UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Sistema de medición dimensional 3D utilizando luz estructurada

T E S I S

QUE PARA OBTENER EL GRADO DE:
INGENIERO EN
TELECOMUNICACIONES

PRESENTA

ULISES MARTÍNEZ GILBÓN

DIRECTOR DE TESIS: M.I. BENJAMÍN VALERA OROZCO

MEXICO, D.F. Octubre de 2004.

Agradecimientos

Gracias a la Universidad Nacional Autónoma de México por darme una educación de calidad y la formación para ser un profesionista comprometido con mi país.

Al proyecto PAPIIT IN11100 de la dirección General de Asuntos del Personal Académico de la UNAM : "Utilización de la visión estéreo para mediciones geométricas precisas sin contacto" el cuál aportó los recursos para el desarrollo del proyecto.

Al Laboratorio de Metrología del Centro de Ciencias Aplicadas y Desarrollo Tecnológico de la UNAM y a sus integrantes:

M.en I. Rigoberto Nava Sandoval, M. en I.Gerardo Ruíz Botello, Ing. Sergio Padilla Olvera, M. en I.José Sánchez Vizcaíno, y M. en I. Benjamín Valera Orozco por su apoyo durante todo el proceso de realización de la tesis.

Al Dr. Victor García Garduño, jefe del Departamento de Telecomunicaciones de la Facultad de Ingeniería de la UNAM por el apoyo brindado en el desarrollo del proyecto.

Dedicatorias

A mis padres, por ser un ejemplo de rectitud, de honestidad y de compromiso. Gracias por haberme apoyado en todas mis decisiones, por acompañarme en mis triunfos y derrotas, y por haberme dado la dicha de ser querido tanto.

A mi hermano, Aris, por haber abierto camino para mi superación, ser mi mejor amigo y ser ejemplo de perseverancia y tenacidad.

A mis abuelitas por haberme dado su tiempo en mi niñez para poder salir adelante.

A mis abuelos que en paz de descansen, han sido guía para mí.

A mis tías por estar al pendiente de mí y de que no me faltara nada.

A mis amigos, que me han acompañado en las buenas y en las malas, a mi hermano Ricardo que siempre lo será, a Victor, a todo el staff de NGWISE: Angel, Amaury, Alex, Axl, Roberto, Hugo, Gustavo e Ivonne, que cada día se vuelven más como de mi familia.

A Ania, por haberme acompañado estos últimos años y complementar así mi vida.

A la Universidad, por haberme dado cabida en sus aulas y dejar huella en mí, con un gran orgullo seré siempre Puma Universitario.

Y un gran agradecimiento a Benjamín Valera, por ser un ejemplo de dedicación y por haber sido mi guía en este trabajo y mi mentor por siempre. Gracias por tu apoyo para que esta tesis por fin llegara a su fin.

Índice temático

Introducción

Objetivo

Desarrollar un sistema de medición dimensional sin contacto, utilizando visión por computadora, técnicas de luz estructurada y algoritmos de control de posición integrados en un instrumento con operación en tiempo real.

Definición del problema

En la actualidad, los instrumentos de medición dimensional han evolucionado rápidamente ya que su uso implica la mejora en los procesos de manufactura y control de calidad. La integración de sistemas computacionales a sistemas mecánicos, hidráulicos, eléctricos y de control, ha marcado la pauta para la constante innovación de estas máquinas.

Aunado a las mejoras, viene también la elevación de los costos de fabricación y mantenimiento de estos instrumentos, ya que para conservar su alta calidad necesitan de partes de gran exactitud y llevar una constante vigilancia y calibración.

Por otro lado, la automatización es un factor importante a considerar, ya que si se desea tener una medición dimensional confiable, es necesario considerar tiempo máquina y tiempo hombre, que para la industria significan dinero.

Es debido a esto que una solución fiable la constituyen los Sistemas de Medición por Visión, SMV, ya que la mayoría de los ajustes pueden ser realizados mediante software, bajando drásticamente los costos, pagando tal vez el precio de una menor exactitud comparada con los instrumentos mecánicos como las Máquinas de Medir por Coordenadas, MMC, o los Sistemas Láser de Medición, SLM. Sin embargo la

automatización y las altas tasas de muestreo de un SMV equilibran sus cualidades frente a los demás sistemas.

En el presente trabajo se desarrolló un SMV que cumple por un lado con un bajo costo, y por otro con exactitud adecuada para algunas aplicaciones, además de ser un proceso autónomo en su mayoría.

Antecedentes

Actualmente, en el Laboratorio de Metrología del CCADET UNAM, se han desarrollado algoritmos y se ha creado infraestructura hardware y software para la medición por visión [31]. En particular, se ha experimentado con técnicas para la visión estéreo con modelos lineales [16] y técnicas de procesamiento de imágenes como auxiliares a los procesos de medición por visión [14]. La experiencia adquirida nos ha permitido plantear un desarrollo propio de un SMV con exactitud y alta tasa de muestreo. Concretamente, comprobamos que la exactitud de un SMV radica en el empleo de modelos no lineales, luz estructurada y extracción de características con nivel sub-pixel. Por otra parte, la automatización del proceso requiere del empleo de técnicas para el control de posición angular desarrolladas en años recientes [34].

La amplia gama de aplicaciones nos motivan a desarrollar y perfeccionar nuestras técnicas para la medición por visión, debido a que los problemas inherentes al procesamiento de imágenes siguen estando abiertos.

En forma adicional, presentamos un esquema general para la medición por visión del presente proyecto en el Congreso Iberoamericano de Reconocimiento de Patrones CIARP 2003 [17].

Descripción de problema a resolver

La medición dimensional implica conocer ciertos puntos de interés en el espacio tridimensional, para eso es necesario hacer un número determinado de mediciones para satisfacer la necesidad de densidad de datos que se requiera, lo que lleva a un arduo trabajo. Con los SMV, se logran capturar varios puntos con una sola lectura lo que hace más eficiente la labor de la medición.

La visión por computadora, en esencia trata de imitar la visión humana, la cual, al ser estereoscópica permite reconocer objetos en tres dimensiones. Un sólo ojo, capta imágenes en dos dimensiones y es el sistema de visión el que se encarga de reconstruir los objetos en 3D en el cerebro, al combinar las imágenes captadas por ambos ojos.

Una forma de visión por computadora es utilizar el mismo principio estéreo, es decir, utilizar dos cámaras. Lo anterior es utilizado en aplicaciones de reconstrucción de objetos, en visión para robots autómatas, entre otras más. Sin embargo, tiene el inconveniente que requiere de un gran procesamiento de las imágenes, lo que lleva a utilizar hardware y software muy robusto.

En este trabajo, se utiliza otro enfoque para lograr la reconstrucción 3D, utilizando solamente una cámara y un sistema de luz estructurada, que en nuestro caso esta constituido por un patrón lineal de luz, generado mediante un láser de línea. De esta forma es posible reconstruir una escena 3D a partir de la imagen 2D capturada por la cámara, debido a que el patrón de luz estructurada proporciona los datos necesarios para poder conocer la dimensión faltante.

La base del instrumento para que pueda reconstruir 3D es la calibración de la cámara y del plano, con lo cual es posible hacer un modelo matemático que transforme las coordenadas 2D a 3D. La calibración, se apoya en un instrumento patrón, que en nuestro caso utilizamos una MMC.

Entonces el problema a resolver consiste en el desarrollo e integración de un instrumento para la medición dimensional de una pieza 3D, utilizando las herramientas desarrolladas por el Laboratorio de Metrología y la experiencia adquirida.

Relevancia y justificación

La relevancia del SMV radica en la alternativa que planteamos para la medición dimensional en la sustitución de instrumentos de medición costosos como MMC y SLM. En forma adicional, aprovechamos las ventajas bien conocidas de los SMV como son la alta tasa de muestreo, la poca inversión en infraestructura y el enfoque orientado hacia la programación [11, 24].

El instrumento planteado en la presente tesis ofrece un enorme ahorro de tiempo y costo de operación, ya que una vez calibrado el SMV, este puede realizar mediciones consecutivas en el mismo espacio de medición. Si se requiere modificar las condiciones de observación (posición relativa de la cámara o foco) basta con recalibrar el instrumento usando las herramientas incluidas en el software desarrollado específicamente para la aplicación.

Lo más importante es que se culmina con un instrumento capaz de digitalizar superficies 3D completas mediante la sincronización de algoritmos de medición y control. El instrumento puede ser utilizado

en servicios especializados que ofrece el Laboratorio de Metrología o bien ser transferido a la industria.

Método

El método empleado se puede sintetizar como el modelado de sistemas físicos e integración de técnicas de procesamiento de imágenes y control automático en un instrumento de medición. El instrumento desarrolla una medición 3D completa en tres etapas principales: Procesamiento de imágenes, algoritmo de medición y algoritmo de control. La figura 1 muestra el proceso de medición dimensional 3D empleado y se describe a continuación.

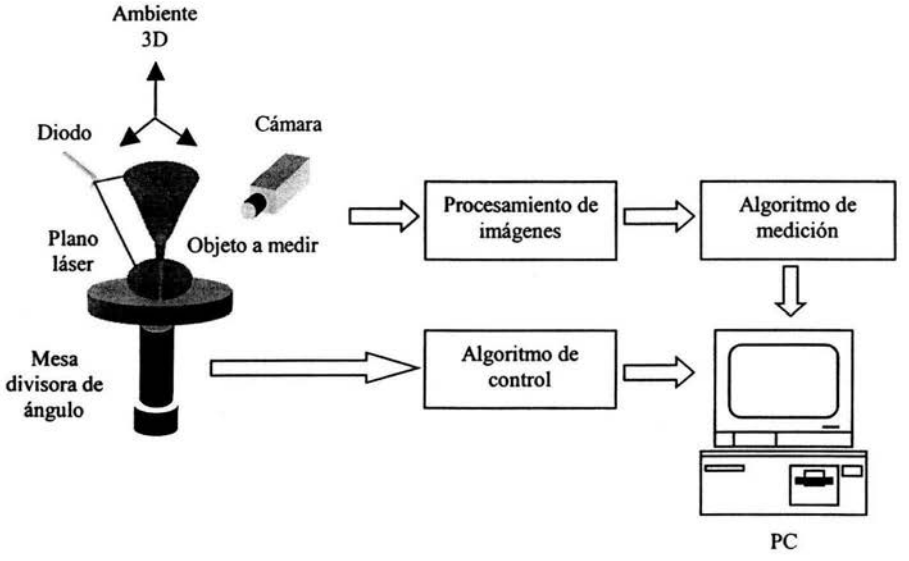


Figura 1. Automatización del SMV propuesto.

En la figura 1 un objeto a medir es colocado en el ambiente global 3D. Un diodo láser de línea genera el patrón de luz estructurada que intersecta la superficie, resaltando algunas características de interés. El patrón de luz genera un plano 3D con perfil aproximadamente gaussiano.

Como parte de la etapa de procesamiento de imágenes, se captura un par de imágenes planas mediante hardware comercial y software desarrollado específicamente para esta aplicación. El par de imágenes es procesado para extraer las características planas del patrón de luz que resalta el objeto a medir.

En la etapa del algoritmo de medición las características 2D son reconstruidas en un ambiente 3D mediante la calibración del arreglo físico. En particular, se calibran la cámara y el plano que forma la luz estructurada mediante un patrón dimensional de mayor exactitud como MMC. La reconstrucción 3D de un conjunto de características constituye una medición parcial a la cual llamaremos *meridiano*.

El algoritmo de control consiste en posicionar angularmente el objeto bajo inspección que es colocado sobre la mesa divisora mediante hardware comercial y software desarrollado específicamente para el control realimentado.

Las etapas de adquisición de imágenes, algoritmo de medición y algoritmo de control son gobernados por el software de desarrollo especifico a esta aplicación en la PC. Los tres algoritmos deben estar sincronizados de manera que una medición sea el conjunto de meridianos que se obtienen al rastrear el objeto en los 360° de la mesa divisora. El rastreo se lleva a cabo de forma discreta y cada meridiano es compensado con la posición angular registrada.

Resultados esperados

Tener un instrumento de medición dimensional utilizando visión por computadora desarrollado completamente en el Laboratorio de Metrología del CCADET. El sistema incluye el desarrollo de un software específico para la medición. Además el software permite la calibración del instrumento, es decir, la calibración de la cámara y el plano de láser mediante una interfase en tiempo real con la MMC.

El software desarrollado utiliza exclusivamente herramientas de programación básicas, prescindiendo de bibliotecas (libraries) y utilerías de terceras partes.

Por otro lado, se consigue el diseño e implementación de los circuitos electrónicos para la sincronización de los algoritmos de procesamiento de imágenes, medición y control.

Todo esto integra un instrumento de medición con calidad de producto terminado y exactitud adecuada para aplicaciones como entretenimiento, diseño artístico, algunas aplicaciones médicas e industriales.

Alcance y limitaciones

La principal limitación en la medición usando un SMV radica en las restricciones naturales que impone una escena al ser proyectada en el

plano imagen. En pocas palabras, un SMV no puede medir lo que no se ve. Por lo tanto, la medición esta limitada a objetos "suaves", es decir, objetos que no contengan oquedades o esquinas pronunciadas.

Resumen de la tesis

En el capitulo 1, se hace una revisión de los conceptos de medición para poder entender la importancia de la metrología en la industria. Además se presenta una breve descripción de los sistemas de medición clásicos y de mayor exactitud en la actualidad. Por otro lado, se plantean los principales fundamentos de los SMV tanto para el procesamiento de las imágenes planas como para el despliegue de las mediciones 3D resultantes. En el capítulo 2, se desglosan paso a paso los conceptos teóricos y matemáticos utilizados en la creación del instrumento. En este capitulo se asientan los fundamentos teóricos para la visión con luz estructurada. Además se describen las etapas que implementan el proceso descrito en la figura 1. En el capítulo 3 se analiza el software desarrollado, lo cual incluye la descripción del equipo empleado, las instrucciones de conexión y operación, y la descripción de las funciones del programa en el ámbito del usuario y del programador. En el capítulo 4 se presentan los resultados obtenidos con el instrumento, las conclusiones y las posibles líneas de trabajo a futuro.

Capítulo 1

Conceptos básicos

El hombre desde que pudo hacer uso de los recursos naturales, ha tenido la necesidad de conocer su entorno, de darse una idea de su ambiente y de su posición en el mundo. Es por eso que se ha dado a la tarea de asignarle un valor a las cosas como: largo, ancho, profundidad, rapidez, peso, temperatura, posición, volumen, etc. Al principio estas mediciones eran comparadas con su propio cuerpo o herramientas de uso cotidiano, de esta forma se crearon unidades de medida como el pie, la cuarta, etc. Sin embargo, con el crecimiento de la población, se fueron integrando comunidades de gran tamaño por lo que se necesitó de la estandarización de las unidades y el desarrollo de instrumentos de medición cada vez más elaborados. Actualmente los sistemas de medidas más usados son el S.I. (Sistema Internacional de Medida) y el sistema inglés. De acuerdo con estas ideas, este capítulo introduce en los conceptos más generales que subyacen al desarrollo del proyecto. Se presta especial importancia a la metrología dimensional como un área de desarrollo tecnológico importante. Se describen los principales instrumentos para la medición geométrica y los recursos que actualmente ofrecen las computadoras para la representación de geometrías espaciales.

1.1. Metrología dimensional

Siempre hemos estado en contacto con la medición dimensional aunque no haya sido de manera científica, hemos sabido de nuestra altura o la de un edificio, el ancho y alto de un dibujo o una hoja, hemos tenido precaución de la profundidad de una alberca, o simplemente las distancias necesarias para llegar de un lugar a otro.

Como se puede ver, estamos acostumbrados la mayor parte del tiempo a conocer la medida en una dimensión de un objeto, en dos dimensiones, cuando queremos darnos cuenta de la superficie que abarca algo, sin embargo, la rama de la metrología dimensional, abarca también la tercera dimensión. Agregar una nueva dimensión a nuestra concepción de medida, nos permite conocer la geometría de objetos tangibles y poder situarlos en el espacio.

1.1.1. La razón de medir

La forma más común de conocer las características tridimensionales de un objeto es usando unidades cúbicas o litros. Sin embargo, es posible determinar las dimensiones de un objeto mediante la posición que ocupa su masa en el espacio, esto es posible mediante las coordenadas, teniendo como referencia un sistema de coordenadas preestablecido. En la vida diaria es probable que no nos interese saber que una esquina de una mesa este en la posición P(0,0,0) y que su esquina opuesta esté en P(0,200,50). Para aplicaciones industriales es muy útil conocer esta característica de los objetos, ya que se pueden representar fácilmente en un dibujo, escalar, e incluso reproducir, lo cual, de otra forma sería más complicado.

En la industria, es muy útil verificar en situaciones de control de calidad de una cadena de producción si las coordenadas de un producto corresponden con las de un modelo patrón, y con esto determinar si se cumplen los requisitos de dimensión y tolerancias necesarios para aceptar o rechazar la producción y con esto hacer las correcciones pertinentes si se requieren. Es el papel del ingeniero diseñar, aplicar y mantener los instrumentos y herramientas útiles para mantener una confiabilidad en las medidas y con esto una mejora en la calidad de los productos maquilados en la industria.

1.1.2. Calibraciones y aseguramiento de la calidad

Básicamente un sistema o instrumento de medición es aquel que nos permite asignar un valor numérico a una característica física de un objeto o evento a medir [18]. Por principio, el valor que despliega el instrumento no es el mismo que el valor verdadero, sino una cantidad muy cercana, es por esto que un instrumento cuenta con propiedades que deben conocerse previamente antes de hacer las mediciones, estas son, entre otras:

- Exactitud: es un indicador de la desviación de la lectura del instrumento respecto a una entrada conocida o patrón. Se da generalmente en porcentajes.

- Precisión: la cual es la propiedad de un instrumento para dar una lectura repetidamente con una cierta exactitud.

Para que un instrumento sea confiable, es necesario calibrarlo previamente, lo cual es hacer una comparación con un patrón establecido o mediante un instrumento de medición con más alta precisión que el instrumento a calibrar y lograr que el valor dado por el instrumento se acerque lo más posible al patrón o a lectura del instrumento más preciso. La calibración no debe pasarse por alto, ya que de esta forma podemos caracterizar el instrumento.

La exactitud de un instrumento puede mejorarse mediante la calibración, sin embargo solamente hasta un cierto límite, determinado por su precisión. La precisión de un instrumento está dada por diversos factores complejos y su cálculo requiere técnicas especiales, generalmente matemáticas, con las cuales es posible determinarla y posiblemente disminuir el error generado a la salida de las variables medidas. Estas técnicas son conocidas como minimización de errores.

1.2. Instrumentos de medición dimensional de alta exactitud

Los procesos de manufactura modernos requieren de mecanismos automáticos exactos y precisos de medida que puedan ir verificando los productos y cumplir así con la calidad necesaria [18]. A este tipo de control se le denomina *en línea,* porque se va haciendo mientras los productos se van elaborando. Hay situaciones en las que no es posible llevar a cabo estos controles, entonces se dice que se realiza una inspección *fuera de línea*, en la cual cada parte o probablemente una porción de la producción es tomada estadísticamente para ser medida y comprobar la calidad de toda la producción. Esta información es periódicamente comparada y se van acumulando variables estadísticas como promedio y desviaciones de toda la producción.

En los procesos de control de calidad de se manifiesta constantemente la necesidad de sistemas de evaluación más acertada de las características geométricas individuales o de subconjuntos de los productos industriales. Generalmente un control más estricto es necesario hacerlo fuera de la cadena de producción. En un principio la inspección era realizada por humanos, haciendo el proceso tardado y con una precisión variable debido a factores humanos como la experiencia, la capacidad visual y el cansancio. Debido a la necesidad de reducir costos y hacer más eficiente los métodos de inspección, se fueron automatizando estos procesos. Entre estos están los Sistemas de Adquisición de Puntos o SAP. Con los cuales se puede reconstruir el

objeto analizado. Existen dos tipos de SAP, los de por contacto y los de sin contacto. En los SAP por contacto el primer lugar lo ocupan las Máquinas de Medir Coordenadas o MMC, mientras que los SAP sin contacto están por un lado las tecnologías ópticas (interferómetros) y por otro lado la visión computarizada, que como más adelante veremos, se ayuda de un láser que entra en la tecnología óptica.

1.2.1. Máquinas de medir en tres ejes

Una Máquina de Medición de Coordenadas (MMC) tiene un palpador montado sobre un sistema de tres ejes perpendiculares y deslizables que permiten el libre movimiento y posicionamiento del palpador en cualquier posición x, y, z dentro del área de trabajo de la máquina. Existen diferentes arreglos de los ejes que brindan ciertas ventajas para usos determinados [6]. El costo de estas máquinas y su desempeño está dado por la precisión y resolución de los transductores de movimiento, constituidos de codificadores ópticos, situados en cada uno de los ejes del instrumento. La resolución que alcanzan estos instrumentos va de 1 a 0.01 μm y considerando que el volumen de trabajo puede ser de aproximadamente de 2 metros cúbicos, es posible darse cuenta lo cuidadoso que debe ser el diseño mecánico de los ejes para lograr estas características. Para lograr alta confiabilidad del sistema se utilizan baleros de "contacto de rodamiento" de alta precisión y baleros de aire hidrostático. Una MMC pequeña con una precisión modesta puede llegar a costar entre $10 000 y $20 000 dólares mientras que más grandes y más precisas alcanzan varios cientos de miles de dólares [6].

El palpador universal es un captador de posición que, mediante contacto, permite localizar puntos sobre cualquier tipo de superficie. Este palpador activa un sistema que permite detener los sensores de los ejes al encontrar una parte de la superficie a medir e inmediatamente se tienen la lectura de los tres transductores de movimiento en cada eje. En una MMC el palpador puede posicionarse en cualquier lugar dentro del volumen de trabajo manualmente mediante una palanca de control que activa motores eléctricos (servomotores) para desplazar el palpador en cada una de las tres direcciones, controlando también la velocidad, figura 1.1 Para esto es utilizado un programa de control computarizado inherente a la MMC.

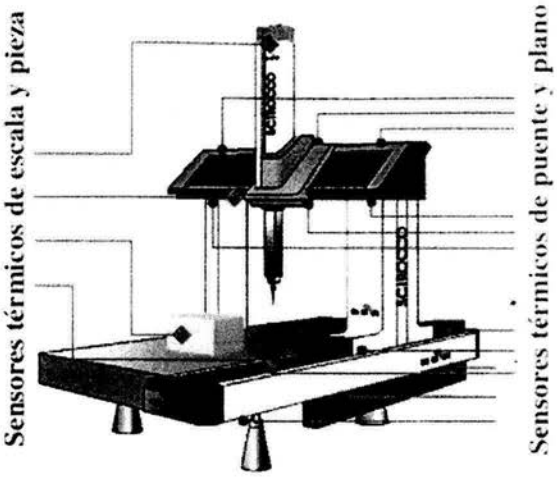


Figura 1.1. Máquina de Medir por Coordenadas, MMC.

Esta característica de tener un control numérico computarizado identifica a las MMC como una herramienta poderosa de medición cuando es combinada con un programa automatizado para hacer varios cálculos geométricos necesarios para extraer las características de las partes de un producto. Un uso importante y común de las MMC es el de verificar de acuerdo a ciertas especificaciones, la primer pieza salida de la producción y generalmente se deja correr un programa que mueve el palpador automáticamente, siguiendo una serie de puntos conocidos o deseados. Del resultado de esta medición es crucial para seguir con la producción o verificar en dónde está el problema.

La calibración de estas máquinas, como se mencionó antes, se realiza mediante un patrón de referencia de alta precisión. Para calibrar y cualificar se utiliza un calibre esférico de incertidumbre certificada, de rango superior al conjunto de la MMC. Se toman lecturas de varios puntos y se caracteriza así el instrumento. La precisión del instrumento se logra de dos maneras: una es utilizando piezas de muy alta calidad, lo cual incrementa su costo considerablemente; y la otra es obtener los datos numéricos y hacer mediante software las correcciones de errores pertinentes a las variables de salida. Este punto se tratará más adelante en los demás sistemas de medición.

Las máquinas de medir por coordenadas ofrecen una muy alta precisión, es por eso su gran utilidad en el control de calidad en la manufactura de productos, y a pesar de su costo ofrecen una solución práctica a la demanda de precisión requerida. Es necesario también

considerar que el mantenimiento de estos aparatos conlleva a un costo similar al del instrumento, debido a que se deben de mantener estrictos controles de temperatura y presión de los recintos donde estas máquinas operan, así como de los elementos y sustancias necesarios para su buen funcionamiento, como son los lubricantes con propiedades eléctricas especiales y resistentes a la corrosión para permitir el movimiento casi sin fricción de los baleros. Además de dispositivos neumáticos para evitar la vibración de las mesas de granito, en las cuales descansan los objetos medidos.

1.2.2. Sistemas láser de medición

Existen básicamente dos familias en los sistemas láser de medición: los indirectos (realizados mediante triangulación) y los directos (por tiempo de vuelo). En los primeros, la imagen del punto analizado desplazándose por el objeto es recogida por una o dos cámaras fijas, cuya separación se conoce con mucha precisión. La distancia a determinar está en función de la correlación que existe entre la imagen del láser medida sobre la toma obtenida en cada cámara. En los segundos, se basa en la medida directa del tiempo que tarda la luz en recorrer la distancia entre el emisor y el blanco y volver. Con éste sistema se obtienen precisiones de 5mm a algunos centímetros con alcances útiles de unos 100 metros [7].

En los sistemas directos, el equipo de medición realiza un barrido sobre el objeto (el cual puede ser manual o automático mediante una programación previa), emitiendo pulso láser siguiendo el patrón determinado. Éste pulso es reflejado al incidir sobre el objetivo. Al determinar el retardo entre la emisión del pulso y la recepción de la señal reflejada, la distancia al blanco es calculada. Además el sistema cuenta con codificadores los cuales determinan los ángulos horizontal y vertical de la proyección del pulso. Con éstos elementos más la distancia medida, se determinan las coordenadas espaciales de cada punto. El ángulo de apertura del haz emitido determina la resolución espacial de la exploración, figura 1.2.

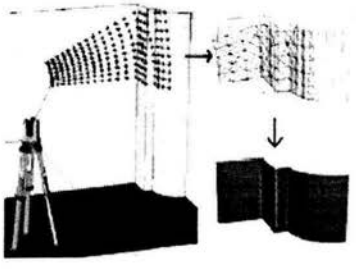


Figura 1.2. Proceso de exploración y procesamiento de datos obtenidos.

Es posible capturar cualquier objeto mediante exploración con láser. La ventaja que presenta la exploración con láser consiste en que, mediante un proceso automático, y sin pasos previos, obtenemos un modelo tridimensional de dimensiones reales, sobre el que podemos comenzar a trabajar de inmediato. El tratamiento posterior permite la obtención, a partir de una nube de puntos no estructurada, de las características geométricas que definen el modelo.

Con el sistema Callidus [28] (desarrollado en Alemania) por ejemplo, se obtienen levantamientos de más de un millón de puntos en aproximadamente 10 minutos, figura 1.3. Esto supone que, por ejemplo, el levantamiento completo de una estación de metro (130x30x15 metros) puede hacerse en menos de dos horas. El mismo trabajo para una estación robotizada con medición sin prisma, ocuparía cerca de un día completo. Se obtienen rendimientos de 120 metros/hora en túneles en condiciones normales de trabajo, siempre con precisiones milimétricas y típicamente con un solo operario.

Además la configuración mecánica permite trabajar en zonas de difícil acceso (pozos, silos), o con limitaciones de permanencia (presencia de radiaciones ionizantes o ambientes tóxicos), donde hacer el levantamiento por métodos clásicos puede resultar inviable. Por otro lado, el procesamiento posterior puede ser muy reducido, según el trabajo a desarrollar. Por ejemplo, la reconstrucción de una planta de oficinas de unos 400 m^2 puede suponer sólo una hora de trabajo.

Figura 1.3. Sistema de medición Callidus.

El LE-100 es un sistema de medida óptico que, en una clara línea de visión por medio de un rayo láser rojo visible (long. de onda 670nm), mide la distancia entre él mismo y un reflector en movimiento, figura 1.4. Calculando la fase del eje entre el rayo láser transmitido y el reflejado el LE – 100 mide, al milímetro, la distancia con precisión. El

principio de medida de fase de eje permite, con unos pocos milisegundos, el cálculo de valores medidos con muy alta reproducibilidad [27].

El LE-100 fue concebido y diseñado para uso industrial. Con un corto tiempo de integración, salida de valores de medición fiables permiten posicionar objetos, movimiento en dirección lineal, a velocidades de hasta 5 m/s. Los valores de posición medidos están disponibles para otros procesamientos por medio de interfaces usuales. Tiene una exactitud de ±2mm. El cálculo de nuevos valores de medida sucede en el menor tiempo posible. Esto es un pre-requisito para proporcionar cambios de posición rápidos en lazos de control automático. Grandes tiempos de integración producen errores de sistema requiriendo gastos de software adicional. El aparato de medición óptica de distancias proporciona un valor de medida absoluto el cual evita ir a una posición de inicio en cada pérdida de alimentación. Esto ahorra tiempo y costos.

Figura 1.4. Sistema de medición Stabila LE-100.

Otra forma de obtener medidas dimensionales mediante dispositivos ópticos es con un arreglo lineal de fotodiodos. El objeto a medir es iluminado mediante una luz trasera para producir un patrón de luz claro-oscuro con transición en los bordes del objeto. Mediante óptica tradicional se puede conocer la trayectoria de los rayos de la fuente a los fotodiodos. La resolución dimensional del arreglo está limitada por la distancia entre los fotodiodos y el número de éstos en el arreglo, en el ejemplo están separados 0.001 in. Sin embargo, de acuerdo al arreglo empleado, si un objeto de 5 in, se detecta en un área de los fotodiodos de 1 in, la resolución será entonces de 0.005 in. Además, si el objeto presenta alguna oscilación o movimiento, se pueden tener mayores errores en la medición [7, 12].

1.2.3. Sistemas de medición por visión

Las oportunidades para la introducción de la visión artificial se encuentran principalmente en tareas de inspección y ensamblaje. Se ha estimado que, en tareas repetitivas, las personas son solamente efectivas entre un 70 y 85%. En un estudio llevado a cabo por la Universidad de Iowa, se le pidió a un grupo de personas que separaran una minoría de pelotas negras de ping pong, en una línea de ensamblaje, en donde la mayoría de las pelotas era de color blanco. Se obtuvo que un 15% de las pelotas negras se dejaron escapar. Aún dos de los operadores más expertos lograron un desempeño del 95% [22].

Las personas tienen un periodo limitado de atención, lo cual las hace susceptibles de distraerse. Además, presentan ciertas inconsistencias en su sensitividad visual en el transcurso del día y de un día a otro. Sin embargo, las personas presentan muchas ventajas respecto a la visión artificial. Las personas son flexibles y pueden ser entrenadas para realizar muchas tareas. Asimismo pueden hacer ajustes para compensar ciertas condiciones que deben ser ignoradas (tonos de color, reflejos, ciertos cambios de posición, etc.).

Según el tipo de aplicación, serán el tipo de imagen que será necesario adquirir (imágenes de rayos X, infrarrojo, etc.) y el análisis que se aplicará. La mayoría de las aplicaciones de la visión por máquina podemos clasificarlas por el tipo de tarea en inspección (medición, calibración, detección de fallas), verificación, reconocimiento, identificación y análisis de localización (posición, guía).

La medición o calibración se refiere a la correlación cuantitativa con los datos del diseño, asegurando que las mediciones cumplan con las especificaciones del diseño. Por ejemplo, el verificar que un cable tenga el espesor recomendado.

La detección de fallas es un análisis cualitativo que involucra la detección de defectos o artefactos no deseados, con forma desconocida en una posición desconocida. Por ejemplo, encontrar defectos en la pintura de un auto nuevo, o agujeros en hojas de papel.

La verificación es el chequeo cualitativo de que una operación de ensamblaje ha sido llevada a cabo correctamente. Por ejemplo, que no falte ninguna tecla en un teclado, o que no falten componentes en un circuito impreso.

El reconocimiento involucra la identificación de un objeto con base en descriptores asociados con el objeto. Por ejemplo, la clasificación de cítricos (limones, naranjas, mandarinas, etc.) por color y tamaño.

Otro ejemplo de reconocimiento podría ser aplicado a células, por área y forma.

Identificación es el proceso de reconocer y clasificar un objeto por el uso de símbolos en el mismo. Por ejemplo, el código de barras, o códigos de perforaciones empleados para distinguir hule espuma de asientos automotrices.

El análisis de localización es la evaluación de la posición de un objeto. Por ejemplo, determinar la posición donde debe insertarse un circuito integrado, "chip". En la figura 1.5 puede verse la localización automática del contorno de un cerebro en una imagen de tomografía. Otro ejemplo sería la localización automática del contorno de la cabeza en una serie de imágenes de tomografía y su posterior reconstrucción tridimensional, figura 1.6.

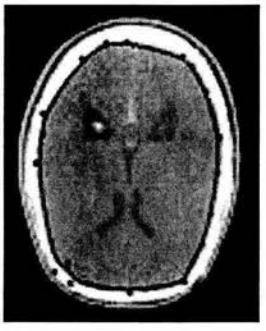


Figura 1.5. Localización automática de contornos.

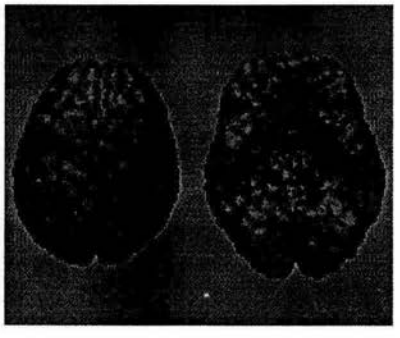


Figura 1.6. Reconstrucción tridimensional.

La tarea de guía se refiere a proporcionar adaptativamente información posicional de retroalimentación para dirigir una actividad. El ejemplo típico es el uso de un sistema de visión para guiar un brazo robótico mientras suelda o manipula partes. Otro ejemplo sería la navegación en vehículos autónomos.

A diferencia de los sistemas de medición por contacto e incluso los sistemas láser la ventaja de los sistemas de medición por visión es que permite capturar varios puntos de interés en décimas de segundo, lo cual lo hace muy competitivo ante los sistemas mecánicos o láser. Algunos sistemas cuentan también con sistemas ópticos para poder hacer enfoques de la pieza de interés y acercamientos, para piezas más pequeñas.

1.3. Principios de visión por computadora

Al final de los años setenta, se empezaron a hacer pruebas con sistemas de visión artificial en los laboratorios de automatización y este campo tuvo un rápido desarrollo. La visión por computadora trata de simular una parte de las habilidades del ojo y cerebro humanos para formar e interpretar imágenes.

1.3.1. Conceptos básicos

Prácticamente un sistema de visión por computadora requiere de que al menos tres funciones estén presentes: formación de imágenes, análisis, e interpretación de imágenes [9].

La *formación de imágenes* se refiere a la combinación de un cierto tipo de iluminación y una cámara o sensor. Existen tres tipos de iluminación de acuerdo a la posición de la fuente con respecto al sistema de captura:

- La iluminación trasera, provee una silueta de la imagen de contraste máximo, lo cual es útil para mediciones dimensionales y detección de la presencia o ausencia de formas.
- Iluminación frontal, puede ser necesaria para lidiar con formas como letras y códigos de barra.
- Iluminación lateral, es utilizada para detección de características tridimensionales.

Para este último tipo de iluminación se utiliza luz polarizada o luz estructurada que proyecta un patrón claro/oscuro geométricamente definido sobre el objeto medido para resaltar ciertas características de éste. El tipo más común de luz estructurada es un simple rayo plano u "hoja" de luz láser o haz de línea. Al proyectar esta luz láser sobre el objeto produce una línea de luz reflejada, la cual sigue el contorno del objeto y esta imagen puede ser procesada para obtener mediciones útiles.

En cuanto a los dispositivos de adquisición de imágenes en un principio se utilizaron cámaras de video para convertir la luz en

señales eléctricas y después en un arreglo bidimensional de la imagen del objeto tridimensional. Más recientemente se usan cámaras con fotosensores de estado sólido usando dispositivos de carga acoplada (CCD) o de carga inyectada (CID). Cada sensor del arreglo provee un elemento individual de la imagen ("píxel" – picture element).

1.3.2. Fundamentos de la imagen digital

El *análisis de la imagen* comienza con el preprocesamiento de ésta, el cual consiste en la captura de la información recibida por los sensores de la cámara mediante una tarjeta de captura, la cual permite almacenar en memoria las imágenes. Se le asigna un valor digital a cada pixel de acuerdo a la consideración que se tome, ya que puede ser binaria (blanco y negro) o una escala de grises. Si se hace de forma binaria se compara cada píxel con un cierto umbral de iluminación y se le asigna un valor binario de 1 ó 0, dependiendo si la señal es mayor o menor que el valor del umbral. De esta forma la imagen es simplificada. Los sistemas de escala de grises proveen de una discriminación más fina de la intensidad de la luz., típicamente de 64 a 256 sombras de gris entre negro y blanco (el ojo humano puede diferenciar alrededor de 64). El inconveniente de usar imágenes con escala de grises radica en su tamaño de almacenamiento en memoria, es por esto que se deben de considerar las características de hardware con que se cuenta.

Algunas veces solamente se requiere una porción de la imagen para identificar el objeto de interés, es por eso que a veces se utilizan ventanas para reducir la información y hacer más eficiente el uso de la memoria. Cuando una imagen cuenta con varios tipos de distorsión o degradación (si está borrosa o desenfocada, bajo contraste, ruido, etc.), se pueden utilizar varios algoritmos para mejorar la calidad de ésta. Una vez que se ha hecho el pre-procesamiento, el análisis de la imagen comienza con detectar las características útiles del objeto analizado como puede ser el reconocimiento de patrones como la extracción de bordes o la segmentación del centro de una línea o su esqueletización.

1.3.3. Representación y descripción

La siguiente etapa, la de *interpretación de la imagen* tiene que ver con la manipulación de los datos que arrojan las imágenes. Para el caso de la metrología dimensional, podemos conocer la presencia de un patrón circular, por ejemplo, y poder medir su radio. En otro caso, podemos reconstruir las dimensiones tridimensionales de un objeto mediante transformaciones 2D → 3D, conociendo previamente las constantes de calibración de la cámara y del plano láser, si es que se usó luz estructurada, o mediante funciones de correlación cruzada

entre dos imágenes tomadas en diferentes posiciones con respecto al objeto estudiado.

Los sistemas de medición por visión por computadora tienen la ventaja de que su costo es inferior al de los dos sistemas antes mencionados, además, las correcciones de exactitud pueden hacerse mediante software.

1.4. Principios de Diseño Gráfico Asistido por Computadora

A inicios de la década de 1950, las computadoras ya permitían el diseño y maquinado de formas 3D que típicamente se realizaban en modelos de arcilla, madera y metales. Estas formas entonces podían usarse en la construcción de productos, principalmente para la industria automotriz. El cuello de botella en este método de producción fue rápidamente descubierto en la falta de software adecuado. En el sentido de maquinar una forma utilizando una computadora, se hizo indispensable la representación de la forma en una manera compatible con la computadora. El método más promisorio para la representación de geometrías 3D fue identificada en términos de curvas y superficies paramétricas.

La teoría de las superficies paramétricas fue entendida en la geometría. Su potencial para la representación de superficies en un ambiente de Diseño Asistido por Computadora no fue bien conocido del todo. La exploración del uso de curvas y superficies paramétricas puede ser visto como el origen del Diseño Gráfico Asistido por Computadora [8, 23].

1.4.1. Geometría tridimensional

El dibujo esquemático asistido por computadora involucra el uso de la computadora para asistir en la producción de diagramas esquemáticos. Se utilizan una serie de líneas y arcos, al igual que en el dibujo a mano, con la diferencia de que son congregados en símbolos y conexiones. El usuario construye el diagrama esencialmente colocando los símbolos en posición dentro del espacio bidimensional de dibujo, y después uniéndolos mediante conectores y líneas. Con los programas de diseño es posible agrupar una serie de objetos dentro de entidades mayores conocidas como formas o modelos, plantillas y símbolos. También es posible dibujar varias líneas conectadas conocidas como polilíneas [38].

Durante muchos años la forma tradicional de representar los objetos para su diseño y posterior manufactura fue utilizando proyecciones

isométricas y/u ortogonales. Sin embargo, el uso de estos métodos requerían de casi 100 000 dibujos y otros documentos para representar algo tan complejo como un aeroplano, traduciéndose esto en días o incluso meses para el diseño de los esquemas y otro tanto para la manufactura de las partes. Debido a esto desde hace más de tres décadas se han desarrollado varios métodos para la representación de la geometría de un objeto usando esquemas que no tengan su proyección en un espacio plano. Estos esquemas involucran la construcción de una sola representación de los componentes geométricos en el espacio tridimensional. De esta forma es posible tener aplicaciones con las cuales se examine el modelo y sea posible extraer información para su análisis y manufactura. El uso de una sola representación elimina el error potencial inherente al uso de múltiples vistas.

Los métodos desarrollados para el diseño en tercera dimensión involucran la representación geométrica como un conjunto de líneas, curvas, superficies, e incluso de sólidos en el espacio. Los modelos en tres dimensiones (3D) son construidos en un espacio 3D, típicamente usando un sistema de coordenadas cartesianas regidos por la ley de la mano derecha. Existe un sistema de coordenadas fijo el cual es usado para la definición total del modelo, llamado sistema de coordenadas globales (SCG), además existe el sistema de coordenadas de trabajo (SCT) usado para asistir en la construcción del modelo.

Las entidades geométricas son instancias de formas geométricas conocidas como primitivas, para las cuales su dimensión, ubicación y orientación se modifica para darle un valor característico a cada entidad. Por ejemplo, una entidad primitiva puede ser un arco de un círculo, para el cual se tendrán que definir el radio, un ángulo de inicio y fin, así como la orientación espacial para ese caso particular.

1.4.2. Representación de objetos

Una forma de representar un modelo en tercera dimensión es mediante la llamada geometría de líneas (wire frame), figura 1.7, en la cual se utilizan líneas y curvas para definir las los contornos del modelo. Es empleada básicamente para hacer bosquejos del modelo. Tiene la ventaja de ser la más rápida representación del objeto en 3D, además de ser más económica en cuanto tiempo de procesamiento computacional y espacio en memoria, lo cual lo hace útil en aplicaciones que solamente requieran la visualización del movimiento de formas simples. Sin embargo, presenta serias deficiencias para aplicaciones más complejas ya que presenta ambigüedad en la representación en ciertas posiciones, formando posiblemente objetos

sin sentido, ya que se pueden encimar líneas y ocultar información importante sobre todo de la profundidad [8].

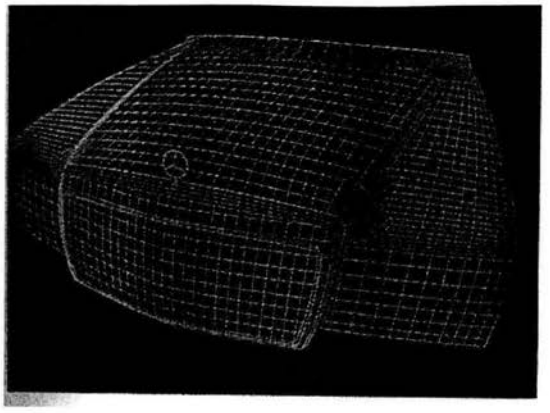


Figura 1.7. Wire frame.

Con el objeto de eliminar las ambigüedades de los modelos (wire-frame) se creó un segundo esquema de modelado 3D: el de superficies. En este esquema se modelan algunas o todas las superficies del objeto. De nuevo, la representación generalmente incluye el uso de entidades geométricas primitivas para formar una nueva entidad, esto es, la superficie. Esta se construye con los límites o bordes y curvas dentro de la superficie, y por lo tanto la representación de superficie se mezcla o se genera con representaciones (wire-frame). La superficie elemental es el plano, el cual se define por dos líneas rectas paralelas, por medio de tres puntos o con una línea y un punto. Otro tipo de superficies utilizadas en aplicaciones DAC (Diseño Asistido por Computadora) son:

- Cilindros tabulados: el cual se define por la proyección de una curva a lo largo de una línea o vector.
- Superficies guíadas(ruled): son aquellas producidas por interpolación lineal entre dos diferentes curvas generadoras o curvas límites. El efecto es de una superficie generada al mover en dirección perpendicular una línea recta con sus puntos finales terminados en la línea curva generadora.
- Superficie de revolución: es generada al hacer girar una curva alrededor de una línea central o vector. Esta superficie es útil al modelar partes torneadas, o aquellas en las que se necesite una simetría axial.
- Superficies de barrido(swept surface): son una extensión de las superficies de revolución, en las cuales una curva se barre o se

mueve a lo largo de una curva arbitraria en vez de un arco circular.

- Superficies escultóricas o curvas de malla: son las más comunes en CAD y se definen usando familias de curvas generadoras, o dos familias que se intersectan formando cruces o una red de parches interconectados.
- Superficie fillet (faja, banda, cinta): La cual se define como una superficie conectando otras dos en una transición suave (generalmente con cierto ángulo de curvatura constante o con cambios graduales).

Para todos los casos debe hacerse hincapié que las superficies son dibujadas como una malla de curvas que se intersectan en la superficie. Sin embargo, esto es solamente para cuestiones de visualización, ya que las superficies son continuas, con cada punto sobre la superficie definido por una relación matemática usada en su definición.

Generalmente para representar una pieza más real, se usa una geometría de superficie en forma de parches, con curvas no necesariamente paralelas para lo cual se necesita utilizar una gran cantidad de estos parches para dar una apariencia suave a la superficie del artefacto. En otras ocasiones es preferente utilizar parches de tres lados para mallas irregulares.

1.4.3. Realismo en la representación de objetos

Para poder representar un objeto en la computadora que parezca a la vista como real, es necesario considerar algunas características del objeto estudiado como son que los detalles internos y la cara posterior desaparecerán de la vista, las sombras que provocará, y que su superficie tendrá diferentes intensidades de luz y de acuerdo a la posición de la fuente.

La generación de imágenes realistas ha sido utilizada en diversos campos que van desde el entretenimiento hasta aplicaciones científicas, pasando por el diseño, la simulación, la publicidad y la educación. El desarrollo que se ha tenido en esta área se ha visto intensificado por los avances en los dispositivos gráficos, como son las tarjetas aceleradoras de video, el incremento en la velocidad de los microprocesadores y además de la creación de software especializado para lograr efectos visuales sorprendentes, figura 1.8.

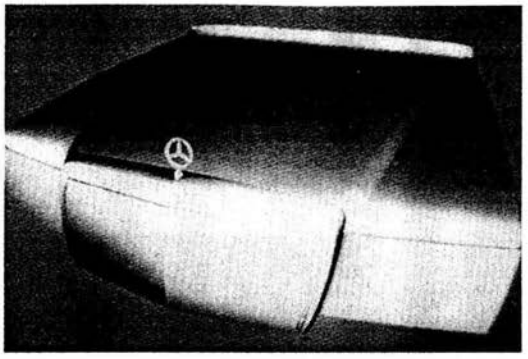


Figura 1.8. Representación realista de primitivas 3D.

La dificultad de presentar una imagen realista radica en la complejidad del mundo, ya que si nos ponemos observar una escena cotidiana nos encontraremos que cada objeto tiene una cierta textura, una iluminación determinada y una posición con respecto a cada uno de los demás objetos lo cual lleva a tener que realizar un gran trabajo para poder incluir todas las características de cada objeto en un ambiente artificial dentro de la computadora.

1.4.4. Software para la graficación por computadora

En la actualidad existen diversos programas para la graficación por computadora, los cuales presentan diferentes características entre las que destacan: la robustez del sistema, la dependencia con el hardware, la velocidad de procesamiento, y la complejidad para la representación de los objetos, entre otras. Se decidió utilizar OpenGL, debido principalmente que es una librería de funciones que permiten interactuar con el hardware de graficación de una manera sencilla y es compatible con la mayoría de lastarjetas de video. Esta interfaz consiste en cerca de 250 comandos diferentes con los cuales se pueden crear objetos y es posible manipularlos dentro de un ambiente tridimensional, realizando las operaciones necesarias.

OpenGL está diseñado para que se pueda comunicar con cualquier hardware de graficación mediante ráfagas de información, soporta todos los eventos y características de las ventanas de Windows. OpenGL no provee herramientas para representar figuras complicadas completas, sino que se basa en la idea de que cada objeto está constituido por objetos primitivos o básicos, como pueden ser las líneas, los puntos y los polígonos [37].

OpenGL cuenta además con las herramientas para poder representar objetos con un gran realismo, al utilizar funciones para poder determinar la profundidad de la escena, lo cual puede lograr mediante

la aparición de niebla en las zonas más alejadas; la suavidad de las uniones de las caras de los objetos para evitar cortes abruptos entre los parches. La presencia de luz para poder distinguir texturas y pliegues de los objetos, así como las sombras que proyectan los objetos iluminados por alguna fuente de luz. También permite la opción de poder utilizar mapas de bits para poder asignarle una textura a un objeto en origen liso.

Capítulo 2

Descripción del sistema para la medición 3D por visión

En el presente capítulo reportamos una técnica para la medición dimensional 3D por visión. En particular, empleamos un sistema de medición sobre la base de la generación de un patrón con luz estructurada. El patrón generado resalta las características de interés en el objeto a medir y facilita el procesamiento de la imagen para la reconstrucción tridimensional. La técnica desarrollada para la medición, utiliza de manera importante herramientas hardware y software desarrolladas en los últimos años en el Laboratorio de Metrología del CCADET UNAM.

2.1. Introducción

Los sistemas de calibración y medición han tenido un avance significativo en las últimas décadas al incluir etapas de control para la automatización de la medición. Otra forma de lograr la automatización es mediante la utilización de sistemas de visión por computadora, lo que permite al usuario obtener múltiples características del objeto a medir con una sola exposición o imagen. Con el desarrollo del hardware computacional y la creación de algoritmos de optimización, es posible lograr mediciones en tiempo real con una gran exactitud y a bajo costo en tiempo de cómputo. Al utilizar un sistema de visión con luz estructurada es posible reconstruir las características de un objeto tridimensional a partir de las imágenes en dos dimensiones captadas por el elemento sensor, en este caso, la cámara CCD y un elemento de

iluminación como un láser de línea, él cual hace sobresalir ciertas características del objeto para su posterior reconstrucción [11].

Para este trabajo, el sistema de visión utiliza un sistema de luz estructurada, el objeto es posicionado en el centro de una mesa giratoria lo que permite manipular el movimiento para reconstruir el objeto en toda su geometría. El sistema consta de tres etapas principales hardware y software: Procesamiento de imágenes, algoritmo de medición y algoritmo de control. En la figura 2.1 se muestra el esquema completo, el cual se detallará en las siguientes secciones.

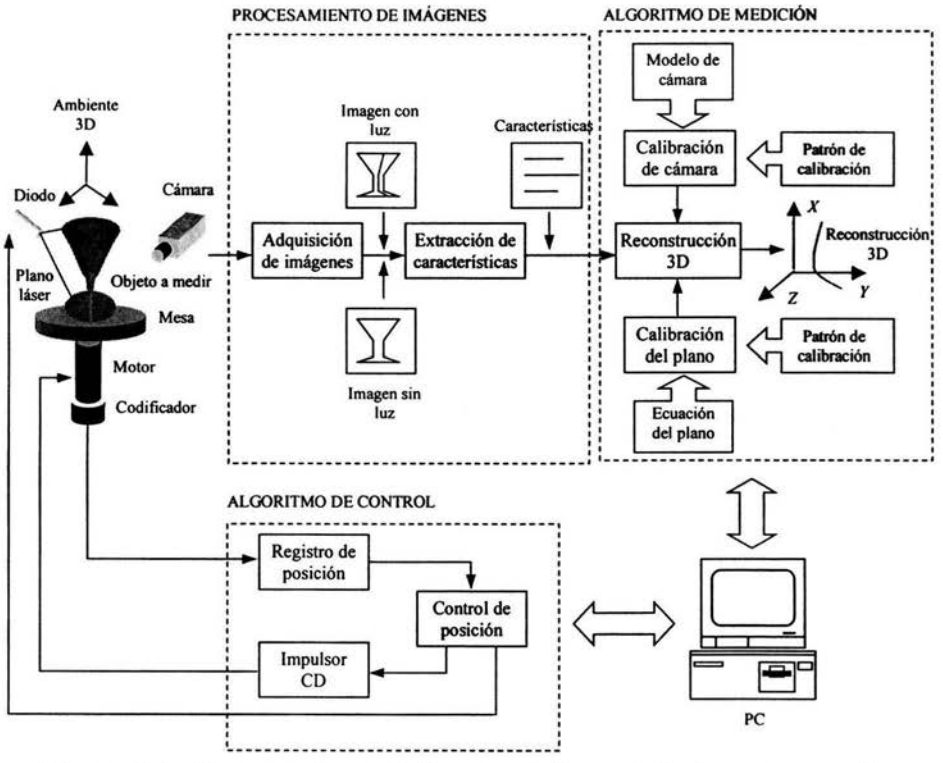


Figura 2.1. Esquema de medición por visión usando luz estructurada.

En la figura 2.1 un objeto a medir es colocado en el ambiente global 3D. Un diodo láser de línea genera el patrón de luz estructurada que intersecta la superficie, resaltando algunas características de interés. El patrón de luz genera un plano 3D con perfil aproximadamente gaussiano.

Como parte de la etapa de procesamiento de imágenes, se capturan dos imágenes que se forman en el ambiente 3D para obtener una

proyección plana, de la cual se debe rescatar su profundidad. El par de imágenes, con luz y sin luz, se procesa con el objeto de extraer las características de interés. En nuestro caso, las características están conformadas por la columna vertebral de la imagen de franja que forma el plano láser con la superficie bajo inspección. Las características constituyen posiciones planas en forma de pares de puntos y son tratadas como un arreglo de dos dimensiones.

En la etapa del algoritmo de medición las características 2D son reconstruidas en un ambiente 3D mediante la calibración del arreglo físico. La calibración de cámara es el método que sintetiza la transformación de la escena 3D a coordenadas 2D al encontrar los parámetros extrínsecos e intrínsecos de tal relación [29]. La calibración de cámara toma como referencia el modelo de cámara y posiciones patrón especificadas con anticipación en el ambiente 3D mediante MMC. Por otra parte, la calibración del plano consiste en ajustar a su representación matemática el plano láser que se genera físicamente. Para ello, los parámetros de la ecuación son encontrados mediante el conocimiento de tres posiciones sobre el plano que constituyen un patrón de calibración. Con los parámetros que se encuentran como parte de los procesos de calibración de cámara y del plano, es posible reconstruir tridimensionalmente las características planas [13]. La reconstrucción 3D de las características constituye una medición parcial a la cual llamaremos *meridiano*.

El algoritmo de control consiste en posicionar angularmente el objeto bajo inspección que es colocado sobre la mesa divisora. Para tal propósito, se registra la posición de la mesa mediante un codificador óptico. El control PID de lazo cerrado se completa al impulsar el motor CD utilizando señales PWM con nivel variable de corriente directa [26, 34]. La rotación de la mesa divisora nos habilita para reconstruir el objeto a medir en toda su geometría, de manera que zonas que están naturalmente ocultas a la cámara puedan ser expuestas. El diodo láser es conmutado con el propósito de obtener las imágenes necesarias para el algoritmo de extracción de características, imágenes con luz y sin luz.

La etapa de adquisición de imágenes y los algoritmos de medición y control son gobernados por el software de desarrollo específico en la PC. Los tres algoritmos deben estar sincronizados de manera que una medición sea el conjunto de meridianos que se obtienen al rastrear el objeto en los 360° de la mesa divisora. El rastreo se lleva a cabo de forma discreta y cada meridiano es compensado con la posición angular registrada.

2.2. Principios de medición por visión usando luz estructurada

Un sistema de luz estructurada consiste en la utilización de patrones de luz geométricamente conocidos que iluminen la escena [1]. El proyector de luz y la cámara están separados por una cierta distancia y deben estar situados en planos no paralelos, por lo tanto, debe existir un cierto ángulo θ entre ellos, figura 2.2.

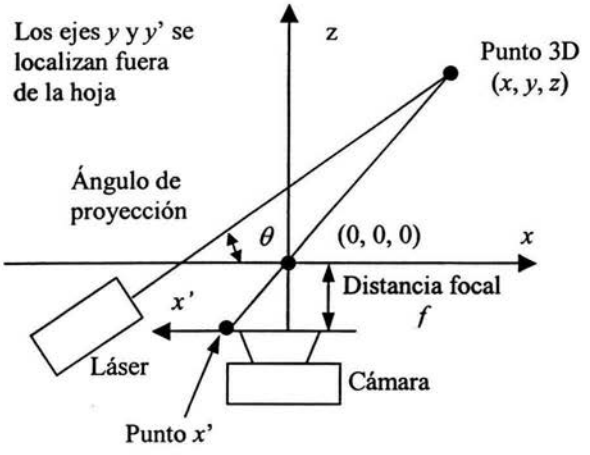


Figura 2.2. Esquema de visión mediante luz estructurada.

Las coordenadas del objeto (x, y, z) están relacionados con las coordenadas de la imagen (x', y') y la proyección del ángulo θ mediante

$$[x\ y\ z]=\frac{b}{f\cot\theta - x`}[x`\ y`\ f]$$

La resolución de este sistema de triangulación lo determina la eficiencia con la cual el ángulo θ y la posición horizontal del punto x' en la imagen, puedan ser medidos. La profundidad se va a obtener al conocer la posición de varios puntos tanto en el ambiente global, como en la imagen. Un haz de línea permite generar varios puntos en una sola iluminación, y es posible determinar la posición de cada uno de los puntos, si se conoce previamente la ecuación del plano que rige el haz de línea.

El patrón de luz que es proyectado sobre el objeto en el ambiente global es capturado por la cámara, figura 2.3. La imagen obtenida contiene ciertas distorsiones (comparadas con el patrón de luz)

determinadas por la forma y la orientación de las superficies del objeto en el que el haz fue proyectado. Cualquier punto del objeto 3-D visto en la imagen plana corresponde a la intersección de la línea de vista de la cámara con el plano de luz, por lo que es posible reconstruirlo. Se van a presentar discontinuidades en la franja de luz cuando la superficie del objeto presente partes ocultas a la cámara y que un poco más atrás, el láser, vuelva a encontrarse con la superficie del objeto.

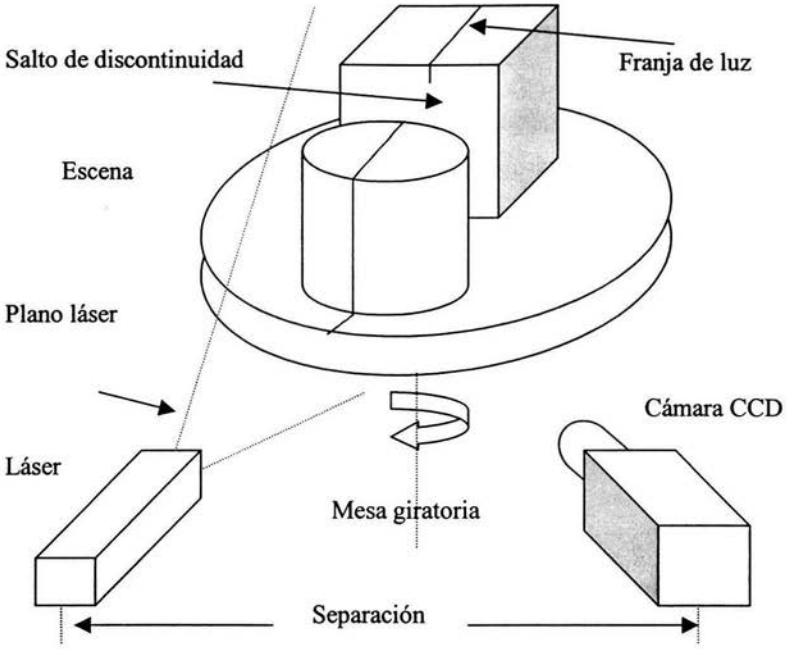


Figura 2.3. Esquema del instrumento de medición.

Las técnicas de luz estructurada se han usado ampliamente en aplicaciones industriales de visión en los cuales es posible controlar la iluminación de la escena. Comúnmente se utiliza un patrón de línea apuntando a una posición fija al igual que la cámara y el objeto se desplaza linealmente para capturar así la forma del objeto; el movimiento se hace en intervalos regulares que correspondan con la captura de las imágenes y el procesamiento de los datos. Otra forma de hacerlo es girar el objeto sobre una plataforma y reconstruirlo de esta forma.

Un punto importante que hay que considerar al utilizar un sistema de luz estructurada es que no es posible obtener información de puntos que no estén visibles para la fuente de luz, ni aquellos que no los pueda captar la cámara.

2.3. Procesamiento de imágenes

La etapa de procesamiento de imágenes incluye la adquisición de las mismas y la extracción de las características del objeto resaltado por el patrón de luz estructurada. Para ello, se utiliza hardware comercial y algoritmos desarrollados e implementados específicamente para el problema descrito en el presente proyecto de tesis.

2.3.1. Adquisición de imágenes

El esquema planteado en la figura 2.1 es implementado en tiempo real mediante los siguientes dispositivos.

Tarjeta para la digitalización de video DT3155 [5]. Es una tarjeta monocromática para la captura de video analógico, NTSC o PAL, mediante una computadora personal y software para el desarrollo de aplicaciones, figura 2.4. Esta tarjeta tiene un desempeño de nivel científico en aplicaciones que requieren alta exactitud y bajo ruido.

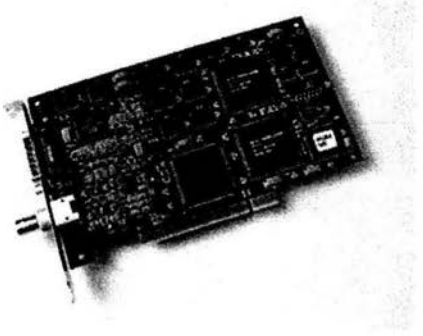


Figura 2.4. Tarjeta para la digitalización de imágenes DT3155.

Diodo láser de luz estructurada [25]. Es un diodo de emisión láser con alto desempeño que genera patrones de luz uniforme, no segmentadas y estrechas, figura 2.5. El haz puede ser enfocado de manera que se cubre un amplio rango de aplicaciones de visión por computadora.

Figura 2.5. Diodo láser de luz estructurada.

Una cámara CCD. Es una cámara analógica de video NTSC con resolución de 500 líneas de TV y elemento transductor CCD de ½ pulgada, figura 2.6. La cámara es complementada con un lente de 16mm.

Figura 2.6. Cámara CCD.

La plataforma hardware es complementada con bases para soportar la cámara y el diodo láser así como una computadora personal con Visual C++ 6 y los controladores de la DT3155 [4].

2.3.2. Extracción de características

En una primera etapa, se realiza la simplificación de la imagen que forma la franja del láser en la escena. Esto se logra mediante la simple resta de dos imágenes, una en la que aparece el objeto sin el patrón de luz estructurada y otra en la que se proyecta el patrón de luz. Este procedimiento se automatiza mediante un dispositivo de conmutación que enciende y apaga el proyector de luz, así como la captura de ambas imágenes [32, 33]. La resta de imágenes resulta de la siguiente expresión

$$I_C(u, v) = I_A(u, v) - I_B(u, v)$$

Una vez que la resta se ha realizado, la imagen se discrimina mediante un umbral para tener en la imagen solamente valores de píxeles superiores a ese umbral *U.*, mediante la siguiente expresión.

$$I(u, v) = I_C(\mathrm{u,v}) \qquad > \mathrm{U}$$

$$I(u, v) = 0 \qquad \text{otro caso}$$

La figura 2.7 ilustra el proceso de simplificación.

(a) (b)

(c)

Figura 2.7. (a) Imagen del objeto. (b) Imagen del objeto iluminado con el patrón de luz estructurada. (c) Resta de las imágenes y umbralización.

La imagen $I(u, v)$ de la figura 2.7.c es una franja clara sobre fondo oscuro de la cual se debe extraer su columna vertebral [10]. Una vez que se ha obtenido la imagen $I(u, v)$, la imagen de franja se procesa al establecer previamente las condiciones para un centro de la línea $\mathbf{p}_i(p_{xi}, p_{yi})$ con un cierto ancho w_i, y la dirección de la línea $\mathbf{d}_i(d_{xi}, d_{yi})$. El centro inicial de la franja se obtiene mediante una búsqueda exhaustiva empezando por la esquina superior izquierda continuando hasta encontrar el primer centro que cumpla con cierta condición establecida de intensidad. La información extraída del primer centro pixel conforma el elemento de línea $\mathbf{e}_i(p_i, w_i, d_i)$ como se muestra en la figura 2.8.

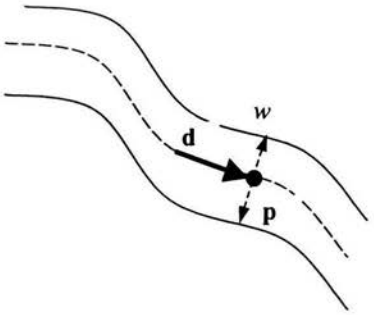


Figura 2.8. Elemento de línea.

Posteriormente, una vez que se ha encontrado el primer elemento de línea $\mathbf{p}_i$, se prosigue a encontrar los restantes elementos utilizando un patrón circular $t(\mathbf{p}, r)$ con centro en $\mathbf{p}_i$ y radio $r_i = 0.6\ w_i$. El conjunto de elementos de línea conforma el esqueleto de la imagen de franja. El patrón se usa como una trayectoria de referencia para encontrar los bordes de la línea $\mathbf{b}_1$ y $\mathbf{b}_2$, como se muestra en la figura 2.9.

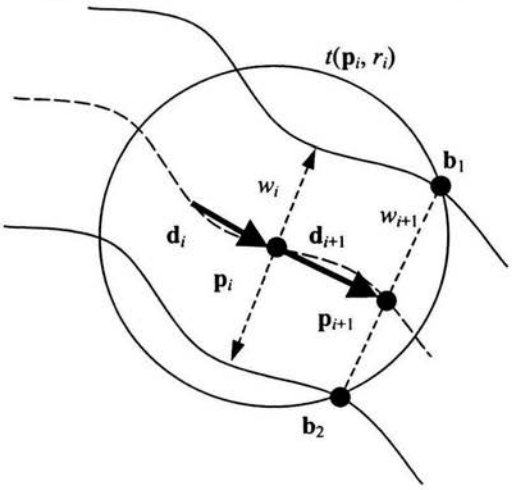


Figura 2.9. Rastreo de los elementos de línea.

La posición del siguiente elemento de línea, $\mathbf{p}_{i+1}$, está dado por

$$\mathbf{p}_{i+1} = \frac{\mathbf{b}_1 + \mathbf{b}_2}{2}$$

el vector de dirección siguiente, $\mathbf{d}_{i+1}$, es

$$\mathbf{d}_{i+1} = \mathbf{p}_{i+1} - \mathbf{p}_i$$

y el ancho del siguiente elemento, w_i, de línea es

$$w_i = \|\mathbf{b}_1 - \mathbf{b}_2\|$$

Los puntos borde $\mathbf{b}_1$ y $\mathbf{b}_2$ se detectan al analizar los niveles de intensidad en escala de grises del patrón circular t. El patrón circular es una serie de $[2\pi\ r]$ muestras tomadas alrededor del perímetro de la circunferencia con centro $\mathbf{p}_i$ y radio r, en otras palabras, $t(\mathbf{p}, r) = (c_0, c_1, \ldots, c_{[2\pi\ r]})$, donde

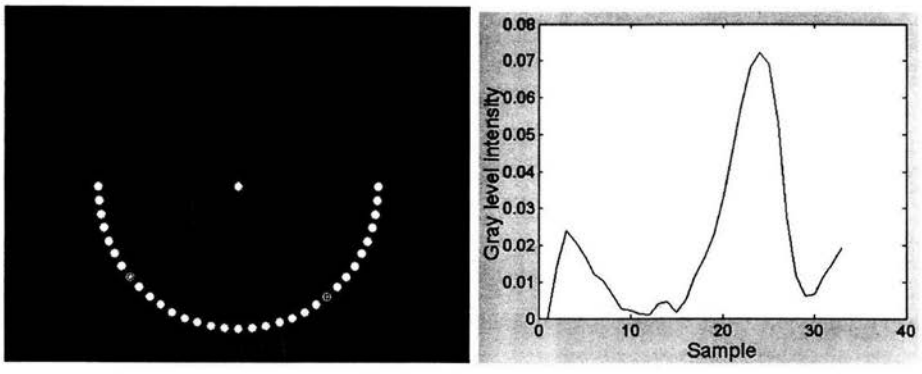
$$c_i = I\ (p_x + |\ \mathbf{p}\ |\ \cos\ (i\ \delta\theta), p_y + |\ \mathbf{p}\ |\ \text{sen}\ (i\delta\theta)\ )$$

para $i = 1, 2, \ldots, [2\ \pi\ r] - 1$ y $\delta\theta = 2\ \pi / [2\ \pi\ r]$. Debido a la naturaleza discreta de la imagen en escala de grises I, las muestras c_i se obtienen mediante una interpolación bilineal [19].

Posteriormente, se obtiene la primera derivada de $t(\mathbf{p}, r)$, $dt(\mathbf{p}, r)$, mediante diferencias finitas. El máximo local de la magnitud $|\ dt(\mathbf{p}, r)\ |$ se detecta y agrupado en pares de acuerdo al signo en $dt(\mathbf{p}, r)$. Una vez que se obtienen los puntos de los bordes, se toman como parte del elemento de línea si cumplen con un cierto valor mayor a cierto umbral, como se muestra en la figura 2.9. Si esto no ocurre, se comienza una nueva búsqueda exhaustiva empezando en la siguiente línea inferior en el primer píxel del lado izquierdo.

En nuestra implementación práctica usamos solamente la mitad de la circunferencia ya que las líneas obtenidas no presentan ángulos menores a 90°.

Se muestra en la figura 2.10 (a) un acercamiento de la franja y del patrón circular generado. En la figura 2.10 (b) se muestra el patrón de escala de grises de la franja en la trayectoria circular.

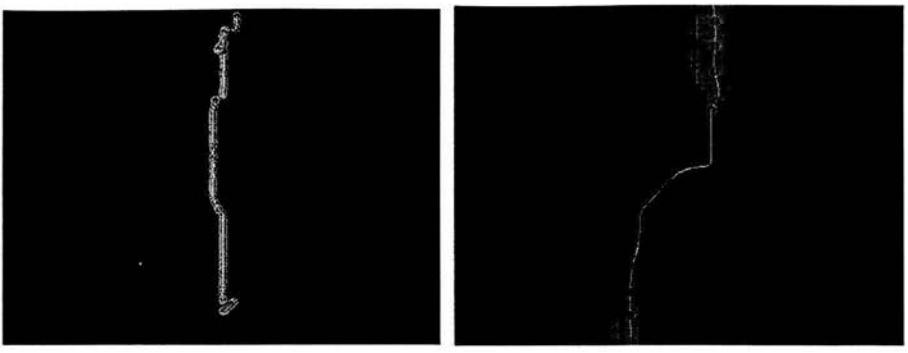


(a) (b)

Figura 2.10. (a) Acercamiento de la imagen de franja analizada. (b) Patrón de la franja analizada por la circunferencia.

La figura 2.11(a) muestra la imagen de franja y la totalidad de elementos de línea extraídos. La figura 2.11(b) es un acercamiento y se

observan los puntos a nivel subpixel obtenidos del esqueleto de la línea.



a) b)

Figura 2.11. (a) Imagen de franja y elementos de línea. (b) Acercamiento.

2.4. Algoritmo de medición

El objetivo de la reconstrucción 3D es la transformación de los puntos 2D resultado del proceso de extracción de características. Para ello, es necesario conocer los principios teóricos básicos para la medición 3D utilizando visión por computadora. Existen dos enfoques: lineal y no lineal. El enfoque lineal es el de menor exactitud, ya que no contempla los fenómenos de distorsión en la lente del sistema de adquisición de imágenes. Sin embargo, es el enfoque con mayor facilidad en la implementación, debido a que las soluciones a los planteamientos son cerradas. Por otra parte, el enfoque no lineal es el de mayor exactitud porque contempla la distorsión que se forma en la imagen y por lo tanto representa con mayor fidelidad el arreglo físico. No obstante, el enfoque no lineal contempla la minimización de funciones de error, lo que hace difícil su implementación utilizando solamente herramientas básicas de desarrollo. En el presente trabajo utilizamos el enfoque no lineal, lo que contempla el modelado no lineal de la cámara. Una vez modelado el sistema físico, entonces es posible calibrar el sistema. La calibración se realiza una sola vez en la puesta inicial del equipo y permanece sin alteraciones a menos que el arreglo varíe (posición de la cámara o de la fuente del luz láser). Con la calibración entonces es posible realizar la reconstrucción de características planas. Los detalles de los procesos mencionados se abordan con mayor profundidad a continuación.

2.4.1. Modelado no lineal de cámara

El modelado de la cámara es el proceso crucial en la reconstrucción 3D, ya que gracias a esto es posible determinar la situación espacial del arreglo físico. El modelado es una representación matemática de la cámara, ya que integra diversas variables como son su posición en el ambiente global, la distancia focal, las distorsiones de la lente, tanto radial como tangencial y la forma que afectan estos factores a la imagen que captura la cámara. En pocas palabras, con el modelado de la cámara, es posible tener una cámara virtual, y manipularla virtualmente, debido a que se sabe cómo se comportaría en la realidad.

Otro aspecto importante en el modelado de la cámara, es que permite su calibración, debido a que por la construcción de los lentes, es posible que tengan imperfecciones o distorsiones, que solamente se pueden comparar con un objeto patrón. Por esto cuando se hace el modelado, se considera también la distorsión de la lente. De esta forma, en la adquisición de imágenes es necesario obtener la imagen sin distorsión, la cual se asemeja más al objeto real.

Se utilizó un modelo no lineal, considerando distorsiones, ya que tiene mayor exactitud que uno no lineal en donde no se consideran las distorsiones [35]. El modelado se hace no lineal debido a las imperfecciones en la lente de la cámara. La imagen que se obtiene resulta de una serie de conversiones, algunas son transformaciones lineales, como rotaciones y traslaciones de ejes, y otras son no lineales, como el caso de las distorsiones [29]. Para la siguiente serie de cuatro conversiones refiérase a la figura 2.12.

Conversión 1

Primero se determina la posición en el espacio de un punto 3D, se convierten las coordenadas globales $[x_w \; y_w \; z_w]^T$ de ese punto a coordenadas de cámara $[x_c \; y_c \; z_c]^T$. Mediante rotaciones y traslaciones, en forma matricial se tiene:

$$\begin{bmatrix} x_c \\ y_c \\ z_c \end{bmatrix} = \begin{bmatrix} r_1 & r_2 & r_3 \\ r_4 & r_5 & r_6 \\ r_7 & r_8 & r_9 \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \tag{2.1}$$

en donde

T_X, T_Y, T_Z: Traslación para la transformación de coordenadas [mm].

R_X, R_Y, R_Z: Rotación para la transformación de coordenadas [rad].

además

$r_1 = \cos(R_y) \cos(R_z)$

$r_2 = \cos(R_z)\ \text{sen}(R_x)\ \text{sen}(R_y) - \cos(R_x)\ \text{sen}(R_z)$
$r_3 = \text{sen}(R_x)\ \text{sen}(R_z) + \cos(R_x)\ \cos(R_z)\ \text{sen}(R_y)$
$r_4 = \cos(R_y)\ \text{sen}(R_z)$
$r_5 = \text{sen}(R_x)\ \text{sen}(R_y)\ \text{sen}(R_z) + \cos(R_x)\ \cos(R_z)$
$r_6 = \cos(R_x)\ \text{sen}(R_y)\ \text{sen}(R_z) - \cos(R_z)\ \text{sen}(R_x)$
$r_7 = -\ \text{sen}(R_y)$
$r_8 = \cos(R_y)\ \text{sen}(R_x)$
$r_9 = \cos(R_x)\ \cos(R_y)$

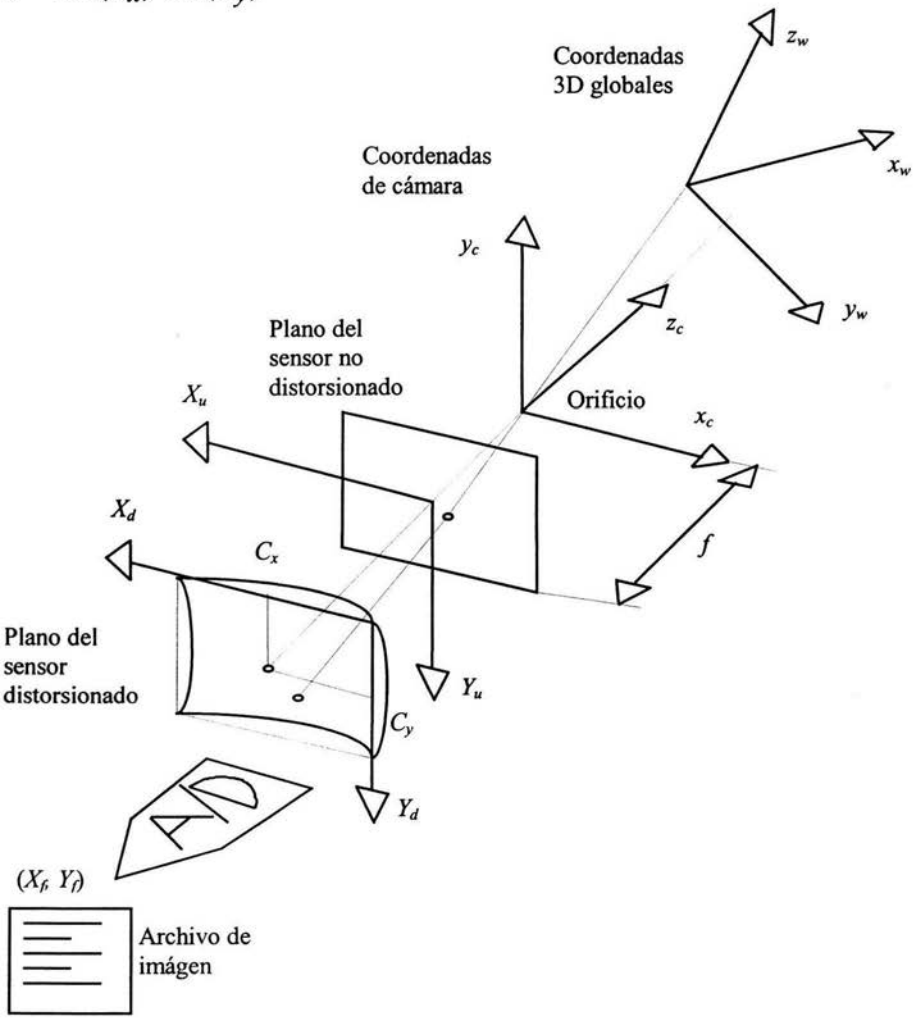


Figura 2.12. Modelo no lineal de cámara.

Conversión 2

Se convierten las coordenadas de cámara $[x_c\ y_c\ z_c]^T$ a coordenadas de sensor no distorsionado $[X_u\ Y_u]^T$. Esto es la proyección que se tendría en el plano imagen si el lente fuera perfecta, es decir plano.

$$X_u = \frac{f x_c}{z_c}$$
$$Y_u = \frac{f y_c}{z_c} \tag{2.2}$$

en donde
f: Longitud focal de la cámara de orificio [mm].

Conversión 3

En este paso se obtiene la proyección 2D distorsionada, al convertir de coordenadas de sensor no distorsionado $[X_u\ Y_u]^T$ a coordenadas de sensor distorsionado $[X_d\ Y_d]^T$. Debido a esta distorsión es que el modelo es no lineal y la solución es no cerrada. Se resuelve con minimización de errores como se verá más adelante

$$X_u = X_d\left[1 + k(X_d^2 + Y_d^2)\right]$$
$$Y_u = Y_d\left[1 + k(X_d^2 + Y_d^2)\right] \tag{2.3}$$

en donde
k: Coeficiente de distorsión de primer orden para un lente radial [1/mm2].

Conversión 4

Esta parte muestra lo que en realidad ve la cámara de acuerdo a sus dimensiones, ya que podría convertir todo el ambiente, pero solamente hará imagen lo que le dan sus dimensiones. Se convierten las coordenadas de sensor distorsionado $[X_d\ Y_d]^T$ a coordenadas de imagen $[X_f\ Y_f]^T$

$$X_f = \frac{sx}{dpx} X_d + C_x$$
$$Y_f = \frac{Y_d}{dpy} + C_y \tag{2.4}$$

en donde
sx: Factor de escala para compensar incertidumbres en el rastreo horizontal del digitalizador.
dpx: Dimensión efectiva en x del pixel en el digitalizador [mm/pixel].
dpy: Dimensión efectiva en y del pixel en el digitalizador [mm/pixel].

C_x, C_y: Coordenadas del centro radial del lente [pixel].

2.4.2. Calibración de cámara

Una vez que se tiene el modelo matemático de la cámara, el siguiente paso es encontrar como se está comportando en la realidad, esto se hace mediante una calibración, para que los valores que nos arroje sean lo más cercanos a la realidad. Recordando del primer capítulo, una calibración, se logra al utilizar un instrumento de medición de mayor exactitud que el instrumento a calibrar. En este caso se utilizó una máquina de medir por coordenadas (MMC).

El procedimiento consiste en tomar una serie de puntos en el espacio donde se vaya a colocar el objeto a medir, estos puntos deben de estar dentro de los límites de la imagen que ve la cámara. Los puntos de referencia no deben ser coplanares. El número de puntos que se utilizan son nueve o más, debido al método analítico empleado [21, 36].

La calibración tiene como objetivo estimar los parámetros del modelo de la figura 2.12 planteado en las ecuaciones (2.1) a (2.4) sobre la base del conocimiento de puntos 3D en el sistema de coordenadas global, $[x_w, y_w, z_w]$ y sus proyecciones en el plano 2D, $[X_f, Y_f]$. El modelado anterior se puede plantear como el siguiente sistema de ecuaciones

$$[X_f, Y_f] = F \{ [x_w, y_w, z_w] [R_x, R_y, R_z, T_x, T_y, T_z, f, k, C_x, C_y, sx, dpx, dpy] \} \quad (2.5)$$

En donde se indica explícitamente en la ecuación la relación entre las variables dependientes $[X_f, Y_f]$ conocidas, las variables independientes $[x_w, y_w, z_w]$ conocidas y los parámetros $[R_x, R_y, R_z, T_x, T_y, T_z, f, k, C_x, C_y, sx, dpx, dpy]$ desconocidos a estimar.

El método aplicado para estimar los parámetros es Levenberg-Marquardt, ya que es bastante popular y está cobrando importancia como método estándar en el modelado de sistemas no lineales [21].

La calibración es un proceso que se puede aplicar una sola vez y perdura para realizar reconstrucciones sucesivas mientras la posición del arreglo no varíe.

2.4.3. Calibración del plano

La calibración del plano consiste en determinar la ecuación del plano del haz láser. Se toman tres puntos ayudados de la MMC, en el camino recorrido por el plano del láser.

Con estos tres puntos es posible determinar la ecuación del plano dada por:

$$Ax + By + Cz + D = 0$$

Aplicando la ecuación en (2.6) a los tres puntos (x_1, y_1, z_1) (x_2, y_2, z_2) (x_3, y_3, z_3) conocidos obtenemos

$$Ax_1 + By_1 + Cz_1 + D = 0$$
$$Ax_2 + By_2 + Cz_2 + D = 0$$
$$Ax_3 + By_3 + Cz_3 + D = 0$$

Que resulta en un sistema no singular de ecuaciones. Resolviendo.

$$\begin{aligned} A &= y_1(z_2 - z_3) + y_2(z_3 - z_1) + y_3(z_1 - z_2) \\ B &= z_1(x_2 - x_3) + z_2(x_3 - x_1) + z_3(x_1 - x_2) \\ C &= x_1(y_2 - y_3) + x_2(y_3 - y_1) + x_3(y_1 - y_2) \\ D &= -\left[x_1(y_2z_3 - y_3z_2) + x_2(y_3z_1 - y_1z_3) + x_3(y_1z_2 - y_2z_1) \right] \end{aligned} \qquad (2.6)$$

La ecuación 2.6 puede ser resuelta para encontrar los coeficientes (A, B, C, D) que definen el plano láser, a partir del conocimiento previo de tres puntos sobre el plano.

2.4.4. Reconstrucción 3D

Una vez que se tienen los coeficientes de la calibración $[R_x, R_y, R_z, T_x, T_y, T_z, f, k, C_x, C_y, sx, dpx, dpy, A, B, C, D]$, el siguiente paso es obtener del objeto analizado la proyección de cada punto de interés 2D $[X_f, Y_f]$ y trasnsformarlo a coordenadas 3D de una posición $[x_w, y_w, z_w]$. De (2.2)

$$x_c = z_c \frac{X_u}{f}$$
$$y_c = z_c \frac{Y_u}{f}$$

sustituyendo en (2.3) y desarrollando

$$z_c \frac{X_u}{f} = r_1 x_w + r_2 y_w + r_3 z_w + T_x$$

$$z_c \frac{Y_u}{f} = r_4 x_w + r_5 y_w + r_6 z_w + T_y$$

$$z_c = r_7 x_w + r_8 y_w + r_9 z_w + T_z$$

sustituyendo la ultima ecuación en las dos primeras

$$(r_7 x_w + r_8 y_w + r_9 z_w + T_z)\frac{X_u}{f} = r_1 x_w + r_2 y_w + r_3 z_w + T_x$$

$$(r_7 x_w + r_8 y_w + r_9 z_w + T_z)\frac{Y_u}{f} = r_4 x_w + r_5 y_w + r_6 z_w + T_y$$

agrupando

$$\left(r_7 \frac{X_u}{f} - r_1\right)x_w + \left(r_8 \frac{X_u}{f} - r_2\right)y_w + \left(r_9 \frac{X_u}{f} - r_3\right)z_w + \left(T_z \frac{X_u}{f} - T_x\right) = 0$$

$$\left(r_7 \frac{Y_u}{f} - r_4\right)x_w + \left(r_8 \frac{Y_u}{f} - r_5\right)y_w + \left(r_9 \frac{Y_u}{f} - r_6\right)z_w + \left(T_z \frac{Y_u}{f} - T_y\right) = 0$$

que forman un sistema singular de dos ecuaciones con tres incógnitas, [xw yw zw]. Para formar un sistema de ecuaciones no singular se incluye la ecuación del plano láser

$$\left(r_7 \frac{X_u}{f} - r_1\right)x_w + \left(r_8 \frac{X_u}{f} - r_2\right)y_w + \left(r_9 \frac{X_u}{f} - r_3\right)z_w + \left(T_z \frac{X_u}{f} - T_x\right) = 0$$

$$\left(r_7 \frac{Y_u}{f} - r_4\right)x_w + \left(r_8 \frac{Y_u}{f} - r_5\right)y_w + \left(r_9 \frac{Y_u}{f} - r_6\right)z_w + \left(T_z \frac{Y_u}{f} - T_y\right) = 0$$

$$Ax_w + By_w + Cz_w + D = 0$$

o en forma matricial

$$\begin{bmatrix} r_7 \frac{X_u}{f} - r_1 & r_8 \frac{X_u}{f} - r_2 & r_9 \frac{X_u}{f} - r_3 \\ r_7 \frac{Y_u}{f} - r_4 & r_8 \frac{Y_u}{f} - r_5 & r_9 \frac{Y_u}{f} - r_6 \\ A & B & C \end{bmatrix} \begin{bmatrix} x_w \\ y_w \\ z_w \end{bmatrix} = \begin{bmatrix} T_x - T_z \frac{X_u}{f} \\ T_y - T_z \frac{Y_u}{f} \\ -D \end{bmatrix}$$

$$[MR][MW2] = [MT]$$

despejando las coordenadas globales

$$[MW2] = [MR]^{-1}[MT] \tag{2.7}$$

con

$$X_d = \frac{dpx}{sx}\left(X_f - C_x\right)$$
$$Y_d = dpy\left(Y_f - C_y\right)$$
$$X_u = X_d\left[1 + k(X_d^2 + Y_d^2)\right]$$
$$Y_u = Y_d\left[1 + k(X_d^2 + Y_d^2)\right]$$

La ecuación (2.7) indica la posibilidad de medir 3D a partir de una proyección plana y las calibraciones de la cámara y el plano láser [13]. Como consecuencia, es posible la medición mediante el esquema planteado en la figura 2.1.

Cabe hacer mención que la reconstrucción tridimensional planteada en la ecuación (2.7) implica primero realizar las calibraciones indicadas en las ecuaciones (2.5) y (2.6). En una situación real, basta con calibrar una sola vez el sistema para realizar mediciones de prácticamente cualquier objeto. La importancia del enfoque no lineal radica en su alta exactitud en la reconstrucción.

2.5. Algoritmo de control

El algoritmo de control que gobierna la mesa divisora permite el posicionamiento angular de objetos de manera que estos puedan ser inspeccionados en toda su geometría por el sistema de visión. La mesa divisora de ángulo es un dispositivo electromecánico integrado por un prototipo mecánico y un sistema electrónico para el control de posición. La medición en una posición del sistema electromecánico conforma un meridiano. Al rotar el sistema electromecánico en posiciones discretas el conjunto de meridianos conforma una medición completa.

2.5.1. Prototipo mecánico

El prototipo contempla los dispositivos para soportar firmemente la mesa giratoria sobre la cual se coloca el objeto a medir. La mesa es impulsada por un motor CD que incluye un codificador óptico de posición angular, figura 2.13.

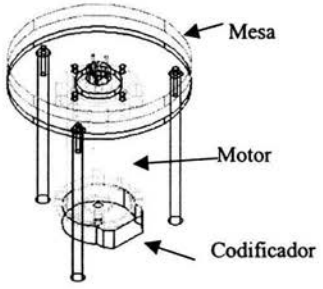


Figura 2.13 Prototipo mecánico.

2.5.2. Sistema electrónico para el control de posición

Se implementó un controlador PID en modo posicional de acuerdo al esquema de la figura 2.14.

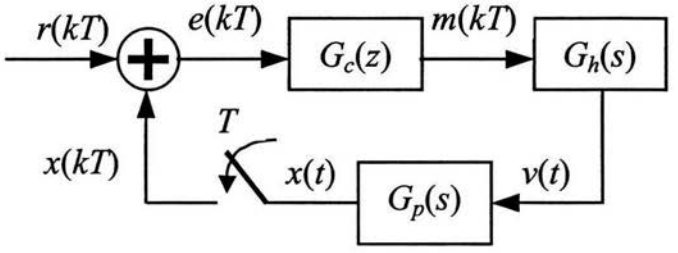


Figura 2.14. Diagrama a bloques para el sistema digital de control de posición.

En donde $G_c(z)$ es el controlador PID digital, $G_h(s)$ es un retenedor de orden cero y $G_p(s)$ es la planta [26]. En el dominio discreto el sistema en conjunto resulta en la siguiente ecuación.

$$G(z) = Z\ \{\ G_h(s)\ G_p(s)\ \}$$

$$G_c(z) = K_p + \frac{K_i}{1 - z^{-1}} + K_d\left(1 - z^{-1}\right)$$

Entonces la respuesta en lazo cerrado es

$$G_l(z) = \frac{G_c(z)G(z)}{1 + G_c(z)G(z)}$$

La ultima ecuación es utilizada para sintonizar los parámetros del controlador (K_p, K_i y K_d). La implementación actual para el control de posición es mediante el esquema de la figura 2.15 y software específico. El software incluye temporizadores de alta resolución con el objeto de cumplir con un periodo de muestreo de T = 10kHz.

En la figura 2.15, la planta esta integrada por el arreglo representado en el prototipo mecánico. Con el objetivo de conseguir operación en tiempo real, la implementación opera el codificador óptico mediante una tarjeta de interfase para computadora PC7266 [30]. Se diseñó un circuito electrónico impulsor CD que suministra las señales PWM a la planta, figura 2.16.

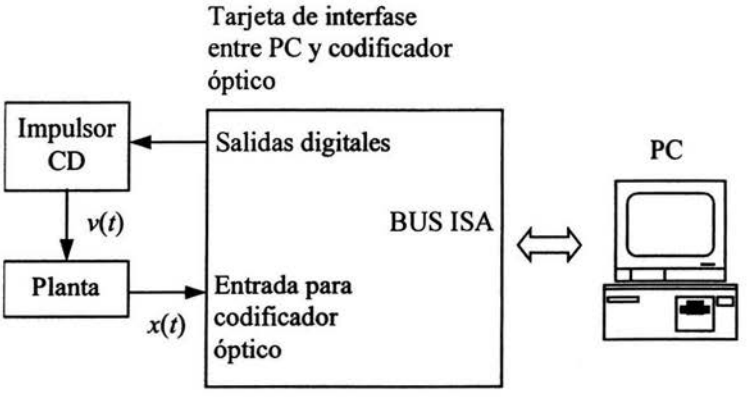


Figura 2.15. Implementación del control de posición.

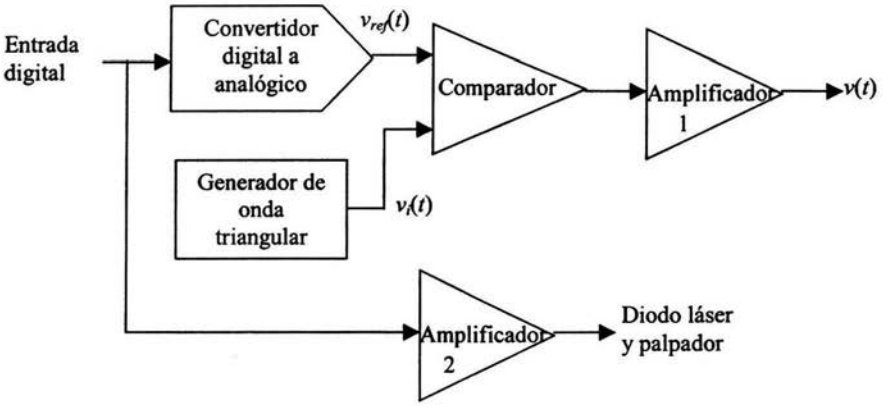


Figura 2.16. Impulsor CD.

En la figura 2.16, una entrada digital de 8 bits es convertida al voltaje analógico $v_{ref}(t)$ y comparado con la señal triangular $v_i(t)$. El comparador genera una señal PWM en donde el nivel de corriente directa es proporcional al voltaje de referencia. El amplificador 1 suministra la corriente que demanda la planta. El amplificador 2 proporciona energía al diodo láser y a un palpador luminoso usado con propósitos de calibración. Las señales involucradas se muestran en la figura 2.17.

La sincronía que implementa el software de desarrollo particular permite realizar lo siguiente en forma secuencial.

- Posicionar la mesa divisora.
- Capturar la escena sin el patrón de luz estructurada.
- Encender la luz estructurada.
- Capturar la escena con el patrón de luz estructurada.
- Apagar la luz estructurada.
- Desarrollar la extracción de características y reconstrucción 3D.
- Ejecutar una nueva posición.

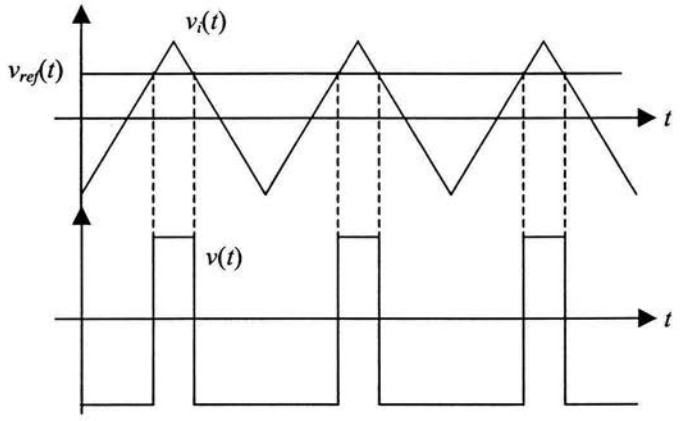


Figura 2.17. Señales de voltaje en el impulsor CD.

Capítulo 3

Implementación de la interfase de usuario

El sistema de medición por visión es operado mediante un programa desarrollado específicamente para esta aplicación llamado "Scan3D". El programa implementa la interfase de usuario y soporta los dispositivos periféricos que complementan el sistema con operación en tiempo real y mínima intervención por parte del usuario. En este capítulo se explicará el programa en dos enfoques fundamentales. El primero es un manual para el usuario que desea utilizar el sistema para la medición por visión. El segundo constituye una guía orientada al programador para facilitar el seguimiento del software que implementa el algoritmo descrito en el capítulo 2. No se incluye el código fuente debido a su gran extensión. En lugar de ello, se describen las clases que conforman el proyecto de programación.

3.1. Introducción

El software "Scan3D" fue desarrollado en el Centro de Ciencias Aplicadas y Desarrollo Tecnológico CCADET, por parte del Laboratorio de Metrología, con el objeto de crear un instrumento de medición dimensional sin contacto utilizando visión por computadora. La plataforma de desarrollo utilizada fue Visual C++ 6.0 [3, 15, 20], debido a las bondades que presenta este lenguaje para crear una interfase amigable tanto con el usuario final como con el programador, además de que es posible interactuar directamente con dispositivos externos como son la tarjeta de captura DT3151 y la tarjeta de control para la mesa divisora de ángulo PC7266.

Se describe el software que implementa el algoritmo del capítulo 2 y que constituye la interfase con el usuario para operar el sistema de medición por visión. Algunas de las tareas principales del software "Scan3D" son las siguientes:

- Capturar y procesar imágenes en tiempo real mediante la plataforma hardware descrita en 2.3.1.
- Extraer características en imágenes de franjas mediante el algoritmo descrito en 2.3.2.
- Realizar los procesos de calibración de cámara y plano con interfase a la MMC en tiempo real como se describió en 2.4.2 y 2.4.3.
- Realizar la reconstrucción 3D a partir de características planas como se describió en 2.4.4.
- Controlar y registrar la posición de la mesa divisora de ángulo descrita en 2.5.2.
- Administrar archivos de medición (guardar, abrir).
- Desplegar reconstrucciones 3D en un ambiente interactivo con operación en tiempo real.

Los requerimientos mínimos para ejecutar y compilar el programa "Scan3D" son los siguientes:

- Computadora Pentium III, 64MB en RAM, 40GB HD, cavidad PCI, cavidad ISA, Windows 2000.
- Tarjeta de digitalización de imágenes DT3155 con su respectivo manejador.
- Tarjeta de interfase entre codificadores ópticos y PC con su respectivo manejador.
- Control Active-X DT-Active Open Layers [4].
- Fuente de entrada de video (cámara).
- Fuente de luz estructurada (diodo láser).
- Mesa divisora de ángulo.
- Visual C++ 6.
- Patrón tridimensional MMC (sólo para la calibración).
- Bases y soportes.

Las características principales del software "Scan3D" son las siguientes:

- Proyecto sobre la base de "documento único".
- Despliegue y procesamiento de video en tiempo real.
- Despliegue interactivo de gráficos 3D en tiempo real.
- Uso de temporizadores de alta velocidad para el controlador PID posicional.
- Uso de controles Active-X.
- Uso de mapas de bits.
- Manejo de variables del tipo VARIANT y SAFEARRAY.

3.2. Descripción en el ámbito del operador

La siguiente descripción tiene como objetivo orientar al usuario final en la operación del software de medición "Scan3D". Se describen las características de la aplicación, su principal función y el modo correcto para emplearlas. La interfase de usuario presenta varios componentes para realizar la medición dimensional 3D. A continuación se explica cada a uno a detalle.

3.2.1. Pantalla principal

En la pantalla principal, como se muestra en la figura 3.1, aparece en la parte superior un menú de opciones y una barra de herramientas con cinco de las opciones más comunes. En la parte central del lado izquierdo hay una pantalla que muestra el ambiente 3D reconstruido. En la esquina superior derecha, se presenta el cuadro de "Video" en tiempo real que corresponde al volumen de visión proyectado en la cámara. Debido a que el recurso de video en vivo es compartido por otros componentes de la aplicación, se cuenta con un botón de control para activar o desactivar la captura. En particular, el proceso de calibración de la cámara utiliza el despliegue de video, de manera que antes de calibrar se requiere detener el despliegue de video de la pantalla principal. Debajo del cuadro de video se encuentra la barra "Progreso" que muestra el estado del proceso de reconstrucción. Debajo de la barra de progreso se cuenta con el control deslizable "Divisiones" para determinar el número de mediciones a realizar en un giro completo de 360°. El número de divisiones permitido se encuentra entre 1 y 400, lo cual proporciona una resolución angular de 0.9°. Por último, debajo del control divisiones se encuentra el botón "Medir" mediante el cual inicia el proceso automático de medición y

reconstrucción. Al presionar el botón "Medir", el proceso puede abortarse mediante este mismo botón, ya que éste se transforma en el botón "Abortar".

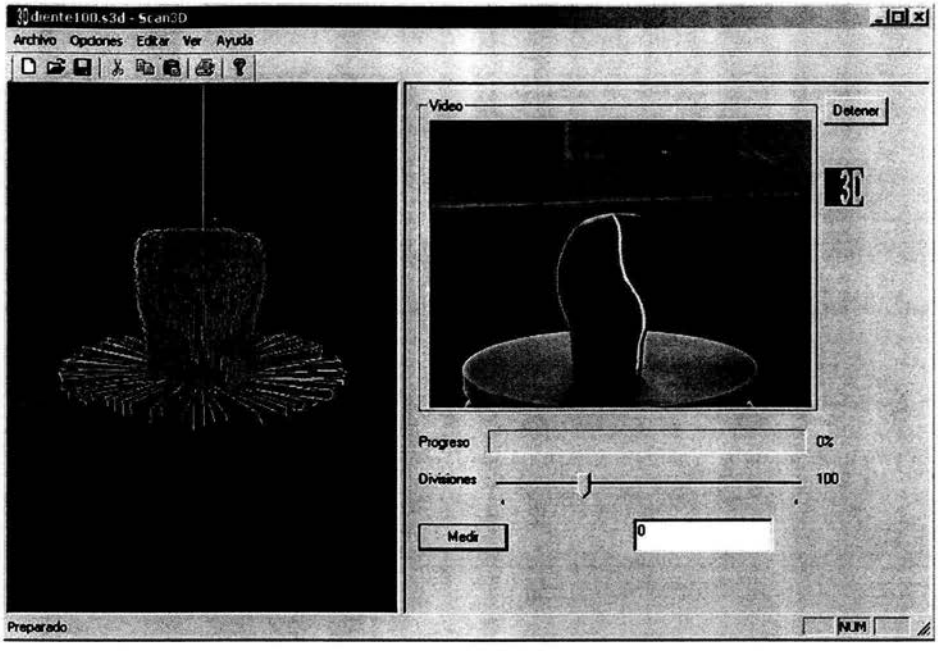


Figura 3.1. Pantalla principal.

3.2.2. Menú de opciones

La totalidad de las opciones del programa son agrupadas en un menú con las siguientes cuatro categorías.

Archivo. Corresponde a las funciones para administrar archivos de medición. Los archivos pueden ser nuevas mediciones, almacenados y abrir las que ya se hayan hecho previamente. Se cuenta con un acceso rápido a los cuatro archivos recientemente abiertos. Las opciones para imprimir y configurar impresora se encuentran en esta categoría. En la figura 3.2 se muestra el menú deslizable con las opciones descritas.

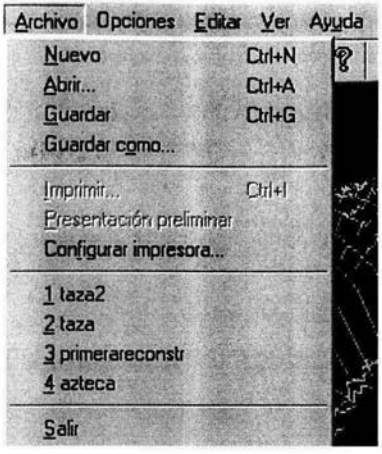


Figura 3.2. Funciones del menú Archivo.

Opciones. Presenta las funciones para la calibración de la cámara y el plano láser, como se describió en 2.4.2 y 2.4.3. En la figura 3.3 se muestra el menú deslizable con las opciones para administrar las calibraciones. Las opciones de calibración son administradas por las cajas de diálogo que se describirán en las siguientes secciones.

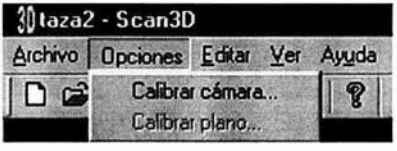


Figura 3.3. Funciones del menú Opciones.

Ver. Simplemente habilita y deshabilita las barra de estado y herramientas, figura 3.4.

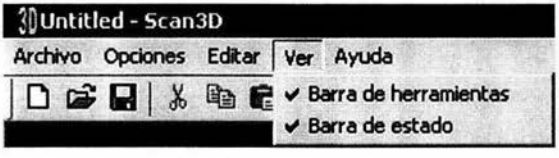


Figura 3.4. Funciones del menú Ver.

Ayuda. Muestra la caja de diálogo "Acerca de...", figuras 3.5 y 3.6.

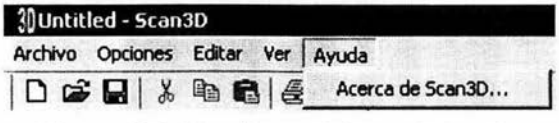


Figura 3.5. Funciones del menú Ayuda.

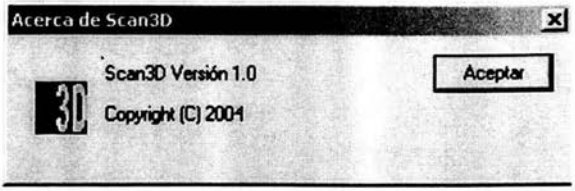


Figura 3.6. Caja de diálogo Acerca de...

3.2.3. Caja de diálogo "Calibración de la cámara"

La calibración de la cámara es indispensable para iniciar cualquier medición cuando el arreglo de la cámara o del láser haya cambiado, o cuando sea necesario modificar la posición de la mesa divisora para ajustarla dentro del espacio de medición. En esta etapa es necesario conectar un palpador luminoso a la tarjeta electrónica. La calibración consiste en tomar nueve o más posiciones patrón dentro de un volumen de medición específico que puedan ser visualizados por la cámara, figura 3.7. Para la generación y registro de las posiciones patrón, se utiliza la Máquina de Medir por Coordenadas, MMC, desarrollada en el CCADET UNAM [2].

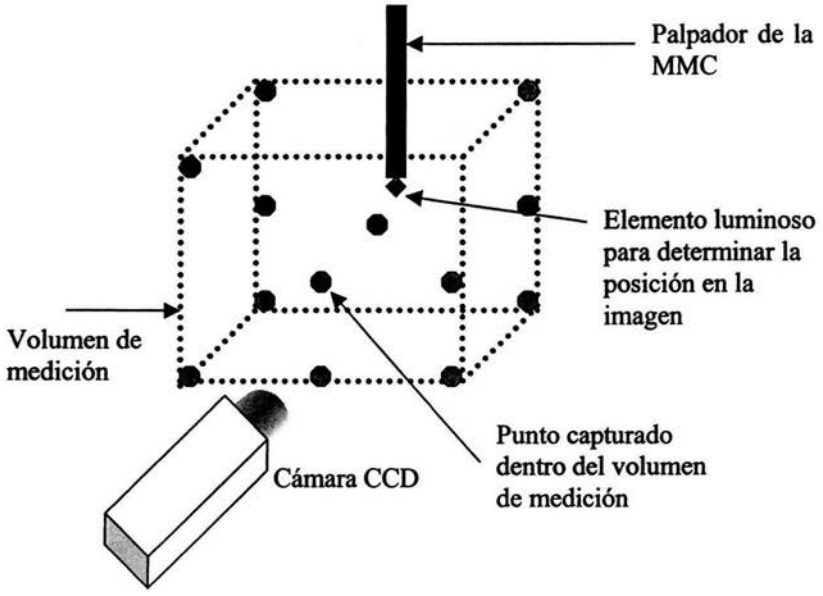


Figura 3.7. Arreglo necesario para la calibración de la cámara.

Como se muestra en la figura 3.8, la posición actual del palpador en la MMC se registra en los tres cuadros "Eje X", "Eje Y" y "Eje Z". Los ejes X, Y y Z pueden ser modificados individualmente mediante los botones "Reset" e "Invierte". El botón "Reset" pone a cero la

lectura del eje correspondiente y el botón "Invierte" cambia el signo de la lectura o sentido de dirección. Los botones anteriores son utilizados para elegir el sistema de coordenadas más apropiado. Al presionar el botón "Procesar" se captura la medición de las coordenadas 3D de la MMC y además se captura y procesa la posición del centroide del palpador luminoso en coordenadas planas 2D (renglón y columna). El par de coordenadas 3D y 2D es ingresado a la lista de puntos. Cuando ya se han registrado las lecturas suficientes se habilita el botón "Guardar" para respaldar en archivo los puntos capturados (in.txt). En cualquier momento puede ser utilizado el botón "Deshacer" con el objeto de eliminar el par de coordenadas más recientemente ingresado a la lista. El botón "Ok" se activa siempre y cuando se haya guardado el archivo con "Guardar". Para cerrar se da clic en el botón "Ok", cuando se hicieron cambios en los parámetros, y "Cancel" si se quiere cerrar la ventana sin modificar los parámetros existentes. Se genera un archivo con los parámetros de la calibración para ser utilizados en la reconstrucción de los puntos de los objetos a medir (out.txt). Con el fin de evitar la eliminación de los parámetros de calibración, al presionar "OK" se pide al usuario la confirmación de la operación. Las coordenadas del cursor sobre la imagen son desplegadas en todo instante mediante el cuadro "Coordenadas imagen". Lo anterior es con el propósito de verificar que efectivamente las coordenadas del centroide corresponden a las ingresadas en la lista.

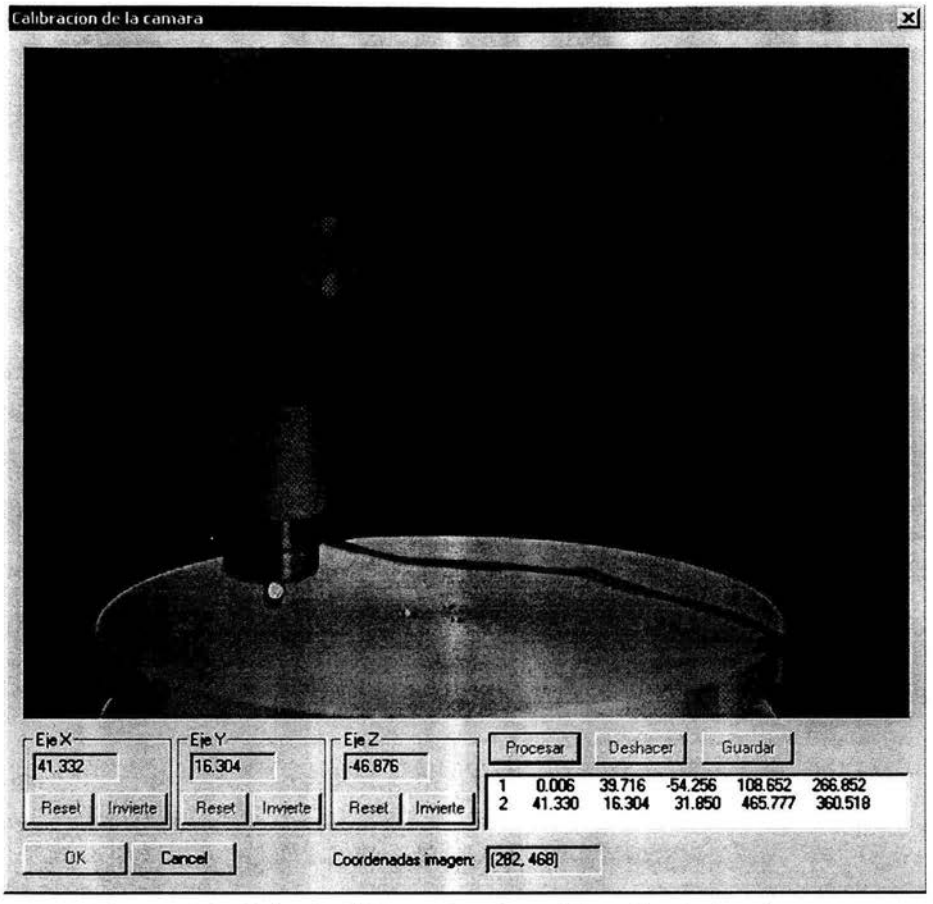


Figura 3.8. Caja de diálogo para la calibración de la cámara.

3.2.4. Caja de diálogo "Calibración del plano láser"

En la calibración del plano es necesario tomar tres puntos para poder encontrar la ecuación del mismo. Adicionalmente, el diodo láser debe ser conectado a la tarjeta electrónica. La calibración del plano debe corresponder con el sistema de ejes definido como parte de la calibración. Por esto la opción para calibrar el plano solamente esta disponible justamente después de haber calibrado la cámara. La figura 3.9 muestra la caja de diálogo auxiliar en la calibración. Es importante hacer notar que en la calibración del plano es imposible modificar el sistema de ejes coordenados, mediante botones de "Reset" e "Inversión" de signo.

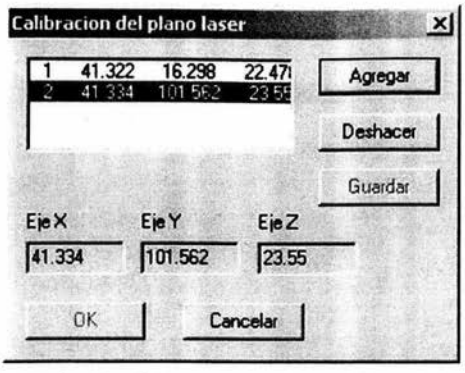


Figura 3.9. Caja de diálogo para la calibración del plano.

La caja de diálogo muestra la posición actual de la MMC en las cajas "Eje X", "Eje Y" y "Eje Z". El botón "Agregar" inserta la posición actual de la MMC en la lista. De esta forma, antes de agregar una lectura, el palpador de la MMC debe ser posicionado sobre el haz del plano que forma la luz láser. El botón "Deshacer" elimina la última lectura ingresada a la lista, con el propósito de enmendar posibles errores. Una vez introducidos los tres puntos que definen el plano, se activa el botón "Guardar". Al presionar el botón, se realiza lo siguiente: se procesan los puntos para obtener la ecuación del plano y se sobrescribe el archivo de texto plano.txt que contiene los cuatro parámetros de la ecuación. Una vez guardado el archivo, el botón "Ok" se activa. Para cerrar se da clic en el botón "Ok", cuando se hicieron cambios en los parámetros, y "Cancel" si se quiere cerrar la ventana sin modificar los parámetros existentes. Con el fin de evitar la eliminación de los parámetros de calibración, al presionar "OK" se pide al usuario la confirmación de la operación.

3.2.5. Procedimiento para realizar una medición

Los siguientes son los pasos indispensables para realizar una medición utilizando el sistema de visión.

1) Colocar los dispositivos asociados al sistema de medición por visión aproximadamente como muestra la figura 3.10. Además, colocar el palpador luminoso en la MMC y utilizar las bases para la cámara y el diodo láser.

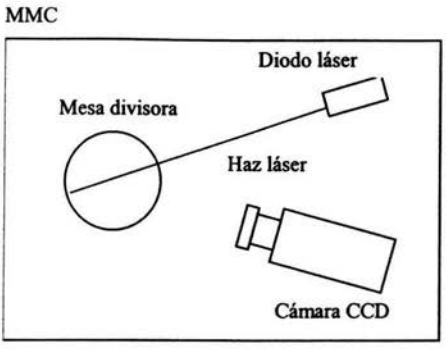


Figura 3.10. Posición de los dispositivos del sistema de medición por visión (vista en planta).

2) Realizar las conexiones de la figura 3.11.

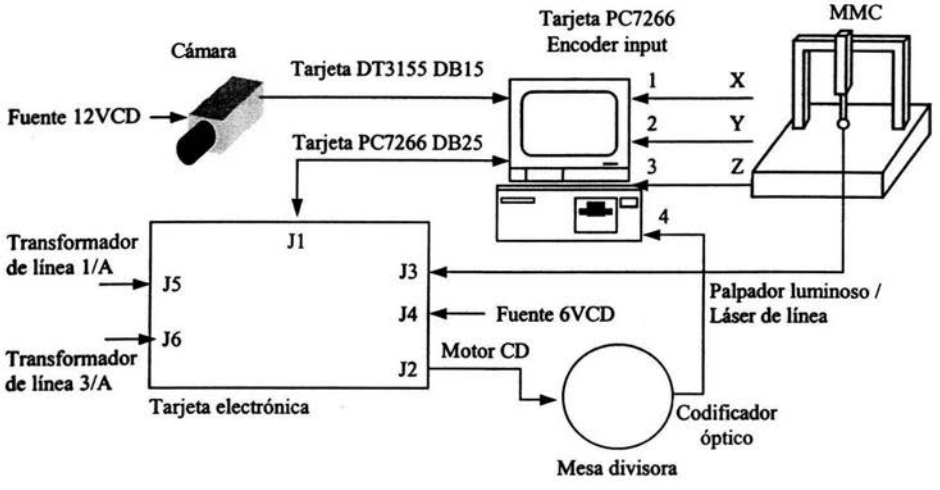


Figura 3.11. Conexiones del sistema de medición por visión.

3) Iniciar los dispositivos en la siguiente secuencia: Energizar la cámara, iniciar el programa "Scan3D" en la PC, energizar la fuente 6VCD y energizar la tarjeta electrónica.

4) Colocar la cámara de forma que su campo de visión abarque el objeto, procurando ocupar la mayor parte de la pantalla. Ajustar la cámara para que la imagen este enfocada y con buena iluminación.

5) Después colocar el láser para que su haz coincida en la medida de lo posible con el eje de giro de la mesa divisora.

6) Una vez ajustados los dispositivos es necesario calibrar el instrumento: la cámara y el plano. Primero conectar y colocar el palpador luminoso. Despues, definir los ejes como se muestra en la figura 3.12. Deshabilitar el video en vivo con el botón "Detener". Posteriormente, en el menú "Opciones" dar clic en "Calibrar cámara...", tomar 9 puntos no coplanares como mínimo, oprimir el botón "Guardar" y "Ok".

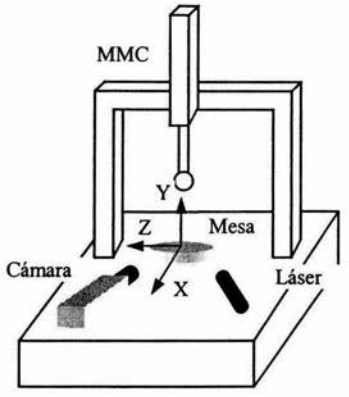


Figura 3.12. Calibración de cámara.

7) Para calibrar el plano, primero colocar y conectar el láser de línea. Después, en el menú "Opciones" dar clic en "Calibrar plano...", tomar 3 puntos no colineales como mínimo, oprimir el botón "Guardar" y "Ok".

8) Una vez calibrada la cámara y el plano, se debe colocar el objeto sobre la mesa divisora dentro del volumen de medición determinado en la calibración. Habilitar el video en vivo presionando el botón "Iniciar".

9) El siguiente paso es determinar cuantas mediciones se desean realizar, es decir, en cuantas partes se desea dividir la captura de los 360° del objeto e ingresarlos en el control deslizable "Divisiones".

10) Iniciar la medición presionando el botón "Medir".

3.3. Descripción en el ámbito del programador

La figura 3.13 muestra las catorce clases en las cuales se organizó el proyecto de programación "Scan3D". El proyecto se construyó sobre la base de la arquitectura documento-vista, utilizando la MFC y el asistente para la creación de proyectos, APP Wizard [3] de Visual C++ 6.0.

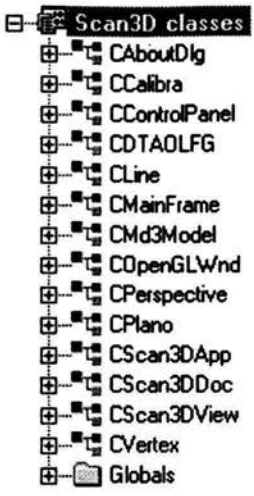


Figura 3.13. Clases empleadas en el software desarrollado, Scan3D.

3.3.1. Clases y métodos creados por el asistente

La siguiente es la descripción de las clases generadas por omisión mediante el asistente para proyectos nuevos de Visual C++ 6.0. Las clases soportan principalmente la arquitectura documento vista y la apariencia final de la aplicación.

CAboutDlg

La clase CAboutDlg, figura 3.14, es utilizada para desplegar la caja de diálogo "Acerca de Scan3D", que contiene información breve del programa. Los métodos se describen a continuación:

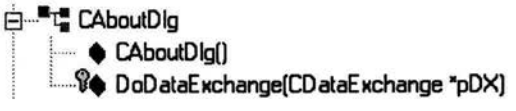


Figura 3.14. Clase CAboutDlg.

CAboutDlg(); Es el constructor por omisión de la clase. No realiza inicializaciones adicionales.

virtual void DoDataExchange(CDataExchange* pDX); Permite el intercambio de datos con otras clases.

CScan3DApp

La clase CScan3DApp es utilizada para el despliegue de la aplicación en forma de caja de diálogo, figura 3.15. Los métodos se describen a continuación.

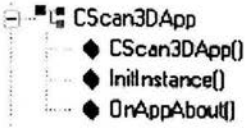


Figura 3.15. Clase CScan3DApp.

CDtApp(); Es el constructor por omisión de la aplicación. No realiza inicializaciones adicionales.

virtual BOOL InitInstance(); Realiza las siguientes inicializaciones estándar. Inicia la apariencia 3D de los controles. Habilita la conexión entre el documento y sus vistas.

afx_msg void OnAppAbout(); Constituye el procedimiento para atender el evento por desplegar la caja de diálogo "Acerca de...".

CScan3DDoc

La Clases CScan3DDoc, la figura 3.16, es utilizada para la administración de los datos de la aplicación, del documento, y la manera en que ellos interactúan con en el programa. Los métodos se describen a continuación.

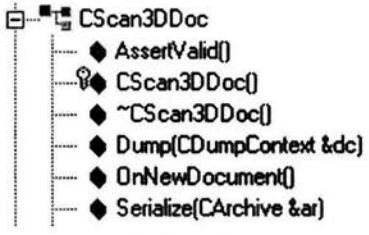


Figura 3.16. Clase CScan3DDoc.

virtual void AssertValid() const; Valida el documento del programa.

CScan3DDoc(); Es el constructor por omisión de la clase. Inicializa el formato de archivo para los datos empleados en el documento.

virtual ~CScan3DDoc(); Es el destructor de la clase.

virtual void Dump(CDumpContext& dc) const; Vacía la estructura de contexto del dispositivo dc.

virtual BOOL OnNewDocument(); Inicializa los datos para crear una nueva estructura del documento para almacenar.

virtual void Serialize(CArchive& ar); Permite el almacenamiento de los datos de memoria con el formato de archivo empleado.

CScan3DView

La clase CScan3DView, en la figura 3.17, presenta los métodos necesarios para el despliegue de la clase de vista por omisión. En nuestro caso, ésta clase no implementa código alguno que altere la interfase de medición, como se puede notar en la siguiente descripción de los métodos.

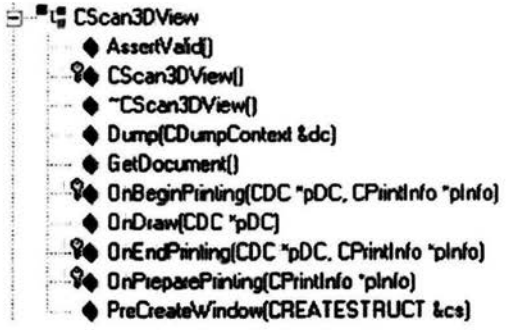


Figura 3.17. Clase CScan3DView.

virtual void AssertValid() const; Valida la clase, llama a el método de CView AssertValid().

CScan3DView(); Es el constructor por omisión de la clase. No realiza inicializaciones adicionales.

virtual ~CScan3DView(); Es el destructor de la clase.

virtual void Dump(CDumpContext& dc) const; Libera el contexto del dispositivo dc.

CScan3DDoc* GetDocument(); Permite el intercambio de datos que pertenecen al documento de mediciones con otras clases.

virtual void OnDraw(CDC* pDC); Son las instrucciones que se deben seguir cada vez que la ventana debe repintarse.

virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo); No aplica en este programa pero es para preparar los datos a imprimir, borra los objetos GDI(Graphic Display Interface) creados.

CMainFrame

La clase CMainFrame, figura 3.18, contiene los métodos necesarios para la asignación de las clases a los paneles divisibles de la aplicación, como se ilustra en la descripción de los siguientes métodos.

Figura 3.18. Clase CMainFrame.

virtual void AssertValid() const; Valida la clase. Manda al método AssertValid() de la clase CFrameWnd.

CMainFrame(); Es el constructor por omisión.

virtual ~CMainFrame(); El destructor de la clase.

virtual void Dump(CDumpContext& dc) const; Libera el contexto de dispositivo.

afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct); Es un prototipo de manejador de mensajes al crear la estructura.

virtual BOOL OnCreateClient(LPCREATESTRUCT lpcs, CCreateContext* pContext); Verifica la creación del cliente, en este caso son las ventanas en las que se va a dividir la vista del programa, se especifica el tamaño y posición de los paneles usados.

afx_msg void OnOpcionesCalibrarplano(); Maneja el evento de mensaje cuando se llama la opción de Calibrar plano.

afx_msg void OnSize(UINT nType, int cx, int cy); Maneja el evento del mensaje de cambiar el tamaño de la ventana de la vista.

afx_msg void OnUpdateFileNew(CCmdUI* pCmdUI); Manejador del evento Crear un nuevo archivo.

afx_msg void OnUpdateFileOpen(CCmdUI* pCmdUI); Manejador del evento Abrir archivo.

afx_msg void OnUpdateFileSave(CCmdUI* pCmdUI); Manejador del evento Salvar archivo.

afx_msg void OnUpdateFileSaveAs(CCmdUI* pCmdUI); Manejador del evento Salvar Como.

virtual BOOL PreCreateWindow(CREATESTRUCT& cs); Hace modificaciones a las propiedades de la ventana.

3.3.2. Clases y métodos de la tarjeta para la captura de imágenes

La siguiente es la descripción de la clase que por omisión es insertada en el proyecto al utilizar el control Actives DT-Active Open Layers. El mencionado control es distribuido comercialmente por el fabricante de la tarjeta para la captura de video, Data Translation.

CDTAOLFG

La clase CDTAOLFG es creada por el ambiente de desarrollo al incluír el control Active-X llamado "DT-Active Open Layers", figura 3.19. El control es suministrado por el fabricante de la tarjeta DT3155, por lo que consideramos que no requiere explicación, ya que sus métodos son los desarrollados y depurados por el fabricante. No requiere de modificaciones y la información de los métodos y variables se puede consultar en la referencia [4].

Figura 3.19. Clase CDTAOLFG.

3.3.3. Clases y métodos para la medición

La siguiente descripción corresponde a las clases que principalmente e implementan los procesos de medición y control en el instrumento para la medición por visión.

CControlPanel

La clase CControlPanel de la figura 3.20 es una de las más importantes de la aplicación. Aquí es donde se implementa el algoritmo para la extracción de características y para el despliegue de las imágenes y video en tiempo real. También aquí se controlan los temporizadores en la implementación del PID digital. Los métodos se describen a continuación.

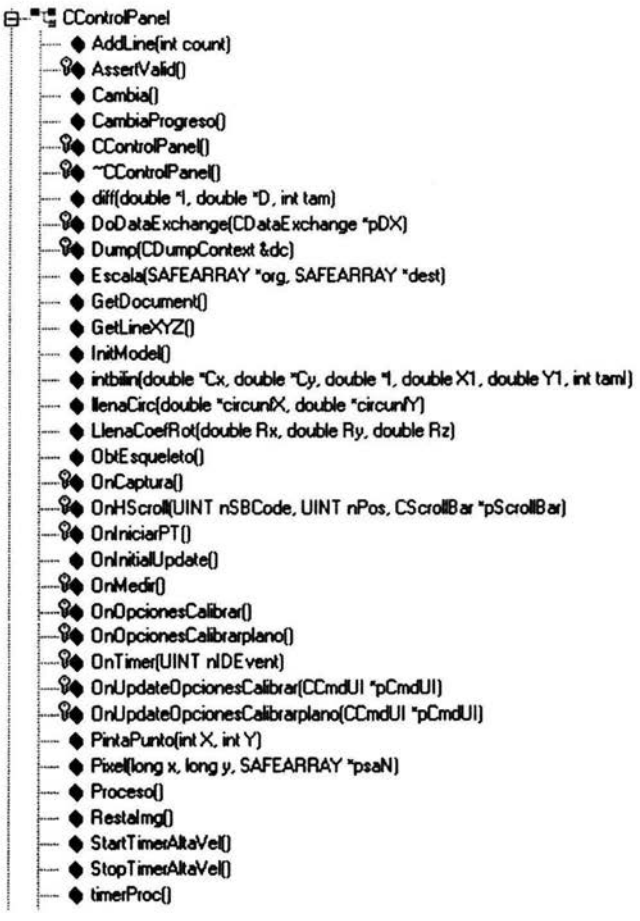


Figura 3.20. Clase CControlPanel.

void AddLine(int count); Añade una línea al modelo 3D con los puntos encontrados en la captura.

virtual void AssertValid() const; Valida la clase CControlPanel.

void Cambia(); Manda instrucciones para cambiar los parámetros y mover el motor.

void CambiaProgreso(); Actualiza la barra de progreso.

CControlPanel(CWnd* pParent = NULL); Es el constructor por omisión. Inicia variables, arreglos VARIANT y SAFEARRAY para el manejo de imágenes.

virtual ~CControlPanel(); Es el destructor por omisión. Detiene el motor.

void diff(double * I, double * D, int size); Esta función hace la ecuación en diferencias de los puntos del patrón circular descrita en la sección 2.3.2.

virtual void DoDataExchange(CDataExchange* pDX); Añade el soporte para el intercambio de datos DDX/DDV entre clases.

virtual void Dump(CDumpContext& dc) const; Libera el contexto de dispositio dc.

void Escala(SAFEARRAY FAR* org,SAFEARRAY FAR* dest); Escala el arreglo origen hacia el arreglo destino para disminuir el tamaño del bitmap en el despliegue de imágenes al 50%.

CScan3DDoc* GetDocument() Lee los atributos del documento de la aplicación.

CLine GetLineXYZ(); Transforma los puntos obtenidos en una línea.

void InitModel(); Crea el modelo necesario para almacenar los puntos de coordenadas 3D.

void intbilin(double * Cx,double * Cy,double *I,double X1,double Y1, int tam); Realiza la interpolación bilineal necesaria en la extracción de características en nivel subpixel. Adicionalmente, crea un archivo de texto con fines de depuración.

int llenaCirc(double * circunfX,double * circunfY); Crea el patrón circular necesario en la definición de elementos de línea del algoritmo de extracción de características.

void LlenaCoefRot(double Rx, double Ry, double Rz); Calcula los parámetros correctos para hacer la rotación, en el proceso de reconstrucción 3D.

int ObtEsqueleto(void); Obtiene las características de la línea de franja que definen la columna vertebral de la imagen como se describió en 2.3.2.

afx_msg void OnCaptura(); Es el manejador del evento por presionar el botón "Image" ya sea capturando la imagen 1 o imagen 2. Captura las imágenes y las almacena en arreglos globales VARIANT como se describió en 2.3.1.

void OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar); Maneja el evento de mover la barra que determina el número de divisiones a realizar.

afx_msg void OnIniciarPT(); Es el manejador del evento por presionar el botón "Start PT" para iniciar o finalizar el despliegue del video en vivo.

void OnInitialUpdate(); Se inicializan la tarjeta de video, y los valores para la barra de progreso.

afx_msg void OnMedir(); Maneja el evento para inicializar y cancelar la aplicación.

afx_msg void OnOpcionesCalibrar(); Maneja el evento del cuadro de diálogo de la calibración.

afx_msg void OnTimer(UINT nIDEvent); Es el manejador del evento que genera un temporizador interno puesto a 500ms para desplegar video en vivo.

afx_msg void OnUpdateOpcionesCalibrar(CCmdUI* pCmdUI); Activa una bandera para poder hacer cambios en la calibración.

afx_msg void OnUpdateOpcionesCalibrarplano(CCmdUI* pCmdUI); Activa una bandera para poder hacer cambios en la calibración.

void PintaPunto(int X,int Y); Superpone un símbolo en forma de punto al arreglo VARIANT que contiene el resultado de la extracción de características.

double Pixel(long x, long y, SAFEARRAY FAR*); Extrae el nivel de intensidad de la imagen en x, y a partir de los datos contenidos en el SAFEARRAY.

void Proceso(); Hace el proceso de tomar las imágenes, restarlas, extraer la línea y las coordenadas de los puntos.

void RestaImg(); Hace la resta algebraica de las imágenes.

void CControlPanel::StartTimerAltaVel(); Inicializa temporizador de alta velocidad.

void CControlPanel::StopTimerAltaVel(); Detiene temporizador de alta velocidad.

void timerProc(); Es el control PID. Funciona con el timer de alta velocidad para controlar el motor.

CCalibra

La clase CCalibra administra el proceso para la calibración de la cámara mediante la caja de diálogo "Calibración de la cámara". Los métodos de la clase se muestran en la figura 3.21 y su descripción se presenta a continuación.

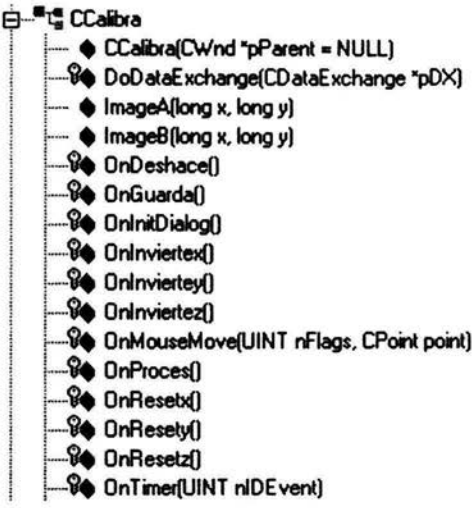


Figura 3.21. Clase CCalibra.

CCalibra(CWnd* pParent = NULL); Es el constructor de la clase. Crea los objetos necesarios para usar el control ActiveX para la captura de las imágenes, así como las variables para la calibración de la cámara y librería PC7266 para el control de la MMC.

virtual void DoDataExchange(CDataExchange* pDX); Permite el intercambio DDX/DDV de datos entre clases.

double ImageA(long x, long y); Toma el valor del pixel (x,y) de la imagen A.

double ImageB(long x, long y); Toma el valor del pixel (x,y) de la imagen B.

afx_msg void OnDeshace(); Elimina el ultimo par de coordenadas ingresados a la lista de la caja de diálogo.

afx_msg void OnGuarda(); Guarda la lista de coordenadas en el archivo de texto out.txt.

virtual BOOL OnInitDialog(); Iniclializa las estructuras para almacenar las imágenes, así como las del control ActiveX de la DT3155 y librería de la PC7266.

afx_msg void OnInviertex(); Manejador del evento del botón invierte X, le cambia el signo a los valores capturados por el puerto de la coordenada X.

afx_msg void OnInviertey(); Manejador del evento del botón invierte Y, le cambia el signo a los valores capturados por el puerto de la coordenada Y.

afx_msg void OnInviertez(); Manejador del evento del botón invierte Z, le cambia el signo a los valores capturados por el puerto de la coordenada Z.

afx_msg void OnMouseMove(UINT nFlags, CPoint point); Manejador del evento cuando el mouse se mueve, muestra las coordenadas de la imagen por donde pasa el mouse.

afx_msg void OnProces(); Manejador del evento del botón Procesar, captura las coordenadas 3D de la MMC, junto con la posición 2D del palpador luminoso.

afx_msg void OnResetx(); Manejador del evento del botón Reset del Eje x, pone en cero el valor de la coordenada X.

afx_msg void OnResety(); Manejador del evento del botón Reset del Eje y, pone en cero el valor de la coordenada Y.

afx_msg void OnResetz(); Manejador del evento del botón Reset del Eje z, pone en cero el valor de la coordenada Z.

afx_msg void OnTimer(UINT nIDEvent); Manejador de los eventos de los temporizadores para actualizar las coordenadas x, y, y z de la MMC.

CPlano

La clase CPlano administra el proceso para la calibración de la cámara mediante la caja de diálogo "Calibración del plano láser". Los métodos de la clase se muestran en la figura 3.22 y su descripción se presenta a continuación.

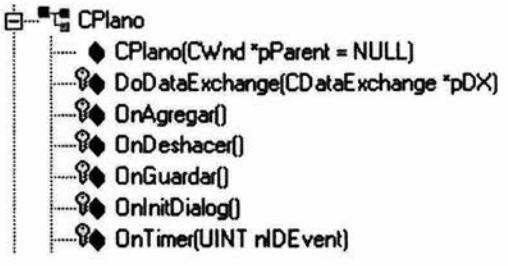


Figura 3.22. Clase CPlano.

CPlano(CWnd* pParent = NULL); Constructor de la clase Plano

virtual void DoDataExchange(CDataExchange* pDX); Permite el intercambio DDX/DDV de datos entre clases.

afx_msg void OnAgregar(); Manejador del evento del botón Agregar. Hace una captura de las coordenadas 3D del palpador.

afx_msg void OnDeshacer(); Manejador del evento del botón Deshacer. Borra la captura hecha previamente.

afx_msg void OnGuardar(); Manejador del evento del botón Guardar. Calcula la ecuación del plano con los tres puntos capturados. Se sobrescribe el archivo de texto plano.txt

virtual BOOL OnInitDialog(); Inicializa las variables para la obtención de las coordenadas 3D para a obtención de la ecuación del pano.

afx_msg void OnTimer(UINT nIDEvent); Manejador de los eventos de los temporizadores para actualizar las coordenadas x, y, y z de la MMC.

3.3.4. Clases y métodos para el despliegue de modelos 3D

La siguiente descripción corresponde a las clases que son utilizadas para el despliegue de modelos 3D bajo el ambiente de programación OpenGL [37].

CVertex

La clase CVertex es un tipo de datos básico que se utiliza en la definición e implementación del modelo 3D que representa las mediciones. La figura 3.23 muestra la clase CVertex.

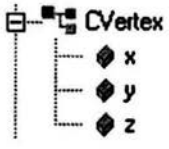


Figura 3.23. Clase CVertex.

En donde los elementos *x*, *y* y *z* representan las coordenadas de un vértice 3D.

CLine

La Clase CLine es un tipo de datos básico que se utiliza en la definición e implementación del modelo 3D que representa las mediciones. La figura 3.24 muestra la clase CLine.

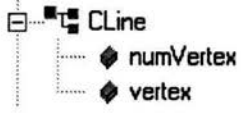


Figura 3.24. Clase CLine.

Un elemento del tipo CLine se compone de un cierto número de vértices, NumVertex, y de un conjunto de vértices, vertex, del tipo CVertex que definen una línea en el espacio.

CMd3Model

La clase CMd3Model es el tipo de datos que define un modelo 3D que representa las mediciones. La figura 3.25 muestra la clase CMd3Model.

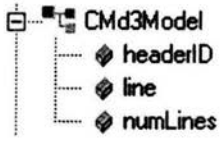


Figura 3.25. Clase CMd3Model.

Un modelo 3D esta compuesto por los siguientes elementos. Un texto de encabezado con propósitos de identificación, headerID, elementos de línea del tipo CLine, y por el número de elementos de línea que completan el modelo, numLines.

En forma adicional, los siguientes métodos se utilizan para manipular los modelos 3D.

BOOL LoadMd3(CArchive &input, CMd3Model *m); Carga el modelo de línea nativo del programa, abre este tipo de archivo.

BOOL SaveMd3(CArchive &output, CMd3Model *m); Guarda el archivo, la medición realizada, con el formato de archivo del programa.

void DrawMd3Wire(CMd3Model *m); Dibuja el modelo de la medición de acuerdo al formato de archivo.

COpenGLWnd

La Clase COpenGLWnd de la figura 3.26 es la encargada de crear el entorno OpenGL, inicializa todas las propiedades de OpenGL, necesarias para poder mostrar la escena 3D de la medición. Los métodos se describen a continuación:

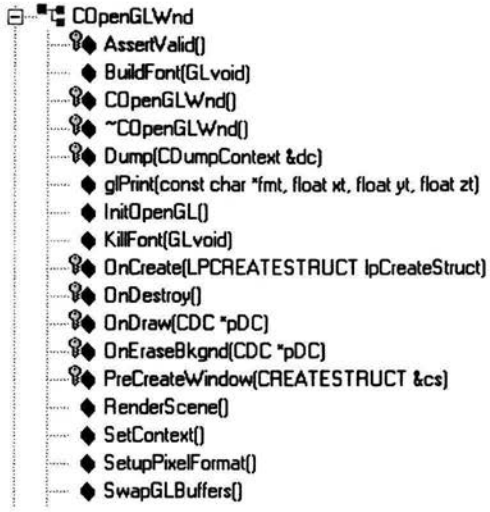


Figura 3.26. Clase COpenGLWnd.

virtual void AssertValid() const; Método heredado para verificar la creación de la clase.

GLvoid BuildFont(GLvoid); Método que construye el formato de letra utilizado en el ambiente.

COpenGLWnd(); Constructor protegido, usado por una creación dinámica.

~COpenGLWnd(); Destructor de la clase.

virtual void Dump(CDumpContext& dc) const; Método heredado para liberar el contexto del dispositivo.

GLvoid glPrint(const char *fmt,float xt,float yt,float zt); Método para desplegar un texto en el ambiente 3D.

BOOL InitOpenGL(); Inicializa los parámetros OpenGL necesarios.

GLvoid KillFont(GLvoid); Destruye el formato de la letra en el ambiente.

afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct); Responde al evento de la creación de la ventana para OpenGL.

afx_msg void OnDestroy(); Responde al evento de la destrucción de la ventana del ambiente OpenGL.

virtual void OnDraw(CDC* pDC); Sobrecarga de este método para el despliegue de esta vista.

afx_msg BOOL OnEraseBkgnd(CDC* pDC); Responde al evento del borrado de la pantalla de OpenGL.

virtual BOOL PreCreateWindow(CREATESTRUCT& cs); Método para validar algunos parámetros necesarios para la ventana en la que se despliega OpenGL.

void RenderScene(); Hace los cálculos necesarios para poder visualizar la escena tridimensional en la pantalla bidimensional.

void SetContext(); Configura la ventana para el despliegue.

BOOL SetupPixelFormat(); Método para configurar propiedades del despliegue.

void SwapGLBuffers(); Cambia los valores de los elementos de la escena 3D de acuerdo al movimiento que se tenga en el entorno.

CPerspective

La Clase CPerspective como se muestra en la figura 3.27, es la encargada de hacer los cambios necesarios en la vista OpenGL cuando suceden eventos del usuario sobre la ventana OpenGL, como son los movimientos del ratón y del teclado para mover la escena, rotarla y hacer acercamientos. Los métodos se describen a continuación.

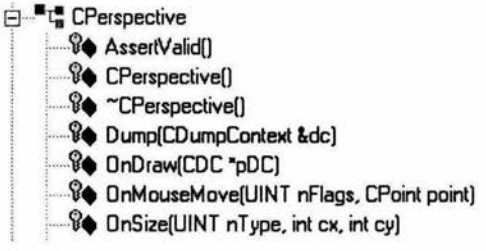


Figura 3.27. Clases CPerspective.

virtual void AssertValid() const; Valida la creación de la clase.

CPerspective(); Es el constructor de la clase. Inicializa las variables para visualizar la escena.

virtual ~CPerspective(); Es el destructor de la clase.

virtual void Dump(CDumpContext& dc) const; Libera el contexto del dispositivo.

virtual void OnDraw(CDC* pDC); Sobrecarga este método para determinar como va a pintar sobre la pantalla.

afx_msg void OnMouseMove(UINT nFlags, CPoint point); Manejador del evento que responde al movimiento del mouse, es aquí

donde se configuran los eventos para poder manipular el objeto 3D, como es rotarlo, hacer zoom, desplazarlo,etc.

afx_msg void OnSize(UINT nType, int cx, int cy); Manejador del evento de cambiar el tamaño de la pantalla.

Capítulo 4

Resultados y conclusiones

En el capítulo tres, presentamos los pasos que se siguen para operar el instrumento de medición, lo que proporciona una idea cualitativa del sistema para la medición por visión. En el presente capítulo evaluamos de manera cuantitativa la medición al estimar el error que se obtiene al aplicar el proceso descrito a un patrón dimensional. En la estimación del error utilizamos dos enfoques: virtual y real. En el enfoque virtual prescindimos del hardware utilizado con el objeto de evaluar aisladamente el desempeño de los algoritmos de medición. Por otra parte, en el enfoque real evaluamos el sistema de medición en su conjunto, incluyendo los algoritmos de procesamiento de imágenes, medición y control operando hardware en tiempo real. Como parte final de los resultados presentamos la medición de una geometría 3D y discutimos su desempeño.

4.1. Resultados

Diseñamos dos experimentos que consisten en medir un objeto conocido, tanto virtual como real, para estimar la exactitud de la medición por visión. El primer experimento nos permitió orientar la implementación final del segundo experimento. Una vez comprobada la exactitud del instrumento, ésta puede ser extrapolada a la medición de una geometría 3D utilizando las capacidades completas de medición y automatización del instrumento final.

4.1.1. Resultados de la medición virtual

En la medición virtual aplicamos el procedimiento descrito en 2.4 a una esfera patrón generada en un ambiente virtual para comparar sus

características conocidas a priori contra aquellas que se obtienen al medir el objeto por visión. La figura 4.1 muestra la esfera patrón generada en un ambiente virtual con radio 25mm y posición del centro en (25, 25, 25)mm. El ambiente fue generado en el programa de traza de rayos PovRay listado en el anexo A.2.1. El ambiente nos permite manipular primitivas geométricas 3D y obtener su proyección en un plano imagen mediante el listado en el anexo A.2.2.

Figura 4.1. Esfera patrón virtual.

Aplicamos el algoritmo de medición por visión descrito en 2.4 para determinar el meridiano 3D que se superpone a la esfera de la figura 4.2.

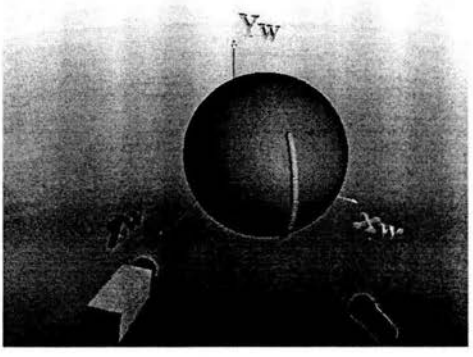


Figura 4.2. Reconstrucción de la esfera patrón virtual.

Los puntos 3D reconstruidos de la figura 4.2 son ajustados a la ecuación de la esfera mediante la minimización de la siguiente función de error.

$$\gamma(h,j,k,a) = \sum_{i=1}^{N}\left[(x_i-h)^2+(y_i-j)^2+(z_i-k)^2-a^2\right]^2 \qquad (4.1)$$

en donde (h, j, k) son las coordenadas del centro de la esfera, a es el radio de la esfera y (x_i, y_i, z_i) son los N puntos 3D reconstruidos a partir de la medición por visión.

La minimización de la ecuación (4.1) se implementa en MatLab mediante el código fuente listado en los anexos A.1.1 y A.2.2. La tabla 4.1 resume los resultados del mejor ajuste obtenido.

	h	*j*	*k*	*a*
Valor nominal (mm)	25	25	25	25
Ajuste no lineal (mm)	25.0685	24.7928	25.0026	25.0603

Tabla 4.1. Ajuste a esfera.

En el ajuste se inicia con un error residual de 9 350.348mm^2 y en el mejor ajuste se consigue 702.55mm^2.

4.1.2. Resultados de la medición real

En la medición real aplicamos el procedimiento descrito en 2.4 a una esfera patrón usando el arreglo de la figura 4.3 y hardware con operación en tiempo real para comparar sus características conocidas mediante MMC contra aquellas que se obtienen al medir el objeto por visión. La figura 4.3 muestra la esfera patrón con radio 11.115 mm y posición del centro en (23.6288, 13.9268, 39.6144)mm.

Figura 4.3. Esfera patrón real.

Aplicamos el algoritmo de medición descrito en 2.4 para determinar un meridiano 3D de la figura 4.4.

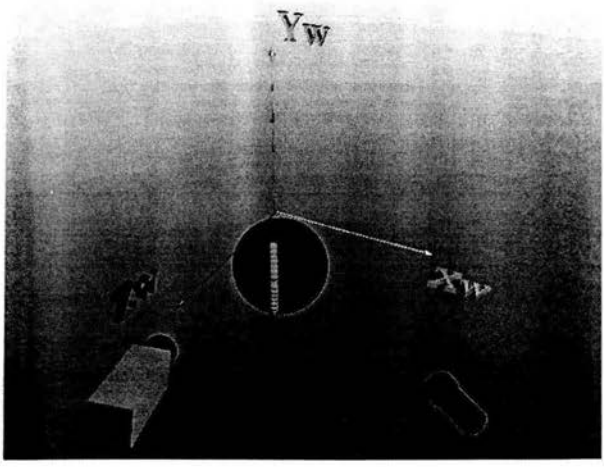


Figura 4.4. Reconstrucción de la esfera patrón real.

Los puntos 3D reconstruidos de la figura 4.4 son ajustados a la ecuación de la esfera mediante la minimización de la función de error en (4.1).

La minimización de la ecuación (4.1) se implementa en MatLab con el código mostrado

en los anexos A.1.1 y A.1.2. La tabla 4.2 resume los resultados del mejor ajuste obtenido.

	h	*j*	*k*	*a*
Valor nominal (mm)	23.6288	13.9253	39.6165	11.115
Ajuste no lineal (mm)	23.6204	13.8511	39.6298	11.146

Tabla 4.2. Ajuste a esfera.

En el ajuste se inicia con un error residual de 142.316714mm^2 y en el mejor ajuste se consigue 123.526959mm^2.

4.1.3. Medición de una geometría libre

En la medición por visión de una geometría libre se utiliza el conjunto de algoritmos implementados en el software de propósito particular "Scan3D" y el hardware con operación en tiempo real. La configuración final del instrumento para la medición por visión se muestra en la figura 4.5.

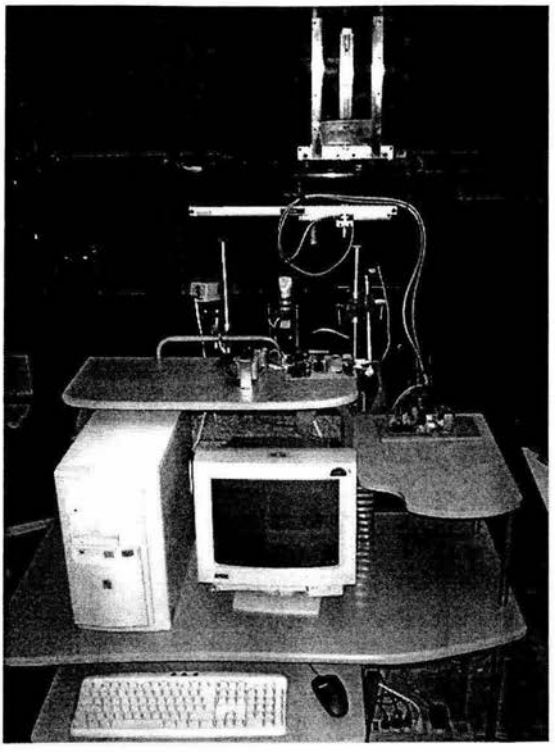
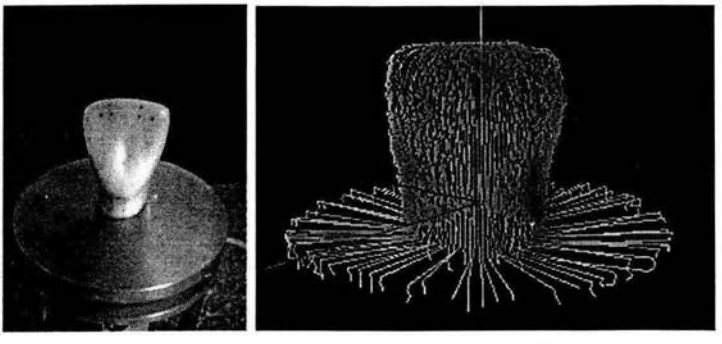


Figura 4.5. Instrumento para la medición por visión utilizando luz estructurada.

La medición 3D del objeto mostrado (un modelo de un diente) en la figura 4.6.a resulta en el modelo de la figura 4.6.b.



(a) (b)

Figura 4.6. Medición usando el sistema de medición por visión. (a) Objeto 3D. (b) Modelo obtenido.

La medición consiste de 100 meridianos y se llevó a cabo en aproximadamente 200s. Por otra parte, según la estimación de la exactitud del sistema descrita en las secciones 4.1.1 y 4.1.2, en la presente medición obtenemos las características de la tabla 4.3.

Volumen de medición (mm)	25, 25, 25
Error máximo (mm)	0.0742

Tabla 4.3. Características del sistema de medición por visión.

4.2. Conclusiones

Se presentó un método para la medición 3D usando un sistema de visión por luz estructurada con alto grado de automatización. La medición en ambientes virtuales nos permite depurar los algoritmos y orientar el desarrollo de herramientas auxiliares. Por otra parte, con la medición en condiciones reales, ponemos en práctica nuestros planteamientos y evaluamos la fiabilidad de aplicar el procedimiento a situaciones específicas.

La medición usando visión puede ser de gran utilidad en las siguientes situaciones:

- En aplicaciones que por su exactitud no requieran del uso de MMC o SLM
- En situaciones en donde la complejidad de los instrumentos MMC o SLM impidan medir confiablemente. Por ejemplo, en un objeto que no facilita el contacto contra el palpador de una MMC o que no permite formar el arreglo óptico necesario para un SLM.
- En mediciones que requieren una alta tasa de muestreo. Por ejemplo, en mediciones en una línea de producción.
- El instrumento posee un alto grado de automatización e interfase amigable con el usuario. El sistema presentado es capaz de realizar mediciones con una gran exactitud en poco tiempo y con una infraestructura de bajo costo. La exactitud del sistema radica en los siguientes aspectos:
- El uso de Máquina de Medir por Coordenadas como patrón de longitud. (dimensional).
- La extracción de características con exactitud subpixel en los algoritmos de visión por computadora.
- El empleo de modelos no lineales de cámara como parte de los procesos de calibración y reconstrucción.

Se presentó un avance de este proyecto en el Congreso Iberoamericano de Reconocimiento de Patrones en La Habana, Cuba, en el mes de Noviembre de 2003, integrando las memorias del Congreso y la publicación en Lecture Notes in Computer Science 2905 [17].

Finalmente, el proyecto utilizó el financiamiento por parte del proyecto PAPIT IN11100 "Utilización de la visión estéreo para mediciones geométricas precisas sin contacto".

4.3. Trabajo a futuro

Los resultados y experiencia obtenida dan la pauta para dos líneas de trabajo: La primera consiste en mejorar la exactitud del sistema al incorporar hardware con capacidades superiores. Por ejemplo, se puede sustituir la cámara actual con resolución de 640X480 por una de 1620X1236. También el láser de línea actual puede ser mejorado con respecto al grosor del patrón generado. En este sentido, el trabajo se enfocaría a implementar el software necesario para soportar el hardware mencionado. Por otra parte, la segunda línea de trabajo consistiría en mejorar la exactitud de la implementación actual del instrumento. Para ello se buscaría mejorar la calidad del modelo 3D al correlacionar múltiples mediciones del mismo objeto en diferentes posiciones.

Anexo

Código fuente

El presente capítulo contiene el código fuente utilizado en la evaluación de los resultados presentados en el capítulo 4.

A.1. MatLab

Los programas desarrollados en MatLab contienen principalmente los algoritmos de procesamiento de imágenes y cálculos numéricos.

A.1.1. AjustaEsfera.m

```
close all
clear all

%Read global coordinates
fid=fopen('3d.txt','r');
a = fscanf(fid,'%f, %f, %f,',[3 inf]);
xw=a(1,1:size(a,2));
yw=a(2,1:size(a,2));
zw=a(3,1:size(a,2));
fclose(fid);

N=size(xw,2);

x=[25 25 25 25];
error0=0;
for i=1:N,
    error(i)=((xw(i)-x(1))^2+(yw(i)-x(2))^2+(zw(i)-x(3))^2-x(4)^2)^2;
    error0=error0+error(i);
end
res=esfera(x,xw,yw,zw);
res0=sum(res(1,1:N).^2,2);
fprintf('\nError inicial, error0=%10.6f
error1=%10.6f\n\n',res0,error0)

x0 = [25 25 25 25];     % Starting guess
options=optimset('LargeScale','on','Display','final','TolFun',0.001);
[x1,resnorm]=lsqnonlin(@esfera,x0,-1000,1000,options,xw,yw,zw);
```

```
res=esfera(x1,xw,yw,zw);
res1=sum(res(1,1:N).^2,2);
errorf=0;
for i=1:N,
    error(i)=((xw(i)-x1(1))^2+(yw(i)-x1(2))^2+(zw(i)-x1(3))^2-
x1(4)^2)^2;
    errorf=errorf+error(i);
end
fprintf('\nError final, error0=%10.6f error1=%10.6f
error2=%10.6f\n\n',resnorm,res1,errorf)
```

A.1.2. Esfera.m

```
function F=esfera(x, xw, yw, zw)
F=(xw-x(1)).^2+(yw-x(2)).^2+(zw-x(3)).^2-x(4).^2;
```

A.2. PovRay

El código PovRay contiene principalmente el código para el despliegue gráfico 3D.

A.2.1. Ambiente3D.pov

```
// Example for calibration and reconstruction
// File: 3D.pov
// Vers: MegaPOV 0.6

#version 3.1;
#include "colors.inc"

global_settings{assumed_gamma 1.8}

sky_sphere{
  pigment{
    gradient y
    color_map { [0.0 color blue 0.6] [1.0 color rgb 1] }
  }
}

// CAMERAS

//Global camera
camera {
  perspective
  location <80, 100, 140>
  right <-4/3, 0, 0>
  up <0, 1, 0>
  direction <1, 0, 0>
  look_at <10, -4, 0>
  angle 52
}

/*Left camera settings
camera {
  perspective
  location <35, 40, 100>
  right <-4/3, 0, 0>
  up <0, 1, 0>
  direction <1, 0, 0>
  look_at <0, 8, 0>
  angle 70
}
*/
// LIGHTS
```

```
// Ambient light
light_source {<100, 100, 100> color Gray}

// Laser light
#declare Count=-60;
#while (Count < 60)
  light_source {<70, 25, 70>
      color Red
      cylinder
      radius 0.1
      point_at <0, Count*1, 0>
      tightness 0
      falloff 1
      jitter
  }
  #declare Count=Count+1.0;
#end

// TEXTURES

// Red
#declare My_Texture_1 =
texture {
   pigment {
       color red 1 green 0 blue 0
   }
   finish {
       ambient .1
       diffuse .1
       specular 1
       roughness .001
       reflection .75
       metallic
   }
}

// Green
#declare My_Texture_2 =
texture {
  pigment {
    color red 0 green 1 blue 0
  }
  finish {
    diffuse 0.5
    phong 0.5
    phong_size 3
    reflection 0.5
  }
}

// Blue
#declare My_Texture_3 =
texture {
  pigment {
    color red 0.2 green 0.2 blue 1
  }
  finish {
    diffuse 0.02
    phong 0.5
    phong_size 3
    reflection 0.5
    metallic
  }
}

// Yellow
#declare My_Texture_4 =
```

```
texture {
  pigment {
    color red 1 green 1 blue 0
  }
  finish {
    diffuse 0.5
    phong 0.5
    phong_size 3
    reflection 0.5
  }
}

// Gray
#declare My_Texture_5 =
texture {
  pigment {
    color red 0.5 green 0.5 blue 0.5
  }
  finish {
    diffuse 0.02
    phong 0.5
    phong_size 3
    reflection 0.01
  }
}

// Other Red
#declare My_Texture_6 =
texture {
  pigment { Red filter 1 }
  finish {
    ambient .1
    diffuse .1
    reflection .2
    specular 1
    roughness .001
    irid {
      0.35
      thickness .5
      turbulence .5
    }
  }
}

// OBJECTS

// Camera
#declare My_Camera=
union {
  box{ <0, 0, 13>, <5, 5, 0> pigment {White}}
  cylinder { <2.5, 2.5, 0>, <2.5, 2.5, -0.7> 1.7 pigment {Yellow}}
  cylinder { <2.5, 2.5, -0.7>, <2.5, 2.5, -3.7> 1.7 pigment {Black}}
  cylinder { <2.5, 2.5, -3.7>, <2.5, 2.5, -4.2> 2.1 pigment {Black}}
}

// Laser diode
#declare My_LaserDiode=
union {
  cylinder { <0, 0, 8>, <0, 0, 0> 2 pigment {Black}}
  cylinder { <0, 0, 0>, <0, 0, -2> 2.5 pigment {Black}}
  rotate <0, 45, 0> rotate <-19.657, 0, 0> translate <70, 25, 70>
}

// Cup
#declare My_Cup=
  sor {
    8,
```

```
    <0.0,   0.1>
    <3.0,   0.2>
    <1.0,   0.3>
    <0.5,   0.4>
    <0.5,   4.0>
    <1.0,   5.0>
    <3.0,  10.0>
    <4.0,  11.0>
    sturm
    scale <4, 4, 4>
    translate <25, 0, 25>
    texture {My_Texture_3}
}

// Base planes
#declare My_Planes=
  mesh {
    // XY plane
    triangle { <0, 0, 0>, <80, 80, 0>, <80, 0, 0>}
    triangle { <0, 0, 0>, <80, 80, 0>, <0, 80, 0>}
    // XZ plane
    triangle { <0, 0, 0>, <0, 0, 80>, <80, 0, 80>}
    triangle { <0, 0, 0>, <80, 0, 80>, <80, 0, 0>}
    // YZ plane
    triangle { <0, 0, 0>, <0, 80, 0>, <0, 80, 80>}
    triangle { <0, 0, 0>, <0, 0, 80>, <0, 80, 80>}
    texture {My_Texture_5}
}

// Laser plane
#declare My_Laser_Plane=
  mesh {
    triangle { <70, 25, 70>, <0, 50, 0>, <0, -50, 0>}
    texture {My_Texture_6}
}

// DRAW AXIS OF WORLD SYSTEM COORDINATE

// X Arrow (Yellow)
cone {
    <50, 0, 0>, 0
    <48, 0, 0>, 1
    texture { My_Texture_4 }
}
cylinder { <0,0,0>, <50,0,0>, 0.3 texture { My_Texture_4 }}
text { ttf "timrom.ttf" "Xw" 0.15, 0 texture {My_Texture_4}
  scale 10 rotate <-90,0,0> translate <53, 1, 15>
}

// Y Arrow (Green)
cone {
    <0, 50, 0>, 0
    <0, 48, 0>, 1
    texture { My_Texture_2 }
}
cylinder { <0,0,0>, <0,50,0>, 0.3 texture { My_Texture_2 }}
text { ttf "timrom.ttf" "Yw" 0.15, 0 texture {My_Texture_2}
  scale 10 translate <3, 53, 1>
}

// Z Arrow (Blue)
cone {
    <0, 0, 50>, 0
    <0, 0, 48>, 1
    texture { My_Texture_3 }
}
cylinder { <0,0,0>, <0,0,50>, 0.3 texture { My_Texture_3 }}
```

```
text { ttf "timrom.ttf" "Zw" 0.15, 0 texture {My_Texture_3}
  scale 10 rotate <0,90,0> translate <3, 15, 73>
}

// DRAW OBJECTS

object {
  My_Camera rotate <0, 19.2886, 0> rotate <-17.7435, 0, 0> translate
<35, 40, 100>
}
My_LaserDiode
My_Cup
//My_Planes
My_Laser_Plane

// Master of calibration
sphere {<0,0,0>,1 pigment {White}}
sphere {<0,25,25>,1 pigment {White}}
sphere {<0,13,50>,1 pigment {White}}
sphere {<25,10,0>,1 pigment {Red}}
sphere {<25,5,25>,1 pigment {Red}}
sphere {<25,40,50>,1 pigment {Red}}
sphere {<50,0,50>,1 pigment {White}}
sphere {<50,38,25>,1 pigment {White}}
sphere {<50,50,0>,1 pigment {White}}


// Objet under inspection
//sphere {<25,25,25>,1 pigment {Yellow}}
```

A.2.2. GenFranja.pov

```
// Example for calibration and reconstruction
// File: 3D.pov
// Vers: MegaPOV 0.6

#version 3.1;
#include "colors.inc"

global_settings{assumed_gamma 1.8}

// CAMERAS

//Right camera settings
/*camera {
  perspective
  location <100, 40, 35>
  right <-4/3, 0, 0>
  up <0, 1, 0>
  direction <1, 0, 0>
  look_at <0, 8, 0>
  angle 70
}
*/
//Left camera settings
camera {
  perspective
  location <35, 40, 100>
  right <-4/3, 0, 0>
  up <0, 1, 0>
  direction <1, 0, 0>
  look_at <0, 8, 0>
  angle 70
}

// LIGHTS

// Ambient light
```

```
light_source {<100, 100, 100> color Gray}

// Laser light
#declare Count=-60;
#while (Count < 60)
  light_source {<70, 25, 70>
      color White
      cylinder
      radius 0.1
      point_at <0, Count*1, 0>
      tightness 0
      falloff 1
      jitter
  }
  #declare Count=Count+1.0;
#end

// TEXTURES

// Red
#declare My_Texture_1 =
texture {
   pigment {
       color red 1 green 0 blue 0
   }
   finish {
       ambient .1
       diffuse .1
       specular 1
       roughness .001
       reflection .75
       metallic
   }
}

// Green
#declare My_Texture_2 =
texture {
  pigment {
    color red 0 green 1 blue 0
  }
  finish {
    diffuse 0.5
    phong 0.5
    phong_size 3
    reflection 0.5
  }
}

// Blue
#declare My_Texture_3 =
texture {
  pigment {
    color red 0.2 green 0.2 blue 1
  }
  finish {
    diffuse 0.02
    phong 0.5
    phong_size 3
    reflection 0.5
    metallic
  }
}

// Yellow
#declare My_Texture_4 =
texture {
```

```
  pigment {
    color red 1 green 1 blue 0
  }
  finish {
    diffuse 0.5
    phong 0.5
    phong_size 3
    reflection 0.5
  }
}

// Gray
#declare My_Texture_5 =
texture {
  pigment {
    color red 0.5 green 0.5 blue 0.5
  }
  finish {
    diffuse 0.02
    phong 0.5
    phong_size 3
    reflection 0.01
  }
}

// Other Red
#declare My_Texture_6 =
texture {
  pigment { Red filter 1 }
  finish {
    ambient .1
    diffuse .1
    reflection .2
    specular 1
    roughness .001
    irid {
      0.35
      thickness .5
      turbulence .5
    }
  }
}

// OBJECTS

// Cup
#declare My_Cup=
  sor {
    8,
    <0.0,   0.1>
    <3.0,   0.2>
    <1.0,   0.3>
    <0.5,   0.4>
    <0.5,   4.0>
    <1.0,   5.0>
    <3.0,  10.0>
    <4.0,  11.0>
    sturm
    scale <4, 4, 4>
    translate <25, 0, 25>
    texture {My_Texture_3}
}

// Base planes
#declare My_Planes=
  mesh {
    // XY plane
```

```
    triangle { <0, 0, 0>, <80, 80, 0>, <80, 0, 0>}
    triangle { <0, 0, 0>, <80, 80, 0>, <0, 80, 0>}
    // XZ plane
    triangle { <0, 0, 0>, <0, 0, 80>, <80, 0, 80>}
    triangle { <0, 0, 0>, <80, 0, 80>, <80, 0, 0>}
    // YZ plane
    triangle { <0, 0, 0>, <0, 80, 0>, <0, 80, 80>}
    triangle { <0, 0, 0>, <0, 0, 80>, <0, 80, 80>}
    texture {My_Texture_5}
}

// Laser plane
#declare My_Laser_Plane=
  mesh {
    triangle { <0, 0, 0>, <100, 100, 100>, <100, 0, 100>}
    triangle { <0, 0, 0>, <100, 100, 100>, <0, 100, 0>}
    texture {My_Texture_6}
}


// DRAW AXIS OF REFERENCE SIYSTEM COORDINATE

// X Arrow (Yellow)
cone {
    <50, 0, 0>, 0
    <48, 0, 0>, 1
    texture { My_Texture_4 }
}
cylinder { <0,0,0>, <50,0,0>, 0.3 texture { My_Texture_4 }}

// Y Arrow (Green)
cone {
    <0, 50, 0>, 0
    <0, 48, 0>, 1
    texture { My_Texture_2 }
}
cylinder { <0,0,0>, <0,50,0>, 0.3 texture { My_Texture_2 }}

// Z Arrow (Blue)
cone {
    <0, 0, 50>, 0
    <0, 0, 48>, 1
    texture { My_Texture_3 }
}
cylinder { <0,0,0>, <0,0,50>, 0.3 texture { My_Texture_3 }}

// DRAW OBJECTS

My_Cup
My_Planes
//My_Laser_Plane

// Master of calibration
sphere {<0,0,0>,1 pigment {White}}
sphere {<0,25,25>,1 pigment {White}}
sphere {<0,13,50>,1 pigment {White}}
sphere {<25,10,0>,1 pigment {Red}}
sphere {<25,5,25>,1 pigment {Red}}
sphere {<25,40,50>,1 pigment {Red}}
sphere {<50,0,50>,1 pigment {White}}
sphere {<50,38,25>,1 pigment {White}}
sphere {<50,50,0>,1 pigment {White}}


// Objet under inspection
sphere {<25,25,25>,1 pigment {Yellow}}
```

Bibliografía

[1]. R. Armes, & R. Kasturi, *Machine Vision*, Mc Graw Hill, USA; 1995.

[2]. G. Bermúdez, B. Valera, J. Sánchez, R. Nava, "Interfase para una máquina de medir por coordenadas", *XIV Congreso de Instrumentación SOMI*, Tonantzintla Puebla, 1999, pp 565-569.

[3]. F. J. Ceballos, *Microsoft Visual C++: Aplicaciones para Win32*, 2ª edición, Ra-Ma, Marzo 2003.

[4]. Data Translation, "SDK DT-Active Open Layers User's Manual", Data Translation, USA; 1999, pp 142.

[5]. Data Translation, "Mach Series DT3155 User Manual", Data Translation, USA; 1995, pp 58.

[6]. M. Ercole, "Máquinas de medir tridimensionales", http://www.metalunivers.com/1pm/pm02/mmt/MMT.htm, 29 de junio de 2004.

[7]. F. J. Escarpa, "Introducción a los sistemas de medición tridimensional con láser", http://concretonline.com/jsp/articulos/construccion15.jsp, 29 de junio de 2004.

[8]. G. Farin, *Curves and surfaces for CAGD*, Academic Press, NY; 1997 pp 429.

[9]. R. C. González y R. E. Woods, *Tratamiento digital de imágenes*, Addison-Wesley/ Diaz de Santos, USA; 1996, pp 175.

[10]. K. Harris, S. N. Efstratiadis, N. Maglaveras, J. Gourassas, C. Pappas, G. Louridas, "Coronary Arterial Tree Extraction based on Artery Tracking and Mathematical Morphology", *Computers in Cardiology*, vol. 25, 1998, pp 769-772.

[11]. G. Hu, G. Stockman, "3-D Surface Solution Using Structured Light and Constraint Propagation", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 11, abril 1989, pp. 390-402.

[12]. Intertronic Internacional, "Sistema láser de medición absoluto", http://www2.intertronic.es/Descargas/Mailings/le100tresp.pdf, 10 de agosto de 2003.

[13]. M. A. G. Izquierdo, M. T. Sanchez, A. Ibañez, L. G. Ullate "Sub-pixel measurement of 3D surfaces by laser scanning", *Sensors and Actuators*, vol. 76, 1999, pp 1-8.

[14]. J. J. Jiménez, E. Rodríguez, *Visión estéreo y estimación de movimiento*, Tesis de licenciatura, Facultad de Ingeniería, UNAM, México D.F, Noviembre de 2000.

[15]. D. J. Kruglinski, *Progrese con Visual C++*, McGrawHill, 2000.

[16]. H. D. Martínez, F. Sotelo, *Metrología dimensional utilizando visión estéreo*, Tesis de licenciatura, Facultad de Ingeniería, UNAM, México D.F., Noviembre de 1999.

[17]. U. Martínez, B. Valera, J. Sánchez & V. García, "Vision system for subpixel laser stripe profile extraction with real time operation", *8th Iberoamerican congress on pattern recognition, CIARP 2003 LNCS 2905*, Habana, Cuaba, November 2003, pp 38-45.

[18]. MetalUnivers, "Metrología dimensional", http://www.metalunivers.com/Arees/metrologiadimensional, 29 de junio de 2004.

[19]. H. Nicolas, *Hiérarchie de modéles de mouvement et méthodes d'estimation associées: application au codage de séquences d'images*, PhD thesis Univerité de Rennes, Rennes France; 1992.

[20]. C. H. Pappas, William H. Murray, *Visual C++ 6.0 Manual de Referencia*, McGrawHill, 1999.

[21]. W. H. Press et al, *Numerical Recipes in C*, Cambridge University Press, USA; 1995, pp 994.

[22]. H. Rios, "Aplicaciones de la visión por computadora", http://www.lania.mx/biblioteca/newsletters/1999-primavera-verano/aplicaciones.html, 29 de junio de 2004

[23]. F. Rogers, J. A. Adams, *Mathematical Elements for Computer Graphics*, McGraw Hill, NY; 1990.

[24]. T. Shen, J. Huang, C. Menq. "Multiple-Sensor Integration for Rapid and High-Precision Coordinate Metrology", *IEEE/ASME Trans. On Mechatronics*, vol. 5, june 2000, pp 110-121.

[25]. StockerYale Inc., *Diode Laser Products Manual SNF-501/SNF-701 DLS-500 Series*, StockerYale, Québec; 1999, pp 38.

[26]. J. Tang, "PID Controller Using the TMS320C31 DSK with On-line Parameter Adjustment for Real-time DC Motor speed and Position Control", *Proceedings of the IEEE International Symposium on Industrial Electronics*, 2001, pp 786-791.

[27]. Toolmart, "Stabila LE 100 laser distance measuring kit", http://www.toolmarts.com/stab_le100.html, 29 de Junio de 2004.

[28]. Trimble, "Callidus 3D laser scanner", http://www.trimble.com/callidus.html, 29 de junio de 2004.

[29]. R. Y. Tsai, "An Efficient and Accurate Camera Calibration Technique for 3D Vision", *Proceedings of IEEE Conference on Comp and Pattern Recognition*, Miami Beach, FL, 1986, pp 364-374.

[30]. US Digital Co., *PC7266 PC to Incremental Encoder Interface Card*, US Digital, Tuesday June 24 2003.

[31]. B. Valera, J. Sánchez, S. Padilla, 2000, "Computer Vision in dimensional Metrology", *Instrumentation & Development*, vol. 4, num. 4, 2000, pp 67-73.

[32]. B. Valera, J. Sánchez, V. García, "Segmentación de imágenes de franjas en tiempo real", *XVII Congreso de Instrumentación SOMI*, Mérida Yucatán, 14 a 18 de Octubre 2002, pp ref. 17BVO302.

[33]. B. Valera, J. Sánchez, "Evaluación de un sistema de luz estructurada para la medición 3D sin contacto", *XVI Congreso de Instrumentación SOMI*, Querétaro Querétaro, 15 a 19 de Octubre 2001, pp ref. MET-2.

[34]. B. Valera, S. Padilla, "Implementación de un controlador PID digital", *Informe técnico CCADET UNAM*, México D. F., Octubre de 2002, pp 50.

[35]. J. Weng, P. Cohen and M. Herniou, "Camera Calibration with Distortion Models and Accuracy Evaluation", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, num. 10, 1992, pp 965-980.

[36]. R. Willson, "Tsai camera calibration software",

www.ius.cs.cmu.edu/afs/cs.cmu.edu/user/rgw/www/TsaiCode.html, 29 de junio de 2004

[37]. R. S. Wright, M. Sweet, *OpenGL Super Bible*, White Group Press, USA; 2000, pp 696.

[38]. N. Zuech, (1988), *Undesrstanding and applying machine vision*, Marcel Dekker, NJ; 2000, pp 416.