UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE INGENIERÍA ELÉCTRICA

BÚSQUEDA DE OBJETOS MEDIANTE CONOCIMIENTO SEMÁNTICO EN AMBIENTES INTERIORES

TESIS PARA OPTAR AL GRADO DE MAGISTER EN CIENCIAS DE LA INGENIERIA MENCION ELECTRICA

MARCELO SAAVEDRA ALCOBA

PROFESOR GUÍA:
PhD. JAVIER RUIZ DEL SOLAR SAN MARTÍN

PROFESOR CO-GUÍA:
PhD. PATRICIO LONCOMILLA ZAMBRANA

MIEMBROS DE LA COMISIÓN:
PhD. ALVARO SOTO ARRIAZA
PhD. JORGE SILVA SANCHEZ

SANTIAGO DE CHILE
2015

RESUMEN DE LA TESIS
PARA OPTAR AL GRADO DE
MAGISTER EN INGENIERÍA ELÉCTRICA
POR: MARCELO SAAVEDRA ALCOBA
FECHA: AGOSTO 2015
PROF. GUÍA: JAVIER RUIZ DEL SOLAR

Resumen

En este trabajo se propone un nuevo enfoque probabilístico para la búsqueda informada de objetos mediante robots. La metodología está basada en un marco bayesiano que usa convoluciones entre verosimilitudes de observación y máscaras de relación espacial para estimar el mapa de probabilidad de encontrar el objeto buscado. Por medio del uso de máscaras de relación espacial, relaciones espaciales complejas entre los objetos pueden ser definidas como sumas ponderadas de las relaciones espaciales básicas utilizando matrices de coocurrencia como pesos.

La estrategia de búsqueda considera dos métodos distintos para seleccionar la ruta por donde debe navegar el robot que son el cálculo del campo de visión óptimo que cubre la mayor probabilidad de encontrar el objeto buscado, y el cálculo de un máximo global de la utilidad esperada sobre las regiones creadas sobre el mapa. Ambos métodos permiten obtener una nueva pose óptima a partir de la pose actual y la información del mapa de probabilidad del objeto buscado.

El algoritmo de búsqueda es validado en un ambiente de oficina con cuatro clases de objetos ("monitor", "teclado", "cpu", y "router") y se consideraron cuatro relaciones espaciales básicas (“muy cerca”, “cerca”, “lejos”, “muy lejos”). Los experimentos combinan estadísticas sobre la coocurrencia de objetos y sobre la detección en ambientes reales, y las pruebas de búsqueda usan una herramienta de simulación realista en el que varias pruebas comparan once algoritmos de búsqueda de objetos.

Las simulaciones se realizaron utilizando diferentes *escalas de vista* en la detección de objetos; es decir que se verifica como varia la detección del sensor (cámara) en diferentes rangos de distancias: 1.0 , 1.25 , 1.50 , 1.75 , 2.0 , 2.25 y 2.50 [mt], esto con el fin de analizar el efecto de la calidad de la detección (tanto la calidad de una cámara como la del algoritmo de detección) en la *tasa de detección*.

Agradecimientos

Agradezco primero a Dios sin cuyo infinito poder no estaríamos acá, por sus bendiciones en todos los momentos de mi vida, a Él se lo debo todo.

Agradezco también al hogar donde Dios me envió, a toda mi familia, en especial a mis padres que supieron inculcarme el deseo de triunfar y a mi querida hermana que siempre supo alentarme en todo momento.

A todos los buenos amigos que hice en Chile, su amistad fue tan valiosa en momentos difíciles para mí que pareció que los hubiera conocido desde siempre.

A todos mis amigos y compañeros del laboratorio de Visión Computacional y Robótica, por todos los momentos gratos compartidos, por esas tardes interminables de ajedrez y por no dudar ni un solo momento en ayudarme cuando lo necesité.

A los profesores del Departamento de Ingeniería Eléctrica de la Universidad de Chile por haber compartido sus conocimientos conmigo.

A los revisores de la presente tesis por sus importantes observaciones y correcciones hechas sin cuyas diferentes visiones mi proceso de aprendizaje y el documento final no habrían alcanzado el nivel logrado.

Al profesor Javier Ruiz del Solar, por darme la oportunidad de haber estado en el programa y por toda su paciencia en mi estadía en la universidad.

A la Universidad de Chile, por haberme permitido beber en sus manantiales de conocimiento y donde pude desarrollarme como persona.

Al gobierno y al pueblo chileno, estaré siempre agradecido porque en todo este tiempo compartí gratos momentos que no olvidaré.

Para terminar voy a prestarme las siguientes palabras:
"Si alguna vez cuentan mi historia, que cuenten que caminé entre gigantes…."
(Ulises en la película Troya, 2004)

Tabla de Contenido

Índice de Tablas

Índice de Figuras

Capítulo 1. INTRODUCCIÓN

1.1. Fundamentación General

La interacción de la robótica con el ser humano evoluciona progresivamente en la actualidad, específicamente en el campo de "robótica de servicio", como por ejemplo los sistemas de atención a los ancianos, minusválidos y otros.

Una de las tareas que surgen en la interacción entre robots y seres humanos es el trabajo de "buscar y traer objetos". Por esta razón, "buscar" es una de las tareas más básicas que se espera de un robot. Los problemas sobre búsqueda de rutas óptimas han sido problemas muy estudiados en el área de la Inteligencia Artificial (IA) [1], pero la búsqueda robótica de objetos en un ambiente real es un trabajo más costoso, ya que los objetos no siempre estarán al alcance del robot y pueden estar ocluidos por otros objetos.

El realizar una búsqueda en un ambiente real implica identificar y elegir un conjunto de acciones apropiadas de un conjunto muy grande de posibilidades. En contraste con los problemas clásicos de búsqueda en la IA, en la búsqueda robótica se desea no volver a lugares ya visitados [2]. Para este propósito los robots deben poseer la capacidad de percibir y representar el mundo de una forma aproximada como lo hacen los seres humanos; y dado que los seres humanos perciben el mundo sobre la base de la información visual, también los robots deben aprovechar esta información visual del ambiente [3].

Para que un robot pueda interactuar con el mundo, necesita tener una representación espacial estructurada del medio ambiente. Estas representaciones se realizan mediante mapas. La mayoría de los trabajos realizados respecto a esta temática, consideran representaciones como mapas métricos [4]**,** mapas topológicos [5] o mapas basados en apariencia [6]**.** Estos tipos de mapas resultan de mucha utilidad en la navegación y localización de robots, pero no contienen información de relevancia para la búsqueda de objetos, ni para planificación de tareas necesarias que permitan realizar una interacción entre robots y humanos. Hoy en día las investigaciones en robótica móvil proponen añadir capas de información semántica sobre los mapas, es decir, añadir información de "Alto Nivel" como es la información del lugar donde se encuentra un robot (ej. salas, pasillos, dormitorios y otros) o la información de los objetos detectados dentro un ambiente (ej. cama, televisor y otros) [7] [8] [9] [10].

Dentro de esta estructura de interpretación de "Alto Nivel", es decir, de la interacción natural que existe entre robots y seres humanos, existen retos por resolver. Como ejemplo se puede mencionar el hecho de dar una orden como "traer la taza de café". Esta tarea puede parecer sencilla para un ser humano, pero para un robot no deja de ser una tarea complicada, dado que en el mundo real no será suficiente un algoritmo reconocedor de objetos sino también percibir el

mundo como lo perciben los seres humanos [3] [7]. Este entendimiento podría ser logrado mediante el uso de una interpretación semántica y el análisis de contexto del ambiente donde se busca el *objeto X*, ya que desde un punto de vista lógico, se sabe que el *objeto X*, en este caso la *taza de café*, no puede estar flotando en el espacio y existe poca probabilidad de que pueda estar en el piso, sin embargo si se detectan objetos intermedios como una *caja de leche*, puede existir más probabilidad que la *taza de café* esté cerca; es decir el *objeto X* en este caso debería estar sobre una mesa, o en un espacio específico donde las personas se sirven el café.

En este trabajo, se propone una nueva metodología para la búsqueda informada de objetos mediante un robot. La metodología para la estrategia de búsqueda considera dos métodos para seleccionar la ruta donde debe navegar el robot, que son el *Análisis del campo de visión óptimo* y el *Análisis del máximo global de la utilidad esperada*. Ambas metodologías permiten obtener una nueva pose óptima a partir de la pose actual y del mapa de probabilidad donde puede encontrarse el objeto de búsqueda, el cual se actualiza mediante detecciones positivas y negativas del objeto primario (objeto de búsqueda)y de los objetos secundarios (objetos que tienen alguna relación espacial con el objeto buscado).

1.2. Planteamiento del Problema

La situación más común al buscar un objeto en un ambiente interior es que el objeto pueda estar fuera del rango de visión del robot. Una búsqueda directa explorando todo el ambiente bastaría para encontrar el objeto, sin embargo se está suponiendo que existe la misma probabilidad de encontrar el objeto en cualquier parte del ambiente. En un mundo real las personas hacen que un ambiente sea dinámico, es decir que los objetos no siempre están en la misma posición o a un fácil alcance visual, pero si en muchos casos manteniendo un orden. Este orden entre objetos puede ser representado mediante una relación espacial tal como la distancia entre objetos. Por ejemplo, en un ambiente interior como una oficina muchos objetos cambiarán de posición frecuentemente por el uso repetitivo de dichos objetos pero se mantendrá cierto orden. Un efecto muy común que puede ocurrir al manipular objetos es la oclusión (ver Figura 1).

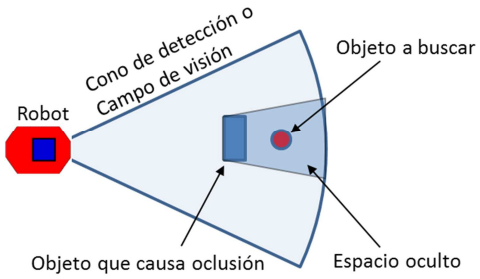


Figura 1. Situación de oclusión de objetos.

En la Figura 1, se puede observar que la cámara perteneciente al robot tiene un rango de detección representado mediante un **campo de visión** o *FOV* (del ing. *field of view* que significa campo de visión). En [11], se menciona que para que un robot realice una búsqueda planificada

necesita resolver dos sub-tareas: ¿dónde se realizará la siguiente vista? y ¿dónde se realizará el siguiente movimiento?. Para la segunda sub-tarea se debe cumplir dos requisitos, el primero es que exista un espacio accesible en el ambiente y la segunda es que exista una alta probabilidad de encontrar el objeto. El “no planificar” estas dos tareas, implica examinar todas las posibles configuraciones de los movimientos y vistas del robot hasta encontrar el objeto buscado, esta estrategia resulta ineficiente por varias razones. Desde esta perspectiva, se puede apreciar que el espacio de búsqueda de objetos es enorme, y que la información incompleta de un ambiente podría hacer que la búsqueda de objetos demore mucho tiempo o que simplemente no se encuentre el objeto buscado.

Es en esta parte donde la información del contexto puede jugar un papel importante a la hora de buscar objetos, ya que los demás objetos detectados pueden tener una relación espacial muy cercana al *objeto X* (siguiendo el ejemplo de buscar una “taza de café”, los objetos con una mayor relación espacial puede ser: caja de leche, caja de café, hervidor, cafetera, vasos y otros objetos similares). Los objetos creados y utilizados en un entorno por los seres humanos no se encuentran en lugares al azar. Por el contrario, las personas diseñan y organizan espacios de manera que sirva para varios propósitos funcionales. Esta organización puede ser expresada en términos de relaciones espaciales. Las relaciones espaciales son abstracciones de la configuración en el espacio de los objetos, tales como las distancias, direcciones o relaciones topológicas. Estas estructuras ayudan a los seres humanos a recordar aspectos de su entorno, y son también de gran utilidad cuando un robot tiene que buscar objetos en ese mismo ambiente [12].

Por ejemplo, dentro de una oficina, se puede ver que el objeto "*teclado*" está a menudo muy cerca del objeto "*monitor*". Para un ser humano, la capacidad de deducir que el objeto *A* tiende a ser "cerca", "muy cerca" o "lejos" de un objeto *B* puede ser una tarea trivial, ya que los seres humanos pueden aprender las relaciones espaciales entre los objetos mediante la observación de un gran número de configuraciones parecidas. Este trabajo se centra en brindar a un robot la capacidad de encontrar objetos mediante la relación espacial semántica de “cercanía” en un ambiente definido. Para representar este conocimiento de cercanía, se interactúa con cuatro clases de objetos ("*monitor*", "*teclado*", "*cpu*", y "*router*") y se considera cuatro relaciones espaciales básicas ("muy cerca", "cerca", "lejos" y "muy lejos"). Los experimentos combinan estadísticas acerca de la coocurrencia de la relación espacial entre objetos y la detección de objetos en entornos reales.

1.3. Objetivos

a) Objetivos Generales

- Proponer una nueva metodología de búsqueda probabilística de objetos conocidos, para robots móviles en ambientes interiores desconocidos usando información semántica y de contexto.

- Implementar la metodología propuesta de búsqueda en diferentes situaciones y evaluar su desempeño.

b) Objetivos Específicos

- Diseñar una representación semántica de ambientes interiores que utilice la información del recorrido del robot y la posición de los objetos detectados.
- Diseñar una técnica que permita obtener la relación espacial entre dos objetos.
- Diseñar una técnica para inferir la posición donde pueda encontrarse el objeto a buscar a partir de las detecciones de otros objetos relacionados
- Diseñar una técnica para la exploración de espacios ocluidos por otros objetos en un ambiente interior.

1.4. Hipótesis

Las hipótesis desarrolladas para este trabajo son las siguientes:

- Una técnica de búsqueda probabilística que utilice la información semántica y la relación espacial entre objetos es capaz de mejorar el proceso de búsqueda de los mismos objetos.
- La coocurrencia entre objetos permite construir mapas semánticos y obtener la información de la relación espacial de los objetos de interés no detectados.

1.5. Originalidad y Aporte de la investigación

Las contribuciones identificadas en la siguiente propuesta de tesis son dos, los que se detallan a continuación.

La primera contribución, es un estudio para la representación del mundo y sus relaciones espaciales, particularmente de lo que ocurre en un ambiente interior, donde se tienen distintas situaciones de la relación espacial entre objetos, que pueden ser representadas mediante variables lingüísticas de distancia y divididas en cuatro casos que son: “muy cerca”, “cerca” ,“lejos” y “muy lejos”. Estas representaciones de las relaciones espaciales son almacenadas mediante matrices de coocurrencia. Para cada caso de relación espacial, se obtiene la frecuencia en que un objeto pueda estar “muy cerca”, “cerca” o “lejos” de otros objetos. De estas matrices, se puede obtener la probabilidad de la cercanía de cada objeto respecto a otro.

La segunda contribución es el desarrollo de la búsqueda indirecta que considera dos métodos para seleccionar la ruta por donde debe navegar el robot. La primera es el cálculo del campo de visión óptimo que tenga la mayor probabilidad de encontrar el objeto de búsqueda, y el segundo método es el cálculo del máximo global de la utilidad esperada sobre regiones creadas sobre el mapa. Ambas metodologías conjuntas permiten obtener una nueva pose óptima a partir de la pose actual y del mapa de probabilidad del objeto primario, el cual se actualiza mediante observaciones positivas y negativas del objeto primario y de los objetos secundarios.

Capítulo 2. TRABAJOS RELACIONADOS

2.1. Representación del mundo mediante mapas

La representación del mundo es probablemente uno de los temas más importantes abordados en la literatura asociada a la robótica móvil. Su importancia fundamental se deriva de la necesidad de contar con un modelo adecuado del medio ambiente para que un robot pueda planificar, razonar, y ejecutar algunas tareas [13]. El uso de modelos del mundo, que incluyen capacidades de inferencia y conocimiento de tipo semántico se remonta a los días del Shakey [14], que fue un robot construido en 1970 en el Stanford Research Institute (SRI). Unos años después dentro del mismo instituto, se crea el sucesor de Shakey, el robot Flakey. Este último utiliza la lógica epistémica de razonar acerca de las creencias, deseos e intenciones de los robots y usuarios [15]. En la Figura 2 se puede ver la estructura física que tenían los robots Shakey y Flakey.

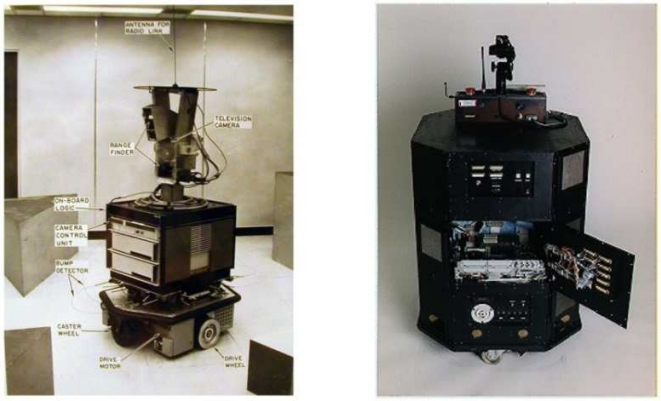


Figura 2. Robots Shakey (izq.) y Flakey de SRI (der.).

El objetivo de crear una representación del ambiente (mapa), es que el robot pueda tener un conocimiento de su espacio de operación. Con la información del mapa el robot podrá reconocer la posición en la que se encuentra, saber qué lugares fueron explorados y planear movimientos a lugares donde falta explorar. El tema de la representación del ambiente fue bastante estudiado durante los últimos años. Inicialmente los investigadores daban al robot una representación del ambiente realizada de manera fuera de línea. Luego apareció el mapeo y localización simultánea, mejor conocido en la literatura como *Simultaneous Localization and Mapping* (SLAM), donde se obtienen mapas probabilísticos para representar el ambiente.

Sin embargo, las representaciones del entorno manejadas por mapas métricos, topológicos y basados en apariencia, están orientados sólo a la navegación robótica y no son suficientes para realizar acciones como búsqueda, seguimiento e interacción con personas. Para hacer frente a la carencia de este tipo de representaciones, se han planteado distintos métodos, tales como las representaciones jerárquicas del espacio, extracción de características de alto nivel,

interpretaciones de la escena, mapas cognitivos y la interacción humano robot (que por sus siglas en inglés se abrevia como HRI) [16].

Martinelli en [17] propone una jerarquía sobre los mapas para modelar el ambiente como se puede ver en la Figura 3, donde en cada escala superior en la jerarquía se reduce la información geométrica y se aumenta una representación distintiva.

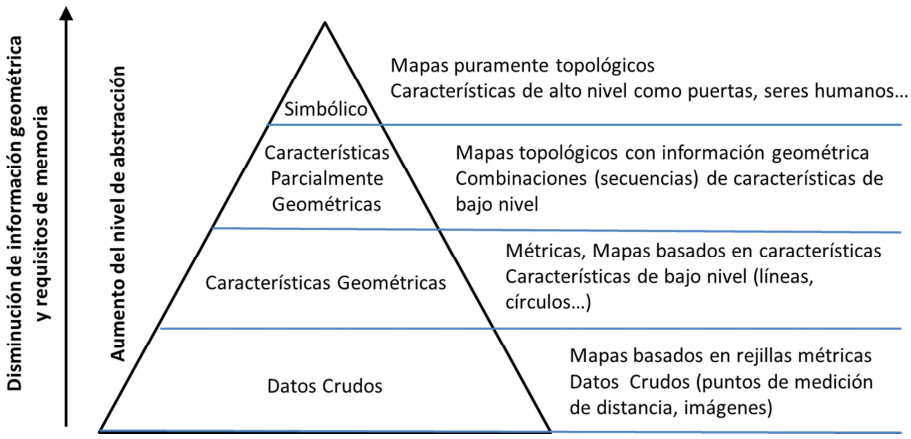


Figura 3. Modelo de jerarquía propuesta por Martinelli en [17].

En [18], Kuipers presentó una jerarquía espacial semántica con modelos espaciales en capas que comprenden datos sensorio-motores basados en puntos de vista, relaciones de lugar y la información métrica. Por otra parte, el modelo *route graph* es introducido por Krieg-Brückner et al. en [19]. Ambas teorías proponen una representación cognitiva inspirada en varias capas del mapa de manera similar, que es al mismo tiempo ventajoso para la navegación del robot.

Estos trabajos presentados utilizan en su mayoría la detección de objetos y el conocimiento semántico para identificar el lugar donde se encuentra el robot, como por ejemplo dormitorios, cocinas, oficinas, pasillos y salones. Sin embargo, otros autores consideran que aún existen cuestiones por resolver, entre ellos: cómo la información semántica puede ser provechosamente utilizada por el robot para planificar y ejecutar tareas.

En [9], Galindo *et al.*, se presenta un enfoque que contiene dos jerarquías paralelas, una espacial y una conceptual, conectados a través de anclaje. La inferencia acerca de los lugares se basa en objetos que se encuentran dentro de cada lugar como se puede ver en la Figura 4, donde los enlaces horizontales están representados por las palabras: "contiene un", y los enlaces verticales están representados por las palabras: "es un". Los enlaces verticales solo simbolizan el paso de una característica de bajo nivel a una de alto nivel de la jerarquía.

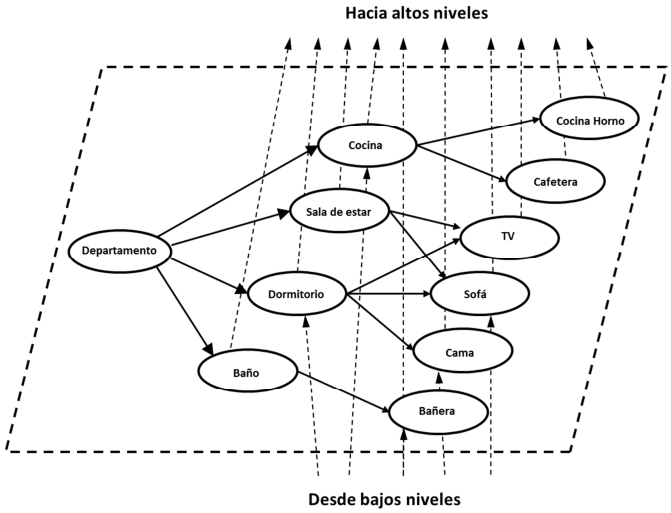


Figura 4. Modelo de jerarquía conceptual presentada en [9].

Posteriormente parte de los mismos autores proponen en [13] un propio mapa semántico integrando una jerarquía espacial de los objetos y lugares con una jerarquía semántica de los conceptos y las relaciones. Utilizando esta información, realizan dos estudios de caso, una para deducir nueva información de la estructura semántica útil para realizar planificación de tareas y la otra para mejorar la eficiencia de la planificación mediante las restricciones semánticas.

En el trabajo de Vasudevan et al. intentan crear una representación espacial en términos de objetos, mediante la codificación de objetos domésticos típicos y puertas dentro de un marco probabilístico jerárquica [16]. Utilizaron un sistema de reconocimiento de objetos basado en SIFT [20] y un sistema de detección de puertas basado en líneas extraídas de escaneos de rango. También propusieron una conceptualización de diferentes lugares, basado en los objetos que se observaron dentro de ellos.

En el trabajo de Viswanathan et al. se propone un enfoque utilizando recursos existentes como el conocimiento de sentido común como las relaciones espaciales de objetos mediante aprendizaje de máquina [21]. Ellos utilizan imágenes marcadas desde la base de datos *LabelMe*, diseñado por Russell et al. [22]. Se entrenan mediante un clasificador automático de los lugares sobre la base de la presencia de los objetos detectados para inferir la probabilidad de que existen los otros objetos, y un tipo de lugar (por ejemplo, la cocina o la oficina).

2.2. Búsqueda de objetos

Como se explicó anteriormente, la búsqueda de objetos en un ambiente real es una tarea muy compleja en el área de robótica, ya que el objeto a buscar no siempre estará visible. Como una alternativa a este problema, Garvey en [23] propuso la idea de la *búsqueda indirecta*, realizando la búsqueda de otro objeto intermedio que mantenga una relación espacial con el objeto buscado. Wixson *et al.* en [24] materializa la idea de la búsqueda indirecta, demostrado una mayor eficiencia tanto teórica como empírica. El problema con la búsqueda indirecta es que la relación

espacial entre el objeto buscado y el objeto intermedio no siempre existe. Además, la detección del objeto intermedio puede no ser más fácil que la detección del objeto buscado. De hecho, en [25] se demostró que la búsqueda de un objeto arbitrario en un espacio 3D es NP-completo. Shubina y Tsotsos proponen un algoritmo que considera el costo y efecto de diferentes acciones, utilizando varios tipos de conocimiento a priori, así como las relaciones espaciales entre objetos [11].

En [26], se resalta la importancia de la detección de objetos en robots de servicio por la fuerte relación que existe entre ambos. Los autores mencionan que un ambiente interior es dinámico y existe una incertidumbre sobre la existencia de objetos de interés porque pueden estar obstruidos por otros objetos o verse muy pequeños en la imagen de la escena. Para solucionar este problema se utiliza la información de contexto para predecir la probabilidad de existencia de objetos de interés a través de objetos ya observados. Para ello, utilizan un modelo de la relación espacial de objetos basados en redes bayesianas y un método de integración propuesto.

En [27] se presenta un método para la búsqueda y localización de objetos, este incluye un enfoque para la visión periférica y una estrategia de planificación. El mecanismo de búsqueda está basado en el reconocimiento de objetos por características SIFT, de hecho, en [3] se describe un sistema inteligente que intenta realizar el reconocimiento de objetos sólidos en un escenario realista. Este método fue probado sobre el robot *Curious George* que fue desarrollado para el *Semantic Robot Vision Contest* (SRVC) [28], que es una competencia para la búsqueda autónoma de objetos.

En el trabajo de Kollar et al., se realiza la búsqueda de un objeto conociendo el mapa del ambiente y la relación espacial existente entre todos los objetos, sin contar con la información del lugar del objeto a buscar [29]. En este trabajo se desarrolla un método probabilístico para obtener los lugares más probables donde se encuentre el objeto utilizando el contexto de objeto-objeto y objeto-escena. Este método aprovecha la coocurrencia de la existencia de varios objetos en imágenes bidimensionales, utilizando etiquetas semánticas en lugares como "cocina". Estas imágenes fueron tomadas del sitio web de *Flickr*, y no toman en consideración las distancias entre objetos. En [30], se presenta un método para utilizar el conocimiento cualitativo de las relaciones espaciales entre objetos con el fin de facilitar la búsqueda visual eficiente para esos objetos. Un modelo computacional para la relación es usado para probar una distribución de probabilidad que guíe la selección de puntos de vista de la cámara. En concreto se examina la relación espacial de "sobre" (“*on*” en inglés), en el sentido de soporte físico (los objetos en un ambiente interior están en su mayoría en una superficie horizontal). Los autores mencionan que los resultados que los resultados obtenidos refuerzan la idea de que la búsqueda indirecta es un método útil en la búsqueda visual activa. En [31] se presenta una solución para realizar la búsqueda de un objeto de destino en un entorno 3D desconocido. En este trabajo un agente genera una ruta de búsqueda sobre la base del conocimiento actual de la localización del objeto a buscar codificado por la distribución de probabilidad durante el proceso de búsqueda. Kasper et al., realizan un estudio sobre las relaciones espaciales en imágenes tridimensionales utilizando un sensor Kinect [32]. Ellos crearon una base de datos utilizando nueve configuraciones de espacio

de oficina diferentes con un total de 168 objetos en 35 clases de objetos. Luego, se encuentran las distancias entre diferentes objetos. También hicieron predicciones sobre la ubicación de los objetos no encontrados mediante la detección de sus objetos circundantes. En el trabajo de Ziegler [33] se propone una estrategia de búsqueda de objetos conocidos en ambientes desconocidos, combinando un enfoque de reconocimiento de objetos, una geometría de la escena y un enfoque de SLAM que genera dinámicamente un mapa semántico etiquetado de la habitación.

En el trabajo de Aydemir et al. [34], se analiza los trabajos anteriores en robótica donde gran parte se basa en el supuesto de que el objeto a buscar estará listo al alcance sensorial del robot. En este trabajo se desea evitar ese supuesto utilizando las relaciones espaciales y una inferencia probabilística a partir de estos. Los autores indican que el método es eficiente para la seleccionar estrategias de búsqueda dadas las probabilidades por las relaciones espaciales. Parte de estos mismos autores desarrollaron un método para la búsqueda de objetos usando relaciones espaciales explícitas entre los objetos con el fin de realizar una búsqueda visual eficiente [12]. Presentaron un modelo computacional usando varias vistas al azar para orientar la cámara del robot a los puntos donde los objetos tienen una alta probabilidad de estar contenidos, usando el término de relación espacial entre objetos "encima de", en un ambiente interior ya que los objetos están en su mayoría en superficies horizontales. Elfring et al., proponen un método de búsqueda activa de objetos que considera coocurrencias entre objetos y el costo de alcanzar los objetos intermedios [35]. Sin embargo, no toman en cuenta las relaciones espaciales entre el objeto principal y los objetos intermedios.

En el trabajo de Loncomilla et al. en [36], se presenta un marco bayesiano para una búsqueda informada usando convoluciones entre verosimilitudes de observaciones y máscaras de relación especial, presentado así una mejora al trabajo presentado en [12]. Se puede observar que la tasa de detección aumenta considerablemente usando este nuevo método.

2.3. Conclusiones de la revisión bibliográfica

La presente propuesta de tesis tiene como base la extracción de características de alto nivel como se puede apreciar en los trabajos presentados en [17] y [9], y sobre el método de búsqueda de objetos se toma como base los métodos presentados en [12], [35] y [36].

Capítulo 3. METODOLOGÍA DE BÚSQUEDA DE OBJETOS

La metodología para la realización del sistema consta de 3 etapas. La primera etapa es la de **Construcción de la matriz de coocurrencia**, realizada en forma *fuera de línea*, es decir que no está enlazado al sistema general. La segunda etapa es la **Creación y actualización del mapa de probabilidad**, **Estrategia de búsqueda** y **Creación de las máscaras de relación espacial desde coocurrencias entre objetos**. La tercera etapa es la puesta en marcha del sistema en un simulador robótico.

3.1. Descripción general del Sistema Propuesto

El Sistema propuesto tiene por objetivo mejorar el proceso de búsqueda de objetos conocidos en un ambiente interior no conocido. El diagrama del esquema general del sistema se puede ver en la Figura 5.

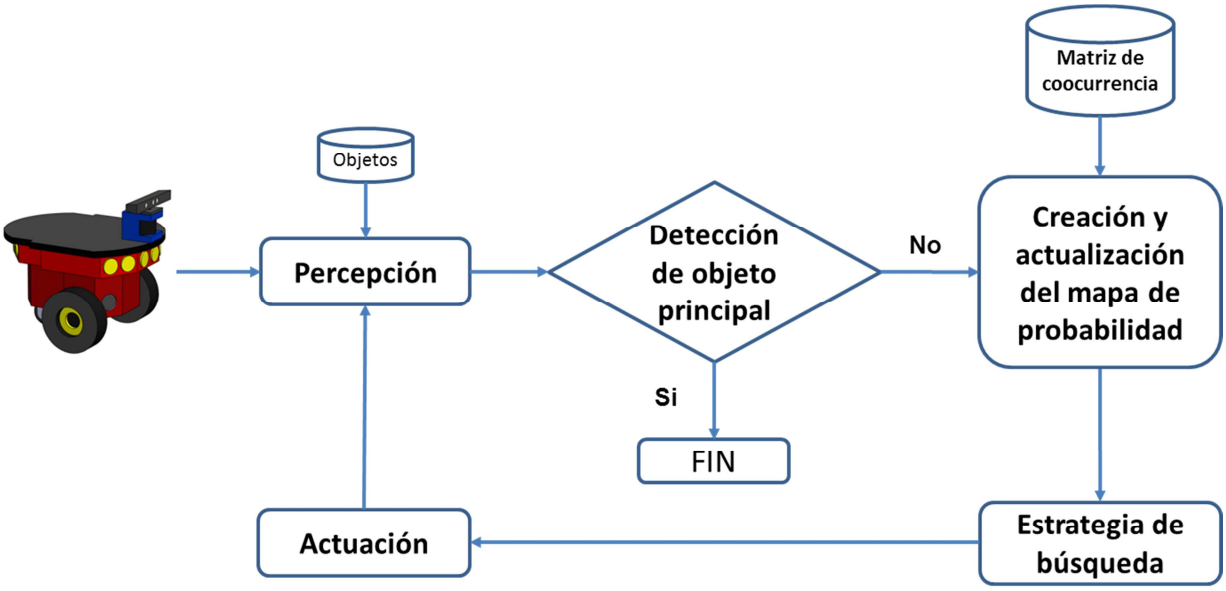


Figura 5. Esquema General Propuesto.

El proceso de búsqueda de un objeto comenzará cuando el robot reciba una orden**.** La primera acción del robot es captar las señales de los sensores (cámaras) y procesarlas, esto se realiza en el módulo de **Percepción.** En este punto se realiza una inicialización de todos los objetos que se pueden detectar. Las salidas de este módulo son 3 casos posibles que son la detección del objeto primario, la detección de uno o más objetos secundarios y la no detección. En el primer caso, si se detecta el objeto primario se procede a terminar el proceso de búsqueda. En los otros casos se pasa al siguiente módulo.

En el proceso de *creación y actualización del mapa de probabilidad*, si se detecta un objeto secundario, se realiza la convolución del mapa con la máscara asociada al objeto detectado,

aumentando así la probabilidad de encontrar el objeto buscado en el mapa. La probabilidad de encontrar al objeto buscado depende de la máscara de acuerdo a la relación espacial (*muy cerca*, *cerca*, *lejos* o *muy lejos*). En caso de no detectar ningún objeto, la probabilidad se reduce de acuerdo al campo de visión del robot, ya que es poco probable que el objeto buscado se encuentre en el área observada

Una vez que el mapa se ha actualizado, en el módulo *estrategia de búsqueda* se ejecutan 2 algoritmos escogiendo la mejor pose siguiente a donde debe llegar el robot. Luego en el módulo de *actuación* se realiza el movimiento evitando colisionar con las paredes u otros objetos presentes en el lugar. Este proceso se repite hasta encontrar el objeto buscado.

3.2. Creación y actualización del mapa de probabilidad

En el trabajo de *Aydemir* et al. en [12], las relaciones espaciales se definen como funciones de un espacio de un par de poses π_A, π_B de dos objetos *A* y *B* en un intervalo [0,1], donde "1" indica que la relación es totalmente cumplida por éstas combinaciones de poses, y "0" que la relación no se aplica en absoluto:

$$Rel_{A,B}: \{\pi_A, \pi_B\} \rightarrow \ [0, 1] \tag{3.1}$$

En este trabajo las relaciones espaciales entre dos objetos son definidas como una distribución de probabilidad de la pose π_A del primer objeto, dada una pose π_B conocida del segundo objeto.

$$Rel_{A,B}(\pi_A, \pi_B) = p(\pi_A|\pi_B) \tag{3.2}$$

Como se trata de una distribución de probabilidad, la suma sobre todas las posibles poses del objeto *A* para una pose fija del objeto *B* es igual a uno:

$$\int_{\pi_A} p(\pi_A|\pi_B) = 1 \tag{3.3}$$

Si la relación espacial es invariante a traslaciones y rotaciones, es decir, que sólo depende de la pose relativa $\pi_{A/B}$ del objeto *A* con respecto al objeto *B*, entonces la expresión de la probabilidad puede reescribirse como:

$$Rel_{A|B}(\pi_{A|B}) = p(\pi_{A|B}) \tag{3.4}$$

El robot se mueve en un espacio parametrizado en dos dimensiones mediante el uso de coordenadas (*x*,*y*). El espacio se cuantifica en celdas cuadradas con tamaño *k*, y es parametrizada mediante el uso de índices (*i*,*j*). Usando la notación de índices, una relación espacial se puede escribir como:

$$R_{A|B}(i, j) = K_{norm} * Rel_{A|B}(ki, kj) \tag{3.5}$$

$$\sum_i \sum_j R_{A|B}(i,j) = 1 \quad (3.6)$$

Donde K_{norm} es una constante de normalización.

El término $p(a_{i,j})$ representa la probabilidad de que el centro del objeto principal *A* se encuentra en la celda (*i*,*j*). Tanto las observaciones positivas como las negativas negativas z_A proporcionan información valiosa para el proceso de búsqueda de objetos, y se puede utilizar para calcular la actualización de probabilidad $p(a_{i,j}|z_A)$. La probabilidad $p(a_0)$ es tratada como un caso especial, y representa la probabilidad de que el objeto esté fuera de la región de búsqueda.

Las detecciones positivas $z_A = true$ proporcionan información acerca de los lugares en los que el objeto tiene una alta probabilidad de encontrarse, por medio de la verosimilitud $p(z_A = true|a_{i,j})$, que es definida sobre la celda (*i*, *j*). La verosimilitud tiene un alto valor sobre la celda donde se ha detectado el objeto y un valor bajo en las otras celdas. Las detecciones negativas $z_A = false$ proporcionan la información $p(z_A = false|a_{i,j})$ sobre las celdas donde los objetos tienen una baja probabilidad de encontrarse, que son las celdas visibles desde el campo de visión actual que tienen una baja probabilidad de contener el objeto.

El problema abordado en este trabajo es encontrar un objeto principal *A*, mediante movimientos apropiados del robot. Así se realiza una búsqueda que continuará hasta que el objeto principal *A* sea encontrado. En consecuencia, el proceso de búsqueda incluye sólo detecciones negativas del objeto principal *A*. Dos casos son considerados:

$$p(a_{i,j}|z_A = false) = \frac{p(z_A = false|a_{i,j})\, p(a_{i,j})}{p(a_0) + \sum_{i,j} p(z_A = false|a_{i,j})\, p(a_{i,j})} \quad (3.7)$$

$$p(a_0|z_A = false) = \frac{p(a_0)}{p(a_0) + \sum_{i,j} p(z_A = false|a_{i,j})\, p(a_{i,j})} \quad (3.8)$$

El objeto secundario *B*, puede producir detecciones z_B positivas y negativas, que pueden ser usadas para calcular una actualización de la probabilidad:

$$p(a_{i,j}|z_B) = \frac{p(z_B|a_{i,j})\, p(a_{i,j})}{p(z_B|a_0)\, p(a_0) + \sum_{i,j} p(z_B|a_{i,j})\, p(a_{i,j})} \quad (3.9)$$

$$p(a_0|z_B) = \frac{p(z_B|a_0)\, p(a_0)}{p(z_B|a_0)\, p(a_0) + \sum_{i,j} p(z_B|a_{i,j})\, p(a_{i,j})} \quad (3.10)$$

El termino $p(z_B|a_{i,j})$ y $p(z_B|a_0)$ son llamados **verosimilitudes-cruzadas**, ya que relacionan la detección de un objeto secundario *B*, con la presencia del objeto principal *A* sobre el mapa. Estas probabilidades pueden ser derivadas considerando las probabilidades $p(b_{u,v})$ para la presencia de un objeto secundario *B*, en la posición (*u*,*v*) de la grilla:

$$p(z_B|a_{i,j}) = \sum_u \sum_v p(z_B|b_{u,v})\, p(b_{u,v}|a_{i,j}) \quad (3.11)$$

$$p(z_B|a_0) = \sum_u \sum_v p(z_B|b_{u,v})\, p(b_{u,v}|a_0) \quad (3.12)$$

$$p(b_{u,v}|a_{i,j}) = R_{B|A}(u-i, j-v) \quad (3.13)$$

$$p(b_{u,v}|a_0) = \frac{1}{n_u n_v} \quad (3.14)$$

El término $p(b_{u,v}|a_0)$ es considerada una constante en (*u*,*v*) cuya suma tiene un valor igual a uno, porque supone que el objeto *B* está en el mapa. El término $p(b_{u,v}|a_{i,j})$ corresponde a la relación espacial entre el objeto principal *A* en la posición (*i*,*j*) y un objeto secundario *B* en la posición (*u*,*v*). Mediante la sustitución de este término con la relación espacial $R_{B|A}$, no hay necesidad de almacenar un mapa para el objeto secundario; sólo el mapa para el objeto principal y las verosimilitudes de las detecciones del objeto secundario son necesarios:

$$p(z_B|a_{i,j}) = \sum_u \sum_v p(z_B|b_{u,v})\, R_{B|A}(u-i, j-v) \quad (3.15)$$

$$p(z_B|a_0) = \frac{1}{n_u n_v} \sum_u \sum_v p(z_B|b_{u,v}) \quad (3.16)$$

Donde $n_u n_v$ es el tamaño del mapa. La ecuación (3.15) puede ser implementada como una convolución en el espacio (*i*,*j*) entre una imagen de la verosimilitud y la máscara $R_{B|A}(i,j)$ que describe la relación espacial entre el objeto principal y los objetos secundarios, que se llamará *máscara de relación espacial*:

$$p(z_B|a_{i,j}) = p(z_B|b_{i,j}) * R_{B|A}(i,j) \quad (3.17)$$

El sistema propuesto es muy versátil, ya que cualquier relación espacial puede ser representada mediante una máscara adecuada. Debe tenerse en cuenta que se pueden añadir objetos secundarios extra al sistema mediante la creación de máscaras de relación espacial adicionales. En caso de que estas relaciones sean encadenadas, como ejemplo de un objeto *A* está cerca de *B*, y el objeto *B* está cerca de *C*, entonces la máscara de la relación de cadena puede ser obtenida por convolución de las máscaras originales:

$$p(z_c|a_{i,j}) = p(z_c|b_{i,j}) * R_{B|A}(i,j) \quad (3.18)$$

$$p(z_c|a_{i,j}) = p(z_c|c_{i,j}) * R_{C|B}(i,j) * R_{B|A}(i,j) \quad (3.19)$$

$$\Rightarrow R_{C|A} = R_{C|B} * R_{B|A} \quad (3.20)$$

3.3. Estrategia de búsqueda

La estrategia de búsqueda considera dos métodos distintos para seleccionar la ruta por donde debe navegar el robot que son el cálculo del campo de visión óptimo que cubre la mayor probabilidad de encontrar el objeto *A* [36], y el cálculo del máximo global de la utilidad esperada sobre las regiones creadas sobre el mapa [35]. Ambos métodos permiten obtener una nueva pose óptima a partir de la pose actual y del mapa de probabilidad del objeto primario, el cual se actualiza mediante observaciones positivas y negativas del objeto primario y de los objetos secundarios.

3.3.1. Análisis del campo de visión óptimo

Este método consiste en encontrar la pose del robot cuya suma de las casillas dentro el campo de visión contenga el mayor valor posible en el mapa de probabilidad del objeto primario. Para este efecto, se generan varios campos de visión a partir de un conjunto de *k* poses aleatorias alcanzables (en este caso 200 poses) calculando que el robot no tarde más de 6 segundos en alcanzar dichas posiciones. Cada campo de visión abarca un conjunto de celdas. La suma total de la probabilidad de las celdas de un campo de visión representa la probabilidad de que el robot encuentre el objeto dentro del cono. Se selecciona la pose que contenga la máxima probabilidad de encontrar el objeto buscado en el área visible, como se ve en el trabajo de [12].

$$\arg\_max_{k=1..N} \sum_{i=1}^{n} \sum_{j=1}^{n} p(a_{i,j}) V(a_{i,j}, k) \quad (3.21)$$

Donde *N* es el número de poses candidatas planteadas, n es el ancho y alto del tamaño del mapa y $V(a_{i,j}, k)$ está definida como:

$$V(a_{i,j}, k) = \begin{cases} 1 & Si\ a_{i,j}\ esta\ dentro\ del\ punto\ de\ vista\ \ k^{ésimo} \\ 0 & En\ otro\ caso \end{cases} \quad (3.22)$$

Este proceso es un modelo óptimo porque evita repetir zonas ya exploradas. Cuando existe la detección de un objeto secundario, se realiza el proceso de convolución que aumenta la probabilidad de encontrar un objeto en un área circular dependiendo el objeto y la máscara asociada, este proceso dirige al robot a un sector que tiene mayor probabilidad de encontrar al objeto principal. El algoritmo de este proceso se puede ver en la Figura 6.

Algoritmo usando el análisis del campo de visión óptimo

1. Elegir un desplazamiento óptimo inicial d* y calcular la probabilidad óptima inicial p*
2. Repetir 200 veces:
    a. Elegir un desplazamiento d consistente en una rotación inicial, una distancia a avanzar y una rotación final. El tiempo que toma el efectuar el desplazamiento debe ser igual a 6 segundos.
    b. Calcular la probabilidad p cubierta por el cono sobre el mapa de probabilidad
    c. Si la probabilidad p es mayor que la probabilidad óptima actual p*, entonces hacer p* = p y hacer d* = d
3. Ejecutar el desplazamiento óptimo d*

Figura 6. Algoritmo usando análisis de visión optimo.

3.3.2. Análisis del máximo global por regiones

Se denomina máximo global porque se genera un análisis región a región sobre todo el mapa. El mapa de probabilidad del objeto primario es dividido en regiones que contienen un cierto número de celdas. Para cada región se calcula una *utilidad esperada* (UE), la cual depende del valor de cada celda y de la distancia entre el robot y la casilla central de la región. La región con la mayor utilidad esperada es elegida como el próximo destino a ser visitado por el robot.

Como primer paso, el mapa se divide en regiones que contengan una cierta cantidad de celdas (ver Figura 7). En el trabajo realizado, el mapa es de 100x100 celdas y cada región contiene 10x10 celdas.

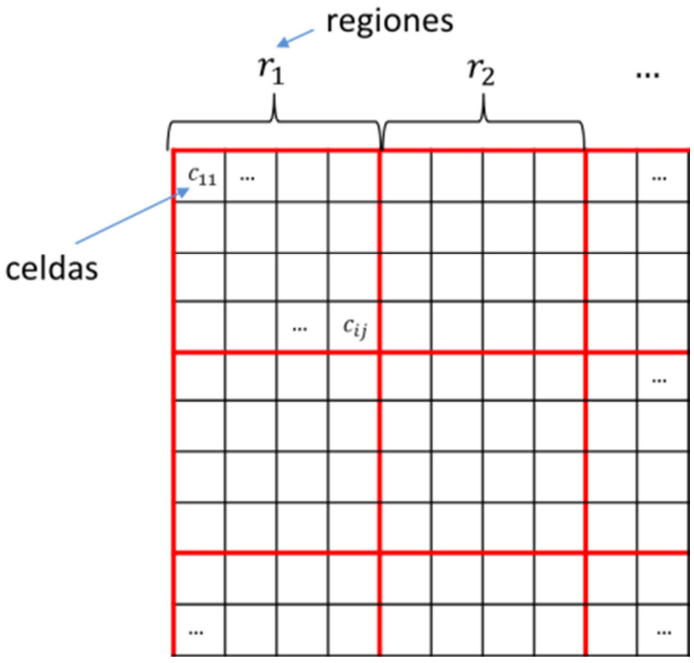


Figura 7. Creación de regiones en el mapa.

Al conjunto de todas las regiones validas se denomina *R*, al inicio de la exploración todas las regiones son válidas.

$$R = \{r_1, r_2 \dots, r_i, \dots, r_M\} \quad (3.23)$$

Donde *M* es el total de regiones existentes. Para determinar a qué región navegar se usa una función de utilidad esperada como se muestra en los trabajos de [1, 35], donde se combina la probabilidad de encontrar el objeto primario dentro de una celda con un costo de viaje:

$$UE(r_i) = [S(r_i) + w * U(r_i)] \quad (3.24)$$

$$S(r_i) = \sum_{c \in r_i} p(c) \quad (3.25)$$

$$U(r_i) = \frac{1}{arctg(d_{c-r})} \quad (3.26)$$

Donde r_i representa a cada una de las regiones, $S(r_i)$ es la suma del valor de las celdas dentro de una región, w actúa como un peso relativo y la utilidad $U(r_i)$ es definida como el recíproco del arco tangente de la distancia euclidiana d_{c-r}, que es la distancia entre el centro de la región y el robot. La suma ponderada de la ecuación (3.23) hace un balance entre la distancia y la probabilidad, evitando así un comportamiento oscilante. El valor de $UE(r_i)$ representa a la utilidad esperada de cada región. Luego se escoge la mejor región $r*$ con la mejor utilidad esperada UE^*:

$$r^* = \arg\_max_{r_i \in M}[UE(r_i)] \quad (3.27)$$

$$UE^* = UE(r^*) \quad (3.28)$$

Si UE^* es mayor a un umbral, el robot debe navegar hacia el centro de la región escogida, una vez aproximado a la región, si el objeto principal no es encontrado la región es marcada como "no valida", esto se realiza quitando la región del conjunto de regiones validas *R*.

3.3.3. Estrategia de búsqueda propuesta

La estrategia de búsqueda propuesta es una combinación basada en los métodos *análisis del campo de visión óptimo* y *análisis del máximo global por regiones*.

Las detecciones positivas y negativas de los objetos se usan para actualizar el mapa de probabilidad. Éste es dividido en regiones, es decir para cada región se calcula la suma de probabilidades de cada casilla. Después se calcula una utilidad esperada (*UE*) para cada región y luego se obtiene UE^* que es la utilidad esperada con mayor valor. Si UE^* es mayor a un umbral, se fija como destino dicha región, caso contrario el destino es seleccionado usando la ecuación (3.29).

Si el objeto principal es encontrado, finaliza la búsqueda. El resultado de este algoritmo es determinístico a diferencia del anterior que analiza poses generadas al azar (ver Figura 8).

Algoritmo de Búsqueda

1. Inicializar mapa de probabilidad del objeto primario y máscaras de convolución.
2. **Mientras** el objeto principal no es encontrado **hacer**
3. Obtener detecciones positivas y negativas para el instante actual
4. Actualizar mapa de probabilidad del objeto principal, utilizando campo de visión
5. **Para** cada objeto secundario detectado **hacer**
6. Calcular la verosimilitud cruzada de detección positiva
7. Actualizar el mapa de probabilidad de objeto principal
8. **Para** cada objeto secundario no detectado **hacer**
9. Calcular la verosimilitud cruzada de detección negativa
10. Actualizar el mapa de probabilidad de objeto principal
11. Calcular la Utilidad Esperada (UE) para las regiones válidas.
12. Calcular la mejor región r^* y la mejor utilidad UE*
13. **Si** UE* > Umbral **entonces**
14. Navegar hacia a la posición central r* evadiendo obstáculos.
15. **Si** el robot está en la mejor región y el objeto primario no es encontrado **entonces**
16. Se marca como región no válida.
17. **Si** UE* < Umbral **entonces**
18. Navegar hacia la pose con el mejor campo de visión alcanzable en 6 segundos evadiendo obstáculos
19. **Si** el objeto primario es encontrado, **entonces** terminar exploración

Figura 8. Algoritmo propuesto de búsqueda.

A continuación en la Tabla 1 y Tabla 2, se realiza un ejemplo ilustrativo para ver la diferencia del uso del uso del análisis del campo de visión óptimo y la estrategia de búsqueda propuesta.

Tabla 1. Comportamiento utilizando análisis del campo de visión óptimo

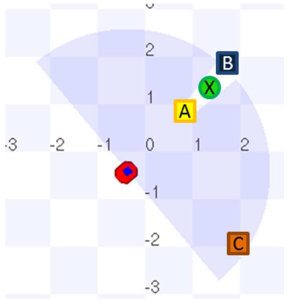
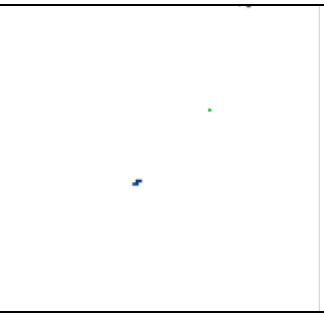
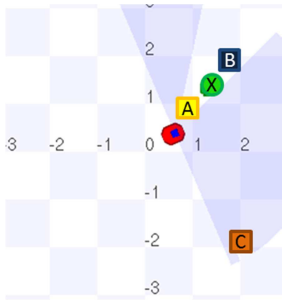
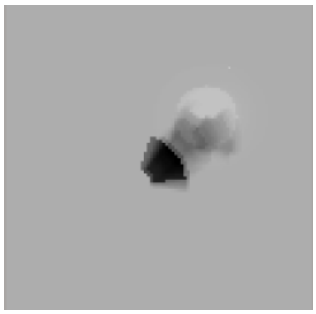
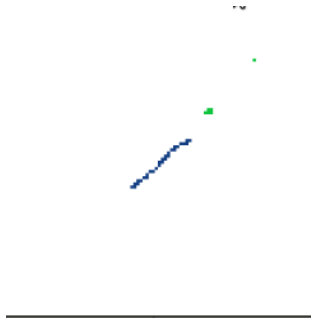
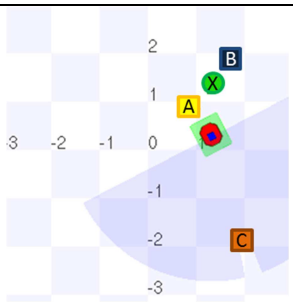
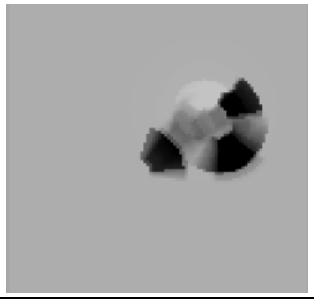
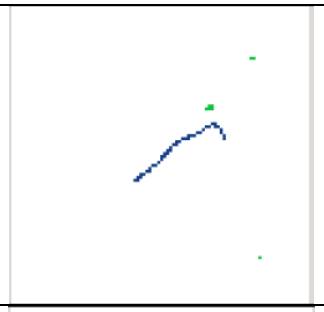
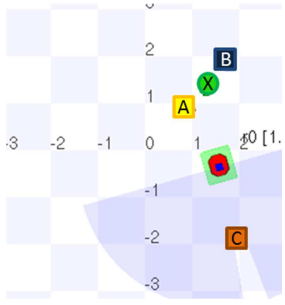
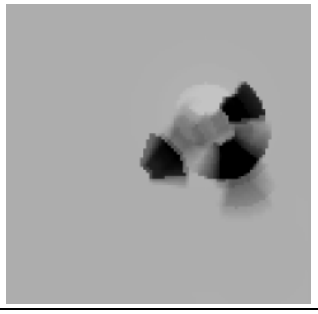
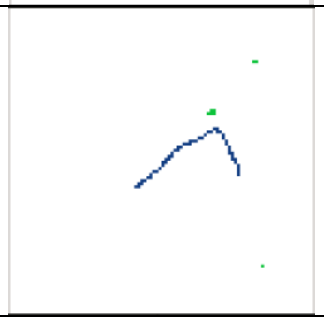
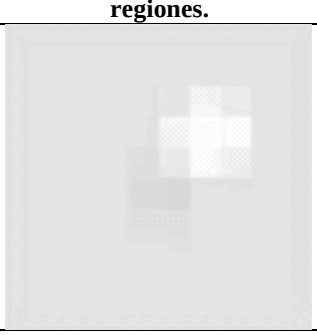
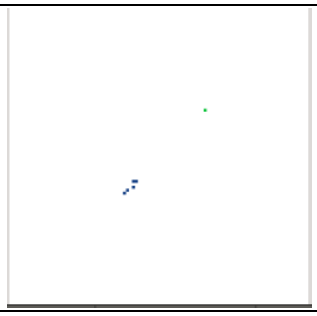
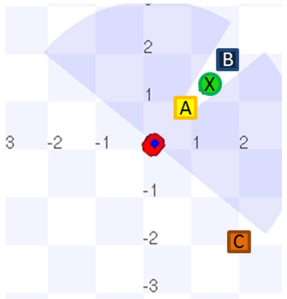
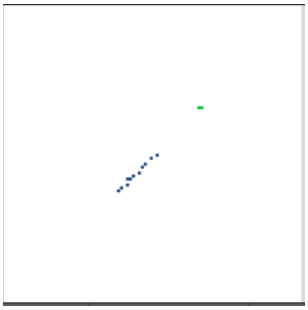
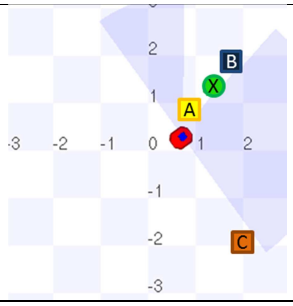
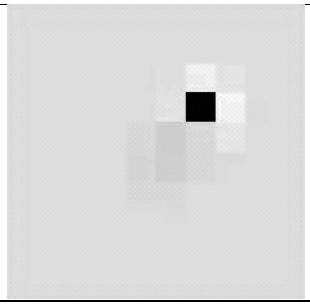
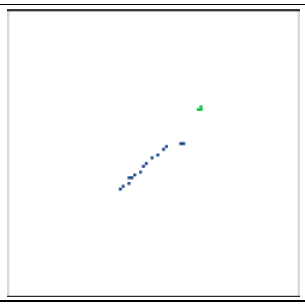
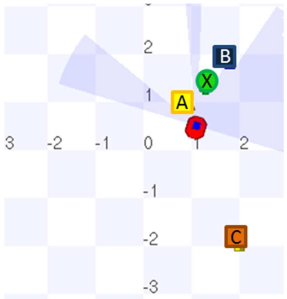
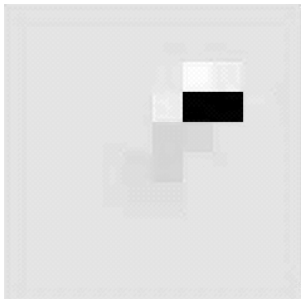
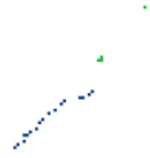
Descripción	Mapa	Mapa probabilidad del objeto primario	Trayectoria del robot
El objeto de búsqueda es *X*. El robot detecta el objeto *A*, y se realiza la convolución con la máscara respectiva			
El robot se acerca al objeto *A*, el objeto primario no es detectado			
El evasor de obstáculos del láser hace que el robot evada el objeto *A*.			
El campo de visión donde se calcula el máximo local, ahora apunta a otra dirección, por lo que el robot se aleja del objeto *X*.			

Tabla 2. Comportamiento utilizando la estrategia de búsqueda propuesta

Descripción	Mapa	Mapa de probabilidad del objeto primario usando análisis del máximo global por regiones.	Trayectoria del robot
El objeto de búsqueda es *X*. El robot detecta el objeto *A*, y se realiza la convolución con la máscara respectiva, luego se obtiene la UE para cada región.			
El robot se aproxima a la mejor región, que está alrededor del objeto *A*.			
Se acercó demasiado a la región y no se encontró el objeto principal *X*, por lo tanto se marca la región como no valida.			
El robot marca como no valida la posición actual y se mueve al siguiente punto. El objeto *X* es detectado (Exploración terminada).			

3.4. Creación de máscaras de relación espacial

Se crean relaciones espaciales complejas entre los objetos definidos como sumas ponderadas de relaciones espaciales básicas. Cada relación espacial básica corresponde a una categoría semántica de distancia significativa para los seres humanos como "muy cerca" (MC), "cerca" (C), "lejos" (L), y "muy lejos" (ML). El uso de estas relaciones espaciales es apropiado, ya que permite al sistema estimar un conjunto de distribuciones de probabilidad básicas a partir de muestras de las posiciones relativas de los objetos en el mundo real. Las máscaras para cada una de las relaciones espaciales se definen en dos versiones, máscaras duras y máscaras suaves. Las máscaras duras se definen por dos umbrales y tienen un perfil rectangular, mientras que las máscaras suaves están definidas por cuatro umbrales y tienen un perfil en forma de trapezoide.

Cada máscara básica es normalizada para que la suma sea uno sobre todas las celdas en el mapa, así una constante de normalización es añadida a las fórmulas.

La separación entre los umbrales de las máscaras duras están definidas como:

$$R_{B|A\,dura}(i,j;a_1,a_2) = K * \begin{cases} 1 & a_1 \leq \sqrt{(ki)^2 + (kj)^2} < a_2 \\ 0 & otro\ caso \end{cases} \quad (3.30)$$

Donde i,j son los índices de la matriz de la máscara de convolución, K y k son constantes de normalización y a_1, a_2 son los umbrales para la separación de la circunferencia. Para cada categoría semántica se utilizan los siguientes umbrales de las máscaras duras:

$$R^{MC}_{B|A\,dura}(i,j) = R_{B|A\,dura}(i,j;0,u_1) \quad (3.31)$$

$$R^{C}_{B|A\,dura}(i,j) = R_{B|A\,dura}(i,j;u_1,u_2) \quad (3.32)$$

$$R^{L}_{B|A\,dura}(i,j) = R_{B|A\,dura}(i,j;u_2,u_3) \quad (3.33)$$

$$R^{ML}_{B|A\,dura}(i,j) = R_{B|A\,dura}(i,j;u_3,\infty) \quad (3.34)$$

Donde u_1, u_2 y u_3 son los umbrales que separan la máscara de convolución, y MC representa muy cerca, C cerca, L lejos y ML muy lejos.

Mientras que la separación entre los umbrales de las máscaras suaves están definidas como:

$$R_{B|A\ suave}(i,j;a_1,a_2,a_3,a_4) = K * \begin{cases} \dfrac{\sqrt{(ki)^2+(kj)^2}-a_1}{a_2-a_1} & a_1 \le \sqrt{(ki)^2+(kj)^2} < a_2 \\ 1 & a_2 \le \sqrt{(ki)^2+(kj)^2} < a_3 \\ \dfrac{a_4-\sqrt{(ki)^2+(kj)^2}}{a_4-a_3} & a_3 \le \sqrt{(ki)^2+(kj)^2} < a_4 \\ 0 & otro\ caso \end{cases} \quad (3.35)$$

Donde i,j son los índices de la matriz de la máscara de convolución, K y k son constantes de normalización y a_1, a_2 y a_3 son los umbrales para la separación de la circunferencia. Para cada categoría semántica se utilizan los siguientes umbrales de las máscaras suaves:

$$R^{MC}_{B|A\ suave}(i,j) = R_{B|A\ suave}(i,j;\ 0,0,u_1-\delta,u_1+\delta) \quad (3.36)$$

$$R^{C}_{B|A\ suave}(i,j) = R_{B|A\ suave}(i,j;\ u_1-\delta,u_1+\delta,u_2-\delta,u_2+\delta) \quad (3.37)$$

$$R^{L}_{B|A\ suave}(i,j) = R_{B|A\ suave}(i,j;\ u_2-\delta,u_2+\delta,u_3-\delta,u_3+\delta) \quad (3.38)$$

$$R^{ML}_{B|A\ suave}(i,j) = R_{B|A\ suave}(i,j;\ u_3-\delta,u_3+\delta,\infty,\infty) \quad (3.39)$$

Donde u_1, u_2 y u_3 son los umbrales que separan la máscara de convolución. Una máscara compleja se puede crear como la suma ponderada de las máscaras básicas tanto duras como suaves:

$$R_{B|A}(x,y) = C^{MC}_{B|A}\,R^{MC}_{B|A}(x,y) + C^{C}_{B|A}\,R^{C}_{B|A}(x,y) + C^{L}_{B|A}\,R^{L}_{B|A}(x,y) + C^{ML}_{B|A}\,R^{ML}_{B|A}(x,y) \quad (3.40)$$

Los cuatro coeficientes: $C^{MC}_{B|A}$, $C^{C}_{B|A}$, $C^{L}_{B|A}$, y $C^{ML}_{B|A}$ son llamados coocurrencias porque indican la frecuencia relativa de ocurrencia de un par de objetos para cada relación espacial. Estas coocurrencias pueden ser construidas a partir de muestras de las posiciones de los dos objetos mediante el cálculo del número de ocurrencias de cada relación espacial básica. Si el conjunto de muestras está dividido en categorías semánticas básicas y las estadísticas son n_{MC} para "muy cerca", n_C para "cerca", n_L para "lejos", y n_{ML} para "muy lejos", las coocurrencias se pueden calcular como:

$$C^{MC}_{B|A} = \frac{n_{MC}}{n_{MC} + n_C + n_L + n_{ML}} \quad (3.41)$$

$$C^{C}_{B|A} = \frac{n_C}{n_{MC} + n_C + n_L + n_{ML}} \quad (3.42)$$

$$C^{L}_{B|A} = \frac{n_L}{n_{MC} + n_C + n_L + n_{ML}} \quad (3.43)$$

$$C^{ML}_{B|A} = \frac{n_{ML}}{n_{MC} + n_C + n_L + n_{ML}} \quad (3.44)$$

Capítulo 4. METODOLOGÍA DE SIMULACIÓN Y VALIDACIÓN

En este capítulo se detalla los pasos para obtener la matriz de coocurrencia mediante ejemplos y los pasos para realizar la simulación realista del robot.

Para calcular la matriz de coocurrencia entre los objetos *teclado, cpu* y *router* como objetos secundarios y *monitor* como objeto principal se obtienen 243 imágenes en 2D, siendo el tamaño de los objetos conocido se usa un algoritmo para estimar la profundidad y la distancia entre los objetos secundarios y el objeto principal en las 243 imágenes. Una vez obtenido las distancias entre los objetos se calcula un umbral para separar las distancias respecto a las categorías "muy cerca", "cerca", "lejos" y "muy lejos", de acuerdo a estas estadísticas se calcula la matriz de coocurrencia.

Para la simulación realística del robot, se realiza una simulación de la detección de objetos en 3D que considera la oclusión de objetos así como la probabilidad de detección para cada objeto de acuerdo a su posición.

4.1. Cálculo del estimador de profundidad

Para elegir los objetos que serán utilizados en esta base de datos, se presentan las siguientes restricciones:

- Los objetos deben ser representados mediante rectángulos
- Los objetos deben estar libres de oclusiones para poder obtener las esquinas de cada rectángulo.
- Se debe contar con los objetos en la realidad.
- Los objetos tienen que ser conocidos y se debe tener las mediciones del largo y ancho.

Dadas estas restricciones, los objetos seleccionados son: *monitor*, *teclado*, *cpu* y *router*, donde el objeto "*monitor*" será seleccionado como objeto primario (objeto a buscar) y los demás objetos como secundarios (aportan información para la búsqueda). El orden que se puede tener al manipular objetos de la vida cotidiana entre personas es muy variable, y se desea obtener una aproximación de distancias entre algunos objetos incluyendo casos extremos que pueden existir por ejemplo una *cpu* debajo de un *monitor*, un *teclado* sobre una *cpu* o un *monitor* que se encuentre solo. Para esto se utilizaran imágenes desde bases de datos en dos dimensiones ya que conseguir bases de datos amplias de escenas con profundidad todavía no es accesible.

Para realizar esta estimación, se estudian los siguientes puntos: modelo de cámara y geometría 3D, obtención de la matriz $[R|t]$ y estimación de profundidad, modelo del sensor *Kinect*, y

estimación de profundidad a través del sensor *Kinect*. Luego de obtener los resultados por los dos modelos, se compara para saber la veracidad del modelo de estimación.

4.1.1. Modelo de cámara y Geometría 3D

Una cámara es un sensor que permite realizar una transformación entre el mundo tridimensional (3D) y una imagen bidimensional (2D). Un modelo simple y muy estudiado es la cámara *Pinhole*, la cual mediante un orificio muy pequeño *C* en una pared deja entrar la luz externa que es proyectada en una pared interior de la cámara oscura. El resultado en esta cámara es una imagen invertida del mundo exterior sobre un plano que se encuentra en el interior de ésta.

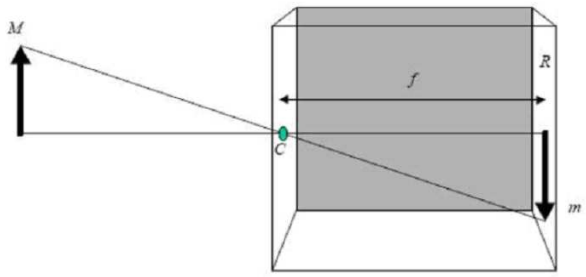


Figura 9. Esquema básico de una cámara Pinhole.

El modelo de cámara *Pinhole* puede ser considerado para ser utilizado como modelo de una cámara normal CCD, como se muestra en la Figura 9 , donde un lente es el encargado de dejar pasar la luz y en la parte posterior de la cámara se tiene un sensor tipo CCD que es el encargado de recibir la luz del exterior y convertirla a valores de voltajes [37].

Una cámara tiene tanto parámetros intrínsecos como extrínsecos, los parámetros intrínsecos son aquellos que describen el funcionamiento de una cámara como por ejemplo la distancia focal, el punto central, y el centro óptico. Los parámetros extrínsecos son los que definen la relación espacial entre la cámara y el mundo real, tales como la rotación y traslación que relacionan la orientación y posición de la cámara respecto al sistema de coordenadas del mundo real [38].

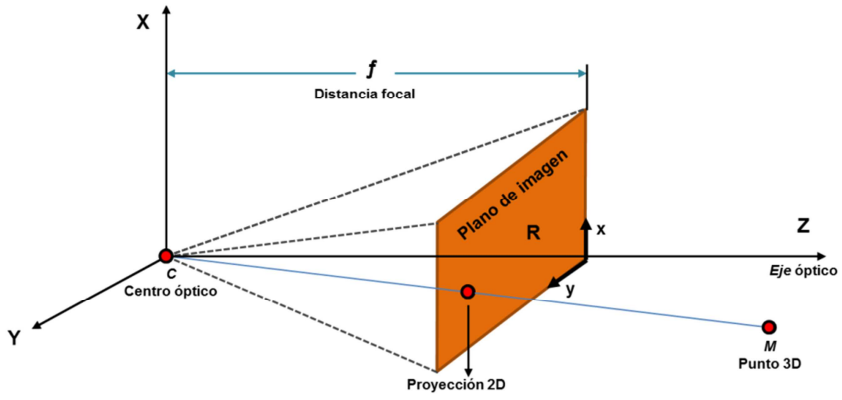


Figura 10. Modelo de una cámara pinhole 3 dimensiones.

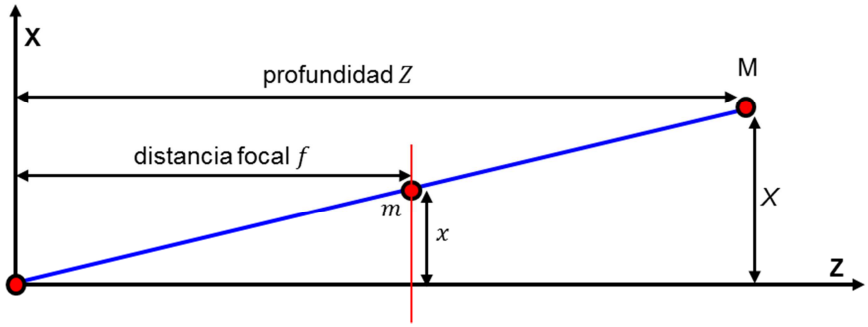


Figura 11. Modelo de una cámara pinhole 2 dimensiones.

En la Figura 10 se puede ver la representación en 3 dimensiones del modelo pinhole y en la Figura 11 la representación en 2 dimensiones.

Para representar este modelo matemáticamente se toman los parámetros intrínsecos que son dos distancias focales f_x y f_y, más el punto principal (c_x, c_y), los parámetros extrínsecos y la posición de los puntos proyectados de 3 dimensiones en el mundo real. Con estos parámetros la proyección de los puntos en modelo físico de la cámara se puede poner en una fórmula matricial simple:

$$s * m' = K\ [R|t] * M' \tag{4. 1}$$

En donde s es un factor de escala homogéneo, el valor de m' es interpretado como la proyección del mundo en 2D y es representado como un vector $[u, v, 1]$. La matriz K son los parámetros intrínsecos de la cámara. La matriz de rotación y traslación $[R|t]$ llamada matriz de parámetros extrínsecos es usada para describir la posición de una cámara alrededor de una escena estática o viceversa. La matriz M' contiene las coordenadas (X, Y, Z) de un punto 3D en el espacio de coordenadas del mundo real. Extendiendo la formula en matrices se tiene:

$$S\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} * \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4. 2}$$

4.1.2. Obtención de la Matriz $[R|t]$ y estimación de profundidad

Para encontrar una estimación de la profundidad a la que se encuentra un objeto se siguen los siguientes pasos:

1. Se obtienen patrones geométricos de los objetos, en este caso el alto y ancho de varios *monitor*es y *teclado*s (en centímetros), y luego se obtiene los promedios de estos valores.

2. Se toma una imagen a cierta distancia de los objetos escogidos (*monitor* y *teclado*) para luego marcar cuatro puntos de las esquinas del objeto en la imagen, de tal forma que se obtenga los valores de alto y ancho en pixeles (ver Figura 12). Estas imágenes son tomadas desde una cámara RGB del *kinect*, donde los valores de los parámetros intrínsecos son [39]:

$$K = \begin{bmatrix} 525 & 0 & 640 \\ 0 & 525 & 320 \\ 0 & 0 & 1 \end{bmatrix} \tag{4. 3}$$

3. Contando con los datos de la distancia (ancho y alto) del objeto en pixeles y en centímetros más la matriz de calibración, se puede obtener una aproximación de la matriz de rotación y traslación, esto se logra con la función ***FindExtrinsicCameraParams2*** de la librería de OpenCV. Esta función minimiza el error de proyección utilizando los parámetros intrínsecos conocidos mediante aproximaciones con funciones jacobianas.

4. Una vez obtenidos los parámetros extrínsecos (matriz $[R|t]$), se realiza una multiplicación por el punto medio del modelo de objeto $[X_R/2,\ Y_R/2,\ 0,\ 1\ ]$. El resultado es una aproximación de las coordenadas de la cámara respecto al objeto en las 3 dimensiones. De estas coordenadas la que nos interesa es el valor de la profundidad Z_c.

$$\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} * \begin{bmatrix} \frac{X_R}{2} \\ \frac{Y_R}{2} \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} X_C \\ Y_C \\ Z_c \end{bmatrix} \tag{4. 4}$$

Figura 12. Imagen de un *monitor* con cuatro puntos seleccionados.

A continuación en la Figura 13 se muestra el esquema del proceso realizado.

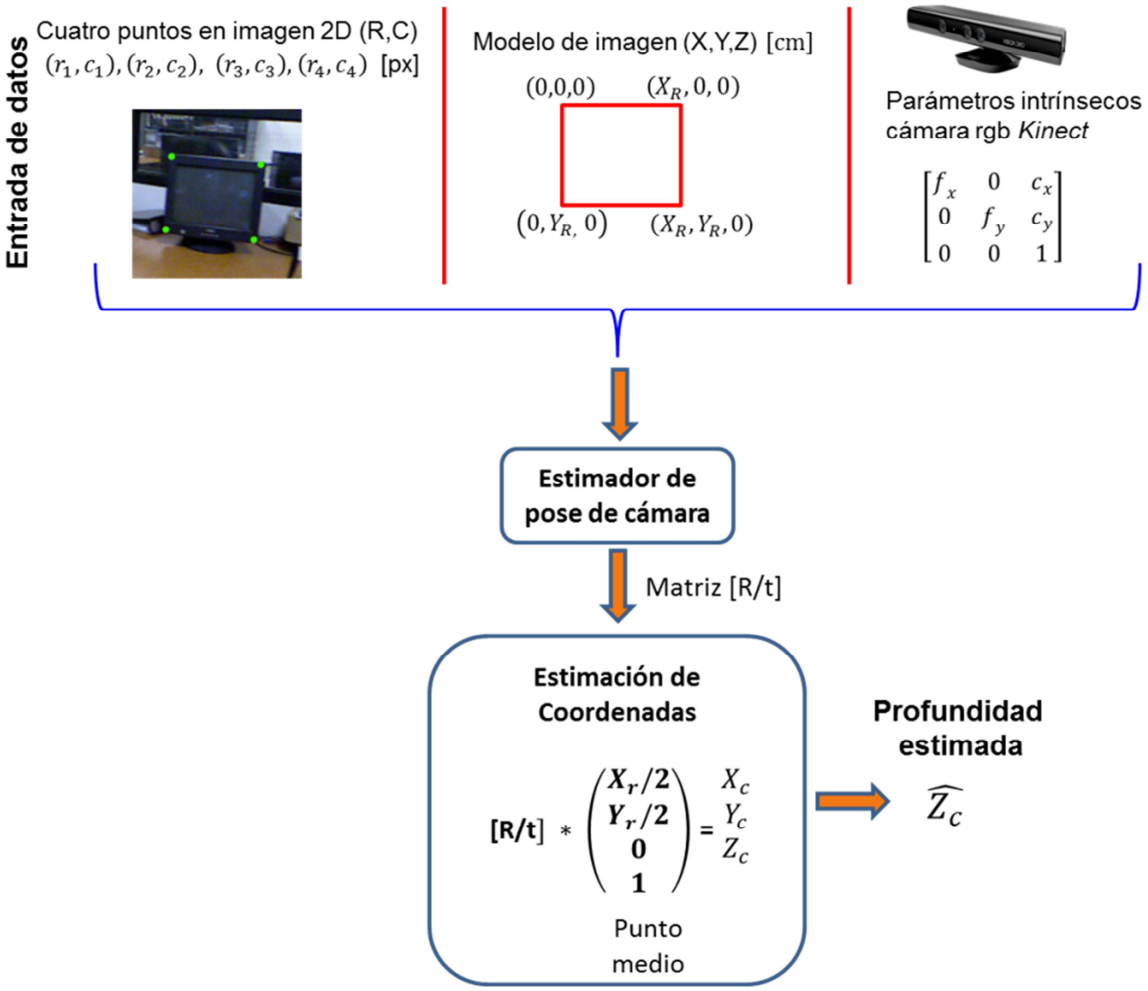


Figura 13. Esquema para obtención de profundidad de un objeto.

4.1.3. Modelo del sensor Kinect

El sensor *Kinect* es un dispositivo desarrollado por Microsoft, que permite a los usuarios controlar e interactuar con la consola sin necesidad de tener contacto físico con un controlador de videojuegos tradicional. Este sensor tiene la forma de una barra horizontal de aproximadamente 23 cm conectada a una pequeña base circular con un eje de articulación de rótula, y está diseñado para ser colocado longitudinalmente por encima o por debajo del televisor.

Figura 14. Forma física del *Kinect* [40].

El sensor *Kinect* se compone de una cámara RGB tradicional (Resolución 640x480 RGB 30fps VGA), un emisor de infrarrojos, un sensor CMOS monocromático de infrarrojos, 4 micrófonos (16bit velocidad de muestreo: 16kHz) y un motor. El emisor de luz infrarroja (IR) y el sensor CMOS monocromo forman la cámara de rango [41]. Ambos se encuentran alineados a lo largo del eje horizontal del dispositivo, a una distancia (denominada "línea base") de 75mm., con ejes ópticos paralelos (ver Figura 14). Esta disposición dentro del *Kinect* facilita los cálculos de profundidad, que se basan en un principio similar al de triangulación activa entre emisor y cámara, esto es, entre los rayos visuales de los puntos proyectados y sus correspondientes proyecciones en la imagen [40]. A diferencia de los métodos tradicionales de triangulación activa, los desarrolladores de la tecnología de la cámara de rango presente en *Kinect* proponen una técnica ingeniosa para la obtención de información 3D, denominada Codificación de Luz (*Light Coding*) [42].

4.1.4. Estimación de profundidad a través del sensor Kinect

Los *drivers* del sensor *Kinect* utilizados en este trabajo funcionan a través de la librería *Kinect SDK*, que es un kit de desarrollo diseñado por Microsoft para facilitar el uso de la programación con Kinect. El programa utilizado entrega tres imágenes que son:

- **Imagen RGB,** que es la típica imagen en color devuelta por una cámara tipo webcam.
- **Imagen de profundidad,** es una imagen monocromática donde el nivel de gris varía respecto a la distancia de los objetos.
- **Imagen fusionada,** que es una imagen donde se realizó una calibración de las dos imágenes anteriores obteniendo así la profundidad y el color original de la imagen.

La cámara de profundidad tiene un diferente rango que la imagen de color, por lo que la librería *SDK Kinect* realiza una calibración de las imágenes entregando una imagen de color corregida con la misma posición de la imagen de profundidad.

Cualquier punto en la imagen fusionada tendrá la misma coordenada en la imagen de profundidad, pudiendo obtener así la profundidad de cualquier objeto presente en la imagen.

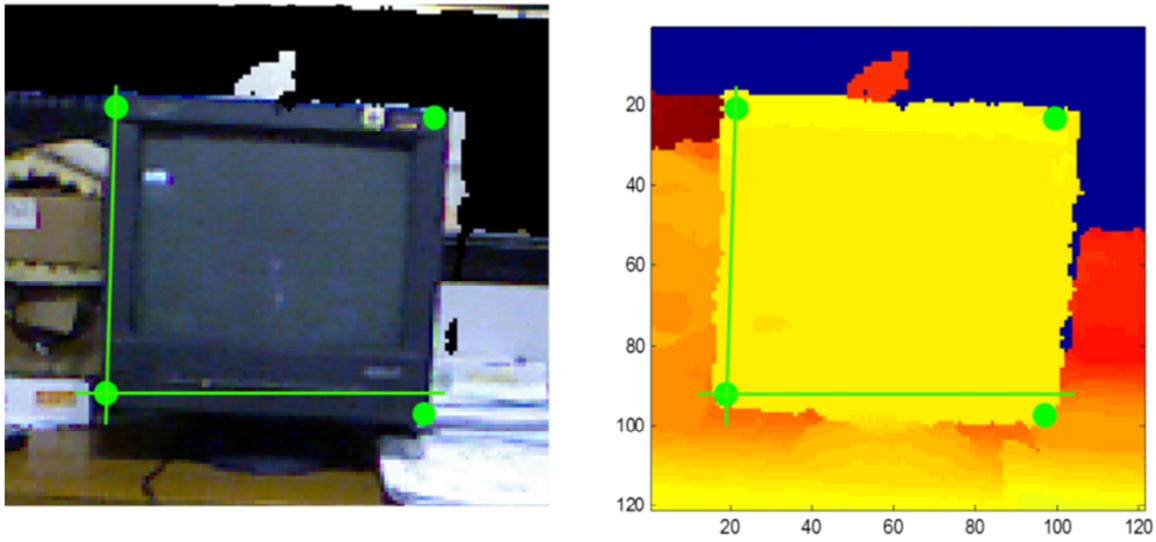


Figura 15. Cuatro puntos seleccionados en la imagen fusionada y de profundidad.

Tomando cuatro puntos sobre un objeto seleccionado, se obtienen cuatro coordenadas en tres dimensiones $[X, Y, Z]$ (ver Figura 15).

$$\begin{bmatrix} (x_1, y_1) = z_1 \\ (x_2, y_2) = z_2 \\ (x_3, y_3) = z_3 \\ (x_4, y_4) = z_4 \end{bmatrix} \rightarrow \text{Punto medio (x,y)} \rightarrow Z_c^k \qquad (4.5)$$

De los cuatro puntos obtenidos, se tomara el valor de la profundidad del punto medio entre los 4 puntos, este valor se guarda en la variable Z_c^k, para luego poder comparar con los resultados con la profundidad obtenida de la estimación en $\widehat{Z_c}$.

4.1.5. Comparación de resultados

Los resultados obtenidos en la ecuación (4.5) dan por resultado la distancia de la cámara al objeto, estas distancias son comparadas para verificar el correcto funcionamiento del resultado del modelo matemático de la ecuación (4.4). Para este efecto se tomó como ejemplo de 31 imágenes de *monitor*es y 25 imágenes de un *teclado* de computadora mediante el sensor *kinect*, teniendo así las imágenes de profundidad e imágenes de color fusionadas que serán usadas para

obtener los 4 puntos de coordenadas en 2 dimensiones. Los resultados se despliegan en la siguiente tabla que se presenta a continuación.

Tabla 3. Comparación de resultados de *monitor*es.

Coordenadas (pixeles)								Z_c estimada mediante Homografía [mm]	Z_c^k promedio por *Kinect* [mm]	Diferencia [mm]
x1	y1	x2	y2	x3	y3	x4	y4			
158	97	213	103	209	151	153	144	1828,5	1897,103	68,60
122	114	179	119	175	164	121	159	1845,5	1889,851	44,35
115	115	171	120	166	166	111	160	1853	1903,524	50,52
115	111	171	117	167	161	111	158	1855,5	1893,919	38,42
112	109	167	114	164	160	107	155	1845,5	1893,635	48,14
125	96	180	102	174	148	118	141	1838,25	1897,038	58,79
120	80	179	87	173	135	115	128	1800	1809,506	9,51
106	89	171	96	167	149	103	143	1617	1640,697	23,70
114	90	184	98	178	152	112	145	1565,25	1548,361	16,89
156	97	228	105	221	161	153	154	1527,5	1515,404	12,10
167	99	237	100	234	160	165	157	1494,5	1540,605	46,10
107	58	190	62	187	133	105	135	1324,75	1256,987	67,76
112	53	196	56	190	129	111	127	1346	1279,905	66,10
109	48	189	52	184	126	106	121	1375,25	1288,769	86,48
72	30	159	32	158	111	74	109	1298	1206,363	91,64
56	74	137	75	134	149	57	145	1391,75	1280,853	110,90
74	74	144	74	144	144	72	138	1474,5	1383,379	91,12
115	43	218	46	218	140	116	142	1085	984,2542	100,75
161	52	264	52	266	146	162	148	1080,75	985,5651	95,18
169	56	273	52	276	147	171	151	1081	987,218	93,78
166	48	268	46	268	139	166	142	1091,75	1006,117	85,63
163	39	266	39	265	133	163	135	1085	992,8902	92,11
159	37	264	36	262	128	161	132	1088,25	1003,311	84,94
158	30	263	27	263	123	160	127	1080,25	983,6907	96,56
264	29	264	124	162	129	160	31	1078,5	958,6766	119,82
162	28	266	25	267	120	166	127	1072,75	976,2924	96,46
139	73	235	71	237	160	143	166	1141,5	1036,473	105,03
146	111	239	117	237	204	143	204	1170,75	1051,911	118,84
139	77	200	77	199	134	139	133	1644,25	1669,854	25,60
115	83	178	83	178	139	116	139	1648,5	1670,486	21,99
119	71	180	71	180	127	120	127	1644,5	1684,913	40,41

Como se puede ver en la Tabla 3, los resultados obtenidos desde el modelo en 2 dimensiones mediante homografía, tienen una aproximación muy cercana a los resultados obtenidos desde el

kinect. El valor máximo del error es de 119.82 [mm] que equivale a 11.98 [cm], y el valor mínimo es de 9.5 [mm], que equivale a decir menos de un centímetro.

Tabla 4. Comparación de resultados de *teclado*s.

Coordenadas (pixeles)								Z_c estimada mediante Homografía [mm]	Z_c^k promedio por *Kinect* [mm]	Diferencia [mm]
x1	y1	x2	y2	x3	y3	x4	y4			
96	177	195	170	204	189	97	196	1212	1148,132	63,87
100	166	195	165	202	182	97	182	1253	1189,77	63,23
93	160	187	162	189	178	87	175	1269,5	1218,612	50,89
59	149	159	148	161	166	50	166	1205	1147,494	57,51
45	149	150	149	150	167	35	166	1183	1110,201	72,80
68	151	164	160	161	177	54	167	1225,5	1162,128	63,37
104	166	212	151	233	168	112	190	1044,25	976,5932	67,66
122	164	230	152	250	169	131	186	1041	989,5661	51,43
149	172	259	159	278	177	158	196	1028,25	999,7492	28,50
159	176	268	159	289	177	169	198	1033,5	1000,563	32,94
154	166	262	151	279	169	164	190	1080	1035,224	44,78
154	165	259	152	279	170	162	189	1051,25	1031,982	19,27
152	158	259	144	277	161	162	181	1064	1032,653	31,35
150	152	255	136	276	153	159	174	1050,75	1021,164	29,59
150	152	256	136	278	155	160	175	1051,75	1005,548	46,20
154	151	259	133	281	150	164	173	1045,75	1015,911	29,84
154	150	259	133	282	151	165	173	1032	1006,715	25,29
122	165	220	146	240	162	132	188	1092,25	1068,967	23,28
130	190	231	171	251	190	141	214	1091,25	1048,019	43,23
134	144	205	141	212	154	136	157	1616,75	1593,015	23,73
100	150	170	144	181	156	103	162	1595	1552,99	42,01
106	137	174	132	186	145	107	151	1594,75	1565,162	29,59
115	138	185	134	193	147	117	153	1601,25	1594,572	6,68
114	148	183	142	193	154	118	163	1598,25	1587,886	10,36

Para el ejemplo con *teclado*s de computadora se obtienen resultados similares, como se puede observar en la Tabla 4. El valor máximo del error es de 72.799 [mm] que equivale a 7.3 [cm], y el valor mínimo es de 6.6785 [mm], que equivale a un valor menor a un centímetro.

Las comparaciones de los valores de profundidad pueden ser visualizadas gráficamente en la Figura 16 y Figura 17.

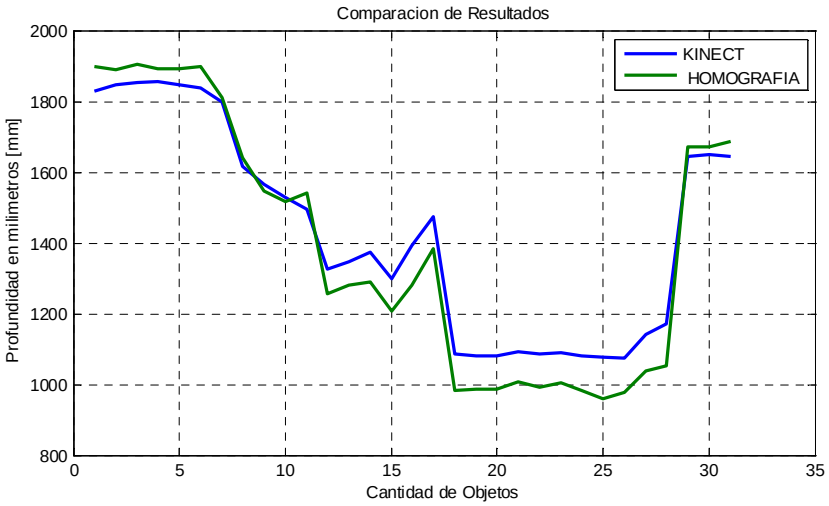


Figura 16. Resultados de los valores de profundidad de "*monitor*es".

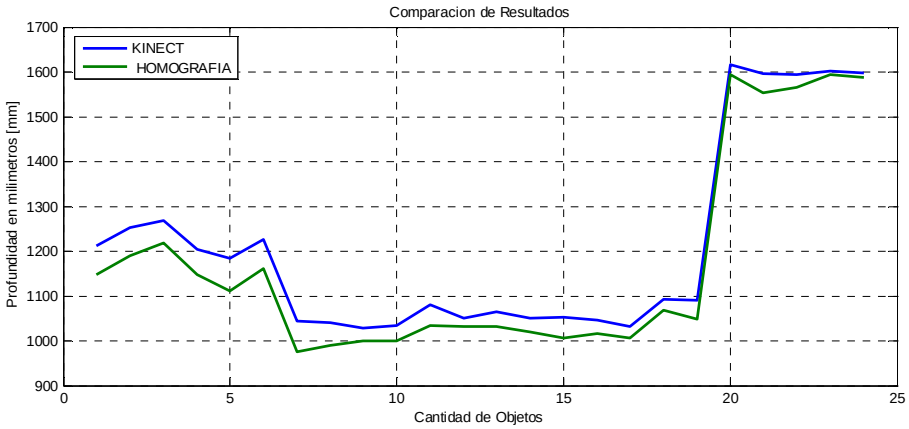


Figura 17. Resultados de los valores de profundidad de "*teclado*s".

4.2. Creación de la matriz de coocurrencia

La *matriz de coocurrencia* es el conjunto de valores de coocurrencia de dos o más objetos con una relación espacial particular. Para generar las matrices de coocurrencia se utilizaron un conjunto de 243 imágenes de oficinas etiquetadas desde la página web de *Flickr* [43] y el proyecto *Labelme* [22]. En cada imagen se obtienen las etiquetas de los objetos: "*monitor*", "*cpu*", "*teclado*" y "*router*". Como se conocen los tamaños promedio de los objetos y los parámetros de la cámara, es posible calcular una aproximación de la pose (posición y orientación) de cada objeto en el espacio mediante una estimación de profundidad. La existencia de uno o más objetos y sus poses en el conjunto de imágenes se utilizaron para construir matrices de coocurrencia con las categorías "muy cerca", "cerca", "lejos", y "muy lejos" de cada uno de los objetos con respecto a otros objetos.

Tomando las poses de un par de objetos, se calcula una distancia que es utilizada para seleccionar si la muestra pertenece a la categoría “muy cerca”, “cerca” o “lejos”. Si en una imagen se detecta un objeto que se encuentre solo, la muestra pertenecerá a la categoría de "muy lejos", ya que indicaría que está lejos del objeto principal. Se utilizó sólo la profundidad y el eje horizontal para calcular las distancias entre los objetos en una imagen, ya que las diferencias en la dirección vertical no afectan a la posición del objeto cuando se transforma a un mapa en grillas 2D.

El estimador de profundidad es aplicado a las 243 imágenes obtenidas, el resultado se puede ver en la Figura 18 donde el *monitor* se encuentra en la posición (0,0), el *teclado* es representado en forma de cuadrado de color rojo, el *cpu* es representado en forma de triángulo de color verde y el *router* está representado como un circulo de color morado.

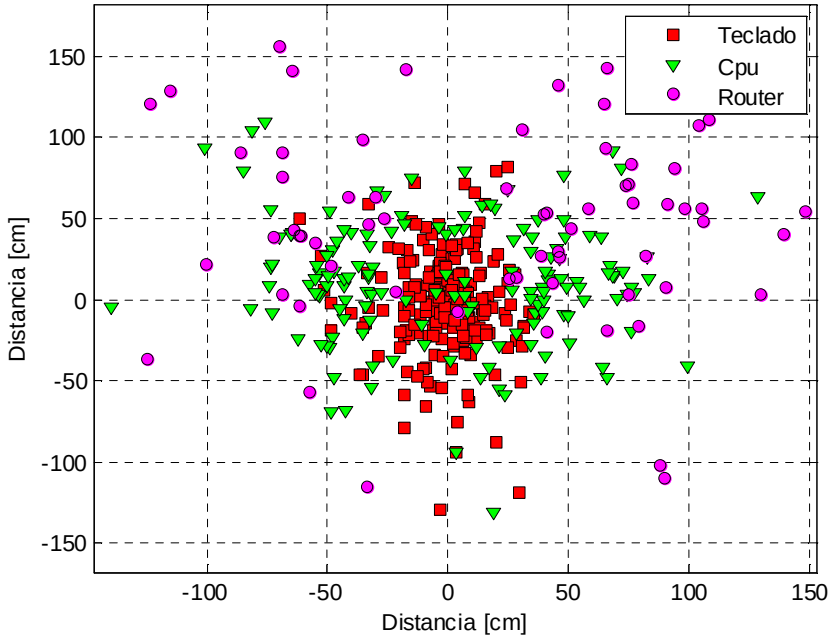


Figura 18. Distancias entre objeto “*monitor*” y objetos secundarios.

Para poder representar las distancias entre objetos en un mapa en dos dimensiones mediante categorías semánticas, se realizan anillos circulares. Los valores de los diferentes radios para cada categoría semántica, se seleccionan teniendo en cuenta las estadísticas de las distancias entre dos objetos mediante el modelado de su proceso de selección como un problema de clasificación. Por lo tanto, el valor del radio óptimo entre dos categorías, por ejemplo, "cerca" y "lejos", es la que genera el mismo error medio de clasificación en ambas clases:

$$\frac{1}{N} * n \cong \frac{1}{M} * m \qquad (4.6)$$

Donde N y M son la cantidad de objetos secundarios de ambas clases, n es la cantidad de datos de una clase antes del umbral y m es la cantidad de datos de la otra clase después del umbral. En la Figura 19 se puede ejemplificar gráficamente la ecuación (4.6).

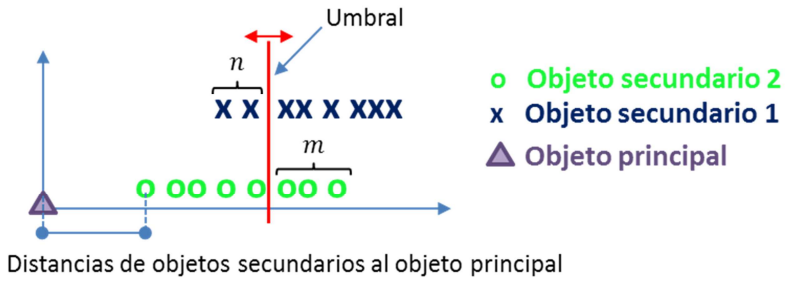


Figura 19. Umbral entre las distancias de objetos secundarios.

Para encontrar éste umbral óptimo se escoge el valor que minimiza la diferencia entre los promedios de los datos que quedan fuera del umbral para cada objeto, de esta forma el resultado es el valor mínimo posible:

$$u^* = \text{argmin}_U \left(abs \left(\frac{1}{N} * n - \frac{1}{M} * m \right) \right) \quad (4.7)$$

Donde U es un conjunto de valores del umbral en cada posición, y u^*es el umbral óptimo encontrado. Los umbrales en forma de anillos obtenidos que separan las categorías son tres. Para la categoría "muy cerca" el umbral es u1=40[cm], para la categoría "cerca" el umbral es u2=70[cm], para la categoría "lejos" el umbral es u3 = 150 [cm] y para la categoría "muy lejos" se toma todos los datos mayores a 180 [cm] (ver Figura 20).

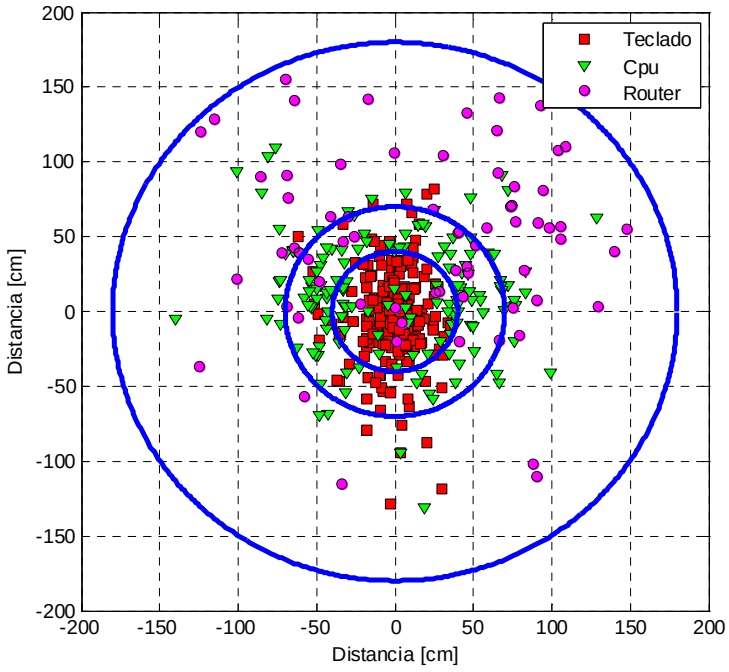


Figura 20. Umbrales en forma de radios.

El resultado del cálculo de la matriz de coocurrencia para el objeto "*monitor*" como objeto principal de acuerdo a las ecuaciones (3.41) al (3.44) se muestran en la Tabla 1.

Tabla 5. Coocurrencias finales de objetos alrededor del objeto principal "*monitor*".

Objeto principal *monitor* / **Categorias semanticas**	**Objetos secundarios**		
	teclado	***Cpu***	***Router***
Muy Cerca	0.773	0.178	0.061
Cerca	0.143	0.491	0.151
Lejos	0.046	0.258	0.485
Muy Lejos	0.038	0.074	0.303

4.3. Diseño de un reconocedor de objetos

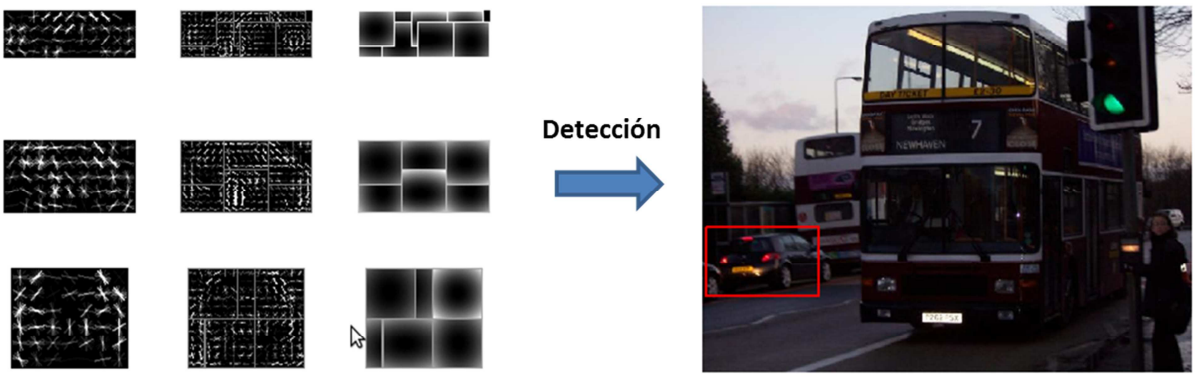
4.3.1. Histograma de Gradiente Orientado (HOG)

La idea general que presenta este descriptor es que la apariencia y forma de un objeto en una imagen puede ser representado por la distribución de la orientación de los gradientes. El descriptor HOG es invariante ante transformaciones geométricas y a los cambios de iluminación, por lo que es ideal para la detección de objetos.

La implementación del sistema se compone de los siguientes pasos como los autores lo mencionan en [44].

1. **División de la imagen en celdas.** Cada celda es dividida en 9 sub-celdas.
2. **Cálculo de gradientes.** Los gradientes se calculan mediante derivadas parciales en la dirección horizontal y vertical, proporcionando información sobre los contornos. Mediante la combinación de los dos gradientes se obtiene la magnitud y orientación del gradiente en cada pixel.
3. **Cálculo de histogramas.** Cada pixel contribuye a la creación de un histograma de orientación de gradientes. El histograma de la celda puede ser generado mediante los obtenidos en cada sub-celda.
4. **Normalización de los histogramas.** Esto suaviza los posibles cambios de iluminación. El descriptor HOG será el conjunto de histogramas de todas las celdas.
5. **Clasificación.** Luego de obtener los descriptores, se realiza la clasificación. Los autores mencionan que usando el la máquina de aprendizaje conocida como SVM lineal se obtienen buenos resultados junto con el descriptor HOG. Esta es capaz de representar una función de decisión mediante la definición de un hiperplano y generar un clasificador.

En la Figura 21 se puede ver los gradientes utilizados para detectar objetos.



a) Descriptor HOG de objeto automóvil b) Detección de objeto automóvil

Figura 21. Ejemplo de detección mediante descriptores HOG [45].

4.3.2. Máquina de Soporte Vectorial (SVM)

Las máquinas de soporte vectorial (*Support Vector Machines*, SVMs) son un conjunto de algoritmos de aprendizaje supervisado desarrollados por Vladimir Vapnik y su equipo en los laboratorios AT&T. Este método reconoce patrones y permite resolver problemas de clasificación y de regresión.

El funcionamiento de este método se puede ejemplificar considerando el caso más básico de clasificación, en el que hay que decidir entre dos clases. Entonces, asumiendo que los datos de entrenamiento $D=\{(\vec{x}_i, y_i), i=1...N\}$ con $y_i \in \{-1,+1\}$ son separables mediante un hiperplano, se debe encontrar el mejor hiplerplano (Figura 22).

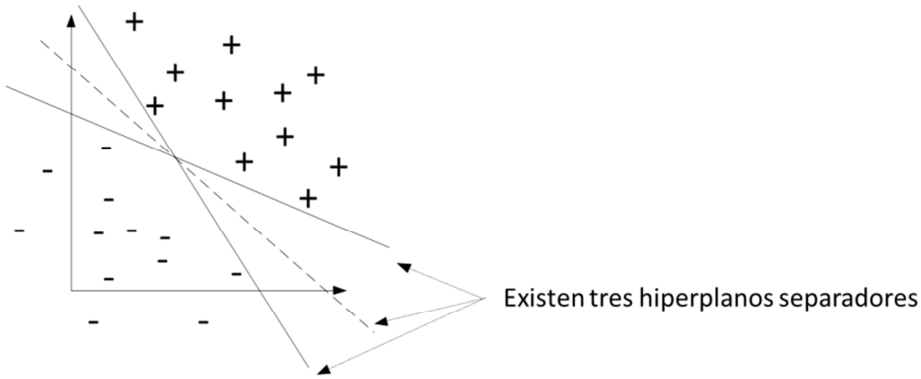


Figura 22. Hiperplanos que separan dos clases.

Intuitivamente, una SVM es un modelo que representa a los puntos de muestra en el espacio, separando las clases por un espacio lo más amplio posible. Cuando las nuevas muestras se ponen en correspondencia con dicho modelo, en función de su proximidad pueden ser clasificadas a una u otra clase.

Más formalmente, una SVM construye un hiperplano o conjunto de hiperplanos en un espacio de dimensionalidad muy alta (o incluso infinita) que puede ser utilizado en problemas de clasificación o regresión. Una buena separación entre las clases permitirá un clasificación correcta. Para más detalles se puede revisar el trabajo de Vapnik en [46].

4.3.3. Clasificación mediante el método *K-means*

K -means es un método de agrupamiento, que tiene como objetivo la partición de un conjunto de n observaciones en k grupos en el que cada observación pertenece al grupo más cercano a la media. Cuanto más cerca este un vector de características de esta media es más probable que pertenezca a esta clase. Se necesita por tanto una medida de distancia que equivale a una medida de semejanza o diferencia entre las diferentes clases y las características. Dos características parecidas pertenecerán con toda seguridad a la misma clase, mientras que dos clases distintas pertenecerán a diferentes clases. La medida más utilizada es la distancia euclidiana generalizada para espacios multidimensionales (aunque pueden utilizarse otro tipo de distancias):

$$d_{ij} = \sqrt{\sum_{k=1}^{n} (c_{kj} - v_{ki})^2} \tag{4. 8}$$

Donde *d* es la distancia mínima entre la clase *j* y la característica *i*, el valor de c_{kj} es el centro de la clase *j*, y el valor de v_{ki} es la característica *i-esima*.

4.3.4. Metodología de diseño del reconocedor de objetos

Utilizando el método de descriptores HOG y la clasificación se muestra el proceso para el reconocimiento de objetos. Este proceso se realiza entrenando el algoritmo con imágenes del objeto deseado y posteriormente es validado con otro conjunto de imágenes del objeto.

4.3.4.A. Preparación del conjunto de datos de entrenamiento

Para la base de datos de entrenamiento en este trabajo se tomaron 500 imágenes recortadas por cada orientación de un objeto en diferentes distancias. Estas 500 imágenes corresponden a 5 distancias desde la cámara al objeto (100 imágenes por cada distancia), los valores son 80, 120, 160, 200 y 250 centímetros. Cada orientación tendrá distintas variaciones respecto a 8 ángulos principales que son 0, 45, 90, 135. 180, 225, 270 y 315 grados de rotación del objeto frente a la cámara. Es decir se tiene en total 4000 imágenes de entrada por cada objeto. las distintas orientaciones consideradas representan al objeto en 3D que puede ser encontrado en un ambiente. Vale mencionar que se considera valores constantes para el **eje *z*** y el ***ángulo de inclinación*** de la cámara desde donde se toman las imágenes del objeto.

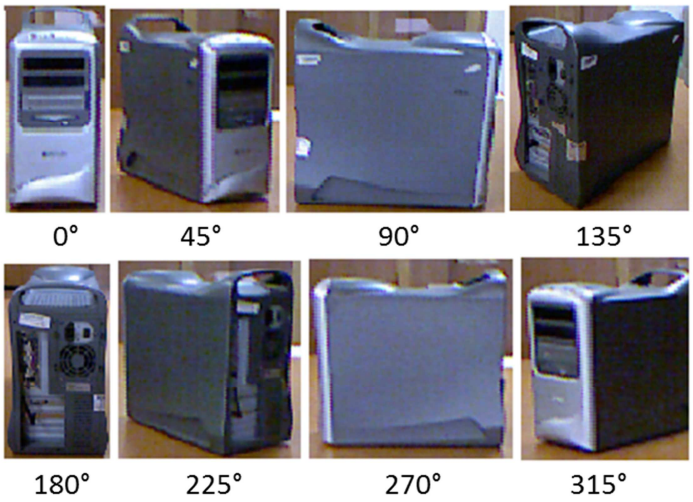


Figura 23. Objeto rotado en 8 ángulos.

Como se puede ver en la Figura 23, el objeto esta rotado de 45 en 45 grados, algunas rotaciones son parecidas (90° y a 270°) y otras muy distintas (0° y a 180°). Es por esta razón que el reconocimiento de objetos en un mundo real es una tarea muy difícil, ya que se aumenta la dimensión del problema el ver al objeto desde otra perspectiva.

Si bien reconocer un objeto rotando alrededor del mismo, es un problema de datos cuasi continuos, tomar 8 ángulos principales discretizan en cierta forma el problema ya que entre cada ángulo principal se realiza una rotación aproximada entre -22.5° y 22.5° como se puede ver en la Figura 24.

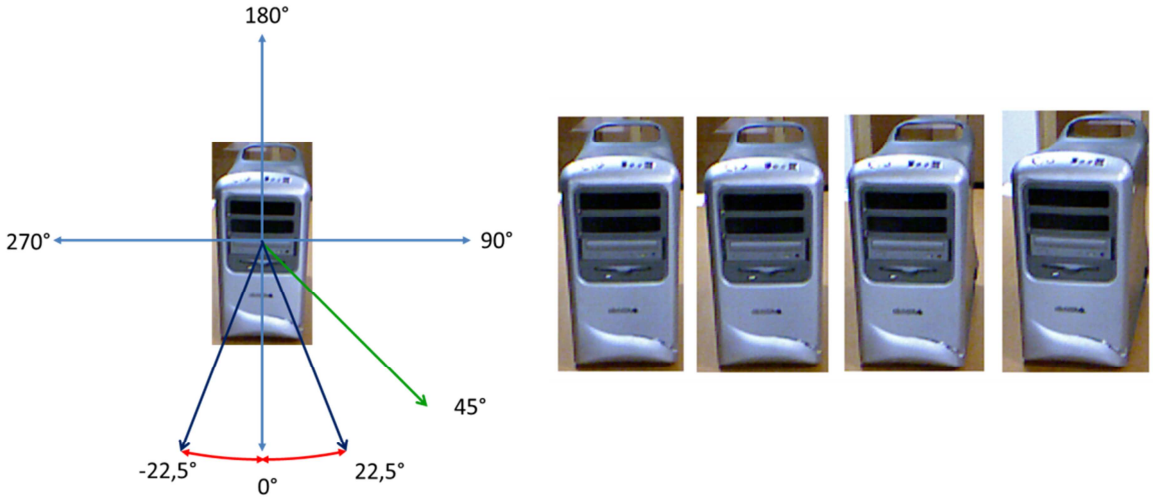


Figura 24. Rotaciones secundarias del objeto.

4.3.4.B. Etapa de entrenamiento

Esta es la etapa de enrolamiento, es decir se guardan la estructura del entrenamiento y una base de datos de los descriptores para ser usadas en la etapa de test. Las imágenes de entrada tienen el

modelo **objeto** y **no-objeto**, a estas imágenes primero se aplica una ecualización para que el histograma de la imagen tenga una distribución uniforme. Luego se aplica el algoritmo de HOG, y se obtiene como resultado un vector de características que se guardan en una BD de descriptores y se aplica un entrenamiento con el clasificador SVM (ver Figura 25).

4.3.4.C. Etapa de clasificación

Al igual que la base de datos de entrenamiento, se tomaron 500 imágenes considerando 5 distancias desde la cámara al objeto (100 imágenes por cada distancia), los valores son 80, 120, 160, 200 y 250 centímetros y la orientación tiene distintas variaciones respecto a 8 ángulos principales que son 0, 45, 90, 135. 180, 225, 270 y 315 grados de rotación del objeto frente a la cámara. La diferencia de esta base de datos respecto a la de entrenamiento es que estas imágenes no son recortadas y contienen un ambiente con otros objetos.

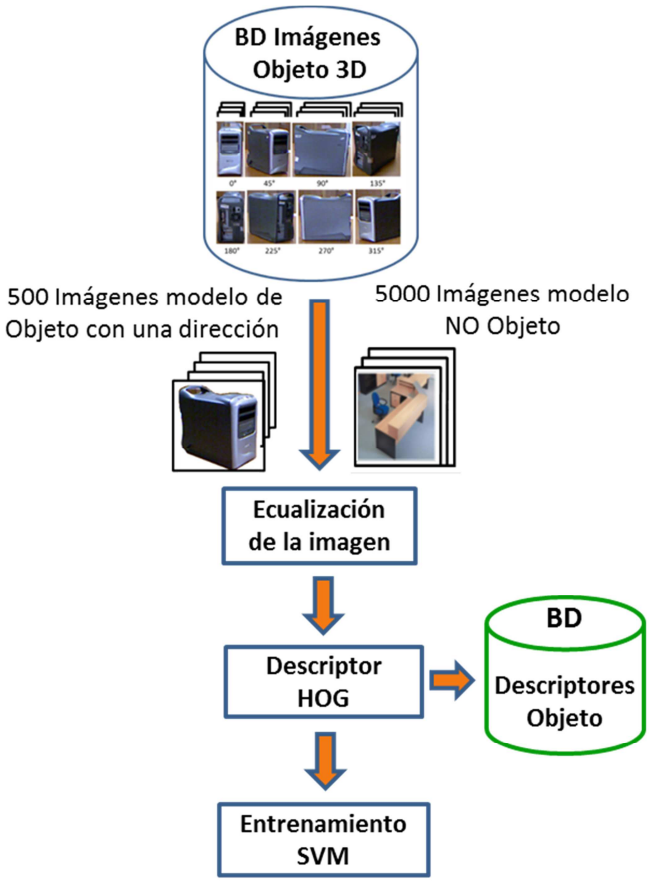


Figura 25. Diagrama del proceso de entrenamiento.

Las imágenes de esta base de datos de *test* serán procesadas mediante un *sliding windows* (ver Figura 26) que es un método para dividir la imagen en regiones de interés (ROI por sus siglas en inglés) barriendo toda la imagen en diferentes tamaños, esto con el fin de encontrar el objeto deseado de forma invariante a la distancia desde donde se toma la imagen.

Figura 26. Método de *sliding widows*(ventana deslizante).

En cada ventana seleccionada, se aplica una ecualización de la imagen para luego aplicar el descriptor HOG. Una vez obtenido el descriptor HOG, este será clasificado por la estructura entrenada previamente con el método de SVM.

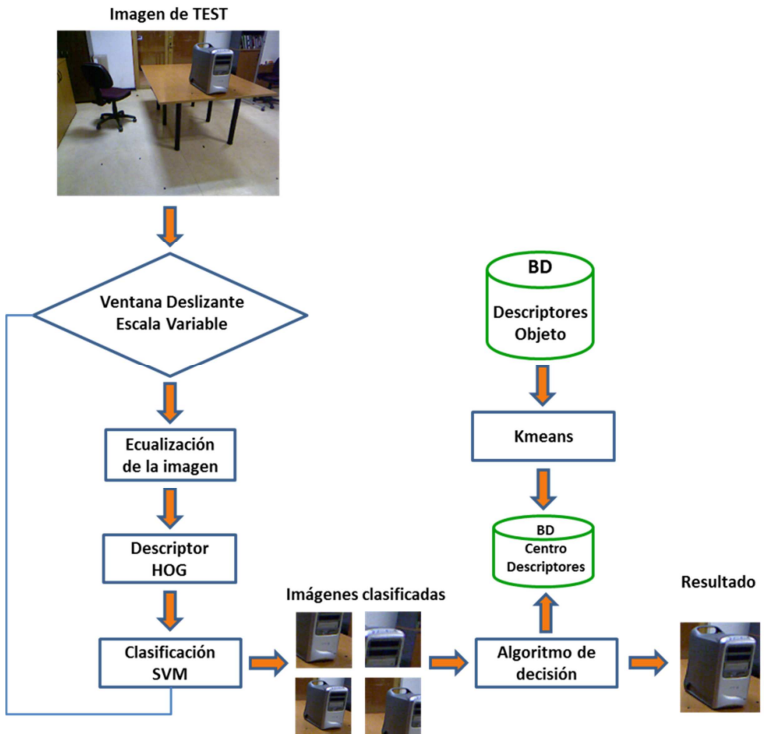


Figura 27. Diagrama del proceso de clasificación.

Al analizar distintos tamaños de regiones de interés traslapadas en cortas distancias, el clasificador puede entregar más de una imagen de detección; es decir las regiones de interés clasificadas como el objeto a buscar, estarán generalmente cerca y tendrán un alto traslape entre ellos (ver figura). Para escoger una sola detección que sea la más cercana al objeto, se usa un **algoritmo de decisión** mediante el uso de *clusters* por el método de *K-means* en el espacio de los descriptores de HOG. Este método contiene 10 centroides de los vectores de características entrenadas previamente. El diagrama del proceso de clasificación se puede ver en la Figura 27.

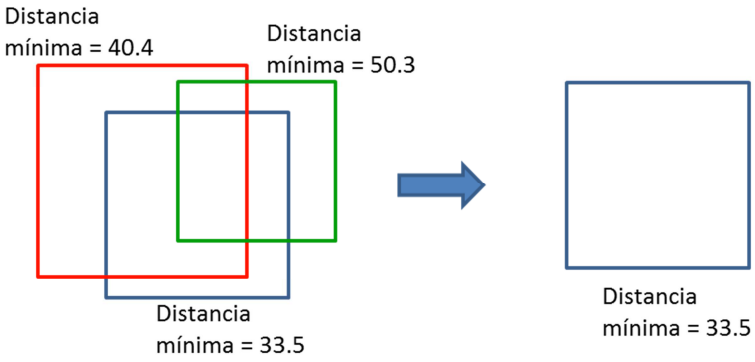


Figura 28. Funcionamiento del Algoritmo de decisión.

Las imágenes clasificadas serán comparadas mediante la distancia euclidiana para ver cuál de las imágenes clasificadas contienen una mínima distancia respecto al objeto deseado entregando una sola imagen como resultado (ver Figura 28).

4.4. Herramienta de simulación robótica Player/Stage

Los desarrollos respecto a la programación de robots han crecido de manera considerable, esto se debe a los avances tecnológicos que permiten dotar a estas máquinas de una mayor capacidad de procesamiento y de almacenamiento, así como los sensores de mayor precisión. Por esta razón las tareas a programar también son más complejas ya que no solo se necesita controlar la plataforma móvil, sino también controlar la interacción con los diferentes sensores y posibles actuadores. Una herramienta muy útil en la programación de robots son los simuladores. Los simuladores ofrecen un entorno virtual en el que se emulan las observaciones de los sensores y los efectos de las órdenes a los actuadores como también el ambiente de trabajo. Sirven como herramientas para la evaluación, depuración y ajuste del programa de control antes de llevar la aplicación al robot real [47].

Aunque la programación de robots puede resultar una tarea muy compleja, no es uno de los principales problemas. Sin embargo el costo de un buen robot puede ser un fuerte impedimento y más aún en el caso de investigar con más de un robot. Este es uno de los motivos por los cuales se utilizan simuladores para probar los programas y comprobar los resultados obtenidos, acercándose lo más posible a los que podrían obtenerse con el robot real. Cabe destacar que las condiciones del entorno simulado no son las mismas que las del entorno real, el cual presenta ruidos en las mediciones de los sensores, situaciones inesperadas, y otros aspectos que a veces no pueden ser reproducidos con fidelidad en la simulación. Los simuladores suelen ser utilizados para comprobar los programas que posteriormente serán implantados en el robot real, de una forma más metódica y exhaustiva, ahorrando en algunos casos bastante tiempo [48].

4.4.1. El proyecto *Player/Stage*

El proyecto *Player/Stage* fue fundado por *Brian Gerkey*, *Richard Vaughan* y *Andrew Howard* en el año 2000, basándose en software que habían desarrollado con *Kasper Stoy* y otros en los

laboratorios de investigación robótica de la Universidad de California del Sur. El nombre del proyecto fue tomado de una célebre frase de *Shakespeare* "*All the world's a stage, And all the men and women merely players*" (El mundo entero es un escenario, y los hombres y mujeres son simplemente actores) [49].

Este proyecto está compuesto por dos componentes bien diferenciados:

- ***Player*** es un servidor que permite controlar los dispositivos de un robot y obtener información de sus sensores. Es una capa *software* que abstrae los detalles del *hardware* del robot, independizándonos del mismo. Los algoritmos de control del robot funcionarán como clientes de *Player* (a través de sockets TCP/IP). Así es posible controlar el robot enviando mensajes que sigan el protocolo de comunicación de *Player* o llamando a funciones de las librerías de *Player*, que pueden abstraer los detalles de comunicación. La distribución actual de *Player* incluye librerías en lenguajes tan diversos como C++, *Java*, *Python* o *Lisp*.

- ***Stage*** es un simulador de robots móviles en 2D que se puede utilizar de manera conjunta con *Player* si se quiere hacer pruebas iniciales de algoritmos o en el caso de no disponer de un robot real.

4.4.2. Arquitectura del simulador Player/Stage

Player utiliza una estructura de cliente/servidor para pasar datos e instrucciones entre el código y el hardware del robot. *Player* es el servidor, y el dispositivo de hardware del robot está suscrito como un cliente que se comunica al servidor a través de un proxy.

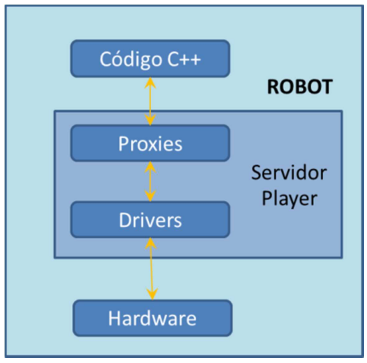


Figura 29. Estructura Cliente/Servidor de *Player*.

En la Figura 29 se muestra la estructura básica del *Player* cuando esta implementado en un robot, en la Figura 30 se muestra como es la estructura básica de *Player* con el simulador *Stage*.

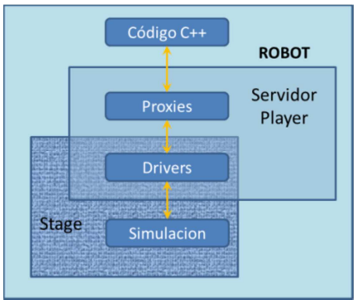


Figura 30. Estructura Cliente/Servidor de *Player* con el simulador *Stage*.

4.4.3. Configuración de Stage

El componente *Stage* puede simular a uno o más robots móviles, sensores, objetos y el ambiente de trabajo para interactuar con el robot mediante un entorno bidimensional. *Stage* además proporciona la simulación de una amplia variedad de sensores y actuadores, tales como sonar, láser, dispositivos de detección de color por visión, odometría y otros.

En la Figura 31 se puede observar una imagen de la versión *Stage* 3.2.2, simulando varios robots, obstáculos y el campo de visión por láser.

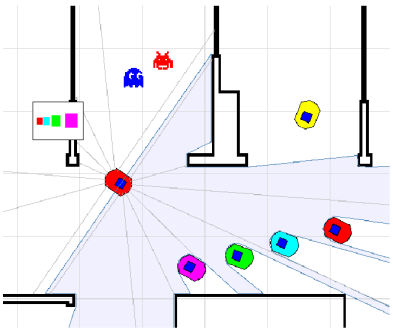


Figura 31. Entorno del simulador *Stage*.

Stage está conformado a su vez por tres componentes:

- Ambiente de trabajo o entorno: descripción del mundo
- Modelo de robot
- Control a través de *Player*

4.5. Simulación de la Percepción del Robot

El módulo de percepción consta de dos partes, el Modelado de la observación de objetos en un ambiente 3D y la Simulación realista de la detección de objetos.

4.5.1. Modelado de la observación de objetos en un ambiente 3D

El simulador *Player/Stage* ha sido diseñado para gestionar robots en entornos 2D [49]. Este simulador trae una herramienta de detección llamado *fiducial detector model*, que simula una detección en 2D, es decir en un plano horizontal. La estructura de esta función tiene como entradas un valor de rango mínimo y rango máximo (por sus siglas en ingles *range_min* y *range_max*), que son intervalos de distancia y el ángulo *fov* (*field of view*), que es el ángulo del campo de visión del robot en grados. Para reconocer a los objetos se utiliza la función *range_max_id* que detecta la ***id*** del objeto (cuál objeto es el detectado). Cualquier objeto con un ***id*** conocido en el plano 2D y que este dentro del *fov* será detectado por el robot. Esta función es muy útil a la hora de simular una exploración robótica con detección de objetos.

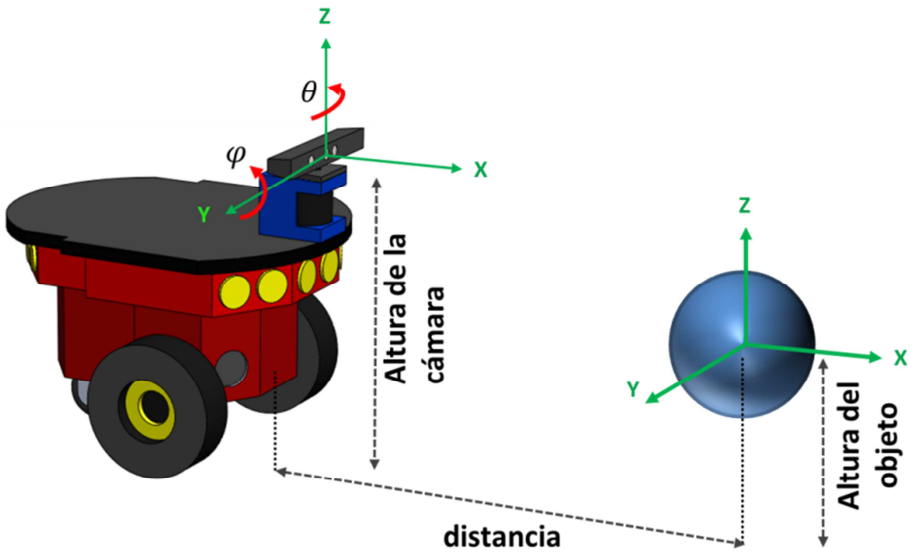


Figura 32. Parámetros geométricos necesarios para simular detección 3D.

Sin embargo, para el modelado de la observación de objetos en un entorno 3D parámetros adicionales deben ser considerados, como la pose de la cámara del robot y la altura del objeto. En la Figura 32 se puede ver la configuración del robot y la cámara utilizada en este trabajo.

Para modelar o simular la observación obtenida por un robot en un ambiente 3D, se utiliza una matriz de transformación que permite llevar la posición de los objetos del sistema de referencia del robot al de la cámara:

$$\begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} = \begin{bmatrix} \cos(-\varphi) & 0 & -sen(-\varphi) \\ 0 & 1 & 0 \\ sen(-\varphi) & 0 & \cos(-\varphi) \end{bmatrix} * \begin{bmatrix} \cos(-\theta) & -sen(-\theta) & 0 \\ sen(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x_{obj} - x_r \\ y_{obj} - y_r \\ z_{obj} - z_r \end{bmatrix} \quad (4.9)$$

Donde φ es el ángulo *pitch* de la cámara, θ es el ángulo *yaw* del robot en el eje horizontal, $[x_{obj}, y_{obj}, z_{obj}]$ es la posición del centro del objeto, $[x_r, y_r, z_r]$ es la posición de la cámara del

robot y $[x_{cam}, y_{cam}, z_{cam}]$ es la posición del objeto en el sistema de la cámara. Entonces la pose relativa del objeto es calculada como:

$$r_d = \sqrt{{x_{cam}}^2 + {y_{cam}}^2 + {z_{cam}}^2} \quad (4.\ 10)$$

$$\theta_d = atan2(y_{cam}, x_{cam}) \quad (4.\ 11)$$

$$\varphi_d = atan2(z_{cam}, x_{cam}) \quad (4.\ 12)$$

Donde r_d es la distancia del objeto hasta la cámara, θ_d es el ángulo *yaw* del objeto en el sistema de la cámara y φ_d es el ángulo *pitch* del objeto en el sistema de la cámara. Un objeto puede ser detectado si éste se encuentra dentro el campo de visión del robot (ver Figura 33), que se define como: $r_d \in [r_{min}, r_{max}]$, $\theta_d \in [\theta_{min}, \theta_{max}]$ y $\varphi_d \in [\varphi_{min}, \varphi_{max}]$.

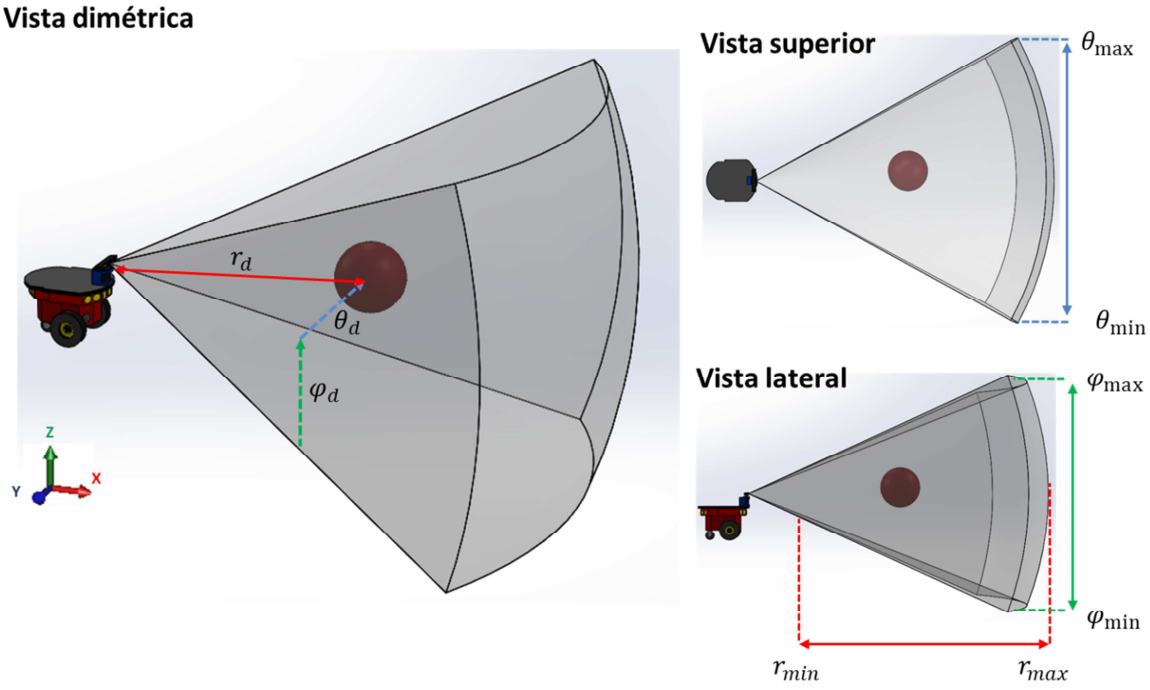


Figura 33. Vistas del modelo detector de objetos 3D.

La simulación de oclusión de objetos en un ambiente 2D es diferente al caso en un ambiente 3D (ver Figura 34). En el caso 3D, se modela cada objeto como una esfera cuyo centro es representado como centro del objeto, y el tamaño del objeto es representado con un radio $\boldsymbol{\rho}$ que varía de acuerdo al tamaño del objeto.

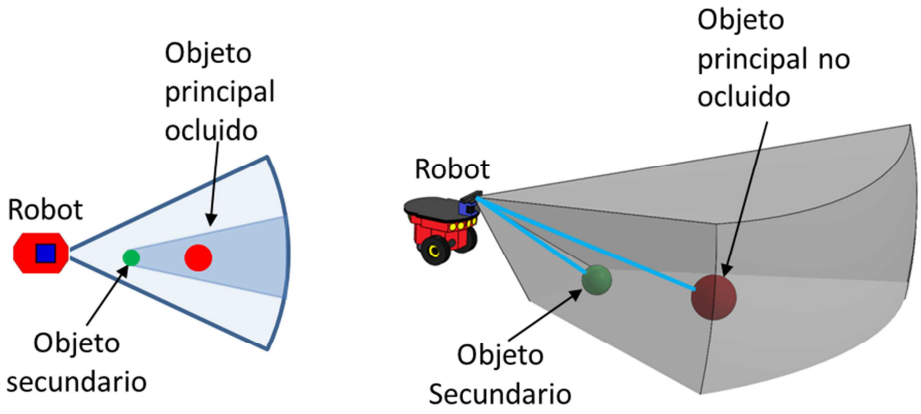


Figura 34. Visualización de la oclusión en un entorno 2D (izquierda) y en un ambiente 3D (derecha).

Para cada par de esferas dentro en el campo de visión, estas son proyectadas en el plano de la imagen y se verifica si intersectan en el mencionado espacio o no. En el caso en que intersecten, una oclusión ha de ser detectada y sólo el objeto más cercano será seleccionado como visible. Este procedimiento se muestra en la Figura 35.

Detección de oclusión

1. Para cada par de objetos
2. $obj_1: (\rho_1, \theta_{d\_1}, \varphi_{d\_1}),\ obj_2: (\rho_2, \theta_{d\_2}, \varphi_{d\_2})$
3. $ang\_proy_1 = atan2(\rho_1, x_{cam\_1})$
4. $ang\_proy_2 = atan2(\rho_2, x_{cam\_2})$
5. . $dist = \sqrt{(\theta_{d\_1} - \theta_{d\_2})^2 + (\varphi_{d\_1} - \varphi_{d\_2})^2}$
6. $Si\ (dist > ang\_proy_1 + ang\_proy_2)$
   obj_1 y obj_2 pueden ser detectados
7. *En otro caso*
   Existe oclusión

Figura 35. Procedimiento para calcular la oclusión de objetos.

Donde θ_{d_1} es el ángulo *yaw* del objeto 1 en el sistema de la cámara, θ_{d_2} es el ángulo *yaw* del objeto 2, φ_{d_1} es el ángulo *pitch* del objeto 1 en el sistema de la cámara y φ_{d_2} el ángulo *pitch* del objeto 2. Las variables ρ_1 y ρ_2 son el radio del objeto 1 y objeto 2, las variables ang_proy_1 y ang_proy_2 son los ángulos proyectados del objeto 1 y objeto 2.

4.5.2. Simulación realista de la detección de objetos

Con el fin de simular de forma realista la detección de objetos, se calculó la información sobre el rendimiento de un sistema específico de detección de objetos bajo diferentes distancias y ángulos de la cámara respecto a cuatro objetos.

Estas estadísticas se utilizaron para construir una LUT (*Look-Up Table*) de detección de objetos, que se utiliza en el simulador cada vez que el robot trata de detectar un objeto situado dentro del campo de visión.

Las estadísticas de la detección de objetos se calculan utilizando 40 imágenes capturadas bajo diferentes *puntos de vista*. Los *puntos de vista* corresponden a las distancias de la cámara a los objetos con 80, 120, 160, 200 y 250 [cm], y los ángulos *yaw* de la cámara al objeto con 0, 45, 90, 135, 180, 225, 270 y 315 grados (los ángulos de la cámara *pitch* y *roll* no son considerados). La representación gráfica de este procedimiento se puede ver en la Figura 36.

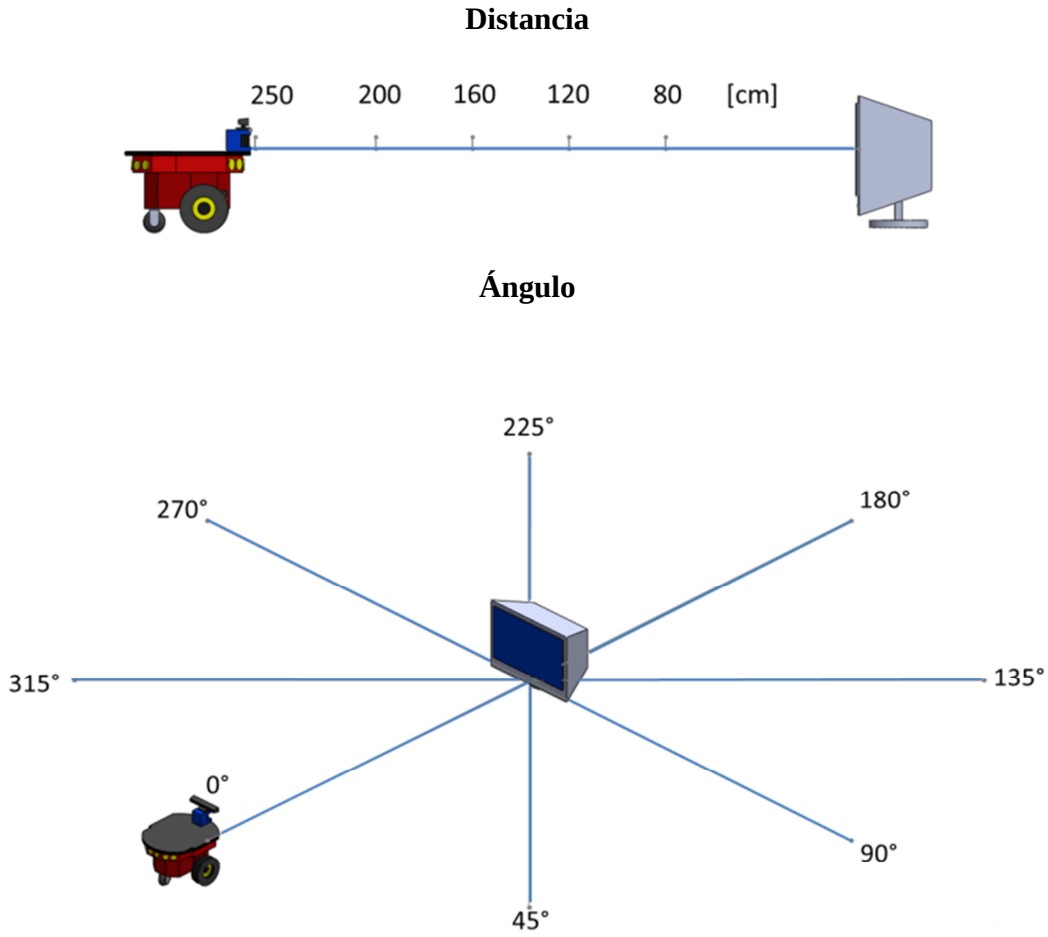


Figura 36. Puntos de vista con diferentes distancias y ángulos.

Para cada *punto de vista*, se capturan 100 imágenes de cada objeto mediante el uso de una secuencia de vídeo corto. En total se obtuvieron 4.000 imágenes de cada objeto.

También para cada *punto de vista*, se calcula una probabilidad de detección de cada objeto contando la cantidad de detecciones en las 100 imágenes disponibles. Se consideran cuatro objetos que son: "*monitor*", "*teclado*", "*cpu*", y "*router*".

Las probabilidades de detección se resumen en la Tabla 6 , y se codificaron en una LUT de detección de objetos.

Tabla 6. LUT de Probabilidad para detección de objetos.

	Monitor Distancias					Teclado Distancias				
Áng.	**80**	**120**	**160**	**200**	**250**	**80**	**120**	**160**	**200**	**250**
0°	0.33	0.40	0.57	0.67	0.84	0.34	0.62	0.80	0.64	0.59
45°	0.12	0.38	0.40	0.85	0.89	0.52	0.85	0.82	0.79	0.45
90°	0.49	0.57	0.67	0.76	0.81	0.05	0.28	0.26	0.64	0.79
135°	0.17	0.75	0.70	0.76	0.74	0.36	0.80	0.42	0.38	0.36
180°	0.45	0.48	0.58	0.59	0.67	0.41	0.43	0.64	0.50	0.33
225°	0.07	0.95	0.74	0.78	0.72	0.60	0.82	0.86	0.63	0.52
270°	0.38	0.55	0.64	0.71	0.84	0.06	0.15	0.25	0.63	0.75
315°	0.10	0.56	0.45	0.65	0.60	0.44	0.86	0.62	0.58	0.68

	Cpu Distancias					Router Distancias				
Áng.	**80**	**120**	**160**	**200**	**250**	**80**	**120**	**160**	**200**	**250**
0°	0.81	0.76	0.69	0.42	0.21	0.51	0.67	0.54	0.32	0.00
45°	0.15	0.31	0.45	0.39	0.22	0.24	0.39	0.62	0.41	0.17
90°	0.66	0.76	0.80	0.69	0.29	0.56	0.65	0.47	0.29	0.00
135°	0.12	0.35	0.49	0.31	0.24	0.52	0.49	0.39	0.33	0.15
180°	0.88	0.94	0.78	0.55	0.26	0.43	0.75	0.54	0.27	0.00
225°	0.11	0.39	0.40	0.35	0.19	0.48	0.46	0.37	0.33	0.13
270°	0.68	0.78	0.82	0.69	0.46	0.55	0.57	0.45	0.34	0.00
315°	0.20	0.32	0.44	0.38	0.25	0.47	0.58	0.48	0.43	0.14

4.6. Actuación

Luego del proceso de *estrategia de búsqueda*, en este módulo se realizará una acción sobre el robot (desplazamiento) para buscar el objeto principal. Una vez obtenido el punto y dirección donde el robot debe llegar, en este módulo se realiza la acción definida incluyendo un comportamiento reactivo usando el láser del robot para evitar colisionar con los obstáculos, paredes u objetos que se presenten en el ambiente.

Capítulo 5. EXPERIMENTOS Y RESULTADOS

5.1. Configuración de los experimentos

Con el fin de validar la metodología propuesta de búsqueda de objetos, se desarrolló una simulación realística de un robot en un ambiente 3D. Este ambiente está basado en el uso del simulador *Player/Stage* [49] e incorpora: (i) estadísticas sobre la coocurrencia de clases del objeto en el mundo real para calcular los parámetros de las máscaras de relaciones espaciales, (ii) un modelo para la observación de los objetos en 3D por el robot, y (iii) el uso de un sistema realista de detección de objetos, cuya probabilidad de detectar el objeto en cada *frame* depende de la pose relativa entre el objeto y la cámara del robot, así como la clase de objeto.

Los experimentos se realizaron en un conjunto de mapas, cada uno con cuatro objetos. Cada mapa contiene un objeto principal *A* (*monitor*), que es el objeto buscado, y tres objetos secundarios que son *B* (*teclado*), *C* (*cpu*), y *D* (*router*). El tamaño de los mapas es 10[m] x 10[m]. Las posiciones de los objetos en cada mapa se seleccionan eligiendo una posición aleatoria para el objeto principal *A*, luego, generando una posición aleatoria para los objetos *B*, *C*, y *D* siguiendo una distribución que representa las coocurrencias. Los objetos se colocan a 0,5[m] del suelo, y la altura de la cámara del robot es de 1[m]. Las orientaciones de los objetos en cada mapa se seleccionan al azar cuantizadas en múltiplos de 45 grados. Los 20 mapas se utilizan para realizar los experimentos, cada mapa es recorrido por el robot la misma cantidad de veces. Un sensor láser de *Player/Stage* se utiliza para evitar colisiones. Las observaciones de los objetos son obtenidas mediante el uso de las observaciones 3D descritas en el capítulo 4 de este trabajo. El campo de visión del sensor tiene un rango horizontal de 75 grados, un rango vertical de 45,6 grados, una profundidad mínima de detección de 0,3[m] y una profundidad máxima de 2,5 [m]. El objeto principal A, se puede detectar hasta 1 [m], y los objetos secundarios B, C, y D pueden ser detectados variando siete *escalas de vista* (EV) que van desde 1, 1.25, 1.50, 1.75, 2.0, 2.25 y 2.50 [m].

5.2. Resultado de los experimentos.

Once algoritmos de búsqueda de objetos son comparados en el mismo conjunto de mapas. Estos algoritmos son:

1. Búsqueda no informada (BNI). Se calcula un mapa de probabilidad $p(a_{i,j})$ para el objeto A mediante el uso de detecciones negativas del mismo objeto, luego el objeto A es buscado mediante los puntos de vista que maximizan la probabilidad que contengan al objeto A.

2. Búsqueda informada usando partículas (BIUP). El algoritmo de Aydemir et al. [12] se utiliza para la construcción de un mapa de probabilidad $p(a_{i,j})$ para el objeto *A* mediante el uso de detecciones negativas de ese objeto. Luego el objeto *A* es buscado mediante puntos de vista que maximicen la probabilidad de que contenga el objeto *A*. Cuando se detecta un objeto

secundario *B*, se genera un conjunto de 500 partículas alrededor de la detección dentro del campo de visión. Cada partícula representa una hipótesis acerca de la ubicación del objeto *A* dada la posición conocida del objeto *B* en el mapa. La relación espacial entre los objetos *A* y *B* induce una distribución de probabilidad de la presencia de A dada una posición conocida de B, que tiene un valor máximo en el mapa. Las partículas con valor de probabilidad asociada igual o mayor que la mitad de la máxima probabilidad se utilizan para seleccionar el siguiente campo de visión óptimo mediante la maximización de la cantidad de partículas aceptadas dentro del campo de visión.

3. Búsqueda informada utilizando convoluciones con información positiva y negativa con máscaras duras (BIC-PN-MD). Un mapa de probabilidad $p(a_{i,j})$ para el objeto principal *A* es estimado mediante el uso de detecciones negativas del objeto A, detecciones positivas y negativas de los objetos *B, C* y *D*, y máscaras de relación espacial. Luego el objeto *A* es buscado mediante los puntos de vista que maximizan la probabilidad de que el objeto esté dentro del campo de visión [36].

4. Búsqueda informada utilizando convoluciones con solo información positiva con máscaras duras (BIC-SP-MD). Similar al algoritmo 3, pero en este caso sólo se utilizan detecciones positivas de los objetos *B, C*, y *D*.

5. Búsqueda informada utilizando convoluciones con información positiva y negativa con máscaras suaves (BIC-PN-MS). Similar al algoritmo 3, pero en este caso se utilizan máscaras suaves de relación espacial.

6. Búsqueda informada utilizando convoluciones con información sólo positiva con máscaras suaves (BIC-SP-MS). Similar al algoritmo 4, pero en este caso se utilizan máscaras suaves de relación espacial.

7. Búsqueda activa explotando relaciones probabilísticas objeto-objeto (BAERP). Considera coocurrencias entre objetos y se obtiene una estrategia que permite el uso de una cadena de objetos intermedios mediante un enfoque probabilístico para la búsqueda de un objeto y el costo de viaje para alcanzar los objetos intermedios [35].

8. Búsqueda informada con convoluciones utilizando información positiva y negativa con máscaras duras y análisis del máximo global (BIC-PN-MD-AMG).

9. Búsqueda informada con convoluciones utilizando información sólo positiva con máscaras duras y análisis del máximo global (BIC-SP-MD-AMG). Es similar al algoritmo 8 pero usando solo la información positiva.

10. Búsqueda informada con convoluciones utilizando información positiva y negativa con máscaras suaves y análisis del máximo global (BIC-PN-MS-AMG). Es similar al algoritmo 8 pero en este caso se utilizan máscaras suaves de relación espacial.

11. Búsqueda informada con convoluciones utilizando información solo positiva con máscaras suaves y análisis del máximo global (BIC-SP-MS-AMG). Es similar al algoritmo 9 pero en este caso se utilizan máscaras suaves de relación espacial.

La tarea consiste en encontrar el objeto *A* antes de que se hayan procesado 1500 vistas. Para cada uno de los 11 modelos se realizan 200 experimentos.

Tabla 7. Resultados de los experimentos.

Métodos	**Tasa de detección (TD %) con diferentes escalas de vista (EV) [m]**						
	EV=1,0	**EV=1,25**	**EV=1,50**	**EV=1,75**	**EV=2,0**	**EV=2,25**	**EV=2,50**
BNI	28	26	38	50	58	65	65
BIUP	31	38	46	53	67	70	70
BIC-PN-MD	49	60	63	64	77	79	85
BIC-SP-MD	41	42	44	44	59	65	67
BIC-PN-MS	53	64	68	71	77	80	88
BIC-SP-MS	42	42	48	55	62	67	69
BAERP	51	63	71	76	73	78	83
BIC-PN-MD-AMG	**77**	**75**	**77**	**87**	**89**	**90**	**93**
BIC-SP-MD-AMG	70	70	76	84	84	87	89
BIC-PN-MS-AMG	**79**	**74**	**86**	**87**	**88**	**94**	**95**
BIC-SP-MS-AMG	74	76	82	82	87	88	91

También se realiza un análisis de sensibilidad con respecto a la capacidad de percepción, por ejemplo, una mejor cámara o mejor detección de objetos en mayores distancias. Para este efecto se utilizan siete *escalas de vista* (EV) diferentes, es decir que el sensor de detección (cámara) tiene los siguientes rangos: 1.0 , 1.25 , 1.50 , 1.75 , 2.0 , 2.25 y 2.50 [mt]. En total son 15.400 experimentos.

Como se puede ver los resultados de la Tabla 7, los métodos con **máscaras de relación suave** y la información con **detección positiva y negativa** son más efectivos. Por esta razón en la Figura 37 se muestran solo los ejemplos que tienen *mascaras suaves* y *detección positiva y negativa* y los demás algoritmos.

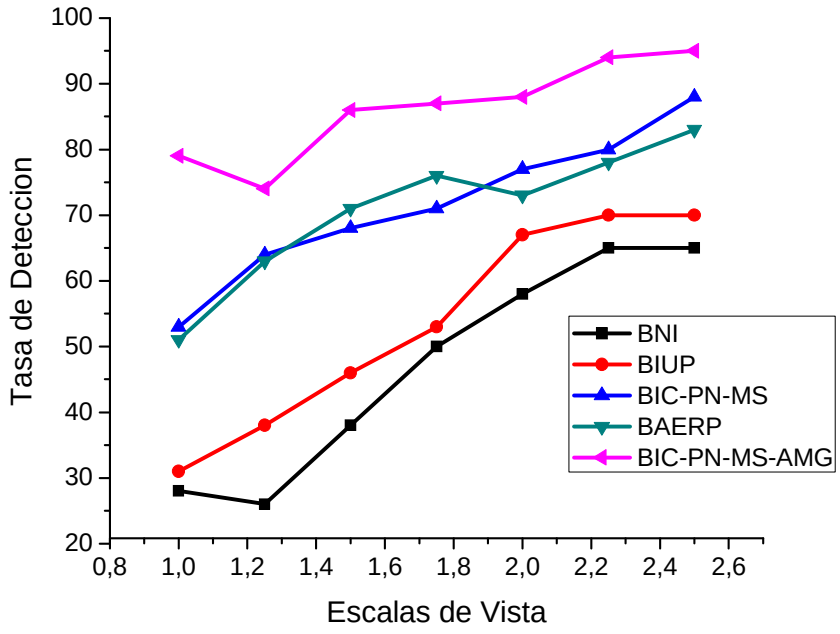


Figura 37. Efecto de la EV en diferentes algoritmos de búsqueda de objetos.

Se probó once modelos bajo las mismas condiciones midiendo una *tasa de detección* (TD). La TD se calcula utilizando 200 exploraciones por cada modelo y en cada *escala de vista* (EV). En la Figura 37 se puede observar que el algoritmo de búsqueda sin información (BNI) tiene porcentajes más bajos de detección, mientras que los 4 modelos propuestos en este trabajo de tesis, tienen mejores resultados, también se puede ver que la combinación de probabilidad negativa y positiva y el uso de máscaras suaves mejoran el rendimiento de la detección. Analizando la obtención del comportamiento de los métodos variando las EV se puede observar que usar una buena detección afecta considerablemente al rendimiento de todos los métodos.

A partir de los resultados, es evidente que el método con convoluciones y análisis del máximo global obtiene un mejor rendimiento que los demás métodos que no incluyen ese tipo de información. Esto sucede porque la información integrada puede utilizar diferentes métodos después del momento en que se ve el objeto. El uso de la información negativa mejora la TD, ya que la información sobre las no detecciones de objetos secundarios se agrega al mapa de probabilidad. El uso de máscaras suaves provoca una mejora respecto a la utilización de máscaras duras debido a que el mapa de probabilidad no tiene transiciones bruscas entre las zonas observadas y no observadas.

5.3. Escalado del tiempo de procesamiento con el número de objetos.

El método propuesto de búsqueda informada puede ser optimizado mediante la observación de las imágenes de verosimilitud cruzada son constantes, excepto alrededor de la zona del área de detección en el caso de las detecciones positivas, y alrededor del campo de visión en el caso de

detecciones negativas. Esto se debe a que la relación espacial entre dos objetos es constante cuando la distancia entre ellos es lo suficientemente grande. Teniendo en cuenta que el cálculo de las convoluciones en áreas de valor constante no es necesario, una focalización de las convoluciones puede ser aplicada en las áreas no constantes. La búsqueda informada utilizando información positiva y negativa escala de forma lineal con el número de objetos secundarios totales, ya que es necesaria una convolución para calcular verosimilitudes cruzadas en las detecciones positivas y negativas.

La búsqueda informada utilizando sólo información positiva escala de forma lineal con el número de objetos secundarios observados ya que necesita calcular solo las verosimilitudes cruzadas en las detecciones positivas. A medida que la cantidad media de los objetos detectados es baja, los métodos utilizan sólo información positiva corren tan rápido como la búsqueda no informada la mayor parte del tiempo. El tiempo de proceso (TP) de los métodos de búsqueda informada utilizando convoluciones escala cuadráticamente con el tamaño de la máscara cuando se utiliza convoluciones focalizadas, y escala linealmente con el tamaño de la máscara cuando se utiliza convoluciones con el mapa completo. El TP de las convoluciones focalizadas no depende del tamaño del mapa. En los experimentos, un pequeño mapa (100x100) y una máscara (32x32) se utilizaron para la representación de un entorno de 6x6[m]. En este caso, el TP del proceso de la búsqueda informada (0,2 segundos por convolución) es menor que el TP del proceso de detección de objetos.

Capítulo 6. CONCLUSIONES Y TRABAJO A FUTURO

En el presente trabajo se propone una nueva metodología para la búsqueda informada de objetos, esta metodología fue probada mediante una herramienta de simulación realista. La metodología está basada en la integración de la información proporcionada por los objetos secundarios en la distribución de probabilidad del objeto principal a ser encontrado. Las relaciones espaciales entre los objetos se estiman mediante el uso de un conjunto de relaciones espaciales básicas que se mezclan usando los valores de coocurrencia como pesos. Luego se divide el mapa en regiones y se obtiene el valor de la máxima utilidad de cada región. Esta metodología puede extenderse para mapas 3D, pero aumentaría de forma significativa el tiempo de procesamiento.

Se usan círculos concéntricos para poder modelar la distribución de las máscaras usando pocos ejemplo por lo que existe un supuesto de unimodalidad. Con una mayor cantidad de datos, se podría calcular la distribución de probabilidad directamente. El modelo de percepción considera factores para distintas poses y distancias del objeto y el robot. El trabajo a futuro incluye la realización de experimentos con un robot real en un ambiente controlado para la validación de los resultados en el mundo real.

También se pretende maximizar la información de probabilidad de encontrar el objeto principal tomando la posición de más de dos objetos secundarios como diferentes hipótesis, es decir si un objeto secundario indica que la mayor probabilidad es que se encuentre muy lejos y otro objeto secundario indica que la mayor probabilidad está cerca, combinar ambas hipótesis y llegar a un grado de creencia en común.

BIBLIOGRAFÍA

[1] S. Russell and P. Norvig, "Artificial intelligence: a modern approach," *Artificial Intelligence, Prentice-Hall, Englewood Cliffs*, vol. 25, 1995.

[2] S. Burlington and G. Dudek, "Spiral search as an eficient mobile robotic search technique," presented at the Centre for Intelligent Machines McGill University, Rue University, Montreal, Canada, 1999.

[3] D. Meger, P.-E. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. J. Little, D. G. Lowe, and B. Dow, "Curious George: An Attentive Semantic Robot," *IROS 2007 Workshop: From sensors to human spatial concepts*, 2007.

[4] S. Thrun, D. Fox, and W. Burgard, "Probabilistic mapping of an environment by a mobile robot," presented at the in: IEEE Int. Conf. on Robotics and Automation, 1998.

[5] H. Choset and K. Nagatani, "Topological simultaneous localization and mapping (SLAM): Toward exact localization without explicit localization," *IEEE Trans. on Robotics and Automation 17 (2),* pp. pp. 125–137, 2001.

[6] B. J. A. Kröse, N. Vlassis, R. Bunschoten, and Y. Motomura, "A probabilistic model for appearance-based robot localization," *Image and Vision Computing 19,* 2001.

[7] Per-Erik Forssén, David Meger, Kevin Lai, Scott Helmer, James J. Little, and D. G. Lowe, "Informed Visual Search: Combining Attention and Object Recognition," 2008.

[8] A. Krishnan and M. Krishna, "A Visual Exploration Algorithm using Semantic Cues that Constructs Image based Hybrid Maps," *presented at the IROS 2010 workshop: Semantic Mapping and Autonomous Knowledge Acquisition,* 2010.

[9] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernandez-Madrigal, and J. Gonzalez, "Multi-hierarchical semantic maps for mobile robotics," in *Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on*, 2005, pp. 2278-2283.

[10] P. Beeson, M. MacMahon, J. Modayil, A. Murarka, B. Kuipers, and B. Stankiewicz, "Integrating multiple representations of spatial knowledge for mapping, navigation, and communication," *Proceedings of the Symposium on Interaction Challenges for Intelligent Assistants,* 2007.

[11] K. Shubina and J. Tsotsos, "Visual search for an object in a 3d environment using a mobile robot," *Computer Vision and Image Understanding 114,* vol. 5, pp. 535-547, 2010.

[12] A. Aydemir, K. Sjöö, and P. Jensfelt, "Object search on a mobile robot using relational spatial information," in *Proc. of the 11th Int Conference on Intelligent Autonomous Systems (IAS-11)*, 2010.

[13] C. Galindo, J. A. Fernández-Madrigal, J. González, and A. Saffiotti, "Robot Task Planning using Semantic Maps " *Robotics and Autonomous Systems,* vol. 56, pp. 955-966, 2008.

[14] N. Nilsson, "Shakey the robot," *Tech. rep., No. 323, Artificial Intelligence Center, SRI International, Melon Park, CA*, 1984.

[15] M. Georgeff and A. Lansky, "Reactive reasoning and planning," *in: AAAI-87*, pp. 677-682, 1987.

[16] S. Vasudevan, S. Gachter, V. Nguyen, and R. Siegwart, "Cognitive maps for mobile robots-an object based approach," *Robot. Auton. Syst. ,* vol. 55, pp. 359-371, 2007.

[17] A. Martinelli, A. Tapus, K. O. Arras, and R. Siegwart, "Multi-resolution SLAM for real world navigation," *in: 11th International Symposium of Robotics Research, Siena, Italy,* 2003.

[18] B. Kuipers, "The Spatial Semantic Hierarchy," *Artificial Intelligence 119,* pp. 191-233, 2000.

[19] B. Krieg-Brückner, T. Röfer, H.-O. Carmesin, and R. Müller, "A Taxonomy of Spatial Knowledge for Navigation and its Application to the Bremen Autonomous Wheelchair," *in: Lecture Notes in Artificial Intelligence,* vol. 1404, pp. 373-397, 1998.

[20] D. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *Int'l J. Computer Vision,* vol. 2, pp. pp. 91-110, 2004.

[21] P. Viswanathan, D. Meger, T. Southey, J. J. Little, and A. Mackworth, "Automated Spatial-Semantic Modeling with Applications to Place Labeling and Informed Search," presented at the Canadian Conference on Computer and Robot Vision,, 2009.

[22] B. Russell, A. Torralba, K. Murphy, and W. Freeman, "Labelme: A database and web-based tool for image annotation," *International Journal of Computer Vision,* vol. 77, pp. 157-173, May 2008.

[23] T. D. Garvey, "Perceptual strategies for purposive vision," *Technical report, SRI International,* vol. 117, 1976.

[24] L. Wixson and D. Ballard, "Using intermediate object to improve efficiency of visual search," *Int. J. Comput. Vis. 18,* vol. 3, pp. 209–230, 1994.

[25] Y. Ye and J. K. Tsotsos, "Sensor Planning for 3D Object Search," *Computer Vision and Image Understanding,* vol. 73-2, pp. 145 - 168, 1999.

[26] Y.-S. Song and S.-B. Cho, "Objects Relationship Modeling for Improving Object Detection Using Bayesian Network Integration," presented at the International Conference on Intelligent Computing, Kunming, China, 2006.

[27] K. Sjö, D. López, P. Chandada, P. Jensfelt, and D. Kragic, "Object search and localization for an indoor mobile robot," *Journal of Computing and Information Technology 17,* pp. 67-80, 2009.

[28] Website:. *http://www.semantic-robot-vision-challenge.org/.*

[29] T. Kollar and N. Roy, "Utilizing object-object and object-scene context when planning to find things," presented at the in ICRA '09: Proceedings of the 2009 IEEE International Conference on Robotics and Automation, 2009.

[30] K. Sjöö, D. G. López, C. Paul, P. Jensfelt, and D. Kragic, "Object search and localization for an indoor mobile robot," *Journal of Computing and Information Technology,* vol. 17, pp. 67-80, 2009.

[31] J. K. Tsotsos and K. Shubina, "Attention and Visual Search : Active Robotic Vision Systems that Search," presented at the The 5th International Conference on Computer Vision Systems, Bielefeld, 2007.

[32] Kasper A., Jäkel R., and D. R., "Using spatial relations of objects in real world scenes for scene structuring and scene understanding," presented at the Proceedings of the 15th International Conference on Advanced Robotics 2011.

[33] L. Ziegler, F. Siepmann, M. Kortkamp, and S. Wachsmuth, "Towards an Informed Search Behavior for Domestic Robots," *Domestic Service Robots in the Real World,* 2010.

[34] A. Aydemir, K. Sjöö, J. Folkesson, A. Pronobis, and P. Jensfelt, "Search in the real world: Active visual object search based on spatial relations," presented at the Int. Conf. Robotics and Automation (ICRA), 2010.

[35] J. Elfring, S. Jansen, R. van de Molengraft, and M. Steinbuch, "Active object search exploiting probabilistic object-object relations," presented at the Proc. of the RoboCup Symposium 2013, Eindhoven, The Netherlands, 2013.
[36] J. Ruiz del Solar, P. Loncomilla, and M. Saavedra, "A Bayesian framework for informed search using convolutions between observation likelihoods and spatial relation masks," presented at the ICAR 2013, Montevideo, Uruguay, 2013.
[37] Á. E. Gómez Sánchez and D. I. Zamorano Acosta, "Visión Estereoscópica y Estimación de Pose para el Posicionamiento de un Brazo Robótico," MSc.Thesis, Departamento de Ingeniería Mecatrónica, Centro Nacional de Investigación y Desarrollo Tecnológico, Cuernavaca, 2008.
[38] F. Calderon Solorio, "Notas de Visión," Postgrado en Ing. Electrica, Universidad Michoacana de San Nicolás de Hidalgo UMSNH, Michoacan - Mexico, 2006.
[39] ROS. (2011). *Kinect Calibration / Technical*. Available: http://www.ros.org/wiki/kinect_calibration/technical
[40] J.R. Ruiz-Sarmiento, C. Galindo, J. Gonzalez-Jimenez, and J. L. Blanco, "Navegación Reactiva de un Robot Móvil usando Kinect," in *Robot 2011 (Robótica Experimental)*, Universidad de Sevilla, España, 2011.
[41] Barak Freedman, Alexander Shpunt, Meir Machline, and Y. Arieli, "Depth Mapping using Projected Patterns," 2010.
[42] PrimeSense. *http://www.primesense.com*.
[43] Website:. *https://www.flickr.com*.
[44] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," *CVPR*, vol. 1, pp. 886–893, 2005.
[45] J. O. Ludwig, D. Delgado, V. Gonçalves, and U. Nunes, "Trainable Classifier-Fusion Schemes: an Application to Pedestrian Detection," in *Intelligent Transportation Systems Conference, ITSC*, St. Louis, 2009.
[46] V. N. Vapnik, "The Nature of Statistical Learning Theory," *Springer-Verlag*, 1995.
[47] T. R. Balaguer, "Diseño e implementación de una capa de software para el control de robots mediante Player/Stage," M.Sc. Thesis, Departamento de Informática, Universidad Carlos III de Madrid, Madrid, 2010.
[48] J.M.Cañas, V.Matellán, and R. Montúfar, "Programación de robots móviles," *RIAI: Revista Iberoamericana de Automática e Informática Industrial*, vol. 3, pp. 99-110, April 2006.
[49] B. P. Gerkey, R. T. Vaughan, and A. Howard, "The Player/Stage project: tools for multi-robot and distributed sensor systems," presented at the In Proceedings of the 11th International Conference on Advanced Robotics ICAR'2003, Coimbra (Portugal).

ANEXOS

POSTERS Y *PAPERS* PRESENTADOS

Poster

Object search using semantic categories and spatial relations between objects

Marcelo Saavedra, Patricio Loncomilla and Javier Ruiz-del-Solar

Publicado en:

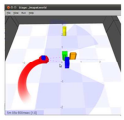
IEEE RAS Summer School on "Robot Vision and Applications"
VI Latin American Summer School on Robotics
Diciembre 2012, .Santiago, Chile.

OBJECT SEARCH USING SEMANTIC CATEGORIES AND SPATIAL RELATIONS BETWEEN OBJECTS

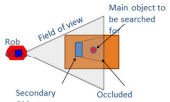
Marcelo Saavedra, Patricio Loncomilla and Javier Ruiz-del-Solar
Advanced Mining Technology Center, Universidad de Chile
msaavedra@ing.uchile.cl

I. INTRODUCTION

One of the tasks expected of a robot is "search and bring" objects. In this work, we focus on giving robots the ability to find objects using existing semantic relations such as "near" or "far," in a given environment.



Simulator Player/Stage



Occlusion of objects

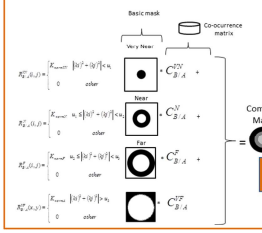
Robot Roby, Forbidden Planet (1956).

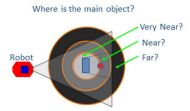
There are 3 important contributions reported in this work.
- The representation of spatial relations as spatial relation masks.
- Basic semantic categories .
- A methodology for computing a co-occurrence matrix associated with a set of basic relation masks.

II. METHODOLOGY

A. *Creating spatial relation masks from co-occurrences*

We focus on four simple spatial relations: "very near" (VN), "near" (N), "far" (F), and "very far" (VF).





The four coefficients $C^{VN}_{B/A}$, $C^{N}_{B/A}$, $C^{F}_{B/A}$ and $C^{VF}_{B/A}$ are called co-occurrences because they indicate the relative frequency of occurrence of a pair of objects for each spatial relation.

A complex mask can be created as a weighted sum of basic masks:

$$R_{B/A}(x,y) = C^{VN}_{B/A} R^{VN}_{B/A}(x,y) + C^{N}_{B/A} R^{N}_{B/A}(x,y) + C^{F}_{B/A} R^{F}_{B/A}(x,y) + C^{VF}_{B/A} R^{VF}_{B/A}(x,y)$$

II. METHODOLOGY

B. *Map update using spatial relation masks*

Spatial relations will be defined in a probabilistic sense: $Rel_{A,B}(\pi_A, \pi_B) = p(\pi_A \mid \pi_B)$

In our case, the robot is in a two-dimensional space. The problem addressed in this work is to find a main object A, by moving the robot appropriately. The search continues until the main object A, is found.

Cross-likelihood

Map update ← $p(a_{i,j} \mid z_B) = \dfrac{p(z_B \mid a_{i,j}) p(a_{i,j})}{\underbrace{p(z_B \mid a_\alpha) p(a_\alpha) + \sum_{i,j} p(z_B \mid a_{i,j}) p(a_{i,j})}_{\text{Evidence}}}$ → Previus map

Cone of detection or Field of view

$p(z_B \mid a_{i,j}) = p(z_B \mid b_{i,j}) * R_{B/A}(i,j)$ ← Spatial relation mask

$p(z_B \mid a_{i,j})$ can be implemented as a convolution in the (*i*, *j*) space between a likelihood image and a mask $R_{B/A}(i,j)$ describing the spatial relation between the main and secondary objects

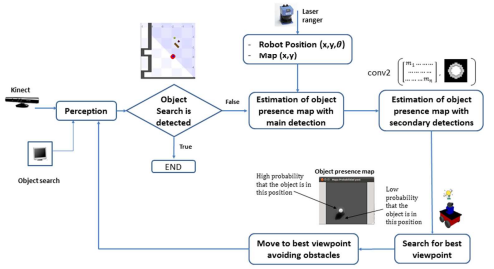
The optimal viewpoint is generated from : $\arg\max_{i=1..N} \sum_{j=1}^{n} \sum_{j=1}^{n} p(a_{i,j}) V(a_{i,j}, k)$

$V(a_{i,j}, k) = \begin{cases} 1 & \text{if } a_{i,j} \text{ is inside the } k^{th} \text{ view cone} \\ 0 & \text{otherwise} \end{cases}$

Where N is the number of candidate poses

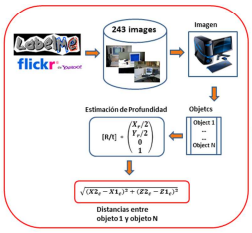
II. METHODOLOGY

Flowchart of object search process



III. RESULTS

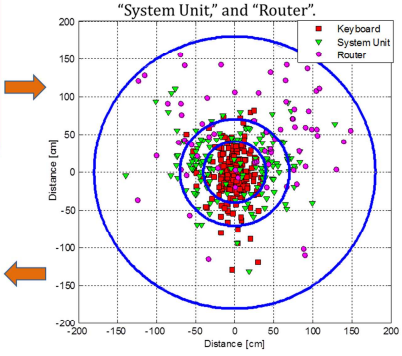
Creation of co-occurrence matrices



Final co-occurrences of objects around the object, "monitor".

Main object Monitor	Secondary objects co-occurrences [%]		
Semantic categories	Keyboard	System Unit	Router
Very Near	0.773	0.178	0.061
Near	0.143	0.491	0.151
Far	0.046	0.258	0.485
Very Far	0.038	0.074	0.303

Relative poses between object "Monitor" and objects "Keyboard," "System Unit," and "Router".



Comparison of object search algorithms

The search methods of objects that will be used to compare with the proposed method are:
- Search uninformed by maximizing likelihood.
- Informed search using particles.
- Spiral search.

Table of Results

Algorithm	Total	Found	%
Informed search using positive and negative information	701	421	60.1%
Informed search using only positive information	**701**	**449**	**64.0%**
Informed search using particles	701	219	31.2%
Uninformed search	701	292	41.7%
Spiral Search	701	380	54.2%

A total of 3505 experiments were executed for comparing the five algorithms

Paper A

Semantic object search using semantic categories and spatial relations between objects

Patricio Loncomilla, Marcelo Saavedra and Javier Ruiz-del-Solar

Publicado en:

RoboCup 2013: Robot World Cup XVII
Lecture Notes in Computer Science Volume 8371, 2014, pp 516-527
Julio 2013, Eindhoven, Holanda

Semantic object search using semantic categories and spatial relations between objects

Patricio Loncomilla, Marcelo Saavedra and Javier Ruiz-del-Solar
Advanced Mining Technology Center & Dept. of Electrical Engineering
Universidad de Chile, Chile
ploncomi@ing.uchile.cl

Abstract. In this work, a novel methodology for robots executing informed object search is proposed. It uses basic spatial relations, which are represented by simple-shaped probability distributions describing the spatial relations between objects in space. Complex spatial relations can be defined as weighted sums of basic spatial relations using co-occurrence matrices as weights. Spatial relation masks are an alternative representation defined by sampling spatial relation distributions over a grid. A Bayesian framework for informed object search using convolutions between observation likelihoods and spatial relation masks is also provided. A set of spatial relation masks for the objects “monitor”, “keyboard”, “system unit” and “router” were estimated by using images from Label-Me and Flickr. A total of 4,320 experiments comparing six object search algorithms were realized by using the simulator *Player/Stage*. Results show that the use of the proposed methodology has a detection rate of 73.9% that is more than the double of the detection rate of previous informed object search methods.

Keywords: Semantic search, Informed search, Object search, Spatial relations, Co-occurrence matrix

1 Introduction

Object search is an important ability for a mobile robot. Some previous work on object search are based on the use of spatial object-place relations, assuming that in a scene the searched object will be readily available to the robot's field of vision. However, there are sets of objects inside a setting that tend to appear near of each other as they have a particular spatial relation, making it possible to infer the existence of an object A, given an object B. For example, when looking inside offices, it can be noted that the object "keyboard" is often very near the object "monitor". For a human, the ability to deduce that object A tends to be "near", "very near" or "far" from object B is a trivial task, since human beings can learn spatial relations between objects by observing a large number of similar settings. In this paper, we focus on giving robots the ability to find objects using existing semantic relations with other objects such as “near” or “far,” in a given environment.

We conducted an exhaustive analysis of spatial relations between real-world objects that share a common use. For this objective, we created a database of spatial co-occurrences where distance values between objects are represented by the linguistic variables "very near," "near," "far," and "very far". We say that two objects "co-occur" when they appear together in several images showing a particular spatial relation.

The main contribution of this work is the use of convolutions for computing the probability of the presence of an object from positive and negative detections of all the objects on the map in a unified way. There are two secondary contributions. The first one is the representation of spatial relations as spatial relation masks, and the use of basic semantic categories such as "very near," "near," "far," and "very far" for generating basic spatial relation masks that can be combined to generate complex spatial relation masks by using a set of weights named co-occurrence values.

The second contribution is the creation of a methodology for computing a co-occurrence matrix associated with a set of basic relation masks from a database of labeled images containing real world objects.

This paper is organized as follows. In Section 2 some related work is presented. In Section 3 the proposed methodology for object search is described. In Section 4 an experimental validation of the methodology is presented. Finally, in Section 5 some conclusions of this work are given.

2 Related Work

The search for objects in a real environment is a very complex task for robots. Garvey in [1] recognized this problem and proposed the idea of indirect search, i.e. searching for another intermediate object that maintains a particular spatial relation with the object being searched for. Wixson et al. materialize the idea of indirect search, demonstrating greater efficiency both theoretically and empirically [2]. But the problem with indirect search is that the spatial relationship between the object sought and the intermediate object does not always exist. Furthermore, the detection of the intermediate object may be more difficult than the detection of the desired/primary object itself. In fact, Ye and Tsotsos demonstrated that the search for an arbitrary object in 3D space is NP-complete [3]. Shubina and Tsotsos propose an algorithm that considers the cost and effect of different actions with different types of prior knowledge and different spatial relations between objects [4]. Kollar et al. perform the search for an object in a known map of the environment, using object-object and object-scene context [5]. They obtain the co-occurrence of the existence of objects in two-dimensional images for learning correlations between object categories, and between objects and place labels (semantic labels such as “kitchen”). These images were taken from the Flickr website, and they do not take the distances between objects into consideration. Viswanathan et al. propose an approach using existing resources: common-sense knowledge of machine learning of object relations [6]. They use marked images from the LabelMe database, designed by Russell et al. [7]. They train an automatic classifier of places based on the presence of the detected objects to infer the probability that the other objects exist, and the kind of place (e.g., kitchen or office) is seen in the setting. Kasper et al. perform a study on spatial relations in three-dimensional images using a Kinect sensor [8]. They created a database using nine different office space settings with a total of 168 objects in 35 object classes. Then, they found the distances between different objects. They also made predictions about the location of unfound objects by detecting their surrounding objects.

Galindo et al. performed a study based on 2D data, combining metric, topological, and semantic aspects on a map [9]. In addition, they proposed a method for learning these semantic representations from sensory data. Vasudevan et al. attempted to create a spatial representation in terms of objects, by encoding typical household objects and doors within a hierarchical probabilistic framework [10]. They used a SIFT [11] based object recognition system and a door detection system based on lines extracted from range scans. They also proposed a conceptualization of different places, based on the objects that were observed inside them.

Aydemir et al. developed a method for object search using explicit spatial relationships between objects in order to perform an efficient visual search [12]. They presented a computational model using several random views to guide the robot's camera to the points where the objects have a high probability of being found by using the spatial-relation term "on" between objects, in an indoor environment since the objects are mostly on horizontal surfaces. The work presented in this paper is an extension and improvement of the model proposed by Aydemir in [12].

3 Methodology

3.1 Map update using spatial relation masks

The proposed methodology is designed for finding an object using informed search, i.e., by using information about other objects, which have a spatial relation with the object to be found. In [12] spatial relations are defined as functions from a space of a pair of poses π_A, π_B of two objects to the interval [0,1], where 1 indicates that the relation is completely fulfilled by these pose combinations, and 0 that the relation does not apply at all:

$$Rel_{A,B} : \{\pi_A, \pi_B\} \rightarrow [0,1] \tag{1}$$

In this work spatial relations will be defined in a probabilistic sense. A spatial relation between two objects is defined as the probability distribution of the pose of the first object given the known pose of the second object:

$$Rel_{A,B}(\pi_A, \pi_B) = p(\pi_A \mid \pi_B) \tag{2}$$

As the relation is treated as a probability distribution, the sum over all possible poses of A for a fixed pose of B is equal to one:

$$\int_{\pi_A} p(\pi_A \mid \pi_B) = 1 \tag{3}$$

If the spatial relation is invariant to translations and rotations, i.e. it only depends on the relative pose $\pi_{A/B}$ of object A with respect to object B, then the expression for the probability can be rewritten as:

$$Rel_{A/B}(\pi_{A/B}) = p(\pi_{A/B}) \tag{4}$$

In our case, the robot is in a two-dimensional space parameterized by using coordinates (x,y). The space is quantized into squared cells with size k, and parameterized by using indexes (i,j). By using the index notation, a spatial relation can be written as:

$$R_{A/B}(i, j) = K_{norm} * Rel_{A/B}(ki, kj) \tag{5}$$

$$\sum_i \sum_j R_{A/B}(i, j) = 1 \tag{6}$$

where K_{norm} is a normalizing constant.

The term $p(a_{i,j})$ represents the probability that the center of the main object A is in the cell (i,j). Both positive and negative observations z_A provide valuable information for the object search process, and can be used to compute an updated probability $p(a_{i,j}|z_A)$. The probability $p(a_0)$ is treated as a special case, and it represents the probability of the object being outside the search region.

Positive detections z_A=*true* provide information about the places where the object has high probability of being, by means of a likelihood $p(z_A$=*true*$|a_{i,j})$, which is defined over the cell (i,j). The likelihood has a high value over the cell where the object was detected, and a low value in the other cells. Negative detections z_A=*false* provide information $p(z_A$=*false*$|a_{i,j})$ about the cells where the objects have low probability of being, which are those cells visible from the current viewpoint that have a low probability of containing the object.

The problem addressed in this work is to find a main object, A, by moving the robot appropriately. The robot search process is applied until the main object, A, is found. In consequence, the search process includes only negative detections of the main object, A, before the object is found. Two cases are considered:

$$p(a_{i,j} \mid z_A = false) = \frac{p(z_A = false \mid a_{i,j})p(a_{i,j})}{p(a_o) + \sum_{i,j} p(z_A = false \mid a_{i,j})p(a_{i,j})} \tag{7}$$

$$p(a_0 \mid z_A = false) = \frac{p(a_0)}{p(a_o) + \sum_{i,j} p(z_A = false \mid a_{i,j})p(a_{i,j})} \tag{8}$$

The secondary object, B, can produce positive and negative detections z_B, which can be used to compute an updated probability:

$$p(a_{i,j} \mid z_B) = \frac{p(z_B \mid a_{i,j})p(a_{i,j})}{p(z_B \mid a_o)p(a_0) + \sum_{i,j} p(z_B \mid a_{i,j})P(a_{i,j})} \tag{9}$$

$$p(a_0 \mid z_B) = \frac{p(z_B \mid a_o)p(a_0)}{p(z_B \mid a_o)p(a_0) + \sum_{i,j} p(z_B \mid a_{i,j})P(a_{i,j})} \tag{10}$$

The terms $p(z_B|a_{i,j})$ and $p(z_B|a_0)$ will be called cross-likelihoods, as they relate the detection of a secondary object, B, with the presence of the main object, A, on the map. These probabilities can be derived by considering probabilities $p(b_{u,v})$ for the presence of a secondary object, B, at locations (u,v) in the grid:

$$p(z_B \mid a_{i,j}) = \sum_u \sum_v p(z_B \mid b_{u,v})p(b_{u,v} \mid a_{i,j}) \tag{11}$$

$$p(z_B \mid a_0) = \sum_u \sum_v p(z_B \mid b_{u,v})p(b_{u,v} \mid a_0) \tag{12}$$

The term $p(b_{uv}|a_0)$ is considered a constant over (u,v) whose sum has a value of 1 because B is supposed to be on the map. The term $p(b_{u,v} \mid a_{i,j})$ corresponds to the spatial relation between the main object, A, at location (i, j) and a secondary object, B, at location (u, v). By replacing this term with the spatial relation $R_{B/A}$, there is no need for storing a map for the secondary object; only the map for the main object and the likelihoods of the detections of the secondary object are needed:

$$p(z_B \mid a_{i,j}) = \sum_u \sum_v p(z_B \mid b_{u,v})R_{B/A}(u-i, j-v) \tag{13}$$

$$p(z_B \mid a_0) = \frac{1}{n_U n_V} \sum_u \sum_v p(z_B \mid b_{u,v}) \tag{14}$$

where $n_U n_V$ is the size of the map.

Equation (13) can be implemented as a convolution in the (*i,j*) space between a likelihood image and a mask $R_{B/A}(i,j)$ describing the spatial relation between the main and secondary objects, which will be named a *spatial relation mask*:

$$p(z_B \mid a_{i,j}) = p(z_B \mid b_{i,j}) * R_{B/A}(i,j) \tag{15}$$

The proposed system is highly versatile because any spatial relation can be represented by an appropriate mask. It must be noted that extra secondary objects can be added to the system by creating additional spatial relation masks. In case these relations are chained, as an example object A is near B, and object B is near C, then the mask of the chained relation can be obtained by convolution of the original masks:

$$P(z_C \mid a_{i,j}) = P(z_C \mid b_{i,j}) * R_{B/A}(i,j) \tag{16}$$

$$P(z_C \mid a_{i,j}) = P(z_C \mid c_{i,j}) * R_{C/B}(i,j) * R_{B/A}(i,j) \tag{17}$$

$$\Rightarrow R_{C/A} = R_{C/B} * R_{B/A} \tag{18}$$

A path for searching for the object can be created by generating optimal viewpoints at each iteration. The optimal viewpoint is generated from a set of *k* random poses reachable in a fixed time, and selecting the one that maximizes the probability of finding the object in the visible area, as shown in [8]:

$$\arg\max_{k=1..N} \sum_{i=1}^{n} \sum_{j=1}^{n} p(a_{i,j}) V(a_{i,j}, k) \tag{19}$$

where *N* is the number of candidate poses, and $V(a_{i,j}, k)$ is defined as:

$$V(a_{i,j}, k) = \begin{cases} 1, & \text{if } a_{i,j} \text{ is inside the k}^{th} \text{ view cone} \\ 0 & \text{otherwise} \end{cases} \tag{20}$$

The navigation algorithm has an impact on the reliability of the object search algorithm. For generating random poses, random sets of parameters for the navigation algorithm must be selected. Simple navigation strategies like going forward and rotating have a easily computable navigation time for a given final pose, and more complex navigation algorithms are not easily parameterized, then there are a trade-off between simplicity of the navigation algorithm (which allows simple and accurate generation of trajectories) and complex navigation approaches (which enable the robot to reach better poses in a given time).

For computing the convolutions, a way of generating appropriate masks from examples of real world images is needed. This will be explained in the following section.

3.2 Creating spatial relation masks from co-occurrences

In this section, a procedure for approximating a complex spatial relation as a weighted sum of basic spatial relations is presented. Each basic spatial relation corresponds to a semantic category meaningful to humans. In this work, we focus on four simple spatial relations: “very near” (VN), “near” (N), “far” (F), and “very far” (VF). The use of these spatial relations is useful as it enables the system to estimate a set of basic probability distributions from samples of relative positions of the objects in the real world. The masks for each of the spatial relations are defined in two versions, *hard masks* and *soft masks*. Hard masks are defined by two thresholds and have a rectangular profile, while soft masks are defined by four numbers and have a trapezoid-shaped profile. Each basic mask is normalized to sum one over all of the cells on the map, thus a normalization constant is added to the formulas.

Equations for hard masks are defined in equations (21) to (25).

$$R_{B/Ahard}(i,j;a_1,a_2) = K * \begin{cases} 1 & a_1 \le \sqrt{(ki)^2+(kj)^2} < a_2 \\ 0 & other \end{cases} \tag{21}$$

$$R^{VN}_{B/Ahard}(i,j) = R_{B/Ahard}(i,j;0,u_1) \tag{22}$$

$$R^{N}_{B/Ahard}(i,j) = R_{B/Ahard}(i,j;u_1,u_2) \tag{23}$$

$$R^{F}_{B/Ahard}(i,j) = R_{B/Ahard}(i,j;u_2,u_3) \tag{24}$$

$$R^{VF}_{B/Ahard}(i,j) = R_{B/Ahard}(i,j;u_3,\infty) \tag{25}$$

Equations for soft masks are similar, but have a smooth transition between values 0 and 1, that is regulated by a gap parameter □. The equations for soft masks are shown in equations (26) to (30).

$$R_{B/Asoft}(i,j;a_1,a_2,a_3,a_4) = K * \begin{cases} \dfrac{\sqrt{(ki)^2+(kj)^2}-a_1}{a_2-a_1} & a_1 \le \sqrt{(ki)^2+(kj)^2} < a_2 \\ 1 & a_2 \le \sqrt{(ki)^2+(kj)^2} < a_3 \\ \dfrac{a_4-\sqrt{(ki)^2+(kj)^2}}{a_4-a_3} & a_3 \le \sqrt{(ki)^2+(kj)^2} < a_4 \\ 0 & other \end{cases} \tag{26}$$

$$R^{VN}_{B/Asoft}(i,j) = R_{B/Asoft}(i,j;0,0,u_1-\delta,u_1+\delta) \tag{27}$$

$$R^{N}_{B/Asoft}(i,j) = R_{B/Asoft}(i,j;u_1-\delta,u_1+\delta,u_2-\delta,u_2+\delta) \tag{28}$$

$$R^{F}_{B/Asoft}(i,j) = R_{B/Asoft}(i,j;u_2-\delta,u_2+\delta,u_3-\delta,u_3+\delta) \tag{29}$$

$$R^{VF}_{B/Ahard}(i,j) = R_{B/Ahard}(i,j;u_3-\delta,u_3+\delta,\infty,\infty) \tag{30}$$

In this work, basic masks defined by a circle of radius u_1 in the case of “very near”, a circular ring of radii u_1 and u_2 in the case of “near,” a circular ring of radii u_2 and u_3 in the case of “far,” and a circular ring of internal radius u_3 and an external radius that cover the whole map in the case of “very far”. The radius values are selected by considering statistics of the distances between objects A and B, and by modeling their selection process as a classification problem.

Thus, the optimal radius value between two categories, e.g., "near" and "far", is the one that generates the same mean classification error in both classes.

A complex mask can be created as a weighted sum of basic hard or soft masks:

$$R_{B/A}(x,y) = C^{VN}_{B/A} R^{VN}_{B/A}(x,y) + C^{N}_{B/A} R^{N}_{B/A}(x,y) + C^{F}_{B/A} R^{F}_{B/A}(x,y) + C^{VF}_{B/A} R^{VF}_{B/A}(x,y) \quad (31)$$

An example of a complex mask sampled over a grid with pixel size 0.1[m] is shown in Figure 1. The four coefficients $C^{VN}_{B/A}$, $C^{N}_{B/A}$, $C^{F}_{B/A}$ and $C^{VF}_{B/A}$ are called co-occurrences because they indicate the relative frequency of occurrence of a pair of objects for each spatial relation. They can be constructed from samples of positions of both objects by computing the number of occurrences of each basic spatial relation. If a set of samples is divided into basic semantic categories and the count is n_{VN} for "very near," n_N for "near," n_F for "far," and n_{VF} for "very far," the co-occurrences can be computed by using equations (32) to (35):

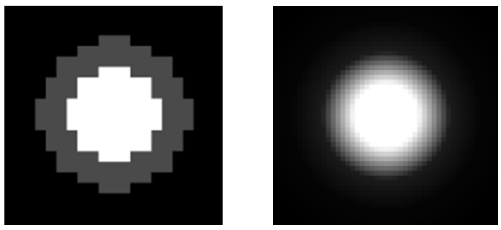
$$C^{VN}_{B/A} = \frac{n_{VN}}{n_{VN} + n_N + n_F + n_{VF}} \quad (32)$$

$$C^{N}_{B/A} = \frac{n_N}{n_{VN} + n_N + n_F + n_{VF}} \quad (33)$$

$$C^{F}_{B/A} = \frac{n_F}{n_{VN} + n_N + n_F + n_{VF}} \quad (34)$$

$$C^{VF}_{B/A} = \frac{n_{VF}}{n_{VN} + n_N + n_F + n_{VF}} \quad (35)$$

A co-occurrence matrix is the set of co-occurrence values of two or more objects with a particular spatial relation.



(a) (b)

Fig. 1. (a) Hard complex mask composed by summing hard masks "very near," "near," and "far," sampled over a grid. (b) The equivalent soft complex mask .

4 Results

4.1 Experimental setup

In order to characterize the proposed object search methodology and to compare it with existing methodologies, a simulation environment that uses data of real world objects was developed. We believe that a simulation environment is an appropriate tool to evaluate object search algorithms, because it allows obtaining repeatable experiments and to better quantify the performance of the

different methodologies. Experiments were performed on maps containing four objects, by using the *Player/Stage*, a robot simulation tool [13]. Each map contains a main object, named A (monitor), to be searched for, and three secondary objects named B (keyboard), C (system unit), and D (router). The size of each map is 6[mt] x 6[mt]. A total of 20 maps were created by picking a random position for the main object, A, and then picking a random position for the objects B, C, and D following a distribution that represents the co-occurrences shown in Table 1. The 20 maps are used to perform the experiments, each map being used the same number of times as the others. A laser sensor from *Player/Stage* is used for avoiding collisions. Observations of the objects are obtained by using a fiducial sensor included in Player/Stage that is able to measure the position of a detected object inside a view cone whose size depends on the object: the main object, A, can be detected up to 1[mt], and the secondary objects, B, C, and D can be detected up to 2[mt]. In each experiment the goal is to find the main object A before 1,500 views have been processed. In the search process, only negative detections of the main object need to be processed because a positive detection causes the search process to finish. Six algorithms of object search are compared on the same set of maps. The first 4 correspond to different variants of the proposed methodology, the fifth algorithm is the one proposed by Aydemir in [12], and the sixth correspond to the baseline algorithm where no information from secondary objects is used. The algorithms are; *Informed search with convolutions using positive and negative information with hard masks*: A probability map $P(a_{i,j})$ for the main object A is estimated by using positive and negative detections of objects B, C, and D, negative detections of object A, and spatial relation masks. Then object A is searched by finding viewpoints that maximize the probability of containing it. The relation masks $R_{A/B}$, $R_{A/C}$ and $R_{A/D}$ needed for updating the probability map $P(a_{i,j})$ from detections of secondary objects are constructed by using co-occurrence matrices. *Informed search with convolutions using only positive information with hard masks*: Similar to algorithm 1, but in this case only positive detections of objects B, C, and D are used. *Informed search with convolutions using positive and negative information with soft masks*: Similar to algorithm 1, but in this case soft spatial relation masks are used. *Informed search with convolutions using only positive information with soft masks*: Similar to algorithm 2, but in this case soft spatial relation masks are used. *Informed search using particles*: The algorithm of Aydemir [12] is used for constructing a probability map $P(a_{i,j})$ for object A by using negative detections of that object. Then object A is searched for by finding viewpoints that maximize the probability of containing A. When a secondary object is detected, a set of particles is generated around the detection inside the current view cone, and the ones which fulfill the spatial relation computed from Table 1 are used to select the next optimal viewpoint. A spatial relation is considered fulfilled when its current value is equal or greater than half of its maximum possible value. *Uninformed search*: A probability map $P(a_{i,j})$ for object A is estimated by using negative detections of that object, then object A is searched for by finding viewpoints that maximize the probability of containing A. This is the baseline algorithm used in [12], and no information from secondary objects is used.

4.2 Creation of co-occurrence matrices.

A set of 243 images from LabelMe [7] and captions of photos on the Flickr website were used for generating co-occurrence matrices. In each image, instances of the objects "monitor," "system unit," "keyboard," and "router" were labeled. As the sizes of the objects and the parameters of the camera are known, it is possible to compute the pose of each object in space. Several instances of the objects on the set of images and their poses were used to construct co-occurrence matrices for the categories "very near," "near," "far," and "very far" for each of the objects with respect to the others. Given the poses of a pair of objects, a distance was computed and used for selecting whether the sample belongs to the categories "very near," "near," or "far". If an object is detected alone in an image, the sample belongs to the category "very far". Only the depth and horizontal axis were used to compute the distances, as differences in the vertical direction do not affect the position of the object when it is transformed onto the 2D grid. The statistics of the distances between the object "monitor" and the objects "keyboard," "system unit," and "router," as well as the delimitation between the basic spatial relations are shown in Figure 2.

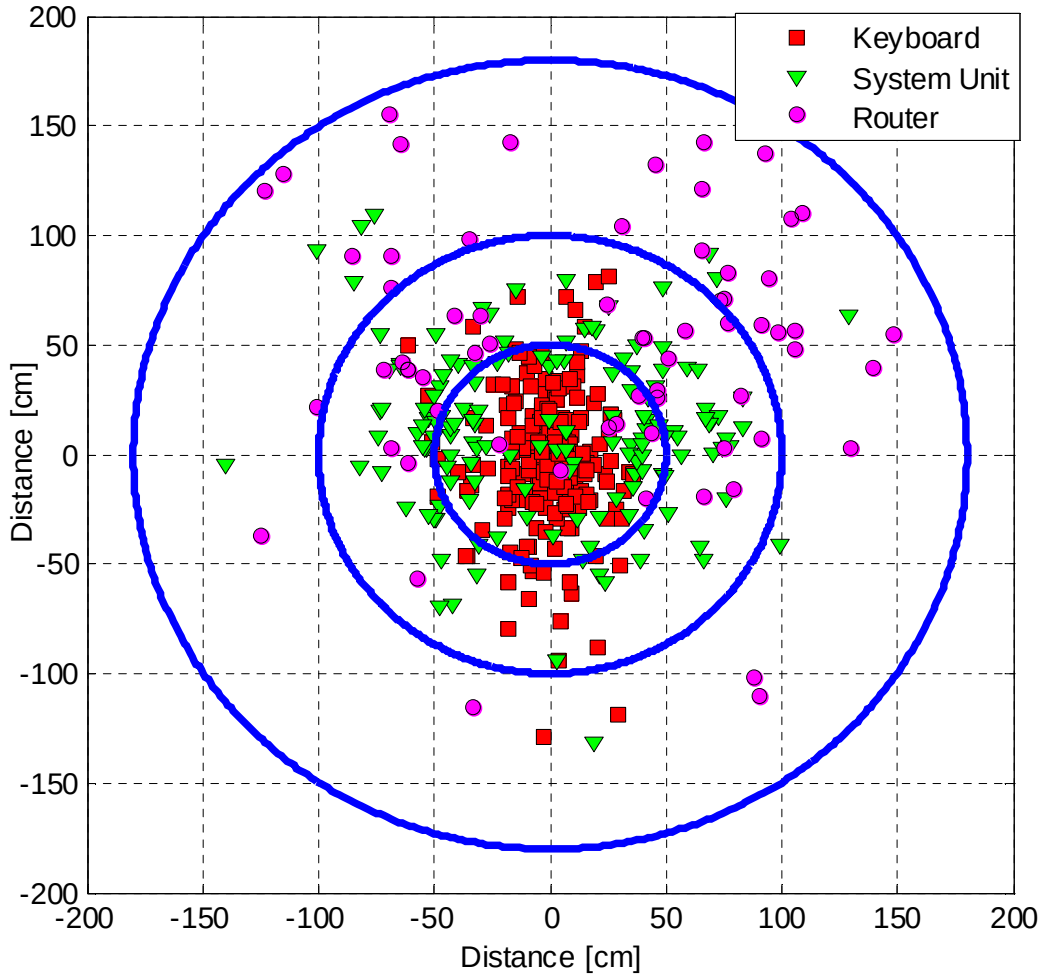


Fig. 2. Relative poses between object "monitor" and objects "keyboard," "system unit," and "router". Thresholds for basic spatial relations are shown as circles.

The final co-occurrences for the object, Monitor, as the main object are shown in Table_1.

Table 1. Final co-occurrences of objects around the object "monitor".

Main object Monitor	**Secondary objects**		
Semantic categories	***keyboard***	***system unit***	***router***
Very Near	0.773	0.178	0.061
Near	0.143	0.491	0.151
Far	0.046	0.258	0.485
Very Far	0.038	0.074	0.303

The basic hard and soft masks are defined by equations (21)-(25) and (26)-(30), respectively. The thresholds that separate the categories very near, near, far and very far are u_1=60[cm], u_2=100[cm] and u_3=150[cm]. The gap parameter for the soft masks is d=20[cm].

4.3 Experiments

A total of 4,320 experiment trials were executed for comparing the six search algorithms explained in Section 4.1. For each algorithm, a total of 720 experiments considering different maps and different initial robot poses were executed. Each experiment is considered successful if the object is found before 1500 sensor frames. The results from the experiments are shown in Table 2.

Table 2. Results from the experiments comparing six variants of object search algorithms. In each variant, a total of 720 experiments were performed. Each experiment is successful if the main object is found before 1,500 sensor frames.

Algorithm	**Number of searches performed**	**Successful searches**	**Detection rate**
Informed search using positive and negative information and hard masks	720	496	68.9%
Informed search using only positive information and hard masks	720	357	49.6%
Informed search using positive and negative information and soft masks	720	532	73.9%
Informed search using only positive information and soft masks	720	383	53.2%
Informed search using particles(Aydemir et al. [12])	720	248	34.4%
Uninformed search	720	208	28.9%

From the results of the experiments, it is evident that the algorithms that use convolutions for integrating information about secondary objects into the probability distribution of the main object have the best performance. This happens because the integrated information can be used several frames after the moment when the object was seen. The integration of positive detections of secondary objects using hard masks improves the detection rate from 28.9% up to 49.6%. The use of soft masks is also a factor that improves the detection rate up to a 53.2%. Finally, the use

of positive and negative information, in addition to soft masks, generates an important improvement in the detection rate that rises up to a 73.9%. The integration of information using masks, the comparison between different kind of masks and the ability of using negative information about secondary objects are all contributions of this work. Aydemir's particle based informed search algorithm [12] performs better than the baseline; however, the best algorithm described in this paper has a detection rate that is more than the double of the detection rate of Aydemir's algorithm.

4.4 Scaling of processing time with the number of objects

The methods can be optimized by observing that the cross-likelihood images are constant except on the detection area in the case of positive detections, and on the view cone in the case of negative detections. Then, focalized convolutions can be applied on these areas. Informed search using positive and negative information scales linearly with the number of total secondary objects. Informed search using only positive information scales linearly with the number of observed secondary objects. As the mean amount of detected objects is low, the methods that use only positive information run as fast as uninformed search the most of the time. Both kind of methods scale quadratically with the size of the mask when using focalized convolutions, and they scale linearly with the size of the mask when using convolutions with the full map. Time of focalized convolutions do not depend on the size of the map. In the experiments, a small map (100x100) and mask (32x32) were used for representing a 6m x 6m environment and the processing time is lower than the processing times of object detection algorithms.

5 Conclusions

In this work, a novel methodology for performing informed search of objects was proposed and tested. The methodology is based on integrating information provided by secondary objects into the probability distribution of the main object to be found. Spatial relations between objects are estimated by using a set of basic spatial relations which are mixed by using co-occurrence values as weights. Six algorithms of object search were compared by using spatial relations estimated from real-world data by performing a total of 4,320 simulations in *player*/*stage*, that include a previous search algorithm that uses spatial relations [12]. The results show that the detection rate of the search process increases from 28.9% to 73.9% when integrating positive and negative detections from the secondary objects into the probability distribution of the main object using soft masks. The integration of positive and negative detections of secondary objects, as well as the use of soft masks increases the detection rate. The obtained detection rate is more than the double of the one obtained by previous informed search algorithms.

Future work includes the creation of a full tridimensional model of the system, management of false detections, comparison with extra object search methods and the realization of experiments with a real robot for validating the results in the real world.

References

[1] T. D. Garvey, "Perceptual strategies for purposive vision," Technical report, SRI International, vol. 117, 1976.

[2] L. Wixson and D. Ballard, "Using intermediate object to improve efficiency of visual search," Int. J. Comput. Vis. 18, vol. 3, pp. 209–230, 1994.

[3] Y. Ye and J. K. Tsotsos, "Sensor Planning for 3D Object Search," Computer Vision and Image Understanding, vol. 73-2, pp. 145 - 168, 1999.

[4] K. Shubina and J. Tsotsos, "Visual search for an object in a 3d environment using a mobile robot," Computer Vision and Image Understanding 114, vol. 5, pp. 535-547, 2010.

[5] T. Kollar and N. Roy, "Utilizing object-object and object-scene context when planning to find things," presented at the in ICRA '09: Proceedings of the 2009 IEEE International Conference on Robotics and Automation, 2009.

[6] P. Viswanathan, D. Meger, T. Southey, J. J. Little, and A. Mackworth, "Automated Spatial-Semantic Modeling with Applications to Place Labeling and Informed Search," presented at the Canadian Conference on Computer and Robot Vision, 2009.

[7] B. Russell, A. Torralba, K. Murphy, and W. Freeman, "Labelme: A database and web-based tool for image annotation," International Journal of Computer Vision, vol. 77, pp. 157-173, May 2008.

[8] A. Kasper, R. Jäkel, and R. Dillmann, "Using spatial relations of objects in real world scenes for scene structuring and scene understanding," presented at the Advanced Robotics (ICAR), 2011 15th International Conference on, Tallinn, Estonia, 2011.

[9] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernandez-Madrigal, and J. Gonzalez, "Multi-hierarchical semantic maps for mobile robotics," in Intelligent Robots and Systems, 2005. (IROS 2005). 2005 IEEE/RSJ International Conference on, 2005, pp. 2278-2283.

[10] S. Vasudevan, S. Gachter, V. Nguyen, and R. Siegwart, "Cognitive maps for mobile robots-an object based approach," Robot. Auton. Syst. , vol. 55, pp. 359-371, 2007.

[11] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," International Journal of Computer Vision, vol. 60, pp. 91-110, 2004.

[12] A. Aydemir, K. Sjöö, and P. Jensfelt, "Object search on a mobile robot using relational spatial information," in Proc. of the 11th Int Conference on Intelligent Autonomous Systems (IAS-11), 2010.

[13] The Player Project. http://playerstage.sourceforge.net/.

Paper B

A Bayesian framework for informed search using convolutions between observation likelihoods and spatial relation masks

Patricio Loncomilla, Marcelo Saavedra and Javier Ruiz-del-Solar

Publicado en:

16th International Conference on Advanced Robotics (ICAR), 2013
Noviembre 2013, Montevideo, Uruguay

A Bayesian framework for informed search using convolutions between observation likelihoods and spatial relation masks

Patricio Loncomilla, Marcelo Saavedra and Javier Ruiz-del-Solar
University of Chile
Advanced Mining Technology Center, Department of Electrical Engineering
Santiago, Chile
{ploncomi, msaavedra, jruizd}@ing.uchile.cl

***Abstract*—In this work, a novel methodology for robots executing informed object search is proposed. The methodology is based mainly on a Bayesian framework that uses convolutions between observation likelihoods and spatial relation masks for estimating the probability map of the object being search for. By using spatial relation masks, complex spatial relations between objects can be defined as weighted sums of basic spatial relations using co-occurrence matrices as weights. The methodology is validated in an office environment in which four object classes ("monitor," "keyboard," "system unit," and "router") and four basic spatial relations ("very near," "near," far," and "very far") are considered. Experiments combine statistics about object's co-ocurrence and about object detection in real environments, and search trials using a realistic simulation tool in which extensive tests comparing six object search algorithms are carried out. Results show that the use of the proposed methodology increases the object detection rate in a search task from 27.5% up to 53.2%.**

Keywords— Semantic search, Informed search, Active search, Spatial relations, Co-occurrence matrix.

I. INTRODUCTION

Object search is an important ability for a mobile robot. Some previous work on object search are based on the use of spatial object-place relations, assuming that in a scene the searched object will be readily available to the robot's field of vision. However, there are sets of objects inside a setting that tend to appear near of each other as they have a particular spatial relation, making it possible to infer the existence of an object A, given an object B. For example, when looking inside offices, it can be noted that the object "keyboard" is often very near the object "monitor". For a human, the ability to deduce that object A tends to be "near", "very near" or "far" from object B is a trivial task, since human beings can learn spatial relations between objects by observing a large number of similar settings. In this paper, we focus on giving robots the ability to find objects using existing semantic relations with other objects such as "near" or "far," in a given environment.

In order to accomplish this goal, we propose a Bayesian framework that uses convolutio[1]ns between observation likelihoods and spatial relation masks for estimating the probability map of the object being search for, and a search procedure that uses this probability map. The main contribution of this methodology is the use of convolutions for computing the probability of the presence of an object from positive and negative detections of all related objects in the environment, in a unified way. The secondary contribution is the use of basic semantic categories such as "very near," "near," "far," and "very far" for generating basic spatial relation masks that can be combined to generate complex spatial relation masks by using a set of weights named co-occurrence values.
In order to characterize and validate the proposed object search methodology, a realistic simulation environment was developed. This environment is based on the use of the Player/Stage simulator [13], but it incorporates: (i) statistics about the co-ocurrence of the object's classes in the real-world to compute the parameters of the spatial relation masks, (ii) a model for the observation of the objects in 3D by the robot, and (iii) the use of a realistic object detection system, whose performance depends on the relative pose between the object and the robot's camera, as well as the object class. We believe that this simulation environment is an appropriate tool to evaluate object search algorithms, because it allows obtaining repeatable experiments and to better quantify the performance of the search methodologies under realistic conditions.

This work corresponds to an improvement of the one presented in [15]. Here, the proposed methodology is better explained, and an informed search procedure is presented. In addition, a realistic 3D simulation environment that includes statistics about the detection of real objects is used.

The paper is organized as follows. In Section II some related work is presented. In Section III the proposed methodology for object search is described. In Section IV an experimental validation of the

978-1-4799-2722-7/13/$31.00 ©2013 IEEE

methodology is presented. Finally, in Section V some conclusions of this work are given.

I. RELATED WORK

The search for objects in a real environment is a very complex task for robots. Garvey in [1] recognized this problem and proposed the idea of indirect search, i.e. searching for another intermediate object that maintains a particular spatial relation with the object being searched for. Wixson et al. materialize the idea of indirect search, demonstrating greater efficiency both theoretically and empirically [2]. But the problem with indirect search is that the spatial relationship between the object sought and the intermediate object does not always exist. Furthermore, the detection of the intermediate object may be more difficult than the detection of the desired/primary object itself. In fact, Ye and Tsotsos demonstrated that the search for an arbitrary object in 3D space is NP-complete [3]. Shubina and Tsotsos propose an algorithm that considers the cost and effect of different actions with different types of prior knowledge and different spatial relations between objects [4]. Kollar et al. perform the search for an object in a known map of the environment, using object-object and object-scene context [5]. They obtain the co-occurrence of the existence of objects in two-dimensional images for learning correlations between object categories, and between objects and place labels (semantic labels such as “kitchen”). These images were taken from the Flickr website, and they do not take the distances between objects into consideration. Viswanathan et al. propose an approach using existing resources: common-sense knowledge of machine learning of object relations [6]. They use marked images from the LabelMe database, designed by Russell et al. [7]. They train an automatic classifier of places based on the presence of the detected objects to infer the probability that the other objects exist, and the kind of place (e.g., kitchen or office) is seen in the setting. Kasper et al. perform a study on spatial relations in three-dimensional images using a Kinect sensor [8]. They created a database using nine different office space settings with a total of 168 objects in 35 object classes. Then, they found the distances between different objects. They also made predictions about the location of unfound objects by detecting their surrounding objects. Galindo et al. performed a study based on 2D data, combining metric, topological, and semantic aspects on a map [9]. In addition, they proposed a method for learning these semantic representations from sensory data. Vasudevan et al. attempted to create a spatial representation in terms of objects, by encoding typical household objects and doors within a hierarchical probabilistic framework [10]. They used a SIFT [11] based object recognition system and a door detection system based on lines extracted from range scans. They also proposed a conceptualization of different places, based on the objects that were observed inside them. Elfring et al. proposed an active object search method that considers object’s co-ocurrences and the cost of reaching the intermediate objects [16]. However, they do not consider spatial relations between the main and the intermediate objects. Aydemir et al. developed a method for object search using explicit spatial relations between objects in order to perform an efficient visual search [12]. They presented a computational model using several random views to guide the robot's camera to the points where the objects have a high probability of being found by using the spatial-relation term "on" between objects, in an indoor environment since the objects are mostly on horizontal surfaces. The work presented in this paper is an improvement of the model proposed by Aydemir in [12].

III. METHODOLOGY

A. Bayesian framework for probability map updating

A spatial relation between two objects is defined as the probability distribution of the pose of the first object, π_A, given the known pose of the second object, π_B:

$$Rel_{A,B}(\pi_A, \pi_B) = p(\pi_A \mid \pi_B) \quad (1)$$

As the relation is treated as a probability distribution, the sum over all possible poses of A for a fixed pose of B is equal to one:

$$\int_{\pi_A} p(\pi_A \mid \pi_B) = 1 \quad (2)$$

If the spatial relation is invariant to translations and rotations, i.e. it only depends on the relative pose, $\pi_{A/B}$, of object A with respect to object B, then the expression for the probability can be rewritten as:

$$Rel_{A/B}(\pi_{A/B}) = p(\pi_{A/B}) \quad (3)$$

The robot moves in a two-dimensional space parameterized by using coordinates (x,y). The space is quantized into squared cells with size k, and

parameterized by using indexes (i,j). By using the index notation, a spatial relation can be written as:

$$R_{A/B}(i,j) = K_{norm} * Rel_{A/B}(ki,kj)$$

$$\sum_i \sum_j R_{A/B}(i,j) = 1 \quad (4)$$

where K_{norm} is a normalizing constant.

The term $p(a_{i,j})$ represents the probability that the center of the main object A is in the cell (i,j). Both positive and negative observations z_A provide valuable information for the object search process, and can be used to compute an updated probability $p(a_{i,j}|z_A)$. The probability $p(a_0)$ is treated as a special case, and it represents the probability of the object being outside the search region.

Positive detections z_A=*true* provide information about the places where the object has high probability of being, by means of a likelihood $p(z_A=true|a_{i,j})$, which is defined over the cell (i,j). The likelihood has a high value over the cell where the object was detected, and a low value in the other cells. Negative detections z_A=*false* provide information $p(z_A=false|a_{i,j})$ about the cells where the objects have low probability of being, which are those cells visible from the current viewpoint that have a low probability of containing the object.

The problem addressed in this work is to find a main object, A, by moving the robot appropriately. The robot search process is applied until the main object, A, is found. In consequence, the search process includes only negative detections of the main object, A, before the object is found. Two cases are considered:

$$p(a_{i,j} \mid z_A = false) = \frac{p(z_A = false \mid a_{i,j})p(a_{i,j})}{p(a_o) + \sum_{i,j} p(z_A = false \mid a_{i,j})p(a_{i,j})}$$

$$p(a_0 \mid z_A = false) = \frac{p(a_0)}{p(a_o) + \sum_{i,j} p(z_A = false \mid a_{i,j})p(a_{i,j})} \quad (5)$$

The secondary object, B, can produce positive and negative detections z_B, which can be used to compute an updated probability:

$$p(a_{i,j} \mid z_B) = \frac{p(z_B \mid a_{i,j})p(a_{i,j})}{p(z_B \mid a_o)p(a_0) + \sum_{i,j} p(z_B \mid a_{i,j})P(a_{i,j})}$$

$$p(a_0 \mid z_B) = \frac{p(z_B \mid a_o)p(a_0)}{p(z_B \mid a_o)p(a_0) + \sum_{i,j} p(z_B \mid a_{i,j})P(a_{i,j})} \quad (6)$$

The terms $p(z_B|a_{i,j})$ and $p(z_B|a_0)$ will be called cross-likelihoods, as they relate the detection of a secondary object, B, with the presence of the main object, A, on the map. These probabilities can be derived by considering probabilities $p(b_{u,v})$ for the presence of a secondary object, B, at locations (u,v) in the grid:

$$p(z_B \mid a_{i,j}) = \sum_u \sum_v p(z_B \mid b_{u,v})p(b_{u,v} \mid a_{i,j}) \quad (7)$$

$$p(z_B \mid a_0) = \sum_u \sum_v p(z_B \mid b_{u,v})p(b_{u,v} \mid a_0) \quad (8)$$

The term $p(b_{uv}|a_0)$ is considered a constant over (u,v) whose sum has a value of 1 because B is supposed to be on the map. The term $p(b_{u,v} \mid a_{i,j})$ corresponds to the spatial relation between the main object, A, at location (i, j) and a secondary object, B, at location (u, v). By replacing this term with the spatial relation $R_{B/A}$, there is no need for storing a map for the secondary object; only the map for the main object and the likelihoods of the detections of the secondary object are needed:

$$p(z_B \mid a_{i,j}) = \sum_u \sum_v p(z_B \mid b_{u,v})R_{B/A}(u-i, j-v) \quad (9)$$

$$p(z_B \mid a_0) = \frac{1}{n_U n_V} \sum_u \sum_v p(z_B \mid b_{u,v}) \quad (10)$$

where $n_U n_V$ is the size of the map.

Equation (9) can be implemented as a convolution in the (i, j) space between a likelihood image and a mask $R_{B/A}(i,j)$ describing the spatial relation between the main and secondary objects, which will be named a *spatial relation mask*:

$$p(z_B \mid a_{i,j}) = p(z_B \mid b_{i,j}) * R_{B/A}(i,j) \quad (11)$$

The proposed system is highly versatile because any spatial relation can be represented by an appropriate mask. It must be noted that extra secondary objects can be added to the system by creating additional spatial relation masks. In case these relations are chained, as an example object A is near B, and object B is near C, then the mask of the chained relation can be obtained by convolution of the original masks:

$$P(z_C \mid a_{i,j}) = P(z_C \mid b_{i,j}) * R_{B/A}(i,j) \quad (12)$$

$$P(z_C \mid a_{i,j}) = P(z_C \mid c_{i,j}) * R_{C/B}(i,j) * R_{B/A}(i,j) \quad (13)$$

$$\Rightarrow R_{C/A} = R_{C/B} * R_{B/A} \quad (14)$$

B. Search strategy

A path for searching for a given object can be created by generating optimal viewpoints at each iteration. The optimal viewpoint is generated from a set of random poses reachable in a fixed time, and selecting the one that maximizes the probability of finding the object in the visible area [8]:

$$\arg\max_{k=1..N} \sum_{i=1}^{n} \sum_{j=1}^{n} p(a_{i,j})V(a_{i,j},k) \quad (15)$$

where N is the number of candidate poses, and $V(a_{i,j},k)$ is defined as:

$$V(a_{i,j},k)=\begin{cases}1, & \text{if } a_{i,j} \text{ is inside the k}^{th} \text{ view cone}\\ 0 & \text{otherwise}\end{cases} \quad (19)$$

In the experiments reported in Section IV we choose to use a simple search strategy, in which the robot, at each iteration, selects the execution of a composed sequence of movements consisting on one initial rotation, followed by a translation and a final rotation (N=3). The parameter's space associated to this sequence of movements is explored randomly, and the best sequence is chosen as the one that maximize the probability of finding the searched object.

Figure 1 shows the proposed object search procedure.

1. *Initialize probabilities and masks*
2. *While main object not detected*
   - *Update the probability map of the main object*
   - *Detect secondary objects*
   - *For each detected secondary object*
     - *Compute cross-likelihood for positive detections*
     - *Update the probability map*
   - *For each non-detected secondary object*
     - *Compute cross-likelihood for negative detections*
     - *Update the probability map*
   - Compute next movements by selecting the optimal pose achievable in a given time. The optimal pose is the one that has the largest probability of detecting the main object inside its view cone.

Fig. 1. Proposed informed search procedure. It uses equations (6)-(9), (13)-(14), (18) and (19).

C. Creating spatial relation masks from co-occurrences

Complex spatial relations between objects are defined as weighted sums of basic spatial relations. Each basic spatial relation corresponds to a semantic category meaningful to humans. In this work, we focus on four simple spatial relations: "very near" (VN), "near" (N), "far" (F), and "very far" (VF). The use of these spatial relations is appropriate as it enables the system to estimate a set of basic probability distributions from samples of relative positions of the objects in the real world. The masks for each of the spatial relations are defined in two versions, *hard masks* and *soft masks*. Hard masks are defined by two thresholds and have a rectangular profile, while soft masks are defined by four thresholds and have a trapezoid-shaped profile. Each basic mask is normalized to sum one over all of the cells on the map, thus a normalization constant is added to the formulas. Hard masks are defined as:

$$R_{B/Ahard}(i,j;a_1,a_2)=K*\begin{cases}1 & a_1\le\sqrt{(ki)^2+(kj)^2}<a_2\\ 0 & other\end{cases} \quad (20)$$

$$R^{VN}_{B/Ahard}(i,j)=R_{B/Ahard}(i,j;0,u_1) \quad (21)$$

$$R^{N}_{B/Ahard}(i,j)=R_{B/Ahard}(i,j;u_1,u_2) \quad (22)$$

$$R^{F}_{B/Ahard}(i,j)=R_{B/Ahard}(i,j;u_2,u_3) \quad (23)$$

$$R^{VF}_{B/Ahard}(i,j)=R_{B/Ahard}(i,j;u_3,\infty) \quad (24)$$

while soft masks are defined as:

$$R_{B/Asoft}(i,j;a_1,a_2,a_3,a_4)=K*\begin{cases}\dfrac{\sqrt{(ki)^2+(kj)^2}-a_1}{a_2-a_1} & a_1\le\sqrt{(ki)^2+(kj)^2}<a_2\\ 1 & a_2\le\sqrt{(ki)^2+(kj)^2}<a_3\\ \dfrac{a_4-\sqrt{(ki)^2+(kj)^2}}{a_4-a_3} & a_3\le\sqrt{(ki)^2+(kj)^2}<a_4\\ 0 & other\end{cases} \quad (25)$$

$$R^{VN}_{B/Asoft}(i,j)=R_{B/Asoft}(i,j;0,0,u_1-\delta,u_1+\delta) \quad (26)$$

$$R^{N}_{B/Asoft}(i,j)=R_{B/Asoft}(i,j;u_1-\delta,u_1+\delta,u_2-\delta,u_2+\delta) \quad (27)$$

$$R^{F}_{B/Asoft}(i,j)=R_{B/Asoft}(i,j;u_2-\delta,u_2+\delta,u_3-\delta,u_3+\delta) \quad (28)$$

$$R^{VF}_{B/Asoft}(i,j)=R_{B/Asoft}(i,j;u_3-\delta,u_3+\delta,\infty,\infty) \quad (29)$$

In this work, basic masks defined by a circle of radius u_1 in the case of "very near", a circular ring of radii u_1 and u_2 in the case of "near," a circular ring of radii u_2 and u_3 in the case of "far," and a circular ring of internal radius u_3 and an external radius that cover the whole map in the case of "very far". The radius values are selected by considering statistics of the distances between objects A and B, and by modeling their selection process as a classification problem. Thus, the optimal radius value between two categories, e.g., "near" and "far", is the one that generates the same mean classification error in both classes.

A complex mask can be created as a weighted sum of basic hard or soft masks:

$$\begin{aligned}R_{B/A}(x,y)=&C^{VN}_{B/A}R^{VN}_{B/A}(x,y)+C^{N}_{B/A}R^{N}_{B/A}(x,y)+\\&+C^{F}_{B/A}R^{F}_{B/A}(x,y)+C^{VF}_{B/A}R^{VF}_{B/A}(x,y)\end{aligned} \quad (30)$$

The four coefficients $C^{VN}_{B/A}$, $C^{N}_{B/A}$, $C^{F}_{B/A}$ and $C^{VF}_{B/A}$ are called co-occurrences because they indicate the relative frequency of occurrence of a pair of objects for each spatial relation. They can be constructed from samples of positions of both objects by computing the number of occurrences of each basic spatial relation. If a set of samples is divided into basic semantic categories and the count is n_{VN} for "very near," n_N for "near," n_F for "far," and n_{VF} for "very far," the co-occurrences can be computed as:

$$C_{B/A}^{VN} = \frac{n_{VN}}{n_{VN} + n_N + n_F + n_{VF}} \quad (31)$$

$$C_{B/A}^{N} = \frac{n_N}{n_{VN} + n_N + n_F + n_{VF}} \quad (32)$$

$$C_{B/A}^{F} = \frac{n_F}{n_{VN} + n_N + n_F + n_{VF}} \quad (33)$$

$$C_{B/A}^{VF} = \frac{n_{VF}}{n_{VN} + n_N + n_F + n_{VF}} \quad (34)$$

IV. RESULTS

A. Experimental setup

In order to characterize and validate the proposed object search methodology, a realistic simulation environment was developed. This environment is based on the use of the *Player/Stage* simulator [13], but it incorporates: (i) statistics about the co-ocurrence of the object's classes in the real-world to compute the parameters of the spatial relation masks, (ii) a model for the observation of the objects in 3D by the robot, and (iii) the use of a realistic object detection system, whose performance depends on the relative pose between the object and the robot's camera, as well as the object class.

B. Modeling the 3D object's observation

Player/Stage is designed to manage 2D environments. Its fiducial detector defines a 2D field of view, and a minimum and a maximum object's detection range. However, for modeling the observation of objects in a 3D environment additional parameters need to be considered, like the pose of the robot's camera and the object's height. Figure 2 shows the robot and camera configuration used in this work.

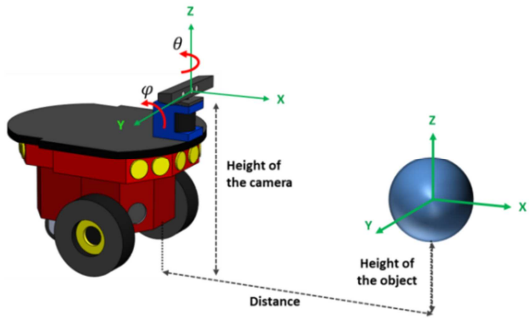


Fig. 2. Geometric parameters needed for simulating 3D detections.

For modeling/simulating the observation obtained by a robot in a 3D environment, the following transformation matrix must be used:

$$\begin{bmatrix} x_{cam} \\ y_{cam} \\ z_{cam} \end{bmatrix} = \begin{bmatrix} \cos(-\varphi) & 0 & -\sin(-\varphi) \\ 0 & 1 & 0 \\ \sin(-\varphi) & 0 & \cos(-\varphi) \end{bmatrix} * \begin{bmatrix} \cos(-\theta) & -\sin(-\theta) & 0 \\ \sin(-\theta) & \cos(-\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x_{obj} - x_r \\ y_{obj} - y_r \\ z_{obj} - z_r \end{bmatrix} \quad (35)$$

where φ is the pitch angle the camera, θ is the yaw angle of the robot on the horizontal axis, $[x_{obj}, y_{obj}, z_{obj}]$ is the position of the center of the object, $[x_r, y_r, z_r]$ is the position of the robot's camera, and $[x_{cam}, y_{cam}, z_{cam}]$ is the position of the object in the camera system. Then, the relative pose of the object is computed as:

$$r_d = \sqrt{x_{cam}^{\ 2} + y_{cam}^{\ 2} + z_{cam}^{\ 2}} \quad (36)$$

$$\theta_d = \text{atan2}(y_{cam}, x_{cam}) \quad (37)$$

$$\varphi_d = \text{atan2}(z_{cam}, x_{cam}) \quad (38)$$

with r_d the distance from the object to the camera, θ_d the yaw angle of the object in the camera system, and φ_d the pitch angle of the object in the camera system. An object can be detected if it is contained in the robot's field of view defined as: $r_d \in [r_{\min}, r_{\max}]$, $\theta_d \in [\theta_{\min}, \theta_{\max}]$ and $\varphi_d \in [\varphi_{\min}, \varphi_{\max}]$.

The simulation of occlusions between objects in a 2D environment is different than in the 3D case (see Figure 3). In the 3D case, we model each object as a sphere whose center is placed on the center of the object, and its radius ρ depends on the object size. For each pair of spheres contained in the field of view, they are projected in the image plane and it is verified whether they intersect in the image space or not. In the case they intersect, an occlusion has being detected and only the nearest object is selected as visible. This procedure is shown in Figure 4.

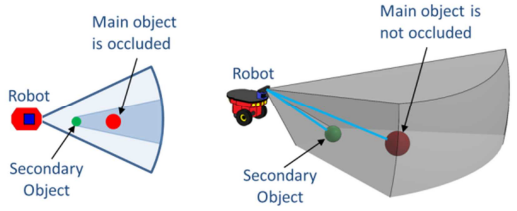


Fig. 3. Occlussion's visualization in a 2D (left), and in a 3D enviroment (right).

for each pair of objects

$obj_1 : (\rho_1, \theta_1, \varphi_1),\ obj_2 : (\rho_2, \theta_2, \varphi_2)$

$ang\_proj_1 = \text{atan}\,2(\rho_1, x_1)$

$ang\_proj_2 = \text{atan}\,2(\rho_2, x_2)$

$dist = \sqrt{(\theta_2 - \theta_1)^2 + (\varphi_2 - \varphi_1)^2}$

if $(dist > ang\_proj_1 + ang\_proj_2)$

obj_1 and obj_2 can be detected

else

occlussion

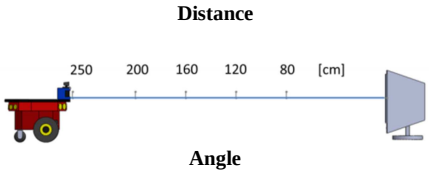
Fig. 4. Procedure for computing objects' occlusions.

C. Realistic Object Detection Simulation

In order to simulate realisticly the detection of objects, statistics about the performance of a specific object detection system, under different camera-object distances and angles, were calculated. These statistics were used to build an object-detection LUT (Look-Up Table), which is used by the simulator every time the robots tries to detect an object placed inside its field of view.

The selected object detection system uses HOG descriptors as features and SVMs as classifiers. For each object class and object orientation a different SVM is used. Each SVM is trained by computing HOG descriptors from five sets of 50 labeled images that show the object under the required orientation for 5 different distances, using the setup described in Figure 5. The SVM training procedure is described in [14].

The object's detection statistics are computed using images captured under 40 different viewpoints. As shown in Figure 5, the viewpoints correspond to camera-object distances of 80, 120, 160, 200 and 250 centimeters, and camera-object yaw angles of 0, 45, 90, 135, 180, 225, 270 and 315 degrees (pitch and roll camera-angles are not considered). For each viewpoint, 100 images of each object are captured by using a short video sequence. In total 4,000 images of each object are obtained. For each viewpoint and each object, a detection probability is computed by counting the amount of detections in the 100 available images. Four objects are considered "monitor," "keyboard," "system unit," and "router". The detection probabilities are summarized in Table I, and coded into the object-detection LUT.



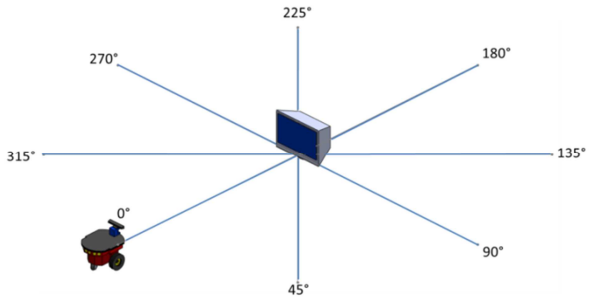


Fig. 5. Viewpoints with different distances and different angles.

TABLE 1. OBJECT DETECTION PROBABILITY FOR: (A) SYSTEM UNIT, (B) MONITOR, (C) ROUTER AND (D) KEYBOARD.

	Distance [cm]					Distance [cm]				
Angles	**80**	**120**	**160**	**200**	**250**	**80**	**120**	**160**	**200**	**250**
0°	0.81	0.76	0.69	0.42	0.21	0.33	0.40	0.57	0.67	0.84
45°	0.15	0.31	0.45	0.39	0.22	0.12	0.38	0.40	0.85	0.89
90°	0.66	0.76	0.80	0.69	0.29	0.49	0.57	0.67	0.76	0.81
135°	0.12	0.35	0.49	0.31	0.24	0.17	0.75	0.70	0.76	0.74
180°	0.88	0.94	0.78	0.55	0.26	0.45	0.48	0.58	0.59	0.67
225°	0.11	0.39	0.40	0.35	0.19	0.07	0.95	0.74	0.78	0.72
270°	0.68	0.78	0.82	0.69	0.46	0.38	0.55	0.64	0.71	0.84
315°	0.20	0.32	0.44	0.38	0.25	0.10	0.56	0.45	0.65	0.60

(A) (B)

	Distance [cm]					Distance [cm]				
Angles	**80**	**120**	**160**	**200**	**250**	**80**	**120**	**160**	**200**	**250**
0°	0.51	0.67	0.54	0.32	0.00	0.34	0.62	0.80	0.64	0.59
45°	0.24	0.39	0.62	0.41	0.17	0.52	0.85	0.82	0.79	0.45
90°	0.56	0.65	0.47	0.29	0.00	0.05	0.28	0.26	0.64	0.79
135°	0.52	0.49	0.39	0.33	0.15	0.36	0.80	0.42	0.38	0.36
180°	0.43	0.75	0.54	0.27	0.00	0.41	0.43	0.64	0.50	0.33
225°	0.48	0.46	0.37	0.33	0.13	0.60	0.82	0.86	0.63	0.52
270°	0.55	0.57	0.45	0.34	0.00	0.06	0.15	0.25	0.63	0.75
315°	0.47	0.58	0.48	0.43	0.14	0.44	0.86	0.62	0.58	0.68

(C) (D)

D. Creation of co-occurrence matrices

A set containing a total of 243 labeled images obtained both from LabelMe [7] and from the Flickr website were used for generating co-occurrence

matrices. In each image, instances of the objects "monitor," "system unit," "keyboard," and "router" were labeled. As the sizes of the objects and the parameters of the camera are known, it is possible to compute the pose of each object in space. Several instances of the objects on the set of images and their poses were used to construct co-occurrence matrices for the categories "very near," "near," "far," and "very far" for each of the objects with respect to the others.

Given the poses of a pair of objects, a distance was computed and used for selecting whether the sample belongs to the categories "very near," "near," or "far". If an object is detected alone in an image, the sample belongs to the category "very far". Only the depth and horizontal axis were used to compute the distances, as differences in the vertical direction do not affect the position of the object when it is transformed onto the 2D grid. As an example, the final co-occurrences for the object "monitor," as the main object, are shown in Table II.

The basic hard and soft masks are defined by equations (20)-(24) and (25)-(29), respectively. The thresholds that separate the categories "very near," "near," "far" and "very far" are u_1=60[cm], u_2=100[cm] and u_3=150[cm]. The gap parameter for the soft masks is d=20[cm].

TABLE I1. FINAL CO-OCCURRENCES OF OBJECTS AROUND THE OBJECT "MONITOR".

Main object Monitor	**Secondary objects**		
Semantic categories	***keyboard***	***system unit***	***Router***
Very Near	0.773	0.178	0.061
Near	0.143	0.491	0.151
Far	0.046	0.258	0.485
Very Far	0.038	0.074	0.303

E. Experiments

Experiments were performed on maps containing four objects. Each map contains a main object, named A (monitor), to be searched for, and three secondary objects named B (keyboard), C (system unit), and D (router). The size of each map is 6[mt] x 6[mt]. A map exemplifying a configuration of the objects is shown in figure 6. A total of 20 maps were created by picking a random position for the main object, A, and then picking a random position for the objects B, C, and D following a distribution that represents their co-occurrences. Objects are placed at 0.5[mt] from the floor, and the height of the camera is 1[mt]. The orientations of the objects in each map are selected randomly as multiples of 45 degrees. The 20 maps are used to perform the experiments, each map being used the same number of times as the others. A laser sensor from *Player/Stage* is used for avoiding collisions. Observations of the objects are obtained by using the 3D obervation system described in Section IV.B. The field of view of the sensor has an horizontal range of 75 degrees, a vertical range of 45.6 degrees, a minimum depth of 0.3[mt] and a maximum depth of 2.5[mt]. The main object, A, can be detected up to 1[mt], and the secondary objects, B, C, and D can be detected up to 2[mt].

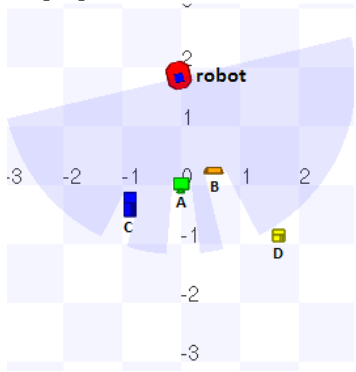


Fig. 6. Map showing the configuration of the objects in a particular search procedure.

In each experiment the goal is to find the main object A before 1,500 views have been processed. In the search process, only negative detections of the main object need to be processed because a positive detection causes the search process to finish.

Six algorithms of object search are compared on the same set of maps, two of them are preexistent and the other four are novel ones that use convolutions for relating detections of secondary objects with the presence of the main object in the map. The first correspond to the baseline algorithm where no information from secondary objects is used. The second algorithm is the one proposed by Aydemir in [12], and the next four algorithms correspond to different variants of the proposed methodology. The algorithms are:

1. *Uninformed search*: A probability map $p(a_{i,j})$ for object A is estimated by using negative detections of that object, then object A is searched for by finding viewpoints that maximize the probability of containing A.

2. *Informed search using particles:* The algorithm of Aydemir [12] is used for constructing a probability map $p(a_{i,j})$ for object A by using negative detections of that object. Then object A is searched for by finding viewpoints that maximize the probability of containing A. When a secondary object B is detected, a set of 500 particles is generated around the detection inside the current view cone. Each particle represents a hypothesis about the location of the object A given the known position of the object B in the map. The spatial relation between objects A and B induce a probability distribution of the presence of A given the position of B, that has a maximum value over the map. The particles with associated probability value equal or

greater than half of the maximum probability are used to select the next optimal viewpoint by maximizing the amount of accepted particles inside the view cone.

3. *Informed search with convolutions using positive and negative information with hard masks*: A probability map $p(a_{i,j})$ for the main object A is estimated by using positive and negative detections of objects B, C, and D, negative detections of object A, and hard spatial relation masks. Then object A is searched by finding viewpoints that maximize the probability of containing it.

4. *Informed search with convolutions using only positive information with hard masks:* Similar to algorithm 3, but in this case only positive detections of objects B, C, and D are used.

5. *Informed search with convolutions using positive and negative information with soft masks:* Similar to algorithm 3, but in this case soft spatial relation masks are used.

6. *Informed search with convolutions using only positive information with soft masks:* Similar to algorithm 4, but in this case soft spatial relation masks are used.

For each algorithm, a total of 720 experiments considering different maps and different initial robot poses were executed. The results are shown in Table III.

TABLE 1II. RESULTS OF OBJECT SEARCH

Method	Number of searches	Successful searches	Detection rate (%)
Uninformed Search	720	198	27.5
Informed search using particles [12]	720	213	29.6
Positive information & hard masks	720	302	41.9
Positive information & soft masks	720	317	44.0
Positive and negative information & hard masks	720	356	49.4
Positive and negative information & soft masks	720	383	53.2

From Table III it can be observed that the baseline method has a DR (Detection Rate) of 27.5%. When the proposed informed search method, with positive information and hard masks, is considered, the DR increases up to 41.9%. When positive and negative information is used, the DR increases up to 49.4%. In both cases, the use of soft masks instead of hard marks increases the DR up to 44.0% and 53.2%, respectively. Aydemir's particle based informed search method performs better than the baseline, however, it has a much lower DR, 29.6%, than the proposed methods.

From the results of the experiments, it is evident that the methods that use convolutions for integrating information about secondary objects into the probability distribution of the main object obtain better performance than methods that do not include such kind of information. This happens because the integrated information can be used several frames after the moment when the object is seen. The integration of negative information improves the detection rate as information about non detections of secondary objects is added to the probability map. The use of soft masks causes an improvement respect to the use of hard masks because the probability map has no sharp transitions between observed and unobserved zones.

F. Scaling of processing time with the number of objects

The proposed informed search method can be optimized by observing that the cross-likelihood images are constant, except around the detection area in the case of positive detections, and around the view cone in the case of negative detections. This occurs because the spatial relation between two objects is constant when the distance between them is large enough. Considering that the computation of convolutions in constant valued areas is not needed, focalized convolutions can be applied in the non-constant areas. Informed search using positive and negative information scales linearly with the number of total secondary objects as it needs to compute cross-likelihoods in positive and negative detections. Informed search using only positive information scales linearly with the number of observed secondary objects as it needs to compute cross-likelihoods only in positive detections. As the mean amount of detected objects is low, the methods that use only positive information run as fast as uninformed search most of the time. The PT (processing time) of informed search methods that use convolutions scale quadratically with the size of the mask when using focalized convolutions, and they scale linearly with the size of the mask when using convolutions with the full map. The PT of focalized convolutions do not depend on the size of the map. In the experiments, a small map (100x100) and mask (32x32) were used for representing a 6m x 6m environment. In this case, the PT of the informed search process (0.2 seconds per convolution) is lower than the PT of the object detection process.

V. CONCLUSIONS

In this work, a novel methodology for performing informed search of objects was proposed and tested. The methodology is based on integrating information provided by secondary objects into the probability distribution of the main object to be found. Spatial relations between objects are estimated by using a set of basic spatial relations which are mixed by using co-occurrence values as weights. Six algorithms of object search were compared by using a realistic simulation environment. The results show that the detection rate of the search process increases

from 27.5% up to 53.2% when using the proposed methodology, which includes integration of positive and negative detections of secondary objects, as well as the use of soft masks. Future work includes the management of false detections in the Bayesian framework, a comparison with other object search methods, and the realization of experiments with a real robot for validating the results in the real world.

ACKNOWLEDGEMENTS

This work was partially funded by Fondecyt Project 1130153.

REFERENCES

[1] T. D. Garvey, "Perceptual strategies for purposive vision," Technical report, SRI International, vol. 117, 1976.

[2] L. Wixson and D. Ballard, "Using intermediate object to improve efficiency of visual search," Int. J. Comput. Vis. 18, vol. 3, pp. 209–230, 1994.

[3] Y. Ye and J. K. Tsotsos, "Sensor Planning for 3D Object Search," Computer Vision and Image Understanding, vol. 73-2, pp. 145 - 168, 1999.

[4] K. Shubina and J. Tsotsos, "Visual search for an object in a 3d environment using a mobile robot," Computer Vision and Image Understanding 114, vol. 5, pp. 535-547, 2010.

[5] T. Kollar and N. Roy, "Utilizing object-object and object-scene context when planning to find things," Proc. of the IEEE Int. Conf. on Robotics and Automation - ICRA 2009.

[6] P. Viswanathan, D. Meger, T. Southey, J. J. Little, and A. Mackworth, "Automated Spatial-Semantic Modeling with Applications to Place Labeling and Informed Search," Proc. of the Canadian Conf. on Computer and Robot Vision, 2009.

[7] B. Russell, A. Torralba, K. Murphy, and W. Freeman, "Labelme: A database and web-based tool for image annotation," Int. Journal of Computer Vision, vol. 77, pp. 157-173, May 2008.

[8] A. Kasper, R. Jäkel, and R. Dillmann, "Using spatial relations of objects in real world scenes for scene structuring and scene understanding," Proc. of the 15th Int. Conf. on Advanced Robotics - ICAR 2011, Tallinn, Estonia, 2011.

[9] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. A. Fernandez-Madrigal, and J. Gonzalez, "Multi-hierarchical semantic maps for mobile robotics," Proc. of the 2005 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems - IROS 2005, pp. 2278-2283, 2005.

[10] S. Vasudevan, S. Gachter, V. Nguyen, and R. Siegwart, "Cognitive maps for mobile robots-an object based approach," Robot. Auton. Syst. , vol. 55, pp. 359-371, 2007.

[11] D. G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," Int. Journal of Computer Vision, vol. 60, pp. 91-110, 2004.

[12] A. Aydemir, K. Sjöö, and P. Jensfelt, "Object search on a mobile robot using relational spatial information," Proc. of the 11th Int Conf. on Intelligent Autonomous Systems (IAS-11), 2010.

[13] R. T. Vaughan and B. P. Gerkey, “Reusable robot code and the Player/Stage Project,” in Software Engineering for Experimental Robotics, ser. Springer Tracts on Advanced Robotics, D. Brugali, Ed. Springer, 2007, pp. 267–289.

[14] O. Ludwig, D. Delgado, V. Goncalves, and U. Nunes, "Trainable Classifier-Fusion Schemes: An Application To Pedestrian Detection," Proc. of the 12th Int. IEEE Conf. on Intell. Transportation Systems, St. Louis, Vol. 1, pp. 432-437, 2009.

[15] P. Loncomilla, M. Saavedra, and J. Ruiz-del-Solar, "Semantic Object Search using Semantic Categories and Spatial Relations between Objects," Proc. of the RoboCup Symposium 2013, 1 July 2013, Eindhoven, The Netherlands (CD Proceedings).

[16] J. Elfring, S. Jansen, R. van de Molengraft, and M. Steinbuch, "Active object search exploiting probabilistic object-object relations," Proc. of the RoboCup Symposium 2013, 1 July 2013, Eindhoven, The Netherlands (CD Proceedings).