Navegación Reactiva de un Robot Móvil usando Kinect

J.R. Ruiz-Sarmiento, C. Galindo, J. Gonzalez-Jimenez, J.L. Blanco
Dpto. de Ingeniería de Sistemas y Automática, Universidad de Málaga, Campus de Teatinos, 29071 Málaga (España)

***Abstract*— Kinect es un novedoso dispositivo que ha aparecido recientemente en el mercado como periférico para el control de videojuegos. Su bajo precio y las prestaciones de los diferentes sensores que incluye lo convierten en un elemento idóneo para diversas aplicaciones robóticas. En este artículo se presenta la integración del sensor Kinect en un robot móvil para mejorar la realización de las tareas de detección de obstáculos y navegación reactiva. Concretamente, se presentan las principales características del sensor, y se proponen mecanismos para afrontar los problemas que surgen en su integración. Las pruebas realizadas ponen de manifiesto la utilidad de este sensor para que un robot móvil pueda desplazarse de forma más segura en entornos con obstáculos a diferentes alturas.**

I. INTRODUCTION

UNO de los principales requerimentos de un robot móvil es disponer de la habilidad para desplazarse de forma segura y robusta en su entorno de trabajo. En la literatura se pueden encontrar multitud de trabajos que han abordado este problema desde muy diversas perspectivas, encontrándonos algoritmos de navegación reactiva [1],[2], planificada [3], híbridos [4], etc.

Más allá de la estrategia de navegación adoptada, la robustez y fiabilidad del sistema dependerá en gran medida del sistema sensorial utilizado para percibir el entorno. La elección del sensor/es a considerar viene determinada principalmente por su precio y por las necesidades particulares de la aplicación, pudiendose encontrar sistemas de navegación basados en sónares [5], escáneres láser [6], en visión [7], etc.

En la práctica, la gran mayoría de robots móviles utilizan una estrategia de navegación híbrida, en la cual la navegación reactiva se encarga de la evitación de obstáculos. Los algoritmos que implementan dicha navegación reactiva suelen trabajar con información procedente de sensores 2D, típicamente escáneres láser, para la percepción del entorno. Este enfoque tiene el claro inconveniente de que el robot sólo es capaz de detectar obstáculos si estos se encuentran en el mismo plano 2D que percibe el escáner, limitación que ha motivado la búsqueda de alternativas, como por ejemplo, crear sensores 3D dotando de un grado de libertad adicional a los sensores 2D [8]. No obstante, las soluciones propuestas son generalmente costosas, y a veces poco efectivas por el alto tiempo de aquisición de un barrido 3D.

Este trabajo ha sido financiado por el Plan Nacional de Investigacion, proyecto DPI08-03527. Contacto: jotaraul@isa.uma.es

La reciente aparición en el mercado del sensor Kinect [9] impulsa líneas de investigación limitadas hasta hoy por la tecnología. Este dispositivo desarrollado por Microsoft está diseñado para controlar videojuegos, pero a la vez presenta características que resultan muy interesantes para aplicaciones de robótica móvil:

- Sensor compacto y ligero que proporciona imágenes RGB y de rango.
- Rápido, con una frecuencia de trabajo de 30 Hz.
- Rango de trabajo aceptable para la detección de obstáculos, desde 1.2 hasta 3.5 metros, con errores por debajo de 2 cm.
- Barato, entorno a 150€.

La utilización del sensor Kinect en el desarrollo de aplicaciones robóticas se está abriendo camino en diferentes plataformas, disponiendose muy recientemente de una API oficial [10].

En este artículo se describe la integración del sensor Kinect en un robot móvil para mejorar su navegación reactiva, gracias a su capacidad de detección de obstáculos de manera densa en todo el rango de alturas del robot. Esta integración, sin embargo, presenta varios problemas a solventar. Por un lado, la cámara de rango de Kinect tiene una resolución de 640x480 píxeles, es decir, en cada escaneo, con una frecuencia de 30 Hz, proporciona 307.200 puntos, lo cual representa un flujo de datos continuo que puede suponer una limitación computacional. Por otro lado, la distancia mínima de funcionamiento es 1.2 metros, es decir, no es capaz de percibir objetos cercanos al robot, con el consiguiente riesgo de colisión para el mismo.

A continuación se describe con más detalle el sensor Kinect. La sección 3 presenta el algoritmo de navegación reactiva utilizado en este trabajo y cómo hace uso de la información tridimensional de este sensor. En la sección 4 se describe cómo se han abordado los problemas que surgen de la integración de Kinect como sensor para cualquier algoritmo típico de navegación reactiva. En la sección 5 se describen y discuten los resultados obtenidos. Finalmente se exponen algunas conclusiones.

II. EL SENSOR KINECT

A. Descripción

El dispositivo Kinect (ver figura 1), distribuido en

Europa desde Noviembre del 2010, ha revolucionado en gran medida el mundo de la investigación relacionado con la percepción del entorno, tanto por sus prestaciones como por su reducido coste. Lanzado inicialmente como periférico para la videoconsola XBOX 360, se está afianzando en la comunidad robótica dado el amplio abanico de aplicaciones en las que se puede emplear.

Desde Junio de 2011 se cuenta con una API oficial de Microsoft [10], si bien, desde poco después de su aparición, la comunidad Open Source hizo público drivers no oficiales como Freenect [11] o CLNUI [12].

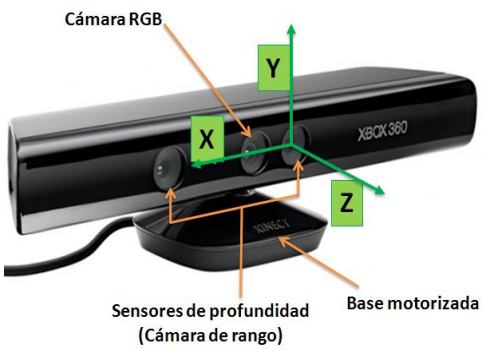

Figura 1. Sensor Kinect.

Kinect está provisto de una cámara RGB, un sensor de profundidad, una matriz de micrófonos y una base motorizada que le permite un movimiento de cabeceo. Además, posee una serie de giróscopos que aportan información sobre su orientación.

La cámara RGB tiene un flujo de datos de 8 bits por canal, siendo capaz de proporcionar imágenes a una frecuencia de 30Hz para resolución VGA (640x480 píxeles). Por su parte, el sensor de profundidad, al que también se le denomina cámara de rango, está compuesto por un proyector de luz infrarroja combinado con un sensor CMOS monocromo. En este caso el flujo de datos es de 11 bits, manteniendo igualmente una resolución VGA (640x480 píxeles) y una frecuencia de muestreo de 30 Hz. El campo de visión de ésta cámara es de 58º en horizontal y 45º en vertical. El rango de trabajo reportado por el fabricante va desde los 1.2m hasta los 3.5m, aunque en las pruebas realizadas en este trabajo, se ha comprobado que el sensor es capaz de detectar objetos poco reflectantes situados a 0.5 metros[1].

Kinect cuenta también con una matriz de cuatro micrófonos capaces de procesar audio de 16 bits con una frecuencia de muestreo de 16 kHz. La disposición en matriz permite determinar la fuente del sonido y eliminar ruido ambiente. En cuanto a la base motorizada, ésta permite inclinar (movimiento de cabeceo) el conjunto de sensores descrito anteriormente hasta ± 27º en el eje Z.

[1] La caracterización del sensor Kinect y sus prestaciones ante objetos de diferentes características, es decir, material, color, condiciones ambientales, etc. están fuera del ámbito de este trabajo.

B. Funcionamiento de la cámara de rango

La cámara de rango de Kinect está formada por dos componentes: un proyector de luz infrarroja (IR) y un sensor CMOS monocromo estándar [13]. Ambos se encuentran alineados a lo largo del eje X del dispositivo, a una distancia (denominada "línea base") de 75mm., con ejes ópticos paralelos (ver figura 1). Esta disposición dentro de Kinect facilita los cálculos de profundidad, que se basan en un principio similiar al de triangulación activa entre emisor y cámara, esto es, entre los rayos visuales de los puntos proyectados y sus corresopndientes proyecciones en la imagen. A diferencia de los métodos tradicionales de triangulación activa, los desarrolladores de la tecnología de la cámara de rango presente en Kinect (PrimeSense [14]) proponen una técnica ingeniosa para la obtención de información 3D, denominada *Codificación de Luz* (Light Coding) [15]. La idea principal consiste en un proceso en dos fases, una primera de calibración, y otra de funcionamiento.

En la fase de calibración, se emplea el proyector de luz infrarroja para proyectar un patrón de puntos (ver figura 2) sobre un plano de la escena, variando su distancia entre posiciones conocidas. A su vez, la cámara captura una imagen del patrón proyectado sobre el plano para cada una de estas distancias. Las imágenes obtenidas se denominan imágenes de referencia y se almacenan en el sensor.

Figura 2. Ejemplo de patrón de puntos proyectado.

En la fase de funcionamiento se emplean las imágenes de referencia para sustituir "virtualmente" al emisor del patrón IR, de tal manera que para cada nueva imagen capturada por el sensor, el cálculo de profundidad se resume a un problema de visión estéreo con configuración ideal: cámaras idénticas, ejes alineados y separados una distancia base de 75 mm.

En cuanto al error cometido por las mediciones, de acuerdo con [11], éste es menor a los 10cm a distancias superiores a los 4m, y menor a los 2cm en mediciones inferiores a los 2.5m.

C. Detección de obstáculos con Kinect

Para el trabajo que nos ocupa es esencial estudiar: (i) la capacidad del sensor Kinect para detectar obstáculos difícilmente detectables por otros sensores, bien por su posición en el espacio o por su grosor, tales como el tablero y las patas de una mesa, sillas, estanterías, etc, y (ii) cómo afecta a dichas detecciones la distancia a la que se encuentran del sensor.

Con el fin de comprobar la detección de estos obstáculos se han realizado una serie de pruebas consistentes en colocar objetos de distinto grosor (1, 2 y 4 cm.) frente al sensor, a una determinada altura, y verificar si son detectados a diferentes distancias. El experimento se realizó en un pasillo (ver figura 3), con el sensor Kinect montado sobre un robot móvil, de tal manera que si éste no era capaz de detectar el objeto, la medición devuelta por el sensor estuviera fuera de rango (distancia muy elevada o 0). Para obviar medidas procedentes del suelo del pasillo o a sus paredes, sólo se han considerado para el estudio las 16 columnas de mediciones centrales de las 640 disponibles, que además cumplan ciertas restricciones de altura.

El robot, situado inicialmente a 4 metros de distancia del obstáculo, se fue acercando hasta varios centímetros, utilizándose la información de odometría para calcular la posición del sensor con respecto al obstáculo.

Se realizaron un total de 5 pruebas con cada grosor de objeto, siendo los resultados obtenidos muy similares a los de los casos señalados a continuación. La figura 4 muestra el resultado de dos pruebas: una para cada grosor del objeto a detectar (los resultados para el caso de 4 cm. de grosor han sido omitidos, ya que el objeto era detectado en todo el rango). En ellas el eje vertical representa si en las 16 columnas estudiadas se detectó la presencia del objeto, y en el eje horizontal la distancia del sensor Kinect a éste. Las distancias han sido discretizadas en intervalos de 5cm.

Figura 3. Proceso de prueba de la detección de Kinect de objetos finos. En este caso el objeto longitudinal es de 4cm de grosor.

El resultado de estas pruebas pone de manifiesto el potencial de Kinect para detectar obstáculos pequeños, de hasta 1cm. de grosor, en un rango aceptable de distancias. Cabe señalar como, para este mismo caso, su detección no es constante, es decir, aunque el objeto es detectado inicialmente a una distancia de 2.5m., hay distancias inferiores a ésta en las que el objeto no es detectado, hecho probablemente relacionado con la discretización vertical del sensor.

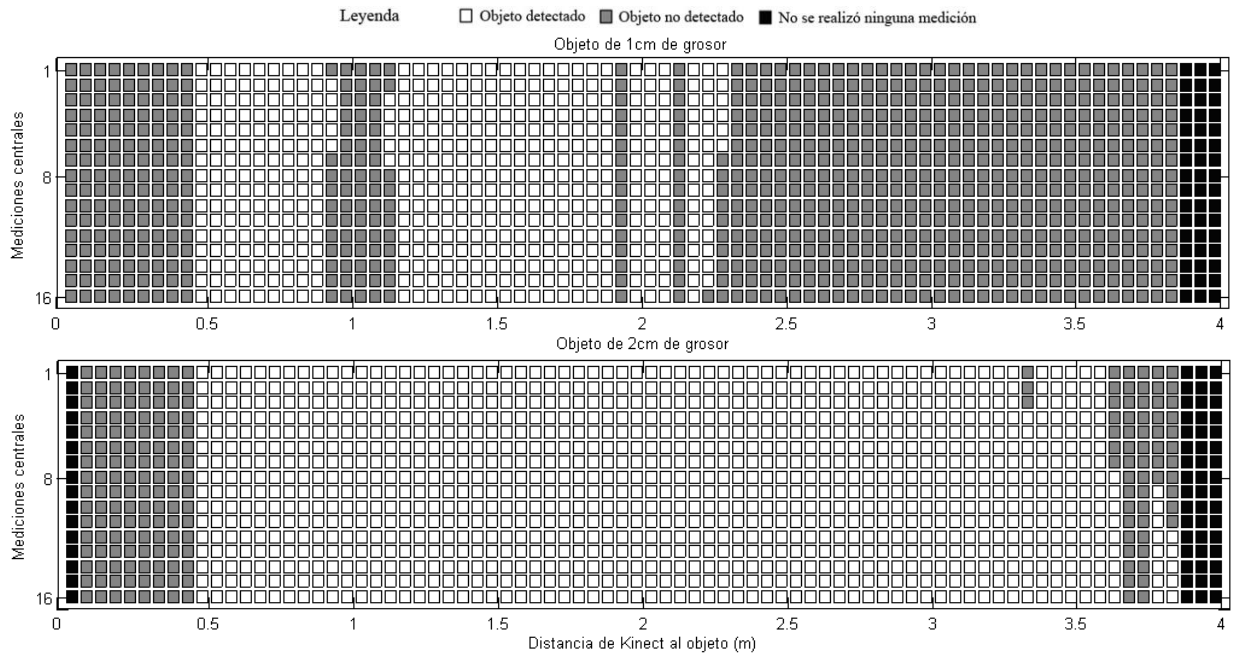

Figura 4. Graficas del resultado de las pruebas de visibilidad.

III. NAVEGACIÓN REACTIVA

Todo robot móvil debe tener una arquitectura software de control que se encargue de planificar y ejecutar las tareas encomendadas al robot en cada momento. En concreto, una de las tareas fundamentales para estos robots es la de navegar hasta un punto de destino de manera segura. En nuestra implementación, las capas altas de la arquitectura de control utilizan un mapa topológico para dividir un comando de navegación en una secuencia de destinos intermedios no excesivamente distantes unos de otros. El camino entre estas localizaciones intermedias se recorre mediante un *navegador reactivo*, que sólo tiene en cuenta los obstáculos existentes en cada instante de tiempo. En esta sección se resume el funcionamiento del navegador reactivo empleado en nuestro robot, explicando el papel que juegan en él los obstáculos detectados mediante el sensor Kinect.

En cualquier algoritmo de navegación reactiva hay que decidir los comandos de velocidades (lineales y rotacionales) instantáneas deseadas para el robot en función del punto de destino y la distribución espacial de los obstáculos. En dicha decisión se deben tener en cuenta simultáneamente dos factores limitantes: (i) un robot móvil es típicamente no holonómico por lo que no puede moverse en cualquier dirección, y (ii) se debe evitar la colisión del robot con los obstáculos, para lo que se debe tener en cuenta el volumen físico ocupado por el robot a lo largo de las trayectorias planificadas. Normalmente ambas restricciones se tienen en cuenta directamente al nivel del algoritmo de navegación reactivo.

En cambio, nuestro método está basado en una serie de transformaciones matemáticas que permiten desacoplar los dos problemas mencionados arriba del algoritmo reactivo propiamente dicho que, operando en una serie de espacios transformados, puede tratar al robot como un simple punto, que es capaz de dirigirse en cualquier dirección sin restricciones cinemáticas.

El proceso completo consta de los siguientes pasos. Primero, se toma la nube de puntos tridimensional formada por los obstáculos detectados por todos los sensores del robot (escáners láser y Kinect en nuestro caso). Estos puntos se simplifican en una nube de puntos bidimensionales, como se explica en la siguiente sección y se ilustra en la figura 7.

A continuación, el problema de la decisión de movimientos debe considerarse en el espacio de configuración (C-Space) del vehículo, típicamente con estructura $R^2 \times S$ (una posición 2D más una orientación) y por lo tanto tridimensional. En dicho espacio el robot se representa completamente como un simple punto, mientras que cada obstáculo puntual se convierte en un sólido, como se ve en la figura 5 (b). A continuación aplicamos una herramienta matemática (que ya presentamos en [1]), basada en la parametrización de familias de trayectorias (*Parameterized Trajectory Generators*, o PTGs) y que permite la transformación de dichos obstáculos tridimensionales a un espacio transformado de parámetros de trayectorias (*Trayectory Parameters Space*, o *TP-Space*). Estos obstáculos se interpretan como la intersección de una familia de posibles trayectorias en el espacio de configuración (la superficie roja de la figura) con los obstáculos tridimensionales, que de esta forma reducen su dimensionalidad a un plano. Además de la simplificación que implica esta reducción de dimensionalidad, cualquier algoritmo reactivo puede aplicarse a continuación sobre estos obstáculos (en el TP-Space) sin tener en cuenta la forma real del robot ni sus restricciones cinemáticas. En nuestro caso, empleamos el algoritmo *Nearness Diagram* [16], que ha demostrado su eficacia para encontrar pasillos libres de obstáculos en entornos con alta densidad de obstáculos.

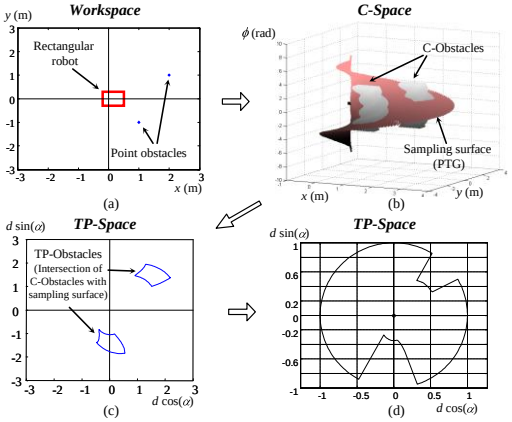

Figura 5. El fundamento del sistema de navegación reactivo empleado en este trabajo. (a) Los puntos representan obstáculos en el espacio de trabajo (Workspace) del robot, que vistos en el espacio de configuración (C-Space) aparecen como sólidos tridimensionales, cuya intersección con la PTG empleada da lugar a (c) los obstáculos en el espacio de parámetros (TP-Space). En la práctica sólo es necesario guardar la distancia mínima a un obstáculo en dichos obstáculos (d) para realizar la navegación reactiva.

IV. INTEGRACIÓN DE KINECT EN LA NAVEGACIÓN REACTIVA

La integración de Kinect en un navegador reactivo no es directa, presentándose principalmente dos problemas: la gran cantidad de datos que proporciona el sensor en cada medición, y la existencia de una zona ciega, en torno a 1m., delante del robot. La solución que proponemos para el primero es realizar un procesamiento rápido de los puntos 3D que simplifique la información sensorial, sin mermar la efectividad en la detección de obstáculos. Para el segundo problema se ha implementado una memoria a corto plazo que almacena y proyecta en el espacio, dependiendo del movimiento del robot, la información percibida para cada pose (x,y,Φ) de éste (ver figura 6).

A. *Simplificación de las medidas*

El sensor Kinect proporciona en cada frame más de 300.000 puntos, cuyo procesamiento por parte de cualquier algoritmo de navegación reactiva a una frecuencia de 30Hz. es difícil de asumir.

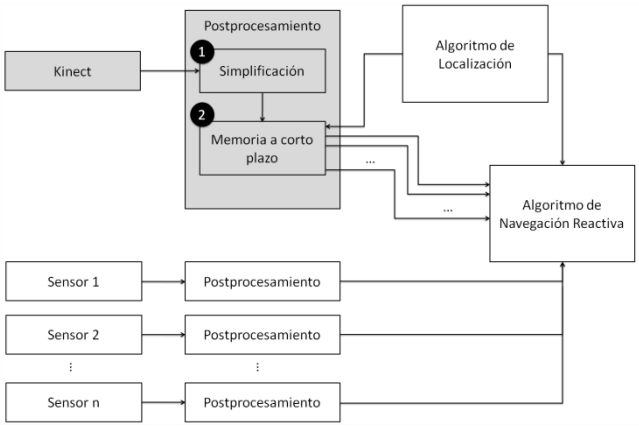

Figura 6. Esquema general de un algoritmo de navegación reactiva que considera diferentes sensores para percibir el entorno, incluido Kinect.

Tal y como ocurre en la mayoría de las aplicaciones de robótica móvil, el robot se considera plano y su forma viene determinada por el mayor polígono que circuscribe las distintas secciones en altura de éste. Por tanto, carece de sentido proporcionarle al navegador reactivo toda la informacion espacial 3D del entorno, que no va a explotar, y que, por su volumen, puede poner en peligro otras tareas simultáneas del robot.

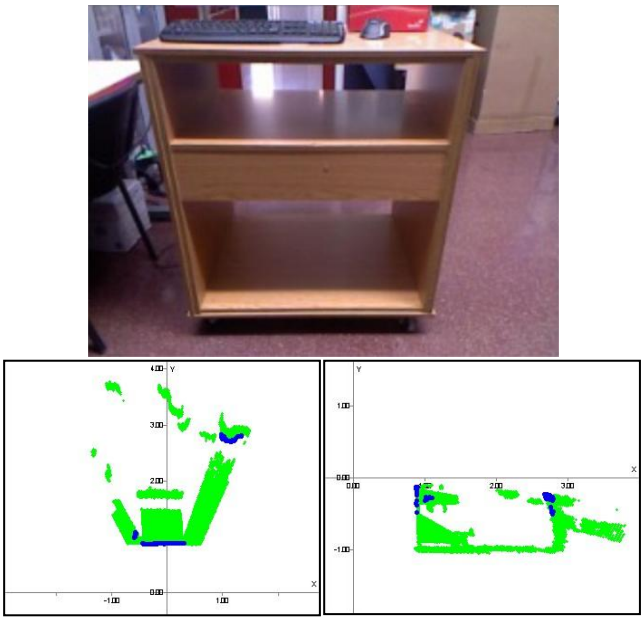

Figura 7. Arriba, imagen de la escena capturada con la cámara RGB. Abajo izquierda, vista superior de la escena percibida por Kinect. Abajo derecha, vista lateral de la misma escena. Los puntos verdes son mediciones desechadas, y los azules mediciones empleadas para la navegación.

Una solución eficiente y que no compromete la eficacia del algoritmo reactivo utilizado consiste en reducir la información 3D obtenida por Kinect y proyectarla sobre el plano 2D en el que se encuentra el robot. Considerando únicamente la medición más cercana, esta simplificación se realizaría calculando el mínimo de cada una de las columnas de la imagen de rango. Es decir, el resultado de esta simplificación consistiría en un barrido horizontal que contiene las mediciones más cercanas de los obstáculos presentes a diferentes alturas (ver figura 7).

Esta técnica podría generalizarse para el caso en que el algoritmo reactivo tenga en cuenta la forma tridimensional del robot para trabajar en un espacio de configuración (X,Y,Z,Φ). Así, si el volumen del robot se modela con más de una sección prismática, la simplificación de medidas de Kinect que se propone aquí se realizaría para cada una de estas secciones, esto es, por bandas en Z. La solución implementada en este trabajo, por tanto, sería un caso particular de una única banda en Z.

B. *Memoria a corto plazo*

El rango de trabajo de Kinect certificado por el fabricante genera una zona ciega de 1.2 m., es decir, objetos que se encuentren dentro de un radio de 1.2 m. podrían pasar desapercibidos para el sensor. Para evitar que el robot colisione con objetos presentes en esta zona, se ha desarrollado una memoria a corto plazo en torno a la posición del robot que mantiene y propaga espacialmente un conjunto de mediciones previas. Dicha memoria se ha implementado mediante una rejilla, de forma análoga a un mapa de ocupación (ver figura 8), y se formaliza matemáticamente de la siguiente forma.

Memoria

Sea *M*, memoria a corto plazo, definida como una matriz *n* x *m*, que discretiza parte del espacio bidimensional de trabajo del algoritmo reactivo. M(i,j) representa la probabilidad de que la celda <i,j>, $c_{i,j}$, esté ocupada por un obstáculo en base a las observaciones, $o_1, \ldots, o_k$, del robot. Es decir, para cada celda se calcula

$$M(i,j) = p(c_{i,j} \mid o_1, \ldots, o_k)$$

Para el cómputo de dicha probabilidad, por razones de eficiencia, se recurre a los denominados *log-odds*, es decir, para cada $c_{i,j}$ en el instante de tiempo *t* se calcula (ver [17]):

$$l^t(c_{i,j}) = \log \frac{p(c_{i,j} \mid o_1, \ldots, o_k)}{1 - p(c_{i,j} \mid o_1, \ldots, o_k)}$$

Nótese que l^t puede tomar cualquier valor real y que, dado dicho valor, se puede recuperar la probabilidad de cada celda como:

$$p(c_{i,j} \mid o_1, \ldots, o_k) = 1 - \frac{1}{e^{l^t(c_{i,j})}}$$

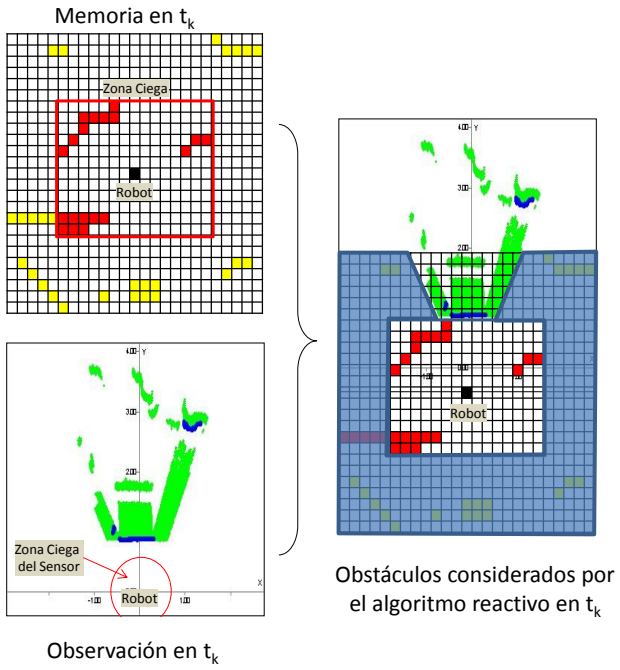

Figura 8. Memoria a corto plazo en un instante de tiempo t_k. La memoria almacena mediciones previas del sensor en una rejilla. A la derecha, los obstáculos considerados por el navegador reactivo resultan de la integración de los puntos de la zona ciega (en rojo) con la observación actual del sensor.

Creación y actualización de la memoria

Sea o_t una observación del sensor Kinect en un instante de tiempo t formada por el conjunto de medidas simplificadas desde una cierta pose del robot representada por la celda $r=(r_x,r_y)$. Sea $c_i=(c_{ix},c_{iy})$ la celda que corresponde a la medición i-esima, una vez considerada la posición y orientación del robot. La memoria se crea y actualiza en base a los siguientes pasos:

1.- *Inicialización*. Inicialmente, en ausencia de observaciones, las celdas de la memoria se inicializan con el valor:

$$l^0(c_{i,j}) = \log \frac{p(c_{i,j})}{1-p(c_{i,j})}$$

donde $p(c_{i,j})$ se establece a un valor de 0.5.

2.- *Actualización de la memoria en base a nuevas observaciones*. Utilizando la regla de Bayes y *log-odds*, la actualización del valor de $l^t(c_{i,j})$ ante una nueva observación o_t, se realiza mediante la siguiente expresión (ver [17]):

$$\forall c_{ij}, l^t(c_{i,j}) = l^{t-1}(c_{i,j}) + \log\left(\frac{p(c_{i,j}\mid o_t)}{1-p(c_{i,j}\mid o_t)}\right) + \log\left(\frac{1-p(c_{i,j})}{p(c_{i,j})}\right)$$

donde la probabilidad a posteriori, $p(c_{i,j}\mid o_t)$ se calcula, aplicando Bayes, en base al modelo del sensor utilizado que corresponde a:

$$p(o_t \mid M, c_{i,j}=1) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(o_t-d_{i,j})^2}{\sigma^2}}$$

donde $d_{i,j}$ es la distancia al centro de la celda $c_{i,j}$. El término $\log\left(\frac{1-p(c_{i,j})}{p(c_{i,j})}\right)$ resulta ser una constante que depende del valor de la probabilidad a priori seleccionado, $p(c_{i,j})$, resultando que si, $p(c_{i,j})=0.5$, el cálculo para la actualización de la memoria se simplifica a:

$$\forall c_{ij}, l^t(c_{i,j}) = l^{t-1}(c_{i,j}) + \log\left(\frac{p(c_{i,j}\mid o_t)}{1-p(c_{i,j}\mid o_t)}\right)$$

Zona Ciega

Dada la posición del robot por la celda $r=(r_x,r_y)$, se define la zona ciega de la memoria, Z_c, como el conjunto de celdas $c_{i,j}$ tal que su distancia a r es menor que un cierto umbral u_c, es decir,

$$Z_c = c_{i,j}, dist(r, c_{i,j}) < u_c$$

donde u_c es la distancia umbral que marca la zona ciega del sensor, y la función *dist* define la distancia de Manhattan en el conjunto de celdas C, es decir:

$$dist : CxC \rightarrow \mathbb{N}$$
$$\forall c_1 = (c_{1x}, c_{1y}), c_2 = (c_{2x}, c_{2y})$$
$$dist(c_1, c_2) = |c_{1x} - c_{2x}| + |c_{1y} - c_{2y}|$$

En nuestra implementación el tamaño de la memoria y de la zona ciega se ha fijado en 75 y 50 celdas, respectivamente, es decir, a 1.5 m. y 1 m., respectivamente, ya que el tamaño de celda considerado es de 2 cm. Por su parte, el algoritmo reactivo recibe como mapa de obstáculos en cada instante t, los puntos simplificados de la observación de Kinect, O_t, junto con puntos (centros) de las celdas de la zona ciega cuya $p(c_{i,j}|o_1,\ldots o_t)$ sea mayor que 0.9 (ver figura 8).

V. Implementación y Discusión

Con el objetivo de probar las mejoras que ofrece el uso de Kinect se han realizado una serie de experiencias empleando para ello al robot móvil SANCHO [18] (ver figura 9), desarrollado por el grupo MAPIR (http://mapir.isa.uma.es).

SANCHO ha sido construido empleando una base comercial Pioneer 3DX sobre la que se han integrado varios dispositivos y sensores para realizar eficazmente tareas de alto nivel. Entre estos componentes hardware destacan una pantalla táctil, dos altavoces y un micrófono para la interacción con el usuario, un conjunto de bumpers, dos sensores de rango 2D Hokuyo [19] y un sensor Kinect para la percepción del entorno. Por su parte, el ordenador que

controla al robot posee un microprocesador Intel® Core™ i7-860 a 2,8GHz y una memoria RAM de 4GB. Todo el software corre bajo el sistema operativo Windows7™ de 32 bits.

Figura 9 Robot móvil Sancho.

La arquitectura de control presente en SANCHO se denomina OpenMORA (Open Mobile Robot Architecture) [20] y ha sido desarrollada por el grupo MAPIR [21]. Está basada en la arquitectura de control presentada en [22] y presenta, entre otras, dos características principales:

- Sigue un diseño e implementación modular, en el que cada módulo implementa un conjunto de habilidades que están dirgidas a alcanzar un objetivo particular.
- La comunicación entre los distintos módulos se realiza mediante una pizarra compartida, implementada en el middle-ware MOOS [23] que está basado en comunicaciones sobre el protocolo TCP/IP.

En las pruebas realizadas con SANCHO se plantearon situaciones en las que sin usar Kinect el robot colisiona con obstáculos situados a diferentes alturas, al no ser percibibles por los lásers 2D, mientras que con su uso se consigue esquivarlos. Un ejemplo de estas situaciones es la mostrada en la figura 10.

Figura 10. Escenario de realización de una de las pruebas. Las líneas azules indican a qué altura percibe la mesa el escáner 2D, y el círculo indica la posición de destino del algoritmo reactivo.

En ella, se le ordena al robot SANCHO a que se desplace a un punto que se encuentra detrás de una mesa. En la figura 11 (a) puede verse cómo los escáners 2D perciben la mesa a una altura en la cual sólo detectan sus laterales, por lo que para éstos hay un espacio libre de obstáculos entre ellos. Sin embargo, el sensor Kinect sí es capaz de detectarla a otras alturas, lo que permite evitar este obstáculo empleando las soluciones propuestas. De este modo, la figura 11 (a) se corresponde con la posición inicial del robot, la figura 11 (b) con como es capaz de bordear el obstáculo, y la figura 11 (c) ilustra el momento alcanza la posición objetivo sin colisiones.

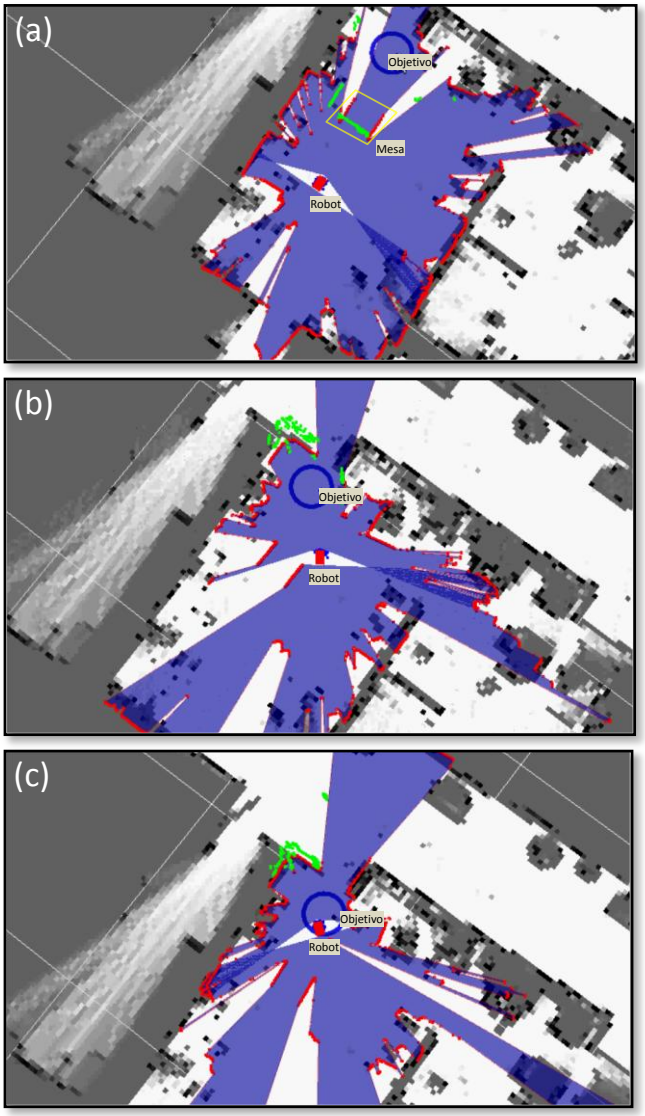

Figura 11. Evolución de la trayectoria del robot tras recibir la orden de desplazarse a una posición objetivo detrás de la mesa. En la figura (a) se ha bordeado de color amarillo la posición de la mesa con fines ilustrativos. Los puntos rojos representan mediciones de los sensores láser 2D, y los puntos verdes mediciones de Kinect.

VI. CONCLUSIONES

En ese trabajo se han presentado las características del sensor Kinect y su integración con éxito en una aplicación en el campo de la robótica móvil. La información 3D aportada por este sensor es empleada por el sistema de navegación de un robot móvil mejorando notablemente la robustez y eficacia del mismo.

En un futuro se plantea la utilización del sensor Kinect en la construcción de mapas métricos, de manera que la información 3D de la cámara de rango también sea útil para la localización del robot en el entorno. Además, la cámara RGB, los micrófonos y el movimiento de cabeceo son prestaciones a explotar que auguran una gran actividad investigadora en torno a este sensor revolucionario.

REFERENCIAS

[1] J.L. Blanco, J. Gonzalez, J.A. Fernandez-Madrigal, "Extending Obstacle Avoidance Methods through Multiple Parameter-Space Transformations", Autonomous Robots, vol. 24 (1), pp. 29-48, 2008.

[2] J.R. Alvarez-Sanchez, F. Lopez, J.M. Cuadra, D. Santos. "Reactive navigation in real environments using partial center of area method". Robotics and Autnomous Systems Vol. 58, n.12, 2010.

[3] Y. Wang and D. Chen. "Autonomous Navigation based on a novel Topological Map". 2009 Asia-Pacific Conference on Information Processing.

[4] Fiorini P. and Shiller Z. "Motion Planning in Dynamic Environments using Velocity Obstacles". 1998, in The International Journal of Robotics Research, vol. 17, no. 7, pp. 760–772.

[5] Zhijun Li, Zhaoxian Xie, Aiguo Ming, "Simultaneously firing sonar ring based high-speed navigation for non-holonomic mobile robots in unstructured environment", International Journal of Vehicle Autonomous Systems, Volume 6, Number 1-2/2008.

[6] R. Vázquez-Martín, P. Núñez, A. Bandera, F. Sandoval. "Curvature-Based Environment Description for Robot Navigation Using Laser Range Sensors", 2009, Sensors 9, no. 8: 5894-5918.

[7] D. Murray, J. J. Little. "Using Real-Time Stereo Vision for Mobile Robot Navigation", in Autonomous Robots, vol. 8 (2), pp. 161-171.

[8] D. Holz, C. Lörken, and H. Surmann. Continuous 3D Sensing for Navigation and SLAM in Cluttered and Dynamic Environments. International Conference on Information Fusion of the (FUSION), 2008. Cologne, Germany, pages 1469-1475.

[9] Kinect (página official): http://www.xbox.com/es-ES/kinect

[10] API oficial Kinect http://research.microsoft.com

[11] OpenKinect: http://www.openkinect.org

[12] Code laboratories: http://codelaboratories.com/

[13] Patente Depth Mapping using Projected Patterns. Pub. No.: US 2010/0118123 A1. Publicada el 13 de Mayo del 2010.

[14] PrimeSense: http://www.primesense.com/

[15] Patente Range Mapping Using Speckle Decorrelation. Número de patente: US 7,433,024 B2. Publicada el 7 de Octubre del 2008.

[16] J. Minguez, L. Montano. Nearness diagram (ND) navigation: collision avoidance in troublesome scenarios, IEEE Transactions on Robotics and Automation, v.20, pp. 45-59, 2004.

[17] S. Thrun. Learning occupancy Grid Maps with Forward Sensor models. Autonomous Robots, n. 15, pp. 111-127, 2003.

[18] J. Gonzalez, C. Galindo, J.L. Blanco, J.A. Fernandez-Madrigal, V. Arevalo, F.A. Moreno, "SANCHO, a Fair Host Robot. A Description", IEEE International Conference on Mechatronics (ICM'09), Malaga (Spain), 2009.

[19] Hokuyo: http://www.hokuyo-aut.jp/

[20] OpenMORA: http://sourceforge.net/p/openmora/home/

[21] Mapir. http://mapir.isa.uma.es

[22] C. Galindo, J. Gonzalez, J.A. Fernandez-Madrigal, "A Control Architecture for Human-Robot Integration. Application to a Robotic Wheelchair", IEEE Transactions on Systems, Man, and Cybernetics, part B, vol. 36, no. 5, pp. 1053-1068, 2006.

[23] The MOOS Cross Platform Software for Robotics Research www.robots.ox.ac.uk/~pnewman/TheMOOS