Congreso Internacional en Ingeniería Electrónica. Mem. Electro 2015, Vol. 37, pp 58-63, Chihuahua, Chih. México
http://depi.itchihuahua.edu.mx/display/memorias_electro/MemoriaElectro2015
ISSN 1405-2172

DISEÑO E IMPLEMENTACIÓN DE AMBIENTE VIRTUAL DEL ROBOT MÓVIL ROBULAB10 PARA CONTROL DE POSICIÓN Y MOVIMIENTO DESDE UN DISPOSITIVO XBOX

Valerio-Zárate Emmanuel, Rojas-Estrada Juan-Antonio*, Patricia Quintero-Alvarez, Ernesto-Jesús Rincón-Martínez, René Sanjuan-Galindo
Instituto Tecnológico de Nuevo León
División de Estudios de Posgrado e Investigación
Posgrado en Ingeniería Mecatrónica
Av. Eloy Cavazos 2001
Tel. 81 8157 0500
e-mail: emval9@hotmail.com, juan.antonio.rojas@itnl.edu.mx, patricia.quintero@itnl.edu.mx, ernesto.jesus.rincon@itnl.edu.mx, rene.sanjuan@itnl.edu.mx
*Autor correspondiente

RESUMEN

El presente trabajo contiene el desarrollo, control y manejo del robot móvil Robulab 10 con un mando Xbox 360 en un entorno de simulación virtual. Para la simulación se escogió el Microsoft Robotics Developer Studio 4 (RDS4) utilizando su ambiente de simulación virtual (VSE). En el trabajo se muestra el desarrollo del modelo en código en visual C#. Se presenta la creación de la entidad con sus características reales, añadiendo el manejo diferencial, sensores digitales y análogos y el control del robot con el mando Xbox 360 en tiempo real. Para alcanzar la posición objetivo, se utiliza una ley de control. En la última etapa se realizaron pruebas estableciendo situaciones de operación, puntos de inicio, intermedios y de llegada. El sistema implementado, se puede acoplar a un sistema de gestión de trayectorias de un robot móvil con remolque.

Palabras Clave: Developer Studio 4, Microsoft Robotics, Robot móvil, Simulación de robots

ABSTRACT

The present paper explains the development, control and drive of the mobile robot Robulab 10 with an Xbox 360 controller in a virtual simulation environment. The Microsoft Robotics Developer Studio 4 (RDS4) was chosen using its Virtual Simulation Environment (VSE). The development of the model in Visual C# is presented in this paper. The creation of de entity with its real characteristics, adding the differential drive, analog and digital sensors and the real time control of the robot with the Xbox 360 controller is presented. At the last stage, tests were made adding operation situations, start, arrival, intermediate points, and position control law. The implemented system works with a robot trajectory planner system of a mobile robot with trailer.

Key words: Robot simulation, Microsoft Robotics Developer Studio 4, Mobile robot.

1. INTRODUCCIÓN

Durante los últimos años ha sido de real importancia la simulación de robots virtuales antes de la construcción del prototipo, esto con el fin de evitar fallos en el manejo, control o dentro del diseño, para así tener determinados ajustes necesarios al momento de su construcción. Los simuladores emulan los sensores y la funcionalidad de los actuadores que sirve como evaluación, depuración y ajuste del control. Las interacciones y comportamientos de un robot que se desarrollan en el ambiente virtual son simuladas con mucha precisión. Este sistema nos da la ventaja de evitar la falta de eficiencia de los algoritmos ya que pueden probarse exhaustivamente y a la vez evitar daños en el robot que conduzcan a pérdidas de tiempo si se prueban los algoritmos directamente en el robot.

Además, crear simulaciones hace que estudiantes e investigadores puedan desarrollar aplicaciones o experimentos sin necesidad de tener un robot físico que puede ser costoso o que solo pueda usarse en un espacio o tiempo determinado. También contribuye a tener colaboraciones con otros centros de investigación con el fin de que se puedan realizar proyectos basados en el mismo robot.

En la actualidad existe una gran variedad de robots de diferente clasificación, dependiendo si son fijos o móviles. Dentro de los robots fijos se encuentran los robots manipuladores industriales de los que existes múltiples tipos de brazos robóticos programables para la ejecución de diversas tareas [1]. Y dentro de los robots móviles existe una diversidad que va desde los aéreos y terrestres hasta los sumergibles, para realizar diversas aplicaciones de acuerdo a lo que el usuario necesite. Es por esto que para a realizar simulaciones es necesario un software para el diseño de este tipo de plataformas que sean capaces de simular las físicas de cualquier entorno.

Existe una extensa diversidad de software de simulación de robots, que van desde los comerciales hasta de código abierto. En [2] se hace un resumen y evaluación de los softwares de simulación existentes hasta ese momento, tomando criterios de evaluación como la fidelidad física de la realidad al entorno gráfico, la fidelidad funcional de un equipo real al simulado, la facilidad del desarrollo y los costos para desarrollarlo.

En relación a este trabajo, dentro del departamento de posgrado de Ingeniería Mecatrónica del Instituto Tecnológico de Nuevo León se establecieron diferentes proyectos en relación al Polígono de Velocidades Admisibles (PVA) [3] que establece una planificación de trayectorias libre de colisiones en un robot móvil diferencial. En [4] se realizó un proyecto utilizando el robot Robulab 10 como robot de prueba

del PVA añadiéndole un remolque. En [5] se realiza la reversa con el PVA utilizando el Robulab 10 y un remolque acoplado. Para continuar con la implementación del PVA en este proyecto se tiene el objetivo de realizar una simulación 3D del Robulab 10 de manera que acepte comandos del mando Xbox 360 en tiempo real y se establezca un sistema de control de posición y de trayectorias de un punto a otro, para que sirva como base para implementar el PVA en la simulación. Además que pueda utilizarse como una forma más de control en un sistema de gestión de trajectorias de un conjunto de robot móvil con remolque.

Se decidió utilizar el RDS4 como herramienta de simulación ya que es un software gratuito que no se necesita de una cuota o un pago por su utilización además de la facilidad de poder ser programado dentro del Visual C# del Visual Studio. También RDS4 utiliza NVIDIA™ PhysX™ Technology que recrea un ambiente realista y permite construir un robot desde cero contando con avanzadas herramientas en 3D, además de ser amigable con los usuarios [6].
El proyecto se encuentra en desarrollo, se espera en un futuro otros dispositivo sensores para poder crear situaciones de operación y trayectorias.

2. CARACTERÍSTICAS DEL MICROSOFT ROBOTICS DEVELOPER STUDIO 4

El RDS4 es un kit de desarrollo de software creado sobre el entorno .NET y permite crear software para robots existentes en el mercado, robots personalizados o que están por construirse. Es una herramienta gratuita para uso personal, académico y de desarrollo [7]. Se basa en un entorno de desarrollo escalable con concurrencia de procesos en la cual se puede diseñar, programar, simular e implementar aplicaciones para diferentes plataformas de robots. El modelo de arquitectura se basa en la estructura cliente-servidor. Además, RDS4 es programado en lenguaje de programación C# que puede ser ejecutado dentro del ambiente de desarrollo de Visual Studio Express que es completamente gratuito [8]. Una de las ventajas que posee es que se puede utilizar tanto para un entorno simulado como para uno real con el robot conectado a la pc o laptop por el puerto Ethernet o Wifi. Una muestra de este software se puede ver en la Figura 1.

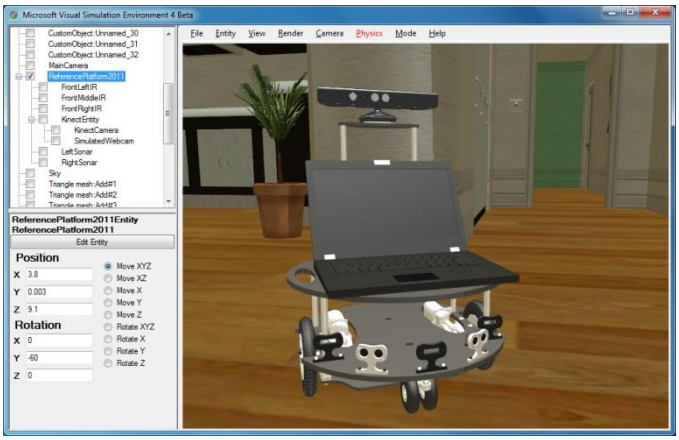

Figura 1. Ambiente de simulación de RDS4.

Hay que resaltar que RDS4 se divide en 4 componentes principales. La Concurrencia y Coordinación en tiempo de ejecución (CCR) y los Servicios de Software Descentralizados (DSS) comprenden el ambiente en tiempo de ejecución. El ambiente de simulación visual (VSE) es un simulador 3D con un simulador de físicas completo donde se puede crear algoritmos y objetos. El lenguaje de programación visual (VPL) es un ambiente de programación gráfica en donde se pueden implementar los servicios.

2.1 Concurrencia y Coordinación en tiempo de ejecución (CCR)

El CCR es una librería gestionada que provee clases y métodos para ayudar a la concurrencia, coordinación y la falla de manejo. También hace posible escribir segmentos de código que operan independientemente y de forma paralela. Los segmentos de código se comunican entre ellos por medio de mensajes cuando es necesario. El CCR es un componente DLL (Dynamic Linking Library), el cual es una librería .NET [7]. La concurrencia es esencial porque al tiempo en que se maneja un robot, también se necesita estar monitoreando los sensores. Sin el procesamiento asíncrono no se podría manejar un robot completo, solo una pieza a la vez.
La estructura del CCR se aprecia en la Figura 2.

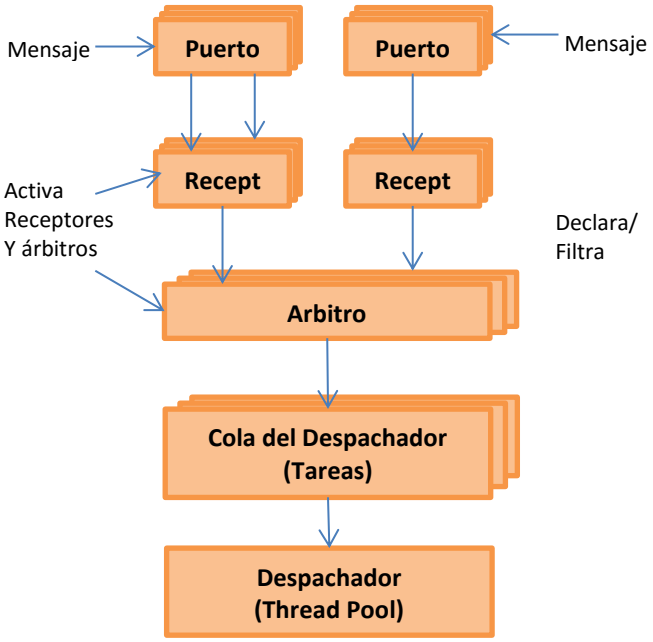

Figura 2. Estructura del CCR.

2.2 Servicios de Software Descentralizados (DSS)

El DSS es el responsable de controlar las funciones básicas de las aplicaciones robóticas. El DSS está construido por encima del CCR. Es el responsable de iniciar y detener los servicios y administrar el flujo de mensajes entre servicios. De hecho, está compuesto por varios servicios que cargan configuraciones de servicios, administra la seguridad, mantienen el directorio de servicios ejecutándose, controla el acceso a archivos locales y recursos embebidos y provee interfaces de usuario a través de páginas web accesibles mediante un navegador web [10].

El DSS utiliza un protocolo llamado Protocolo DSS (DSSP). El DSSP está basado en la arquitectura REST (Representational State Transfer) de servicios web que permite que dispositivos se puedan comunicar en un entorno distribuido.

2.3 Entorno de Simulación Virtual (VSE)

El VSE está diseñado para utilizarse en una variedad de escenarios con alta demanda de fidelidad, visualización y escalamiento. Esto, ofrece a los programadores una alternativa de acceso al mundo de la robótica sin tener la necesidad de adquirir ningún tipo de Hardware. La integración del NVIDIA™ PhysX™ Technologies establece una muy fuerte simulación de físicas. El motor de reproducción está basado en el Microsoft XNA Framework. El RDS4 tiene incluidos algunos modelos de robots como son LEGO NXT, iRobot Create, Pioneer 3DX o el brazo robótico KUKA LBR3 [6]. La simulación del robot LEGO se muestra en la Figura 3.

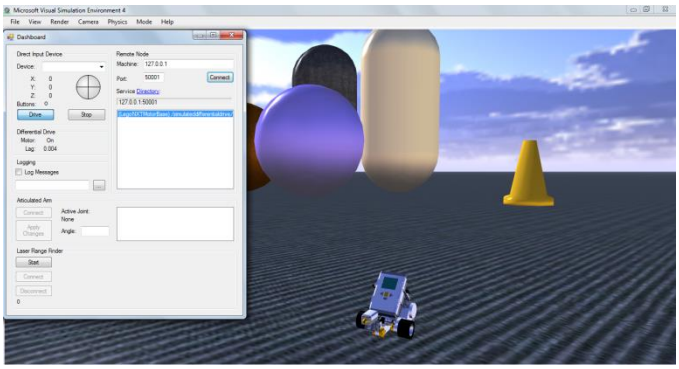

Figura 3. Simulación del LEGO NXT Brick.

2.4 Lenguaje de Programación Virtual (VPL)

RDS4 contiene un entorno de desarrollo de aplicaciones diseñada en un modelo de programación basado en el flujo de datos gráfico llamado VPL. Está enfocado en programadores novatos con conocimientos básicos de variables y lógica. Pero no está limitado a principiantes. Este entorno consiste en arrastrar y colocar una secuencia de conexiones de bloques con entradas y salidas que pueden conectarse a otros bloques, que representan actividades o servicios, dentro de un área de diseño, véase la Figura 4.

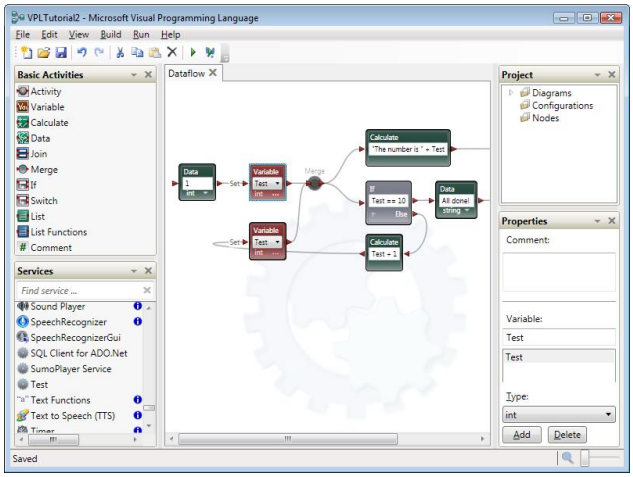

Figura 4. Diagrama en VPL.

3. DESARROLLO

Nuestro trabajo se enfoca en el desarrollo de una simulación del robot móvil Robulab 10 para que sirva de base para implementar la simulación del PVA en un robot móvil Robulab10.

Se desarrolló un nuevo servicio DSS utilizando una plantilla

```
SkyDomeEntity sky = new
SkyDomeEntity("skydome.dds", "sky_diff.dds");
SimulationEngine.GlobalInstancePort.Insert(sky);

LightSourceEntity sun = new LightSourceEntity();
sun.State.Name = "Sun";
sun.Type = LightSourceEntityType.Directional;
sun.Color = new Vector4(0.8f, 0.8f, 0.8f, 1);
sun.Direction = new Vector3(0.5f, -.75f, 0.5f);
SimulationEngine.GlobalInstancePort.Insert(sun);
```

en Visual Studio proporcionada en la instalación del RDS4. En este servicio se insertaron elementos del ambiente (en código C#) como el terreno, el cielo y la cantidad de luz. Un ejemplo de parte del código para agregar estos elementos se muestra en la Figura 5.

Figura 5. Código para insertar elementos a la simulación.

Para insertar la entidad del robot móvil se diseñó una entidad en donde prácticamente se dibujó el robot con las características del robot Robulab 10. El robot se creó así para insertarle dos ruedas locas en vez de una, ya que las entidades creadas a partir de la entidad diferencial solo pueden tener una. En la Figura 6 se muestra parte del código para la creación de la entidad visual y en la Figura 7 se muestra el resultado en la simulación.

```
BoxShapeProperties chassisDesc = new
BoxShapeProperties(
                    "chassis",
                    mass,
                    new Pose(new Vector3(0,
chassisClearance + chassisDimensions.Y /
2.0f, 0)),
                    chassisDimensions);

                chassisDesc.Material = new
MaterialProperties("chassisMaterial", 0.0f,
0.5f, 0.5f);

                BoxShape chassis = new
BoxShape(chassisDesc);
                chassis.State.Name =
"ChassisShape";

base.State.PhysicsPrimitives.Add(chassis);
```

Figura 6. Fragmento del código para crear el chasis del robot.

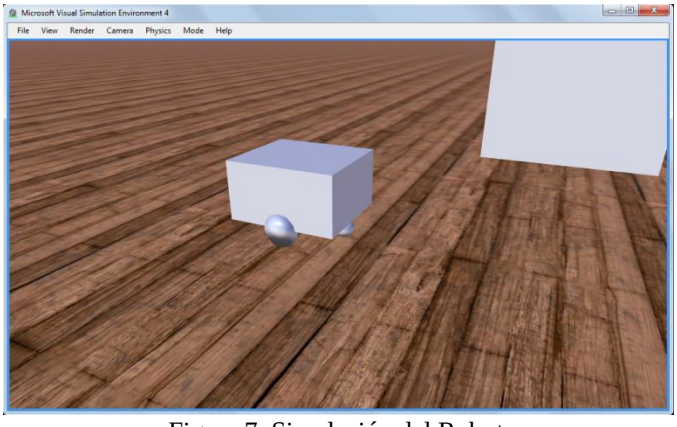

Figura 7. Simulación del Robot.

El robot tomó forma de robot móvil diferencial y para darle movimiento fue necesario crear un socio para el manejo diferencial. Se ajustaron parámetros de acuerdo a las características reales del robot. Una vez configurado, se añadió el servicio del simple Dashboard dentro del archivo .xml o manfiest del servicio DSS del robot. Este simple dashboard tiene un joystick virtual para mover al robot o también puede ser controlado por un control de Xbox 360. En la Figura 8 se muestra el Joystick virtual en el simple dashboard.

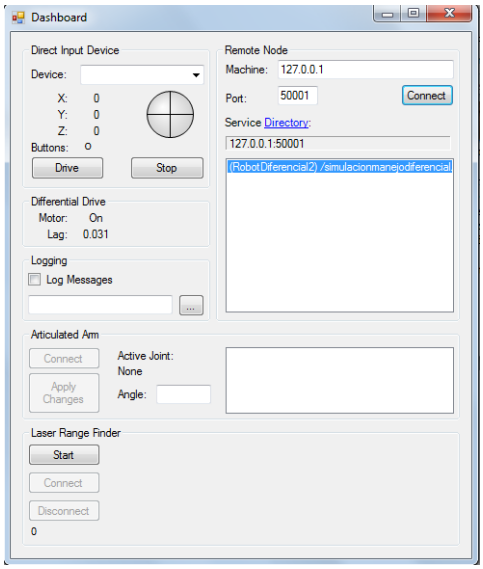

Figura 8. Se aprecia el joystick virtual en la parte superior izquierda.

3.1 Diseño y elaboración del modelo 3D del Robulab 10 e integración a la simulación

Después de realizar las mediciones del robot Robulab 10, se diseñó un modelo 3D del mismo. El programa seleccionado para la elaboración del diseño fue *Solidworks*. El robot fue creado a base de ensambles para crear el modelo final. En la Figura 9 se muestra el diseño en Solidworks del robot y en la Figura 10 el robot real.

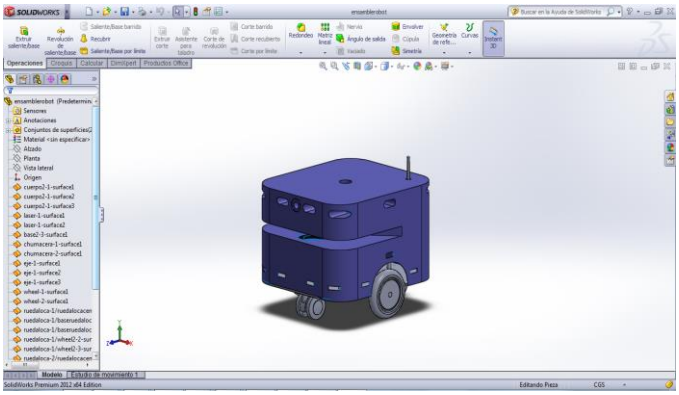

Figura 9. Modelo 3D del Robulab 10 en Solidworks.

Figura 10. Robot móvil Robulab 10.

Como el RDS4 solo acepta modelos en formato .obj, se tuvo que realizar una conversión de formatos. Primero, desde Solidworks se guardó el modelo en formato .stl. Una vez obtenido el modelo en .stl se utilizó el software Blender para convertir el formato de .stl a .obj. Debido que al pasar el modelo en formato .stl, este solo guarda las mallas en 3D de la figura pero no trae consigo las propiedades de color o textura. Por esto se le añadió el color utilizando Blender como se muestra en la Figura 11.

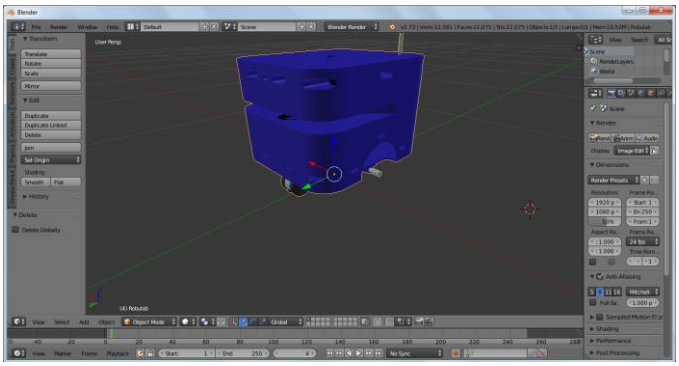

Figura 11. Edición de color del modelo del Robulab 10 en Blender.

Una vez editado se guardaron los modelos .obj en la carpeta de media dentro de la instalación del RDS4 para poder ser utilizados.

3.2 Simulación del Diseño completo

Para controlar el robot se utilizó un control de Xbox 360. Se realizó una prueba en el manejo diferencial moviéndolo a través del terreno creado y haciéndolo colisionar ante obstáculos. En la Figura 12 se ve el resultado de la simulación.

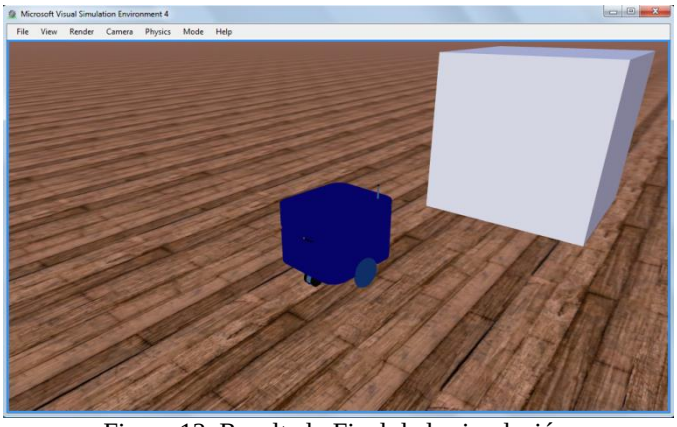

Figura 12. Resultado Final de la simulación.

4. CONCLUSIONES

Los resultados de la simulación fueron exitosos. La creación de una entidad tipo diferencial en la simulación se pudo controlar utilizando un mando Xbox 360. A demás se logró realizar un modelo en 3D e insertarlo dentro del ambiente de simulación. Se hicieron otros ensayos creando ambientes con otros obstáculos. Se puede destacar que los detalles del ambiente son muy cercanos a la realidad, característica que se buscaba desde el planteamiento del proyecto. Se ha ganado mucha experiencia en el diseño de otros objetos y su encadenamiento al ambiente de simulación virtual básico.

5. TRABAJOS FUTUROS

En la etapa actual del desarrollo del proyecto, el robot simulado aún no cuenta con los sensores análogos y digitales instalados. Se desea crear las entidades de los sensores así como sus respectivos servicios socios para ser implementados dentro de la simulación. Una vez estén implementados, se podrán crear situaciones para control de posición y trayectorias utilizando las lecturas de los sensores. Esta situación también será de suma importancia para la incorporación del ambiente virtual creado al sistema de gestión de trayectorias de un conjunto de robot móvil con remolque que actualmente está en curso y que utiliza el Robulab10 y un remolque instrumentado.

6. REFERENCIAS

[1] B. A. Ollero. Robótica: manipuladores y robots móviles. Marcombo S.A. 2001.

[2] Jeff Craighead, Robin Murphy, Jenny Burke, and Brian Goldiez. “A Survey of Commercial & Open Source Unmanned Vehicle Simulators”, in Robotics and Automation, 2007 IEEE International Conference on, pp 852 – 857, Roma, Italia. 10-14 April 2007.

[3] P. Quintero Álvarez, J. A. Rojas Estrada, A. A. Fernández Ramírez y J. G. Ramírez Torres. “Técnicas para evasión de obstáculos en Robótica Móvil”, en CISCI 2010. Orlando florida. Julio, 2010.

[4] J. A. Rojas Estrada, P. Quintero Álvarez, L.A. Montoya Soto, H.A. Villarreal Hdez. “Implementación del pva para la planificación de trayectorias con obstáculos de un robot móvil en un módulo de microcontrolador”, en Electro 2012.

[5] Hibram A. VILLARREAL-HERNÁNDEZ, Juan A. ROJAS-ESTRADA, Patricia QUINTERO-ALVAREZ, René SANJUAN-GALINDO. “Implementación de un tractor virtual para un robot móvil con remolque para ejecución de reversa”, en Revista Iberoamericana de Sistemas, Cibernética e Informática: RISCI.Diciembre, 2014.

[6] Microsoft, “Microsoft Robotics Developer Studio 4 (RDS4)”, Microsoft [En linea]. Disponible en: http://msdn.microsoft.com/en-us/library, sitio visitado marzo 2015

[7] Kyle Johns, Trevor Taylor. Professional Microsoft Robotics Developer Studio, Wrox, 2008.

[8] “Visual Studio Express”. [En línea]. Disponible en: https://www.visualstudio.com/es-es/products/visual-studio-express-vs.aspx, sitio visitado mayo 2015