Planificador de Vistas para Reconstrucción Tridimensional de Objetos

por

Juan Irving Vásquez Gómez

Tesis sometida como requisito parcial para obtener el grado de

Maestro en Ciencias en el Área de Ciencias Computacionales en el

Instituto Nacional de Astrofísica, Óptica y Electrónica

Supervisada por:

Dr. Luis Enrique Sucar Succar, INAOE
Dr. Efraín López Damián, CIIDIT

Resumen

Un robot necesita un modelo 3D de cualquier objeto para poder manipularlo. Debido al limitado campo de visión de un sensor y a las oclusiones presentadas por los objetos, es necesario un conjunto de vistas para completar el modelo. Un interesante problema es cómo seleccionar esas vistas de forma óptima de acuerdo a ciertos criterios.

En esta tesis proponemos un novedoso algoritmo para seleccionar la siguiente mejor vista en la reconstrucción tridimensional de objetos desconocidos. El algoritmo utiliza una representación volumétrica para representar el estado de la reconstrucción. La siguiente mejor vista es seleccionada de un conjunto de vistas candidatas alrededor del objeto, mediante el uso de una función de utilidad y una de dos estrategias de búsqueda. La función de utilidad considera los porcentajes de voxels visibles en cada vista, la calidad y la distancia de navegación. Las estrategias de búsqueda se basan en dos aspectos. Una estrategia está basada en una descomposición jerárquica del espacio de búsqueda y otra basada en un proceso proceso de trazado de rayos multiresolución.

El algoritmo fue probado en simulación con siete objetos de diferentes complejidades, mostrando buenos resultados en términos de calidad de los modelos, tiempo de cómputo y distancia de navegación. El algoritmo también fue probado con mediciones de un sensor físico (cámara estereoscópica), como resultado se generó exitosamente el modelo de un objeto real.

Abstract

For manipulating an unknown object, a robot needs a 3D model of it. Given the limited field of view of a camera and self occlusions, a set of views is required to build a complete 3D model. So, an important problem is how to select these views optimally according to certain criteria.

We propose a novel algorithm (view planner) to select the next-best-view (NBV) for a range camera to model 3D arbitrary objects. We use a volumetric representation. We propose a new utility function considering amount of unknown information, quality and navigation. We also propose two novel strategies to accelerate the search of the NBV. One strategy is based on a hierarchical decomposition of the search space and the other is based on a multi-resolution of ray tracing.

We have tested our planner in simulation with 7 different 3D objects, showing good results in terms of quality of the models and computation time required, and at the same time reducing the distance that the sensor has to travel to obtain the set of views. We also have tested the planner in a robot with a stereo camera, as a result we reconstructed a real object.

Agradecimientos

Primero agradezco a Dios por brindarme las fuerzas para llegar hasta este punto en mi vida y por todas las satisfacciones que ha habido en ella.

Expreso mi gratitud hacia el Consejo Nacional Ciencia y Tecnología (CONACyT) por apoyar el desarrollo científico del país mediante los programas de posgrado.

Quiero expresar mi gratitud a mis asesores, Dr. Luis Enrique Sucar Succar y Dr. Efraín López Damián, quienes han guiado el desarrollo de esta tesis y han participado con sus ideas en la misma.

Después quiero agradecer a todos mis compañeros y amigos del inaoe quienes me han brindado su apoyo durante toda la estancia.

Por otra parte, quiero agradecer a mis amigos de candela con quienes he descubierto mi otra pasión, "la salsa", en especial a kika quien me apoyó siempre y me compartió su alegría.

Dedicatorias

A mis padres, Roberto Enrique Vásquez Martínez y Batilde Valeria Gómez Carrera, quienes me han inculcado las ganas de realizar mis sueños y me han apoyado incondicionalmente durante toda mi vida.

Índice general

Índice de figuras

Índice de tablas

Nomenclatura

$()'$	Vector transpuesto
γ	Ángulo de giro del sensor (*swing*)
ϕ	Ángulo de rotación vertical del sensor (*tilt*)
θ	Ángulo de rotación horizontal del sensor(*pan*)
b	Parámetro "base" de la estrategia multiresolución
k	Número de etapas de la estrategia multiresolución
l	Nivel de discretización de la esfera de vistas
len	Longitud de una arista de un voxel
M	Mapa de voxels
R^{eje}_{angulo}	Rotación sobre un eje de coordenadas
V	Conjunto de vistas candidatas o esfera de vistas
v	Estado del sensor o vista
RTO	Reconstrucción tridimensional de objetos
RTR	Resolución de trazado de rayos
SMV	Siguiente mejor vista

Capítulo 1

Introducción

La planificación de vistas es un recurso imprescindible en la reconstrucción tridimensional de objetos (RTO). En este capítulo se abordan las motivaciones de esta tesis, la problemática actual de la planificación de vistas para reconstrucción tridimensional de objetos por robots móviles, los objetivos y aplicaciones de esta tesis, y un resumen de nuestra propuesta de solución.

1.1. Motivación

En la actualidad los sensores de rango, capaces de observar superficies con precisión de milímetros, en conjunto con algoritmos de fusión de mallas tridimensionales han permitido la generación de modelos tridimensionales de gran precisión a partir de objetos físicos. Dichos modelos tienen un gran número de aplicaciones, por ejemplo, videojuegos, medicina, análisis de estructuras, recorridos virtuales, etc. Únicamente en robótica podemos encontrar aplicaciones en manipulación, reconocimiento de objetos, seguimiento de objetos (*object tracking*), etc.

Un ejemplo de reconstrucción tridimensional de objetos es el proyecto Miguel Ángel Digital [Levoy et al, 2000] cuyo objetivo fue escanear las esculturas esculpidas por Miguel Ángel, con la intención de analizar la forma en que el autor las esculpió, conservar en formato digital las obras de arte y analizar los cambios que

han sufrido con el tiempo para poder conservarlas.

Sin embargo, actualmente gran parte del proceso de reconstrucción tridimensional de objetos es realizada manualmente. Automatizar por completo dicho proceso tendría grandes beneficios, por ejemplo, aumentar la productividad, reducir costos, reducir la necesidad de personal altamente capacitado, etc. En robótica, la automatización dotaría con mayor autonomía para interactuar con el ambiente.

1.1.1. La reconstrucción de objetos

En la reconstrucción tridimensional de objetos intervienen cuatro elementos: el objeto, el sensor de rango, el sistema de posicionamiento y el espacio de vistas. El objeto es la entidad física a modelar. El sensor de rango es el dispositivo que permite adquirir la información geométrica en tres dimensiones del objeto. El sistema de posicionamiento permite colocar el sensor o el objeto en diferentes posiciones para poder observarlo. El espacio de vistas está formado por las combinaciones posibles de las configuraciones del sensor con las configuraciones del sistema de posicionamiento.

El objetivo de la RTO es adquirir un modelo geométrico tridimensional (3D) de la superficie del objeto haciendo uso del sensor de rango.

1.1.2. La planificación de vistas

Para reconstruir la superficie de un objeto es necesario un conjunto de imágenes de rango desde diferentes posiciones, también llamadas vistas. Ésto debido a que los sensores de rango sólo pueden ver una parte del objeto y también debido a las oclusiones que presentan los objetos mismos.

Determinar el conjunto de vistas necesario para reconstruir un objeto, cumpliendo las restricciones del sensor y del sistema de posicionamiento, es lo que entendemos como el problema de planificación de vistas (PPV). Dicho problema tiene dos clases que surgen a partir de la información *a priori* que se conoce del

objeto:

- **Planificación basada en un modelo (PBM)**. Esta categoría involucra un conocimiento previo del objeto y se subdivide en dos problemas:

  Si se conoce completamente la superficie del objeto, el problema consiste en determinar el conjunto mínimo de vistas, de tal forma que se observe la superficie completa del objeto. Este problema es equivalente al problema de la galería de arte, el cual se define como: *dado un conjunto de paredes encontrar la mínima cantidad de guardias que puedan vigilar todas las paredes de la galería.*

  Si se conoce parcialmente la forma del objeto, es decir, se tiene un modelo con un cierto grado de fidelidad, el problema consiste en: *determinar el conjunto de vistas necesarias que mejoren el modelo y provean la información faltante.*

- **Planificación no basada en un modelo (PNBM)**. En esta categoría se desconoce completamente la forma del objeto. En este caso no se puede hacer una planificación del conjunto de vistas como en el caso anterior, y utilizar un conjunto de vistas predeterminado puede ser insuficiente debido a la variación de tamaño y forma de los objetos. Por lo tanto, el problema consiste en: *determinar de forma iterativa las vistas que observen la superficie del objeto.*

1.2. Problema

En robótica, cuando un agente (robot) se encuentra con un objeto desconocido es posible que requiera agregar a su base de conocimiento la forma tridimensional del objeto. Dicha forma, representada muchas veces como una malla triangular, le serviría para diversas tareas. Por ejemplo: planificar los movimientos para poder tomarlo, reconocerlo cuando lo encuentre de nuevo, o incluso sólo para aumentar

su conocimiento del ambiente. En la figura 1.1 se muestra a un robot de servicio en una situación en la que se pregunta cómo puede tomar una taza que se encuentra sobre una mesa.

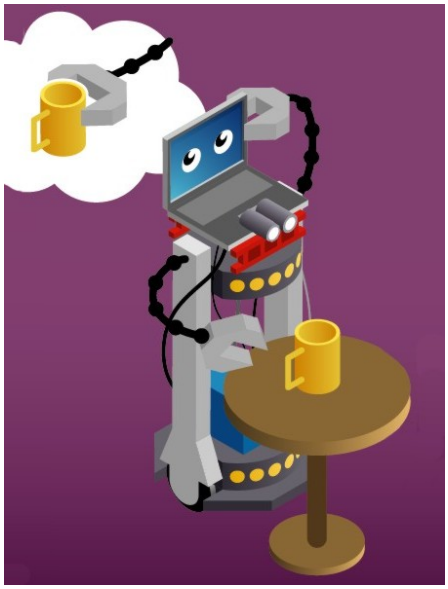


Figura 1.1: Ilustración de un robot de servicio tratando de manipular un objeto. El robot se encuentra en una situación en la cual necesita un modelo geométrico de la taza para poder planificar los movimientos de sujeción.

Dado que el problema es una planificación no basada en un modelo, la reconstrucción y la planificación se pueden ver como un único proceso formado por cuatro pasos que se repiten hasta alcanzar el criterio de paro: posicionamiento del sensor, captura de la imagen de rango, actualización de la representación, determinación de la siguiente posición y configuración del sensor.

Durante la reconstrucción, el agente debe viajar a cada posición, desde la cual toma una imagen y actualiza el modelo del objeto, gastando tiempo y energía, tanto en el viaje como en la captura de la imagen y actualización del modelo. En la figura 1.2 se muestra a un robot de servicio tomado imágenes de rango desde diferentes puntos para lograr reconstruir un objeto.

Si se reduce la distancia viajada por el robot idealmente se reduce el tiempo y la energía gastada, el ahorro de esta última es uno de los principales retos en

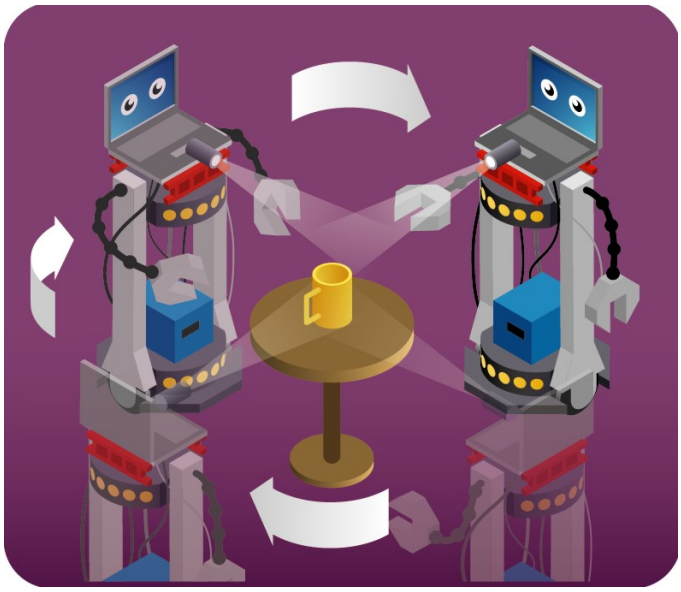


Figura 1.2: Ilustración de la reconstrucción de un objeto. El robot necesita moverse a diferentes posiciones para reconstruir un objeto.

robótica móvil [Mei et al, 2005], ya que la mayoría de los robots son alimentados por baterías. Por otra parte, si se reduce la cantidad de imágenes tomadas a su vez se reduciría: el tiempo de observación, el tiempo de registro y el traslape excesivo entre imágenes. Por lo tanto, es deseable que la cantidad de vistas y la distancia recorrida sea pequeña, conservando la calidad de las tomas y manteniendo un traslape que facilite el registro.

Actualmente la mayor parte de las técnicas de planificación están diseñadas para generar modelos de alta calidad sin importar el consumo de energía o la navegación, ya que son utilizadas en ambientes controlados y con sistemas de posicionamiento de alta precisión. Por otra parte, los trabajos de planificación que reducen la distancia de navegación sólo lo hacen a partir de un modelo o mapa previo del objeto. En resumen, hasta el momento no existe una técnica de PNBM para reconstrucción de objetos que minimice tanto la cantidad de vistas como la distancia de navegación y maximice la calidad de las observaciones.

1.3. Objetivo de la tesis

En esta tesis, el objetivo general es desarrollar un planificador de vistas para reconstrucción tridimensional de objetos[1]. El planificador debe reconstruir la mayor cantidad posible de la superficie del objeto y además debe buscar reducir la distancia de navegación y la cantidad de vistas para completar el modelo manteniendo la calidad en las tomas.

Para lograr el objetivo general se contemplan los siguientes objetivos particulares:

- Desarrollar una función de utilidad que permita evaluar la deseabilidad de las vistas, es decir, medir qué tan provechosas son para el proceso de reconstrucción.
- Desarrollar una estrategia de búsqueda que permita hallar la siguiente vista en tiempos aceptables para el proceso de reconstrucción.
- Implementar el algoritmo propuesto y hacer pruebas en simulación y en un robot móvil.

1.4. Solución propuesta

En esta tesis se planificador de vistas capaz de reconstruir objetos de diferente complejidad geométrica. El algoritmo está formado por tres partes esenciales:

- Una esfera de vistas. Al igual que en trabajos anteriores, nuestro planificador utiliza un conjunto discreto de vistas candidatas.
- Una función de utilidad. La función permite evaluar qué tan buena es una

[1]Un planificador de vistas contempla la planificación de las vistas y generación del modelo. Por otra parte, un algoritmo de planificación de vistas es parte de un planificador y únicamente de determina las vistas, en la literatura el algoritmo de planificación es también conocido como *next-best-view algorithm*.

vista candidata, a partir de ciertas condiciones que consideramos debe cumplir una buena vista.

- Una estrategia de búsqueda. La estrategia de búsqueda explora las vistas de la esfera, evalúa dichas vistas con la función de utilidad y determina la siguiente vista en un tiempo adecuado para la aplicación.

El planificador fue probado en simulación y con datos de un sensor real, con lo que se logró reconstruir exitosamente objetos de diferentes complejidades, reducir la distancia de navegación y determinar la siguiente vista en tiempos aceptables. Logros que nos permiten afirmar que se obtuvo un planificador más completo que trabajos anteriores.

1.5. Contribuciones

Las contribuciones de esta tesis son dos:

- Una función de utilidad que evalúa la deseabilidad de una vista a partir del área desconocida, el traslape, la calidad de la toma y la distancia de navegación.

- Dos estrategias de búsqueda que permiten determinar la siguiente vista en tiempos aceptables.

1.6. Organización de la tesis

En el capítulo 2 se abordan los conceptos y algoritmos utilizados en esta tesis. En el capítulo 3 se analizan los trabajos existentes en planificación de vistas para reconstrucción tridimensional de objetos (PVRTO), exponiendo sus capacidades y limitaciones. En el capítulo 4 se explica la constitución de nuestro planificador, con especial detalle en nuestras aportaciones. En los capítulos 5 y 6 se muestran los

resultados de nuestro planificador; en el capítulo 5 los resultados correspondientes a simulación y en el capítulo 6 los correspondientes a datos reales. En el capítulo 7 se concluye la tesis con un análisis de los resultados obtenidos y las contribuciones hechas; también se mencionan las posibles mejoras al planificador y las posibles líneas de investigación siguientes a esta tesis.

Capítulo 2

Marco teórico

En este capítulo se explican los conceptos y algoritmos que se utilizan en la planificación de vistas con métodos volumétricos. Los conceptos presentados sirven de base para comprender los aportes de esta tesis.

2.1. Sensor de rango

El sensor de rango es el dispositivo que permite adquirir la información geométrica en tres dimensiones del objeto. De acuerdo a Chen y Scott [Chen et al, 2008, Scott et al, 2003] un sensor pertenece a una de dos clases: sensores de rango pasivos o sensores de rango activos.

Los sensores de rango pasivos son aquellos que no emiten energía y a través de una técnica de visión computacional pueden obtener la forma tridimensional de una superficie, por ejemplo, cámaras monoculares y cámaras binoculares (estereoscópicas). Las técnicas computacionales utilizadas con éste tipo de sensores están basadas en la forma que los humanos percibimos el ambiente. Algunos ejemplos de dichas técnicas son *shape from shading*, *shape from texture*, *focus*, etc. Para más información consulte [Jain et al, 1995, Zhang et al, 1999].

Los sensores de rango activos, por su parte, utilizan un dispositivo de proyección que les permite cuantificar distancias. Podemos clasificar a este tipo de

sensores en triangulación y tiempo de vuelo. Los sensores de triangulación están basados en el principio de reconocer y triangular la posición de patrón proyectado en el objeto, por ejemplo "luz estructurada". Este tipo de sensores son altamente precisos y su aplicación en reconstrucción de objetos ha sido muy amplia. Los sensores por tiempo de vuelo se basan en medir el tiempo que tarda en regresar un rayo emitido para posteriormente determinar la distancia del objeto detectado, este tipo de sensores son más utilizados en mediciones donde las distancias son grandes, como es el caso de modelado de objetos de gran escala [Blaer et Allen, 2006a].

2.1.1. Representación de un sensor de rango

De acuerdo a Chen [Chen et al, 2008] un sensor se representa con un vector de seis parámetros espaciales $(x, y, z, \phi, \theta, \gamma)$. x, y y z son los grados de libertad de posición. ϕ,θ,γ son los grados de libertad de orientación, conocidos también como ángulos de *pan*, *tilt* y *swing*. Ver figura 2.1. La configuración de dichos parámetros es también llamada "vista".

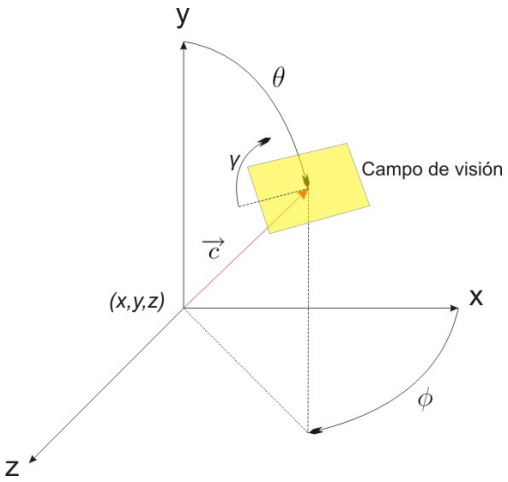


Figura 2.1: La ilustración muestra los parámetros de posición y orientación de un sensor de rango.

Para un sensor binocular, con la asunción de que ambas cámaras son paralelas y que la separación entre cámaras (*baseline*) es fija, el estado puede ser modelado

como un vector de nueve dimensiones, $v = (x, y, z, \phi, \theta, \gamma, d, f, a)$, donde d,f,a son los parámetros ópticos. d es la distancia al plano de la imagen f es la distancia focal. a es el diámetro de apertura de la pupila de la lente.

Para un sensor de rango 3D que opera con el principio de triangulación, la configuración de sensor puede ser definida como un vector de siete parámetros $(x, y, z, \phi, \theta, \gamma, \psi)$, donde ψ el ángulo del barrido.

2.1.2. Oclusiones

Las oclusiones, o áreas de oclusión, son generados cuando una superficie no es vista debido a las limitaciones de la técnica utilizada o por la posición y orientación del sensor. Por ejemplo, un sensor de rango activo genera oclusiones en las áreas en las cuales el rayo emitido no pudo hacer contacto con la superficie. En la figura 2.2 se muestra las oclusiones generadas por un sensor de rango. Los planos de oclusión se forman en la unión de la superficie de oclusión con el espacio vacío.

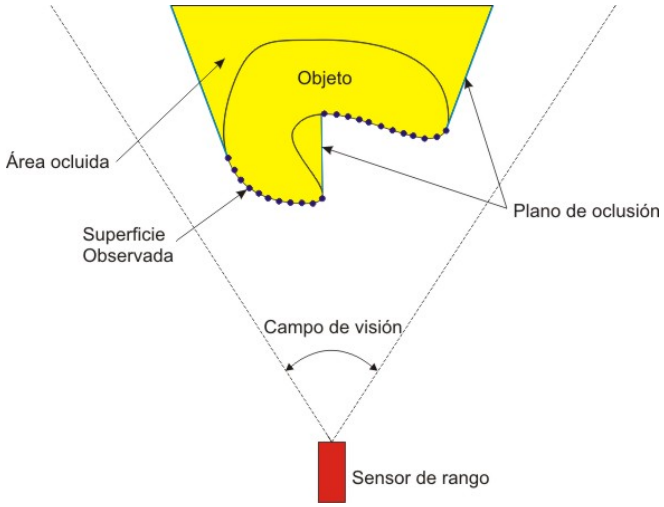


Figura 2.2: .
Las oclusiones o áreas de oclusión están determinadas por las áreas que que no son observadas por un sensor de rango. A la intersección del espacio vacío con las áreas de oclusión se le llama plano de oclusión.

2.1.3. Restricciones del sensor de rango

Para adquirir la información acerca de la superficie del objeto es necesario cumplir con una serie de restricciones impuestas por el sensor. Estas restricciones deben ser tomadas en cuenta en el momento de la planificación, para poder reconstruir exitosamente un objeto.

Visibilidad

La restricción de visibilidad concierne a qué superficies pueden se observadas desde una posición del sensor (figura 2.3(a)). Una sección de la superficie es visible si el producto punto de la normal de ésta y el rayo director del sensor es menor que 0 [Chen et al, 2008], expresado en la ecuación (2.1).

$$\vec{n} \cdot \vec{c} = \|\vec{n}\|\|\vec{c}\| \cos(180 - \theta) < 0, (0 <= \theta <= 180) \tag{2.1}$$

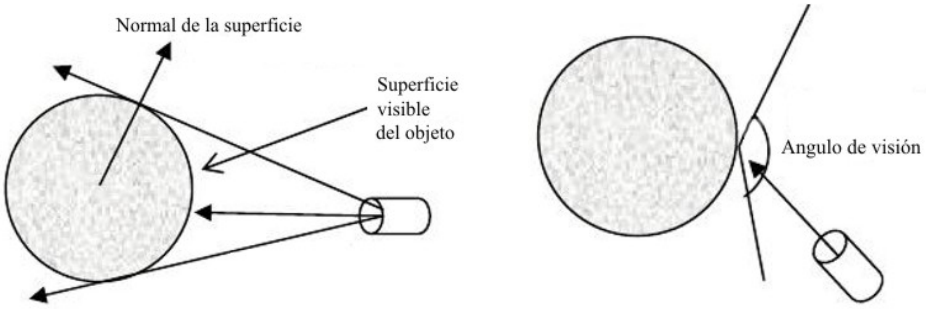
donde θ, en este caso, indica el ángulo formado entre el rayo director y la normal de la superficie.

Ángulo de visión

Mientras la visibilidad indica qué superficies son vistas desde una posición del sensor, el ángulo de visión (figura 2.3(b)) indica en qué posiciones se puede colocar el sensor para observar determinada superficie [Chen et al, 2008].

El espacio de posiciones está dentro de un cono, en donde se cumple que el ángulo formado entre el rayo director y la normal de la superficie está limitado por un θ_{max}.

$$\theta = \pi - cos^{-1}\left(\frac{\vec{n} \cdot \vec{c}}{\|\vec{n}\| \times \|\vec{c}|}\right) < \theta_{max} \tag{2.2}$$



(a) Visibilidad. (b) Ángulo de visión.

Figura 2.3: Visibilidad y ángulo de visión. La visibilidad indica qué superficies son visibles desde una posición del sensor. El ángulo de visión indica donde se puede colocar el sensor para observar una superficie. Imagen tomada de [Chen et al, 2008].

Campo de visión

El campo de visión (CDV) de un sensor delimita un cono en el cual las superficies son detectables. Una cámara CCD tiene un CDV limitado por el área del sensor y la longitud focal de la lente. Esta restricción implica que la sección de la superficie a observar debe estar dentro del campo de vista.

Resolución

La resolución en una imagen se puede entender de dos formas: una relacionada con la cantidad de lineas y columnas en la imagen; otra relacionada con el tamaño de la característica más pequeña detectable. Durante el texto, cuando sólo utilicemos la palabra resolución nos referiremos a la cantidad de filas y columnas en una imagen.

Resolución 3D

Se puede definir la resolución 3D como la mínima característica 3D en la escena que es detectable por el sistema de visión, es decir, qué tan pequeño es un voxel. Esta resolución es provista por el sensor, si el valor es pequeño entonces el sensor es capaz de detectar variaciones más pequeñas en la superficie observada.

Por lo tanto si queremos asegurar que una característica del objeto es detectable, el sensor debe estar configurado para que al menos dos voxels representen la característica mínima detectable.

Profundidad de campo

La profundidad de campo de una lente es el espacio en el cual las superficies observadas se encuentran en foco. Esta restricción se cumple cuando la superficie a observar se encuentra dentro de la profundidad de campo del sensor.

2.2. Imágenes de rango

Una imagen de rango es una matriz $[d_{uv}]\,(u \in [0...N-1], v \in [0...M-1])$ de distancias sensadas en la dirección $R_{\phi}^{\overrightarrow{y}} R_{\theta}^{\overrightarrow{z}} \overrightarrow{n}_{uv}$ desde $(x_0, y_0, z_0)'$ (Ver figura 2.4). Donde ϕ y θ indican la orientación del sensor y $\overrightarrow{n}_{uv}$ son vectores normalizados calculados para la geometría de un sensor específico (Ver [Massios et Fisher, 1998] para más detalles).

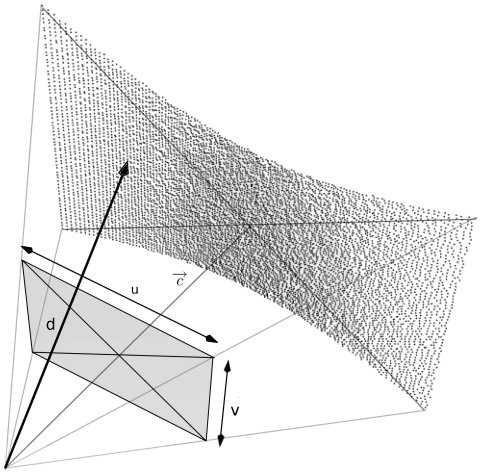


Figura 2.4: En la ilustración se muestran los puntos capturados en una imagen de rango.

Las coordenadas 3D en el mundo de un punto observado, $\overrightarrow{p}_{uv}$, son calculadas

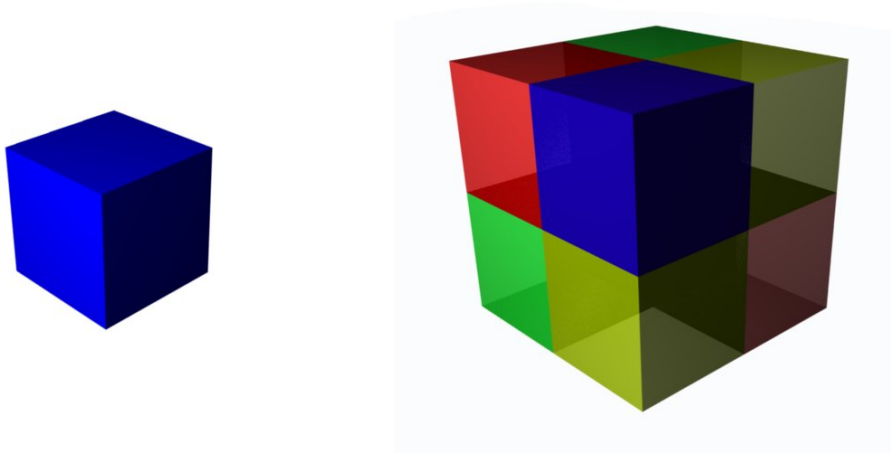
con la ecuación (2.3). La función 2.4 representa la misma operación.

$$\overrightarrow{p}_{uv} = \overrightarrow{c} + d_{uv} R_{\phi}^{\overrightarrow{y}} R_{\theta}^{\overrightarrow{z}} \overrightarrow{n}_{uv} \tag{2.3}$$

$$ConvertirACoordenadas3D(p_{uv}) = \overrightarrow{c} + d_{uv} R_{\phi}^{\overrightarrow{y}} R_{\theta}^{\overrightarrow{z}} \overrightarrow{n}_{uv} \tag{2.4}$$

2.3. Mapa de voxels

Un *voxel* es la unidad mínima de volumen que compone un objeto (Figura 2.5(a)), de la misma forma que un pixel representa una área en un espacio 2D, un *voxel* representa un volumen en espacio 3D. Un mapa de *voxels* (M) es una arreglo de tres dimensiones en el cual cada elemento es un *voxel* (Figura 2.5(b)).



(a) Voxel (b) Mapa de voxels

Figura 2.5: Voxel y mapa de voxels. La figura a) muestra un voxel. La figura b) muestra un mapa de voxels de resolución 2 x 2 x 2

Un *voxel* es un cubo de lado igual a *len*, este cubo es identificado por sus coordenadas $(i, j, k)'$ en el mapa de *voxels*. Cada voxel tiene asociados tres atributos: i) una etiqueta, ii) la normal de la superficie y iii) su calidad.

Las etiquetas identifican un voxel de acuerdo a su tipo y pueden tomar uno de 6 valores posibles:

- ***Sin marcar***. Es un voxel en una área que no ha sido observada por el sensor.
- ***Ocupado***. Es un voxel cuya posición coincide con los puntos de la superficie del objeto.
- ***Vacío***. Es un voxel en una área observada por el sensor en la cual no se han encontrado puntos de la superficie.
- ***Oculto***. Es un voxel que durante una observación ha sido ocluido por un voxel *ocupado*.
- ***Occplane***. Es un voxel oculto que es adyacente a un voxel vacío en cualquiera de sus caras.
- ***Borde***. Es un voxel ocupado que es adyacente a un voxel sin marcar en cualquiera de sus caras.

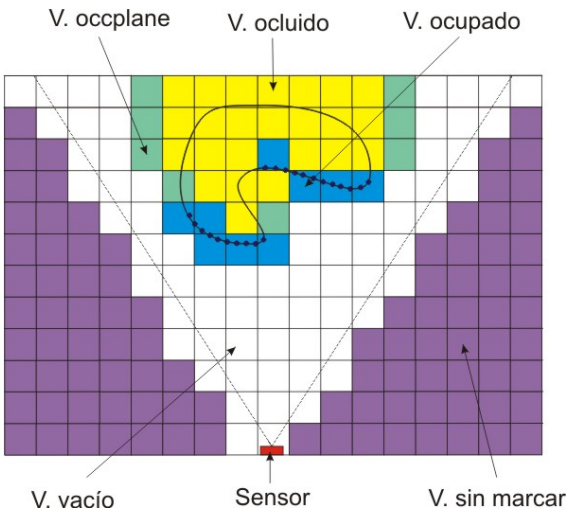


Figura 2.6: Etiquetas de voxels. Los voxels con diferentes etiquetas se muestran con un color diferente. En la figura sólo se muestran 5 tipos de voxels

2.3.1. Normal y calidad de un voxel

Los atributos normal y la calidad de un voxel, están definidas únicamente para los voxels de tipo *ocupado*. La normal de un voxel ocupado es un vector que representa la normal de la superficie en el punto que representa el voxel. La figura 2.7 ilustra un ejemplo de las normales de los voxels ocupados. La calidad de un voxel es un valor flotante, entre 0 y 1, relacionado con la forma en que fue observada una superficie. Cuando el rayo director del sensor se coloca perpendicular a la superficie entonces la superficie es tomada con la máxima calidad, conforme el ángulo entre la superficie y el rayo director deja de ser perpendicular la calidad disminuye. Éstos atributos se determinan durante una observación del sensor utilizando el algoritmo 1.

Algoritmo 1: Actualización de normal y calidad

Entrada: Conjunto de voxels ocupados(S); vector director del sensor($\overrightarrow{c}$); imagen de rango (Im)
Salida : Voxels ocupados con normal y calidad actualizadas (S)
Variables: Voxel ocupado (s); normal del voxel ocupado s ($\overrightarrow{n}_s$); calidad del voxel ocupado s (q_s); vector auxiliar ($\overrightarrow{n}$); valor flotante (q)

foreach $s \in S$ **do**
  $\overrightarrow{n} \leftarrow$ Determinar la normal para s utilizando *mínimos cuadrados* sobre una sección de 3×3 puntos de Im;
  if $\overrightarrow{n}_s$ *tiene un valor previo* **then**
    $\overrightarrow{n}_s \leftarrow$ Promedio de $\overrightarrow{n}_s$ con $\overrightarrow{n}$;
  else
    $\overrightarrow{n}_s \leftarrow \overrightarrow{n}$;
  end
  $q \leftarrow$ Determinar el producto punto entre $\overrightarrow{n}_s$ y $\overrightarrow{c}$;
  if $q > q_s$ **then**
    $q_s \leftarrow q$;
  end
end
Regresar S;

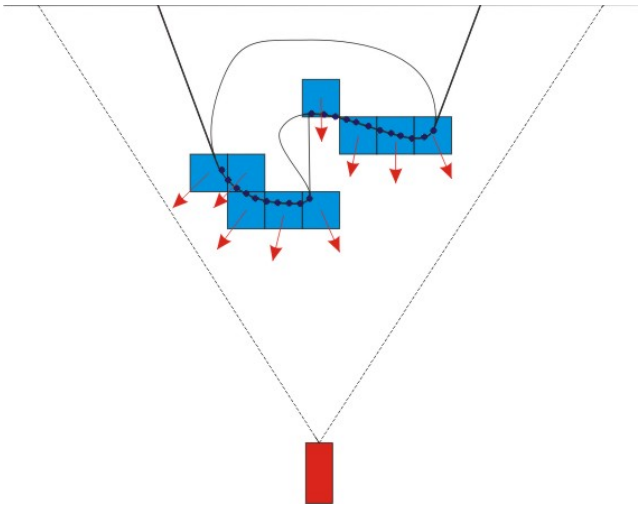


Figura 2.7: Mapa de voxels y normales de la superficie. Los cubos muestran los voxels *ocupados* y las flechas indican las normales de la superficie de sus respectivos voxels

2.3.2. Dimensiones y resolución del mapa de voxels

Durante el resto del texto llamaremos dimensión del mapa de voxels a las medidas que tiene el mapa con respecto al mundo real. La resolución del mapa de voxels se referirá a la cantidad de elementos que contiene el mapa de voxels en cada una de sus tres dimensiones. La resolución o tamaño de un voxel hará referencia a la longitud de uno de sus lados (len). Incrementar la resolución del mapa de voxels implica reducir el tamaño de un voxel mientras se conservan las dimensiones del mapa de voxels.

2.4. Trazado de rayos

Dado un voxel con índices $(i, j, k)'$ y un vector $\overrightarrow{d}$, trazar un rayo en el mapa de voxels significa recorrer los voxels a partir de $(i, j, k)'$ en la dirección de $\overrightarrow{d}$, los voxels recorridos están determinados por el algoritmo de Bresenham [Bresenham, 1965]. En la figura 2.8 se muestra el trazado de un rayo en el mapa de voxels.

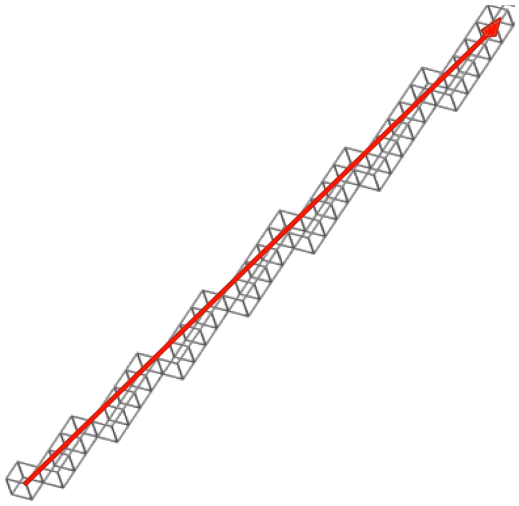


Figura 2.8: Trazado de rayos a través del mapa de voxels. En la figura se muestra como un rayo recorre el mapa de voxels de acuerdo al algoritmo de Bresenham

La función (2.5) indica que se lanza un rayo dentro del mapa de voxels desde la posición del voxel inicial, v_i, hasta la posición del voxel final, v_f. Cada voxel recorrido durante el trazado del rayo es marcado con la etiqueta e.

$$TrazarRayoMarcar(v_i, v_f, e) \tag{2.5}$$

2.5. Actualización del mapa de voxels

En cada iteración del proceso de reconstrucción, después de que se toma una imagen de rango y se transforman los puntos de la imagen al sistema de referencia del mapa de voxels, es necesaria una actualización de las etiquetas. Para ello son necesarios dos pasos, en el primero se transforman los puntos observados a posiciones de voxels y luego se actualizan las etiquetas mediante trazado de rayos.

A continuación se explican los dos pasos de la actualización del mapa de voxels, conversión de puntos y actualización de etiquetas.

2.5.1. Conversión de puntos al mapa de voxels

En este primer paso se convierte cada punto observado al índice correspondiente en el mapa de voxels, es decir, se asigna a cada punto un voxel, asumiendo que cada punto a convertir está dado en coordenadas del mundo real.

Dado el centro del mapa de voxels con índices $(0, 0, 0)'$ y con coordenadas en el mundo $(x_w, y_x, z_w)'$ y dado el tamaño de voxel como len, cada punto con coordenadas $(x, y, z)'$, cae en el voxel con índices $(ConvertirPunto(x-x_w), ConvertirPunto(z-z_w), convertirPunto(z-z_w))$, donde la expresión $ConvertirPunto(x)$ está determinada por la función (2.6).

$$ConvertirPunto(x) = Redondear\left(\frac{x + signo(x)\frac{l}{2}}{l}\right) \tag{2.6}$$

2.5.2. Actualización de las etiquetas

Una vez que se tiene el conjunto de puntos de la imagen de rango, o nube de puntos, con índices del mapa de voxels se actualizan las etiquetas. El objetivo de esta etapa es marcar qué espacios son ocupados por la superficie observada y marcar qué espacios están libres. El algoritmo utilizado primero marca cada punto observado como un voxel ocupado en el mapa, en seguida traza un rayo desde la posición del sensor a cada punto observado y marca todos los voxels recorridos como vacíos. El algoritmo 2 detalla el proceso.

2.6. Resumen

En este capítulo se mencionaron los conceptos base de la planificación de vistas por métodos volumétricos. Se estableció la representación de un sensor de rango y las restricciones que deben cumplir las vistas. También se definió el concepto de mapa de voxels y su forma de actualización. En el capítulo siguiente se revisan y se analizan los trabajos más sobresalientes en la planificación de vistas.

Algoritmo 2: Actualización de etiquetas

Entrada: Mapa de voxels(M); Posición del sensor en el espacio (c); imagen de rango (I)

Salida : Mapa de voxels con etiquetas actualizadas (M)

Variables: Posición del sensor en el mapa de voxels (c^{voxel}); Punto en la columna u y fila v de la imagen de rango (p_{uv}); Posición en el mundo real del punto p_{uv} (p_{uv}^{mundo}); posición en el mapa de voxels del punto p_{uv} (p_{uv}^{voxel}); punto de intersección del rayo que genera el punto p_{uv} con los límites del mapa de voxels (i_{uv}^{voxel})

1 $c^{voxel} \leftarrow$ `ConvertirPunto(`c`);` `/* Función 2.6 */`
2 **foreach** $p_{uv} \in I$ **do** `/* Para cada punto en la imagen */`
3 $p_{uv}^{mundo} \leftarrow$ `ConvertirACoordenadas3D(`p_{uv}`);` `/* Función 2.4 */`
4 $p_{uv}^{voxel} \leftarrow$ `ConvertirPunto(`p_{uv}^{mundo}`);`
5 $i_{uv}^{voxel} \leftarrow$ Calcular la intersección del rayo que genera el punto p_{uv} con los límites de M;
6 Hacer un trazado de rayos desde la posición del sensor, c^{voxel}, hasta el punto p_{uv}^{voxel} marcando como *vacíos* los voxels recorridos, si se encuentran en el camino voxels *ocupados* éstos no se modifican;
7 Hacer un trazado de rayos del punto p_{uv}^{voxel} hasta i_{uv}^{voxel}, marcando como *ocultos* los voxels *sin marcar* encontrados;
8 Marcar p_{uv}^{voxel} como *ocupado*;
9 **end**
10 **foreach** $o_i^{voxel} \in M$ **do** `/* Para cada voxel oculto */`
11 Marcar o_i^{voxel} como *occplane* si es adyacente a un voxel *vacío* en alguna de sus caras;
12 **end**
13 **foreach** $b_j^{voxel} \in M$ **do** `/* Para cada voxel borde */`
14 Marcar b_j^{voxel} como *ocupado*;
15 **end**
16 **foreach** $ocu_j^{voxel} \in M$ **do** `/* Para cada voxel ocupado */`
17 Marcar ocu_j^{voxel} como *borde* si es adyacente a un voxel *sin marcar* en alguna de sus caras.
18 **end**

Capítulo 3

Trabajo relacionado

3.1. Introducción

Como se vio en el capítulo 1, el problema de planificación de vistas consiste en determinar un conjunto de vistas que pueda reconstruir la superficie de un objeto, mientras se cumplen las restricciones del sensor. En este capítulo se revisan los trabajos más importantes que se han desarrollado para resolver dicho problema.

De acuerdo a [Scott et al, 2003], los trabajos que se han desarrollado en planificación de vistas para reconstrucción de objetos pueden dividirse en dos categorías: métodos basados en el modelo y métodos no basados en el modelo.

Las métodos[1] basadas en el modelo hacen uso de un modelo previo con cierto grado de fidelidad para planificar el conjunto de vistas que permita observar la superficie completa del objeto, por ejemplo [Wang et al, 2007]. El requerir un modelo *a priori* es una restricción que impide a este tipo de métodos ser aplicados en la solución de nuestro problema.

A diferencia de los métodos anteriores, los métodos no basadas en el modelo no cuentan con información *a priori* del objeto, o si cuentan con alguna información, ésta es mínima (posición y posibles dimensiones), esta característica las hace apli-

[1]La denominación métodos se utiliza para denominar un conjunto algoritmos de planificación que determinan la siguiente mejor vista de forma similar

cables al problema que buscamos resolver. Estos métodos se pueden separar en dos clases, diferenciadas por el dominio de razonamiento: i) los métodos basados en la superficie y ii) los métodos volumétricos.

En la siguiente sección se explican los requerimientos que consideramos debe cumplir un planificador de vistas. Más adelante se revisan los métodos más importantes en reconstrucción no basada en un modelo. Se clasifican los trabajos en dos categorías, los basados en la superficie y los volumétricos. Dentro de los métodos volumétricos se mencionan algunos trabajos que fueron desarrollados para reconstrucción de ambientes, éstos son muy similares a la reconstrucción de objetos. Nuestra revisión es más extensa en los métodos volumétricos que en los métodos basados en la superficie, dado que nuestro algoritmo es un método volumétrico.

3.2. Requerimientos

idealmente los algoritmos de planificación deben cumplir con ciertos requerimientos y especificaciones que les permitan lidiar con diferentes problemas suscitados durante una reconstrucción. A continuación se mencionan dichos requerimientos, clasificados en tres categorías: general; objeto; sensor y sistema de posicionamiento. Muchos de los requerimientos que a continuación se mencionan fueron propuestos en [Scott et al, 2003].

3.2.1. General

- **Especificación de la calidad del modelo**. El modelo generado debe tener asociada una medida cuantificable de qué tan preciso es con respecto del objeto real.

- **Algoritmo generalizable**. El planificador debe ser aplicable a una amplia gama de sensores de rango, sistemas de posicionamiento, y objetos.

- **Calidad de la toma**. El planificador debe procurar que el sensor esté posicionado de forma perpendicular a la superficie a observar, para asegurar una buena calidad en las tomas.

- **Vistas generalizadas**. En adición a la posición del sensor, el algoritmo planificador de vistas debe planificar los parámetros del sensor, a través del uso de vistas generalizadas que incorporen todos los parámetros.

- **Traslape**. El planificador debe proveer un traslape entre imágenes capturadas, de tal forma que se facilite la etapa de registro.

- **Robustez**. El algoritmo planificador debe ser capaz de superar errores críticos suscitados durante la reconstrucción, necesitando la mínima asistencia del usuario.

- **Finalización automática**. El algoritmo debe ser capaz de reconocer cuando el objetivo de reconstrucción se ha alcanzado o el incremento del conocimiento acerca del objeto se ha estancado, de tal forma que se finalice el proceso.

3.2.2. Objeto

- **Conocimiento *a priori* mínimo**. El planificador debe ser efectivo a pesar de sólo contar con conocimiento *a priori* mínimo del objeto, por ejemplo únicamente dimensiones aproximadas y centroide.

- **Restricciones de forma**. El planificador debe ser capaz de reconstruir un objeto independientemente de la forma geométrica del objeto.

- **Restricciones del material**. Si se cumple que, dada la constitución física del objeto (material que constituye la superficie), el sensor puede observar la superficie entonces el planificador debe de determinar las vistas para reconstruir el objeto.

3.2.3. Sensor y sistema de posicionamiento

- ***Frustum***. El *frustum*, o volumen de visualización de un dispositivo, debe ser modelado y contemplado en la reconstrucción. Éste incluye el campo de vista, la profundidad del campo, y longitud de observación.

- **Posicionamiento en 6D**. El planificador debe ser compatible con sistemas de posicionamiento sin restricciones en los tres grados de libertad de posición y los tres de orientación.

- **Ahorro de energía**. El planificador debe reducir la energía que gasta el sistema de posicionamiento en la reconstrucción del objeto, sobre todo en sistemas de posicionamiento alimentados por baterías.

3.3. Métodos basados en la superficie

Los métodos basados en la superficie, en su mayoría, son métodos por síntesis, es decir determinan la siguiente mejor vista a partir de analizar la superficie registrada. La mayoría de los algoritmos analizan la superficie observada y determinan la siguiente vista en los bordes de dicha superficie.

Esta tesis se basa en métodos volumétricos, por lo cual, a continuación sólo se revisan los métodos basados en la superficie más importantes.

Maver y Bajcsy [Maver et Bajcsy, 1993] propusieron un algoritmo de planificación basado en bordes de oclusión. El algoritmo fue planteado en dos etapas y consideraba de forma separada las oclusiones generadas por el sensor y las generadas por el objeto. El enfoque del método esta en detectar los bordes de oclusión en una imagen de rango. Para ello se calculan los bordes de la imagen (de forma similar a una imagen visual). Se calcula el histograma de dicha imagen y se determina la siguiente vista a partir del mayor valor del histograma. El método fue incapaz de observar la superficie completa del objeto. Sin embargo, fue el precursor de muchos trabajos, debido a que estableció los diferentes tipos de oclusiones.

Mas tarde, en [García et Velázquez, 1998] se propuso otro algoritmo basado en bordes de oclusión. Este trabajo fue planteado en dos etapas. En la primera etapa se propuso un algoritmo de votación que utiliza las normales de la superficie. En la segunda etapa se hace un análisis de visibilidad para completar las áreas no vistas durante la primera etapa.

De los trabajos revisados no se encontró alguno que proveyera características de traslape, calidad de las tomas y disminución de la distancia de navegación. A su vez, dichos algoritmos son complejos de implementar debido a las representaciones de superficie utilizadas, por ejemplo, mallas triangulares o PLY[2] tienen estructuras propias y requieren de un manejo específico; a diferencia de los métodos volumétricos que únicamente requieren una matriz tridimensional. Por otra parte, los tiempos de cálculo son muy similares a los algoritmos por métodos volumétricos.

3.4. Métodos volumétricos

Los métodos volumétricos planifican las vistas a través del análisis del mapa de voxels que representa el estado de la reconstrucción. El objetivo es determinar la vista que aumente el conocimiento de la superficie. En la mayoría de los casos, dicha vista es determinada a partir de un conjunto de vistas candidatas mediante la optimización de una función de utilidad que evalúa cada vista.

A continuación se describen los trabajos más sobresalientes, ordenados de manera cronológica.

3.4.1. Papadopoulos-Orfanos y Schmitt

En [Papadopoulos-Orfanos et Schmitt, 1997] se abordó el problema de digitalizar la superficie de un objeto utilizando un sensor láser 3D con un pequeño

[2]PLY es un formato de descripción de objetos utilizado por investigadores que utilizan modelos poligonales.

campo de visión. En ese trabajo se utilizó un mapa de voxels con 3 etiquetas diferentes: *sin marcar*, *ocupado* y *vacío*.

La reconstrucción del objeto utiliza dos etapas. En la primera etapa se hace una exploración exhaustiva del espacio mediante el seguimiento de una ruta en zig-zag sobre el plano xy para luego incrementar la coordenada z y hacer nuevamente el recorrido en zig-zag, durante esta etapa la orientación del sensor se fijó y sólo traslaciones fueron empleadas para observar el espacio. Como consecuencia, muchas superficies ocluidas no son observadas. En la segunda etapa se busca eliminar las oclusiones dejadas por la primera, haciendo uso de la orientación del sensor. Para la segunda etapa los autores sólo mencionaron algunas posibles soluciones que no fueron probadas.

El objetivo de ambas etapas fue colocar el sensor lo más cerca posible del objeto, debido al pequeño campo de visión del sensor (de 5 a 10 centímetros), mientras se evitaban colisiones.

El énfasis del trabajo estuvo en la actualización del mapa de voxels, en hacer un etiquetado robusto y en la abolición de colisiones. Acertadamente, los autores consideraron un sensor con seis grados de libertad. Sin embargo, no hicieron una planificación de vistas como tal, debido a que el trabajo se reduce a una exploración exhaustiva del espacio siguiendo una ruta predefinida, la cual se modifica durante la reconstrucción para evitar colisiones con el objeto. Además, el hacer una exploración exhaustiva requiere capturar muchas imágenes y hacer muchos movimientos con el sensor.

3.4.2. Massios y Fisher

En [Massios et Fisher, 1998] se desarrolló un método para determinar la siguiente vista, con la novedad de que los autores incluyeron un criterio de calidad, el cual tiene el objetivo de mejorar la calidad promedio de las superficies previamente escaneadas. A diferencia de trabajos previos los autores definieron y usaron

cuatro tipos de voxels: *vacío*, *visto* (*ocupado*), *no-visto* (*sin marcar*) y *occplane*. Además incorporaron a cada voxel *visto* el atributo de calidad.

Los autores utilizaron una esfera de vistas para restringir el conjunto de vistas candidatas, el método utilizado para generar este conjunto fue tomar un icosaedro como base y por medio de divisiones recursivas alcanzar el nivel de división deseado.

Con el fin de determinar la siguiente mejor vistalos autores desarrollaron una función de objetivo $f_{total}(v)$, representada como una suma pesada de los términos de visibilidad y calidad, dicha función se muestra en la ecuación (3.1).

$$f_{total}(v) = w_v f_{visibility}(v) + w_q f_{quality}(v) \tag{3.1}$$

donde v es una vista del conjunto de vistas candidatas.

El criterio de visibilidad prefiere vistas con mayor cantidad de voxels *occplane*, y el criterio de calidad prefiere vistas que observen superficies tomadas con baja calidad. La forma para optimizar la función fue una evaluación exhaustiva utilizando trazado de rayos.

La introducción del concepto de calidad promedio para el mapa de voxels y la inclusión del término de calidad en la función objetivo son contribuciones importantes. Sin embargo, para mejorar la calidad promedio es necesario tomar imágenes de zonas previamente observadas, lo que implica un aumento tanto en la cantidad de vistas como en la distancia recorrida. Por lo tanto, sería deseable hacer una predicción de la calidad de la superficie que se va a tomar y buscar otra posición que obtenga una mejor calidad.

Entre las deficiencias de este algoritmo están el costo computacional, la falta de un criterio de traslape entre vistas y la omisión de un criterio que minimice la energía consumida.

3.4.3. Wong, Dumont y Abidi

En [Wong et al, 1999] se propusieron tres algoritmos de planificación basados en búsqueda. Dichos algoritmos determinan un conjunto de vistas candidatas y seleccionan la siguiente mejor vista mediante la optimización de una función de utilidad. Las etiquetas utilizadas fueron *desconocido* (*oculto*) y *conocido* (*ocupado*). La siguiente mejor vista fue definida como la vista desde la cual el mayor número de voxels *desconocidos* es visible.

El primer método, llamado *The Optimal Method*, genera una esfera de vistas a través de una técnica similar a una división de paralelos y meridianos, para después hacer una búsqueda exhaustiva en la esfera, por lo cual este método es computacionalmete costoso. El segundo método, llamado *The Surface Normal Method*, determina un pequeño conjunto de vistas candidatas sumando las normales de la superficie de los voxels desconocidos en cada eje y determina la siguiente mejor vista como el vector resultante de dicha suma. El segundo método no es capaz de reconstruir objetos con auto-oclusiones, es por eso que el tercer método es una combinación de los dos primeros, utilizando por omisión *The surface Normal Method* hasta que existe auto-oclusión de una vista, en ese momento se cambia el algoritmo a *The Optimal Method*.

El planificador de vistas tiene la capacidad de reconstruir objetos de diferentes complejidades en relativamente pocas vistas y la capacidad de ser independiente del sensor que se utilice. Sin embargo, tiene las desventajas de no proveer un traslape entre vistas, no restringir las posiciones inalcanzables para el sensor y no considerar la distancia viajada por el sistema de posicionamiento.

3.4.4. Banta y otros

En [Banta et al, 2000] se propuso un algoritmo para planificación de vistas que une varios algoritmos desarrollados previamente los autores de este trabajo o por otros autores. El algoritmo, dependiendo de la iteración en la reconstrucción,

selecciona uno de tres métodos para determinar la siguiente vista.

Método basado en bordes. Previamente desarrollado en [Banta et al, 1995] y similar al propuesto por [Maver et Bajcsy, 1993]. Este método analiza la imagen de rango tomada en búsqueda de evidencia de oclusiones, las oclusiones son detectadas midiendo los niveles de intensidad en los bordes de una imagen de rango, por lo tanto, la siguiente mejor vista está determinada por el borde con mayor intensidad en dirección al centro de la esfera de vistas.

Método basado en centroide. En este método se calcula el centroide de los voxels *ocultos* y se calcula la siguiente vista apuntando hacia el centro del objeto, de tal forma que se pase por la cara del voxel más cercano al centroide.

Método de particionamiento. El método utiliza un algoritmo de particionamiento hasta lograr n particiones de los voxels *ocultos*, para después orientar el sensor a la partición más grande, este método fue propuesto en [Banta et Abidi, 1996].

El algoritmo general determina en la primera iteración la vista inicial de forma arbitraria. En la segunda iteración, la vista es determinada por el método de bordes. En las iteraciones tercera y cuarta se utiliza el método del centroide. Para las vistas siguientes utiliza el método de particionamiento. Cuando este último método no logra determinar la siguiente vista, el algoritmo general selecciona la vista que está mas lejana a las vistas previas.

La selección de un método en función de la iteración de la reconstrucción permite utilizar las mejores cualidades de cada método, lo que se considera un acierto. Sin embargo, en las últimas iteraciones el algoritmo puede no ser óptimo, debido a que el último recurso para determinar la siguiente vista se basa en determinar una vista alejada de las anteriores; una decisión que en el peor de los casos recorrería todas las vistas restantes para encontrar una vista que provea nueva información.

Entre las ventajas del método está el ser un algoritmo independiente del sensor y la forma de los objetos. Entre sus desventajas está el no considerar un traslape en las vistas, el no restringir las posiciones del sensor y el no considerar la distancia viajada por el sistema de posicionamiento.

3.4.5. Null y Sinzinger

En [Null et Sinzinger, 2006] se desarrollaron dos algoritmos, uno para reconstrucción de objetos denominado *Exterior NBV algorithm*; otro para reconstrucción de ambientes interiores denominado *Interior NBV algorithm*. El algoritmo para reconstrucción de objetos está desarrollado para un sensor montado en una plataforma móvil con tres grados de libertad, dos para su posición en el plano xy y uno más para su orientación (θ).

El método propuesto genera un conjunto de vistas candidatas y maximiza una función objetivo para determinar la siguiente mejor vista. Las vistas candidatas son generadas mediante la discretización del plano alrededor del objeto (plano sobre en el que se mueve el robot). La siguiente mejor vista es aquella desde la cual se visualiza la mayor cantidad de voxels.

En [Null et Sinzinger, 2006] los autores utilizaron una búsqueda exhaustiva de la siguiente mejor vista en todo el conjunto de vistas candidatas, contabilizando la cantidad de voxels *desconocidos* (definidos como *sin marcar*) mediante un trazado de rayos. Una de las contribuciones del trabajo fue restringir los rayos al espacio que corresponde al posible objeto, con lo que se disminuye el costo computacional; sin embargo, la ganancia en tiempo no fue significativa. El algoritmo no es generalizable, debido a que está restringido a plataformas con tres grados de libertad. El algoritmo no contempla un traslape entre vistas ni contempla de la distancia viajada por la plataforma con tres grados de libertad.

3.4.6. Blaer y Allen

En [Blaer et Allen, 2006a, Blaer et Allen, 2006b, Blaer et Allen, 2007] los autores abordan el problema de adquisición de datos y planificación de vistas para reconstrucción de objetos exteriores de gran escala. Los algoritmos propuestos son una combinación de planificación basada en un modelo y no-basada en un modelo. Dichos algoritmos proceden en dos etapas. En la primera etapa se utiliza un mapa

en 2D (proporcionado *a priori*), con el que se hace una planificación para explorar la mayor cantidad de la superficie del objetivo. En la segunda etapa se utiliza un método volumétrico para completar el modelo de la primera.

El objetivo de la primera etapa es obtener un modelo 3D inicial del objeto, el cual servirá de entrada para la segunda etapa. En la primera etapa el sistema utiliza un mapa 2D para planificar un conjunto mínimo de vistas para cubrir la superficie del objeto. Una vez determinado el conjunto de vistas inicial se planifica la ruta más corta para recorrerlas.

En la segunda etapa, el modelo 3D obtenido es representado como un mapa de voxels, a partir del cual se determina iterativamente cada siguiente vista con el objetivo de observar las superficies faltantes. En esta segunda etapa se utilizan cuatro tipos de etiquetas para los voxels: *desconocido*, *ocupado*, *vacío* y *frontera* (definidos como *occplane*). Las vistas candidatas son los voxels que coinciden con el piso vacío del mapa 2D inicial. La siguiente mejor vista es aquella desde la cual se observa la mayor cantidad de voxels *frontera*.

Una de las principales contribuciones del trabajo fue la unión de métodos basados en el modelo y métodos no basados en el modelo para resolver un mismo problema. Otra contribución fue la incorporación de un algoritmo de aproximación para recorrer las vistas buscando la mínima distancia. Sin embargo, en la segunda etapa del algoritmo no se incorpora criterio alguno que permita minimizar la distancia que recorre el robot. El algoritmo no es generalizable debido a la restricción de requerir un mapa del ambiente. Los modelos generados no cuentan con un índice de calidad. Además, su principal limitación es requerir un mapa previo del objeto.

3.4.7. Reconstrucción de ambientes

La reconstrucción de objetos y ambientes son problemas muy relacionados. Podemos definir a ambos de la siguiente forma: Dado un sensor de rango el ob-

jetivo es encontrar un conjunto de vistas que permita observar toda la superficie, del objeto o del ambiente. En reconstrucción de ambientes las siguientes vistas deben caer dentro del espacio libre previamente observado, a diferencia de la reconstrucción de objetos donde de antemano se conoce el espacio libre.

Debido a la semejanza entre los problemas de reconstrucción de ambientes y reconstrucción de objetos, es posible aplicar algunos conceptos y técnicas desarrollados de uno en el otro. A continuación se mencionan dos trabajos que propusieron técnicas potencialmente utilizables para reconstrucción de objetos.

En [Sanchiz et Fisher, 1999] se desarrolló un algoritmo para determinar la siguiente mejor vista, en el que se incorporó una novedosa función de utilidad, la cual permite evaluar que tan deseable es una vista con base en ciertos criterios.

El algoritmo propuesto [Sanchiz et Fisher, 1999] fue pensado para aplicarse a un sensor colocado en un robot con cinco grados de libertad: tres grados correspondientes a la posición del sensor en el ambiente (x, y, z) y dos grados correspondientes a la orientación del sensor (*pan* y *tilt*). La representación del ambiente utilizada fue un mapa de voxels. Los tipos de voxels utilizados fueron: *sin marcar*, *vacío*, *ocupado*, *oculto* y *occplane*.

Con el fin de evaluar que tan deseable es una vista, en [Sanchiz et Fisher, 1999] se establecieron los siguientes criterios: 1)Proveer traslape con los datos previamente adquiridos, 2) Eliminar áreas de plano de oclusión y 3) Observar áreas no vistas. Una vez establecidos los criterios, se propuso una función matemática, f_{area}, que aplicada a los porcentajes de voxels visibles desde una vista, evalúa qué tan deseable es ésta, la función f_{area} se muestra en la ecuación (3.2):

$$f_{area} = \left(5a_{ov}^3 + -10.5a_{ov}^2 + 6a_{ov}\right)\left(1 - \frac{1}{2}|a_{op} - a_{us}|\right) \tag{3.2}$$

donde a_{ov} es el porcentaje de traslape, a_{op} es el porcentaje de voxels de plano de oclusión y a_{us} es el porcentaje de área no vista.

La función f_{area} fue definida como el criterio básico y otros criterios, como

navegación o calidad, pueden ser agregados si son representados como factores f_i que multiplican al factor básico, como se muestra en la ecuación (3.3).

$$f = f_{area} \prod_i (1 + f_i) \tag{3.3}$$

La función de área fue uno de los principales aportes del trabajo mencionado, ya que, refleja qué tan deseable es una vista de acuerdo a los criterios que proponen los autores. Otro de los aportes fue el método para encontrar la siguiente mejor vista, el cual está basado en la combinación de una búsqueda exhaustiva y una búsqueda tipo *hill climbing*. Sin embargo, el método de búsqueda tiene la desventaja de caer en mínimos locales. Los resultados del método fueron mostrados en simulación sin lograr completar el ambiente objetivo.

Más tarde en [Lozano et al, 2002] se extendió el trabajo desarrollado en [Sanchiz et Fisher, 1999]. En este trabajo se definió una nueva función de utilidad y se utilizó el mismo algoritmo de búsqueda. Se lograron mejores resultados pero sin lograr explorar completamente la superficie del ambiente debido a que el algoritmo se estancaba en mínimos locales.

La función de utilidad presentada en [Lozano et al, 2002] fue diseñada para medir qué tan deseable es una vista de acuerdo a los criterios establecidos: proveer traslape, resolver oclusiones, observar áreas no vistas y mejorar la calidad actual del mapa de voxels. Sin embargo, la función tiene la desventaja de que aunque no haya voxels en una vista las evaluaciones son cercanas al valor máximo.

3.5. Resumen

La tabla 3.1 muestra el resumen de las características de cada algoritmo revisado. Dentro de los planificadores por métodos volumétricos para RTO, los potencialmente aplicables a robots móviles se reportaron en [Banta et al, 2000], [Wong et al, 1999] y [Massios et Fisher, 1998], debido a que son algoritmos gene-

ralizables.

Con base en el análisis de los algoritmos llegamos a la conclusión de que actualmente no hay un algoritmo de planificación de vistas para reconstrucción tridimensional de objetos no basada en un modelo que sea aplicable a robots móviles y que además asegure un traslape, reduzca la distancia de navegación, observe las superficies con buena calidad y reduzca la cantidad de vistas.

Trabajo	Calidad de la toma.	Alg. Generalizable	Traslape en vistas	Finalización autom.	Conoc. *a priori* min.	Libertad de forma	Restricciones de pos.	Criterio de distancia
[Papadopoulos-Orfanos et Schmitt, 1997]	-	-	-	S	S	S	S	-
[Massios et Fisher, 1998]	S	S	-	S	S	S	S	-
[Wong et al, 1999]	-	S	-	S	S	S	-	-
[Banta et al, 2000]	-	S	-	S	S	S	-	-
[Null et Sinzinger, 2006]	-	-	-	S	S	S	-	-
[Blaer et Allen, 2007]	-	-	-	S	-	S	S	P
[Sanchiz et Fisher, 1999]	-	-	S	-	S	S	S	-
[Lozano et al, 2002]	S	-	S	-	S	S	S	-

Tabla 3.1: Estado del arte de los métodos volumétricos. S indica que el algoritmo si cumple con la característica, - indica que no la cumple, y P indica que la cumple de forma parcial. Los criterios evaluados son los mostrados en la sección 3.2

El método propuesto en esta tesis se basa en muchos de los conceptos propuestos por los algoritmos revisados, empezando por una representación volumétrica que es fácil de implementar. De [Massios et Fisher, 1998] y [Wong et al, 1999] tomamos la idea de la esfera de vistas, debido a que reduce la complejidad del problema. De [Sanchiz et Fisher, 1999] tomamos la idea de una función de utilidad basada en porcentajes de voxels para evaluar una vista; cabe mencionar que la función de [Sanchiz et Fisher, 1999] fue desarrollada para reconstrucción de ambientes y no contempla la navegación ni la calidad de las tomas. De [Lozano et al, 2002]

tomamos el factor de calidad que utiliza en su función de utilidad; debido a que es el único trabajo que hace una predicción en las calidades de las tomas.

Nuestros aportes radican una novedosa función de utilidad y un par de estrategias de búsqueda. La función de utilidad incorpora criterios que consideramos deben existir en cada toma, y la estrategia de búsqueda permite hallar la siguiente mejor vista en tiempos aceptables y competentes con las técnicas más recientes.

En el siguiente capítulo se explica en detalle la conformación de nuestro planificador de vistas.

Capítulo 4

Planificador de vistas

En este capítulo se explica nuestro algoritmo de planificación de vistas. En la introducción de este capítulo se muestra el algoritmo general y en las demás secciones se detalla cada parte. Nuestras principales aportaciones, la función de utilidad y las estrategias de búsqueda, se encuentran en las secciones 4.4.1 y 4.4.2 respectivamente.

4.1. Introducción

Nuestro algoritmo planificador comienza tomando una imagen de rango del objeto desde una posición inicial establecida arbitrariamente. Los puntos adquiridos de la imagen de rango son convertidos al sistema de coordenadas del mapa de voxels generando un modelo parcial del objeto, mediante el etiquetado de los voxels. Entonces, la estrategia de búsqueda utiliza una función de utilidad para evaluar un conjunto de vistas de una esfera de vistas alrededor del objeto. La función considera la cantidad de voxels en la vista, sus normales y la distancia desde la posición actual del sensor. Despúes de la evaluación la siguiente vista es seleccionada. El siguiente paso consiste en mover el sensor de rango (robot) a la posición seleccionada para tomar nuevamente una imagen de rango y obtener un modelo más completo. El proceso se repite hasta que un criterio de paro es alcan-

zado: las vistas candidatas no proveen nueva información. El algoritmo 3 resume nuestro método.

Algoritmo 3: Planificador de Vistas Para Reconstrucción Tridimensional de Objetos

Entrada: Vista inicial (v_0)
Salida : Modelo del objeto (M)
Variables: Imagen de rango (Im); Siguiente mejor vista (nbv)

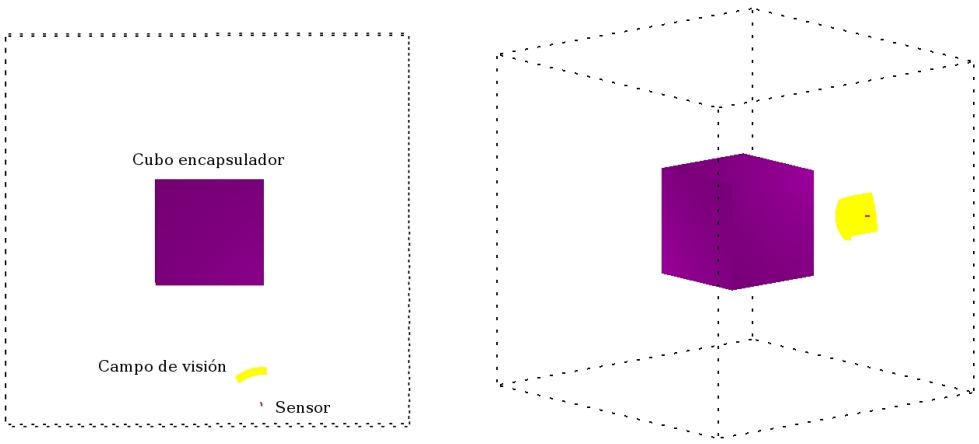
```
Posicionar el robot en v_0;
Im ← Capturar imagen de rango;
M ← Actualizar modelo M con la imagen Im;
while true do
    nbv ← Determinar la siguiente mejor vista ;
    if nbv no provee información then          /* criterio de paro */
        Devolver M;
        Terminar reconstrucción;
    end
    Posicionar el robot en nbv;
    Im ← Capturar imagen de rango;
    M ← Actualizar modelo M con la imagen Im;
end
```

4.2. Representación de la reconstrucción

Los mapas de voxels han sido ampliamente utilizados para representar el espacio 3D en diversas áreas de la computación. En planificación de vistas, además de ser utilizados como dominio de razonamiento, son utilizados como representación del modelo reconstruido, teniendo la desventaja de ser modelos de baja precisión o modelos precisos que ocupan gran cantidad de memoria. No obstante, a pesar de las desventajas su aplicación sigue siendo exitosa, puesto que, el espacio de almacenamiento no es actualmente un problema y la precisión del modelo final puede ser independiente del mapa de voxels, como se hace en [Blaer et Allen, 2007] al

utilizar las mediciones del sensor como modelo final. Una muy importante ventaja del mapa de voxels, en comparación con otras representaciones, es su facilidad de implementación y transferencia a otras estructuras, gracias a algoritmos como *Marching cubes* [Lorensen et Cline, 1987].

Debido a las ventajas que nos proporcionan los mapas de voxels, hemos desarrollado nuestro método basado en esta estructura. Al igual que trabajos previos, como [Sanchiz et Fisher, 1999, Massios et Fisher, 1998], a cada voxel se le han asignado tres atributos: etiqueta, normal y calidad. Las etiquetas posibles para cada voxel son: *ocupado*, *oculto*, *occplane*, *sin marcar* y *vacío* (definidos en la sección 2.3). La normal y la calidad sólo son asignadas a los voxels de tipo *ocupado*.



(a) Vista superior (b) Vista en perspectiva

Figura 4.1: Configuración inicial del mapa de voxels. Las líneas puntedas delimitan el mapa de voxels. En el centro se muestran los voxels sin marcar formando el cubo encapsulador del objeto. Dentro del mapa se muestra también la posición del sensor y el campo de visión del mismo.

El mapa de voxels, al inicio de la reconstrucción, está formado únicamente por dos tipos de voxels, *sin marcar* y *vacíos*. Los voxels *sin marcar* forman un cubo que encapsula al posible objeto. El cubo es llamado cubo encapsulador. Alrededor del cubo encapsulador se asignan voxels *vacíos* representando el espacio donde se puede colocar el sensor. La figura 4.1 muestra el mapa de voxels al inicio de una

reconstrucción.

4.3. Esfera de vistas

Una vista está determinada por seis parámetros $(x, y, z, \phi, \theta, \gamma)$, establecidos por el sistema de posicionamiento, más los parámetros del sensor utilizado (ver sección 2.1.1). Durante el proceso de reconstrucción es necesario determinar cada siguiente vista, hacer ésto en el espacio de configuraciones completo es muy complejo. Utilizar un conjunto discreto de vistas, o vistas candidatas, reduce la complejidad computacional del problema. El conjunto de vistas candidatas más utilizado es una esfera de vistas, que consiste un conjunto finito de puntos equidistantes del objeto, en los cuales se puede colocar el sensor.

Al utilizar una esfera de vistas suponemos que se cumplen los siguientes puntos:

- El objeto es visible desde cualquier vista. No hay obstáculos que se interpongan entre el objeto y el sensor.
- El campo de visión es de un tamaño tal, que permite observar por completo al objeto. Ésto es posible, dado que asumimos que el conocimiento *a priori* del objeto provee las dimensiones aproximadas del mismo.
- La resolución 3D del sensor es tal que al menos un punto en la imagen de rango representa al objeto.
- La distancia a la que se coloca el sensor es tal que el objeto se encuentra en el campo de profundidad del sensor.

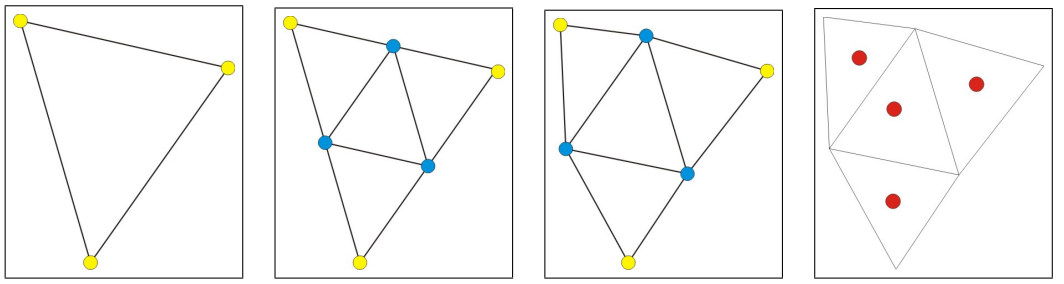
Con las suposiciones anteriores se eliminan de la planificación los parámetros intrínsecos del sensor. Así mismo, el parámetro de rotación del sensor (γ) puede ser eliminado dado que el campo de visión cubre el objeto. Por lo tanto únicamente consideraremos una vista como la configuración de cinco de los parámetros establecidos por el sistema de posicionamiento, ver ecuación (4.1):

$$v = (x, y, z, \phi, \theta) \tag{4.1}$$

4.3.1. Generación de la esfera de vistas

Los puntos de la esfera pueden ser obtenidos mediante diferentes técnicas: división de paralelos y meridianos [Connolly, 1985], mapas de discretización esférica [García et Velázquez, 1998] y *sphere tessellation* [Massios et Fisher, 1998]. *Sphere tessellation* es originalmente una técnica de gráficos computacionales utilizada para aproximar la forma de una esfera. La técnica parte de la forma de un sólido platónico y a través de subdivisiones en sus caras genera un modelo más refinado. En comparación con las técnicas antes mencionadas, *sphere tessellation* logra una discretización casi uniforme, motivo por el cual es utilizada como nuestro método para generar la esfera de vistas.

Las dimensiones de la esfera de vistas dependen de las dimensiones del objeto. El centro de la esfera (o) y el radio (r) deben de estar configurados de tal forma que se cumplan las suposiciones hechas en la sección anterior.



(a) Cara original (b) División (c) Proyección (d) Vistas Candidatas

Figura 4.2: Pasos del algoritmo *sphere tessellation*.

El algoritmo 4 muestra la forma en que generamos la esfera de vistas. Este algoritmo parte de la forma de un icosaedro (definido por un conjunto E de 12 vértices, y un conjunto C de 20 caras, donde cada cara está determinada por 3 vértices) y a través de l iteraciones determina un conjunto V de vistas candidatas. A l le llamaremos nivel de discretización. En cada iteración el algoritmo toma

los vértices de cada cara (Figura 4.2(a)), divide las aristas formadas por dichos vértices y genera un conjunto de nuevas caras (Figura 4.2(b)), finalmente proyecta cada vértice para que esté a una distancia r del centro (Figura 4.2(c)). Una vez finalizadas las l iteraciones, se toma el centroide de cada cara, se calcula la orientación del sensor hacia el centro de la esfera y se toma como un vista candidata (Figura 4.2(d)).

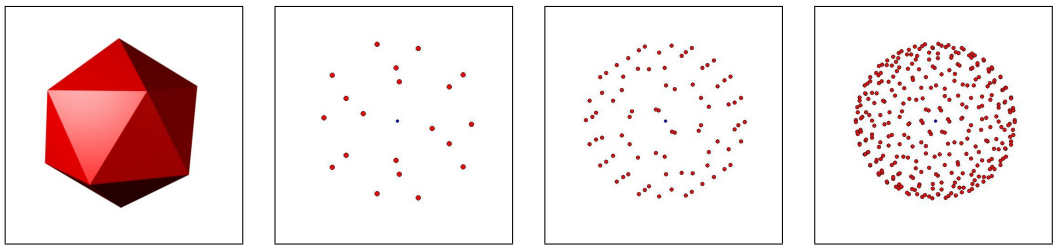
Algoritmo 4: Creación de Esfera de Vistas

Entrada: Nivel de discretización (l); radio (r); centro (o)
Salida : Conjunto de vistas (V)
Variables: Conjunto de caras (C); conjunto de caras temporal (C_{temp}); conjunto de cuatro caras (C_{cuatro}); cara (c); entero (n)

```
C ← Caras de un icosaedro con radio r y centro en o;
n ← 0;
V ← ∅;
while n < l do
    C_temp ← ∅;
    foreach c ∈ C do
        C_cuatro ← ∅;
        C_cuatro ← Dividir la cara c;        /* Se generan cuatro caras */
        Proyectar los vértices de cada cara del conjunto C_cuatro;
        Agregar las caras C_cuatro al conjunto C_temp;
        Eliminar c del conjunto C
    end
    C ← C_temp;
    incrementar n;
end
foreach c ∈ C do
    p ← Calcular centroide de la cara c;
    p ← Proyectar el punto p a una distancia r del centro;
    v ← Determinar la orientación del sensor desde p hacia o;
    Agregar v a V;
end
```

Con cada iteración del algoritmo se generan cuatro nuevas caras por cada cara existente. Por lo tanto, para un nivel de discretización l la cantidad de vistas generadas es $20 * 4^l$. En la figura 4.3 se muestran las esferas de vistas obtenidas

con diferentes valores de l.



(a) Icosaedro (b) $l = 0$ (c) $l = 1$ (d) $l = 2$

Figura 4.3: Icosaedro y esferas de vistas con diferentes niveles de discretización.

4.4. Siguiente mejor vista

El método de planificación que describimos al inicio del capítulo establece que en cada iteración es necesario determinar la vista que nos revele la cantidad óptima de información, o siguiente mejor vista (SMV). Con este objetivo hemos establecido un método basado en búsqueda, el cual utiliza las vistas candidatas para determinar cuál de ellas es la SMV.

El método de la siguiente mejor vista está formado por dos componentes. El primer componente es una función de utilidad, la cual determina qué tan deseable es una vista candidata. El segundo componente es una estrategia de búsqueda que determina qué vistas se deben evaluar.

4.4.1. Función de utilidad

La función de utilidad debe evaluar qué tan deseable es una vista, tomando en cuenta que una buena vista es aquella que:

1. Visualiza áreas no vistas y provee un porcentaje de traslape con la información actual del modelo, facilitando la fusión de la información en un único modelo.

2. Reduce la distancia de navegación desde la vista anterior intentando minimizar la distancia de navegación de toda la reconstrucción.

3. Observa la mayor cantidad posible de superficie desconocida en un intento por minimizar globalmente la cantidad de vistas necesarias para completar el modelo.

4. Adquiere la imagen de rango con buena calidad, es decir, coloca el rayo director del sensor lo más perpendicular posible a la superficie a observar.

Dadas las características anteriores, la función de utilidad está formada como la combinación de una serie de factores, o funciones matemáticas, que evalúan una característica en particular de la vista. Dichos factores son: área, navegación, oclusión y calidad.

Con el objetivo de evaluar una vista candidata, se hace un trazado de rayos simulando un sensor de rango dentro del mapa de voxels. Con el trazado de rayos se recolecta información de forma similar a capturar una imagen. La información recolectada es utilizada por cada uno de los factores para llevar a cabo la evaluación.

Factor de área

El objetivo del factor de área es observar áreas no vistas y al mismo tiempo observar una cierta cantidad de áreas previamente vistas. Así que este factor se basa en medir los porcentajes de cada tipo de voxel que son visibles desde una vista candidata, de tal forma que la siguiente vista mantenga un compromiso entre porcentajes.

Matemáticamente, el factor de área está representado como una suma de funciones f_i, donde f_i evalúa el porcentaje de un tipo de voxel i y retorna un valor en el rango [0,1]. f_i debe ser máxima cuando el porcentaje x_i de voxels i en la imagen sea el óptimo (α_i), es decir, $f_i = 1$ cuando $x_i = \alpha_i$. Cuando el porcentaje

es 0 significa que no hay voxels de ése tipo en la vista (una vista no deseable), por lo tanto, f_i es 0. Cuando el porcentaje es 1 significa que desde la vista sólo se observan voxels de un tipo (una vista que tampoco es deseable) así que la evaluación es 0. La figura 4.4 muestra el comportamiento deseado para f_i.

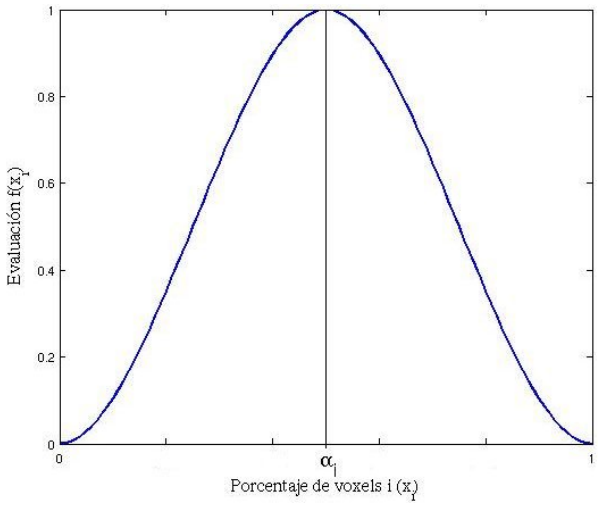


Figura 4.4: Comportamiento de la función f_i

Para que la función f_i, que a partir de ahora llamaremos f, observe el comportamiento antes mencionado debe estar sujeta a las siguientes restricciones:

$$f(0) = 0 \quad f'(0) = 0$$
$$f(\alpha) = 1 \quad f'(\alpha) = 0$$
$$f(1) = 0 \quad f'(1) = 0$$

$$f(x) > 0 \text{ para todo } x \in [0, 1]$$

$$f'(x) > 0 \text{ para todo } x \in [0, \alpha]$$

$$f'(x) < 0 \text{ para todo } x \in [\alpha, 1]$$

Dado que el pico de la función depende de α no podemos utilizar una función gaussiana. Tampoco podemos utilizar un único polinomio de tercer grado ya que no cumpliría con todas las restricciones. Por lo tanto, la función que utilizamos

para cumplir las restricciones está dividida en dos partes, cada parte formada por un polinomio de tercer grado, dicha función se muestra en la ecuación (4.2):

$$f(x) = \begin{cases} A_1x^3 + B_1x^2 + C_1x + D_1, & 0 < x \leq \alpha \\ A_2x^3 + B_2x^2 + C_2x + D_2, & \alpha < x < 1 \end{cases} \tag{4.2}$$

Aplicando las restricciones a la función (4.2) obtenemos dos sistemas de cuatro ecuaciones y cuatro incógnitas para cada ecuación:

$$\begin{aligned} D_1 &= 0 & A_2\alpha^3 + B_2\alpha^2 + C_2\alpha + D_2 &= 1 \\ A_1\alpha^3 + B_1\alpha^2 + C_1\alpha + D_1 &= 1 & A_2 + B_2 + C_2 + D_2 &= 0 \\ C_1 &= 0 & 3A_2\alpha^2 + 2B_2\alpha + C_2 &= 0 \\ 3A_1\alpha^2 + 2B_1\alpha + C_1 &= 0 & 3A_2 + 2B_2 + C_2 &= 0 \end{aligned}$$

Resolviendo los sistemas de ecuaciones obtenemos los coeficientes de (4.2) en función de α:

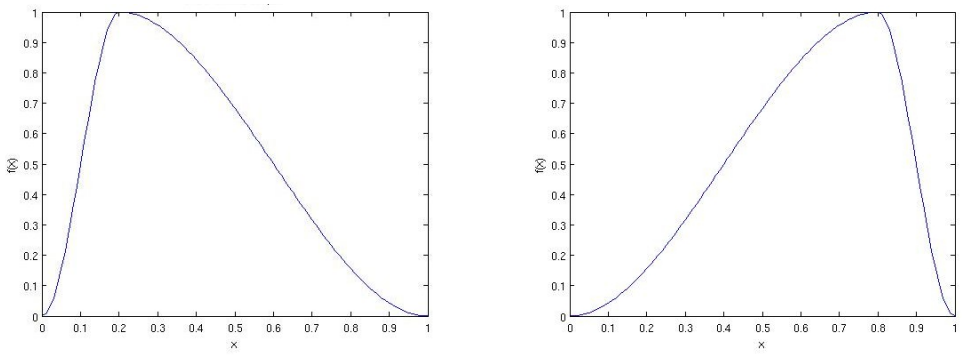
$$\begin{aligned} A_1 &= -\tfrac{2}{\alpha^3} & A_2 &= -\tfrac{2}{(\alpha-1)^3} \\ B_1 &= \tfrac{3}{\alpha^2} & B_2 &= \tfrac{3(\alpha+1)}{(\alpha-1)^3} \\ C_1 &= 0 & C_2 &= -\tfrac{6\alpha}{(\alpha-1)^3} \\ D_1 &= 0 & D_2 &= \tfrac{3\alpha-1}{(\alpha-1)^3} \end{aligned}$$

Nótese que los coeficientes C_1 y D_1 son cero, por lo tanto, la función f en función de α y x está determinada por la ecuación (4.3):

$$f(x,\alpha) = \begin{cases} -\tfrac{2}{\alpha^3}x^3 + \tfrac{3}{\alpha^2}x^2, & x \leq \alpha \\ -\tfrac{2}{(\alpha-1)^3}x^3 + \tfrac{3(\alpha+1)}{(\alpha-1)^3}x^2 - \tfrac{6\alpha}{(\alpha-1)^3}x + \tfrac{3\alpha-1}{(\alpha-1)^3}, & x > \alpha \end{cases} \tag{4.3}$$

La figura 4.5 muestra el comportamiento de la función (4.3) para diferentes valores de α. En dicha figura se observa que la función cumple con las restricciones antes establecidas.

Debido a que una vista debe observar un porcentaje de área desconocida y un porcentaje de traslape, f_{area} debe evaluar los porcentajes de voxels *occplane*



(a) $\alpha = 0{,}2$ (b) $\alpha = 0{,}8$

Figura 4.5: Comportamiento de la función f para diferentes valores de α

y voxels *ocupados*. Dado que queremos observar 80 % de área desconocida y un 20 % de traslape (porcentajes propuestos por [Lozano et al, 2002]) la función de área está definida como:

$$f_{area} = f\left(x, \alpha_x\right) + f\left(y, \alpha_y\right) \tag{4.4}$$

donde x es el porcentaje de voxels *occplane*, y el porcentaje de voxels *ocupados*, $\alpha_x = 0{,}8$ y $\alpha_y = 0{,}2$.

Factor de navegación

El objetivo del factor de navegación es evaluar una vista candidata con base en la distancia entre la configuración actual del sensor y la configuración de la vista candidata, de tal forma que considere como mejores vistas las que estén mas cerca de la configuración actual. Con la incorporación de este factor se busca que la distancia entre la posición actual del sensor y la vista seleccionada sea pequeña con respecto a la distancia a la vista candidata más lejana, esperando que al terminar la reconstrucción, la distancia total sea menor que si no se usa el factor.

Antes de desarrollar el factor de navegación es necesario tener una función de distancia entre dos vistas. La distancia ideal debería estar basada en una métrica

en el espacio de configuraciones del sistema de posicionamiento o robot. Así mismo la función de distancia debería contemplar la evasión de obstáculos y del objeto.

Debido a que tomamos una esfera de vistas, la distancia ortodrómica parece ser la más adecuada en nuestro caso. Dicha distancia se define como el camino más corto entre dos puntos de una superficie esférica (figura 4.6). En un trabajo futuro esta distancia se podría reemplazar por una distancia en el espacio de configuraciones del robot que incorpore la evasión de obstáculos.

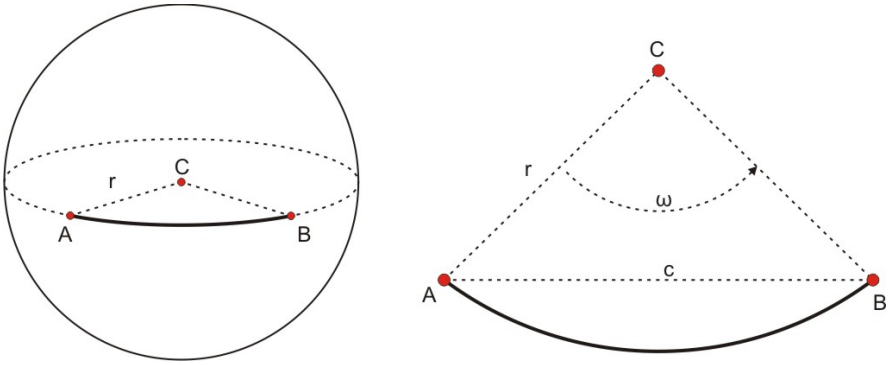


Figura 4.6: Distancia ortodrómica. La distancia ortodrómica entre los puntos A y B sobre la superficie de una esfera se encuentra marcada en la figura con una línea más gruesa. Todos los puntos de la línea son equidistantes al centro de la esfera.

La distancia ortodrómica entre dos puntos A y B sobre la superficie de una esfera con centro en C y radio r se obtiene mediante la ecuación (4.5).

$$d\,(A, B) = \omega * r \tag{4.5}$$

donde:

$$\omega = \arccos\left(\frac{2r^2 - c^2}{2r^2}\right) \tag{4.6}$$

y

$$c = ||B - A|| \tag{4.7}$$

Como la distancia ortodrómica máxima d_{MAX} está determinada por dos puntos separados por un ángulo ω de 180 grados, podemos normalizar la distancia

ortodrómica como:

$$d_{normalizada}\left(A,B\right)=\frac{\omega*r}{d_{MAX}} \tag{4.8}$$

donde

$$d_{MAX}=2\pi*r \tag{4.9}$$

Regresando al desarrollo del factor de navegación, se busca que dicho factor considere a las vistas candidatas más cercanas como mejores. Así que planteamos que este debe de cumplir los siguientes criterios (la figura 4.7 muestra el comportamiento deseado):

- La vista con la menor distancia debe tener la mayor evaluación, 1.
- La vista con la mayor distancia debe tener la menor evaluación, la cual será igual a ρ, siendo que $\rho \in [0,1]$. Al modificar el valor de rho se puede establecer que tanto se penalizan las vistas con distancias largas.
- Una vista con distancia mayor que otra siempre debe tener menor evaluación.
- En distancias cortas la evaluación decrece de manera más lenta.

Considerando el factor de navegación como una función (f) de la distancia ortodrómica normalizada, las restricciones anteriores se establecen de la siguiente forma:

$$\begin{aligned} &f\left(0\right)=1 \\ &f\left(1\right)=\rho \quad \text{para} \quad 0\leq\rho<1 \\ &f'(0)=0 \\ &f'\left(x\right)<0 \quad \text{para todo} \quad x\in(0,1] \end{aligned}$$

donde x es la distancia ortodrómica normalizada.

Para cumplir con las restricciones utilizamos un polinomio de grado dos mostrado en la ecuación (4.10). La ecuación (4.11) muestra la derivada del polinomio.

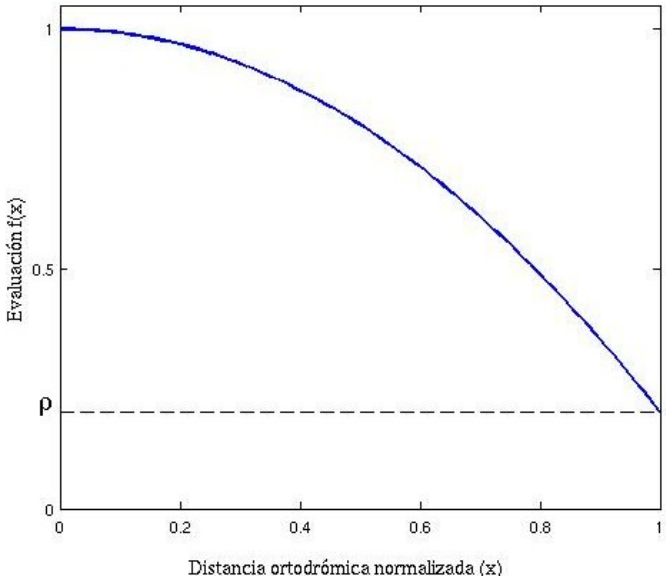


Figura 4.7: Comportamiento deseado del factor de navegación

$$f(x) = Ax^2 + Bx + C \tag{4.10}$$

$$f'(x) = 2Ax + B \tag{4.11}$$

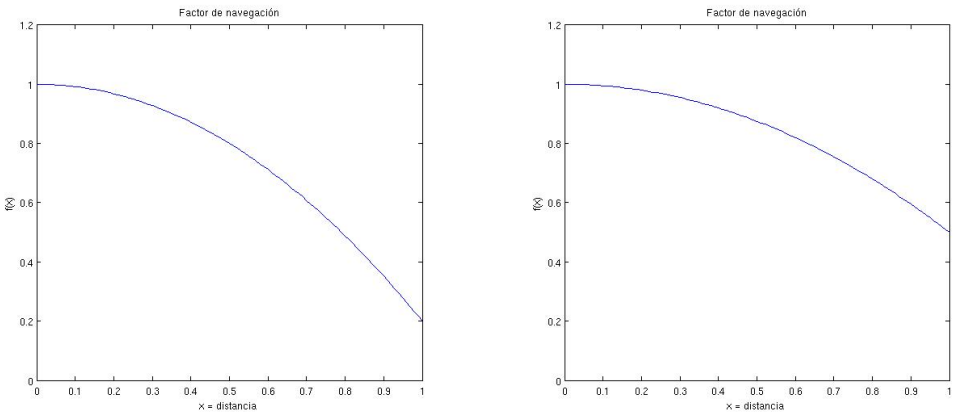
Aplicando las tres primeras restricciones al polinomio (4.10) y su derivada (4.11) obtenemos el siguiente sistema de ecuaciones:

$$\begin{aligned} C &= 1 \\ A + B + C &= \rho \\ B &= 0 \end{aligned}$$

Resolviendo el sistema, el valor de A en función de ρ es:

$$A = \rho - 1$$

Finalmente verificamos que la última condición se cumple. Sustituimos el valor



(a) $\rho = 0{,}2$ (b) $\rho = 0{,}5$

Figura 4.8: Comportamiento del factor de navegación con diferentes valores de ρ.

de los coeficientes en (4.11). Después aplicamos la condición $f'(x) < 0$ para obtener:

$$f'(x) = 2(\rho - 1)x < 0$$

Como podemos observar, el factor $(\rho - 1)$ es negativo cuando $\rho \in [0, 1]$ y x es positiva, dada la definición de distancia. Por lo tanto el resultado es negativo, cumpliéndose la condición.

Dado que hemos verificado que las condiciones se cumplen, el factor de navegación está dado por la ecuación (4.12):

$$f_{navegacion} = (1 - \rho)\, x^2 + 1 \tag{4.12}$$

donde:

$$x = d_{normalizada}\,(A, B) \tag{4.13}$$

En la figura 4.8 se muestran los comportamientos del factor de navegación con diferentes valores de ρ. En dicha gráfica podemos observar que modificando el valor de ρ se modifica el peso que tiene el factor de navegación. Entre más se acerque el valor de ρ a 1 el factor tiene menos peso en distancia largas.

Si se deseara utilizar otra medida de distancia en lugar de la distancia or-

todrómica bastaría con reemplazar (4.13) por otra distancia normalizada.

Factor de oclusión

El factor de oclusión tiene por objetivo observar la mayor cantidad de área desconocida. Entre mayor sea el área desconocida (*voxels occplane*) que provea una vista, mayor es la evaluación. Este factor se define en la ecuación (4.14):

$$f_{oclusion} = \frac{n\_occplane}{n\_rayos} \tag{4.14}$$

donde n_rayos es la cantidad de rayos lanzados para evaluar una vista y $n\_occplane$ la cantidad de *voxels occplane* observados.

Factor de calidad

El objetivo del factor de calidad es evaluar una vista a partir de la calidad que tendrá la observación. Una observación tiene buena calidad cuando el sensor es colocado perpendicularmente a la superficie observada.

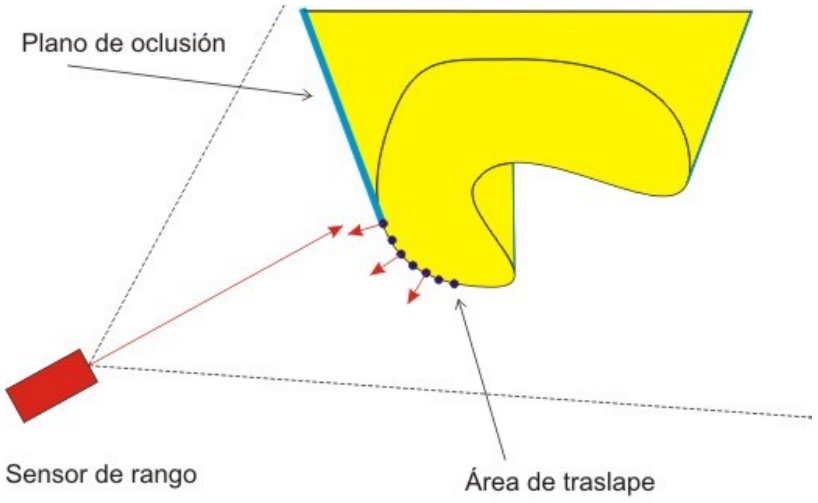


Figura 4.9: Sensor perpendicular a la superficie de traslape. El sensor es colocado de forma perpendicular a la superficie de traslape que se representa con voxels ocupados

Si desde una vista candidata se observa una área de traslape y una área de

plano de oclusión (Figura 4.9), la superficie de traslape puede proporcionar una predicción de la calidad de la superficie a observar. Si se coloca el sensor de forma perpendicular a la superficie de traslape es muy probable que la superficie a observar tenga una calidad similar. Además, al colocar el rayo director perpendicular a la superficie de traslape se mejora su calidad.

Basado en lo anterior, el factor de calidad evalúa como mejores vistas aquellas en que el rayo director sea perpendicular a la superficie de traslape, es decir, a los voxels ocupados. Por lo tanto este factor se define en la ecuación (4.15).

$$f_{calidad\_ocup} = \frac{\sum_{i=1}^{n\_ocup} \cos(\alpha_i)}{n\_ocup} \tag{4.15}$$

donde α_i es el ángulo formado entre la normal del voxel ocupado i y el rayo director de la vista, y n_ocup es la cantidad de voxels *ocupados*. Este factor fue previamente utilizado en [Lozano et al, 2002].

Combinación de factores

La función de utilidad es una combinación de los factores que mencionamos previamente, así mismo debe cumplir dos características:

- El factor de área, llamado primario, debe ser determinante, es decir, cuando la evaluación sea 0 la función completa debe ser 0.

- Los demás factores, secundarios, deben contribuir en la función de utilidad.

La función de utilidad que proponemos se muestra en la ecuación (4.16) como un producto del factor de área por la suma de los factores secundarios:

$$f_{utilidad} = f_{area} * (f_{calidad} + f_{navegacion} + f_{oclusion}) \tag{4.16}$$

4.4.2. Estrategia de búsqueda

Con el objetivo de determinar la siguiente mejor vista, la estrategia de búsqueda determina qué vistas se deben evaluar. Dicha estrategia hace uso de la función de utilidad cuando desea evaluar una vista candidata.

Estrategia Exhaustiva

La estrategia más sencilla es una estrategia exhaustiva. Dicha estrategia consiste en evaluar todas las vistas de la esfera. La evaluación es llevada a cabo mediante un trazado de rayos desde la vista candidata (la cantidad de rayos es la misma que la cantidad usada por el sensor de rango utilizado en la reconstrucción. La forma en la que se lanzan los rayos simula la geometría del sensor). Se contabilizan los voxels encontrados y se utiliza la función de utilidad para determinar qué tan buena es. La siguiente mejor vista es aquella vista candidata que tuvo la mayor evaluación. Esta estrategia tiene la inherente desventaja de ser computacionalmente muy costosa.

Estrategia Jerárquica Recursiva

La estrategia jerárquica recursiva evalúa una menor cantidad de vistas haciendo uso de una discretización jerárquica de la esfera de vistas. Dicha estrategia consiste de dos pasos. En el primer paso se evalúan las vistas candidatas del icosaedro (nivel de discretización $l = 0$) y se selecciona a la vista mejor evaluada. En el segundo paso se hace una división de la cara correspondiente a la vista seleccionada (alcanzando un nivel más de discretización). Con esta división se obtiene un nuevo conjunto de vistas candidatas, se evalúan dichas vistas candidatas, y se selecciona la mejor evaluada. El segundo paso se puede repetir recursivamente hasta alcanzar un nivel de discretización deseado.

El algoritmo 5 detalla la estrategia jerárquica recursiva. El algoritmo recibe como datos de entrada el nivel de discretización deseado (l) y el conjunto de vistas

candidatas del icosaedro (V) y devuelve la SMV.

Algoritmo 5: Estrategia Jerárquica Recursiva

Entrada: Nivel de discretización(l)
Salida : Siguiente mejor vista(smv)
Variables: Entero (i); conjunto de caras(C); conjunto de vistas (V); conjunto de vistas (H); cara (c)

$C \leftarrow$ Conjunto de caras de un icosaedro;
$V \leftarrow$ Determinar un conjunto vistas a partir de C; `/* El centroide de una cara es una vista */`
Evaluar el conjunto de vistas V;
$v \leftarrow$ Vista mejor evaluada de V;
$i \leftarrow 0$;
while $i < l$ **do**
  $c \leftarrow$ Determinar la cara que corresponde a la vista v;
  $H \leftarrow$ Dividir c; `/* Se generan cuatro caras */`
  Evaluar el conjunto de vistas H;
  $v \leftarrow$ Vista mejor evaluada de H;
  $smv = v$;
  Incrementar i;
end
Regresar smv;

Resolución Ideal de Trazado de Rayos

Para evaluar una vista se hace un trazado de rayos. A la cantidad de rayos trazados le denominamos resolución de trazado de rayos (RTR). Una RTR igual a la utilizada por la cámara de rango que observa el objeto la denominamos RTR completa.

Cuando se hace un trazado de rayos en el mapa de voxels, muchos de los rayos recorren los mismo voxels. Cuando una RTR es muy grande y el mapa de voxels tiene baja resolución existe información redundante ya que los voxels son evaluados de forma repetida por diferentes rayos. Con la resolución de trazado de

rayos ideal, se busca que los rayos trazados, en un radio de acción, toquen todos los voxels sin que existan rayos redundantes.

El cálculo de la RTR ideal se basa en crear una dispersión de los rayos adecuada para un cierto mapa de voxels. Así que para un mapa de voxels con longitud de voxel len y una cámara de rango con ángulos de apertura horizontal α y vertical β colocado a una distancia r del centro del mapa de voxels la resolución ideal esta determinada por la ecuación (4.17).

$$R_{ideal} = pts\_h \times pts\_v \tag{4.17}$$

donde pts_h es la cantidad de puntos horizontales, pts_v la cantidad de puntos verticales y se determinan aplicando las fórmulas (4.18) y (4.19).

$$pts\_h = \frac{\alpha}{\operatorname{arc\,sen}\left(\frac{len}{r}\right)} \tag{4.18}$$

$$pts\_v = \frac{\beta}{\operatorname{arc\,sen}\left(\frac{len}{r}\right)} \tag{4.19}$$

Estrategia Multiresolución

El trazado de rayos para evaluar una vista es demasiado costoso, debido a que prácticamente se simula un sensor dentro del mapa de voxels. La estrategia multiresolución busca reducir este costo computacional mediante la variación de la cantidad de rayos trazados, mientras la cantidad de vistas evaluadas permanece constante.

La estrategia multiresolución está formada por k etapas. En cada etapa se evalúa un subconjunto de vistas candidatas a una resolución determinada. En la primera etapa se evalúan todas las vistas con una resolución mínima. Se seleccionan las mejores vistas. Se evalúan las vistas seleccionadas con una resolución más alta y se vuelven a seleccionar las mejores. Al final sólo queda un conjunto pequeño que es valuado con la resolución ideal. El algoritmo 6 resume la estrategia

multiresolución.

Algoritmo 6: Estrategia Multiresolución

Entrada: Esfera de vistas (V), número de etapas (k), resolución mínima(R_{min})
Salida : Siguiente mejor vista(smv)
Variables: Resolución ideal (R_{ideal}); resolución para la etapa i (R_i); cantidad de vistas a seleccionar en la etapa i (n_i)

for *i=1:k-1* **do**
  Determinar la resolución R_i; `/* Ecuación 4.20 */`
  Evaluar V con resolución R_i;
  $V \leftarrow$ Seleccionar las n_i vistas mejor evaluadas;
end
Evaluar V con resolución R_{ideal};
$smv \leftarrow$ Seleccionar la vista mejor evaluada;
Regresar smv;

La resolución utilizada en cada etapa se debe de incrementar con respecto de la etapa anterior. Qué patrón sigue el incremento en la resolución no es algo que esté definido. Así que proponemos un incremento homogéneo desde la resolución mínima (R_{min}), hasta alcanzar la resolución ideal(R_{ideal}). Por lo tanto, la resolución para cada etapa se determina con la ecuación (4.20):

$$R_i = R_{ideal} - (k - i) * \left(\frac{R_{ideal} - R_{min}}{k - 1} \right) \tag{4.20}$$

La cantidad de vistas seleccionadas debe disminuir conforme avanzan las etapas. El número de vistas seleccionadas en cada etapa repercute en el desempeño del algoritmo. Con pocas vistas seleccionadas, el algoritmo es rápido pero se pierde eficacia. Con muchas vistas seleccionadas se gana eficacia pero se consume mucho tiempo.

Determinar cuantas vistas son seleccionadas en cada etapa queda a consideración del usuario del algoritmo. Sin embargo, como ayuda, proponemos una forma basada en una función exponencial para determinar cuántas vistas son seleccio-

nadas en cada etapa, según lo muestra la ecuación (4.21):

$$n_i = b^{k-i} \tag{4.21}$$

donde b es un parámetro establecido por el usuario.

Con la función anterior esperamos un comportamiento en el cual la reducción de vistas sea proporcional a la etapa. Por ejemplo, si se establece el parámetro $b = 5$ la selección de vistas es 125, 25, 5, 1; si se establece $b = 10$ la selección de vistas es 1000, 100, 10, 1.

La cantidad de etapas (k) tiene una relación inversa con el parámetro b. Al incrementar dicho parámetro la cantidad de etapas disminuye y se hace una selección más rigurosa. La relación entre k y el parámetro b está determinada por la ecuación (4.22).

$$k = argmax\left(b^k < |V|\right) \tag{4.22}$$

donde $|V|$ es el número de vistas candidatas.

Si se utiliza el incremento homogéneo de la resolución y la selección exponencial la estrategia es independiente de la discretización de la esfera. La ventaja de esto es que se pueden eliminar o agregar vistas al conjunto de vistas candidatas sin que se afecte la planificación (ventaja que no la tiene la estrategia jerárquica recursiva). A su vez, dado que la estrategia utiliza la RTR ideal, la planificación es independiente de la resolución del mapa de voxels.

4.5. Criterio de paro

Saber cuándo se ha observado toda la superficie del objeto es complicado dado que no conocemos el objeto. Existen criterios de paro que han sido utilizados en trabajos previos. En [Banta et al, 2000] se utiliza el criterio de paro de terminar la reconstrucción cuando el porcentaje de voxels ocupados no incrementa. Sin embargo, este criterio no es adecuado, ya que, se detiene el incremento únicamente

cuando se observan superficies previamente observadas. Otro criterio de paro es cuando ya no haya voxels *occplane* en el mapa, sin embargo, este criterio tampoco es adecuado ya que pueden existir voxels *occplane* inalcanzables para el sensor, por ejemplo, las superficies interiores de un jarrón.

En nuestro planificador hemos establecido el criterio de paro después que es determinada una vista (ver algoritmo 3 al inicio del capítulo). Si la vista determinada sólo observa una cantidad de voxels *occplane* menor que un umbral entonces el proceso termina. Este criterio de paro fue utilizado en [Blaer et Allen, 2007].

El umbral es determinado por el usuario del algoritmo. Cuando este umbral se establece cercano a cero el algoritmo tomará muchas imágenes de la superficie del objeto hasta que no queden voxels *occplane* visibles. Por el contrario, cuando el umbral es establecido con un número grande el algoritmo tomará menos imágenes pero es muy probable que queden superficies del objeto sin ser observadas. De acuerdo a lo observado en nuestros experimentos sugerimos un criterio de paro menor al $0.1\,\%$ de la cantidad de voxels en el cubo encapsulador.

4.6. Resumen

En este capítulo se explicó el planificador de vistas que se propone en esta tesis. Dicho planificador es un algoritmo iterativo de posicionamiento del robot, captura de la imagen, actualización del mapa de voxels y determinación de la siguiente mejor vista.

Las dos aportaciones principales de ésta tesis, una función de utilidad y dos estrategias de búsqueda, forman parte de la determinación de la SMV. La función de utilidad, se basa en la evaluación de una vista mediante una función matemática que evalúa las áreas de voxels visibles, la distancia de navegación, la cantidad de voxels *occplane* y la calidad predicha. Dicha función de utilidad se propone como un producto entre el factor de área y la suma de los demás factores. Por otra parte, las estrategias de búsqueda hacen uso de la discretización de la esfera de

vistas y de la resolución del trazado de rayos para reducir el costo computacional de determinar la SMV.

En el siguiente capítulo se muestran los resultados de utilizar en simulación el planificador con diversos objetos sintéticos. Más adelante, en el capítulo 6 se muestran los resultados del planificador en un ambiente real controlado.

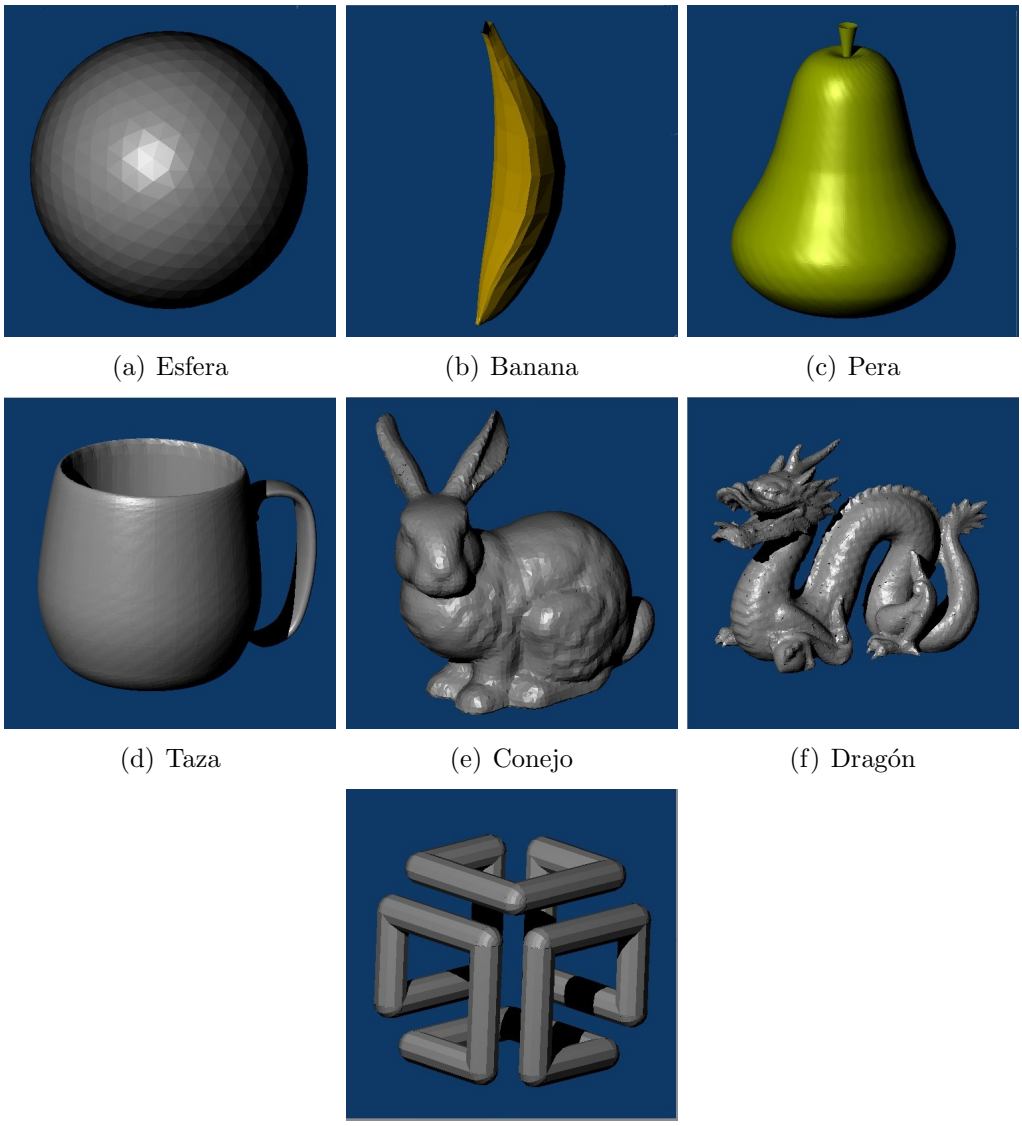
Capítulo 5

Resultados en simulación

En este capítulo se detallan las pruebas hechas con el planificador de vistas en simulación. La primera prueba consitió en utilizar cada estrategia en la reconstrucción de diversos objetos de diferentes complejidades. El objetivo fue medir el tiempo, la cantidad de vistas requeridas, la distancia de navegación y la calidad de los modelos en cada reconstrucción. La segunda prueba consistió en reconstruir objetos utilizando y no utilizando el factor de navegación. El objetivo fue medir el impacto de este factor en el planificador. La última consitió en utilizar diferentes resoluciones en el mapa de voxels.

Realizar las pruebas en simulación implicó utilizar objetos sintéticos y un sensor de rango virtual que analizara los objetos y produjera una imagen de rango similar a la de una cámara de rango real. La figura 5.1 muestra los objetos utilizados en las pruebas. El sistema de posicionamiento se tomó como un sistema de posicionamiento libre (*free-flyer*) con 5 grados de libertad (x, y, z, ϕ, θ).

La implementación fue realizada con el lenguaje de programación C, haciendo uso de la biblioteca OpenGl y bibliotecas para el manejo del mapa de voxels y trazado de rayos [Lozano et al, 2002, Sanchiz et Fisher, 1999].



(a) Esfera (b) Banana (c) Pera

(d) Taza (e) Conejo (f) Dragón

(g) Logo SGI

Figura 5.1: Objetos sintéticos utilizados en las pruebas del planificador de vistas.

5.1. Reconstrucción de objetos

El objetivo de esta prueba fue analizar los resultados del planificador utilizando las diferentes estrategias y los diferentes objetos. Con cada estrategia de búsqueda se reconstruyó cada uno de los objetos.

5.1.1. Configuración

La tabla 5.1 muestra cada estrategia utilizada y su respectiva configuración. La columna "T. por Iter." indica el tiempo de cómputo promedio por iteración.

Estrategia	**T. por Iter.**	**Configuración**
Exhaustiva 20	25.69 s	Se utilizaron 20 vistas y una resolución de trazado de rayos de 320×320
Exhaustiva 80	135.50 s	Se utilizaron 80 vistas y una resolución de trazado de rayos de 320×320
Jerárquica	30.58 s	La estrategia utilizó 2 etapas, 20 vistas en la primer etapa y un RTR de 320×320
Multiresolución	6.25 s	Se utilizó $k = 2$, $b = 10$ y 80 vistas. Las resoluciones de trazado de rayos para cada etapa fueron $R_1 = 40 \times 40$ y $R_2 = 105 \times 105$.

Tabla 5.1: Configuración de las estrategias de búsqueda.

Las pruebas se realizaron en una máquina AMD Turion 64 X2 de 1.9 GHz y 2 GB de memoria RAM con Sistema Operativo Ubuntu 8.04. La cámara de rango simulada tuvo una resolución de 320×320 puntos con ángulos de apertura de 45° en horizontal y 45°en vertical. La resolución del mapa de voxels fue $205 \times 205 \times 205$ voxels para el ambiente y $55 \times 55 \times 55$ voxels para el cubo encapsulador. Lo que significa, 8,615,125 voxels para el ambiente y 166,375 voxels para el objeto. El tamaño de un voxel se tomó como 0.02 m. El radio de la esfera de vistas fue 2m.

5.1.2. Resultados

La figura 5.2 muestra los modelos obtenidos por el planificador con la estrategia multiresolución (los modelos obtenidos con las demás estrategias son similares). Las tablas 5.2, 5.3, 5.4 y 5.5 muestran los resultados obtenidos con cada estrategia. Las tablas indican la cantidad de vistas tomadas, la cantidad de voxels ocupados, la calidad promedio del modelo obtenido (la calidad es un valor normalizado entre 0 y 1 y carece de unidades, ver sección 2.3.1) y la distancia ortodrómica total.

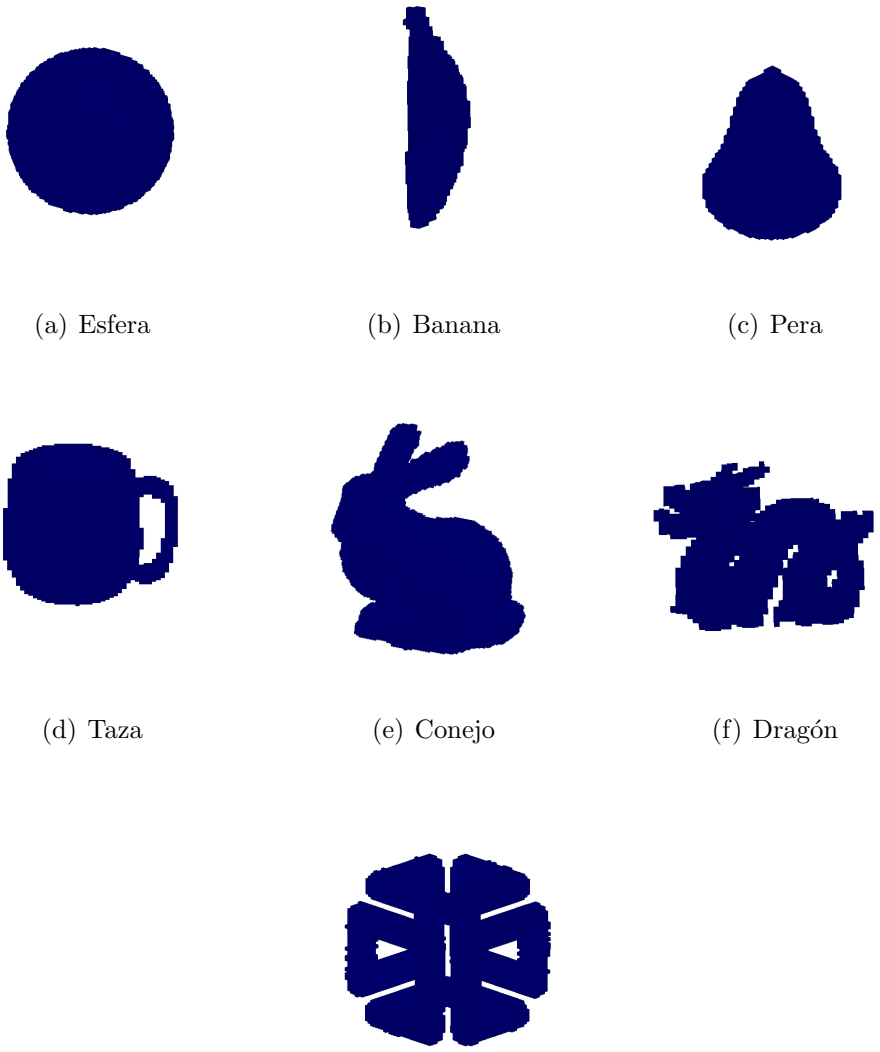
Los resultados muestran que el planificador, con la estrategia multiresolución y la exhaustiva 80, fue capaz de reconstruir todos los objetos. Los casos marcados con un asterisco ("*") indican que no se encontró la siguiente vista y el modelo quedó con voxels *occplane* visibles. El problema se debió a que la cantidad de vistas evaluadas (20) no fue suficiente.

5.1.3. Análisis

En esta prueba el objetivo fue comprobar si el planificador es capaz de reconstruir objetos con diferentes complejidades geométricas. Los resultados muestran que nuestro planificador es capaz de reconstruir dichos objetos.

Utilizando la estrategia multiresolución se obtuvieron los mejores resultados; se completaron los modelos y el tiempo de cómputo fue el menor. En el experimento pudimos notar también que la eficacia del algoritmo dependió de la cantidad de vistas candidatas. Con una mayor cantidad de vistas candidatas, en general, el algoritmo puede observar una mayor cantidad de superficie. Por otra parte, conforme el objeto tiene mas oclusiones o huecos es necesaria una mayor cantidad de vistas.

En [Wong et al, 1999] se utilizó una configuración similar a la nuestra y se reconstruyeron los objetos esfera, taza y logo SGI. La tabla 5.6 muestra la comparación de la cantidad de vistas requeridas por cada algoritmo. Las cantidades son similares, lo que indica que nuestro planificador produce resultados similares en



(a) Esfera (b) Banana (c) Pera

(d) Taza (e) Conejo (f) Dragón

(g) Logo SGI

Figura 5.2: Modelos generados por el planificador de vistas.

Objeto	Vistas	Vxls. occup.	Calidad	Distancia(m)
Esfera	6	7160	0.739	17.39
Pera	7	4343	0.813	14.77
Banana	4	1051	0.739	7.74
Taza	7*	7378	0.692	17.49
Conejo	9	5371	0.832	19.04
Dragón	7*	3482	0.725	17.48
Logo SGI	11*	11861	0.832	28.34

Tabla 5.2: Resultados del uso de la estrategia exhaustiva 20.

Objeto	Vistas	Vxls. occup.	Calidad	Distancia(m)
Esfera	6	7200	0.757	17.70
Pera	7	4330	0.803	16.40
Banana	4	1050	0.803	9.54
Taza	8	7736	0.718	21.76
Conejo	9	5392	0.825	21.32
Dragón	9	3595	0.777	24.46
Logo SGI	12	11952	0.843	34.02

Tabla 5.3: Resultados del uso de la estrategia exhaustiva 80.

Objeto	Vistas	Vxls. occup.	Calidad	Distancia(m)
Esfera	6	7141	0.730	14.71
Pera	7	4307	0.801	17.49
Banana	3*	934	0.711	6.28
Taza	8	7781	0.722	22.41
Conejo	9	5382	0.831	22.47
Dragón	9	3586	0.790	23.27
Logo SGI	11*	11854	0.838	27.53

Tabla 5.4: Resultados del uso de la la estrategia jerárquica.

Objeto	Vistas	Vxls. occup.	Calidad	Distancia(m)
Esfera	6	7200	0.757	17.70
Pera	7	4342	0.803	16.40
Banana	4	1049	0.756	7.15
Taza	8	7736	0.718	21.76
Conejo	9	5395	0.821	23.56
Dragón	9	3639	0.798	22.14
Logo SGI	12	11900	0.832	32.53

Tabla 5.5: Resultado del uso de la estrategia multiresolución.

Objeto	Nuestro planificador	[Wong et al, 1999]
Esfera	6	5
Taza	8	10
Logo SGI	12	13

Tabla 5.6: Comparación de cantidad de vistas.

cuanto a cantidad de vistas. No podemos hacer una comparación estricta ya que la cantidad de voxels en ambos mapas es similar pero no la misma y las vistas iniciales no son las mismas (desconocemos las vistas iniciales de [Wong et al, 1999]). El traslape, la calidad de las tomas y la navegación no fueron consideradas por dicho trabajo, así que no podemos compararlos con nuestros resultados.

Nuestros resultados de calidad de las tomas y distancia de navegación pueden ser utilizados como referencia para trabajos futuros que consideren dichos atributos.

5.2. Distancia de navegación

El objetivo de esta segunda prueba fue mostrar el efecto que tiene el factor de navegación en la reconstrucción. La prueba consistió en comparar los resultados cuando el factor de navegación es usado y cuando no es usado.

Se hicieron dos experimentos en la prueba. En el primero se reconstruyó cada uno de los objetos desde una vista inicial común y se compararon resultados. En el segundo experimento se reconstruyó un objeto desde 20 vistas iniciales diferentes y se compararon los promedios. El objetivo del primer experimento fue analizar el efecto en diferentes objetos. El objetivo del segundo fue determinar estadísticamente el efecto del factor de navegación en la reconstrucción de un objeto.

5.2.1. Configuración

En esta prueba se estableció el parámetro $\rho = 1$ (el factor de navegación no tiene efecto en la evaluación de una vista) y se comparó contra los resultados de usar el parámetro $\rho = 0$. Cuando $\rho = 1$ decimos que la reconstrucción es sin factor y cuando $\rho = 0$ decimos que la reconstrucción es con factor.

La resolución del mapa de voxels y la configuración de la estrategia multiresolución fueron las mismas de la prueba anterior.

5.2.2. Resultados

La tabla 5.7 compara los resultados del primer experimento. Las columnas con título "nav" muestran los resultados con el factor de navegación y las columnas con "sin-nav" muestran los resultados sin el factor de navegación. Por su parte, la tabla 5.8 compara los resultados del segundo experimento.

Objeto	**Vistas**		**Vxls. occup.**		**Calidad**		**Distancia**	
	nav	sin-nav	nav	sin-nav	nav	sin-nav	nav	sin-nav
Esfera	6	6	7200	7180	0.757	0.752	17.70	20.94
Pera	7	7	4342	4361	0.803	0.824	16.40	19.74
Banana	4	4	1049	1054	0.765	0.810	7.15	9.50
Taza	8	8	7736	7786	0.718	0.724	21.76	22.76
Conejo	9	9	5395	5354	0.721	0.814	23.56	23.86
Dragón	9	9	3639	3544	0.798	0.785	22.14	28.48
Logo SGI	12	12	11900	11875	0.832	0.833	32.53	37.06

Tabla 5.7: Efecto del factor de navegación en la reconstrucción. Las columnas con "nav" indican que se utilizó el factor de navegación.

Objeto	**Vistas**		**Vxls. occup.**		**Calidad**		**Distancia**	
	nav	sin-nav	nav	sin-nav	nav	sin-nav	nav	sin-nav
Taza	8.9	8.6	7832.3	7822.0	0.74	0.74	23.74	27.31

Tabla 5.8: Efecto promedio del factor de navegación. Las columnas con "nav" indican que se utilizó el factor de navegación.

.

5.2.3. Análisis

Como se puede observar en la tabla 5.7 hay una reducción significativa en la distancia de navegación, aproximadamente del 15 % en cuatro de los siete objetos probados. La reducción se hizo particularmente en los objetos con forma en su mayoría convexa. Sin embargo, estos resultados se obtuvieron a partir de una sola vista inicial. En el segundo experimento, donde se utilizaron 20 diferentes vistas iniciales, pudimos nuevamente comprobar que el factor de navegación reduce la distancia total, notando que hubo un ligero incremento en el promedio de vistas tomadas.

En la figura 5.3 se muestran los caminos recorridos en la reconstrucción del objeto banana del primer experimento. Como se observa en dicha figura, el camino recorrido con el factor de navegación tiene recorridos entre vistas mas pequeños, lo que hace que en la mayoría de los casos se reduzca la distancia total de navegación.

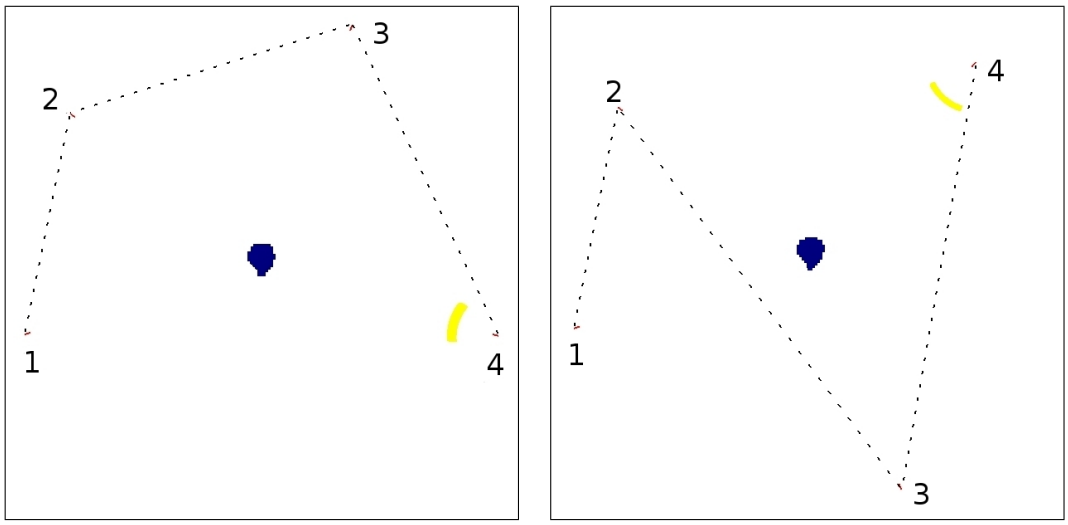
Dado que en el mejor de los casos se puede lograr una reducción de la distancia de navegación hasta en 33 % y en promedio se reduce en 13 % es muy recomendable incluir el factor de navegación en una reconstrucción. Con la inclusión del factor de navegación se reduce energía gastada por el sistema de posicionamiento.

5.3. Resolución del mapa de voxels

El objetivo de esta tercera prueba fue analizar el efecto del cambio de resolución del mapa de voxels en la reconstrucción.

5.3.1. Configuración

En esta prueba utilizamos cuatro diferentes resoluciones del mapa de voxels para reconstruir el objeto taza. Los tamaños de voxel fueron 0.05, 0.03, 0.02 y 0.01. La estrategia utilizada fue la multiresolución con parámetros $k = 2$, $b = 10$, $R_1 = 40 \times 40$. Dado que R_2 es igual a la resolución ideal, ésta dependió de la



(a) Camino con el factor de navegación (b) Camino sin el factor de navegación

Figura 5.3: Panorámica superior del camino de reconstrucción del objeto *banana*. Los números indican la iteración del algoritmo. Las vistas no se encuentran a la misma altura ya que es un espacio de tres dimensiones.

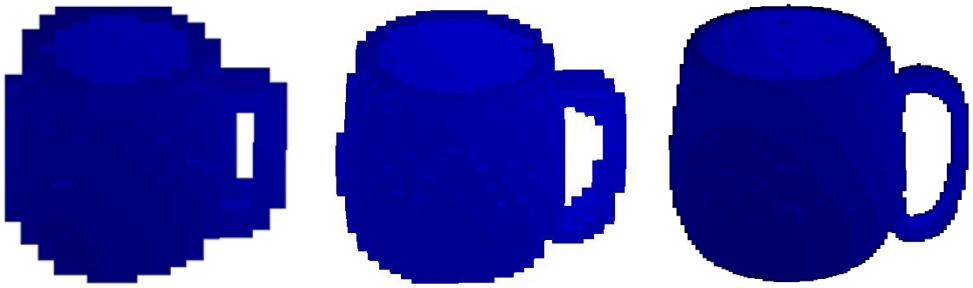
resolución del mapa de voxels y fue calculada automáticamente por el algoritmo. El criterio de paro fue establecido en 100 voxels para la resolución de 0.1 y 10 voxels para las demás resoluciones. La cantidad de vistas candidatas fue 80.

5.3.2. Resultados

La figura 5.4 muestra tres de los cuatro modelos reconstruidos. La tabla 5.9 muestra los resultados de cada reconstrucción. La columna *V. mapa* indica los voxels en el el mapa y la columna *V. cubo* los voxels en el cubo encapsulador. *T. vista* indica el tiempo promedio por iteración para determinar la vista. *Calidad* indica la calidad promedio del mapa de voxels. *Distancia* indica la distancia ortodrómica viajada.

5.3.3. Análisis

En los resultados pudimos notar que el algoritmo es capaz de reconstruir objetos a diferentes resoluciones. Además notamos que: i) La cantidad de vistas



(a) Res. 0.05 (b) Res. 0.03 (c) Res. 0.01

Figura 5.4: Comparación de modelos obtenidos a diferentes resoluciones. En la izquierda se muestra el modelo con un tamaño de voxel de 0.05 m, en la derecha se muestra el modelo con un tamaño de voxel de 0.01 m.

Res.	**V. mapa**	**V. cubo**	**Vistas**	**T. vista**	**Calidad**	**Dist.**
0.5	614,125	15,625	5	1.27 s	0.70	14.34
0.3	2,628,072	54,872	7	2.81 s	0.72	18.81
0.2	8,615,125	166,375	8	6.90 s	0.72	21.76
0.1	66,430,125	1,157,625	11	44.80 s	0.72	27.16

Tabla 5.9: Resultados de la reconstrucción del objeto taza con diferentes resoluciones del mapa de voxels.

aumenta con la resolución debido a que se necesitan observar más partes de la superficie para descartar voxels *occplane*; ii) el criterio de paro debe incrementar o disminuir de forma directa al la cantidad de voxels en el mapa; iii) el tiempo de cómputo se incrementa de forma directa al incremento en cada dimensión del mapa.

5.4. Comparación con otro enfoque

En nuestro planificador determinamos la siguiente vista como aquella que muestra traslape, reduce la distancia de navegación y procura buena calidad de las tomas, es decir busca un compromiso entre factores (CF). En la mayoría de los trabajos previos [Massios et Fisher, 1998, Wong et al, 1999, Blaer et Allen, 2007,

Banta et al, 2000, Null et Sinzinger, 2006, Blaer et Allen, 2007] la siguiente vista es aquella que provee la mayor cantidad de área desconocida (SVPMAD). En ésta prueba comparamos ambos enfoques. Los resultados muestran que nuestro enfoque es mejor que el enfoque utilizado por trabajos previos en los aspectos que consideramos debe tener una vista.

5.4.1. Configuración

La prueba consistió en utilizar un objeto y reconstruirlo desde diferentes vistas iniciales, tanto con nuestra función de utilidad como con una función que reflejara el enfoque SVPMAD.

El objeto reconstruido fue la taza, la cual consideramos un objeto difícil de reconstruir por sus auto-oclusiones y huecos. Se utilizó un conjunto de 20 vistas iniciales distribuidas alrededor del objeto. Se reconstruyó el objeto con el enfoque CF y con el enfoque SVPMAD con cada una de las 20 vistas iniciales, en total se hicieron 40 reconstrucciones, 20 con CF y 20 con SVPMAD.

Enfoque CF fue representado la función de utilidad (5.1).

$$CF = f_{area} * (f_{calidad} + f_{navegacion} + f_{oclusion}) \tag{5.1}$$

Por otro lado, el enfoque SVPMAD fue representado con con la función de utilidad (5.2).

$$SVPMAD = \frac{n\_occplane}{n\_rayos} \tag{5.2}$$

donde n_rayos es la cantidad de rayos lanzados para evaluar una vista y $n\_occplane$ la cantidad de *voxels occplane* observados.

Utilizamos la función de utilidad (5.2) por que, en nuestra representación, la mayor cantidad de área desconocida está representada como la mayor cantidad de voxels *occplane*. No consideramos los voxels *ocultos* debido a que, por definición, estos voxels siempre están ocluidos por un voxel *ocupado* o por un voxel *occplane*.

5.4.2. Resultados

La tabla 5.10 muestra el promedio de los resultados obtenidos. La figura 5.5 muestra la comparación gráfica de los resultados, en dicha figura los datos fueron normalizados para su mejor comprensión. Los datos completos se muestran en el anexo A.

Enfoque	Vistas	Distancia	Calidad	V. ocupados
SVPMAD	8.4	30.2	0.72	7809.6
CF	8.9	23.74	0.74	7832.3

Tabla 5.10: Comparación del enfoque CF con el enfoque SVPMAD

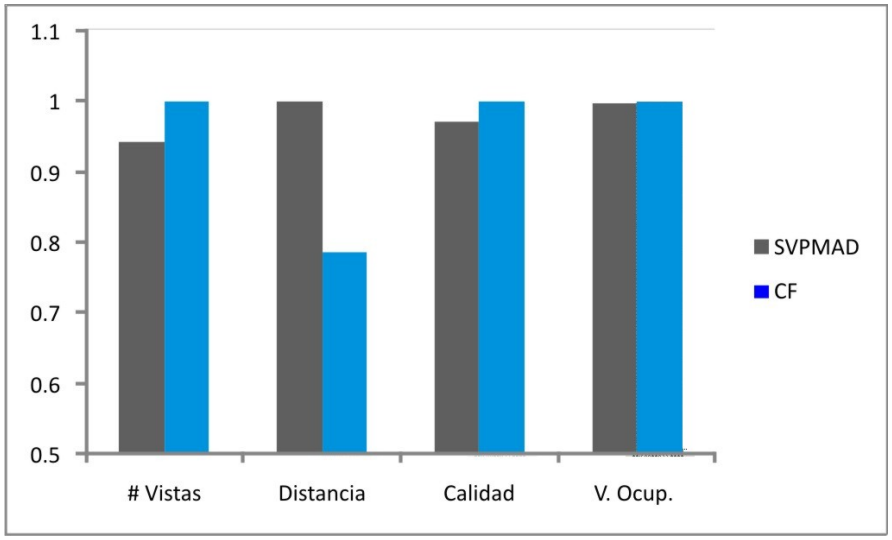


Figura 5.5: Comparación gráfica del enfoque CF con el enfoque SVPMAD.

5.4.3. Discusión

En los resultados se muestra que nuestro enfoque (CF) logró en promedio menor distancia, mayor calidad, y mayor cantidad de voxels ocupados.

Por otra parte la cantidad de vistas es mayor en CF respecto a SVPMAD. Ésto sucede por que con CF existió traslape en todas las vistas; mientras que con SVPMAD no en todas las vistas existió el traslape. Para ser específicos, con CF

la primer vista planificada [1] mostró en el 100 % de los casos un traslape cercano al 20 % de la imagen, mientras que, con SVPMAD únicamente el 20 % de los casos mostró traslape (el traslape fue menor al 2.2 % de la imagen).

5.4.4. Conclusión

De acuerdo a los resultados del experimento el enfoque CF mostró traslape en vistas, reducción de distancia de navegación y mejora de la calidad de las tomas. Por otra parte, el mismo enfoque CF requirió mayor cantidad de vistas debido al traslape que tienen todas las vistas.

Por lo tanto, consideramos que nuestro planificador con el enfoque CF proporciona una reconstrucción de objetos en la que se reduce la distancia de navegación, se proporciona un traslape en vistas y se mejora la calidad de las tomas. Como consecuencia el robot encargado de la reconstrucción ahorrará energía en la navegación, el proceso de registro será mas fácil gracias al traslape en las tomas y las tomas tendrán una calidad mejor comparado contra no utilizar nuestro enfoque.

5.5. Alta discretización y alta resolución

En esta prueba se utilizó el planificador con una alta resolución y un alto nivel de discretización de la esfera. La resolución del voxel fue de 0.1, equivalente a 66,430,125 voxels para el ambiente y 1,157,625 voxels para el cubo encapsulador. La cantidad de vistas candidatas se elevó a 320; cuatro veces más que las pruebas anteriores. Además se incrementó la resolución de la cámara de rango a 400 por 400 puntos, conservando los ángulos de apertura de 45 grados $\times$ 45 grados. La estrategia utilizada fue la multiresolución con configuración $k = 3$, $b = 7$ y resolución mínima de 40×40.

Los resultados de la reconstrucción, iteración por iteración, se muestran en

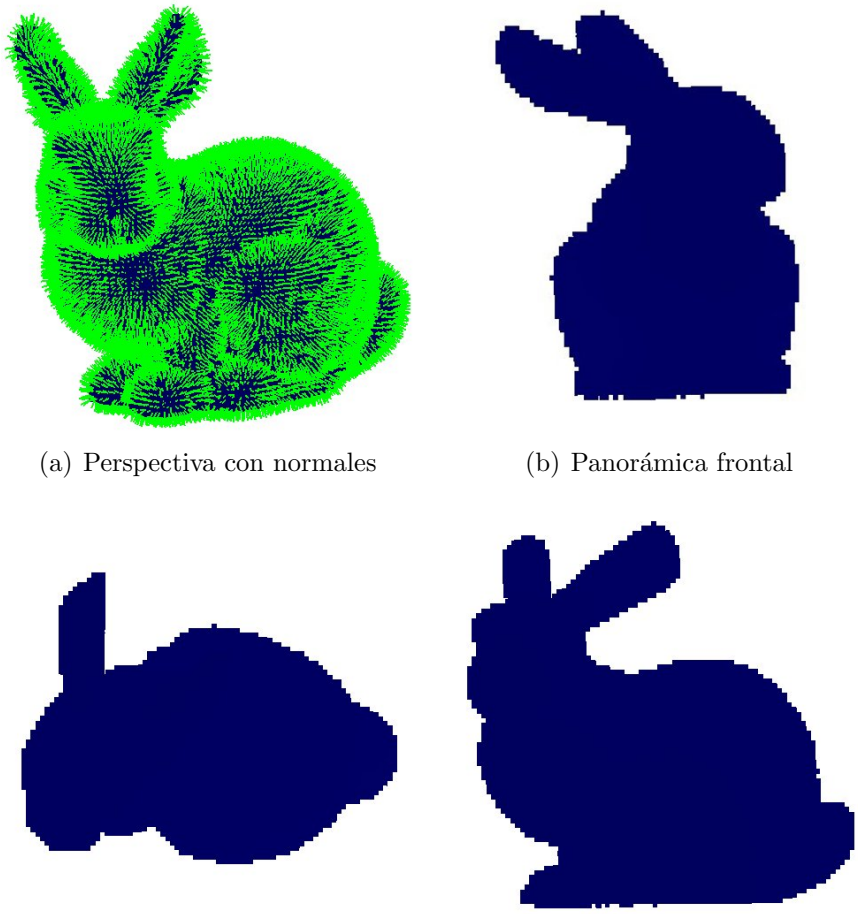
[1] La primer vistas planificada es la consecutiva a la vista inicial. No se considera la vista inicial por que ésta, por definición, no tiene traslape.

la tabla 5.11. La columna *# vista* indica el índice de vista determinada por el planificador en ésa iteración. *Eval* muestra la evaluación de la función de utilidad para la vista seleccionada. *V. occplane v.* indica la cantidad de voxels visibles desde la vista utilizada (cuando este valor es menor que el criterio de paro el proceso termina). *Tiempo* indica el tiempo de cómputo. *V. ocupados* indica la cantidad de voxels ocupados en el mapa. El modelo resultante se observa en la figura 5.6.

En la tabla se puede ver que el tiempo de cómputo fue de 118 segundos en promedio. Aunque el incremento en la cantidad de vistas fue de cuatro veces (80 a 320), el tiempo de cómputo sólo incrementó un poco más del doble (44 s a 118 s), gracias a la estrategia de multiresolución, que en esta ocasión se estableció con $k = 3$.

Iteración	**# Vista**	**Eval.**	**V. occplane v.**	**Tiempo**	**Calidad**	**Dist.**	**V. ocupados**
0	1	0	0	0.6767	0.682	0	6,395
1	84	3.1940	7391	107 s	0.6811	2.40	9,289
2	240	3.4278	6948	110 s	0.7000	1.84	12,376
3	186	2.8066	4999	121 s	0.7094	2.27	15,645
4	98	2.0072	3197	118 s	0.7278	2.77	18,735
5	148	0.5000	995	122 s	0.7568	2.50	19,820
6	281	0.0710	389	119 s	0.7858	1.57	20,358
7	293	0.0244	318	114 s	0.8074	4.45	20,809
8	309	0.0091	194	116 s	0.8243	2.46	21,084
9	261	0.0048	124	114 s	0.8319	3.82	21,295
10	230	0.0038	106	117 s	0.8437	4.10	21,500

Tabla 5.11: Datos de la reconstrucción de objeto conejo utilizando 230 vistas candidatas y una resolución de voxel de 0.1.



(a) Perspectiva con normales (b) Panorámica frontal

(c) Panorámica superior (d) Panorámica izquierda

Figura 5.6: Panorámicas del modelo del objeto conejo obtenido por el planificador.

5.6. Resumen

En este capítulo se presentaron las pruebas en simulación del planificador. Las pruebas involucraron las diferentes estrategias, diferentes objetos, el factor de distancia y diferentes resoluciones del mapa de voxels.

La prueba con las diferentes estrategias y diferentes objetos nos llevó a la conclusión de que nuestro planificador, con la estrategia multiresolución, es capaz de reconstruir objetos de forma geométrica arbitraria en tiempos aceptables, siempre y cuando la cantidad de vistas candidatas sea adecuada para la forma geométrica del objeto.

La prueba con el factor de navegación nos llevó a la conclusión que es mejor la inclusión del factor de navegación en una reconstrucción, dado que es muy probable que la distancia total se vea reducida.

En la tercera prueba pudimos mostrar que el planificador de vistas es capaz de operar a diferentes resoluciones del mapa de voxel, como consecuencia el modelo puede ser utilizado en diferentes aplicaciones.

En el siguiente capítulo se detalla el experimento que se hizo utilizando mediciones de un sensor físico. Las mediciones fueron tomadas con una cámara estereoscópica mientras se observaba un florero de talavera.

Capítulo 6

Resultados con datos de sensor físico

En este capítulo se detalla el experimento de reconstrucción de un objeto real. El objetivo fue probar que el planificador es capaz de ser utilizado en un robot móvil con datos de un sensor físico. En la sección 6.1 se detallan los componentes involucrados en la reconstrucción. En la sección 6.2 se explica como se llevaron a cabo los procesos requeridos por el planificador de vistas. En la sección 6.3 se muestran y se discuten los resultados obtenidos en el experimento.

6.1. Ambiente de reconstrucción

En el experimento realizado se utilizó una cámara estereoscópica, el robot Markovito [Aviles-Arriaga et al, 2009] con navegación manual, una esfera de vistas restringida y un objeto físico. La figura 6.1 muestra el ambiente de reconstrucción con los elementos mencionados.

Figura 6.1: Ambiente de reconstrucción.

6.1.1. Sensor de rango

Se utilizó la cámara estereoscópica STH-LMDCS-VAR/-C, cuyo fabricante es Videre [Videre Design, 2004]. Así mismo, se utilizó el sistema *Small Vision System* (SVS) [Konolige et Beymer, 2007, Konolige, 1997] para llevar a cabo los procesos de calibración, rectificación de las imágenes y determinación de las profundidades de la superficie.

6.1.2. Sistema de posicionamiento

El sistema de posicionamiento fue el robot Markovito con una navegación manual. El robot markovito cuenta con un módulo de localización y navegación que le permite desplazarse de un lugar a otro. Sin embargo, la precisión en la localización del robot no es adecuada para la tarea de reconstrucción. Por lo tanto, la navegación del robot fue manual, es decir, una persona colocó el robot en las posiciones determinadas.

Debido a los tiempos asignados a esta tesis y debido también a que no estaba en nuestros objetivos, no se implementó un algoritmo de navegación que solucionara el problema de localización. En un trabajo futuro se podría implementar dicho

algoritmo.

Grados de libertad

Dado que el robot sólo se desplaza en la superficie del área de trabajo y es capaz de girar sobre su propio eje, los grados de libertad del sistema de posicionamiento se limitan a tres: dos coordenadas de posición y una de orientación (x_r, y_r, θ_r).

Error de posicionamiento

El error de posicionamiento se origina cuando el sistema de posicionamiento no es capaz de alcanzar exactamente la posición deseada, ocasionando que las posiciones de los puntos observados no sean las correctas.

Las consecuencias del error de posicionamiento pueden ser contrarestadas con una etapa de registro de la información dentro de la actualización del mapa de voxels. En dicha etapa se fusiona la información en un solo modelo. Por ejemplo, en [Blaer et Allen, 2007] se utiliza el algoritmo *iterative closest point* [Zhang, 1994], el cual utiliza el traslape entre imágenes de rango para forzar los puntos de una imagen a coincidir con los puntos comunes de otra imagen, de tal forma que ambas formen una única nube de puntos.

La implementación de nuestro planificador no cuenta con una etapa de registro que fusione la información. Sin embargo, dado que nuestro planificador asegura que existe un traslape entre imágenes, en un trabajo futuro se puede incorporar dicha etapa.

En el experimento realizado, las consecuencias del error de posicionamiento no son gráficamente perceptibles, sin embargo, éstas existen. Para determinar la magnitud de las consecuencias del error de posicionamiento es necesario determinar la distancia que existe entre un punto observado y la posición del objeto real; determinar esa distancia es un proceso que no fue hallado en la literatura revisada y desde nuestro punto de vista es un proceso complejo. Por tanto no proveemos

explícitamente la magnitud del error de posicionamiento.

6.1.3. Espacio de Vistas

El espacio de vistas fue definido como el conjunto de combinaciones posibles de la configuración del sensor y la configuración del sistema de posicionamiento. Debido a que suponemos que el sensor está calibrado para observar el objeto y que el centroide del objeto está a la misma altura que las cámaras, los únicos parámetros a planificar son los parámetros del robot, (x_r,y_r,z_r).

Dado que el planificador de vistas trabaja con vistas definidas por cinco parámetros ,(x, y, z, ϕ, θ), y el robot únicamente tiene tres grados de libertad ,(x_r, y_r, θ_r), son necesarios dos procesos: uno que restrinja las vistas candidatas a las que son alcanzables y otro que convierta los parámetros del planificador a parámetros del robot.

Restricción de vistas

Para asegurar que las vistas son alcanzables por el planificador podemos hacer dos cosas: una es eliminar todas las vistas inalcanzables; otra es generar un conjunto de vistas compatible. Dado que sabemos que el sensor está a la misma altura que el objeto optamos por generar un anillo de vistas candidatas en lugar de la esfera de vistas.

Dado el radio (r), el centro del objeto ($o(0, h, 0)$) y la cantidad de vistas (n), una vista v_i del anillo está determinada según la ecuación (6.1):

$$v_i = (x_i, y_i, z_i, \phi_i, \theta_i) \tag{6.1}$$

$$x_i = r \sin \varphi_i$$
$$y_i = h$$
$$z_i = r \cos \varphi_i$$
$$\phi_i = \varphi_i - \frac{\pi}{2}$$
$$\theta_i = 0$$

donde $\varphi_i = i * \frac{2\pi}{n}$, $0 \leq i < n$, y h es la altura del centroide del objeto.

Conversión de parámetros

Este proceso convierte los parámetros del planificador (x, y, z, ϕ, θ) a parámetros del robot (x_r, y_r, θ_r). Con las suposiciones echas en esta sección 6.1.3 la conversión de parámetros es directa:

$$x_r = z$$
$$y_r = x$$
$$\theta_r = \phi$$

6.1.4. Objeto

El objeto utilizado como objetivo a modelar fue un florero de talavera (Figura 6.2). El florero tiene 35 centímetros de alto y 17 centímetros de diámetro en su parte más ancha. La textura es brillante y con diferentes figuras. Las figuras impresas en el florero facilitan la correlación de puntos en el algoritmo de visión; sin embargo, los reflejos en la superficie del objeto le afectan negativamente a dicho algoritmo.

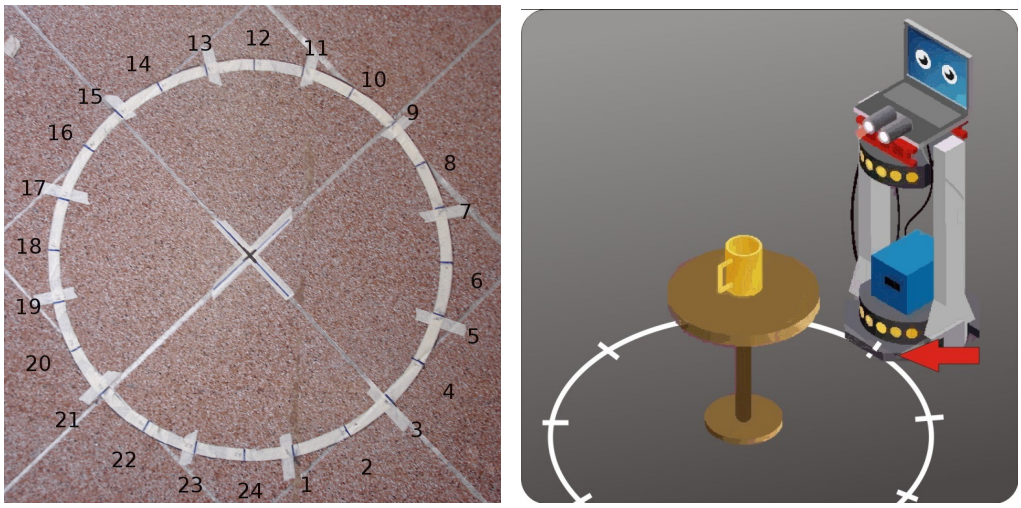
Figura 6.2: Florero de talavera.

6.2. Pasos de la Reconstrucción

La reconstrucción del objeto se hizo utilizando el planificador de vistas explicado en el capítulo 4. Dicho planificador contempla los pasos de: posicionamiento del robot, captura de la imagen de rango, actualización del mapa y planificación de la siguiente vista. A continuación se explican cada uno de estos pasos.

6.2.1. Posicionamiento del robot

El posicionamiento fue llevado a cabo manualmente colocando al robot en la posición que el planificador indicaba. Para facilitar la tarea se establecieron marcas en el piso indicando la posición de cada vista. La figura 6.3(a) muestra las 24 marcas establecidas. Cada que se debía posicionar el robot, éste era alineado con la marca que correspondía, de tal forma que la lente izquierda de la cámara apuntara hacia el centro del objeto (la lente izquierda indica la orientación de la cámara). En la figura 6.3(b) se muestra la alineación del robot.



(a) Marcas en el piso del anillo de vistas (b) Alineación del robot

Figura 6.3: Posicionamiento del robot. En (a) se muestran las vistas marcadas en el piso, los números muestran el índice de la vista. En (b) se muestra la alineación del robot al momento de colocarlo en una vista

6.2.2. Captura de la imagen de rango

La imagen de rango fue capturada con el sofware SVS [Konolige, 1997]. La calibración de la cámara fue hecha con el sofware *Small Vision System Calibration* al principio de cada reconstrucción.

Los parámetros del software SVS (*unique*, *speckle*, *multiscale*, *disparity*, *window*, *xoff*) fueron establecidos de forma manual en cada iteración del algoritmo de planificación. El objetivo de la configuración manual de los parámetros es lograr una buena observación de la superficie del objeto.

Idealmente los parámetros del sensor deben ser determinados por el planificador. Nuestro planificador no contempla dichos parámetros debido a que fue desarrollado pensando en un sensor genérico. En un trabajo futuro se podrían incluir estos parémetros en la planificación, de tal forma que la siguiente mejor vista busque la mejor configuración.

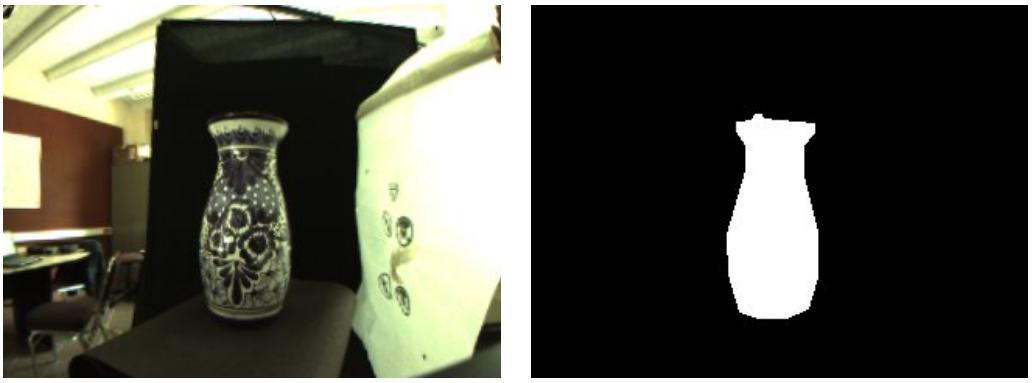
Segmentación del objeto

Las cámaras estereoscópicas determinan la posición en el espacio 3D a partir de la disparidad entre puntos. Por lo tanto todos los puntos que tengan disparidad en las imágenes proporcionan una profundidad, pertenezcan o no al objeto. Así que surge el problema: ¿cómo delimitar los puntos que sólo pertenecen al objeto que queremos reconstruir del resto de la imagen?, determinar dichos puntos es un problema de investigación no resuelto completamente. Trabajos como [Stiene et al, 2006] intentan obtener la silueta a partir de imágenes de rango y trabajos como [Stein et Hebert, 2007] buscan determinar los bordes de oclusión a través del movimiento de la cámara.

Desarrollar una técnica o implementar alguna existente que logre segmentar al objeto sin controlar el ambiente es un trabajo que queda fuera del alcance de esta tesis. Por lo anterior optamos por controlar de forma parcial el ambiente e implementar una sencilla técnica de segmentación.

El control del ambiente consistió en colocar una pantalla negra debajo y detrás del objeto. A partir del ambiente controlado se utilizó el software *Small Vision System* para obtener las imágenes. La segmentación utilizó la imagen rectificada (Figura 6.4(a)) y siguió los siguientes pasos: convertir la imagen a escala de grises y hacer una umbralización; hacer operaciones morfológicas para eliminar puntos aislados; conectar regiones; y seleccionar como el objeto a la región con centro más cercano al centro de la imagen. El resultado de la segmentación se puede observar en la figura 6.4(b). Una vez realizada la segmentación se eliminaron de la imagen de rango todos los puntos que no pertenecían al objeto.

Cabe mencionar que en cada iteración el umbral fue controlado manualmente para asegurar que el objeto fuera segmentado de forma correcta. En un trabajo futuro se podría automatizar por completo la segmentación.



(a) Ambiente controlado (b) Objeto segmentado

Figura 6.4: Segmentación del objeto. En la izquierda se muestra el escenario controlado durante la adquisición de una imagen de rango. En la derecha se muestra el resultado de la segmentación por umbralización.

6.2.3. Transformación de puntos

Una vez que se ha tomado la imagen de rango con la información de la superficie del objeto, cada punto capturado está representado en coordenadas del sistema de referencia de la cámara estereoscópica (figura 6.5). Por lo tanto es necesario trasladar cada punto al sistema de referencia del mapa de voxels antes de actualizar el mapa.

La figura 6.6) muestra tanto el sistema de referencia de la cámara como el sistema de referencia del mapa de voxels. Los ejes del sistema de referencia del mapa de voxels son X,Y,Z y el origen del dicho sistema es o. Los ejes del sistema de referencia de la cámara son u, v, w y el origen es e. Un punto de la superficie es considerado como p, sus coordenadas con respecto de la cámara son (u_p, v_p, w_p) y sus coordenadas con respecto del mapa de voxels son (x_p, y_p, z_p).

Para trasladar un punto de un sistema de coordenadas a otro es necesario conocer la orientación de los ejes y la posición del origen del primer sistema con respecto del segundo. En este caso, es necesario conocer e,u,v,w con respecto de los ejes X,Y,Z para poder la traslación (figura 6.6).

De acuerdo a nuestro planificador, una vista está representada como un vector con origen en (x, y, z) y orientación (ϕ, θ). Así que el origen del sistema de coor-

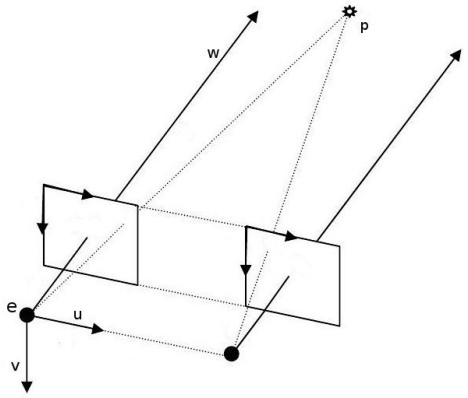


Figura 6.5: Sistema de coordenadas de la cámara estereoscópica. e representa el origen del sistema de coordenadas, dicho origen se encuentra fijo en el lente izquierdo de la cámara estereoscópica. u, v y w son los ejes base del sistema de coordenadas. p indica un punto observado en la superficie del objeto.

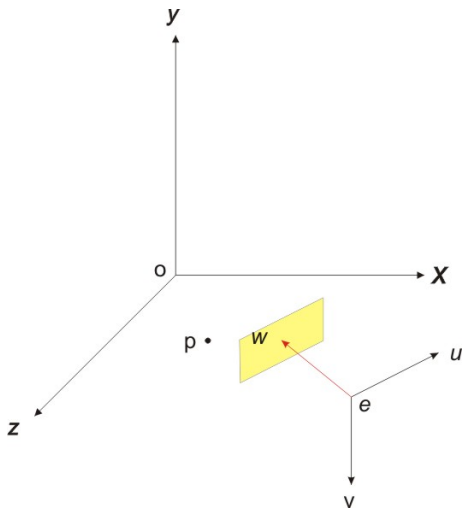


Figura 6.6: Sistemas de referencia. La figura muestra un punto de la superficie del objeto (p), el sistema de coordenadas del mapa de voxels (X,Y,Z) y el sistema de coordenadas de la cámara (u,v,w).

denadas de la cámara estereoscópica, e, está determinada por las coordenadas de posición de una vista, según se expresa en la ecuación (6.2).

$$e = \begin{bmatrix} x_e \\ y_e \\ z_e \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \tag{6.2}$$

Si por el momento no consideramos la orientación de la vista entonces cada vector base del sistema de referencia de la cámara en coordenadas del mapa de voxels está por la expresión (6.3):

$$u_0 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, v_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, w_0 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{6.3}$$

Por otra parte, las rotaciones de *pan* y *tilt* son rotaciones sobre los ejes Y y Z respectivamente:

Rotación de $pan(\phi)$:

$$R_Y^{\phi} = \begin{bmatrix} cos(-\phi) & 0 & sen(-\phi) \\ 0 & 1 & 0 \\ -sen(-\phi) & 0 & cos(-\phi) \end{bmatrix} \tag{6.4}$$

Rotación de $tilt(\theta)$:

$$R_Z^{\theta} = \begin{bmatrix} cos(-\theta) & -sen(-\theta) & 0 \\ sen(-\theta) & cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{6.5}$$

Por lo tanto, si utilizamos las rotaciones de *pan* (ecuación 6.4), y *tilt* (ecuación 6.5) en los vectores base (ecuaciones 6.3) obtenemos u,v y w con respecto de X,Y,Z (lo que necesitábamos conocer):

$$u = R_Y^{\phi} R_Z^{\theta} u_0 = \begin{bmatrix} -sen(\phi) \\ 0 \\ cos(\phi) \end{bmatrix} \tag{6.6}$$

$$v = R_Y^{\phi} R_Z^{\theta} v_0 = \begin{bmatrix} cos^2(\theta) \\ -sen(\theta) \\ sen(\phi)cos(\theta) \end{bmatrix} \tag{6.7}$$

$$w = R_Y^{\phi} R_Z^{\theta} w_0 = \begin{bmatrix} sen(\phi)cos(\theta) \\ cos(\theta) \\ sen^2(\phi) \end{bmatrix} \tag{6.8}$$

Una vez determinados e, u, v, w con respecto del sistema de coordenadas del mapa de voxels, un punto p capturado por la cámara, en coordenadas del mapa de voxels, está dado por la ecuación (6.9).

$$\begin{bmatrix} x_p \\ y_p \\ z_p \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & x_e \\ 0 & 1 & 0 & y_e \\ 0 & 0 & 1 & z_e \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_u & x_v & x_w & 0 \\ y_u & y_v & y_w & 0 \\ z_u & z_v & z_w & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_p \\ v_p \\ w_p \\ 1 \end{bmatrix} \tag{6.9}$$

Para obtener más información acerca de transformación de coordenadas véase [Shirley, 2005].

6.2.4. Siguiente mejor vista

El experimento se realizó colocando el robot en la vista 1, se tomó la imagen, se actualizó el mapa de voxels, y se determinó la siguiente mejor vista. Una vez determinada la siguiente vista se colocó el robot en la nueva posición y se repitió el proceso hasta que se alcanzó el criterio de paro.

Para determinar la siguiente mejor vista utilizamos nuestro planificador con la

estrategia multiresolución. Los parámetros fueron $b = 10$, resolución mínima (R_1) de 40×40 y $k = 2$. La resolución ideal calculada (R_2) fue 276×210.

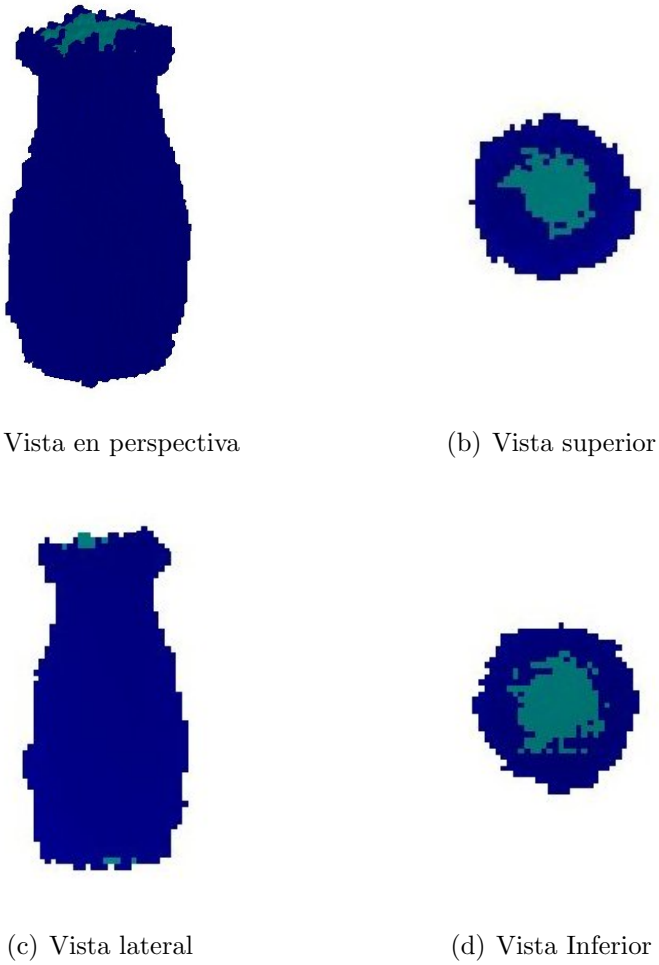
El radio del anillo de vistas fue de 49.7 cm. El tamaño del cubo del escenario fue de 1 m por lado. El cubo encapsulador fue de 21 cm por lado. La resolución de un voxel fue de 0.6 cm. El mapa de voxels tuvo una resolución de $172 \times 172 \times 172$ voxels (5,088,448 de *voxels*) y el cubo encapsulador $68 \times 68 \times 68$ voxels (314,432 voxels). El umbral de paro se estableció en 50 voxels *occplane* en la siguiente mejor vista. Este umbral de paro fue establecido en 50 voxels dado que es un umbral que funcionó de forma adecuada en las reconstrucciones con configuraciones similares realizadas en simulación.

6.3. Resultados

El modelo obtenido se muestra en la figura 6.7. Como se puede observar, el modelo es similar al objeto real. Los voxels *occplane* restantes indican las áreas correspondientes no fueron observadas debido a las limitaciones del sensor.

La tabla 6.1 muestra los datos de la reconstrucción. *# vista* indica el índice de la vista determinada por el planificador en ésa iteración. *Eval* muestra la evaluación de la función de utilidad para la vista seleccionada. *V. occplane v.* indica la cantidad de voxels visibles desde la vista utilizada (cuando este valor es menor que el criterio de paro el proceso termina). *Tiempo* indica el tiempo que se tarda la estrategia en determinar la siguiente vista. *V. ocupados* indica la cantidad de voxels ocupados en el mapa de voxels. Los datos mostrados en cada renglón fueron capturados al terminar cada iteración.

Se tomaron 5 vistas hasta alcanzar el criterio de paro. En la figura 6.9 muestra cada una de las iteraciones de la reconstrucción. En la figura 6.10 se muestran las posiciones del robot en cada iteración.



(a) Vista en perspectiva (b) Vista superior

(c) Vista lateral (d) Vista Inferior

Figura 6.7: Modelo generado por el planificador de vistas del florero de talavera. En azul se muestran los voxels *ocupados* y en cian se muestran los voxels *occplane*.

6.4. Análisis

En el experimento anterior se pudo comprobar que nuestro planificador es capaz de reconstruir objetos cuando se incorporan datos de sensores reales. El resultado fue un modelo del florero adecuado para manipulación. Las áreas sin observar (figura 6.7) se debieron a dos limitantes: una es la capacidad de la técnica binocular para reconstruir superficies; otra es el reducido alcance del sistema de posicionamiento, el cual no es capaz de colocar el sensor en las partes superiores e inferiores del objeto.

Iteración	# vista	Eval.	V. occplane v.	Tiempo (s)	Calidad	Dist. (m)	V. ocupados
0	1	0	0	0	0.682	0	1791
1	19	2.8479	2326	4.27 s	0.712	65.057	3422
2	14	2.7581	1987	4.70 s	0.719	65.057	4959
3	8	1.8059	858	4.68 s	0.720	78.069	6517
4	3	0.0373	92	4.63 s	0.728	65.057	7693

Tabla 6.1: Resultados de la reconstrucción del florero de talavera.

El tiempo de cómputo de la siguiente mejor vista fue de 4.57 s en promedio (tabla 6.1), incluso si se escoge un $b = 5$ en la estrategia este tiempo se reduce casi a la mitad. Por otra parte, el tiempo de actualización del mapa fue de 1.85 segundos en promedio. Si sumamos ambos tiempos tenemos un tiempo promedio por iteración de 6.42 segundos. Con el tiempo logrado, podemos decir que el planificador produce soluciones adecuadas para la reconstrucción de objetos.

En la reconstrucción se pudieron observar las características que deseábamos de una reconstrucción: traslape entre vistas, cantidad de imágenes tomadas relativamente pequeña y una corta distancia de navegación. La figura 6.8 muestra el camino recorrido.

El planificador propuesto en esta tesis es capaz planificar cada vista. Sin embargo, para poder reconstruir un objeto de forma automática es necesario resolver los problemas de segmentación del objeto, navegación y fusión de información. Ésta tesis puede ser base de trabajos futuros que lleguen a realizar la RTO de forma automática; no sólo en robots móviles, sino en diversas aplicaciones donde se requiera generar modelos 3D de objetos físicos.

En trabajos futuros se puede mejorar la planificación que se hace en nuestro algoritmo. Por ejemplo, incluir los parámetros del sensor, mejorar el desempeño o planificar vistas cuando hay más de un objeto en la escena.

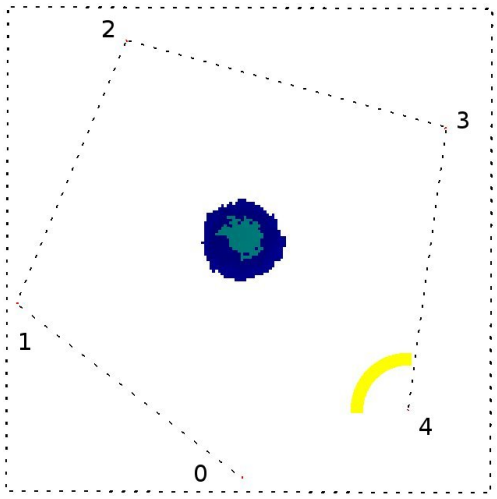


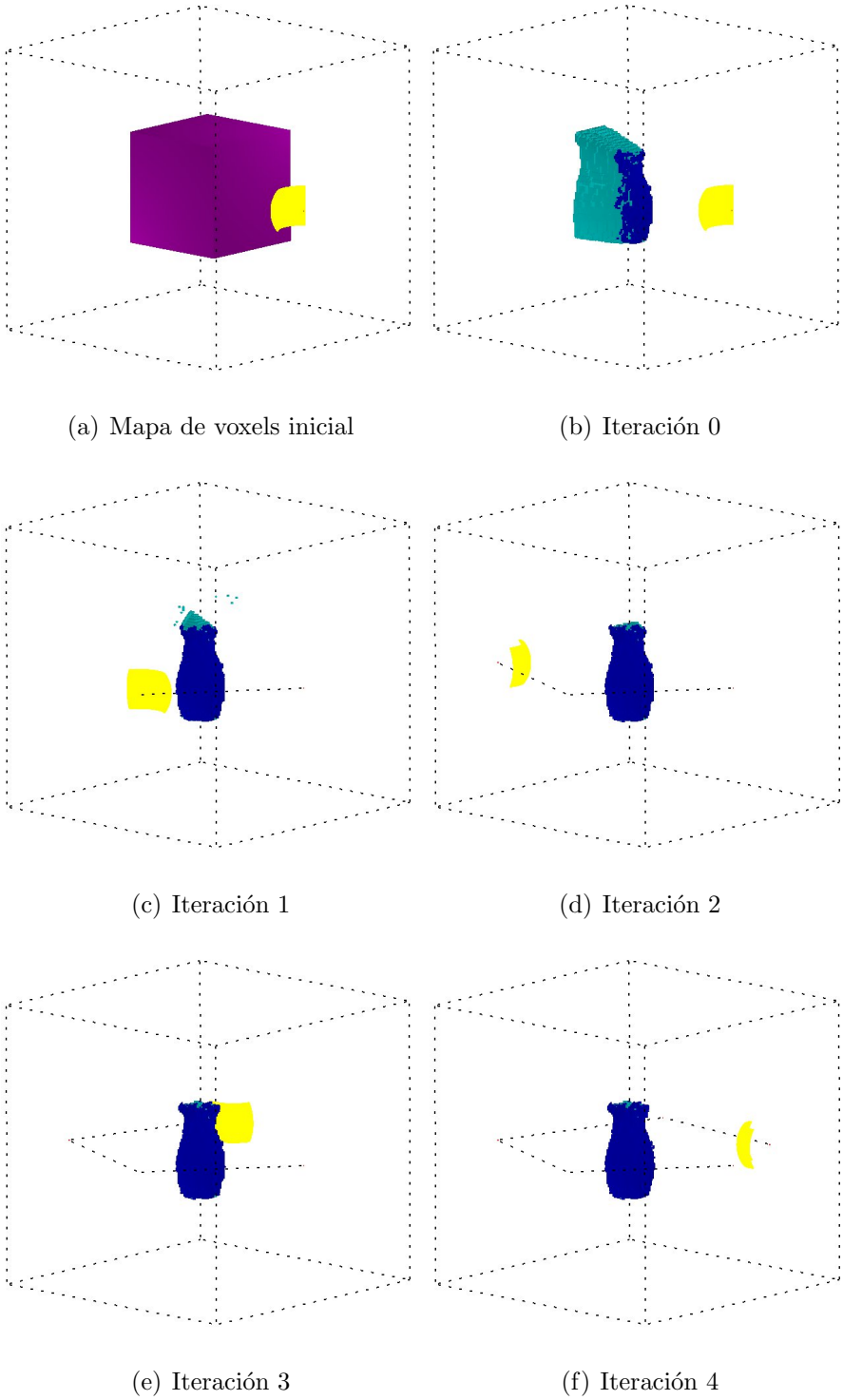
Figura 6.8: Panorámica superior del camino de reconstrucción. Los números indican la iteración del algoritmo de planificación. Las líneas punteadas indican la relación de secuencia entre vistas, estas líneas no indican el camino recorrido por el robot durante la navegación manual.

6.5. Resumen

En este capítulo se mostró la reconstrucción de un objeto físico con el planificador. Se mostraron también cada uno de los procesos que se realizaron para hacer la reconstrucción: posicionamiento del robot; captura de la imagen de rango; transformación de puntos.

Para el posicionamiento del robot se utilizó un posicionamiento por el usuario. Para la captura de la imagen de rango se utilizó el software *svs* y se complementó con la implementación de una sencilla técnica de segmentación. La transformación de puntos fue realizada de acuerdo a la configuración de la cámara estereoscópica.

Con el experimento realizado comprobamos que el planificador es aplicable a robots móviles y que la planificación cumplió las características que deseábamos. Además mostramos que es necesario resolver varios problemas para realizar una reconstrucción automática, como son segmentación, navegación y planificación de parámetros del sensor.



(a) Mapa de voxels inicial

(b) Iteración 0

(c) Iteración 1

(d) Iteración 2

(e) Iteración 3

(f) Iteración 4

Figura 6.9: Iteraciones de la reconstrucción del florero. las figuras muestran cada una de las iteraciones de la reconstrucción, a excepción de (a) que muestra la configuración inicial del mapa de voxels

(a) Iteración 0

(b) Iteración 1

(c) Iteración 2

(d) Iteración 3

(e) Iteración 4

Figura 6.10: Posiciones del robot. las figuras muestran la posición del robot en cada iteración de la reconstrucción

Capítulo 7

Conclusiones y Trabajo Futuro

Generar un modelo geométrico a partir un objeto físico es un problema que surge en diferentes áreas de la computación, en especial en robótica. Para llevar a cabo una reconstrucción no basada en un modelo es necesario hacer una planificación iterativa de las vistas que permitan reconstruir el objeto.

Los trabajos existentes en planificación de vistas, en su mayoría, han abordado el problema con el objetivo de generar un modelo de gran precisión en ambientes controlados. De los trabajos existentes para la reconstrucción de objetos por robots móviles, la mayoría no contempla la navegación, y quienes lo hacen, requieren un modelo o mapa previo del objeto.

En esta tesis se ha presentado un algoritmo de planificación de vistas para reconstrucción tridimensional de objetos. La planificación es hecha de forma iterativa hasta cumplir con el criterio de paro. Cada siguiente mejor vista es determinada a partir de un conjunto discreto de vistas que rodean al objeto.

Las contribuciones de la tesis fueron una función de utilidad y dos estrategias de búsqueda. La función de utilidad evalúa que tan deseable es una vista candidata. Las estrategias de búsqueda permiten obtener soluciones en tiempos aceptables para la aplicación (en el experimento donde se reconstruyó un florero de talavera el tempo de cómputo fue de 4.57 s en promedio).

La función de utilidad, a diferencia de trabajos anteriores en reconstrucción de objetos por métodos volumétricos, incorpora tres criterios que consideramos muy importantes en una reconstrucción: el traslape entre tomas, la calidad de las tomas y la distancia de navegación.

Una estrategia de búsqueda es la estrategia de jerárquica recursiva. Dicha estrategia hace uso de una discretización jerárquica de la esfera de vistas. La otra estrategia es la estrategia multiresolución que determina la SMV a través de una serie de etapas, en dichas etapas se va aumentando la resolución de rayos trazados mientras se descartan las vistas peor evaluadas. En los experimentos la mejor estrategia fue la estrategia multiresolución, ya que fue capaz de reconstruir en tiempos aceptables todos los objetos probados.

El planificador propuesto fue probado en dos fases. En la primera fase se experimentó con objetos sintéticos y un sensor simulado, obteniéndose buenos resultados en cuanto a la calidad del modelo, reducción de la distancia viajada y tiempo de cómputo. En la segunda fase se utilizaron mediciones de un sensor físico, con lo que se logró la reconstrucción de un objeto real.

7.1. Conclusiones

El objetivo de desarrollar un planificador de vistas para reconstrucción tridimensional de objetos fue alcanzado y se llegó a las siguientes conclusiones:

- Nuestro planificador de vistas para reconstrucción tridimensional de objetos es capaz de reconstruir objetos de diferentes complejidades mientras se tengan los parámetros de cantidad de vistas candidatas y criterio de paro configurados adecuadamente. Las vistas planificadas proveen traslape facilitando el registro, reducen en la mayoría de los casos la distancia de navegación y mejorar la calidad de las tomas. Además, el algoritmo de planificación es generalizable, requiere conocimiento *a priori* mínimo del objeto, es indepen-

diente de la forma del objeto y es capaz de terminar de forma automática la reconstrucción.

- En nuestro planificador la resolución del modelo, la cantidad de vistas candidatas, el criterio de paro, el traslape y el costo de navegación son parámetros configurables, lo que hace al planificador adaptable a diversas aplicaciones no sólo en robótica móvil.

- El incorporar un criterio de navegación a la función de utilidad permite reducir en la mayoría de los casos la distancia total de navegación. En el mejor de los casos la reducción es hasta del 33 % y en promedio es de 13 % respecto a cuando no se usa. Lo anterior nos permite decir que es mejor incluir el factor que excluirlo en la determinación de una vista, ya que es muy probable que la distancia de navegación se vea reducida.

- Con los experimentos realizados podemos decir que la aplicación del planificador en robots móviles es asequible en cuanto a tiempo de ejecución. En dichos experimentos, el tiempo de cómputo para nuestra mejor estrategia fue menor que 118 segundos, aún con el uso de una resolución de 66,430,125 voxels en el mapa, considerada con alta resolución, y 320 vistas candidatas. Si comparamos este tiempo con la navegación y la adquisición de imágenes de rango podemos decir que el tiempo es adecuado (el tiempo de navegación del robot Markovito [Aviles-Arriaga et al, 2009] está en el orden de minutos).

7.2. Trabajo Futuro

Esta tesis plantea dos tipos de trabajos futuros. El primero contempla mejoras y pruebas al planificador. El segundo contempla los tareas que deben ser resueltas para lograr una automatización completa del proceso de reconstrucción.

La mejoras y pruebas sugeridas para el planificador son:

- Incorporación de una etapa de registro de la información. El objetivo de la etapa es fusionar las diferentes nubes de puntos en un único modelo. Nuestra sugerencia es utilizar el algoritmo *Iterative Closest Point* [Zhang, 1994].

- Mejora en la implementación del planificador. La implementación actual del planificador ocupa, en altas resoluciones, alrededor de 1GB de memoria RAM. Con una mejor implementación este consumo de memoria se podría reducir. Por otra parte la implementación de la estrategia multiresolución puede ser mejorada. La sugerencia es conservar las evaluaciones en cada etapa para evitar cálculos repetidos.

- Sustituir la función de distancia ortodrómica con una distancia normalizada sobre el espacio de configuraciones del robot. Debido a que, el robot no necesariamente sigue una trayectoria sobre la superficie de una esfera para alcanzar un punto.

- Probar el planificador utilizando un sensor de rango de alta resolución. En esta prueba sugerimos utilizar el repositorio de imágenes de rango de la universidad de Stanford [Stanford University, 2009].

- Incorporar información 3D del ambiente para restringir las vistas candidatas. En nuestro planificador suponemos que afuera del cubo encapsulador del objeto hay espacio vacío. Sería interesante incorporar información 3D del ambiente a ése espacio para evitar colisiones del robot o del sensor.

Los tareas que deben ser resueltas para lograr una automatización completa del proceso de reconstrucción son:

- Desarrollar o implementar un algoritmo de navegación que sea eficaz en el posicionamiento y orientación, de tal forma que el error de posicionamiento sea menor que la resolución del voxel.

- Desarrollar un algoritmo de segmentación que halle los bordes de oclusión del objeto a reconstruir. El algoritmo debe ser compatible con las suposiciones y restricciones de nuestro planificador.

- Incorporar al planificador los parámetros de configuración del sensor.

Apéndice A

Datos de reconstrucción

En este anexo se muestran los datos de cada reconstrucción del experimento 5.4 del capítulo 5. En la tabla A.1 se muestran los resultados de cada reconstrucción utilizando el enfoque CF. En la tabla A.2 se muestran los datos de cada reconstrucción utilizando el enfoque SVPMAD.

Inicio	# vistas	Vxl. ocup.	Calidad	Dist. (m)
1	9	7859	0.749	24.92
2	8	7791	0.730	25.68
3	8	7787	0.694	26.60
4	8	7784	0.727	24.70
5	8	7751	0.715	26.95
6	7	7647	0.701	20.36
7	9	7882	0.750	29.73
8	8	7796	0.724	27.39
9	8	7816	0.729	27.37
10	8	7767	0.713	27.45
11	10	7935	0.757	25.47
12	8	7829	0.724	26.47
13	9	7864	0.756	29.43
14	9	7869	0.744	30.4
Continúa en la siguiente página...				

15	8	7742	0.717	23.10
16	9	7839	0.736	30.60
17	10	7921	0.754	28.54
18	10	7893	0.768	29.47
19	9	7825	0.754	30.63
20	9	7844	0.744	30.93

Tabla A.1: Resultados del enfoque CF.

Inicio	# vistas	Vxl. ocup.	Calidad	Dist.(m)
1	8	7779	0.717	29.18
2	9	7851	0.731	33.94
3	8	7782	0.724	28.25
4	9	7902	0.738	33.69
5	10	7880	0.735	35.30
6	9	7830	0.729	32.10
7	10	7907	0.736	34.75
8	8	7758	0.675	28.19
9	8	7879	0.710	26.99
10	8	7802	0.723	29.75
11	7	7740	0.712	24.56
12	9	7902	0.737	32.85
13	9	7845	0.728	31.42
14	8	7771	0.677	23.74
15	7	7721	0.695	29.34
16	8	7771	0.729	31.96
17	9	7786	0.713	31.53
18	9	7812	0.727	30.84
19	7	7688	0.713	27.87
20	8	7786	0.722	27.80

Tabla A.2: Resultados del enfoque SVPMAD.

Bibliografía

[Aviles-Arriaga et al, 2009] Aviles-Arriaga H., Sucar L., Morales E., Vargas B., et Corona E. (2009). Markovito: A Flexible and General Service Robot. En Springer Berlin / Heidelberg editores, *Design and Control of Intelligent Robotic Systems*, volume 177/2009 of *Studies in Computational Intelligence*, pp. 401–423. Springer.

[Banta et Abidi, 1996] Banta J. et Abidi M. (1996). Autonomous placement of a range sensor for acquisition of optimal 3-D models. En *Proceedings of the IEEE 22nd International Conference on Industrial Electronics, Control and Instrumentation (Taipei, Taiwan)*, volume 3, pp. 1583 – 1588.

[Banta et al, 2000] Banta J. E., Wong L. M., Dumont C., et Abidi M. A. (2000). A Next-Best-View System for Autonomous 3-D Object Reconstruction. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 30:589–598.

[Banta et al, 1995] Banta J. E., Zhien Y., Wang X. Z., Zhang G., Smith M. T., et Abidi M. A. (1995). A best-next-view algorithm for three-dimensional scene reconstruction using range images. En *XIV session of intel systems and advanced manufacturing symposium, SPIE*, pp. 418–29.

[Blaer et Allen, 2006a] Blaer P. et Allen P. (2006a). Two Stage View Planning for Large-Scale Site Modeling. En *Third International Symposium on 3D Data Processing, Visualization, and Transmission (3DPVT'06)*, pp. 814–821.

[Blaer et Allen, 2006b] Blaer P. et Allen P. (2006b). View Planning for Automated Site Modeling. En *Proceedings of the IEEE International Conference on Robotics and Automation, 2006. ICRA 2006 (Orlando, FL)*, pp. 2621–2626.

[Blaer et Allen, 2007] Blaer P. et Allen P. (2007). Data acquisition and view planning for 3-D modeling tasks. En *IEEE/RSJ International Conference on Intelligent Robots and Systems, 2007. IROS 2007 (San Diego, CA.)*, pp. 417–422.

[Bresenham, 1965] Bresenham J. E. (1965). Algorithm for computer control of a digital plotter. *IBM Systems Journal*, 4:25–30.

[Chen et al, 2008] Chen S., Li Y., Zhang J., et Wang W. (2008). *Active Sensor Planning for Multiview Vision Tasks*. Springer-Verlag.

[Connolly, 1985] Connolly C. (1985). The determination of next best views. En *IEEE International Conference on Robotics and Automation 1985, ICRA85*, volume 2, pp. 432–435.

[García et Velázquez, 1998] García M. A. et Velázquez S. (1998). A two-stage algorithm for planning the next view from range images. En *Procceedings of the 6th British Machine Vision Conference*, pp. 720–729.

[Jain et al, 1995] Jain R., Kasturi R., et Schunck B. G. (1995). *Machine vision*. McGraw-Hill Inc., New York, NY, USA.

[Konolige, 1997] Konolige K. (1997). Small Vision Systems: Hardware and Implementation. En *Proceedings of the Eighth International Symposium on Robotics Research (Hayama, Japan)*, pp. 203–212.

[Konolige et Beymer, 2007] Konolige K. et Beymer D. (2007). *SRI Small Vision System User's Manual*. SRI International.

[Levoy et al, 2000] Levoy M., Pulli K., Curless B., Rusinkiewicz S., Koller D., Pereira L., Ginzton M., Anderson S., Davis J., Ginsberg J., Shade J., et Fulk D. (2000). The digital michelangelo project: 3D scanning of large statues. En *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pp. 131–144.

[Lorensen et Cline, 1987] Lorensen W. E. et Cline H. E. (1987). Marching cubes: A high resolution 3d surface construction algorithm. *ACM SIGGRAPH (Special Interest Group on Computer Graphics and Interactive Techniques) Computer Graphics*, 21(4):163–169.

[Lozano et al, 2002] Lozano M. T., Devy M., et Sanchiz J. M. (2002). Perception Planning for an Exploration Task of a 3d Environment. En *Proceedings of the 16 th international conference on pattern recognition, ICPR'02*, volume 3, pp. 704– 707.

[Massios et Fisher, 1998] Massios N. A. et Fisher R. B. (1998). A Best Next View Selection Algorithm Incorporating a Quality Criterion. En *British Machine Vision Conference, BMVC98*, pp. 780–789.

[Maver et Bajcsy, 1993] Maver J. et Bajcsy R. (1993). Occlusions as a guide for planning the next view. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:417–433.

[Mei et al, 2005] Mei Y., Lu Y.-H., Hu Y., et Lee C. (2005). A case study of mobile robot's energy consumption and conservation techniques. En *Proceedings of 12th International Conference on Advanced Robotics, 2005. ICAR '05.*, pp. 492–497.

[Null et Sinzinger, 2006] Null B. et Sinzinger E. (2006). Next best view algorithms for interior and exterior model acquisition. En Springer Berlin / Heidelberg

editores, *Advances in Visual Computing*, volume 4292/2006 of *Lecture Notes in Computer Science*, pp. 668–677. Springer.

[Papadopoulos-Orfanos et Schmitt, 1997] Papadopoulos-Orfanos D. et Schmitt F. (1997). Automatic 3-d digitization using a laser rangefinder with a small field of view. En *Proceeding of the International Conference on Recent Advances in 3-D Digital Imaging and Modeling 1997 (Ottawa, Ont., Canada)*, pp. 60–67.

[Sanchiz et Fisher, 1999] Sanchiz J. et Fisher R. (1999). A next-best-view algorithm for 3d scene recovery with 5 degrees of freedom. En *British Machine Vision Conference 1999 (Nottingham, UK)*, pp. 163–172.

[Scott et al, 2003] Scott W. R., Roth G., et Rivest J.-F. (2003). View planning for automated three-dimensional object reconstruction and inspection. *ACM Computing Surveys (CSUR)*, 35(1):64–96.

[Shirley, 2005] Shirley P. (2005). *Fundamentals of computer graphics*. A K Peters, Ltd., second edition.

[Stanford University, 2009] Stanford University (2009). The Stanford 3D Scanning Repository. http://www-graphics.stanford.edu/data/3Dscanrep/. Consultado en junio de 2009.

[Stein et Hebert, 2007] Stein A. et Hebert M. (2007). Combining local appearance and motion cues for occlusion boundary detection. En *British machine vision conference 2007 (bmvc07)*.

[Stiene et al, 2006] Stiene S., Lingemann K., Nuchter A., et Hertzberg J. (2006). Contour-Based Object Detection in Range Images. En *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pp. 168–175.

[Videre Design, 2004] Videre Design (2004). *STH-MDCS-VAR/C Stereo Head User's Manual*.

[Wang et al, 2007] Wang P., Krishnamurti R., et Gupta K. (2007). View Planning Problem with Combined View and Traveling Cost. En *IEEE International Conference on Robotics and Automation 2007 ICRA07 (Roma, Italy)*, pp. 711–716.

[Wong et al, 1999] Wong L. M., Dumont C., et Abidi M. A. (1999). Next Best View system in a 3-d Object Modeling Task. En *Proceedings of EEE International Symposium on Computational Intelligence in Robotics and Automation 1999. CIRA99.*, pp. 306–311.

[Zhang et al, 1999] Zhang R., Tsai P.-S., Cryer J. E., et Shah M. (1999). Shape from shading: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(8):690–706.

[Zhang, 1994] Zhang Z. (1994). Iterative point matching for registration of free-form curves and surfaces. *International Journal of Computer Vision*, 13(2):119–152.