UNIVERSIDAD MICHOACANA DE
SAN NICOLÁS DE HIDALGO

FACULTAD DE INGENIERÍA ELECTRICA
DIVISIÓN DE ESTUDIOS DE POSGRADO

IMPLEMENTACIÓN EN TIEMPO REAL PARA SEGUIMIENTO DE OBJETOS EN SECUENCIAS DE VÍDEO

TESIS

QUE PARA OBTENER EL GRADO DE:
MAESTRÍA EN CIENCIAS EN INGENIERÍA ELÉCTRICA

PRESENTA:
JESÚS EDUARDO ALCARAZ CHÁVEZ

DIRECTOR DE TESIS
DR. FÉLIX CALDERÓN SOLORIO

AGOSTO 2013

IMPLEMENTACIÓN EN TIEMPO REAL PARA SEGUIMIENTO DE OBJETOS EN SECUENCIAS DE VÍDEO

TESIS

Que para obtener el grado de

MAESTRO EN CIENCIAS EN INGENIERÍA ELÉCTRICA

presenta

Jesús Eduardo Alcaraz Chávez

Dr. Félix Calderón Solorio

Director de Tesis

Universidad Michoacana de San Nicolás de Hidalgo

Agosto 2013

Resumen

En el presente trabajo de tesis se implementa un seguidor, detector de objetos, en una secuencia de vídeo. Esta implementación trabaja en tiempo real, y requiere sólo una muestra del objeto de interés, proporcionada por un click y arrastre de ratón.
El seguidor detector, está basado en el algoritmo de SURF (Speeded Up Robust Features), debido a la robustez que presenta en cuanto a cambios de luminosidad y a las posibles transformaciones que a una imagen se refiere, como lo son: el escalamiento, cizallamiento, rotación traslación.
La finalidad es poder concretar un seguidor, detector en tiempo real, dado la velocidad con que funciona el proceso para generar los vectores de características. Que posteriormente son necesarios para aplicar la correspondencia entre el objeto de interés y una secuencia de vídeo cualquiera que aloje el objeto de interés.
Una vez que se generaron los puntos de interés, tanto de la imagen de muestra del objeto, como los puntos en la secuencia de vídeo, se implementa la búsqueda de la correspondencia mediante vecinos cercanos. Se puede acelerar el proceso mediante k-d tree, sin embargo, existe una diversidad de métodos que puede afectar el rendimiento final de la implementación. Finalmente se utiliza el algoritmo de RANSAC para buscar la transformación afín, que demarque el objeto de interés, para adecuarse a la deformación del objeto.

Al concluir todo este proceso, se pudo constatar que es posible seguir objetos a partir de una sola muestra, y aunque el objeto tenga deformaciones rígidas en su seguimiento. De igual forma se constato que es posible retomar el objeto si sale de escena.

Abstract

In this thesis a follower detector objects in a video sequence is implemented. This implementation works in real time, and requires only a sample of the object of interest, provided by a mouse click and drag. The detector tracker is based on the algorithm of SURF (Speeded Up Robust Features), due to its robustness against changes in brightness as possible transformations of an image, such as scaling, shearing, rotation, translation.

The purpose is to realize a follower detector in real time, given the speed with which the process works to generate the feature vectors. They are then required to apply the correspondence between the object of interest and any video stream that hosts the object of interest.

Once generated the points of interest, both of the sample image of the object, as the points in the video stream, is implemented by the correspondence search nearby neighbors. It can speed up the process by kd tree, however, a variety of methods that can affect the final performance of the implementation. Finally we use the RANSAC algorithm to find the affine transformation, to demarcate the object of interest, to adapt to the deformation of the object.

On completion of the process, it was found that it is possible to track objects from a single sample, and if the object is rigid deformation monitoring. Likewise, it was verified that it is possible to return the object if it leaves the stage.

Contenido

Resumen III
Abstract V
Contenido VII
Lista de Figuras IX
Lista de Tablas XII
Lista de Símbolos XIII

1. Introducción 1
   1.1. Motivación 2
   1.2. Planteamiento del Problema 3
   1.3. Objetivos y Alcances de la Tesis 4
      1.3.1. Objetivo general 4
      1.3.2. Alcances 5
   1.4. Descripción del Sistema 6
   1.5. Metas 7
   1.6. Descripción y Organización del Documento 7

2. Estado del Arte del Seguimiento y Detección de Objetos 9
   2.1. Seguimiento de Objetos 9
   2.2. Modelado para la representación del objeto 10
   2.3. Selección de características para el seguimiento 14
   2.4. Detección de objetos 16
   2.5. Detección de Puntos de Interés 16
   2.6. Sustracción del Fondo 19
   2.7. Segmentación 24
   2.8. Aprendizaje supervisado 24
   2.9. Modelos de Seguimiento (Tracking) 27
   2.10. Conclusión 29

3. Detección de objetos 31
   3.1. SIFT 31
      3.1.1. Detección de Extremos en el Espacio Escala 32

3.1.2. Refinamiento de Puntos Clave 36
3.1.3. Supresión de puntos situados a lo largo de bordes 38
3.1.4. Asignación de la Orientación de los puntos clave 40
3.1.5. Descriptor de Puntos de interés 44
3.2. SURF 46
3.2.1. Detección de puntos de interés 47
3.2.2. Aproximación al Espacio-Escala vía Filtros de Caja 48
3.2.3. Muestro del Espacio-Escala 53
3.2.4. Aproximación de los operadores diferenciales 58
3.2.5. Selección de Características 63
3.2.6. Descripción Local 63
3.2.7. Orientación del Punto de interés 64
3.2.8. Creación del Descriptor 66

4. Seguimiento y Correspondencia de Objetos 69
4.1. Correspondencia de Características 69
4.1.1. Indexación Eficiente de Vecinos Próximos 72
4.1.2. Indexación rápida para la correspondencia 72
4.1.3. Evaluación del tiempo entre SIFT y SURF 73
4.1.4. Transformada de Hough 78
4.1.5. Cúmulos con la transformación de Hough 81
4.1.6. Solución para los parámetros afines 83
4.1.7. RANSAC 86
4.1.8. Conclusiones 96

5. Base de Desarrollo 97
5.0.9. Sistema Implementando 99
5.0.10. Evaluación del Sistema 103
5.1. Conclusión 111

6. Conclusiones 112
6.1. Conclusiones Generales 112
6.2. Trabajos Futuros 113

Referencias 115

Lista de Figuras

2.1. Representación del objeto 12
2.2. Puntos de Interes detectados por a) Harris, b) KLT, c)Operador SIFT d)Operador SURF 17
2.3. Mezcla de Guassianas para las sustracción del fondo. 21
2.4. Espacio propio de descomposición basado en la sustracción de fondo . 22
2.5. Conjunto de filtros rectangulares utilizados por Viola. 27

3.1. Creación del espacio-escala Gaussiano. 35
3.2. En rojo: Píxel en estudio. En verde: Vecinos en escala actual. En amarillo: Vecinos de escala anterior y posterior. 36
3.3. Puntos claves detectados. 37
3.4. De izquierda a derecha aplicación de un umbral mínimo 39
3.5. De izquierda a derecha puntos que no fueron descartados 41
3.6. Arriba: Ventana 16 $\times$16 alrededor del punto de interés. Abajo izquierda: $m(x, y)$. Abajo derecha: $\theta(x, y)$. 42
3.7. Izquierda: Región de gradientes 16 $\times$ 16. Centro: Ventana circular Gaussiana. Derecha: Histograma final del punto de interés. 43
3.8. Izquierda: Ejemplo de histograma con orientación simple. Derecha: Ejemplo de histograma con orientaciones múltiples 44
3.9. Izquierda: Subdivisiones 4 $\times$ 4. Centro: Ventanas circulares gaussianas. Derecha: Descriptor compuesto por 16 histogramas de 8 bins. 45
3.10. Ejemplo de Imagen Integral. 50
3.11. Ejemplo de Imagen Integral, en la imagen de Lena aplicando filtro de caja de 15 $\times$ 15 50
3.12. Ejemplo de Imagen Integral. 51
3.13. Imagen Original Samurái 52
3.14. Izquierda: Imagen con filtros de caja y L=13, Derecha Imagen con $\sigma = 1.708$ 53
3.15. De izquierda a derecha Imagen con filtros de caja cambiando el tamaño L, a su vez filtro gaussiano en un escala mayor. 53
3.16. imágenes integrales permite el aumento de escala del filtro 54

3.17. Filtro D_{yy} (parte superior) y D_{xy} (parte inferior) para dos escalas sucesivas (9×9 15×15). 56
3.18. Representación gráfica de las longitudes de los lados de filtro para tres octavas 57
3.19. Aproximación de los operadores de derivada de orden 1 a lo largo de x 59
3.20. Aproximación de los operadores de derivada de orden 1 a lo largo de x 59
3.21. Filtros de caja de segundo orden 60
3.22. Arriba a la derecha filtro de SURF $\tilde{D}^L_{yy}(x, y)$; 61
3.23. Kernel exacto del Hessiano con una delta de Dirac y su aproximación 62
3.24. Operador Determinante de Hessiano aplicado a la imagen de Lena con $L = 33$ 63
3.25. Filtros Haar wavelet, calculo de la respuesta en la dirección de x(izquierda) y y (derecha) 64
3.26. Asignación de la orientación de cada sector Un vector de orientación es calculado como representante de todas las respuestas Gaussianas . 65
3.27. Respuestas de Haar en las sub-regiones alrededor del punto de interés 67
3.28. Las entradas del descriptor de una sub-región representa el tipo de patrón de intensidad subyacente izquierda 68
3.29. Debido a la integración global SURF, se mantiene más sólido a la perturbación de diferentes imagenes, que el descriptor SIFT más local en funcionamiento. 68

4.1. Ejemplo de la correspondencia de puntos característicos de SURF . . 70
4.2. Correspondencia entre Puntos de Interés 71
4.3. Si el contraste entre dos puntos de interés es diferente (oscuro en un fondo claro vs claro en un fondo oscuro), el candidato no es considerado como correspondencia válida 73
4.4. De izquierda a derecha cálculos de puntos de interés así como las direcciones dominantes de cada punto de SURF y SIFT respectivamente. 74
4.5. De izquierda a derecha cálculos de puntos de interés así como las direcciones dominantes de cada punto de SURF y SIFT respectivamente. 75
4.6. De izquierda a derecha cálculos de puntos de interés así como las direcciones dominantes de cada punto de SURF y SIFT respectivamente. 76
4.7. Obtención del objeto de interés mediante el ratón 77
4.8. Correspondencia de un objeto 78
4.9. Algoritmo para detectar rectas en una imagen 81
4.10. Grafo con las longitudes de las líneas por cada ángulo 82
4.11. Correspondencia de un objeto 91
4.12. Búsqueda del mejor parámetro T 92
4.13. Valores atípicos en la correspondencia 93
4.14. Valores típicos (inliers), valores atípicos (outliers) en la búsqueda de la transformación afín 94

4.15. Correspondencia de un objeto mejores parámetros de T 95

5.1. En blanco puntos de interés generados con datos primitivos de C y OpenCV, Círculos con variedad de colores, puntos de interés generados por funciones métodos de OpenCV optimizadas 98
5.2. Vídeo Taxi de Hamburgo 99
5.3. Sistema implementado 100
5.4. Selección del Objeto de Interés 101
5.5. Puntos Característicos del tanto del objeto de interés como deun cuadro del vídeo 101
5.6. Correspondencia entre la imagen muestra en la secuencia de Vídeo . . 102
5.7. Demarcación del objeto de interés en el vídeo mediante el algoritmo de RANSAC 103
5.8. Detección de objetos, imagen A logo de Ubuntu, imagen B discos collage de Ubuntu. 104
5.9. Detección de objetos con un solo ejemplo imagen A taxi, imagen B recorrido del taxi de Hamburgo. 105
5.10. Punto de intéres aislado 106
5.11. Detección de objetos en la secuencia de vídeo Taxi de Hamburgo . . 107
5.12. Detección de objetos en la secuencia de vídeo Taxi de Hamburgo . . 108
5.13. Detección de objetos en una secuencia de vídeo 109
5.14. Oclusión parcial de objetos en una secuencia de vídeo 110

Lista de Tablas

4.1. Tiempo entre el algoritmo de SURF y SIFT en la Figura 3.33. 74
4.2. Tiempo entre el algoritmo de SURF y SIFT en la Figura 3.34. 75
4.3. Tiempo entre el algoritmo de SURF y SIFT en la imagen 3.35. . . . 75

5.1. Correspondencia de puntos entre la imagen de muestra A, e imagen B del Taxi de Hamburgo 106
5.2. Punto de intéres aislado 107

Lista de Símbolos

I	Imagen original de muestra.
σ	Factor de escala.
s	Sub-niveles o escalas de las octavas generadas.
$L(x, y, \sigma)$	Imagen con un factor de escala determinado.
$G(x, y, \sigma)$	Espacio Escala Gaussiano.
$D(x, y, \sigma)$	Diferencia de Gaussiana.
$\mathbf{x}$	offset del punto característico.
$\widehat{\mathbf{x}}$	función para descartar puntos de bajo contraste
$\mathbf{H}$	Matriz Hessiana para calcular la ubicación y la escala del punto clave
α	Vector propio de $\mathbf{H}$ con la magnitud más grande
β	Vector propio de $\mathbf{H}$ con la magnitud más pequeña
Tr	Traza de Matriz Hessiana.
$Det(H)$	Determinante de la Matriz Hessiana.
r	Relación entre los Vectores propios α y β.
θ	Angulo de inclinación del punto de interés.
m	Modulo de los puntos de interés.
p	Punto de interés.
$H(p, \sigma)$	Hessiano de un punto y escala determinado.
$D_{xx}(p, \sigma)$	Convolución de la derivada de segundo orden con la imagen I en el punto p
M	Tamaño del kernel Gaussiano
L	Tamaño del filtro de caja que se va a utilizar.
I_c	Imagen integral.
B_Γ	Función rectangular.
$\tilde{D}_y^{(L)}$	Harr wavelets en dirección y.
$\tilde{D}_x^{(L)}$	Harr wavelets en dirección x.
$L^*u^*v^*$	Espacios de color
$L^*a^*b^*$	Espacios de color
M_h	Matriz de momentos Harris
$w(x, y)$	vecindario.
$\sigma^2\nabla^2 G$	Laplaciano de Gaussiana
U	Datos observados

Θ	Espacio de parámetros
$\tilde{theta}$	modelo de párametros
P	Probabilidad de una muestra no contaminada.
J_s	Función de costo
η_0	Umbral predefinido.
T	Transformación afín

Capítulo 1

Introducción

Hoy en día el procesamiento digital de imágenes tiene múltiples aplicaciones a nivel industrial, militar, educativo, social, artístico, etc, y poco a poco se ha ido incorporando a nuestra vida cotidiana en aspectos tan simples como útiles, desde una cámara web que mantienen en foco el rostro de la persona que la usa, hasta cámaras fotográficas que detectan los rostros, las condiciones de iluminación y que son capaces de utilizar tales condiciones para determinar el nivel de zoom y filtros necesarios para captar el momento indicado de un rostro sonriente, lo cual, contribuye a la obtención de una mejor fotografía. Por tanto, las imágenes digitales tienen, sin lugar a duda, mucha importancia y se encuentran en aspectos de la vida diaria tan simples que a veces pasamos por alto su existencia. La noción que se tiene de una imagen es la de formas y colores visibles plasmados físicamente en un lienzo, en cambio una imagen digital se puede describir como una imagen representada por bits, los cuales necesitan ser interpretados para poder ser captados por el ojo humano, como un conjunto de puntos de colores llamado píxeles [Yilmaz et al., 2006a] .

Una de las aplicaciones más importantes de las imágenes digitales son los sistemas de visión, que capturan el acontecer del medio en el que se encuentran para que se pueda extraer información relevante y usarla con un fin específico. La rápida mejora en cuanto a calidad y resolución de los sensores de cálculo en la ultima década ha

favorecido la creación de nuevos algoritmos y aplicaciones que potencializan la eficiencia del procesamiento, lo cual es consecuencia directa de la llamada Ley de Moore, que expresa que los precios bajan a medida que las prestaciones suben [Moore, 1965]. Esto trae mejoras significativas a uno de los avances más notables en la historia de la humanidad: la computadora, la cual aumenta sus capacidades a pasos agigantados y da la posibilidad de resolver por software problemas que antes solo se solucionaban por hardware, haciendo posible con ésto la detección y seguimiento de objetos, sin la necesidad de grandes infraestructuras. El seguimiento de objetos es el proceso de estimar en el tiempo la ubicación de uno o más objetos móviles mediante el uso de cámaras [Zhang y Lu, 2001].

Los principales retos que hay que tener en cuenta en el diseño de un seguidor de objetos, están relacionados con la similitud de aspecto entre el objeto de interés y el resto de objetos en la escena, así como la variación de aspecto del propio objeto. Dado que el aspecto tanto del resto de objetos como el fondo, puede ser similar al del objeto de interés, esto puede interferir en el seguimiento, aunado a que el objeto pueda salirse del cuadro de la imagen. Por ello que en el presente capítulo se mencionan las razones que motivaron a la realización de un seguidor, detector de objetos, la metodología utilizada, los alcances del mismo, los requisitos para su funcionamiento y la descripción de los capítulos subsecuentes.

1.1. Motivación

El motivo central, es poder concretar un seguidor detector a tiempo real, donde el objeto de interés no necesite un aprendizaje o estudio previo, donde al detector seguidor se le indique qué seguir en una secuencia de vídeo de origen cualquiera, y por ende, con un solo ejemplo se pueda iniciar la búsqueda. Por tanto el presente trabajo, tiene su motivación en la escasa seguridad que existe en nuestro país, y el aumento creciente de la delincuencia, no obstante, éste, solamente atisba una parte

de trabajos futuros, que centran como base esta tesis, sin embargo es evidente que el presente trabajo contiene en su haber, muchas variables y desventajas, sin embargo, estas desventajas y la inseguridad, son el fundamento de motivación de este trabajo de tesis.

1.2. Planteamiento del Problema

La formulación general del problema de la detección y seguimiento de un objeto, se puede plantear de la siguiente manera: dado una secuencia de vídeo, de una escena en particular, se debe determinar la posición (en la imagen) del objeto de interés, así como la trayectoria que éste genere en los cuadros subsecuentes. De igual forma, retomar la posición del objeto si el seguidor pierde el objeto de interés, validando el objeto a detectar con un conjunto de características que describan al objeto seguido. En cuanto a la parte de la detección, también se suele clasificar en dos problemas bien diferenciados: identificación y verificación, como se menciona en [Yilmaz et al., 2006a]. En los problemas de identificación, la entrada del sistema es la imagen de un objeto desconocido, marcando la identidad del mismo determinada a partir de un conjunto de características extraídas del seguimiento. Por otro lado, los problemas de verificación, el sistema necesita confirmar o rechazar la identidad pretendida de la imagen del objeto de entrada al proceso.

Aunque existen muchas técnicas de seguimiento y detección de objetos, los enfoques que existen, han demostrado eficacia prometedora, pero las tareas de seguimiento y detección de objetos todavía son difíciles de completar satisfactoriamente, debido a las variaciones de iluminación, los contrastes del entorno donde se hace el seguimiento y la posición de los objetos con respecto a la cámara, así como las oclusiones que hacen que el seguidor pierda el objetivo, por ende, estos suelen ser algunos de los principales problemas de la detección y del seguimiento de objetos [Kalal et al., 2010]. El grado de complejidad aumenta en la medida que el entorno donde se implementa el seguimiento

no es un ambiente controlado, por ejemplo: los lugares abiertos, al aire libre. Luego entonces, el problema que se aborda en este trabajo de tesis, radica en la posibilidad de generalizar el objeto de muestra inicial, al poder seleccionar cualquier objeto de interés de una secuencia de vídeo, y a partir de esa muestra única, iniciar la detección y el seguimiento del objeto indicado A diferencia de lo que existe en el estado del arte, no se contempla un entrenamiento previo, no existe un clasificador de base para poder hacer lo antes mencionado. Otro problema importante en el que se centra este trabajo de tesis es: dado una única muestra inicial, poder seguir el objeto, a pesar de las deformaciones rigidas que se generen y los posibles cambios de iluminación, así como la posible perdida del objeto de interés, por inclusiones de objetos diversos contenidos en la escena del vídeo.

1.3. Objetivos y Alcances de la Tesis

1.3.1. Objetivo general

El objetivo general de la presente tesis, es el desarrollo de un sistema, que pueda generalizar la búsqueda y el seguimiento de objetos, que se pueda adaptar a los cambios en el seguimiento del objeto seleccionado por un click y arrastre de ratón, y que a pesar de que la muestra del objeto sufra cambios notables en el transcurso de este proceso, por cambios de iluminación, rotación, escalamiento o cizallamiento, pueda seguirse con el rastreo del objeto, de igual forma, en el caso de que el objeto sea perdido, o cubierto por otro objeto de la escena, (oclusión), se pueda retomar la búsqueda, aunque la apariencia del objeto recapturado no sea la misma que la muestra inicial.

1.3.2. Alcances

Las restricciones y consideraciones en el proceso de detección y seguimiento del objeto respecto del ambiente y las variables involucradas en el presente desarrollo son los que se describen a continuación

- **Cambio de posición**.

  Se asume que el objeto móvil de interés varia su aspecto cuando se proyecta sobre el plano de la imagen, por ejemplo al girar, por lo que en el presente trabajo de tesis se implementa un seguidor que debe de ser robusto a éstos cambios en el objeto.

- **Iluminación ambiente**.

  Se asume que la dirección, la intensidad y el color de la luz de ambiente influye en el aspecto del objeto de interés. Asimismo, los cambios en la iluminación global son con frecuencia un reto en las escenas al aire libre, por tanto se prevee que el seguidor debe adaptarse a estos cambios de iluminación.

- **Ruido**.

  En el presente trabajo, se asume que el proceso de adquisición de imágenes introduce en la señal de la imagen, un cierto grado de ruido que depende de la calidad del sensor. Las observaciones del objeto de interés pueden dañarse y por tanto afectar el rendimiento del seguidor.

- **Oclusiones**.

  En el presente trabajo, se asume que el objeto de interés puede ser parcialmente o totalmente cubierto por otros objetos en la escena, por tanto se prevee que el detector interactué en la perdida total del objeto, para nuevamente ser detectado apenas sea visible en escena y así continuar con el seguimiento del objeto dado. Las oclusiones son generalmente debidas a:

a) Un objeto de interés que se mueve detrás de un objeto estático, como por ejemplo una columna.

b) Otros objetos que se mueven en la escena de manera que entorpecen la visión de un objeto de interés.

1.4. Descripción del Sistema

Kalal propone en [Kalal et al., 2010] la adquisición de un objeto de interés mediante un solo click en un cuadro de la secuencia de vídeo, una vez adquirido el objeto de interés, se inicia el seguimiento y se genera la trayectoria del objeto. Puesto que el enfoque de Kalal, [Kalal et al., 2010] que es una aportación nueva e innovadora al aprendizaje automático y de tiempo real, a partir de la generación de esta trayectoria se va adaptando a las modificaciones del objeto. Debido a que obtiene las características a partir de los pequeños cambios del objeto, para posteriormente ser guardados en una memoria temporal, de esta forma, cuando el objeto se pierde, puede ser detectado nuevamente por esta memoria, clasificador que se va generando en el trascurso del seguimiento. Por ende, es notable sin duda la idea, dado que de un solo ejemplo, se puede hacer el seguimiento y la detección por la memoria generada, es decir, mientras detecto y sigo se que es el objeto que me interesa, de esta forma, se podra discriminar objetos que aparezcan en la escena.
Luego entonces, basándose en la idea de Kalal, donde se puede generalizar lo que se requiere seguir, surge la idea de implementar un seguidor detector con características similares. Basando el trabajo de tesis en algoritmos generadores de características, robustos a la invarianza de rotación, traslación, cambios en la iluminación. Es por eso, que se optó por la inclusión de los algoritmos de SIFT y SURF [Lowe, 2004], [Bay et al., 2008], que se basan en vectores de características, la idea se complementa al observar el aprendizaje del ser humano, que basa la discriminación de objetos y su clasificación en características para poder hacer una identificación adecuada de cualquier

objeto de la vida cotidiana. Debido a ello es que se integran los mencionados algoritmos, dado que a partir de una sola muestra se pueden obtener puntos verdaderamente característicos, con los que es posible hacer la detección y el seguimiento basándose en un solo ejemplo, por tal razón, se describen ampliamente estos dos algoritmos en el Capítulo 3.

1.5. Metas

La principal meta del presente trabajo de tesis, consiste en el desarrollo de un sistema de detección y seguimiento de objetos propio, donde se pueda generalizar lo que se quiere seguir y no se base la búsqueda en un objeto en particular, ahora bien, se dará una sola muestra por un click y arrastre de ratón, teniendo la posibilidad de seleccionar a voluntad un objeto de interés en una escena cualquiera, mismo que respalde los experimentos y pruebas necesarias para determinar sí, a partir de ese solo ejemplo y de sus vectores de características, es posible hacer el seguimiento y la detección de un objeto a pesar de las deformaciones que del objeto se generen.

1.6. Descripción y Organización del Documento

El esquema de lo que resta de este trabajo de tesis es como sigue: el Capítulo 2 describe el estado del arte de la detección y seguimiento de objetos, en algunos casos en tiempo real, en otros sólo la detección de un objeto específico en una imagen fija; ya sea por color, distancia euclidiana, o puntos de interés entre otros.
El Capítulo 3 por con siguiente, profundiza en dos de los algoritmos mencionados en el estado del arte, como lo es el algoritmo de SIFT y SURF. Así mismo, la correspondencias entre las características de una imagen y otra, se aborda ampliamente debido a su implementación en este trabajo de tesis y a su naturaleza, de sólo hacer el seguimiento con un solo ejemplo de un objeto cualquiera.

Posteriormente, el Capítulo 4 aborda el problema de la demarcación del objeto de interés, buscando la transformación afín, homografía del objeto de interés, con la finalidad de que la marca se apegue lo mayormente posible al objeto rastreado. Además, se hace mención de la transformada de Hough y el algoritmo de RANSAC para tal búsqueda.

Enseguida el Capítulo 5 presenta la evaluación del sistema desarrollado, así como el rendimiento de la implementación en cuanto al seguimiento y detección se refiere, dado que solo se da una muestra inicial con la que se pretende hacer la búsqueda y el seguimiento a pesar de que la apariencia del objeto se haya modificado. Finalmente el Capítulo 6 presenta las conclusiones y trabajos futuros así como las referencias.

Capítulo 2

Estado del Arte del Seguimiento y Detección de Objetos

En este capítulo se dará una visión global sobre el estado actual del seguimiento y detección de objetos mediante el procesamiento de imágenes, así como los procesos, pasos previos y algoritmos más utilizados para esta labor. Finalmente se introduce la línea general por la cual se realizarán las implementaciones y el estudio en el que se basa este proyecto.

2.1. Seguimiento de Objetos

El seguimiento de objetos mediante videocámaras es un tema muy explorado en el campo de la visión por computadora debido a su gran utilidad [Yilmaz et al., 2006b]. Esta clase de seguimiento es utilizado en sistemas de reconocimiento basados en movimiento, sistemas de videovigilancia, adquisición de metadatos para bases de datos de vídeo, interacción hombre-máquina, etc. En el tema de la videovigilancia, es un hecho que el análisis de vídeo requerido para un sistema de este tipo se divide en tres principales etapas, las cuales son: detección de objetos interesantes en escena, seguimiento de éstos y posteriormente su reconocimiento. Teniendo en cuenta el objetivo

de este proyecto, únicamente se expondrán las dos primeras etapas de este proceso, las cuales están directamente relacionadas con el tema tratado y se dejará para un estudio posterior todo lo referente al reconocimiento final de dichos objetos.

2.2. Modelado para la representación del objeto

En un escenario de seguimiento, un objeto puede ser definido como algo que es de interés para su análisis. Por ejemplo: los barcos en el mar, los peces dentro de un acuario, los vehículos en una carretera, aviones en el aire, la gente caminando, o burbujas en el agua, son un conjunto de objetos que puede ser importantes para hacer un seguimiento en un dominio específico. Los objetos pueden ser representados por sus formas y apariencias. El primer punto es la detección del objeto en escena, pero para ello, hay que definir antes una representación para estos objetos. Para llevar a cabo esta tarea hay diversos modelos que se pueden dividir en dos grandes grupos: modelos basados en forma y modelos basados en apariencia [Yilmaz et al., 2006b].

Los modelos basados en forma. Utilizan determinadas características referentes a la forma del objeto para hacer el seguimiento. Este tipo de modelos suelen ser los más utilizados para la mayoría de problemas del seguimiento (*tracking*) y existen diversas versiones:

- **Modelo de puntos**: El objeto está representado por un solo punto (su centro) [Veenman et al., 2001] (ver Figura 2.1, (a)), o por un conjunto de puntos [Serby et al., 2004] (ver Figura 2.1, (b)).

- **Modelo de formas geométricas primitivas**: El objeto está representado por una elipse, rectángulo o alguna otra forma geométrica primitiva [Comaniciu et al., 2003a] (ver Figura 2.1, (c)). Esta forma es idónea para representar objetos rígidos pero en la práctica es muy utilizada también para cualquier otro tipo de objetos no-rígidos.

- **Modelo de formas articuladas**: Este método es una evolución del método anterior, ya que se basa en la utilización de formas geométricas primitivas, pero en este caso, para representar cada una de las partes rígidas de un determinado objeto, separadas por una articulación [Comaniciu et al., 2003a] (ver Figura 2.1, (d)).

- **Modelo de silueta y contorno**: Se basa en utilizar el contorno del objeto para definir sus fronteras y así poder obtener la parte interior, llamada silueta [Yilmaz et al., 2004] (ver Figura 2.1, (e)). Este método resulta idóneo para representar objetos no-rígidos complejos, pero es mucho más costoso que los anteriores.

- **Modelo de esqueleto**: Consiste en la extracción del esqueleto del objeto mediante el uso previo de su silueta [Ballard y Brown, 1982] (ver Figura 2.1, (f)). Este método es muy utilizado cuando se tiene previsto realizar el paso de reconocimiento de objetos

El segundo tipo son los **Modelos basados en apariencia**. Los cuales utilizan la información cromática o de contornos que ofrecen cada uno de sus píxeles. Hay un número de formas de representar las características de apariencia de los objetos. Nótese que las representaciones de formas también se puede combinar con las representaciones de apariencia [Cootes et al., 2001] para el seguimiento. Algunas representaciones de apariencia comunes en el contexto del seguimiento de objetos son las siguientes:

- **Modelo de densidades de probabilidad**: Define un objeto únicamente por las estimaciones de las funciones de densidades de probabilidad. Para extraer esta medida existen diversos métodos, ya sea mediante Gaussianas [Zhu y Yuille, 1996], ventanas Parzen, [Elgammal et al., 2002] o histogramas [Comaniciu et al., 2003a]

- **Patrones**: Este modelo se basa en formas geométricas o siluetas predefinidas para identificar objetos en escena [Fieguth y Terzopoulos, 1 27]. Cada una de

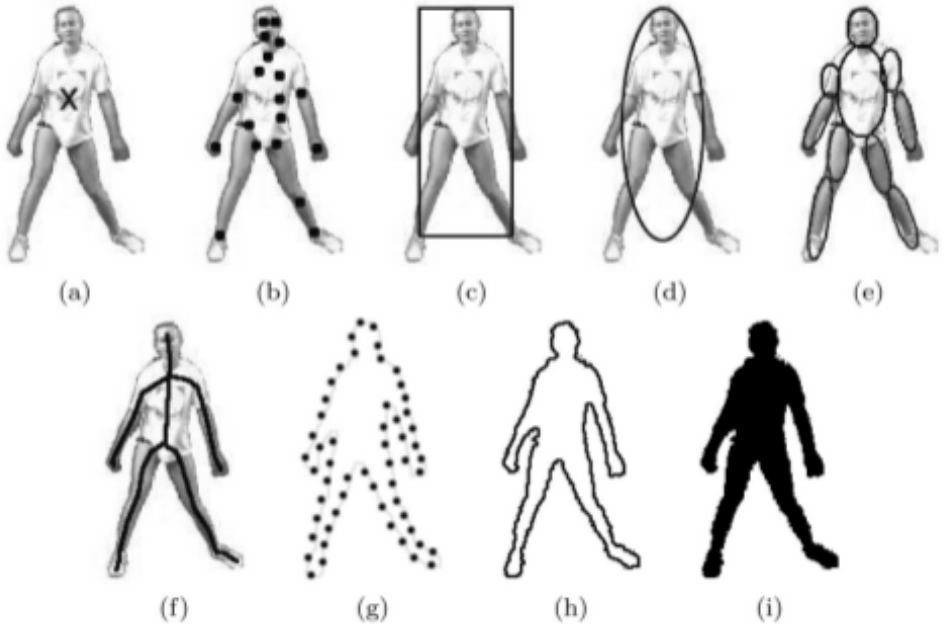


Figura 2.1: Representación del objeto. (a) Centroide (b) Múltiples Puntos, (c) Parche rectangular, (d) Parche elíptico (e) Partes basadas en múltiples parches (f) Esqueleto del objeto, (g) Puntos de control del contorno del objeto (h)Contorno completo del objeto, (h) Silueta del objeto [Yilmaz et al., 2006b].

estas siluetas contendrá información cromática del objeto. Es un método sencillo pero sólo apto para objetos que no varían excesivamente en el tiempo.

- **Modelo de apariencia activo**: Este modelo se basa en generar información referente a la forma y apariencia del objeto. La forma viene definida por una serie de marcas basándose en un sistema similar a la detección del contorno, y cada una de estas marcas contiene un vector de apariencia con diversas características como el color, textura o la magnitud del gradiente [Edwards et al., 1998].

- **Modelo de apariencia multivista**: Este modelo se basa en guardar la información de diferentes vistas de un mismo objeto, para así ser menos susceptible a cambios de forma o apariencia (por ejemplo, personas que giran sobre sí mismas). Para almacenar estas vistas se utiliza una codificación mediante subespacios a través de métodos como el PCA (Principal Component Analysis) [Shlens,

2005a] y el ICA (Independent Component Analysis) [Moghaddam y Pentland, 1997].

.

Otros enfoques para aprender las características del objeto es mediante la información de un conjunto de clasificadores, por ejemplo, las máquinas de soporte vectorial [Avidan, 2001] o redes Bayesianas [Park, 2004]. Una limitación de los modelos de apariencia multivista es que la apariencia en todas las vistas se requieren antes de tiempo. En general, existe una fuerte relación entre las representaciones de objeto y los algoritmos de rastreo. Las representaciones del objetos suelen ser elegidos de acuerdo con el dominio de aplicación. Para el seguimiento de objetos, donde los objetos son pequeños en una imagen, suele ser apropiado, como lo muestra Veenman et al en [Veenman et al., 2001] utilizan la representación de punto para hacer un seguimiento en la secuencia de vídeo de un plato de semillas. Del mismo modo, Shafique et al en [Shafique y Shah, 2003] utilizan la representación punto, para seguir el alejamiento de las aves. Para los objetos cuyas formas se pueden aproximar por rectángulos o elipses, formas geométricas primitivas que son más apropiadas para el seguimiento, como por ejemplo Comaniciu et al en [Comaniciu et al., 2003a] utiliza una forma elíptica y emplea un histograma de los colores a partir de la región elíptica para modelar la apariencia del objeto. En 1998 Black y Jepson [Black y Jepson, 1998] utilizan vectores propios para representar la apariencia del objeto, los vectores propios se generan a partir de plantillas de objetos rectangulares. Para el seguimiento de objetos con formas complejas, como otro ejemplo seria, el de los seres humanos, lo apropiado seria basarse en una silueta que represente a la persona. Un ejemplo de aplicar siluetas a seres humanos es la de Haritaoglu et al en [Haritaoglu et al., 2000] dado que utilizan siluetas para el seguimiento de objetos en una aplicación de vigilancia.

2.3. Selección de características para el seguimiento

Seleccionar las características adecuadas es fundamental en el seguimiento. En general, las características más notables de un objeto facilitan la distinción del objeto en el espacio de características. La selección de características está estrechamente relacionadas con la representación del objeto. Por ejemplo, el color se utiliza como una característica de histograma basado en apariencia representativa del objeto de interés, mientras que para el contorno basado en la representación, los bordes del objeto se utilizan generalmente como características. En general, muchos algoritmos de rastreo utilizan una combinación de estas características. Los detalles de esta selección de características se describen a continuación [Yilmaz et al., 2006a].

- **Color**. El color aparente de un objeto está influenciado principalmente por dos factores físicos, 1) la distribución de energía espectral de la fuente luminosa, y 2) las propiedades de reflectancia de la superficie del objeto. En el procesamiento de imágenes, el color RGB (rojo, verde, azul) de espacio de color se utiliza normalmente para representar el color. Sin embargo, el espacio RGB no es un espacio de color perceptualmente uniforme. Es decir, las diferencias entre los colores en el espacio RGB no se corresponden con las diferencias de color percibidas por los seres humanos [Paschos, 2001]. Además, las dimensiones RGB están altamente correlacionadas. Por el contrario, $L^*u^*v^*$ y $L^*a^*b^*$ son espacios de color perceptualmente uniformes, mientras que el HSV (Tono, Saturación, Valor) es un espacio de color aproximadamente uniforme. Sin embargo, estos espacios de color son sensibles al ruido [Songet 1996] En resumen, no hay una última palabra en la que el espacio de color es más eficiente, para aplicar de una manera más precisa el seguimiento o la detección del objeto.

- **Bordes**. Son los límites de los objetos. Suelen generar fuertes cambios en la

intensidad de la imagen. La detección de bordes se utiliza para identificar estos cambios. Una propiedad importante de los bordes es que son menos sensibles a los cambios de iluminación en comparación con las características de color. Los algoritmos que hacen un seguimiento en base al contorno de los objetos suelen utilizar los bordes, como la característica representativa, debido a su simplicidad y exactitud. El método de detección de bordes más popular es el detector de bordes de Canny [Canny, 1986]. Una evaluación de los algoritmos de detección de bordes es proporcionada por [Bowyer et al., 2001].

- **Flujo óptico**. Flujo óptico es un denso campo de vectores de desplazamiento que define la traslación de cada píxel en una región. Se calcula utilizando la restricción de brillo, que supone la constancia del brillo de los píxeles correspondientes en los marcos consecutivos [Horn y Schunck, 1981].El flujo óptico se utiliza comúnmente como una característica de movimiento basado en la segmentación y el seguimiento de los objetos. Las técnicas más populares para el cálculo de flujo óptico denso incluyen métodos de Horn y Schunck [Horn y Schunck, 1981], Lucas y Kanade [Lucas y Kanade, 1981], Black y Anandan, P. [Black y Anandan, 1996], y finalmente Szeliski y James M. Coughlan [Szeliski y Coughlan, 1997].

Las características mayormente se eligen manualmente por el usuario dependiendo del dominio de la aplicación. Sin embargo, el problema de la función de selección automática ha recibido mucha atención en la comunidad del reconocimiento de patrones. Los métodos automáticos de selección de características se pueden dividir en métodos de filtrado y métodos de contenedor [Blum y Langley, 1997]. Los métodos de filtro intentan seleccionar las características basándose en un criterio general, por lo que las características no deben de ser correlacionadas. El método de contenedor selecciona las características basándose en la utilidad para un problema especifico, donde, la clasificación del rendimiento usa un subconjunto de características. El Análisis de

componentes Principales por sus siglas en ingles (PCA) es un ejemplo del método de filtro para la reducción de características, el cual implica la transformación de un número de posibles variables correlacionadas en un número más pequeño de variables no correlacionadas llamados componentes principales, como lo muestra en [Tieu y Viola, 2004].

2.4. Detección de objetos

Todos los seguidores de objetos requieren un mecanismo de detección de objetos, ya sea en cada cuadro o cuando el objeto aparece en un determinado cuadro del vídeo. Un enfoque común para la detección de objetos es utilizar la información en un solo marco. Sin embargo, algunos métodos de detección de objetos hacen uso de la información temporal, calculada a partir de una secuencia de cuadros, para reducir el número de falsas detecciones. Esta información temporal es usualmente en la forma de diferenciación del marco, que pone de relieve el cambio en las regiones de cada marco consecutivo. Dado el objeto en la imagen, es entonces donde comienza la tarea del seguidor, dando la correspondiente trayectoria en los marcos subsecuentes [Yilmaz et al., 2004]. Enseguida se mencionarán los métodos más populares en el contexto del seguimiento de objetos en aras de la exhaustividad.

2.5. Detección de Puntos de Interés

- **Detector de puntos**: Este tipo de detector se basan en encontrar puntos característicos en una imagen utilizando su textura (Figura 2.2). Hay diversos algoritmos para realizar esta tarea como por ejemplo Moravec en [Moravec, 1979], Harris [Harris y Stephens, 1988a], detector KLT [Shi y Tomasi, 1994] y detector SIFT [Lowe, 2004].

  Para encontrar los puntos característicos, el operador de Moravec calcula la

variación de la intensidad de la imagen en un parche de 4×4 respecto a la horizontal, vertical, diagonal, y antidiagonal, seleccionando el mínimo de las cuatro variantes como valores representativos de la ventana. Un punto se declara interesante si la variación de intensidad es un máximo local en un parche de 12×12 .

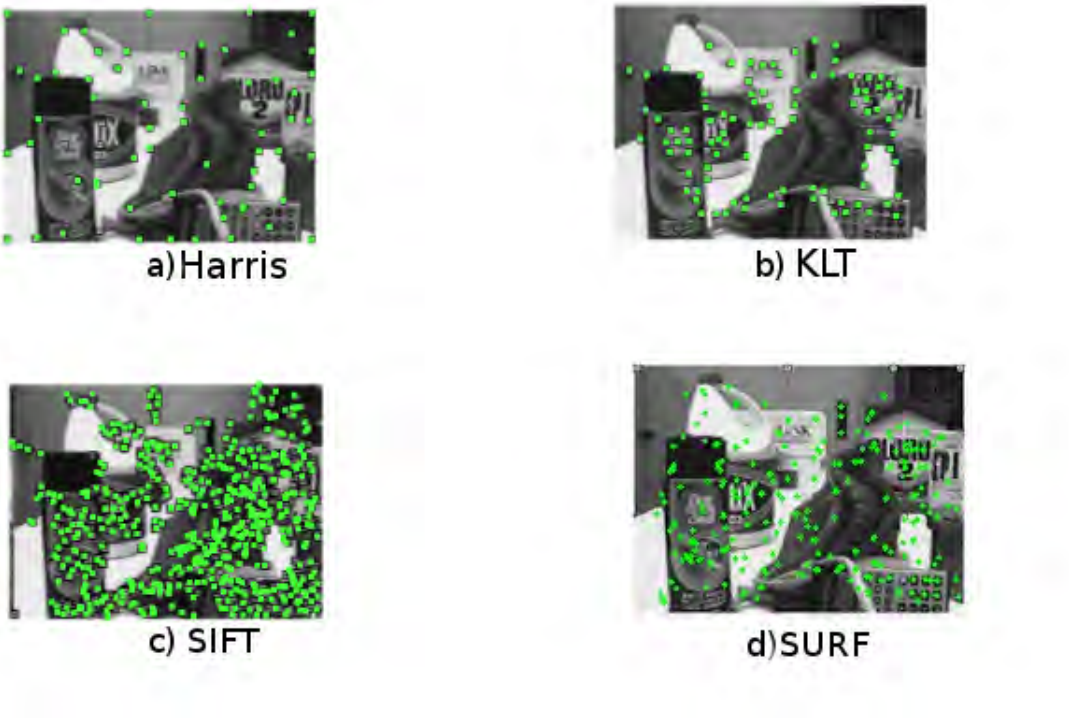


Figura 2.2: Puntos de Interes detectados por a) Harris, b) KLT, c)Operador SIFT d)Operador SURF

El detector de esquinas de Harris [Harris y Stephens, 1988a] es, probablemente, el más comúnmente usado, debido a su elevada invarianza ante escalamiento, rotación, cambios de iluminación y ruido en la imagen. Este detector calcula la derivada de primer orden de la imagen, en las direcciones I_x y I_y para resaltar las variaciones de intensidad de las direcciones, entonces, se calcula la matriz de momentos que se muestra enseguida.

$$M_h = \sum_{x,y \in w(x,y)} \begin{pmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{pmatrix} \tag{2.1}$$

La cual codifica esta variación, se evalúa para cada píxel de un pequeño vecindario $w(x, y)$. Un punto de interés es identificado usando el determinante y la traza de M_h que miden la variación en una vecindario local $R = det(M_h) - k \cdot tr(M_h)^2$ donde k es una constante. El punto de interés es marcado por una umbralización R después de aplicar supresión-non-máxima como se puede ver en la Figura 2.2(a). La misma matriz de momentos M dada en la Ecuación 2.1 se utiliza en la etapa de detección de puntos de interés del seguimiento KLT. La confianza del punto de interés, R, es calculada usando el mínimo valor propio de M_h, λ_{min}. Los puntos de interés candidatos son seleccionados por un umbral R. Entre los puntos candidatos, KLT elimina los candidatos que están especialmente cerca uno del otro. Algunos autores han obtenido buenos resultados experimentales con k = 0.04. Este es el caso de Davison y Murray en [Davison y Murray, 2002], utilizan un detector de esquinas Harris con una subventana w de 15×15.

- **Harris-Laplace**: Los puntos de interés detectados por Harris-Laplace son invariantes a rotación y escala [Mikolajczyk y Schmid, 2001]. Estos puntos son detectados por una función de Harris seleccionada en el espacio de escalas por el operador Laplaciano. La escala seleccionada determina el tamaño de la región de interés. En [Jensfelt et al., 2006] Jensfelt y Kragic, proponen utilizar el detector Harris-Laplace en lugar del detector de diferencia de gaussianas usado originalmente por SIFT. La razón por la que hace esta elección es porque Harris-Laplace detecta características localizadas de forma más precisa espacialmente, lo cual, es mejor cuando estas características son usadas para reconstrucción y localización y no sólo para reconocimiento.

- **SURF**: (*Speeded Up Robust Features*) es un detector invariante a rotación y a

escala y un descriptor presentado por Bay en [Bay et al., 2008]. Este detector se basa en la matriz Hessiana por su precisión y el bajo coste computacional. Según sus autores, este método supera a otros métodos existentes respecto a la repetibilidad, robustez y distinción de los descriptores. Al igual que con las características SIFT, se utiliza sólo el detector SURF dejando a un lado su descriptor, ya que se considera parte de un problema diferente.

- **SIFT**: El descriptor SIFT cuyo acrónimo hace referencia al título de *Scale-Invariant Feature Transform*, fue desarrollado y presentado por Lowe en [Lowe, 2004] como un algoritmo capaz de detectar puntos característicos estables en una imagen. Estos puntos son invariantes frente a diferentes transformaciones como traslación, escala, rotación, iluminación. Originalmente fue desarrollado para el reconocimiento de objetos de manera general. Realiza la correspondencia entre puntos basada en los vectores de características de cada punto que componen el descriptor de la imagen.

2.6. Sustracción del Fondo

La detección de objetos puede lograrse mediante la construcción de una representación de la escena, llamado modelo de fondo para luego encontrar desviaciones del modelo de cada trama entrante. Cualquier cambio significativo en una región de la imagen del modelo de fondo significa un objeto en movimiento. Los píxeles que constituyen las regiones en proceso de cambio están marcados para su posterior procesamiento. Por lo general, se aplica un algoritmo de componentes principales para obtener regiones conectadas correspondientes a los objetos. Este proceso se conoce como la sustracción de fondo.

La sustracción del fondo ha sido estudiada desde finales de los 70 [Jain y Nagel, 1979]. Sin embargo, este enfoque se hizo popular a raíz de la obra de Wren et al en [Wren et al., 1997]. Con el fin de obtener cambios graduales en el tiempo, Wren [Wren

et al., 1997], propone modelar el color de cada pixel $I(x, y)$ de un fondo estacionario con una sola gaussiana $3D$ (Y,U y V espacio de color) , $I(x, y) \sim N(\mu(x, y), \sum(x, y))$. Los parámetros del modelo, la media $\mu(x, y)$ y la covarianza $\sum(x, y)$, son aprendidos de la observaciones de los colores en los fotogramas consecutivos. Una vez que el modelo de fondo es derivado, en cada uno de sus píxeles, la probabilidad de que el color proceda de $N(\mu(x, y), \sum(x, y))$ es calculado, y los píxeles que se desvían del modelo se etiquetan como los píxeles del primer plano.

Sin embargo, una sola Gaussiana no es un buen modelo para escenas al aire libre [Gao et al., 2000] dado que múltiples colores se pueden observar en una determinada ubicación, debido al movimiento repetitivo de objetos, sombras o reflectancia. Una mejora sustancial en el modelado del fondo se consigue mediante modelos estadísticos multimodales para describir por píxel el color del fondo. Por ejemplo, Stauffer en [Stauffer y Grimson, 2000] utilizan una mezcla de Gaussianas para modelar el color del píxel. En este método, un píxel de un fotograma se compara con el modelo de fondo, comparando cada gaussiana hasta una gaussiana coincidente. Si se encuentra una coincidencia, la media y la varianza de la Gaussiana correspondiente, se actualiza. De lo contrario, una nueva gaussiana con la media igual a la del color del píxel actual y algunas variaciones iniciales se introducen en la mezcla. Cada píxel se clasifica en función de sí la distribución correspondiente representa el fondo del proceso. Las regiones en movimiento, que se detectan utilizando este enfoque, junto con los modelos de fondo, se muestra en la Figura 2.3.

Otro enfoque es incorporar regiones basándose en la información de la escena en lugar de sólo el uso de la información del color. Elgammal et al en [Elgammal et al., 2000] emplearon la estimación de densidad de un kernel no paramétrico para modelar el fondo del píxel. Durante el proceso de sustracción, el píxel actual se corresponde no sólo con el píxel actual en el modelo de fondo, sino también con los más cercanos. Por lo tanto, este método puede manejar fluctuaciones en la cámara o pequeños movimientos en el fondo. Li y Leung en [Li y Leung, 2002] fusionaron las características de textura

Figura 2.3: Mezcla de Guassianas para las sustracción del fondo. (a)Secuencia de imágenes de una persona caminando en la escena, (b)El valor más alto de la media en cada posición del píxel, la cual representa a los colores del píxel más persistente, por lo tanto representa el fondo fijo (c)Los valores de la Gaussiana con el segundo peso más alto, de los cuales representan los colores que se observan con menos frecuencia,(d) Resultado de la sustracción del fondo [Yilmaz et al., 2006a].

y color para llevar acabo la sustracción del fondo sobre los bloques de 5×5 píxeles. Dado que la textura no varia mucho con cambios de iluminación, el método es menos sensible a la iluminación. Toyama et al en [Toyama et al., 1999] propone un algoritmo de tres niveles para hacer frente al problema de la sustracción de fondo. Además de la sustracción de niveles de píxel, los autores utilizan la región y la información de nivel del marco. En el nivel del píxel, los autores proponen utilizar el filtrado de Wiener [Wiener, 1964] para hacer predicciones probabilísticas del color del fondo esperado.

Un enfoque alternativo para la sustracción del fondo es la de representar las variaciones de intensidad de un píxel en una secuencia de imágenes como estados discretos que corresponden a los acontecimientos en el medio ambiente. Por ejemplo, para el seguimiento de los automóviles en una autopista, los píxeles del fondo constituyen un estado, el automóvil sería parte del primer plano el cual conformarían otro de los estados y así mismo las sombras que se proyecten seria otro estado. Rittscher et al en [Rittscher et al., 2000] emplearon modelos ocultos de Markov (HMM) para clasificar bloques pequeños de la imagen como pertenecientes a los estado antes mencionados. En el contexto de la detección de la luz dentro y fuera de los eventos,

de una habitación . Stenger et al en [Stenger et al., 2001] utilizaron HMMs para la sustracción del fondo, La ventaja de utilizar modelos ocultos de Markov (HMM) es que ciertos eventos, que son difíciles de modelar correctamente utilizan enfoques no supervisados, que pueden aprender usando un conjunto de entrenamiento. En lugar de modelar la variación de los píxeles individuales, Nuria M. Oliver en [Oliver et al., 2000] propone un enfoque integral mediante la descomposición de eigen espacios. Para las k marcos de entrada $I_i : i = 1...k$, de tamaño $n \times m$, una matriz de fondo B de tamaño $k \times L$ es formada por una cascada de m filas en cada cuadro uno tras otro, donde $I = (n \times m)$ y la descomposición del valor propio, se aplica a la covarianza de B, es decir $C = B^T B$. El fondo se representa por los eigen vectores η más descriptivos U_i, donde $i < \eta < K$ embarcan todas las posibles iluminaciones en el campo de visión (FOV) por sus siglas en ingles. Por lo tanto, este enfoque es menos sensible a la iluminación, los objetos en el primer plano son detectados por las proyección de la imagen, para el espacio propio y encontrar las diferencias, las imágenes reconstruidas y las reales. Se muestran las regiones de objetos detectados utilizando el enfoque de espacio propio en la Figura 2.4.

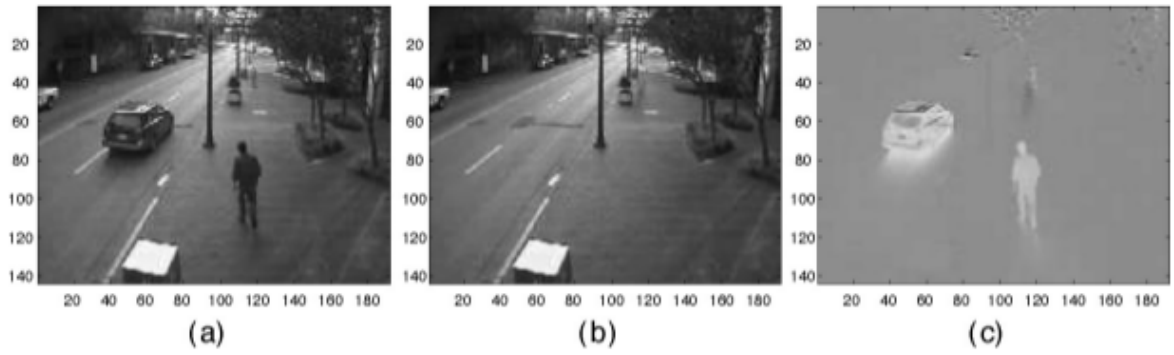


Figura 2.4: Espacio propio de descomposición basado en la sustracción de fondo (el espacio se construye con los objetos en el campo de visión de la cámara): (a)Una imagen de entrada con objetos (b) imagen reconstruida después de la proyección de imagen de entrada en el Eigen espacio (c)la diferencia de la imagen, se tiene en cuenta que los objetos en el primer plano son claramente identificables. [Yilmaz et al., 2006a]

Una limitación de los enfoques antes mencionados es que requieren un fondo controlado. Esta limitación se aborda por Monnet et al en [Monnet et al., 2003], Zhong y Sclaroff en [Zhong y Sclaroff, 2003]. Ambos métodos son capaces de hacer frente a la variación del fondo en el tiempo (por ejemplo, las olas en el agua, el movimiento de las nubes, las escaleras mecánicas) . Estos métodos modelan las regiones de la imagen con el proceso autorregresivo (ARMA) que proporcionan un camino para aprender y predecir los patrones de movimiento en la escena. Un proceso ARMA es un modelo de serie de tiempo que se compone de la suma de los componentes autorregresivos y media móvil, donde un procesos autorregresivo puede ser descrito como una suma ponderada de los valores anteriores y así mismo, un error de ruido blanco.

En resumen, la mayoría del estado del arte del seguimiento, para cámaras fijas, por ejemplo, Haritaoglu et al en [Haritaoglu et al., 2000] y Collins et al en [Collins et al., 2001] usan métodos de sustracción de fondo para detectar regiones de interés. Esto se debe a los últimos métodos de sustracción que tienen la capacidad de modelar la iluminación cambiante, el ruido y el movimiento periódico de las regiones del fondo y por tanto, pueden detectar con precisión los objetos en una variedad de circunstancias. Además, estos métodos son computacionalmente eficientes. En la práctica, la sustracción del fondo proporciona regiones incompletas de objetos en muchos casos. Es decir, los objetos pueden ser puestos en diferentes regiones, o puede haber agujeros en el interior del objeto, ya que no hay garantías de que las características del objeto será diferente de las características del fondo. La limitación más importante de la extracción del fondo es el requisito de cámaras fijas. El movimiento de la cámara normalmente distorsiona los modelos de fondo. Estos métodos se pueden aplicar al vídeo adquirido por cámaras móviles por los modelos de fondo para la regeneración de las pequeñas ventanas temporales, por ejemplo, tres marcos, a partir de cero [Kanade et al., 1998] o por el movimiento de compensación del sensor, por ejemplo, la creación de fondo mosaico [Rowe y Blake, 1996], [Irani y Anandan, 1998]. Sin embargo, ambas soluciones requieren supuestos en escenas planas y movimientos pequeños en cuadros

sucesivos.

2.7. Segmentación

El objetivo de los algoritmos de segmentación de la imagen es para dividir la imagen en regiones perceptualmente similares [Shi y Malik, 2000]. Cada algoritmo de segmentación hace referencia a dos problemas, los criterios para una buena partición y el método para lograr la partición eficiente [Shi y Malik, 2000].

- **Agrupación Media-Cambiante** ($MeanShiftClustering$)

  Para el problema de la segmentación de imágenes, Comaniciu et al en [Comaniciu et al., 2002] proponen el enfoque de media cambiante (Mean-Shift) para encontrar grupos en el espacio de color $[l, u, v, x, y]^T$ donde $[l, u, v]^T$ representan el color y $[x, y]^T$ representa la localización espacial. Dada una imagen el algoritmo, se inicializa con una gran número de hipótesis de los centroides de los cúmulos elegidos al azar a partir de los datos. Entonces, cada centroide del cúmulo es trasladado a la media de los datos que yacen en el interior de la elipsoide multidimensional centrada en el centro del cúmulo.

2.8. Aprendizaje supervisado

La detección de objetos puede ser realizada mediante el aprendizaje de cambios del objeto en forma automática a partir de un mecanismo de aprendizaje supervisado. Aprender de las diferentes formas del objeto exime de la obligación de almacenar un conjunto completo de plantillas, dado un conjunto de ejemplo de aprendizaje. Los métodos de aprendizaje supervisado generan una función que mapea a partir de las entradas salidas deseadas. Una formulación estándar de aprendizaje supervisado es el problema de la clasificación, donde el aprendizaje aproxima el comportamiento de una función a partir de la generación de una salida en la forma de un valor constante,

llamada regresión o etiquetación de clases que de igual forma podremos llamarla clasificación. En el contexto de la detección de objetos, los ejemplos de aprendizaje se componen de pares de características de objeto y una clase de objeto asociado en ambas cantidades se define manualmente.

La selección de características juega un papel importante en el rendimiento de la clasificación. Por lo tanto, es importante utilizar un conjunto de características que discriminen una clase de la otra. Además de las características mencionadas anteriormente, es posible incluir más características tales como el área del objeto, la orientación del objeto, el aspecto del objeto en forma de función de densidad, un ejemplo de ello es el histograma. Una vez que las características son seleccionadas, los cambios de aspecto del objeto pueden ser aprendidos por la elección del método de aprendizaje supervisado. Estos enfoques de aprendizaje son los que se mencionan a continuación: redes neuronales, [Rowley et al., 1998], boosting adaptativo [Viola et al., 2005], arboles de decisión [Grewe y Kak, 1995], y las máquinas de soporte vectorial [Papageorgiou et al., 1998]. Estos métodos de aprendizaje calculan una hipersuperficie que separa una clase de la otra, en un espacio tridimensional. Los métodos de aprendizaje supervisado por lo general requieren una gran colección de muestras del objeto que se desea aprender. Adicionalmente, la colección de muestras deben de ser etiquetadas manualmente. Un posible enfoque para reducir la cantidad de datos de forma manual de las etiquetas es la Co-formacion ($Co - training$) el cual es un algoritmo del aprendizaje máquina que se utiliza sólo cuando hay pequeñas cantidades de datos etiquetados y grandes cantidades no etiquetados [Blum y Mitchell, 1998]. La idea principal detrás del $Co - training$ es la formación de dos clasificadores, utilizando un pequeño conjunto de datos etiquetados, donde las características utilizadas para cada clasificador son independientes. Después del entrenamiento, cada clasificador es usado para asignar datos no etiquetados al conjunto de entrenamiento del otro clasificador. Se ha demostrado que, a partir de un pequeño conjunto de datos etiquetados con dos conjuntos de características estadísticamente independientes, $Co - training$

puede proporcionar una regla de clasificación muy precisa [Blum y Mitchell, 1998]. Co-training ha sido utilizado con éxito para reducir la cantidad de interacciones manuales requeridas para el entrenamiento en el contexto del AdaBoost [Levin et al., 2003] y las máquinas de soporte vectorial [Kockelkorn et al., 2003].

- **Impulso adaptable (Adaptive Boosting)**: El Impulso (*Boosting*) es un método iterativo para encontrar un clasificador muy preciso mediante la combinación de muchos clasificadores, de los cuales, solo pueden ser moderadamente precisos [Freund y Schapire, 1997], la fase de entrenamiento del algoritmo Adaboost, el primer paso es construir una distribución inicial de los pesos sobre el conjunto de entrenamiento. El mecanismo de impulso (*boosting*) selecciona un clasificador de base que proporciona el error mínimo, donde el error es proporcional a los pesos de los datos clasificados erróneamente. Enseguida, los pesos asociados con los datos mal clasificados por el clasificador de base se incrementan. Así, el algoritmo alienta a la selección de otro clasificador que tiene mejor rendimiento en los datos mal clasificados en la siguiente iteración.

  En el contexto de la detección de objetos, los clasificadores débiles pueden ser operadores simples tales como un conjunto de umbrales, aplicados a las características de objetos extraídos de la imagen. En [Viola et al., 2005] utiliza el Framework Adaboot para detectar peatones. En su enfoque, perceptrones fueron elegidos como los clasificadores débiles, los cuales eran entrenados en base a las características obtenidas de la imagen, mediante una combinación de operadores espaciales y temporales. Los operadores de extracción de características tienen la apariencia de filtros rectangulares, y se muestran en la Figura 2.5. Los operadores en el dominio temporal tiene la apariencia de la diferenciación de los marcos que codifican información del movimiento. La diferenciación de marcos, fotogramas en el dominio temporal, reduce el número de falsas detecciones, mediante la aplicación de la detección de objetos en las regiones donde se produce

movimiento.

Figura 2.5: Conjunto de filtros rectangulares utilizados por Viola [Viola et al., 2005] para extraer características usadas en el Framework Adaboost. Cada filtro es compuesto por tres regiones, blanco, gris claro, gris oscuro con peso asociados 0, -1, y 1 respectivamente, con el fin de calcular las características en una ventana, estos filtros son convolucionados con la imagen.

2.9. Modelos de Seguimiento (Tracking)

Una vez escogido el método para detectar los objetos en escena, el siguiente paso es escoger un modelo para realizar el propio seguimiento (tracking) del objeto. Para ello, al igual que en los pasos anteriores se dispone de diversas posibilidades que se pueden dividir en tres grandes grupos: seguimiento por puntos, núcleo y silueta [Yilmaz et al., 2006b].

1. **El seguimiento por puntos**: El seguimiento por puntos se usa en el caso de tener un modelo de objeto basado en puntos. El objetivo es relacionar los puntos en un determinado cuadro, con los mismos puntos en el cuadro anterior. Los algoritmos para establecer estas correspondencias se dividen en dos grandes grupos:

   - **Métodos deterministas de correspondencia**: En este tipo de métodos se relacionan los puntos de un cuadro en un instante de tiempo $t-1$ con los mismos puntos en t. Para ello, se utilizan todas las combinaciones posibles a la hora de asociar dichos puntos y posteriormente se escogen las

correspondencias correctas mediante métodos de asignación óptima, como, por ejemplo, el Algoritmo Hungarian [Kuhn y Yaw, 1955].

- **Métodos estadísticos de correspondencia**: En este caso, se trata al igual que en el caso anterior, de encontrar relaciones entre puntos. Pero esta vez, teniendo en cuenta una cierta incertidumbre (cuantificada en forma de error) en las medidas [Isard y Blake, 1998].

2. **El seguimiento de Núcleo (Kernel Tracking)**: A diferencia del anterior, basado en puntos muy concretos de la imagen, éste se basa en una región primitiva de donde se extraen una serie de parámetros para realizar el seguimiento de cada objeto. Se puede dividir este tipo de seguimiento en tres métodos:

   - **Seguimiento basado en patrones**: Se basa en la utilización de un patrón para buscar el objeto a través de la imagen, normalmente mediante correlaciones [Schweitzer et al., 2002] o algún método similar. Este método tiene bastantes pérdidas cuando la apariencia del objeto varía mucho durante la escena. Sin embargo, funciona muy bien para el seguimiento de objetos que no modifican su forma.

   - **Seguimiento basado en modelos de apariencia**: Este modelo se basa en la búsqueda de ciertas características del objeto, por ejemplo, podrían ser histogramas de color, gradientes u otros modelos de obtención de densidad, mediante los cuales prevé la dirección en la que se desplaza el objeto sin necesidad de recorrer toda la imagen. Uno de estos métodos es el Mean Shift media cambiante (*Mean Shift*) [Comaniciu et al., 2003b].

   - **Seguimiento mediante modelos de apariencia multivista**: Consisten en aumentar el aprendizaje de un objeto basándose en múltiples vistas generadas de forma *offline*. Es decir, con un procesamiento previo a la exposición del sistema a un seguimiento a tiempo real, con el objetivo de

hacer al seguimiento menos susceptible a cambios drásticos de la forma del objeto. Para ello, existe un método basado en subespacios llamado Eigenspace [Black y Jepson, 1998], basado en (PCA) [Shlens, 2005b].

3. **Seguimiento Basado en Siluetas**:está orientado al seguimiento de objetos complejos, para los cuales no se puede usar una simple forma geométrica. Este tipo de seguimiento se puede dividir en dos grandes grupos:

   - **Seguimiento de forma**: Este método es similar al seguimiento mediante patrones, ya mencionado anteriormente, pero utilizando siluetas complejas como patrón de búsqueda [Cole et al., 2004].
   - **Seguimiento de contorno**: Este método no considera una silueta fija, sino que intenta seguir cuadro a cuadro la evolución del contorno de cada objeto para obtener así un seguimiento más preciso en objetos que tienden a cambiar de forma durante la escena [Isard y Blake, 1998].

2.10. Conclusión

Una vez que se ha indagado en el estado del arte del seguimiento y detección de objetos, los algoritmos, métodos, ideas generalizadas sobre este tema de interés, con las características necesarias para apegarse a la idea principal de este trabajo de tesis, serían los algoritmos de SIFT y SURF [Lowe, 2004], [Bay et al., 2008]. Para ello, se considera importante profundizar sobre el funcionamiento de estos dos algoritmos, dado su robustez que se mencionan es sus correspondientes investigaciones. Donde se muestra que sobresalen de los demás algoritmos debido a que presentan distintas ventajas que no tienen los demás algoritmos mencionados. Se dejaron de lados algunos otros algoritmos de vital importancia y notable trascendencia en el estado del arte del seguimiento, como lo es el seguimiento mediante el flujo óptico. Dado que tiene algunas limitaciones en cuanto a la perdida del objeto se refiere, de igual forma el numero

de niveles de las pirámides que se necesitan para poder implementarlo. Es necesario un algoritmos con los cuelas se pudiera basar el seguimiento en características que no se pudieran mezclar con el entorno donde se implementa el seguimiento, por eso el algoritmo *Meanshift Camshift* basando el seguimiento en un histograma no es apto para esta implementación. Es evidente entonces que, basándose en estos algoritmos SIFT y SURF, poder seguir un objeto en tiempo real con constantes deformaciones geométricas es posible, a pesar de sólo dar una muestra inicial. Surge entonces la conclusión de optar por estos dos algoritmos, dadas las características antes mencionadas y por ende se describirán más afondo en el capitulo siguiente para constatar si efectivamente es posible con una sola muestra hacer seguimiento en tiempo real.

Capítulo 3

Detección de objetos

3.1. SIFT

El algoritmo de SIFT proporciona una imagen en coordenadas invariantes a la escala en el ámbito local. A partir de las características locales, se busca conseguir invariancia a la escala, a la orientación, parcialmente a cambios de iluminación, etc. También se puede utilizar para buscar correspondencia entre diferentes puntos de vista de una misma escena. El algoritmo SIFT se compone principalmente de cuatro etapas que se describen siguiendo la implementación de Lowe [Lowe, 2004].

1. **Detección de Extremos en el Espacio Escala**: La primera etapa del algoritmo realiza una búsqueda sobre las diferentes escalas y dimensiones de la imagen identificando posibles puntos de interés, invariantes a los cambios de orientación y escalado. Esto se lleva a cabo mediante la función Diferencia de Gaussiana DoG(*Difference of Gaussian*).

2. **Localización de puntos clave**: Para seleccionar los puntos clave, también llamados puntos de interés, de forma precisa, se aplica una medida de estabilidad sobre todos ellos para descartar aquellos que no sean adecuados.

3. **Asignación de la Orientación**: Se asignan una o más orientaciones a cada

punto de interés extraído de la imagen basándose en las direcciones locales del gradiente de la imagen. Todas las operaciones posteriores son realizadas sobre los datos transformados según la orientación, escala y localización dentro de la imagen asignados en esta etapa, proporcionando así la invarianza respecto de estas transformaciones.

4. **Descriptor del Punto de Interés**: La última etapa hace referencia a la representación de los puntos clave como una medida de los gradientes locales de la imagen en las proximidades de dichos puntos clave y respecto de una determinada escala. Cada punto de interés corresponde a un vector de características compuesto por 128 elementos, que le confiere una invarianza parcial a deformaciones, así como a cambios de iluminación.

Un aspecto importante de este enfoque, es que genera un gran número de características, que cubren densamente la imagen bajo todo el rango de escalas y ubicaciones. Una imagen típica de 500 $\times$ 500 píxeles, dará lugar a aproximadamente 2000 características estables (aunque este número depende del contenido de la imagen y de varios parámetros). La cantidad de características, es particularmente importante para el reconocimiento de objetos. Donde la capacidad de detectar pequeños objetos en fondos no controlados, requiere que por lo menos 3 características de cada objeto coincidan correctamente para una identificación fiable.

3.1.1. Detección de Extremos en el Espacio Escala

La primera etapa de la detección de puntos de interés es identificar las ubicaciones y escalas que se puedan repetir bajo diferentes puntos de vista de un mismo objeto. El descriptor SIFT es construido a partir del espacio-escala Gaussiano de la imagen original, en el cual se pueden detectar de manera efectiva las posiciones de los puntos claves, invariantes a cambios de escala de la imagen. El espacio-escala Gaussiano de una imagen $L(x, y, \sigma)$ es definido como la convolución de funciones en dos dimensiones,

Gaussianas $G(x, y, \sigma)$ de diferentes valores σ con la imagen original $I(x, y)$:

$$L(x, y, \sigma) = G(x, y, \sigma) * I(x, y). \tag{3.1}$$

donde

$$G_\sigma(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{3.2}$$

donde $[x, y]^T$ son las coordenadas espaciales y σ que es utilizada como el factor de escala. Para detectar puntos claves estables en el espacio-escala, no se utiliza la función L calculada anteriormente sino una función que deriva de ella: DoG (Diferencia de Gaussiana) que se forma a partir de la derivada de la Gaussiana escalada espacialmente. Esta función DoG, $D(x, y, \sigma)$, se obtiene mediante la sustracción de escalas posteriores en cada octava, de la cual se abordara su descripción en los párrafos subsecuentes:

$$D(x, y, \sigma) = L(x, y, k\sigma) - L(x, y, \sigma). \tag{3.3}$$

Se utiliza diferencia de gaussiana debido a que se aproxima considerablemente a el Laplaciano de Gaussiana ($\sigma^2\nabla^2 G$) analizado por, Lindeberg en [Lindeberg, 1994]. Lindeberg mostró que la normalización del laplaciano con un factor σ^2 es requerido para una verdadera invarianza a escala. En una detallada comparación experimental desarrollada por Mikolajczyk, y Schmid en [Mikolajczyk y Schmid, 2004] se encontró que los máximos y mínimos de esta función proporcionan las características más estables comparado con otras funciones utilizadas en el estado del arte (Hessiano, gradiente, Harris). La relación entre D y $\sigma^2\nabla^2 G$ se puede comprender a partir de la ecuación de difusión de calor (parametrizando en términos de σ en lugar de lo más habitual $t = \sigma$):

$$\frac{\partial G}{\partial \sigma} = \sigma\nabla^2 G. \tag{3.4}$$

De esto, se puede ver que $\nabla^2 G$ puede ser calculada por una aproximación de diferencias finitas $\frac{\partial G}{\partial \sigma}$, mediante las diferencias de escalas cercanas $k\sigma$ y σ:

$$\sigma \nabla^2 G = \frac{\partial G}{\partial \sigma} \approx \frac{G(x, y, k\sigma) - G(x, y\sigma)}{k\sigma - \sigma}. \tag{3.5}$$

Por lo tanto,

$$G(x, y, k\sigma) - G(x, y, \sigma) \approx (k-1)\sigma^2 \nabla^2 G. \tag{3.6}$$

Esto muestra que la función diferencia de gaussiana tiene diferentes escalas por un factor constante que ya incorpora la normalización de escala σ^2 requerido para el Laplaciano invariante a escala. El factor $(k-1)$ en la ecuación es una constante en todas las escalas y, por lo tanto no influye en la localización de extremos es por eso que se aproxima mediante la diferencia de Gaussiana debido a la complejidad del cálculo del Laplaciano.

Al conjunto de las imágenes Gaussianas suavizadas junto con las imágenes DoG se le llama octava. El conjunto de las octavas es construido mediante el muestreo sucesivo de la imagen original por un factor de 2. Cada una de las octavas (i.e., duplicado σ) es a su vez dividida en un numero entero de sub-niveles o escalas s. Una vez se ha procesado una octava completa, la primera imagen de la siguiente octava se obtiene mediante el muestreo de la primera de las imágenes de la octava predecesora con un valor de σ del doble respecto a la actual.

Esto se traduce en una gran eficiencia del algoritmo para un número de escalas pequeño. El proceso descrito puede verse representado en la Figura 3.1. Dado que el espacio-escala $L(x, y, \sigma)$ representa la misma información a diferentes niveles de escala, el modo particular del muestreo permite una reducción de la redundancia. De esta manera se producen $s + 3$ imágenes por cada una de las octavas y por lo tanto $s + 2$ DoG imágenes donde se llevará a cabo la búsqueda de extremos. De acuerdo con los resultados de David Lowe en [Lowe, 2004], es el valor de $s = 3$ el que mejores

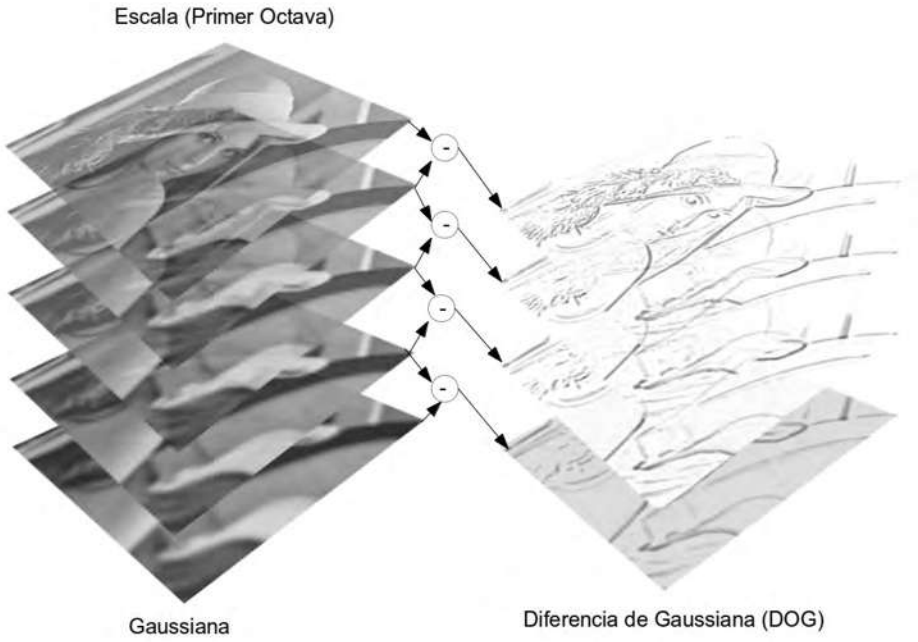


Figura 3.1: Creación del espacio-escala Gaussiano. En cada una de las escalas, también llamadas octavas, la imagen se convoluciona repetidamente con funciones Gaussianas para producir el conjunto de imágenes gaussianas mostradas en la parte izquierda de la imagen. Las imágenes obtenidas son substraídas en parejas adyacentes para producir las imágenes DOG mostradas a la derecha. Después de cada octava, las imágenes Gaussianas son muestreadas por un factor de 2, y se repite el proceso. Fuente [Lowe, 2004]

resultados consigue.

A partir de los cálculos anteriores, se obtienen los máximos y mínimos locales del espacio $D(x, y, \sigma)$. Todos los píxeles de cada imagen de la pirámide son comparados con sus ocho vecinos de la propia imagen y con los nueve vecinos de la escala anterior y posterior, lo cual es mostrado en la Figura 3.2.

Un punto será seleccionado como punto clave sólo si es mayor que sus 26 vecinos o menor que todos ellos. El costo de realizar todas las comparaciones no es elevado, ya que la mayoría de puntos se van descartando a medida que se explora la imagen. Cabe señalar que sólo se podrán detectar puntos clave en escalas centrales de $D(x, y, \sigma)$,

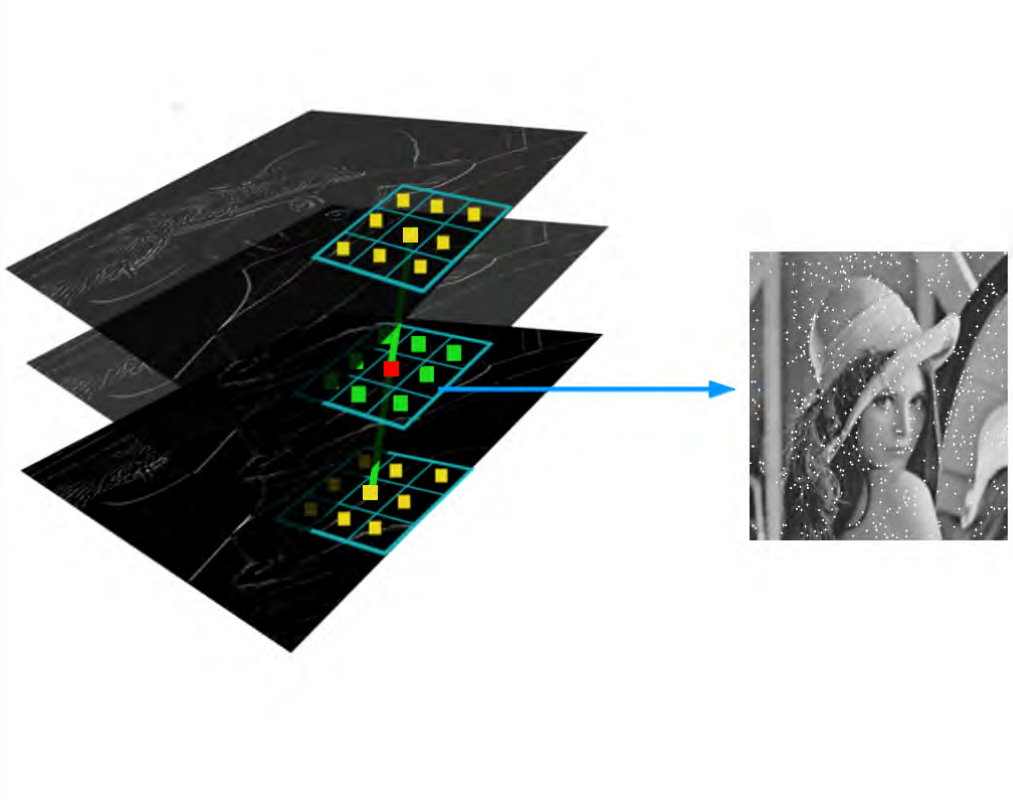


Figura 3.2: En rojo: Píxel en estudio. En verde: Vecinos en escala actual. En amarillo: Vecinos de escala anterior y posterior.

debido a que no existen imágenes vecinas en las escalas laterales.

3.1.2. Refinamiento de Puntos Clave

Una vez que un punto clave es encontrado mediante la comparación de un píxel con sus vecinos, el siguiente paso, es llevar acabo un ajuste detallado de los datos cercanos para la ubicación, escala y la proporción de la curvatura principal. Esta información permite rechazar a los puntos que tienen poco contraste (y son por lo tanto, sensibles al ruido) o están mal localizados a lo largo de un borde. La implementación inicial de este enfoque [Lowe, 1999] simplemente encuentra puntos claves en la ubicación y escala del punto de muestra central. Sin embargo, recientemente Brown ha desarrollado un método en [Brown y Lowe, 2002b] para el montaje de una función cuadrática 3D

Figura 3.3: Puntos claves detectados.

a los puntos de muestra de la zona, para determinar la ubicación de interpolación máxima. Sus experimentos demostraron que esto proporciona una mejora sustancial a la igualación y la estabilidad. El enfoque utiliza la expansión de Taylor (hasta los términos cuadráticos) de la función de espacio-escala, $D(x, y, \sigma)$, cambia de manera que el origen está en el punto de muestra:

$$D(\mathbf{x}) = D + \frac{\partial D^T}{\partial \mathbf{x}}\mathbf{x} + \frac{1}{2}\mathbf{x}^T\frac{\partial^2 D}{\partial \mathbf{x}^2}\mathbf{x}. \tag{3.7}$$

donde D y sus derivadas son evaluadas en el punto $\mathbf{x} = [x, y, \sigma]^T$ donde $\mathbf{x}$ es el offset del punto. La locación del extremo $\widehat{\mathbf{x}}$, es determinada tomando la derivada de esta función con respecto a $\mathbf{x}$ y se iguala a cero quedando.

$$\widehat{\mathbf{x}} = -\frac{\partial^2 D^{-1}}{\partial \mathbf{x}^2}\frac{\partial D}{\partial \mathbf{x}}. \tag{3.8}$$

Brown sugiere que, el Hessiano y la derivada de D son aproximadas mediante el uso de las diferencias de puntos de muestreo vecinos. El sistema lineal resultante de 3×3 puede resolverse con un costo mínimo. Si el offset $\widehat{\mathbf{x}}$ es mayor que 0.5 en cualquier dimensión, entonces esto significa que el valor extremo se encuentra más cerca de un punto de muestra diferente. En esta caso, la muestra del punto es cambiada y la interpolación es realizada sobre ese punto. El offset final $\widehat{\mathbf{x}}$ es añadido en la ubicación de su punto de muestra para obtener la estimación interpolada para la ubicación del extremo. El valor de la función del extremo, esta dada por $D(\widehat{\mathbf{x}})$ es muy útil para descartar puntos de bajo contraste. Esto puede ser obtenido sustituyendo la Ecuación (3.8) en la Ecuación (3.7)

$$D(\widehat{\mathbf{x}}) = D + \frac{1}{2}\frac{\partial D^T}{\partial \mathbf{x}}\widehat{\mathbf{x}}. \tag{3.9}$$

Para ello, se establece un umbral mínimo al que deben llegar los puntos clave para no ser rechazados. Aquellos que no cumplan la condición $|\widehat{\mathbf{x}}| > 0.03$ serán eliminados de la lista (asumiendo que los píxeles de la imagen están en el rango de $[0, 1]$). Tomando estos datos como referencia, la imagen que se muestra en la Figura 3.4 demuestra la supresión de puntos que son sensibles al ruido. Asimismo el paso siguiente seria la supresión de puntos mal situados a lo largo de los bordes, lo cual se mencionaría en el siguiente apartado.

3.1.3. Supresión de puntos situados a lo largo de bordes

Para la estabilidad, no es suficiente rechazar puntos clave con bajo contraste, la diferencia de Gaussiana tendrá una respuesta fuerte a lo largo de los bordes, incluso si la locación a lo largo del borde está mal determinado y por lo tanto inestable a pequeñas cantidades de ruido. Un pico mal definido en la diferencia de Gaussiana

Figura 3.4: De izquierda a derecha aplicación de un umbral mínimo, $|\widehat{\mathbf{x}}| > 0.03$, para la supresión de puntos de bajo contraste, nótese como van disminuyendo el numero de puntos en cada una de las escalas.

tendrá una gran curvatura principal a través del borde, pero una pequeña en la dirección perpendicular. La curvatura principal puede ser calculada por una matriz Hessiana de 2×2. $\mathbf{H}$, calcula la ubicación y la escala del punto clave.

$$\mathbf{H} = \begin{bmatrix} D_{xx} & D_{xy} \\ D_{xy} & D_{yy} \end{bmatrix}. \tag{3.10}$$

Los vectores propios de H (α y β) son proporcionales a la curvatura principal de D. Siguiendo el método utilizado por Harris, y Stephens en [Harris y Stephens, 1988b], pudiendo evitar explícitamente el cálculo de los valores propios (eigenvalues), ya que solo interesa el radio. Dado α como el valor propio con la magnitud más grande y β como la menor, podremos calcular la suma de los valores propios de la traza de $\mathbf{H}$ y el producto de su determinante.

$$Tr(\mathbf{H}) = D_{xx} + D_{yy} = \alpha + \beta. \tag{3.11}$$

$$Det(\mathbf{H}) = D_{xx}D_{yy} - (D_{xy})^2 = \alpha\beta. \tag{3.12}$$

En el caso improbable de que el determinante sea negativo, es decir que la curvatura tenga diferente signo, entonces el punto es descartado, por no ser un extremo. Siendo r la relación entre ellas ($\alpha = r\beta$), y utilizando la traza y determinante de la matriz se define un nuevo criterio.

$$\frac{Tr(\mathbf{H})}{Det(\mathbf{H})} = \frac{(\alpha+\beta)^2}{\alpha\beta} = \frac{(r\beta+\beta)^2}{r\beta^2} = \frac{(r+1)^2}{r}. \tag{3.13}$$

Criterio que depende sólo de la relación de los valores propios en lugar de sus valores individuales. El valor $\frac{(r+1)^2}{r}$ está en un mínimo cuando los dos valores propios son iguales y aumenta con r. Por lo tanto, para comprobar que la relación de curvaturas principales es inferior a un umbral, r, estableciendo un valor de $r = 10$, ello implicará la eliminación de los puntos clave que tengan una relación entre respuesta paralela y perpendicular al borde superior a 10.

$$\frac{Tr(\mathbf{H})^2}{Det(\mathbf{H})} < \frac{(r+1)^2}{r}. \tag{3.14}$$

Como se deduce en la Figura 3.4 y 3.5 , son rechazados un alto porcentaje de puntos que provienen de la fase inicial de SIFT. Así, sólo quedarán los que son realmente invariantes al cambio. Desde el punto de vista computacional es una ventaja, ya que se calcularan muchos menos descriptores en etapas posteriores.

3.1.4. Asignación de la Orientación de los puntos clave

Se define una región de 16 × 16 píxeles alrededor del punto, donde se calculan la orientación del punto clave seleccionado, y donde a cada uno de los píxeles se les calculara su gradiente. Ésta viene determinada por su modulo e inclinación, ambos parámetros se calcularan utilizando diferencias entre píxeles.

Figura 3.5: De izquierda a derecha puntos que no fueron descartados, y sólo quedan los que son realmente invariantes al cambio.

$$m(x,y) = \sqrt{(L(x+1,y) - L(x-1,y))^2 + (L(x+1,y) - L(x-1,y))^2} \tag{3.15}$$

$$\theta(x,y) = tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)}{L(x+1,y) - L(x-1,y)}\right) \tag{3.16}$$

La imagen utilizada para hacer los cálculos anteriores será la imagen de la pirámide L, Figura 3.5, donde se detectó el punto de interés que esté siendo analizado.

Después de realizar el proceso de la Figura 3.6, se agrupa la información en forma de histograma, para cada punto clave. De tal forma que cada histograma de orientaciones estará formado por 36 contenedores (bins) para cubrir el rango total de 360°. A medida que se añade al histograma, cada orientación $\theta(x_1, y_1)$ de la región, 16×16, se pondera por su módulo $m(x_1, y_1)$ y por una ventana circular Gaussiana con valor σ igual a

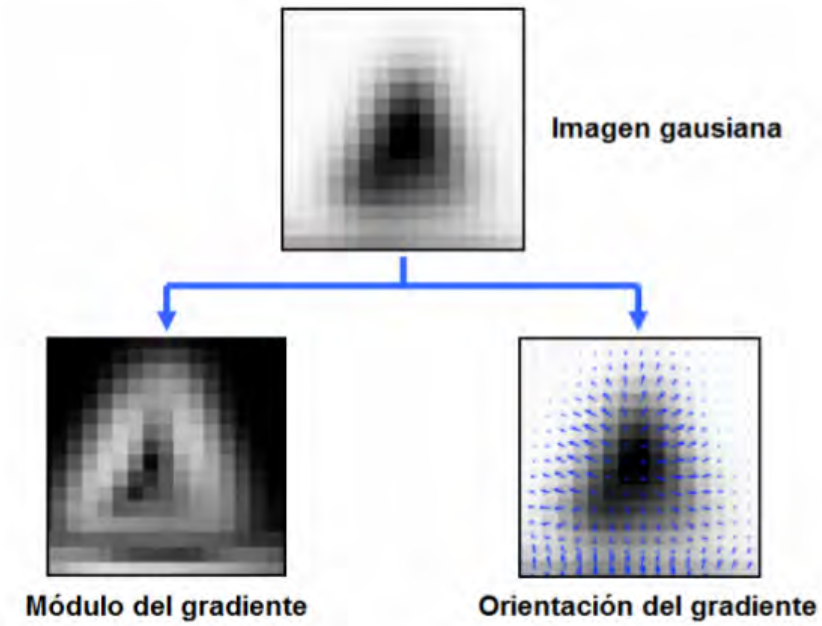


Figura 3.6: Arriba: Ventana 16 $\times$16 alrededor del punto de interés. Abajo izquierda: $m(x, y)$. Abajo derecha: $\theta(x, y)$.

1.5 veces la escala del punto clave como lo muestra la Figura 3.7. Existen diversos motivos para realizar estas dos ponderaciones.

- Dar mayor peso a las orientaciones con módulos elevados, que por tanto son más importantes.

- Dar mayor importancia a los puntos cercanos al punto clave, es decir, los puntos centrales de la ventana.

Los picos más altos de cada histograma son las direcciones dominantes de los gradientes locales, y por lo tanto la orientación final del punto clave. Sin embargo, en algunas ocasiones no es necesario quedarse sólo con el pico más alto. El proceso será el

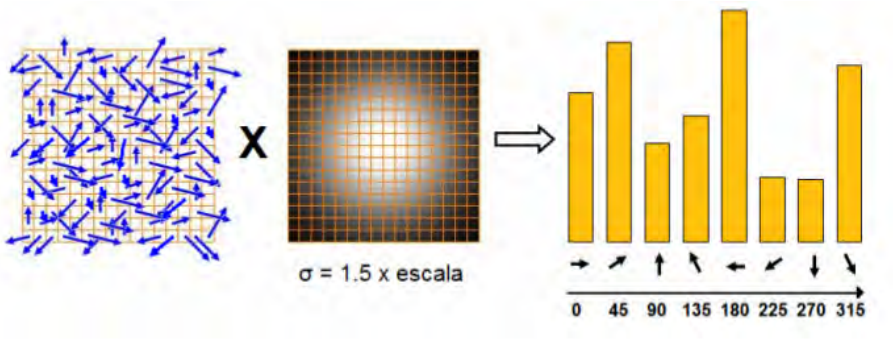


Figura 3.7: Izquierda: Región de gradientes 16 × 16. Centro: Ventana circular Gaussiana. Derecha: Histograma final del punto de interés.

siguiente:

- Detección del pico de mayor tamaño

- Búsqueda de picos secundarios que tengan una altura mayor al 80 % del principal. Si no hay ninguno, se toma sólo el mayor.

- A cada pico seleccionado, se interpola su posición para una mayor precisión. Se lleva a cabo mediante la construcción de una parábola entre él mismo y sus vecinos laterales.

$$P_n(x) = P_{n-1}(x) + a_n(x - x_0) \cdot .. \cdot (x - x_{n-1}) \tag{3.17}$$

Para localizaciones con múltiples picos elevados de similar magnitud, se obtienen varias orientaciones para un mismo punto de la imagen. Es decir, al construir los descriptores en la siguiente etapa del algoritmo, los puntos claves con orientaciones múltiples tendrán asignados varios descriptores, que sólo diferirán en su inclinación.

Hay que precisar que la mayoría serán orientaciones simples, y sólo a un porcentaje de entre el 10 % y 20 % de los puntos se les construirán múltiples descriptores. Por tanto, este hecho no producirá un aumento significativo del tiempo de procesado total.

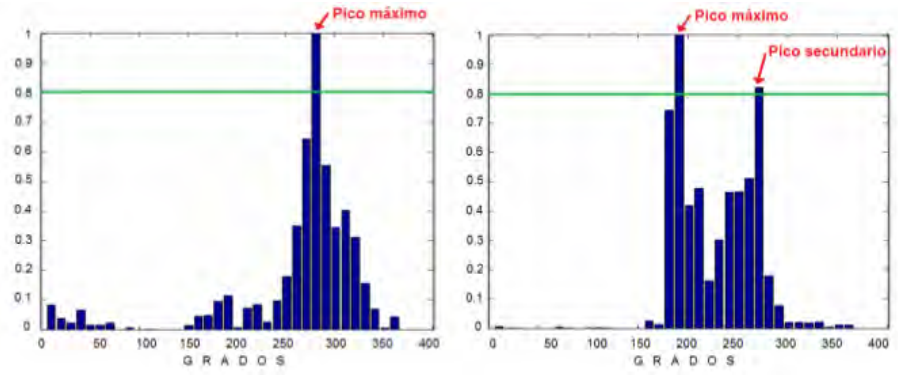


Figura 3.8: Izquierda: Ejemplo de histograma con orientación simple. Derecha: Ejemplo de histograma con orientaciones múltiples

3.1.5. Descriptor de Puntos de interés

Los parámetros calculados hasta ahora, forman un sistema de coordenadas 2D que describe localmente cada región de la imagen, y por tanto proporciona invariancia a esos mismos parámetros. El siguiente paso es, utilizando toda esa información, obtener un descriptor para cada zona de interés. Éste nos aportará además robustez ante posibles variaciones de iluminación y cambios de puntos de vista 3D. El proceso parte de las regiones 16×16 ya multiplicadas por la ventana Gaussiana con σ igual a 1.5 veces la escala. Éstas se dividen en subregiones de 4×4 píxeles con el objetivo de resumir toda esa información en pequeños histogramas de sólo 8 contenedores (bins), es decir, 8 orientaciones.

Antes de realizar esta adaptación de la información, cada gradiente de la ventana de 16×16 se rota tantos grados como especifique la orientación principal del punto clave (calculada en la etapa anterior), y así será independiente a la inclinación de la imagen.

Ahora bien para cada punto clave, mientras antes se tenia un gran histograma con 36 posibles orientaciones (que provenía de 256 muestras alrededor del punto), ahora se tendrán 16 pequeños histogramas de 8 contenedores (bins) cada uno de ellos. Para evitar cambios abruptos entre fronteras, cada subregión es filtrada de nuevo por una ventana circular Gaussiana (en esta ocasión de tamaño 4x4) con un factor $\sigma = 0.5 \times$ escala como se ve en la Figura 3.9.

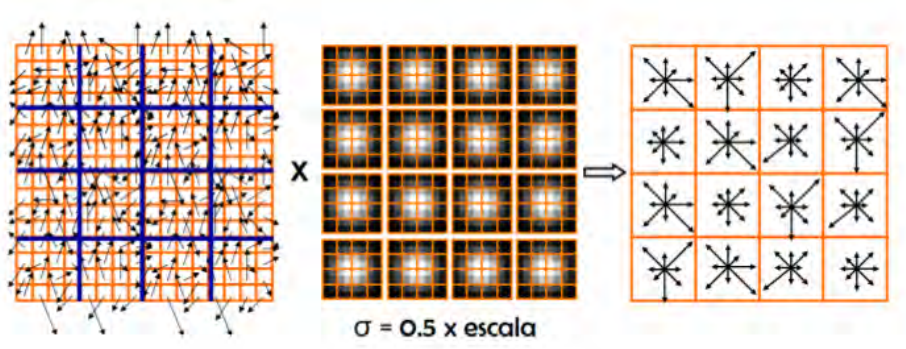


Figura 3.9: Izquierda: Subdivisiones 4×4. Centro: Ventanas circulares gaussianas. Derecha: Descriptor compuesto por 16 histogramas de 8 bins.

Existen dos parámetros que marcan la complejidad del descriptor: el número de orientaciones de cada histograma (r) y el ancho de la matriz (n). En esté caso, el tamaño será de 128 elementos

$$\text{Tamaño} = r \times n^2 = (8).(4)^2 = 128$$

Es necesario realizar una serie de modificaciones para conseguir mayor robustez ante posibles cambios de iluminación. El objetivo es ser invariante a dos tipos de variación.

- Luminosidad
- Contraste

Hasta el momento, el descriptor es invariante a la primera de las dos características, la luminosidad. Esto es así debido a que los gradientes están calculados mediante

diferencias entre píxeles vecinos. Por tanto, el hecho de sumar una constante de luz a la imagen no influirá en el resultado final. Para conseguir ser robusto a cambios de contraste, sólo hay que normalizar a la unidad cada uno de los sub-histogramas del total de 16 que tiene cada descriptor. Así, un cambio de contraste en el que cada píxel y gradiente sean multiplicados por una constante, será automáticamente cancelado por la normalización.

3.2. SURF

El descriptor SURF, cuyo acrónimo hace referencia al título, Speed-Up Robust Feature, fue desarrollado por Herbert Bay [Bay et al., 2008] como un detector de puntos de interés y descriptor robusto. El descriptor SURF guarda cierto parecido con la filosofía del descriptor SIFT [Lowe, 2004] si bien presenta cambios notables, que quedaran expuestos en los siguientes apartados, los autores afirman que este detector y descriptor presenta 2 mejoras resumidas en los siguientes conceptos:

- Velocidad de cálculo considerablemente superior sin ocasionar perdida del rendimiento.
- Mayor robustez ante posibles transformaciones de la imagen.

Estas mejoras se consiguen mediante la reducción de la dimensionalidad y complejidad en el cálculo de los vectores de características de los puntos de interés obtenidos, mientras continúan siendo suficientemente característicos e igualmente repetitivos. A continuación se describirán en detalle las etapas para la creación de los descriptores de SURF. Pero antes se presentan, a modo de resumen previo, las diferencias más significativas respecto del descriptor SIFT:

- La normalización o longitud de los vectores de características de los puntos de interés es considerablemente menor. Concretamente se trata de vectores con una

dimencionalidad de 64, lo que supone una reducción de la mitad de la longitud del descriptor SIFT.

- El descriptor SURF siempre la misma imagen, la original.
- Utiliza el determinante de la matriz Hessiana para calcular tanto la posición como la escala de los puntos de interés.

El algoritmo de SURF esta compuesto por tres pasos consecutivos:

- Detección de puntos de interés.
- Descriptores de puntos de interés.
- Correspondencias de características.

Como el método de SIFT, los primeros dos pasos se basan en una representación espacio-escala. La originalidad del método SURF es que las operaciones son aceleradas por el uso de una imagen integral y técnicas de filtro de caja que se detallan en la secciones de la aproximación al espacio-escala y selección de puntos de interés, respectivamente.

3.2.1. Detección de puntos de interés

La primera de las etapas del descriptor SURF es análoga a la del descriptor SIFT en cuanto a la detección de puntos de interés se refiere. Sin embargo el procedimiento para su obtención se basa en diferencias sustanciales que se detallan a lo largo de esta Sección.

El descriptor SURF hace uso de la matriz Hessiana, más concretamente, del valor del determinante de la matriz, para la localización y la escala de los puntos. El motivo para la utilización de la matriz Hessiana es respaldado por su rendimiento en cuanto a la velocidad de cálculo y a la precisión. Lo realmente novedoso del detector incluido en

el descriptor SURF respecto de otros detectores es que no utiliza diferentes medidas para el cálculo de la posición y la escala de los puntos de interés individualmente, sino que utiliza el valor del determinante de la matriz Hessiana en ambos casos. Por lo tanto dado un punto $p = [x, y]^T$ de la imagen I, la matriz Hessiana $H(p, \sigma)$ del punto p perteneciente a la escala σ se define como:

$$H(p, \sigma) = \begin{bmatrix} D_{xx}(p, \sigma) & D_{xy}(p, \sigma) \\ D_{xy}(p.\sigma) & D_{yy}(p, \sigma) \end{bmatrix}. \tag{3.18}$$

donde $D_{xx}(p, \sigma)$ representa la convolución de la derivada de segundo orden de la Gaussiana $\frac{\partial^2}{\partial x^2} g(\sigma)$ con la imagen I en el punto p. De manera análoga ocurre con los términos $D_{xy}(p, \sigma), D_{yy}(p, \sigma)$ de la matriz. A pesar de que los filtros gaussianos son óptimos para el análisis del espacio-escala [Lindeberg, 1990], se ha implementado una alternativa a los filtros gaussianos en el detector SURF debido a una serie de limitaciones de estos filtros como la necesidad de ser discretizados, la falta de prevención total del indeseado efecto aliasing. Por ende, se utilizan los filtros tipo caja (box-filters) [Simard et al., 1998]. Estos filtros aproximan las derivadas parciales de segundo orden de las Gaussianas y pueden ser evaluados de manera muy rápida usando imágenes integrales, cuya definición se encuentra ampliamente detallada en [Derpanis et al., 2007], [Viola y Jones, 2004]. Sin embargo, se mencionara en el siguiente apartado la aproximación al espacio-escala con filtros de caja e imágenes integrales.

3.2.2. Aproximación al Espacio-Escala vía Filtros de Caja

Definir un método que es invariante a cambios de escala, generalmente se logra mediante la simulación de ampliaciones de la imagen. Esta representación del espacio-escala puede ser obtenida por una convolución Gaussiana en las diferentes escalas de la imagen (como SIFT [Lowe, 2004]). Para acelerar este procedimiento, cuyo consu-

mo es del orden de $(O(logM(M)))$ donde M es el tamaño del kernel Gaussiano, el algoritmo de SURF se aproxima al kernel Gaussiano y sus derivadas espaciales por funciones rectangulares denominados filtro de caja o ("box-filer"), Para dichas funciones la complejidad de convolución es lineal $(O(WH)) = O(N)$ utilizando imágenes integrales, en SURF el espacio-caja es definido por la siguiente tripleta (x, y, L) donde L caracteriza el tamaño de los filtros de caja usados.

En esta subsección, primeramente se recalca la definición de imagen integral y la convolución con un filtro de caja. Así mismo se hace una comparación entre el espacio-escala normal y la integración del filtro de caja que utiliza el algoritmo de SURF.

Imágenes Integrales y Filtros de Caja

Considerando I como una imagen en tono de grises (tomando valores en el rango de 0 a 255), lo cual sería una forma fácil de aplicar robustez al cambio de color (tales como una corrección de balance de blancos). La imagen integral se define de la siguiente manera.

$$\widehat{I}(x, y) = \sum_{i=0}^{i \leq x} \sum_{j=1}^{j \leq y} I(i, j). \tag{3.19}$$

En la Figura 3.10 se ilustra el valor de la imagen integral en el punto $[x, y]^T$, donde se muestra la suma de todas las intensidades en el cuadro dorado. Una imagen integral puede ser creada de manera recursiva para minimizar el numero de cálculos necesarios para obtenerla, comienza en la esquina superior izquierda y trabaja por una fila a la vez.

La suma de las intensidades en en cuadro dorado es la siguiente donde u y v son coordenadas del otro punto que forma el rectángulo:

$$c * I(x, y) = I_c(x, y) - I_c(x, v) - I_c(u, y) + I_c(u, v). \tag{3.20}$$

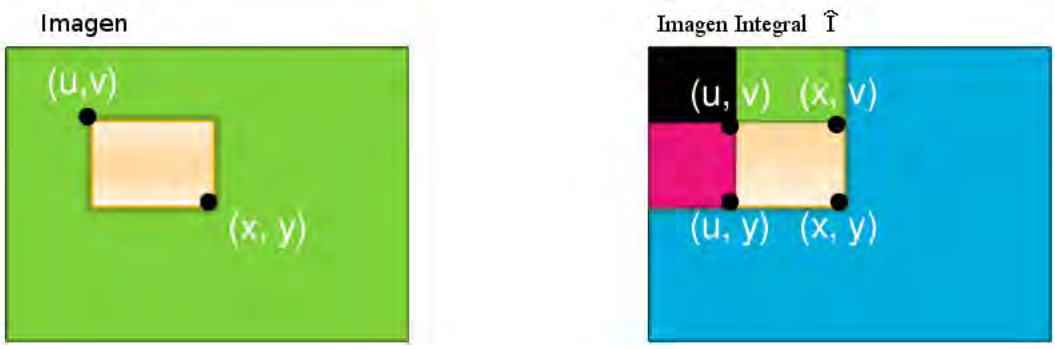


Figura 3.10: Ejemplo de Imagen Integral.

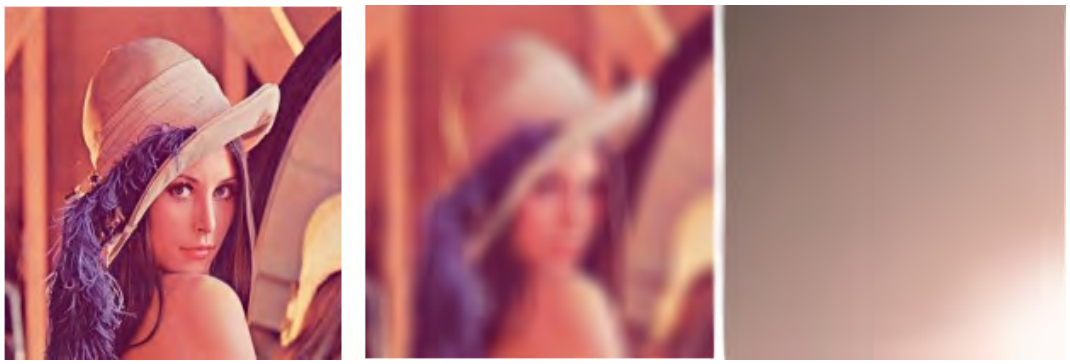


Figura 3.11: Ejemplo de Imagen Integral, en la imagen de Lena aplicando filtro de caja de 15×15 .

Ahora la convolución de I con una función rectangular 2D B_Γ con soporte rectangular $\Gamma = [\text{-W}, \text{W}] \times [-\text{H}, \text{H}]$, donde W y H son enteros.

$$B_\Gamma(x, y) = \begin{cases} 1 & \text{si}(x, y) \in \Gamma \\ 0 & \text{de otra manera} \end{cases} \tag{3.21}$$

y Γ es un rectángulo de dimensiones $\text{W} = \frac{R-1}{2}$ donde R es impar, $\text{H} = \frac{R'-1}{2}$, $R' \in 2\mathbb{Z}+1$. El calculo de tal imagen integral $\widehat{I}$ permite convolucionar la imagen I con una caja B_Γ en tres operaciones y cuatro accesos a memoria usando la siguiente formula.

$$(B_\Gamma * I)(x,y) = \widehat{I}(x-\mathrm{W}, y-\mathrm{H}) + \widehat{I}(x+\mathrm{W}, y+\mathrm{H}) - \widehat{I}(x-\mathrm{W}, y+\mathrm{H}) - \widehat{I}(x+\mathrm{W}, y-\mathrm{H}). \tag{3.22}$$

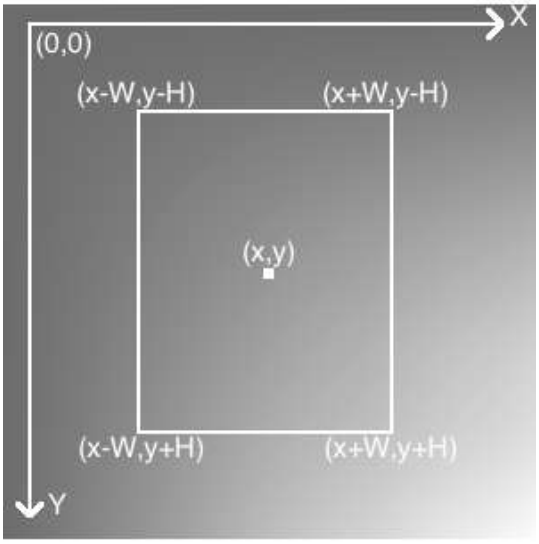


Figura 3.12: Ejemplo de Imagen Integral.

Ahora bien, el espacio-escala Gaussiano es representado por una imagen I, el cual se obtiene al convolucionar un kernel Gaussiano.

$$I_\sigma = G_\sigma * I. \tag{3.23}$$

donde g_σ es el centro del kernel 2-D donde el parámetro σ usualmente es el parámetro de escala.

$$G_\sigma(x,y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{3.24}$$

Haciendo uso de la técnica de filtrado de caja antes mencionado, tal representación se puede aproximar con la siguiente convolución discreta $\widehat{I}_L = B_L * I$ donde B_L es un filtro con un ancho de banda L es decir, con un soporte cuadrado $\left[-\frac{L}{2}, \frac{L}{2}\right] \times \left[-\frac{L}{2}, \frac{L}{2}\right]$

en un entorno discreto. Los valores tomados por L son enteros e impares. Los valores correspondientes de σ son dados en [Gwosdek et al., 2011] :

$$\sigma^2(B_L) = \frac{L^2 - 1}{12}. \tag{3.25}$$

Es importante mencionar que el filtro de caja debe ser normalizado por un factor L^2 para obtener un núcleo unitario, lo cual no se considera dentro del tiempo de cálculo, puesto que la normalización se lleva a cabo sólo una vez durante la etapa de detección de puntos característicos. En la siguiente Figura 3.14 se ilustra la aproximación del espacio-escala obtenido usando filtros de caja. De igual forma se hace una comparación con el espacio-escala Gaussiano.

Figura 3.13: Imagen Original Samurái

Como se puede observar en la Figura 3.14 son más o menos similares. Sin embargo, la aproximación espacio-escala por medio de filtros de caja exhibe algunos objetos verticales y horizontales debido a la anisotropía del filtro de caja.

Nótese que la mejor aproximación al espacio-escala se puede obtener iterando filtros de caja. Véase por ejemplo la siguiente Figura 3.15 donde se incrementa el tamaño del filtro en base a lo que se menciona en [Gwosdek et al., 2011].

Figura 3.14: Izquierda: Imagen con filtros de caja y L=13, Derecha Imagen con $\sigma = 1.708$

Figura 3.15: De izquierda a derecha Imagen con filtros de caja cambiando el tamaño L, a su vez filtro gaussiano en un escala mayor.

3.2.3. Muestro del Espacio-Escala

Similar al algoritmo de SIFT, el espacio-escala en el enfoque de SURF esta basado en un muestreo regular de parámetros de escala L. La representación de escala es dividida en octavas: una nueva octava corresponde con el doble del tamaño del kernel (lo que significa que las escalas son una serie geométrica). Cada octava es dividida en diferentes niveles (o intervalos), cada una corresponde al incremento del tamaño del filtro implicado. En SURF, se define s como el índice de octava e i el índice del intervalo, el muestreo de escala se obtiene con la siguiente relación:

$$\sigma = \frac{1.2}{3}(2^s \times i + 1) = \frac{1.2}{3} \times L = 0.4L.. \tag{3.26}$$

donde $0.4L$ es la escala que se utiliza para todos los cálculos de puntos clave y se corresponde con el parámetro de detección de escala, el análogo del parámetro de la longitud de la caja. Una formula más precisa debería haber sido utilizada, puesto que

$$\sigma(B_L) \approx \frac{L}{\sqrt{12}} \approx 0.29L. \tag{3.27}$$

Contrario al algoritmo de SIFT [Lowe, 2004] la imagen no se submuestrea, por un factor de 2 en cada octava. Debido a la utilización de filtros de caja e imágenes integrales no se tiene que aplicar iterativamente el mismo filtro a la salida de una capa previamente filtrada, en su lugar se aplican filtros de caja de tamaños previamente expuestos exactamente a la misma velocidad, directamente en la imagen original e incluso en paralelo.

Por lo tanto, el espacio-escala se analiza incrementando el tamaño del filtro en lugar de reducir iterativamente el tamaño de la imagen como se muestra en la Figura 3.16.

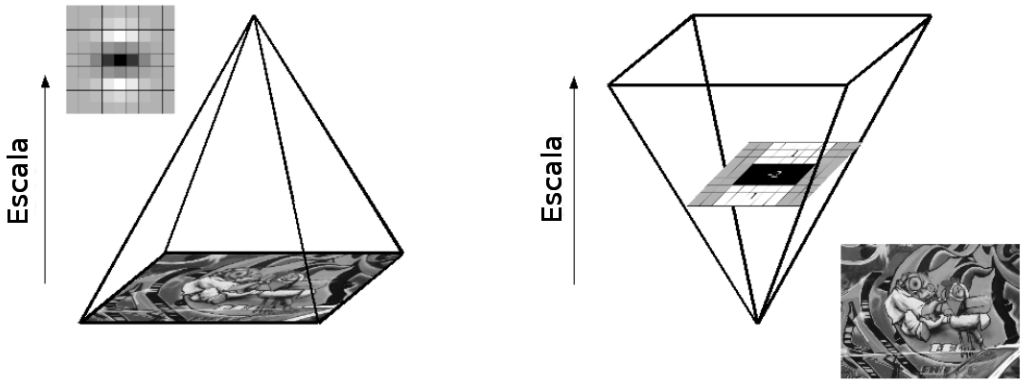


Figura 3.16: En lugar de reducir iterativamente el tamaño de la imagen (a la izquierda), el uso de imágenes integrales permite el aumento de escala del filtro con un costo constante

La salida del filtro de 9×9 que se introducirá en los párrafos subsecuentes son

considerados como la capa de escala inicial, a la cual se referirá como escala $L = 1.2$ (la cual es una aproximación de la derivada de Gaussiana con $\sigma = 1.2$). Las capas siguientes se obtienen filtrando la imagen con máscaras gradualmente más grandes, teniendo en cuenta la naturaleza de la imagen integral y la estructura especifica de los filtros.

El espacio escala se divide en octavas. Una octava representa una serie de respuestas del filtro obtenidos mediante la convolución de la misma imagen de entrada con un filtro de tamaño mayor. En total, una octava incluye un factor de escala de 2 (que implica que uno tiene a más del doble del tamaño del filtro, ver más abajo). Cada octava se subdivide en un número constante de los niveles de escala, debido a la naturaleza discreta de las imágenes integrales. La diferencia mínima en la escala entre 2 escalas posteriores depende de la longitud L de las áreas positivas o negativas de la segunda derivada de orden parcial en la dirección de la derivada (x o y), que se fija en un tercio de la longitud del tamaño del filtro. Para el filtro de 9×9, esta longitud L es de 3. Para dos niveles sucesivos, se debe aumentar este tamaño por un mínimo de 2 píxeles (un pixel a cada lado) con el fin de mantener el tamaño impar y por lo tanto garantizar la presencia del píxel central. Esto se traduce en un aumento del tamaño total de la máscara de píxeles por 6. Sin embargo, se debe de tener en cuenta que, para dimensiones diferentes a partir de L (por ejemplo, la anchura de la banda central para el filtro vertical en la Figura 3.17), se ajusta la base de la máscara introduciendo un redondeo del posible error. Sin embargo, ya que estos errores son típicamente muchos más pequeños que L, ésta es una aproximación aceptable.

La construcción del espacio escala comienza con el filtro de 9×9, que calcula la respuesta gota para las imágenes de escalas pequeñas. Es decir el punto de interes es representado por una burbuja, gota la cual el diamentro de esta representa la escala a la que pertenece ese punto de interes.

Entonces filtros con un tamaño de 15×15, 21×21 y 27×27 se aplican, por lo que incluso más de un cambio de escala de 2 se puede lograr. Pero esto es necesario, ya que

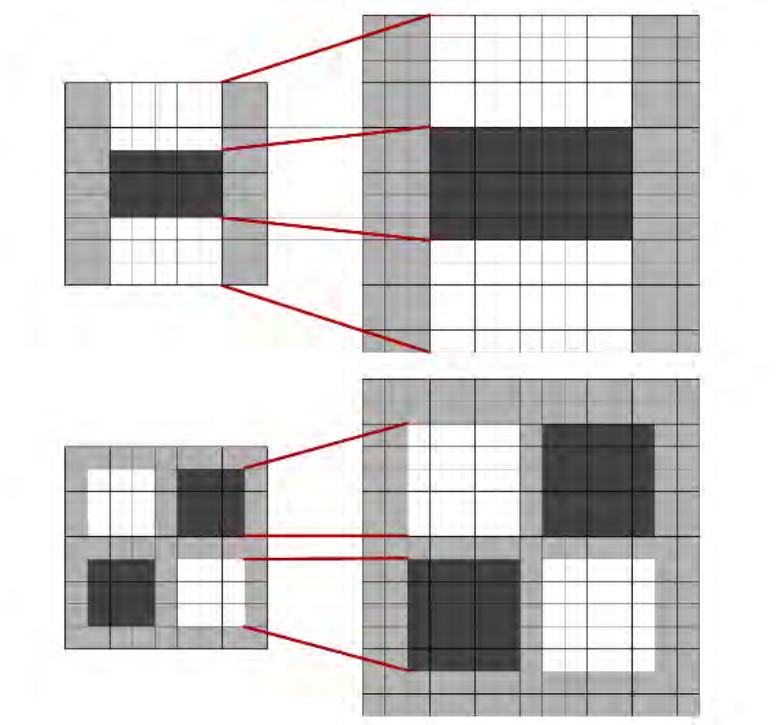


Figura 3.17: Filtro D_{yy} (parte superior) y D_{xy} (parte inferior) para dos escalas sucesivas (9×9 15×15). La longitud del lóbulo oscuro sólo puede ser aumentada por un número par de píxeles con el fin de garantizar la presencia de un píxel central (parte superior).

una supresión no máxima 3D es aplicada tanto espacial como sobre las escalas vecinas. Por lo tanto la primera y la ultima respuesta del Hessiano no puede contener dichos máximos en sí misma, ya que se utilizan por razones de comparación solamente. Por lo tanto, después de la interpolación, la escala más pequeña posible es $\sigma = 1.6 = 1.2\frac{12}{9}$ correspondiente a un filtro de 12×12 y la más alta para $\sigma = 3.2 = 1.2\frac{24}{9}$, para más detalle ver [Bay, 2009].

Consideraciones similares son válidas para las otras octavas. Para cada nueva octava, el aumento de tamaño del filtro se duplica (va del 6 al 12, 24 al 48). Al mismo tiempo, los intervalos de muestreo para la extracción de los puntos de interés se puede doblar, así como para cada nueva octava. Esto reduce el tiempo de cálculo y la pérdida de precisión es comparable a la imagen sub-muestreada de las aproximaciones tradicio-

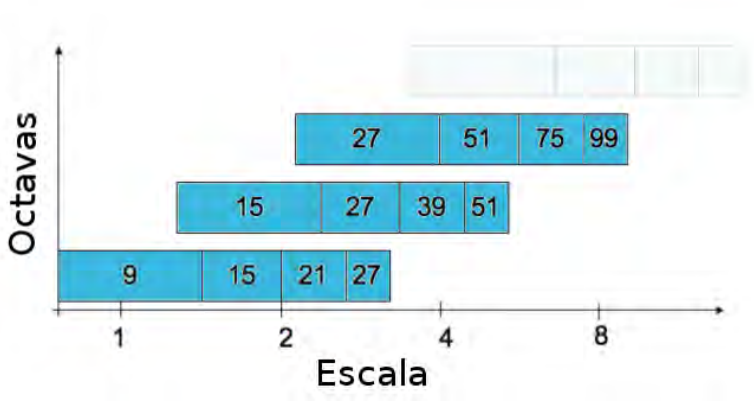


Figura 3.18: Representación gráfica de las longitudes de los lados de filtro para tres octavas diferentes. El eje horizontal representa la escala logarítmica. Tenga en cuenta que las octavas se superponen con el fin de cubrir todas las escalas posibles sin problemas [Bay et al., 2008].

nales. Los tamaños del filtro para la segunda octava son 15, 27, 39, 51. Un tercio de la octava se calcula con los tamaños del filtro 27, 51, 75, 99 y, si el tamaño de la imagen original es todavía más grande que los tamaños de los filtros correspondientes, el análisis del espacio escala se lleva a cabo por una cuarta octava de filtro 51, 99, 147 y 195. En la Figura 3.18 se da una visión general de los tamaños del filtro para las tres primeras octavas, Se debe de tener en cuenta que más octavas pueden ser analizadas, pero el número de puntos de interés detectado por cada octava decae muy rápidamente. Los cambios a gran escala, especialmente entre los primeros filtros dentro de estas octavas (del 9 al 15 es un cambio de 1.7) hace que el muestreo de escalas sea en crudo. Por lo tanto, también se implementa un espacio escala con un muestra más fino de las escalas. Ésta primero duplica el tamaño de la imagen, mediante la interpolación lineal, y a continuación, comienza la primera octava mediante el filtro de tamaño 15 $\times$ 15. Tamaños de filtros adicionales son 21, 27, 33 y 39. Entonces comienza una segunda octava. De nuevo el uso de filtros que ahora aumentan el tamaño por 12 píxeles después de una tercera y cuarta octava. Ahora bien, el cambio de escala entre los dos primeros filtros es solo 1.4($\frac{21}{5}$). La escala más baja para

la versión exacta que puede ser detectada a través de la interpolación cuadrática es $s = (1.2\frac{18}{9})/2 = 1.2$. A medida que la norma de Frobenius permanece constante para filtros de cualquier tamaño, donde la escala ya se ha normalizado, y no se requiere ningún otra ponderación de la respuesta del filtro, ver [Lindeberg y Bretzner, 2003] .

3.2.4. Aproximación de los operadores diferenciales

Con el fin de detectar puntos de interés y construir descriptores robustos, los operadores diferenciales de primer y segundo orden son calculado en el espacio-escala. obtenido por la convolución de un kernel Gaussiano (como el que se menciona en SIFT [Lowe, 2004]) esto se reduce a calcular la operación de convolucion siguiente:

$$D_{xy}^{(\sigma)} I_\sigma(x, y, \sigma) = (D_{xyg_\sigma}) * I(x, y). \tag{3.28}$$

Sin embargo en el enfoque de SURF, este calculo se aproxima con una convolución con una combinación lineal de filtros de caja, la cual se definirá en el siguiente párrafo.

$$\tilde{D}_{xy}^{(L)} * I. \tag{3.29}$$

La derivada de primer orden en la escala L se calcula utilizando Harr wavelets $\tilde{D}_x^{(L)}$ y $\tilde{D}_y^{(L)}$:

$$\tilde{D}_y^{(L)}(x, y) = \begin{cases} +1 & \text{Si}(x, y) \in [-L, L] \times [-L, -1] \\ -1 & \text{si } (x, y) \in [-L, L] \times [-1, L] \\ 0 & \text{en otro caso} \end{cases} \tag{3.30}$$

Esto por consiguiente se reduce a.

$$(\tilde{D}_x^{(L)} * I)(x, y) = \sum_{i=-1}^{L} \sum_{j=1}^{L} (I(x + i, y + j) - I(x + i, y - j))\text{..} \tag{3.31}$$

Lo cual es calculable con 7 adiciones con filtros de caja ver Ecuación (3.18).

$$\tilde{D}_x^{(L)} * I = B_{\Gamma 1} * I - B_{\Gamma 2} * I. \tag{3.32}$$

donde,

$$\Gamma_1 = [-L, L] \times [-L, 0] \text{ y } \Gamma_2 = [-L, L] \times [0, L]. \tag{3.33}$$

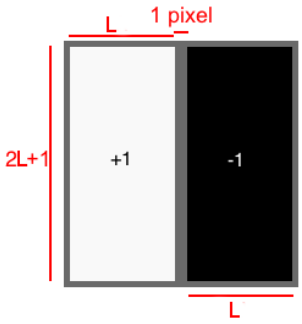


Figura 3.19: Aproximación de los operadores de derivada de orden 1 a lo largo de x

La Figura 3.20 hace una comparación del operador de derivada de Gauassiana original y su correspondiente aproximación utilizando filtros de caja. Asimismo en SURF, la derivada de segundo orden en la escala L se aproxima por:

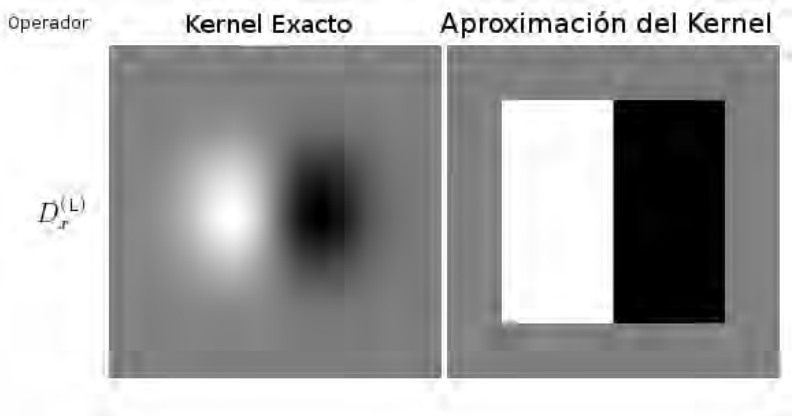


Figura 3.20: Aproximación de los operadores de derivada de orden 1 a lo largo de x

$$\tilde{D}_{yy}^{(L)}(x,y) = \begin{cases} -2 & \text{Si}(x,y) \in [-L,L] \times \left[-\frac{L}{2}, \frac{L}{2}\right] \\ +1 & \text{si } (x,y) \in [-L,L] \times \left[-\frac{3L}{2}, \frac{3L}{2}\right] \setminus [-L,L] \times \left[-\frac{L}{2}, \frac{L}{2}\right] \\ 0 & \text{en otro caso} \end{cases} \tag{3.34}$$

$$\tilde{D}_{xy}^{(L)}(x,y) = \begin{cases} +1 & \text{Si}(x,y) \in [0,-L] \times [0,+L] \cap [0,+L] \times [-L,0] \\ -1 & \text{Si } (x,y) \in [-L,0] \times [-L,0] \cap [0,+L] \times [0,+L] \\ 0 & \text{en otro caso} \end{cases} \tag{3.35}$$

Donde el parámetro L es definido en la Ecuación 3.25, los filtros correspondientes son como se muestran en la Figura 3.21:

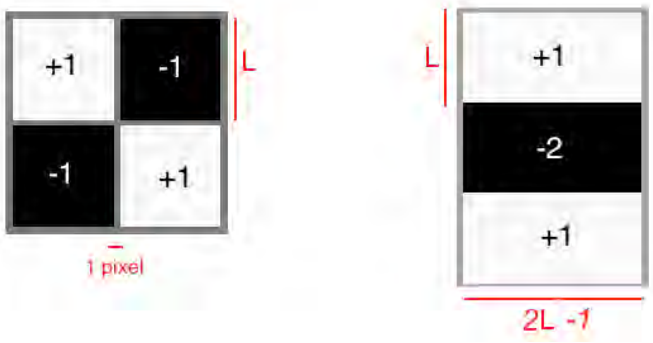


Figura 3.21: Filtros de caja de segundo orden con parámetro L. Izquierda $\tilde{D}_{xy}^{(L)}$ derecha $\tilde{D}_{yy}^{(L)}$ donde los operadores de definen en las Ecuaciones 3.33 y 3.34 respectivamente

Una vez más, la convolución con estos operadores se calcula utilizando la imagen integral. Por ejemplo, se obtiene,

$$\tilde{D}_{yy}^{(L)} * I = B_{\Gamma_1} * I - 3 \times B_{\Gamma_2} * I. \tag{3.36}$$

donde,

$$\Gamma_1 = [-L, L] \times \left[-\frac{3L}{2}, \frac{3L}{2}\right], \Gamma_2 = [-L, L] \times \left[\frac{L}{2}, \frac{L}{2}\right]. \tag{3.37}$$

y,

$$\tilde{D}_{xy}^{(L)} * I = -B_{\Gamma 1} * I + B_{\Gamma 2} * I + B_{\Gamma 3} * I - B_{\Gamma 4} * I. \tag{3.38}$$

Así,

$$\Gamma_1 = [0, L] \times [0, L], \Gamma_2 = [-L, 0] \times [0, L], \Gamma_3 = [0, L] \times [-L, 0], \Gamma_4 = [-L, 0] \times [-L, 0]. \tag{3.39}$$

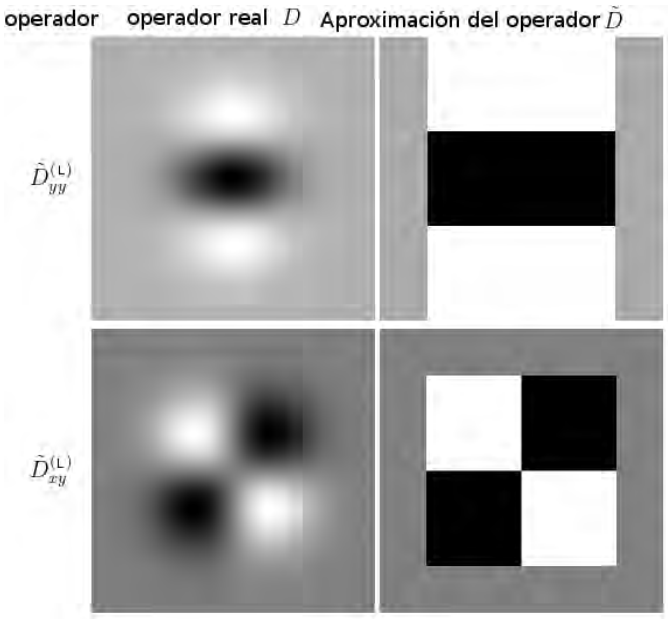


Figura 3.22: Arriba a la derecha filtro de SURF $\tilde{D}_{yy}^{L}(x,y)$; Arriba a la izquierda el filtro correspondiente de SIFT $\tilde{D}_{yy}^{L}(x,y)$; Abajo a la derecha filtro de SURF en $\tilde{D}_{xy}^{L}(x,y)$; abajo a la izquierda el filtro correspondiente de SIFT en $\tilde{D}_{xy}^{L}(x,y)$

Una comparación entre las derivadas de segundo orden de la Gaussiana y su aproximación con filtros de caja se muestran a continuación.

Así como lo menciona Lindeberg en [Lindeberg, 1994], las características invariantes a escala pueden ser detectadas mediante el uso de escalas normalizadas y derivadas de segundo orden en el espacio-escala representado en la imagen. Estas características corresponden a gotas, bordes o esquinas. Contrario a SIFT [Lowe, 2004] en donde se aproxima al laplaciano con diferencias de Gaussiana (DoG), SURF aproxima a la normalización de escala mediante el determinante del Hessiano (DH), operador que se define enseguida.

$$DH^{L}(I) = \sigma^{4} det(Hg_{\sigma} * I) = \sigma^{4}((D_{xx}g_{\sigma} * I) \times (D_{yy}g_{\sigma} * I) - (D_{xy}g_{\sigma} * I)^{2}). \quad (3.40)$$

con esta fórmula.

$$D\tilde{H}^L(u) := det(\tilde{H}^{(L)} * I) := (\tilde{D}^L_{xx} * I) \times (\tilde{D}^L_{yy} * I) - (r(L)\tilde{D}^{(L)}_{xy})^2 \tag{3.41}$$

$$= \left\{ \left(\tilde{D}^L_{xx} * I \right) \times \left(D^L_{yy} * I \right) - \left(r(L) \left(\tilde{D}^L_{xy} * I \right)^2 \right) \right\}$$

donde la derivada de segundo orden se calcula en la escala L usando la ecuación anterior. El factor $r(L)$ corresponde con la razón de la norma de Frobenius (raíz cuadrada de la suma de los cuadrados de cada uno de los coeficientes de la matriz) donde la aproximación y el Hessiano exacto del kernel Gaussiano se define como sigue:

$$r(L) = \frac{\| \tilde{D}^{(L)}_{xx} \|_F}{\| \tilde{D}^{(L)}_{xy} \|_F} . \frac{\| \tilde{D}^{(L)}_{xy} \|_F}{\| \tilde{D}^{(L)}_{xx} \|_F} = 0.912... \simeq 0.9 \tag{3.42}$$

De acuerdo con SURF, el peso relativo $r(L)$ de la respuesta del filtro es usado para balancear la expresión del determínate del Hessiano. Esto es necesario para la conservación de la energía entre el kernel Gaussiano y su aproximación. En la Figura 3.23 se muestra la imagen obtenida al aplicar el operador exacto y su aproximación con una delta de Dirac δ. De igual forma en la Figura 3.24 se muestra un ejemplo al aplicar tal filtro.

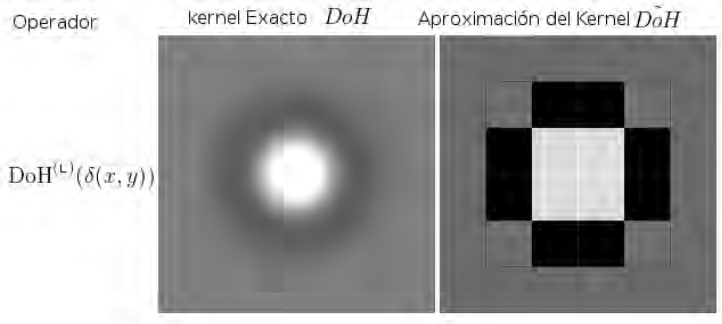


Figura 3.23: Kernel exacto del Hessiano con una delta de Dirac y su aproximación

Figura 3.24: Operador Determinante de Hessiano aplicado a la imagen de Lena con $L = 33$

3.2.5. Selección de Características

Los puntos de interés son puntos en el espacio-escala que corresponden a un máximo local citado en el operador del determinante del Hessiano, aplicado a la representación del espacio-escala en la imagen. Estos puntos son seleccionados considerando un vecindario $3 \times 3 \times 3$, realizando así una comparación exhaustiva de todos los puntos del espacio-escala con sus 26 vecinos más cercanos.

Con el fin de obtener una representación compacta de la imagen y también para hacer frente al ruido, el algoritmo selecciona la característica más salientes (i.e) de un conjunto de máximos locales, esto es logrado usando un umbral t_H en la respuesta del operador del Determinante del Hessiano para cada uno de los puntos de interés candidatos $[x, y]^T$ en la escala L.

$$\mid D\tilde{H}^L(I)(x, y) \mid > t_H \tag{3.43}$$

3.2.6. Descripción Local

De la etapa anterior, se obtiene un conjunto de puntos de interés N en el espacio-escala $M_i : (x_i, y_i, L_i)_{i,\ldots N}$ lo cual permite codificar características invariantes a escala en la imagen. Sin embargo antes de construir el descriptor local, primero se tiene que definir para cada punto de interés una orientación dominante siguiendo el procedi-

miento que se detalla en los párrafos subsecuentes, a fin de lograr invarianza total a la rotación.

3.2.7. Orientación del Punto de interés

Con ese propósito, primeramente se calcula la respuesta Haar wavelet en la dirección de x y y, para cada punto de interés M_i, así mismo se considera un vecindario $B_{6Li}(x_i, y_i)$ donde $6Li$ se define como el radio del disco y L como la escala donde el punto de interés ha sido detectado con centro en (x_i, y_i), En consonancia con el resto, también el tamaño de los wavelets es dependiente de la escala y ajustado a una longitud de 4σ. Por lo tanto se pueden volver a utilizar imágenes integrales para el filtrado rápido. Enseguida se muestras los filtros utilizados Figura 3.25.

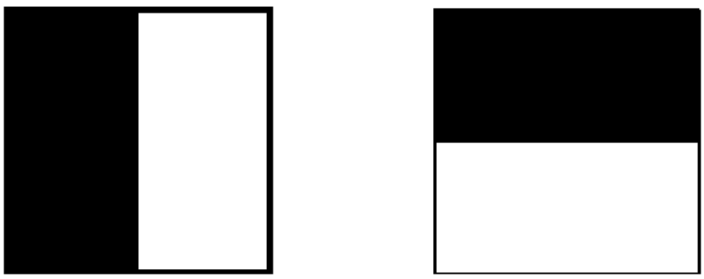


Figura 3.25: Filtros Haar wavelet, calculo de la respuesta en la dirección de x(izquierda) y y (derecha), La parte oscura tiene un peso de -1 y la parte blanca de +1

El calculo de la respuesta wavelet en esa escala L_i en ese vecindario es obtenido por una convolución con un filtro gaussiano (lo cual se puede ver en la definición correspondiente de la respuesta Haar wavelets en la Ecuación 3.43.) para evitar efectos secundarios, todas estas muestras del gradiente se ponderan usando un kernel gaussiano, con una desviación estándar igual a $s = 2[L_i]$, la cual depende de la distancia Euclidiana de la muestra del punto de interés.

$$\phi_j := \left(\tilde{D}_x^{L_i} * I(x_j, y_j), \tilde{D}_y^{L_i} * I(x_j, y_j)\right) \times g_s(x_j - x_i, y_j - y_i) \forall (x_j, y_j) \in B_{6L_i}(x_i, y_i) \tag{3.44}$$

Contrario a SIFT, en el que un histograma se construye para estimar la orientación dominante, el algoritmo de SURF calcula el máximo de la función siguiente:

$$\Psi(\theta) = \begin{pmatrix} \Psi_x(\theta) \\ \Psi_y(\theta) \end{pmatrix} = \sum_{j \in J(\theta)} \phi_j \tag{3.45}$$

donde,

$$J(\theta) := \left\{ j, < \phi_j \in [\theta - \frac{\pi}{6}, \theta + \frac{\pi}{6}] \text{ y } (x_j, y_j) \in B_{6\sigma_i}(x_i, y_i) \right\}$$

La orientación dominante es estimada calculando la suma de todas las respuestas dentro de una ventana deslizante de tamaño $\frac{\pi}{3}$ ver la Figura 3.26.

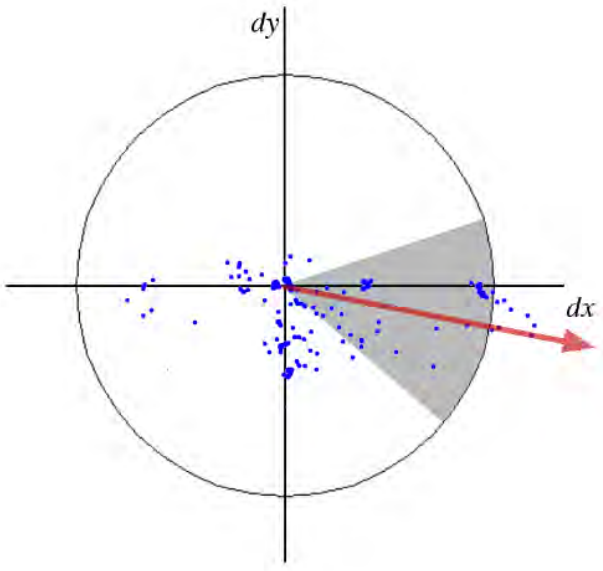


Figura 3.26: Asignación de la orientación de cada sector Un vector de orientación es calculado como representante de todas las respuestas Gaussianas de Haar en los puntos de muestreo contenidos en cada sector circular de valor $\frac{\pi}{3}$

La respuesta horizontal y vertical dentro de la ventana se suman. Las dos respuestas sumadas producen un vector de orientación local, por tanto el vector de mayores dimensiones en todas las ventanas define la orientación del punto de interés.

3.2.8. Creación del Descriptor

Es en esta última etapa del proceso donde se concreta la creación del descriptor SURF. Se construye como primer paso una región cuadrada de tamaño $20L_i$ alrededor del punto de interés y orientada en relación a la orientación calculada en la etapa anterior. Esta región es a su vez dividida en 4×4 sub-regiones dentro de cada una de las cuales se calculan las respuestas de Haar de puntos con una separación de muestreo de 5×5 en ambas direcciones. Por simplicidad, se consideran dx y dy las respuestas de Haar en las direcciones horizontal y vertical, respectivamente, relativas a la orientación del punto de interés. En la Figura 3.27 están representadas tanto las respuestas de Haar en cada una de las sub-regiones como las componentes dx y dy uno de los vectores. Para dotar a las respuestas dx y dy de una mayor robustez ante deformaciones geométricas y errores de posición, éstas son ponderadas por una Gaussiana de valor $\sigma = 3.3\sigma_i$ centrada en el punto de interés. En cada una de las sub-regiones se suman las respuestas dx y dy obteniendo así un valor de dx y dy representativo por cada una de las sub-regiones.
Al mismo tiempo se realiza la suma de los valores absolutos de las respuestas $\mid dx \mid$ y $\mid dy \mid$ en cada una de las sub-regiones, obteniendo de esta manera, información de la polaridad sobre los cambios de intensidad. En resumen, cada una de las sub-regiones queda representada por un vector v de componentes:

$$v = \left(\sum_{i=1}^{32} dx, \sum_{i=1}^{32} dy, \sum_{i=1}^{64} \mid dx \mid, \sum_{i=1}^{64} \mid dy \mid \right) \tag{3.46}$$

y por lo tanto, englobando las 4×4 subregiones, resulta un descriptor SURF con una longitud de 64 valores para cada uno de los puntos de interés identificados.

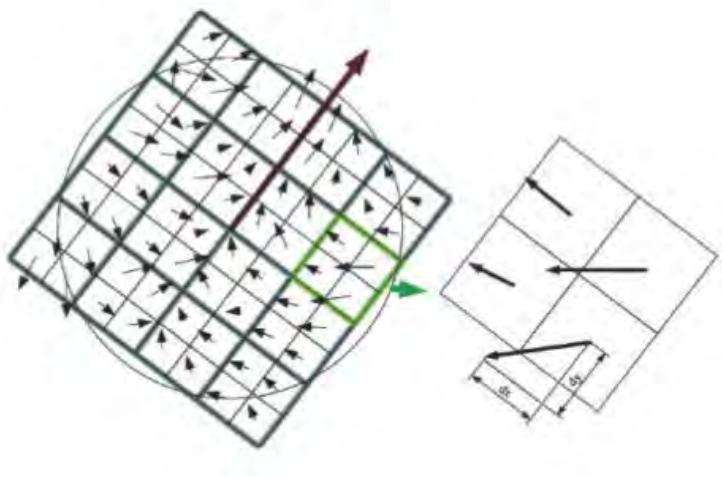


Figura 3.27: Respuestas de Haar en las sub-regiones alrededor del punto de interés

La respuesta Wavelet es invariante a sesgos en la iluminación (offset). La invarianza de contraste (un factor de escala) se consigue girando el descriptor en un vector unitario.

La Figura 3.28 muestra las propiedades del descriptor, para tres imágenes con una intensidad distintiva dentro de una sub-región, se podría imaginar combinaciones de estos patrones de intensidad locales, dando como resultado un descriptor distintivo.

SURF, como se ha mencionado con anterioridad, es hasta cierto punto, similar al concepto de SIFT en que ambas se centran en la distribución espacial de la información del gradiente. Sin embargo, SURF supera en casi todos los casos al algoritmo de SIFT. Se argumenta que esto es debido al hecho de que de SURF integra la información gradiente en un sub-parche, mientras que SIFT depende de las orientaciones de los gradientes individuales. Esto hace que SURF sea menos sensible al ruido, como lo ilustra la Figura 3.27. Es decir se suman los valores de las direcciones en contrario a SIFT.

Como se puede ver en la Figura 3.29 donde se muestran dos imágenes una totalmente limpia y otra con un poco de ruido, se puede ver cómo se obtienen las direcciones correspondientes de cada una de las imágenes. Sin embargo, al sacar la dirección

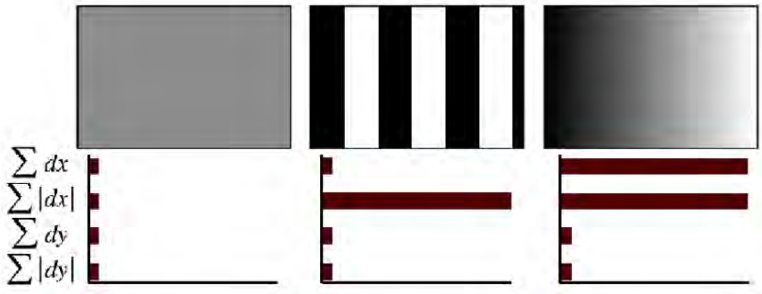


Figura 3.28: Las entradas del descriptor de una sub-región representa el tipo de patrón de intensidad subyacente izquierda: En caso de una región homogénea, todos los valores son relativamente bajos. Medio: En presencia de frecuencias en la dirección x, el valor de $\sum_{i=0}^{64} \mid dx \mid$ es alta, pero todos los demás se mantienen bajos. Si la intensidad aumenta progresivamente en la dirección x ambos valores $\sum_{i=0}^{32} dx$ y $\sum_{i=0}^{64} \mid dx \mid$ son altos.

dominante en la imagen de ruido se puede ver como se generan diferentes direcciones, a diferencia de SURF el cual suma los valores y por ende se puede obtener una dirección dominante más precisa.

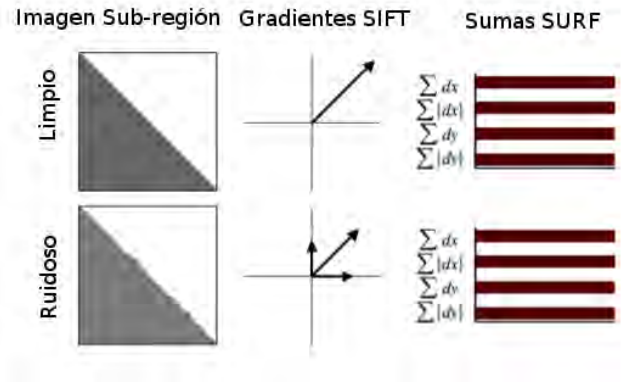


Figura 3.29: Debido a la integración global SURF, se mantiene más sólido a la perturbación de diferentes imagenes, que el descriptor SIFT más local en funcionamiento.

Capítulo 4

Seguimiento y Correspondencia de Objetos

4.1. Correspondencia de Características

De los pasos anteriores, dos imágenes correspondientes pueden ser representadas por dos conjuntos de puntos de interés con su correspondiente descriptor SURF. Recordando que el descriptor SURF es un vector con una dimecionalidad de 64 compartimientos, con norma unitaria Euclidiana. El paso de la correspondencia se realiza hasta este punto, comparando exhaustivamente los vectores, combinado con la distancia del radio del vecindario más próximo (NN-DR), La distancia Euclidiana entre el descriptor de la primera imagen y cualquier descriptor de la segunda imagen son calculados. Solo los más cercanos y los segundos vecinos más cercanos son considerados. Si el radio de estas dos distancias es menor a 0.8 la correspondencia entre el descriptor consultado y su candidato es validado. Tal umbral permite descartar numerosas discordancias, un ejemplo de lo que se menciona en la correspondencia de características puede ser visto en la Figura 3.30

Una vez que se ha calculado los descriptores, sólo queda realizar la correspondencia entre los puntos de interés, ya sea de dos imágenes o de un conjunto de imágenes

Figura 4.1: Ejemplo de la correspondencia de puntos característicos de SURF

(e.q.). Para ello Lowe [Lowe, 2004] propone el método de vecinos más próximos. Este método está basado en la distancia Euclidiana entre vectores.

El algoritmo consiste en calcular los descriptores de una imagen A, en el cual se quiere encontrar un objeto que está definido en una imagen B. Cuando ya se tienen los vectores de los descriptores de todos los puntos de interés, por cada punto de interés de la imagen A se calcula una distancia que existe entre un vector descriptor y todos los demás vectores de los puntos característicos de la imagen B. Una vez que se tiene calculadas todas las distancias por cada punto de interés de la imagen A, se calcula la relación entre las dos distancias menores. Si esta relación es menor que el valor umbral, existe una correspondencia entre el punto de interés de A y el punto de interés de B más cercano. Para calcular el grado de similitud entre ellos existen varias alternativas, sin embargo como ya se ha hecho mención se utiliza la distancia Euclidiana.

$$Ds_t = \sqrt{\sum_{i=1}^{64}(a_i - b_i)^2} \tag{4.1}$$

donde Ds_t es la distancia Euclidiana que hay entre los 64 elementos de ambos des-

criptores. Realizando una serie de operaciones entre cada descriptor de la imagen A con cada una de la imagen B, se podrá decidir cuales son correspondientes con mayor probabilidad, eligiendo siempre el que tenga la distancia Euclidiana menor.

El coste computacional total de las comparaciones es muy elevado si no se fijan restricciones. Por ejemplo, si se sabe con certeza que ciertos puntos de la imagen A no pueden aparecer de ningún modo en una zona determinada de la imagen B, se podrá descartar los descriptores de esa zona y así ahorrar el coste de las comparaciones. Por tanto, en función del tipo de aplicación que se quiera implementar y de la información que se tenga a *priori*, se podrá optimizar el código de una forma u otra. Un ejemplo del la correspondencia de puntos de interés es el que se muestra en la Figura 3.31.

Figura 4.2: Correspondencia entre Puntos de Interés

Una cuestión importante que queda como medida característica distintiva es, cómo la fiabilidad de la correspondencia varía en función de la cantidad de características dentro de la base de datos, o del conjunto que se obtiene de cada imagen, en los párrafos subsecuentes se mencionan algunas maneras eficientes de trabajar con grandes cantidades de información utilizando cúmulos (clustering) así como una aproximación, mejora a los vecindarios próximos.

4.1.1. Indexación Eficiente de Vecinos Próximos

No se conoce algoritmo que pueda identificar a los vecinos más cercanos de una manera exacta en espacios dimensionales elevados. Y que sean más precisos que la búsqueda exhaustiva. El descriptor que ahora se describe, tiene un vector de características de tamaño 64 y el mejor algoritmo, tal como el k-d Tree [Friedman et al., 1977] proporciona no más que una aceleración de búsqueda exhaustiva de aproximadamente 10 espacios dimensionales. Por lo tanto, se ha usado una aproximación al algoritmo, llamado algoritmo Mejor-Caja-Primero , *Best-Bin-First(BBF)* [Beis y Lowe, 1997], esto es aproximado en el sentido de que devuelve el vecino más cercano con alta probabilidad.

El algoritmo BBF utiliza una búsqueda modificada para ordenar el algoritmo K-d Tree, para que los contenedores en el espacio de características sean buscados en orden de su distancia más cercana consultando su ubicación. Esta búsqueda de prioridad, fue examinada por primera vez, por [Arya et al., 1994] y proporcionan más estudios de sus propiedades computacionales en [Arya et al., 1998]. Esta forma de búsqueda requiere el uso de una cola de prioridad basado en una pila para la determinación eficiente de la búsqueda.

4.1.2. Indexación rápida para la correspondencia

Para la rápida indexación durante la correspondencia, el signo del Laplaciano (traza de la Matriz Hessiana) para el punto de interés subyacente se incluye. Generalmente, el punto de interés es encontrado en una estructura tipo burbuja, el signo del Laplaciano distingue manchas brillantes sobre fondos oscuros a partir de la inversa. Esta característica está disponible sin costo computacional extra, ya que se calcula ya en la fase de detección. En la etapa de correspondencia, sólo se comparan las características si tienen el mismo contraste como lo muestra la Figura 3.32, por lo tanto esta información mínima permite una rápida adaptación, sin reducir el rendi-

miento del descriptor. Es evidente que esto también es una ventaja para métodos de indexación más avanzados, por ejemplo para el k-d Tree, esta información adicional define un hiperplano significativo para dividir los datos, en lugar de elegir al azar un elemento o característica utilizando estadística.

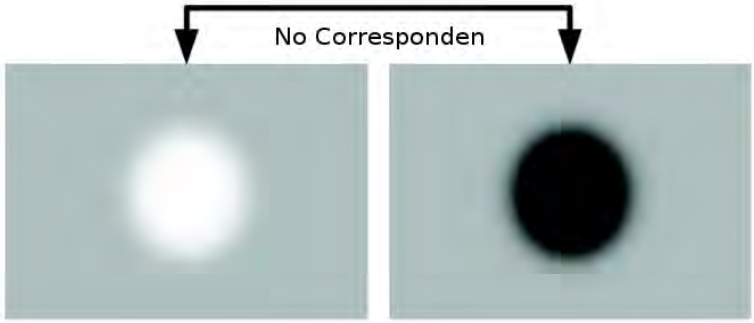


Figura 4.3: Si el contraste entre dos puntos de interés es diferente (oscuro en un fondo claro vs claro en un fondo oscuro), el candidato no es considerado como correspondencia válida

4.1.3. Evaluación del tiempo entre SIFT y SURF

Si bien es cierto que en las secciones anteriores, se ha mencionado, la diferencia de calculo y el tiempo de respuesta, entre estos dos algoritmos, es evidente que faltaría, tal demostración. Ahora bien, los siguientes cálculos se han implementando con diferentes imágenes de distintos tamaños, dado que la idea principal de este proyecto es tomar una segmento de un marco de un secuencia de vídeo, no importando la forma inicial del recorte. Es decir, no es necesario que el segmento sea cuadrado para hacer un buen cálculo. De hecho, puede ser rectangular en la mayoría de los casos. Es por esto que el cálculo se hace con diferentes tamaños y a colores o en su defecto blanco y negro. El equipo donde se implementaron los siguientes experimentos fue en una computadora con procesador intel i7-2600 segunda generación, con 8G de Ram. Así como una laptop con procesador i7-26700Qm segunda generación, con 6G

de memoria Ram, bajo el sistema operativo Linux. Igualmente todas las implementaciones están desarrolladas en el lenguaje de programación C y C++, incluyendo la librería OpenCV. Los experimentos se ejecutaron con algunos servicios funcionando en el *background* de las dos computadoras. Sin embargo no afecto la ejecución de los programas.

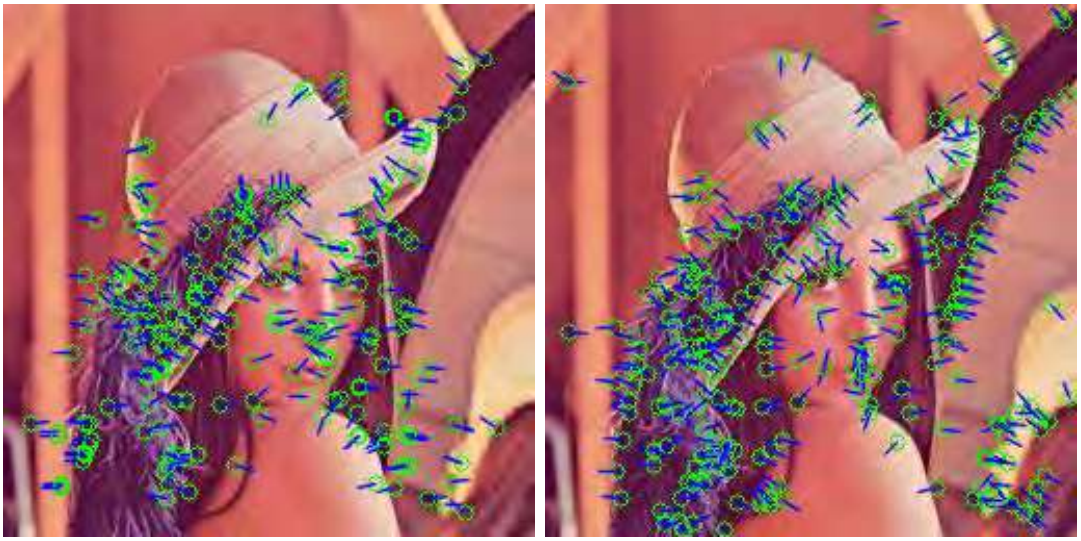


Figura 4.4: De izquierda a derecha cálculos de puntos de interés así como las direcciones dominantes de cada punto de SURF y SIFT respectivamente.

La imagen de Lena de 256×256 que se muestra en la Figura 3.33 arrojó los siguientes resultados en cuanto a tiempo se refiere.

Iteración / Algoritmo	1	2	3	4	5	Promedio
SURF	14.7ms	13.7ms	14.1ms	12.8ms	13.2ms	13.7ms
SIFT	114.8ms	114.2ms	112.0ms	109.7ms	113.9ms	112.9ms

Tabla 4.1: Tiempo entre el algoritmo de SURF y SIFT en la Figura 3.33.

Como se puede ver en los resultados, es evidente el tiempo que se tarda en encontrar los puntos característicos uno del otro.Sin embargo esta no es la única prueba que se

hizo, enseguida los siguientes ejemplos. La Figura 3.34 muestra el taxi de hamburgo de medidas 252×138 en escala de grises

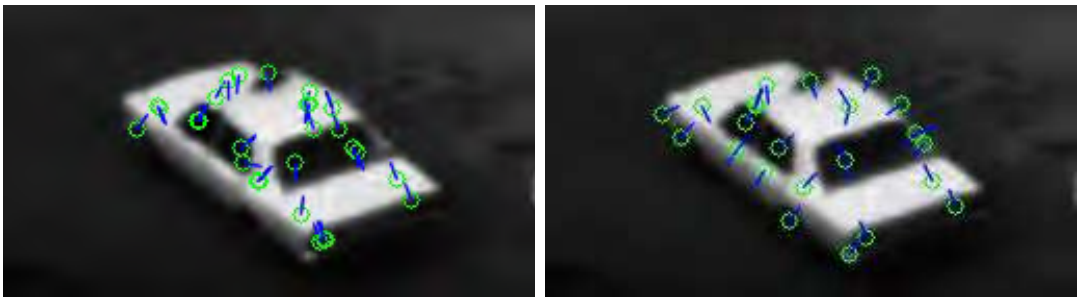


Figura 4.5: De izquierda a derecha cálculos de puntos de interés así como las direcciones dominantes de cada punto de SURF y SIFT respectivamente.

Iteración / Algoritmo	1	2	3	4	5	Promedio
SURF	6.1ms	6.9ms	5.7ms	6.8ms	6.52ms	6.40ms
SIFT	29.5ms	30.3ms	31.6ms	28.9ms	30.5ms	30.1ms

Tabla 4.2: Tiempo entre el algoritmo de SURF y SIFT en la Figura 3.34.

Asimismo se aplicaron los mismo algoritmos a imágenes de mucho mayor tamaño como la imagen de 1200 × 746 Figura 3.35, con estos ejemplos se ilustra la velocidad con que se desempeña el algoritmo de SURF en cuanto al cálculo de los puntos de interés y creación del descriptor se refiere.

Iteración / Algoritmo	1	2	3	4	Promedio
SURF	151.2ms	119.9ms	129.7ms	126.8ms	131.9ms
SIFT	1034.5ms	1011.6ms	1010.9ms	1014.5ms	1017.875

Tabla 4.3: Tiempo entre el algoritmo de SURF y SIFT en la imagen 3.35.

El propósito de este trabajo de tesis está enfocado en detectar un objeto y seguirlo, a pesar de las deformaciones que de él surjan en el trascurso del seguimiento.

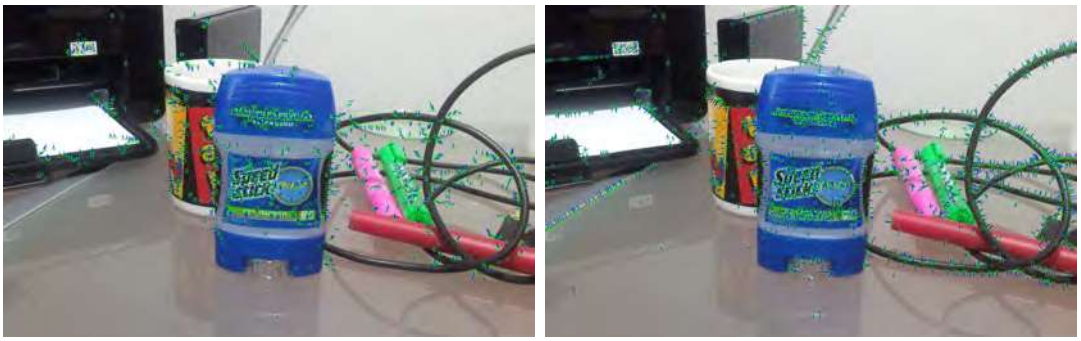


Figura 4.6: De izquierda a derecha cálculos de puntos de interés así como las direcciones dominantes de cada punto de SURF y SIFT respectivamente.

El objeto será seleccionado por un click y arrastre de ratón, demarcando el objeto, por una figura en la mayoría de los casos rectangular. Si bien es un hecho que tiene bastante precedente en el estado del arte del seguimiento y la detección de objetos, la mayoría de los enfoques parten de redes neuronales, clasificadores en cascada, (como lo muestran en [Viola et al., 2005]), aprendizaje supervisado, seguimiento por color, enfocados en un objeto en particular, sin la posibilidad de seleccionar otro objeto de interés. El trabajo se centra en sólo un recorte o muestra, obtenido por el click y el arrastre del ratón, aprovechando uno de los algoritmos que se describieron en el Capítulo 3 como lo es el algoritmo de SURF, donde se obtienen puntos característicos de una imagen en concreto y a su vez se hace correspondencia entre los puntos característicos de otra imagen que contengan el objeto de interés. Hecho notable debido a que con una sola muestra se puede hacer el seguimiento de un objeto de manera robusta y adaptable indistintamente de las modificaciones rígidas del objeto en cuestión, aunado a que si el objeto sale de escena es retomado cuando este es enfocado nuevamente. Un ejemplo de la obtención de la muestra se puede ver en la Figura 4.1.

Partiendo de algunos ejemplos sobre la detección, donde un factor claro e importante es la homografía, la transformación afín de la imagen, o del marco del vídeo, con el único propósito de adecuar el rectángulo, cuadro, o en su defecto polígono irregular que indique lo que se esta siguiendo. La homografía en este caso se realiza teniendo

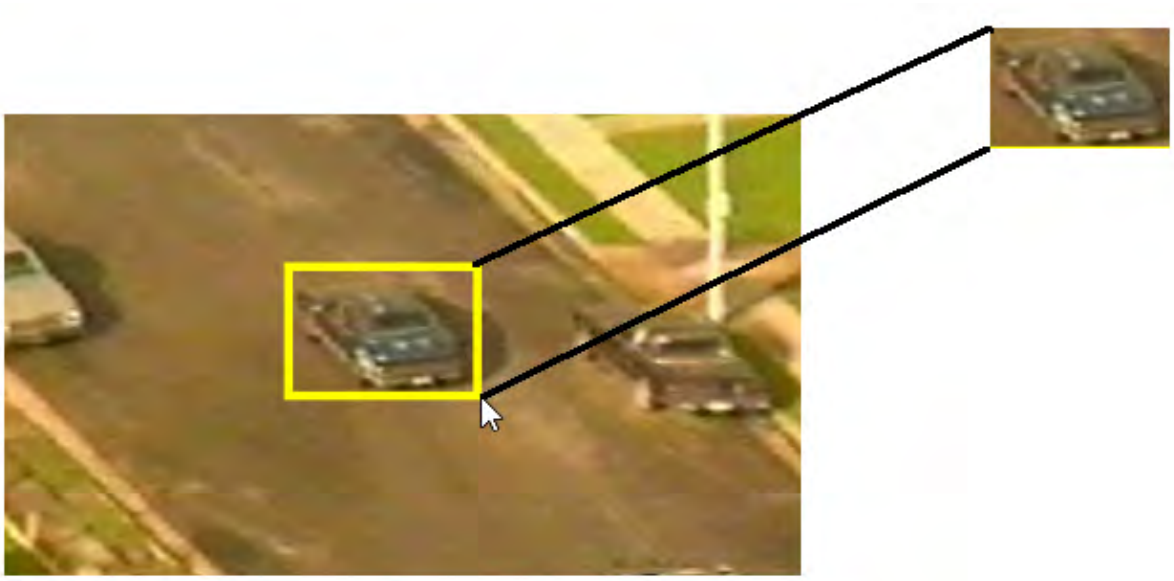


Figura 4.7: Obtención del objeto de interés mediante el ratón

en cuenta que se conoce el tamaño real del rectángulo en el que está dispuesto la muestra u objeto de interés, para de esta forma obtener la orientación y la posición adecuada del objeto. La homografía que da la posición real del objeto en función de la imagen o del marco del vídeo responde a la siguiente ecuación

$$\begin{bmatrix} t_x \\ t_y \\ t \end{bmatrix} = M \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{4.2}$$

Donde la matriz M se calcula utilizando los datos de las dimensiones reales del rectángulo y los datos dónde se encuentra el rectángulo en la imagen. En la siguiente Figura 4.2 se puede ver un ejemplo

Asimismo en el siguiente apartado se menciona el uso de los cúmulos de puntos de intéres mediante la transformada Hough, para el manejo de parámentros afines y la búsqueda de los mejores valores que se adecúen a la transformación del objeto seguido, no sin antes mencionar y resaltar que hay un extenso número de métodos, técnicas

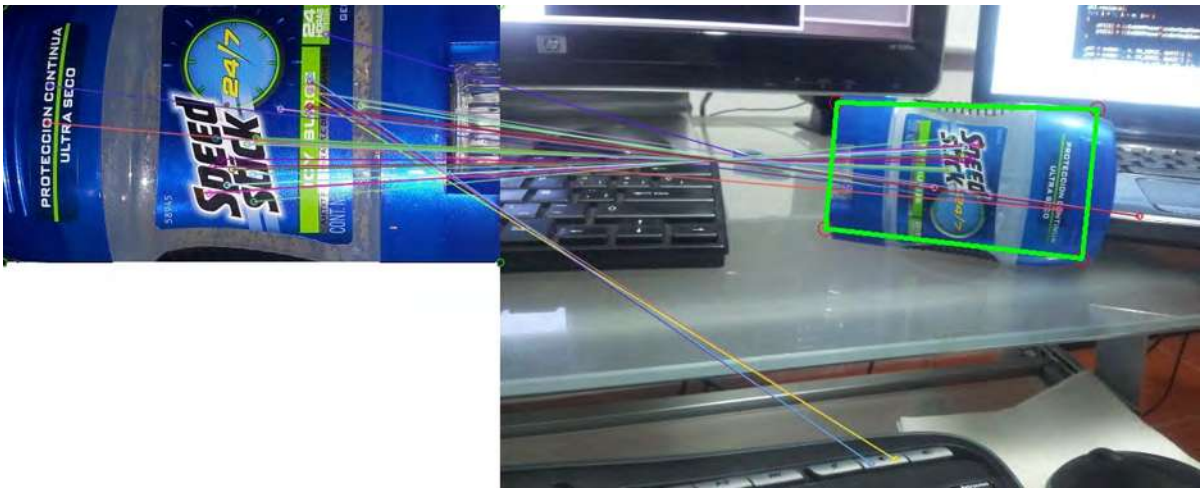


Figura 4.8: Correspondencia de un objeto

para encontrar estos valores. Sin embargo, sólo unos cuantos se pueden utilizar para el reconocimiento y seguimiento en tiempo real. No hay una manera establecida para encontrar esta transformación, no existen funciones, métodos definidos que indiquen una solución real de estos parámetros afines. Es decir, se puede abordar de diferentes maneras la búsqueda de la transformación afín. Debido a esto, se puede afectar el rendimiento en la búsqueda de la mencionada transformación. Entonces, el desempeño dependería directamente de aplicar los algoritmos, métodos, funciones, más óptimos para esta búsqueda. Por ello sólo se mencionan estos dos métodos debido a su rendimiento en cuanto a la búsqueda de los parámetros de la transformación afín se refiere y de igual forma al estudios pasos previos de estos métodos, algoritmo, para tal labor, los cuales se describen a continuación.

4.1.4. Transformada de Hough

La Transformada de Hough es una técnica para la detección de figuras en imágenes . Con la transformada de Hough es posible encontrar algunos tipos de figuras que puedan ser expresadas matemáticamente, tales como rectas, circunferencias o elipses. En el análisis automático de imágenes, es común encontrar el problema de detectar

figuras, simples como rectas círculos etc, [Eichmann y Dong, 1983]. Sin embargo, como primer paso, se puede usar un detector de bordes para obtener los puntos de la imagen que pertenecen a la frontera de la figura deseada. Debido a las imperfecciones, ya sea de la imagen captada o del detector de bordes, existen muchos puntos que pertenecen a la curva y que faltan en la imagen; también pueden existir desviaciones espaciales entre la figura ideal (por ejemplo, una recta) y los puntos ruidosos del borde detectado. Por estas razones, usualmente no es trivial agrupar los bordes detectados en un conjunto apropiado de figuras, ya sean circunferencias o cualquier otra figura. El objetivo de la transformada de Hough es resolver este problema, haciendo posible realizar agrupaciones de los puntos que pertenecen a los bordes de posibles figuras a través de un procedimiento de votación sobre un conjunto de figuras parametrizadas.

La transformada de Hough usa una matriz, llamada acumulador, cuya dimensión es igual al número de parámetros desconocidos del problema. Para construir el acumulador es necesario discretizar los parámetros que describen la figura. Cada celda del acumulador representa una figura cuyos parámetros se pueden obtener a partir de la posición de la celda. Por cada punto en la imagen, se buscan todas las posibles figuras a las que puede pertenecer ese punto. Esto se logra buscando todas las posibles combinaciones de valores para parámetros que describen la figura (los posibles valores se obtienen a partir del acumulador). Si es así, se calculan los parámetros de esa figura, y después se busca la posición en el acumulador correspondiente a la figura definida, y se incrementa el valor que hay en esa posición. Las figuras se pueden detectar buscando las posiciones del acumulador con mayor valor (máximos locales en el espacio del acumulador). La forma más sencilla de encontrar estos picos es aplicando alguna forma de umbral, pero distintas técnicas podrían dar mejores resultados en distintas circunstancias, determinando donde se encuentran las figuras y cuantas hay. Por los errores que se pueden cometer detectando bordes, existirán imperfecciones en el espacio acumulador, lo que puede hacer que no sea trivial encontrar los picos correctos y por tanto las figuras apropiadas. Si bien no es un ejemplo de un objeto

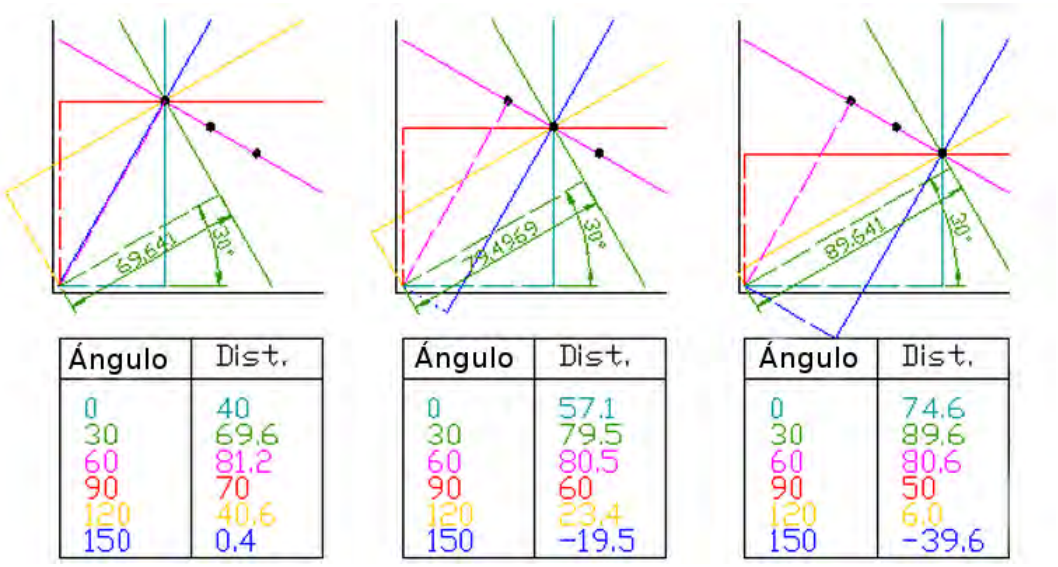
como tal, el siguiente pseudocódigo 1 muestra como detectar una recta en una imagen cualquiera, de la forma $y = m * x + n$, donde la dimensión del acumulador seria de dos ya sea su representación en coordenadas cartesianas (m, n) o en coordenadas polares (ρ, θ), son desconocidos. Las dos dimensiones del acumulador corresponden a los valores cuantificados para (ρ, θ).

Algoritmo 1: Algoritmo para detectar rectas en una Imagen

```
while Detectar los bordes de la imagen;
do
    Por cada punto en la imagen;
    if Si el punto (x, y) esta en un borde:;
    then
        while Por todos los posibles ángulos θ;
        do
            Calcular ρ para el punto (x, y) con un ángulo θ;
            Incrementar la posición (ρ, θ) en el acumulador;
        end
    end
    Buscar las posiciones con los mayores valores en el acumulador;
end
Devolver las rectas cuyo valores son los mayores en el acumulador.;
```

Por cada punto se dibujan un número de líneas que pasan por el mismo, con distintos ángulos, líneas contiguas en la Figura 4.3. De igual forma por cada línea se dibuja una recta perpendicular a ésta la cual pasa por el origen, líneas discontinuas. La longitud y el ángulo de cada línea discontinua se calcula, los resultados se muestran en las tablas subsecuentes, esto se repite por cada punto.

Finalmente se crea un grafo con las longitudes de la líneas por cada ángulo, conocido como grafo de espacio de Hough. El punto donde las curvas se intersecan da la distancia y el ángulo. Esta distancia y este ángulo indican la recta que se interseca con



Ángulo	Dist.
0	40
30	69.6
60	81.2
90	70
120	40.6
150	0.4

Ángulo	Dist.
0	57.1
30	79.5
60	80.5
90	60
120	23.4
150	-19.5

Ángulo	Dist.
0	74.6
30	89.6
60	80.6
90	50
120	6.0
150	-39.6

Figura 4.9: Algoritmo para detectar rectas en una imagen

los puntos anteriores.

El grafo muestra el punto rosado donde se interceptan las curvas, este punto corresponde a la recta rosada de la Figura 4.3, que pasa por los tres puntos negros. Ahora bien, este ejemplo demarca las secciones subsecuentes para la detección de un objeto cualquiera, basándose en el conjunto de puntos característicos que sólo son representativos del objeto de interés.

4.1.5. Cúmulos con la transformación de Hough

Para maximizar el rendimiento del reconocimiento de objetos, para objetos pequeños o tapados por un objeto de la escena, se desea identificar objetos con el menor número de características coincidentes, se ha encontrado en el estado del arte de la correspondencia de puntos de de interés, que es posible reconocer con sólo tres características un objeto [Lowe, 2004]. Una imagen contiene alrededor de 2000 o más características, que pueden provenir de muchos objetos diferentes, así como del fondo de la escena. Siguiendo lo que se mencionó en la correspondencia de características

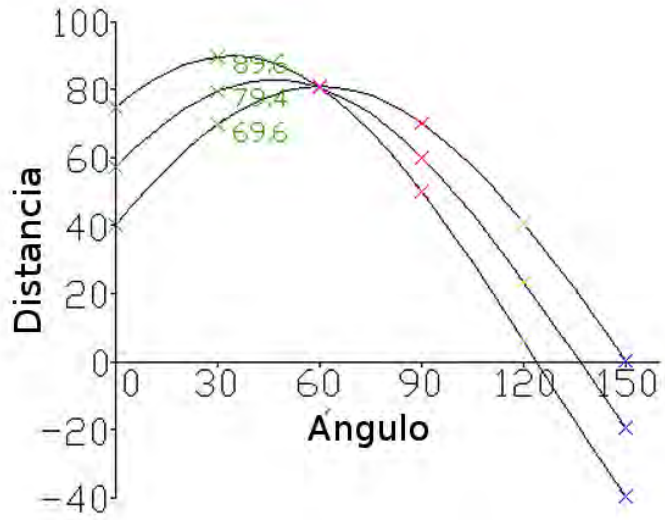


Figura 4.10: Grafo con las longitudes de las líneas por cada ángulo

sección 3.16 donde se permite descartar muchos falsos coincidentes que surgen del desorden del fondo. Sin embargo, esto quita coincidencias de otros objetos validos, que a menudo todavía se tienen que identificar correctamente, los cuales contienen menos del 1 % de valores típicos(inliers) entre el 99 % de valores atípicos (outliers).

Muchos métodos de ajuste robustos como RANSAC o mínimos cuadrados tiene un mal desempeño cuando el porcentaje de valores típicos (inliers) caen por debajo del 50 %. Sin embargo, un rendimiento mucho mejor se puede obtener con un conjunto de características en pose del espacio usando la transformación de Hough como lo menciona Hough, Ballard, Grimson en [Ballard, 1987], [Hough, 1962], [Grimson, 1990]. La transformada de Hough identifica grupos de características con una interpretación uniforme por el uso de cada característica para votar por todos los objetos base que sean consistentes con la característica. Cuando se encuentra un conjunto de características, se vota por la misma muestra de un objeto. La probabilidad de que la interpretación sea correcta es mucho mayor que la de una característica individual de cada uno de los puntos de interés. Cada uno de los puntos característicos generados especifica 4 parámetros: Ubicación, escala a la que pertenece, orientación dominante

y cada punto tiene un registro de los parámetros del punto significativo en relación con la imagen en la que se encontró. Por lo tanto, se puede utilizar la transformada de Hough prediciendo la ubicación del modelo, la orientación y la escala de la hipótesis de partida. Esta predicción tiene margenes de error grandes, dado que la transformada de similitud implícita en estos 4 parámetros es sólo una aproximación a la totalidad de los 6 grados de libertad espacial de un objeto 3D y no se tienen en cuenta ninguna deformación no rígida. Por lo tanto, se utilizan amplios contenedores (bins) de 30 grados para la orientación, un factor de escala de 2 y 0.25 veces el entrenamiento máximo proyectado en las dimensiones de la imagen (según la escala prevista) para la ubicación. Para evitar el problema de los efectos de frontera en la asignación del contenedor, cada punto clave coincide con los votos para los 2 contenedores más próximos en cada dimensión dando un total de 16 entradas para cada hipótesis y ampliar aun más el rango de la pose. En la mayoría de las implementaciones de la transformada de Hough, una matriz multidimensional se utiliza para representar los contenedores. Sin embargo, muchos de los posibles contenedores quedarán vacíos, y es difícil de calcular el rango de posibles valores de ubicación debido a su dependencia mutua (por ejemplo, la dependencia de la discretización local en la escala seleccionada). Estos problemas pueden evitarse mediante el uso de una función hash pseudo-aleatoria de los valores de ubicación para insertar votos en una tabla hash de una sola dimensión, en la que se detectan fácilmente colisiones.

4.1.6. Solución para los parámetros afines

La transformada Hough es usada para identificar todos los cúmulos con al menos 3 entradas por cada cajón (bin), acumulador. Cada cúmulo se somete después a un procedimiento de verificación geométrica en la que se realiza una solución de mínimos cuadrados para los mejores parámetros afines de la imagen en relación a la formación de la nueva imagen. Una transformación afín correctamente representa rotaciónes 3D de una superficie plana bajo una proyección ortogonal, sin embargo la aproximación

puede ser pobre para la rotación 3D de objetos no planos. Una solución más general sería la de resolver la matriz fundamental [Luong y Faugeras, 1995]. No obstante la solución de la matriz fundamental requiere al menos 7 puntos que coincidan, en comparación con soló 3 que la solución afín en la práctica requiere para una buena estabilidad. Ahora bien, al realizar el reconocimiento con tan sólo 3 características, la transformación afín proporciona un punto de partida mucho mejor y se podrán ver los errores en la aproximación afín al permitir grandes errores residuales.

Si se imagina la colocación de una esfera alrededor de un objeto, entonces la rotación de la esfera con unos 30 grados, no moverá ningún punto sobre ella más de 0.25 veces el diámetro proyectado de la esfera. Para los ejemplos de los objetos 3D mencionados en [Lowe, 2004] una solución afín funciona dado que no permite errores residuales de hasta 0.25 veces las dimensión máxima proyectada del objeto. Un enfoque más general se da en [Brown y Lowe, 2002a] en donde la solución inicial se basa en una transformada de similitud, que luego progresa hacia la matriz fundamental. Son en estos casos en lo que se encuentran un número suficiente de coincidencias.
El modelo de la transformación afín de un punto $[x\ y]^T$ a un punto de la imagen $[u\ v]^T$ puede ser escrito como sigue:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \tag{4.3}$$

donde el modelo de traslación es $[t_x\ t_y]^T$ y la rotación afin y escala son representados por los parámetros m_i. Luego entonces, se resuelve para los parámetros de la tranformación, por lo que la ecuación anterior se puede reescribir para juntar las incógnitas en un vector columna:

$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \\ & & \cdots & & & \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u \\ v \\ \vdots \end{bmatrix} \tag{4.4}$$

Esta ecuación muestra una sola correspondencia, pero para cualquier número de correspondencias, pueden ser añadidas, por cada correspondencia se contribuye con dos filas al principio y dos al final de la matriz. Al menos tres correspondencias son necesarias para proporcionar una solución. Se puede reescribir el sistema lineal como:

$$Ax = b \tag{4.5}$$

La solución de mínimos cuadrados para los parámetros x pueden ser determinados resolviendo las ecuaciones normales correspondientes.

$$x = [A^T\ A]^{-1} A^T b \tag{4.6}$$

La cual minimiza la suma de los cuadrados de las distancias, desde la ubicación del modelo, proyectando en las ubicaciones correspondientes de la imagen. Este enfoque de mínimos cuadrados fácilmente podría extenderse a resolver 3D y los parámetros internos de objetos articulados y flexibles. Los valores atípicos se pueden eliminar por la comprobación de la concordancia entre cada característica de la imagen y el modelo o imagen de interés [Lowe, 1991].

Dada la solución más exacta de mínimos cuadrados, ahora se requiere que cada correspondencia concuerden dentro de la gama media de error que se utiliza para los parámentros en los contenedores (bins) de la transformada de Hough. Si hay menos de 3 puntos, quedan descartados los valores extremos, y entonces la correspondencia es rechazada. Como valores extremos, se descartan la solución de mínimos cuadrados

y se resolverán con los puntos restantes, y se itera el proceso. Adicionalmente una fase de correspondencias de barrido arriba hacia abajo se utiliza para agregar más correspondencias de acuerdo con el modelo proyectado. Esto pudo haberse perdido en contenedor (bin) de la transformada Hough debido a la similitud de la aproximación y de otros errores. La decisión final de aceptar o rechazar una hipótesis del modelo se basa en un modelo probabilístico detallado en [Lowe, 2001]. Este método calcula inicialmente falsas correspondencias con el modelo base, dado el tamaño del modelo proyectado, el número de características en una región, y la precisión del ajuste. Posteriormente se realiza un análisis Bayesiano de la probabilidad de que el objeto esté presente, basándose en el número real de características encontradas. Se acepta el modelo si la probabilidad final para una interpretación correcta es superior a 0.98. Para los objetos que se proyectan en pequeñas regiones de una imagen, 3 características pueden ser suficiente para un reconocimiento fiable. Para los objetos de mayor tamaño que cubren una amplia parte de una imagen con mucha textura, el número de falsos positivos es mayor y un máximo de 10 características pueden ser necesarias.

4.1.7. RANSAC

RANSAC es una abreviatura de *Random Sample Consensus*. Se trata de un método iterativo para calcular los parámetros de un modelo matemático de un conjunto de datos observados que contiene los valores atípicos (outliers). Se trata de un algoritmo que produce un resultado razonable sólo con una cierta probabilidad, la cual aumenta a medida que se les permite más iteraciones. Una hipótesis básica, es que los datos consisten en datos típicos (inliers), es decir, datos cuya distribución se explica por un conjunto de parámetros de los modelos, aunque pueden estar sujetos a ruidos y sean datos atípicos (outliers), que son datos que no encajan en el modelo. Los valores atípicos pueden llegar, por ejemplo, a partir de los valores del ruido o de las mediciones erróneas o hipótesis erróneas acerca de la interpretación de los datos. RANSAC

también asume que, dado un conjunto (generalmente pequeño) de valores típicos (inliers), existe un procedimiento que puede estimar los parámetros de un modelo que explica o se ajusta a esta información de manera óptima.

La entrada del algoritmo de RANSAC es un conjunto U de datos observados, una proporción desconocida de estos datos es coherente con un modelo de parámetros desconocidos a partir de un espacio de parámetros Θ. El objetivo es encontrar los parámetros del modelo θ' de un espacio de parámetros Θ que maximiza una función de costo $J_S(\theta, \mathrm{U}, \Delta)$. En la formulación estándar, la función de costo J_S es el tamaño del soporte del modelo con los parámetros θ, es decir, el número de datos de U que son coherentes con él. Los datos con los errores más pequeños que Δ se consideran constantes para apoyar el modelo. La función de error $\rho(\theta, \mathrm{x})$ representa la distancia del modelo de los datos típicamente dados.

El algoritmo de RANSAC lleva a cabo la maximización de J_S como sigue. Dos pasos son ejecutados repetidamente:

- Una etapa de generación de hipótesis
- Paso de verificación

En la etapa de generación de hipótesis, una hipótesis θ_k de los parámetros del modelo es calculado a partir de un subconjunto S_k (llamado muestra) seleccionado de los datos de entrada U al azar. La probabilidad de que una muestra contiene al menos un valor atípico se incrementa exponencialmente, con el tamaño de la muestra, por lo tanto, el tamaño de la muestra $m = |S|$ es elegido para ser tan pequeño como sea posible, para determinar los parámetros del modelo de una forma exclusiva. Por ejemplo, una línea se define por dos puntos distintos ($m = 2$) y una matriz fundamental se define por siete correspondencias punto a punto en una posición general. Luego entonces, en la etapa de verificación se calcula la calidad de los parámetros del modelo hipotético. La función de costo J_S es la cardinalidad del soporte del modelo con los parámetros θ_k, entonces el máximo de la función de costo J_S se recupera con una probabilidad

predefinida (confianza) de $1 - \eta_0$ (normalmente establecido al 95 %) donde η_0 es la probabilidad de retornar una mala solución. En el número de las pruebas de la hipótesis. Hay dos tipos de muestras: contaminadas, que contienen al menos un caso atípico (outlier), y muestras no contaminadas (todo valor típico (inlier) o sin valores atípicos). Sólo estos últimos son de interés. Como los parámetros del modelo, son calculados a partir de los datos que incluyen valores atípicos son arbitrarios.

Sea P la probabilidad de que una muestra no contaminada G de tamaño m es seleccionada al azar de un conjunto U, N datos.

Algoritmo 2: Algoritmo para detectar rectas en una Imagen

Entrada: U, δ, η_0;

Salida: $\tilde{\theta}, G$;

while *hasta que la probabilidad* $\eta = (1 - P(G'_k))^k$ *del modelo de búsqueda sea mayor que* G_{k-1} *en el* k*-ésimo paso cae por debajo del umbral* η_0 **do**

Generación de la Hipótesis

a) Seleccionar una muestra aleatoria de tamaño mínimo m de U b) Estimar el modelo de parámetros θ_k ajustando la muestra. **Verificación**

(a) Calcular G_k=$J(\theta_k)$ del modelo (b) Establecer G'_k=$max(G'_{k-1}, G_k)$ y si G'_k (es decir cuando se alcanza un nuevo máximo), posteriormente, se guardan los parámetros del modelo θ_k

end

$$P(G) = \frac{\binom{G}{m}}{\binom{N}{m}} = \prod_{j=0}^{m-1} \frac{G-j}{N-j} \leq \varepsilon^m \tag{4.7}$$

donde ε es la fracción de un valor típico (inlier) $\varepsilon = \dfrac{I}{m}$.

El número de valores típicos (inliers) G no se conoce. Dejando G'_k ser el mayor soporte de un modelo hipotético encontrado hasta la $k-$ésima muestra inclusive. $G'_k = |G'_k|$ el proceso de muestreo es terminado cuando la probabilidad de encontrar un mejor modelo (con el soporte más grande que G'_k) que cae bajo un umbral. Es decir, cuando la probabilidad η de perder una serie de valore típicos (inliers) G'^+ de

tamaño $G'^{+}| \geq G'_k$ dentro de k muestras, cae bajo un umbral predefinido η_0.

$$\eta = (1 - P(G'_k))^k \tag{4.8}$$

El número de muestras que se han de elaborar para satisfacer $\eta \leq \eta_0$ es:

$$k_{\eta_0}(M'_k) = \frac{ln(\eta_0)}{ln(1 - P(M'_k))} \tag{4.9}$$

El muestreo uniforme de m correspondencias puede ser visto también como un muestreo no uniforme del espacio de parámetros Θ. Existen $\binom{N}{M}$ posibles muestras fuera de $\binom{G}{m}$ que están situadas cerca de los parámetros del modelo óptimo θ, mientras que el resto de las muestras (muestras contaminadas) generan modelos con parámentros dispersos sobre Θ. Esta observación también es explotada en la transformada de Hough.

Donde bajo dos condiciones (la aproximación de) todos los extremos locales de la función de costo $J(\theta)$ pueden ser recuperados eficientemente. La primera condición es que la función de costo debe ser una suma de las contribuciones de todos los datos de los puntos. Segundo, debe de haber un procedimiento eficiente de todos los parámetros de θ consistente con un único punto. La transformada de Hough, procede de la siguiente manera. Primero, la parte permisible del espacio de paramentos θ es discretizado en una rejilla, y un acumulador es asignado a cada punto de la cuadricula. Entonces, cada punto emite un voto para todos los contenedores que contiene los parámetros de un modelo coherente con el punto. Los votos de los diferentes puntos de datos son resumidos en los contenedores.

La transformada de Hough trata fácilmente la presencia de múltiples objetos, que a menudo ocurre en la detección de primitivas geométricas en la imagen. Por lo contrario, la transformada de Hough no es adecuada para la detección de modelos de altas dimensiones, debido a que el espacio de parámetros discreatizados necesario para representarlo crece exponencialmente con las dimensiones del modelo. Por lo

tanto, la trasformada de Hough es adecuada, por ejemplo, para la detección de líneas, puntos, círculos (modelos de 2 dimensiones), y mientras las dimensiones aumentan la transformada de Hough es menos adecuado, un ejemplo seria la detección de una transformación afín (6 dimensiones). Sin embargo, se han publicado un número de variantes de la transformada de Hough. Transformada de Hough Aleatorizada donde puede ser localizada la transformada y RANSAC, la transformada de Hough aleatoria genera muestras de la misma manera como lo hace RANSAC. Los parámetros de todos los modelos hipotéticos se almacenan en una estructura de datos que permite la búsqueda de parámetros similares. Cuando algunos de los parámetros aparecieron un número de veces no aleatorias, se detectan los valores típicos (inliers) a esos parámetros como en la etapa de verificación de RANSAC. Los valores típicos (inliers) detectados del modelo se retiran de la muestra y el procedimiento continua. Una cuestión importante a la hora de aplicar el algoritmo de RANSAC descrito anteriormente, es decir cómo clasificar las correspondencias como valores típicos o valores atípicos

Como se puede ver en la Figura 4.2, el calculo de la transformación afín de la imagen junto con las dimensiones del objeto buscado, contribuyen para adecuar las líneas de color verde intenso al objeto de interés, y de esta manera adaptarse a las deformaciones del objeto.

Ahora bien, cabe mencionar y profundizar en el algoritmo utilizando para adecuar las líneas a la deformación del objeto, es el algoritmo de RANSAC, el cual es aplicado para encontrar la transformación afín del objeto de interés, partiendo de la muestra inicial y la metodología antes expuesta. Luego entonces, tendríamos que ajustar una transformación afín, a un conjunto de puntos característicos, sin embargo se tiene un problema inicial, hay algunos puntos de interés incorrectos.

Ahora entonces, a partir de lo antes expuesto y basándose en la similitud en cuanto a la metodología de estos dos métodos, se opta por el algoritmo de RANSAC debido a que, en reiteradas investigaciones [Bay et al., 2008], [Lowe, 2004], hacen mención de su buen funcionamiento siempre y cuando los valores tipicos sean mayores que el 50 %.

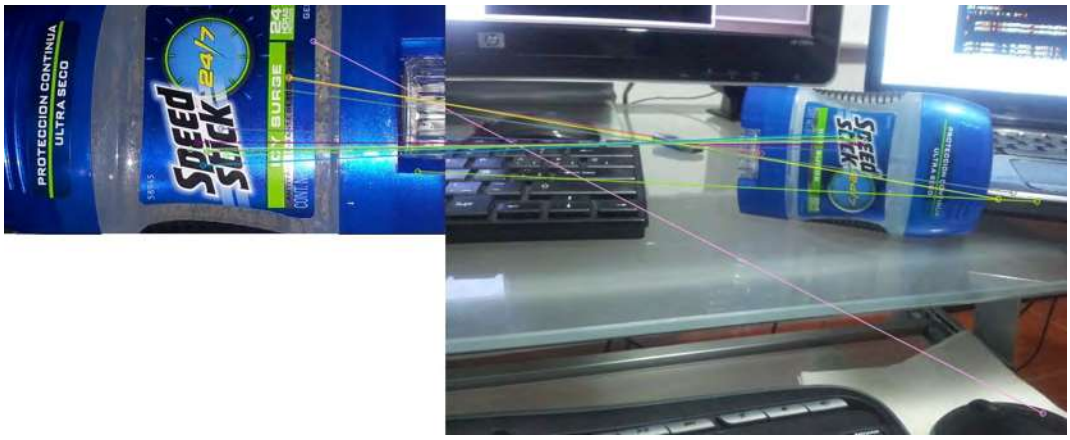


Figura 4.11: Correspondencia de un objeto

Ahora bien, la cuestión de como adaptar una transformación afín a un conjunto de caracterśticas coincidentes como es visible en la Figura 4.5, aunque existe un problema: hay algunas coincidencias incorrectas. Ahora considerando una transformación afín la cual mapea un punto en otro punto, dando entonces T como:

$$T = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \tag{4.10}$$

donde se tiene a su vez un conjunto de n correspondencias que vendrían siendo los puntos coincidentes en la Figura 4.5 como ejemplo.

$$\begin{aligned} [x_1, y_1] &\rightarrow [x'_1, y'_1] \\ [x_2, y_2] &\rightarrow [x'_2, y'_2] \\ [x_3, y_3] &\rightarrow [x'_3, y'_3] \\ [x_4, y_4] &\rightarrow [x'_4, y'_4] \\ [x_5, y_5] &\rightarrow [x'_5, y'_5] \end{aligned} \tag{4.11}$$

Ahora bien considerando entonces una correspondencia de un punto caracterśtico tenemos que:

$$[x_1, y_1] \rightarrow [x'_1, y'_1], \tag{4.12}$$

$$\begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} = \begin{bmatrix} x'_1 \\ y'_1 \\ 1 \end{bmatrix}. \tag{4.13}$$

$$\begin{aligned} ax_1 + by_1 + c &= x'_1 \\ dx_1 + ey_1 + f &= y'_1 \end{aligned}. \tag{4.14}$$

entonces se tienen 2 ecuaciones, 6 incógnitas, de las cuales se necesitan al menos 3 correspondencias, las cuales pueden ajustarse n por mínimos cuadrados. Como se puede ver, es un sistema grande aunque es relativamente fácil de resolver. En otras palabras, encontrar una transformaciń afín T que se mapeen de la imagen uno lo más cerca posible de su correspondencia, una ejemplo es el que se muestra en la Figura 4.6.

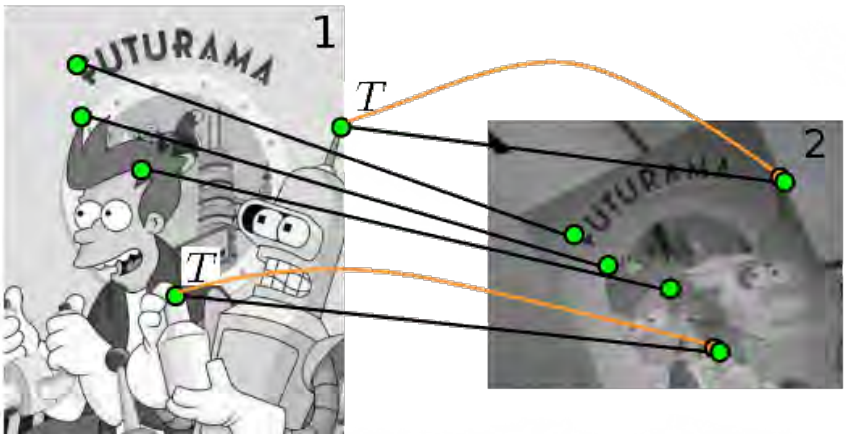


Figura 4.12: Búsqueda del mejor parámetro T

Sin embargo, como fue mostrado en la Figura 4.5, y Figura 4.7, se tienen datos erróneos, es decir correspondencias incorrectas visibles en la figura 4.7 los puntos en rojo no corresponden entre las dos imágenes. Estos valores erróneos conocidos como valores atípicos (outliers) pueden causar problemas con el ajuste de la transformación.

Es entonces donde entra el algoritmo de RANSAC, para buscar la mejor hipótesis, que arroje parámetros que se apeguen a la transformación, siguiendo los paso previos que se mencionaron.

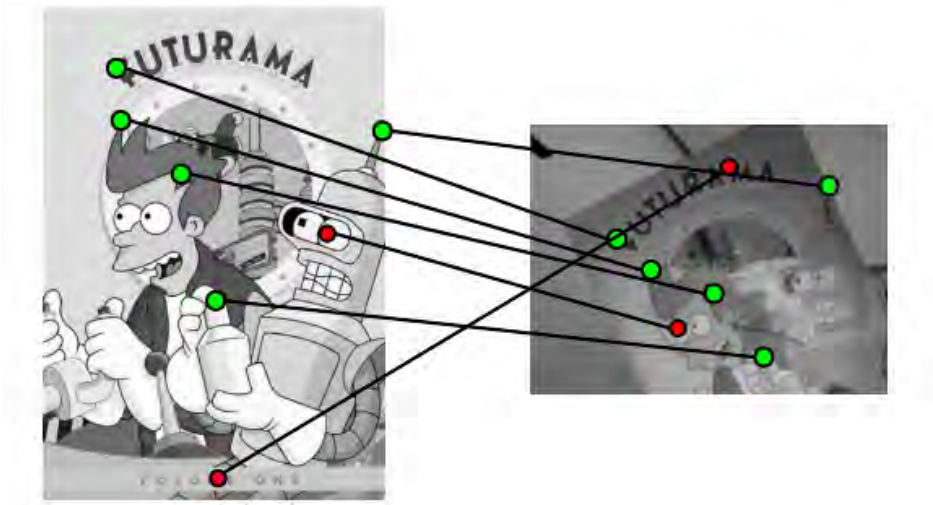


Figura 4.13: Valores atípicos en la correspondencia

Por tanto, el ajuste mediante RANSAC se haría siguiendo los pasos que se muestran en el Algoritmo 3.

Algoritmo 3: Ajuste de los valores de la Transformación afín

```
Seleccionar tres características aleatoriamente;
Resolver para la transformación afín T;
Contar el número de correspondencias que son inlier en T;
if Si T tiene el mayor número de inliers hasta el momento, se guarda then
    while Se repite para una determinada N do
        se regresa la mejor T
    end
end
```

de estos pasos previos, mencionados anteriormente, se desprende la siguiente pregunta ¿Cómo resolver para T dado 3 correspondencias ?, la respuesta seria: dados tres corres-

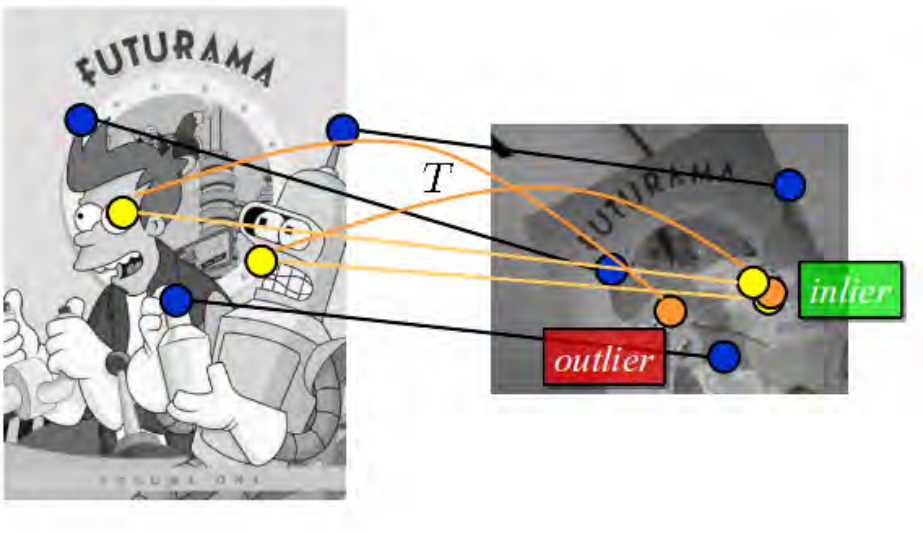


Figura 4.14: Valores típicos (inliers), valores atípicos (outliers) en la búsqueda de la transformación afín

pondencias, se genera un sistema lineal con 6 ecuaciones, como se muestra enseguida.

$$\begin{aligned} ax_1 + by_1 + c &= x'_1 \\ dx_1 + ey_1 + f &= y'_1 \\ ax_2 + by_2 + c &= x'_2 \\ dx_2 + ey_2 + f &= y'_2 \\ ax_3 + by_3 + c &= x'_3 \\ dx_3 + ey_3 + f &= y'_3 \end{aligned} \tag{4.15}$$

Se desprende entonces dos sistemas lineales de 3×3

$$\begin{aligned} ax_1 + by_1 + c &= x'_1 \\ ax_2 + ey_2 + c &= x'_2 \\ ax_3 + by_3 + c &= x'_3 \end{aligned} \tag{4.16}$$

$$\begin{aligned} dx_1 + ey_1 + f &= y'_1 \\ ax_2 + by_2 + f &= y'_2 \\ dx_3 + ey_3 + f &= y'_3 \end{aligned} \tag{4.17}$$

Posteriormente se resuelve el sistema

$$\begin{aligned} ax_1 + by_1 + c &= x'_1 \\ ax_2 + ey_2 + c &= x'_2 \\ ax_3 + by_3 + c &= x'_3 \end{aligned} \tag{4.18}$$

Ahora entonces podemos reescribir la matriz de la siguiente forma

$$\begin{bmatrix} x_1 & y_2 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} x'_1 \\ x'_2 \\ x'_3 \end{bmatrix} \tag{4.19}$$

Se resuelve para encontrar la mejor T, es decir, la mejor transformación afín que se adecue a los inlier y a la deformación del objeto buscado, cuestión que se representa en la siguiente figura.

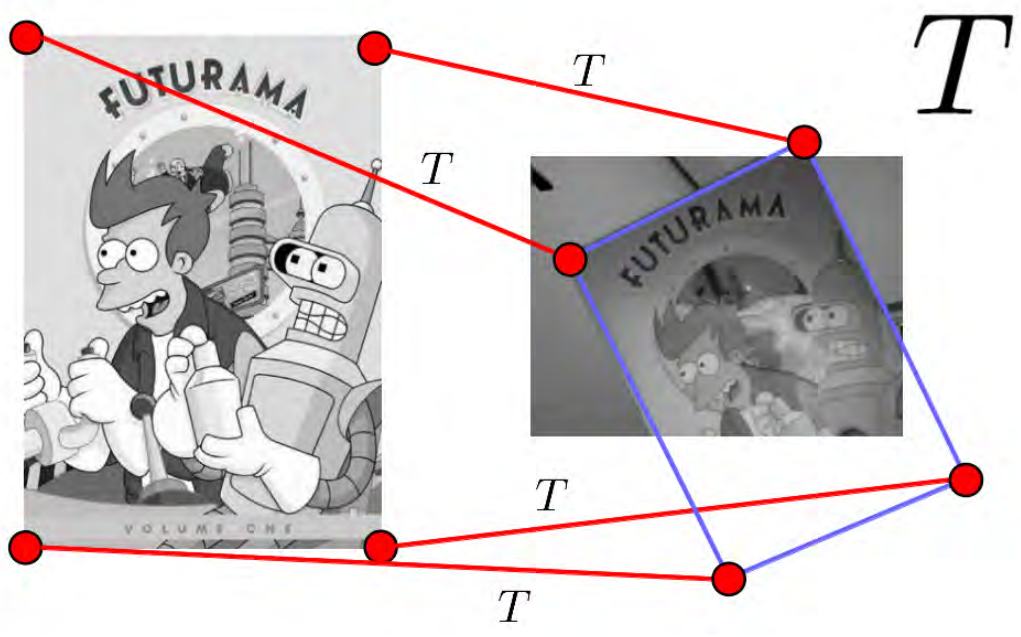


Figura 4.15: Correspondencia de un objeto mejores parámetros de T

4.1.8. Conclusiones

De los experimentos realizados a lo largo de este capitulo se puede deducir, que al igual, como en distintas investigaciones como la de [Bay et al., 2008]. Relacionadas con la eficiencia y velocidad de calculo de los los puntos de interés y descriptores. Se hace mención que el mejor algoritmo en cuanto a estas prestaciones se refiere, es el algoritmo de SURF. Dado que acelera el calculo al dejar de lado la diferencia de Gaussiana y las convoluciones que se necesitan para aplicar los filtro, e igualmente al no disminuir constantemente las imágenes requeridas para obtener los puntos de interés necesarios.

Al aplicar aproximaciones con filtro de caja e imágenes integrales, SURF acelera el proceso de generación de los vectores de características, y puntos de interés. En cuanto a las dimensiones del descriptor, es de mucho menor tamaño el de SURF que el de SIFT. No obstante la idea original de este trabajo de tesis no es verificar cúal de los dos algoritmos es más veloz, sino obtener el mejor algoritmo que se adapte a los cambios del objeto a seguir, aunado a que pueda ser aplicado en tiempo real. Como se ilustró en las pruebas, queda claro que el algoritmo de SURF puede ser el algoritmo que mejor se adapte a los requerimientos del seguimiento en vídeo, a partir de un solo ejemplo, tomado por un click de ratón debido a todo lo expuesto en las secciones anteriores. Asimismo queda claro que el proceso de correspondencias puede alentar el proceso a tiempo real, si no se aplica un método adecuado como el kd-Tree , RANSAC, entre otros algoritmos, detallados a lo largo de este Capítulo.

Este trabajo de tesis trata de adecuar la marca que resalta al objeto buscando mediante la transformación afín, u homografía, como fue visible en la Figura 4.5 donde la marca se adecua al objeto que se esta siguiendo. Sin embargo, es una labor compleja debido a que se esta integrado a la búsqueda en tiempo real, cuestión qué, ciertamente aunque los puntos de interés son mapeados en la secuencia del vídeo o la imagen y siguen siendo característicos, es posible que la figura que intenta resaltar al objeto de interés no se adecúe completamente.

Capítulo 5

Base de Desarrollo

En el presente Capítulo, se aborda la implementación realizada y las partes que lo incorporan, desde la generación de puntos característicos en una imagen y por ende en una secuencia de vídeo, así como la correspondencia entre estos puntos. De igual forma se exponen los resultados obtenidos al ejecutar la implementación realizada.

El sistema se desarrolló en los lenguajes de programación C, C++ incluyendo la librería OpenCV, a su vez, fue ejecutado en una maquina con procesador i7-2600LGA1155 con memoria Ram de 8G y una laptop con procesador i7-2670QM con 6G de Ram ambas bajo el sistema operativo Linux. Se utilizaron los dos lenguajes debido a que la librería en su constante actualización, presenta cambios notables en funciones, métodos, utilizados en el desarrollo del sistema. Razón por lo cual algunos prototipos se tuvieron que modificar para adecuar el sistema a cada mejora de la librería, pues su tipos de datos primitivos, funciones métodos clases, variaban en su funcionamiento. Un claro ejemplo seria las funciones que generan los vectores de características y los puntos de interés **cvExtractSURF**, no con esto se quiere decir que se tomaron todas las funciones existentes en la librería, si no sólo funciones específicas optimizadas para mejorar el rendimiento del sistema.

Para poder comprender el funcionamientos de los algoritmos en cuanto a una imple-

mentación se refiere, se programó una parte del algoritmo de SIFT y SURF utilizando únicamente datos primitivos de OpenCV y C lo cual no fue integrado en la implementación final del seguidor de este trabajo de tesis, debido a que la implementación tiene que trabajar en tiempo real. Ahora entonces, la parte que se programó únicamente con C y algunos datos primitivos de OpenCV, consiste en la extracción de los puntos característico, el refinamiento de tales puntos mediante las diferencias de Gaussiana y su correspondiente espacio escala. Igualmente, en el caso de SURF los filtros de caja con imágenes integrales para verificar que ciertamente se acelera el proceso al utilizar estos filtros de caja e imágenes integrales. Así mismo, con la finalidad de analizar el rendimiento de las funciones programadas mediante estos datos primitivos y los métodos funciones ya establecidas y optimizadas de la librería. Se hizo la siguiente prueba de emparejamiento entre los puntos característicos.

Figura 5.1: En blanco puntos de interés generados con datos primitivos de C y OpenCV, Círculos con variedad de colores, puntos de interés generados por funciones métodos de OpenCV optimizadas

Los puntos en blanco, puntos característicos, son los generados por el programa con

datos primitivos de C y OpenCV, elaborado desde cero basándose específicamente en el algoritmo, sin ninguna función o método establecido por alguna librería. Los círculos de colores, son los puntos característicos generados por la librería por los métodos establecidos como ejemplo **DescriptorExtractor**, **FeatureDetector** etc, entre otros. Dado los resultados arrojados en varias imágenes se optó por utilizar las funciones de OpenCV debido la naturaleza de este trabajo de tesis, que tiene como objetivo primordial hacer todo el proceso en tiempo real, situación que requiera de más tiempo, si el proyecto de investigación se hubiera seguido programando desde cero, programación que no era necesaria porque el fin no era optimizar el algoritmo ni mejorarlo, sino aplicarlo para el seguimiento y detección de objetos, a pesar de las deformaciones rígidas que del objeto surgieran en el seguimiento, basándose en una sola muestra inicial.

5.0.9. Sistema Implementando

El sistema que se implemento, es mostrado en la Figura 5.2 donde cada cuadro del diagrama muestra los pasos necesarios para hacer la detección de algún objeto de interés. Sin embargo es necesario explicar cada parte de este proceso, aplicado a la secuencia de vídeo Taxi de Hamburgo.

Figura 5.2: Vídeo Taxi de Hamburgo

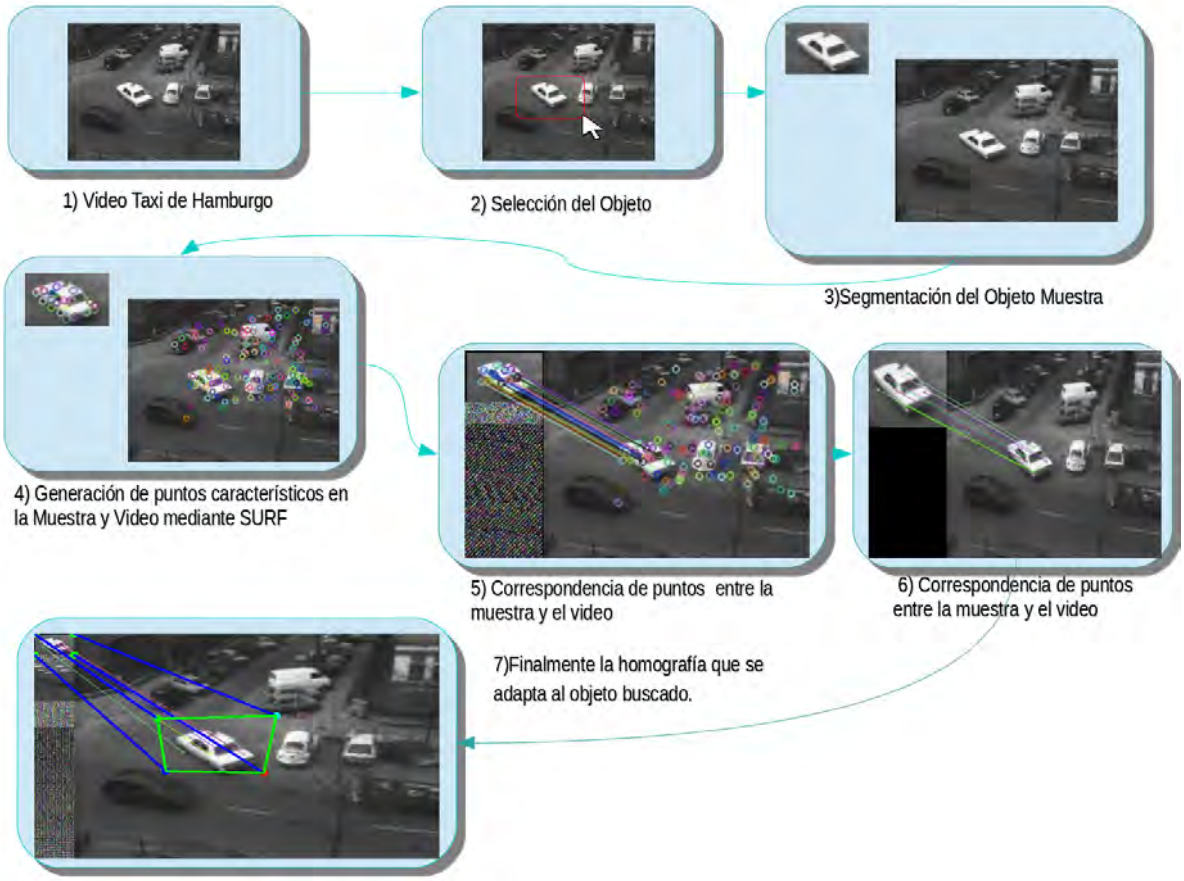


Figura 5.3: Sistema implementado

Ahora bien teniendo cualquier vídeo, la idea es poder seleccionar cualquier objeto interesante en la escena. Sin embargo la escena de este ejemplo como ya se había hecho mención es el Taxi de Hamburgo el cual inicialmente solo era un conjunto de imágenes las cuales se tuvieron que unir para generar mencionado vídeo. No con esto se quieres decir que el ejemplo es ideal, que funciona adecuadamente por características que ayuden a lo que se implemento. Se unieron las imágenes debido al uso continuo en otra exploraciones como lo fuel el flujo óptico. El segundo paso del sistema es tomar con el ratón el objeto de interés de la secuencia de vídeo ver Figura 5.4.

Una vez que se ha seleccionado el objeto de interés, es recortado para posteriormente generar los puntos caracterśticos de la imagen obtenida, a su vez cuadro por

Figura 5.4: Selección del Objeto de Interés

cuadro de la secuencia de vídeo son obtenidos los puntos característicos mediante el algoritmo de SURF el cual fue descrito en el Capítulo 3. Cada punto clave tiene información relevante, como los la posición, la escala de la que proviene, el angulo predominante y en el caso de la correspondencia información sobre el punto que le corresponde entre la imagen de muestra y la secuencia de vídeo o en una imagen cualquiera. Estos puntos caracterśticos iniciales son mostrados en la siguiente Figura 5.5.

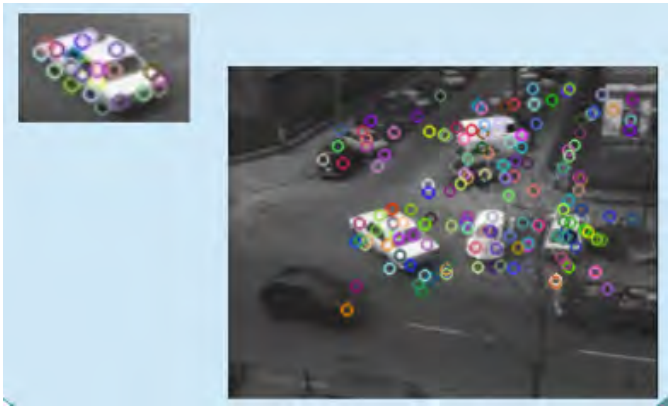


Figura 5.5: Puntos Característicos del tanto del objeto de interés como deun cuadro del vídeo

Posteriormente ahora que se tienen los puntos, el siguiente paso es hacer la correspondencia entre la imagen muestra y los cuadros del vídeo. Es importante mencionar que cada punto tiene información necesaria para poder hacer la correspondencia entre puntos, igualmente este proceso de buscar la correspondencia guarda información sobre la posible posición del punto correspondiente. Por tanto se descartan todos los puntos que estén muy alejados, aunado que el proceso que se hace es la distancia Euclidiana entre los descriptores mientras menor sea la distancia el punto de la muestra corresponde al punto clave del vídeo. En esta parte del proceso se opto por vecinos cercanos al igual que la distancia Euclidiana. Sin embargo tanto en el estado del arte como en la misma librería utilizada OpenCV, recomiendan una modificación al proceso de búsqueda que acelere el proceso como la aproximación a los vecinos prx́imos y hacen mención de incluir la siguiente librería FLANN *Fast Library for Approximate Nearest Neighbors*. Ahora bien una vez que se hace el proceso se dibujan lineas que hagan correspondencia entre la imagen muestra y la secuencia de vídeo como se ejemplifica en la Figura 5.6.

Figura 5.6: Correspondencia entre la imagen muestra en la secuencia de Vídeo

Finalmente la figura que demarca el objeto se implemente mediante el uso del algoritmo RANSAC aunque las investigaciones latentes como las de Lowe en [Lowe, 2004] mencionan la transformada de Hough en algunas de sus variantes. Como se puede ver en la Figura 5.7 la figura trata de apegarse al objeto de interés en algunas

ocasiones como se vera más adelante no se completa la figura debido a los falsos positivos errores en la generación de puntos u en su defecto a la implementación en tiempo real.

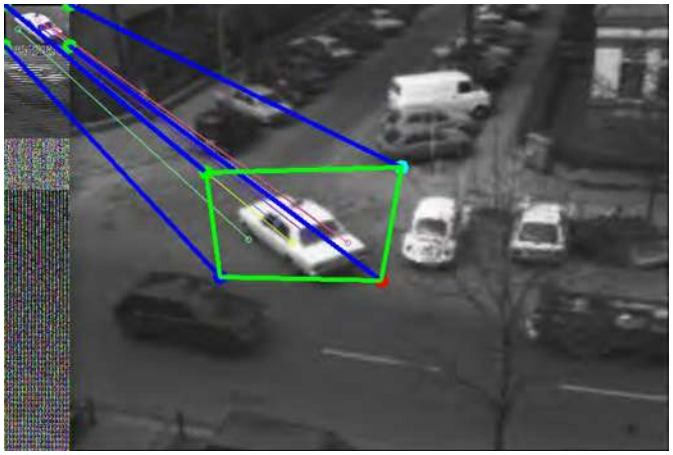


Figura 5.7: Demarcación del objeto de interés en el vídeo mediante el algoritmo de RANSAC

5.0.10. Evaluación del Sistema

En la detección y seguimiento de objetos, el rendimiento y evaluación se reporta en base, a la verificación, identificación de un cierto conjunto de características, definidas por el objeto de interés, es decir, qué tan eficiente y acertada es la búsqueda del objeto deseado. No obstante la idea central de este proyecto de investigación es la detección y seguimiento a partir de una sola muestra, queda claro entonces que lo expuesto en el capitulo 3, da como base el uso del algoritmo de SURF. Debido a los resultados arrojados en cuanto a su velocidad al generar los vectores de características y su robustez ante cambios de iluminación

Correspondencia de Muestra Inicial

con el fin de corroborar lo antes mencionado se hicieron pruebas con una sola muestra, obteniendo resultados notables, dado que la muestra inicial no correspondía

a la forma que el objeto tenia en la imagen de búsqueda pero si los puntos característicos. Es decir, se podía detectar y seguir los puntos siempre y cuando siguieran siendo caracterśticos, frase que se a mencionado a lo largo de todo el desarrollo de este trabajo de tesis.

Por ende, los resultado son muy prometedores con una sola muestra, la Figura 5.1 muestra un ejemplo de lo antes mencionado. Donde es visible que la muestra no corresponde porque se encuentra en sentido inverso y rotado algunos grados lo cual a pesar de estos cambios, el objeto es detectado y marcado, tratándose de apegar lo más posible a la muestra, imagen A, con su correspondiente en la imagen B. Igualmente

Figura 5.8: Detección de objetos, imagen A logo de Ubuntu, imagen B discos collage de Ubuntu.

tanto en el algoritmo de SIFT y SURF se hicieron las pruebas pertinentes, para verificar la correspondencia de puntos característicos partiendo de dos imágenes, en las cuales el objeto de interés es un automóvil, tomado de una secuencia de imágenes llamada el taxi de hamburgo.

Secuencia de vídeo antes descrita en el sistema implementado la cual se puede ver en la figura 3.2. Los puntos característicos que son correspondientes entre las dos imágenes, donde inicialmente se generan los de la imagen A, en este caso el taxi, así mismo, la correspondiente imagen B donde se aprecia el recorrido del taxi, aplicando a cada una los algoritmos de SIFT y SURF, sin embargo en la imagen 3.2 son

mostrados solo los puntos generados por el algoritmo de SURF y su correspondiente búsqueda. Es importante resaltar, que el algoritmo de SURF y SIFT unicamente son utilizados para buscar estos puntos característicos invariantes de cada imagen, una vez que se generan, se implementan la búsqueda de la correspondencia entre cada punto característico de la imagen A en la imagen B, ya sea por vecinos próximos, k-d Tree los cuales fueron mencionados en el capitulo 3. En este caso, el resultado obtenido de la correspondencia entre los puntos son las líneas de colores que representan el mapeo de la imagen A con su correspondiente punto en la imagen B, sin embargo, es evidente que aunque se esta indicando tal correspondencia, faltaría integrar el rectángulo que se apague a la transformación afín que indique que ese conjunto de puntos corresponde a la muestra inicial dada.

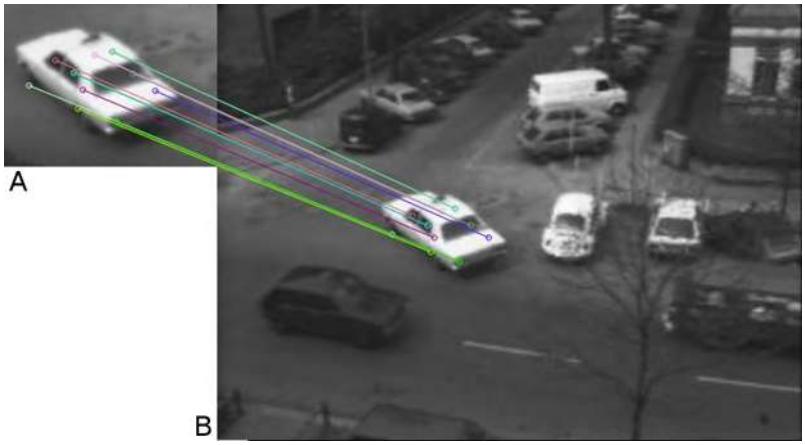


Figura 5.9: Detección de objetos con un solo ejemplo imagen A taxi, imagen B recorrido del taxi de Hamburgo.

Ahora bien, en la siguiente tabla se pueden apreciar los mencionados valores de la imagen anterior, que tienen la mejor correspondencia entre las dos imágenes, donde se indica el número del punto clave en la imagen de muestra A, y su correspondiente punto clave en la imagen de búsqueda B.

Es decir, cada punto tiene asociado un descriptor, valores, que sean mencionado constantemente en este trabajo de tesis, los cuales empatan con sus correspondientes va-

Puntos caracterśticos imagen A	Puntos característicos imagen B
Punto clave 3	Punto Clave 417
Punto clave 7	Punto Clave 179
Punto clave 11	Punto Clave 216
Punto clave 12	Punto Clave 218
Punto clave 16	Punto Clave 179
Punto clave 20	Punto Clave 222
Punto clave 27	Punto Clave 244
Punto clave 29	Punto Clave 535
Punto clave 34	Punto Clave 559
Punto clave 41	Punto Clave 556
Punto clave 43	Punto Clave 532

Tabla 5.1: Correspondencia de puntos entre la imagen de muestra A, e imagen B del Taxi de Hamburgo

lores entre una imagen y otra. Para profundizar y ser un poco más explicito en cuanto a la correspondencia se refiere, en la prueba anterior, se aisló uno de los puntos de esa lista, para demostrar la concordancia entre los puntos característicos. En esta caso el resultado arrojado seria el siguiente:

Figura 5.10: Punto de intéres aislado

Como se puede notar, hay una incongruencia entre los puntos, no obstante, es de resaltar que no siempre se tendrá el mismo orden entre las correspondencias de los puntos caracterśticos o el mismo indice, este puede cambiar conforme se vayan ad-

Puntos caracterśticos imagen A	Puntos característicos imagen B
Punto clave 1	Punto Clave 417

Tabla 5.2: Punto de intéres aislado

quiriendo los puntos claves, sin embargo, lo que no se altera de cada punto clave es la coordenada y el ángulo predominante de cada punto. Dada la naturaleza de éste experimento, el cual esta basado en una imágen fija. Ahora bien, el fin del proyecto consiste en hacer el seguimiento en una secuencia de vídeo, es entonces aquí donde las coordenadas del segundo conjunto de puntos característicos varía, en este caso los puntos generados de los marcos de la secuencia de vídeo, luego entonces, se puede decir que se trata del mismo punto el cual a lo largo del seguimiento no deja de ser característico. Por ende, se puede hacer un seguimiento a partir de una sola muestra, como se puede ver en la siguiente Figura 5.4.

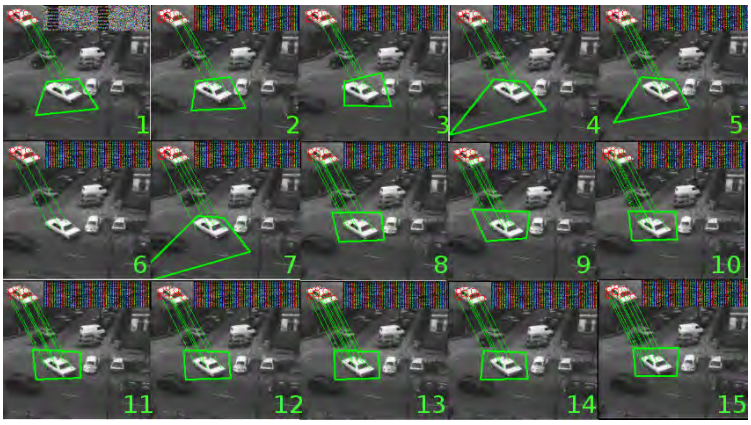


Figura 5.11: Detección de objetos en la secuencia de vídeo Taxi de Hamburgo

Se incluyen las 15 imágenes que componen el vídeo o secuencia del Taxi de Hamburgo de izquierda a derecha numeradas, dado que el trabajo, esta implementado en vídeo y tiempo real, y con ésto se trata de ejemplificar el funcionamiento de la implementación, pero, no es muy notorio el movimiento del taxi que se va alejando del volkswagen que se alcanza apreciar en la escena. Por ende, la siguiente Figura 5.5 contribuye para

poder observar el seguimiento de una manera adecuada, queda claro entonces que en la Figura 5.5 imagen **A** es detectado y generados los puntos caracterśticos de la muestra del taxi, la muestra fue obtenida de una imagen de la secuencia, es notorio que aunque existe una pequeña diferencia entre cuadros, es consistente la marca del objeto que se esta siguiendo en los cuadros subsecuentes. De igual forma es importante mencionar las líneas en color verde las cueles indican la correspondencia entre los puntos de interés de la imagen de muestra.

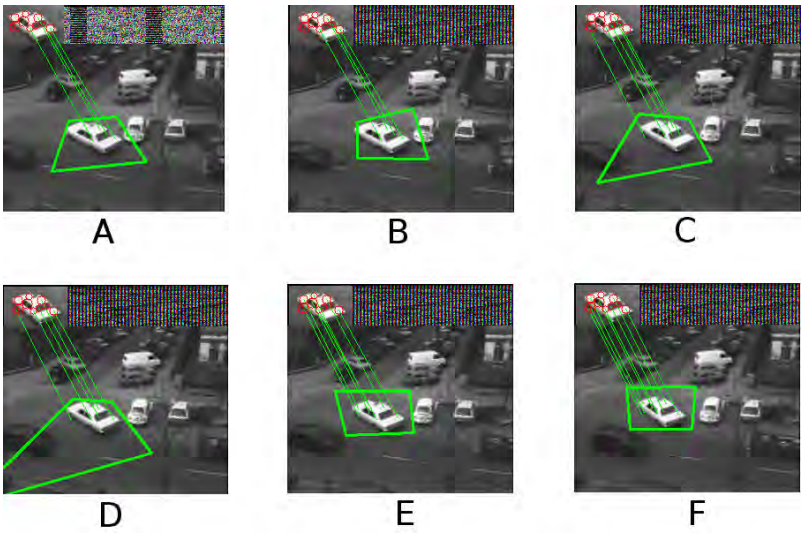


Figura 5.12: Detección de objetos en la secuencia de vídeo Taxi de Hamburgo

Igualmente es de destacar como se trata de apegar en forma de un rectángulo, o figura geométrica amorfa, la detección del conjunto de puntos de intéres en cada marco del vídeo, como es visible en en cada una de las imágenes desde la imagen **A** hasta la imagen **F** aunque en algunos casos, como el de la imagen **D** o igualmente en la Figura 5.4 marco 6, donde no se dibuja ningún cuadro, marca, objeto geométrico que especifique que es lo que se esta siguiendo. Porque la transformación afín que indica los parámetros necesarios para tratar de apegar la figura geométrica que demarque el objetos seguido, es deficiente, debido a la utilización del algoritmo de RANSAC, que no siempre va a generar resultados adecuados que se apeguen a lo que se esta

siguiendo debido a la naturaleza de la implementacion, que es en tiempo real, no en todos los casos es posible adecuar esta líneas rectas al objeto que se esta siguiendo, pero el cual al seguir haciendo la correspondencia de los puntos en cada marco del vídeo es posible retomar la figura geométrica que demarca al objeto de interés, como se muestra en la Figura 5.5 las imágenes **E** y **F**.

Igualmente, en la siguiente imagen Figura 5.6, se muestra una secuencia de vídeo tomada de una cámara web y se puede observar cómo, aunque el objeto de interés sea tapado, en una porción la detección continúa a pesar de la luminosidad los cambios de luz y la poca calidad de los cuadros que contienen al objeto de interés, cuestión que los algoritmos de SURF y SIFT mencionan en sus correspondientes investigaciones, donde se presume la robustez ante estos fenómenos, obstáculos, para la detección y el seguimiento del objeto de interés.

Figura 5.13: Detección de objetos en una secuencia de vídeo

Queda claro entonces que los resultados arrojados, son muy prometedores, debido a que con una sola muestra dada, es posible hacer el seguimiento y la detección, aun cuando el objeto se haya perdido y sobre todo que es posible tomar cualquier objeto

y hacer su seguimiento y detección en tiempo real. No obstante, se ejemplifica más a detalle en la siguiente Figura 5.7 enfocando este ejemplo en la obstrucción parcial del objeto de interés por objetos de la escena mencionado anteriormente, donde en la imagen **A** de la Figura 5.7 se ve claramente un dedo en esta escena que trata de tapar el logotipo de Ubuntu, que es la muestra inicial, el cual se busca en la secuencia de vídeo de la cámara web, en las posteriores imágenes se ve el desplazamiento del mencionado dedo, por tanto es notorio que la muestra correspondiente de la secuencia de vídeo es detectada aun cuando el dedo cubre una parte del objeto a seguir. En algunos se puedo perder el objeto por completo, pero siguiendo la misma filosofía al ya tener los puntos de interés de la muestra inicial, es posible retomar el objeto y demarcarlo en base a su transformación correspondiente, lo cual se nota en las posteriores imágenes de la figura 5.7 de la **F** a la **J** de igual forma como en el ejemplo anterior la correspondencia entre la muestra y la secuencia de vídeo es más robusto que la búsqueda de la transformación afín homografía, por el algoritmo de RANSAC, implementando en este trabajo de tesis.

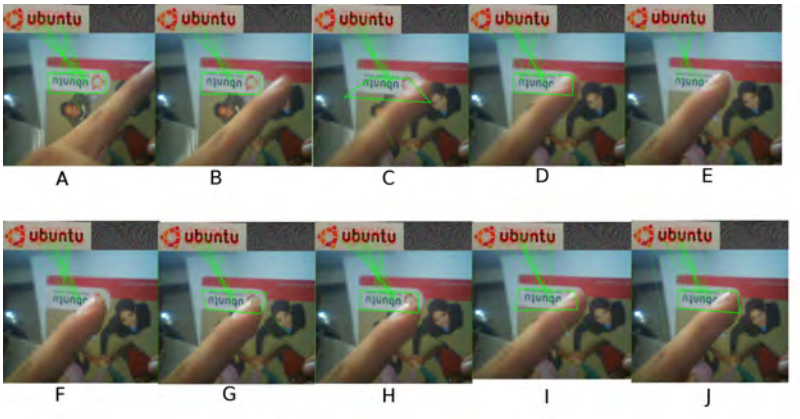


Figura 5.14: Oclusión parcial de objetos en una secuencia de vídeo

5.1. Conclusión

En este Cápitulo se presento la implementación de un seguidor, con una única muestra, se demostró que es posible seguir objetos que difieren de la apariencia de la muestra inicial, en la búsqueda del objeto. Por consiguiente, partiendo de estos experimentos, se puede afirmar que es posible el desarrollo de seguidores que puedan generalizar lo que se quiere seguir. Es decir, tener la posibilidad de decisión en cuanto a lo que se quiere seguir y detectar, y así mismo, poder aprender de ellos en la medida que se vaya modificando la apariencia del mismo, por tanto, tendríamos seguidores detectores, con la posibilidad de aprender en base a una muestra y por ende más robustos a modificaciones en cuanto al aspecto inicial del objeto.

Capítulo 6

Conclusiones

6.1. Conclusiones Generales

La visión por computadora es un campo muy estudiado, pero a su vez muy profundo, con un sin número de posibilidades, de mejoras constantes a los diferentes algoritmos expuestos, a los diferentes métodos de búsquedas de objetos de interés y al universo que conlleva darle visión a un objeto inanimado cómo lo es una computadora o un robot.

En este trabajo de tesis, se ha implementado un sistema de seguimiento de objetos a partir de un solo ejemplo dado por un click de ratón utilizando algunos de los algoritmos expuestos en capítulos anteriores como lo es el algoritmo de SURF. Queda claro que el trabajo de tesis expuesto tiene un sin numero de posibilidades, de mejoras, de aplicaciones, de desarrollo en donde a partir del objeto que es seguido, se pueda aprender, a pesar de las deformaciones rigidas que del objeto emanen. Queda entendido que estos dos algoritmos presentados, como base de un sistema de seguimiento a tiempo real, pueden ser la base para proyectos de investigacion posteriores. De igual forma la diferencia que existe entre SURF y SIFT es palpable y sobre todo notable, pues en este trabajo, la velocidad de calculo de uno, en este caso el de SURF es mucho más provechosa cuando de tiempo real se habla, pues el calculo de los vectores de

características son de mucho menor tamaño y su tiempo de computo es mucho menor que el de SIFT.

Ahora bien, cuando de un seguimiento en un vídeo se habla la implementacíon del algoritmo de SURF para hacer seguimiento y detección, a pesar de tener un solo ejemplo, es bastante buena dado que se pueda seguir lo indicado y retomarlo si el objeto de interés es perdido, y aunque existan deformaciones visibles del objeto, es capas de adecuarse siempre y cuando los vectores de características sigan ciendo característicos. Por ende es algo con lo que se puede trabajar en próximas trabajos los cuales puedan retomar este trabajo de tesis, profundizando u optimizando la búsqueda de correspondencias de los objetos buscados.
Por consiguiente, queda claro de igual forma, que es posible, basar el seguidor, no en un objeto especifico, sino, generalizar lo que se quiere seguir, es decir, de una secuencia de vídeo cualquiera, poder seleccionar cualquier objeto de interés que en ella exista.

6.2. Trabajos Futuros

Se tienen varias líneas de investigación para exploraciones futuras, como se pudo apreciar a lo largo de toda la tesis, el seguimiento se hace a partir del vector de características generado por un solo ejemplo, aplicando el algoritmo de SURF, si bien es algo que ha generado buenos resultados, es sin duda algo en lo que se puede trabajar ampliamente, debido que este algoritmo al generar los vectores de características utiliza imágenes integrales que se pueden implementar siguiendo la investigación de [Viola y Jones, 2004] donde generan un clasificador utilizando imágenes integrales. O en su defecto poder actualizar el descriptor generado de la imagen, o la marca del objeto detectado en la secuencia de video. Por eso en ocaciones se hacia mencion de la tranformacion afin o la homografia del objeto, porque apartir de estos datos puede ser posible generar un clasificador o memoria que guarde los pequeños cambios del descriptor. Igualmente estos cambion seria importantes almacenarlos en una base de

datos, para posteriormente tener registro de lo buscado.

A su vez, otra línea de investigación podria ser, la de tomar como base este trabajo de tesis y aplicarla un AR.Drone. Para hacer el seguimiento desde una perspectiva aérea, e indagar en el campo de la robótica y de los vehículos no tripulados. Para hacer seguimiento y búsquedas de objetos de interés, a partir de un único ejemplo, se retoma el AR.Drone debido a su capacidad de vuelo y su maniobrabilidad en el aire. Existen muchos ejemplos, pruebas donde la versatilidad de vuelo de este robot aéreo es un factor importante para otros experimentos, donde se manejan enjambres para construir prototipos de casas y edificios, así como pequeños prototipos de vehículos no tripulados en el ámbito armamentista.

Referencias

[Arya et al., 1998] Arya, S., Mount, D., Netanyahu, N., Silverman, R., y Wu, A. (1998). An optimal algorithm for approximate nearest neighbor searching fixed dimensions. *Journal of the ACM (JACM)*, 45(6).

[Arya et al., 1994] Arya, S., Mount, D. M., Netanyahu, N. S., Silverman, R., y Wu, A. Y. (1994). An optimal algorithm for approximate nearest neighbor searching in fixed dimensions. In *ACM-SIAM SYMPOSIUM ON DISCRETE ALGORITHMS*, pages páginas 573–582.

[Avidan, 2001] Avidan, S. (2001). Support vector tracking. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. en 2001 IEEE Computer Society Conference on*, volume 1, pages páginas I–184–I–191 vol.1.

[Ballard, 1987] Ballard, D. H. (1987). Readings in computer vision: issues, problems, principles, and paradigms. chapter Generalizing the hough transform to detect arbitrary shapes, pages páginas 714–725. Morgan Kaufmann Publishers Inc., San Francisco, CA, EUA.

[Ballard y Brown, 1982] Ballard, D. H. y Brown, C. M. (1982). *Computer Vision*. Prentice-Hall, Englewood Cliffs, NJ.

[Bay, 2009] Bay, H. (2009). *From wide-baseline point and line correspondences to 3D*. PhD thesis, ETH Zurich.

[Bay et al., 2008] Bay, H., Ess, A., Tuytelaars, T., y Gool, L. J. V. (2008). Speeded-up robust features (surf). *Computer Vision and Image Understanding*, 110(3):páginas, 346–359.

[Beis y Lowe, 1997] Beis, J. S. y Lowe, D. G. (1997). Shape indexing using approximate nearest-neighbour search in high-dimensional spaces. In *CVPR*, pages páginas, 1000–1006. IEEE Computer Society.

[Black y Anandan, 1996] Black, M. J. y Anandan, P. (1996). The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Comput. Vis. Image Underst.*, 63(1):75–104.

[Black y Jepson, 1998] Black, M. J. y Jepson, A. D. (1998). Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. *Int. J. Comput. Vision*, 26(1):63–84.

[Blum y Mitchell, 1998] Blum, A. y Mitchell, T. (1998). Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, COLT' 98, pages 92–100, Nueva York, EUA. ACM.

[Blum y Langley, 1997] Blum, A. L. y Langley, P. (1997). Selection of relevant features and examples in machine learning. *ARTIFICIAL INTELLIGENCE*, 97:245–271.

[Bowyer et al., 2001] Bowyer, K. W., Kranenburg, C., y Dougherty, S. (2001). Edge detector evaluation using empirical roc curves. *Computer Vision and Image Understanding*, 84(1):77–103.

[Brown y Lowe, 2002a] Brown, M. y Lowe, D. (2002a). Invariant features from interest point groups. In *In British Machine Vision Conference*, pages 656–665.

[Brown y Lowe, 2002b] Brown, M. y Lowe, D. G. (2002b). Invariant features from

interest point groups. In Rosin, P. L. y Marshall, A. D., editors, *BMVC*. British Machine Vision Association.

[Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698.

[Cole et al., 2004] Cole, L., Austin, D., y Cole, L. (2004). Visual object recognition using template matching. In *Proceedings of Australian Conference on Robotics and Automation*.

[Collins et al., 2001] Collins, R. T., Lipton, A. J., Fujiyoshi, H., y Kanade, T. (2001). Algorithms for cooperative multisensor surveillance. In *Surveillance, Proceedings of the IEEE*.

[Comaniciu et al., 2002] Comaniciu, D., Meer, P., y Member, S. (2002). Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:603–619.

[Comaniciu et al., 2003a] Comaniciu, D., Ramesh, V., y Meer, P. (2003a). Kernel-based object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(5):páginas, 564–575.

[Comaniciu et al., 2003b] Comaniciu, D., Ramesh, V., y Meer, P. (2003b). Kernel-based object tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 25(5):564–575.

[Cootes et al., 2001] Cootes, T. F., Edwards, G. J., y Taylor, C. J. (2001). Active appearance models. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(6):páginas, 681–685.

[Davison y Murray, 2002] Davison, A. J. y Murray, D. W. (2002). Simultaneous localization and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24:865–880.

[Derpanis et al., 2007] Derpanis, K. G., Leung, E. T. H., y Sizintsev, M. (2007). Fast scale-space feature representations by generalized integral images. In *ICIP (4)*, pages 521–524. IEEE.

[Edwards et al., 1998] Edwards, G. J., Taylor, C. J., y Cootes, T. F. (1998). Interpreting face images using active appearance models. In *Proceedings of the 3rd. International Conference on Face & Gesture Recognition*, FG '98, pages páginas, 300–, Washington, DC, EUA. IEEE Computer Society.

[Eichmann y Dong, 1983] Eichmann, G. y Dong, B. Z. (1983). Coherent optical production of the hough transform. *Appl. Opt.*, 22(6 páginas 830–834).

[Elgammal et al., 2002] Elgammal, A., Duraiswami, R., Harwood, D., Davis, L. S., Duraiswami, R., y Harwood, D. (2002). Background and foreground modeling using nonparametric kernel density for visual surveillance. In *Proceedings of the IEEE*, pages páginas 1151–1163.

[Elgammal et al., 2000] Elgammal, A. M., Harwood, D., y Davis, L. S. (2000). Nonparametric model for background subtraction. In *Proceedings of the 6th European Conference on Computer Vision-Part II*, ECCV '00, pages 751–767, Lodres Reino Unido. Springer-Verlag.

[Fieguth y Terzopoulos, 1 27] Fieguth, P. y Terzopoulos, D. (1997 , páginas, 21-27). Color-based tracking of heads and other mobile objects at video frame rates. In *in Proc. IEEE Conf. on Computer Vision and Pattern Recognition*.

[Freund y Schapire, 1997] Freund, Y. y Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *J. Comput. Syst. Sci.*, 55(1):119–139.

[Friedman et al., 1977] Friedman, J. H., Bentley, J. L., y Finkel, R. A. (1977). An

algorithm for finding best matches in logarithmic expected time. *ACM Trans. Math. Softw.*, 3(3):páginas, 209–226.

[Gao et al., 2000] Gao, X., Boult, T. E., Coetzee, F., y Ramesh, V. (2000). Error analysis of background adaption. pages 503–510.

[Grewe y Kak, 1995] Grewe, L. L. y Kak, A. C. (1995). Interactive learning of a multiple-attribute hash table classifier for fast object recognition. *Computer Vision and Image Understanding*, 61(3):387–416.

[Grimson, 1990] Grimson, W. E. L. (1990). *Object recognition by computer: the role of geometric constraints*. MIT Press, Cambridge, MA, EUA.

[Gwosdek et al., 2011] Gwosdek, P., Grewenig, S., Bruhn, A., y Weickert, J. (2011). Theoretical foundations of gaussian convolution by extended box filtering. In Bruckstein, A. M., ter Haar Romeny, B. M., Bronstein, A. M., y Bronstein, M. M., editors, *SSVM*, volume 6667 of *Lecture Notes in Computer Science*, pages 447–458. Springer.

[Haritaoglu et al., 2000] Haritaoglu, I., Harwood, D., y Davis, L. S. (2000). W4: Real-time surveillance of people and their activities. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:809–830.

[Harris y Stephens, 1988a] Harris, C. y Stephens, M. (1988a). A combined corner and edge detector. In *In Proc. of Fourth Alvey Vision Conference*, pages 147–151.

[Harris y Stephens, 1988b] Harris, C. y Stephens, M. (1988b). *A combined corner and edge detector*, volume 15, pages 147–151. Manchester, UK.

[Horn y Schunck, 1981] Horn, B. K. P. y Schunck, B. G. (1981). Determining optical flow. *ARTIFICAL INTELLIGENCE*, 17:185–203.

[Hough, 1962] Hough, P. (1962). Method and Means for Recognizing Complex Patterns. U.S. Patent 3.069.654.

[Irani y Anandan, 1998] Irani, M. y Anandan, P. (1998). Video indexing based on mosaic representations.

[Isard y Blake, 1998] Isard, M. y Blake, A. (1998). Condensation - conditional density propagation for visual tracking. *International Journal of Computer Vision*, 29:5–28.

[Jain y Nagel, 1979] Jain, R. y Nagel, H.-H. (1979). On the analysis of accumulative difference pictures from image sequences of real world scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):206–214.

[Jensfelt et al., 2006] Jensfelt, P., Kragic, D., Folkesson, J., y Björkman, M. (2006). A framework for vision based bearing only 3D SLAM. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA'06)*, Orlando, FL.

[Kalal et al., 2010] Kalal, Z., Mikolajczyk, K., y Matas, J. (2010). Face-tld: Tracking-learning-detection applied to faces. In *ICIP*, pages 3789–3792. IEEE.

[Kanade et al., 1998] Kanade, T., Collins, R., Lipton, A., Burt, P., y Wixson, L. (1998). Advances in cooperative multi-sensor video surveillance. In *Darpa Image Understanding Workshop*, pages 3–24. Morgan Kaufmann.

[Kockelkorn et al., 2003] Kockelkorn, M., Lüneburg, A., y Scheffer, T. (2003). Using transduction and multi-view learning to answer emails. In Lavrac, N., Gamberger, D., Blockeel, H., y Todorovski, L., editors, *PKDD*, volume 2838 of *Lecture Notes in Computer Science*, pages 266–277. Springer.

[Kuhn y Yaw, 1955] Kuhn, H. W. y Yaw, B. (1955). The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97.

[Levin et al., 2003] Levin, A., Viola, P. A., y Freund, Y. (2003). Unsupervised improvement of visual detectors using co-training. In *ICCV*, pages 626–633. IEEE Computer Society.

[Li y Leung, 2002] Li, L. y Leung, M. K. H. (2002). Integrating intensity and texture differences for robust change detection. *IEEE Transactions on Image Processing*, 11(2):105–112.

[Lindeberg, 1990] Lindeberg, T. (1990). Scale-space for discrete signals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12:páginas, 234–254.

[Lindeberg, 1994] Lindeberg, T. (1994). Scale-space theory: A basic tool for analysing structures at different scales. *Journal of Applied Statistics*, pages páginas, 224–270.

[Lindeberg y Bretzner, 2003] Lindeberg, T. y Bretzner, L. (2003). Real-time scale selection in hybrid multi-scale representations. pages páginas, 148–163. Springer Verlag.

[Lowe, 1991] Lowe, D. G. (1991). Fitting parameterized three-dimensional models to images. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 13:441–450.

[Lowe, 1999] Lowe, D. G. (1999). Object recognition from local scale-invariant features. In *ICCV*, pages 1150–1157.

[Lowe, 2001] Lowe, D. G. (2001). Local feature view clustering for 3d object recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 682–688. Springer.

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):páginas, 91–110.

[Lucas y Kanade, 1981] Lucas, B. D. y Kanade, T. (1981). An iterative image registration technique with an application to stereo vision (ijcai). In *Proceedings of the 7th International Joint Conference on Artificial Intelligence (IJCAI '81)*, pages 674–679.

[Luong y Faugeras, 1995] Luong, Q.-T. y Faugeras, O. (1995). The fundamental matrix: theory, algorithms, and stability analysis. *International Journal of Computer Vision*, 17:43–75.

[Mikolajczyk y Schmid, 2001] Mikolajczyk, K. y Schmid, C. (2001). Indexing based on scale invariant interest points. In *Proceedings of the 8th International Conference on Computer Vision, Vancouver, Canada*, pages 525–531.

[Mikolajczyk y Schmid, 2004] Mikolajczyk, K. y Schmid, C. (2004). Scale & affine invariant interest point detectors. *Int. J. Comput. Vision*, 60(1):63–86.

[Moghaddam y Pentland, 1997] Moghaddam, B. y Pentland, A. (1997). Probabilistic visual learning for object representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7):páginas, 696–710.

[Monnet et al., 2003] Monnet, A., Mittal, A., Paragios, N., y Ramesh, V. (2003). Background modeling and subtraction of dynamic scenes. pages 1305–1312.

[Moore, 1965] Moore, G. E. (1965). Cramming more components onto integrated circuits. *Electronics*, 38(8).

[Moravec, 1979] Moravec, H. (1979). Visual mapping by a robot rover. In *Proceedings of the 6th International Joint Conference on Artificial Intelligence*, pages 599–601.

[Oliver et al., 2000] Oliver, N. M., Rosario, B., y Pentland, A. P. (2000). A bayesian computer vision system for modeling human interactions. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22(8):831–843.

[Papageorgiou et al., 1998] Papageorgiou, C. P., Oren, M., y Poggio, T. (1998). A general framework for object detection. In *Proceedings of the Sixth International Conference on Computer Vision*, ICCV '98, pages 555–, Washington, DC, EUA. IEEE Computer Society.

[Park, 2004] Park, S. (2004). A hierarchical bayesian network for event recognition of human actions and interactions. In *Association For Computing Machinery Multimedia Systems Journal*, pages 164–179.

[Paschos, 2001] Paschos, G. (2001). Perceptually uniform color spaces for color texture analysis: an empirical evaluation. *Trans. Img. Proc.*, 10(6):932–937.

[Rittscher et al., 2000] Rittscher, J., Kato, J., Joga, S., y Blake, A. (2000). A probabilistic background model for tracking. In Vernon, D., editor, *Computer Vision - ECCV 2000, 6th European Conference on Computer Vision, Dublin, Ireland, June 26 - July 1, 2000, Proceedings, Part II*, volume 1843 of *Lecture Notes in Computer Science*, pages 336–350. Springer.

[Rowe y Blake, 1996] Rowe, S. y Blake, A. (1996). Statistical mosaics for tracking. *Image and Vision Computing*, 14:549–564.

[Rowley et al., 1998] Rowley, H., Baluja, S., y Kanade, T. (1998). Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38.

[Schweitzer et al., 2002] Schweitzer, H., Bell, J. W., y Wu, F. (2002). Very fast template matching. In *Proceedings of the 7th European Conference on Computer Vision-Part IV*, ECCV '02, pages 358–372, London, UK, UK. Springer-Verlag.

[Serby et al., 2004] Serby, D., Koller-Meier, E., y Gool, L. V. (2004). Probabilistic object tracking using multiple features. In *Proceedings of the Pattern Recognition, 17th International Conference on (ICPR'04) Volume 2 - Volume 02*, ICPR '04, pages páginas, 184–187, Washington, DC, EUA. IEEE Computer Society.

[Shafique y Shah, 2003] Shafique, K. y Shah, M. (2003). A non-iterative greedy algorithm for multi-frame point correspondence. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 51–65.

[Shi y Malik, 2000] Shi, J. y Malik, J. (2000). Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

[Shi y Tomasi, 1994] Shi, J. y Tomasi, C. (1994). Good features to track. In *1994 IEEE Conference on Computer Vision and Pattern Recognition (CVPR'94)*, pages 593 – 600.

[Shlens, 2005a] Shlens, J. (2005a). http://www.snl.salk.edu/ shlens/pub/notes/pca.pdf.

[Shlens, 2005b] Shlens, J. (2005b). A tutorial on principal component analysis. In *Systems Neurobiology Laboratory, Salk Institute for Biological Studies*.

[Simard et al., 1998] Simard, P., Bottou, L., Haffner, P., y LeCun, Y. (1998). Boxlets: A fast convolution algorithm for signal processing and neural networks. In Kearns, M. J., Solla, S. A., y Cohn, D. A., editors, *NIPS*, pages páginas, 571–577. The MIT Press.

[Stauffer y Grimson, 2000] Stauffer, C. y Grimson, W. E. L. (2000). Learning patterns of activity using real-time tracking. *IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE*, 22:747–757.

[Stenger et al., 2001] Stenger, B., Ramesh, V., Paragios, N., F.Coetzee, y Buhmann, J. (2001). Topology free hidden markov models: Application to background modeling. In *In IEEE International Conference on Computer Vision*, pages 294–301.

[Szeliski y Coughlan, 1997] Szeliski, R. y Coughlan, J. M. (1997). Spline-based image registration. *International Journal of Computer Vision*, 22(3):199–218.

[Tieu y Viola, 2004] Tieu, K. y Viola, P. (2004). Boosting image retrieval. *Int. J. Comput. Vision*, 56(1-2):páginas 17–36.

[Toyama et al., 1999] Toyama, K., Krumm, J., Brumitt, B., y Meyers, B. (1999). Wallflower: principles and practice of background maintenance.

[Veenman et al., 2001] Veenman, C. J., Reinders, M., y Backer, E. (2001). Resolving motion correspondence for densely moving points. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23:páginas, 54–72.

[Viola et al., 2005] Viola, P., Jones, M. J., y Snow, D. (2005). Detecting pedestrians using patterns of motion and appearance. *Int. J. Comput. Vision*, 63(2):153–161.

[Viola y Jones, 2004] Viola, P. A. y Jones, M. J. (2004). Robust real-time face detection. *International Journal of Computer Vision*, 57(2):137–154.

[Wiener, 1964] Wiener, N. (1964). *Extrapolation, Interpolation, and Smoothing of Stationary Time Series*. The MIT Press.

[Wren et al., 1997] Wren, C., Azarbayejani, A., Darrell, T., y Pentland, A. (1997). Pfinder: Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19:780–785.

[Yilmaz et al., 2006a] Yilmaz, A., Javed, O., y Shah, M. (2006a). Object tracking: A survey. *ACM Comput. Surv.*, 38(4).

[Yilmaz et al., 2006b] Yilmaz, A., Javed, O., y Shah, M. (2006b). Object tracking: A survey. *ACM Comput. Surv.*, 38(4).

[Yilmaz et al., 2004] Yilmaz, A., Li, X., y Shah, M. (2004). Contour based object tracking with occlusion handling in video acquired using mobile cameras. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26:páginas, 1531–1536.

[Zhang y Lu, 2001] Zhang, D. y Lu, G. (2001). Segmentation of moving objects in image sequence: A review. In *Circuits, Systems and Signal Process*, pages 143–183.

[Zhong y Sclaroff, 2003] Zhong, J. y Sclaroff, S. (2003). Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, pages 44–, Washington, DC, EUA. IEEE Computer Society.

[Zhu y Yuille, 1996] Zhu, S. C. y Yuille, A. (1996). Region competition: Unifying snakes, region growing, and bayes/mdl for multi-band image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:páginas, 884–900.