Reconocimiento de objetos deformables en condiciones variables usando modelos estadísticos de forma y apariencia

por

Pável Herrera Domínguez

Tesis sometida como requisito parcial para obtener el grado de

Maestro en Ciencias en el Área de Ciencias Computacionales en el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Supervisada por:

Dr. Leopoldo Altamirano Robles, INAOE

Reconocimiento de objetos deformables en condiciones variables usando modelos estadísticos de forma y apariencia.

por

Pável Herrera Domínguez

Tesis sometida para
obtener el grado de Maestro en Ciencias

supervisada por
Dr. Leopoldo Altamirano Robles

Agradecimientos

- Al seminario de estudiantes y sus asistentes por intentar entender lo que los demás hacen.
- Al seminario de visión por los comentarios y sugerencias.
- A todos aquellos que aportaron al contenido de esta tesis con comentarios, sugerencias, correcciones o simplemente ponían atención.
- A Google por su apoyo incondicional a la búsqueda de conocimiento.
- 2 836 35 2667 33 64 8432 9 64 46774722466.

G.C.T. oooo0'

Resumen

En esta tesis se presentan herramientas matemáticas y computacionales usadas para resolver el problema de reconocer objetos deformables en condiciones no controladas, en particular bajo condiciones de ambientes de iluminación variable y condiciones de oclusión parcial del objeto de interés. Usando dichas herramientas se proponen y elaboran algoritmos que permiten mejorar y extender métodos existentes, constituyendo éstas extensiones las aportaciones principales de esta tesis. Primeramente se propone estructurar los elementos de la representación del objeto formando partes del mismo, como resultado se obtienen relaciones entre los elementos; ya sea topológicas en el caso de un grafo, o espaciales en el caso de cúmulos de los elementos de la representación. Una vez que se tienen estructurados dicho los elementos se proponen varios algoritmos de inferencia, dos para el caso de oclusión parcial del objeto y uno para el caso de iluminación variable. Para el caso de oclusión se propone almacenar los cúmulos de elementos en una estructura jerárquica, que representan partes del objeto. Esta estructura es usada para unir los elementos de la representación e ir recuperando información parcial del objeto. Además, se usan las relaciones topológicas como sistema de vecindades para construir un campo aleatorio de Markov, utilizando este campo para inferir los parámetros de los elementos del modelo que se cree que no están presentes debido a la oclusión. Por otro lado, en el caso de la iluminación variable se usan las relaciones topológicas realizando una búsqueda sobre los elementos de la representación con el fin de recuperar la mayor parte del modelo en la imagen. Se realizaron experimentos para validar cada algoritmo. Para el caso de oclusión parcial se hicieron experimentos con las clases *faces* y *car-side* de la base de datos Caltech 101. Con la finalidad de evaluar el comportamiento de los algoritmos, se modificaron las imágenes de la base de datos mencionada agregando oclusión sintética a las imágenes. Respecto al algoritmo propuesto para iluminación variable se experimentó con las imágenes de rostros de la base de datos Yale B. En todos los experimentos con los algoritmos propuestos se obtuvieron mejores tasas de detección promedio; alcanzando hasta 20 puntos porcentuales más comparando con los resultados que se alcanzan al aplicar el algoritmo base *SumMaxMaps*.

Abstract

This thesis contains mathematical and computational tools to solve the problem of recognize deformable objects in uncontrolled environments, in variable lighting conditions and object with partial occlusion. With those tools is possible to propose and implement algorithms that allow to improve existing methods, these improvements are the main contributions of this thesis. First, we propose to partition the object in parts formed with the elements of the object representation, as a result we have a graph or a set of clusters. Thus, with the structured representation of the object we propose several algorithms to recognize the object, two for the partial occlusion case, and one algorithm for the variable lighting environments. For occlusion, the first method consists in arrange the clusters in a hierarchy; later the algorithm uses the hierarchy to merge the parts and retrieve partial information of the object. The second method uses the graph to set a neighborhood system to build a Markov random field; after, this field is employed to infer the model parameters that are assumed to be hidden by the occlusion. Additionally, the proposed algorithm for recognizing the object in variable lighting conditions uses the graph to perform a search over the elements assumed to be in the image in order to get the rest of the elements not found because of the lighting conditions. Experiments were performed in order to validate the proposed algorithms. For the occlusion case the experiments were done over *faces* and *car-side* classes from Caltech database. With the objective of investigate the behavior of the algorithms, images from the database were modified, adding synthetic occlusion. The experiment for the algorithm that considers variable lighting environments was performed over images taken from Yale B database. The results for all the performed experiments show improvements up to 20 % points in the detection rates, comparing them with the results obtained by using *SumMaxMaps*, the original algorithm for the used representation.

Índice general

Índice de figuras

Índice de Tablas

Índice de Algoritmos

Lista de Símbolos

B	Conjunto de filtros de Gabor
B_m	Conjunto de filtros de Gabor deformados para la imagen I_m
B_i	i-ésimo filtro de Gabor, con sus parámetros (x_i, y_i, θ_i)
$< I, B_i >$	Respuesta al filtro B_i de la imagen I
$\hat{H}$	Subconjunto de elementos B_i *presentes* en una posición de la imagen

Capítulo 1

Introducción

En esta parte del documento se describirá el problema a resolver en esta tesis, sus objetivos y la estructura general su contenido.

1.1. Motivación

Uno de los retos en visión por computadora es el reconocimiento de objetos dentro de una imagen tomada en cualquier ambiente o estado. Reconocer un objeto en una imagen consiste en obtener información del estado del objeto; esta información puede ser, por ejemplo, la posición del objeto en la imagen, si éste está o no en la imagen, o quién está en la imagen en caso de ser personas, entre otras. Tener la posibilidad de reconocer objetos en ambientes no controlados como lo son ambientes donde la iluminación es variable y/o existe oclusión parcial, puede ser útil en aplicaciones de robótica, vigilancia, identificación, y otros. Todas estas aplicaciones han llevado al desarrollo de diversas tecnologías en el área de reconocimiento.

Son varias las dificultades en la tarea de reconocer un objeto, por ejemplo, si se quiere reconocer un automóvil dentro de una imagen, como la que se muestra en la figura 1.1(a), donde se observa que existen elementos externos que ocluyen parcialmente el automóvil, es necesario un método que permita decidir que partes del objeto están; con el fin de obtener una imagen como la que se muestra en la figura 1.1(b), que representa haber reconocido un automóvil ocluido en la figura 1.1(a).

(a) (b)

FIGURA 1.1: a) Imagen de un objeto ocluido. b) Imagen del objeto recuperado

1.1.1. Problemática

Para reconocer un objeto en una imagen, en general existen varias situaciones que complican el reconocimiento, algunas de ellas son:

- Oclusión, ya sea por el mismo objeto o por objetos externos.
- Diferentes estados de iluminación en el ambiente, donde se encuentra el objeto.
- Cambios de aspecto del objeto, en su color y en su textura.
- Las diferentes poses en las que el objeto puede aparecer.
- Las deformaciones que puede presentar el objeto en la imagen, ya sea provocadas por la cámara o por el objeto en sí mismo.
- La asignación de etiquetas a imágenes de ejemplo del objeto, para construir un modelo del objeto, el cual será usado en el reconocimiento, es tedioso cuando el número de ejemplos crece.

1.1.2. Antecedentes

En los últimos años se han desarrollado algoritmos para tratar de resolver alguna de las problemáticas mencionas en la subsección anterior.
Uno de las principales contribuciones para resolver este tipo de problemas fue publicada por Cootes et al., quienes propusieron ajustar una plantilla deformable a

la imagen, minimizando una medida de error entre el ajuste y la imagen [CET+01]. Este trabajo fue extendido por Baker el at., quienes incluyeron en su método los efectos provocados por oclusión [BMX+04]. Por otra parte en el trabajo de Ayala-Raggi et al., se tomaron en cuenta los cambios de iluminación sobre el objeto [ARARCE09]. Por otro lado, Zhu et al. desarrollaron una representación basada en filtros de Gabor perturbables, ésta permite representar objetos deformables y poder reconocerlos; además, esta representación tiene la ventaja de que puede ser construida con un bajo nivel de supervisión [WSGZ09]. Este último trabajo se ha usado para resolver el problema de oclusión parcial en el objeto, mencionado en la problemática, usando gramáticas visuales [WZ10]. Existen otros trabajos que han permitido avances en los puntos mencionados algunos de los cuales se revisarán con cierto detalle en el capítulo 3.

Los logros obtenidos en esta área aún no resuelven completamente el problema, aunque se han logrado avances significativos. Esto último motivó la propuesta de esta tesis, que consiste en plantear algoritmos que favorezcan el reconocimiento de objetos en condiciones de ambiente no controladas, que además sea independiente de algún objeto en particular, y que aproveche los avances ya existentes en la resolución a este tipo de problemas.

1.2. Enunciado del problema

Desde el punto de vista matemático se representa una imagen por la letra I. Suponiendo que se tiene un conjunto de imágenes $\{I_1, ..., I_n\}$, que representan ejemplos del objeto O, tomadas en condiciones controladas, y una imagen I' del objeto O tomada en condiciones de oclusión parcial o diferentes condiciones de iluminación; el problema a resolver en este trabajo es el siguiente: reconocer automáticamente el objeto **O** en la imagen I'.

1.3. Preguntas de investigación

Las preguntas más importantes que se quieren contestar en este trabajo son:

- ¿Es posible reconocer un objeto parcialmente oculto?

- En caso de ser posible el punto anterior, ¿es posible saber que partes del objeto *no están* en una posición en la imagen?.
- ¿Es posible reconocer un objeto en una imagen tomada en condiciones de iluminación variable?
- ¿Qué tipo de modelo es necesario usar para contesar las preguntas anteriores?
- ¿Qué grado de supervisión es necesario para poder construir un modelo que represente al objeto y que ayude a contestar las preguntas anteriores?

1.4. Objetivo de la tesis

El objetivo principal de la tesis es tener un método de aprendizaje que sea capaz de construir un modelo que permita reconocer un objeto aún si éste está parcialmente ocluido o si se encuentra en condiciones de iluminación variable. Asimismo elaborar algoritmos de inferencia sobre el modelo que consideren el problema de la oclusión y cambios de iluminación, para mejorar o mantener el reconocimiento del objeto.

Objetivos específicos

- Definir un método para aprender, con un bajo nivel de supervisión, modelos estadísticos de forma y apariencia a partir de un conjunto de imágenes de un objeto.
- Definir un método de ajuste del modelo aprendido a una imagen nueva, considerando que el objeto puede estar parcialmente ocluido o si se encuentra en un ambiente donde las condiciones de iluminación son variables.
- Definir un conjunto de imágenes prueba para medir el desempeño de los algoritmos propuestos.

1.5. Solución propuesta

Se propone tomar los elementos de la representación *active basis* y estructurarlos en partes del objeto, para poder usar los elementos estructurados se modificará el

algoritmo de inferencia de la representación *active basis* de varias formas. Para el caso de oclusión se proponen dos variantes: la primera busca unir las partes del objeto usando una jerarquía, la segunda es usar una de las estructuras aprendidas como sistema de vecindades en un campo aleatorio de Markov. Por otro lado, para el caso del reconocimiento en condiciones de iluminación variable, se propone un método de inferencia que usa histéresis para encontrar los elementos del modelo en la imagen, considerando que no se encuentran en ella debido a los cambios de iluminación.

1.6. Experimentos y Resultados

Para medir el rendimiento de las modificaciones propuestas para el caso de oclusión parcial se modificaron las imágenes de la base de datos CalTech 101 clases *Faces* y *car-sise* [FFFP06], agregando oclusión parcial sintética a diferentes niveles, con el objetivo de medir cómo se comporta el algoritmo cuando éste es ocluido. En cuanto a los experimentos para ver el comportamiento de las modificaciones para el caso de iluminación parcial se usaron imágenes de la base de datos de *The Yale Face Database B* [GBK00], que contiene imágenes en condiciones de iluminación variable. En todos los casos se midieron las tasas de detección de cada algoritmo con un conjunto de prueba, y en cada experimento se reportaron mejoras respecto a las tasas de detección respecto al algoritmo original usado para la representación *active basis*.

1.7. Contribuciones

Las principales aportaciones de esta tesis están en las dos etapas de reconocimiento de objetos: aprendizaje e inferencia.

1. En la etapa de aprendizaje la contribución es la elaboración de un método para construir un modelo a partir de imágenes de ejemplo del mismo objeto, específicamente como estructurar los elementos de la representación *active basis* en *partes* del objeto. Para lo que se proponen dos enfoques, tener el modelo estructurado como unión de cúmulos y como un grafo. Ninguno de estos enfoques requiere supervisión extra más allá de tener el objeto

delimitado por un recuadro, ni tampoco información acerca del tipo de objeto que se está modelando.

2. En cuanto a la etapa de inferencia se proponen diferentes métodos para usar las estructuras compuestas de elementos de la representación *active basis*, para reconocer parcialmente un objeto en condiciones de oclusión e iluminación variable. Estos métodos son: un método jerárquico basado en partes para reconocer el objeto parcialmente, otro basado en campos aleatorios de Markov para completar la información de los elementos del modelo que se asume que fueron ocluidos, y un método para inferir en imágenes con iluminación variable que usa histéresis para completar los elementos del modelo que fueron afectados por los cambios de iluminación. Siendo estos algoritmos parte de las contribuciones principales de este trabajo.

1.8. Estructura del documento

A continuación se describe la distribución del material contenido en este documento:

- En el capítulo 2 se presentan algunos elementos básicos de estadística, probabilidad, y reconocimiento de objetos. En la parte final de este capítulo se detallan algunos elementos de la teoría y los algoritmos usados para la representación *active basis*.
- En el capítulo 3 se dará una breve discusión acerca del estado actual de la tecnología en reconocimiento de objetos, con la finalidad de justificar las decisiones tomadas en el desarrollo de la tesis.
- El capítulo 4 contiene las contribuciones principales de este trabajo, las cuales son los algoritmos propuestos para la estructuración de los modelos *active basis* y como aplicarlos en una etapa de inferencia para reconocer objetos en condiciones de oclusión e iluminación variable.
- En capítulo 5 se presentan los resultados al aplicar los algoritmos elaborados en este trabajo en imágenes obtenidas de algunas bases de datos adaptadas; para medir el rendimiento de los mismos.

- En el capítulo 6 se proponen otras posibles aplicaciones usando algunos de los algoritmos contenidos en este trabajo.
- En el capítulo 7 se dan las conclusiones del trabajo, algunas ideas a seguir para posibles extensiones, mejoras y aplicaciones de lo presentado en este trabajo.
- Finalmente se incluye la bibliografía consultada y dos apéndices que contienen las tablas de resultados de los experimentos y una posible extensión a esta tesis.

Capítulo 2

Marco Teórico

En este capítulo se presentan los conceptos necesarios que se usan a lo largo del documento, en particular conceptos básicos de probabilidad y conceptos relacionados con el reconocimiento de objetos. En el capítulo 3 se revisan los trabajos relacionados con este tema.
También se presentan en este capítulo los detalles teóricos sobre *active basis*, y se explica que son los filtros de Gabor. *Active basis* es la representación que se usa en este trabajo, y como se verá más adelante esta representación está compuesta por filtros de Gabor.

2.1. Conceptos preliminares

Los conceptos que se presentan a continuación serán usados para desarrollar parte de la teoría y los algoritmos de la representación *active basis*, en su mayoría son conceptos básicos de estadística y probabilidad, tratando de que el documento sea auto contenido.

2.1.1. Variable aleatoria

Dado un experimento aleatorio con un espacio de muestra $\mathcal{S}$. Una función X, la cual asigna a cada elemento $c \in \mathcal{S}$ uno y solo un número real $X(c) = x$, es llamada variable aleatoria. El espacio de X es el conjunto de los números reales $\mathcal{A} = \{x | x = X(c), c \in \mathcal{S}\}$ [HCM65].

2.1.2. Función de distribución de probabilidad

Una función $F_X(x) = P(X \leq x)$ es una función de distribución acumulativa de la variable aleatoria X si cumple con:

- F es no decreciente,
- F es continua a la derecha,
- $\lim_{x\to-\infty} F[x] = 0$ y $\lim_{x\to\infty} F[x] = 1$.

Familia Exponencial

Sea una familia $\{f(x;\theta); \theta \in \Omega\}$ de funciones de densidad de probabilidad, donde Ω es el intervalo $\Omega = \{\theta; \gamma < \theta < \delta\}$, γ y δ son constantes conocidas, y

$$f(x;\theta) = \exp[p(\theta)K(x) + S(x) + q(\theta)] \; si \; a < x < b,$$

$$= 0 \; en \; otro \; caso, \tag{2.1}$$

donde K y S son funciones de x, q y p funciones de θ.
Una función de densidad de probabilidad de la forma que se muestra en la ecuación (2.1) se dice que es de la familia exponencial [HCM65].
Algunas distribuciones, como la distribución gaussiana, la Gamma, la de Poisson, y la binomial es posible escribirlas como distribuciones de esta familia.
Esta familia será usada para modelar las distribuciones de las respuestas de los filtros de Gabor a la imagen, lo cual se explicará más adelante cuando se describan la representación *active basis*.

2.1.3. Teorema de Bayes

El teorema de Bayes es una de las herramientas más usadas en inferencia estadística, aplicado tanto en aprendizaje por computadora como en visión. Este resultado es una relación entre las probabilidades condicionales de dos variables. En la siguiente ecuación se muestra la definición de probabilidad condicional:

$$P(A|B) = \frac{P(A,B)}{P(B)}, \tag{2.2}$$

donde $P(A|B)$ denota la probabilidad del evento A una vez que se conoce el resultado del evento B.
Desarrollando $P(A|B)$ y $P(B|A)$ se tiene

$$P(B|A) = \frac{P(A,B)}{P(A)}, \tag{2.3}$$

$$P(B)P(A|B) = P(A,B) = P(A)P(B|A). \tag{2.4}$$

Por lo que al despejar se tiene que

$$P(A|B) = \frac{P(A)P(B|A)}{P(B)} \; si \; P(B) \neq 0. \tag{2.5}$$

Que es justamente el teorema de Bayes, este resultado se puede extender de varias formas para calcular otras probabilidades, de las cuales no se abundara en este documento [HCM65].

2.1.4. Modelos estadísticos

Un modelo estadístico en el contexto de este trabajo, está definido por un conjunto de ecuaciones las cuales describen el comportamiento de un objeto de estudio en términos de variables aleatorias y sus distribuciones asociadas. Por ejemplo, en un modelo *estadístico de forma*, la variable aleatoria representa las características de forma respecto a la respuesta a un filtro particular y la distribución de dicha variable estará parametrizada por una función de la familia exponencial.

2.1.5. Divergencia de Kullback-Leibler

Una manera de medir la *distancia* entre dos distribuciones es la divergencia de Kullback-Leiber (que formalmente no es una distancia puesto que no es simétrica) también conocida como entropía relativa. Se define con la ecuación (2.6) en el caso discreto y como se muestra en la ecuación (2.7) en el caso continuo [KL51],

$$KL(p,q) = \sum_k p_k log_2(\frac{p_k}{q_k}), \tag{2.6}$$

$$KL(p,q) = \int_{-\infty}^{\infty} p(x) log_2(\frac{p(x)}{q(x)}) dx, \text{[1]} \tag{2.7}$$

donde p y q son las distribuciones que se busca comparar.
Este concepto será usado más adelante para medir la *distancia* entre la distribución de las respuestas de ciertos filtros al fondo de una imagen y la distribución de las respuestas de los filtros en partes de las imágenes que contienen un objeto en particular.

2.1.6. Función de verosimilitud

Una función de verosimilitud $L(a)$ es la función de densidad de probabilidad para la ocurrencia de una configuración, dada una muestra $(x_1, ..., x_n)$, asumiendo la función de probabilidad $f(x; a)$ y que el parámetro a es conocido [Hoe84].

$$L(a) = f(x_1; a)...f(x_n; a) \tag{2.8}$$

2.1.7. Estimador de máxima verosimilitud

Un estimador de máxima verosimilitud es el valor del parámetro a tal que la función de verosimilitud alcanza un máximo [Hoe84].

2.2. Campos aleatorios de Markov

En esta sección se dará un repaso sobre los campos aleatorios de Markov, los cuales serán usados como herramienta para poder completar información acerca del estado del objeto, cuando se hace inferencia en el caso de oclusión.
Un campo aleatorio de Markov (MRF por sus siglas en inglés) es un modelo gráfico de una distribución de probabilidad. Este modelo consiste en un grafo no dirigido $G = (S, E)$ en el cual los nodos S representan relaciones entre las variables aleatorias.
Sea X_S el conjunto de variables aleatorias asociadas al conjunto de nodos S. Entonces, las aristas E representan las relaciones de independencia condicional usando la siguiente regla: dados los conjuntos disjuntos de nodos A, B y C, X_A

[1] log_2 representa el logaritmo base 2

es condicionalmente independiente de X_B dado X_C si no hay ningún camino de algún nodo del conjunto A a alguno del conjunto B que no pase por algún nodo de C. Esto se escribe como $X_A \perp\!\!\!\perp X_B|X_C$. Si tal camino existe entonces son dependientes.
La vecindad de N_n de un nodo n está definida como el conjunto de nodos que están conectados a n a través de las aristas del grafo, es decir:

$$N_n = \{n \in N|(n,m) \in E\}. \tag{2.9}$$

Dada la vecindad de un nodo, éste es independiente de otros nodos del grafo. Esto es, se puede escribir la probabilidad de X_n como en la ecuación (2.10).

$$P(X_n|X_N - X_n) = P(X_n|X_{N_n}). \tag{2.10}$$

La ecuación (2.10) es la propiedad de Markov, que además es la razón por la que se les da el nombre de campos aleatorios de Markov.
La propiedad de Markov dice que la distribución del campo X está completamente determinada por las distribuciones locales condicionales $P(X_n|X_{N_n})$.
No es completamente claro cómo construir la distribución conjunta global de estas funciones. Una forma de definir dicha distribución conjunta es usando una distribución de Gibbs, la cual se describirá a continuación.
La distribución de Gibbs en un grafo G se representa por la siguiente ecuación:

$$P(x) = \frac{1}{Z}\prod_{c\in C}\Phi_c(x_c), \tag{2.11}$$

donde el producto es sobre todos los *cliques maximales* en el grafo. Un *clique* es un subconjunto de nodos, tal que todos los nodos del subconjunto están conectados con todos los nodos del subconjunto. Un *clique maximal* es un subconjunto de nodos tal que si se le agrega otro nodo éste ya no puede ser extendido a otro *clique*. Z es la función de partición definida por la ecuación:

$$Z = \sum_x \prod_{c\in C}\Phi_c(x_c). \tag{2.12}$$

Donde $\Phi_c(x_c)$ se escribe usualmente como

$$\Phi_c(x_c) = e^{\frac{1}{T}V_c(x_c)}, \tag{2.13}$$

donde T denota la *temperatura*, la cual usualmente se toma como 1. Por lo tanto se puede escribir $P(x)$ como:

$$P(x) = \frac{1}{Z} e^{\frac{1}{T} - U(x)}, \tag{2.14}$$

donde

$$U(x) = \sum_{c \in C} V_c(x), \tag{2.15}$$

que es conocida como la energía; tanto $\Phi_c(x_c)$ y $V_c(x_c)$ serán referidos como los potenciales del *clique*.

El teorema de Hammersley-Clifford asegura que la probabilidad conjunta de cualquier MRF puede ser descrita con una distribución de Gibbs. Y aun más, para cualquier distribución Gibbs existe un MFR para el cual dicha distribución de Gibbs es la distribución conjunta del campo. Es decir, el teorema de Hammersley-Clifford establece una equivalencia entre los campos aleatorios de Markov y los modelos de Gibbs. Esto resuelve el problema de cómo especificar la distribución conjunta de un MRF en términos de funciones locales; pues basta con especificar los potenciales en cada *clique maximal* para definir dicha distribución conjunta.

Optimización

Una de las aplicaciones de estos modelos es encontrar la configuración más probable del campo dado un conjunto de observaciones, para ello se maximiza la función de distribución de Gibbs dada por la ecuación (2.14). Al calcular el logaritmo de la ecuación (2.14) se puede ver que es suficiente minimizar el potencial U. Existen varias formas de encontrar dicha configuración; depende esencialmente de la naturaleza de las variables, si son continuas o discretas. En el caso continuo, por su naturaleza se puede aplicar casi cualquier algoritmo de optimización del tipo de descenso de gradiente. En el caso discreto existen algoritmos como el de *Metropolis* [BS00], y el de *Recocido Simulado* [KGJV83].

Para más detalles sobre campos aleatorios de Markov se puede revisar el trabajo de Geman et al. [GGR84].

2.3. Conceptos sobre reconocimiento de objetos

En las siguientes tres subsecciones se describen los pasos o decisiones a ser tomadas para hacer reconocimiento de objetos. Este contenido es un breve repaso de los conceptos que se hablarán en el capitulo 3, donde se revisan los trabajos relacionados con estos temas. Estas decisiones caen en los siguientes tres puntos:

1. Representación
2. Aprendizaje
3. Reconocimiento

2.3.1. Representación

La representación del objeto se refiere a la manera de expresar el *objeto* en términos numéricos para la computadora. En la ecuación 2.16 se expresa de manera genérica la representación del objeto, es decir,

$$\theta = (x_1, ..., x_n), \tag{2.16}$$

donde las variables x_i son características que pueden ser medidas geométricas, estadísticas o características de color del objeto. En el caso de que los modelos sean modelos estadísticos, θ será una variable o vector aleatorio, al cual se le asocia una distribución de probabilidad que es lo que será *aprendido*.
Existen otras formas de describir la representación de un objeto como las representaciones estructurales que agregan relaciones entre los elementos que componen la representación teniendo como resultado una estructura.

2.3.2. Aprendizaje

Una vez que se tiene una representación del objeto, la siguiente decisión a tomar es definir una distribución asociada a dicha representación y estimar sus parámetros. Existen dos variantes principales que corresponden a asumir un modelo discriminativo o un modelo generativo; existen también modelos híbridos, pero por ahora solo se describirán los modelos discriminativos y generativos. La

diferencia entre los modelos generativos y los modelos discriminativos radica en que lado de la ecuación (2.17) toman en cuenta, esto es cómo representan la distribución de θ, según la expresión:

$$P(\theta|Image) = \frac{P(Image|\theta)P(\theta)}{P(Image)} \tag{2.17}$$

Donde $Image$ representa las observaciones y θ representa el modelo a priori que se conoce.

Modelos Discriminativos

Los modelos discriminativos *aprenden* la parte izquierda de la ecuación (2.17). Esto significa que construyen límites entre las regiones que corresponden a la distribución de las clases dadas las observaciones θ, véase figura 2.1.
Cabe mencionar que este tipo de modelos tienen la ventaja de que suelen ser más fáciles de aprender respecto a los modelos generativos. Sin embargo sólo son suficientes para delimitar regiones en el espacio de características.

Modelos Generativos

Los modelos generativos usan la parte derecha de la ecuación (2.17), y lo que hacen es aprender la distribución de las posibles categorías. Esta forma de considerar el modelo, no solo permite diferenciar cosas entre sí; sino que también posibilita el generar ejemplos a partir de las distribuciones aprendidas. En la figura 2.1 se puede ver que los modelos generativos aprenden las distribuciones de cada una de las regiones. Esto puede ser muy conveniente en visión por computadora ya que se puede obtener información de más alto nivel.

2.3.3. Inferencia en el reconocimiento

Una vez que se tiene el modelo de los datos representados por θ y dependiendo de que se aprendió (si un modelo discriminativo o generativo), la representación y el modelo pueden ser usados para hacer reconocimiento. El objetivo en este caso es buscar o estimar los parámetros del modelo que mejor ajustan a la imagen.
Existen varias formas de decir que se reconoció un objeto en una imagen:

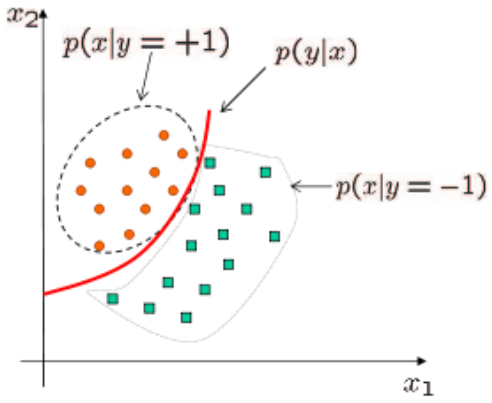


FIGURA 2.1: Se muestra la diferencia entre los modelos generativos y los modelos discriminativos. $p(y|x)$ representa el modelo aprendido por los modelos discriminativos, se puede observar que delimita dos regiones. Por otro lado los modelos generativos representados por $p(x|y = -11)$ y $p(x|y = 1)$, describen zonas las zonas correspondientes a cada etiqueta.

- Verificación: Responde a la pregunta si una subimagen corresponde o no al objeto.
- Detección: Responde la pregunta si en la imagen está el objeto y donde.
- Identificación: Contesta la pregunta quién está; en este caso se refiere a un reconocimiento entre ejemplos del mismo objeto, por ejemplo identificar una persona.
- Etiquetar: Se refiere a poner etiquetas a partes de las imágenes que se cree que corresponden a objetos de esa categoría.
- Categorizar el contexto: Se refiere a poner etiquetas a la imagen describiendo que es lo que puede encontrarse en la imagen; en este caso son etiquetas *suaves* las cuales no dan más información.

Dependiendo de cada una de estas categorías se define el tipo de modelo que se requiere, y el tipo de inferencia que se hace. A continuación se presentan dos tipos de inferencia comúnmente usados en el caso de detección e identificación

Inferencia de arriba a abajo (*Top-Down*)

Este tipo de inferencia como su nombre lo dice es ir de lo general a lo particular; lo que hace es buscar los parámetros del modelo global y luego irlos detallando. A

pesar de que puede ser muy rápida, tiene el inconveniente de que es muy fácil que encuentre óptimos locales, y falle en la tarea de reconocimiento.

Inferencia de abajo a arriba (*Button-up*)

En este caso la idea es inferir a partir de detalles encontrados e ir construyendo hipótesis cada vez más generales; tiene la desventaja de que este tipo de inferencia suele ser lenta. Este tipo de inferencia, dada su naturaleza, es útil cuando se usan descriptores locales.

2.4. Filtros de Gabor

En esta sección se describirá que son los filtros de Gabor, este concepto es muy importante pues la representación *active basis*, descrita más adelante, está compuesta por filtros de Gabor.
Los filtros de Gabor son un caso particular de los filtros de cuadratura o pares de cuadratura, los cuales no se detallan en este documento [Jah91]. Los *kernels* de los filtros de Gabor se ven como elementos bases de Fourier que son multiplicados por gaussianas; con esto se puede decir que los filtros de Gabor responden en puntos de la imagen donde hay componentes que localmente tienen una frecuencia espacial particular en cierta orientación [Jah91]. Estos filtros vienen en parejas, un elemento de la pareja recupera componentes simétricos en una dirección particular y la otra parte recupera los componentes asimétricos.
Los filtros de Gabor matemáticamente se representan de la siguiente forma:

$$G_{symetric} = cos(k_x x + k_y y)e^{-\frac{x^2+y^2}{2\sigma^2}}, \tag{2.18}$$

$$G_{antisymetric} = sin(k_0 x + k_1 y)e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{2.19}$$

Otra forma de escribir esto en términos de una sola función es la siguiente:

$$G(x,y) = e^{-\frac{(\frac{x}{\sigma_x})^2+(\frac{y}{\sigma_y})^2}{2}} e^{ix}. \tag{2.20}$$

Con la ecuación (2.20) se va a construir un banco de filtros $\{B_{x,y,s,\alpha}\}$ que se puede ver gráficamente en la figura 2.2. Cada elemento de dicho banco será expresado y

calculado por:

$$B_{x,y,s,\alpha}(x', y') = \frac{G(\frac{\hat{x}}{s}, \frac{\hat{y}}{s})}{s^2} \tag{2.21}$$

donde

$$\hat{x} = (x' - x)cos\alpha - (y' - y)sin\alpha, \tag{2.22}$$

$$\hat{y} = (x' - x)sin\alpha + (y' - y)cos\alpha, \tag{2.23}$$

s será la escala y α la orientación.

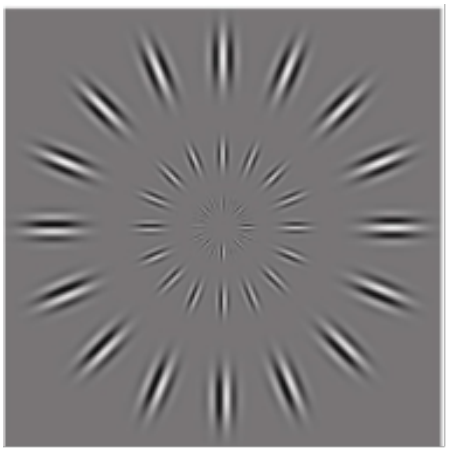


FIGURA 2.2: Ejemplo de una familia de filtros de Gabor

2.5. Modelos *Active Basis*

En esta sección se detallan los modelos *active basis*, cómo se entrenan y cómo se usan en el reconocimiento de objetos. Además se dará un repaso a la teoría usada para explicar el porqué los algoritmos descritos más adelante funcionan y porqué lo hacen de esa manera, sin embargo, esta sección no es fundamental para el entendimiento de los mismos.[2]

[2]El contenido del la mayor parte de esta sección está basado en los trabajos de Wu, Y. N et al. [WSF+07, WSGZ09], que son los trabajos donde se propone el uso de *active basis*.

2.5.1. Representación

La representación *active basis* expresa una imagen de la siguiente forma:

$$I_m = \sum_{i=0}^{n} c_{m,i} B_{m,i} + \epsilon_m, \tag{2.24}$$

$$B_i \approx B_{m,i} \tag{2.25}$$

donde $B_{m,i}$ son filtros de Gabor, $\{c_{m,i}, i = 1, ..., n\}$ son coeficientes, y ϵ_m es el residuo de la imagen I_m (la información de la imagen I_m no explicada por el termino $\sum_{i=0}^{n} c_{m,i} B_{m,i}$). Para definir $B_i \approx B_{m,i}$, supóngase

$$B_i = B_{x_i, y_i, s, \alpha_i}$$

$$B_{m,i} = B_{x_{m,i}, y_{m,i}, s, \alpha_{m,i}}$$

entonces $B_i \approx B_{m,i}$ si y sólo si existen $(\delta_{m,i}, d_{m,i})$ tales que

$$x_{m,i} = x_i + d_{m,i} \sin \alpha_i$$

$$y_{m,i} = y_i + d_{m,i} \cos \alpha_i$$

$$\alpha_{m,i} = \alpha_i + \delta_{m,i}$$

$$d_{m,i} \in [-b_1, b_1], \delta_{m,i} \in [-b_2, b_2] \tag{2.26}$$

Esto es a cada elemento B_i se le permite cierto grado de deformación, este comportamiento se ilustra en la figura 2.3.

Esta forma de representar la imagen es muy similar a la usada por *wavelets* [Mal89], la diferencia de los *wavelets* con esta representación es que cada elemento $B_{m,i}$ representa una subfamilia de filtros de Gabor.

El objetivo de estos modelos es representar el objeto del cual se está aprendiendo un modelo, como suma de los filtros elegidos B_i, mientras que el fondo de la imagen quede representado como el residuo ϵ_m. Para lograr este objetivo se deben elegir los B_i correctos que representen el objeto. Posteriormente el modelo aprendido se usa para reconocer objetos en una imagen nueva donde aparece el objeto.

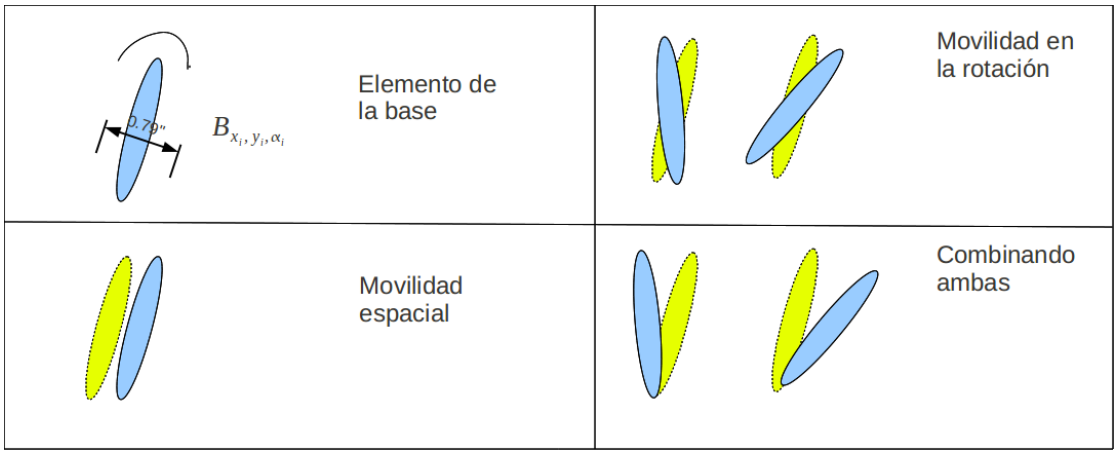


FIGURA 2.3: Representación de la *movilidad* que se le permite a cada uno de los elementos de la bases usados para formar el modelo

Notación

A partir de aquí, cuando se use la letra B se refiere al conjunto de filtros de Gabor previamente aprendidos. Cuando se mencione a B_i se refiriere a un elemento particular de B, el cual tiene sus parámetros implícitos de posición, de orientación y de escala.

2.5.2. Algoritmo de aprendizaje

En esta subsección se describe cómo se aprenden los parámetros de la representación *active basis*, el resultado del algoritmo descrito más adelante será usado como entrada para una de las aportaciones de la tesis.
Como ya se mencionó anteriormente en la ecuación (2.24), una imagen se representa como suma de ciertos filtros de Gabor multiplicados por los coeficientes $c_{m,i}$ más un residuo.
Para poder explicar cómo elegir los elementos B_i, primero se asumirá lo siguiente:

$$p(I_m|B_m) = q(I_m)\frac{p(c_{m,1},...,c_{m,n})}{q(c_{m,1},...,c_{m,n})}, \tag{2.27}$$

donde I_m es la m-ésima imagen y B_m es una deformación de un conjunto B de filtros de Gabor que mejor explica a la imagen I_m, $q(I_m)$ es la distribución de referencia de la forma de una familia exponencial, $p(c_{m,1},...,c_{m,n})$ es la probabilidad conjunta de los coeficientes de los filtros, $q(c_{m,1},...,c_{m,n})$ es la distribución de

los coeficientes de los filtros asumiendo la distribución de referencia y $p(I_m|B_m)$ es la probabilidad de la imagen I_m dado B_m. Donde además las distribuciones marginales $p(c_{m,i})$ se asumirá que son independientes entre sí. La distribución $q(I_m)$ que modela el residuo ϵ_m, se considerará igual para todas las imágenes. La justificación de esto está descrita más adelante, ver la subsección 2.5.4.
Considerando que se tiene una distribución de referencia con sus parámetros ya conocidos, el objetivo es elegir elementos $B_{m,i}$ de tal forma que la *distancia* entre las distribuciones $p(I_m|B_m)$ y $q(I_m)$ sea máxima. Esta estimación de distancia está dada por la divergencia de Kullback-Leiber, ya definida en la subsección 2.1.5. A continuación se darán paso a paso las herramientas para elegir los elementos B_i, simplificando el modelo hasta tener una forma sencilla de elegirlos.

Logaritmo de la verosimilitud y la divergencia de Kullback-Leiber

Para aprender B y $\{B_m \approx B, m = 1, ..., M\}$ se puede maximizar el promedio del logaritmo de la verosimilitud,

$$\frac{1}{M}\sum_{m=1}^{M}\log\frac{p(I_m|B_m)}{q(I_m)} = \frac{1}{M}\sum_{m=1}^{M}\log\frac{p(c_{m,1},...,c_{m,n})}{q(c_{m,1},...,c_{m,n})}. \tag{2.28}$$

Se puede observar que la ecuación (2.28) converge a $KL(p(c_{m,1},...,c_{m,n})||\ q((c_{m,1},...,c_{m,n})))$ cuando $M \to \infty$, a condición de que $p(c_{m,1},...,c_{m,n})$ pueda ser consistentemente estimado a partir de imágenes de entrenamiento. Para maximizar el cociente del logaritmo de la verosimilitud, se va a elegir B y deformarla en $\{B_m \approx B\}$ para maximizar $KL(p(c_{m,1},...,c_{m,n})||\ q(c_{m,1},...,c_{m,n}))$ de modo que el contraste alcanzado entre la textura de fondo y la forma que sobresale es máximo.

Parametrización asumiendo una distribución de la familia exponencial

Es posible simplificar la divergencia de Kullback-Leiber suponiendo el siguiente modelo de la familia exponencial:

$$p(c;\lambda) = \frac{1}{Z(\lambda)}\exp\{\lambda h(r)\}q(c), \tag{2.29}$$

donde $\lambda > 0$ es el parámetro, $r = |c|^2$, y

$$Z(\lambda) = \int \exp\{\lambda h(r)\} q(c) dc = E_q[\exp\{\lambda h(r)\}], \tag{2.30}$$

es la constante de normalización. $h(r)$ es una función creciente, tal que $p(c;\lambda)$ asigna más probabilidad que $q(c)$ en aquellos c con un r más largo, si se tiene que $r = |c|^2$
Sea $p(r;\lambda)$ y $q(r)$ las densidades de $r = |c|^2$ bajo $p(c;\lambda)$ y $q(c)$ respectivamente, entonces $p(c;\lambda)/q(c) = p(r;\lambda)/q(r) = \exp\{\lambda h(r)\}/Z(\lambda)$.

Estimación de p_i

Se puede estimar $q(c)$ construyendo histogramas a partir de imágenes naturales. $p_i(c)$ se estima a partir de $\{c_{m,i} =< I_m, B_{m,i} >, m = 1, ..., M\}$, ajustando la densidad $p(c;\lambda_i)$ a $\{c_{m,i}\}$. Específicamente, se define la media como:

$$\mu(\lambda) = E_\lambda[h(r)] = \int h(r) \frac{1}{Z(\lambda)} \exp\{\lambda h(r)\} q(r) dr, \tag{2.31}$$

Para estimar el valor del parámetro λ_i se resuelve la ecuación siguiente:

$$\mu(\lambda_i) = \frac{1}{M} \sum_{m=1}^{M} h(r_{m,i}), \tag{2.32}$$

donde $r_{m,i} = |c_{m,i}|$, entonces se tiene que $\hat{\lambda} = \mu^{-1}(\sum_{m=1}^{M} h(r_{m,i})/M)$. $\hat{\lambda}_i$ es el estimador de máxima verosimilitud que maximiza $\sum_{m=1}^{M} \log[p(c_{m,i};\lambda)/q(c_{m,i})]$ sobre λ_i. Por lo que se estima $p_i(c)$ con $p(c;\hat{\lambda}_i)$.

Selección de B_i

Al sustituir en la ecuación (2.28) el modelo paramétrico y usando una distribución de la forma de la familia exponencial se tiene que:

$$\frac{1}{M} \sum_{i=1}^{M} \log \frac{p(c_{m,i};\hat{\lambda}_i)}{q(c_{m,i})} = KL(p(c;\hat{\lambda}_i)||q(c)). \tag{2.33}$$

Ésta es una función creciente de $\frac{1}{M}\sum_{m=1}^{M} h(r_{m,i})$. Por lo tanto, se elige B_i y se perturba para formar $\{B_{m,i}\}$ maximizando el índice de búsqueda $\sum_{m=1}^{M} h(r_{m,i})$.

Perturbar B_i en $\{B_{m,i}\}$ es una función monótona creciente de $r = |c|^2$. Esto justifica que, dado B_i, se puede perturbar B_i en $\{B_{m,i}\}$ para maximizar $| < I_m, B_{m,i} > |$, sujeto a la restricción de no traslape. Este $B_{m,i}$ es el estimador de máxima verosimilitud dado B_i.

Así que, la estimación de λ_i, la perturbación de B_i en $B_{m,i}$ y la selección de B_i siguen el principio de máxima verosimilitud.

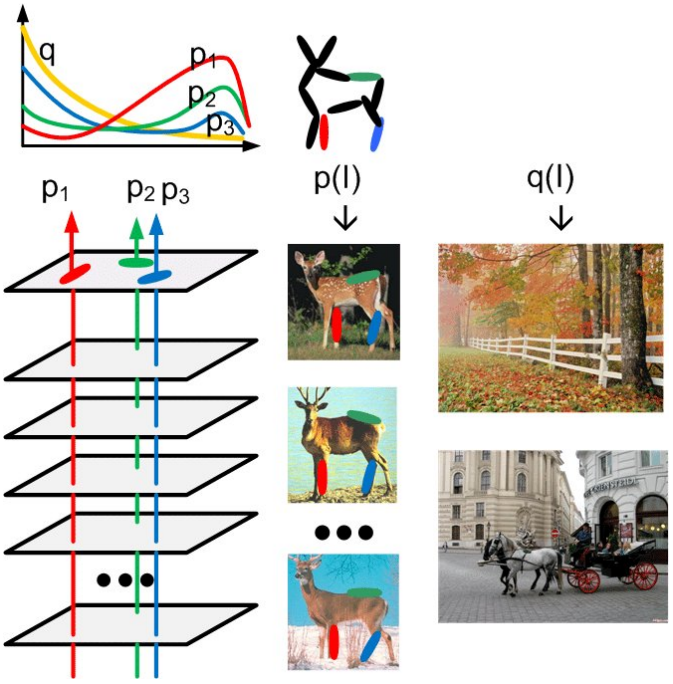


FIGURA 2.4: Muestra cómo se comporta el algoritmo 3 de manera gráfica. Dada la distribución de referencia, se estiman los histogramas de las respuestas y usando los histogramas se eligen los filtros de Gabor cuyas distribuciones de las respuestas se alejan más a la distribución de referencia.

Algoritmo

Resumiendo lo anterior, se quiere elegir un conjunto de filtros B_i que representa un conjunto de $\{B_{m,i}\}$ *perturbados*, los cuales formarán juntos un modelo o plantilla de un objeto del cual se está entrenando el modelo. Para ello se asumen varias cosas, entre ellas que la distribución de las respuestas de los filtros sigue una distribución de la familia exponencial, dichas distribuciones son independientes, lo que se traduce en que los filtros elegidos no se traslapan entre sí. Además, se asume que se conoce la distribución $q(I_m)$ del fondo. Finalmente para elegir los elementos se busca maximizar la *distancia* entre las distribuciones $q(I_m)$ y $p(I_m|B)$, eligiendo

los elementos B_i que cumplan las condiciones ya mencionadas. En la figura 2.4 se ejemplifica lo descrito en este párrafo. A continuación se describe la parte

Algorithm 1 Convolución de $\{I_1, .., I_m\}$ con los filtros de Gabor

Require: $I = \{I_1, ..., I_M\}$
1: **for** m=1 to M **do**
2: **for** $(x, y) \in D$ and α **do**
3: $SUM1_m(x, y, s, \alpha) = whitening(| < I_m, B_{x,y,s,\alpha} > |^2)$
4: **end for**
5: **end for**
6: **return** SUM1

Algorithm 2 Maximización Local de SUM1

Require: $SUM1$
1: **for** m=1 to M **do**
2: **for** $(x, y) \in D$ and α **do**
3: $MAX1_m(x, y, s, \alpha) = \max_{d\in[-b_1,b_1]\delta\in[-b2,b2]} SUM1_m(x + d\cos(\alpha), y + d\sin(\alpha), s, \alpha + \delta)$
4: Record $TRACK1_m(x, y, s, \alpha)$
5: **end for**
6: **end for**
7: **return** MAX1

algorítmica de la construcción del modelo, la cual se puede ver en pseudocódigo en el algoritmo 3. Este se divide en varias partes, las cuales construyen el modelo asegurando las propiedades descritas anteriormente.

Convolución (algoritmo 3 en la línea 1): Lo que se hace es aplicar los filtros de Gabor previamente construidos a diferentes orientaciones α en todas las imágenes, s no varía. El pseudocódigo de esta parte se muestra en el algoritmo 1, en esta parte no solo se convolucionan las imágenes con los filtros, también se aplica la función h denotada como la función *whitening* en el pseudocódigo. Hasta aquí no se ha elegido nada, simplemente se calculan todas las respuestas de los filtros.

Maximización Local (algoritmo 3 en la línea 2): En cada punto o *píxel* de la imagen se busca la mayor respuesta en una vecindad de dicho píxel. Esta parte permite poder decir que $B_i \approx B_{m,i}$. Esto se puede interpretar como que se le permite al modelo cierto grado de deformación del objeto tanto en la parte de aprendizaje como en la parte de inferencia. El pseudocódigo de esta etapa se muestra en el algoritmo 2.

Selección (algoritmo 3 en la línea 3): En esta parte se elige el siguiente elemento de la base. Para ello, dadas las respuestas en todas las imágenes a los filtros se

Algorithm 3 Algoritmo de aprendizaje a partir de $\{I_1, .., I_m\}$(Sheared Sketch)

Require: $I = \{I_1, ..., I_M\}$
1: $SUM1 = Convolution(I)$(see algorithm (1))
2: $MAX1 = LocalMaximization(SUM1)$(see algorithm (2))
3: **Selection:**
4: $i \leftarrow 1$
5: **for** $m = 0$ to M **do**
6: $SUM2_m \leftarrow \vec{0}$
7: **end for**
8: Find (x_i, y_i, α_i) by maximazing $\sum_{m=1}^{M} h(MAX1_m(x_i, y_i, s, \alpha))$
9: Compute λ_i from $\sum_{m=1}^{M} h(MAX1_m(x_i, y_i, s, \alpha)) - logZ(\lambda_i)$
10: **for** $m = 1$ to M **do**
11: Update $SUM2_m \leftarrow SUM2_m + \lambda_i h(MAX1_m(x_i, y_i, s, \alpha_i))$
12: **end for**
13: **Non-Maximum suppression:**
14: **for** $m = 1$ to M **do**
15: Retrive $(x_{m,i}, y_{m,i}, \alpha_{m,i}) \leftarrow TRACK1_m(x_i, y_i, s, \alpha_i)$
16: **if** $MAX1_m(x_i, y_i, s, \alpha_i) > 0$ **then**
17: **for** $(x, y) \in D$ and α **do**
18: **if** $corr(B_{x_{m,i}, y_{m,i}, s, \alpha_{m,i}}, B_{x,y,s,\alpha}) > \epsilon$ **then**
19: $SUM1(x, y, s, \alpha) \leftarrow 0$
20: **end if**
21: **end for**
22: Re-compute MAX1 using **Local Maximization**
23: **end if**
24: **end for**
25: **if** $i = n$ **then**
26: STOP
27: **else**
28: $i \leftarrow i + 1$
29: GOTO: **Selection**
30: **end if**
31: **return** $\{B = \{B_1, ..., B_n\}, \Lambda\}$

toma el filtro cuya suma sea mayor, esto representa elegir un borde saliente del objeto. Con esto se asegura la parte de maximización de la ecuación (2.28).

Supresión (algoritmo 3 en la línea 13): Una vez que se eligió un elemento para la base se remueven todos aquellos filtros que se traslapan con dicho elemento, esto se hace como se muestra en la línea 18, buscando todos los filtros cuya correlación con el filtro sea mayor que cero. Este paso asegura la independencia entre las distribuciones de las respuestas de los filtros.

Resultado

Después de haber elegido los elementos de la base B se tienen n filtros de Gabor B_i con sus parámetros (x_i, y_i, α_i) y los parámetros $(\lambda_i, \log Z_i)$ de la distribución de la familia exponencial asociada a las respuestas de las imágenes al i-ésimo filtro.

2.5.3. Etapa de inferencia

En esta parte se explicará el algoritmo usado por la representación *active basis* para reconocer un objeto en una imagen nueva, este algoritmo por la forma en que está estructurado lleva el nombre de *SumMaxMaps*. En importante mencionar que este algoritmo será modificado como parte de las contribuciones de la tesis.
Una vez entrenado el modelo, se tiene un conjunto de $B = \{B_i \approx B_{x_i,y_i,s_i,\alpha_i}\}$ con sus respectivos parámetros $(\lambda_i, \log z_i)$ de la distribución de las respuestas a los filtros modelados por una función de la familia exponencial. Este modelo puede ser usado para reconocer el objeto en una imagen, esto es para detectar y hacer un bosquejo del objeto encontrado. Si I es la imagen en donde se quiere detectar o ajustar el modelo, lo que se necesita es saber la localización de los B_i. Para encontrar dichos parámetros espaciales es necesario encontrar el lugar donde se maximiza la verosimilitud de $P(I|B)$, para ello se sugiere como medida de ajuste la ecuación (2.34) que es el logaritmo de la verosimilitud[3],

$$Match(I, B) = log(\frac{P(I|B)}{q(I)}) = \sum_{i=0}^{n} \lambda_i h(| < I, B_i > |^2) - logZ(\lambda_i). \tag{2.34}$$

Algoritmo

En la figura 2.5 se muestra la idea gráfica del algoritmo que encuentra los mejores parámetros de $B_i = B_{x_i,y_i,s,\alpha_i}$ en la imagen nueva usando programación dinámica, de manera global lo que hace es construir mapas $SUM1$, $MAX1$, $SUM2$, $MAX2$. Estos mapas se hacen para asegurar algunas propiedades para maximizar la ecuación (2.34). $SUM1$ contiene las respuestas de los filtros, $MAX1$ contiene los máximos de las respuestas locales en cada punto, $SUM2$ representa la suma en cada pixel dada por la ecuación (2.34) de los filtros B ya elegidos, y finalmente

[3]Aunque también se sugieren otras formas de estimar que tan bien se ajusta el modelo a la imagen pero en este trabajo no se abundará al respecto.

$MAX2$ representa el lugar donde se maximiza la ecuación (2.34). El algoritmo 4 es el pseudocódigo que encuentra los mejores parámetros del modelo en la imagen. De manera similar al algoritmo de aprendizaje primero se filtran la imagen I con el banco de filtros de Gabor y estas respuestas se maximizan localmente, construyendo $SUM1MAP$ y $MAX1MAP$. Lo que cambia respecto a la etapa de aprendizaje es a partir de la línea 7, lo que se hace es usar B como filtro de forma sobre las respuestas de los filtros ya maximizadas en $MAX1$. Esto se hace para cada posición en la imagen, obteniendo el valor del logaritmo de la verosimilitud para cada posición, que representa que tan bien se ajusto el modelo en ese punto. Una vez calculadas las respuestas a los filtros de forma se busca la posición que maximiza la verosimilitud, para obtener la posición donde mejor se ajusta la plantilla.

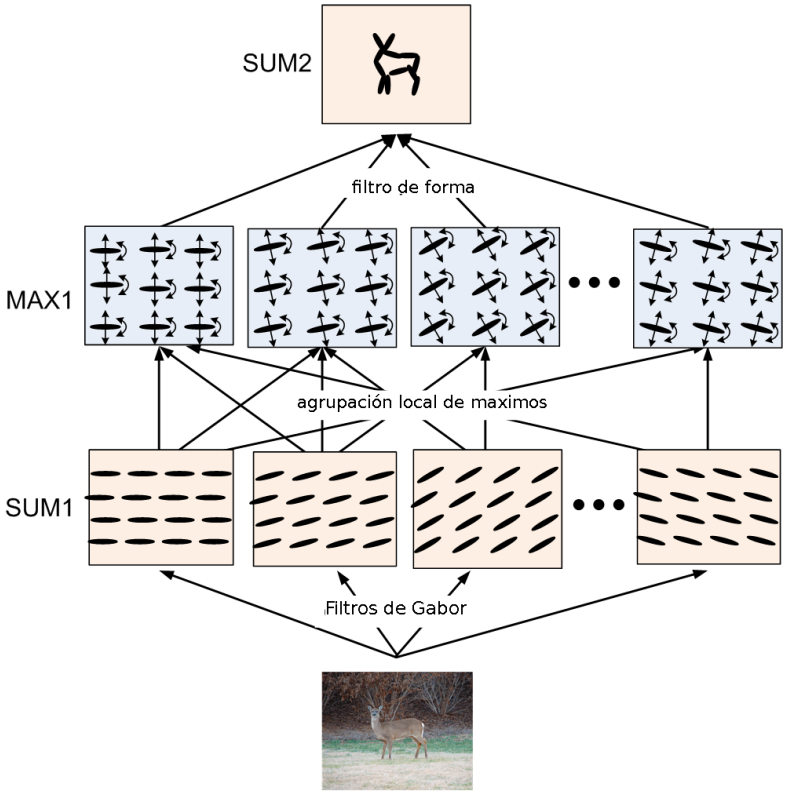


FIGURA 2.5: Algoritmo SumMaxMaps, el cual está dividido en varias etapas, SUM1 que filtra la imagen, MAX1 agrupa los máximos locales de las respuestas, SUM2 aplica un filtro de forma al mapa MAX1 y finalmente a partir del mapa SUM2 es posible encontrar el lugar donde se ajusta mejor la plantilla del objeto

Algorithm 4 SumMaxMaps

Require: $I, \{B_i, \lambda_i, \log Z_i, \vec{x_i}, \alpha_i\}$

1: **for** all $(x, y) \in D$ and α **do**
2: $\quad SUM1(x, \theta) \leftarrow h(| < I, B_{x,y,s,\alpha} > |)$
3: **end for**
4: **for** $(x, y) \in D$ and α **do**
5: $\quad MAX1(x, y, \theta) \leftarrow \underset{\Delta x, \Delta y, \Delta \alpha}{\text{máx}} SUM1(x + \Delta x, y + \Delta y, \alpha + \Delta \alpha)$
6: **end for**
7: **for** $(x, y) \in D$ **do**
8: $\quad SUM2(x, y) \leftarrow \sum_{i=1}^{n} [\lambda_i MAX1(x + x_i, y + y_i, \alpha_i) - \log Z_i]$
9: **end for**
10: **return** $argmax_{(x,y)\in D} SUM2(x, y)$

Análisis de complejidad

En esta parte se desarrolla el análisis de complejidad de la etapa de reconocimiento, para el entrenamiento se aplica un análisis similar. Como se puede observar este algoritmo tiene cuatro pasos SUM1, MAX1, SUM2 y MAX2. Cada uno de los pasos es independiente del anterior por lo que la complejidad total es la suma de cada una de ellas. Antes de dar la complejidad total, los parámetros que se usarán son los siguientes, D es el número de píxeles de la imagen donde se va a buscar el objeto, $NumElements$ se refiere al número de elementos en la base $B = \{B_i\}$, $NumOrient$ número de posibles orientaciones de cada elemento, $ShiftSize$ es el parámetro relacionado con la variabilidad que se le da a cada elemento de la base.

SUM1: Esta etapa consiste sólo en filtrar las imágenes, lo que implica aplicar $NumOrient$ filtros a la imagen, se puede hacer esto usando la transformada rápida de Fourier completando las imágenes a que sus dimensiones sean potencias de 2 se tiene que la complejidad es $O(NumOrient * D \log D)$.

MAX1: Esta etapa maximiza localmente la respuesta de cada uno de los filtros por lo que su complejidad es $O(D * shiftSize)$.

SUM2: Esta etapa aplica el filtro de forma a la imagen en cada píxel usando lo que ya se tiene en la etapa MAX1. La complejidad en este paso es $O(D * numElements)$.

MAX2: Finalmente esta etapa encuentra la mayor respuesta al filtro de forma sobre todos los posibles lugares donde puede estar el objeto, por lo que la complejidad es $O(D)$.

Complejidad total: Uniendo la complejidad de cada etapa se tiene que la complejidad de todo el proceso de inferencia es $O((NumOrient * D \log D)) + D * shiftSize + D * NumElements + D)$ esto se convierte en $O(D * NumElements)$ después de tomar el termino más grande de la expresión previa. Este resultado es para cada escala posible de la imagen.

2.5.4. Teoría

En esta subsección se presentan los fundamentos teóricos que se usan como base para desarrollar los algoritmos usados para la representación *active basis*. Sin embargo, el entendimiento de esta parte no es fundamental para el uso de los algoritmos descritos, pero es útil para entender cómo funcionan.

Distribución de probabilidad de las intensidades de las imágenes

Asumiendo que se tiene un conjunto de imágenes de entrenamiento $\{I_m, m = 1, ..., M\}$ representadas por la ecuación (2.24) es posible estimar la distribución de $\{c_{m,i}, i = 1, ..., n\}$, así como la distribución del residuo ϵ_m. Dadas estas dos distribuciones, se puede calcular la distribución de I_m respecto de B_m (las bases) esto es $p(I_m|B_m)$. Dadas estas distribuciones maximizando la verosimilitud es posible tener una forma de aprender B_i y posteriormente poder inferir información en una nueva imagen.
Para simplificar se usará una notación matricial más clara, I_m será tratada como un vector de $|D| \times 1$ donde $|D|$ es el número de píxeles de la imagen I_m. $B = (B_{i,0}, B_{i,1}, i = 1, ..., n)$ se puede manejar como una matriz de $|D| \times 2n$, donde cada $B_{i,\eta}(\eta = 0, 1)$ es un vector de $|D| \times 1$. Por lo tanto cada B_m se puede tratar como una matriz de $|D| \times 2n$. Se puede escribir $C_m = (c_{m,0}, c_{m,1}, m = 1, ..., n)^t$ como un vector de $2n \times 1$. En consecuencia la ecuación (2.24) se puede reescribir como $I_m = B_m C_m + U_m$.

Descomposición lineal

Asumiendo que $B_m C_m$ es la proyección de I_m al subespacio generado por los vectores columnas de B_m, entonces queda $C_m = (B_m^t B_m)^{-1} B^t I_m$. Si B_m es ortogonal, entonces $C_m = B^t I_m$. ϵ_m cae en un subespacio de dimensión $|D| - 2n$ que es ortogonal a las columnas de B_m. No hay pérdida de generalidad en tal suposición, porque sí ϵ_m no es ortogonal a B_m, se puede proyectar ϵ_m en B_m, y dejar que $B_m C_m$ absorba esta proyección. Entonces se puede escribir $\epsilon_m = \hat{B}_m \hat{C}_m$ donde $\hat{B}_m$ es una matriz de $|D| \times (|D| - 2n)$ cuyas columnas son ortogonales a las de B_m y $\hat{C}_m$ es un vector de $(|D| - 2n) \times 1$. Entonces, la representación queda expresada como la siguiente ecuación:

$$I_m = B_m C_m + \hat{B}_m \hat{C}_m \tag{2.35}$$

No existe un mapeo lineal uno a uno entre I_m y $(C_m, \hat{C}_m)$. $\hat{B}_m$ y $\hat{C}_m$ pueden ser construidas implícitamente mediante modelación estadística.

Forma y textura

Dado lo anterior es posible especificar la función de densidad $p(I_m|B_m)$. Para la representación lineal dada por (2.35) se tiene que

$$p(I_m|B_m) = p(C_m, \hat{C}_m)|J_m| = p(C_m)P(\hat{C}_m|C_m)|J_m|, \tag{2.36}$$

donde $|J_m|$ es el valor absoluto del determinante de la matriz Jacobiana de la transformación lineal de I_m a $(C_m, \hat{C}_m)$. $p(C_m)$ es la distribución de los coeficientes para representar el primer plano de la imagen, y $p(\hat{C}|C_m)$ es la distribución de la textura residual del fondo, dados los coeficientes del primer plano de la imagen. Esta distribución $p(I_m, B_m)$ está completamente determinada por $p(C_m)$ y $p(\hat{C}_m|C_m)$.

Sea $q(I_m)$ la distribución de referencia. Análogamente que para la ecuación (2.36) se puede escribir $q(I_m) = q(C_m)q(\hat{C}_m|C_m)|J_m|$, con el mismo Jacobiano $|J_m|$. Ahora se quiere construir $p(I_m, B_m)$ modificando $q(I_m)$. Específicamente, se puede asumir que $p(\hat{C}_m|C_m) = q(\hat{C}_m|C_m)$, lo que significa que la distribución condicional de la textura residual de fondo en $p(I_m, B_m)$ se supone que es la misma que $q(I_m)$.

Entonces,

$$p(I_m|B_m) = q(I_m)\frac{p(C_m)}{q(C_m)} = q(I_m)\frac{p(c_{m,1},...,c_{m,n})}{q(c_{m,1},...,c_{m,n})}, \tag{2.37}$$

donde se sustituye $p(C_m)$ por $q(C_m)$ para construir la densidad $p(I_m)$ de $q(I_m)$. El modelo expresado por la ecuación (2.37) combina la textura y la forma. $q(I_m)$ modela la textura del fondo, con B_m y $p(C_m)$ modelando la forma saliente. La forma saliente resalta de la textura de fondo, y es modelada por la relación de probabilidad $\frac{p(C_m)}{q(C_m)}$.

Substitución de densidad y máxima entropía

La forma (2.37) es un esquema de substitución usado en la técnica de proyección en la búsqueda [Fri87]. Esta substitución también es válida en una reducción diferencial no lineal de I_m, o si C_m es discreta. Esta forma permite construir un modelo sobre las intensidades de la imagen en lugar de hacerlo en el espacio de características. Este modelo generativo hace posible la elección de características muestreando la imagen. El modelo dado por la ecuación (2.37) puede ser justificado por el principio de máxima entropía [DPDPL97]: $p(I_m|B_m)$ es la distribución que mejor aproxima a $q(I_m)$ en términos de la divergencia de Kullback-Leibler, sobre todas las distribuciones que comparten $p(C_m)$.

Elección de la distribución de referencia

Asumiendo que $q(I_m)$ es estacionaria. Las siguientes elecciones de $q(I_m)$ son posibles:

1. Distribución gaussiana de ruido blanco. Esta distribución es más comúnmente elegida en el modelo lineal, en el caso del ajuste de mínimos cuadrados se asume implícitamente.

2. Una distribución marginal no gaussiana para subimágenes de imágenes *naturales*. Ésta es la distribución $q(I_m)$ que se usa en este trabajo. En particular, se asume que las distribuciones marginales de $< I_m, B_{x,y,s,\theta} >$ son todas iguales a las de las distribuciones marginales. Dicha distribución marginal es altamente no-gaussiana, con una cola larga que permite bordes fuertes ocasionales. También se asume que $q(I_m)$ hereda la independencia

ortogonal de las distribuciones de ruido blanco gaussiano. Dicha distribución es la modificación más simple a la distribución gaussiana, y provee un mejor modelo que la distribución gaussiana para un fondo U_m, permitiendo bordes fuertes en U_m.
En la figura 2.6 se muestra el histograma marginal de $h(|< I_m, B_{x,y,s,\alpha} >|)$ construido a partir de imágenes *naturales*.

3. Un campo aleatorio de Markov que coincide con las distribuciones marginales de las respuestas de los filtros.

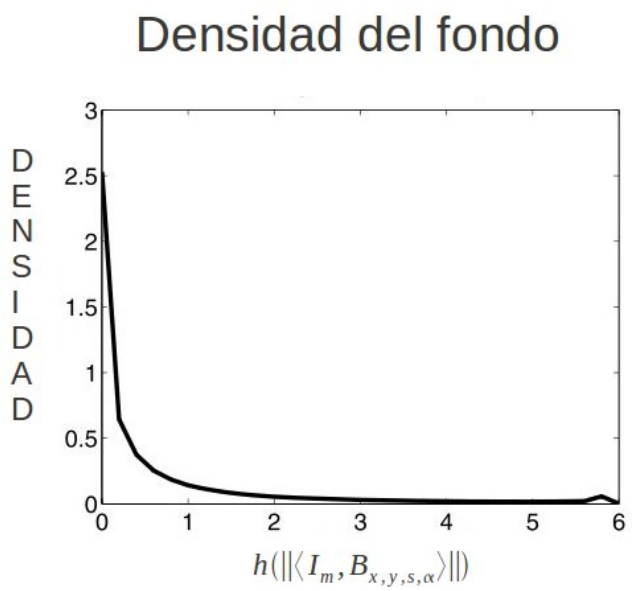


FIGURA 2.6: Histograma de la densidad de las respuestas sobre todos los valores de (x, y, s) de $h(|< I_m, B_{x,y,s,\alpha} >|)$ a partir de imágenes naturales, [WSGZ09].

Transformación y normalización

En esta subsección se explica por qué es necesario el uso de una sigmiode y la transformación suavizadora $h(r)$.

Sigmoide

Sea $p_{on}(r)$ la densidad de $r = |< I, B_{x,y,s,\alpha} >|^2$ cuando $B_{x,y,s,\alpha}$ está en un borde, y sea $p_{off}(r)$ la densidad de r cuando no está sobre un borde, asumiendo que $p_{on}(r)$ tiene una cola más larga que $p_{off}(r)$. Sea $q(r)$ y $p_i(r)$ las densidades de $r = |c|^2$ bajo

$q(c)$ y $p_i(r)$ respectivamente. Es razonable asumir $q(r) = (1-\rho_0)p_{off}(r)+\rho_0 p_{on}(r)$ y $p_i(r) = (1-\rho_i)p_{off}(r) + \rho p_{on}(r)$. Esto es, tanto $q(r)$ y $p_i(r)$ son mezclas de las mismas distribuciones p_{on} y p_{off}, con $\rho_i > \rho_0 > 0$. Cuando $r \to \infty$, $log\frac{p_i(r)}{q(r)} \to log(\frac{\rho_i}{\rho_0}) > 0$, esto es, converge a una constante positiva. Por lo que se puede asumir el siguiente modelo log-lineal: $log[p_i(c)/q(c)] = log(p_i(r)/q(r)) = \lambda_i h(r) + cte$, donde $\lambda_i > 0$ y $h(r)$ alcanza una saturación fija cuando $r \to \infty$. Esto justifica la saturación en la transformación de la sigmoide.

Normalización de *blanqueo*

Para hacer las respuestas de los filtros comparables entre diferentes imágenes de entrenamiento, es necesario normalizar estas. Sea

$$\sigma^2(s) = \frac{1}{|D|A} \sum_{\alpha} \sum_{(x,y)\in D} | < I, B_{x,y,s,\alpha} > |^2, \tag{2.38}$$

donde $|D|$ es el número de píxeles en I, y A es el número total de orientaciones. σ^2 mide el espectro de potencia de I alrededor de la frecuencia $1/s$. Para cada imagen de entrada se normaliza $| < I, B_{x,y,s,\alpha} > |^2$ cambiándola por $| < I, B_{x,y,s,\alpha} > |^2/\sigma^2(s)$. Ésta se le llama *normalización de blanqueo*, ya que hace que el espectro sea plano sobre s.

Transformación de *blanqueo*

La transformación de blanqueo acerca $q(I_m)$ a la distribución de ruido blanco, la cual es la hipótesis nula más simple. Ésta también conduce a una expresión explicita de $\mu(\lambda)$ y $\log Z(\lambda)$.
Sea $F(t) = q(r > t)$, esto es, la probabilidad de que $r > t$ bajo $q(r)$ o $q(I_m)$. La razón por la que $h(r) = -\log F(r)$ es llamada transformación de *blanqueo* es debido a que $Pr(h(r) > t) = Pr(-\log(F(r))) = P(F(r) < e^{-t}) = e^{-t}$, lo que significa que $h(r)$ sigue una distribución exponencial con una esperanza unitaria. Ésta es la distribución de r si $q(I_m)$ es ruido blanco gaussiano. Esto es el porqué la energía local de r es la suma de los cuadrados de las respuestas al seno y coseno de Gabor, y ambas siguen distribuciones gaussianas independientes si $q(I_m)$ es ruido blanco gaussiano. Por lo tanto la suma de sus cuadrados sigue una distribución χ^2, la cual es una distribución exponencial. La distribución tiene esperanza uno

ya que se normaliza la imagen para tener σ^2 unitaria, ver ecuación (2.38).
Esta transformación cambia una distribución con una cola larga $q(r)$ a una distribución exponencial con una cola corta. Con la transformación de *blanqueo*, asumiendo $p(c;\lambda) = \frac{1}{Z(\lambda)} exp(\lambda h(r)) q(c)$, $h(r) \approx Exp(1-\lambda)$, que es una distribución con $\mu(\lambda) = 1/(1-\lambda)$ y $Z(\lambda) = 1/(1-\lambda)$. Además, λ_i puede ser estimada por $\hat{\lambda} = 1 - M/(\sum_{m=1}^{M} h(r_{m,i}))$.

2.5.5. Síntesis sobre *active basis*

Para construir la representación *active basis* se asume que el fondo de las imágenes sigue una distribución q la cual se estima a partir de imágenes aleatorias. Una vez que se tiene la distribución de referencia y un conjunto de imágenes de ejemplo del objeto el cual se quiere modelar, se eligen un conjunto de filtros de Gabor, B_i, donde cada uno representa una subfamilia de filtros que son perturbaciones del filtro elegido. Estos filtros se eligen maximizando la divergencia de Kullback-Leibler entre la distribución de referencia y la distribución de las respuestas de los filtros, simplificando esta elección asumiendo que las distribuciones son de la forma de una distribución de la familia exponencial. Al final del proceso de entrenamiento se tiene un conjunto de filtros de Gabor que representan una plantilla del objeto y cada filtro junto con los parámetros de la distribución de cada uno de los filtros de la plantilla. Ya que se tiene la plantilla, ésta se usa para reconocer el objeto en una imagen nueva, maximizando la ecuación (2.34) usando un proceso llamado *SumMaxMaps*, el cual aplica la plantilla como si fuera un filtro de forma en cada posición de la imagen, teniendo como resultado la posición y un bosquejo del objeto.

2.6. Resumen

En este capítulo se presentaron algunos fundamentos acerca de probabilidad, conceptos relacionados con campos aleatorios de Markov, el reconocimiento de objetos y los filtros de Gabor que se usan como fundamentos de la tesis. Posteriormente se explica lo referente a los modelos *active basis*, usados en esta tesis, se revisaron los algoritmos existentes para *active basis* tanto el algoritmo *Shared Sketch* que se usa para aprender la representación como el algoritmo *SumMaxMaps* de reconocimiento, al final de la sección referente a *active basis*

se presentan los fundamentos teóricos usados en el desarrollo de los algoritmos existentes para la representación *active basis*.

Capítulo 3

Trabajo Relacionado

En este capítulo se discuten los trabajos relacionados con esta tesis, para ello este capítulo se estructurará de la siguiente manera. Primero se dará una taxonomía de cómo se estructurará el orden de los trabajos incluidos. Posteriormente se habla de los trabajos relacionados separados en tres secciones representación, aprendizaje y reconocimiento. Al final se dará un resumen acerca material contenido en este capítulo, en el cual se concluirá cómo se encuentra el estado de la tecnología actual en cuanto a reconocimiento de objetos deformables en condiciones variables.
Existen varios paradigmas usados para reconocer objetos, en esta revisión sólo se describirán aquellos que tengan alguna relación con este trabajo, con el fin de acotar el área de estudio. En este caso se asumirá que para reconocer un objeto en una imagen, el objeto requiere ser representado de alguna forma en la computadora, posteriormente se necesita una forma de aprender o entrenar los parámetros de dicha representación, resultando un modelo del objeto; y una vez que se tiene un modelo del objeto, éste se usará para reconocer el objeto en una imagen nueva.

3.1. Taxonomía

Siguiendo el paradigma mencionado para reconocer objetos, se presentarán los avances en cada uno de los pasos mencionados para reconocer un objeto, representación, aprendizaje y reconocimiento. En cuanto a la representación del objeto se mencionarán los trabajos acerca de cómo representar objetos y además se hablará de los avances en representaciones estructuradas. En cuanto al aprendizaje

se discutirán avances en aspectos importantes como la supervisión requerida para entrenar el modelo y el tipo de modelo que se aprende, que puede ser discriminativo o generativo. Finalmente, el último paso referente al reconocimiento del objeto en una imagen nueva, se discutirán los avances de los algoritmos existentes para algunos de los modelos descritos, también se hablará sobre los trabajos existentes respecto a la invariancia del algoritmo de reconocimiento.

3.2. Representación

Existen varias representaciones del objeto, en este caso sólo se mencionarán las relacionadas con descriptores locales, esto quiere decir las representaciones cuyos elementos describan localmente un objeto. Por una parte están las representaciones basadas en descriptores que dependen de respuestas a filtros. Por ejemplo, las representaciones basadas en puntos SIFT que representan al objeto a través de puntos de interés representado por un descriptor de características [Low04]. Otra forma de describir localmente un objeto en una imagen, que ha resultado bastante útil; es la representación basada en filtros de Gabor usada por los modelos *active basis*, propuesta por Wu et al., que representa el objeto como suma de filtros de Gabor, cada filtro puede ser *deformado* o perturbado localmente en orientación y posición permitiendo deformaciones en el objeto [WSGZ09]. Ésta representación ha sido extendida por Hu et al. para poder representar poses del objeto, quienes construyen un modelo 3D uniendo filtros de Gabor [HZ10].
Por otro lado está la representación usada por los *Active Apparence Models* (AAM), ésta representación expresa el objeto como un punto en un espacio vectorial cuya base son los eigenvectores obtenidos de la matriz de covarianza de un conjunto de descriptores de forma y textura del objeto [CET+01].

3.2.1. Representaciones jerárquicas

Una forma de representar los objetos es el basado en partes, esta forma de representar el objeto es una idea propuesta por Fischler y Elschlager, quienes describen que es posible describir un objeto como partes de este, manteniendo relaciones entre dichas partes [FE73]. Esta idea ha sido tomada por varios autores y lo que ha variado en los nuevos enfoques es el tipo de características que se usan y como se relacionan. Por ejemplo; Wu, B et al. usan este esquema construyendo

una jerarquía con partes compuestas por *pedazos de bordes*, aplicado a detección de peatones [WN05].
Otra variante para construir estructuras del modelo es construir gramáticas visuales, que es similar a las gramáticas formales salvo que en este caso se tienen elementos visuales; esta idea en su momento fue mencionada por David Marr [Mar82], y ha sido retomada por varios autores. Tal es el caso de los trabajos de L. Zhu et al. y de S. Zhu et al. , en estos trabajos se propone el uso de gramáticas visuales usualmente construidas a mano y posteriormente usar esta gramática para construir un árbol de *parseo* de la imagen con el fin de interpretar su contenido [ZCY07, ZLH+08, ZM06]. Este tipo de ideas han dado buenos resultados aunque aún tienen el problema de definir la gramática.

3.3. Aprendizaje

A partir de la representación se busca aprender un modelo que explique un conjunto de ejemplos del objeto. Para aprender este modelo existen dos paradigmas principales, los modelos discriminativos y los modelos generativos. Cada uno estima los parámetros de una distribución de los parámetros de la representación del objeto, los modelos discriminativos aprenden la distribución de $P(clase|Objeto)$, mientras que los modelos generativos $P(Objeto|clase)P(clase)$, en ambos casos el termino $Objeto$ es una representación del objeto y $clase$ puede ser SI o NO dependiendo si es o no el objeto o incluso el nombre del objeto. Ambos son útiles para reconocer un objeto, la diferencia es el resultado; los modelos generativos dan la posibilidad de generar ejemplos del objeto reconocido, mientras que los discriminativos sirven para decidir si es o no el objeto.
En el caso de los modelos generativos están los trabajos de Cootes et al. en donde muestran cómo construir una plantilla del objeto, de forma (modelos ASM) y apariencia (modelos AAM), a partir de etiquetas características en imágenes de ejemplo del objeto, tienen la desventaja de requerir dichas etiquetas, pues se eligen manualmente en la mayoría de los casos [CET+01, CTC+95]. Por otra parte está el trabajo de Zhu et al., quienes proponen cómo aprender los parámetros de la representación de los modelos *active basis* a partir de únicamente imágenes de ejemplo del objeto alineadas en escala y posición [WSGZ09]. Para los modelos *active basis* existen avances para reducir las restricciones en las imágenes de ejemplo requeridas para entrenar el modelo; Si, W et al. proponen

un algoritmo basado en maximización de la esperanza (EM) para aprender modelos *active basis* aun cuando el objeto en las imágenes de ejemplo no están alineadas, permitiendo que el objeto pueda estar en diferentes posiciones, escalas y orientaciones [SGZW09].
Por otro lado en cuanto a los modelos discriminativos, una de las técnicas más populares es una técnica basada en *boosting*, que consiste en construir una cascada de clasificadores; esta técnica, conocido como *AdaBoost* de Viola-Jones, consiste en construir una cascada de clasificadores a partir de las respuestas de imágenes de ejemplo del objeto a ciertos filtros [VJ01]. Este trabajo tiene la desventaja que para aprender un buen modelo del objeto se requieren un conjunto grande de imágenes de ejemplo y tiene la ventaja de que los modelos aprendidos son muy eficientes en el reconocimiento.

3.4. Reconocimiento o inferencia

En esta etapa existen trabajos diferentes para cada una de las posibles formas de reconocimiento, se listarán los trabajos relacionados sobre cómo se encuentran los parámetros de los modelos respectivos en imágenes nuevas del objeto.
Uno de los paradigmas más populares es la inferencia bayesiana sobre los datos observados conociendo el modelo, tal es el caso de Ommer et al. quienes usan inferencia bayesiana para unir las partes del objeto detectadas obtenidas usando algún clasificador [OB10].
Por otra parte está el algoritmo *SumMaxMaps*, el cual construye mapas a partir de respuestas a filtros, resultando al final un mapa que contiene una medida de que tan bien se ajusto un modelo a cada punto en la imagen. Este algoritmo es usado para hacer inferencia sobre la representación *active basis* [WSGZ09].

3.4.1. Invariancia

Un aspecto importante de cada modelo y su algoritmo de inferencia es la invariancia, esto es, bajo qué condiciones el algoritmo sigue *reconociendo* el objeto. Entre los principales aspectos que incluye la invariancia, están la invariancia a deformaciones como rotaciones y cambios de escala del objeto, y la invariancia a cambios de iluminación y oclusión.
En cuanto a algoritmos que son invariantes a deformaciones geométricas se tiene

el método de ajuste usado para AAM, propuesto por Cootes el a., quienes van moviendo los parámetros de deformación minimizando una medida de error [CET+01]. El algoritmo original se ha extendido para poder tolerar oclusión del objeto en la imagen donde se quiere reconocer, tal es el caso de la propuesta de Baker et al., quienes muestran resultados muy buenos para reconocer una persona con casi la mitad de la cara ocluida [BMX+04]. Existen trabajos para tolerar cambios en la iluminación como los trabajos de Ayala-Raggi et al., ellos recuperan el estado del iluminación del objeto, que en su caso son rostros, ajustando los parámetros de un modelo del estado de iluminación del objeto [ARARCE09].
En cuanto a trabajos relacionados directamente con oclusión y deformaciones geométricas, uno de los trabajos más recientes es el propuesto por Wu, T et al., quienes hacen un análisis de cómo afecta la oclusión y el cambio de escala al reconocimiento, y proponen que para resolver este problema hay que hacer inferencia con una gramática AND/OR, compuesta de partes del objeto modeladas con *active basis*, uniendo información de tres procesos que ellos llaman *alfa*, *beta*, *gama*, [WZ10]. Este trabajo tiene el problema que necesita la gramática para poder funcionar. Siguiendo esta misma línea de aplicar *active basis* y *SumMaxMaps* usando un modelo 3D es posible reconocer el objeto aún sin saber la vista del objeto en la imagen; W. Hu et al. muestran cómo poder resolver el problema de reconocer el objeto sin importar su pose o vista en la imagen [HZ10]. En este trabajo la forma de ajustar los modelos aprendidos les permite tolerar oclusiones generadas por la vista del objeto, aunque realmente no manejan el problema de que el objeto esté parcialmente ocluido.

3.5. Discusión

En esta sección se muestran tablas comparativas de los tres puntos mencionados. En cuanto a la representación en la tabla 3.1, se muestra la comparativa de algunas de las representaciones revisadas. En esta tabla se busca resaltar las ventajas que ofrece mantener una representación estructural, que es el caso de este trabajo. La mayor ventaja de tener una representación estructurada es la cantidad de información que es posible recuperar a partir de la estructura; sin embargo, la mayor desventaja es como construir dicha estructura que es algo que se busca resolver en esta tesis.

TABLA 3.1: Trabajos relacionados el tipo de representación del objeto o imagen

Clase	Trabajos	Pros	Contras
Características Locales	Sift	Robustos a escala y transformaciones afines	
	Filtros de Gabor	Son buenos descriptores de textura	
	Parches	Capturan textura	No hay modelo explicito de forma más allá de las relaciones espaciales
Estructurales	G. AND/OR[ZM06]	La cantidad de información que se puede recuperar de la imagen	Construir la gramática
	Wu, B[WN09]	Reconoce peatones uniendo muchos pedazos detectados	La forma de partir es muy sencilla y manual
	AB+estructura	No depende del objeto, tiene los beneficios de *Active Basis*(AB)	

Posteriormente se tiene la comparativa entre las diversas formas de aprender la representación, esta comparativa se muestra en la tabla 3.2. En este caso uno de los principales problemas es el grado de supervisión, teniendo como compromiso que tipo de modelo se construye y si es útil o no para hacer reconocimiento. En ese sentido se tiene que los algoritmos discriminativos ofrecen modelos que se pueden usar muy eficientemente en el reconocimiento, pero no se recupera mucha información a partir de estos modelos. Por otro lado, los modelos generativos ofrecen más ventajas en cuanto a la información que recuperan, siendo esta información de gran utilidad en el reconocimiento y aplicaciones posteriores. En cuanto al grado de supervisión, existen avances importantes que perminten aprender estos modelos con un nivel bajo de supervisión, tal es el caso de los modelos *active basis*.

Finalmente en cuando a la inferencia o reconocimiento, en la tabla 3.3 se muestran las ventajas y desventajas de los algunos algoritmos relacionados con la inferencia o reconocimiento del objeto en la imagen cuando este está en condiciones de oclusión y de iluminación variable. Como se muestra en la tabla, una de las principales desventajas de estos trabajos es que el modelo requerido usualmente requiere un nivel alto de supervisión. Este problema se busca resolver usando las extensiones

TABLA 3.2: Trabajos relacionados con el aprendizaje en la tarea de reconocimiento

Clase	Trabajos	Pros	Contras
M. generativos	AAM[CET+01]	Sintetizan el objeto	Nivel alto de supervisión
	AB[WSGZ09]	Representación compacta, permite deformar y separar el modelo, y requiere un nivel bajo de supervisión	No considera oclusión y cambios de iluminación
M. discriminativos	Adaboost[VJ01]	Muy rápidos	No recupera más información aparte de la clase, requieren muchos ejemplos

a los modelos *active basis*. Como se puede observar, los algoritmos propuestos en esta tesis presentan varias ventajas en el reconocimiento de objetos en condiciones de oclusión e iluminación variable.

3.6. Resumen

En este capítulo se describieron algunos de los principales trabajos sobre reconocimiento de objetos. Los trabajos se describieron en tres etapas: la representación, el aprendizaje y la inferencia. En cuanto a la representación se discutieron los avances que hay para representar un objeto, siendo las representaciones estructurales las que dan más ventajas. En cuanto al aprendizaje del modelo asociado a la representación se hablo de los modelos generativos y discriminativos, revisando además el nivel de supervisión requerido para cada algoritmo. En cuanto a la etapa de inferencia se discutieron los trabajos para reconocer el objeto en imágenes, se revisó además los trabajos referentes a la invariancia de los algoritmos de reconocimiento. Entre las tcnicas más completas relacionadas con el reconocimiento de objetos, se encuentra *active basis*, por sus ventajas en el aprendizaje e inferencia. Sin embargo, aún no resuelven el problema de reconocer objetos en condiciones oclusión e iluminación variable, por lo que este trabajo consiste en resolver dichas deficiencias.

TABLA 3.3: Trabajos relacionados con la inferencia en la tarea de reconocimiento

Clase	Trabajos	Pros	Contras
Inferencia	SumMaxMaps[WSGZ09]	Permiten inferir sobre información local de una manera muy sencilla	Lento comparado con AdaBoost
	PHAB[HDAR10]	Consideran el problema de oclusión, recupera un modelo parcial del objeto	No mejora mucho los porcentajes de detección respecto a *SumMaxMaps*
	SumMaxMaps+H	Recuperan el modelo aun en casos de iluminación variable	No recupera información de la iluminación
	SumMaxMaps+CAM	Recuperan el modelo, mejora las tasas de detección	Es lento en comparación con otros algoritmos
Invariancia	Baker,S[BMX+04]	AAM robustos a oclusión	No recupera información explicita de las partes ocluidas
	(IAAM)[ARARCE09]	Recupera el estado de iluminación	Requiere un conjunto de puntos elegidos manualmente para construir el modelo
	$(\alpha + \beta + \gamma)$[WZ10]	Precisión alta	Requiere gramática
	3D+2D+AB[HZ10]	Invariante a la pose, y recupera el ángulo de vista	Solo soporta oclusiones generadas por la pose

Capítulo 4

Algoritmo Propuesto

El problema que se quiere resolver es reconocer objetos deformables en condiciones variables, como oclusión e iluminación variable. Para ello, en este capítulo se describe como se propone una solución a este problema. Para explicar la solución este capítulo se organizó de la siguiente forma: primero se presentará el esquema que se propone para resolver el problema, posteriormente se dan un conjunto de observaciones usadas en la realización de los algoritmos y después se describirán los algoritmos propuestos; que son una extensión al algoritmo de aprendizaje *SharedSketch* usado para entrenar los modelos *active basis* y varias modificaciones al algoritmo de reconocimiento *SumMaxMaps*, que es el usado en la representación *active basis*.

4.1. Esquema propuesto

El procedimiento general para resolver el problema consiste en:

1. Dado un conjunto de imágenes con el objeto presente se construirá un modelo usando una representación, que en este caso será *active basis* [WSGZ09].
   Dado el modelo, éste será estructurado y se construirán relaciones entre los elementos de la representación de tal forma que el objeto este dividido en subpartes. Esto se puede ver gráficamente en la figura 4.1, donde la contribución está enmarcada y *SharedSketch* se refiere al algoritmo de aprendizaje descrito en el algoritmo 3.

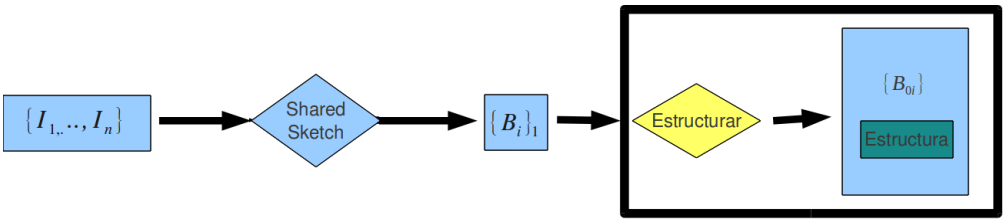


FIGURA 4.1: Esquema propuesto para extender el algoritmo de aprendizaje usado para *active basis*, esta extensión consiste en estructurar los elementos de la representación *active basis*

.

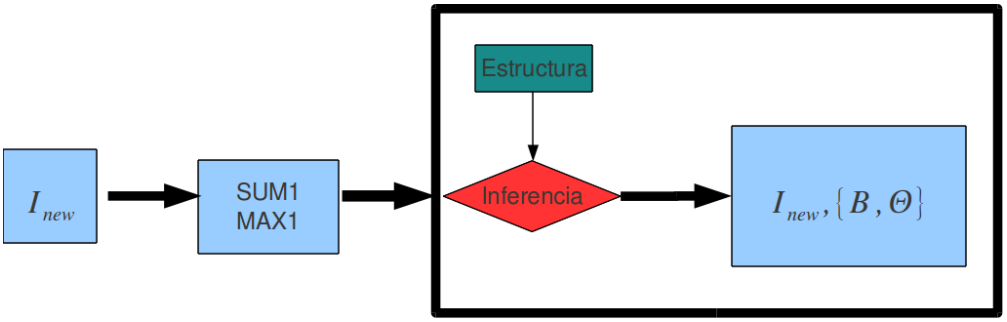


FIGURA 4.2: Esquema propuesto para modificar el algoritmo de *SumMaxMaps*, para reconocer un objeto en una imagen tomada en condiciones no controladas (oclusión e iluminación variable)

2. Después de que se estructuró el modelo, éste es usado para reconocer el objeto en una imagen nueva en una etapa de inferencia. Para ello se propone usar las primeras dos etapas del algoritmo *SumMaxMaps* y agregar una etapa nueva, la cual usa las estructuras aprendidas. Como resultado se obtienen los parámetros para cada elemento de la base del modelo ajustado a la imagen y la posición global del objeto en la imagen. Esto se puede ver gráficamente en la figura 4.2.

4.2. Observaciones e idea del método

El objetivo es resolver el problema de reconocimiento en el caso de ambientes variables, como lo son ambientes con oclusión e iluminación variable. Partiendo de estas premisas, se hacen las siguientes observaciones:

1. No tiene caso aprender todas las posibles formas de que un objeto pueda ser ocluido, pues no se sabe cómo puede aparecer parcialmente el objeto en la imagen.

2. Si se tiene el modelo estructurado por partes, es posible inferir su estado de visibilidad a partir de las partes en lugar de inferir sobre el modelo completo.

3. Se puede decidir qué elementos de la plantilla son más probables de estar presentes y cuáles son las que no están usando la función *Match*, descrita a continuación:

$$Match(I, B) = \sum_{i=0}^{n} \lambda_i h(| < I, B_i > |^2) - logZ(\lambda_i). \tag{4.1}$$

4. Es posible tener el modelo separado en partes del objeto a partir del modelo global aprendido, con el objetivo de no perder las relaciones espaciales aprendidas en la etapa de entrenamiento y poder usar la información local de cada parte para tener información del modelo global.

Los elementos anteriores llevaron a proponer un método basado en la idea de agrupar en *subpartes* del objeto los elementos B_i del modelo aprendido B.
Mientras que en la etapa de inferencia o reconocimiento se usará como base de los algoritmos la siguiente premisa. Se elige un subconjunto $\hat{H} \subset H$ donde $H = \{B_i \in B\}$, $\hat{H}$ representa el conjunto de elementos que si están presentes. Una vez que se tiene $\hat{H}$ y sus elementos agrupados es posible obtener información del estado del objeto que se está buscando.
Para obtener $\hat{H}$ se requiere eliminar los elementos no presentes de H; para decidir qué elementos no están presentes, se elegirán los elementos tales que su logaritmo de la verosimilitud sea menor o igual a cero, la construcción de este conjunto se puede ver de manera gráfica en la figura 4.3. Esto surgió al observar que si se quiere maximizar la ecuación (4.1) entonces los elementos que se deben eliminar son los ya mencionados. Quedando $\hat{H}$ formado con los elementos que el logaritmo de su verosimilitud sea mayor que cero. Este criterio en cada punto (x, y) de la imagen se puede ver de escribir como la siguiente expresión:

$$B_i \in \hat{H} \Leftrightarrow \lambda_i * MAX1MAP(x + x_i, y + y_i) - \log Z_i > 0 \tag{4.2}$$

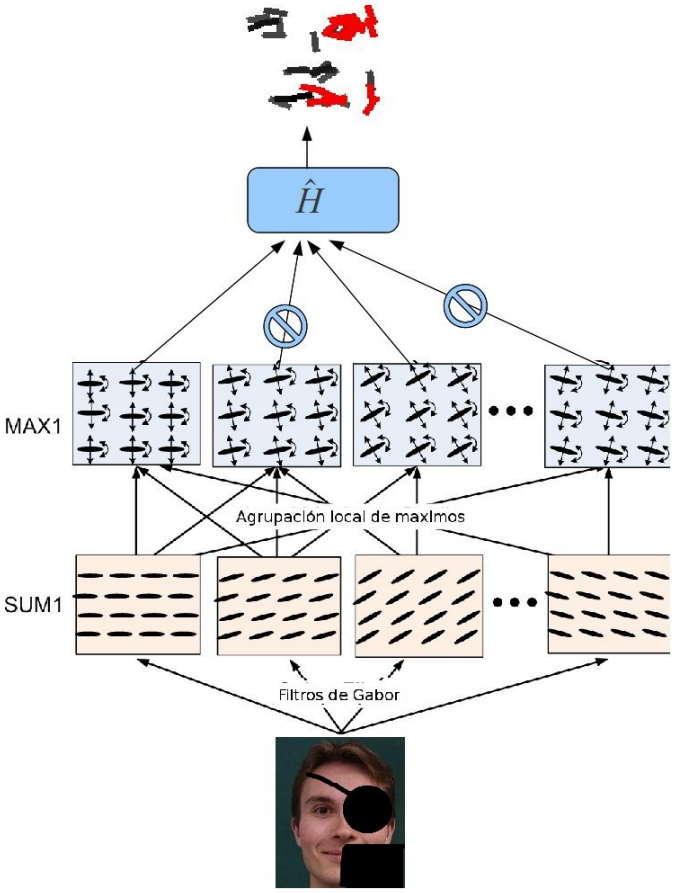


FIGURA 4.3: Se muestra como se construye el conjunto $\hat{H}$. Después de las primeras dos etapas del algoritmo SumMaxMaps, se elige en cada posición los elementos que cumplen con el criterio de la ecuación (4.2).

Simplemente sumar la verosimilitud de cada uno de los elementos de $\hat{H}$ no es suficiente, pues lo que encontraría son zonas con bordes fuertes que no necesariamente corresponden con el objeto que se está buscando[1].

4.3. Aprendizaje de la estructura

Para aprender la plantilla o modelo del objeto se uso como algoritmo de entrenamiento el propuesto por Wu et al. [WSGZ09], el cual se describió en el capítulo 2 algoritmo, 3. Una vez que se tienen los elementos de la base o plantilla del objeto con sus respectivos parámetros, estos se agrupan en cúmulos que representan una parte del objeto. En la figura 4.4 representa el modelo gráfico del modelo en partes.

[1]Este algoritmo se le denotará *simpleSUM* en los experimentos

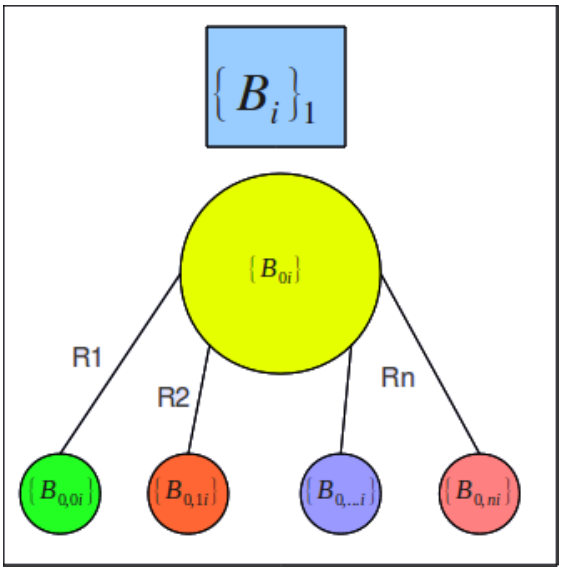


FIGURA 4.4: Ejemplificación del la estructura aprendida

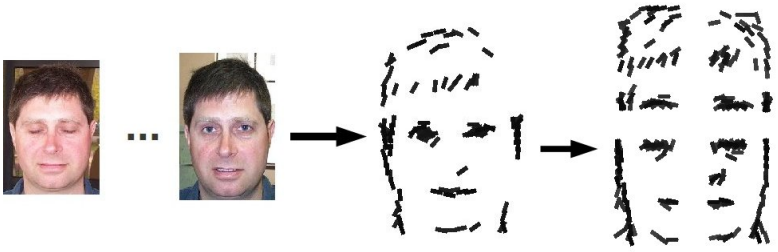


FIGURA 4.5: A partir de imágenes de ejemplo se construye el modelo del objeto usando el algoritmo *SharedSketch* y una vez dado el modelo, éste se secciona en varias partes.

Para poder hacer estos cúmulos de filtros de Gabor se propusieron varias formas de agrupar dichos elementos, con el fin de tener un algoritmo que sea independiente del objeto.

- Lo más sencillo fue partir el modelo en 4 partes, para ello se colocó el modelo en un rectángulo y se dividió. De esta forma los modelos generados son como los que se muestran en la figura 4.5.
- Observando que los elementos cercanos de las bases corresponden a *subpartes* del objeto, resulta natural pensar que si se agrupan los elementos, los cúmulos son las partes que resaltan del objeto. Para construir estos cúmulos es necesario definir una distancia entre los elementos. Considerando a cada elemento como un segmento se sugiere usar la distancia entre dos segmentos

como la distancia entre elementos de la base, dicha distancia está dada por:

$$d_{seg}(q_1, q_2) = min_{P \in q_1, Q \in q_2} d(P, Q), \tag{4.3}$$

donde q_1 y q_2 son dos segmentos y $d(P, Q)$ es la distancia euclidiana entre los dos puntos P y Q. Una vez que se tiene la distancia se procede de la siguiente forma:

1. Se toman dos elementos a y b de la plantilla.
2. Si los dos elementos a y b se encuentran a una distancia menor que algún umbral T fijo, se dice que a y b pertenecen a la misma subparte del objeto.
3. Se repite el paso 1 hasta que no haya modificaciones.

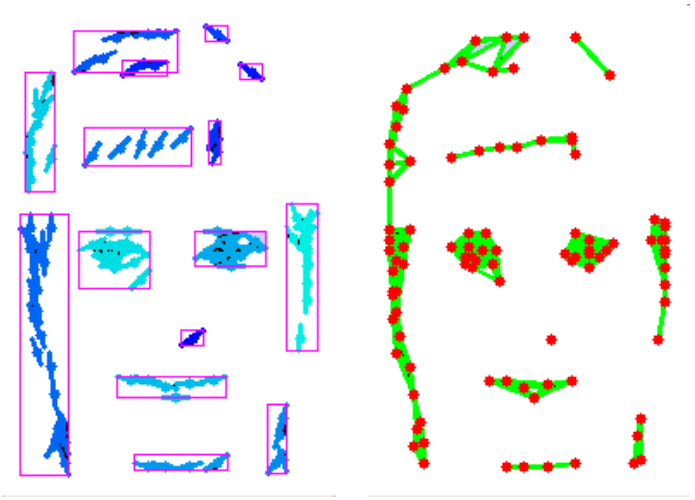
Los pasos anteriores se pueden hacer usando el algoritmo *Union-Find* [Cor01]. En la figura 4.6(a) se muestra como queda el modelo estructurado en partes.

- Siguiendo la idea anterior es posible construir un grafo con los elementos de la base. En este caso cada vértice corresponde a cada elemento de la base y tiene una posición dada por las coordenadas de dicho elemento, mientras que las aristas corresponden a elementos tales que su distancia es menor que algún umbral definido, usando la ecuación (4.3). La figura 4.6(b) muestra el grafo generado a partir de rostros.

El pseudocódigo que construye la estructura se muestra en el algoritmo 5, lo que hace este algoritmo es ir partiendo en pedazos el objeto en varios niveles, la función *split* en la línea 5 es la que parte el modelo en partes agrupadas del objeto. En el algoritmo 6 se muestra cómo se construye el grafo; lo que hace éste algoritmo es para cada dos elementos de la base, si estos se encuentran a una distancia menor que un umbral entonces se construye una arista entre los elementos.

4.4. Inferencia en el caso de oclusión parcial

Una vez que se tiene la estructura del modelo partido en varias partes agrupadas, las preguntas a responder serían dónde y cómo está el objeto, además, cuales



(a) Cúmulos generados usando la distancia (4.3)

(b) Grafo generado usando la distancia (4.3)

FIGURA 4.6: Estructuras a partir de modelos *active basis*

Algorithm 5 Aprendizaje y estructuración del modelo

Require: $Itrain = I_1, ..., I_n$

1: $\{B_i\}$ ←learn from Itrain at scale Scale, algorithm 3
2: **if** $level = max\_level$ **then**
3: **return** $\{B, \emptyset\}$
4: **else**
5: $Basis\_parts \leftarrow Split\{B_i\}$ using spatial relations
6: **for** $part \in Basis\_Parts$ **do**
7: **for** $img \in Itrain$ **do**
8: $nTrain \leftarrow \{nItrain\} \cup subimage(img, part.subWindow)$
9: **end for**
10: $\{part.B_i\} \leftarrow$ Learn-recursive-model using nItrain
11: $\{part.P(this|Parts)\} \leftarrow$ Probability of *this* part given that is a subpart
12: **end for**
13: $G \leftarrow DoGraph(\{B_{i,x,y,\theta}\}, graphThreshold)$(algorithm 6)
14: **return** $\{B, Basis\_parts, G\}$
15: **end if**

Algorithm 6 Construcción del grafo a partir de $\{B_{i,x,y,\theta}\}$

Require: $\{B_{i,x,y,\theta}\}, graphThreshold$
1: $Edges \leftarrow \emptyset$
2: $Vertices \leftarrow \emptyset$
3: **for** $i \in B$ **do**
4: $\quad Vertices \leftarrow Vertices \cup \{i\}$
5: $\quad$ **for** $j \in B$ **do**
6: $\quad\quad$ **if** $dist(i,j) \leq graphThreshold$ **then**
7: $\quad\quad\quad Edges \leftarrow Edges \cup (i,j)$
8: $\quad\quad$ **end if**
9: $\quad$ **end for**
10: **end for**
11: **return** $(Vertices, Edges)$

son los parámetros de la plantilla. Para contestar estas preguntas se modificó el algoritmo 4 de inferencia explicado en el capítulo 2 de varias formas, tratando de alcanzar un mejor desempeño.

4.4.1. Inferencia sobre las partes

El primer método consiste en unir las partes usando programación dinámica, esto es hacer un paso intermedio en el algoritmo *SumMaxMaps* después de la etapa $SUM1$. La idea para esto es: dados los elementos agrupados y cuales elementos de la base están presentes en la región donde se está reconociendo el objeto (el subconjunto $\hat{H}$), pasar esta información hacia una jerarquía de partes y unirlas hasta llegar al punto donde se tiene el objeto completo.

Algoritmo

Los pseudocódigos de los algoritmos son descritos en los algoritmos 7 y 8. En el algoritmo 7 se muestra la modificación del algoritmo *SumMaxMaps* descrito en el capítulo 2 algoritmo 4 al calcular *SUM2MAP*, mientras que en el algoritmo 8 se muestra cómo se hace la inferencia sobre los elementos de la plantilla que *aparecen* en la imagen. El algoritmo 7 lo que hace es construir el conjunto $\hat{H}$ en cada posición de la imagen, una vez que se tiene el conjunto $\hat{H}$ se pasa al algoritmo 8. Lo que hace el algoritmo 8 es primero unir los elementos en $\hat{H}$ en las partes del objeto dando una idea de si una parte del objeto está o no; posteriormente las partes del objeto se vuelven a unir en partes más generales del objeto hasta llegar al punto

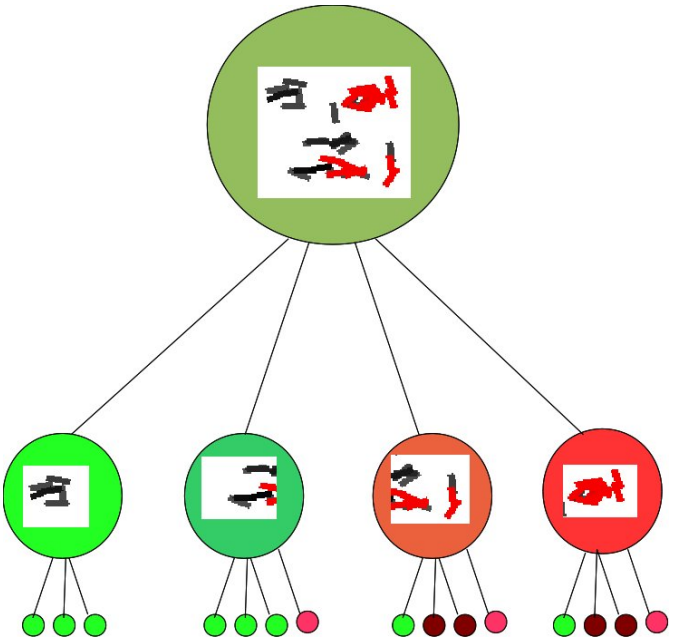


FIGURA 4.7: Se muestra como se unen las partes en el algoritmo *PHAB*, dado el conjunto $\hat{H}$ se agrupan en partes del objeto, luego cada parte pasa al objeto recuperando información acerca de su estado global.

donde se tiene el objeto completo, teniendo como resultado una medida de cómo está el objeto en la imagen en esa posición[2]. Una manera ilustrativa de cómo se unen las partes de un modelo del objeto partido una vez se muestra en la figura 4.7.

Algorithm 7 Computo de SUM2MAP para bases incompletas

for $x = 1$ to Width **do**
  for $y = 1$ to Height **do**
    for $t = 1$ to NumElements **do**
      if $\lambda_t * MAX1MAP(x + B_{t,x}, y + B_{t,y}, B_{t,\theta}) - \log Z_t > 0$ **then**
        $\hat{H} = \hat{H} \cup \{t\}$
      end if
    end for
    $SUM2MAP(x, y) \leftarrow solve(\hat{H})$(algorithm 8)
  end for
end for

[2]Este algoritmo será llamado *Active Basis Jerárquico por Parcial* y se denotara *PHAB* en la parte de experimentos.

Algorithm 8 Inferencia usando programación dinámica dado el conjunto $\hat{H}$

Require: $\hat{H}, Parts$
Initialize $Table$ to 0
$level \leftarrow lowest\_level$
for $t \in H$ **do**
  $Table[level][t] \leftarrow 1$
end for
$level \leftarrow level - 1$
while $level \geq 0$ **do**
  for $part \in Parts[level]$ **do**
    $prob \leftarrow 0$
    for $subpart \in part$ **do**
      $prob \leftarrow prob + Table[level + 1][subpart] * P(subpart|part)$
    end for
    $Table[level][part] \leftarrow prob$
  end for
  $level \leftarrow level - 1$
end while
return $Table[0][0]$

Análisis de complejidad

En el capítulo 2 se mencionó que la complejidad del algoritmo de inferencia *SumMaxMaps* es $O(d * NumElements)$. La modificación presentada implica hacer inferencia en cada posible posición, lo que tiene una complejidad de $O(NumElements * NumLevels)$ donde $NumLevels$ es el número de veces que se partió el modelo en submodelos. Por lo tanto la complejidad final es $O(d * (NumElements + NumElements * NumLevels))$.

4.4.2. Inferencia usando campos aleatorios de Markov

Una vez que se tiene la estructura y un grafo, el cual servirá como sistema de vecindades entre los elementos de las bases, se puede plantear el problema de reconocer el objeto incluso con oclusión como un problema de información incompleta, para ello las partes ocluidas se tratarán como información con ruido donde no se conoce su valor. Una forma de resolver este problema es plantearlo como un problema en un campo aleatorio de Markov (MFR) y tratar de predecir los valores *no* conocidos. Como ya se mencionó en el capítulo 2 para poder modelar

un campo aleatorio de Markov se necesita un potencial ligado a un sistema de vecindades para definir la distribución de Gibbs y por tanto tener un MRF.

Construcción del campo

Durante la construcción del modelo se asume que los elementos de la base son independientes entre sí, esto se hace para evitar traslapes entre los elementos y tener una función de ajuste fácil de calcular. Pero una vez que se tienen las bases se observa cómo algunos elementos juntos forman partes del objeto, por lo que se puede pensar que la independencia supuesta, en general, no es cierta. Intuitivamente esto resulta porque las respuestas sobre puntos en los bordes deben ser muy parecidas a lo largo del borde. Partiendo de esto el grafo que se construyó en la etapa de entrenamiento se puede usar como sistema de vecindades, y lo que se necesita definir es el potencial para el sistema de vecindades. El potencial que se usa se muestra en la siguiente ecuación:

$$U(x) = \sum_{k=1}^{N} U_c(k, x_k) - \lambda_{MRF} * \sum_{y \in V_k} U_v(y - x_k), \tag{4.4}$$

donde V_k es el conjunto de vecinos que están en el subconjunto $\hat{H}$, que son los elementos que se asume están presentes. U_c es el potencial *local* entre lo predicho y lo observado expresado como la ecuación (4.5) y U_v es el potencial de lo predicho y lo que se conoce de la vecindad el cual está dado por ecuación (4.6). Esto es,

$$U_c(k, x_k) = (H(k) - x_k)^2 \tag{4.5}$$

$$U_v(x) = (x)^2 \tag{4.6}$$

Para minimizar este potencial se itera la recurrencia dada por la ecuación siguiente:

$$x_t = x_{t-1} - \gamma \nabla U \tag{4.7}$$

Algoritmo

En los algoritmos 9 y 10 se muestra cómo se hace este procedimiento. El algoritmo 9 muestra la modificación al algoritmo *SumMaxMaps*, que agrega una etapa entre MAX1 y SUM2, esta etapa intermedia es la construcción del conjunto $\hat{H}$ y la

regularización del campo aleatorio de Markov. Esta regularización, descrita en el algoritmo 10, lo que hace es propagar la información contenida en el conjunto $\hat{H}$ al resto de los elementos vecinos que se desconoce su valor, esto se hace minimizando el potencial mencionado anteriormente usando un algoritmo de descenso de gradiente. Esta forma de resolver el problema será denotada como *SumMaxMaps+MRF* en el capítulo de experimentos.

Algorithm 9 Computo de SUM2MAP usando CAM

for $x = 1$ to Width **do**
  for $y = 1$ to Height **do**
    for $t = 1$ to NumElements **do**
      if $\lambda_t * MAX1MAP(x + B_{t,x}, y + B_{t,y}, B_{t,\theta}) - \log Z_t > 0$ **then**
        $H = H \cup \{(t, MAX1MAP(x + B_{t,x}, y + B_{t,y}, B_{t,\theta}))\}$
      else
        $H = H \cup \{(t, -\infty)\}$ Element not present
      end if
    end for
    $\hat{H} \leftarrow solveMRF(H)$ (algorithm 10)// Contains all the regularizated responces
    $SUM2MAP(x, y) \leftarrow \sum_{t=1}^{NumElements} \lambda_t * \hat{H}(t) - logZ_t$
  end for
end for

Algorithm 10 Optimización del campo

for $t = 1$ to NumElements **do**
  if $H(t).response \geq 0$ **then**
    $\delta(t) \leftarrow 1$
  else
    $\delta(t) \leftarrow 0$
  end if
end for
for $it = 1$ to Iterations **do**
  $X_{it} \leftarrow X_{it-1} + \gamma \nabla(U(X_{it-1}, \delta))$
end for
return $X_{Iterations}$

4.5. Inferencia en el caso de iluminación variable

En el caso del reconocimiento de objetos en condiciones de iluminación variable, en este trabajo sólo interesa saber la posición del objeto no el estado de iluminación

del objeto, dada esta premisa se propone el siguiente algoritmo, el cual se basa en encontrar los elementos faltantes usando histéresis.

4.5.1. Observaciones acerca del problema

Para plantear la solución al problema de reconocimiento se usaron las siguientes observaciones.

- Usar *active basis* es similar a aplicar un filtro de forma a un conjunto de bordes dados por la parte de maximización de las respuestas a los filtros de Gabor en ciertas orientaciones. De aquí surgió la idea de hacer algo similar a lo que hace el detector de bordes de Canny [Can87], el cual expande las posiciones donde las respuestas a ciertos filtros son máximas en dirección de los puntos vecinos en la imagen cuyas respuestas estén muy cerca al valor del máximo. De esta forma se pueden encontrar bordes *ocultos* por la iluminación.
- Ya que en este caso solo se tienen las respuestas a los filtros de Gabor en cierta orientación, la idea es hacer una especie de *histéresis* usando el grafo que se construyó en la etapa de aprendizaje, sobre los elementos de las bases que ya se encontraron (conjunto $\hat{H}$).

4.5.2. Algoritmo

Primero se construye un conjunto inicial $\hat{H}$ que representa los elementos máximos en el algoritmo de Canny, por lo que en este caso el papel de los máximos lo tendrán los elementos que el logaritmo de su verosimilitud sea mayor que un umbral establecido. Una vez que ya se construyo $\hat{H}$ lo que ahora sigue es agregar elementos de la base cuyas respuestas a la imagen sean *cercanas* al nodo actual, para ello se hace una búsqueda en profundidad o en amplitud visitando los vecinos de los elementos que ya se encuentran en $\hat{H}$. Después de haber agregado los nuevos elementos al conjunto $\hat{H}$ se aplica nuevamente la inferencia por partes. Los algoritmos 11 y 12 muestran como se hace esto. Específicamente el algoritmo 11 dado un conjunto de elementos iníciales $\hat{H}$ explora el modelo visto como un grafo, visitando los vecinos cuyas respuestas cumplan que son cercanas al elemento actual de la búsqueda. En cuanto al algoritmo 12 lo que hace es construir un

conjunto inicial $\hat{H}$ y pasarlo al algoritmo de búsqueda, una vez que se agregaron elementos al conjunto $\hat{H}$ este se pasa al algoritmo de inferencia por partes, que trata el problema como un problema de oclusión parcial.

Algorithm 11 Inferencia con Histéresis

```
1: Queue ← Ĥ
2: while Queue not empty do
3:    b ← Queue.pop
4:    for v ∈ Neiborhood(b) do
5:       if |response(b) − response(v)| ≤ TOL and response(v) ≥ LOW then
6:          Q.push(v)
7:          Ĥ ← Ĥ ∪ {v}
8:       end if
9:    end for
10: end while
11: return Ĥ
```

Algorithm 12 Cálculo de SUM2MAP para iluminación variable usando histéresis

```
1: for x = 1 to Width do
2:    for y = 1 to Height do
3:       for t = 1 to NumElements do
4:          if λ_t * MAX1MAP(x + B_{t,x}, y + B_{t,y}, B_{t,θ}) − log Z_t > 0 then
5:             Ĥ = Ĥ ∪ {(t, MAX1MAP(x + B_{t,x}, y + B_{t,y}, B_{t,θ}))}
6:          end if
7:       end for
8:       Ĥ ← solveHysteresis(Ĥ) (algorithm 11)
9:       SUM2MAP(x, y) ← solve(Ĥ)(algorithm 8)
10:   end for
11: end for
```

4.5.3. Análisis de complejidad

La complejidad es $O(D*(numElements+numElements+E+NumElements*NumLevels))$, donde E es el número de aristas que se tienen en el modelo. Esto se obtiene a partir de aplicar el filtro de forma en cada píxel y en cada píxel se hace una búsqueda en amplitud o profundidad.

4.6. Resumen

En este capítulo se presento un esquema para resolver el problema de reconocer un objeto en condiciones no controladas. Posteriormente se dieron observaciones para construir un modelo el cual se uso para resolver el problema de reconocimiento en condiciones variables, además se explico cómo construir un conjunto $\hat{H}$ el cual representa los elementos de la plantilla que están presentes en la imagen. En cuanto a los algoritmos se presentaron varias formas de estructurar el modelo entrenado usando la representación *active basis*, teniendo como resultado relaciones espaciales y topológicas. Estas relaciones fueron usadas para reconocer objetos en condiciones de oclusión parcial e iluminación variable. Para oclusión se presentaron dos modificaciones al algoritmo *SumMaxMaps*, una usando programación dinámica para unir partes del objeto y otra que consistía en estimar los valores de los elementos no presentes de la plantilla usando campos aleatorios de Markov. Para iluminación variable, se presentó una forma de reconocer objetos en condiciones de iluminación variable usando histéresis para recuperar los elementos de la plantilla que fueron afectados por el cambio en la iluminación cuando se capturo la imagen.

Capítulo 5

Experimentos

En este capítulo se presentan los experimentos realizados para validar los algoritmos propuestos en el capítulo anterior. Primero se menciona como fueron implementados los algoritmos ya mencionados y como se midieron los resultados obtenidos. Posteriormente se describe cada uno de los experimentos; para ello se menciona en qué consiste el experimento, con qué objetivo se hizo y cuáles son los resultados obtenidos. Finalmente se da una discusión general sobre los resultados obtenidos.

5.1. Detalles de la implementación

Todos los algoritmos descritos en el capítulo 4 fueron implementados en C++ usando las librerías publicas *OpenCV*[1] y *Boost*[2]; se uso como base el código disponible en la página de los autores que propusieron la representación *active basis*[3], estos códigos fueron modificados y extendidos para poder hacer lo que se propone en esta tesis, y está descrito en el capítulo 4.
Todos los algoritmos fueron ejecutados en la misma computadora y compilados de la misma forma. A continuación se muestran las características de la computadora y del compilador:

- Sistema operativo: Ubuntu 10.04 32 bits.

[1] http://sourceforge.net/projects/opencvlibrary/
[2] http://www.boost.org/
[3] http://www.stat.ucla.edu/~ywu/AB/active_basis_cpp.html

- Versión del Kernel: 2.6.32-21-generic
- Memoria RAM: 3GB.
- Procesador: Intel Core 2 DUO E7200 2.53 GHz.
- Compilador: g++ versión 4.4.3.
- Opciones de optimización del compilador: -O2

En cuanto a los parámetros de la distribución de referencia q, mencionada en el capítulo 2, subsección 2.5.2; se estimaron una sola vez a partir de un conjunto de 475 imágenes aleatorias tomadas del conjunto de datos proporcionados por los autores de la representación *active basis*, los parámetros calculados se usaron para todos los experimentos mencionados en este capítulo.

5.2. Tasa de detección

Para medir el rendimiento de los algoritmos se calculó la tasa de detección en un conjunto de imágenes que contiene al objeto, midiendo si el algoritmo encontraba correctamente el objeto en la imagen. Para esto, tomando el resultado arrojado por los algoritmos de reconocimiento, tanto el algoritmo *SumMaxMaps* como las modificaciones propuestas, se construyó el menor rectángulo que envuelve completamente los elementos de la plantilla; este se denotará como B^c. El rectángulo B^c se comparó con el rectángulo proporcionado por las bases de datos de donde se tomaron las imágenes de prueba, denotado por B^{gt}. Para decir que fue correctamente detectado el objeto se uso el siguiente criterio:

$$B^c \; es \; correcto \; \Leftrightarrow \; \frac{A(B^c \cap B^{gt})}{A(B^c \cup B^{gt})} \geq \frac{1}{2}, \tag{5.1}$$

donde $A(\bullet)$ denota el área del rectángulo.
Finalmente para calcular la tasa de detección primero se cuenta en la variable *positivos* el número de imágenes donde el criterio anterior se considera correcto y en la variable *negativos* las que se consideran incorrectas; y posteriormente se aplica la siguiente expresión:

$$tasa \; de \; deteccion = \frac{positivos}{positivos + negativos}. \tag{5.2}$$

5.3. Experimento: Reconocimiento de rostros ocluidos

El primer experimento consiste en detectar rostros ocluidos de manera sintética a diferentes niveles de oclusión usando los siguientes algoritmos: *PHAB*, descrito en el capítulo 4 subsección 5.3; *SumMaxMaps*, descrito en el capítulo 2 subsección 2.5.3 y *simpleSum* mencionado en el capítulo 2, subsección 4.2. Este experimento se hizo con el objetivo de medir el comportamiento de los algoritmos cuando se ocluye el objeto cada vez más con objetos cuya textura puede generar bordes.

5.3.1. Diseño del experimento

Este experimento se hizo con 50 imágenes de la clase *faces* de la base de datos Caltech 101 [FFFP06], estas imágenes fueron modificadas agregando oclusión sintética a las imágenes. La oclusión fue generada por objetos que tienen una textura no lisa a diferentes niveles, los niveles de oclusión se refieren al porcentaje de píxeles del objeto que son ocluidos, de manera que el nivel 0 es la imagen original, el nivel 1 representa el 10 % de oclusión del objeto, el nivel 2 es 20 % y así sucesivamente.
En cuanto a los detalles de la configuración del modelo, se construyó un modelo con 100 elementos (filtros de Gabor) aprendidos de un conjunto de rostros sin oclusión. La configuración para los elementos de las bases como se menciona en la sección (2.26), es la siguiente:

- Variabilidad de α: 3
- Variabilidad de d: 3
- Número de orientaciones: 15
- Número de elementos activos: 100

Cada algoritmo se corrió 10 veces, sobre 10 diferentes subconjuntos de entrenamiento y prueba extraídos del mismo conjunto. Para cada partición se eligieron 50 % de las imágenes para entrenamiento y 50 % para prueba. El conjunto de entrenamiento contenía únicamente imágenes no ocluidas y las imágenes eran los

rostros acotados dentro de los rectángulos que proporciona la base de datos. El conjunto de prueba consistió del resto de las imágenes no elegidas para el entrenamiento en sus versiones con oclusión sintética a diferentes niveles. En la tabla 5.1, en el primer renglón se muestran ejemplos a diferentes niveles de oclusión de las imágenes usadas.

5.3.2. Resultados

En los resultados que se muestran en la gráfica de la figura 5.1 se puede observar que el algoritmo propuesto *PHAB* supera al algoritmo *SumMaxMaps* cuando el objeto se ocluye cada vez más. En la tabla 5.1 se muestran los bosquejos de las imágenes reconocidas bajo diferentes niveles de oclusión. En la tabla 5.1, en el tercer renglón se puede apreciar que el modelo no se recupero totalmente pues las *manos* que ocluyen el rostro no lo permiten. En el apéndice A sección A.1 se muestran las tablas de los resultados numéricos y los resultados de las pruebas de significancia estadística.
De los resultados se puede decir lo siguiente:

- La modificación *simpleSum* se comportó como se esperaba, pues esta modificación al algoritmo *SumMaxMaps* al usar únicamente los elementos de $\hat{H}$ confunde el objeto fácilmente con zonas que tienen bordes *fuertes* como árboles, nubes y otros. Este algoritmo únicamente reconoce el objeto en imágenes donde el fondo no tenía texturas.

- Agrupar los elementos y hacer inferencia sobre el conjunto $\hat{H}$ que contiene los elementos de la base *presentes* es significativamente mejor que sumar su verosimilitud (algoritmo *simpleSum*).

- Comparando los algoritmos *SumMaxMaps* con *PHAB* se tiene una mejora en la tasa de detección, la cual en la mitad de los experimentos no es significativa estadísticamente. Pero el algoritmo *PHAB* tiene el beneficio de poder tener un bosquejo parcial del objeto, dando información sobre las partes que no están.

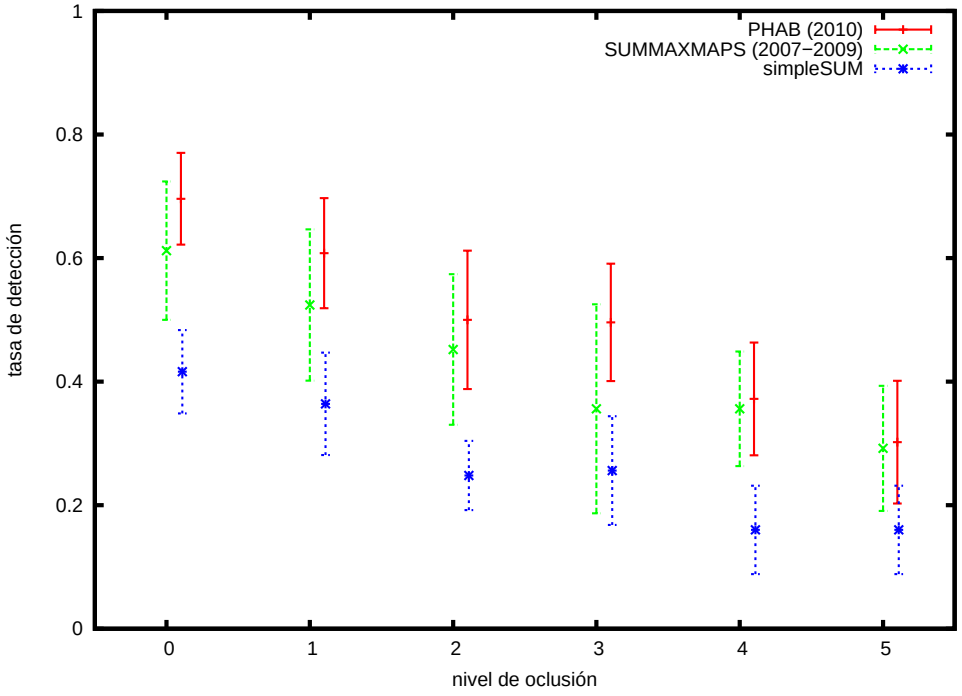
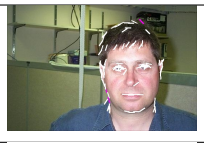
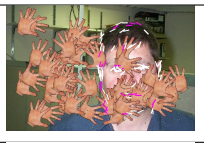
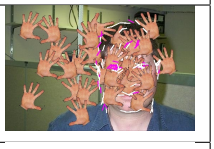


FIGURA 5.1: La gráfica muestra los resultados al correr 10 veces el algoritmo *PHAB* y *SumMaxMaps* con 10 conjuntos diferentes de entrenamiento y prueba, se muestra la media y la desviación estándar de las 10 corridas.

TABLA 5.1: Ejemplos de imágenes con un incremento en el nivel de oclusión. En el primer renglón se muestran las imágenes de prueba, en el segundo la imagen con el bosquejo y las predicciones de los elementos no presentes, y en el tercer renglón los bosquejos recuperados.

	Nivel 0	Nivel 1	Nivel 2	Nivel 3	Nivel 4
Img ocluida					
Bosquejo					
					

5.4. Experimento: Reconocimiento de rostros ocluidos usando MRF

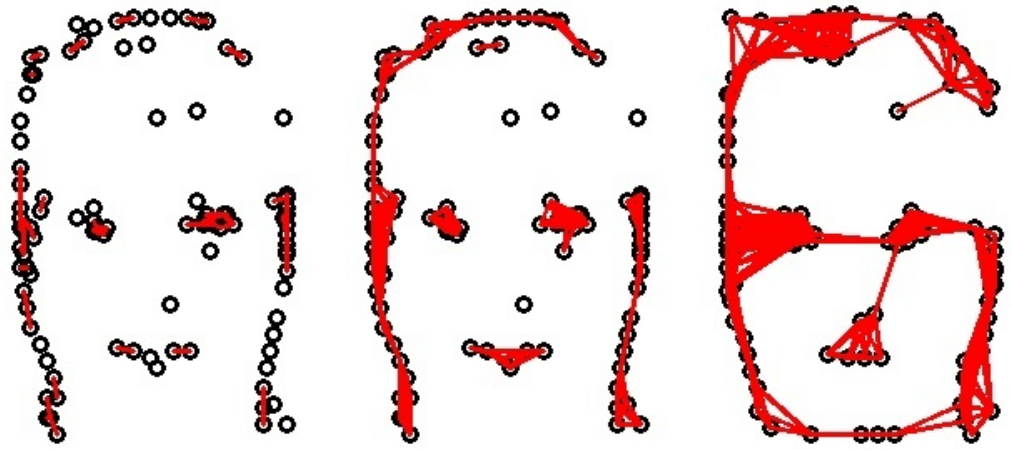
En este experimento se usó el algoritmo utilizando campos aleatorios de Markov, descrito en los algoritmos 9 y 10 en el capítulo 4; y se comparó con el algoritmo *SumMaxMaps*.

5.4.1. Diseño del experimento

El diseño del experimento es similar al descrito en la sección 5.3, salvo que se hicieron experimentos para diferentes valores del parámetro *graphThreshold* que influye en la construcción del grafo, que se usa como sistema de vecindades en el algoritmo. Este parámetro aparece en la parte de resultados como $U = X$ donde U representa a $4X$ pixeles de distancia. Se corrió el algoritmo 10 veces con 10 diferentes cortes en el conjunto de prueba y entrenamiento al igual que el experimento anterior. De forma análoga que el experimento anterior se usaron 100 elementos (filtros de Gabor), los parámetros de variabilidad del cada elemento fueron iguales al experimento anterior. Este experimento se presenta de manera independiente para poder mostrar los resultados únicamente al variar el parámetro de construcción del grafo.

5.4.2. Resultados

La gráfica de la figura 5.3 muestra los resultados al correr el algoritmo *SumMaxMaps* usando campos aleatorios de Markov variando el parámetro para construir el grafo, denotado en la gráfica como *SumMaxMaps+MRF(U=X)*. En la gráfica se puede ver una tendencia favorable al algoritmo *SumMaxMaps+MRF* comparado con el algoritmo *SumMaxMaps*, mientras que en la figura 5.2 se muestran los grafos construidos con los diferentes parámetros probados usados en este experimento. En el apéndice A se muestran las tablas con los resultados numéricos al aplicar estos algoritmos.
A partir de este resultado se puede decir que la información de los vecinos favorece a la integración de la información del modelo. De los resultados obtenidos al variar el parámetro de construcción del grafo se puede afirmar lo siguiente:



(a) (b) (c)

FIGURA 5.2: a) Representa el grafo al correr el algoritmo 6 con el parámetro graphThreshold=4 pix, b) representa el grafo al correr el algoritmo 6 con el parámetro graphThreshold=12 pix y c) representa el grafo al correr el algoritmo 6 con el parámetro graphThreshold=144 pix

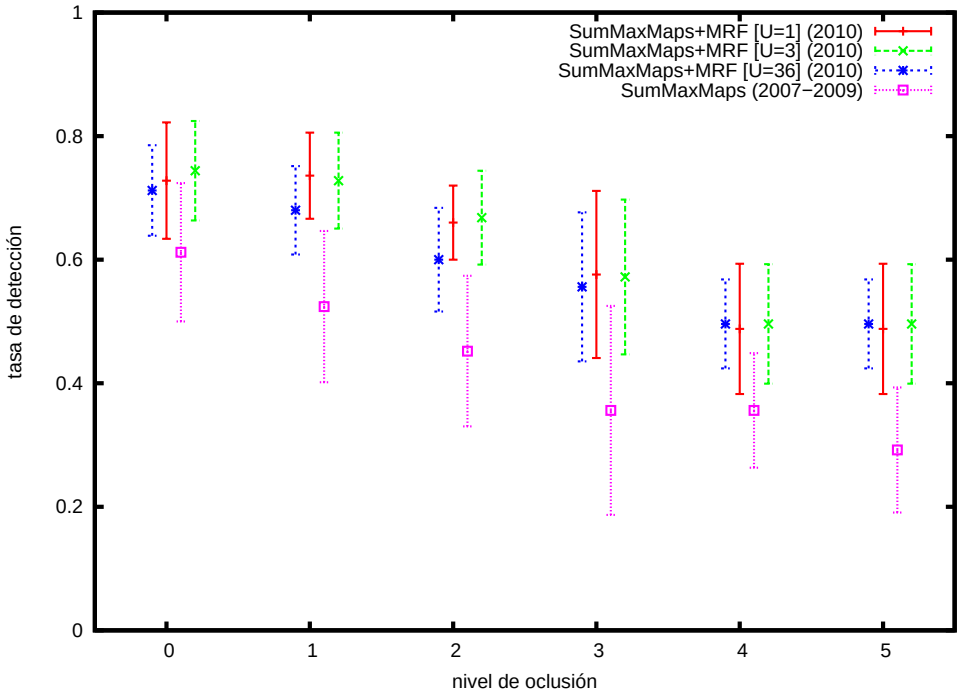


FIGURA 5.3: La gráfica muestra los resultados que da el algoritmo usando campos aleatorios de Markov.

- Cuando el nivel de oclusión es *bajo*, el usar grafos construidos con un parámetro U correspondiente a 1 y 3 supera al algoritmo usando el grafo construido con el parámetro $U = 36$. Esto se debe a que la información de los vecinos no se requiere tanto como cuando el objeto está más ocluido.
- Cuando el objeto está muy ocluido (niveles 3 en adelante), el algoritmo funciona prácticamente igual, esto se debe a que el conjunto $\hat{H}$ está prácticamente vacío, teniendo pocos elementos para propagar información.

5.5. Experimento: Reconocimiento de autos ocluidos

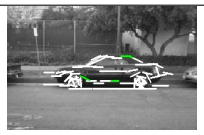
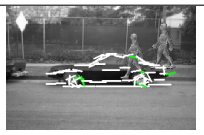
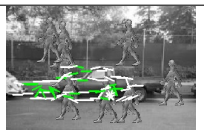
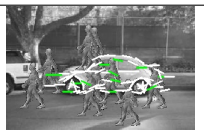
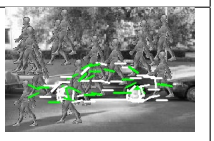
En este experimento se hacen pruebas con objetos diferentes a rostros, en este caso se experimentó con autos tomados de manera lateral obtenidos de la subclase *car-side* del repositorio Caltech 101 [FFFP06]; al igual que en el caso de rostros se ocluyeron las imágenes con objetos ajenos a las imágenes.

5.5.1. Diseño del experimento

Este experimento se realizo con un conjunto de imágenes de autos, el cual constaba de 100 imágenes de autos estacionados lateralmente. Se corrió el algoritmo 10 veces sobre 10 diferentes conjuntos de entrenamiento y prueba extraídos del mismo conjunto. Análogamente que con rostros se aplico el algoritmo a las imágenes con 5 niveles de oclusión, los cuales fueron generados artificialmente. De la misma forma que en rostros el nivel 0 representa la imagen original, el nivel 1 representa el objeto ocluido menos del 10 % y así sucesivamente.
La configuración para los elementos de las bases como se menciona en la sección (2.26) es la siguiente:

- Variabilidad de α: 3
- Variabilidad de d: 3
- Número de orientaciones: 15
- Número de elementos activos: 50

TABLA 5.2: Ejemplos de imágenes con un incremento en el nivel de oclusión.

	Nivel 0	Nivel 1	Nivel 2	Nivel 3	Nivel 4
Img ocluida					
Bosquejo					

5.5.2. Resultados

Los resultados se muestran en la gráfica de la figura 5.4, donde se puede observar que a pesar de que ambos algoritmos tienen porcentajes de detección bastante altos el algoritmo por partes supera al algoritmo *SumMaxMaps*. En el segundo renglón de la tabla 5.2, se muestra el *bosquejo* encontrado por el algoritmo *PHAB* aplicado a cada una de las imágenes del primer renglón. En el apéndice A se muestran los resultados de las tasas de detección y los resultados de las pruebas de significancia estadística respecto al algoritmo *SumMaxMaps*.

Se puede decir que los valores grandes de la varianza reportada para estos resultados se debe a que el conjunto de imágenes de prueba contiene autos, como camionetas, que son significativamente diferentes a los autos *promedio*, además, si estas imágenes con camionetas son ocluidas en las partes que son similares a los autos *promedio*, es muy difícil que el algoritmo encuentre alguna parte que pueda usar en la inferencia.

5.6. Experimento: Reconocimiento de rostros bajo iluminación variable

Se realizó este experimento para investigar el comportamiento del algoritmo descrito en la sección 4.5 bajo condiciones de iluminación variable, el cual usa histéresis para completar los elementos del modelo.

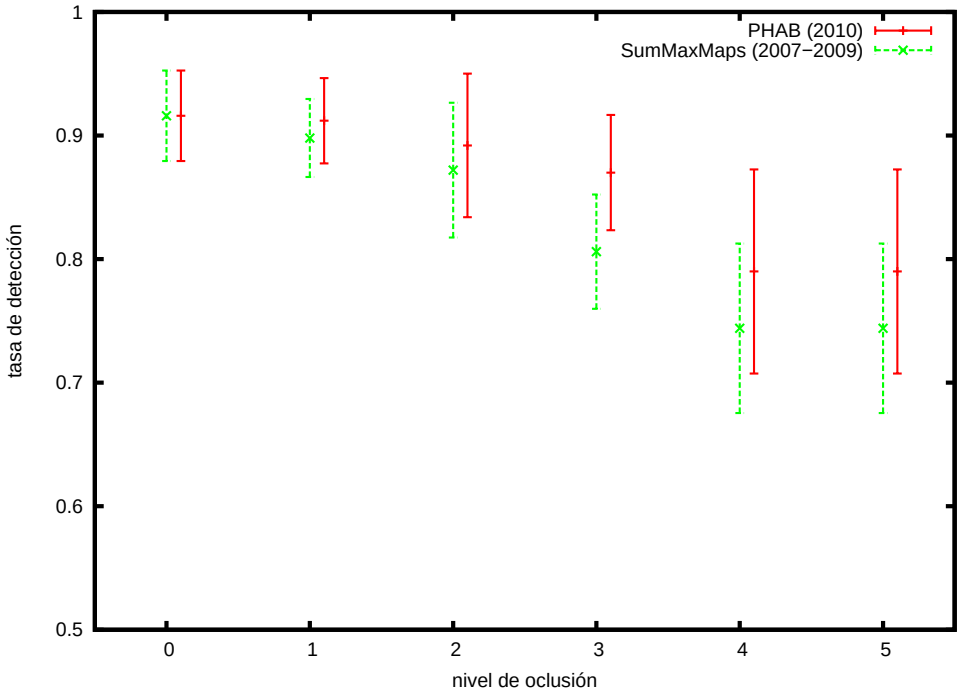


FIGURA 5.4: La gráfica muestra los resultados al correr 10 veces el algoritmo *PHAB* y *SumMaxMaps* con 10 conjuntos diferentes de entrenamiento y prueba, se muestra la media y la desviación estándar de las 10 corridas.

5.6.1. Diseño del experimento

En este experimento se usaron los rostros de la base de datos *The Yale Face Database B* [GBK00], que contiene imágenes tomadas con iluminación variable. En este caso el modelo usado se entreno con imágenes de la misma persona, además estas imágenes se eligieron manualmente de tal forma que la iluminación fuera de frente para que no hubiera sombras que el modelo pudiera aprender. Algunos ejemplos de estas imágenes se pueden observar en la figura 5.5. El conjunto de prueba consistió de un conjunto con 50 imágenes diferentes a las que se usaron para entrenar el modelo. Una vez que se construyó el modelo, se usó para reconocer el objeto variando los parámetros de la histéresis en el algoritmo 4.5. Cada nivel de histéresis que se muestra en la figura 5.6 corresponde a un decremento de 0.5 en el umbral inferior (la variable *LOW* en la línea 5 del algoritmo 11), esto se puede

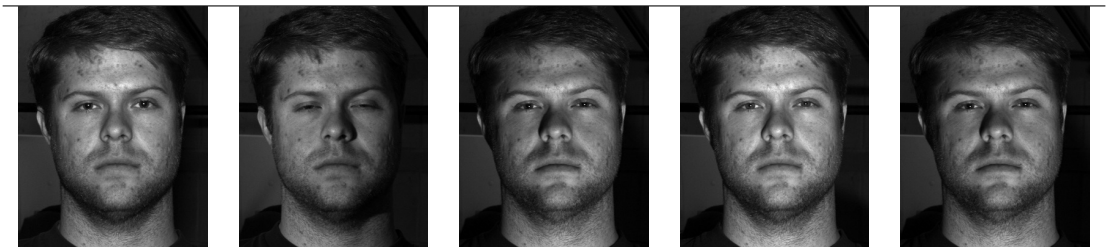


FIGURA 5.5: Subconjunto de imágenes elegidas para entrenar. Se eligieron manualmente de tal forma que "visiblemente"tuvieran una iluminación uniforme sobre el rostro

expresar de la siguiente forma

$$B_i \in \hat{H} \Leftrightarrow \lambda_i MAX1MAP(x + x_i, y + y_i) - \log Z_i \in [-.5x, 0] \tag{5.3}$$

y que además cada elemento B_i sea encontrado por la búsqueda en amplitud.

5.6.2. Resultados

Los resultados se muestran en la gráfica de la figura 5.6, donde la línea recta es la referencia, esta recta se obtiene a partir de los resultados del algoritmo *SumMaxMaps*[WSGZ09]. Se puede observar que el algoritmo que usa histéresis supera al algoritmo *SumMaxMaps* cuando se cambian los rangos de histéresis permitiendo que se expanda más la búsqueda, lo cual es algo que se espera, ya que la búsqueda va agregando elementos a $\hat{H}$ teniendo más información para la inferencia. La figura 5.7 muestra los resultados al aplicar el algoritmo con histéresis con dos diferentes parámetros en los umbrales de histéresis.

5.7. Discusión

En cuanto a los resultados de detección de rostros ocluidos se puede observar que el algoritmo que usa campos aleatorios de Markov presenta una mejoría notable en la tasa detección respecto al algoritmo de inferencia por partes[4], en este caso, esto se debe a que los campos aleatorios completan la información de sus vecinos,

[4]En las tablas del apéndice A se presentan las tasas de detección de manera cuantitativa.

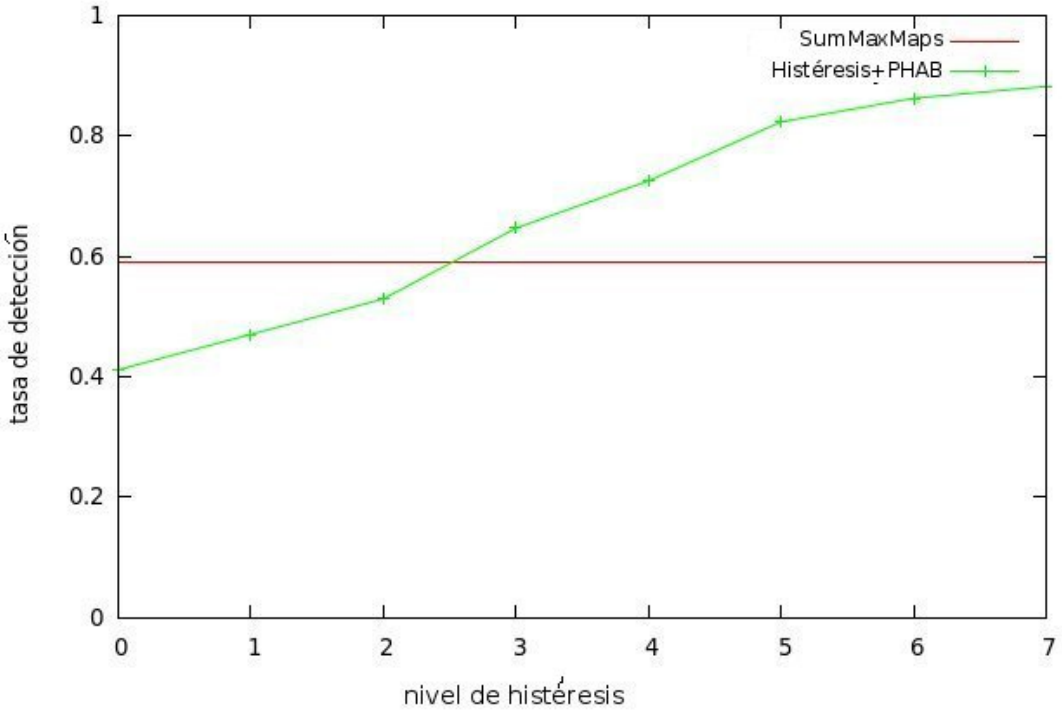


FIGURA 5.6: Resultados de detección en la base de datos *The Yale Face Database B*. La recta es la línea de referencia usando el algoritmo *SumMaxMaps*, mientras que la curva muestra las tasas de detección al variar los parámetros de histéresis.

(a) (b)

FIGURA 5.7: Ejemplos de imágenes al correr el mismo algoritmo con diferentes parámetros de histéresis, se puede observar que al incrementar los rangos el conjunto $\hat{H}$ tiene mas elementos y por tanto se puede inferir mas acerca del objeto.

lo que favorece la inferencia. La desventaja es el aumento de la complejidad computacional de maximizar el campo, pues maximizar el campo únicamente es más lento respecto de unir las partes como lo hace el algoritmo *PHAB*.
En cuanto a detección de autos, las tasas de detección son altos para ambos algoritmos, llegando a tener tasas de deteccion arriba del 90 %, esto está relacionado con el ambiente en el que están las imágenes, pues aunque se incluyó oclusión, el resto del ambiente no genera formas que sean similares a autos, lo que si pasa en el caso de los rostros.
Con respecto al algoritmo que usa histéresis, los resultados obtenidos muestran que al aumentar los rangos que permiten agregar elementos al conjunto $\hat{H}$ durante la búsqueda en profundidad que hace la histéresis, la tasa de detección aumenta. Este resultado ya se esperaba puesto que el algoritmo que hace la inferencia final tiene más información para estimar que partes del objeto están.
La principal limitación de las extensiones propuestas es la complejidad computacional respecto al algoritmo *SumMaxMaps*, ya que esta aumenta al agregar las extensiones descritas en el capítulo 4. Cabe mencionar que cada uno de los algoritmos puede ser optimizado para mejorar los tiempos de ejecución, pero esto último queda fuera de los objetivos de este trabajo.

5.8. Resumen

En este capítulo se presentaron varios experimentos medir como se comportaban las modificaciones al algoritmo *SumMaxMaps*. Estos experimentos fueron los siguientes:

- Usando una base de datos con rostros ocluidos artificialmente, se compararon los algoritmos *PHAB* y *SumMaxMaps*, además se presentaron los resultados al correr el algoritmo *simpleSum*, estos resultados se presentan con el propósito de analizar el comportamiento si no se usaba la estructura construida.
- En el segundo experimento se trabajó con el algoritmo que usa campos aleatorios de Markov, usando la misma base de datos del experimento anterior, teniendo resultados favorables respecto al algoritmo *SumMaxMaps*.
- El tercer experimento fue reconocer autos ocluidos, usando la inferencia por partes, con el objetivo de experimentar con objetos diferentes a rostros.

- Finalmente el último experimento se hizo usando el algoritmo con histéresis con una base de datos que constaba de imágenes de rostros tomadas en condiciones de iluminación variable.

En la mayoría de los experimentos se observaron mejoras significativas estadísticamente en los porcentajes de detección, comparados con el algoritmo *SumMaxMaps*, tanto en reconocimiento de objetos en condiciones de oclusión como en condiciones de iluminación variable.

Capítulo 6

Aplicaciones

En este capítulo se presentan algunas aplicaciones, de los algoritmos presentados en la tesis, en algunas ramas de visión por computadora, segmentación y recuperación de imágenes con objetos similares a un objeto particular.

6.1. Segmentación

Esta aplicación se planteó al observar que el modelo ajustado en la imagen delimita el objeto bastante bien, por ello surgió la idea de usar los parámetros del modelo para segmentar el objeto en la imagen, esta aplicación podría ayudar a mejorar la calidad de la segmentación de objetos en la imagen y favorecer la realización de tareas como el etiquetado de imágenes.

La aplicación consiste en lo siguiente; una vez que se tiene el bosquejo del objeto reconocido en la imagen, para segmentar dicho objeto se propone rodear los elementos del bosquejo suponiendo que dichos elementos constituyen bordes del objeto, en el algoritmo 13 se muestra el pseudocódigo para segmentar varias ocurrencias del objeto. La función *FIND* en la línea 3 puede ser alguno de los algoritmos de reconocimiento, ya sea el algoritmo *SumMaxMaps* propuesto por Wu et al. [WSGZ09], o alguna de las extensiones propuestas en el capítulo 4. La función *GetSegment* 5 regresa la región delimitada por el modelo, y para delimitar el modelo únicamente se rodean los elementos de la plantilla.

Para ilustrar los resultados arrojados por este algoritmo, se construyó el modelo para automóviles y se agregaron varios automóviles no incluidos en la fase de entrenamiento a varias escalas en diversas partes de una imagen en un ambiente

(a) (b)

FIGURA 6.1: Segmentación de objetos usando *active basis*

real, en la figura 6.1(a) se muestra esta imagen. Posteriormente, se aplicó el algoritmo 13 a la imagen que se construyo, teniendo como resultado un conjunto de segmentos que representan el objeto en la imagen. En la figura 6.1(b) se muestra el resultado obtenido al aplicar el algoritmo 13 en la imagen 6.1(a), que muestra cómo se segmentaron siete automóviles a diferentes escalas, se puede observar que al segmentar cada automóvil quedan algunas zonas no cubiertas por el algoritmo, esto se debe a que el modelo no contiene todos los bordes del objeto.

Algorithm 13 Segmentación de I dados la imagen I y el modelo B

1: $found \leftarrow true$
2: **while** $found$ **do**
3: $\{B, \Lambda\} \leftarrow FIND(I, B)$
4: **if** $\{B, \Lambda\} \neq \emptyset$ **then**
5: $Segment \leftarrow GetSegment(\{B, \Lambda\}, I)$
6: REMOVE(Segment,I)
7: **else**
8: $found \leftarrow false$
9: **end if**
10: **end while**

6.2. Recuperación de imágenes con objetos similares

Esta aplicación surgió como una forma de tratar de encontrar más ejemplos de un objeto y poder construir un mejor modelo a partir de un sólo ejemplo del objeto, resultando una forma de recuperar imágenes que contienen un objeto similar al que

se dio de ejemplo. La idea de esta aplicación es usar el modelo del objeto aprendido de una subimagen que contiene al objeto que se desea buscar, para después usarlo para encontrar imágenes con objetos similares. Partiendo de un modelo sencillo, éste se ajusta a un conjunto de imágenes, se toma la imagen que mejores resultados proporciona, se vuelve a aprender el modelo y así sucesivamente, hasta cumplir algún criterio de paro. Esta idea es muy similar a la que se propone para hacer *part-templates* en el trabajo de Wu et al. [WSGZ09]. La diferencia es que aquí sí se sabe que buscar, pues se selecciona un rectángulo que contiene al objeto que se quiere buscar y no se asume que el objeto está en todas las imágenes. Al final se regresa la información del modelo aprendido, las imágenes que contienen al objeto buscado y donde está presente en cada imagen. De esta forma se pueden recuperar imágenes similares dando como entrada una imagen y a la vez se aprende un modelo más general del objeto que se está buscando. La figura 6.2 ilustra la idea descrita en el texto anterior.

En el algoritmo 14 se muestra el pseudocódigo que recupera imágenes siguiendo lo descrito en el párrafo anterior. La línea 3 lo que hace es encontrar una base o plantilla para las imágenes que se tienen de ejemplo hasta el momento en que se llama la función. En la línea 5 se encuentran los mejores parámetros para el modelo en cada una de las imágenes.

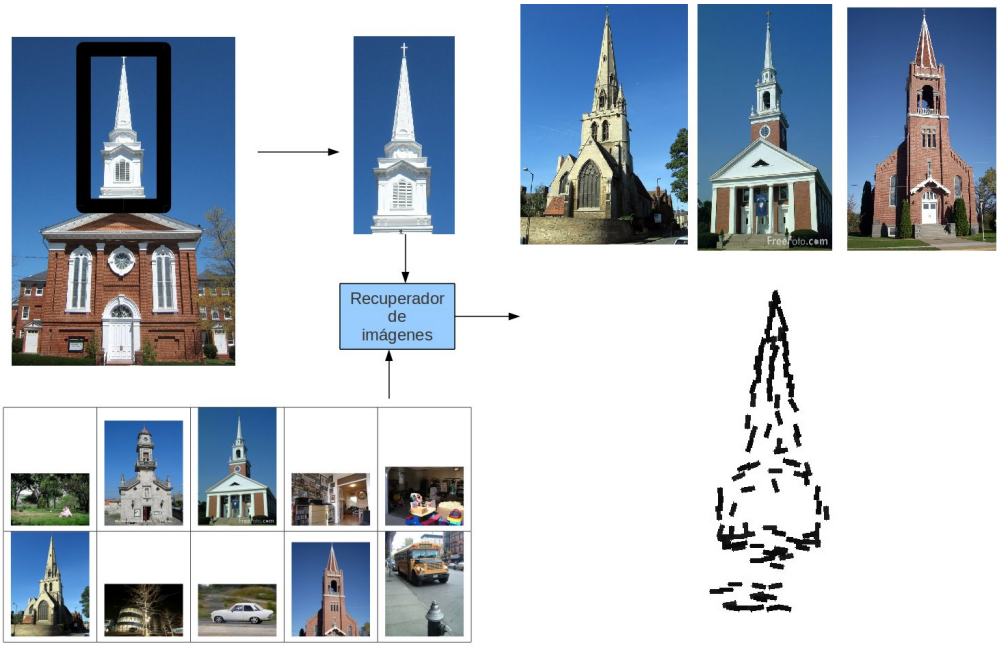


FIGURA 6.2: Idea de cómo recuperar imágenes con objetos similares y un modelo del objeto a partir de una imagen de ejemplo en el rectángulo y un conjunto de imágenes que pueden contener o no el objeto de interés.

En la figura 6.3 se muestra cómo va evolucionando el modelo y las imágenes recuperadas a partir del modelo obtenido en la iteración anterior.

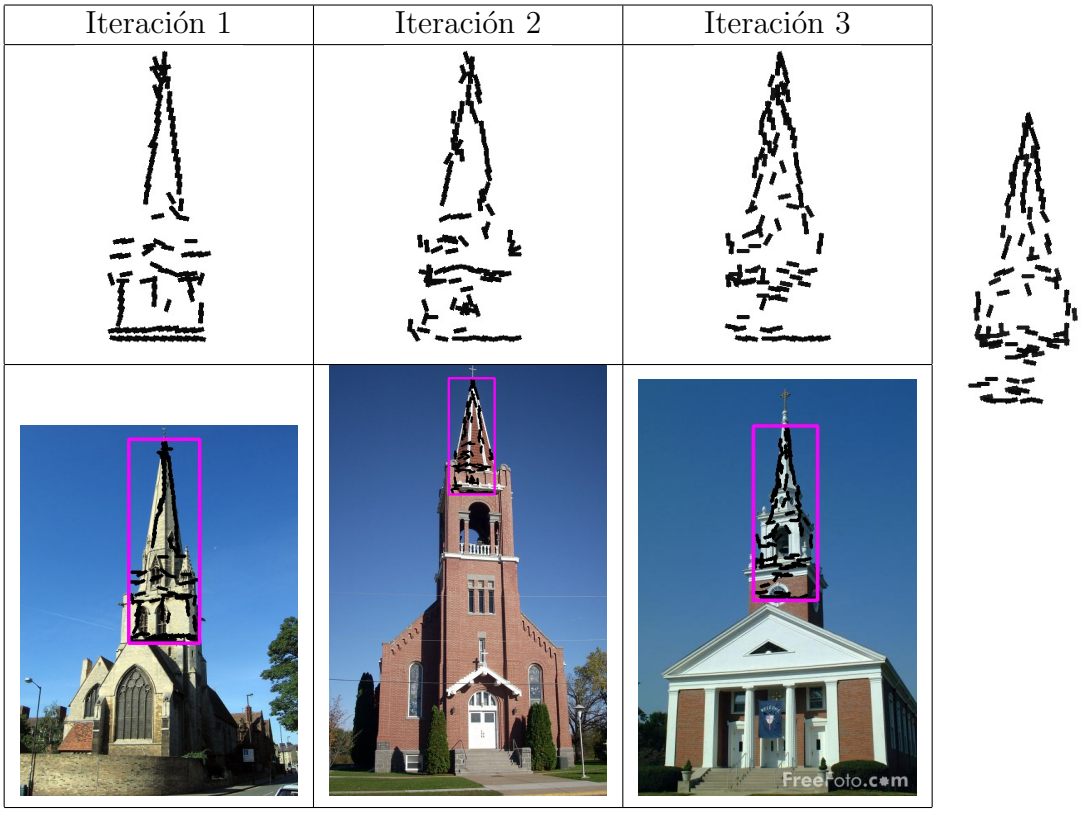


FIGURA 6.3: En el primer renglón se muestran los modelos construidos hasta la primera, segunda, y tercera iteración y en el segundo se muestran las imágenes recuperadas. En la parte derecha de la gráfica se muestra el modelo final aprendido después de recuperar 3 imágenes.

Esta aplicación únicamente funciona con objetos claramente significativos en la imagen. Se hizo el experimento con objetos no tan relevantes, por ejemplo recuperar imágenes que contienen ojos de personas y las imágenes de entrada eran paisajes, personas y en ambientes con más objetos. Lo que se recuperó es algo que se parece al objeto de entrada, sin embargo, las imágenes y las regiones donde se predice que se encontró el objeto contextualmente no corresponden a lo que se busca. Esto indica que la técnica tiene limitaciones y es un buen ejercicio el buscar criterios de su uso, a fin de delimitar su campo de aplicación.

Algorithm 14 Recuperación de K imágenes de $\{I_1, .., I_n\}$ similares a ObjImage

$train = \{ObjImage\}$
for $iter = 1$ to numIters **do**
  $B \leftarrow Learn(train)$
  for $i = 1$ to n **do**
    $\{rect_i, score_i\} \leftarrow find(I_i)$
  end for
  $mx \leftarrow argmax_{score}\{rect_i, score_i\}$
  $nSubImg \leftarrow SubImage(I_{mx}, rect_{mx})$
  $train \leftarrow train \cup \{nSubImg\}$
  $I \leftarrow I - \{I_{mx}\}$
  return K-best scores and B
end for

6.3. Resumen

En este capítulo se presentaron dos aplicaciones usando los algoritmos que existen para la representación *active basis*, esto incluye las modificaciones propuestas en la tesis. Las aplicaciones son:

- Usar los modelos *active basis*, para segmentar objetos en una imagen. Esta aplicación puede ser útil en tareas de etiquetado de imágenes.
- La otra aplicación consiste en usar los algoritmos de aprendizaje y reconocimiento para poder recuperar imágenes que contienen el objeto de interés; además, con esta aplicación se puede construir un modelo más general del objeto.

Estas aplicaciones fueron únicamente probadas con ejemplos sencillos con el objetivo de ilustrar el posible alcance de las mismas, por lo que no se presentan resultados cuantitativos.

Capítulo 7

Conclusiones y Trabajo Futuro

7.1. Resumen

En este trabajo se planteó el problema de reconocer objetos deformables en condiciones variables, como lo son oclusión e iluminación variable. Siendo los objetivos los siguientes puntos:

- Construir el modelo de un objeto el cual permita reconocer objetos en imágenes tomadas en condiciones variables, usando un nivel bajo de supervisión.
- Proponer métodos que permitieran usar el modelo aprendido y poder reconocer objetos cuando éstos se encuentran en condiciones variables.

Para poder lograr los objetivos en este trabajo de tesis se realizaron los siguientes puntos:

- Se propuso un algoritmo para estructurar el modelo o plantilla de un objeto usando la representación *active basis* previamente aprendida; teniendo como resultado dos posibles estructuras, la primera consistió en tener un modelo *active basis* como unión de cúmulos que representan *partes* del objeto, la segunda consistió en tener el modelo expresado como un grafo donde cada elemento es un vértice y las aristas están dadas por la cercanía entre los elementos de la base.

- Se propusieron varios algoritmos para usar las posibles estructuras aprendidas, tanto en condiciones de oclusión parcial como de iluminación variable. Para oclusión parcial se propuso una forma de usar una jerarquía con los cúmulos que forman parte del objeto y un algoritmo de inferencia usando campos aleatorios de Markov que usa el grafo como sistema de vecindades. Para iluminación variable se propuso un algoritmo que usa histéresis para completar la información del modelo suponiendo que usar la representación *active basis* es similar a un detector de bordes de Canny.

- Se experimentó con los algoritmos en imágenes con iluminación variable obtenidas de la base de datos de Yale [GBK00], y en imágenes de rostros y autos con oclusión parcial sintética obtenidas de la base de datos Caltech 101 clase *Faces* y *car-side* [FFFP06]. Los resultados de estos algoritmos fueron favorables comparados con el algoritmo *SumMaxMaps* usado para la representación *active basis* [WSGZ09].

- Se propusieron dos aplicaciones para usar el modelo y los parámetros recuperados a partir de la imagen donde se reconoció el modelo. La primera aplicación propuesta consiste en usar la plantilla del objeto para segmentar el objeto. La segunda aplicación fue emplear la plantilla para recuperar imágenes similares a un objeto particular y a la vez construir un modelo más general del objeto.

7.2. Conclusiones

Como conclusiones de este trabajo de tesis se puede mencionar que, con respecto a los algoritmos de inferencia se obtienen resultados favorables en el reconocimiento de objetos parcialmente ocluidos, particularmente usando *SumMaxMaps* con campos aleatorios de Markov. Sin embargo, a pesar de que se presentaron experimentos donde se muestra que es posible recuperar información acerca del estado del objeto, la complejidad computacional que implica la inferencia aún no es competitiva comparada con algoritmos basados en técnicas como, por ejemplo, *boosting*. Respecto a los resultados con cambios de iluminación se mostró que es posible recuperar la posición del objeto en la imagen, pero aun está el problema sin resolver de recuperar el estado de iluminación.
En general las conclusiones de este trabajo son:

- Es posible construir una estructura del objeto sin depender del objeto usando la representación *active basis*.
- La agrupación de los elementos en partes del objeto contribuye sustancialmente al reconocimiento de objetos parcialmente ocluidos.
- A partir de relaciones espaciales y topológicas de los elementos es posible recuperar más información sobre el resto de los elementos del modelo, ya sea mediante histéresis para el caso de iluminación variable o a través de la regularización de un campo aleatorio de Markov en el caso de oclusión parcial. Obteniendo mejoras en las tasas de detección de hasta un 20 % respecto al algoritmo *SumMaxMaps* en ambos casos.

La aportación principal de este trabajo se puede considerar como una extensión a los modelos *active basis* para el caso de oclusión. En este contexto se propusieron dos formas de reconocer objetos en condiciones de oclusión, usando inferencia por partes y mediante campos aleatorios de Markov; que es algo que hasta donde se investigo no existen trabajos concretos para resolver ese problema, aunque sí existen algunos estudios que buscan resolver el problema [HZ10, WZ10].

7.3. Trabajo futuro

A continuación se proponen algunas ideas para un posible trabajo posterior:

- En la parte de aprendizaje se propone estructurar los elementos en otras representaciones, todo esto en un esquema de aprendizaje no supervisado, con el objetivo de usar las ideas propuestas en la tesis en otras representaciones.
- En la parte de reconocimiento se propone unir las ideas usadas para reconocimiento de objetos ocluidos e iluminación variable usando el algoritmo de campos aleatorios de Markov. Esta extensión puede surgir de manera natural puesto que ambos algoritmos usan el mismo grafo.
- En cuanto a la representación *active basis* y algoritmo de inferencia *SumMaxMaps* es posible mejorar la velocidad de reconocimiento usando tecnologías como GPUs pues el aplicar el filtro en cada punto de la imagen es independiente de los otros puntos por lo que se puede hacer en paralelo.

- En cuanto a las aplicaciones mencionadas en el capítulo 6, éstas se pueden implementar en aplicaciones más grandes, como en competencias de reconocimiento y recuperación de información para medir sus resultados comparándolos con algoritmos tradicionales en esas áreas.
- Además se propone un algoritmo para aprender modelos a partir de imágenes con el objeto incompleto, ésta idea es explicada en el apéndice B.

7.4. Publicaciones

Algunos de los resultados mostrados en este trabajo de tesis fueron reportados en el trabajo: *Herrera-Domínguez Pável and Altamirano-Robles Leopoldo. "A Hierarchical Recursive Partial Active Basis Model"*, que aparece en las memorias de *Second Mexican Conference of Patter Recognition 2010* (MCPR 2010) [HDAR10].

Referencias

[ARARCE09] S.E. Ayala-Raggi, L. Altamirano-Robles, and J. Cruz-Enriquez. Recovering 3D Shape and Albedo from a Face Image under Arbitrary Lighting and Pose by Using a 3D Illumination-Based AAM Model. In *Proceedings of the 6th International Conference on Image Analysis and Recognition*, page 593. Springer, 2009.

[BMX+04] S. Baker, I. Matthews, J. Xiao, R. Gross, T. Ishikawa, and T. Kanade. Real-Time Non-Rigid Driver Head Tracking for Driver Mental State Estimation. 2004.

[BS00] I. Beichl and F. Sullivan. The metropolis algorithm. *Computing in Science & Engineering*, 2(1):65–69, 2000.

[Can87] J. Canny. A computational approach to edge detection. *Readings in computer vision: issues, problems, principles, and paradigms*, page 184, 1987.

[CET+01] T.F. Cootes, G.J. Edwards, C.J. Taylor, et al. Active appearance models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):681–685, 2001.

[Cor01] T.H. Cormen. *Introduction to algorithms*. The MIT press, 2001.

[CTC+95] T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, et al. Active shape models-their training and application. *Computer vision and image understanding*, 61(1):38–59, 1995.

[DPDPL97] S. Della Pietra, V. Della Pietra, and J. Lafferty. Inducing features of random fields. *IEEE transactions on pattern analysis and machine intelligence*, 19(4):380–393, 1997.

[FE73] MA Fischler and RA Elschlager. The representation and matching of pictorial structures. *IEEE Transactions on Computers*, 100(22):67–92, 1973.

[FFFP06] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006.

[Fri87] J.H. Friedman. Exploratory projection pursuit. *Journal of the American statistical association*, 82(397):249–266, 1987.

[GBK00] A.S. Georghiades, P.N. Belhumeur, and D.J. Kriegman. From Few to Many: Generative Models for Recognition Under Variable Pose and Illumination. In *Proceedings of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition 2000*, page 277. IEEE Computer Society, 2000.

[GGR84] S. Geman, D. Geman, and S. Relaxation. Gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):721–741, 1984.

[HCM65] R.V. Hogg, A.T. Craig, and J. McKean. *Introduction to mathematical statistics*. Macmillan New York, 1965.

[HDAR10] Pavel Herrera-Domínguez and Leopoldo Altamirano-Robles. A Hierarchical Recursive Partial Active Basis Model. In *Advances in Pettern Recognition*, 2010.

[Hoe84] Paul G. Hoel. *Introduction to mathematical statistics*. Wiley, 1984.

[HZ10] W. Hu and S.C. Zhu. Learning a Probabilistic Model Mixing 3D and 2D Primitives for View Invariant Object Recognition. *IEEE Conference on Computer Vision and Pattern Recognition*, 2010.

[Jah91] B. Jahne. *Digital image processing: concepts, algorithms and scientific applications*. Springer-Verlag London, UK, 1991.

[Kan06] G.K. Kanji. *100 statistical tests*. SAGE publications Ltd, 2006.

[KGJV83] S. Kirkpatrick, C.D. Gelatt Jr, and M.P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671, 1983.

[KL51] S. Kullback and R.A. Leibler. On information and sufficiency. *The Annals of Mathematical Statistics*, 22(1):79–86, 1951.

[Low04] D.G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[Mal89] SG Mallat. A theory for multiresolution signal decomposition: the wavelet representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 11(7):674–693, 1989.

[Mar82] D. Marr. *Vision: A computational approach*. Freeman & Co., San Francisco, 1982.

[OB10] B. Ommer and JM Buhmann. Learning the compositional nature of visual object categories for recognition. *IEEE transactions on pattern analysis and machine intelligence*, 32(3):501, 2010.

[SGZW09] Z. Si, H. Gong, S.C. Zhu, and Y.N. Wu. Learning Active Basis Models by EM-Type Algorithms. *Statistical Science*, 2009.

[VJ01] P. Viola and M. Jones. Rapid Object Detection using a Boosted Cascade of Simple. In *Proc. IEEE CVPR 2001*, 2001.

[WN05] B. Wu and R. Nevatia. Detection of multiple, partially occluded humans in a single image by bayesian combination of edgelet part detectors. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV 2005*, volume 1, 2005.

[WN09] B. Wu and R. Nevatia. Detection and segmentation of multiple, partially occluded objects by grouping, merging, assigning part detection responses. *International Journal of Computer Vision*, 82(2):185–204, 2009.

[WSF+07] Y.N. Wu, Z. Si, C. Fleming, S.C. Zhu, and L.A. UCLA. Deformable Template As Active Basis. In *IEEE 11th International Conference on Computer Vision, 2007. ICCV 2007*, pages 1–8, 2007.

[WSGZ09] Y.N. Wu, Z. Si, H. Gong, and S.C. Zhu. Learning active basis model for object detection and recognition. *International Journal of Computer Vision*, pages 1–38, 2009.

[WZ10] Tianfu Wu and Song-Chun Zhu. A numerical study of the bottom-up and top-down inference processes in and-or graphs. *International Journal of Computer Vision*, pages 1–27, 2010.

[ZCY07] L. Zhu, Y. Chen, and A. Yuille. Unsupervised learning of a probabilistic grammar for object detection and parsing. *Advances in neural information processing systems*, 19:1617, 2007.

[ZLH+08] L. Zhu, C. Lin, H. Huang, Y. Chen, and A. Yuille. Unsupervised structure learning: hierarchical recursive composition, suspicious coincidence and competitive exclusion. *ECCV08*, 2008.

[ZM06] S.C. Zhu and D. Mumford. A stochastic grammar of images. *Foundations and Trends® in Computer Graphics and Vision*, 2(4):362, 2006.

Apéndice A

Tablas de Resultados

Se muestran las tablas de resultados de los experimentos 1, 2 y 3 de reconocimiento de objetos ocluidos.

A.1. Rostros

En la tablas A.1 a A.6 se muestran todos los resultados de los algoritmos al aplicarlos a imágenes con rostros ocluidos a diferentes niveles, en cada tabla se muestra el algoritmo, el porcentaje de detección, la diferencia porcentual entre los promedios de cada algoritmo contra el resultado del algoritmo de referencia y el resultado de las pruebas de significancia estadística usando la prueba Z-test[Kan06].

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	61.200±11.207 %	0.000 %	
SumMaxMaps+CAM(U=1)	72.800±9.432 %	11.600 %	SI
SumMaxMaps+CAM(U=3)	74.400±8.040 %	13.200 %	SI
SumMaxMaps+CAM(U=36)	71.200±7.332 %	10.000 %	SI
PHAB	69.600±7.419 %	8.400 %	SI

Tabla A.1: Se muestran los resultados aplicando los algoritmos mencionados a rostros en el nivel 0 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	52.400±12.244 %	0.000 %	
SumMaxMaps+CAM(U=1)	73.600±6.974 %	21.200 %	SI
SumMaxMaps+CAM(U=3)	72.800±7.756 %	20.400 %	SI
SumMaxMaps+CAM(U=36)	68.000±7.155 %	15.600 %	SI
PHAB	60.800±8.908 %	8.400 %	SI

Tabla A.2: Se muestran los resultados aplicando los algoritmos mencionados a rostros en el nivel 1 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	45.200±12.192 %	0.000 %	
SumMaxMaps+CAM(U=1)	66.000±6.000 %	20.800 %	SI
SumMaxMaps+CAM(U=3)	66.800±7.600 %	21.600 %	SI
SumMaxMaps+CAM(U=36)	60.000±8.390 %	14.800 %	SI
PHAB	50.000±11.207 %	4.800 %	NO

Tabla A.3: Se muestran los resultados aplicando los algoritmos mencionados a rostros en el nivel 2 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	35.600±16.919 %	0.000 %	
SumMaxMaps+CAM(U=1)	57.600±13.529 %	22.000 %	SI
SumMaxMaps+CAM(U=3)	57.200±12.528 %	21.600 %	SI
SumMaxMaps+CAM(U=36)	55.600±12.060 %	20.000 %	SI
PHAB	49.600±9.499 %	14.000 %	SI

Tabla A.4: Se muestran los resultados aplicando los algoritmos mencionados a rostros en el nivel 3 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	35.600±9.269 %	0.000 %	
SumMaxMaps+CAM(U=1)	48.800±10.553 %	13.200 %	SI
SumMaxMaps+CAM(U=3)	49.600±9.666 %	14.000 %	SI
SumMaxMaps+CAM(U=36)	49.600±7.200 %	14.000 %	SI
PHAB	37.200±9.130 %	1.600 %	NO

Tabla A.5: Se muestran los resultados aplicando los algoritmos mencionados a rostros en el nivel 4 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	29.200±10.127 %	0.000 %	
SumMaxMaps+CAM(U=1)	48.800±10.553 %	19.600 %	SI
SumMaxMaps+CAM(U=3)	49.600±9.666 %	20.400 %	SI
SumMaxMaps+CAM(U=36)	49.600±7.200 %	20.400 %	SI
PHAB	30.200±9.928 %	1.000 %	NO

TABLA A.6: Se muestran los resultados aplicando los algoritmos mencionados a rostros en el nivel 5 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	91.600±3.666 %	0.000 %	NO
PHAB	91.600±3.666 %	0.000 %	NO

TABLA A.7: Se muestran los resultados aplicando los algoritmos mencionados a autos en el nivel 0 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	89.800±3.156 %	0.000 %	NO
PHAB	91.200±3.453 %	1.400 %	SI

TABLA A.8: Se muestran los resultados aplicando los algoritmos mencionados a autos en el nivel 1 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	87.200±5.455 %	0.000 %	NO
PHAB	89.200±5.810 %	2.000 %	SI

TABLA A.9: Se muestran los resultados aplicando los algoritmos mencionados a autos en el nivel 2 de oclusión

A.2. Automóviles

En las tablas A.7-A.12 se muestran los resultados de los algoritmo *SumMaxMaps* y *PHAB* aplicado a la detección de autos ocluidos parcialmente. Cada tabla muestra los porcentajes, la diferencia porcentual respecto a los resultados al solo usar *SumMaxMaps* y el resultado de las pruebas de significancia estadística usando la prueba Z-test[Kan06].

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	80.600±4.630 %	0.000 %	NO
PHAB	87.000±4.669 %	6.400 %	SI

Tabla A.10: Se muestran los resultados aplicando los algoritmos mencionados a autos en el nivel 3 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	74.400±6.859 %	0.000 %	NO
PHAB	79.000±8.258 %	4.600 %	SI

Tabla A.11: Se muestran los resultados aplicando los algoritmos mencionados a autos en el nivel 4 de oclusión

Algoritmo	Porcentaje de detección	vs SumMaxMaps	S.E.
SumMaxMaps	74.400±6.859 %	0.000 %	NO
PHAB	79.000±8.258 %	4.600 %	SI

Tabla A.12: Se muestran los resultados aplicando los algoritmos mencionados a autos en el nivel 5 de oclusión

Apéndice B

Aprendizaje a partir de imágenes ocluidas

Al intentar construir un modelo a partir de objetos que no se sabe si en todas las imágenes aparecen completos, lo que se podría esperar del algoritmo propuesto de Wu et al.[WSGZ09] es que en el modelo aparezcan ciertos bordes correspondientes a los bordes que provienen del objeto que ocluyen al objeto que se está modelando, o que al haber oclusión los parámetros de la distribución de la familia exponencial, que parametrizan las respuestas a los filtros de Gabor no sean lo suficientemente buenos para poder hacer un buen reconocimiento posteriormente.

B.1. Observaciones

Si existen objetos ocluidos en el conjunto de entrenamiento, y las partes ocluidas no se repiten en ciertas zonas especificas del objeto entonces, se puede pensar que los histogramas no consten de una cola alargada hacia la derecha como se espera, si no que tienen frecuencias altas en respuestas bajas de los filtros de Gabor en una misma posición, y frecuencias bajas en respuestas altas a los filtros de Gabor.

B.2. Algoritmo

Para solucionar este problema se propone modificar las líneas 8 y 9 en el algoritmo 3, que es donde se elige y se encuentran los parámetros del siguiente elemento de la

base. La idea entonces es crear mapas de votación los cuales ayudaran a saber si en la i-ésima imagen cierta posición se considera ocluida o no. Para lo que se propone un algoritmo iterativo el cual estima un modelo para las imágenes asumiendo que todas están sin oclusión. Una vez que se tiene un modelo inicial se remueven las respuestas, que evaluadas bajo el modelo, sean menores que un umbral, en este caso menores que 0. Una vez que se remueven estas respuestas se vuelven a calcular los histogramas y se aprende un modelo nuevo estimando nuevamente todos los parámetros. Los algoritmos 15 y 16 muestran como se hace esto.

Algorithm 15 Aprendizaje a partir de imágenes ocluidas

$I = \{I_1, .., I_n\}$
$Maps[1, .., n][1, .., w][1, .., h] = 1$
for $iter = 1$ to $numIterations$ **do**
  $B \leftarrow Learn(I, Maps)$
  for $i = 1$ to n **do**
    $Maps[i] \leftarrow EstimateVotingOcclusionMaps(MAX1_n, B, Map[i])$ (algorithm 16)
  end for
end for

Algorithm 16 Estimación de mapas oclusión por votación

for $B_i \in B$ **do**
  if $\lambda_i * MAX1_n(x_i, y_i, \alpha_i) - \log Z_i <= 0$ **then**
    $MAP[x_i][y_i] \leftarrow 0$
  end if
end for