1998.
- [42] Ting Liu, Andrew W. Moore, Alexander Gray, and Ke Yang. An investigation of practical approximate nearest neighbor algorithms. pages 825–832. MIT Press, 2004.

- [43] T. W. R. Lo and J. P. Siebert. Local feature extraction and matching on range images: 2.5D SIFT. *Computer Vision and Image Understanding*, 113(12):1235–1250, December 2009.
- [44] Dawid G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, nov 2004.
- [45] Dawid G. Lowe. Object recognition from local scale-invariant features. In *International Conference on Computer Vision, 20–25 September, 1999, Kerkyra, Corfu, Greece, Proceedings*, volume 2, pages 1150–1157, 1999.
- [46] Jiri Matas, Ondrej Chum, Martin Urban, and Tomás Pajdla. Robust wide-baseline stereo from maximally stable extremal regions. *Image Vision Comput.*, 22(10):761–767, 2004.
- [47] K. Mikolajczyk and J. G. Matas. Improving descriptors for fast tree matching by optimal linear projection. In *ICCV*, pages 1–8, 2007.
- [48] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, October 2005.
- [49] Krystian Mikolajczyk and Cordelia Schmid. Indexing based on scale invariant interest points. In *ICCV*, pages 525–531, 2001.
- [50] Krystian Mikolajczyk and Cordelia Schmid. Scale & affine invariant interest point detectors. *International Journal of Computer Vision*, 60(1):63–86, 2004.
- [51] Krystian Mikolajczyk and Cordelia Schmid. A performance evaluation of local descriptors. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(10):1615–1630, October 2005.
- [52] Paul Milgram, Haruo Takemura, Akira Utsumi, and Fumio Kishino. Augmented reality: A class of displays on the reality-virtuality continuum. pages 282–292, 1994.
- [53] Tsutomu Miyashita, Peter Georg Meier, Tomoya Tachikawa, Stephanie Orlic, Tobias Eble, Volker Scholz, Andreas Gapel, Oliver Gerl, Stanimir Arnaudov, and Sebastian Lieberknecht. An augmented reality museum guide. In *ISMAR*, pages 103–106. IEEE, 2008.

- [54] Marius Muja and David G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *International Conference on Computer Vision Theory and Application VISSAPP'09*, pages 331–340. INSTICC Press, 2009.
- [55] David Nister and Henrik Stewenius. Scalable recognition with a vocabulary tree. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 2161–2168, Washington, DC, USA, 2006. IEEE Computer Society.
- [56] Mark S. Nixon and Alberto S. Aguado. *Feature extraction and image processing*. Newnes, Oxford, 2002.
- [57] M. Ozuysal, P. Fua, and V. Lepetit. Fast keypoint recognition in ten lines of code. In *CVPR*, pages 1–8, 2007.
- [58] Rousseeuw P.J. Least median of squares regression. *Journal of the American Statistical Association*, 79(388):871–880, 1984.
- [59] Edward Rosten and Tom Drummond. Fusing points and lines for high performance tracking. In *IN INTERNATIONAL CONFERENCE ON COMPUTER VISION*, pages 1508–1515. Springer, 2005.
- [60] Patrice Simard, Léon Bottou, Patrick Haffner, and Yann LeCun. Boxlets: A fast convolution algorithm for signal processing and neural networks. In Michael J. Kearns, Sara A. Solla, and David A. Cohn, editors, *NIPS*, pages 571–577. The MIT Press, 1998.
- [61] Patrice Y. Simard, Léon Bottou, Patrick Haffner, and Yann LeCun. Boxlets: a fast convolution algorithm for signal processing and neural networks. In *Proceedings of the 1998 conference on Advances in neural information processing systems II*, pages 571–577, Cambridge, MA, USA, 1999. MIT Press.
- [62] Iryna Skrypnyk and David G. Lowe. Scene modelling, recognition and tracking with invariant image features. In *ISMAR*, pages 110–119. IEEE Computer Society, 2004.
- [63] Gabriel Takacs, Vijay Chandrasekhar, Natasha Gelfand, Ying Xiong, Wei-Chao Chen, Thanos Bismpligianis, Radek Grzeszczuk, Kari Pulli, and Bernd Girod. Outdoors augmented reality on mobile phone using loxel-based visual feature organization. In *Proceedings of the 1st ACM international conference on Multimedia information retrieval*, MIR '08, pages 427–434, New York, NY, USA, 2008. ACM.

- [64] Tinne Tuytelaars and Krystian Mikolajczyk. Local invariant feature detectors: A survey. *Foundations and Trends in Computer Graphics and Vision*, 3(3):177–280, 2007.
- [65] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511–I–518 vol.1, 2001.
- [66] Daniel Wagner, Gerhard Reitmayr, Alessandro Mulloni, Tom Drummond, and Dieter Schmalstieg. Real-time detection and tracking for augmented reality on mobile phones. *IEEE Trans. Vis. Comput. Graph*, 16(3):355–368, 2010.
- [67] Nan Zhang. Computing optimised parallel speeded-up robust features (p-surf) on multi-core processors. *International Journal of Parallel Programming*, 38(2):138–158, 2010.