Videojuego para la simulación de robots seguidores de línea en entornos tridimensionales

Oscar Fernando Gómez Sandoval, Urbano Eliécer Gómez Prada,
Universidad Pontificia Bolivariana, Bucaramanga

***Resumen*— Este artículo presenta una vista general de un videojuego en desarrollo que permite el diseño físico y lógico de un robot seguidor de línea y la observación del comportamiento de dicho robot en un entorno tridimensional animado. El juego busca brindar un marco de trabajo común para el desarrollo de habilidades de programación y la creación de algoritmos de seguimiento de alta calidad que puedan compararse de manera objetiva y precisa, además de servir como herramienta de apoyo durante la creación de un robot físico.**

***Palabras clave*—Videojuegos, Simulación, Robótica, Seguidores de línea.**

I. INTRODUCCIÓN Y CONTEXTO

UN seguidor de línea es un robot que tiene la capacidad de desplazarse sobre una trayectoria dada mediante el seguimiento completamente autónomo de una ruta indicada visualmente , generalmente en forma de una línea negra sobre un fondo blanco o viceversa (Ilustración 1).

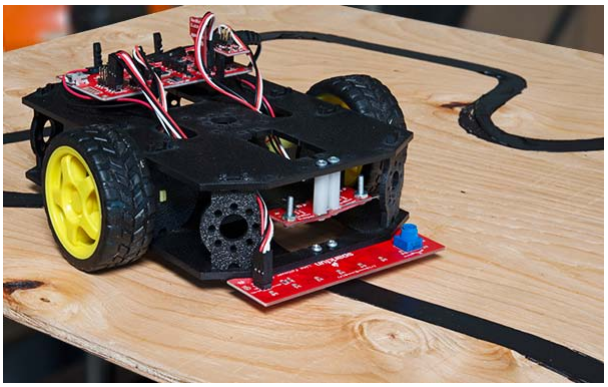

Ilustración 1 Seguidor de línea efectuando un recorrido. Foto original por SparkFun Electronics, licencia Creative Commons BY. Edición por el autor O. Gómez.

El funcionamiento de un seguidor de línea se soporta en la implementación de un controlador que incorpora un algoritmo definido por el diseñador del robot. Dicho controlador –cuya complejidad puede variar entre un circuito analógico sencillo hasta un programa ejecutándose en una tarjeta Arduino o un computador– debe basarse en las lecturas de uno o más sensores para determinar si la trayectoria del robot es adecuada o no y, en consecuencia, ajustar la velocidad de rotación de los motores que están acoplados a las ruedas del robot para mantenerlo sobre la línea que se está siguiendo. Por ejemplo, en el instante mostrado en la Ilustración 2, uno de los dos sensores del robot (verde) capta la línea, mientras que el otro (rojo) no lo hace. El controlador del robot podría usar esa información para concluir que el robot no debería ir en línea recta porque se saldría de la línea por la derecha, y corregiría el rumbo haciendo que el motor derecho gire más rápido que el izquierdo.

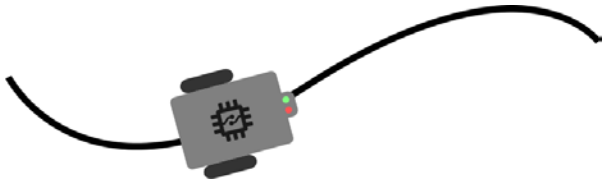

Ilustración 2 Funcionamiento de un seguidor de línea de dos sensores.

Los seguidores de línea suelen utilizarse en ámbitos académicos como una forma interesante de enseñar nociones de robótica, sistemas de control o inteligencia artificial. Así, por ejemplo, la realización de un seguidor de línea puede incorporar el aprendizaje de programación de microcontroladores, control de velocidad por modulación de ancho de pulso (PWM), aplicación de lógica difusa, algoritmos genéticos o control PID, entre otras temáticas [1][2][3].

Ahora bien, el desarrollo de un robot requiere del adecuado manejo de diversos factores mecánicos, ambientales y económicos que pueden influir seriamente en su desempeño. El material del robot, la calidad de los motores y sensores, el nivel de carga de las baterías e incluso la luz medioambiental pueden marcar la diferencia entre un robot operativo y uno no funcional. Dependiendo de las habilidades que quieran desarrollarse, puede ser deseable el tener que manejar estos factores. Sin embargo, si lo que se desea es enfocarse en mejorar habilidades de algoritmia y control, lo ideal sería removerlos por completo.

Actualmente, los autores se encuentran finalizando el desarrollo de un videojuego consistente en un simulador de seguidores de línea que remueve una gran cantidad de factores físicos difíciles de controlar y se enfoca en que los jugadores diseñen los algoritmos de sus robots para que optimicen el seguimiento lo más posible. El juego busca brindar un marco

Artículo presentado el 6 de julio de 2016.
Este trabajo fue desarrollado en el marco del proyecto de investigación "Desarrollo de un software simulador de robots seguidores de línea", financiado por la Universidad Pontificia Bolivariana Seccional Bucaramanga.
O. Gómez es profesor de la Universidad Pontificia Bolivariana Seccional Bucaramanga (e-mail: oscar.gomezs@upb.edu.co).
U. Gómez es profesor de la Universidad Pontificia Bolivariana Seccional Bucaramanga (e-mail: urbano.gomez@upb.edu.co).

de trabajo común para el desarrollo de habilidades de programación y la creación de algoritmos de seguimiento de alta calidad que puedan compararse de manera objetiva y precisa, y que incluso pueda servir como herramienta de apoyo durante la creación de un robot físico.

A la fecha, el videojuego es completamente funcional. Para usarlo, el jugador inicialmente debe crear el controlador de su robot en el lenguaje de programación LUA –un lenguaje de *scripting* ampliamente utilizado en la industria de videojuegos [4]– y un diseño básico de la estructura física del robot consistente en indicar la distancia entre sus ruedas y la posición de sus sensores. Una vez listo el robot, éste puede cargarse en el juego, que simula su comportamiento en una pista (que también puede ser diseñada por los jugadores como se explica más adelante). La visualización de la simulación puede ser básica –mostrando la pista tal cual es: una línea negra sobre fondo blanco– o inmersiva –representando a los robots en una carrera de alta velocidad de hasta tres participantes simultáneos en un escenario tridimensional– (véanse la Ilustración 3 y la sección Interfaz gráfica).

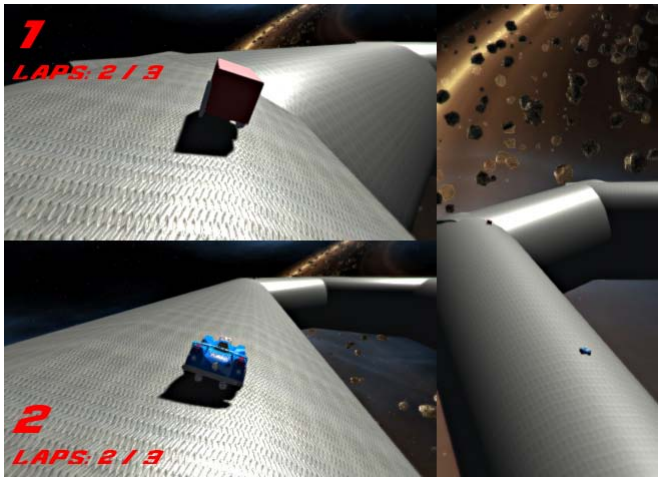

Ilustración 3 Carrera de dos jugadores ejecutándose en una versión inicial del escenario *Warp*, incluido con el juego.

Para lograr el resultado descrito, el desarrollo del software se realizó en dos etapas. En la primera, se definió la manera como se simularía el comportamiento de los robots, y en la segunda, se estableció la visualización de dichas simulaciones. Este artículo presenta una revisión general de las consideraciones que se tuvieron en cuenta durante dichas etapas, así como algunas conclusiones de diseño a las cuales se llegó al momento de implementar el videojuego.

II. SIMULACIÓN FÍSICA

Teniendo en cuenta los objetivos del proyecto, se minimizaron en la simulación los factores físicos que afectaran el movimiento del robot. Así, se estableció que las ruedas de los robots siempre rodarían sin deslizar y que el peso del robot se ignoraría, de modo que no se tendrían en cuenta los efectos de la inercia ni la inercia rotacional en su movimiento. También se establecieron restricciones al diseño físico del robot: aunque un robot podría tener una cantidad ilimitada de variantes al momento de su construcción, la opinión de los autores es que sólo ciertos factores clave aportan significativamente al desarrollo de algoritmos de seguimiento interesantes, y sólo dichos factores deberían ser tenidos en cuenta en el modelado, mientras que los demás deberían simplemente restringirse (véase la sección Discusión). A manera de ejemplo, considérese el robot Dynabot II mostrado en la Ilustración 4. Dicho robot dispone de dos ruedas alineadas en un mismo eje, dos sensores infrarrojos y una rueda de apoyo omnidireccional,[1] todo acoplado a una base de acrílico. Factores como la posición de la rueda de apoyo y el tamaño o la ubicación de las ruedas son poco relevantes al momento de diseñar y simular, dado que se puede asumir que los jugadores nunca los establecerían de maneras que dificultaran el funcionamiento del robot.

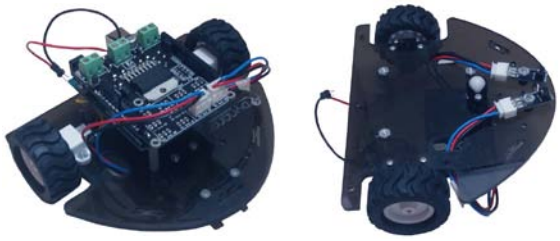

Ilustración 4 Robot Dynabot II. Izquierda: vista superior. Derecha: vista inferior.

Con base en todo lo anterior, y teniendo en cuenta que dos ruedas son suficientes para controlar el movimiento del robot por completo a la vez que se minimiza el peso de éste, y que no es común encontrar robots que usen más ruedas, se definió que los robots simulados tendrían dos ruedas de igual tamaño acopladas a un mismo eje, igual que el Dynabot II. La longitud del eje, esto es, la distancia entre las ruedas, sí sería definible por el jugador al diseñar su robot, ya que afecta directamente la manera como el robot gira y no tiene un valor óptimo por sí solo, de manera que los jugadores pueden variarla para lograr comportamientos distintos y dar un mejor control a su algoritmo. Con los sensores, por otro lado, hubo una cantidad mayor de flexibilidad: los jugadores pueden diseñar robots que hagan uso de cualquier cantidad de sensores (que, por supuesto, puede limitarse como parte de las reglas de una competición o torneo que se lleve a cabo usando el videojuego), que pueden ubicarse en cualquier posición con respecto a las ruedas. La decisión de permitir que los jugadores ubiquen los sensores libremente se debió al hecho de que el posicionamiento y la cantidad de sensores de un robot son importantes para maximizar el efecto de un buen algoritmo, de modo que un diseño flexible en ese sentido seguramente permitirá la creación de resultados interesantes, e incluso el estudio de algoritmos avanzados de seguimiento. En cuanto al tipo de sensores, el juego sólo permite el uso de sensores que captan el color del suelo que esté justo debajo de ellos.

[1] La función de esta rueda es actuar como punto de apoyo para mantener al robot estable y permitir su libre desplazamiento y rotación.

Finalmente, y con el fin de que los robots simulados tuviesen un comportamiento cercano a la realidad, varias características técnicas del robot (velocidad de giro máxima y aceleración angular de los motores, radio de las ruedas, etc.) se definieron a partir de una completa caracterización que se hizo al Dynabot II por medio de mediciones físicas directas y cálculos realizados a partir de procesamiento cuadro a cuadro de videos del robot funcionando [5].

Una vez definidas las restricciones anteriores, se procedió a desarrollar el motor de física que se encargaría de enmarcar el comportamiento de los robots de los jugadores en un entorno realista desde un punto de vista físico. Dicho motor de física, en breve, consiste en un modelo de simulación compuesto por ecuaciones diferenciales que describen el comportamiento físico de un cuerpo móvil, y que son integradas de manera repetitiva en pasos discretos que corresponden a pequeños cambios de tiempo [6], lo cual es equivalente a una simulación numérica. Para la creación del motor, inicialmente se desarrolló un modelo matemático exacto de un seguidor de línea con la restricción de que las ruedas del robot siempre girarían a velocidades angulares constantes. Posteriormente, se desarrolló una simulación numérica bajo la misma restricción, y se implementaron dos métodos para efectuar las integraciones de las ecuaciones del modelo de simulación: el método de Euler y el método de Runge Kutta de cuarto orden (RK4).[2] Los dos métodos se compararon con el modelo matemático exacto en una herramienta desarrollada por los autores específicamente para ello, verificándose que el método RK4 brindaba niveles superiores de precisión con menores requerimientos de cómputo, lo cual era esperado según la teoría [7]. Finalmente, se extendió el modelo de simulación para que tuviera en cuenta aceleraciones angulares en las ruedas.

Por supuesto, para simular un seguidor de línea, el simulador debe contar con la pista sobre la cual está funcionando el robot. Dicha pista se define por medio de un archivo de imagen y un archivo de texto (Ilustración 5); el primero define la forma de la pista, mientras que el segundo contiene algunos parámetros de ésta: la escala, que indica a cuántos centímetros reales equivale cada píxel de la imagen de la pista, la posición del punto de inicio de la pista y el ángulo de orientación inicial de los robots al comenzar el recorrido. El simulador se basa en esa información para obtener, a partir de los píxeles de la imagen, los valores leídos por los sensores de un robot en cualquier instante de tiempo.

```
cmperpixel=0.85
initialpoint=238,566
initialangledeg=98
```

Ilustración 5 Definición de una pista.

III. INTERFAZ GRÁFICA

Tras desarrollarse la simulación por completo, se creó la capa de interfaz gráfica del juego, para lo cual se generaron escenarios tridimensionales y se adquirieron modelos de vehículos de la Unity Asset Store. Los escenarios tridimensionales se modelaron con base en algunas pistas de ejemplo que se incluyen con el videojuego, y permite simular dichas pistas en un entorno inmersivo; por ejemplo, el escenario *"Landscapes"*, mostrado en la Ilustración 6, es una visualización de la pista de la Ilustración 5.

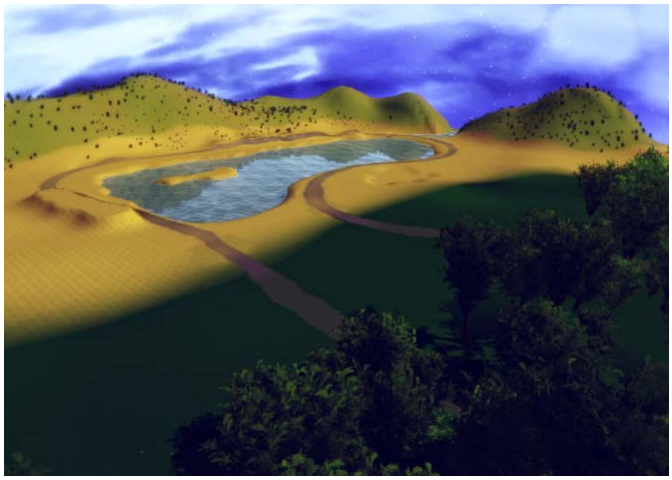

Ilustración 6 Versión inicial del escenario *Landscapes*, incluido en el juego.

Con el fin de hacer más interesantes las competencias en el videojuego, los entornos inmersivos se presentan en una escala considerablemente mayor a la de los robots, de manera que el movimiento de estos se efectúa con una sensación aumentada de velocidad; dicha sensación es únicamente visual, y no afecta en absoluto la ejecución de la simulación. Además, es posible ejecutar carreras con hasta tres competidores simultáneos; la pantalla se divide para presentar la carrera desde el punto de vista de cada competidor, y la interfaz de usuario presenta información con el estado de cada participante en la carrera.

Las carreras también pueden ejecutarse sin usar escenarios inmersivos. En este modo, que se usa por defecto para todas las pistas añadidas por los jugadores, la pista se presenta como es en realidad, y ésta y los robots están a la misma escala, de manera que la visualización es fiel a la realidad. La Ilustración 7 muestra un ejemplo de este tipo de visualización, usando

[2] Más precisamente, se usaron el método de Euler y la regla de Simpson. RK4 y el método de Euler permiten aproximar soluciones de ecuaciones diferenciales de la forma $\dot{y} = f(x, y)$, siendo $\dot{y} \equiv \partial y/\partial x$. RK4 se reduce a la regla de Simpson si la ecuación que se quiere resolver es de la forma $\dot{y} = f(t)$, esto es, f es independiente de y, tal como sucede en este caso, dado que la aceleración es independiente de la velocidad y la velocidad es independiente de la posición [7].

como base la pista utilizada en la competencia RobotChallenge 2015, llevada a cabo en Viena, Austria.

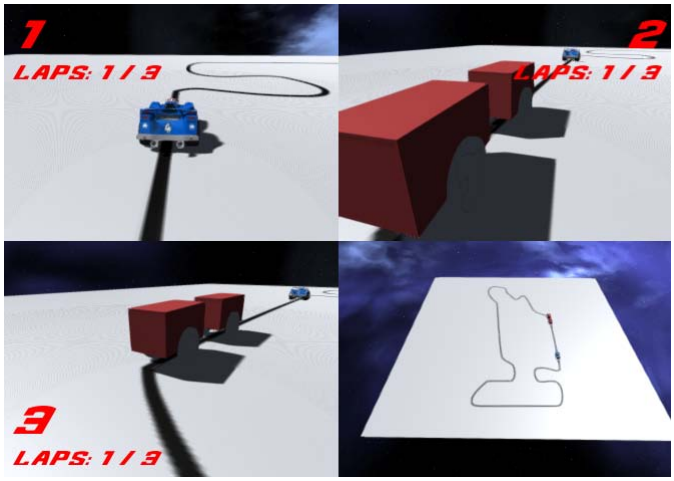

Ilustración 7 Carrera desarrollándose sin un escenario inmersivo.

Por otro lado, la mayoría de los modelos 3D de los vehículos que representan a los robots simulados provienen de la tienda de recursos de Unity [8]. Dichos modelos fueron escalados y animados de manera que tuviesen un comportamiento visual acorde con el resto del entorno y con el estado del robot simulado.

IV. DISCUSIÓN

El videojuego que se está desarrollando está compuesto por dos capas de software principales: la simulación (*back end*) y la visualización (*front end*). Ambas capas se desarrollaron de forma que estuviesen completamente desacopladas, de modo que es posible, por ejemplo, ejecutar simulaciones sin visualización alguna, lo cual puede ser muy útil para ejecutar carreras completas en un instante con el fin de estudiar los resultados de inmediato.

Por supuesto, existen muchas posibles mejoras que pueden incorporarse al juego. En opinión de los autores, una de las más importantes es la de incorporar a la simulación los efectos de la inercia y la inercia rotacional de los robots, que si bien no son factores fundamentales para la creación de buenos algoritmos de seguimiento, sí afectan el realismo físico de la simulación dado que su omisión genera que el comportamiento simulado se aleje del comportamiento de un robot real que ejecute un algoritmo equivalente.

Finalmente: varias de las decisiones tomadas por los autores al momento de diseñar el simulador no sólo afectan la calidad de la simulación, sino que también definieron (o removieron) mecánicas del videojuego final. Aunque todas las decisiones de diseño se tomaron intentando prever las dinámicas que surgirían una vez las personas utilizaran el juego, la única forma cierta de comprobar si dichas decisiones fueron acertadas o no es mediante la futura realización de pruebas (*playtesting*) [9], que son fundamentales cuando se quiere realizar juegos de calidad [10][11].

REFERENCIAS

[1] C. Restrepo, J. Molina y C. Torres, «Algoritmo genético para la ubicación óptima de sensores en un robot seguidor de línea,» *Scientia et Technica*, nº 41, 2009.

[2] J. C. Diaz, «On the Straight and Narrow,» The University of Tulsa, 9 Marzo 2004. [En línea]. Disponible: http://www.ens.utulsa.edu/~diaz/cs2503/text/LineFollower/LineFollower/node1.html.

[3] A. Sitoula, «PID Tutorials for Line Following,» Let's Make Robots, 2014. [En línea]. Disponible: http://letsmakerobots.com/node/39972.

[4] N. Llopis, «Lua - 14th Annual Front Line Awards,» *Game Developer Magazine*, vol. 19, nº 1, Enero 2012.

[5] J. D. Miranda y C. A. Gamboa, «Caracterización del Sistema Robótico Seguidor de Línea Avanzado con plataforma Arduino: Dynabot II,» *Sin publicar*, 2016.

[6] I. Millington, Game Physics Engine Development, San Francisco, California: Morgan Kaufmann Publishers, 2007.

[7] E. Süli y D. Mayers, An Introduction to Numerical Analysis, Nueva York: Cambridge University Press, 2003.

[8] Unity Technologies, «Unity Asset Store,» [En línea]. Disponible: https://www.assetstore.unity3d.com. [Último acceso: 6 julio 2016].

[9] R. Hunicke, M. Leblanc y R. Zubek, «MDA: A formal approach to game design and game research,» de *Proceedings of the Challenges in Games AI Workshop, Nineteenth National Conference of Artificial Intelligence*, Menlo Park, 2004.

[10] T. Fullerton, Game Design Workshop: A Playcentric Approach to Creating Innovative Games, segunda ed., Burlington: Morgan Kaufmann, 2008.

[11] J. Schell, The Art of Game Design: A Book of Lenses, segunda ed., Boca Ratón, Florida: CRC Press, 2014.

Oscar Fernando Gómez Sandoval es Magíster en Ingeniería de la Universidad de los Andes, Bogotá, e Ingeniero Electrónico de la Universidad Pontificia Bolivariana Seccional Bucaramanga. Sus intereses investigativos giran principalmente alrededor del desarrollo de videojuegos con aplicaciones adicionales al entretenimiento. Es profesor adscrito a la Facultad de Ingeniería de Sistemas e Informática de la Universidad Pontificia Bolivariana Seccional Bucaramanga, y coordinador del Semillero de Investigación en Ingeniería de Sistemas e Informática (SIINFO). Es coordinador general del grupo Wikimedistas de Colombia, administrador de Wikipedia en español y miembro activo del movimiento Wikimedia. Es miembro de la IEEE Computer Society.

Urbano Eliécer Gómez Prada es Magíster en Ingeniería, Área Informática y Ciencias de la Computación e Ingeniero de Sistemas de la Universidad Industrial de Santander. Su línea de investigación es la de Ingeniería de Software y Bases de Datos. Es profesor adscrito a la Facultad de Ingeniería de Sistemas e Informática de la Universidad Pontificia Bolivariana Seccional Bucaramanga, líder de la línea estratégica de investigación en Tecnologías de la Información y las Comunicaciones (TIC) de la misma universidad y director del Grupo de Investigación en Informática (GIINFO). Es miembro de la Association for Computing Machinery (ACM).