MODELADO CINEMÁTICO Y SIMULACIÓN REALISTA DEL MANIPULADOR MÓVIL DE BAJO COSTE *TURTLEBOT2 + WIDOW-X* EN *ROS*

Marina Aguilar, Ana Cruz-Martín y Juan-Antonio Fernández-Madrigal
Departamento de Ingeniería de Sistemas y Automática
Universidad de Málaga
marinaaguilar@uma.es, anacm@uma.es, jafernandez@uma.es

Resumen

El manipulador móvil compuesto de la plataforma Turtlebot2 y el brazo Widow-X supone una solución de bajo coste muy interesante para laboratorios docentes y de investigación. Lamentablemente, el conjunto no dispone de un simulador completo en el popular framework de programación ROS; tampoco el fabricante ofrece el modelado cinemático del brazo. En este artículo se describe el trabajo realizado sobre ambas cuestiones para nuestro robot CRUMB, y cómo se han implementado en Gazebo, un simulador realista para ROS donde también se ha añadido la posibilidad de incluir ruido sensorial simple, algo que la mayoría de los simuladores no ofrecen. Se incluyen también algunos experimentos de manipulación y navegación que demuestran las posibilidades de nuestro trabajo.

Palabras clave: manipulador móvil, simulación, modelado cinemático, *Turtlebot2*, *Widow-X*, *ROS*, *Gazebo*.

Figura 1: Manipulador móvil *CRUMB*.

1. INTRODUCCIÓN

Como parte del proyecto puente de investigación denominado *CRUMB* [1] (*Cognitive-RoboticssUpporting Mobile Base*), financiado a través del Plan Propio de Investigación de la Universidad de Málaga, se adquirió un manipulador móvil de bajo coste sobre el que implementar desarrollos de Robótica Cognitiva. Este robot está formado por una base móvil *Turtlebot2* [2] y un brazo manipulador *Widow-X* [3], mostrados en la figura 1.

Para facilitar que varios miembros del equipo investigador pudieran trabajar de manera simultánea con esta plataforma robótica, se decidió crear un simulador realista compatible con el robot real, de manera que los algoritmos desarrollados en el primero pudieran utilizarse en el robot *CRUMB* con mínimos cambios, y que se incorporaran efectos suficientemente realistas en cinemática, dinámica y ruido de los sensores. Como el robot debía ser capaz de realizar tareas de manipulación, en primer lugar era necesario obtener el modelo cinemático del brazo manipulador *Widow-X*, no disponible por parte del fabricante, y resolver los problemas directo e inverso. Además, el simulador del robot completo debía estar integrado en *ROS* [4], dada su popularidad como *framework* de programación en la comunidad Robótica y por tanto, su idoneidad para compartir desarrollos de diversos investigadores.

En este artículo se presenta el trabajo realizado bajo los mencionados objetivos, concretamente el modelado cinemático del brazo y la simulación realista del manipulador móvil en *Gazebo* [5] (que se encarga de la dinámica realista), así como la implementación de un interfaz de programación prácticamente idéntico al del robot real (es decir, los *topics* de *ROS* necesarios) y la inclusión de ruido simple en los sensores.

El texto se organiza de la siguiente forma: en la sección 2 se describe el cálculo del modelado cinemático directo e inverso del brazo manipulador *Widow-X*; en la sección 3 se detalla cómo se ha implementado el simulador para *CRUMB* en *ROS* y *Gazebo*, incluyendo el modelado geométrico y de los parámetros físicos y de control que hemos realizado; en la sección 4 se muestran los experimentos realizados con el simulador completo; finalmente, en la sección 5 se resumen las conclusiones y las líneas de trabajo futuras.

2. PROBLEMAS CINEMÁTICO DIRECTO E INVERSO EN EL BRAZO *WIDOW-X*

Para realizar tareas de manipulación (p.ej., *pick and place*) con el brazo *Widow-X* es necesario al menos su modelado cinemático. Este brazo cuenta con 5 g.d.l y una pinza paralela como efector final.

El modelo cinemático se basará en el producto de las matrices de paso correspondientes a cada una de las articulaciones. Para calcular dichas matrices de paso se necesitan los parámetros de Denavit-Hartenberg [6], que son (ver figura 2):

- a_i: distancia desde el eje Z_i hasta el eje Z_{i+1} medida a lo largo de X_i.
- α_i: ángulo que forman los ejes Z_i y Z_{i+1} medido sobre X_i.
- d_i: distancia desde el eje X_{i-1} hasta el eje X_i medida a lo largo de Z_i.
- θ_i: ángulo que forman los ejes X_{i-1} y X_i medido sobre Z_i.

En el caso del *Widow-X*, estos parámetros son los mostrados en el cuadro 1. Con ellos se obtienen las matrices de paso 1, 2, 3, 4 y 5.

Cuadro 1: Parámetros D-H del manipulador *Widow-X*.

Articulación	a_{i-1}	α_{i-1}	d_i	θ_i
1	0	0	0	θ_1
2	0	-π/2	0	θ_2-β
3	d2	π	0	θ_3-γ
4	L3	0	0	θ_4+π/2
5	0	π/2	0	θ_5

$$T_1^0 = \begin{bmatrix} \cos\theta_1 & -\sin\theta_1 & 0 & 0 \\ \sin\theta_1 & \cos\theta_1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

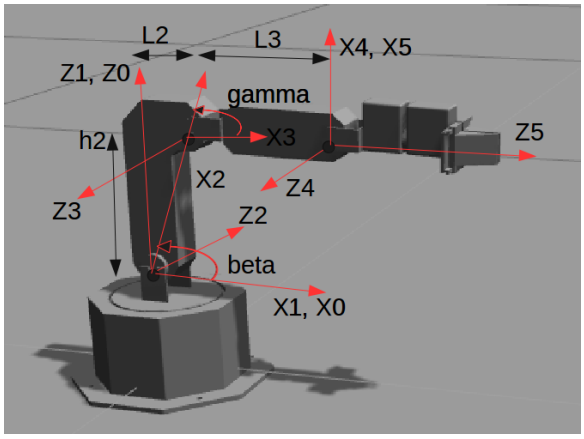

Figura 2: Ejes y dimensiones en el *Widow-X* para el cálculo de los parámetros Denavit-Hartenberg.

$$T_2^1 = \begin{bmatrix} \cos(\theta_2-\beta) & \sin(\theta_2-\beta) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \sin(\theta_2-\beta) & -\cos(\theta_2-\beta) & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

$$T_3^2 = \begin{bmatrix} \cos(\theta_3-\beta) & \sin(\theta_3-\beta) & 0 & d_2 \\ \sin(\theta_3-\beta) & -\cos(\theta_3-\beta) & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (3)$$

$$T_4^3 = \begin{bmatrix} \cos(\theta_4+\pi/2) & -\sin(\theta_4+\pi/2) & 0 & L_3 \\ \sin(\theta_4+\pi/2) & \cos(\theta_4+\pi/2) & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (4)$$

$$T_5^4 = \begin{bmatrix} \cos\theta_5 & -\sin\theta_5 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ \sin\theta_5 & \cos\theta_5 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (5)$$

El modelo cinemático será el indicado en 6. El punto obtenido con el mismo será el nombrado como O_m en la figura 3, que corresponde a la muñeca.

$$T_5^0 = T_1^0 T_2^1 T_3^2 T_4^3 T_5^4 \quad (6)$$

El problema cinemático directo consiste básicamente en la realización del producto anterior con una postura articular dada. El inverso necesita la resolución de las mismas ecuaciones cuando tal postura es desconocida. En nuestro caso se ha obtenido con un procedimiento algebraico mediante desacoplo de la muñeca respecto del brazo: en primer lugar se obtienen las variables articulares θ_1,

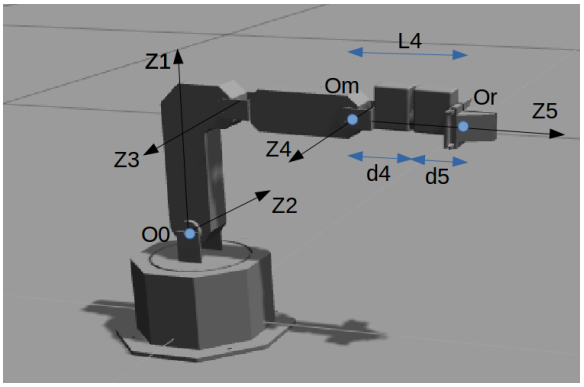

Figura 3: Posición de la muñeca y efector final.

θ_2 y θ_3 en función de un punto genérico, y a continuación θ_4 y θ_5 para una orientación genérica de la muñeca.

Para ello, partiendo de la matriz T_5^0, se modifica la última columna para introducir el punto genérico de la muñeca (p_x, p_y, p_z), obteniéndose la matriz Tp_5^0 de la ecuación 7, en la que los elementos r_{ij} son los elementos ij del modelo cinemático.

$$Tp_5^0 = \begin{bmatrix} r_{11} & r_{12} & r_{13} & p_x \\ r_{21} & r_{22} & r_{23} & p_y \\ r_{31} & r_{32} & r_{33} & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7}$$

La matriz Tp_5^0 debe ser igual a T_5^0, con lo que se obtiene la ecuación 8d.

$$Tp_5^0 = T_1^0 T_2^1 T_3^2 T_4^3 T_5^4 \tag{8a}$$
$$[T_1^0]^{-1} Tp_5^0 = T_2^1 T_3^2 T_4^3 T_5^4 \tag{8b}$$
$$T_0^1 Tp_5^0 = T_2^1 T_3^2 T_4^3 T_5^4 \tag{8c}$$
$$Tp_5^1 = T_2^1 T_3^2 T_4^3 T_5^4 \tag{8d}$$

Resolviendo el sistema de ecuaciones resultante se obtienen las variables articulares θ_1, θ_2 y θ_3 (ecuaciones 9a, 9b y 9c, respectivamente).

$$\theta_1 = A\tan 2(p_y, p_x) \tag{9a}$$
$$\theta_2 = \beta + A\tan 2(k_1, k_2) - A\tan 2(k, \pm\sqrt{k_1^2 + k_2^2 + k^2}) \tag{9b}$$
$$\theta_3 = \theta_2 - A\sin\left(\frac{-p_z - d_2 sin\beta}{L_3}\right) \tag{9c}$$

donde k, k_1 y k_2 son, respectivamente, 10a, 10b y 10c.

$$k = p_z^2 + d_2^2 - L_3^2 + (p_x\cos\theta_1 + p_y\sin\theta_1)^2 \tag{10a}$$
$$k_1 = 2p_x\cos\theta_1 d_2 + 2p_y\sin\theta_1 d_2 \tag{10b}$$
$$k_2 = 2p_z d_2 \tag{10c}$$

Para obtener θ_4 y θ_5, se utiliza la orientación de la muñeca. Sea R_5^0 la submatriz de rotación de T_5^0 para una orientación genérica $[\hat{n}, \hat{o}, \hat{a}]$ (ecuación 11).

$$R_5^0 = [\hat{n}, \hat{o}, \hat{a}] = \begin{bmatrix} n_x & o_x & a_x \\ n_y & o_y & a_y \\ n_z & o_z & a_z \end{bmatrix} \tag{11}$$

Se tiene la igualdad 12:

$$R_5^0 = R_3^0 \cdot R_5^3 \tag{12}$$

donde R_3^0 es conocido ya que es el producto en la ecuación 13a de las submatrices de rotación mostradas, que dependen de θ_1, θ_2 y θ_3, ya obtenidos previamente, y R_5^3 el producto de la ecuación 13b, cuyo resultado depende de θ_4 y θ_5.

$$R_3^0 = R_1^0 \cdot R_2^1 \cdot R_3^2 \tag{13a}$$
$$R_5^3 = R_4^3 \cdot R_5^4 \tag{13b}$$

Al resolver el sistema de ecuaciones resultante se obtienen las variables articulares de la muñeca, ecuaciones 14a y 14b.

$$\theta_4 = A\cos(-k_3) - \frac{\pi}{2} \tag{14a}$$
$$\theta_5 = A\sin(n_x\sin\theta_1 - n_y\cos\theta_1) \tag{14b}$$

donde k_3 es:

$$k_3 = a_z\cos(\theta_2 - \theta_3) + a_x\sin(\theta_2 - \theta_3)\cos\theta_1 + a_y\sin(\theta_2 - \theta_3)\sin\theta_1 \tag{15}$$

Finalmente, se obtiene la postura del efector final O_r en función de la de la muñeca O_m, obtenida a su vez del modelo cinemático (figura 3), de donde se deduce la ecuación 16.

$$\vec{P_m} = \vec{P_r} - L_4 \cdot \hat{Z_5} \tag{16}$$

3. MODELADO DEL ROBOT PARA SU SIMULACIÓN

En esta sección se describe el modelado geométrico, físico y de controladores necesario para la simulación en *Gazebo* del brazo manipulador *Widow-X*, así como el de la base móvil *Turtlebot2* y del robot completo.

3.1. BRAZO *WIDOW-X*

El modelado del brazo *Widow-X* se divide en modelado geométrico, modelado de parámetros físicos, modelado de los controladores y configuración de los *plugins* necesarios en *Gazebo* [7].

3.1.1. Modelado geométrico

La geometría del brazo se ha modelado partiendo de los ficheros *CAD .stl* facilitados por *Robotnik Automation S.L.L.* [8], distribuidores de este manipulador. Dicho modelado se describe en formato *XML*, donde se diferencian dos elementos: eslabones y articulaciones.

En los eslabones se indican dos tipos de geometrías: la visible y la de detección de colisiones, donde se añade el fichero *.stl* junto con el origen de coordenadas y la orientación del eslabón. Para indicar la relación entre eslabones se utilizan las articulaciones. En este trabajo se utilizan uniones de revolución, prismáticas y fijas. En las dos primeras se indica el punto de contacto, el eje de giro, los límites del movimiento y cuáles son los eslabones fijo y móvil. También se puede indicar si una pieza imita el movimiento de otra, como en el caso de la pinza. Para las uniones fijas se indican las dos piezas que se unen, en este caso la base del brazo con la superficie superior de la base móvil.

También es necesario añadir propiedades de color y autocolisión en los ficheros de configuración, para que el modelo utilice la detección de colisiones para no atravesarse a sí mismo y para mejorar su aspecto visual.

3.1.2. Modelado de parámetros físicos

Para que el modelo sea lo más realista posible se han incluido en el mismo parámetros físicos como la inercia, el amortiguamiento y la fricción.

Para cada uno de los eslabones se ha calculado la masa, el centro de masas y el tensor de inercia respecto al centro de masas. El software utilizado ha sido *MeshLab* [9] para los centros de masa y *V-REP* [10] para la masa y el tensor de inercia.

Para comprobar que las inercias son correctas *Gazebo* dispone de un visor. En la figura 4 se observa, por ejemplo, cómo la geometría del brazo y la generada por la inercia son coincidentes.

El amortiguamiento y la fricción de los eslabones se han obtenido experimentalmente bajo la premisa de que el brazo no presente vibraciones. Se ha aumentado la resolución de los métodos numéricos durante la simulación para mejorar los resultados obtenidos.

Finalmente, para poder realizar tareas con la pinza hay que indicar cuáles son las partes móviles y fijas de la misma, el tiempo para agarrar y soltar el objeto y el rozamiento y el contacto. Todos estos parámetros se han obtenido de manera experimental persiguiendo que las piezas no se queden pegadas ni resbalen.

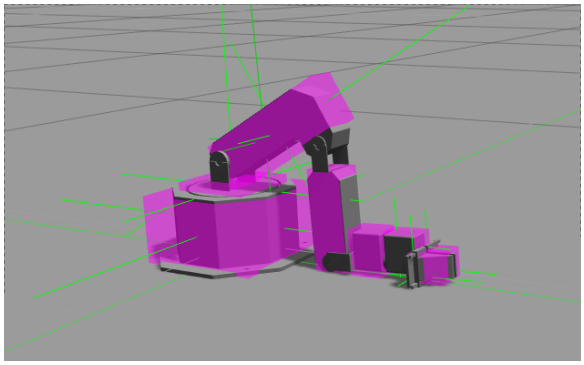

Figura 4: *Widow-X* en el visor de inercia de *Gazebo*.

3.1.3. Modelado de controladores

Se ha elegido un control de esfuerzo/posición para la simulación de las articulaciones, ya que experimentalmente es el que mejores resultados ha ofrecido. El controlador PID se ha sintonizado de manera experimental.

También se han incluido el esfuerzo y la velocidad máxima de las articulaciones. Para estos datos se han utilizado los del brazo *Widow-X* real.

3.1.4. *Plugins Gazebo*

Para añadir las funcionalidades necesarias para la simulación realista de los modelos en *Gazebo* ha sido necesario incluir los siguientes *plugins*:

- *Ros_Control*: necesario para simular cualquier robot, permite que los controladores puedan actuar sobre las articulaciones.

- *Mimic*: necesario para las articulaciones de la pinza, donde las dos partes se mueven de manera simétrica y al mismo tiempo.

3.2. PLATAFORMA MÓVIL *TURTLEBOT2*

La plataforma *Turtlebot2* está basada en una base diferencial *Kobuki* [11] que proporciona energía para el ordenador externo, sensores (cámara *kinect*, *encoders* magnéticos incrementales, *IMU*, *bumpers*, sensores anticaída y antivuelco), actuadores, indicadores y botones. El modelo de la plataforma ya estaba disponible en el repositorio *ROS*, junto con el *stack* de software de navegación que se utiliza en el experimento 2. Sin embargo, para que su aspecto visual sea más real se ha modificado la geometría de colisión para que coincida con la de *CRUMB*.

3.3. MANIPULADOR MÓVIL *CRUMB*

Uniendo los modelos del brazo *Widow-X* y de la plataforma *Turtlebot2* se ha obtenido el robot completo *CRUMB*, mostrado en la figura 5.

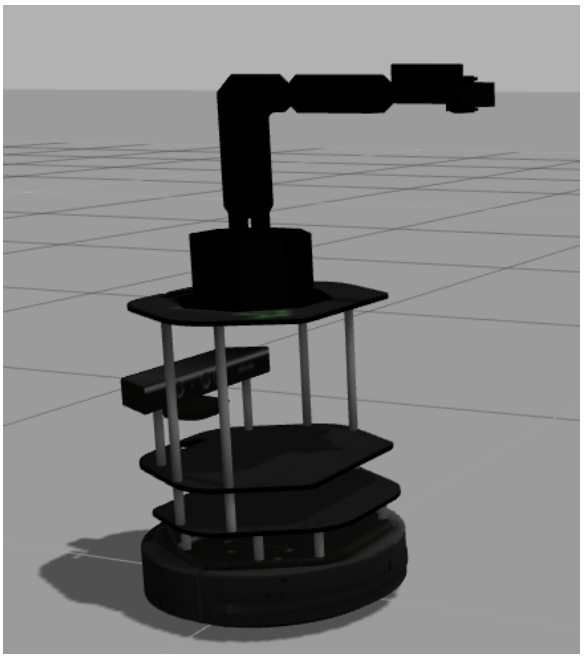

Figura 5: Modelo completo de *CRUMB* en *Gazebo*.

Este modelo viene ya asociado en *ROS* con los *topics* de programación más importantes existentes en el robot real:

- Comandos de actuación y estado de todas las articulaciones del brazo
- Lectura de los sensores *IMU*, cámara *kinect*, *bumpers*, sensores anticaída.
- Datos de odometría de la plataforma.
- Datos del estado completo del sistema, base móvil y brazo.

Sin embargo, otros *topics* del robot real han tenido que ser incluidos adicionalmente:

- Lectura de los sensores *wheel drop* y LEDs indicadores.
- Lectura de todos los sensores de la plataforma en un mismo *topic*.

También se han añadido *topics* útiles para simulación que no existen en real, como la postura real de la plataforma, el tiempo real y el de simulación.

Finalmente, para que el modelo sea más realista, se ha implementado ruido simulado en el sensor inercial y en la cámara *kinect*, obteniéndose así 2 modelos de *CRUMB* simulado distintos, uno con ruido y otro sin ruido, una característica que no es habitual en la simulación con *Gazebo* y que puede ser muy útil para realizar experimentos de diversos tipos.

Como ejemplo de esta última característica, la figura 6 muestra la lectura de la cámara *kinect* sin ruido y con ruido gaussiano.

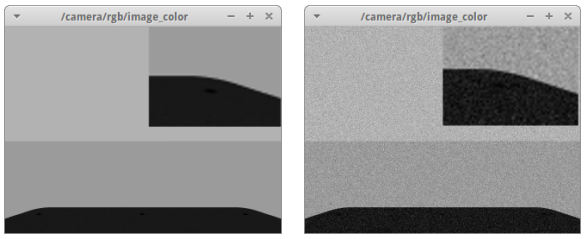

Figura 6: Cámara *kinect* sin ruido (izda) y con ruido (dcha).

4. EXPERIMENTOS

Para validar el trabajo presentado aquí y mostrar su utilidad, se han desarrollado tres aplicaciones experimentales, que se detallan a continuación.

4.1. EXPERIMENTO 1. *PICK-AND-PLACE*

En este experimento el robot permanece fijo en el escenario mostrado en la figura 7 mientras el brazo va moviendo los objetos según programa el usuario. Hay tres posiciones posibles de *pick-and-place*: dos para los objetos y una libre.

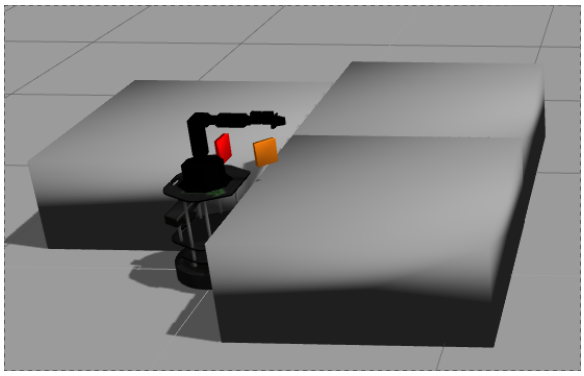

Figura 7: Escenario para *pick-and-place*.

En la figura 8 se observa cómo *CRUMB* va moviendo los objetos hacia el hueco libre. En la referencia [12] se encuentra el vídeo para ver la secuencia completa.

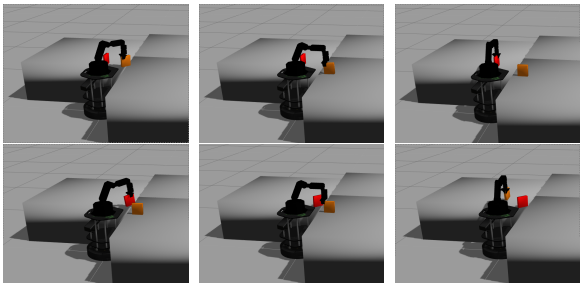

Figura 8: Secuencia de movimiento *pick-and-place*.

4.2. EXPERIMENTO 2. MAPEADO DEL ENTORNO Y NAVEGACIÓN

En el segundo experimento, con el *stack* de navegación de *ROS*, *CRUMB* crea un mapa de su entorno para, posteriormente, navegar a una posición indicada evitando obstáculos. En la figura 9 se muestra el escenario creado en *Gazebo*.

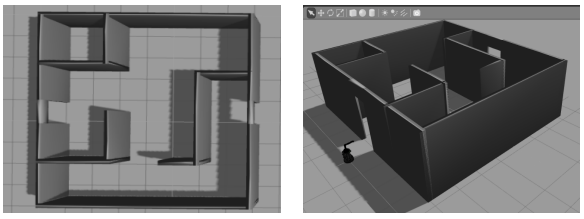

Figura 9: Escenario para mapeado y navegación.

Una vez lanzados los nodos del *stack* de navegación de *ROS*, se genera el mapa de la figura 10 con los datos recopilados por la cámara *kinect*.

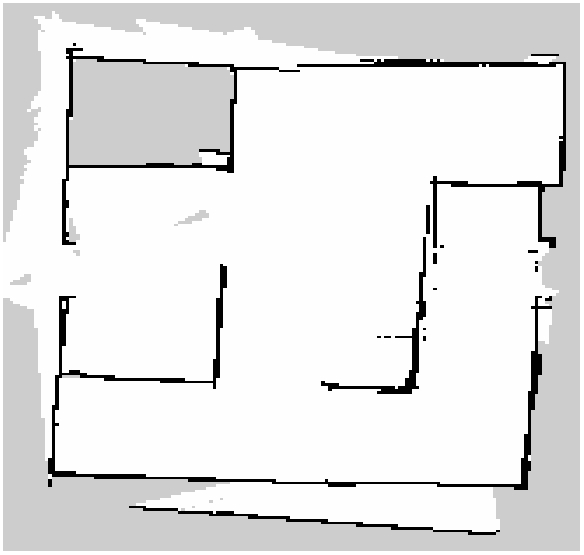

Figura 10: Mapa creado del entorno.

En la figura 11 se observa cómo *CRUMB* es capaz de ir a un punto indicado en el mapa esquivando las paredes del entorno. Lo más relevante respecto al presente trabajo es que dicha navegación se realiza de manera físicamente realista.

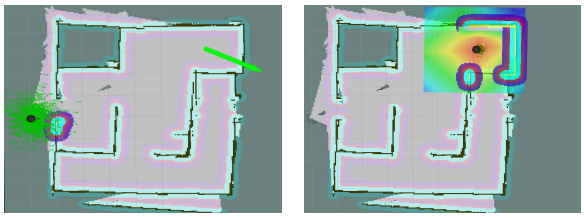

Figura 11: *CRUMB* llegando al punto objetivo.

Si se incluyeran nuevos objetos en el escenario, utilizando el mismo mapa, el robot llega al punto objetivo esquivando con éxito estos nuevos objetos, como se aprecia en la figura 12.

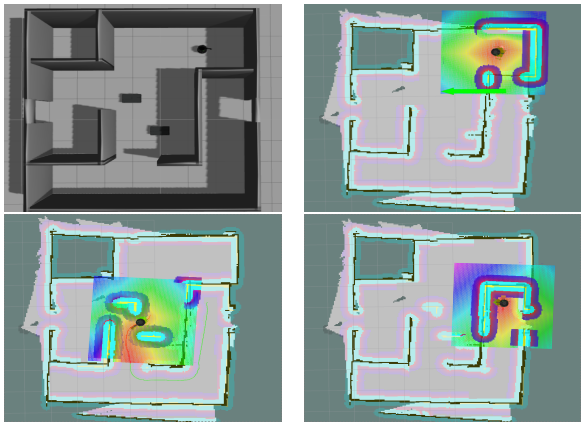

Figura 12: *CRUMB* navega en un escenario con obstáculos.

En la referencia [13] se puede ver la secuencia completa de este experimento.

4.3. EXPERIMENTO 3. MANIPULACIÓN MÓVIL

En este tercer experimento se combina la navegación con la manipulación de objetos. En el escenario de la figura 13 *CRUMB* navega hasta una superficie donde se encuentra un objeto, lo recoge y lo lleva hasta otra superficie para soltarlo.

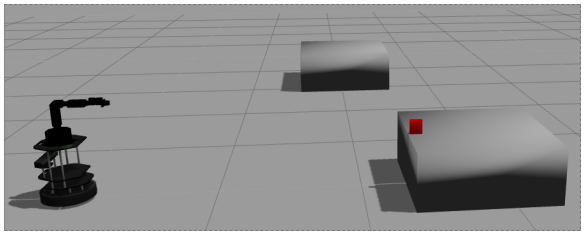

Figura 13: Escenario para la manipulación móvil.

En la figura 14 se ve cómo *CRUMB* es capaz de realizar la tarea. En la referencia [14] se encuentra

el vídeo con la secuencia completa de este experimento, donde se puede apreciar mejor el realismo de la simulación.

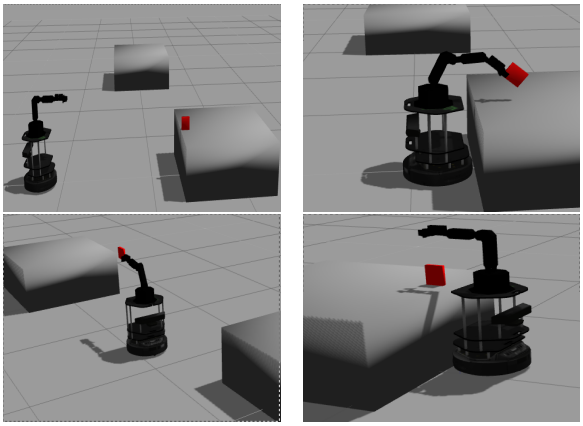

Figura 14: *CRUMB* realizando el experimento 3.

5. CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS

En este trabajo se ha presentado una simulación en *ROS* y *Gazebo* del manipulador móvil de bajo coste *CRUMB*, compuesto por un brazo *Widow-X* y una plataforma *Turtlebot2*. Esta simulación permite ejecutar las mismas aplicaciones que se ejecutarían en el robot real programándolas de modo prácticamente idéntico gracias a que se han añadido todos los *topics* de *ROS* necesarios. Además, al incluir elementos geométricos (visuales y de colisión), parámetros físicos, de control y ruido en los sensores *IMU* y en la cámara *kinect*, se ha conseguido que el modelo tenga un aspecto muy realista.

Este trabajo también ha incluido el modelado cinemático del brazo *Widow-X*, no disponible anteriormente, así como la resolución de la cinemática directa e inversa del mismo.

Para validar la simulación se han llevado a cabo diversos experimentos con éxito: *pick-and-place*, creación de mapas, navegación con obstáculos y navegación combinada con manipulación.

También se han escrito tres manuales: de instalación, uso y desarrollo, y se ha subido todo el código necesario para su uso al popular repositorio de software *GitHub* del proyecto [15], donde está disponible públicamente.

El trabajo futuro incluye de manera natural el uso de esta simulación en docencia en ingeniería, donde se debería evaluar el aprendizaje del alumno y las diferencias encontradas con el robot real; asimismo, este trabajo ha constituido una base importante del proyecto *CRUMB*, donde se pretende en el futuro implementar diversas líneas de investigación relacionadas con la Robótica Cognitiva en la que los integrantes del proyecto trabajan actualmente, como el aprendizaje por refuerzo o la inferencia bayesiana aplicada a sistemas de percepción robóticos.

Agradecimientos

Este trabajo ha sido financiado por el Plan Propio de Investigación de la Universidad de Málaga. Los autores agradecen a la empresa *Robotnik Automation S.L.L* su amabilidad y disposición, así como los ficheros CAD del brazo. También desean agradecer a los compañeros integrantes del proyecto *CRUMB* sus valiosas aportaciones durante la realización de este trabajo.

English summary

REALISTIC SIMULATION OF THE LOW-COST *TURTLEBOT2* + *WIDOW-X* MOBILE MANIPULATOR IN *ROS*

Abstract

The combination of a Turtlebot2 mobile-platform with a Widow-X robot-arm offers an interesting low-cost solution for research, teaching, or generic educational purposes. Unfortunately, neither does this hardware combination have a complete simulator in the robot programming-framework ROS, nor does the manufacturer offer a kinematic model of the robotic arm. In this article we propose a solution for both issues related to our mobile manipulator. Besides, we describe how robot simulation was developed in Gazebo, a realistic simulator integrated with ROS, and the possibility to include simple sensor-noise, which is not always available on most simulators. Finally, we demonstrate the applicability of our simulator by means three experiments that involve pick-and-place, mapping, and navigation tasks.

Keywords: mobile manipulator, robot simulation, kinematic modelling, *Turtlebot2*, *Widow-X*, *ROS*, *Gazebo*.

Referencias

[1] Blog CRUMB. [Online]. Disponible: http://babel.isa.uma.es/crumb/, Accedido: 20/06/2018.

[2] Turtlebot. [Online]. Disponible: http://www.turtlebot.com, Accedido: 20/06/2018.

[3] Widow-X. [Online]. Disponible: http://www.trossenrobotics.com/widowxrobotarm, Accedido: 20/06/2018.

[4] Robot Operating System (ROS). [Online]. Disponible: http://www.ros.org, Accedido: 20/06/2018.

[5] Gazebo. [Online]. Disponible: gazebosim.org/, Accedido: 20/06/2018.

[6] J. J. Craig, *Introduction to robotics: mechanics and control*, vol. 3. Pearson Prentice Hall Upper Saddle River, 2005.

[7] Plugins Gazebo. [Online]. Disponible: http://gazebosim.org/tutorials?tut=ros_gzplugins, Accedido: 09/06/2018.

[8] Robotnik Automation S.L.L. [Online]. Disponible: http://www.robotnik.es/, Accedido: 20/06/2018.

[9] MeshLab [Online]. Disponible: http://www.meshlab.net/ , Accedido: 20/06/2018.

[10] V-REP. [Online]. Disponible: http://www.coppeliarobotics.com/, Accedido: 20/06/2018.

[11] Sensores Kobuki [Online]. Disponible: http://www.robotnik.es/web/wp-content /uploads/2014/04/TB_robot.pdf , Accedido: 20/06/2018.

[12] Vídeo aplicación 1. [Online]. Disponible: http://babel.isa.uma.es/crumb/index.php/2016/12/04/crumb-grasping-objects/, Accedido: 20/06/2018.

[13] Vídeo aplicación 2. [Online]. Disponible: http://babel.isa.uma.es/crumb/index.php/2016/12/04/navigation-with-crumb/, Accedido: 20/06/2018.

[14] Vídeo aplicación 3. [Online]. Disponible: http://babel.isa.uma.es/crumb/index.php/2016/12/04/crumb-moving-objects-around/, Accedido: 20/06/2018.

[15] Repositorio GitHub [Online]. Disponible: https://github.com/ , Accedido: 20/06/2018.