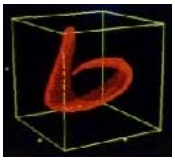
Universidad Tecnológica de la Mixteca

"Sistema de Reconocimiento de Objetos de única instancia".

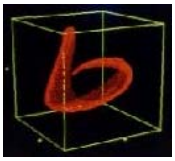
Tesis que para obtener el titulo de Ingeniero en Computación presenta:

Iván Antonio García Pacheco.

Huajuapan de León, Oaxaca., Febrero del 2001



Índice.



Introducción.

El procesamiento de las imágenes esta relacionado con la capacidad de percepción que poseemos los humanos. Una máquina de visión, por otro lado, se relaciona con una máquina encargada del procesamiento digital de una imagen.

Un campo de aplicación de las técnicas de procesamiento de imágenes consiste en la resolución de problemas relacionados con la percepción automatizada. En este caso, el interés se centra en los procedimientos para extraer la información de la imagen de forma conveniente para el procesamiento por computadora. A menudo esta información tiene poco en común con los rasgos visuales que los seres humanos emplean para interpretar el contenido de una imagen.

Los problemas actuales de la percepción automatizada, que utilizan rutinariamente técnicas de procesamiento de imágenes, son el reconocimiento automático de caracteres, los reconocimientos militares, el tratamiento automático de huellas digitales, las muestras de sangre, las imágenes de rayos X y el procesamiento automático de las imágenes aéreas y de satélite para la predicción del tiempo y evaluación de cultivos.

Cabe mencionar, que el procesamiento digital de imágenes implica procedimientos que normalmente se expresan en forma de algoritmos.

De esta manera, con excepción de la adquisición de las imágenes y su representación, la mayor parte de las funciones de tratamiento de la imagen pueden ser implementadas en *software*.

Uno de los problemas actuales de la percepción automatizada, es la construcción de modelos que puedan ser usados por la computadora, para que esta pueda reconocer estos objetos. Tradicionalmente, los modelos se han construido a través de la representación de la información contenida en el objeto usando líneas, arcos, etc., dando lugar a los llamados modelos geométricos. Desafortunadamente, al construir este tipo de modelos se pierde una cantidad muy significativa de información, que puede ser crucial al tratar de reconocer los objetos.

Recientemente se ha propuesto como una alternativa a los modelos geométricos, el uso de otro tipo de modelos basados en la apariencia que presentan los objetos, de tal manera que es posible mediante el uso de ellos, preservar no solamente el contorno, sino también otro tipo de información tal como la textura, el color, etc.

La técnica de Espacios Fundamentales (*Eigenspaces*), propuesta por *Turk y Pentland* [Turk & Pentland, 1991] para el reconocimiento de rostros, y por *Murase y Nayar* [Nayar, 1996] para el reconocimiento de objetos en general, es una de las técnicas que emplean modelos basados en la apariencia de los objetos. Tal técnica inicia con la toma de un conjunto de imágenes del objeto a reconocer y mediante el empleo del Análisis de Componentes Principales (PCA por sus siglas en inglés) encuentra un conjunto de imágenes representativas, que permiten construir una representación compacta del objeto (modelo) dentro de un espacio vectorial denominado *espacio fundamental*. El proceso de reconocimiento se realiza por la proyección de un objeto a reconocer dentro del espacio fundamental que contiene a los modelos de los objetos previamente construidos, localizando el modelo que corresponda a tal imagen.

Debido a lo anterior, el objetivo del presente trabajo de tesis es desarrollar un algoritmo que sea capaz de generar un espacio fundamental para fines de reconocimiento de un objeto, usando múltiples perspectivas del mismo. Durante este desarrollo, se demuestra que las técnicas del

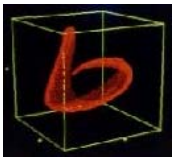
Procesamiento Digital de Imágenes y la representación de Espacios Fundamentales, pueden aplicarse en conjunto, para la resolución de tal problema. De igual manera, se muestran los resultados obtenidos al representar una imagen en un espacio como si fuera un punto cualquiera[1] proyectado a un espacio vectorial, parte crucial para el desarrollo de este trabajo.

El material de este trabajo esta organizado en 4 capítulos divididos en áreas temáticas. En el Capítulo 1 se trata de manera extensa los dos puntos fundamentales para el desarrollo del sistema propuesto: en primera instancia se trata la *representación digital de imágenes* y su aplicación a este trabajo, para posteriormente definir la *técnica de los espacios fundamentales,* su representación y los conceptos necesarios para formar un espacio de este tipo.

En el Capitulo 2 se exponen los criterios matemáticos necesarios para definir un espacio fundamental y se explican de manera concreta las etapas que forman un sistema de reconocimiento en general, así mismo se demuestra la utilidad de cada una de estas etapas para dar soporte a una solución viable de la hipótesis inicial.

En el Capítulo 3 se muestra la implementación formal del sistema propuesto; explicando detalladamente los procesos que se llevan a cabo para conformar un espacio bien definido con las imágenes utilizadas, se muestran algoritmos y procedimientos programados que fueron necesarios para hacer de este un sistema eficiente.

[1] Ver **Apéndice A** para comprender la *representación de un punto en un espacio vectorial.*



Capítulo 1

Identificación de Objetos

1.1 Preliminares.

1.1.1 Representación Digital de Imágenes.

Una imagen está compuesta por variaciones de luminosidad en función de coordenadas espaciales, y temporales en el caso del video.

Una imagen digital se puede representar matemáticamente por medio de una matriz $F=(f_{ij})$ de dimensiones $m \times n$, donde i denota a la fila y j a la columna de dicha matriz, respectivamente. De esta forma el producto de m y n es el número de puntos requeridos para representar la imagen.

El término *imagen* se refiere a una función bidimensional de intensidad de luz $f(x,y)$ donde x e y representan las coordenadas espaciales y el valor de f en un punto cualquiera (x,y) es proporcional al brillo (o nivel de gris) de la imagen en ese punto.

Una imagen digital es una imagen que se ha discretizado tanto en las coordenadas espaciales como en el brillo. Una imagen digital puede considerarse como una matriz cuyos índices de fila i y columna j, identifican

un punto de la imagen y el valor del correspondiente elemento de la matriz indica el nivel de gris en ese punto.

Los elementos de una distribución digital de este tipo se denominan elementos de la imagen, o más comúnmente *pixels* o *pels*, abreviaturas de su denominación inglesa *"pictures elements"*. [González & Woods, 1996]

En la representación digital de los niveles de gris de las imágenes, la imagen es presentada como un arreglo de dos dimensiones. Donde cada número representa la intensidad o nivel de gris de la imagen como una posición relativa. Si cada nivel de gris es representado con 8 *bits* (1 *byte*), entonces el nivel de gris permitido es de 2^8 o 256 posibles valores. Estos niveles son usualmente asignados a valores enteros dentro del rango de 0 a 255, donde el 0 representa el nivel de intensidad oscuro y 255 el nivel de intensidad claro.

En una imagen a color, la representación es similar, excepto que en cada locación de la matriz, el número representa tres colores primarios: rojo, verde y azul.

Para una representación a color, de 24 *bits* por *pixel*, el número es dividido en tres segmentos de 8 *bits*. Cada segmento representa la intensidad de uno de los colores primarios.

Desde su nivel más básico, el Procesamiento Digital de Imágenes requiere una computadora para procesar las imágenes y dos elementos clave para la entrada y salida: un digitalizador de imágenes y un dispositivo para el despliegue de imágenes (Véase Figura 1.1.1.1).

El tipo de sistema que se plantea implantar en la realización de este tipo de trabajos, lleva a cabo las siguientes operaciones: Adquisición de las imágenes, almacenamiento, procesamiento y presentación.

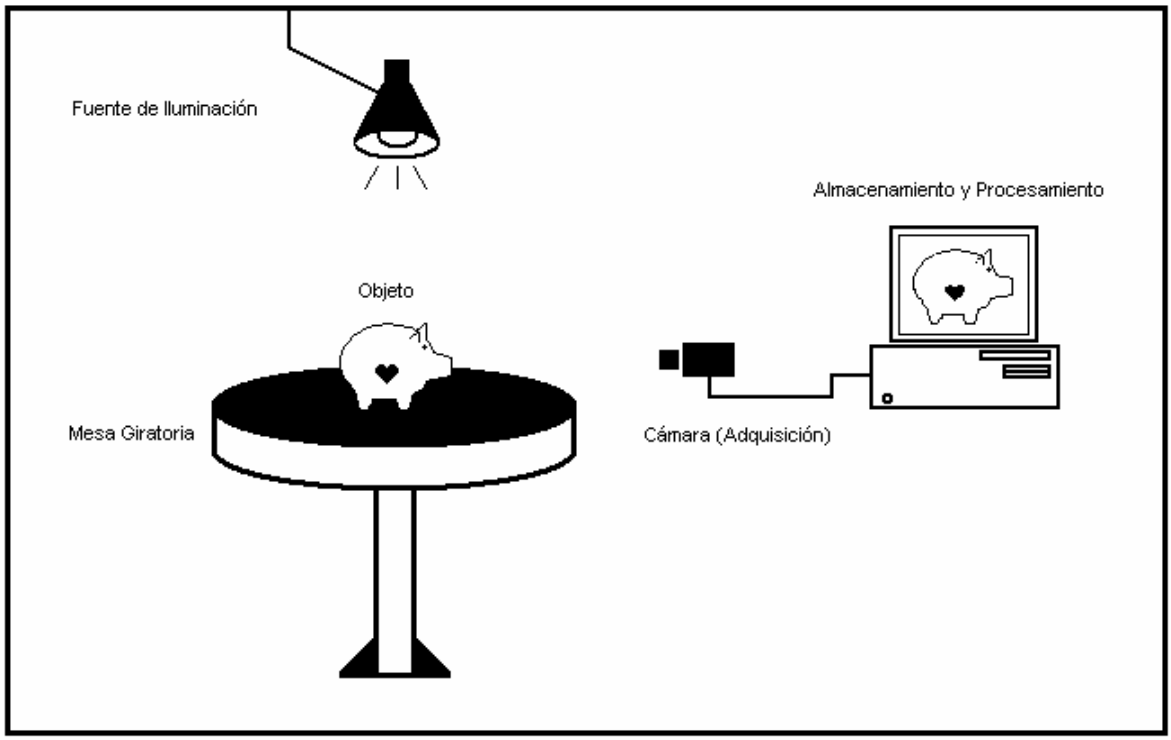


FIGURA 1.1.1.1. Esquema que muestra la forma en que se realiza la toma de las imágenes requeridas para construir un modelo basado en la apariencia de un objeto. Aún cuando en la figura no se señala explícitamente, la fuente de iluminación puede poseer la propiedad adicional de cambiar su posición respecto al objeto. La cámara puede poseer la propiedad adicional de descender y elevarse, para construir un modelo más completo.

1.2 Aplicación de los Espacios Fundamentales.

Cada **modelo basado en apariencia**[2] es parametrizado por las variables de la tarea de visión a realizar. En el caso del reconocimiento de objetos, estas variables pueden incluir la posición del objeto (respecto al sensor) y las condiciones de iluminación (posición de la fuente o fuentes de iluminación, intensidades de ellas, etc.). Si el objeto es no rígido, los parámetros de deformación pueden servir como variables adicionales. Nótese que estas variables son continuas dentro de algún rango de valores conveniente para dicha tarea.

A continuación se introducirán las siguientes definiciones:

Definición I) Sean $q_1, q_2, \ldots, q_m$ las variables de la tarea de visión a realizar. Se definen los *grados de libertad* (*DOF* por sus siglas en inglés) de la tarea, como el valor m (el número total de variables a considerar). [Baker, 1996]

Definición II) Sea $q = [q_1, q_2, \ldots, q_m]^T$ el vector formado por las variables de la tarea de visión. Para cualesquiera valores de q (dentro del rango definido), el sensor produce una imagen del objeto a procesar. Al considerar todos los posibles valores para q, se obtiene un codominio continuo de imágenes $i(q)$. A este codominio de imágenes se le conoce

[2] Suponiendo que se ha tomado un conjunto de imágenes de entrenamiento de un objeto, *Shapiro* y *Costa* definen el modelo basado en apariencia de un objeto, como una descripción del objeto en términos de cualesquiera características que sean detectables en las imágenes del objeto (no solo geométricas), éstas incluyen los diferentes valores de intensidad, textura, color, etc. [Shapiro & Costa, 1994]

como el espacio visual de trabajo (*visual workspace*) de la tarea. [Baker, 1996]

1.2.1 Identificación de Objetos mediante el uso de Espacios Fundamentales.

La representación de espacios fundamentales ha atraído recientemente la atención de los investigadores en el campo de la visión computacional.

La idea básica es representar las imágenes o características de la imagen en un espacio transformado donde las características individuales no estén correlacionadas. [Chandrasekaran, 1996]

Desde hace ya algunos años, han sido propuestas interesantes aplicaciones relacionadas con el procesamiento de imágenes, basadas en la representación de espacios fundamentales. Éstas incluyen el reconocimiento de rostros y la codificación de video. Sin embargo, la comunidad de investigadores de la visión computacional ha pasado por alto, en gran parte, progresos paralelos en el Procesamiento de Señales y el Álgebra Lineal referentes a la actualización de algoritmos eficientes para espacios fundamentales.

Estos nuevos desarrollos son significativos por dos razones: la primera es que él adoptarlos hará que alguno de los algoritmos actuales sea más robusto y eficiente, y la segunda, más importante aún; es el hecho de que la actualización de las representaciones de los espacios fundamentales abrirá nuevas e interesantes aplicaciones en la visión computacional, tales como el reconocimiento activo y el aprendizaje.

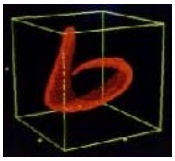
En contraste, la óptica tradicional trata una región de la imagen como una "cosa" en movimiento, debido a esto, no puede distinguir entre cambios en las perspectivas de vista o la configuración del objeto. Si los cambios en las posiciones de vista son significantes, la "cosa" no será la misma y puede ser que el reconocimiento falle.

Por otro lado, la técnica de representación de espacios fundamentales permite describir modelos simples de imágenes en dos dimensiones (2D), diseñados para explicar el cambio en las posiciones de vista o estructura del objeto.

Las representaciones de espacios fundamentales pueden proporcionar una codificación aproximada de un conjunto grande de imágenes en términos de un número pequeño de imágenes de base ortogonal. [González, 1997]

Las técnicas estándares de espacios fundamentales confían en el ajuste de mínimos cuadrados entre una imagen y un espacio fundamental, y pueden conducir a resultados pobres cuando existe la presencia de ruido en la imagen de entrada. Debido a esto, el problema correspondiente al espacio fundamental es reformulado como uno de estimación robusta.

Más que un intento por representar todas las vistas posibles de un objeto desde todas sus posiciones posibles, es más práctico representar un conjunto más pequeño de vistas canónicas y permitir una transformación parametrizada de la imagen de entrada y el espacio fundamental.



Capítulo 2

Representación de los Espacios Fundamentales

2.1 Representación Matemática.

La técnica de los espacios fundamentales es uno de los enfoques que emplea exclusivamente el Álgebra Lineal (en su forma original), lo que hace fácil su comprensión e implementación, sí se cuenta con estudios básicos en esta área, además de contar con algunos conocimientos extra sobre Análisis Numérico. El proceso de creación de los modelos de los objetos es independiente de la geometría, textura, color, etc., de los objetos, y sólo es restringido a que los objetos sean rígidos y monomórficos. Esto significa que este proceso es el mismo para cualquier objeto que cumpla con dichas características, pues no requiere el ajuste de ningún parámetro dependiente del tipo de objeto a modelar. Un espacio fundamental puede definirse en términos algebraicos.

Cada imagen de tamaño n x n puede ser vista como un vector columna de tamaño (n x n) x 1. En base a esto, podemos calcular el valor esperado de nuestras imágenes para obtener el “promedio” de estas, necesario para calcular la covarianza C_x de nuestro conjunto.

El valor esperado de la variable aleatoria x, recibe usualmente otros nombres como esperanza, o valor medio, o media. También utiliza las siguientes notaciones E(x), μ_X *(ó* μ *simplemente) ó* m_x. [Hernández, 1982]

Ahora bien, supongamos que *M* es el conjunto de imágenes de muestra, y que estas imágenes son presentadas como columnas:

$\Gamma_1, \Gamma_2, \ldots, \Gamma_M$ *donde cada* Γ_i *es de tamaño* $N^2 * 1$

Entonces el valor esperado esta dado por :

$$m_x = \frac{1}{M} \sum_{n=1}^{M} \Gamma_n$$

De igual forma, es necesario calcular una cantidad que indique la desviación Φ_n de las *M* imágenes como método de extracción.

$$\Phi_n = \Gamma_n - m_x$$

En base a estos dos conceptos, podemos formar la matriz de covarianza. La entrada *Cij* es un indicativo de cómo “covarían” los vectores *i, j;* mientras que *Cii* determina la varianza de la columna *i*.

$$C_x = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T$$

La covarianza entre dos variables es un resumen estadístico que nos indica si las puntuaciones están relacionadas entre sí. La formulación clásica, se simboliza por la letra griega sigma σ_{xy} cuando ha sido calculada en la población. Cuando en un estudio se mide la relación bivariada entre más de dos variables, frecuentemente la información se expresa en forma matricial.

La estructura de esta matriz es de naturaleza simétrica. [Hernández, 1982]

Si nos planteamos intuitivamente cual es la mejor representación bidimensional posible de un objeto en más de 2 dimensiones, estaremos de acuerdo en que esta será la que nos muestre una mayor extensión del mismo, o la que nos muestre más cosas de él. El concepto estadístico de varianza de una variable, es un buen indicador de la "extensión" que toman un conjunto de objetos "vistos" desde ella.

Teniendo en cuenta lo anterior, se calculan los valores propios (*eigenvalues*) y los correspondientes vectores propios (*eigenvectores*)[3].

Existen diversos métodos que facilitan esta labor entre los cuáles se encuentran:

- *Transformación de Jacobi para matrices simétricas.*
- *Reducción de una matriz simétrica a la forma tridiagonal: Usando reducciones de Householder.*
- *Uso de matrices de Hermitian (Matrices Hermitian).*
- *Reducción de una matriz general a la forma de Hessenberg.*
- *Algoritmo QR para matrices reales de Hessenberg.*

[3] Ver **Apéndice A** para una mejor comprensión de ambos términos

Puesto que, como se demuestra más adelante, se sabe que la matriz de covarianza es simétrica por naturaleza y debido a que nos interesa conocer tanto los valores propios como sus correspondientes vectores propios a la vez, es recomendable utilizar el método desarrollado por *Jacobi* (1845) y que recibió el nombre de *método iterativo de Jacobi*.

2.2 Transformaciones de *Jacobi*.

El método de *Jacobi* para la obtención de los valores propios ha atraído gran interés debido a que se puede implementar de manera paralela y es más aproximado que los métodos *QR* usados para resolver el mismo problema. [Zhon, 1998]

Existen dos tipos básicos de *Jacobi*, estos son, *Jacobi* de un solo lado (*one-sided Jacobi*) y *Jacobi* de dos lados (*two-sided Jacobi*). El método tradicional de Jacobi de dos lados para la obtención de los valores propios trabaja desarrollando una secuencia de actualizaciones similares

$$A \leftarrow Q^T AQ$$

con la propiedad de que cada nueva A es “mas diagonal” que su predecesor

El método de *Jacobi* consiste de una secuencia de transformaciones ortogonales de la forma:

$$A \rightarrow P_1^{-1}.A.P_1 \rightarrow P_2^{-1}.P_1^{-1}.A.P_1P_2 \rightarrow P_3^{-1}.P_2^{-1}.P_1^{-1}.A.P_1.P_2.P_3 \rightarrow etc.$$

Cada transformación (*rotación de Jacobi*) es una rotación diseñada para alinear cada uno de los elementos que se encuentren fuera de la diagonal de la matriz. Las transformaciones sucesivas eliminan los ceros previamente fijados, sin embargo los elementos de la diagonal se vuelven cada vez más pequeños con gran precisión. Los vectores propios son obtenidos mediante la acumulación de los productos de las transformaciones,

$$X_R = P_1.P_2.P_3.....$$

mientras que los elementos de la matriz diagonal final son los valores propios.

El funcionamiento del algoritmo de Jacobi implementado para solucionar el problema de los valores propios y vectores propios es descrito detalladamente en el Capítulo 3 de esta tesis.

2.3 Aproximación al *Eigen* reconocimiento.

Los *pixeles* que componen una imagen pueden considerarse como un vector. Así, una imagen prototipo se puede considerar como un vector base, útil para describir otras imágenes usando el *producto interno*.

De igual forma, un conjunto de imágenes puede formar un espacio para describir una variable en una dimensión. Por ejemplo, una secuencia de tres imágenes de una pelota en distintas vistas, puede ser usada para definir un espacio de una pelota girando.

Una imagen en particular, se puede representar por un vector con los tres productos internos obtenidos mediante las tres imágenes de ejemplo.

Es necesario computar este producto interno en una localización apropiada, pero puesto que la correlación es una secuencia de productos internos, es posible encontrar la correlación máxima, y entonces se describe a la imagen del vector de productos internos obtenido en ésa posición. [Reignier, 1995]

El problema con esta representación es que puede llegar a ser muy costosa computacionalmente hablando, mientras el número de imágenes aumente. Sin embargo, el conjunto de imágenes se puede reducir a un conjunto mínimo ortogonal, y la correlación con este conjunto base usarla para describir el contenido.

Ésta es la idea detrás de la codificación de un espacio fundamental, hecha popular por *Turk* y *Pentland*. [Turk & Pentland, 1991]

Para construir un espacio fundamental se parte de una base de datos con imágenes. Después se obtiene una imagen promedio, se calcula la variación entre las imágenes y finalmente, se utiliza un algoritmo para obtener la covariación basada en las imágenes de entrada.

Para n x n *pixeles* se obtiene una imagen de n^2 *pixeles*, esta matriz de covarianza tiene, en teoría, n^2 coeficientes.

Afortunadamente, dicha covariación es altamente diagonalizable, y se puede utilizar un algoritmo rápido para computar y diagonalizarla.

Los componentes principales de la matriz de covarianza forman un conjunto ortogonal de la base, que son el eje del espacio fundamental.

Supongamos que se tiene una base de datos que contiene 100 imágenes distintas, donde cada una es de k x k *pixeles*. Cada imagen es *normalizada* primero, es decir, se corta el fondo y se colocan pequeñas marcas en ella. De esta manera se reduce la variación en cada imagen.

La galería de imágenes normalizadas se denomina *conjunto*.

Los números de la escala de grises no representan en sí a la imagen, sino que son usados para calcular la variación entre una imagen y las demás.

Cada imagen de *k* x *k* *pixeles* puede ser vista como un vector columna de tamaño (*k* x *k*) x 1. Después, se calcula el *valor esperado* para las 100 imágenes, y de esta forma se obtiene el *promedio de la imagen*, necesario para calcular la covariación de nuestro conjunto, C_x.

Para este ejemplo, C_x es una matriz de tamaño k^2 que representa la variación entre las imágenes. Ésta matriz, es simétrica.

Puesto que la entrada C_{ij} en la matriz de covariación representa una medida de cómo los componentes *i* y *j* varían juntos, está claro que $C_{ij}=C_{ji}$. [Ammoura, 1997]

2.4 Componentes Principales del Sistema.

La idea de construir un sistema para el reconocimiento de objetos envuelve distintos componentes. Estos "subsistemas" pueden ser implementados usando diferentes teoremas matemáticos y/ o algoritmos.

2.4.1 Adquisición y Disposición.

Dado que es imposible obtener íntegro el espacio visual de trabajo para una tarea de visión ya dada, por tratarse de una representación continua, la técnica de espacios fundamentales parte de la adquisición de un conjunto de valores discretos para las variables de la tarea de visión. Este muestreo

discreto puede ser uniforme o no uniforme. Estas imágenes se usan posteriormente para crear una representación compacta que servirá no solo para reconocer las imágenes muestreadas sino también para reconocer aquellas imágenes comprendidas entre cualquier par de imágenes muestreadas consecutivamente. Este conjunto puede ser generado con distintas imágenes del objeto. 10 objetos representados con 4 diferentes imágenes cada uno generará un conjunto de entrenamiento con 40 imágenes $M = 40$.

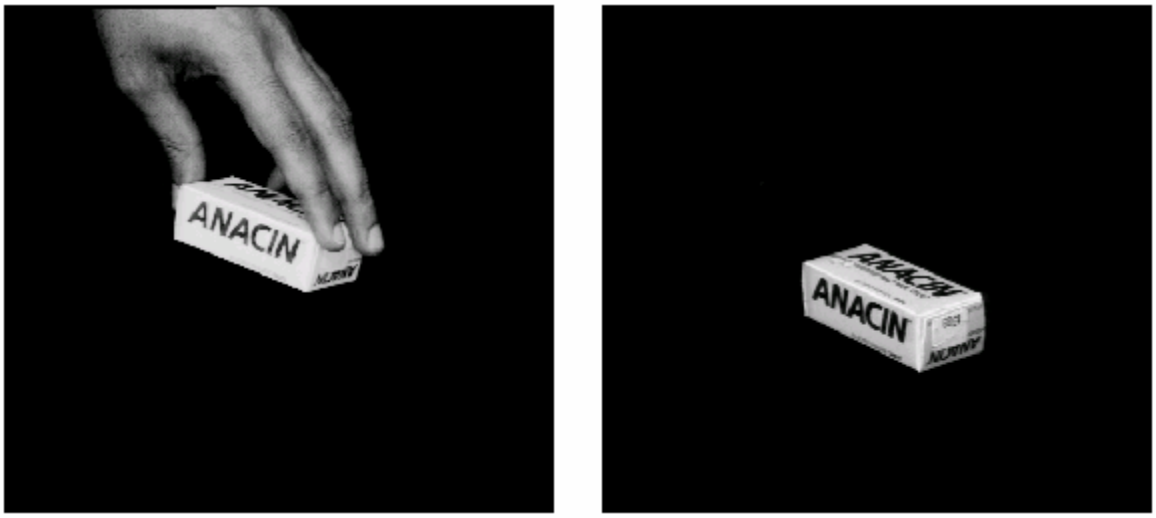


Figura 2.2.1.1. Adquisición de las imágenes que serán proyectadas al espacio fundamental.

Las cuatro imágenes pueden diferir en las condiciones de iluminación.

Para alcanzar una invariancia en escala, se restringen las imágenes para poseer un tamaño fijo e igual para todas. Además, las imágenes sólo deben contener al objeto a reconocer; esto implica que el objeto deberá ser segmentado del resto de la imagen.

Sea R_k el número de imágenes de muestra obtenidas para cada grado de libertad q_k. Entonces, el número total de imágenes adquiridas será $M = \prod_{k=1}^{m} R_k$. Sean estas imágenes: $\{i_1, i_2, \ldots, i_M\}$.

2.4.2 Análisis de Componentes.

Las imágenes en el conjunto tienden a estar correlacionadas en un alto grado, si y sólo si el desplazamiento entre imágenes consecutivas es pequeño. Es por ello que el siguiente paso consiste en aplicar una técnica de compresión que explote esta propiedad, permitiendo además obtener una representación compacta del objeto. El Análisis de Componentes Principales (PCA), basado en la transformación de *Hotelling* [González & Woods, 1996], es empleado para tal propósito.

Por medio de esta técnica se obtiene un conjunto de imágenes (denominados vectores fundamentales o propios), que se usarán como una base ortogonal para representar las imágenes individuales $\{i_1, i_2, \ldots, i_M\}$. [Altamirano, 1999]

Debido al hecho de que solucionar los *eigen* sistemas es un asunto de investigación profunda, este trabajo de ninguna forma es un tratamiento para dicho problema. La tentativa aquí, es simplemente plantear una solución posible que puede ser utilizada en nuestro sistema de reconocimiento.

Como primer paso para la construcción del modelo del objeto, el promedio *m* de todas las imágenes en el conjunto es restado de cada imagen. Esto asegura que el vector fundamental con el valor fundamental más grande represente la dimensión del subespacio en el cual la varianza de las imágenes es máxima en el sentido de la correlación. En otras palabras, esta es la dimensión más importante del espacio fundamental:

$$m_x = \frac{1}{M}\sum_{n=1}^{M}\Gamma_n$$

Enseguida, una matriz de imágenes *P* es construida, sustrayendo *m* de cada imagen y transformando las imágenes i_k en vectores columna $i'_{k'}$, de tal manera que las columnas de la imagen son apiladas para formar tal vector columna. Sea *P*:

$$P = \{i'_1 - m, i'_2 - m, \ldots, i'_M - m\}$$

P es de tamaño *O x M*, donde *O* es el número de *pixeles* en cada imagen y *M* es el número total de imágenes en el conjunto de muestras. Para calcular los vectores fundamentales del conjunto de imágenes, se define la matriz de covarianza.

Ésta, es una matriz cuadrada de tamaño N^2 y no es una medida solamente de la variación, sino también de la covariación de los vectores columna dados. *C* es una matriz muy grande, debido a que una imagen está formada por un gran número de *pixeles*.

El siguiente paso consiste en hallar los vectores fundamentales e_k y los correspondientes valores fundamentales λ_k de *C*. El cálculo se puede realizar por cualquier algoritmo conocido, aunque se debe notar que tal cálculo es intensivo, debido al tamaño de *C*, por lo que se deberá usar un algoritmo con mucha precisión.

Aún cuando todos los vectores fundamentales son requeridos para la perfecta reconstrucción de alguna imagen particular del conjunto de imágenes de muestra, sólo unos pocos son suficientes para el reconocimiento visual. Debido a esto, solamente se considerarán *K* vectores, seleccionando aquellos que poseen los *K* valores fundamentales más grandes. El resultado es un conjunto de valores fundamentales $\{\lambda_k \mid k = 1, \ldots, K\}$, donde $\{\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_k\}$, y un correspondiente conjunto de

vectores fundamentales ortonormales $\{ e_k \mid k = 1, 2, \ldots, K\}$. Cabe mencionar que cada vector fundamental es de tamaño N, esto es, el tamaño de una imagen. La selección del valor de K es seleccionado típicamente para un valor de 20 o menor.

2.4.3 Construcción del modelo de un objeto.

Cada imagen de muestra i'_j es proyectada al espacio fundamental a través de sustraer el promedio *c* de ella, y entonces calcular el producto interno del resultado con cada uno de los *k* vectores fundamentales. El resultado es un punto f_j en el espacio fundamental:

$$f_j = [e_1, e_2, \ldots, e_k]^T(i'_j - c)$$

A través de la proyección de todas las imágenes de muestra en esta forma, se obtiene un conjunto discreto de puntos. Debido a que imágenes consecutivas están fuertemente correlacionadas, sus proyecciones están también cercanas una de otra.

Aprovechando esta propiedad, los puntos discretos son interpolados para obtener una representación continua denominada *manifold*, que será el modelo del objeto procesado. (Véase la Figura 2.2.3.1 para observar una interpretación de un *manifold*). Se denotará a este *manifold* por *f(q)*.

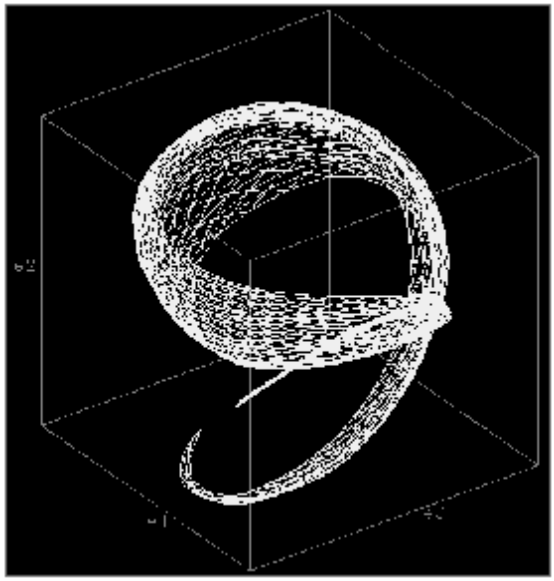


FIGURA 2.2.3.1. Aspecto de un manifold. Aunque el manifold normalmente reside en un espacio de 3 dimensiones, y por tanto no es posible obtener una representación gráfica de él, para propósitos ilustrativos, en la figura se ha parametrizado por la posición del objeto frente a la cámara, y por la dirección de la fuente de iluminación. Nótese que el inicio y el fin del manifold coinciden, pues se termina de capturar la apariencia del objeto en la misma posición donde se comenzó. Imagen generada mediante SLAM: Software Library for Appearance Matching. Department of Computer Science, Columbia University [Nene, 1994].

2.4.4 Reconocimiento del Objeto.

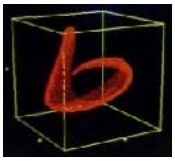
Para la etapa de reconocimiento, se supondrá que se cuenta con una imagen i_c que contiene un objeto que ha sido previamente segmentado de una escena que lo contenía. Además, se supondrá que i_c ha sido normalizada en escala (al tamaño seleccionado previamente para las imágenes de muestra). El promedio *c* es restado de la imagen, y el vector resultante i'_c es proyectado al espacio fundamental para obtener el punto:

$$f_c = [e_1, e_2, \ldots, e_k]^T(i'_c - c)$$

Entonces el problema de reconocimiento (*matching*) se reduce a encontrar la distancia mínima d_r y f_c y el *manifold f(q)*:

$$d_r = \min_q Pf_c - f(q)P$$

Si d_r está dentro de algún margen preestablecido, se concluye que i_c pertenece al *manifold f(q)*, y por tanto, i_c es reconocido como el objeto al que pertenece el *manifold*. En la práctica, el *manifold* es almacenado en memoria como una lista de puntos *k-dimensionales*, obtenidos por re-muestrear el *manifold*, por lo que el problema se reduce a encontrar el punto del *manifold* más cercano a f_c. Esto puede realizarse mediante un proceso de búsqueda exhaustivo, o bien, con un método más eficiente.



Capítulo 3

Implementación del Sistema

3.1 Introducción.

El lograr el reconocimiento de objetos a través de imágenes, tiene repercusiones inmediatas en diversas ramas del conocimiento, como en las Redes Neuronales y la Inteligencia Artificial, en particular en la robótica, y en tecnologías como el control de calidad y la manufactura de productos, así como en la evaluación de cultivos para determinar la existencia de plagas, solo por citar algunas.

Sin embargo, debido a que las leyes que rigen el aspecto visual de los humanos no son fáciles de describir, el lograr que las computadoras realicen el proceso de reconocimiento se vuelve una tarea extremadamente difícil. Tradicionalmente los modelos se han construido a través de la representación geométrica del contorno de los objetos, dando lugar a los denominados modelos geométricos. Este tipo de representaciones ha sido ampliamente investigado, sin embargo, el reconocimiento de objetos a través de este enfoque presenta algunos problemas que hasta la fecha son motivo de investigación. De manera general, no es posible modelar a un objeto solamente a través de sus propiedades geométricas, pues al hacerlo se pierde una cantidad muy significativa de información acerca de él,

información que puede ser crucial cuando se le quiera reconocer. Además, cabe mencionar que los enfoques geométricos usualmente tratan de representar al objeto en tres dimensiones (3D), lo que requiere una gran cantidad de procesamiento del sistema de reconocimiento.

Recientemente se ha propuesto el uso de modelos basados en la apariencia que presenta el objeto, de tal manera que es posible que mediante el uso de estos, se preserven no solamente las características geométricas del objeto sino también otro tipo de información tal como la textura, el color y otras características contenidas dentro del contorno.

Este capítulo da un panorama general en torno a los avances obtenidos en la aplicación de los modelos basados en apariencia. Se hace énfasis particular en el uso de espacios fundamentales. Este enfoque fue seleccionado, debido principalmente a las razones que se expondrán en las siguientes secciones de este trabajo.

3.2 Selección de la Técnica de Espacios Fundamentales.

Como se ha mencionado a lo largo del desarrollo de este trabajo, se ha seleccionado la técnica de espacios fundamentales para cubrir los objetivos de esta tesis. Esta selección se debe, principalmente, a las siguientes razones:

- Es uno de los enfoques, que ha mostrado convincentemente la viabilidad del uso de la apariencia, en el reconocimiento de objetos. Esto se desprende de los resultados obtenidos por

Nene, Nayar y Murase, reportados en [Nene, 1994] y [Nene & Nayar, 1994].

- El proceso de creación de los modelos de los objetos es independiente de la geometría, textura, color, etc. de los objetos, y sólo es restringido a que los objetos sean rígidos y monomórficos. Esto significa que este proceso es el mismo para cualquier objeto rígido y monomórfico, pues no requiere el ajuste de ningún parámetro dependiente del tipo de objeto a modelar.

- Una vez creados los modelos, se requiere de un espacio mínimo para almacenarlos. Esto se debe a que este enfoque incorpora un esquema de comprensión, que permite modelar los objetos con mucha menos información que la contenida en las imágenes empleadas para crear los modelos.

- Permite reconocer los objetos en tiempo real, una vez creados los modelos. Si se descarta el tiempo requerido para la construcción de los modelos (que se realiza una sola vez), el proceso de reconocimiento se reduce al cálculo de algunos productos internos (ó productos punto), y a una búsqueda dentro de un espacio multidimensional.

- Se han hecho contribuciones posteriores a su propuesta original, lo que hace que sea un método cada vez más robusto (por ejemplo, Ohba e Ikeuchi en *"Recognition of the Multi Specularity Objects using the Eigen-Window"*[4], proponen una

[4] Ohba, K., Ikeuchi, K., *"Recognition of the Multi Specularity Objects using the Eigen-Window"*, Proceedings of International Conference on Pattern Recognition, August 1996.

estrategia para el reconocimiento de objetos parcialmente ocultos, siguiendo este enfoque).

De acuerdo a esta selección, en la siguiente sección se presenta el método propuesto para solucionar la hipótesis inicial, así como una solución viable al problema planteado al inicio del trabajo.

3.3 Construcción Formal del Sistema.

3.3.1 Normalización de las Imágenes.

La construcción del Sistema de Reconocimiento sigue los criterios expuestos en la sección 2.2 del capítulo anterior. Inicialmente se tiene un conjunto de imágenes de prueba al que llamaremos *conjunto de prueba*.

Estas imágenes forman parte de la base de datos de imágenes de COIL (*Columbia Object Image Library*). *Department of Computer Science, Columbia University*. En total, se cuenta con 370 imágenes distribuidas en 5 objetos distintos.

Como se puede observar en la Figura 3.3.1.1, cada imagen es adquirida con cierto grado de libertad, de manera tal que en conjunto las tomas de los objetos con 5 grados de diferencia entre cada una, representan un objeto en rotación.

En un principio, las imágenes no están procesadas, es decir, carecen de una normalización adecuada que permita iniciar el procesamiento en forma directa.

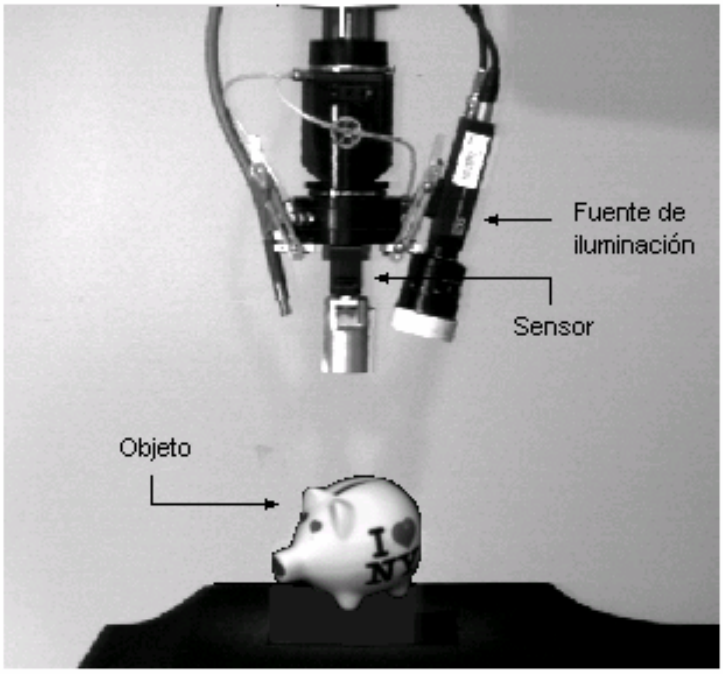


Figura 3.3.1.1. Adquisición de las imágenes que serán normalizadas para la tarea de reconocimiento.

Las imágenes de este conjunto de prueba no son cuadradas, es decir no son de un tamaño *N* x *N*, donde *N* representa el número de *pixeles* a lo ancho y largo de la imagen.

Debido a esto, el proceso de normalización que se presenta genera una imagen perfectamente cuadrada, de 128 x 128 *pixeles* específicamente.

La normalización procesa un conjunto de 25 imágenes como máximo (Figura 3.3.1.2) y una sola imagen como mínimo. Estás imágenes son almacenadas para continuar con el procesamiento cuando se desee.

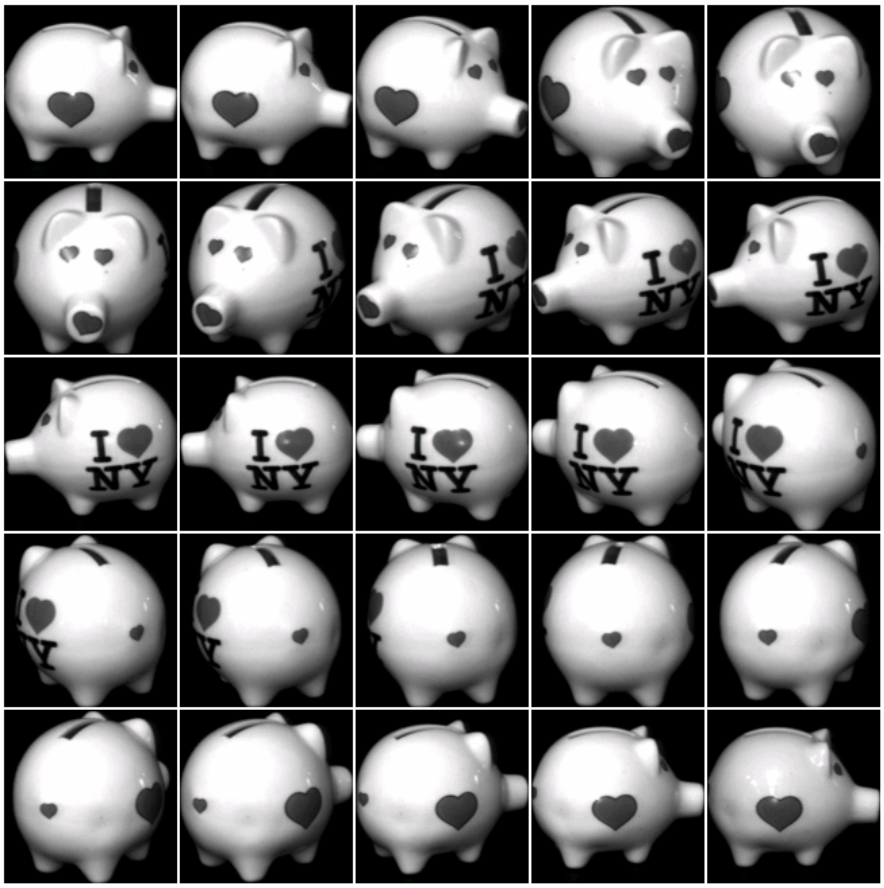


FIGURA 3.3.1.2. Imágenes de COIL (Columbia Object Image Library). Estas imágenes corresponden a las 25 tomas utilizadas para la implementación del sistema.

El algoritmo que se encarga de normalizar el conjunto de prueba se resume a continuación:

- Es necesario hacer la lectura de cada una de las imágenes del conjunto de prueba, a la cuál se le denominará *Imagen*.

- Se aplica una segmentación a *Imagen* mediante el umbral por histéresis, de esta manera obtenemos otra imagen con ciertas limitaciones en sus niveles de gris, supongamos que la imagen obtenida se denomina *Imagen Segmentada*.

- Es necesario determinar el tamaño de la región de interés contenida dentro de la imagen, por lo que es conveniente determinar el *rectángulo mas pequeño* de la *Imagen Segmentada* obtenida en el paso anterior. Mediante la determinación de dicho rectángulo obtenemos las coordenadas x_1, x_2, y_1, y_2 que lo conforman. Es fácil notar que el ancho y el largo de nuestra región esta dado por x_2-x_1 y y_2-y_1, respectivamente.

  Ahora bien, el aspecto fundamental de esta normalización consiste en la manipulación de las coordenadas obtenidas, de tal manera que aseguremos la obtención de una imagen perfectamente cuadrada.

  Definamos *Ancho_Reg* = x_2-x_1 y *Largo_Reg* =y_2-y_1.

  Sí *Ancho_Reg* < *Largo_Reg*

  $x_1 = x_1 - (Largo\_Reg - Ancho\_Reg)/2;$
  $x_2 = x_2 + (Largo\_Reg - Ancho\_Reg)/2;$

  $Nuevo\_Ancho\_Reg = x_2 - x_1$

  $x_2 = (Nuevo\_Ancho\_Reg \mathrel{!}= Largo) \ ? \ x_2 + 1 : x_2$

  De lo contrario

  $y_1 = y_1 - (Ancho\_Reg - Largo\_Reg)/2;$
  $y_2 = y_2 + (Ancho\_Reg - Largo\_Reg)/2;$

  $Nuevo\_Largo\_Reg = y_2 - y_1$

  $y_2 = (Nuevo\_Ancho\_Reg \mathrel{!}= Largo) \ ? \ y_2 + 1 : y_2$

- La manipulación de las coordenadas es muy clara en la parte anterior, ya que al incrementar el valor de una coordenada en *x* o en *y* estamos aumentando un píxel a cualquiera de los extremos de la región. La condición utilizada para calcular un nuevo valor de x_2 ó y_2 nos permite asegurar la obtención de una imagen cuadrada.

- Una vez que se determina una imagen perfectamente cuadrada es conveniente recortarla de la *Imagen* original. De esta manera estaríamos desechando el fondo de la imagen, el cuál no es útil para el procesamiento posterior. La imagen recortada se denomina *Nueva Imagen*. Hasta el momento hemos logrado normalizar a la imagen de cierta manera, es decir que conseguimos eliminar el fondo, sin embargo, es conveniente que el proceso de normalización se extienda un poco más todavía. Es recomendable obtener una imagen de 128 x 128 *pixeles* que facilite la continuación de nuestro procesamiento.

- Debido a lo anterior, es conveniente utilizar ciertas transformaciones que nos permitan escalar nuestra *Nueva Imagen* a un tamaño de 128 x 128 *pixeles*. Para este caso, se utiliza un *mapa de afinación* obtenido como resultado de aplicar una *matriz de escalado*. De ser necesario la imagen resultante es llenada con ceros (niveles oscuros) afuera de la región de la imagen original. Generalmente los puntos transformados mienten sobre las coordenadas de los *pixeles*, a causa de este problema es utilizado un esquema apropiado de interpolación, específicamente la *Interpolación Gaussiana*[5]. Por último, cabe mencionar que se deben tener presentes dos aspectos muy importantes para el uso de la interpolación, la rapidez y la calidad.

Las imágenes normalizadas mediante este proceso tienen el aspecto de la Figura 3.3.1.3. Como se puede observar, la imagen resultado es perfectamente cuadrada y solo denota la región de interés para nuestro análisis.

FIGURA 3.3.1.3. Imágenes obtenidas mediante el proceso de normalización descrito en esta sección.

Una vez que se tiene todas las imágenes normalizadas, es recomendable almacenarlas en un directorio específico para facilitar su lectura posterior.

[5] Ver Apéndice C para una mejor comprensión de la función de Interpolación usada en esta sección.

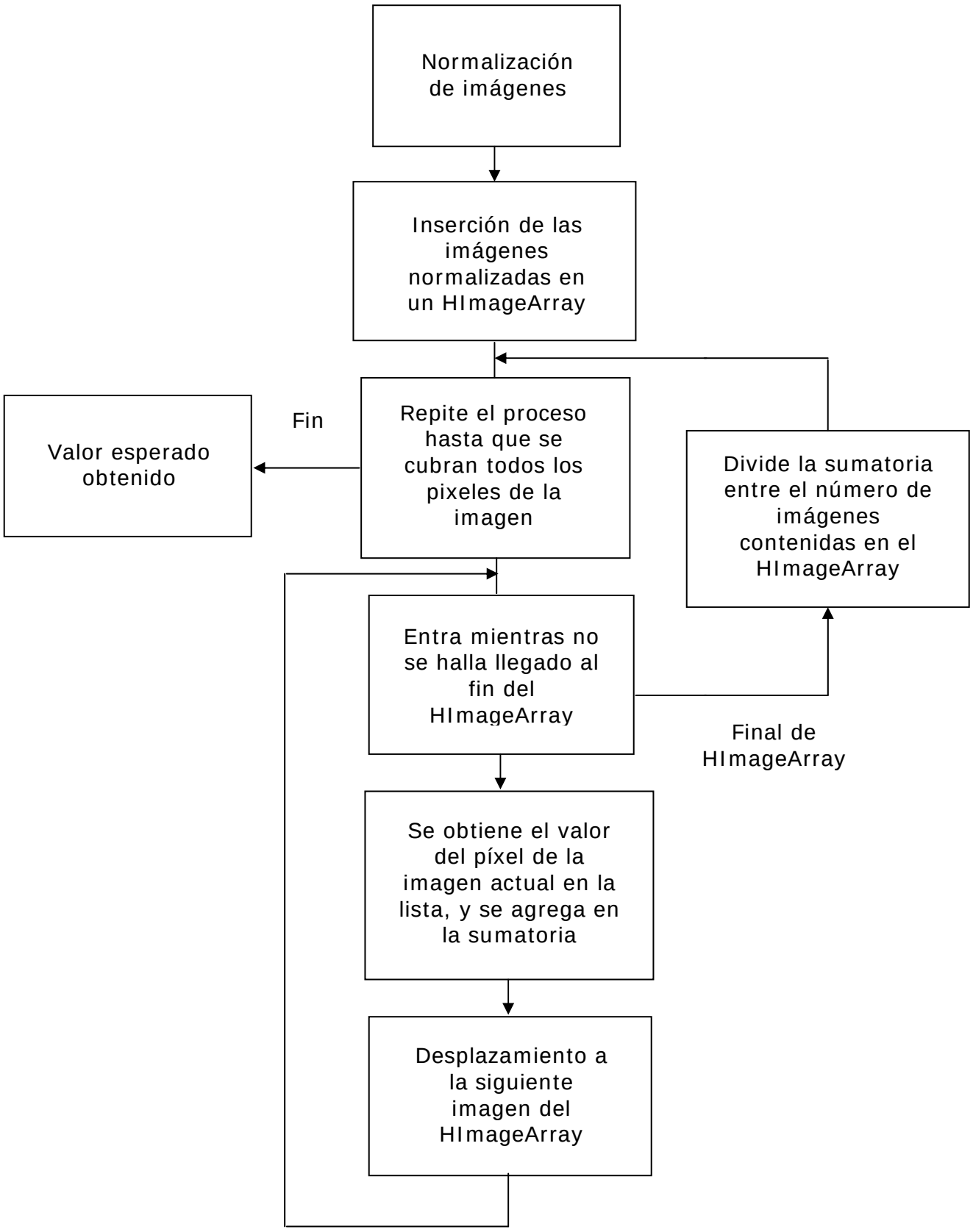
3.3.2 Cálculo del Valor Esperado.

Para iniciar la construcción del modelo es necesario obtener el promedio de las *M* imágenes normalizadas que forman parte de un objeto. Las imágenes deben ser vistas como columnas Γ_M y el resultado obtenido se debe almacenar en un contenedor que facilite las operaciones del álgebra lineal que se tendrán que aplicar, por ejemplo en una matriz.

Es importante mencionar que esta tesis emplea una metodología de programación que permite expresar estructuras de datos, algoritmos y optimizaciones para el álgebra lineal, denominada *Matrix Template Library* (MTL). [Siek, 1999]. La meta de MTL es facilitar el desarrollo de librerías y aplicaciones de la computación científica. En adición, las técnicas de programación utilizadas son altamente aplicables y útiles para reducir costos de desarrollo.

De manera similar, se utiliza el concepto de una imagen de tipo *HImage* que funciona como una matriz de números reales a la que se le pueden aplicar determinadas operaciones matemáticas. Un arreglo de imágenes de tipo *HimageArray* tiene la funcionalidad de una lista de imágenes que puede ser indexada como cualquier arreglo.

La obtención del valor esperado no es más que una simple operación de promedio entre los *n pixeles* de cada imagen , el valor que se obtiene es colocado en la posición *n* de una matriz de MTL. A continuación se resume este proceso con un sencillo diagrama que indica los procedimientos implicados para determinar una sumatoria de valores.


Normalización de imágenes
Inserción de las imágenes normalizadas en un HImageArray
Repite el proceso hasta que se cubran todos los pixeles de la imagen
Fin
Valor esperado obtenido
Divide la sumatoria entre el número de imágenes contenidas en el HImageArray
Entra mientras no se halla llegado al fin del HImageArray
Final de HImageArray
Se obtiene el valor del píxel de la imagen actual en la lista, y se agrega en la sumatoria
Desplazamiento a la siguiente imagen del HImageArray

El valor esperado queda almacenado en una matriz M_x de MTL la cuál será utilizada para determinar la matriz de covarianza.

Para fines de ilustración, la Figura 3.3.2.1 muestra el valor esperado para las 9 imágenes mostradas en la Figura 3.3.1.1.

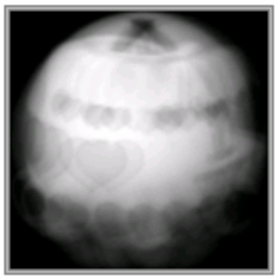
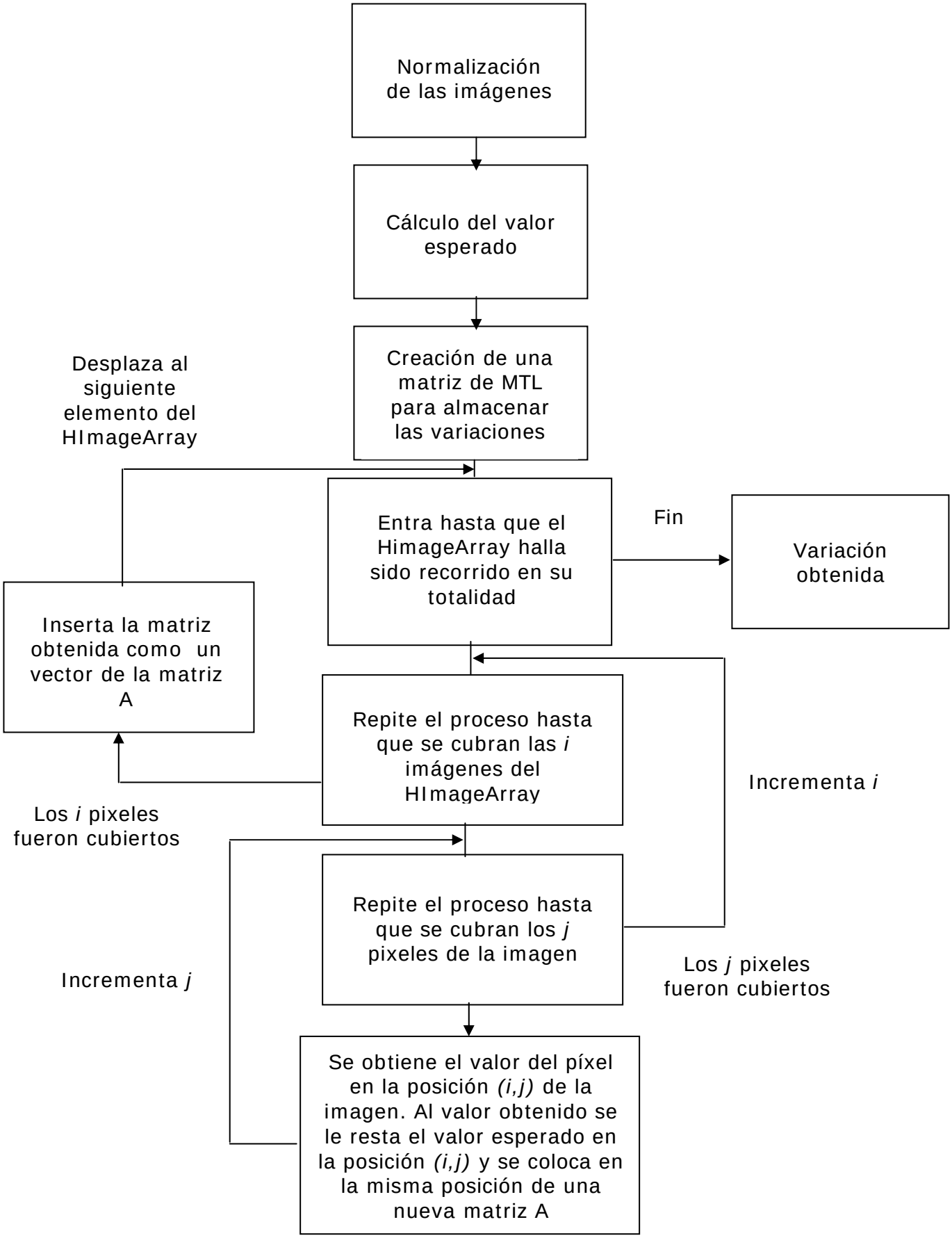


FIGURA 3.3.2.1. Imagen obtenida mediante el proceso de obtención del valor esperado descrito en esta sección.

3.3.3 Cálculo de la Variación entre Imágenes.

El siguiente paso es determinar la variación entre las imágenes normalizadas y el valor esperado obtenido. Cabe mencionar que las imágenes deberán ser vistas como una matriz donde los valores de los *pixeles* determinarán la variación buscada.

El diagrama siguiente ilustra el proceso implementando para realizar tal función. De igual forma, los procedimientos anteriores siguen indicándose para una mayor comprensión, los valores utilizados en esta sección serán utilizados como parámetros para los procesos posteriores.


Normalización de las imágenes
Cálculo del valor esperado
Creación de una matriz de MTL para almacenar las variaciones
Desplaza al siguiente elemento del HImageArray
Entra hasta que el HimageArray halla sido recorrido en su totalidad
Fin
Variación obtenida
Inserta la matriz obtenida como un vector de la matriz A
Repite el proceso hasta que se cubran las *i* imágenes del HImageArray
Incrementa *i*
Los *i* pixeles fueron cubiertos
Repite el proceso hasta que se cubran los *j* pixeles de la imagen
Incrementa *j*
Los *j* pixeles fueron cubiertos
Se obtiene el valor del píxel en la posición *(i,j)* de la imagen. Al valor obtenido se le resta el valor esperado en la posición *(i,j)* y se coloca en la misma posición de una nueva matriz A

3.3.4 Cálculo de la Matriz de Covarianza.

Para estos fines se introducirá una nueva matriz que se usará para determinar sólo los valores propios necesarios y sus correspondientes vectores ortonormales. Esta matriz se denomina *L*. [Turk & Pentland, 1991]

Recordando que:

$$\Phi_i = \Gamma_i - m_x \quad y \quad A = \left(\Phi_1 \Phi_2 \ldots \Phi_M\right)$$

Teniendo lo anterior en cuenta, se puede rescribir C_x en términos de *A*:

$$C_x = \frac{1}{M} \sum_{n=1}^{M} \Phi_n \Phi_n^T = AA^T$$

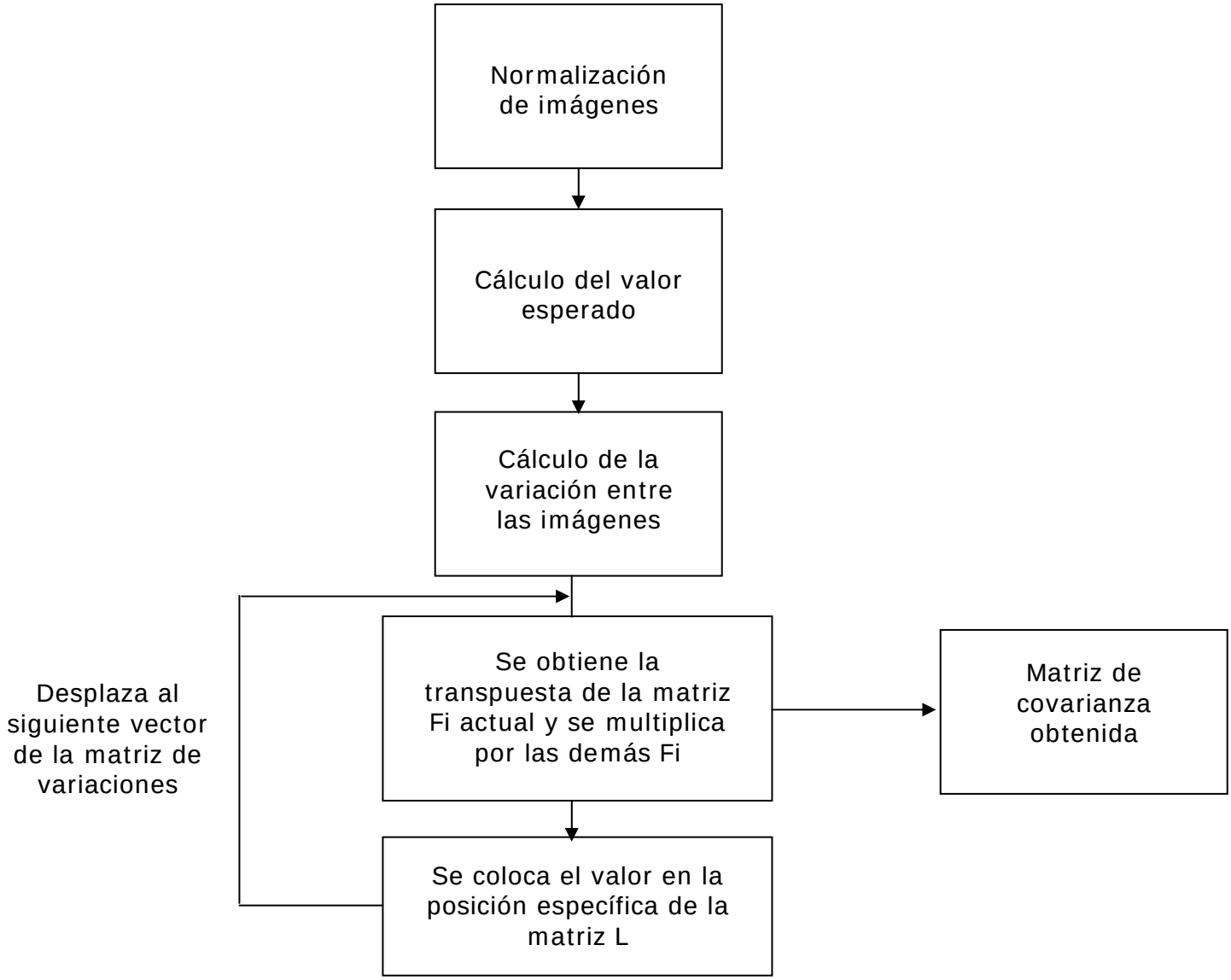
La matriz *L* se define como:

$$L = A^T A = \begin{vmatrix} \Phi_1^T \Phi_1 & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \Phi_M^T \Phi_M \end{vmatrix} \quad tal\ que\ L_{mn} = \Phi_m^T \Phi_n$$

Y da como resultado una matriz de tamaño *M* x *M*. Es importante notar que:

$$\Phi_m^T \Phi_n = \Phi_n^T \Phi_m \quad donde\ m, n \in [1, M] \Leftrightarrow L\ es\ simétrica$$

El diagrama siguiente ilustra el proceso implementando para realizar tal función.



3.3.5 Obtención de los Valores Propios y Vectores Propios.

La rotación básica de *Jacobi* $\boldsymbol{P}_{pq}$ es una matriz de la forma:

$$P_{pq} = \begin{vmatrix} 1 & & & & & & \\ & \ldots & & & & & \\ & & c & \ldots & s & & \\ & & . & 1 & . & & \\ & & -s & \ldots & -c & & \\ & & & & & \ldots & \\ & & & & & & 1 \end{vmatrix}$$

Todos los elementos en la diagonal son la unidad excepto los dos elementos *c* en las filas (y en las columnas) *p* y *q*. Todos los elementos fuera de la diagonal son cero excepto los dos elementos *s* y *–s*. Los números *c* y *s* son el *coseno* y el *seno* de un ángulo de rotación ϕ, por lo que $c^2 + s^2 = 1$.

Una rotación de este tipo, es usada para transformar la matriz ***A*** de acuerdo a

$$A' = P_{pq}^{T}.A.P_{pq} \qquad \text{[Ec. 1]}$$

$P_{pq}^{T}.A$ cambia solo las filas *p* y *q* de ***A***, mientras que $A.P_{pq}$ cambia solo las columnas *p* y *q*. Es importante mencionar que los subíndices *p* y *q* no denotan a los componentes de $\boldsymbol{P}_{\boldsymbol{pq}}$, pero se debe tener presente que dependiendo de la rotación que se haga, estos nos indican a que filas y columna afectó tal rotación. Así, los elementos cambiantes de ***A*** están solamente en las filas *p* y *q* y las columnas se indican abajo:

$$A' = \begin{vmatrix} & \cdots & a'_{1p} & \cdots & a'_{1q} & \cdots & \\ \cdot & & \cdot & & \cdot & & \cdot \\ a'_{p1} & \cdots & a'_{pp} & \cdots & a'_{pq} & \cdots & a'_{pn} \\ \cdot & & \cdot & & \cdot & & \cdot \\ a'_{q1} & \cdots & a'_{qp} & \cdots & a'_{qq} & \cdots & a'_{qn} \\ \cdot & & \cdot & & \cdot & & \cdot \\ & \cdots & a'_{np} & \cdots & a'_{nq} & \cdots & \end{vmatrix}$$

Multiplicando la ecuación [Ec. 1] y usando la simetría de ***A***, se obtiene las siguientes fórmulas:

$$a'_{rp} = ca_{rp} - sa_{rq} \qquad r \neq p, r \neq q \qquad \text{[Ec. 2]}$$

$$a'_{rq} = ca_{rq} + sa_{rp} \qquad \text{[Ec. 3]}$$

$$a'_{pp} = c^2 a_{pp} + s^2 a_{qq} - 2sca_{pq} \qquad \text{[Ec. 4]}$$

$$a'_{qq} = s^2 a_{pp} + c^2 a_{qq} + 2sca_{pq} \qquad \text{[Ec. 5]}$$

$$a'_{pq} = \left(c^2 - s^2\right)a_{pq} + sc\left(a_{pp} - a_{qq}\right) \qquad \text{[Ec. 6]}$$

La idea del método de *Jacobi* es convertir en cero los elementos que están fuera de la diagonal mediante series de rotaciones. Por consiguiente, haciendo $a'_{pq} = 0$, se obtiene que la ecuación [Ec. 6] proporciona la siguiente expresión para el ángulo de rotación ϕ

$$\theta \equiv \cot 2\phi \equiv \frac{c^2 - s^2}{2sc} = \frac{a_{qq} - a_{pp}}{2a_{pq}}$$

Si se asume que $t \equiv s/c$, la definición de θ puede ser reescrita como:

$$t^2 + 2t\theta - 1 = 0$$

La raíz pequeña de la ecuación corresponde a un ángulo de rotación menor que $\Pi/4$ en magnitud; esto permite que en cada etapa se obtenga la reducción más estable. Usando la forma de la fórmula cuadrática con el discriminante en el denominador, se puede escribir esta raíz como

$$t = \frac{\text{sgn}\,(\theta)}{|\theta| + \sqrt{\theta^2 + 1}}$$

Si θ es más grande que θ^2 puede presentarse un desbordamiento en la computadora, por lo que es recomendable hacer *t = 1/(2*θ*).* Entonces se tiene que:

$$c = \frac{1}{\sqrt{t^2 + 1}}$$
$$s = tc$$

Cuando se utilizan las ecuaciones [Ec. 2] – [Ec. 6], se rescriben para reducir al mínimo cualquier error. La ecuación [Ec. 6] es reemplazada por:

$$a'_{pp} = 0 \qquad \text{[Ec. 7]}$$

La idea en las ecuaciones de resta es fijar la nueva cantidad igual a la vieja cantidad mas una pequeña corrección. De esta manera se pueden utilizar las ecuaciones [Ec. 6] y [Ec. 7] para eliminar a_{qq} de [Ec. 4], obteniendo:

$$a'_{pp} = a_{pp} - ta_{pq}$$

Similarmente,

$$a'_{qq} = a_{qq} + ta_{pq}$$
$$a'_{rp} = a_{rp} - s\left(a_{rq} + \tau a_{rp}\right)$$
$$a'_{rq} = a_{rq} + s\left(a_{rp} - \tau a_{rq}\right)$$

donde τ *(= tan* ϕ */ 2)* esta definida por:

$$\tau = \frac{s}{1+c}$$

Se puede notar la convergencia del método de *Jacobi* considerando la suma de los cuadrados de los elementos fuera de la diagonal:

$$S = \sum_{r \neq s} \left| a_{rs} \right|^2$$

El grupo de ecuaciones [Ec. 2] – [Ec. 6] mostrado anteriormente implica que:

$$S' = S - 2\left| a_{pq} \right|^2$$

Desde que la transformación es ortogonal, la suma de los cuadrados de los elementos de la diagonal crece proporcionalmente a $2|a_{pq}|^2$. La secuencia de *S's*, sin embargo, decrece monolíticamente. Puesto que la secuencia es limitada bajo cero, y puesto que podemos elegir a_{pq} para cualquier elemento que queramos, la secuencia puede hacerse converger a cero.

Eventualmente se va obteniendo la matriz ***D***. Los elementos de la diagonal nos dan los valores propios de la matriz original ***A***, puesto que:

$$D = V^T . A . V$$

donde $V = P_1 . P_2 . P_3 \ldots$

las P_i*'s* sucesivas son las matrices de rotación de *Jacob*i. Las columnas de ***V*** son los valores propios (puesto que ***A . V = V . D***). Estos pueden ser computados aplicando

$$V' = V . P_i \qquad \text{[Ec. 8]}$$

en cada etapa del cálculo, donde V es inicialmente la matriz identidad. En detalle, la ecuación [Ec. 8] es:

$$\begin{aligned} v'_{rs} &= v_{rs} \qquad (s \neq p, s \neq q) \\ v'_{rp} &= cv_{rp} - sv_{rq} \\ v'_{rq} &= sv_{rp} + cv_{rq} \end{aligned}$$

Se pueden rescribir estas ecuaciones en términos de τ como se hizo anteriormente para minimizar el error.

Una vez definidos los términos que implican la formulación de nuestro espacio fundamental, es necesario conocer más a fondo el proceso que se tiene que seguir para cubrir tal tarea. Debido a esto, las siguientes secciones tratan de dar una explicación mas clara sobre la aplicación de esta técnica al problema del reconocimiento.

El algoritmo que calcula los valores y vectores propios esta basado en la implementación de los términos definidos en esta sección de la tesis. Dicho algoritmo utiliza dos grandes refinamientos:

- En los primeros tres recorridos se realiza la rotación de *pq* sólo sí $\left| a_{pq} \right| > \in$ para un cierto valor de umbral

$$\in = \frac{1}{5}\frac{S_0}{n^2}$$

donde S_0 es la suma de los módulos en la diagonal,

$$S_0 = \sum_{r<s} \left| a_{rs} \right|$$

- Después de cuatro barridos, sí $\left| a_{pq} \right| << \left| a_{pp} \right| \; y \; \left| a_{pq} \right| << \left| a_{qq} \right|$, se hace $\left| a_{pq} \right| = 0$ y salta la rotación. El criterio usado en la

comparación es $\left| a_{pq} \right| < 10^{-(D+2)} \left| a_{pp} \right|$, donde *D* es el número de dígitos decimales significativos, y de manera similar para $\left| a_{qq} \right|$.

El algoritmo de *Jacobi* recibe como parámetro de entrada a la matriz de covarianza *L* de tamaño *n* x *n*. Para la salida, los elementos de la superdiagonal de *L* son destruidos, sin embargo la diagonal y subdiagonal permanecen sin cambios y proporcionan información completa sobre la matriz simétrica original *L*.

El vector *d[1...n]* utilizado, regresa los valores propios de *L*. Durante el cómputo, este vector almacena la diagonal actual de *L*. La matriz *V(1..n)(1...n)* devuelve el vector propio normalizado que pertenece a la *k*-ésima columna de *d[k].*

La variable *nrot* representa el número de rotaciones que serán necesarias para alcanzar la convergencia.

3.3.6 Obtención de las Eigenimágenes.

Considerando el vector propio P_i y su valor propio asociado, los cuáles fueron calculados aplicando el método de *Jacobi* a la matriz *L*, entonces:

$$LP_i = \lambda_i P_i \Rightarrow A^T A P_i = \lambda_i P_i \Rightarrow AA^T A P_i = \lambda_i A P_i \Rightarrow C_x A P_i = \lambda_i A P_i$$

Es importante notar que $\boldsymbol{AP_i}$ es un vector columna.

Observando esta última consecuencia y comparándola con la definición de un vector propio, se puede notar que los *M* vectores propios $\boldsymbol{AP_i}$ son vectores propios de C_x. Es decir, los vectores propios calculados para *L* también son valores propios de *C*.

De esta manera, es posible construir el espacio de imágenes utilizando el concepto definido por *Ammoura*: [Ammoura, 1997]

$$Eigenimagen\ u_i = \sum_{k=1}^{M} P_{ik} \Phi_k$$

Como se puede observar, las eigenimágenes son equivalentes en número a los vectores propios de *L* y mantienen la misma dimensión que las imágenes de entrada (128^2 x 1, en nuestro caso particular). El vector propio P_i es utilizado para generar la eigenimagen u_i.

$$u = AP = \overline{\Phi_0 \quad \Phi_1 \quad \ldots \quad \Phi_M} \begin{vmatrix} P_0 \\ p_1 \\ \ldots \\ P_m \end{vmatrix}$$

$$u = \left| \begin{array}{cccc|c} \Phi_0[0] & \Phi_1[0] & ... & \Phi_M[0] & P[0] \\ \Phi_0[1] & \Phi_1[1] & ... & \Phi_M[1] & P[1] \\ ... & ... & ... & ... & ... \\ \Phi_0[128^2-1] & \Phi_1[128^2-1] & ... & \Phi_M[128^2-1] & P[M] \end{array} \right|$$

A continuación se muestra un conjunto de imágenes que ilustra el proceso descrito en esta sección. Estas imágenes son las proyecciones de los vectores al espacio fundamental, las eigenimágenes (Figura 3.3.6.1).

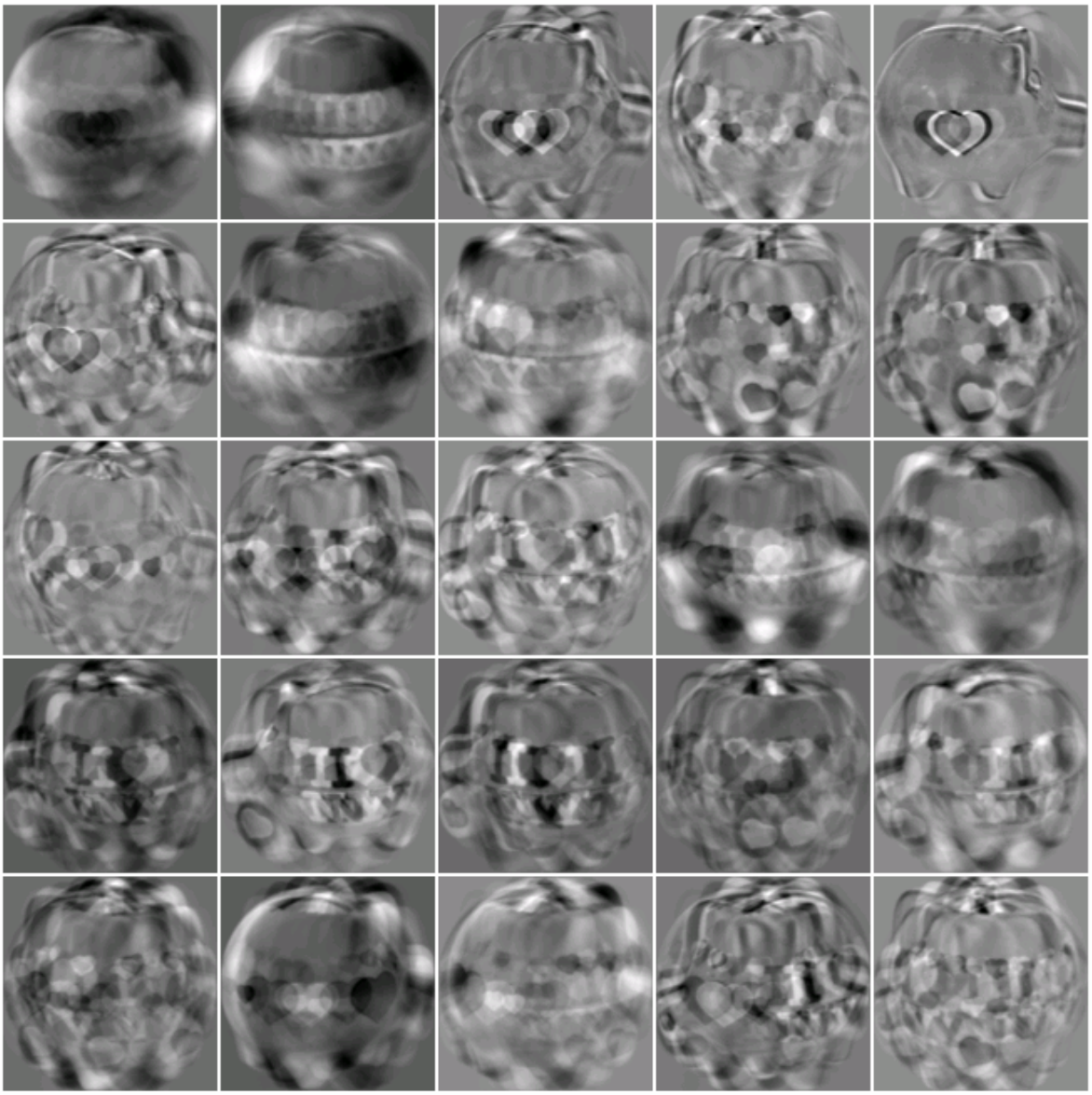
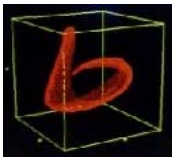


Figura 3.3.6.1. Eigenimágenes obtenidas por el proceso descrito en esta sección. Estos resultados corresponden al conjunto de imágenes normalizadas de la figura 3.3.1.1.

3.3.7 Proyección y Reconocimiento.

Como se esperaba, las eigenimágenes definen un conjunto de puntos donde cada uno es una combinación lineal de la base $\{U_i\}$. Después de inicializar el sistema con el conjunto de prueba y las eigenimágenes, un conjunto de clases puede ser definido. Cada clase es un vector de M x 1 que representa la contribución de cada eigenimagen.



Apéndice A.

Fundamentos de Álgebra Lineal.

Debido al contenido matemático de este trabajo es necesario definir ciertos términos que son esenciales para el desarrollo del mismo. De esta manera se hace más fácil la comprensión de este trabajo de tesis.

Representación en espacios vectoriales.

La idea de usar parejas de números para localizar puntos en el plano y ternas para localizar puntos en el espacio tridimensional se concibió por primera vez con claridad a mediados del siglo XVII. A fines del siglo XIX, matemáticos y físicos empezaron a darse cuenta que no era necesario quedarse en las ternas. Se reconoció que los conjuntos ordenados de cuatro números (a_1, a_2, a_3, a_4) se podrían considerar como puntos en el espacio "tetradimensional", los conjuntos ordenados de cinco números $(a_1, a_2, a_3, a_4, a_5)$ como puntos en el espacio "pentadimensional", etc.

Un espacio vectorial (o espacio lineal) *V* sobre un campo[6] *F* consiste de un conjunto de ***n-adas ordenadas*** en el que están definidas dos operaciones (llamadas adición y multiplicación por escalares, respectivamente), tal que para cualquier par de elementos *x* y *y* en *V* exista

[6] La palabra "campo" se puede interpretar como "campo de los números reales" (denotado por R) o "campo de los números complejos".

un elemento único $x + y$ en V, y para cada elemento a en F y cada elemento x en V exista un elemento único ax en V.

Los elementos $x + y$ y ax se denominan, respectivamente, suma de x y y y el producto de a y x. [Spence & Insel, 1986]

Los elementos del campo F se llaman ***escalares*** y los elementos del espacio vectorial V se llaman ***vectores***. En nuestro caso particular, no se debe confundir la palabra "vector" con la entidad física ya conocida, la palabra "vector" la utilizaremos para describir cualquier elemento en un espacio vectorial.

Un objeto de la forma $(a_1, \ldots, a_n)$, donde los valores o entradas a_i son elementos de un campo F, se denomina n-dimensional con valores de F.

Dos n-dimensionales $(a_1, \ldots, a_n)$ y $(b_1, \ldots, b_n)$ se definen como iguales si y sólo si $a_i = b_i$ para $i = 1, 2, 3, \ldots, n$.

Es común que en el estudio del espacio tridimensional, el símbolo (a_1, a_2, a_3) tenga dos interpretaciones geométricas distintas. Puede interpretarse como un punto, en cuyo caso a_1, a_2 y a_3 son las coordenadas (Figura 1.a) o se puede interpretar como un vector, en cuyo caso a_1, a_2 y a_3 son las componentes (Figura 1.b). Por tanto, se puede concluir que una *n-ordenada* $(a_1, \ldots, a_n)$ se puede concebir como un "punto generalizado" o como un "vector generalizado" desde el punto de vista matemático, la distinción no tiene importancia.

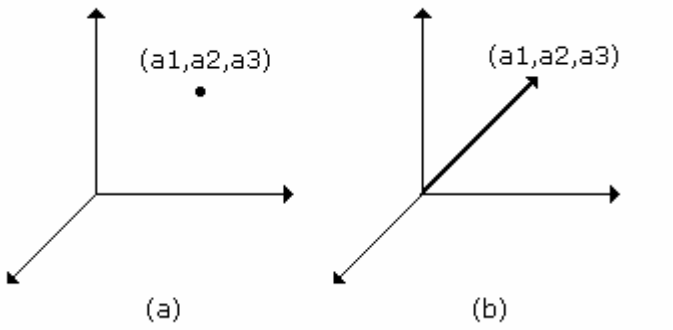


Figura 1. Interpretación del símbolo (a_1, a_2, a_3) en el espacio tridimensional.

Valores propios y vectores propios.

En muchos problemas de ciencias y matemáticas, se da un operador lineal $T : V \to V$ y es importante determinar aquellos escalares λ para los cuales la ecuación $Tx = \lambda x$ tiene soluciones diferentes de cero. En esta sección se analiza este problema.

Definición. Si A es una matriz de n x n, entonces se dice que un vector diferente de cero x en R^n es un ***eigenvector*** de A, si Ax es un múltiplo escalar de x; es decir,

$$Ax = \lambda x$$

para algún escalar λ. El escalar λ se denomina ***eigenvalor*** de A y se dice que x es un vector propio correspondiente a λ.

Uno de los significados de la palabra “eigen” en alemán es el de “propio”; por ello, también se le da a los eigenvalores los nombres de ***valores propios***, ***valores característicos***, o bien, ***raíces latentes***. [Anton, 1993]

Por ejemplo, el vector $x = \begin{vmatrix} 1 \\ 2 \end{vmatrix}$ es un vector propio de

$$A = \begin{vmatrix} 3 & 0 \\ 8 & -1 \end{vmatrix}$$

correspondiente al valor propio $\lambda = 3$, debido a que

$$A=\begin{vmatrix} 3 & 0 \\ 8 & -1 \end{vmatrix}\begin{vmatrix} 1 \\ 2 \end{vmatrix}=\begin{vmatrix} 3 \\ 6 \end{vmatrix}=3x$$

Los valores propios y los vectores propios tienen una interpretación geométrica útil en R^2 y R^3. Si λ es un valor propio de A correspondiente a x, entonces $Ax = \lambda x$, de modo que la multiplicación por A dilata a x, contrae a x o invierte la dirección de x, dependiendo del valor de λ (Ver Figura 2).

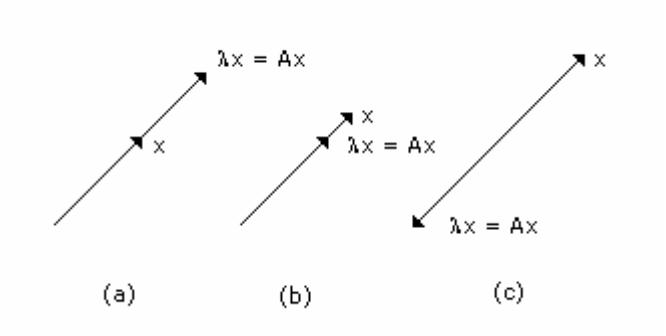


Figura 2. (a) Dilatación, $\lambda > 1$. (b) Contracción, $0 < \lambda < 1$. (c) Inversión de la dirección, $\lambda < 0$.

Para encontrar los valores propios de una matriz A de n x n, se vuelve a escribir $Ax = \lambda x$ como

$$Ax = \lambda Ix$$

o, lo que es equivalente,

$$(\lambda I - A)x = 0$$

Para que λ sea un valor propio, debe haber una solución diferente de cero de esta ecuación. Sin embargo, la ecuación anterior tendrá una solución diferente de cero si y sólo si

$$\det(\lambda I - A) = 0$$

Esto se conoce como ***ecuación característica*** de *A*; los escalares que satisfacen esta ecuación son los valores propios de *A*. Cuando se desarrolla, el determinante $\det(\lambda I - A)$ es un polinomio en λ conocido como ***polinomio característico*** de *A*.

Por ejemplo, para hallar los valores propios de la matriz

$$A = \begin{vmatrix} 3 & 2 \\ -1 & 0 \end{vmatrix}$$

Dado que

$$\lambda I - A = \lambda \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix} - \begin{vmatrix} 3 & 2 \\ -1 & 0 \end{vmatrix} = \begin{vmatrix} \lambda - 3 & 2 \\ -1 & \lambda \end{vmatrix}$$

el polinomio característico de *A* es

$$\det(\lambda I - A) = \det \begin{vmatrix} \lambda - 3 & -2 \\ 1 & \lambda \end{vmatrix} = \lambda^2 - 3\lambda + 2$$

y la ecuación característica de *A* es

$$\lambda^2 - 3\lambda + 2 = 0$$

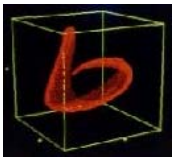
Las soluciones de esta ecuación son $\lambda = 1$ y $\lambda = 2$; estos son los valores propios de A.

Cabe mencionar que existen algunas aplicaciones que requieren escalares complejos y espacios vectoriales complejos. En dichos casos, se permite que las matrices tengan valores propios complejos. Sin embargo, en este trabajo, sólo se consideran valores propios reales.

En muchos problemas prácticos, la matriz A a menudo es demasiado grande como para que convenga determinar la ecuación característica. Como resultado, se aplican diversos métodos de aproximación para obtener los valores propios; en el Capítulo 2 de este trabajo se mencionarán algunos de estos métodos.

Ahora que se ha visto cómo se encuentran los valores propios, se analizará el problema de encontrar los vectores propios. Los vectores propios de A correspondientes a un valor propio λ son los vectores diferentes de cero que satisfacen $Ax = \lambda x$. También se puede decir que los vectores propios correspondientes a λ son los vectores diferentes de cero en el espacio de soluciones de $(\lambda I - A)x = 0$. A este espacio de soluciones se le conoce como ***eigenespacio*** de A correspondiente a λ**.**

Los vectores propios y los valores propios se pueden definir tanto para los operadores lineales como para las matrices. Un escalar λ se denomina ***valor propio*** de un operador lineal $T : V \rightarrow V$, si existe un vector diferente de cero x en V tal que $Tx = \lambda x$. El vector x es un ***vector propio*** de T correspondiente a λ. De modo equivalente, los vectores propios de T que corresponden a λ son los vectores diferentes de cero en el núcleo de $\lambda I - T$. Este núcleo es el ***eigenespacio*** de T correspondiente a λ**.**



Apéndice B.

Interpolación Gaussiana.

Las imágenes digitales puede ser procesadas en una variedad de formas. Una de las más comunes es llamada “filtrado” y consiste en generar una nueva imagen como resultado de procesar los *pixeles* de una imagen existente.

Cada *píxel* en la imagen de salida es computado en función de uno o varios *pixeles* de la imagen original, situada generalmente cerca de la localización del *píxel* de salida. Sí la función usada realiza un tipo de interpolación (lineal, cúbica o *Gaussiana*), entonces el resultado lucirá más “liso” que el original, pero se necesita tener cuidado en que los valores de salida no sean computados de muchos pixeles de entrada, o la imagen que resulta puede obtenerse enmascarada.

Los filtros *Gaussianos* son una clase de filtros lineales con los pesos escogidos de acuerdo a la forma de una función *Gaussiana*. El filtro *Gaussiano* es un muy buen filtro para remover ruido trazado de una distribución normal.[7] La función *Gaussiana* en una dimensión es

$$g(x) = e^{-\frac{x^2}{2\sigma^2}}$$

[7] El hecho de que los pesos sean escogidos de una distribución *Gaussiana* y que el ruido sea también distribuido de esta manera es mera coincidencia.

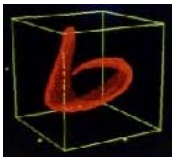
donde el parámetro de la extensión *Gaussiana* σ determina la anchura del *Gaussiano*. Para el procesamiento de una imagen, la función *Gaussiana* discreta en dos dimensiones,

$$g[i, j] = e^{-\frac{(i^2 + j^2)}{2\sigma^2}},$$

es usada como un filtro de alisado. Las funciones *Gaussianas* cumplen cinco propiedades que las hacen útiles en las tareas de visión. Estas propiedades indican que los filtros *Gaussianos* son efectivamente filtros pasa-baja desde la perspectiva de ambos dominios, el espacio y la frecuencia, son eficientes de implementar, y pueden ser utilizados por los ingenieros en aplicaciones prácticas de visión. Las cinco propiedades son sumarizadas a continuación.

1. En dos dimensiones, las funciones *Gaussianas* son rotacionalmente simétricas. Esto significa que la cantidad de alisado realizada por el filtro será la misma en todas las direcciones. En general, los bordes en una imagen no estarán orientados en alguna dirección en particular, esto se sabe por adelantado; consecuentemente, no existe ninguna razón *a priori* para alisar más en una dirección que en otra. La propiedad de simetría rotacional implica que un filtro *Gaussiano* no predispondrá la detección de bordes en una dirección en particular.
2. La función *Gaussiana* tiene un solo lóbulo. Esto significa que un filtro *Gaussiano* reemplaza cada píxel de una imagen con el promedio de los pesos de los pixeles vecinos en semejanza al peso de un vecino que decrece monolíticamente con la distancia desde el píxel central.
3. El lóbulo sencillo en una transformada de *Fourier* o *Gaussiana* significa que la imagen alisada no será corrompida por contribuciones de señales de frecuencia no deseadas, mientras que la mayoría de señales deseables serán retenidas.

4. La anchura, y por lo tanto el grado de alisado, de un filtro *Gaussiano* esta parametrizado por σ, y la relación entre σ y el grado de alisado es muy simple. Un σ muy largo implica un filtro *Gaussiano* muy ancho y en consecuencia un alisado más grande.
5. Los filtros *Gaussianos* largos pueden ser implementados eficientemente porque las funciones *Gaussianas* son separables. Las convoluciones *Gaussianas* bidimensionales pueden realizarse mediante la convolución de la imagen con una función *Gaussiana* unidimensional y después efectuar la convolución del resultado con el mismo filtro unidimensional orientado ortogonalmente usado en la primer etapa. Así, la cantidad de cómputo requerida para un filtro *Gaussiano* 2-D crece linealmente en la anchura del filtro mascará en vez de crecer cuadráticamente.



Apéndice C.

Matrix Template Library.

La construcción de *software* para la computación científica es una tarea difícil. Los códigos son a menudo largos y complejos, y requieren grandes cantidades de conocimiento en el dominio para su construcción. Estos códigos procesan conjuntos grandes de datos por lo que requieren adicionalmente de eficiencia y alto rendimiento. Se requiere de conocimiento considerable de arquitecturas modernas de computadoras y compiladores para hacer las optimizaciones necesarias, lo cual es una tarea intensiva y a lo largo vuelve complicado el código.

Durante la década pasada se mostraron avances significativos en el área de la construcción de este tipo de *software*.

Esta tesis utiliza como una librería, *Matrix Template Library* (MTL) un paquete de alto rendimiento para el álgebra lineal numérica. MTL provee un conjunto variado de operaciones básicas del álgebra lineal, equivalentes al Nivel 1, Nivel 2 y Nivel 3 de BLAS (*Basic Linear Álgebra Routines*), sin embargo MTL trabaja con un conjunto mucho más grande de tipos de datos. MTL es único entre bibliotecas de álgebra lineal porque cada algoritmo (para la mayor parte) es implementado con sólo una función macro. Desde el punto de vista del mantenimiento del software, la reutilización de código da a MTL una ventaja significativa sobre BLAS u otras bibliotecas orientadas a objetos como TNT (*Tools Numerical*

Templates), que todavía tienen diferentes subrutinas para diferentes formatos de matriz. Debido a la reutilización de código proporcionada por la programación genérica, MTL emplea menos líneas de código que *NetLib Fortran* BLAS, mientras que proporciona funciones de mejor funcionamiento. La implementación de MTL es de 8, 284 palabras (de acuerdo a la utilería *wc* de *UNIX*) para algoritmos y de 6, 900 palabras para contenedores densos (*dense containers*). *NetLib BLAS* emplea un total de 154, 495 palabras y las versiones de alto rendimiento de BLAS (con las cuáles MTL es competitivo) son aún mayores.

El encapsulamiento de datos se ha aplicado a la información de la matriz y el vector, lo cual se refleja en una interfaz simple de MTL debido a que la información de entrada y salida esta en términos de los objetos matriz y vector, en lugar de números enteros, números de punto flotante y apuntadores.

MTL define un conjunto de componentes para representar objetos del álgebra lineal, predominantemente matrices y vectores. Los componentes de matrices en MTL cubren la mayoría de los formatos de matrices más comunes usados en la actualidad. Cada formato de matriz provee ventajas en espacio y/ o eficiencia en el tiempo para matrices de cierta estructura diferente de cero.

La Figura 1 muestra el modelo característico que describe los formatos de matrices representados en MTL. Este modelo puede también describirse por medio de una gramática de configuración, mostrada en la Figura 2. En este contexto una *configuración* es una selección de atributos matriciales que especifican un formato particular de matriz.

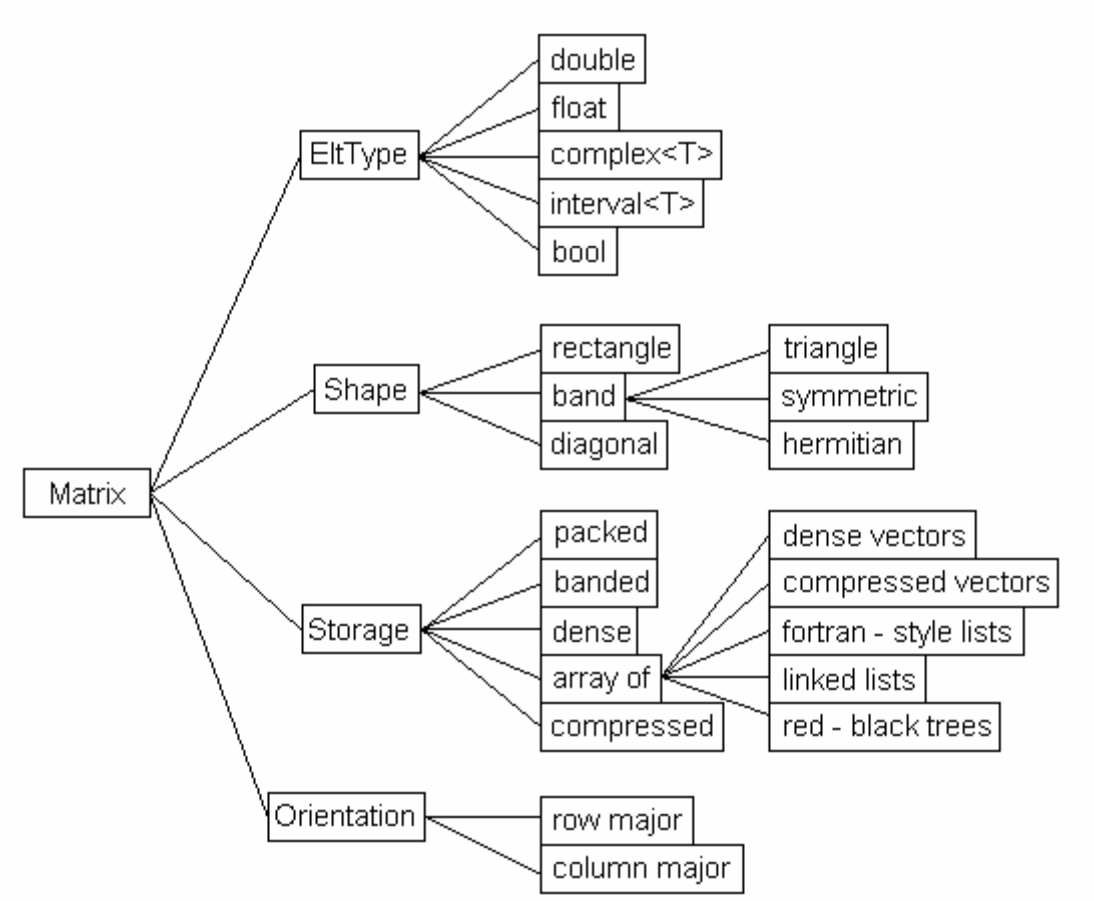
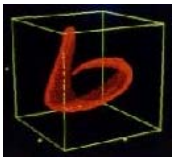


Figura 1. Diagrama característico para formatos comunes de matrices.

```
Matrix:          matrix[EltType, Shape, Storage, Orientation]
EltType:         float | double | complex[EltType] | interval[EltType]
Shape:           rectangle | diagonal | Banded
Banded:          banded | triangle | symmetric | hermitian
Storage:         dense | packed | banded | banded_view |
| compressed | array of OneD | envelope
OneD:            dense vector | compressed vector | pair vector
| fortran-style list | linked list | red-black tree
Orientation:     row major | column major
```

Figura 2. Gramáticas de configuración de MTL



Referencias.

[Alvarado, 1999] Altamirano Robles, Luis Carlos., Altamirano Robles, Leopoldo & Alvarado Mentado, José Matías.
Informe Técnico.
"Reconocimiento de Objetos usando Modelos Basados en Apariencia".
Instituto Politécnico Nacional.
Centro de Investigación en Computación.
Número: 45. Serie: Azul.
Agosto 1999.

[Ammoura, 1997] Ammoura, Ayman H.
"Eigenface".
Topics in Computer Vision.
University of Alberta.
Department of Computing Science.
1997.

[Antón, 1993] Anton, Howard.
Introducción al Álgebra Lineal.
Edición en español.
Drexel University. 1993.

[Baker, 1996] Baker, Simon., Nayar, Shree K. & Murase, Hiroshi.
"Parametric Feature Detection".
Department of Computer Science.
Columbia University. New York, NY. USA.
NNT Basic Research Laboratory.
Morinosato Wakamiya, Atsugi-shi. 1996.

[Chandrasekaran, 1996] Chandrasekaran, S., Manjunath, B. S. Wang, Y. F., Winkeler, J. & Zhang, H. *"An Eigenspace Update Algorithm for Image Analysis".* Computer Science Technical Report. April 1996.

[Domínguez, 1994] Domínguez Torres, Alejandro. *"Procesamiento Digital de Imágenes".* Revista Soluciones Avanzadas. Noviembre de 1994.

[González, 1997] González Rosiles, José Gerardo. *"Classroom Notes".* Georgia Tech. College of Computing. April 2, 1997 - June 4, 1997

[González & Woods, 1996] C. González, Rafael & E. Woods, Richard. Tratamiento Digital de Imágenes. Edición en español. Addison-Wesley Iberoamericana, S.A. 1996.

[Hadley, 1969] Hadley, G. Álgebra Lineal. Linear Álgebra. Fondo Educativo Interamericano. Edición Bilingüe. Colombia, 1969.

[Hernández, 1982] Hernández Lerma, Onésimo. Elementos de Probabilidad y Estadística. Universidad Autónoma Metropolitana. Fondo de Cultura Económica. 1982.

[Jain, 1997] Jain, Ramesh., Kasturi, Rangachar., Schunck, Brian G.

Machine Vision.
McGraw Hill International Editions
Computer Science Series, 1997.

[Kirby & Sirovich, 1990] Kirby, M. & Sirovich, L.
"Application of the Karhunen-Loeve procedure for the characterization of Human Face".
IEEE: Transactions on Pattern Analysis and Machine Intelligence.
1990.

[Kriegman, 1997] Kriegman, David J., Hespanha, Joao P. & Belhumeur, Peter N.
“Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection”.
IEEE Transactions On Pattern Analysis and Machine Intelligence., Vol. 19, No. 17.
July 1997.

[Meyer, 1999] Meyer, Carl D.
Matrix Analysis and Applied Linear Algebra.
Chapter 7.
Editorial Siam. 1999.

[Moghaddam, 1994] Pentland, A., Moghaddam, B. & Starner, T.
"View-based and Modular Eigenspace for face Recognition".
Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition.
1994.

[Murakami & Kumar, 1982] H. Murakami & B. V. K. V. Kumar.
"Efficient Calculation of Primary Images from a set of Images".
IEEE T-PAMI, Volumen 4, Número 5.
September 1982.

[Nayar, 1996] Nayar, Shree K., Murase, Hiroshi. & Nene, Sameer A. *"Parametric Appearance Representation".* Early Visual Learning, New York Oxford. Oxford University Press. 1996.

[Nene, 1994] Nene, S.A., Nayar, S. K. & Murase, H. *"Software Library for Appearance Matching (SLAM)".* Proceedings of ARPA Image Understanding Workshop, Monterey. November 1994.

[Nene & Nayar, 1994] Nene, S.A., Nayar, S. K. & Murase, H. *"SLAM: A Software Library for Appearance Matching (SLAM)".* Technical Report, Columbia University. November 1994.

[Preciado, 1986] Preciado, Gustavo. & Trigueros, María. Álgebra Lineal. Nivel Superior. Programa Nacional de Formación y Actualización de profesores de Matemáticas. Cinvestav del Instituto Politécnico Nacional. Primera Edición. 1986.

[Reignier, 1995] Reignier, Patrick. *"Identifying the Face by Eigenspace Decomposition".* The European Computer Vision Network. MED DST. 1995

[Siek, 1999] Jeremy G. Siek, B. S. *"A Modern Framework for Portable High Performance Numerical Linear Algebra".* Thesis. Department of Computer Science and Engineering. Notre Dame, Indiana.

April 1999.

[Teukolsky, 1992] Press, William H., Teukolsky, Saul A., Vetterling, William T. & Flannery, Brian P.
Numerical Recipes in C.
The Art of Scientific Computing.
Second Edition.
Cambridge University Press. 1992.

[Turk & Pentland, 1991] Turk, M. & Pentland, A.
"Eigenfaces for Recognition".
Journal of Cognitive Neuroscience.
1991.

[Shapiro & Costa, 1994] Shapiro, L. G. & Costa, M. S.
"Appearance-Based 3D Object Recognition ".
Object Representation in Computer Vision.
Vol. I, in Proceedings of International NSF-ARPA Workshop, New York City, NY, USA.
December 1994.

[Zhon, 1998] B. B. Zhon., R. D. Brent., Kahn, M.
One-Sided Jacobi Algorithm for the Symmetric EigenValue Problem
The Australian National University.
Canberra, Australia. 1998.