CIMAT Centro de Investigación en Matemáticas, A.C.

Métodos de Calibración de un Digitalizador Óptico

Tesis

que para obtener el grado de

Maestro en Ciencias

presenta

Edgar Román Arce Santana

Coodirectores de Tesis
Dr. José Luis Marroquín Zaleta
M.C. Gilberto Marrufo Quirino

Guanajuato, Gto., Febrero del 2000

A mi esposa Adriana Clarisa

y a mis hijas Saraí y María Paula

Agradecimientos

Quiero agradecer al Dr. José Luis Marroquín Zaleta y al M.C. Gilberto Marrufo Quirino por la paciencia y atención que tuvieron para el desarrollo de esta tesis. Su sapiencia, consejos y comentarios aportaron el conocimiento necesario y suficiente para que pudiera llevar a buen fin mis estudios.

En especial quisiera agradecer a mi esposa e hijas, que me dieron su paciencia y apoyo incondicional en este ir y venir durante estos dos años. Su comprensión, su alegría y disposición que compartieron conmigo, solamente los fines de semana, fueron suficientes para motivarme e impulsarme para concluir esta meta en mi vida académica.

También quisiera agradecer a todos los profesores del área de cómputo del CIMAT, especialmente al Dr. Mariano Rivera, cuyos consejos, ideas y el trabajo minucioso de revisión de la tesis hizo posible no solo mejorar la calidad de ésta misma, sino de ampliar el conocimiento en los diferentes temas abordados en la tesis.

Quisiera dar también un agradecimiento a mis compañeros de estudios dentro de esta institución, Felix Calderón y Carlos Galván que con su disposición, consejos y ayuda fue posible muchas veces entender y discutir los diferentes temas abordados tanto en la tesis, como en los cursos impartidos durante los estudios de la maestría.

En cuanto a la posibilidad de poder venir a superarme en esta institución, quiero reconocer el total apoyo que tuve del director de la Facultad de Ciencias de la Universidad Autónoma de San Luis Potosí, Fis. Benito Pineda Reyes, así como al Rector de la UASLP, Ing. Jaime Valle Méndez. Por el mismo motivo quisiera agradecer al PROMEP (Programa de Mejoramiento de Profesores) por darme esta oportunidad de superación en las condiciones más favorablemente posibles.

Finalmente quisiera dar mi gratitud a todo el personal del CIMAT, doctores, profesores, personal administrativo y demás trabajadores que permitieron con su amabilidad y disponibilidad ayudar, en lo que fue posible, a que el transcurrir estos años fuera agradable y propicio para el estudio, y algunas veces para la distracción y el esparcimiento.

Índice General

1 Introducción

La motivación del presente trabajo es la reconstrucción tridimensional de objetos a partir de un modelo físico. Su importancia se vuelve evidente cuando es necesario obtener mediciones de objetos que permitan, a través de ellas, alguna forma de reproducción o análisis del objeto estudiado. La automatización de este proceso lleva directamente a pensar en una posible alternativa para su realización, *La Digitalización Tridimensional*, misma que involucra un conjunto de conocimientos tanto de *Cómputo* como de *Matemáticas Aplicadas*.

La utilidad de dicho dispositivo es variada; puede ser utilizado en la industria para reconstrucción de piezas, para mejorar la calidad de la producción al medir piezas y compararlas con algún patrón, para la creación de moldes a partir de un modelo, etc.

Otro atractivo importante, es su interés científico, ya que por sus características existen problemas que pueden ser atacados desde diferentes enfoques, representando un reto resolverlos utilizando distintos métodos, exponiendo sus ventajas y desventajas.

Así, bajo esta idea, el producto final de este trabajo de tesis es la creación de un prototipo de digitalizador que reconstruya objetos en tres dimensiones. Los principios en que se basa son ópticos, y aunque en la actualidad existen una variedad de este tipo de digitalizadores, es de gran interés probar y proporcionar alternativas que intenten mejorar su calidad.

1.1 Antecedentes

Actualmente existen digitalizadores que trabajan en base a diferentes principios, siendo los más populares los de contacto y los ópticos (*Scanners*). A continuación se describen brevemente, permitiendo entender las diferencias que los caracterizan.

I Digitalizadores de Contacto.- Su principio se basa en palpar la superficie del objeto a reconstruir tridimensionalmente. Las coordenadas obtenidas de cada contacto son referenciadas a un origen, que generalmente corresponde al de una mesa de coordenadas, donde se encuentra instalado el palpador. Sus principal desventaja es que dado que las mediciones son proporcionadas al tocar la superficie del objeto, y si éste es fácilmente deformable no será posible utilizar este método. También debido a que existe contacto, es posible dañar el mismo palpador. Otra desventaja inherente es el tiempo, ya que para recorrer todo el objeto es necesario una gran cantidad de puntos, y el mover el palpador a cada uno de ellos requiere un tiempo considerable.

II Rastreadores Láser (scanners).- Los digitalizadores de este tipo utilizan generalmente un rayo láser, que al pasar sobre el objeto forman una línea que define su contorno. Esta línea es captada por una cámara, de la que se obtiene una imagen, que al ser procesada en una computadora, permite conocer las coordenadas de cada punto. Al barrer el láser completamente el objeto, es entonces posible reconstruirlo totalmente. Una de sus ventajas principales es que no existe contacto alguno, por lo cual no es posible producir alguna deformación. Su desventaja son las oclusiones, que se presentan cuando por algún motivo la línea, generada por el rayo láser, que pasa sobre el objeto no es continua, ocasionando problemas en el procesos de digitalización. Este problema puede en ocasiones ser resuelto utilizando más de una cámara.

III Proyectores de Franjas.- Éste otro método. a diferencia del anterior, proyecta un conjunto de franjas al mismo tiempo y utilizando principios ópticos, se recupera por medio de una sola imagen las dimensiones del objeto. Se puede observar inmediatamente que una de sus características es la rapidez de digitalización, pero desgraciadamente la calibración de dichos dispositivos es complicada y continúa siendo tema de investigación.

1.2 Justificación

En el presente trabajo se optará por un digitalizador del tipo scanner, pero se buscará, en principio, obtener mediciones lo más precisas posibles, proponiendo para ello alternativas en dos aspectos importantes:

I El tratamiento de las líneas generadas por el rayo láser.- Se pretende utilizar métodos robustos que detecten las líneas proyectadas sobre el objeto con una exactitud a nivel subpixel, mejorando por lo tanto la exactitud del digitalizador.

II Una diferente alternativa de calibración.- Los métodos clásicos de calibración tratan de estimar los parámetros del modelo de las cámaras, que una vez conocidos permitan la reconstrucción tridimensional. Aquí se propone un método completamente diferente, que trate de encontrar simplemente la correspondencia entre las imágenes y el objeto en el espacio, buscando al mismo tiempo mejorar la calidad del dispositivo.

1.3 Desarrollo de la Tesis

En el segundo capítulo se especifica la configuración del digitalizador, describiendo los elementos que lo componen, su funcionamiento y las características de cada uno de ellos. Se presenta un esquema en el que pueden apreciarse la razón de su disposición y los términos que después serán utilizados en el desarrollo de la tesis.

El tercer capítulo explica como se realizó la detección de líneas, la razón por la cual se utilizaron los filtros de *Gabor*, obteniendo con ello mediciones a nivel subpixel.

La determinación de los puntos de calibración fue tratada en el cuarto capítulo. En él se describe de que manera fueron generados y un algoritmo denominado *MPM-MAP* que permitió localizarlos a nivel subpixel, mejorando así la calidad del digitalizador.

El capítulo cinco fue el más extenso. En el se desarrollaron dos métodos diferentes de *Calibración de Cámaras*. El primero se basó en *Transformaciones Proyectivas*, en el cual, teniendo un conjunto de puntos cuyas coordenadas en el espacio tridimensional son conocidas, así como sus correspondientes en las imágenes. De esta forma es posible obtener una matriz de proyección que mapeé puntos de la imagen a puntos en un plano en el espacio, pudiendo reconstruir de esta manera la forma un objeto. El segundo método propuesto fue la utilización de *Redes Neuronales*, que considerado como un aproximador de funciones desconocidas o difíciles de modelar, provee una diferente y atractiva alternativa, ya que existen un conjunto de factores tales como el tipo de lente, campo de visión o apertura angular de la cámara que intervienen en la digitalización y que no son considerados en el método anterior.

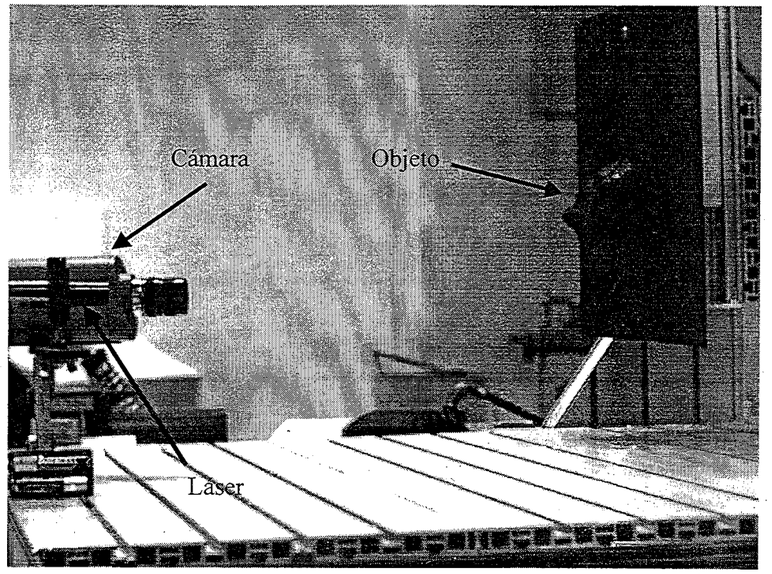
El sexto capítulo es la parte experimental, en ella se hace una comparación de los métodos de calibración y los resultados obtenidos en la digitalización de algunas figuras.

El último capítulo, el séptimo, se llega a conclusiones, que de manera objetiva permitan describir las ventajas y desventajas del trabajo propuesto.

En cada tema tratado en esta tesis se presenta el fundamento teórico, siendo después desarrollados los algoritmos específicos que, una vez codificados en algún lenguaje de programación, permitan su aplicación y prueba en la construcción del digitalizador.

2 Arquitectura del Digitalizador

Los elementos que componen el digitalizador son una *Cámara Digital* de marca *Cohu*, monocromática de 480×640 pixeles, un *Diodo de Luz Láser* de bajo costo con cabeza para proyectar un línea y una *Mesa de Coordenadas*. La mesa tiene incluida un base que puede ser movida en cualquiera de los tres ejes coordenados X, Y y Z por medio de motores de paso controlados por una computadora, donde es posible sujetar el objeto a digitalizar. A su vez, la mesa de coordenadas tiene una plataforma fija en la que fueron colocado el láser y la cámara, como se muestra en la imagen siguiente.



Digitalizador

La disposición de estos elementos permite que el láser proyecte una línea sobre el objeto y que ésta sea captada por la cámara, que a su vez es almacenada en una computadora. Hay que recalcar que la cámara y el láser están sujetos por medio de una base a la plataforma de la mesa, de forma que una vez colocados no sea posible moverlos, ya que de esta configuración dependerá el proceso de digitalización, descrito posteriormente. Un esquema representativo, así como algunas características del dispositivo se ilustran en la figura 1.

La orientación de los ejes coordenados corresponden al sistema utilizado por la mesa de coordenadas. La precisión de los movimientos que son posible realizar en cada eje son del orden de milésimas de centímetro.

Como se puede observar en la figura 2, la línea proyectada por el láser puede considerarse como un plano, que al pasar por el objeto traza una línea sobre su superficie, misma que será captada por la cámara, cuya imagen puede considerarse como otro plano.

En la figura 2 están representados los dos planos correspondientes. A cada punto en uno le corresponde uno y solo uno del otro, de tal forma que sea posible, por algún método, encontrar la correspondencia entre el *Plano del Láser* y el *Plano de la Imagen*.

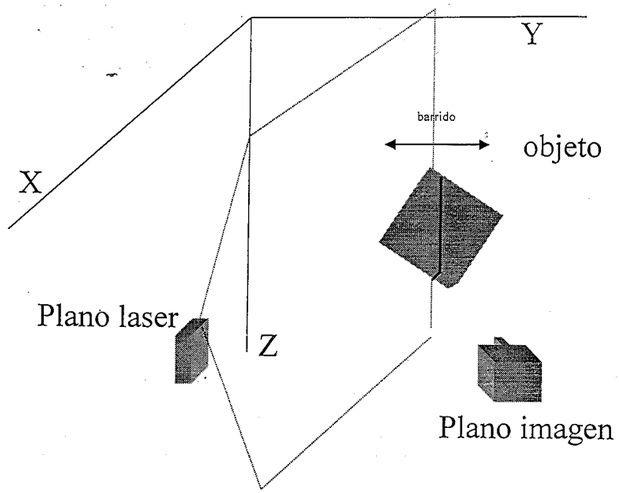


Figura 1: Esquema del Digitalizador

Ya se dijo que la cámara y el emisor de rayo láser se encuentran fijos, esto quiere decir que el plano del láser con respecto al plano de la imagen no se moverá, lo que significa que, sabiendo el mapeo entre ellos, es posible conocer las coordenadas X y Z de cada punto de la línea proyectada por el láser sobre el objeto. Ahora si éste es desplazado a lo largo del eje Y, y ya que el desplazamiento de la mesa de coordenadas (conocido) es controlado por la computadora, es entonces posible entonces realizar la reconstrucción tridimensional.

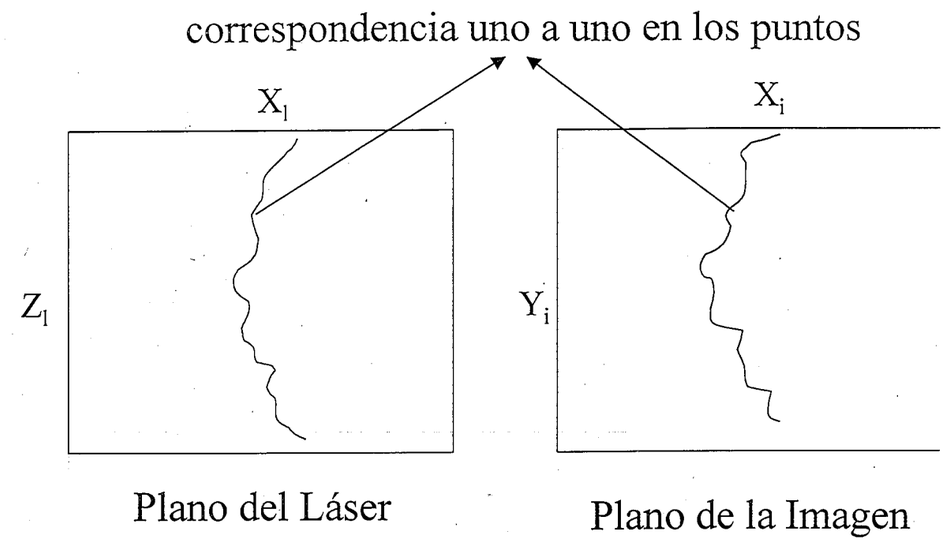


Figura 2: Planos

3 Detección de Líneas

El procesamiento de imágenes para detectar la línea, generada por el rayo láser proyectado sobre el objeto es parte clave del digitalizador. Para el método de calibración, la preciza detección de los puntos de la línea permitirán mapear puntos entre el plano de la imagen y del láser que posteriormente para el proceso de digitalización permitirá la medición de objetos.

La línea en sí, describe el contorno, y por lo tanto los detalles del objeto. Su ubicación exacta en la imagen es de importancia, primordialmente para mejorar la exactitud en el proceso de reconstrucción.

Un ejemplo típico de las líneas a tratar se muestra en la figura 3, así como el detalle de un renglón de la misma en la figura 4.

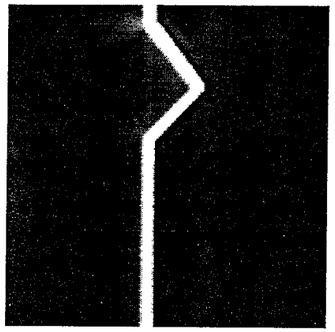


Figura 3: Linea del Laser

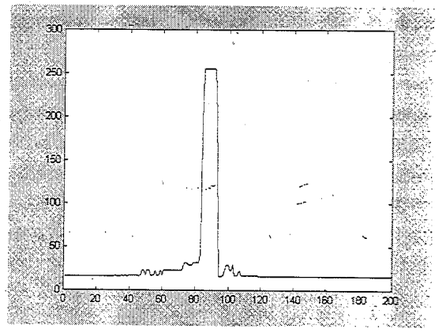


Figura 4: Detalle

En el detalle puede observarse una región de algunos pixeles de longitud donde la intensidad es máxima, que es precisamente la parte de interés, y ruido en los extremos, siendo

entonces necesario identificar dentro de esta región el punto (coordenadas del pixel) que represente la línea.

Existen varias maneras de realizar este trabajo, entre ellas se podría pensar en las siguientes:

1. Una manera es encontrar el sitio con valor de intensidad máximo del tono de gris en la imagen y tomar éste como la coordenada donde pasa la línea. Aunque muy simple, este método incurre en errores, ya que si la región abarca algunos pixeles, no se sabría que pixel elegir. Si se calculara el promedio de la coordenada de estos pixeles, también se incurriría en un error porque dentro de la misma región pueden existir pequeñas variaciones en la intensidad de gris, por lo que tampoco sería muy exacto.

2. Otra forma de detectar líneas, podría ser calculando su centro de masa, con la expresión,

$$\frac{\sum_i f(x_i)\, x_i}{\sum_i x_i}$$

donde $f(x_i)$ es el nivel de gris en el pixel x_i. El problema que se presenta aquí es que, debido a que siempre existe ruido en la imagen, como se observa en la figura 4, puede ocasionar que el centro de masa se salga de la región de interés. Una posibilidad de solucionar este problema sería primero aplicar un umbral a la imagen.

3. En el presente trabajo se utilizó un método más robusto, los *Filtros de Cuadratura*, pero antes de definirlos es necesario introducir lo que es la *fase local* y su utilidad en la detección de líneas, aspectos importantes para crear dichos filtros.

3.1 Fase Local

Una línea en una imagen puede considerarse que tiene aproximadamente la forma de la cresta de una onda cosenoidal.

Como puede observarse en la figura 5 el centro de la línea coincide en posición con la parte más alta de una de las crestas de la onda, y es este punto el que es de interés localizar, ya que nos proporcionará la ubicación exacta de un punto por donde pasa la línea. Ahora, esta posición tiene que ver con el argumento de la función coseno, la fase, teniendo para este caso particular el valor de cero.

Una forma de localizar la fase de la cosenoide es suponer un operador que retrasa su fase 90°, tal operador convertiría una señal

$$g(x) = \cos(\omega_0 x)$$

en

$$g'(x) = \cos\left(\omega_0 x + \frac{\pi}{2}\right) = -\sin(\omega_0 x)$$

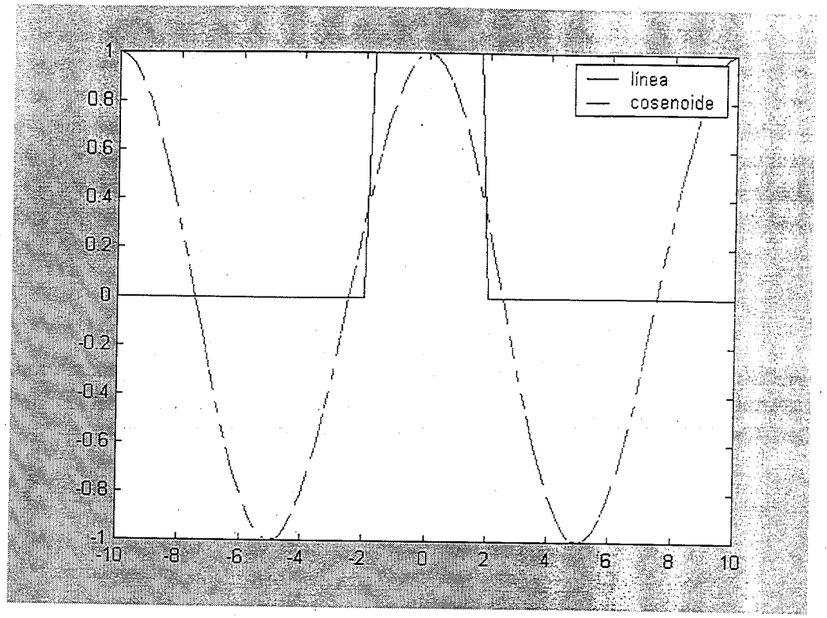


Figura 5: Línea y onda cosenoidal

de tal forma que la fase podría obtenerse al calcular

$$\arctan\left(\frac{-g'(x)}{g(x)}\right)$$

Dicho operador es conocido como *Operador de Hilbert*, cuya función de transferencia está definida como

$$H_{\mathcal{H}}(u) = \begin{cases} -iH(u) & \text{sí } u < 0 \\ iH(u) & \text{si } u \geq 0 \end{cases} \tag{1}$$

donde $H(u)$ es la función de transferencia de $g(x)$. Tal operador es complejo e impar debido a que la señal $g(x)$ es real, por lo que ambas pueden combinarse en una señal compleja

$$g_A = g(x) - ig'(x) \tag{2}$$

conocida como función analítica o señal analítica.

3.2 Filtros de Cuadratura

Los filtros de cuadratura[1] son una alternativa para obtener un par de señales al convolucionar un renglón de la imagen con ellos. Esencialmente estos filtros son parecidos a una

función analítica, excepto por que la función de transferencia no tiene magnitud uno y puede ser cualquier función real. En nuestro caso se optó por un par de filtros conocidos de *Gabor*[1], que juntos forman un filtro de cuadratura.

El objetivo principal de este filtro, pasa banda, fue que permitiera localizar en cada renglón de la imagen, la línea que fuera proyectada por el rayo láser al pasar sobre el objeto. Ésto se logra haciendo que la mitad del periodo de la frecuencia (ω_0) se asemeje lo más posible al ancho de la línea. El filtro complrjo se diseñó en el dominio del espacio y por definición su construcción fue:

$$h(x) = g(x)\left[\cos(\omega_0 x) + i\sin(\omega_0 x)\right]$$

donde $g(x)$ es una función *Gaussiana* con desviación standard σ igual a una cuarta parte del periodo de ω_0. Las gráficas de la figura 6 y 7 corresponden al filtro que se utilizaría para procesar la línea del ejemplo.

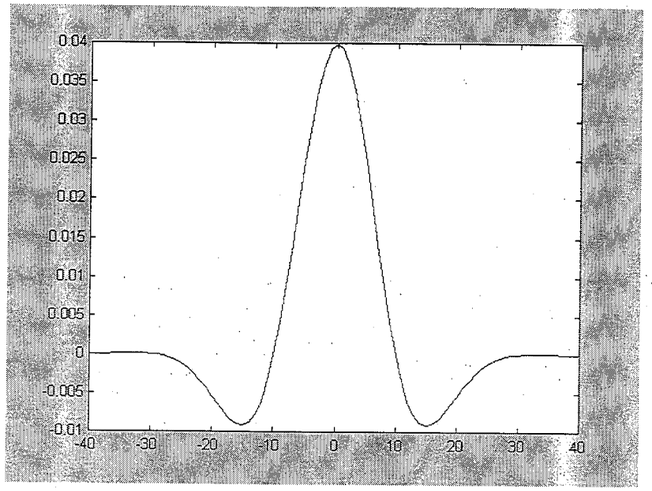


Figura 6: Parte real del filtro de Gabor

La figura 8 muestra (segunda gráfica) que, la magnitud obtenida al aplicar el filtro de *Gabor* a un renglón de una imagen depende solamente de la energía de la señal y de que tan bien ésta concuerda con el filtro pasa banda de la parte real del filtro. La fase, por otro lado es indiferente al *DC* de la imagen, y depende de la relación que existe entre la parte par e impar de la señal relativa al centro del filtro. De esta forma, cuando el centro del filtro, tanto la parte real como la imaginaria, se encuentra en el centro de la línea, el valor de la fase tomará el valor de cero. Hay que hacer notar que este valor concuerda exactamente con el valor pico de la magnitud, determinando de esta forma las coordenadas del subpixel que será tomado como representativo de la línea para su posterior procesamiento.

Un filtro de cuadratura debe cumplir con (1) y (2), por lo que es necesario demostrar que los filtros de Gabor cumplen con estos requisitos.

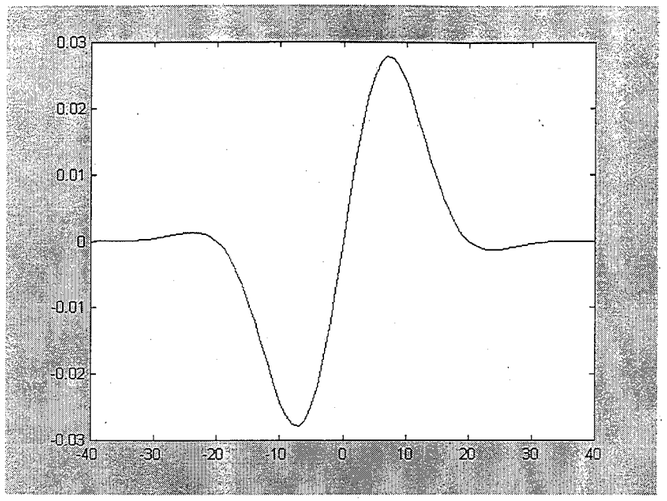


Figura 7: Parte imaginaria del filtro de Gabor

Retomando estas dos condiciones tenemos que

$$H_{xi}(u) = \begin{cases} -iH_{\text{real}}(u) & \text{sí } u < 0 \\ iH_{\text{real}}(u) & \text{si u} \geq 0 \end{cases} \tag{3}$$

$$h(x) = h_{\text{real}}(x) - ih_{xi}(x) \tag{4}$$

Donde $H_{xi}(u)$ es la *DFT* (Transformada Discreta de Fourier) de la *Transformada de Hilbert,* definida más adelante, de $h_{xi}(x)$, $H_{\text{real}}(u)$ es la DFT de $h_{\text{real}}(x)$ y $h(x)$ es la función analítica del filtro, donde

$$h_{\text{real}}(x) = g(x)\cos(\omega_0 x)$$

es la parte real del filtro de Gabor, donde $g(x)$ es una ventana *Gaussiana* y su *Transformada de Fourier* es

$$DFT\{h_{\text{real}}(x)\}(u) = H_{\text{real}}(u) = \frac{N}{2}\left[\delta(u-\omega_0) + \delta(u+\omega_0)\right] \circledast G(u)$$

donde $\circledast$ denota convolución.

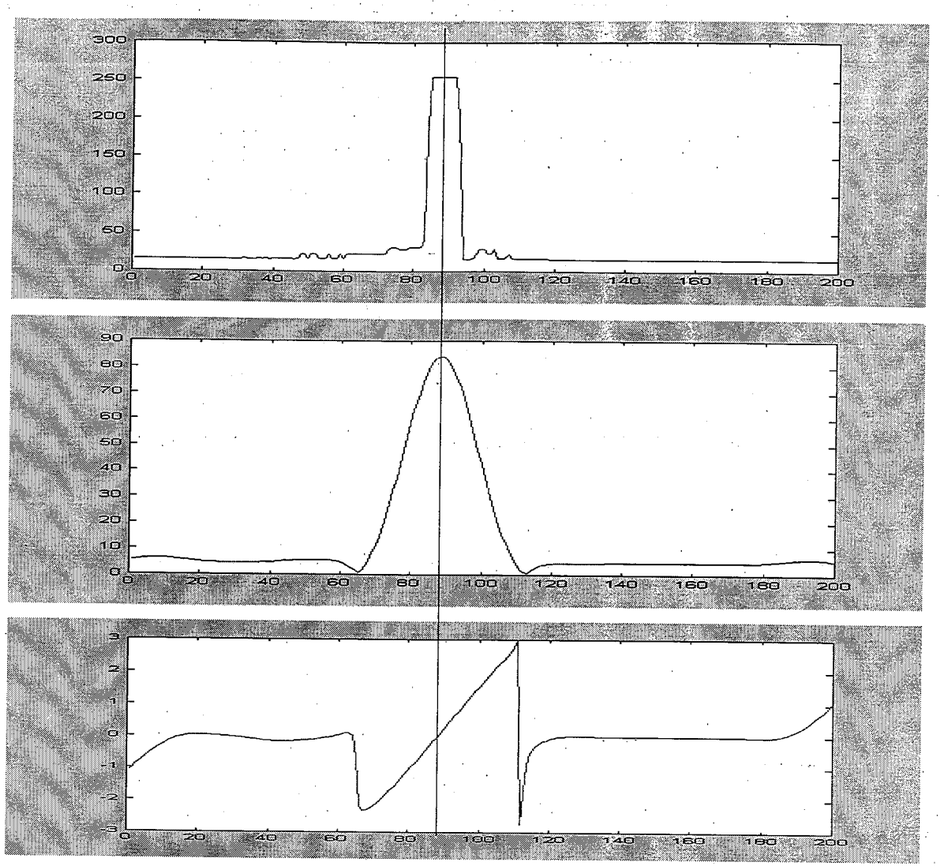


Figura 8: Línea, magnitud y fase obtenidas de aplicar los filtros de cuadaratura.

Se tiene que por (3), la *DFT* de la *Transformada de Hilbert* de $h_{xi}(x)$ se define como:

$$
\begin{aligned}
H_{xi}(u) &= H_{\text{real}}(u) \cdot i \cdot sign(u) \\
&= \left\{ \frac{N}{2} \left[\delta(u-\omega_0) + \delta(u+\omega_0) \right] \circledast G(u) \right\} \cdot i \cdot sign(u) \\
&= \left\{ \frac{N}{2} \left[\delta(u-\omega_0) - \delta(u+\omega_0) \right] \circledast G(u) \right\} \cdot i \\
&= \frac{N}{2} \sum_{k=0}^{N-1} \left[\delta(k-\omega_0) - \delta(k+\omega_0) \right] G(u-k) \cdot i \\
&= \frac{N}{2} \left[G(u-\omega_0) - G(u+\omega_0) \right] \cdot i
\end{aligned}
$$

Como puede verse, el hecho de que, tanto $g(x)$ (Gaussiana) como $\cos(\omega_0 x)$ sean funciones pares y reales, el resultado de la *DFT* de la *Transformada de Hilbert* es un función impar e imaginaria, por lo que su IDFT (Inversa de la DFT) es,

$$
\begin{aligned}
IDFT\{H_{xi}(u)\} &= \frac{1}{2} i \cdot IDFT\{G(u-\omega_0) - G(u+\omega_0)\} \\
&= \frac{1}{2} i \cdot \left[g(x) e^{i\omega_0 x} - g(x) e^{-i\omega_0 x} \right] \\
&= -g(x) \sin(\omega_0 x)
\end{aligned}
$$

Por lo tanto, la función analítica (4) correspondiente al filtro *real* sería

$$h(x) = g(x) \left[\cos(w_0 x) + i \sin(w_0 x) \right]$$

que es igual al filtro de *Gabor* propuesto, con lo que termina la demostración.

Ya teniendo el filtro adecuado para cada línea de la imagen, éste es aplicado y se procede a calcular, como ya se mencionó, la magnitud y fase. En seguida se busca el pixel correspondiente a la magnitud máxima. Alrededor de este pixel, la fase debe tener un cruce por cero, por lo que se procede a calcularlo para cada renglón con el siguiente algoritmo :

Algoritmo 1 *Detecta Coordenadas de Líneas*

1. Sea $f(x)$ un renglón de la imagen y $h(x)$ el filtro correspondiente, *Calcula Mag(x)* $= \| f(x) \circledast h(x) \|$ y *Ang(x)* $= atan2(f(x) \circledast h(x))$,
2. *Calcula la ecuación de la recta que cruza por cero en la fase, al rededor del punto de máxima magnitud.*

$$x_{\max} = arg\ max(Mag(x))$$

$$m = \frac{[Ang\,(x_{\max} - 1) - Ang\,(x_{\max} + 1)]}{[(x_{\max} - 1) - (x_{\max} + 1)]}$$

$$g\,(x) = mx - m\,(x_{\max} + 1) + Ang\,(x_{\max} + 1)$$

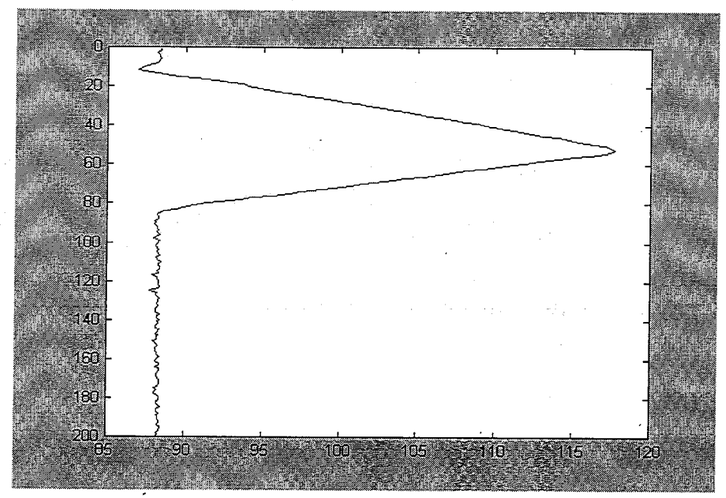
3. *Cuando $g\,(x) = 0$ existe un cruce por cero, por lo que*

$$x = (x_{\max} + 1) - \frac{Ang\,(x_{\max} + 1)}{m}$$

donde x la coordenada corresponde al subpixel que representa la línea en un renglón y.

4. *Se repiten los pasos del 1 al 3 para cada renglón.*

El resultado de aplicar este algoritmo a la línea de ejemplo que se utilizó en esta sección es



Coordenadas de los cruces por cero de la imagen de prueba

4 Puntos de Calibración

El objetivo de la calibración es obtener un método que permita encontrar puntos del *Plano del Láser* que correspondan a aquellos en el *Plano de la Imagen*, pero antes de describir cada uno de los métodos propuestos, es conveniente explicar las características de la configuración *(setup)* usada para la calibración.

Lo primero que se hizo fue buscar la manera de obtener puntos en la imagen, cuya ubicación en el plano del láser fuera conocida. Esto se logró utilizando una figura geométrica, que al pasar el rayo láser sobre ella, permitiera sin ninguna ambigüedad encontrar en la imagen las coordenadas de los puntos de esta línea con resolución de subpixel, una vez procesada con los filtros de *Gabor*.

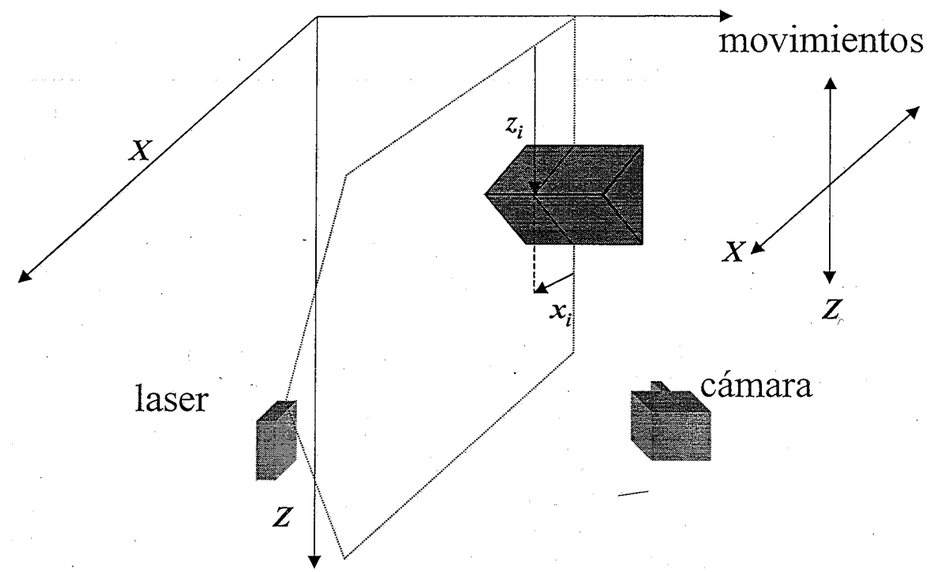


Figura 9: Configuración de la Calibración.

La figura por la que se optó fue un prisma, ya que la pasar el rayo por un costado, figura 9, forma dos rectas que se intersectan en un punto del cual, una vez conocidos los parametros de las rectas, es fácil calcular sus coordenadas a nivel subpixel, aumentado de tal forma la precisión obtenida en la parte de detección de líneas.

4.1 Detección de Puntos de Calibración

Para la estimación de las rectas que forma el láser al pasar por el prisma, es necesario un procedimiento que obtenga los parámetros de sus ecuaciones.

Las coordenadas de los puntos obtenidos en las imágenes no corresponden exactamente a dos rectas, existe una desviación, debida principalmente al proceso de digitalización de

la cámara. Dicha diferencia se supuso se debía a ruido *Gaussiano*, motivo por el cual se recurrió a un método estadístico que, sabiendo que se trataban de rectas, las modelara en forma eficiente, eligiendo para tal propósito un algoritmo denominado MPM-MAP[2] (*Maximizador de Marginales a Posteriori y Máximo a Posteriori*), por sus siglas en inglés.

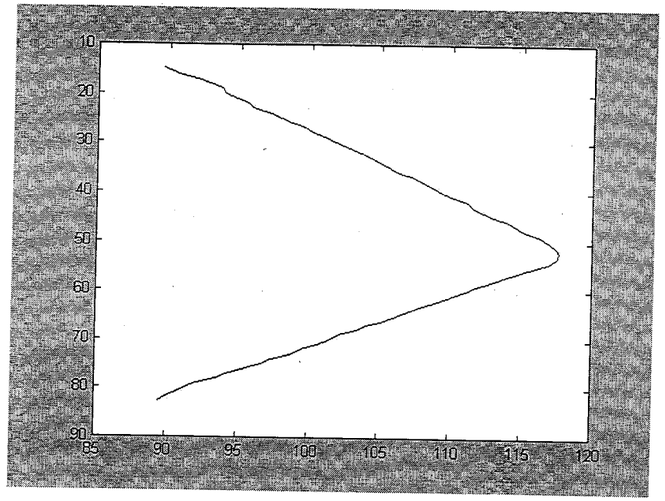


Figura 10: Coordenadas de los subpixeles que forman las rectas

La figura 10 muestra un ejemplo de las coordenadas de los subpixeles que forman las rectas, ya suprimidos aquellos que corresponden a la recta vertical que no es parte del canto del prisma, antes de aplicar el algoritmo *MPM-MAP*.

Cada imagen es adquirida por medio de un frame-grabber , instalado en una computadora, después de haber movido el prisma a una posición aleatoria (pero conocida) dentro del plano de láser por la mesa de coordenadas, también controlada por una computadora y después procesada por el algoritmo descrito a continuación.

4.2 Algoritmo MPM-MAP

Una de las principales razones para utilizar este algoritmo es que permite ajustar lo mejor posible los valores de los parámetros de un modelo, de forma que teniendo un conocimiento apriori de ellos, modele el conjunto de datos con que se dispone, en este caso se refiere a las coordenadas de los subpixeles de las dos rectas dentro en una región de las imágenes de calibración. Pero ya que, tanto los valores de estos modelos, así como la ubicación de las regiones de soporte de las rectas dentro de la imagen son desconocidas, este enfoque realiza una estimación de ellos en dos pasos iterativos: el primero es, dada una estimación de los parámetros del modelo, se estiman las regiones de soporte, las cuales se utilizan a su vez en el segundo paso para estimar a su vez los parámetros.

La manera de atacar este problema es desde un enfoque probabilístico en el cual se especifica la distribución del ruido observado y la distribución de probabilidad apriori del

conjunto de posibles modelos. El algoritmo está basado en la teoría de *Estimación Bayesiana*, donde los parámetros de los modelos que generaron los datos son estimados, en este caso las ecuaciones de las dos rectas que representan las líneas.

Iniciemos explicando en forma general este algoritmo en el contexto del trabajo de tesis, introduciendo para ello la siguiente notación.

Sea X el conjunto de valores correspondientes a las coordenadas x de los subpixeles, tal que $g = \{g(x) \,|x \in X, g(x) \in Y\}$ son las observaciones de las coordenadas y de los subpixeles. Sea también $\{\phi(r;\theta_k), k = 1...K\}$ el conjunto de modelos a estimar (en este caso, dos líneas rectas) y que están representados por el vector de parámetros $\theta = \{\theta_1, \theta_2, .., \theta_k\}$. Cada modelo es válido en una región $R_k \subseteq X$, de tal forma que el valor $f(x)$ dado por

$$f(x) = \sum_{k=1}^{K} \phi(x;\theta_k) b_k(x) \tag{5}$$

representa la verdadera coordenada y generada por el modelo a estimar y $b_k(x)$ es la función indicadora para la región R_k:

$$b_k(x) = \begin{cases} 1, & \text{si } x \in R_k \\ 0, & \text{de otra forma} \end{cases}$$

La verosimilitud de los datos, dados los modelos y las regiones de soportes es:

$$P(g|b,\theta) = \frac{1}{Z_X} \prod_{x \in X} \prod_{k=1}^{k} l_k(x)^{b_k(x)} \tag{6}$$

donde Z_X es una constante de normalización y

$$l_k(x) = P(g(x)\,|\theta_k, b_k(x) = 1)$$

que puede ser obtenida del modelo de ruido observado.

Habiendo hecho estas definiciones es posible calcular la distribución a posteriori usando la regla de Bayes:

$$P(b,\theta|g) = \frac{P(g,b \mid \theta)\, P(\theta)}{P(g)}$$

donde

$$P(g,b|\theta) = P(g|b,\theta)\, P(b|\theta)$$

es la verosimilitud de los datos completos.

Por otra parte, cuando el vector de parámetros θ es desconocido, se puede asumir que tiene una distribución a priori uniforme, por lo que $P(\theta) =$ constante y $P(g)$ es una constante de normalización.

En este enfoque se consideran tanto a b, como a θ como una matriz y un vector aleatorios, cuyos estimadores óptimos son obtenidos por medio de la minimización del valor esperado de una función apropiada de costo, tomada con respecto a la distribución a posteriori.

La función de costo propuesta en el método es la siguiente:

$$C\left(\hat{b},\hat{\theta},b,\theta\right) = \left[1-\delta\left(\theta-\hat{\theta}\right)\right] + \sum_{x\in X}\left[1-\delta\left(b(x)-\hat{b}(x)\right)\right] \tag{7}$$

Donde $b(x)$ es el vector de unos y ceros que indica el modelo al que pertenecen las coordenadas x del subpixel. Para este caso de dos rectas, este vector solamente podrá tener para cada coordenada los posibles valores,

$$b(x) \in \{[0,1],[1,0]\}$$

Para que δ sea igual a uno, su argumento debe tomar el valor de cero y δ toma el valor cero si tiene cualquier otro valor. Así, el primer término de la función de costo indica que el vector de parametros $\hat{\theta}$ de los modelos debe ser el ideal, mientras que las funciones indicadoras de soporte de las regiones $\hat{b}$ tienen que minimizar el número de errores, por lo que los estimadores óptimos de $\hat{\theta}$ y $\hat{b}$ son,

$$\left(\hat{\theta}^*,\hat{b}^*\right) = \arg\min_{\hat{b},\hat{\theta}} Q\left(\hat{\theta},\hat{b}\right) = E\left[C\left(\hat{b},\hat{\theta},b,\theta\right)\right]$$

Con el propósito de minimizar Q, el método lo realiza en dos pasos; primero minimiza con respecto a $\hat{b}$, manteniendo fija $\hat{\theta}$ y luego de haber obtenida la $\hat{b}$ óptima se fija ésta y se minimiza con respecto a $\hat{\theta}$.

Al hacer lo anterior se tiene que el primer paso corresponde a minimizar,

$$\sum_{b}\sum_{x\in X}\left[1-\delta\left(b(x)-\hat{b}(x)\right)\right]P\left(b,\bar{\theta}\mid g\right) \tag{8}$$

$$= \sum_{x\in X}\sum_{b} P\left(b,\bar{\theta}\mid g\right) - \sum_{b}\sum_{x\in X}\delta\left(b(x)-\hat{b}(x)\right)P\left(b,\bar{\theta}\mid g\right)$$

$$= |X| - \sum_{x\in X}\sum_{b:b(x)=\hat{b}(x)} P\left(b,\bar{\theta}\mid g\right)$$

$$= |X| - \sum_{x\in X}\sum_{k=1}^{K}\sum_{b:b_k(x)=\hat{b}(x)} P\left(b,\bar{\theta}\mid g\right)\hat{b}_k(x)$$

$$= |X| - \sum_{x\in X}\sum_{k=1}^{K}\pi_k(x)\,\hat{b}_k(x)$$

donde

$$\pi_k(x) = \sum_{b:b_k(x)=1} P\left(b,\bar{\theta}\mid g\right) = E\left[b_k(x)\right]$$

que es la probabilidad marginal aposteriori de la región de soporte k para la coordenada x. De tal forma que para minimizar la expresión (8) se toma como el valor de $\hat{b} = \bar{b}$ como:

$$\bar{b}_k(x) = \begin{cases} 1, \text{ si } \pi_k(x) > \pi_{k'}(x) \text{ para } k' \neq k \\ 0, \text{ de otra forma} \end{cases}$$

Este estimador es llamado *Maximizador de Marginales a Posteriori* o estimador *MPM* para una b dado $\bar{\theta}$.

Para minimizar Q con respecto a $\hat{\theta}$ para un valor fijo de $\hat{b} = \bar{b}$ se considera el primer termino de(7):

$$\int\left[1-\delta\left(\theta-\hat{\theta}\right)\right]dP\left(\bar{b},\theta|g\right) = 1 - P\left(\bar{b},\hat{\theta}|g\right)$$

de tal forma que el estimador de θ es encontrado al maximizar (6) con respecto a θ.

4.3 Determinación de Rectas Usando MPM-MAP

Ahora se desarrollará el algoritmo en forma completa, para el caso de las dos rectas que se mencionaron en la sección anterior. Ya se dijo que las observaciones $g(x)$ son las coordenadas y de las rectas que se generan al pasar el rayo de láser por el costado de la figura de calibración, pero que éstos tienen cierta dispersión de una recta ideal $f(x)$, la cual puede modelarse como ruido Gaussiano. Cada punto entonces estaría generado por:

$$g(x) = f(x) + \varepsilon$$

$$\varepsilon \sim N\left(0, \sigma^2\right)$$

$$\varepsilon = g(x) - f(x)$$

donde $f(x)$ está dada por (5).

Se tienen dos modelos ϕ_1 y ϕ_2, cuyos parámetros θ_1 y θ_2 corresponden a $\theta_k = \{\alpha_k, M_k, B_k, \sigma_k\}$ para $k = 1, 2$, dende α_k representa las proporciones de mezclas de los dos modelos y cuyos parámetros M_k, B_k son la pendiente y la ordenada al origen de cada recta y σ_k es la desviación estandar del error ε de la distrubución normal corespondiente a la coordenada en cada una de las rectas.

Haciendo estas suposiciones, tenemos entonces que la verosimilitud podría ser expresada como

$$P(g|b, \theta) = \frac{1}{Z_X} \Pi_{x \in X} \Pi_{k=1}^{K} l_k(x)^{b_k(x)}$$

Si definimos

$$l_k(x) = P(x|b_k(x) = 1, \theta_k) P(b_k(x) = 1|\theta_k)$$

donde

$$P(x|b_k(x) = 1, \theta_k) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x) - f_k(x))^2}{2\sigma_k^2}\right\}$$

(modelo del ruido observado) y

$$P(b_k(x) = 1|\theta_k) = \alpha_k$$

(la proporción de mezcla), entonces

$$l_k(x) = \frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x) - f_k(x))^2}{2\sigma_k^2}\right\}$$

Para el primer paso del algoritmo, MPM, teniendo una estimación inicial de los parámetros α_k, M_k,B_k y σ_k, se calculan las marginales a posteriori con,

$$\begin{aligned}
\pi_k(x) &= \frac{\frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x)-f_k(x))^2}{2\sigma_k^2}\right\}}{\sum_k \frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x)-f_k(x))^2}{2\sigma_k^2}\right\}} \\
&= \frac{\frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x)-M_k x-B_k)^2}{2\sigma_k^2}\right\}}{\sum_k \frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x)-M_k x-B_k)^2}{2\sigma_k^2}\right\}}
\end{aligned}$$

y se maximizan, obteniendo

$$\bar{b}_k(x) = \begin{cases} 1, \text{ si } \pi_k(x) > \pi_{k'}(x) & \text{para } k' \neq k \\ 0, & \text{de otra forma} \end{cases}$$

En el paso MAP se maximiza la verosimilitud, que es equivalente a minimizar la *log-verosimilitud negativa*, una vez obtenida la estimación de las regiones de soporte, con respecto a θ_k de la siguiente forma:

$$\begin{aligned}
E &= -\ln P\left(g|\bar{b}, \hat{\theta}\right) = -\ln \frac{1}{Z_X} \Pi_{x \in X} \Pi_{k=1}^{K} l_k(x)^{\bar{b}_k(x)} \\
&= -\ln\left[\frac{1}{Z_X} \Pi_{x_i \in X} \Pi_k \frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{(g(x) - f_k(x))^2}{2\sigma_k^2}\right\}\right]^{\bar{b}_k(x)} \\
&= -\sum_x \sum_k \bar{b}_k(x) \left\{\ln \alpha_k - \ln \sigma_k - \frac{(g(x) - M_k x - B_k)^2}{2\sigma_k^2}\right\} + cte
\end{aligned}$$

Ahora podemos minimizar esta función con respecto a los parámetros M_k, B_k, σ_k y α_k. Para M_k y B_k, tenemos que

$$\frac{\partial E}{\partial M_k} = \sum_{x \in X} \bar{b}_k(x) \frac{g(x) - M_k x - B_k}{2\sigma_k^2} x = 0$$

$$\sum_{x \in X} \bar{b}_k(x) M_k(x)^2 + \sum_{x \in X} \bar{b}_k(x) B_k x = \sum_{x \in X} \bar{b}_k(x) g(x) x \tag{9}$$

$$\frac{\partial E}{\partial B_k} = \sum_{x \in X} \bar{b}_k(x) \frac{g(x) - M_k x - B_k}{2\sigma_k^2} = 0$$

$$\sum_{x \in X} \bar{b}_k(x) M_k x + \sum_{x \in X} \bar{b}_k(x) B_k = \sum_{x \in X} \bar{b}_k(x) g(x) \tag{10}$$

que al resolver (9) y (10) se obtienen dichos parámetros.
En el caso de σ_k se tiene

$$\frac{\partial E}{\partial \sigma_k} = \frac{\sigma_k^2 \sum_{x \in X} \bar{b}_k(x) - \sum_{x \in X} \bar{b}_k(x) (g(x) - M_k x - B_k)^2}{\sigma_k^3} = 0$$

$$\sigma_k^2 = \frac{\sum_{x\in X} \bar{b}_k(x)\left(g(x) - M_k x - B_k\right)^2}{\sum_{x\in X} \bar{b}_k(x)}$$

En cuanto a α_k, debemos tomar en cuenta la restricción $\sum_k \alpha_k = 1$. Introduciendo los multiplicadores de Lagrange λ, queda,

$$\pounds(\lambda) = E + \lambda\left(\sum_k \alpha_k - 1\right)$$

Y derivando con respecto a α_k e igualando a cero,

$$-\sum_{x\in X} \frac{\bar{b}_k(x)}{\alpha_k} + \lambda = 0$$

$$\begin{aligned} \lambda &= \sum_{x\in X} \frac{\bar{b}_k(x)}{\alpha_k} \\ \lambda &= |X| \end{aligned}$$

Por lo tanto,

$$\frac{\partial E}{\partial \alpha_k} = -\sum_{x\in X} \frac{\bar{b}_k(x)}{\alpha_k} + |X| = 0$$

$$\alpha_k = \frac{1}{|X|}\sum_{x\in X} \bar{b}_k(x)$$

Algoritmo 2 *Determinación de rectas*

1. Se proponen valores iniciales de $\theta_k^{(t)}$para M_k, B_k, σ_k y α_k para $k = 1, 2$, en la iteración $t = 0$.

2. Paso MPM, se calculan las marginales con $\theta_k^{(t)}$ y se estiman las $\bar{b}_k^{(t+1)}(x)$ para cada x, de la forma

$$\bar{b}_k^{(t+1)}(x) = \begin{cases} 1, \text{ si } \pi_k(x) > \pi_{k'}(x) \text{ para } k' \neq k \\ 0, \text{ de otra forma} \end{cases}$$

donde

$$\pi_k(x) = \frac{\frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{\left(g(x) - M_k^{(t)}x - B_k^{(t)}\right)^2}{2\sigma_k^{2(t)}}\right\}}{\sum_k \frac{\alpha_k}{\sqrt{2\pi}\sigma_k} \exp\left\{-\frac{\left(g(x) - M_k^{(t)}x - B_k^{(t)}\right)^2}{2\sigma_k^{2(t)}}\right\}}$$

3. Paso MAP, se estiman los parámetros de $\theta_k^{(t+1)}$ una vez obtenidas las $\bar{b}_k^{(t+1)}$:

$$\begin{pmatrix} \sum_{x\in X} \bar{b}_k^{(t+1)}(x)\, x^2 & \sum_{x\in X} \bar{b}_k^{(t+1)}(x)\, x \\ \sum_{x\in X} \bar{b}_k^{(t+1)}(x)\, x & \sum_{x\in X} \bar{b}_k^{(t+1)}(x) \end{pmatrix} \begin{bmatrix} M_k^{(t+1)} \\ B_k^{(t+1)} \end{bmatrix} = \begin{bmatrix} \sum_{x\in X} \bar{b}_k^{(t+1)}(x)\, g(x)\, x \\ \sum_{x\in X} \bar{b}_k^{(t+1)}(x)\, g(x) \end{bmatrix}$$

de donde podemos despejar los valores de $M_k{}^{(t+1)}$y $B_k^{(t+1)}$

4. Calculamos $\sigma_k^{(t+1)}$ y $\alpha_k^{(t+1)}$.

$$\left(\sigma_k^2\right)^{(t+1)} = \frac{\sum_{x\in X} \bar{b}_k^{(t+1)}(x)\left(g(x) - M_k^{(t+1)}x - B_k^{(t+1)}\right)^2}{\sum_{x\in X} \bar{b}_k^{(t+1)}(x)}$$

$$\alpha_k^{(t+1)} = \frac{1}{|X|}\sum_{x\in X} \bar{b}_k^{(t+1)}(x)$$

5. hacemos $t = t + 1$

6. Repetimos los pasos de 2 a 5.

La figura 11 muestra el resultado de aplicar el algoritmo a la imagen ejemplo, después de haber sido filtrada y localizados los cruces por cero.

Las ecuaciones encontradas fueron:

$$y1 = (-1.0391)\, x + 175.3455$$

$$y2 = (1.3048)\, x - 102.7854$$

y su punto de intersección $(118.6621, 52.0445)$.

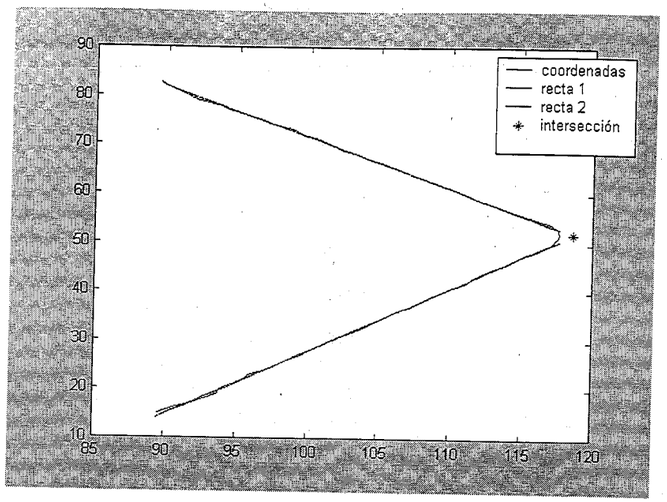


Figura 11: Resultados del algoritmo MPM-MAP

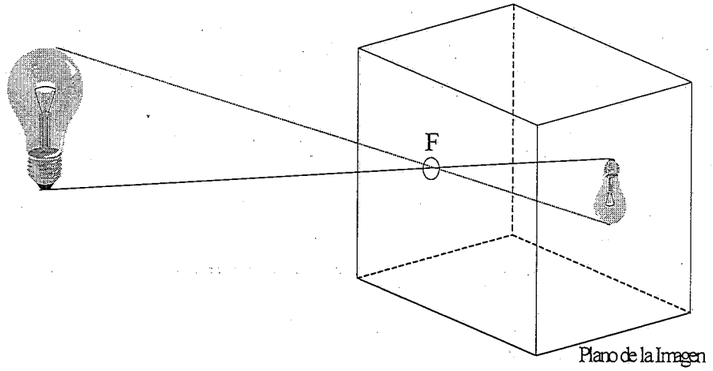
5 Métodos de Calibración

Una vez localizadas las coordenadas en la imagen de los puntos de calibración, se aplicaron dos métodos diferentes para realizar la correspondencia entre puntos en cada uno de los planos. El primero de ellos se basó en encontrar la matriz de proyección de un plano al otro, debido a que intuitivamente sería lo ideal, si se tuviera la certeza de tener una configuración perfecta, sin ningún tipo de error, ya sea en la cámara, el emisor del rayo láser o el movimiento perfecto de la mesa de coordenadas.

El segundo de ellos utiliza una red neuronal. La razón de esta decisión fue debido a que ésta es un buen aproximador universal de cualquier función desconocida, además que, por las características de la configuración del digitalizador, no se conocen todos los elementos que intervienen en él, siendo así una opción excelente esta función de mapeo.

5.1 Transformación de Planos Proyectivos

La relación que existe entre un punto en el plano del láser y uno en la imagen, es lo que se llama una transformación proyectiva entre dos planos. En ella un punto y su imagen, por definición, están alineados con un punto llamado *ojo de la transformación* que no pertenece a ninguno de los dos planos. En el caso del digitalizador, un punto físico correspondiente al objeto y su proyección en el plano de la imagen se encuentran también alineados con el centro focal F de la cámara, por lo que puede entonces modelarse con una *transformación proyectiva* en la cual F es el ojo de la transformación.



Modelo de la cámara.

Esta correspondencia uno a uno entre puntos puede ser representada por una matriz de 3×3, cuando se usan coordenadas homogéneas para definir los puntos en la imagen. Así, conociendo la matriz de transformación que lleva puntos de la imagen al láser, podemos conocer la posición de puntos físicos del objeto en el plano *X, Z (coordenadas de la mesa)*, y conociendo además el desplazamiento del barrido de éste es posible la reconstrucción tridimensional.

Antes de describir el método que lleva a obtener aproximadamente esta matriz, es necesario introducir algunas definiciones que permitieron después establecer el teorema que demuestra su existencia.

5.1.1 Coordenadas Homogéneas y Plano Proyectivo

Las coordenadas homogéneas son una herramienta analítica de la geometría proyectiva que permite facilitar el desarrollo de fórmulas de transformación que serían más complicadas si se utilizara coordenadas *Cartesianas*.

Es fácil convertir cualquier cualquier punto en coordenada *Cartesianas* $[x, y]$ a coordenadas homogéneas, simplemente se agrega otra coordenada con valor igual a uno, es decir $[x, y, 1]$. Como se puede ver, estas coordenadas se encuentran contenidas en lo que es llamado el *plano proyectivo estándar*, que corresponden al plano *Euclidiano* en R^3 con $z = 1$. No es necesario que el tercer componente de las coordenadas homogéneas sea uno, cualquier punto con coordenadas $[xk, yk, k]$ puede ser llevado a este plano proyectivo estandar al dividir cada coordenada por k, es decir

$$[xk, yk, k] = [x, y, 1]$$

por lo que puede existir un conjunto de puntos en coordenadas homogéneas que tengan el mismo punto en este plano proyectivo, por ejemplo:

Sean dos puntos

$$Q = [xk_1, yk_1, k_1]$$

y

$$R = [xk_2, yk_2, k_2]$$

con $k_1 \neq k_2$, entonces $Q = R$ en el plano proyectivo estandar ya que

$$[xk_1, yk_1, k_1] = [x, y, 1]$$

y

$$[xk_2, yk_2, k_2] = [x, y, 1]$$

Por otro lado, existe una clase de puntos llamados *puntos ideales* contenidos en este plano. Un punto ideal $P = [a, b, 0]$ define una línea l en el plano $z = 1$ que pasa por el origen $O\,(0, 0, 1)$ y cuya dirección es P, ver figura ??. Así, l puede ser definida en forma paramétrica como

$$\{P_t = (at, bt, 1) \in R^3 | t \in R\}$$

donde a, b y 0 son los números directores de l, por lo que para cada t,

$$[at, bt, 1] = \left[a, b, \frac{1}{t}\right]$$

y P es el límite del punto Euclidiano P_t sobre l, es decir

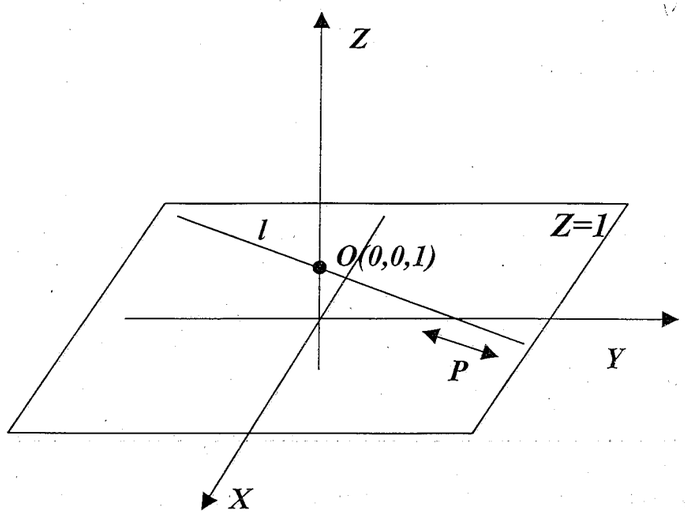


Figura 12: Punto ideal en el plano proyectivo estandar

$$P = \lim_{t\to\infty} P_t = \lim_{t\to\infty} \left[a, b, \frac{1}{t}\right] = [a, b, 0]$$

que son las coordenadas homogéneas estándar de P.

Este punto tiene la característica de ser el punto al que convergen todas las líneas que son paralelas en plano proyectivo estándar. Lo anterior se demuestra al definir otra línea m, la cual se puede describir paramétricamente como

$$\{P_t = (at + x_0, bt + y_0, 1)\,|t \in R\}$$

por lo que tenemos que

$$P = \lim_{t\to\infty} P_t = \lim_{t\to\infty} \left[a + \frac{x_0}{t}, b + \frac{y_0}{t}, \frac{1}{t}\right]$$

y P es el límite del punto Euclidiano P_t en m cuando $t \to \infty$.

$$P = [a, b, 0]$$

Finalmente podemos definir que las coordenadas homogéneas estandar de un punto $P = [p_1, p_2, p_3]$ en el plano proyectivo pueden ser de la forma

1. $[x, y, 1]$ si P es un punto en el plano *Euclidiano* $z = 1$ cuyas coordenadas *Cartesianas* son $(x, y, 1)$.

2. $[a, b, 0]$ si P es un punto ideal asociado a todas las líneas en el plano Eucliadiano $z = 1$ con números directores $a, b, 0$.

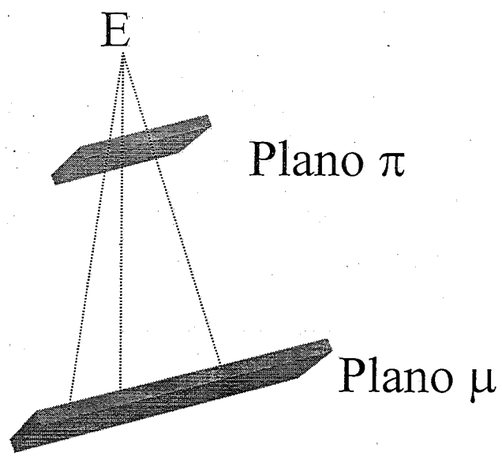
Con estas definiciones podemos entonces decir que,

$$\text{plano proyectivo } = \{\text{puntos } \textit{Euclidianos}\} \cup \{\text{puntos ideales}\}$$

5.1.2 Teorema Fundamental de la Geometría Proyectiva entre Planos

El teorema fundamental de la Geometría Proyectiva[3] para transformaciones entre dos planos establece lo siguiente:

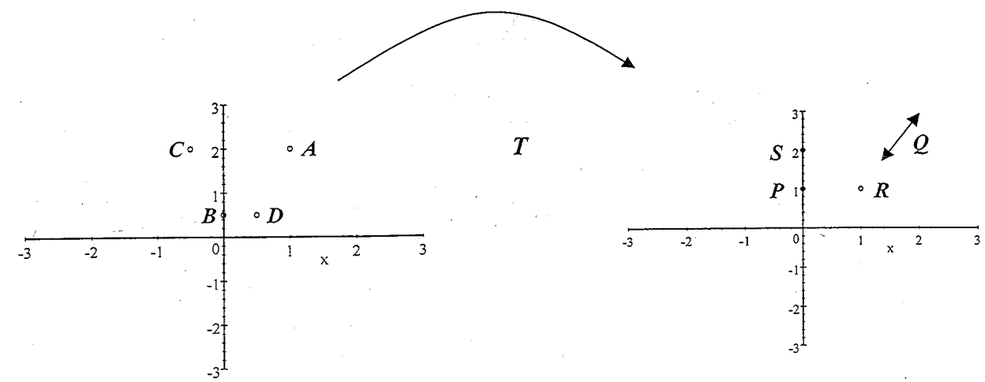
Sea π y μ dos planos en el espacio proyectivo. Si A, B, C y D son cuatro puntos distintos en π, de los cuales tres de ellos no son colineales, y P, Q, R y S son puntos en μ, tres de ellos no colineales, existe una y solo una transformación proyectiva de π a μ en relación al centro de proyección E, y por lo tanto de A a P, B a Q, C a R y D a S.



Planos proyectivos

Para entender mejor el teorema, a continuación se describe un ejemplo que ilustra este teorema y la existencia de una matriz que realiza dicha transformación.

Dado dos conjuntos de cuatro puntos (en coordenadas homogéneas) en el plano projectivo , $A[1, 2, 1]$, $B[0, 1, 2]$, $C[-1, 4, 2]$, $D[1, 1, 2]$ y $P[0, 1, 1]$, Q$[1, 1, 0]$, $R[1, 1, 1]$, $S[0, 2, 1]$ de los cuales tres de ellos no son colineales, la transformación proyectiva T que lleva A a P, B a Q, C a R y D a S,



Transformación proyectiva T

se puede encontrar de la siguiente forma:

Sea d el vector correspondiente al punto D, una combinación lineal de los otros vectores a, b y c.

$$d = \langle 1, 1, 2\rangle = \frac{2}{3}\langle 1, 2, 1\rangle + \langle 0, 1, 2\rangle + \frac{-1}{3}\langle -1, 4, 2\rangle$$

Aquí la notación vectorial $< -, -, - >$, significa que, un vector $< p_1, p_2, p_3 >$ es representativo de las coordenadas homogéneas $[q_1, q_2, q_3]$, si se cumple que

$$[p_1, p_2, p_3] = [q_1, q_2, q_3]$$

La multiplicación por la matriz

$$M_1 = \begin{pmatrix} \frac{2}{3} & 0 & \frac{1}{3} \\ \frac{4}{3} & 1 & \frac{-4}{3} \\ \frac{2}{3} & 2 & \frac{-2}{3} \end{pmatrix} = \begin{pmatrix} \frac{2}{3}a & b & \frac{-1}{3}c \end{pmatrix}$$

cuyas columnas se obtienen al multiplicar los vectores a, b y c por los escalares $2/3$, 1 y $-1/3$ respectivamente, defme una transformación proyectiva T_1 del plano proyectivo, para el cual

$$T_1(I_x) = T_1([1, 0, 0]) = \left[\frac{2}{3}, \frac{4}{3}, \frac{2}{3}\right] = [1, 2, 1] = A$$

$$T_1(I_y) = T_1([0, 1, 0]) = [0, 1, 2] = B$$

$$T_1(I_o) = T_1[0,0,1] = \left[\frac{1}{3}, \frac{-4}{3}, \frac{-2}{3}\right] = [1,-4,2] = C$$

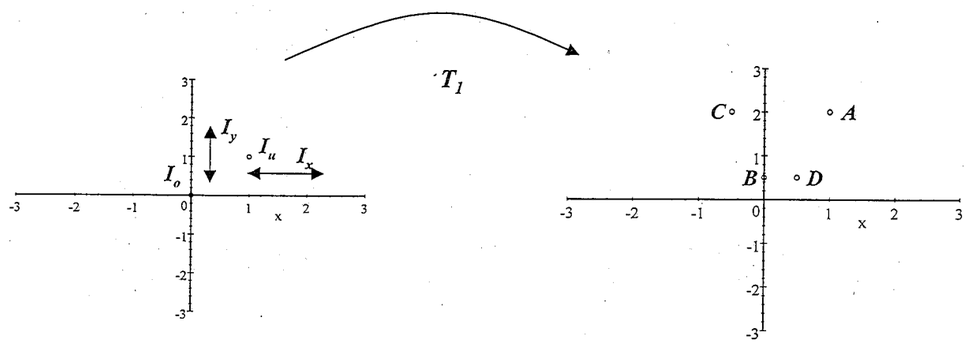
donde I_x es el punto ideal en la dirección de x, I_y el punto ideal en la dirección de y y I_o es el punto *origen* en el plano proyectivo estandar (en coordenadas homogéneas). Y ya que

$$\langle 1,1,1\rangle = \langle 1,0,0\rangle + \langle 0,1,0\rangle + \langle 0,0,1\rangle$$

entonces

$$T_1(I_u) = T_1([1,1,1]) = [1,1,2] = D$$

donde I_u es el punto *unidad* en coordenadas homogéneas. La representación gráfica de esta transformación es.

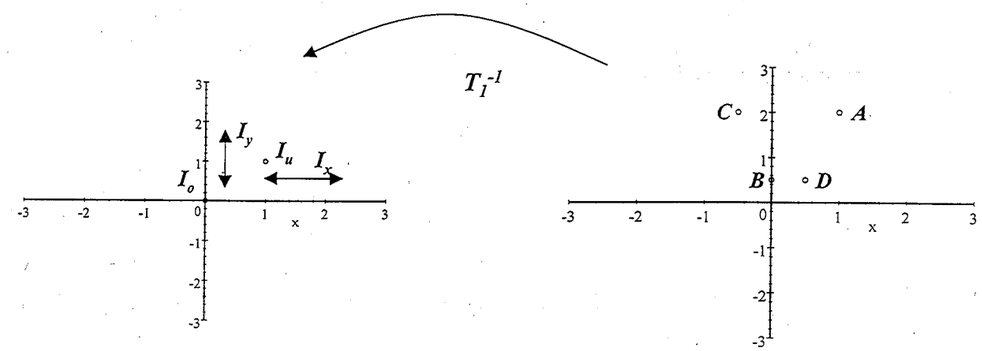


Transformación proyectiva T_1

La inversa T_1^{-1} de T_1 representada por la matriz

$$M_1^{-1} = \begin{pmatrix} 1 & \frac{1}{3} & \frac{-1}{6} \\ 0 & \frac{-1}{3} & \frac{2}{3} \\ 1 & \frac{-2}{3} & \frac{1}{3} \end{pmatrix}$$

define la transformación $T_1^{-1}(A) = I_x$, $T_1^{-1}(B) = I_y$, $T_1^{-1}(C) = I_o$ y $T_1^{-1}(D) = I_u$.



Transformación proyectiva T_1^{-1}

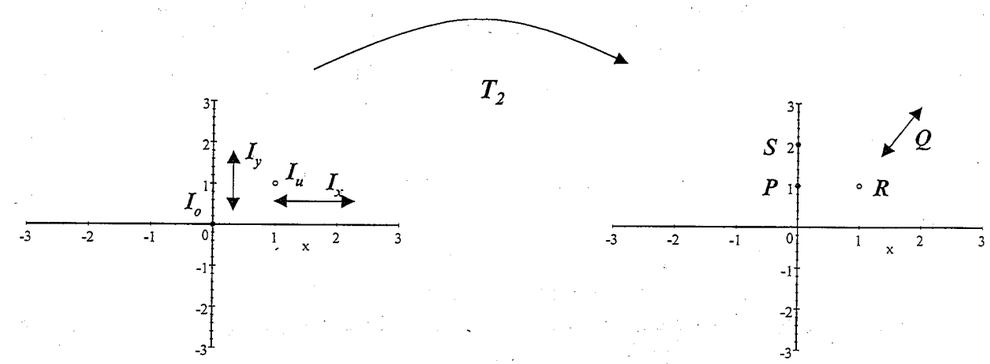
De la misma forma, se tiene el vector

$$s = \langle 0,2,1\rangle = 2\langle 0,1,1\rangle + \langle 1,1,0\rangle + (-1)\langle 1,1,1\rangle$$

que es un combinación lineal de los vectores p, q y r ,y que la multiplicación por la matriz

$$M_2 = \begin{pmatrix} 0 & 1 & -1 \\ 2 & 1 & -1 \\ 2 & 0 & -1 \end{pmatrix}$$

define una transformación proyectiva T_2 par la cual $T_2(I_x) = P$, $T_2(I_y) = Q$, $T_2(I_o) = R$, $T_2(I_u) = S$.



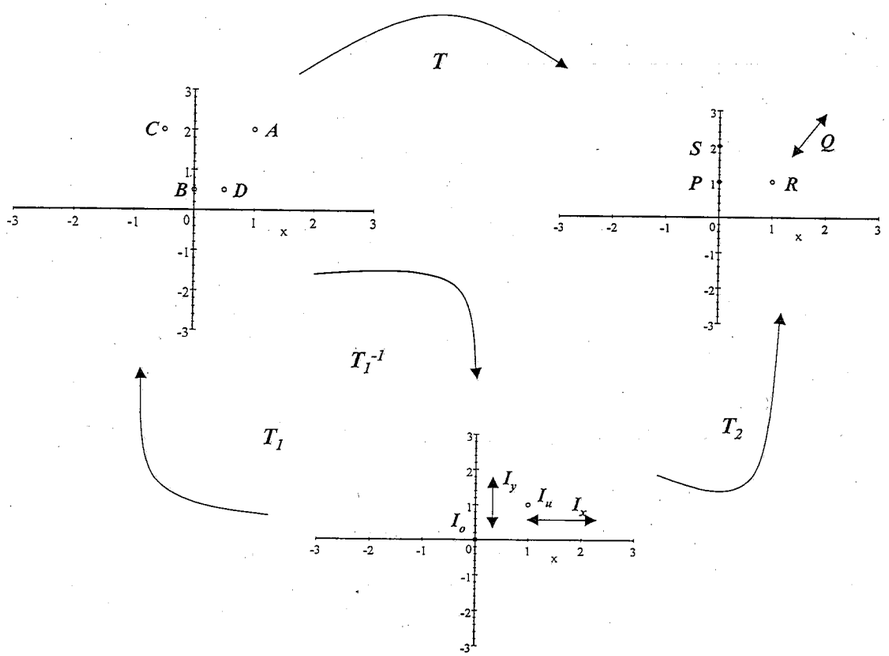
Transformación proyectiva T_2

Por lo tanto si T es una transformación compuesta por $T = T_2 \circ T_1^{-1}$, que puede ser representada por la matriz

$$M_2 M_1 = \begin{pmatrix} 0 & 1 & -1 \\ 2 & 1 & -1 \\ 2 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & \frac{1}{3} & \frac{-1}{3} \\ 0 & \frac{-1}{3} & \frac{2}{3} \\ 1 & \frac{-2}{3} & \frac{1}{3} \end{pmatrix} = \begin{pmatrix} -1 & \frac{1}{3} & \frac{1}{3} \\ 1 & 1 & 0 \\ 1 & \frac{4}{3} & -\frac{2}{3} \end{pmatrix}$$

entonces, $T(A) = P$, $T(B) = Q$, $T(C) = R$, $T(D) = S$.

La transformación proyectiva completa T entre los dos planos proyectivos puede verse gráficamente como:



Transformación proyectiva $T = T_2 \circ T_1^{-1}$

5.1.3 Método de Transformación Proyectiva

En el *Teorema Fundamental de la Geometría Proyectiva* se demuestra que un transformación entre dos planos puede ser realizada por una matriz de 3×3 cuando los puntos en cada plano se representan en coordenadas homogéneas.

De esta manera, si un punto p en el plano de la imagen tiene coordenadas *Euclidianas* (x_i, y_i) y la transformación proyectiva entre el plano de la imagen y el del láser está dada por la matriz:

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix}$$

entonces, las coordenadas homogéneas de la imagen del punto p en el plano del láser están dadas por

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \begin{pmatrix} x_i \\ y_i \\ 1 \end{pmatrix}$$

y sus correspondientes coordenadas *Euclidianas* pueden ser calculadas como.

$$x_l = \frac{m_{11}x + m_{12}y + m_{13}}{m_{31}x + m_{32}y + m_{33}}$$

y

$$y_l = \frac{m_{21}x + m_{22}y + m_{23}}{m_{31}x + m_{32}y + m_{33}}$$

Así, el método de calibración puede ser visto como, dado un conjunto de n puntos $a_1, a_2, ..., a_n$ en el plano de la imagen y sus correspondientes puntos $b_1, b_2, ..., b_n$ en el plano del láser, cuyas coordenadas son conocidas en los dos casos, encontrar la matriz M que realiza dicha transformación.

Las coordenadas en ambos planos son conocidas, por lo que tenemos un conjunto de n ecuaciones de la forma:

$$Ma_i = b_i' \text{ para } i = 1, .., n$$

donde b_i' tiene la forma $(h_i b_{1i}, h_i b_{2i}, h_i)$. Por lo que se puede establecer el siguiente conjunto de ecuaciones.

$$\begin{aligned} a_{1i}m_{11} + a_{2i}m_{12} + m_{13} &= h_i b_{1i} \\ a_{1i}m_{21} + a_{2i}m_{22} + m_{23} &= h_i b_{2i} \\ a_{1i}m_{31} + a_{2i}m_{32} + m_{33} &= h_i \end{aligned}$$

o equivalentemente si se substituye h_i

$$\begin{aligned} a_{1i}m_{11} + a_{2i}m_{12} + m_{13} - (a_{1i}m_{31} + a_{2i}m_{32} + m_{33})\, b_{1i} &= 0 \\ a_{1i}m_{21} + a_{2i}m_{22} + m_{23} - (a_{1i}m_{31} + a_{2i}m_{32} + m_{33})\, b_{2i} &= 0 \end{aligned}$$

estas expresiones pueden ser escritas en forma matricial para lo n puntos, de la forma.

$$\begin{pmatrix} a_{11} & a_{21} & 1 & 0 & 0 & 0 & -a_{11}b_{11} & -a_{21}b_{11} & -b_{11} \\ 0 & 0 & 0 & a_{11} & a_{21} & 1 & -a_{11}b_{21} & -a_{21}b_{21} & -b_{21} \\ a_{12} & a_{22} & 1 & 0 & 0 & 0 & -a_{12}b_{12} & -a_{22}b_{12} & -b_{12} \\ 0 & 0 & 0 & a_{12} & a_{22} & 1 & -a_{12}b_{22} & -a_{22}b_{22} & -b_{22} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{1n} & a_{2n} & 1 & 0 & 0 & 0 & -a_{1n}b_{1n} & -a_{2n}b_{1n} & -b_{1n} \\ 0 & 0 & 0 & a_{1n} & a_{2n} & 1 & -a_{1n}b_{2n} & -a_{2n}b_{2n} & -b_{2n} \end{pmatrix} \begin{pmatrix} m_{11} \\ m_{12} \\ m_{13} \\ m_{21} \\ m_{22} \\ m_{23} \\ m_{31} \\ m_{32} \\ m_{33} \end{pmatrix} = \mathbf{0}$$

que es un sistema de ecuaciones de la forma $Am = \mathbf{0}$, que para $n > 4$ está sobredeterminado y puede ser resuelto de la siguiente manera.

Ya que la norma de la solución de un sistema homogéneo de ecuaciones es arbitraria, se busca una solución de norma unitaria en el sentido de mínimos cuadrados[4]. Por lo tanto, se requiere minimizar

$$\|Am\|^2 = (Am)^T Am = m^T A^T Am$$

sujeta a la restricción

$$m^T m = 1$$

Introduciendo los multiplicadores de Lagrange, ésto es equivalente a minimizar el Lagrangeano:

$$\pounds(\lambda) = m^T A^T Am - \lambda\left(m^T m - 1\right)$$

Derivando con respecto a m e igualando a cero, queda

$$\begin{aligned} A^T Am - \lambda m &= 0 \\ A^T Am &= \lambda m \end{aligned}$$

El significado de esta ecuación es que el multiplicador λ corresponde a un valor propio de $A^T A$ y la solución $m = e_\lambda$ su vector propio. Al substituir $A^T Am$ por λe_λ y m por $e_\lambda$en el Lagrangeano, se obtiene.

$$\begin{aligned} \pounds(\lambda) &= m^T \lambda e_\lambda - \lambda m^T e_\lambda + \lambda \\ \pounds(\lambda) &= \lambda \end{aligned}$$

Por lo tanto, el mínimo se encuentra cuando λ es igual a cero, lo que significa que su vector e_λ será el que, una vez escrito en forma de matriz cuadrada de 3×3, se tomará como el que realice la transformación proyectiva entre los dos planos.

5.2 Redes Neuronales

Este método se basa en el paradigma de *Redes Neuronales*, su uso práctico se debe al hecho de ser un aproximador universal de funciones, desconocidas o difíciles de modelar, siendo ésta la principal característica aprovechada en el presente trabajo.

Ya se mencionó que existen una serie de elementos involucrados en la configuración del digitalizador tridimensional, los cuales son desconocidos y/o difíciles de representar, tales como la aberración del lente, los ejes de la mesa de coordenadas no estén totalmente perpendiculares, etc., y ya que el método de *Transformación Proyectiva* no los contempla en su modelo matemático, el uso de una red neuronal es una buena alternativa de calibración.

La descripción de la red utilizada, así como su topología, el algoritmo de aprendizaje y su funcionamiento se detallan en seguida.

5.2.1 Redes Neuronales de Dos Capas

Este clase de red ha demostrado su importancia en aplicaciones prácticas y un considerable número de artículos describen y demuestran sus propiedades, por lo que es importante entender su estructura y funcionamiento.

Lo primero a definir es su topología, figura (13),ya que a través de ella se caracteriza el tipo de aprendizaje y su funcionamiento. Su descripción gráfica y su explicación es la siguiente,

Consta de una capa de entrada compuesta de n elementos, llamados *neuronas*, cuyo funcionamiento es distribuir a cada neurona de la capa oculta su valor x_i de entrada multiplicado por los pesos $w_{ij}^{(1)}$que une el elemento de entrada i con la neurona oculta j, el superíndice (1) representa los pesos de esta primera capa. La función de transferencia f, que por lo general es igual en las neuronas de la misma capa, es evaluada con la integración de los datos de entrada a la neurona, es decir $f\left(\sum_{i=1}^{n+1} w_{ij}^{(1)} x_i\right)$. El bias es un umbral que permite, dependiendo del tipo de función f, exitar o activar la neurona.

Originalmente cada neurona fue llamada perceptrón, y su función de activación era la función escalón,

$$f(x) = \begin{cases} 1, \text{ si } x \geqslant \theta \text{ (bías)} \\ 0, \text{ de otra forma} \end{cases}$$

este umbral está representado como el peso w_{n+1j}, y el valor x_{n+1} es siempre igual a uno, de tal forma que pueda ser substituida por la función escalón cuyo umbral sea cero,

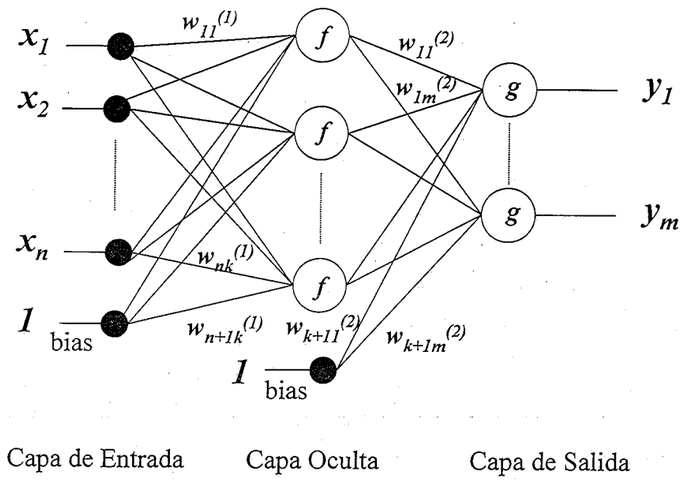


Figura 13: Red Neuronal de dos capas.

$$f(x) = \begin{cases} 1, \text{ si } x \geqslant 0 \\ 0, \text{ de otra forma} \end{cases} \quad \text{para } x = \sum_{i=1}^{n} w_{ij}^{(1)} x_i - w_{n+1j}^{(1)}$$

El algoritmo de *Propagación Hacia Atrás*, explicado más adelante, sirve para encontrar los pesos w_{ij} que permiten a la red interpolar prácticamente cualquier función. Está basado en *descenso de gradiente*, por lo que la función escalón es substituida por una sigmoide o tangente hiperbólica, según sea el caso, y es ésta la que generalmente es usada en la capa oculta.

De igual forma, la salida de cada neurona de la capa oculta es transferida a la capa de salida, después de haber sido multiplicada por el peso correspondiente.

Finalmente la función de activación de la capa de salida, que puede ser de cualquier tipo dependiendo de la aplicación, proporciona el paso final del mapeo de los valores de entrada n-dimensional a los m-dimensional de salida.

Ha habido mucha investigación referente a esta clase de red y se ha demostrado que puede aproximar en forma adecuada cualquier función de un espacio n-dimensional finito a otro, si se proporciona el número adecuado de neuronas ocultas.

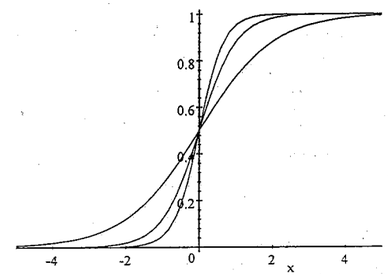
5.2.2 Algoritmo de Propagación Hacia Atrás

El algoritmo presentado aquí se basa en un enfoque gráfico[5], el cual se reduce a un problema de etiquetar un grafo. Este método no solo es más fácil de visualizar, sino que puede ser eficientemente implantado.

El algoritmo de *Propagación Hacia Atrás* busca el mínimo de una función de error en el espacio de pesos w_{ij}, usando el método de descenso de gradiente. El procedimiento de ajustar los pesos es considerado el proceso de aprendizaje de la red neuronal. Ya que es necesario el cálculo del gradiente de la función de error, se debe garantizar que ésta sea derivable, por lo que las funciones de activación utilizadas deben tener dicha propiedad. Una de las funciones más usadas para este propósito es la sigmoide, definida por la expresión,

$$S_c(x) = \frac{1}{1 + e^{-cx}}$$

La constante c controla la forma de la función. La figura siguiente muestra su apariencia para diferentes valor de c, $c = 1$, $c = 2$, $c = 3$. Para $c = \infty$, la sigmoide converge a la función escalón en el origen.



Sigmoide para c=1, c=2 y c=3

La derivada de esta función con respecto a x está dada por

$$\frac{dS(x)}{dx} = \frac{e^{-x}}{(1 + e^{-x})^2} = S(x)(1 - S(x))$$

El Proceso de Aprendizaje de la Red. Las redes neuronales de esta clase necesitan ajustar los pesos entre sus elementos para poder interpolar en forma adecuada la función deseada, que como ya se dijo se realiza minimizando una función de error en el espacio de pesos. De hecho la red es la implantación de una cadena de composición de funciones que transforma un vector de entrada a uno de salida, dicha composición es llamada la *función de red.*

La red debe entonces tratar que la función de la red φ se aproxime lo más posible a la función real f, la cual no ha sido dada explícitamente y que solamente conocemos a través de algunos ejemplos.

La aproximación entre funciones se logra encontrando la combinación óptima de pesos w_{ij}, lo cual se conoce como el problema de *Aprendizaje de la Red.*

El proceso de aprendizaje se lleva a cabo presentando a la red un conjunto de datos de entrenamiento $\{(\mathbf{x}_1, \mathbf{t}_1), ... (\mathbf{x}_p, \mathbf{t}_p)\}$ que consiste de p pares ordenados de vectores de n y m dimensiones que son llamados los patrones de entrada y salida. Supongamos que las funciones en cada nodo son continuas y diferenciables y que los pesos son seleccionados en forma aleatoria. En cada nodo de la capa de salida, después de atravesar la red un patrón de entrada por la capa oculta, se obtiene un valor de salida o_i, que por lo general es diferente al patrón deseado t_i, paso llamado *feed-forward.* El objetivo del algoritmo es entonces minimizar la función de error de la red, definida como,

$$E = \frac{1}{2} \sum_{q=1}^{p} \|\mathbf{o}_q - \mathbf{t}_q\|^2$$

Después de lograr dicha minimización, se espera que cualquier entrada desconocida sea interpolada adecuadamente por la red.

El algoritmo de propagación hacia atrás,busca precisamente un mínimo local para la función de error. La manera en que lo realiza es ajustando los pesos por medio de un proceso iterativo que calcula el gradiente.

El primer paso para la minimización es extender la red neuronal con nodos que calculen el error en cada unidad de salida, como se ilustra a continuación.

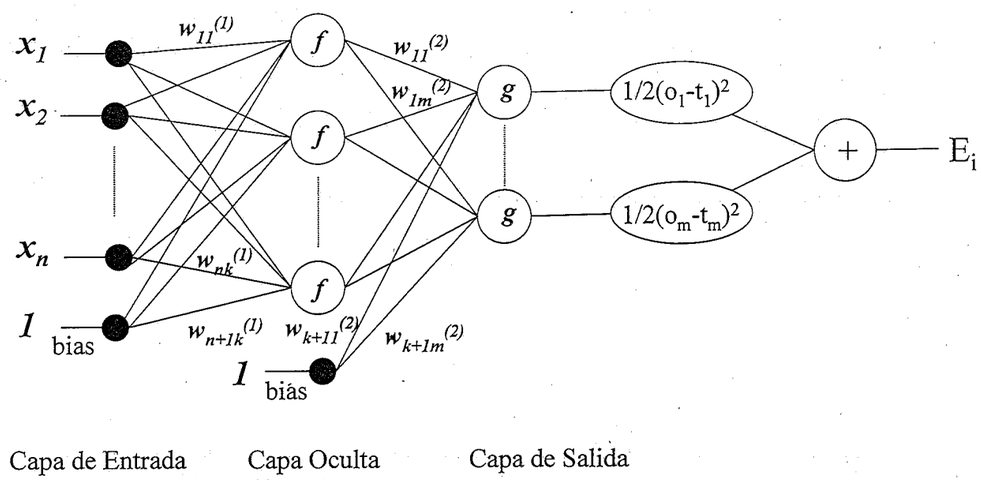


Figura 14: Extensión de la Red con Nodos de Cálculo de Error

Cada neurona de salida es conectada a un nodo que evalúa la función $\frac{1}{2}\left(o_j^{(2)} - t_j\right)^2$, donde t_j es el patrón j para la j-ésima neurona de salida, cuyo valor es $o_j^{(2)}$. Posteriormente se acumula el error de cada patrón en $E = E_1, E_2, ..., E_p$, que es la función de error.

El proceso de aprendizaje busca ahora minimizar E, usando un proceso iterativo de descenso de gradiente, para el cual es necesario calcular,

$$\nabla E = \left(\frac{\partial E}{\partial w_1}, \frac{\partial E}{\partial w_2}, ..., \frac{\partial E}{\partial w_l}\right)$$

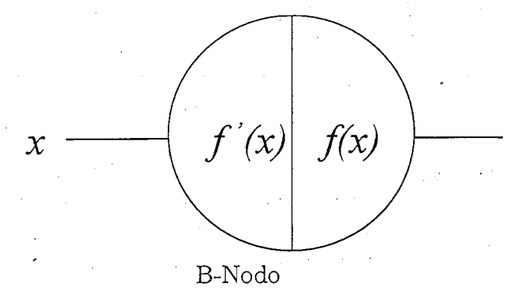
Cada peso es luego actualizado con

$$\triangle w_i = -\gamma \frac{\partial E}{\partial w_i} \text{ para } i = 1..l$$

donde γ es la constante de aprendizaje y es el parámetro que define la longitud del paso en la dirección del gradiente negativo. Bajo esta idea, se espera que se encuentre el mínimo de E cuando $\nabla E = 0$.

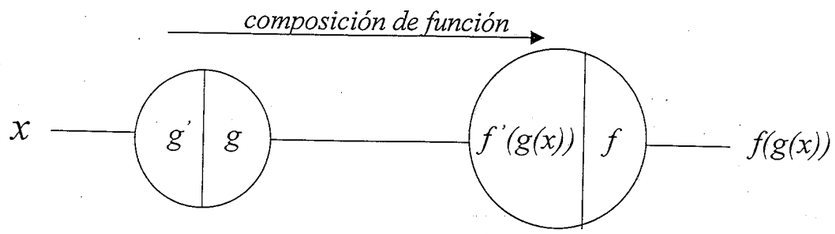
Diagramas B para el Algoritmo de Propagación Hacia Atrás. La derivada parcial del error con respecto a los pesos de la capa de salida es directa, como se verá más adelante, pero en cambio con respecto a la capa oculta es más compleja, debido a que existe una composición de funciones, debido a las conexiones que existen entre los nodos en las diferentes capas, por lo que hay que aplicar sucesivamente la regla de la cadena.

Una manera de realizar estos cálculos y reflejarlos en la estructura de la red es por medio de los *diagramas* B[5]. Cada neurona en la red está dividida en dos partes, la parte izquierda evalúa la derivada parcial f' de la función, tomando como argumento la entrada a la unidad y la parte derecha la evaluación de la función f.



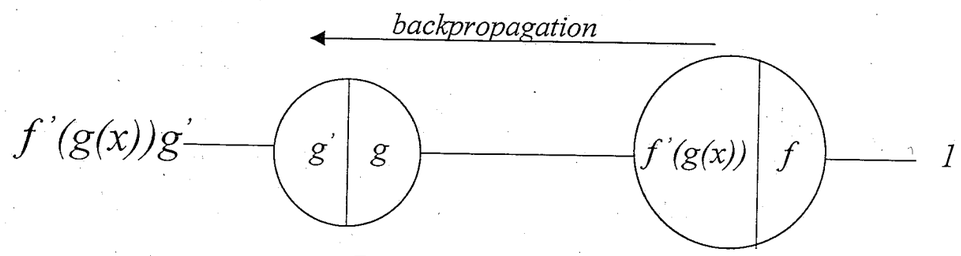
La red es evaluada en dos pasos: en el primero de ellos, tanto la derivada como la función son evaluadas cuando la información fluye de izquierda a derecha, ambos resultados son almacenados en cada nodo, pero solo la parte derecha es trasmitida a la siguiente capa de la red. La parte izquierda de los nodos es utilizada después en el paso de propagación hacia atrás, el segundo paso, que ayudará a calcular las derivadas parciales del error con respecto a los pesos.

Dentro de la red existen dos tipos diferentes de conexiones entre las capas, las cuales tienen diferente composición de funciones, es decir la manera en que se van calculando las derivadas parciales. La primera de ellas ocurre entre la capa de salida y los nodos de cálculo de error en la red neuronal extendida, figura 14. Cuando la información fluye hacia adelante *(feed forward)* cada nodo calcula la composición de funciones representada en el siguiente diagrama *B*.



Resultado de Feed-Forward.

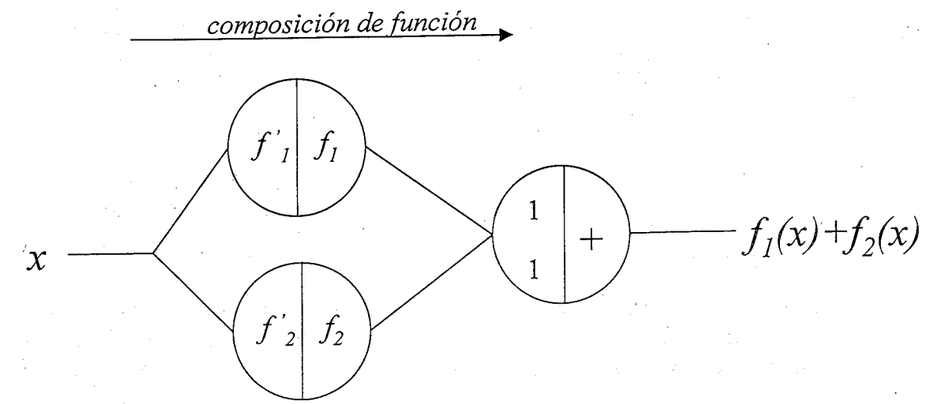
En el paso de propagación hacia atrás (*backpropagation*) la entrada a la red es la constante uno. Ahora el flujo de información va de derecha a izquierda y la información que llega a cada nodo es multiplicada por la parte izquierda de éste, como se indica a continuación.



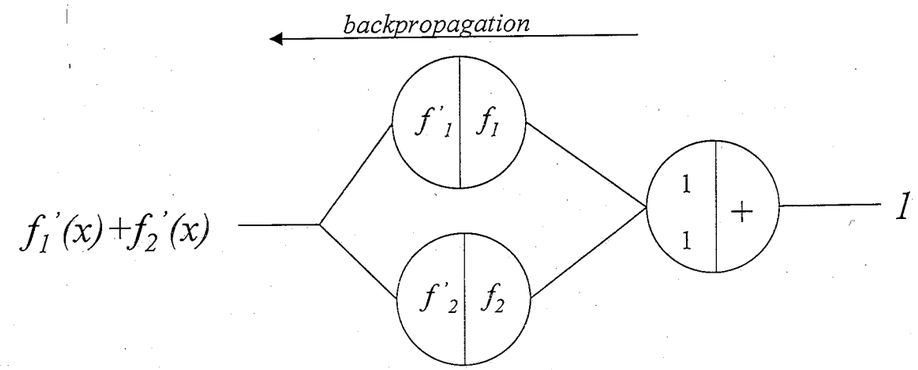
Propagación Hacia Atrás.

Se puede visualizar la propagación hacia atrás como un forma de implantación de la regla de la cadena.

El segundo tipo de composición es la *función adición.* En este caso la función en el nodo es la adición de dos o más funciones que provienen de la capa anterior, en la red de dos capas se presenta tanto en la capa oculta como en la de salida. Par mostrar gráficamente este caso solo se consideraron dos entradas. El flujo correspondiente hacia adelante como hacia atrás es el siguiente:



Adición de funciones



Propagacióna Hacia Atrás

En esta última figura se muestra que el paso de propagación hacia atrás calcula la derivada de $(f_1 + f_2)$ evaluada en x.

Aprendizaje de la Red Usando Propagación Hacia Atrás. Hasta ahora hemos considerado las derivadas parciales con respecto a la información que entra a cada nodo, pero en el caso de la red neuronal cada unidad evalúa su función con respecto al valor $w_{ij}o_i$, para poder así propagar hacia adelante la información necesaria, y a su vez calcula la derivada con respecto a este valor, es decir $\partial E/\partial\left(w_{ij}o_i\right)$. Debido a que lo que se necesita para calcular el gradiente es $\partial E/\partial w_{ij}$ y ya que se puede considerar o_i como una constante, la derivada se puede calcular como,

$$\frac{\partial E}{\partial w_{ij}} = o_i \frac{\partial E}{\partial\left(w_{ij}o_i\right)}$$

De tal forma que también es necesario almacenar en cada nodo la salida correspondiente o_i para poder implantar completamente las derivadas parciales.

Así el algoritmo completo de propagación hacia atrás sería el siguiente:

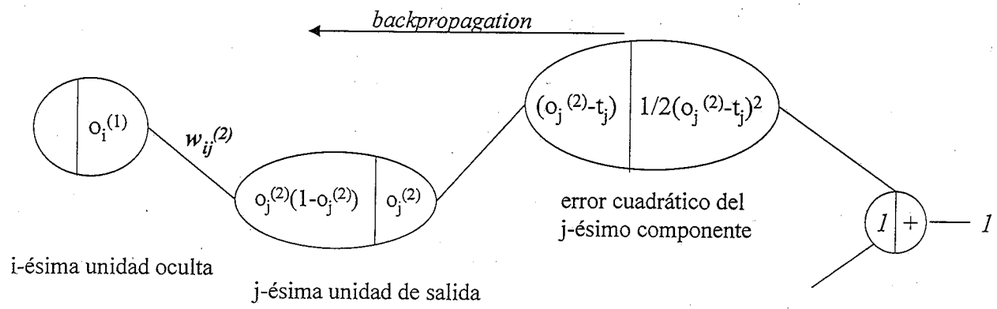
Algoritmo 3 *Aprendizaje Usando Propagación Hacia Atrás*

1. Propagación Hacia Adelante

El vector $\mathbf{x}$ es presentado a la red. Los vectores $\mathbf{o}^{(1)}$ y $\mathbf{o}^{(2)}$ son calculados y almacenados. Las derivadas y las funciones de activación son evaluadas y almacenadas en cada neurona.

2. Propagación Hacia Atrás

Las derivadas parciales del error con respecto a los pesos de las neuronas de salida se pueden calcular recorriendo la red en sentido de derecha a izquierda, dando como entrada el valor de uno a la red. La ruta de propagación hacia atrás se muestra en el siguiente *diagrama B*.



Propagación Hacia Atrás de la capa de salida.

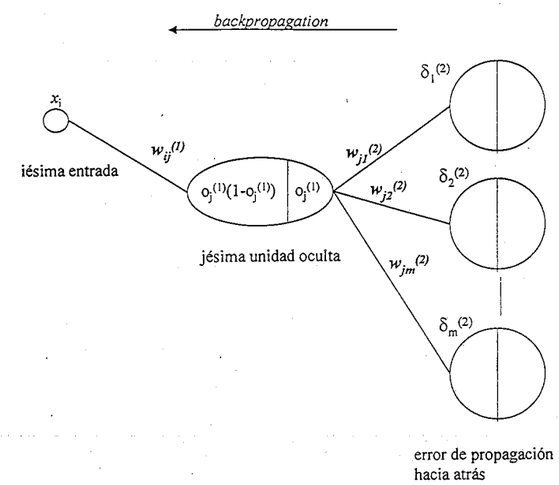
Multiplicando los términos que se encuentran en la ruta de propagación hacia atrás y si definimos

$$\delta_j^{(2)} = o_j^{(2)}\left(1 - o_j^{(2)}\right)\left(o_j^{(2)} - t_j\right)$$

como el error de propagación hacia atrás, entonces la derivada parcial del error con respecto a w_{ij} es

$$\frac{\partial E}{\partial w_{ij}} = o_i^{(1)}\delta_j^{(2)}$$

En el caso de las derivadas parciales con respecto a los pesos que van de la capa de entrada a la oculta son un poco más complejas, pero recordando la composición de una función de adición se tiene que la ruta a seguir es la siguiente.



Propagación Hacia Atrás de la capa oculta.

y definiendo el error de propagación hacia atrás para las unidades ocultas como

$$\delta_j^{(1)} = o_j^{(1)}\left(1 - o_j^{(1)}\right)\sum_{q=1}^{m} w_{jq}^{(2)}\delta_q^{(2)}.$$

Entonces la derivada parcial es

$$\frac{\partial E}{\partial w_{ij}^{(1)}} = \delta_j^{(1)} x_i$$

3. Ajuste de Pesos

Ya teniendo las derivadas parciales, lo que resta por hacer es ajustar los pesos en la dirección negativa del gradiente. Una constante de aprendizaje γ es definida como la longitud de paso a seguir en dicha dirección. La modificación en los pesos se realiza mediante.

$$\Delta w_{ij}^{(2)} = -\gamma o_i^{(1)}\delta_j^{(2)} \text{ para } i = 1, ..., k+1; j = 1, ..., m$$

y

$$\Delta w_{ij}^{(1)} = -\gamma x_i \delta_j^{(1)} \text{ para } i = 1, ..., n+1; j = 1, ..., k$$

Es común tener más de un patrón de entrenamiento para la red, por lo que cada peso $w_{ij}^{(1)}$ tendrán diferentes ajustes por cada uno de ellos y la correción final seria

$$\Delta w_{ij}^{(1)} = \Delta_1 w_{ij}^{(1)} + \Delta_2 w_{ij}^{(1)} + ... + \Delta_p w_{ij}^{(1)}$$

Que es la correción exacta siguiendo la dirección negativa del gradiente, conocida como ajuste en *lote (batch)*.

Por otro lado el ajuste en *línea* tiene la ventaja de agregar algún tipo de *ruido* en la dirección del gradiente, lo que evita caer en un mínimo local de la función de error.

5.2.3 Mínimos Cuadrados No Lineales

El algoritmo de *Propagación Hacia Atrás* busca minimizar la función de error utilizando descenso de gradiente. Esta búsqueda suele ser muy lenta principalmente cuando un mínimo en la función objetivo se encuentra en un valle angosto, lo que conduce a una gran cantidad de oscilaciones hasta encontrar el valor mínimo, como lo muestra la figura 15.

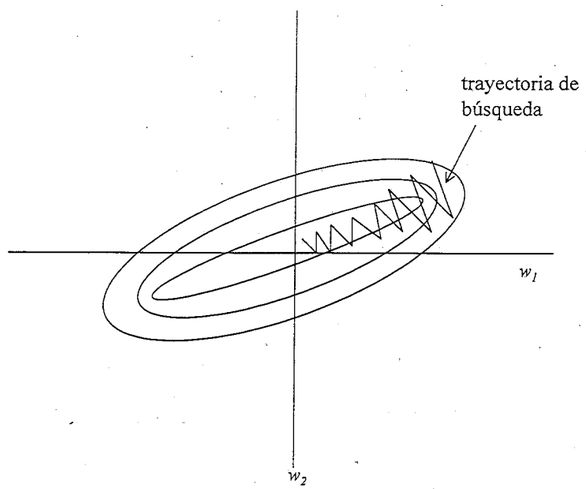


Figura 15: Descenso de Gradiente.

Por otra parte el método es aplicable para cualquier clase de función objetivo. Sin embargo hay ocasiones en que es posible utilizar métodos más eficientes debido a que la

función objetivo tiene una forma especial, y este es el caso de la función de error de la red neuronal propuesta en este trabajo, que es del tipo,

$$E(\mathbf{x}) = \sum_{j=1}^{m} f_j^2(\mathbf{x}) \tag{11}$$

La minimización de dicha función es llamada *Mínimos Cuadrados No Lineales*[6] y existen métodos adecuados para resolverlos. Entre ellos existe uno fácil de implantar, además de ser rápido y eficiente, denominado *Levenberg-Marquardt*, debido a sus creadores, pero antes de describir este algoritmo y como fue utilizado en la red neuronal, es necesario explicar algunos conceptos importantes para su entendimiento.

Derivadas de la Suma de Cuadrados de Funciones. Es importante para el método poder calcular el vector gradiente y de la matriz Hessiana. La primeras derivadas parciales de la función (11) con respecto a x_i están dadas por

$$g_j = 2\sum_{i=1}^{m} f_i \frac{\partial f_i}{\partial x_j} \tag{12}$$

Si definimos la matriz *Jacobiana* como

$$J = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_m} \\ \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1} & & \frac{\partial f_m}{\partial x_n} \end{bmatrix}$$

entonces el vector gradiente puede ser escrito como

$$g(\mathbf{x}) = 2J^T(\mathbf{x}) f(\mathbf{x})$$

Ahora, diferenciando (12) con respecto a x_k obtenemos el *kj-ésimo* elemento de la matriz *Hessiana*:

$$G_{kj} = 2\sum_{i=1}^{m} \left\{ \frac{\partial f_i}{\partial x_k}\frac{\partial f_i}{\partial x_j} + f_i \frac{\partial^2 f_i}{\partial x_k \partial x_j} \right\}$$

Si definimos como T_i la matriz Hessiana para f_i

$$T_i(\mathrm{x}) = \nabla^2 f_i$$

entonces la matriz Hessiana completa de $E(\mathrm{x})$ es

$$G(\mathrm{x}) = 2J^T(\mathrm{x})\,J(\mathrm{x}) + 2\sum_{i=1}^{m} f_i(\mathrm{x})\,T_i(\mathrm{x})$$

Si además se define

$$S(\mathrm{x}) = \sum_{i=1}^{m} f_i(\mathrm{x})\,T_i(\mathrm{x})$$

se tiene que

$$G(\mathrm{x}) = 2J^T(\mathrm{x})\,J(\mathrm{x}) + 2S(\mathrm{x})$$

Método de Newton. En el método de Newton se considera que la función a minimizar puede ser considerada a una cuadrática en las proximidades de un punto dado x_k por lo que la serie de Taylor puede proporcionar el gradiente de la función en $x_{k+1} = x_k + p_k$ como

$$g_{k+1} = g(x_k + p_k) = g_k + Gp_k$$

ya que la derivadas de mas alto orden son cero. Si x_{k+1} fuera un mínimo de la función, entonces g_{k+1} sería igual a cero y p_k sería

$$p_k = -G^{-1}g_k \tag{13}$$

En las funciones que no son cuadráticas en $x_k + p_k$ no se encontrará por lo general un mínimo, por lo que es necesario realizar estos cálculos iterativamente.

Ahora, bajo este mismo razonamiento y tomando en cuenta las derivadas de las funciones desarrolladas en la sección anterior, tenemos que

$$\left(J_k^T J_k + S_k\right) p_k = -J_k^T f_k \tag{14}$$

$$x_{k+1} = x_k + p_k$$

El principal problema de aplicar el método de *Newton* es el cálculo de $S(\mathrm{x})$, ya que la otra parte que forma la matriz *Hessiana* involucra solamente las primeras derivadas. Tratando de evitar estos costosos cálculos, fue lo que dio origen a otros métodos especializados en mínimos cuadrados no lineales.

El Método de Gauss-Newton. Refiriéndose a la ecuación (14), en este método se ignora tal término S_k. La justificación a esta idea es debida a que entre más $f(\mathrm{x}) \longrightarrow 0$ cuando $x \longrightarrow x^*$ también $S(x) \longrightarrow 0$, por lo que una buena aproximación de la matriz *Hessiana* es

$$J_k^T J_k p_k = -J_k^T f_k$$

Otra característica del método es que $J_k^T J_k$ es al menos semipositiva definida, por lo que casi siempre es posible de invertir.

Bajo estos argumentos hay que mencionar que los posibles problemas en los que se puede incurrir en este método es que la matriz JJ^T sea singular, además que si $E(x^*)$ es substancialmente mayor que cero el método tenderá a converger muy lentamente. Este último caso es conocido como el de los *grandes residuos.*

El Método Levenberg Marquardt. Una manera de solventar el problema de la singularidad de la matriz $J_k^T J_k$ es utilizando este método. La ecuación (14) es modificada para obtener

$$\left(J_k^T J_k + \mu_k I\right) p_k = -J_k^T f_k$$

donde μ_k es un escalar e I la matriz identidad de orden n. Para valores grandes de μ_k, la matriz $\left(J_k^T J_k + \mu_k I\right)$ es positiva definida y p_k es entonces una dirección de descenso debido a (13). Mientras $x_k \longrightarrow x^*$ se requiere que $\mu_k \longrightarrow 0$ de tal manera que el método adquiera el comportamiento del de *Gauss-Newton.*

Mientras $\mu_k \longrightarrow \infty$, $\mu_k I$ domina a $J_k^T J_k$, haciendo que $p_k = (\mu_k I)^{-1} J_k^T f_k$ represente un paso infinitesimal el la dirección de descenso de gradiente. Por otro lado si $\mu_k = 0$, p_k representa un tamaño de paso suficientemente largo en la dirección de máximo descenso. Dentro de estos dos extremos, se tiene entonces la posibilidad de tener una razón de convergencia adecuada que permita al método encontrar un mínimo en un tiempo relativamente corto.

Originalmente el método propuesto por *Levenberg*, escogía una μ_k en cada iteración, lo que llevaba a, primero a buscar un valor adecuado de μ_k y segundo a resolver cada vez el sistema $\left(J_k^T J_k + \mu_k I\right) p_k = -J_k^T f_k$, que ya en la práctica era costoso.

Marquardt (1963) mejoró la eficiencia del algoritmo al establecer una estrategia para seleccionar los valores de μ_k. Inicialmente se escoge un valor positivo para μ_0 y un factor $v > 1$ con el cual μ_k será incrementado o decrementado. Al principio el valor de μ_k es reducido por un factor v con la finalidad de obligar al algoritmo a tener un comportamiento como el de *Gauss-Newton.* Si esto falla en reducir el valor de la función objetivo, el valor de μ_k es inceremetado sucesivamente por un factor v hasta obtener una reducción.

Algoritmo 4 *Levenberg-Marquardt*

x_0 =valor inicial
$\mu_0 = 0.01$
$v = 10$
$k = 0$

Repite

$\mu_k = \mu_k / v$

Repite

Resuelve $\left(\mathbf{J}_k^T \mathbf{J}_k + \mu_k \mathbf{I}\right) \mathbf{p}_k = -\mathbf{J}_k^T \mathbf{f}_k$

$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{p}_k$

Si $E(\mathbf{x}_{k+1}) \geq E(\mathbf{x}_k)$ *entonces* $\mu_k = \mu_k v$

Hasta $E(\mathbf{x}_{k+1}) < E(\mathbf{x}_k)$ o $\mu_k >$ (valor máximo de μ)

$\mu_{k+1} = \mu_k$

$k = k + 1$

Hasta $\left\|2\mathbf{J}_k^T \mathbf{f}_k\right\| \leq$ (valor mínimo del gradiente) o $\mu_k >$ (valor máximo de μ)

Implantación de Levenberg-Marquardt a la Red Neuronal de Dos Capas. Aplicar el método de *Levenberg-Marquardt* para la red neuronal de dos capas requiere modificar la estructura de los *B-nodos* utilizados en la *Propagación Hacia Atrás*. La razón principal de este cambio se debe a que ahora la función a minimizar toma en cuenta las salidas $o_j^{(2)}$ de todas las neuronas para cada patrón $t_j^{(q)}$ de los p patrones, considerados para el proceso de aprendizaje con el fin de poder construir adecuadamente la matriz *Jacobiana*, necesaria para la aplicación del método, es decir, la función de error es ahora definida como,

$$E = \frac{1}{2} \sum_{q=1}^{p} \sum_{j=1}^{m} \left(f_j^{(q)}\right)^2$$

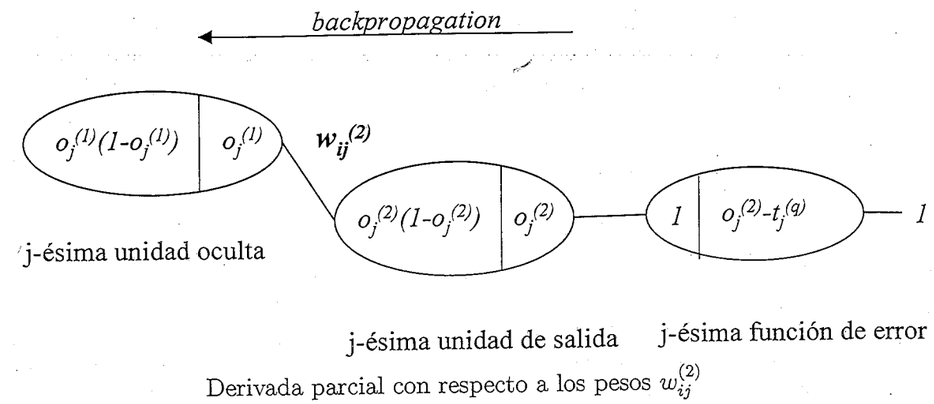
con

$$f_j^{(q)} = o_j^{(2)} - t_j^{(q)}$$

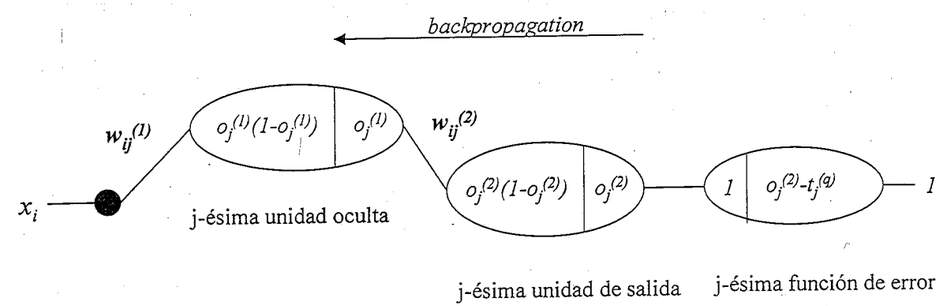
y cuya matriz J es

$$J = \begin{bmatrix} \frac{\partial f_1^{(1)}}{\partial w_{11}^{(1)}} & \cdots & \frac{\partial f_1^{(1)}}{\partial w_{n+11}^{(1)}} & \frac{\partial f_1^{(1)}}{\partial w_{11}^{(2)}} & \cdots & \frac{\partial f_1^{(1)}}{\partial w_{k+11}^{(2)}} \\ \frac{\partial f_2^{(1)}}{\partial w_{12}^{(1)}} & \cdots & \frac{\partial f_2^{(1)}}{\partial w_{n+12}^{(1)}} & \frac{\partial f_2^{(1)}}{\partial w_{12}^{(2)}} & \cdots & \frac{\partial f_2^{(1)}}{\partial w_{k+12}^{(2)}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_m^{(1)}}{\partial w_{1m}^{(1)}} & \cdots & \frac{\partial f_m^{(1)}}{\partial w_{n+1m}^{(1)}} & \frac{\partial f_m^{(1)}}{\partial w_{1m}^{(2)}} & \cdots & \frac{\partial f_m^{(1)}}{\partial w_{k+1m}^{(2)}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_m^{(p)}}{\partial w_{1m}^{(1)}} & \cdots & \frac{\partial f_m^{(p)}}{\partial w_{n+1m}^{(1)}} & \frac{\partial f_m^{(p)}}{\partial w_{1m}^{(2)}} & \cdots & \frac{\partial f_m^{(p)}}{\partial w_{k+1m}^{(2)}} \end{bmatrix}$$

La modificación en los *B-nodos* y en la ruta de *Propagación Hacia Atrás*, para el cálculo de estas derivadas se muestra en las siguientes gráficas.



Derivada parcial con respecto a los pesos $w_{ij}^{(2)}$



Derivada parcial con respecto a los pesos $w_{ij}^{(1)}$

Si definimos

$$\delta_j^{(2)} = o_j^{(2)} \left(1 - o_j^{(2)}\right)$$

entonces las expresiones correspondientes a las derivadas parciales de los pesos de la capa oculta a la de salida, siguiendo la ruta de propagación hacia atrás es

$$\frac{\partial \left(o_j^{(2)} - t_j^{(q)}\right)}{\partial w_{ij}^{(2)}} = \delta_j^{(2)} o_i^{(1)}$$

En cuanto a derivadas parciales de los pesos de la capa de entrada a la oculta son

$$\frac{\partial \left(o_j^{(2)} - t_j^{(q)}\right)}{\partial w_{ij}^{(1)}} = \delta_j^{(2)} w_{ij}^{(2)} o_j^{(1)} \left(1 - o_j^{(1)}\right) x_i$$

El algoritmo completo para utilizar el método *Levenberg-Marquardt* para minimizar la función de error está compuesto de dos partes, la primera para calcular la matriz Jacobiana y la segunda el algoritmo que la utiliza.

Algoritmo 5 *Cálculo de la Matriz Jacobiana para el Algoritmo Levenberg-Marquardt para Red Neuronal de Dos Capas*

1. Para cada patrón $q = 1..p$

2. Propaga-Hacia-Adelante

El vector $\mathbf{x}^{(q)}$ es presentado a la red. Los vectores $\mathbf{o}_j^{(1)}$ y $\mathbf{o}_j^{(2)}$ son calculados y almacenados para cada patrón q. Las derivadas y las funciones de activación son evaluadas y almacenadas en cada neurona.

3. Propagación Hacia Atrás

Las derivadas parciales del error con respecto a los pesos de las neuronas de salida se calculan, como ya fue indicado.

$$\delta_j^{(2)} = o_j^{(2)}\left(1 - o_j^{(2)}\right)$$

$$\frac{\partial\left(o_j^{(2)} - t_j^{(q)}\right)}{\partial w_{ij}^{(2)}} = \delta_j^{(2)} o_i^{(1)}$$

Y para los pesos de la capa de entrada a la oculta.

$$\frac{\partial\left(o_j^{(2)} - t_j^{(q)}\right)}{\partial w_{ij}^{(1)}} = \delta_j^{(2)} w_{ij}^{(2)} o_j^{(1)}\left(1 - o_j^{(1)}\right) x_i$$

4. Se calcula el error y se almacenan las derivadas parciales en la matriz *Jacobiana.*

Para cada neurona de salida y para cada patrón se calcula

$$f_j^{(q)} = o_j^{(2)} - t_j^{(q)}$$

Y en cada renglón de la matriz $\mathbf{J}$ se almacenan la derivadas parciales de $f_j^{(q)}$, como ya fue indicado.

5. Calculamos $\mathbf{Je} = \mathbf{J}^T\mathbf{f}$.

Donde $\mathbf{f}$ es el vector cuyos elementos son $f_j^{(q)}$, para cada patrón q y cada salida j.

Algoritmo 6 *Levenberg-Marquardt para Red Neuronal de Dos Capas*

1. Valores Iniciales

$\mathbf{w}_0$ =valor inicial de los pesos
$\mu_0 = 0.01$
$v = 10$
$k = 0$

2. *Se calcula* $\mathbf{J}$ *y* $\mathbf{Je}$ *con el algoritmo anterior.*

3. *Repite hasta* $\|2\mathbf{Je}\| \leq$ (valor mínimo del gradiente) o $\mu_k >$ (valor máximo de μ) o $\sum_{q=1}^{p}\sum_{j=1}^{p}\left(o_j^{(2)} - t_j\right)^2 >$ (valor mínimo de error)

$\mu_k = \mu_k / v$
Repite
Resuelve $\left(\mathbf{J}_k^T\mathbf{J}_k + \mu_k\mathbf{I}\right)\mathbf{p}_k = -\mathbf{Je}$
$\mathbf{w}_{k+1} = \mathbf{w}_k + \mathbf{p}_k$
Si $E(x_{k+1}) \geq E(x_k)$ *entonces* $\mu_k = \mu_k v$
Hasta $E(x_{k+1}) < E(x_k)$ o $\mu_k >$ (valor máximo de μ)
Actualiza los $\mathbf{w}^{(1)}$ y $\mathbf{w}^{(2)}$ con $\mathbf{w}_{k+1}$.
Se calcula $\mathbf{J}$ *y* $\mathbf{Je}$
$\mu_{k+1} = \mu_k$
$k = k + 1$

5.2.4 Comparación de Algoritmos de Aprendizaje

Con el propósito de comprobar la rapidez de aprendizaje de la red neuronal al utilizar el algoritmo anteriormente descrito, se realizó una comparación con otros dos métodos.

Un algoritmo con el cual se comparó es el tradicional *Descenso de Gradiente*, el otro método utilizado se denomina *QRprop*[5]. Este algoritmo trabaja con dos estrategias, dependiendo de los valores que vaya tomando el gradiente en cada iteración. En la primera de ellas, la idea principal es ajustar los pesos w_i utilizando solamente una tasa de aprendizaje $\gamma_i^{(k)}$ individual en cada iteración k, que depende del signo de la derivada parcial de la función de error con respecto a cada uno de los pesos w_i en dos iteraciones sucesivas, es decir, sí $\nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} > 0$, donde $\nabla_i E^{(k)}$ es el i-ésimo componente del vector gradiente en la k-ésima iteración, se trata de avanzar rápidamente en esa dirección, ya que indica que un mínimo local se encuentra más adelante.

Por otro lado, segunda estrategia , si $\nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} \leq 0$, se debe a que un mínimo local se ha dejado atrás en la dirección w_i, ya que ha cambiado de signo del gradiente, o que se ha llegado a un mínimo local al tener un valor de cero, también en esa misma dirección. En ambos casos se encuentra cerca un mínimo local y el algoritmo realiza una aproximación cuadrática del gradiente con respecto a la función de error en esa dirección.

El algoritmo se puede resumir en los siguientes pasos:

Algoritmo 7 *QRprop*

1. Actualización de las tasas de aprendizaje individuales

$$
\begin{array}{l}
Si\ \nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} = 0\ entoces \\
\quad Si\ \nabla_i E^{(k)} \neq \nabla_i E^{(k-2)}\ entonces \\
\qquad q_i = \max\left(d, \min\left(\frac{1}{u}, \left|\frac{\nabla_i E^{(k)}}{\nabla_i E^{(k)} - \nabla_i E^{(k-2)}}\right|\right)\right) \\
\quad \text{Sino} \\
\qquad q_i = \frac{1}{u}
\end{array}
$$

$$
\gamma_i^{(k)} = \begin{cases} \min\left(u \cdot \gamma_i^{(k-1)}, \gamma \max\right),\ si\ \nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} > 0 \\ \gamma_i^{(k-1)} \qquad\qquad si\ \nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} < 0 \\ \max\left(q_i \cdot \gamma_i^{(k-1)}, \gamma \min\right)\ si\ \nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} = 0 \end{cases}
$$

2. Actualización de pesos

$$
w_i^{(k+1)} = \begin{cases} w_i^{(k)} - \gamma_i^{(k)} \cdot sign\left(\nabla_i E^{(k)}\right)\ si\ \nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} \geq 0 \\ w_i^{(k)} \qquad\qquad de\ otra\ forma \end{cases}
$$

$$
Si\ \ \nabla_i E^{(k)} \cdot \nabla_i E^{(k-1)} < 0\ entonces\ hacer\ \nabla_i E^{(k)} = 0
$$

Las constantes u y d generalmente satisfacen $u > 1$ y $d < 1$. Los valores $\gamma \min$ y $\gamma \max$ sirven para evitar grandes oscilaciones en los pesos, y cuyos valores son dados de antemano.

La comparación se realizo con una red neuronal compuesta por dos neuronas de entrada, dos de salida y tres ocultas. El número de patrones presentados a la red fue de 150 y se utilizó como función de error de la red la suma de los errores cuadráticos de cada neurona de salida. Los resultados obtenidos se muestran en la gráfica de la figura 16.

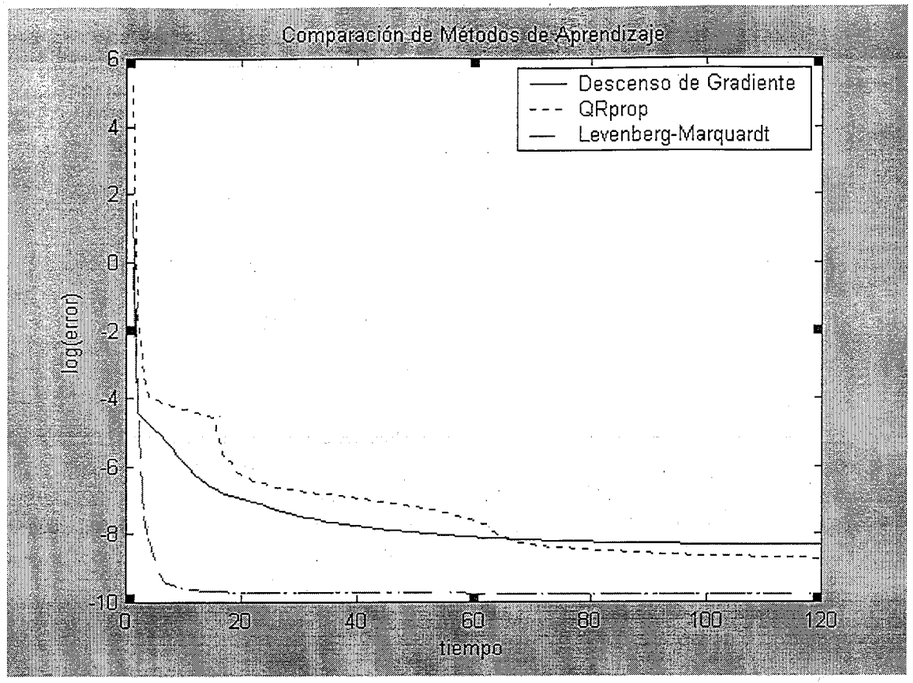
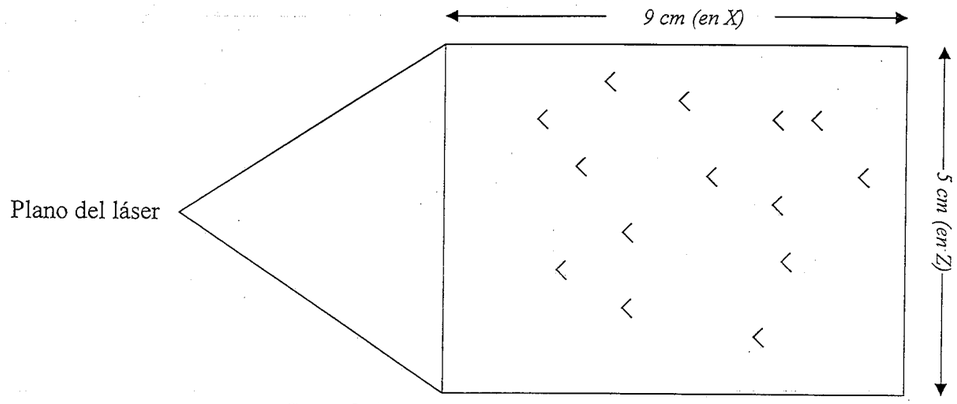


Figura 16: Comparación de métodos de aprendizaje.

6 Experimentos

Los primeros experimentos realizados fueron con el objetivo de medir la exactitud del digitalizador con respecto a los dos métodos de calibración propuestos, la matriz de proyección y la red neuronal. Para esto se realizó la calibración con 150 puntos, cuyas coordenadas fueron seleccionadas en forma aleatoria, dentro de una superficie de aproximadamente 9×5 cm.



Superficie de calibración y de prueba

Los parámetros de entrenamiento de la red neuronal utilizada en este experimento fueron: dos entradas, tres neuronas en la capa oculta y dos salidas.

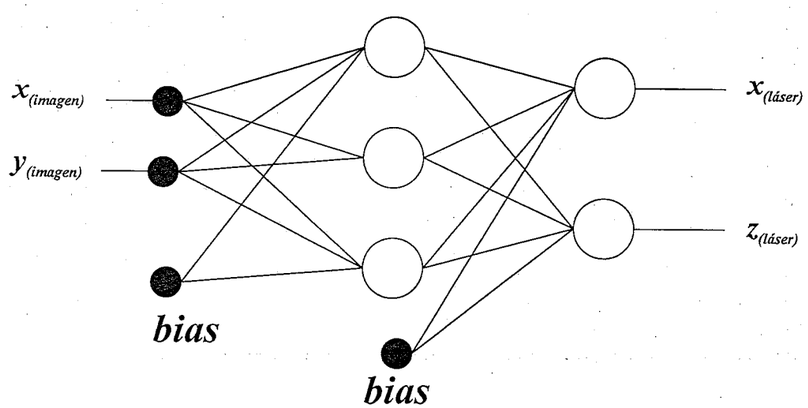
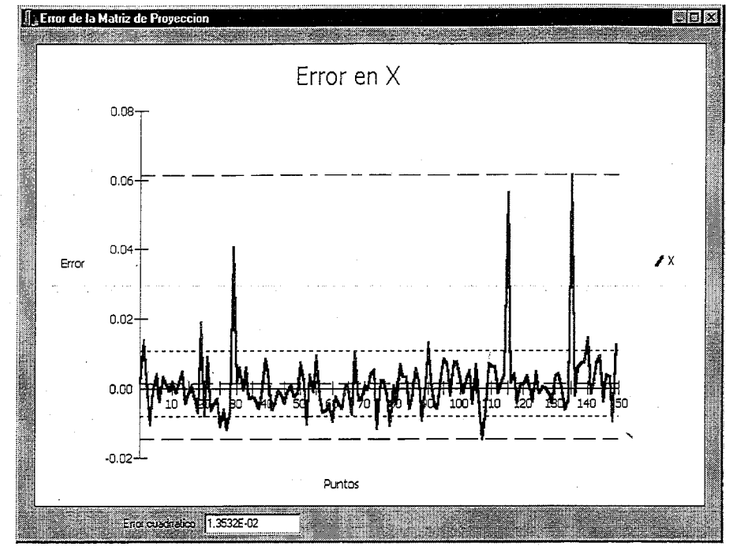


Figura 17: Estructura de la red neuronal

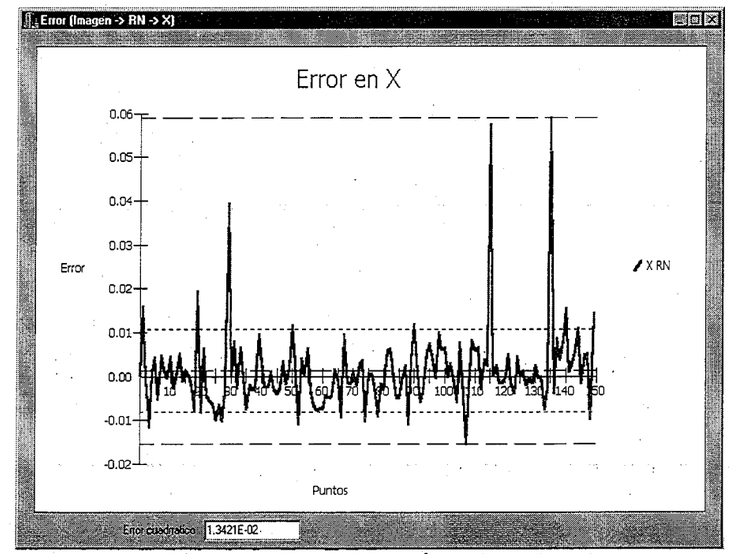
Como puede observarse en la figura 17, las entradas a la red neuronal fueron las coordenadas de los subpixeles de las imágenes de los puntos de calibración y las salidas fueron las coordenadas, conocidas, en el plano del láser (X, Z) de estos mismos. El tiempo transcurrido en el entrenamiento fue de aproximadamente doce minutos en 5000 épocas, una época es una iteración en la cual todo el conjunto de patrones de entrada y de salida son presentados a la red para su entrenamiento. La sumatoria del error cuadrático obtenido al final de este proceso fue de 5.8686e-03 cm.

Posteriormente, se tomaron otros 150 puntos de prueba, también con coordenadas conocidas y seleccionados en forma aleatoria dentro de la misma área, los cuales fueron usados para comprobar la precisión de los métodos.

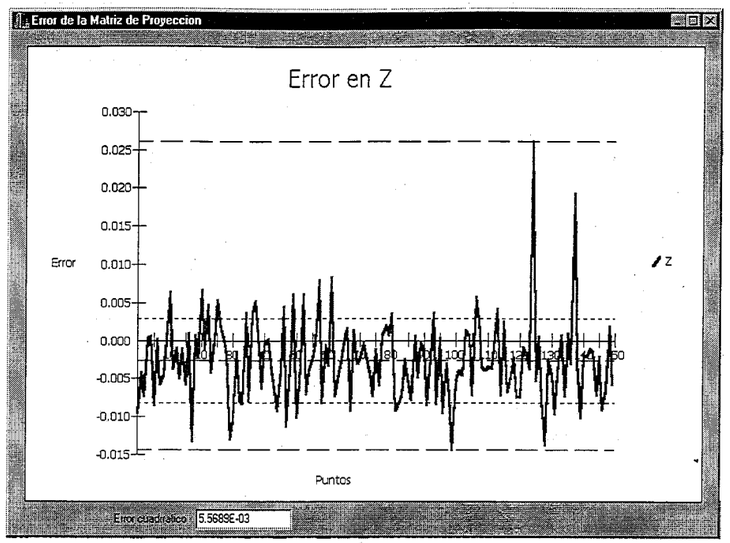
Los resultados obtenidos, tanto para el eje X, como para el Z se muestran a continuación:



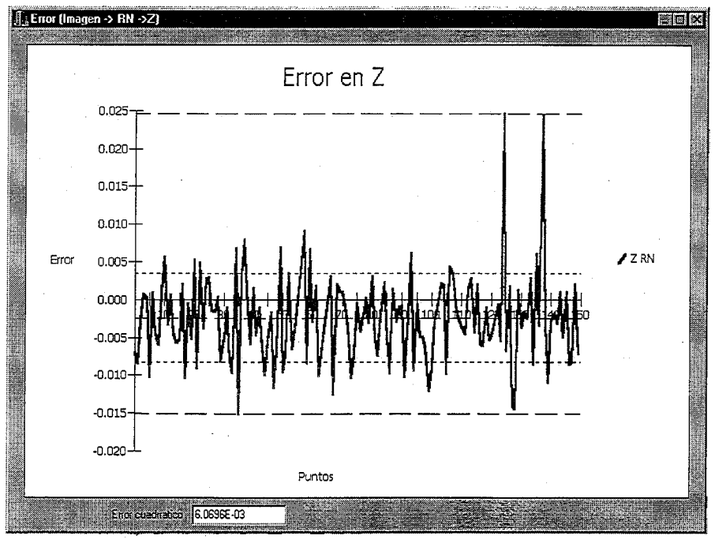
Error en la coordenada X, matriz de proyección.



Error en la coordenada X, red neuronal.



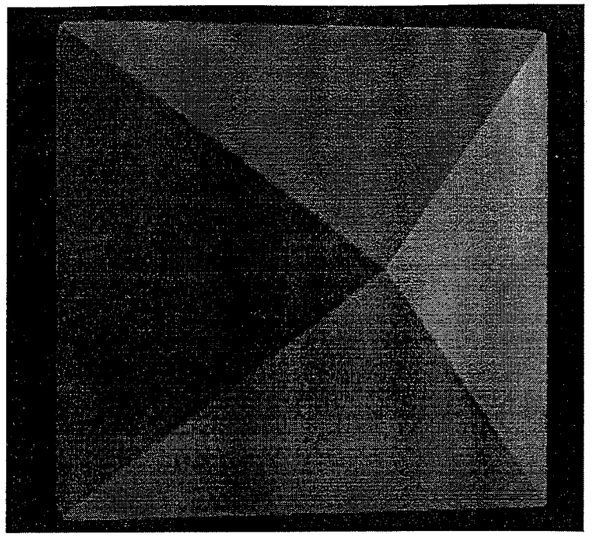
Error en la coordenada Z, matriz de proyección.



Error en la coordenada Z, matriz de proyección.

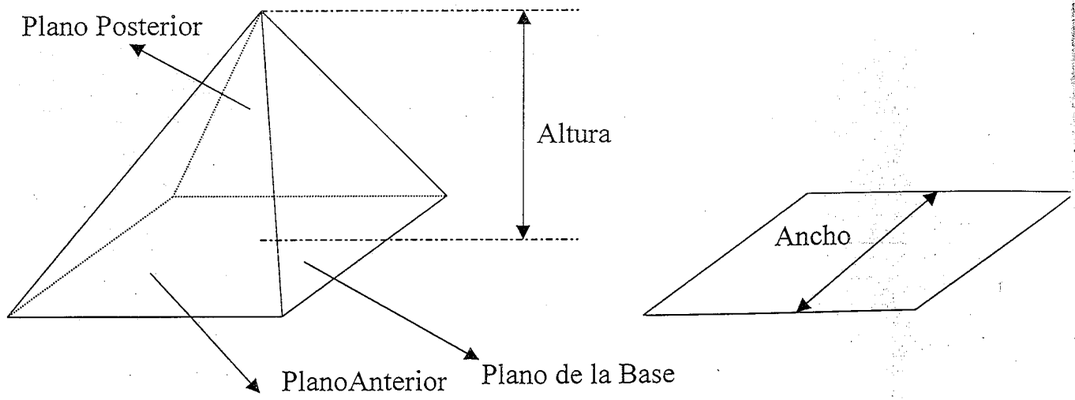
Las gráficas anteriores, muestran respectivamente el error, en centímetros, incurrido en cada punto de prueba y el error cuadrático de cada método al calcular la coordenadas X, Z.

Otro experimento que se realizó, también con el objetivo de comprobar la precisión, fue digitalizar una pirámide de base cuadrada, previamente calibrada con una máquina de medición por coordenadas.



Pirámide Real

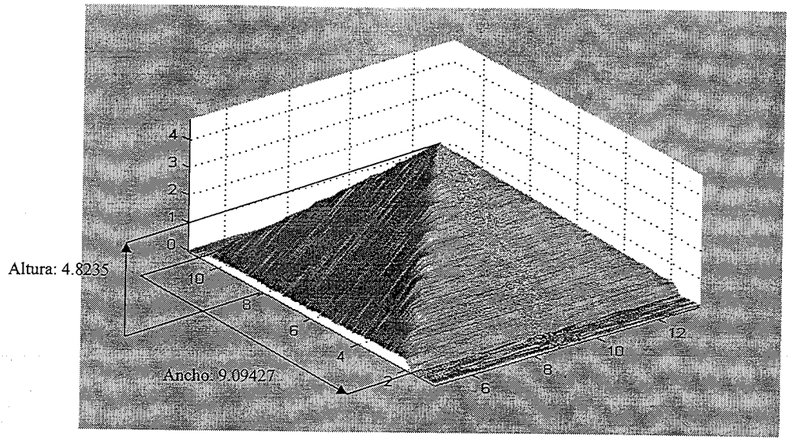
De acuerdo a las especificaciones proveídas de la figura, se pudo hacer la comparación de la altura y la distancia entre dos rectas de la base, el ancho.



Elementos a comparar

La comparación se realizo utilizando los dos métodos de calibración mencionados, y los resultados fueron:

Comparación de Mediciones entre Maquina de Medición de Coordenadas y
El Modelo Propuesto Utilizando Matriz de Proyecciones

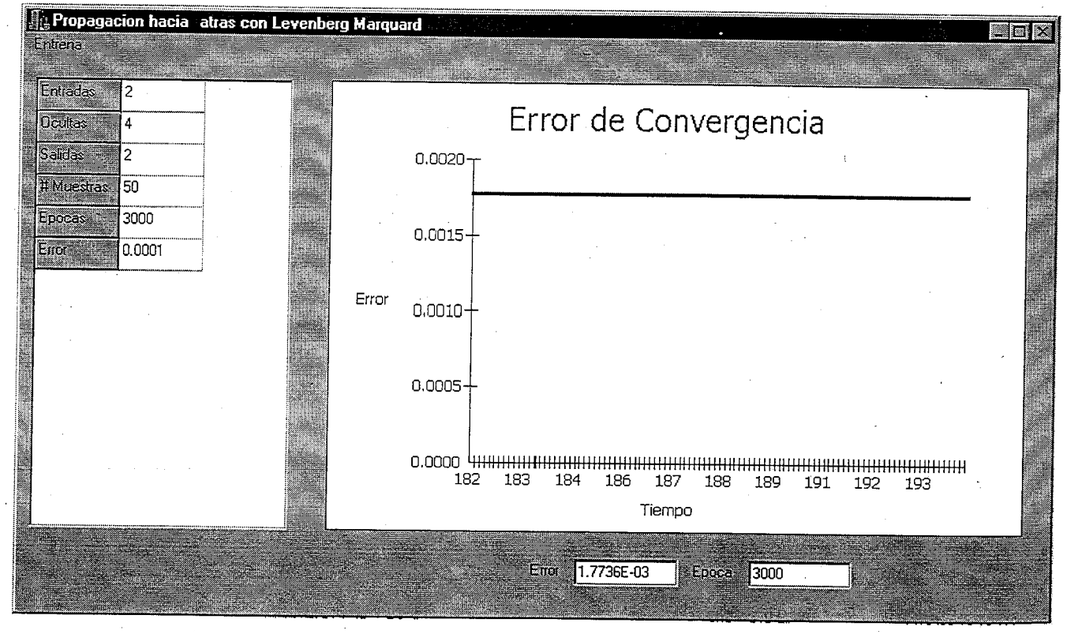


Altura: Valor Nominal: 4.85, Valor Real 4.806, Valor Medido 4.8235, Desviación: 0.0175
Ancho: Valor Nominal: 8.95, Valor Real 8.947, Valor Medido 9.0942, Desviación: 0.1472

Mediciones en cm

Figura 34.- Digitalización con la Matriz de Proyección

Para el caso de la Red Neuronal, se entrenó con las siguientes especificaciones.

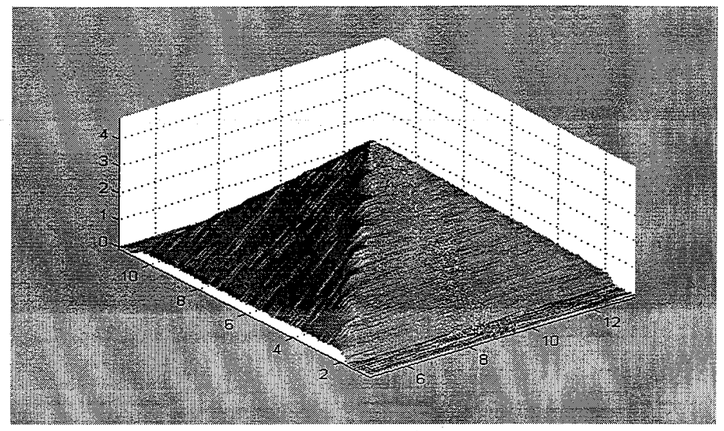


Estructura y Error de Converegncia de la Red Neuronal

La figura muestra el *Error Cuadrático*, el número de épocas y el tiempo que tardó la red para alcanzar estos valores.

Los resultados obtenidos al utilizar la red neuronal en la digitalización fueron los siguientes.

Comparación de Mediciones entre Maquina de Medición de Coordenadas y
El Modelo Propuesto Utilizando la Red Neuronal



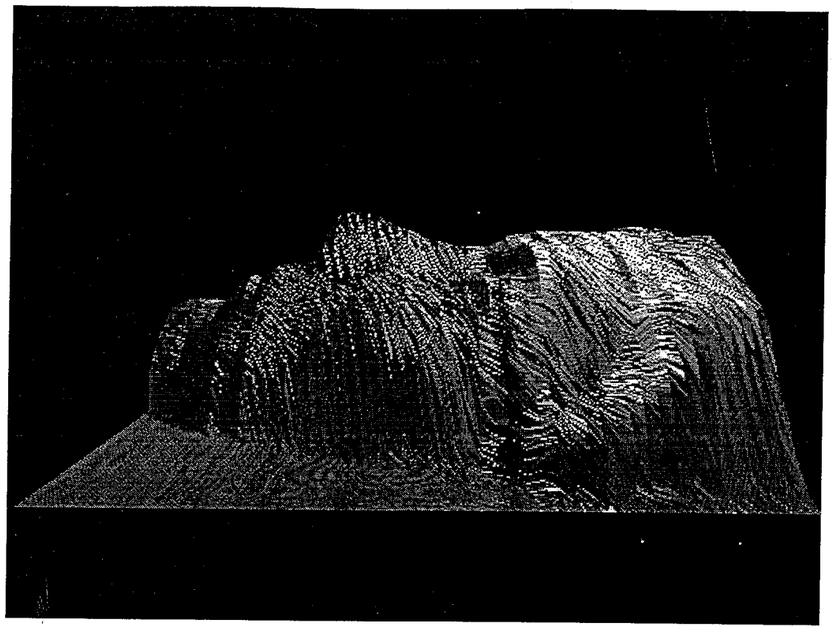
Altura: Valor Nominal: 4.85, Valor Real 4.806, Valor Medido 4.8441, Desviación: 0.0381
Ancho: Valor Nominal: 8.95, Valor Real 8.947, Valor Medido 9.0959, Desviación: 0.1489

Mediciones en cm

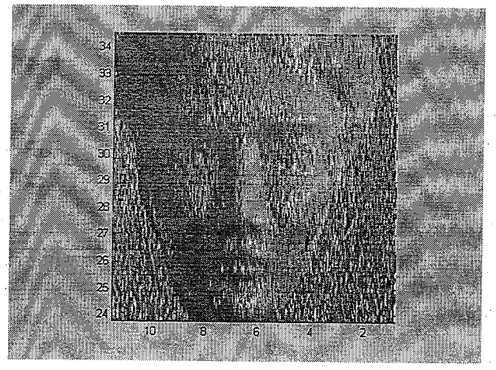
Digitalización con la Red Neuronal

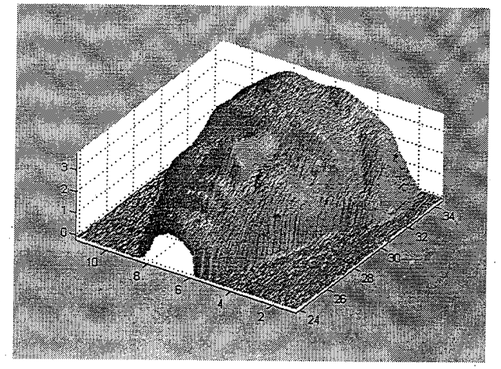
En ambos métodos, las mediciones se obtuvieron; para el caso de la altura, con la diferencia entre el valor más alto y el mas bajo en X y para la distancia entre dos de las rectas que forma la base, se midió utilizando las intersecciones de los planos *Plano Posterior y Plano Anterior* con el *Plano de la Base*, cuyas ecuaciones fueron calculadas utilizando[7].

Finalmente se hicieron otras digitalizaciones como prueba, pero solamente se muestra una como ejemplo.



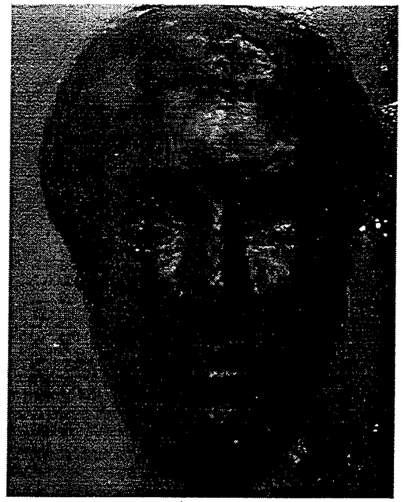
Reconstruida Utilizando un Trazador de Rayos





Reconstruida Interpolando la Superficie

Estas figuras corresponden a una cara hecha de plastilina; la primera de ellas fue reconstruida con un programa de trazo de rayos, en el que se puede indicar posición de la cámara, luminosidad y textura, la segunda es simplemente utilizando todos los puntos de la digitalización e interpolando superficies entre ellos.



Cara de Plastilina Real

7 Conclusiones

En esta parte de la tesis, se tratará de llegar a algunas conclusiones acerca de los resultados obtenidos en los experimentos, así como el trabajo futuro que permita experimentar y por lo tanto conocer más el comportamiento del digitalizador, principalmente en la red neuronal como método de calibración. Pero antes de comenzar estas explicaciones, es conveniente decir que todo el conocimiento necesario para llevar a cabo este trabajo fue proporcionado por los cursos impartidos en la maestría y/o por la orientación de los investigadores del área de cómputo del CIMAT.

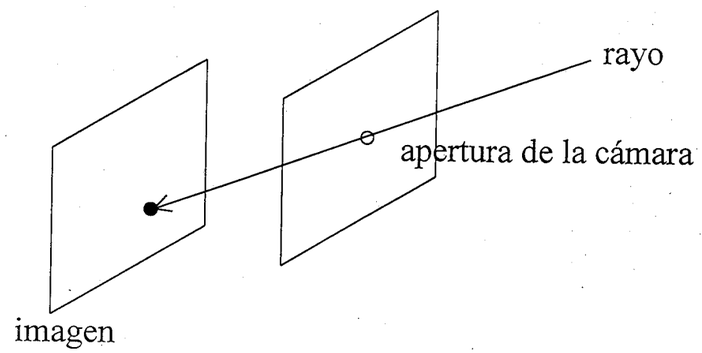
7.1 Resultados

Los resultados significativos obtenido en la sección *Experimentos*, se pueden dividir en los siguientes puntos:

1. El error cuadrático obtenido por los dos métodos de calibración al calcular las coordenadas de los puntos de prueba, se encuentra en el orden de décimas de milímetro en la coordenada X, del plano del láser, y de centésimas de milímetro en la coordenada Z, lo que significa dos cosas; que la detección de líneas utilizando filtros de *Gabor*, así como la aplicación del algoritmo *MPM-MAP* mejoraron la precisión de la digitalización al obtener coordenadas en la imagen a nivel subpixel y, segundo, que los dos métodos reflejan adecuadamente el modelo físico que interviene en la reconstrucción tridimensional.

2. Una característica importante, también relacionada con los filtros de *Gabor* y el algoritmo *MPM-MAP* fue su robustés al ruido; los filtros, por un lado, son robustos al ruido que se pudiera tener en las imágenes obtenidas de las líneas proyectadas por el rayo láser, ya sea sobre el prisma utilizado en la calibración, como en la digitalización, repercutiendo en forma definitiva en la precisión del dispositivo, por otro lado el algoritmo *MPM-MAP* estimó en forma adecuada las rectas que sirvieron para ubicar las coordenadas de los puntos de calibración, a pesar de la desviación de las coordenadas de los puntos en la imagen con respecto a los puntos que idealmente formaban las rectas al pasar el rayo por un canto de la pirámide, mejorando también la precisión.

3. El hecho de que, el error cuadrático obtenido al aplicar la red neuronal a los puntos de prueba no mejora en forma significativa el error incurrido por la matriz de proyección, no importando el número de épocas con que se entrenara la red, aunado a que las gráficas de error de los dos métodos siguen, casi en forma idéntica, el mismo patrón, muestra que el modelo físico de la cámara, para las dimensiones de la superficie del plano láser en el que se realizó el experimento, se acerca en forma definitiva al modelo de *cámara obscura*, llamado *pinhole* el cual significa que solamente un rayo proveniente de cualquier punto puede entrar a la cámara por un diminuto agujero, creando una correspondencia uno a uno entre puntos visibles, rayos, y puntos en la imagen,



Modelo de la cámara obscura.

dicha correspondencia es precisamente la base de la transformación proyectiva utilizada por el método de la matriz de proyección.

4. Otro aspecto relacionado con el punto anterior es, asumiendo que el modelo de *pinhole* simula en forma ideal al modelo físico del experimento, entonces la red neuronal es también una muy buena aproximación de él.

5. En cuanto al experimento con la pirámide de base cuadrada, la dimensión de la altura, obtenida de acuerdo al reporte de la máquina de medición por coordenadas, difiere de la obtenida por el digitalizador en décimas de milímetro. La diferencia que se obtuvo al comparar la distancia entre dos rectas de la base fue mayor, del orden de milímetros, pero esta diferencia así como la correspondiente a la altura, desgraciadamente no son muy significativas, debido principalmente a que el reporte que acompañaba a la figura calibrada no indica la manera en que habían sido determinadas estas distancias.

7.2 Trabajo Futuro

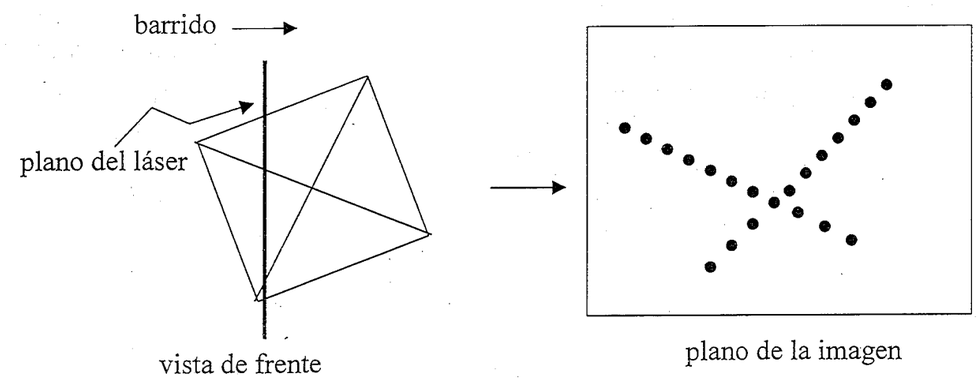
Dentro de los trabajos a realizar para facilitar y mejorar la calidad del digitalizador, podríamos enumerar los siguientes:

I. Comprobar la precisión de los métodos de calibración.- Como ya se mencionó la red neuronal, así como la matriz de proyección modelaron en forma adecuada el modelo físico del digitalizador, pero estos experimentos fuero realizados dentro de una área de dimensiones pequeñas, plano del láser, por lo que sería conveniente comprobar la precisión de los métodos de calibración a distancias mayores, es decir aumentando el área donde estarían ubicados los puntos de calibración con la finalidad de digitalizar objetos de mayores dimensiones, y comprobar que la no linearidad introducida por la cámara, debida a factores como tipo de lente, campo de visión o apertura angular incurran en errores en el modelo proyectivo, comparándolo al mismo tiempo con la red neuronal, la cual, escogiendo adecuadamente sus parámetros, puede aproximar en forma precisa cualquier la función en la que intervienen estos factores que nos son fáciles de determinar.

II. Oclusiones.- Éste es un problema que se encontró al digitalizar la cara de plastilina, mostrada en la sección anterior, y que se puede presentar en muchos objetos, en los cuales al pasar el rayo láser sobre su superficie se encuentre con partes del objeto mismo que lo obstaculicen, produciendo en la imagen captada por la cámara regiones donde no se pueda localizar la línea proyectada por el rayo. Este problema podría se resuelto con más cámaras, lo cual sería relativamente fácil de implantar, pero aún así siempre existirían pequeñas zonas ocluídas ya que este problema es inherente al enfoque óptico del digitalizador, lo cual obligaría a tratar las oclusiones de alguna forma.

III. Experimentación en una máquina de medición de coordenadas.- Con la finalidad de verificar la precisión de la digitalización, sería de gran ayuda poder contar con una máquina de medición de coordenadas, en la cual se instalaría también la cámara y el rayo láser, aprovechándose al mismo tiempo el movimiento que tiene en los tres eje coordenados para realizar la calibración y la digitalización, teniéndose de esta forma el mismo error o incertidumbre, además de saberse de manera precisa cómo se realizarían las mediciones de algunas características de un objeto de calibración, permitiendo hacer más objetiva la comparación entre el método de contacto y el óptico aquí propuesto.

IV. Aspectos prácticos de calibración.- Pensando en construir un digitalizador con fines prácticos, sería conveniente buscar una manera de calibrar que utilizara el único movimiento que se realiza al digitalizar un objeto, es decir, el barrido en el eje Y de la mesa de coordenadas, de tal forma que no se tuviera la necesidad de movimientos en los otros dos ejes, reduciendo por lo tanto de manera significativa el costo del dispositivo. Una posibilidad sería tener como objeto de calibración una pirámide regular de base cuadrada,



cuya altura abarcara toda el área del plano del láser y ubicada de tal forma que al moverla solamente en el eje Y, el láser fuera proyectando rectas que se intersectaran en diferentes puntos, no todos colineales, dentro del plano del la imagen.

Referencias

[1] Signal Processing for Computer Vision, Gosta H. Granlund and Hams Knutsson, Kuwer Academic Publisher, 1995.

[2] The MPM-MAP Algorithm for Image Segmentation, J. L. Marroquín, F. Calderón, S. Botello, and B. Vemuri, Reporte Técnico, january 27 2000.

[3] Projective Geometric and its Aplications to Computer Graphics, Michel A. Penna and Richard R. Patterson, Prentice-Hall, 1986.

[4] Introductory Techniques for 3-D Computer Vision, Emanuele Trucco, Alessandro Verri, Prentice-Hall, 1998.

[5] Neural Networks a Systematic Introduction, Raul Rojas, Springer-Verlag, 1996.

[6] Introduction to Non-Linear Optimization, L.E. Scales, Department of Computer Sciences University of Liverpool.

[7] Reporte de Proyecto *PAJUSTA*, Desarrollo Tecnológico de la Maestría en Cómputo del C.I.M.A.T.

Otras Referencias Bibliográficas

1. Apuntes del Curso de Procesamiento de Señales I, José Luis Marroquin, 1999.
2. Digital Image Processing; Concepts, Algorithms, and Scientific Applications 4th edition, Bernd Jahne, Springer-Verlag, 1997.
3. Introduction to Linear Algebra, Gilbert Strang, Wellesley-Cambridge Press, 1998.
4. Neural Networks for Patter Recognition, Christopher M. Bishop, Oxford Clarendon Press, 1995.