Universidad Nacional de La Plata

Facultad de Informática

Segmentación espectral de imágenes obtenidas con cámaras de tiempo de vuelo

Tesina de Licenciatura en Informática

Alumno: Lorenti, Luciano Rolando

Director: Giacomantone, Javier Oscar

Índice general

Capítulo 1

Introducción

El propósito de un método de segmentación es descomponer una imagen en sus partes constitutivas. La segmentación es generalmente la primera etapa en un sistema de análisis de imágenes, y es una de las tareas más críticas debido a que su resultado afectará a las etapas siguientes. Se han propuesto un gran número de técnicas y algoritmos para realizar ésta tarea. Algoritmos de visión por computador, en particular de segmentación, que han sido utilizados con éxito en ambientes industriales, con colores e iluminación controlada, no obtienen resultados similares en contextos diferentes. Una alternativa para abordar problemas en que las condiciones de contorno no permiten una segmentación adecuada es incorporar información de profundidad, es decir, la distancia a la que se encuentran los objetos que conforman la escena respecto al dispositivo de captura. Los principales métodos ópticos para obtener mediciones de rango de una escena son: interferometría, triangulación y tiempo de vuelo [1].

Las mejoras realizadas en el campo de las tecnologías de escaneo 3D han hecho posible que las cámaras basadas en el principio de tiempo de vuelo sean más accesibles [2]. Las cámaras de tiempo de vuelo son dispositivos de captura que generan simultáneamente imágenes en tonos de grises e información 3D de la escena. La cámara utiliza su propia fuente de iluminación en forma de una matriz de diodos emisores de luz infrarroja modulada en amplitud. A partir de las ondas reflejadas por los objetos, dos imágenes son formadas: una imagen de distancia (rango), que es calculada a partir de la diferencia de fase entre señal emitida y la reflejada en cada píxel, y una imagen de intensidad, que es calculada a partir de la amplitud de la señal reflejada. Las principales ventajas con respecto a otras técnicas de medición 3D es la posibilidad de obtener

imágenes a velocidades compatibles con aplicaciones de tiempo real, y la posibilidad de obtener nubes de puntos 3D desde un sólo punto de vista [3] [4].

Mientras la literatura sobre la segmentación de escenas basadas en información de color o niveles de intensidad es extremadamente vasta, el número de trabajos que intentan segmentar utilizando las dos fuentes de información es aún limitada. Recientemente han sido propuestas técnicas para segmentar objetos que operan sobre imágenes de rango e intensidad con el objetivo de definir bordes más precisos en presencia, tanto de ruido como de oclusiones en distintos planos [2] [5].

La fusión de la información de profundidad junto con la información de intensidad permite obtener descripciones de las escenas que tienen en cuenta tanto la geometría de los objetos como la información de luminancia. En éste contexto, la segmentación de imágenes consiste en utilizar algoritmos que utilicen ambas fuentes de información y no sólo los niveles de intensidad. Con esta perspectiva el problema de segmentación puede ser formulado como la búsqueda de formas efectivas para particionar adecuadamente un conjunto de muestras con información de intensidad y distancia.

Un primer acercamiento al problema consiste en realizar dos procesos de segmentación independientes, uno aplicado a la imagen de intensidad y el otro aplicado a la imagen de distancia, y luego combinar los resultados obtenidos. Una segunda alternativa consiste en considerar a cada punto como un patrón de dimensión superior que contiene las coordenadas geométricas del punto en el espacio y la información de intensidad.

Recientemente han sido propuestos diversos algoritmos de agrupamiento, tanto jerárquicos como particionales, para abordar el problema de segmentar objetos en imágenes de intensidad. En particular los métodos de agrupamiento espectral tienden a determinar la estructura subyacente en un conjunto de patrones, donde otros métodos convencionales por la disposición y características particulares de los agrupamientos, no obtienen los resultados esperados [6] [7] [8].

En el marco de esta tesina presenta particular interés la posibilidad de utilizar métodos de agrupamiento espectral en imágenes de rango y de intensidad para abordar problemas de segmentación complejos.

Para esto se implementó una librería de funciones que permite capturar y documentar imágenes de tiempo de vuelo utilizando la cámara SwissRanger SR4000. Se definieron dos métodos

de segmentación que combinan la información de intensidad y de rango: el primero de ellos segmenta objetos parcialmente ocluidos, y el segundo es un método segmentación espectral que permite segmentar adecuadamente objetos de niveles de intensidad similares ubicados a distancias próximas u objetos cercanos con niveles de intensidad diferentes. Por último se desarrolló un entorno de software que implementa el método de segmentación espectral propuesto y facilita la visualización de los resultados.

El primer capítulo de la presente tesina expone una descripción general de los métodos actuales para la obtención de imágenes de rango. En el segundo capítulo se detalla el principio de funcionamiento de las cámaras de tiempo de vuelo y se analizan las características principales de la cámara utilizada, denominada MESA SwissRanger SR4000. En el tercer capítulo se presentan los conceptos principales de agrupamiento espectral. En el cuarto capítulo se detallan los métodos propuestos y las implementaciones realizadas. Por último, el capítulo final presenta las conclusiones y propone líneas de trabajo futuras.

Capítulo 2

Adquisición de imágenes 3D

Las imágenes obtenidas con cámaras tradicionales no pueden registrar toda la información que la escena tridimensional provee. El carácter parcial de la información capturada dificulta la aplicación de algoritmos de segmentación y análisis sobre éstas imágenes. Gran parte de las dificultades se deben a la proyección que se realiza del mundo 3D para obtener una representación bidimensional. El proceso de captura pierde la información de profundidad de los objetos. Esto crea ambigüedades entre el tamaño de los mismos y la distancia relativa con respecto al dispositivo sensor. Adicionalmente las cámaras que no cuentan con una fuente de iluminación activa y por lo tanto, para realizar la captura, dependen de la iluminación ambiente, son altamente sensibles a sus variaciones. Esto produce que el mismo objeto pueda parecer diferente cuando las condiciones de iluminación son distintas.

Una alternativa para afrontar estas dificultades y otorgarle mayor robustez a los algoritmos de segmentación aplicados sobre imágenes obtenidas con cámaras tradicionales es incorporar la información de profundidad perdida en el proceso de captura. En el campo de la visión por computador se han desarrollado técnicas que permiten reconstruir la información 3D de la escena y por lo tanto obtener la información de profundidad o rango de los objetos presentes en ella.

La información de profundidad medida por un sensor 3D constituye una nube de puntos en el espacio. Puede ser dada por una grilla rectangular en coordenadas cartesianas $z(x, y)$, o en coordenadas polares $R(\alpha, \phi)$. La forma más simple y a menudo más conveniente de representar y almacenar las coordenadas de una superficie de una escena es utilizar un mapa de profundidad

[9]. Un mapa de profundidad M es una matriz donde la posición (x, y) de un punto corresponde a las fila x y a la columna y de la matriz, y la medición de profundidad z correspondiente a ese punto se almacena en $M(x, y)$. Este tipo de información es llamado mapa de profundidad o imagen de profundidad. Los mapas de profundidad, conocidos también como imágenes de rango, o imágenes $2\frac{1}{2}$D se asemejan a las imágenes en escala de grises generadas por una cámara 2D, con la diferencia que la información de profundidad reemplaza a la información de intensidad. En conjunto con el mapa de profundidad, muchos sensores 3D también entregan una señal de amplitud.

Los sensores 3D se han convertido en un gran desafío en el diseño de sensores de imágenes. La adquisición de datos 3D con sistemas ópticos son preferidos sobre otros métodos alternativos, basados en ultra-sonido o microondas. La razón es que los sistemas ópticos permiten velocidades de capturas muy altas, son sistemas seguros y presentan una resolución lateral muy grande [10]. Los sistemas ópticos de medición 3D, en forma teórica, deben producir una estimación de profundidad independientemente de la reflectividad de la superficie del objeto, su distancia al sensor y las condiciones de iluminación. Diversas variantes de éstos tipos de sensores fueron relevados por Jarvis [11] en 1982. Besl [12] en 1986 presentó una descripción de las diferentes técnicas de medición de rango y de los sensores comerciales existentes en ese momento. Mas recientemente, Chen, Brown y Song [13] presentaron un estudio general de las técnicas utilizadas principalmente en el modelado de objetos 3D. Blais [14] en 2003 realizó una revisión de los últimos 20 años con respecto al desarrollo de sensores de rango desde una perspectiva orientada a la industria. Schwarte et al [1] presentaron una clasificación enfocada en los principios de adquisición de las información tridimensional de la escena.

Las técnicas descriptas por los autores se pueden categorizar como técnicas de obtención de superficies 3D sin contacto que utilizan ondas de luz para realizar la medición. La figura 2.1 muestra la clasificación propuesta por Schwarte et al [1] que será la adoptada en las secciones que se desarrollarán a continuación.

2.1. Medición óptica de imágenes de rango

A partir del conocimiento de los principios físicos subyacentes que definen los límites de la incertidumbre en la medición, es posible escoger el sensor óptimo que se adapte a los restricciones

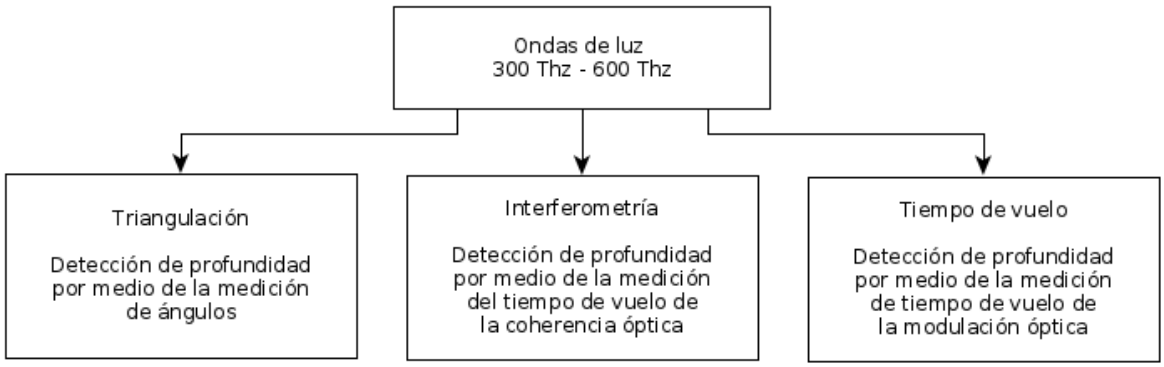


Figura 2.1: Clasificación de técnicas de obtención de superficies 3D

definidas por el dominio de problema y comprender de una forma más precisa los resultados obtenidos. Schawrte et al [1] mostraron que todos los sensores conocidos están basados en uno de los tres principios siguientes :

1. Triangulación
2. Medición por tiempo de vuelo
3. Interferometría

Los tres principios difieren en términos de como la incertidumbre en la medición escala con respecto a la distancia de objeto [15].

El parámetro más críticos en los sistemas de medición 3D son el rango de medición Δz y la resolución de profundidad δz. La figura 2.2 ilustra los rangos de medición y resolución cubiertos por los tres principios de medición mencionados. La figura muestra la incertidumbre relativa $\frac{\delta z}{z}$ como una función de la profundidad del objeto z. La incertidumbre de medición más baja es alcanzada por los métodos basados en la interferometría. Algunas técnicas basadas en este principio permiten incrementar el rango de profundidad de micrómetros a metros [16]. Las técnicas de triangulación pueden ser usadas con gran precisión desde el rango de los milímetros hasta el rango de los 100km, aunque sus principios también se aplican para la medición de distancias astronómicas. La incertidumbre en la medición de los métodos basados en el principio de tiempo de vuelo es prácticamente independiente de la distancia y se encuentra en el rango de los milímetros.

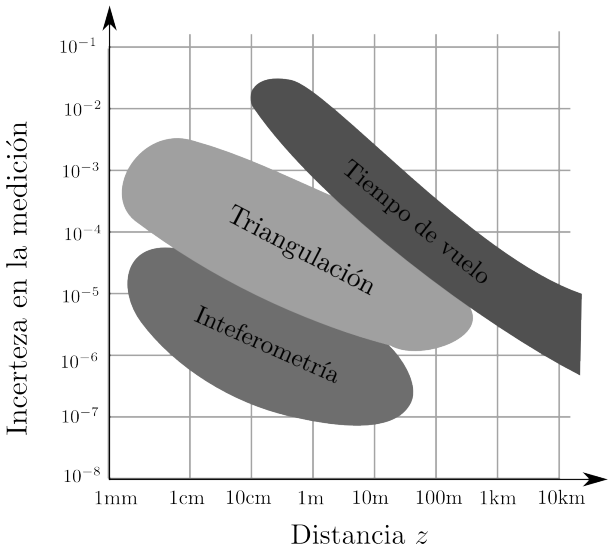


Figura 2.2: Resolución relativa de los métodos ópticos de medición [1]

2.2. Triangulación

La triangulación es la técnica más usada para la medición óptica de rango [1]. El método de triangulación utiliza una aproximación geométrica para obtener las mediciones de distancia. Se construye un triángulo en el que uno de sus vértices es el punto de interés cuya distancia se quiere conocer y los dos puntos restantes son partes conocidas del sistema de medición. La distancia del objetivo puede ser determinada midiendo los ángulos del triángulo o la base del mismo. A pesar de basarse en el mismo principio, las técnicas de triangulación pueden parecer muy diferentes. Desde un punto de vista general se pueden distinguir:

1. Técnicas basadas en el foco
2. Triangulación pasiva basada en fotogrametría y estereoscopía.
3. Triangulación activa con luz estructurada.
4. Forma a partir del sombreado

Técnicas basadas en el foco

El problema de obtener la información de profundidad a partir del foco consiste en reconstruir la profundidad de la escena cambiando activamente la óptica de la cámara hasta que el punto de interés esté enfocado. Un objeto se mueve con respecto al sistema de captura y se toma una secuencia de imágenes que corresponden con diferentes niveles de foco. Se subdivide a las imágenes en un conjunto de regiones disjuntas y se computa una medida de foco por cada una de las regiones de las imágenes de la secuencia. El valor de la medición de foco aumenta a medida que la nitidez o el contraste de la imagen aumenta y alcanza la máxima nitidez cuando la imagen está enfocada. Por lo tanto las regiones enfocadas más nítidas se pueden detectar y extraer. La profundidad de la superficie del objeto capturado en la región puede ser obtenida conociendo la posición de los lentes y la longitud focal que resultó en las regiones enfocadas más nítidas [17].

Las técnicas que adquieren la información de profundidad a partir del desenfoque no son técnicas de búsqueda como la anterior. Miden la cantidad de desenfoque de una imagen y necesita, en los sistemas más simples, sólo dos imágenes con configuraciones ópticas distintas para obtener el mapa de profundidad de toda la escena. El grado de desenfoque es un indicativo fuerte de la profundidad de los objetos ya que se incremente a medida que el objeto se aleja de la distancia mínima de enfoque [18]. Es un procedimiento más rápido en comparación con la técnica basada en el foco, pero menos precisa. El nivel de foco en las dos imágenes puede ser variado modificando la configuración del foco de la lente, o cambiando el tamaño de la apertura de la lente.

Triangulación pasiva

Las técnicas de triangulación pasiva incluyen las diferentes formas de fotogrametría y visión estéreo [19]. Éstos sistemas utilizan múltiples cámaras de forma simultánea para capturar la escena. El modelo básico del sistema consiste en dos cámaras que tienen sus ejes ópticos paralelos, siendo la distancia que los separa la línea de base. Cuando un mismo punto es capturado desde distintas perspectivas su proyección en cada imagen tendrá una posición diferente. Ésta diferencia en las posiciones proyectadas se la conoce como disparidad. Dadas dos cámaras paralelas A y B ubicadas a una distancia b conociendo la longitud focal f de las cámaras y

conociendo la ubicación del píxel proyectado en la imagen capturada por A, $p_A = (x_A, y_A)$ y la ubicación de píxel proyectado en la imagen capturada por B, $p_B = (x_B, y_B)$, la distancia z al punto capturado puede ser calculada por:

$$z = \frac{bf}{d}$$

donde d es la disparidad, es decir, el desplazamiento relativo de una característica de una imagen con respecto a otra $d = p_A - p_B$.

Dado que cada punto a medir debe ser identificado desde ambas perspectivas sin ambigüedad, es necesario resolver el problema de correspondencia entre los píxeles de las imágenes capturadas. La relación inversa que existe entre la profundidad y la disparidad indica que para obtener mediciones más precisas se requiere extender la línea de base. Este requerimiento complica la tarea de establecer correspondencias de características y resulta en sistemas de medición más voluminosos. Adicionalmente, existen ciertos fenómenos que dificultan la tarea de corresponder características entre las imágenes: la falta de texturas en los objetos capturados, patrones regularmente repetidos en las imágenes, oclusión y efectos perturbadores debidos a las condiciones lumínicas.

El esfuerzo computacional requerido para resolver el problema de correspondencia no debe ser subestimado. Los efectos de sombras son también problemas típicos que no pueden ser dejados de tener en cuenta. La visión estéreo funciona correctamente para ciertas escenas, preferiblemente ricas en contraste y de objetivos relativamente planos. Para escenas industriales, sin embargo, son por lo general poco adecuadas. Si bien los problemas de sombras pueden ser minimizados utilizando un mayor número de cámaras, realizando lo que se conoce como sistemas de triangulación de múltiples puntos de vista, ésta mejora trae como consecuencia un gran aumento en el tiempo de cómputo.

La posibilidad de obtener un cuadro entero a partir de un solo momento de captura es una de las principales ventajas que presentan estos sistemas. Además, la ausencia de partes móviles los vuelve altamente confiables. Sin embargo, a pesar de su simplicidad teórica, los sistemas comerciales basados en este principio tienden a ser complejos.

Triangulación Activa

En los sistemas de triangulación activos la escena es iluminada por una fuente de luz coherente desde una dirección y es visualizada desde otra. El triángulo queda formado por el punto donde se emite el haz de luz, el detector y el objeto iluminado. Estos sistemas difieren por el tipo de luz estructurada utilizada: si se utiliza un solo punto, o una luz codificada y el método de escaneo: si se mueve el objeto o si se mueve el sensor.

Al utilizar como fuente de iluminación un rayo láser de baja divergencia, la interacción de ésta radiación con la superficie del objeto producirá un punto iluminado que podrá ser detectado fácilmente por un sensor, típicamente un charge-coupled device (CCD por sus siglas en inglés). La orientación y la posición de la fuente de luz son, por lo general, conocidas. A partir de la ubicación del punto capturado por el sensor puede ser calculada la línea que pasa por el punto sensado y el centro de la cámara. El punto 3D resulta en la intersección entre el haz del láser y la línea de la cámara. Si la orientación y la posición de láser no se conocen, es posible calcular las coordenadas utilizando dos o más cámaras como en los métodos de triangulación pasivos.

Un problema evidente de iluminar un solo punto de la escena, es que las coordenadas tridimensionales de las superficies deben ser extraídas un punto a la vez. Esto requerirá piezas mecánicas que permitan controlar la dirección del rayo láser de manera rápida y precisa, lo que no siempre es posible. Por otra parte los tiempos necesarios para capturar una imagen 3D utilizando la proyección de un solo punto puede requerir varios segundos e incluso minutos.

Con el objetivo de remediar este problema, se han desarrollado técnicas que reemplazan el punto láser por patrones de iluminación más complicados. Al utilizar diferentes fuentes de luz es posible capturar mayor cantidad de puntos por cuadro. Cuando una franja láser o una matriz de puntos es utilizada, es posible reconstruir más puntos al mismo tiempo en un solo cuadro.

Un rayo láser puede ser extendido a un plano poniendo un lente cilíndrico en frente del láser. En vez de formar un solo punto en la superficie, la intersección del plano con la superficie del objeto formará una curva. La cámara captura la escena y para cualquier píxel que resulte de la proyección de la curva, el correspondiente punto 3D en la superficie del objeto será encontrado calculando la intersección entre el rayo que pasa por el píxel y la ecuación del plano láser. El movimiento de escaneo bidimensional necesario cuando se proyecta un solo punto, puede ser reemplazado por un barrido unidimensional que recorra toda la superficie. El uso de una matriz

de puntos láser permite muestrear una región en lugar de una línea, y puede ser potencialmente la solución más rápida para la adquisición de superficies. Sin embargo, el problema de hacer corresponder cada haz de luz con su punto proyectado adquirido por la cámara es más complejo que en el caso de un solo haz de luz.

Las técnicas de triangulación que proyectan luz estructurada sobre el objeto 3D no requieren escaneo. Para materiales que dispersan la luz, la proyección puede ser usada para producir texturas en la superficie. Por ejemplo, la proyección de ruido es usualmente utilizada en la fotogrametría digital que utiliza varios puntos de captura. Otros patrones de proyección incluyen rejillas de líneas o sinusoidales. La triangulación activa por proyección de texturas funciona muy bien en superficies suaves y no especulares.

La técnica de moiré es un patrón de interferencia de baja frecuencia espacial creado cuando dos rejillas regularmente espaciadas son superpuestas una sobre la otra. La información de profundidad utilizando la técnica de moiré se obtiene iluminando la escena con una fuente de luz que pasa por una rejilla, y visualizando al escena a través de una segunda rejilla idéntica. La luz proyectada es modulada espacialmente en amplitud por la rejilla, y la rejilla de la cámara demodula el patrón y crea franjas de interferencia cuyas fases son proporcionales al rango [20]. Los patrones moiré son más útiles cuando los objetos tienen superficies planas relativamente grandes y pequeñas variaciones de profundidad.

Otros métodos proponen la utilización de patrones proyectivos y la detección del mismo patrón a partir de múltiples vistas utilizando sistemas estereoscópicos [21]. La proyección secuencial de patrones codificados es un modo elegante de lidiar con las ambigüedades de rango. Los métodos más populares para la proyección de patrones usan patrones de codificación binaria o patrones de franjas con fases desplazadas. Los métodos que utilizan patrones codificados son muy populares debido a los proyectores de bajo costo disponibles y que permite la adquisición de volúmenes 3D rápidamente con unos pocos cuadros.

Técnicas basadas en el sombreado

Éstas técnicas utilizan el sombreado exhibido por una imagen en escala de grises para inferir las formas de las objetos, basándose en los mapas de reflectancia que relacionan los niveles de intensidad de la imagen con las orientaciones de las superficies. La obtención de los mapas de reflectancia no es una cuestión trivial dado que para cada punto en la imagen se conoce

solamente la reflectancia en el punto correspondiente. Para algunos puntos (llamados puntos singulares) la reflectancia determina la normal local, pero para la mayoría de los puntos no lo hace. Consecuentemente, obtener las formas de la superficie no puede ser encontrada por operaciones locales solamente [22].

Éstas técnicas se basan en el modelo de superficie lambertiano, que indica que la intensidad observada en la superficie no varía de acuerdo al punto de vista. Para muchas superficies la fracción de luz incidente que es dispersa en una dirección dada es una función de los ángulos involucrados:

- El ángulo de incidencia entre la normal local y el haz del sensor.
- El ángulo que forma el haz de luz y la normal locales.
- El ángulo de fase entre el haz de luz y el haz del sensor.

Otra alternativa posible para la utilización de la información proporcionada por el sombreado de las superficies consiste en tomar dos o más imágenes desde el mismo punto de vista pero con diferentes condiciones de iluminación. El objetivo es estimar los vectores normales de las superficies. Esta técnica se denomina estéreo fotométrico.

Técnicas basadas en texturas

Las formas que exhiben las texturas de los objetos en una imagen constituyen una fuente de información de las superficies de los mismos. La disposición de patrones regulares en una imagen depende de la ubicación relativa del objeto con respecto a la cámara. La obtención de formas a partir de texturas estima la forma de las superficies observadas a partir de la distorsión de la textura creada durante el proceso de captura.

Malik y Rosenholtz [23] presentan el problema de la extracción de formas a partir de las texturas haciendo una comparación con las técnicas estereoscópicas. Si se consideran dos porciones cercanas de una superficie en una escena, con la misma textura, o texturas lo suficientemente similares, la apariencia de las dos porciones en la imagen monocular serán ligeramente desemejantes debido a la distinta relación geométrica que tienen con respecto al ojo de la cámara. Por lo tanto se obtienen imágenes desde distintos puntos de vistas de la misma textura, a pesar de ser una sola imagen. Esto sugiere un procedimiento que consiste en dos etapas:

1. Calcular la distorsión de la textura a partir de la imagen
2. Interpretar la distorsión de la textura para inferir la orientación 3D y la forma de la superficie en la escena.

La distorsión de la textura representa la misma información que la disparidad de los sistemas estéreos.

2.3. Interferometría

La interferometría es descripta por la superposición de dos ondas monocromáticas de frecuencia v, amplitud U_1 y U_2 y fase α_1 y α_2, respectivamente, lo que resulta en otra onda monocromática de la misma frecuencia v, pero con diferente fase y diferente amplitud [24]. En la configuración más sencilla, el interferómetro de Michelson, un haz de láser monocromático y coherente es dividido en dos rayos por un divisor de haz. Un rayo es proyectado en un espejo de distancia fija x_1 (haz de referencia) mientras que el otro rayo es enviado al objeto ubicado a una distancia variable x_2 (haz de medición). Ambos rayos son reflejados de regreso al separador, que los proyecta a un detector de integración. El haz de medición viaja una distancia diferente que el haz de referencia por lo que ya no estarán en fase.

Cuando los haces de luz se encuentran en el sensor, se superponen e interfieren, y la diferencia de fase entre ellos crea un patrón de zonas claras y oscuras, que se conoce como franjas de interferencia. Las áreas claras son lugares donde la amplitud de los haces se suma debido a la interferencia constructiva, las zonas oscuras son lugares donde la amplitud de los haces se sustrae debido a la interferencia destructiva.

La aplicación básica de la interferometría es la medición precisa de pequeñas distancias (micrómetros hasta varios centímetros).

2.4. Tiempo de Vuelo

Los sistemas basados en el principio de tiempo de vuelo emiten una señal a la escena y calculan la distancia midiendo el tiempo que demoró el pulso de luz en viajar desde el dispositivo hasta un punto de referencia. Esta medición indirecta de la distancia es posible dado que la velocidad de la luz se conoce con precisión [25]:

$$c = 3{,}10^8 \frac{m}{s}$$
$$c = 2{,}150 \frac{m}{\mu s}$$
$$c = 2{,}015 \frac{m}{ns}$$
$$c = 2{,}015 \frac{mm}{ps}$$

El principio, denominado tiempo de vuelo directo, consiste en la emisión de un pulso de luz con velocidad c hacia el objetivo ubicado a una distancia D y la contabilización por parte de un cronómetro del tiempo requerido por el pulso para alcanzar el objetivo y regresar de el. La recepción del pulso de luz por el mecanismo detector detiene la contabilización del tiempo.

Sea Δt es el tiempo que demoró el pulso de luz en recorrer dos veces la distancia D (al viajar hasta el objeto y volver de el), se debe considerar la mitad del tiempo transcurrido.

$$D = \frac{c\Delta t}{2}$$

Ejemplo Si el tiempo cronometrado $\Delta t = 6{,}67127ns$ la distancia medida será de un metro:

$$\begin{aligned} 6{,}67ns &= 0{,}000000000667s \\ D &= \frac{c\Delta t}{2} m \\ &= \frac{2999792458 \cdot 0{,}000000000667}{2} m \\ &= 1{,}000m \end{aligned}$$

Es evidente que el problema básico para establecer sistemas de tiempo de vuelo para la medición de distancia es la realización de mecanismos medidores de tiempo muy precisos. Para poder alcanzar una resolución de 1 cm, se requiere una resolución temporal de 67 picosegundos.

Una de las ventajas principales de este tipo de sistemas es que no producen información incompleta de rango como lo hacen los sistemas de triangulación. Esto se debe a que la iluminación y la observación son colineales, lo que evita los efectos de sombras.

Una variación del principio de tiempo de vuelo directo, son los sistemas de tiempo de vuelo de modulación de amplitud. En este tipo de sistemas la fuente de energía opera continuamente, pero la señal emitida es modulada sinusoidalmente en el tiempo. Cuando la señal modulada

alcanza a un objeto y retorna al sensor, ésta seguirá estando modulada en amplitud, pero ya no estará en fase con la señal emitida.

Calculando el desplazamiento de fase entre la señal emitida y la reflejada, es posible calcular el tiempo que demoró la señal en alcanzar el objeto que la reflejó y regresar de él y por lo tanto la distancia en la que se encontraba el objeto. La relación entre el rango R y la fase ϕ está dada por:

$$R = \frac{c}{2f}$$

donde f es la frecuencia de modulación y c es la velocidad de propagación de la señal.

Una de las ventajas principales de los sensores que utilizan la modulación de amplitud es la posibilidad de modificar la frecuencia de modulación para adaptarla al rango que se desea medir e incluso utilizar diferentes longitudes de onda en forma simultánea para alcanzar rangos más extensos y mayor resolución en las mediciones. Si se utilizan frecuencias de modulación altas, se consigue una mayor resolución de profundidad pero un menor campo de profundidad y si se utilizan frecuencias bajas de modulación se obtiene un campo de profundidad más grande y una menor precisión. Han surgido avances en los últimos años que permitieron el desarrollo de sensores basados en éste principio que alcanzan velocidades de captura muy altas.

Un inconveniente que presenta la utilización de sensores de modulación de amplitud es que sufre ambigüedades debido a que las diferencias de profundidad correspondientes a los desplazamientos de fase que son múltiplos de 2π no se pueden resolver.

Capítulo 3

Cámaras de tiempo de vuelo

Las cámaras de rango matriciales de tiempo de vuelo, o cámaras de tiempo de vuelo (ToF por sus siglas en inglés) son sensores activos capaces de adquirir la geometría tridimensional de la escena mediante la técnica de tiempo de vuelo por modulación de amplitud [26]. En la actualidad existen varios productos comerciales desarrolados por empresas independientes como Mesa Imaging [27], PMD Technologies [28], SoftKinetic [29] y Microsoft [30]. Éstas cámaras pueden ser usadas para estimar la estructura 3D directamente, sin la necesidad de algoritmos tradicionales de visión por computador.

Su utilización en aplicaciones civiles comenzó alrededor del año 2000, debido a que los procesadores se hicieron lo suficientemente rápidos para estos dispositivos. Los sistemas abarcan rangos de profundidad desde unos pocos metros hasta 60 metros. La resolución de la distancia es de hasta 1 centímetro. La resolución lateral de las cámaras de tiempo de vuelo son generalmente bajas comparadas con las cámaras de vídeo 2D. La mayoría de las productos comerciales cuentan con una resolución de 320×240 píxeles. Comparado con los sistemas láser 3D, las cámaras ToF operan a una velocidad muy alta, adquiriendo hasta 100 imágenes por segundo y permiten medir simultáneamente rango e intensidad por cada píxel.

3.1. Principio de medición de profundidad

Las cámaras de tiempo de vuelo utilizan una fuente de luz modulada en amplitud con una frecuencia f_m que varía entre 10-100 Mhz que ilumina el campo de visión. El principio de medición mostrado en la figura 3.1, consiste en tomar cuatro mediciones de la luz sinusoidal

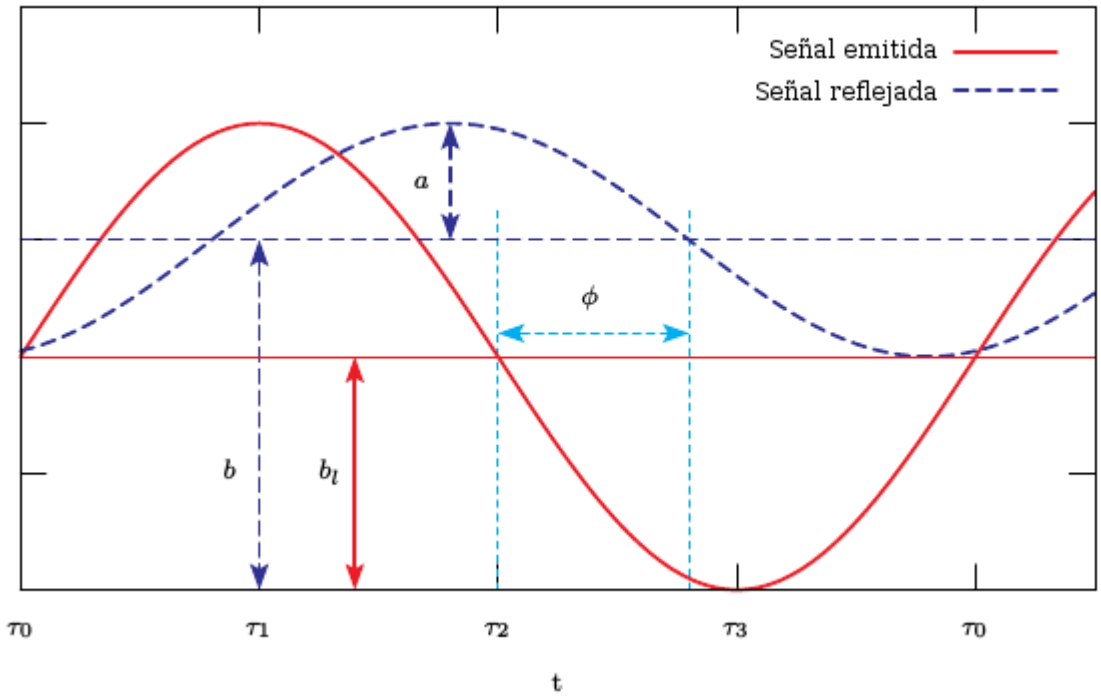


Figura 3.1: Señal de entrada recibida muestreada en 4 puntos por cada período de modulación T

reflejada en $0°, 90°, 180°$ y $270°$ de la fase por cada período $T = \frac{1}{f_m}$. El desplazamiento de fase de un píxel θ_i, la amplitud a_i y la intensidad b_i puede ser calculada de la siguiente manera:

$$\phi_i = \text{atan}\left(\frac{S_i(t_0) - S_i(t_2)}{S_i(t_1) - S_i(t_3)}\right)$$

$$a_i = \frac{\sqrt{(S_i(t_0) - (S_i(t_2))^2 + (S_i(t_1) - S_i(t_3))^2}}{2}$$

$$b_i = \frac{\sum_{j=0}^{3} S_i(t_j)}{4}$$

Donde $S_i(t_0), S_i(t_1), S_i(t_2), S_i(t_3)$ son las mediciones en $0°, 90°, 180°$ y $270°$, respectivamente.

La distancia d_i puede calcularse entonces a partir de la información de fase:

$$d_i = \frac{\alpha_m}{2}\frac{\phi_i}{2\pi}$$

donde α_m es la longitud de onda de la señal de modulación

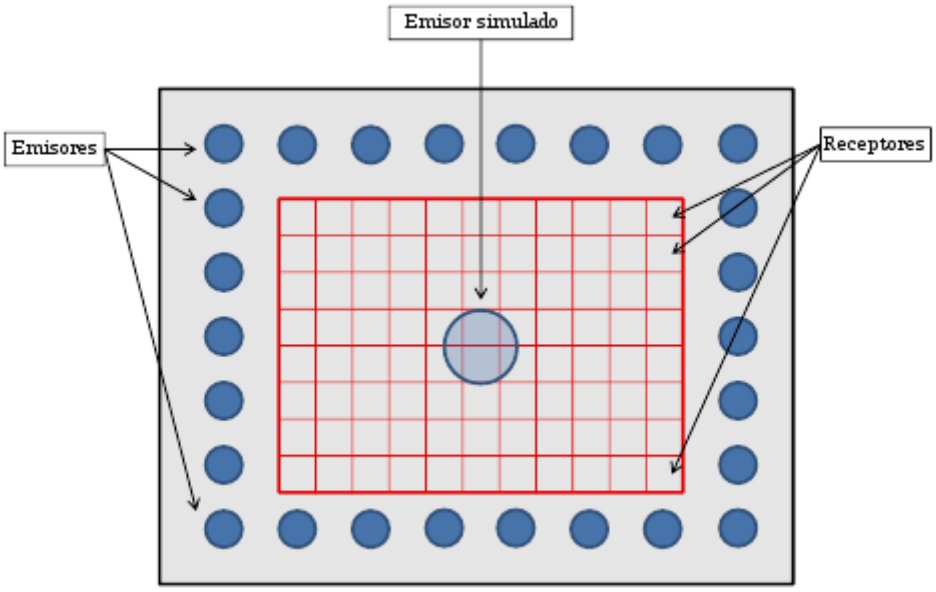


Figura 3.2: Esquema de un sensor ToF matricial. La matriz CCD/CMOS de pixeles lock-in está en rojo. Los emisores (azul) se distribuyen alrededor y en el centro de la matriz de píxeles, simulando un emisor coposicionado.

3.2. Cámaras de tiempo de vuelo matriciales

Una cámara de tiempo de vuelo puede ser interpretada conceptualmente como una organización matricial con N_f filas y N_c columnas de múltiples dispositivos, cada uno de ellos compuesto por un emisor y un receptor co-posicionado. En la práctica, la implementación de los dispositivos basados en la yuxtaposición de dispositivos de medición individuales no es realizable. Actualmente no es posible integrar $N_f \times N_c$ emisores y $N_f \times N_c$ receptores en un solo chip, especialmente para grandes valores de N_f y N_c. Sin embargo, no es necesario que cada receptor requiera un emisor y co-posicionado, en su lugar un solo emisor puede proveer la radiación que será reflejada por la escena y recolectada por una multitud de receptores. La figura 3.2 muestra yn esquema de un sensor matricial con emisores simulados. Una vez que los receptores están separados de los emisores, el dispositivo puede ser implementado con un CCD/CMOS lock-in pixel [25] e integrado en una matriz de $N_f \times N_c$.

3.3. Control de las cámaras de tiempo de vuelo

El ajuste de las cámaras TOF en escenas dinámicas es aún tarea dificultosa. La precisión depende de un número de parámetros, dos de ellos pueden ser controlados por el usuario:

- La frecuencia de modulación
- El tiempo de integración

3.3.1. Frecuencia de modulación

La frecuencia de modulación es uno de los parámetros mas importantes a la hora de configurar una cámara de tiempo de vuelo debido a que la precisión es proporcional a ella. Sin embargo, la frecuencia de modulación también determina la distancia máxima de medición. Si la frecuencia de modulación aumenta, también lo hace la precisión, mientras que la distancia máxima mensurable se decrementa.

Las mediciones están sujetas a un fenómeno denominado ambigüedad ilustrado en la figura 3.3. Esto se debe a la periodicidad de la señal que es usada para la medición de la señal. Si en la escena hay objetos ubicados a distancias mayores a la distancia máxima correspondiente a un período entero de modulación, la medición de su posición es ambigua. Por ésta razón la distancia de una fase entera es referida como rango de no-ambigüedad. La distancia máxima mensurable D puede ser calculada conociendo la frecuencia de modulación:

$$D = \frac{c}{2f_m}$$

donde c es la velocidad de la luz y f_m es la frecuencia de modulación. Con una frecuencia de modulación de $30Mhz$, la distancia máxima mensurable es de 4.997 m considerando a la velocidad de la luz como $299792458\frac{m}{s}$. Si se utiliza una cámara con un rango de medición de $5m$, cualquier objeto ubicado mas allá de los $5m$ y cuya intensidad es lo suficientemente fuerte para ser detectada por la cámara será interpretado como si el objeto se encontrara en el rango de no ambigüedad. Un objeto ubicado a $7m$ será medido como si se encontrara a $2m$.

3.3.2. Tiempo de integración

Cuando la cantidad de fotones recibidos excede la cantidad máxima que el sensor del pixel puede tomar, éste se satura. El fenómeno es particularmente notable en presencia de iluminación

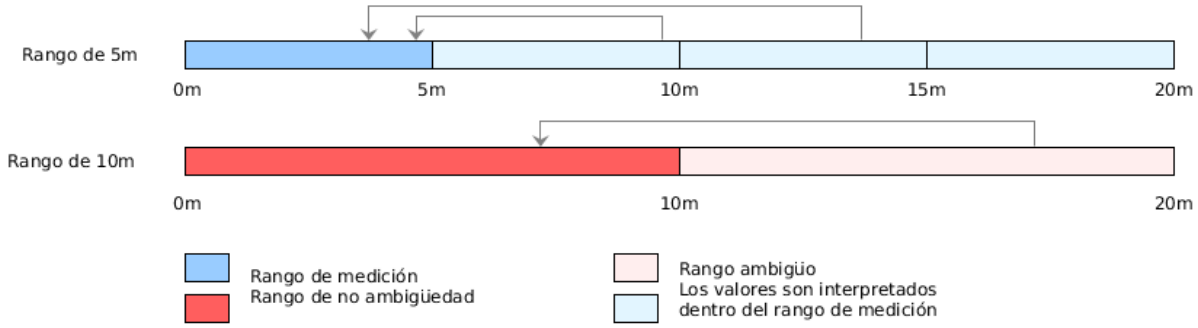


Figura 3.3: Ilustración del rango de no ambigüedad

infrarroja extrema (iluminación solar externa) o en presencia de objetos altamente reflectantes. El tiempo de integración indica la cantidad de tiempo que el píxel puede colectar luz. Con un tiempo de integración mas largo, mayor será la cantidad de fotones colectados y mayor probabilidad habrá de saturación.

La reflectancia de los objetos es un parámetro importante para la elección del tiempo de integración y, por lo tanto, para realizar una medición precisa. Típicamente una escena está compuesta por objetos de diferente reflectancia: desde objetos que reflejan el 2 % para neumáticos de caucho negro hasta el 100 % para el papel blanco [25].

Un escena que contiene solo un pequeño objeto con alta reflectancia cerca del sensor se utiliza para explicar el balance que se debe conseguir: dado que el tiempo de integración se ajusta solamente para todos los píxeles de forma conjunta, si configura la cámara con un tiempo de integración pequeño, lo suficiente para capturar al objeto cercano, los objetos del fondo serán capturados con baja precisión. Si, por el contrario, se utiliza un tiempo de integración alto, los objetos del fondo podrán ser capturados con mayor precisión pero se saturarán los píxeles que reciban la luz reflejada por el objeto cercano.

Tiempos de integración elevados implican iluminar la escena en un intervalo de tiempo mas largo. El incremento del consumo de energía calienta la cámara. Por lo tanto en ambientes no debidamente refrigerados, el tiempo de integración puede causar distorsión. Debido a que son tomadas cuatro muestras para producir la medición de fase, se requiere de cuatro períodos de integración separados. Por lo tanto el tiempo total requerido para capturar una imagen de profundidad es cuatro veces el tiempo de integración sumado al tiempo de lectura. Claramente el tiempo de integración influye en la cantidad de cuadros por segundo que se pueden obtener.

Elección del tiempo de integración óptimo

No es necesariamente la mejor estrategia evitar la saturación de todo el conjunto de datos. Hacer foco en un objeto pequeño decrementará la precisión del resto de la escena. El nivel de la señal para objetos con baja reflectancia difusa será baja si objetos con alta reflectancia se encuentran en el mismo rango de visión durante el proceso de medición.

Con el objetivo de determinar el tiempo de integración óptimo tres aspectos se deben tener en cuenta de forma simultánea:

1. El tiempo de adquisición necesario.
2. La calidad de medición necesaria.
3. El tipo de los objetos medidos.

La cantidad de cuadros por segundo que se pueden adquirir es inversamente proporcional al tiempo de integración. Fijando un tiempo de integración pequeño resultará en una tasa de cuadros por segundo mas elevada. Además si se utilizan tiempos pequeños de integración la posibilidad que un objeto se mueva entre las cuatro mediciones del ángulo de fase es mas pequeña, por lo que disminuirá la presencia de artefactos de movimiento en los objetos.

Sin embargo, el ruido en la medición es inversamente proporcional al tiempo de integración: tiempos de integración mas elevados permiten capturar una mayor cantidad de luz, lo que resulta en menos ruido.

La cantidad de luz que es reflejada por el objeto medido es proporcional a su reflectancia en el dominio infrarrojo. Por lo tanto, para alcanzar la misma calidad de medición se necesita un tiempo de integración mayor para objetos con baja reflectancia con respecto a los que reflejan mayor proporción de luz. La intensidad de la luz decrece a medida que viaja por el espacio. Por lo tanto para alcanzar la misma calidad en la medición, los objetos ubicados mas lejos de la cámara requerirán mayor tiempo de integración con respecto a aquellos ubicados cerca.

3.4. Errores en la medición

Un número de errores sistemáticos y no sistemáticos se presentan al realizar capturas utilizando cámaras de tiempo de vuelo. Lange [25] presenta una extensa descripción de éstos errores. A continuación se detallarán los mas importantes.

Las medidas de profundidad con cámaras ToF se enfrentan a la aparición de errores sistemáticos y no sistemáticos. En general, los errores sistemáticos pueden ser controlados mediante la calibración del dispositivo y los no sistemáticos utilizando filtros [31].

3.4.1. Errores sistemáticos

Imprecisiones en la modulación

Uno de los errores sistemáticos que se presentan en la medición son la distorsiones de profundidad que aparecen como consecuencia del hecho que la luz infrarroja emitida no puede ser generada, en la práctica, como se planifica en la teoría (generalmente sinusoidal) debido a irregularidades en el proceso de modulación. Por lo general la gráfica del error con respecto a la distancia presenta una forma sinusoidal.

Tiempo de integración

El tiempo de integración puede causar diferentes mediciones de profundidad. El tiempo de integración afecta el rango de las mediciones que la cámara puede medir con precisión.

Baja amplitud

Pueden surgir imprecisiones en la medición cuando la amplitud de la señal reflejada es baja. La precisión de la profundidad está altamente relacionada con la cantidad de luz incidente. A mayor amplitud reflejada, mayor sera la precisión de la medida. Las amplitudes bajas se registran por lo general en el borde de la imagen debido a que el poder de la luz emitida en los bordes del emisor es menor que en el centro, provocando una sobre estimación de profundidad. Una posibilidad para evitar estos problemas es aplicar un umbral con el objetivo de filtrar los píxeles cuyo nivel de amplitud no supere ese valor, pero esta solución puede descartar grandes porciones de la imagen.

Temperatura

Los valores de profundidad sufren un desplazamiento en toda la imagen cuando la temperatura de la cámara se desestabiliza. Es por esto que muchos modelos cuentan con un sistema de refrigeración.

3.4.2. Errores no sistemáticos

La distorsión en la relación señal-ruido en escenas no uniformemente iluminadas. Las áreas con menor iluminación son más susceptibles a ruido que las de mayor iluminación. Este tipo de error es altamente dependiente de la amplitud, el tiempo de integración y la uniformidad de profundidad en la escena. Una escena con distintas profundidades puede conducir a áreas de amplitud baja (los objetos mas lejanos) que serán afectadas fuertemente por ruido.

Interreflexión

Los errores debido a la interreflexión o la recepción múltiple de luz se debe a la interferencia de múltiples reflexiones de luz capturada en cada sensor del píxel. Las recepciones múltiples de luz se deben principalmente a los bordes de los objetos y sus concavidades. Los errores generados por los bordes pueden ser removidos comparando el ángulo de incidencia de los píxeles vecinos. Por otro lado, es un tema de investigación actual el problema de múltiples reflexiones originadas por concavidades.

Artefactos debidos al movimiento

Cada una de las cuatro muestras de medición de fase son tomadas en exposiciones separadas. Esto significa que para obtener una medición de distancia, se requieren cuatro exposiciones consecutivas. Si un objeto en la escena se mueve durante estas exposiciones, se pueden introducir errores en la medición. Esta interferencia introducida debido al movimiento se presenta como un ruido aleatorio en las aristas de los objetos.

3.5. MESA SwissRanger SR4000

El modelo de cámara de tiempo de vuelo MESA SwissRanger SR4000 mostrada en la figura 3.5 es la cuarta generación de sensores desarrollados por la empresa suiza MESA Imaging. Sus especificaciones se encuentran detalladas en la figura 3.4

3.5.1. Datos de salida de la cámara

A partir del desplazamiento de fase y la información de amplitud la cámara entrega los siguientes datos por cada píxel:

Tamaño de la matriz de píxeles	176 × 144
Rango de operación	5m a 10m
Precisión absoluta	±1 cm o ±1 %
Frame Rate	10 - 30 fps
Interfase de comunicación	Fast Ethernet
Temperatura	0 - 50^{o}C
Alimentación	12V, 2A
Seguridad de ojos	Clase 1
Dimensión	65 × 65 × 76 mm (Ancho × Alto × Profundidad)

Figura 3.4: Especificaciones de la cámara Tof SwissRanger SR4000

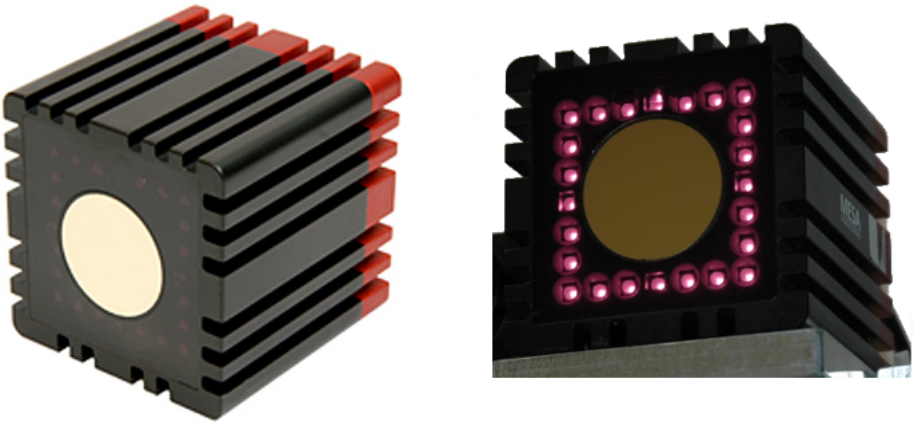


Figura 3.5: Ilustración del rango de no ambigüedad

- Datos de distancia en coordenadas polares
- Coordenadas (x,y,z)
- Amplitud
- Mapa de confianza (una estimación de la confianza de los datos)

Datos de distancia

Los datos de distancia son entregados por el ángulo de fase calculado. Los ángulos de fase para todos los píxeles se obtienen como un puntero a un arreglo de $176 \times 144 = 25344$ enteros cortos sin signo. Los datos son almacenados como una secuencia de 144 filas de 176 elementos. Los ángulos de fase que se almacenan en el arreglo solo utilizan los 14 bits mas significativos de los 16 disponibles por cada píxel. Los dos bits menos significativos son reservados.

El valor de distancia es entonces:

$$d = \frac{\phi . d_{\max}}{2^{14}}$$

donde ϕ es el valor de 14 bits del píxel codificado por los 14 bits mas significativos y $d_{\max}$ la máxima distancia mensurable por la frecuencia de modulación utilizada por la cámara. El valor correspondiente a los primeros 14 bits del valor hexadecimal 0×0000 corresponde al origen del sistema de coordenadas intrínseco del sistema y el valor correspondiente a los 14 bits mas significativos del valor hexadecimal $0 \times 3FFF$ corresponde a la distancia radial de fase completa.

Coordenadas cartesianas

La librería provista por la empresa cuenta con un conjunto de funciones que convierten los 14 bits de distancia radial en coordenadas cartesianas, expresadas en metros. Los cálculos se realizan por la librería en la CPU host. Ésta transformación incluye una corrección que compensa la distorsión radial de la óptica. El sistema de coordenadas usado aquí utiliza la regla de la mano derecha como se puede ver en la figura 3.6, con la coordenada z creciendo a lo largo del eje óptico que se aleja de la cámara, la coordenada y incrementa verticalmente hacia arriba y la coordenada x incrementa horizontalmente hacia la izquierda, todos desde el punto de vista de la cámara.

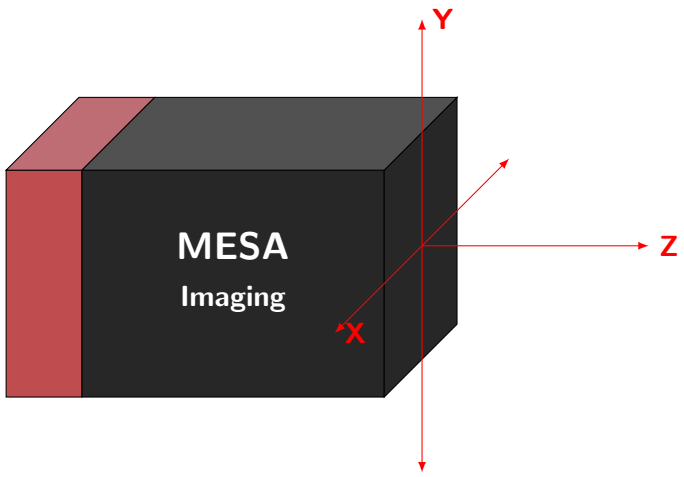


Figura 3.6: Los ejes de mi carreta

3.5.2. API

Apertura y clausura de la cámara

La cámara es un objeto instanciado del tipo **SRCAM** definido en el archivo libMesaSR.h. Las funciones de la API toman una variable de éste tipo como argumento. De forma de obtener la información provista por la cámara, es necesario abrirla primero, es decir, inicializar la comunicación via Ethernet.

La cámara puede ser inicializada utilizando la función SR_OpenETH(). La función requiere la dirección IP como segundo parámetro.

```
1  int SR_OpenETH(SRCAM *srCam, const char *addr);
```

Otra opción para la apertura es abrir un stream SwissRanger (SRS). La función

```
1  int SR_OpenFile(SRCAM *srCam, const char *filename);
```

puede ser usada para abrir un stream almacenado con anterioridad. Es importante destacar que las funciones relacionadas con los streams SRS no funcionan en los sistemas GNU/Linux.

Todas éstas funciones abren una cámara, sea real o virtual, y deben tener una función de cierre SR_Close() asociada. La función SR_Close() libera los recursos almacenados y cierra la conexión con la cámara si ésta estuviera establecida.

```
1  int SR_Close(SRCAM srCam);
```

Captura de imágenes

La función SR_Acquire() dispara una captura en la cámara y transfiere los datos de la cámara a la PC. Las imágenes adquiridas pueden ser tomadas utilizando la función SR_GetImage(). Luego de ésta función por lo general, las coordenadas esféricas son transformada en coordenadas cartesianas.

```
int SR_Acquire(SRCAM srCam);
```

El registro ImgEntry representa una imagen capturada por la cámara. Cuenta con el tipo de imagen de la que se trata (si es de amplitud o de rango), el tipo de datos de la matriz de la imagen (unsigned short, float), el puntero a los datos y el ancho y al alto de la imagen.

```
typedef struct _ImgEntry
{
  enum ImgType imgType; //Tipo de imagen
  enum DataType dataType; //Tipo de datos
  void* data;       // Puntero a los datos
  WORD  width;      // Ancho de la imagen
  WORD  height;     // Alto de la imagen
} ImgEntry;
```

La función **SR_GetImageList()** retorna la cantidad de imágenes capturadas por la función **SR_Acquire** y un arreglo con las imágenes capturadas. Las imágenes pueden ser accedidas directamente por el miembro data del registro ImgEntry o usando la función SR_GetImage() con el índice correspondiente. La lista de imágenes disponibles puede ser afectada por el parámetro de configuración AcquireMode que se modifica utilizando la función SR_SetMode()

```
int   SR_GetImageList(SRCAM srCam, ImgEntry **imgEntryArray)
```

La función SR_GetImage retorna el puntero a la imagen capturada según el índice pasado como parámetro

```
void* SR_GetImage(SRCAM srCam, unsigned char idx)
```

Control de parámetros

Tiempo de integración El tiempo de integración puede ser configurado con la función **SR_SetIntegrationTime**. Recibe el objeto correspondiente a la cámara, y el tiempo de integración que es un valor en el rango de 0 a 255.

```
int SR_SetIntegrationTime(SRCAM srCam, unsigned char intTime);
```

El tiempo de integración se establece mediante la siguiente fórmula

$$IT = 0{,}300\text{ms} + (intTime),0{,}100\text{ms}$$

La cantidad de cuadros por segundo entonces

$$FR = 4(IT + RO)$$

donde RO, el tiempo de lectura es aproximadamente 4.6ms. Por defecto el parámetro del tiempo de integración es de 30 (0x13). Lo que conduce a un tiempo de integración de 3.3ms y una tasa de 31.5 fpx aproximadamente. La tasa mas rápida está limitada por la velocidad de transferencia de la imagen a la PC. Éste tiempo depende de la cámara y es alrededor de 6ms.

La función

```
SR_GetIntegrationTime()
```

entrega el tiempo de integración actual

Umbral de amplitud El parámetro de umbral de amplitud define la amplitud mínima que necesita ser medida para aceptar la medición. El parámetro puede ser usado para suprimir valores con niveles de amplitud bajos que pueden ser inestables. El umbral de amplitud puede ser configurado con;

```
int SR_SetAmplitudeThreshold(SRCAM srCam, unsigned short val)
```

donde el rango de valores de val va de 0 a $2^{16} - 1$. El efecto es que las distancias y los valores (x,y,z) de un píxel con una amplitud mas baja que val son puestas a 0.

La función

```
SR_GetAmplitudeThreshold()
```

Frecuencia de Modulación La frecuencia de modulación puede ser configurada con la función

```
int SR_SetModulationFrequency(SRCAM srCam, enum ModulationFrq modFrq)
```

donde el tipo enumerativo tiene los siguientes valores.

modFrq	1	6	8	9	10	11
MHz	30	15	29	31	14.5	15.5

Filtros y funciones auxiliares

Los modos de captura pueden ser modificados utilizando la función

```
1 int SR_SetMode(SRCAM srCam, int mode)
```

que toma un argumento entero. Este número es una combinación de las banderas especificadas en el enumerativo AcquireMode. Algunos de los valores principales del enumerativo son:

- Filtro de Medianas: AM_MEDIAN. Aplica un filtro de medianas 3×3 en la imagen. El procesamiento se realiza en el host.
- Conversión a grises: (AM_CONV_GRAY). Este modo compensa los valores de intensidad devueltos por la cámara. La imagen resultante se asemeja más a una imagen de niveles de gris convencional obtenida con una cámara pasiva. El procesamiento se realiza en el host.
- Filtro Adaptativo: (AM_DENOIS_ANF). Este valor aplica un filtro de ruido implementado en hardware. El filtro combina la información de distancia y amplitud en una vecindad de 5×5. Reduce el ruido mientras preserva aristas o estructuras pequeñas. El filtro se realiza en el hardware de la cámara.

3.6. Simulación

El software Blensor desarrollado por Gschwandtner et al. [32] [33] provee un entorno unificado de simulación y modelado que es capaz de simular distintos tipos de sensores, considerando las propiedades técnicas de cada uno. El entorno se integra con el software Blender [34], una aplicación para la creación de contenido 3D. Con ésta combinación es posible modelar escenarios de prueba con un nivel de detalle arbitrario y simular la salida del sensor directamente dentro del entorno modelado.

El sistema tiene en cuenta la propiedades de los sensores, como modelos de ruido, efectos físicos como la reflexión, refracción y reflectancia.

El software utiliza algoritmos relacionados con la técnica del trazado de rayos. El software consiste en modificaciones al código base de Blender para emitir varios rayos de forma simultánea. Mediante el uso de esta interfaz, los sensores pueden configurar un conjunto de direcciones de rayos y emitirlos utilizando el núcleo modificado de Blender. Esta modificación procesa todos los rayos, calcula reflexiones si es necesario y devuelve las distancias en la que

se produjo la colisión. Finalmente, el código del sensor calcula sensor de ruido dependientes y otras características físicas.

Para construir un escena estática o dinámica para la simulación del sensor, se pueden utilizar todas las herramientas básicas de Blender. Cualquier objeto puede ser agregado a la simulación. Algunas propiedades de los materiales (por ejemplo el parámetro de reflexión difusa) tiene un impacto en la mediciones simuladas de profundidad. En Blensor, las cámaras determinan el punto de visión del sensor. Una vez que la escena fue modelada y animada, el usuario selecciona la cámara que actuará de sensor, ajusta las propiedades físicas y eventualmente simula el proceso de escaneo.

Capítulo 4

Agrupamiento espectral

Los métodos de agrupamiento espectral utilizan los autovectores y autovalores extremos de las matrices derivadas de los patrones de entrada. De forma opuesta a los algoritmos de agrupamiento tradicionales como k-medias que permiten agrupar patrones con formas geométricas convexas, el agrupamiento espectral puede resolver problemas en escenarios mucho más complejos, tales como espirales entrelazadas, u otras formas no lineales arbitrarias.

Diferentes versiones de agrupamiento espectral han sido aplicadas satisfactoriamente en segmentación de imágenes [7] [35], minería de texto [36], reconocimiento de voz [37] y métodos de propósito general para análisis de datos y agrupamiento [8] [38] [39] [40] [41].

En una de las primeras referencias científicas a los métodos de agrupa espectral [42] se sugería que los autovectores de las matrices de adyacencia podrían ser usados para determinar las particiones subyacentes del conjunto de datos. Fiedler [43] asoció el segundo menor autovalor de la matriz laplaciana de un grafo con su

conectividad y sugirió particionar el grafo dividiendo los vértices de acuerdo a su valor en el autovector correspondiente. Una revisión excelente de la evolución de los métodos de agrupamiento espectral puede ser encontrada en [44].

Una línea de análisis realiza una conexión del agrupamiento espectral con el particionado de grafos [7] [6]. El particionado de grafos es el problema de dividir los vértices de un grafo en conjuntos disjuntos optimizando una función criterio. La matriz derivada a partir de la distancia de los patrones puede ser vista como la matriz de adyacencia W de un grafo no dirigido en donde los pesos de las aristas están dados por una función de distancia.

Los métodos de segmentación espectral no hacen suposiciones acerca de la estructura global de los datos. En su lugar, utilizan la evidencia local sobre cuán probable es que dos patrones pertenezcan a la misma clase y luego realizan una decisión global para dividir los patrones en conjuntos disjuntos de acuerdo a algún criterio. Por lo general se obtiene una representación de dimensión inferior en la que las relaciones entre los patrones son preservadas lo más posible. Lo que hace atractivo a los métodos de agrupamiento espectral es que la obtención de la nueva representación se consigue mediante la obtención de los autovectores asociados a las matrices de semejanza obtenidas a partir de los patrones de entrada. Es decir que los autovectores son tratados como coordenadas geométricas de un conjunto de patrones[45]. Sin embargo, para obtener una solución discreta a partir de los autovectores usualmente requiere resolver un problema de agrupamiento en el nuevo espacio.

Los algoritmos de agrupamiento espectral pueden ser vistos como un algoritmo de tres etapas.

1. Construcción de un grafo G de semejanza para todo el conjunto de patrones.
2. Mediante la obtención de los autovectores de la matriz de adyacencia W del grafo G, los patrones son embebidos en un espacio en donde es más sencillo hacer el agrupamiento.
3. Aplicación de un algoritmo de agrupamiento clásico para particionar en el nuevo espacio.

La representación de dimensión obtenida en el segundo paso se lo conoce como **spectral embedding**. Esta técnica es utilizada, no solo en algoritmos de agrupamiento, sino también en métodos de reducción de dimensión [46] [41].

4.1. Grafo de Semejanza

Sea $X = \{x_1, ..., x_n\} \in \Re^n$ el conjunto de patrones que se requiere particionar en k subconjuntos y sea $f : X^2 \to \Re$ una función de semejanza que asigna valores altos a patrones similares y valor bajos a patrones desemejantes. Con el objetivo de realizar agrupamiento espectral, primero es necesario representar los datos de un grafo pesado no dirigido de semejanza $G = (V, E)$. En él, cada punto $x_i \in X$ es representado por un vértice $v_i \in V$ y cada arista perteneciente a E es una tríada $(x_i, x_j, w_{ij}) \in E$ donde $x_i, x_j \in X$ y $w_{ij} = f(x_i, x_j)$ es el peso de la arista. Debido a que G es un grafo no dirigido $w_{ij} = w_{ji}$

Una forma de representar un grafo es utilizar una matriz bidimensional W. Para cada arista (x_i, x_j, w_{ij}), la entrada en la matriz W es $W(i,j) = w_{ij}$. El peso de w_{ij} será cero cuando los vértices v_i y v_j no están conectados [47], es decir, son completamente desemejantes. Como los métodos de agrupamiento espectral tienen como objetivo particionar los vértices dejando en el mismo grupo a aquellos que tengan mayor semejanza y en diferentes grupos a aquellos con semejanza baja, es crítica la elección de un método efectivo para construir esa matriz de adyacencia.

Estudios recientes en la teoría espectral de grafos [48] [46] [49] han demostrado que el grafo de semejanza debe modelar la estructura geométrica local de los patrones. Basándose en esta regla es posible utilizar cuatro modelos para construir W [50].

- Grafos de k vecinos más próximos: en éste tipo de grafos el vértice v_i es conectado con v_j cuando v_j está entre los k-vecinos más próximos de v_i o v_i está entre los k-vecinos más próximos de v_j.

- Grafos de k vecinos más próximos simétrico: otra alternativa consiste en conectar v_i y v_j cuando v_i está entre los vecinos más próximos de v_j y v_j está entre los vecinos mas próximos de v_i. Este caso es posible utilizar como peso de las aristas 0 y 1 según estén conectados o no.

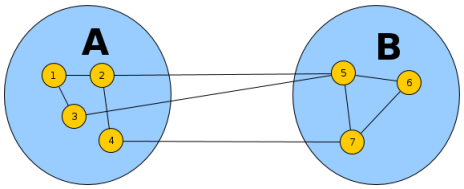
- Grafo de vecindad ϵ: en éste tipo de grafos los vértices son conectados sólo cuando la distancia $||x_i - x_j||^2$ es menor que ϵ. Esto puede conducir a grafos con componentes desconectados si ϵ no se elije adecuadamente.

- Grafo completamente conectado: en este caso se conectan todos los vértices con semejanza positiva.

Debido a que el grafo de semejanza debe modelar la vecindad local, la medida de semejanza debe ser elegida cuidadosamente. Un ejemplo de función muy utilizada es

$$w_{ij} = e^{\frac{-||x_i - x_j||^2}{\alpha^2}}$$

donde α controla el ancho de la vecindad. Además de los problemas de escoger un valor apropiado para α la construcción de estos tipos de grafos sufre problemas de eficiencia. Se reportó en [51] que el enfoque más utilizado para afrontar el costo computacional y de memoria requerido es poner en cero gran parte de los elementos de la matriz.



$$\mathbf{1}_A = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{1}_B = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Figura 4.1: Vectores indicadores

4.2. Notación

Sea W la matriz de adyacencia de G. Para cada vértice el grado d_i es calculado como la suma de los pesos de las aristas que salen de el:

$$d_i = \sum_{j=1}^{n} w_{ij}$$

La matriz de grado D se define como la matriz diagonal tal que $D_{ii} = d_i$

Para un subconjunto A de vértices de V, su vector indicador se denota como $\mathbf{1}_A = (f_1, f_2, ..., f_n)$, donde $f_i = 1$ si el vértice $v_i \in A$ y $f_i = 0$ en el caso contrario.

4.3. Particionado de Grafos

Dado un grafo $G = (V, E)$. Sean dos subconjuntos $A_1, A_2 \subseteq V$ que satisfacen $A_1 \cap A_2 = \emptyset$ y $A_1 \cup A_2 = V$. Se define el corte entre los dos conjuntos como la suma de los pesos de las aristas que conectan a los conjuntos.

$$\text{cut}(A_1, A_2) = \sum_{\substack{v_i \in A_1 \\ v_j \in A_2}} w_{ij}$$

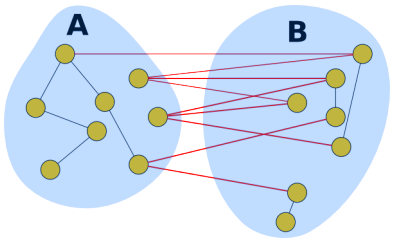
Donde w_{ij} es el peso de la arista entre el vértice v_i y el vértice v_j. El corte describe la semejanza entre los dos subconjuntos del grafo. Un valor pequeño de corte indica mayor separabilidad de los dos subconjuntos.

Wu y Leahy [52] propusieron un método de agrupamiento basado en el criterio de corte mínimo denominado MinCut. El método busca particionar el grafo en k subgrafos de forma tal que el máximo corte entre los subgrafos sea minimizado. Este problema puede ser resuelto de forma eficiente calculando recursivamente los cortes mínimos que dividen los segmentos existentes. Los autores mostraron que este criterio global óptimo puede ser usado para producir buenos agrupamientos en algunas imágenes.

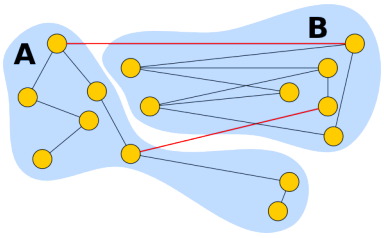
Es posible extender la definición de corte de dos conjuntos para el caso de múltiples conjuntos

$$cut(A_1, A_2, ..., A_k) = \sum_{i=1}^{k} cut(A_i, \overline{A_i})$$

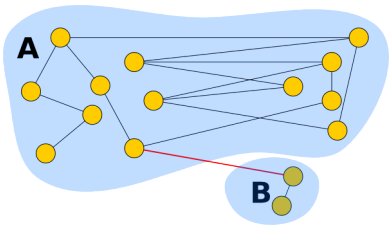
donde $\overline{A_i}$ es el conjunto complemento de A_i, es decir, $\{v_j | v_j \notin A_i\}$. Si bien el corte mínimo es una forma intuitiva para particionar el grafo, sufre de un problema crítico, mostrado en la figura 4.2. En muchos casos separa pequeños conjuntos de vértices del resto del grafo, debido a que el valor de corte es pequeño. Es posible, sin embargo, incorporar algunas estrategias de normalización de la función objetivo para evitar éste problema. Una función objetivo altamente utilizada es el criterio de cortes normalizados.



(a) Un valor de corte alto. cut$(A, B) = 9$



(b) Un valor de corte bajo. cut$(A, B) = 2$



(c) Un valor de corte bajo. cut$(A, B) = 1$

Figura 4.2: Criterio de corte mínimo. Considerando todas las aristas con un peso igual a 1, es fácil notar que el criterio MinCut puede producir pequeños grupos de patrones. A menor valor de cut mejor la partición escogida.

4.4. Cortes Normalizados

El criterio de cortes normalizado fue propuesto por Shi y Malik [6] con el objetivo de evitar el inconveniente de obtener clústeres de pequeñas cantidades de vértices. Tal como lo hace MinCut, el criterio de cortes normalizados mide cuán relacionados están dos o más grupos. La diferencia radica en que en lugar de calcular el valor total de los pesos de las aristas que conectan la partición con el resto del grafo, mide la proporción que esas aristas representan para el conjunto con respecto al total de las aristas que salen de él.

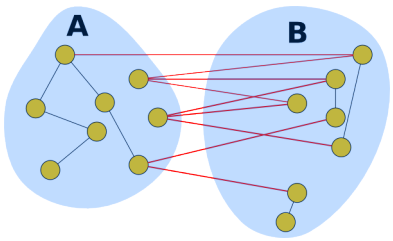
La función objetivo Ncut se define como

$$\text{Ncut}(A_1, ..., A_k) = \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A_i})}{\text{assoc}(A_i, V)}$$

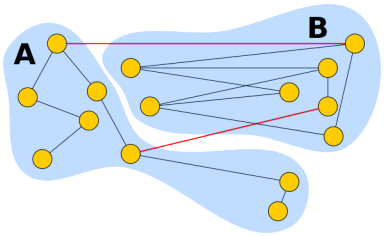
donde $\text{assoc}(A_i, V) = \sum_{v_j \in A_i} d_j$.

Con éste criterio de partición el corte que devuelve patrones aislados no tendrá un valor pequeño de Ncut. Las aristas que conectan a los vértices del conjunto con el resto del grafo representarán un gran porcentaje del total de aristas que salen de el como muestra la figura 4.3.

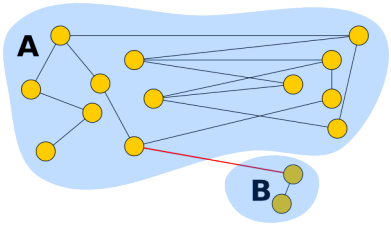
La incorporación de la normalización de la función objetivo convierte al problema de partición de grafos en un problema NP-Hard [53]. Sin embargo, Shi y Malik mostraron que es posible encontrar buenas particiones de forma eficiente expresando el criterio como un problema de minimización de traza. La expresión del problema en forma matricial utiliza un tipo de matriz derivada de los datos denominada matriz laplaciana del grafo [48].



(a) Ncut$(A, B) = \frac{9}{14} + \frac{9}{12} = 0{,}69$



(b) Ncut$(A, B) = \frac{2}{9} + \frac{2}{10} = 0{,}42$



(c) Ncut$(A, B) = \frac{1}{14} + \frac{1}{2} = 0{,}57$

Figura 4.3: Criterio de cortes normalizados. Considerando todas las aristas con un peso igual a 1, es posible notar que el criterio de cortes normalizados entrega grupos de tamaño balanceado. A menor valor de Ncut mejor la partición escogida.

4.4.1. Matriz Laplaciana del grafo

La matriz laplaciana del grafo se define como

$$L = D - W$$

A continuación se detallan las propiedades más importantes de L [54] [55]:

1. Para cualquier vector arbitrario $f \in \Re^n$

$$f^t L f = \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2$$

2. L es simétrica y positiva semi definida

3. El autovalor más pequeño es 0 con autovector $\mathbf{1}$

4. L tiene n autovalores no negativos $0 = \lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$

La relación de la matriz laplaciana con el criterio de cortes normalizados puede ser observada si se considera la utilización de un vector indicador escalado. Un vector indicador escalado f para un conjunto A tendrá la forma $f_A = c\mathbf{1}_A$ es decir:

$$f_{A_i} = \begin{cases} c & \text{si } i \in A \\ 0 & \text{caso contrario} \end{cases}$$

Considerando la propiedad 1 de la matriz laplaciana:

$$\begin{aligned} f^t L f &= \frac{1}{2} \sum_{i,j=1}^{n} w_{ij}(f_i - f_j)^2 \\ &= \frac{1}{2} \left(\sum_{\substack{v_i \in A \\ v_j \in A}} w_{ij}(c-c)^2 + \sum_{\substack{v_i \notin A \\ v_j \in A}} w_{ij}(-c)^2 + \sum_{\substack{v_i \in A \\ v_j \notin A}} w_{ij}(c)^2 \right) \\ &= \frac{1}{2} \left(2 \sum_{\substack{v_i \in A \\ v_j \notin A}} w_{ij} c^2 \right) \\ &= c^2 cut(A, \overline{A}) \end{aligned}$$

Si cada vector indicador es escalado por $c = \frac{1}{\sqrt{assoc(A,V)}}$ entonces

$$f^t L f = \frac{\text{cut}(A,\overline{A})}{\text{assoc}(A,V)} = \text{Ncut}(A, \overline{A})$$

4.5. Expresión en forma matricial

Sea $F \in \Re^{n \times k}$ la matriz formada por k vectores indicadores $f_1, f_2, ..., f_k$ puestos en columnas. Si se considera que:

$$f_i^t L f_i = (F^t L F)_{ii}$$

donde $(F^t L F)_{ii}$ es el elemento i-ésimo de la diagonal principal de $F^t L F$, entonces es posible expresar el criterio de cortes normalizados como:

$$\begin{aligned}\text{Ncut}(A_1, ..., A_k) &= \sum_{i=1}^{k} \frac{\text{cut}(A_i, \overline{A_i})}{\text{assoc}(A_i, V)} \\ &= \sum_{i=1}^{k} f_i^t L f_i \\ &= \text{Tr}(F^t L F)\end{aligned}$$

Donde Tr() denota la traza de la matriz.

Es importante destacar que los vectores indicadores son ortogonales y que

$$f_i^t D f_i = \sum_{j=1}^{n} d_j f_{ij}^2 = \frac{\sum_{v_j \in A_i} d_j}{\text{assoc}(A_i,V)} = \frac{\text{assoc}(A_i,V)}{\text{assoc}(A_i,V)} = 1$$

es decir que $F^t D F = I$.

Entonces minimizar el criterio de cortes normalizados equivale a encontrar la matriz H, es decir los k vectores indicadores escalados de los subconjuntos $A_1, A_2, ..., A_k$, que minimizan la traza de la matriz $F^t L F$, sujeto a la restricción $F^t D F = I$:

$$\begin{aligned}\min_{A_1,...,A_k} \quad & \text{Tr}(F^t L F) \\ \text{s.t.} \quad & F^t D F = I\end{aligned}$$

Debido a que que f_{ij} puede valer solo 0 o $\frac{1}{\sqrt{assoc(A_i,V)}}$, la función objetivo es discreta. El carácter discreto de los vectores indicadores convierte al problema en NP-Hard.

4.6. Aproximación de cortes normalizados

Para abordar computacionalmente el problema es posible modificar la restricción es posible debilitar la restricción que impone un estructura discreta de los vectores indicadores.

$$\begin{aligned}\min_{F \in \Re^{n \times k}} \quad & \text{Tr}(F^t L F) \\ \text{s.t.} \quad & F^t D F = I\end{aligned}$$

Una última conversión a la expresión del problema es indispensable para convertirlo en un problema de minimización cuya solución puede ser encontrada de forma eficiente. Si se substituye $T = D^{\frac{1}{2}}F$ se obtiene un problema de minimización de traza estándar [56][50]:

$$\begin{aligned} \min_{T \in \Re^{n \times k}} \quad & \mathrm{Tr}(T^t D^{-\frac{1}{2}} L D^{-\frac{1}{2}} T) \\ \text{s.t.} \quad & T^t T = I \end{aligned}$$

La matriz $L_{\mathrm{sym}} = D^{-\frac{1}{2}} L D^{-\frac{1}{2}}$ se la conoce como matriz laplaciana normalizada.

Este problema de minimización de traza estándar se resuelve cuando la matriz T contiene los primeros k autovectores de L_{sym} como columnas [8] [56]. Una vez obtenido los autovectores de L_{sym} es posible calcular la matriz H volviendo a substituir $H = D^{-\frac{1}{2}}T$. Esta matriz contendrá una aproximación continúa a los vectores discretos buscados.

Es necesario como último paso convertir los vectores reales continuos a particiones discretas. Para esto se debe aplicar un algoritmo de agrupamiento como k-medias, utilizando a las filas de la matriz como patrones. Según se detalla en [50] [8] al utilizar los autovectores de la matriz L_{sym}, previo a la etapa final de agrupamiento es necesario normalizar las filas de los patrones debido a que pueden ocurrir perturbaciones si se encuentran vértices con grados muy bajos.

4.7. Algoritmo de clsutering espectral normalizado

Por lo expuesto anteriormente, es posible resumir el algoritmo de clustering espectral normalizado en seis etapas básicas. Dado un conjunto de patrones $X = \{x_1, x_2, ..., x_n\}$

1. Construir el grafo de semejanza de X utilizando unos de los métodos descriptos. Sea W la matriz de adyacencia del grafo y D la matriz de grados.

2. Calcular la matriz laplaciana normalizado $L_{\mathrm{sym}} = D^{-\frac{1}{2}}(D - W)D^{-\frac{1}{2}}$.

3. Determinar $f_1, f_2, ..., f_k$ los k autovectores correspondientes a los autovectores más pequeños.

4. Construir la matriz $F \in \Re^{n \times k}$ a partir de $f_1, f_2, ..., f_k$ puestos en columnas.

5. Normalizar las filas de F de forma que $\forall_{i=1,2,\ldots,n} \sqrt{\sum_{j=1}^{k} F_{ij}^2} = 1$

6. Tratar cada fila de F como un vértice en $\Re^k$, particionar éstos vectores en k clústeres.

4.7.1. Análisis de complejidad

Un problema crítico para la implementación de los algoritmos de agrupamiento espectral es la memoria necesaria para almacenar las matriz laplaciana, cuyo número de elementos es el cuadrado de la cantidad de patrones. Es por esto que en la práctica se utilizan los grafos de vecinos más próximos o los grafos de vecindad ϵ.

Una implementación típica para construir un grafo de vecinos más próximos es la descripta en [57]: Se utiliza una max heap de tamaño t. Se inserta de forma secuencial los patrones que se encuentran a una distancia que es menor al valor máximo de la heap y luego se reconstruye la heap. Debido a que reconstruir una heap es de orden $log(t)$, la complejidad de generar una matriz dispersa es:

$$O(n^2) + O(n^2 log(t))$$

El costo $O(n^2)$ puede ser reducido utilizando arboles KD, u otras estructuras de particionado de espacio. En algunos conjuntos de datos es posible obtener en tiempo constante los vecinos de un patrón dentro de un radio r. En el caso de las imágenes, por ejemplo, dado un píxel ubicado en la posición (x, y) es posible obtener en tiempo constante todos los píxeles vecinos ubicados dentro de un radio r. En este caso la construcción del grafo de radio ϵ tiene complejidad temporal lineal.

Resolver un problema de autovalores estándar toma $O(n^3)$ operaciones [7], siendo n el número de vértices en el grafo. Eso lo vuelve impracticable para volúmenes de datos muy grandes. Afortunadamente, los grafos están localmente conectados por lo que resultan muy dispersos y solo se requieren unos pocos autovectores extremos. Debido a que las matrices de grafos no dirigidos son simétricas y positiva semi definida es posible utilizar el algoritmo de Lanczos [58] que puede calcular los autovectores en $O(mn)$ [59] donde m es el número de aristas del grafo, es decir, el número de elementos no cero de la matriz de adyacencia. Para matriz dispersas con un número de aristas proporcional al número de vértices da un tiempo de ejecución de

$$O(n^2)$$

La complejidad temporal de la última etapa dependerá del algoritmo de clustering utilizado para particionar el espacio de autovectores. Sin embargo, el espacio generado tendrá una dimensión relativamente pequeña por lo que el orden de ejecución no debería dominar sobre las etapas anteriores.

4.8. Agrupamiento espectral aplicado en imágenes

Los algoritmos de clustering espectral han sido aplicado con éxito en el contexto de segmentación de imágenes en diversas investigaciones [7] [35] [60]. En estos trabajos se construye una matriz de afinidad dada una imagen de entrada F. La creación del grafo pesado no dirigido se realiza utilizando cada píxel de la imagen como un vértice, tal como muestra la figura 4.4. Los pesos de las aristas que unen los vértices deben reflejar la probabilidad de que dos píxeles pertenezcan a un mismo objeto. En imágenes de intensidad tradicionales es posible utilizar el valor de intensidad del píxel y su ubicación espacial. En el artículo escrito por Shi y Malik [6] recomiendan utilizar la función de pesos

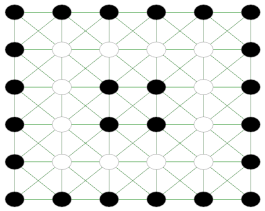
$$W(i,j) = e^{\frac{-\|F(i)-F(j)\|_2}{\alpha_I}} * \begin{cases} e^{\frac{-\|X(i)-X(j)\|_2}{\alpha_X}} & \text{si } \|X(i)-X(j)\|_2 \leq r \\ 0 & \text{caso contrario} \end{cases}$$

en donde $X(i)$ es la coordenada espacial (x, y) del píxel i y α_X, α_I, r son parámetros configurables de la función. El grafo generado por ésta función es un grafo de vecindad r.

En la figura 4.4 se muestra una imagen de entrada F y el grafo G generado utilizando la función propuesta por Shi y Malik con $\alpha_I = 0{,}1, \alpha_X = 0{,}1$ y $r = 2$.

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

(a) Imagen de entrada



(b) Grafo G de vecindad $\epsilon = 2$ generado a partir de la imagen.

Figura 4.4: Imagen de entrada y grafo generado a partir de ella

Una vez calculada la matriz L_{sym} derivada de G, es posible obtener los autovectores y embeber a los píxeles de la imagen en un espacio de dimensión inferior para realizar el agrupamiento. En la figura 4.5 se muestra la representación de los píxeles de la imagen en un espacio de dimensión 1 generado por el autovector correspondiente al tercer autovalor mas pequeño de L_{sym}.

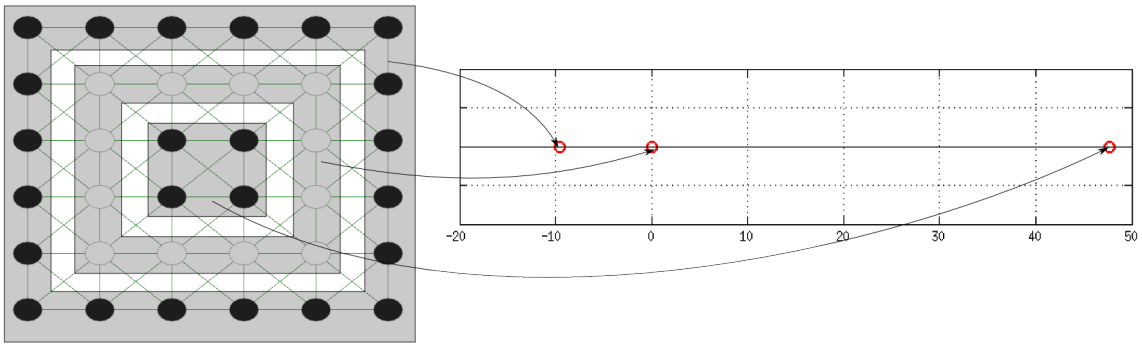


Figura 4.5: Patrones embebidos en un espacio de dimensión 1.

En el nuevo espacio generado realizar el agrupamiento es más sencillo debido a que los píxeles de cada objeto resultan en clústeres de mínima varianza. Los resultados del algoritmo de segmentación son mostrados en la figura 4.5.

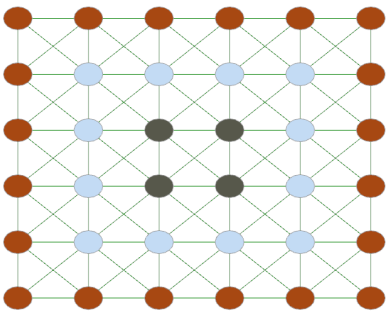


Figura 4.6: Resultado del algoritmo de clustering

Capítulo 5

Segmentación de imágenes ToF

Recientemente han surgido diversas investigaciones cuyo objetivo es combinar adecuadamente la información de intensidad y de rango provista por las cámaras de tiempo de vuelo para mejorar los resultados de la segmentación [61] [2]. A continuación se proponen dos métodos que combinan la información de la imagen de intensidad y la imagen de rango en coordenadas cartesianas para realizar la segmentación.

El primero consiste en un algoritmo de detección de objetos parcialmente ocluidos. Realiza, en una primera etapa, una extracción de bordes mejorada combinando la información útil de las dos imágenes. Luego, a partir de las regiones delimitadas por los bordes, analiza cuáles pertenecían al mismo objeto, pero debido a una oclusión parcial fue segmentado como dos.

El segundo método es un algoritmo de segmentación espectral que combina los autovectores de los grafos de semejanza obtenidos a partir de las dos imágenes para realizar el particionado.

El entorno SpectralGUI implementa el algoritmo presentado en el capítulo 4 y el algoritmo de agrupamiento espectral propuesto. Permite visualizar todas las etapas intermedias del método y evaluar los resultados de los algoritmos.

En la siguiente sección se detalla el entorno TOFCapture, la segunda sección detalla el algoritmo de segmentación de objetos parcialmente ocluidos, en la tercera sección el algoritmo de clustering espectral propuesto y la última sección detalla el entorno SpectralGUI.

5.1. Aplicación TOFCaptureGUI

TOFCapture es una aplicación multiplataforma que permite la captura, almacenaje y documentación de las imágenes obtenidas con la cámara SwissRanger SR4000. El software utiliza como base solamente las funciones multiplataforma de la librería descripta en 3.5 provista por la empresa y el framework Qt [62] para la interfaz gráfica.

El desarrollo de la aplicación fue motivado por la falta de portabilidad por parte de la librería provista por la empresa SwissRanger: las funcionalidades de la librería relacionadas con el descubrimiento de la cámara en la red y la persistencia de las captura está restringida solamente a los sistemas operativos Microsoft Windows. Otra motivación fue la carencia de funciones destinadas a la documentación de las imágenes.

La aplicación provista por la empresa, denominada SR 3D View funciona solamente en los sistemas operativos Microsoft Windows y permiten capturar las imágenes en formato ASCII, pero no almacenan los parámetros de la cámara al momento de hacer la captura. Es posible almacenar, sin embargo, en el formato propietario de la empresa, que sí almacena la información relacionada con la cámara pero no es posible abrir este formato de archivo en otro sistema operativo.

TOFCapture fue desarrollado con el objetivo de facilitar la captura, documentación y visualización de las imágenes para luego poder aplicar algoritmos de segmentación sobre las mismas. Uno de los objetivos principales consiste en documentar cuáles son los parámetros de configuración de la cámara al momento de hacer la captura, y poder adjuntar algún tipo de comentario adicional sobre las figuras. Otro requisito es el de guardar las imágenes en un formato portable. Por último, es importante poder visualizar de forma sencilla las capturas realizadas para luego poder hacer una selección de las mejores.

El software, como muestra la figura 5.1, se divide en tres componentes principales: una librería para realizar capturas con la cámara llamada TOFCapture, una librería para la persistencia y documentación de las imágenes llamada TOFCaptureIO y un entorno gráfico, denominado TOFCaptureGUI, que provee una interfaz amigable que utiliza las funciones de las dos librerías anteriores

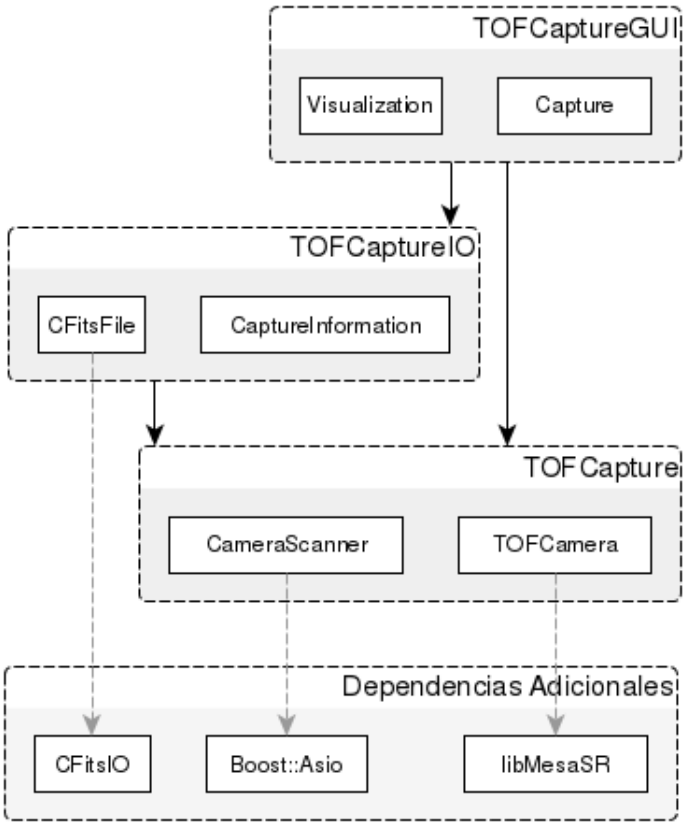


Figura 5.1: Diagrama de dependencias de TOFCaptureGUI

5.1.1. TOFCapture

TOFCapture es la librería base que hace de mediador entre la cámara y el usuario. Permite descubrir las cámaras conectadas a la red ethernet y realizar capturas con las cámaras seleccionadas. Utiliza la librería Boost::ASIO para localizar la cámara en la red y la librería libMesaSR provista por la empresa SwissRanger para configurar la cámara y realizar capturas.

La librería está compuesta por dos clases principales: **TOFCamera** que representa a la cámara y **CameraFinder** que realiza la búsqueda de la cámara en la red. La clase **TOFCamera** es la clase que hace de mediador entre la API de SwissRanger y el usuario. Contiene los métodos de apertura y clausura de la cámara. El método de apertura recibe la IP de la cámara que se desea abrir. Una vez abierta es posible especificar los parámetros de configuración descriptos en el capítulo 3 y realizar capturas.

```
1 void TOFCamera::openCamera(const string & ip);
2 void TOFCamera::closeCamera();
```

El método **TOFCamera::capture** entrega la imagen de amplitud, la imagen de rango en coordenadas esféricas y la imagen de rango en coordenadas cartesianas. Cada una de las imágenes es una instancia de la clase **Image**. La imagen de amplitud y la imagen de rango en coordenadas esféricas utilizan unsigned short como tipo de datos para cada píxel y la imagen de rango en coordenadas cartesianas utiliza un punto 3D (x, y, z) de tipo double por cada píxel de la imagen. Los valores de los datos corresponden a los entregados por la librería de la empresa.

```
1 void TOFCamera::capture(AmplitudeImage &amplitudeImage,
2                         RangeImageSpherical &rangeImageSpherical,
3                         RangeImageCartesian &rangeImageCartesian);
```

El método **vector<string> getIpByScanning()** de la clase **CameraFinder** busca las cámaras de tiempo de vuelo SR4000 conectadas a la red por el puerto ethernet. La cámara escucha peticiones en el puerto 2477. TOFCapture envía un paquete de 4 bytes a la dirección de broadcast de IPv4 por el puerto 2477. Cuando la cámara recibe el paquete, envía un mensaje a la IP que realizó la petición indicando el número de modelo. Cuando TOFCapture recibe los paquetes de las cámaras puede extraer la dirección IP desde donde provinieron.

5.1.2. TOFCaptureIO

TOFCaptureIO es el componente encargado de almacenar y recuperar las imágenes desde el disco. Almacena las imágenes de intensidad, de rango en coordenadas cartesianas y de rango en coordenadas esféricas. Almacena, a su vez, los parámetros de captura: tiempo de integración, frecuencia de modulación, umbral de amplitud, máximo rango de medición, fecha de captura y una descripción provista por el usuario.

TOFCaptureIO almacena las imágenes en el formato FITS (Flexible Image Transport System) [63]. FITS es un estándar abierto, cuya última versión fue aceptada en 2008, que define un formato de archivo útil para el almacenamiento, transmisión y procesamiento de imágenes científicas. A diferencia de otros formatos, FITS fue diseñado específicamente para almacenar datos científicos y por lo tanto incluye muchas herramientas para la descripción precisa del origen de los datos como el manejo de unidades de medición y de diferentes sistemas coordenados.

Una de las principales virtudes del formato FITS es que los metadatos de las imágenes son almacenados en una cabecera ASCCI legible, de forma que el interesado puede examinar las cabeceras de un archivo de procedencia desconocida. Es posible, además, almacenar más de una imagen por archivo. El formato admite otras fuentes de información además de imágenes como datos estructurados.

TOFCapture cuenta con la clase **CaptureInformation** que almacena toda la información referente a una captura: las imágenes de rango, y la imagen de amplitud y los parámetros de la captura. La clase TOFDataIO contiene métodos estáticos para almacenar y recuperar la información de las capturas:

```
1  void TOFCaptureIO::save(const string & filename, CaptureInformation &info);
2  CaptureInformation *TOFCaptureIO::read(const string &filename);
```

La clase TOFCaptureIO utiliza la clase **FitsFile** para almacenar los datos.

5.1.3. TOFCaptureGUI

La interfaz gráfica provista por TOFCaptureGUI permite utilizar las librerías TOFCapture y TOFCaptureIO de forma sencilla. El software presenta dos ventanas principales: la primera de ellas, mostrada en la figura 5.2 es utilizada para capturar y almacenar las imágenes. En la barra lateral permite seleccionar los parámetros de captura y permite registrar la imagen deseada.

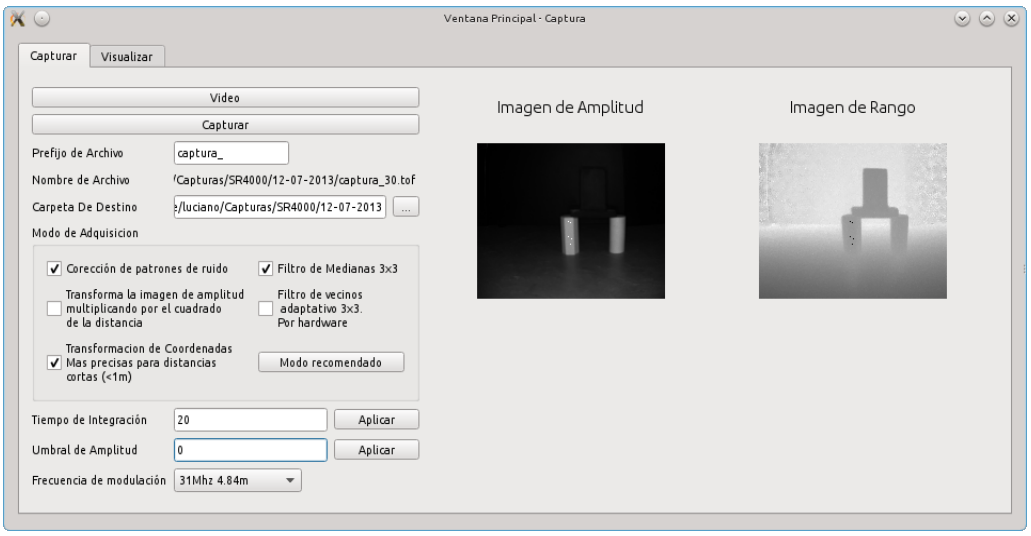


Figura 5.2: Captura de imágenes

La segunda ventana se utiliza para visualizar las imágenes capturadas dentro de una carpeta. Como se puede ver en la figura 5.3, una vez seleccionada una carpeta, el programa lista todas las imágenes junto con sus metadatos.

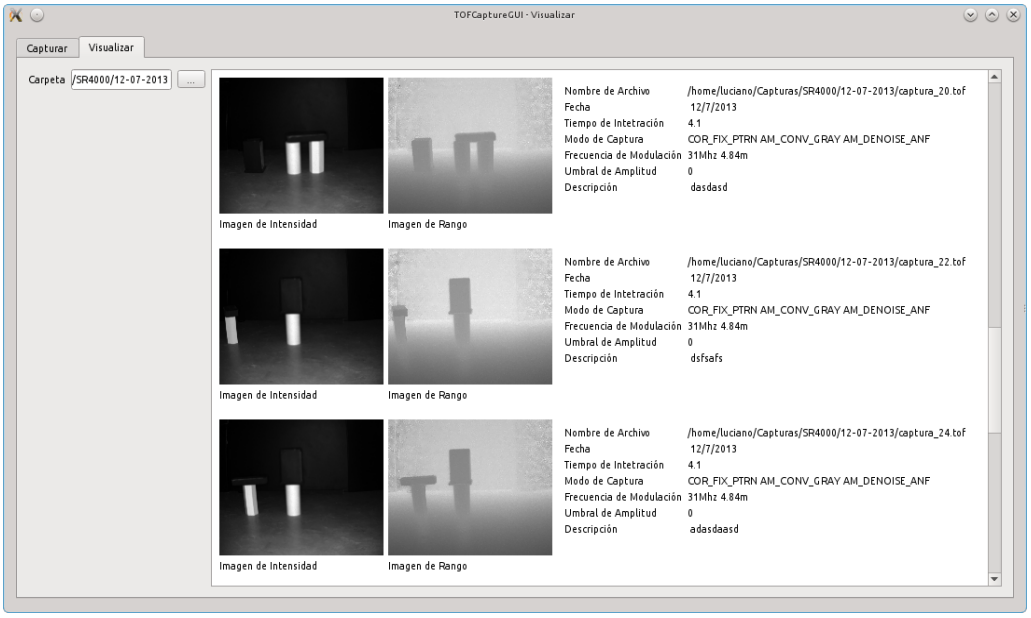


Figura 5.3: Visualización de capturas realizadas

5.2. Segmentación de objetos parcialmente ocluidos

Diversos métodos se utilizan en imágenes de intensidad para la detección de bordes basados en operadores derivativos. Tanto las imágenes de intensidad como de rango son afectadas por el ruido. Es necesario, por lo tanto, considerar el efecto que este ruido produce en las imágenes al aplicar operadores basados en derivadas de primer y segundo orden. Entre los detectores de bordes clásicos más utilizados, que consideran la influencia del ruido en el proceso de segmentación, podemos mencionar un método robusto como el algoritmo de Canny [64]. Este algoritmo minimiza la inclusión de falsos bordes, minimiza, a su vez, la distancia entre los bordes reales y los detectados y establece un criterio que integra respuestas múltiples que corresponden a un mismo borde.

En las imágenes de intensidad generadas por una cámara de tiempo de vuelo predominan los detalles generados por la textura de los objetos y en las de rango los detalles y ruido asociados al fondo de la escena 3D. Recientemente han sido propuestas técnicas para segmentar objetos que operan sobre imágenes de rango e intensidad con el objetivo de definir bordes más precisos en presencia, tanto de ruido como de oclusiones en distintos planos [65] [2] [5]. Estos métodos intentan determinar los bordes de los objetos en forma más precisa, no siempre obteniendo contornos cerrados. La falta de contornos cerrados impide la consideración de oclusiones parciales y por lo tanto la obtención de una segmentación correcta.

El método presentado a continuación propone una solución integral para los dos problemas anteriores utilizando información útil de las dos imágenes generadas por una cámara de tiempo de vuelo. Para esto utiliza el método propuesto por Danciu, et ál. [2] para obtener información de bordes adecuada. Obtenidos los segmentos clausurados, se realiza una etapa que determina cuáles de esos conjuntos de puntos delimitados por los contornos eran originalmente un solo objeto pero fue segmentado como dos debido a una oclusión parcial.

5.2.1. Extracción de contornos usando operaciones lógicas

Danciu, et ál. [2] proponen un método mejorado de extracción de contornos para segmentación de imágenes adquiridas con cámaras TOF, que combina información de las aristas obtenidas a partir la imagen de distancia y de la imagen de intensidad. Para esto definen y utilizan una operación lógica basada en la operación AND que considera cada píxel y su vecindad para

determinar la presencia de contornos, como muestra la figura 5.4.

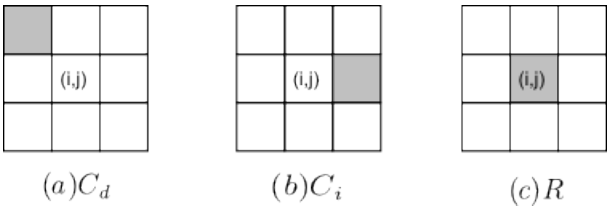


Figura 5.4: Operador AND aplicando a los vecinos 8 de (i,j) a) contornos de la imagen de distancia, b) contornos de la imagen de intensidad, c) imagen resultante

Sean $D(x, y)$ e $I(x, y)$ las imágenes de distancia e intensidad, respectivamente, que provee la cámara TOF, ambas de dimensión n x m. El método propuesto obtiene una imagen resultante $R(x, y)$ de dimensión n x m que contiene información mejorada de los contornos de los objetos con respecto a la que se podría obtener utilizando un algoritmo de detección de bordes a $D(x, y)$ o $I(x, y)$ por separado. El método es descripto a continuación:

1. Se aplica el método Canny a las imágenes $D(x, y)$ e $I(x, y)$.
   Sea $B_d(x, y)$ la información de bordes extraída a partir de la imagen de distancia y sea $B_i(x, y)$ la información de bordes extraída a partir de la imagen de intensidad. $B_d(x, y)$ y $B_i(x, y)$ de dimensión n x m.

2. Por cada píxel (p, k) se inspecciona su entorno delimitado por un círculo de radio 2 en $B_i(x, y)$ y $B_d(x, y)$ simultáneamente.
   Sea $C_{B_d(p,k)}$ el entorno del píxel (p, k) de $B_d(x, y)$ y $C_{B_i(p,k)}$ el entorno del píxel (p, k) de $B_i(x, y)$.

3. Si al menos un píxel de contorno es encontrado $C_{B_d(p,k)}$ y en $C_{B_i(p,k)}$ entonces el píxel (p, k) en $R(x, y)$ es considerado como un píxel de contorno.

5.2.2. El método propuesto

El método propuesto se divide en dos etapas principales: la primera consiste en extraer correctamente las fronteras de los objetos capturados y la segunda en determinar cuáles de esos conjuntos de puntos delimitados por los contornos eran originalmente un solo objeto pero fueron segmentados como dos debido a una oclusión parcial.

Extracción de Bordes Inicial

1. Se utiliza el método descripto en la sección anterior para obtener una imagen de bordes $R(x, y)$

Clausura de Contornos

2. $NR = NOT(R(x, y))$

3. Utilizando NR localizan los componentes conectados W_i en la imagen de rango.

4. Por cada componente conectado W_i se estima el plano que mejor describe a ese conjunto y se calcula el error cuadrático medio SSE_{W_i} [66].

5. Si $\sum_{i=1}^{n} SSE_{W_i} >$ umbral entonces se dilatan los bordes de R y se continúa en 2, sino continúa en 6.

Análisis de Objetos Ocluidos

6. Se calcula para cada conjunto W_i

   - La media de intensidad.
   - La media de distancia.
   - El centroide de la nube de puntos.

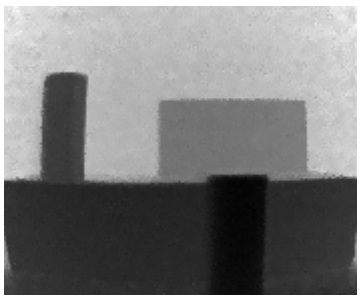
   y se estima

   - El vector normal del plano que mejor que describe a la superficie [66].

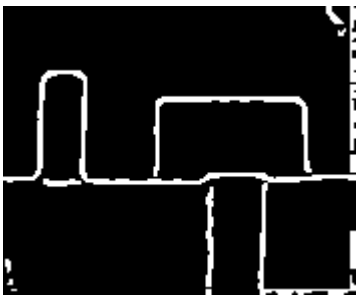
7. Se comparan los componentes conectados entre sí buscando la similitud entre las características calculadas en el paso anterior. Si dos conjuntos resultan similares se unen.

5.2.3. Resultados Experimentales

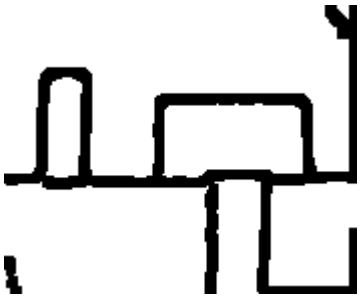
Aplicamos el algoritmo propuesto sobre una serie de capturas realizadas con la cámara TOF SwissRanger SR4000



(a) Imagen de Intensidad



(b) Extracción de Bordes Inicial

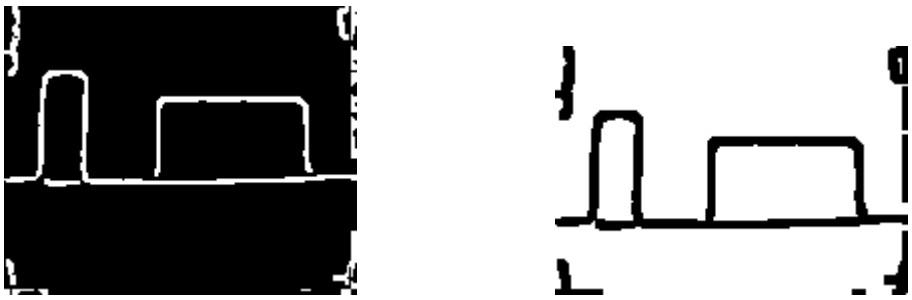


(c) Contornos Clausurados

(d) Extracción de Superficies Ocluídas

Figura 5.5: Segmentación utilizando el método propuesto

En la figura 5.5 se pueden visualizar imágenes que describen la información obtenida en cada etapa de algoritmo. La imagen 5.5b muestra los bordes extraídos mediante el método descripto en la secciones anteriores utilizando una imagen de intensidad 5.5a y de rango. En la imagen 5.5c se muestran las fronteras de los objetos correctamente delimitados y en la imagen 5.5d se puede percibir como el objeto que se encontraba parcialmente ocluido fue segmentado como uno sólo.



(a) Extracción de Bordes (b) Contornos Cerrados

Figura 5.6: Clausura de Contornos

En la figura 5.6 se puede observar como el método propuesto cierra correctamente los contornos de los objetos posibilitando la segmentación posterior. La imagen 5.6a muestra el resultado de aplicar el método descripto. En ella se puede apreciar objetos no correctamente delimitados debido a la presencia de grietas en las fronteras de los mismos. La imagen 5.6b muestra los bordes correctamente cerrados utilizando el método propuesto.

(a) Imagen de Intensidad (b) Extracción de Bordes Inicial

Figura 5.7: Segmentación de Objetos Ocluidos

La figura 5.7 muestra dos imágenes que comparan la segmentación de los objetos delimitados por los contornos basándose solamente en los componentes conectados de a 8 (Figura 5.7a) y los objetos segmentados utilizando el método propuesto (Figura 5.7b). El objeto parcialmente ocluido es segmentado como uno solo puesto que su valor de intensidad, su vector normal y su posición en el espacio se encuentra dentro de un umbral determinado.

5.3. Segmentación espectral de imágenes TOF

Existen diversos métodos que utilizan algoritmos de segmentación espectral sobre las imágenes obtenidas con cámaras de tiempo de vuelo [67] [68] [26]. El método presentado a continuación utiliza el criterio de cortes normalizados para embeber a los patrones en un nuevo espacio generado por los autovectores del grafo de semejanza generado a partir de la imagen de intensidad y el grafo generado a partir de una imagen de rango mejorada.

Teniendo en cuenta las diferencias relativas de profundidad entre los objetos, algunas aristas de los objetos pueden no ser distinguidas [69]. Por esta razón, las aristas de los objetos deben ser realzadas para mejorar la segmentación. En este trabajo, en lugar de utilizar la imagen de rango en coordenadas cartesianas, se adopta el método propuesto en [70] que genera a partir de la imagen de rango en coordenadas cartesianas una imagen de rango mejorada. Esta imagen provee información sobre las orientaciones de los objetos y presenta una información mas precisa sobre los contornos de los objetos.

5.3.1. Imagen de rango mejorada

El método propuesto por Pully [70] consiste en representar a la imagen de rango como una imagen RGB, basándose en el vector normal estimado para ese punto. Dado el vector normal estimado $n_i = [\, x_i \; y_i \; z_i \,]$ para el punto i-ésmo ubicado a una profundidad d_i se genera una imagen RGB en el que el píxel i-esimo tendrá por valor:

$$\begin{cases} \mathrm{R}_{\,i} = & k_1 \cos^{-1}(x_i) \\ \mathrm{G}_{\,i} = & k_2 \sin^{-1}(y_i) \\ \mathrm{B}_{\,i} = & k_3 d_i \end{cases}$$

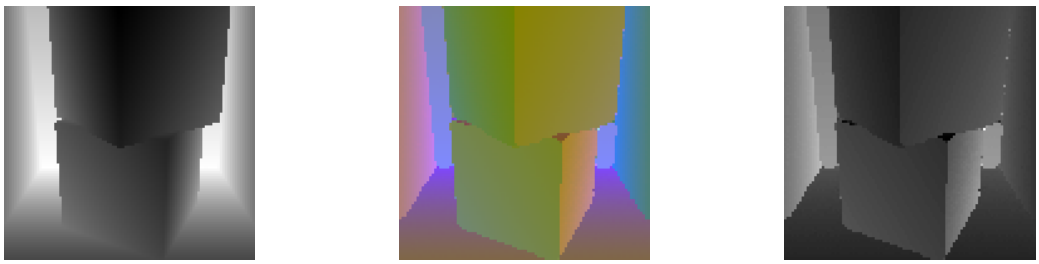
donde k_1, k_2 y k_3 son constantes positivas que escalan los valores de los píxeles al rango $[\, 0, 255 \,]$. Es importante notar que

1. El uso de la componente x_i e y_i es suficiente para representar al vector normal dado que $z_i = \sqrt{1 - x_i^2 - y_i^2}$
2. La información contenida en la imagen determina completamente al punto en $\Re^3$. Esto significa que es posible obtener la información geométrica local (el vector normal a la superficie) y la información global (la información de distancia)

A esta imagen RGB, mostrada en en la figura 5.8b, se la conoce con el nombre de ERI (Ehnacned Range Image) [67]. Luego la imagen RGB es convertida a una imagen de intensidad. Esta imagen es denominada NERI (Normal Ehnacned Range Image) [69] debido a que mejora la aristas y las superficies de los objetos en la imagen de rango que asigna colores diferentes a las superficies que presentan diferentes orientaciones y hace las aristas mas distintivas.

El procedimiento, entonces, para generar la imagen NERI buscada es:

1. Dada una imagen de rango de dimensión $m \times n$.
2. Por cada punto i de la imagen ubicado a una distancia d_i se estima el vector normal $n_i = [x_i \ y_i \ z_i]$.
3. Se construye una imagen RGB de dimensión $m \times n$ done el píxel i tiene como componentes $[x_i \ y_i \ d_i]$.
4. Se convierte la imagen RGB a escala de grises.



(a) Rango (b) Imagen ERI (c) NERI

Figura 5.8: Proceso de creación de la imagen NERI

5.3.2. Método propuesto

El método propuesto cuya representación esquemática puede observarse en la figura 5.9, utiliza la imagen de intensidad y la imagen de rango mejorada para embeber a los píxeles en un nuevo espacio de características donde la información útil de ambas imágenes se puede aprovechar con facilidad. Las imágenes atraviesan un proceso similar en el que se calcula el grafo de semejanza y se obtienen los autovectores asociados a sus matrices laplacianas respectivas.

Luego los autovectores son unidos para generar el nuevo espacio. El agrupamiento, entonces, se realiza en el nuevo espacio de características y por último se retoma la representación original teniendo en cuenta los conjuntos encontrados.

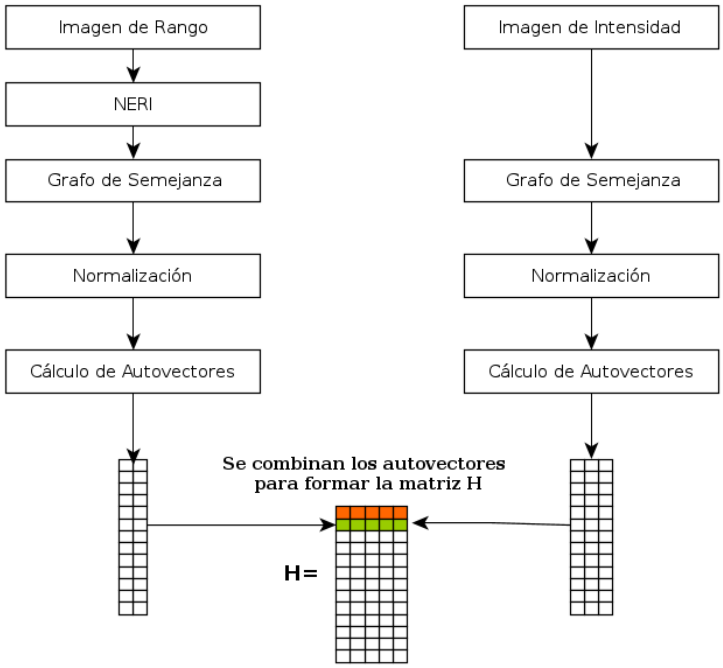


Figura 5.9: Representación esquemática del método propuesto

El algoritmo propuesto se detalla a continuación:

Sea $I(i)$ una imagen de intensidad y $R(i)$ una imagen de rango, ambas de dimensión $n \times m$.

1. A partir de la imagen $R(i)$ se genera una imagen de rango mejorada $NERI(i)$.

2. *a*) Se construye la matriz de afinidad W_I de forma que

$$W_I(i,j) = e^{\frac{-\|F(i)-F(j)\|_2}{\alpha_I}} * \begin{cases} e^{\frac{-\|X(i)-X(j)\|_2}{\alpha_X}} & \text{si } \|X(i)-X(j)\|_2 < r \\ 0 & \text{c. c.} \end{cases}$$

donde $X(i)$ es la locación espacial del nodo i y $F(i) = I(i)$.

b) Se construye la matriz de afinidad W_R de forma que

$$W_R(i,j) = e^{\frac{-\|F(i)-F(j)\|_2}{\alpha_I}} * \begin{cases} e^{\frac{-\|X(i)-X(j)\|_2}{\alpha_X}} & \text{si } \|X(i)-X(j)\|_2 < r \\ 0 & \text{c. c.} \end{cases}$$

donde $X(i)$ es la locación espacial del nodo i y $F(i) = NERI(i)$.

3. *a*) Se calcula la matriz laplaciana normalizada asociada a W_I

$$L_{NI} = D_I^{-1/2}(D_I - W_I)D_I^{-1/2}$$

b) Se calcula la matriz laplaciana normalizada asociada a W_R

$$L_{NR} = D_R^{-1/2}(D_R - W_R)D_R^{-1/2}$$

4. Se genera la matriz $H_I \in R^{k\times n}$ que contiene como columnas primeros k autovectores del sistema $L_{NI}u = \lambda u$. Se genera la matriz $H_R \in R^{l\times n}$ que contiene como columnas los primeros l autovectores del sistema $L_{NR}u = \lambda u$.

5. Se obtienen los autovectores correspondientes a las matrices laplacianas no normalizadas. $PI = D_I^{-1/2}H_I$ y $PR = D_R^{-1/2}H_R$

6. Se normalizan las filas de PI y PR

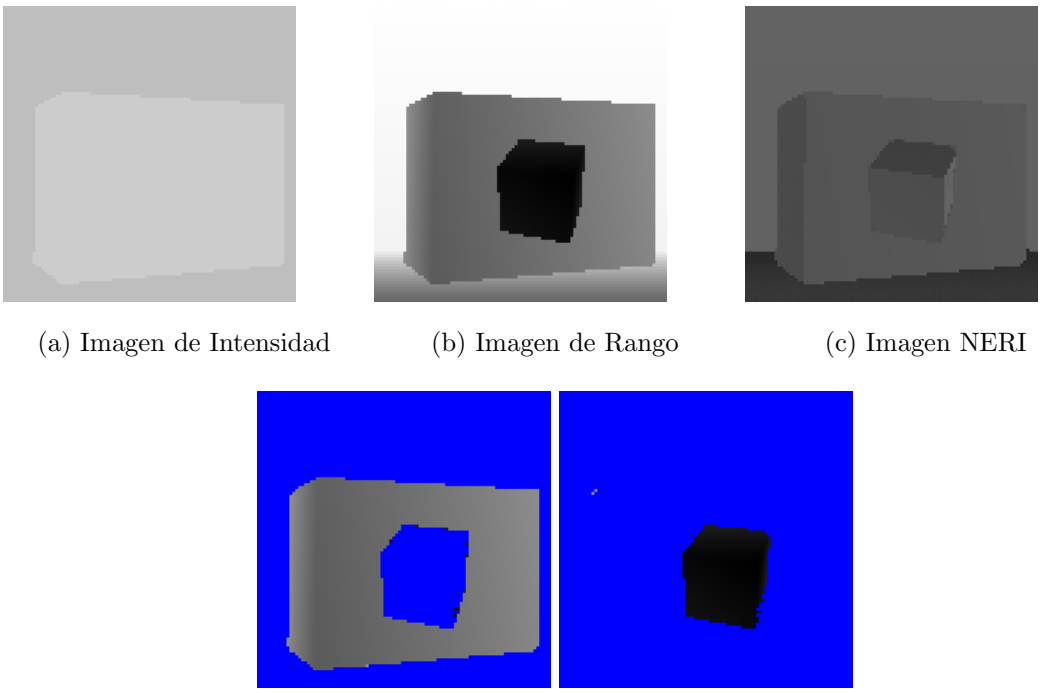
7. Se forma la matriz P_{IR} concatenando las matrices PI y PR

8. Para $i = 1, \ldots, n$, sea $y_i \in R^m$ los vectores correspondientes a la i-ésima fila de P_{IR}, se segmentan los puntos $y_i \in R^m$ con algoritmo k-medias en clusters $C_1, \ldots, C_k$.

5.3.3. Resultados Experimentales

El método propuesto fue aplicado a imágenes artificiales y a imágenes reales. Las capturas reales fueron obtenidas utilizando la cámara de tiempo de vuelo MESA SwissRanger SR4000 [27] y las capturas reales fueron simuladas utilizando el software Blensor [32].

En la figura 5.10 se puede visualizar el resultado de aplicar el algoritmo propuesto sobre una captura artificial.



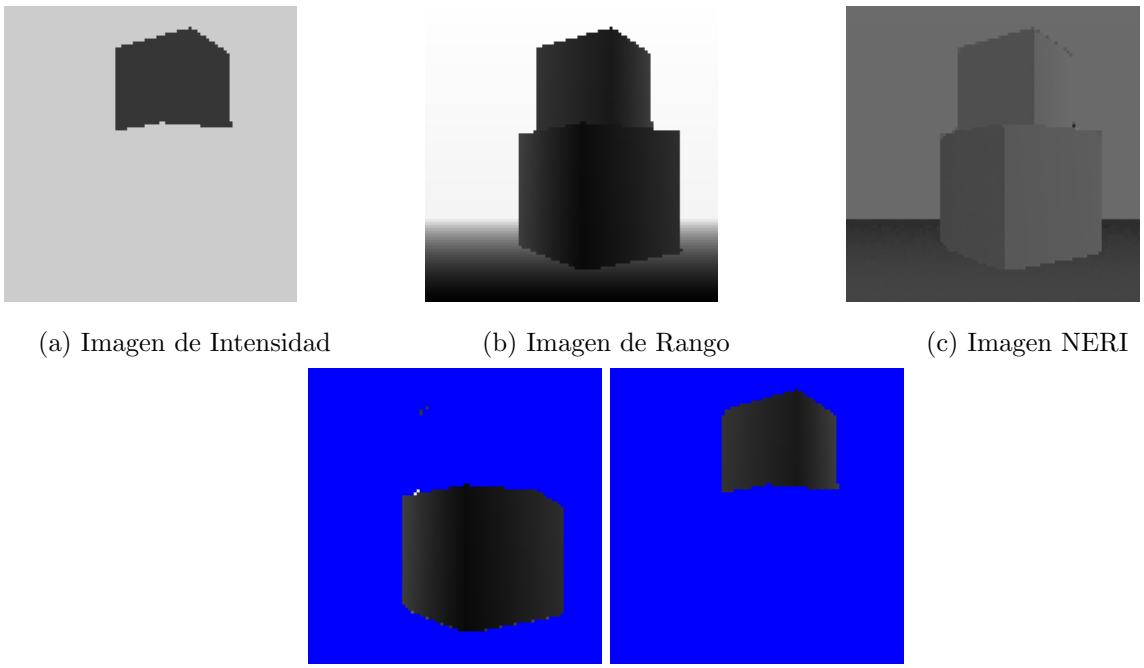
(a) Imagen de Intensidad (b) Imagen de Rango (c) Imagen NERI

(d) Resultado de la Segmentación

Figura 5.10: Segmentación utilizando el método propuesto

La imagen 5.10a muestra dos objetos de niveles de intensidad iguales y ambos con niveles de intensidad próximos al del fondo. En la imagen 5.10b se puede observar que la información de distancia es útil para realizar una segmentación correcta. La imagen 5.10c muestra como la imagen de rango mejorada, incorpora información sobre las orientaciones de los objetos, conservando la información de distancia.

Aplicando k-medias sobre el espacio generado por los autovectores, el algoritmo permite extraer correctamente las partes constitutivas de la imagen como se puede observar en las imágenes 5.10d.



(a) Imagen de Intensidad (b) Imagen de Rango (c) Imagen NERI

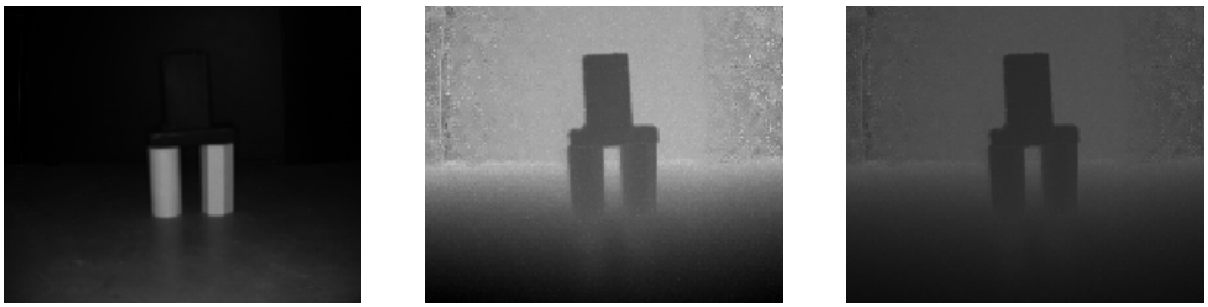
(d) Resultado de la Segmentación

Figura 5.11: Segmentación utilizando el método propuesto

En la figura 5.11 se puede visualizar el resultado de aplicar el algoritmo propuesto sobre otra captura artificial. La imagen 5.11a muestra como en la imagen de intensidad el segundo objeto tiene niveles de intensidades muy próximos a los del fondo y en la imagen 5.11b como los dos objetos que integran la imagen son difíciles de separar uno del otro. El algoritmo puede combinar correctamente la información de las dos imágenes. Utilizando correctamente los datos de distancia para separar el fondo del objeto situado en la parte inferior de la imagen, y los datos de intensidad para separar un objeto del otro como presenta la figura 5.11d.

La figura 5.12 muestra el resultado de aplicar el algoritmo propuesto sobre una captura real. La imagen de amplitud 5.12a presenta 3 objetos sobre un fondo negro, todos a la misma distancia. Uno de los objetos tiene un nivel de intensidad similar al del fondo, lo que dificulta su segmentación. En la imagen de rango 5.12b los objetos se distinguen claramente del fondo pero no uno del otro. El método combina correctamente la información de ambas imágenes ruidosas para segmentar los tres objetos presentes en la escena, como se muestra en la 5.12d.

(a) Imagen de Intensidad (b) Imagen de Rango (c) Imagen NERI



(d) Resultado de la Segmentación

Figura 5.12: Segmentación utilizando el método propuesto sobre una captura real

5.4. SpectralGUI

El entorno SpectralGUI permite la aplicación de algoritmos espectrales de agrupamiento y la visualización de todas las etapas involucradas en el proceso. La aplicación consta de 5 librerías básicas y una interfaz de usuario como muestra la figura 5.13.

Graph

La librería Graph cuenta con la definición de la interfaz de un grafo en la clase abstracta **AbstractGraph**. Implementa un grafo genérico por medio de la clase **Graph** que utiliza una representación similar a la de listas enlazadas [47]. Los grafos pueden ser almacenados a disco por medio de la librería Serialization de Boost [71].

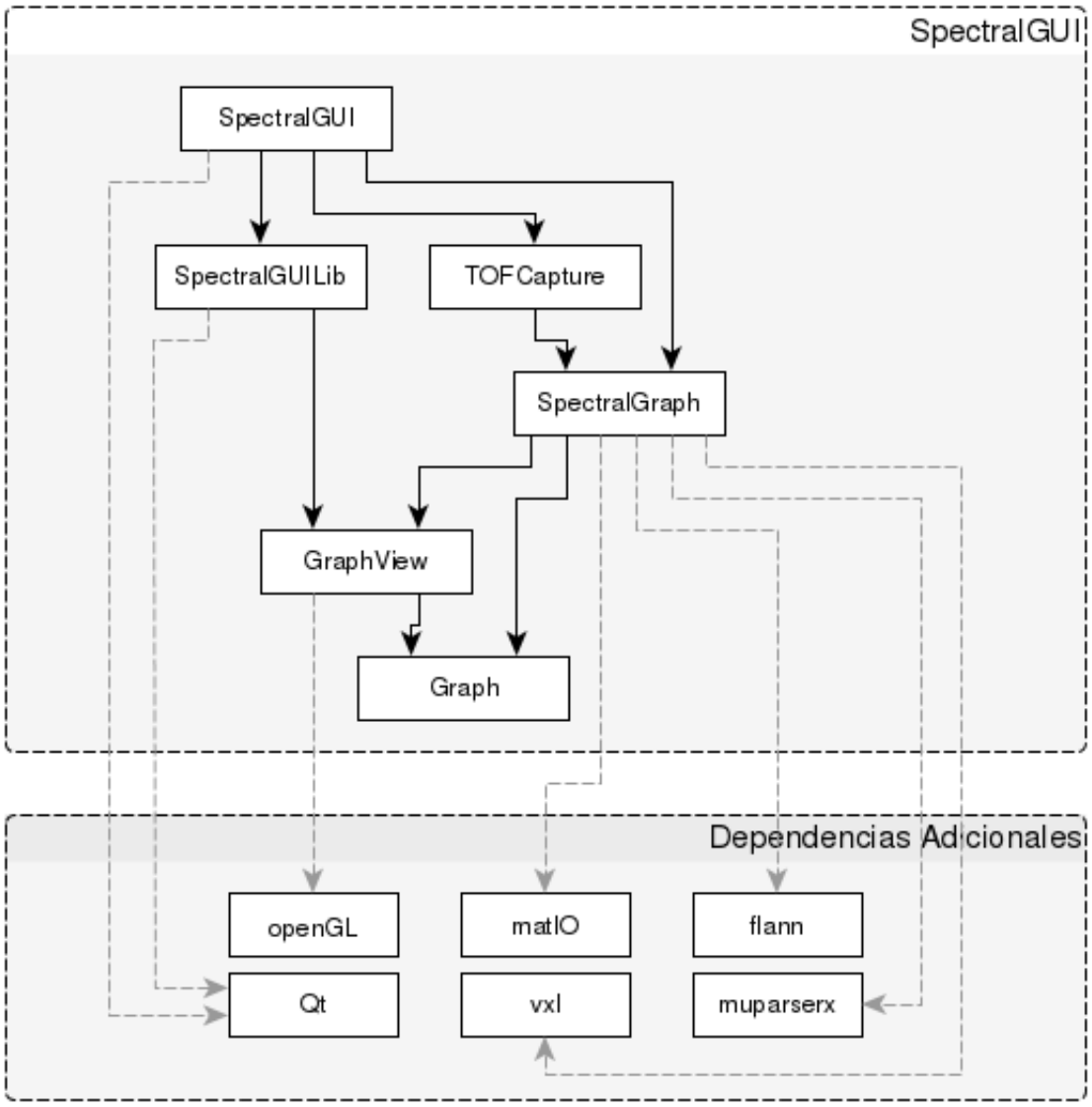


Figura 5.13: Diagrama de dependencias de SpectralGUI

GraphView

GraphView representa una escena compuesta por un conjunto de vértices y de aristas. Implementa las clases necesarias para la visualización de un grafo utilizando OpenGL. Provee un conjunto de Layouts para organizar los nodos en el entorno y las interfaces para construir nodos visuales a partir de datos.

SpectralGraph

El objetivo principal de SpectralGraph consiste en la creación de grafos de semejanza a partir de conjuntos arbitrarios de datos, en el análisis del espectro del grafo resultante y en la aplicación de algoritmos de agrupamiento espectral sobre grafos. Esta librería hace uso del grafo provisto por la librería Graph para analizar su espectro. Los archivos fuentes están dispuestos en un conjunto de directorios que reflejan los componentes principales de la librería:

- EigenvectorCalculator: En esta carpeta se encuentra la clase homónima **EigenvectorCalculator** que permite recibir cualquier tipo de grafo definido y retorna los autovectores y autovalores correspondientes. Hace uso de la librería VXL[72] para realizar la descomposición en autovectores de la matriz laplaciana del grafo. **EigenvectorCalculator** contiene el punto de entrada para la invocación de la librería LASO2 provista por VXL, que implementa el algoritmo de Lanczos con ortogonalización selectiva [73].

- Metric: En esta carpeta se encuentran definidas una serie de funciones de semejanza y desemejanza útiles al momento de la construcción del grafo. La clase **CustomMetric** provee la posibilidad de especificar una función de semejanza o desemejanza en tiempo de ejecución, la expresión es evaluada utilizando la librería muparserx [74].

- DataReader: La interfaz **DataReader** define una conjunto de operaciones para la lectura de datos desde cualquier tipo de fuente. SpectralGraph implementa dos lectores, uno para archivos separados por coma, definido en la clase **CSVReader** y otro que para leer archivos de Matlab .mat **MatReader**. **MatReader** utiliza la librería matIO [75] para realizar la lectura.

- Graph/Builder: Aquí se encuentra la definición de la clase **GraphBuilder** encargada de construir un grafo a partir de un conjunto de datos. El tipo de grafo a construir

dependerá del algoritmo de creación de aristas elegido. Los algoritmos de creación de aristas se encuentran definidos en Graph/Builder/EdgeCreator en donde se encuentra una clase por cada tipo de grafo mencionado en 4.1.

- Segmentation/KWayNCut: Implementa el algoritmo de segmentación propuesto por Shi y Malik explicado en 4.

TOFSegmentation

En este módulo se encuentra la implementación del método espectral propuesto. La clase **TOFSegmentation** permite aplicar el algoritmo sobre imágenes obtenidas con la librería TOF-Capture (.tof) o imágenes simuladas con Blensor (.pcd). Contiene también el código necesario para convertir una imagen de rango a una imagen de rango mejorada. La clase **NormalLTS** estima el vector normal para un conjunto de puntos dados utilizando mínimos cuadrados recortados [70].

SpectralGUILib

Contiene la clase **GraphWidget** que implementa un componente visual para el framework Qt [62] encargado de desplegar en pantalla la escena dibujada por GraphView.

5.4.1. SpectralGUI

SpectralGUI cuenta con la implementación del entorno multiplataforma que permite la aplicación de algoritmos de agrupamiento espectral. Hace uso de todas las librerías mencionadas anteriormente para utilizar los algoritmos sobre datos arbitrarios e imágenes obtenidas con cámaras de tiempo de vuelo. Permite visualizar a su vez todas las etapas intermedias de los algoritmos con el objetivo de facilitar el análisis detallado de los mismos.

La aplicación esta dividida en dos partes

- Agrupamiento espectral sobre conjuntos arbitrarios de datos
- Agrupamiento espectral sobre imágenes TOF

Agrupamiento espectral sobre conjuntos arbitrarios de datos

Esta ventana permite aplicar el algoritmo de clustering espectral propuesto conjuntos arbitrarios de datos. Permite importar conjuntos de datos desde archivos de datos de Matlab (.mat), imágenes PNG (.png) , imágenes capturadas con TOFCapture (.tof) e imágenes Blensor (.pcd). A su vez brinda la posibilidad de exportar e importar los grafos generados en archivos con extensión graph.

Una vez seleccionado el conjunto de datos se pueden visualizar los vértices del grafo. Luego se debe especificar el algoritmo que se utilizará para calcular las aristas y las medidas de semejanza que se utilizarán, para esto el programa muestra un asistente que permite seleccionar todos éstos parámetros. Luego se procede al cálculo de autovectores y al agrupamiento.

La barra de herramientas ubicado en la parte izquierda de la pantalla va mostrado gradualmente las funcionalidades a medida que se avanza en el proceso de creación del grafo. La figura 5.14 muestra todas las opciones que presenta: apertura del conjunto de datos, cálculo de la aristas, cálculo de autovectores y segmentación.

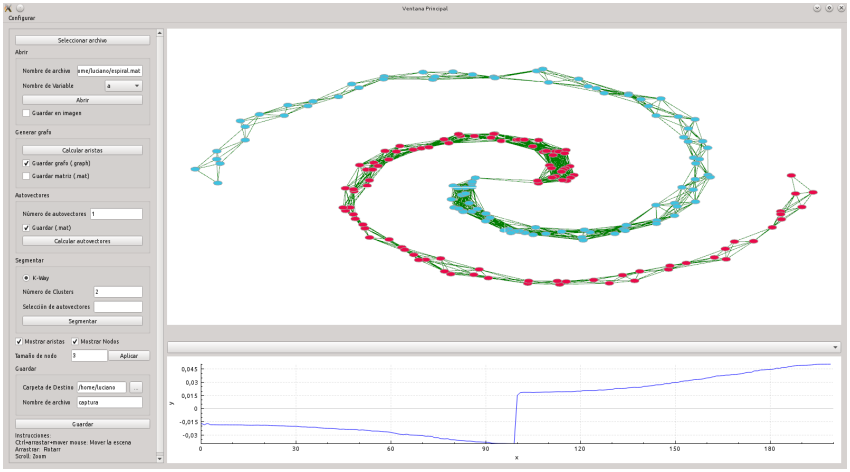


Figura 5.14: Captura de pantalla de SpectralGUI aplicado a un conjunto de datos de dimensión 2. El gráfico ubicado en la parte inferior representa a los valores del autovector correspondiente al segundo autovalor mas pequeño del grafo generado.

Agrupamiento espectral sobre imágenes TOF

Esta ventana permite aplicar el algoritmo de clustering espectral propuesto sobre las imágenes de rango e intensidad. Permite seleccionar el archivo capturado por TOFCapture (.tof) o

un archivo simulado con Blensor (.pcd). La barra lateral, como muestra la figura 5.15 permite configurar los parámetros de la función de pesos de las aristas de los dos grafos.

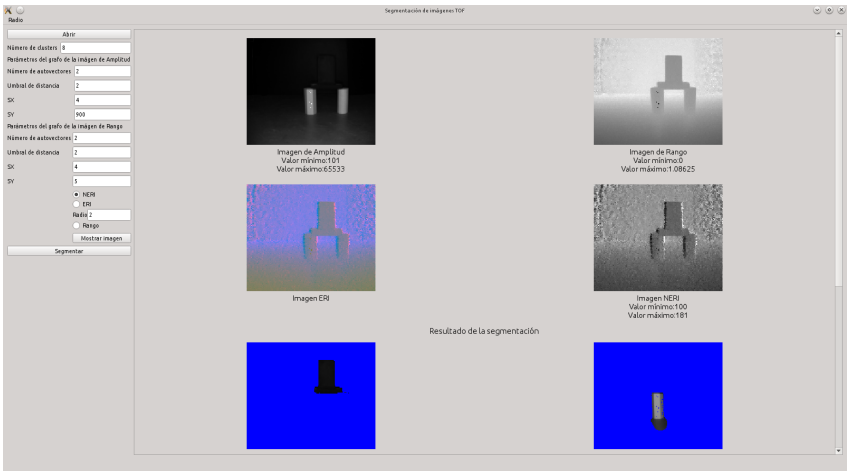


Figura 5.15: Captura de pantalla de SpectralGUI aplicado a una imagen de intensidad y de rango capturada con una cámara de tiempo de vuelo SwissRanger SR4000

Capítulo 6

Conclusiones y trabajo futuro

El trabajo realizado en la presente tesina se vincula con el proyecto "11-F011- Procesamiento paralelo y distribuido. Fundamentos y aplicaciones en Sistemas Inteligentes y Tratamiento de imágenes y vídeo" acreditado por la UNLP dentro del Programa de Incentivos llevado a cabo por el III-LIDI (Instituo de investigación en informática - LIDI).

Para cumplir con los objetivos propuestos fue necesario investigar las principales características de las cámaras de tiempo de vuelo, su modelo geométrico y los datos que brinda. Esto permitió el desarrollo de la aplicación TOFCapture que facilita la adquisición de imágenes a partir de la cámara de tiempo de vuelo MESA SwissRanger SR4000 y su posterior documentación. Esta aplicación sirvió como herramienta para la obtención de datos requeridos por los experimentos ya que es posible tener un registro detallado del las capturas que se realizan y permite poder recuperar con facilidad las imágenes para evaluarlas con diferentes algoritmos de segmentación. Además, su estructura modular y multiplataforma permite reutilizar las librerías en aplicaciones con diferentes requerimientos.

Se debió analizar también, distintos métodos de segmentación específicos para cámaras TOF. Como resultado del análisis, se implementaron diversos algoritmos viables para las imágenes que este tipo de cámaras brindan. El algoritmo de segmentación de objetos parcialmente ocluidos fue presentado en el XVIII Congreso Argentino de Ciencias de la Computación [76]. Los resultados preliminares obtenidos permiten observar el cierre correcto de los contornos de los objetos y la segmentación de objetos ocluidos de características simples.

En el contexto de la segmentación de imágenes obtenidas con cámaras de tiempo de vuelo,

se puso un énfasis particular en el análisis de los fundamentos teóricos de los algoritmos de agrupamiento espectral. Este análisis posibilito la propuesta de un método de agrupamiento que utiliza los fundamentos teóricos estudiados. El método de segmentación espectral fue presentando en el XIX Congreso Argentino de Ciencias de la Computación [35]. Los resultados obtenidos con éste método aplicado tanto en imágenes de intensidad y rango simuladas como reales presentan resultados preliminares satisfactorios. El algoritmo combina adecuadamente la información provista por ambas imágenes incluso en presencia de ruido. Permite segmentar objetos con niveles de intensidad próximos, ubicados a distintas distancias, u objetos cercanos con niveles de intensidad diferentes o con orientaciones diferentes.

A su vez el estudio de los fundamentos del agrupamiento espectral condujo al desarrollo del entorno SpectralGUI que facilita la aplicación de métodos espectrales de agrupamiento y permite realizar fácilmente variaciones en los algoritmos para evaluar los resultados. Su estructura modular y multiplataforma permite reutilizar las librerías en aplicaciones con diferentes requerimientos.

Como trabajo futuro se propone un análisis detallado de la influencia de los parámetros de la función de pesos en los autovectores de las matrices laplacianas. Otro aspecto para un trabajo de investigación futuro es modificar el método propuesto combinando la información de intensidad y profundidad en la función de pesos.

Capítulo 7

Bibliografía

[1] R. Schwarte, G. Häusler, and R. Malz, *Computer Vision and Applications: A Guide for Students and Practitioners*, ch. Three-Dimensional Imaging Techniques. Vol. 1 of *Electronics & Electrical* [19], 2000.

[2] G. Danciu, M. Ivanovici, and V. Buzuloiu, "Improved contours for ToF cameras based on vicinity logic operations," in *Proceedings of the International Conference on Optimization of Electrical and Electronic Equipments*, pp. 989–992, 2010.

[3] M. Cazorla, D. Viejo, and C. Pomares, "Study of the sr4000 camera," in *XI Workshop de Agentes Físicos, Valencia*, 2010.

[4] N. Blanc, T. Oggier, G. Gruener, J. Weingarten, A. Codourey, and P. Seitz, "Miniaturized smart cameras for 3d-imaging in real-time," in *Sensors, 2004. Proceedings of IEEE*, pp. 471–474 vol.1, 2004.

[5] S. Oprisescu, C. Burlacu, and A. Sultana, "A new contour extraction algorithm for tof images," in *Signals, Circuits and Systems (ISSCS), 2011 10th International Symposium on*, pp. 1–4, 2011.

[6] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 1997.

[7] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, pp. 888–905, Aug. 2000.

[8] A. Y. Ng, M. I. Jordan, and Y. Weiss, "On spectral clustering: Analysis and an algorithm," in *Advances in Neural Information Processing Systems*, pp. 849–856, MIT Press, 2001.

[9] J. A. Beraldin, F. Blais, L. Cournoyer, G. Godin, and M. Rioux, "Active 3D Sensing," pp. 22–46, Apr. 2003.

[10] B. Büttgen, T. Oggier, M. Lehmann, R. Kaufmann, and F. Lustenberger, "Ccd/cmos lock-in pixel for range imaging: challenges, limitations and state-of-the-art," in *In Proceedings of 1st Range Imaging Research Day*, pp. 21–32, 2005.

[11] R. A. Jarvis, "A perspective on range finding techniques for computer vision," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-5, no. 2, pp. 122–139, 1983.

[12] P. J. Besl, "Advances in machine vision," in *Machine Vision and Applications* (J. L. C. Sanz, ed.), vol. 1, ch. Active Optical Range Imaging Sensors, pp. 1–63, New York, NY, USA: Springer-Verlag New York, Inc., 1988.

[13] F. Chen, G. M. Brown, and M. Song, "Overview of three-dimensional shape measurement using optical methods," *Optical Engineering*, vol. 39, no. 1, pp. 10–22, 2000.

[14] F. Blais, "Review of 20 years of range sensor development," *Proc. SPIE*, vol. 5013, pp. 62–76, 2003.

[15] G. Häusler, "About the scaling behaviour of optical range sensors," in *Fringe*, vol. 97, pp. 147–155, 1997.

[16] J. E. Decker, J. R. Miles, A. A. Madej, R. F. Siemsen, K. J. Siemsen, S. de Bonth, K. Bustraan, S. Temple, and J. R. Pekelsky, "Increasing the range of unambiguity in step-height measurement with multiple-wavelength interferometry-application to absolute long gauge block measurement," *Appl. Opt.*, vol. 42, pp. 5670–5678, Oct. 2003.

[17] S. Nayar and Y. Nakagawa, "Shape from focus," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 16, no. 8, pp. 824–831, 1994.

[18] P. Favaro and S. Soatto, "A geometric approach to shape from defocus," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 27, no. 3, pp. 406–417, 2005.

[19] B. Jähne and H. Haussecker, *Computer Vision and Applications: A Guide for Students and Practitioners*. Electronics & Electrical, Academic Press, 2000.

[20] H. Takasaki, "Moiré topography," *Appl. Opt.*, vol. 9, pp. 1467–1472, Jun 1970.

[21] A. van Haasteren and H. Frankena, "Real-time displacement measurement using a multicamera phase-stepping speckle interferometer," *Appl. Opt.*, vol. 33, pp. 4137–4142, Jul 1994.

[22] B. K. P. Horn, "Shape from shading," ch. Obtaining Shape from Shading Information, pp. 123–171, Cambridge, MA, USA: MIT Press, 1989.

[23] J. Malik and R. Rosenholtz, "Computing local surface orientation and shape from texture for curved surfaces," *International Journal of Computer Vision*, vol. 23, no. 2, pp. 149–168.

[24] B. Saleh, *Fundamentals of photonics*. Hoboken, N.J: Wiley-Interscience, 2007.

[25] R. Lange, "3d time-of-flight distance measurement with custom solid-state image sensors in cmos/ccd-technology," 2000.

[26] C. Mutto, *Time-of-flight cameras and Microsoft Kinect*. New York: Springer, 2012.

[27] "Mesa imaging." www.mesa-imaging.ch, Dec. 2013.

[28] "Pmd technologies." http://www.pmdtec.com, Dec. 2013.

[29] "Softkinetic." http://www.softkinetic.com/, dic 2013.

[30] "Microsoft kinect." http://www.xbox.com/es-ES/Xbox360/Accessories/kinect/Home, Dec. 2013.

[31] S. Foix, G. Alenya, and C. Torras, "Lock-in time-of-flight (tof) cameras: A survey," *Sensors Journal, IEEE*, vol. 11, no. 9, pp. 1917–1926, 2011.

[32] M. Gschwandtner, R. Kwitt, A. Uhl, and W. Pree, "BlenSor: Blender Sensor Simulation Toolbox Advances in Visual Computing," vol. 6939 of *Lecture Notes in Computer Science*, ch. 20, pp. 199–208, Berlin, Heidelberg: Springer Berlin / Heidelberg, 2011.

[33] M. Gschwandtner, *Support Framework for Obstacle Detection on Autonomous Trains*. PhD thesis, Department of Computer Science, University of Salzburg, Austria, 2013.

[34] B. Foundation, "Blender." `http://blender.com/`, 2000–2004.

[35] L. Lorenti and J. Giacomantone, "Segmentación espectral de imágenes utilizando camaras de tiempo de vuelo," in *Proceedings of XVIII Congreso Argentino de Ciencias de la Computación*, pp. 1601–1608, (Red UNCI), 2013.

[36] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," in *Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '01, (New York, NY, USA), pp. 269–274, ACM, 2001.

[37] F. R. Bach and M. I. Jordan, "Learning spectral clustering," in *Advances in Neural Information Processing Systems 16*, Cambridge, MA: MIT Press, 2004.

[38] L. Zelnik-manor and P. Perona, "Self-tuning spectral clustering," in *Advances in Neural Information Processing Systems 17*, pp. 1601–1608, MIT Press, 2004.

[39] I. Dhillon, Y. Guan, and B. Kulis, "Kernel k-means: spectral clustering and normalized cuts," p. 556, 2004.

[40] C. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proc. SIAM Data Mining Conf*, pp. 606–610, 2005.

[41] L. Lorenti, L. Violini, and J. Giacomantone, "Selección sub-óptima del espectro asociado a la matriz de afinidad," in *XVIII Congreso Argentino de Ciencias de la Computación*, 2013.

[42] W. E. Donath and A. J. Hoffman, "Lower bounds for the partitioning of graphs," *IBM J. Res. Dev.*, vol. 17, pp. 420–425, Sept. 1973.

[43] M. Fiedler, "Algebraic connectivity of graphs," *Czechoslovak Mathematical Journal*, vol. 23, no. 98, pp. 298–305, 1973.

[44] D. A. Spielman and S.-H. Teng, "Spectral partitioning works: Planar graphs and finite element meshes," *Linear Algebra and its Applications*, vol. 421, pp. 284–305, Mar. 2007.

[45] S. X. Yu and J. Shi, "Multiclass spectral clustering," in *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ICCV '03, (Washington, DC, USA), pp. 313–, IEEE Computer Society, 2003.

[46] M. Belkin and P. Niyogi, "Laplacian eigenmaps and spectral techniques for embedding and clustering," in *Advances in Neural Information Processing Systems 14*, pp. 585–591, MIT Press, 2001.

[47] M. Weiss, *Estructuras de datos en Java TM : compatible con Java TM2.* Madrid, etc: Addison-Wesley, 2000.

[48] F. R. K. Chung, *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92).* American Mathematical Society, Dec. 1996.

[49] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A Global Geometric Framework for Nonlinear Dimensionality Reduction," *Science*, vol. 290, pp. 2319–2323, Dec. 2000.

[50] U. Luxburg, "A tutorial on spectral clustering," *Statistics and Computing*, vol. 17, pp. 395–416, Dec. 2007.

[51] W.-Y. Chen, Y. Song, H. Bai, C.-J. Lin, and E. Y. Chang, "Parallel spectral clustering in distributed systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 3, pp. 568–586, 2011.

[52] Z. Wu and R. Leahy, "An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1993.

[53] D. Wagner and F. Wagner, "Between min cut and graph bisection," in *Proceedings of the 18th International Symposium on Mathematical Foundations of Computer Science*, MFCS '93, (London, UK, UK), pp. 744–750, Springer-Verlag, 1993.

[54] B. Mohar, "The laplacian spectrum of graphs," in *Graph Theory, Combinatorics, and Applications*, pp. 871–898, Wiley, 1991.

[55] B. Mohar, "Some applications of laplace eigenvalues of graphs," in *Grap Symmetry: Algebraic Methods and Applications, VOLUME 497 OF NATO ASI SERIES C*, pp. 227–275, Kluwer, 1997.

[56] H. Lütkepohl and H. Lutkepohl, *Handbook of Matrices*. Wiley, 1 ed., Feb. 1997.

[57] R. Bekkerman, M. Bilenko, and J. Langford, "Scaling up machine learning: Parallel and distributed approaches," in *Proceedings of the 17th ACM SIGKDD International Conference Tutorials*, KDD '11 Tutorials, (New York, NY, USA), pp. 4:1–4:1, ACM, 2011.

[58] G. H. Golub and C. F. Van Loan, *Matrix Computations (3rd Ed.)*. Baltimore, MD, USA: Johns Hopkins University Press, 1996.

[59] M. Newman, *Networks: An Introduction*. New York, NY, USA: Oxford University Press, Inc., 2010.

[60] G. Hegde and C. Ye, "Extraction of planar features from swissranger sr-3000 range images by a clustering method using normalized cuts," in *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pp. 4034–4039, 2009.

[61] A. Bleiweiss and M. Werman, "Fusing time-of-flight depth and color for real-time segmentation and tracking," in *Dynamic 3D Imaging*, pp. 58–69, Springer, 2009.

[62] "Qt." http://qt-project.org/, Ene 2014.

[63] "Flexible image transport system (fits)." http://fits.gsfc.nasa.gov/, Ene 2014.

[64] J. Canny, "A computational approach to edge detection," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. PAMI-8, no. 6, pp. 679–698, 1986.

[65] R. Benlamri, "Range image segmentation of scenes with occluded curved objects.," *Pattern Recognition Letters*, vol. 21, no. 12, pp. 1051–1060, 2000.

[66] K. Klasing, D. Althoff, D. Wollherr, and M. Buss, "Comparison of surface normal estimation methods for range sensing applications," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 3206–3211, 2009.

[67] G. Hegde, C. Ye, and G. Anderson, "An extended normalized cuts method for real-time planar feature extraction from noisy range images," in *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pp. 1190–1195, 2010.

[68] G. Hegde and C. Ye, "A recursive planar feature extraction method for 3d range data segmentation," in *Systems, Man, and Cybernetics (SMC), 2011 IEEE International Conference on*, pp. 3119–3124, 2011.

[69] C. Ye and G. Hegde, "Robust edge extraction for swissranger sr-3000 range images," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 2437–2442, 2009.

[70] K. Pulli and M. Pietikäinen, "Range image segmentation based on decomposition of surface normals," in *Scandinavian conference on image analysis*, vol. 2, pp. 893–893, may 1993.

[71] "Boost serialization." http://www.boost.org/doc/libs/1_55_0/libs/serialization/doc/index.html, jan 2014.

[72] "Vxl (the vision-x-libraries)." http://vxl.sourceforge.net, jan 2014.

[73] B. Parlett, D. Scott, and C. U. B. E. R. LAB., *The Lanczos Algorithm with Selective Orthogonalization*. Defense Technical Information Center, 1978.

[74] "muparserx." https://code.google.com/p/muparserx/, Ene 2014.

[75] "Mat file i/o library." http://sourceforge.net/projects/matio/, Ene 2014.

[76] L. Lorenti, J. Giacomantone, and C. Manresa-Yee, "Segmentación de objetos parcialmente ocluidos utilizando cámaras tof," in *XVIII Congreso Argentino de Ciencias de la Computación*, 2012.