5.

Uso de simuladores en Robótica Educativa

Gonzalo Zabala†, Ricardo Morán♠

Centro de Altos Estudios en Tecnología Informática

Facultad de Tecnología Informática Universidad Abierta Interamericana

gonzalo.zabala@uai.edu.ar;

ricardo.moran@uai.edu.ar

Resumen

Durante el siglo XXI, la robótica ha tomado un rol significativo como recurso didáctico para la enseñanza de las ciencias y el pensamiento computacional. Sin embargo, la pandemia dispuso un límite físico que obligó a encontrar recursos que permitieran acceder a contenidos similares dentro de ámbitos virtuales. En este artículo se presenta el análisis realizado para la selección de un simulador de robótica basado en un motor de física, y un desarrollo sobre el mismo para la creación de un ambiente sencillo y de características similares al mundo físico, que permitiera mantener la actividad de forma remota y con recursos de hardware limitados.

Palabras clave: educación, robótica, simulación, pensamiento computacional, programación visual

Introducción

La incorporación de robots en las aulas proporciona a los estudiantes un entorno de aprendizaje reflexivo y práctico. Al colaborar en equipos para diseñar, construir, programar y documentar robots autónomos, no solo adquieren conocimientos tecnológicos, sino que también desarrollan habilidades esenciales necesarias para el mundo laboral actual. (Eguchi, 2017). Por otra parte, dado que la demanda de profesionales en tecnología, incluyendo programadores, diseñadores e ingenieros, sigue en aumento, el estudio de la robótica puede ser una manera efectiva de fomentar vocaciones en estos campos, así como de promover la participación de mujeres, que están subrepresentadas en la industria tecnológica (Miller & Nourbakhsh, 2016). Es por estas razones que los robots se han convertido en un recurso didáctico presente en muchas escuelas primarias y secundarias del mundo.

Sin embargo, el costo de compra y renovación de materiales, y el temor que algunos docentes presentan ante la complejidad de interactuar con el mundo físico (Reich-Stiebert & Eyssel, 2016), siguen siendo un límite para que esta expansión sea completa. Además, situaciones como las vividas en la pandemia, presentan la necesidad de contar con recursos que puedan utilizarse en forma remota y virtual.

Por otra parte, el aumento del poder de cómputo de las placas gráficas y la disminución de su costo, han permitido la popularización del uso de simuladores de robótica basados en un motor de física. Estos simuladores abren las puertas a la construcción de ambientes realistas, donde la robótica simulada no sea simplemente un desafío de programación, sino que también entren en juego las mismas variables que en el mundo real.

Por estos motivos, desde el 2020 han crecido las propuestas educativas de robótica basadas en simuladores. En este artículo se presenta una comparación entre los simuladores que permiten generar mundos realistas, y la construcción de un ambiente de sumo para la organización de torneos virtuales de largo plazo. Selección de un simulador de robótica para el uso en educación

El uso de simuladores de robótica se ha expandido en diferentes áreas de la industria. Por ejemplo, es un recurso imprescindible en el entrenamiento de una red neuronal de un robot utilizando aprendizaje por refuerzo (Zhao et al., 2020). También es utilizado para la capacitación de operarios en el uso de robots industriales (Tosello et al., 2019). Para seleccionar uno de ellos para ser utilizado en el ámbito educativo, es necesario definir en primer término las características que dicho software debe presentar.

Características deseables en un simulador de robótica para educación

A continuación se presentan cuáles son las características que debe presentar un simulador para que su inserción en la educación sea exitosa (Hughes, Shimizu, et al., 2019):

- El software debe ser gratuito o de bajo costo para alcanzar la mayor cantidad de instituciones educativas posibles. Efectivamente, uno de los motivos por los cuales la robótica física no ha llegado a todas las aulas es por su costo de compra y mantenimiento.
- La interfaz de uso debe ser sencilla e intuitiva. En general, esto no es una característica habitual del software orientado a la industria.
- La instalación debe ser lo suficientemente simple como para no convertirse en una barrera de entrada.
- No debe tener altos requerimientos de cómputo, dado que el costo que no está en el software se trasladaría al equipamiento.
- Debe tener un motor de física subyacente que acerque el comportamiento de los robots al mundo real.

A partir de estos requerimientos, se realizó una comparación entre diferentes programas disponibles en el mercado. A continuación se presentan los más cercanos a los requerimientos mencionados anteriormente.

Virtual robotics toolkit

Conocido como VRT, es un software de la compañía Cogmation compatible con Windows y MacOS. Posee un motor de física avanzado, que facilita la creación de mundos realistas dentro de un conjunto de bloques y componentes electrónicos del mundo Lego Ev3. Tiene un conjunto de robots prearmados y permite importar modelos desde herramientas de CAD basadas en LEGO. La programación también se puede realizar desde los mismos entornos que los modelos físicos. Por último, posee un conjunto de herramientas de aprendizaje, como ejercicios y tutoriales (Simulate FIRST LEGO League & WRO | Virtual Robotics Toolkit, 2020)

Lamentablemente, el software es cerrado y su costo es significativo.

CoderZ

CoderZ es un simulador que se ejecuta en la nube, evitando la necesidad de un equipamiento potente para su ejecución. Contiene un set de robots prearmados, que sólo la empresa puede modificar o ampliar. Presenta una serie de actividades y desafíos orientados a la educación. Con respecto a la programación, ofrece diversos lenguajes como Java y Python, y lenguajes visuales como Blockly (CoderZ Homepage - CoderZ, 2019).

También en este caso el software es cerrado, y aunque tiene un costo bajo, exige una suscripción mensual para su uso.

EyeSim VR

Este software basado en Unity, puede integrarse con un casco de realidad virtual para tener una experiencia aún más inmersiva. Estádiseñado para trabajar con un conjunto de robots específicos (Eyebots), pero permite crear diferentes entornos de simulación con diversos objetos. Puede utilizarse Python y C para la programación de los robots. El software es gratuito, pero cerrado. Tanto su instalación como la interfaz tienen un nivel de complejidad elevado (Bräunl, 2023).

CoppeliaSim

Posee un motor de física avanzado, que simula con precisión el comportamiento de objetos físicos. Permite crear no sólo entornos, sino también robots completos. Posee una biblioteca de modelos ya creada de ambientes y dispositivos. Está basado en una arquitectura de control distribuido, donde cada objeto puede ser controlado en forma independiente, mediante un script embebido (Python o Lua), un plugin en C o C++ y hasta desde un cliente remoto en una gran variedad de lenguajes (Robot simulator CoppeliaSim: create, compose, simulate, any robot - Coppelia Robotics, 2020). Es gratuito y de código abierto, aunque su interfaz es compleja y confusa, y su operación complicada.

Gazebo

Es uno de los programas de simulación de robótica más utilizados en la academia. Tiene una excelente integración con ROS (Robotics Operating System). Su motor de física es robusto, simulando con precisión la dinámica del mundo real. Contiene una amplia biblioteca de modelos, que crece constantemente dada la comunidad activa que posee. Es gratuito y de código abierto. Corre en Linux y MacOS, y aunque puede correr en Windows, su instalación y operación en este sistema operativo es compleja (Gazebo, 2014).

Webots

Este ambiente de simulación también es muy popular en el ámbito académico y hobbista. Su motor de física es ODE, muy robusto y con antigüedad en el mercado. Permite crear

servidores con ejecución en la nube, liberando al usuario de necesitar máquinas potentes para su uso. Cuenta con una gran biblioteca de mundos, robots y objetos, además de que se pueden diseñar los propios. Es gratuito y de código abierto, ejecutándose de forma sencilla en Windows, Linux y MacOS. La programación de los robots se puede realizar en C, C++, Java, Python, Matlab o ROS. Su interfaz de uso es muy sencilla, así como las herramientas para la construcción de mundos, robots y modelos (Michel, 2004).

	Virtual Robotics Toolkit	CoderZ	Eyesim VR	CoppeliaSim	Gazebo	Webots
Gratuito			x	x	x	x
Código abierto				x	x	x
Uso sencillo	x	x				x
Bajos requerimientos		x				
Orientado a educación	x	x	x			
Integración de actividades didácticas		x				
Diferentes robots	x	x		x	x	x
Modificación de actuadores y sensores				x	x	x
Ejecución en la nube		x		x	x	x
Concurrencia de robots	x		x	x	x	x
Programación visual		x		x	x	x
Programación textual		x	x	x	x	x

Tabla 1 - Comparación entre simuladores

Selección final del simulador

Dadas sus características, tanto Gazebo como Webots son los ambientes que más se acercaron a los requerimientos definidos. En el caso de Webots, no sólo su facilidad de uso es mayor, sino que tiene la posibilidad de configurar ciertas características gráficas que permiten su ejecución en hardware menos potente. Finalmente, Robocup también se inclinó por el uso de Webots para la simulación en Rescue Maze, con lo cual finalmente se determinó el uso de este simulador.

Desarrollo de la plataforma de Sumo

En Argentina, desde el año 2000 se llevan a cabo las **Olimpíadas Nacionales de Robótica y Feria de Proyectos**, conocidas como la **Roboliga**. En el 2020, se había proyectado una edición especial para celebrar su vigésimo aniversario, pero lamentablemente, la pandemia impidió su realización. Además, el **Instituto Nacional de Educación Tecnológica** solicitó la colaboración del autor de este artículo para mantener las actividades relacionadas con la robótica y la programación en el contexto virtual que predominaba en las instituciones educativas en ese momento. Por estos motivos, tras realizar un estudio para seleccionar el simulador más adecuado a las necesidades actuales, se optó por crear un entorno de simulación para **Sumo Robótico**. Esta competencia, que cuenta con una gran cantidad de participantes, involucra diversos aspectos de la robótica y presenta un problema sencillo con reglas claras para los competidores.

Figura 1 - Sumo real en la Roboliga

La lucha de Sumo Robótico consiste en dos robots cuyo objetivo es sacar al oponente fuera de la arena, conocida como dohyo. Para que el robot pueda detectar sus límites con sus sensores, habitualmente es negra con un borde alrededor de color blanco. Los robots cuentan con sensores de color o brillo para el piso, y sensores de distancia para detectar al oponente, entre otros.

A pesar de que el motor de física de Webots es muy preciso, las peleas de Sumo son una prueba estresante, dado que se generan colisiones que deben ser evaluadas constantemente. Por este motivo, se eligió un robot ya presente en la biblioteca de forma cilíndrica (Elisa - 3), dado que esta superficie presenta un plano sencillo para estas colisiones.

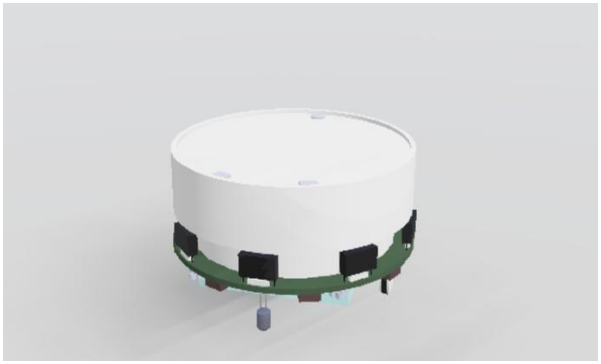

Figura 2 - Elisa - 3

Durante las primeras pruebas, las peleas entre robots carecían de definición, ya que resultaba

imposible que dos robots cilíndricos se empujaran sin que, al resbalar, dejaran de ejercer presión. Como consecuencia, todas las peleas concluían en empate. Para abordar este desafío, se optó por buscar un robot prearmado que se asemejara a los modelos desarrollados por los estudiantes. La elección recayó en el Microbot, un robot con una estructura irregular que se acopla fácilmente a otro para arrastrarlo hacia el borde del área de combate. Además, el Microbot está equipado con un sensor de distancia orientado hacia el suelo, dos paragolpes y un sistema de movimiento diferencial. Para adaptarlo a la competencia de sumo, se ajustó la posición y dirección del sensor de distancia, y se incorporaron dos sensores de brillo dirigidos al suelo para detectar la línea blanca. A los sensores se les agregó un umbral de ruido, para asemejarlos a los sensores del mundo real. Además, se trabajó en la malla de colisión, dado que su estructura original no respetaba el formato real del robot.

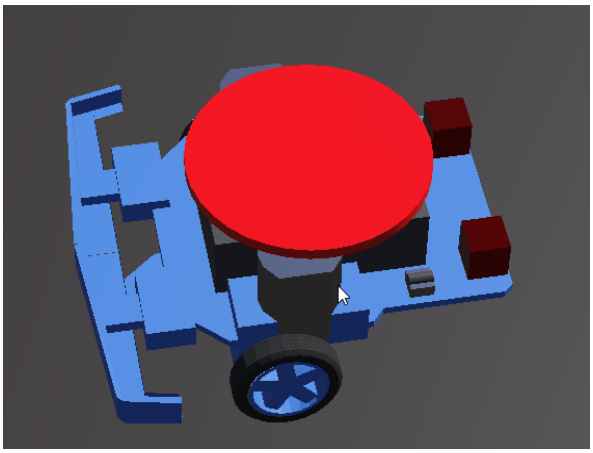

Figura 3 - Robot final

Para la competencia, se creó un ambiente con el dohyo de color negro con borde blanco. En un primer momento se intentó construir un cilindro de una altura de 5 cm como en la competencia del mundo real, pero por las características constructivas de los cilindros en Webots (es un conjunto de sectores circulares), se generaban colisiones aberrantes contra el piso. Por lo tanto, se definió la zona de lucha como una imagen plana aplicada al terreno base.

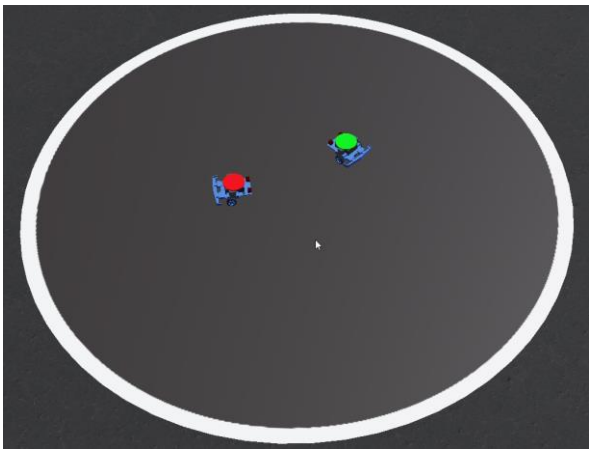

Figura 4 - Dohyo

Las reglas de la competencia de Sumo robótico que se incorporaron al modelo son las siguientes:

a) La lucha tiene una duración de 2:30 minutos continuos.
b) Ambos robots comienzan en un punto predefinido del dohyo, mirando en una dirección determinada.
c) Si el punto central del robot sale por fuera de la línea blanca, se considera que salió del dohyo y queda fuera de combate.
d) Si un robot queda sin cambio de posición durante 20 segundos, se considera fuera de combate.
e) Si uno de los robots llega a 15 segundos sin cambio de posición, y el otro lleva 10 segundos o más en el mismo estado, se realiza una relocalización. Esta consiste en ubicar a los robots en el punto inicial, pero modificando la dirección a la que apuntan.
f) Si al finalizar los 2:30 ninguno de los robots queda fuera de combate, la pelea se considera empatada.

Para poder realizar el arbitraje de la pelea, se desarrolló un sistema automático que no sólo verifica las reglas de juego, sino que también habilita la carga del código de los dos robots

competidores, realiza el posicionamiento original y las relocalizaciones necesarias durante la lucha.

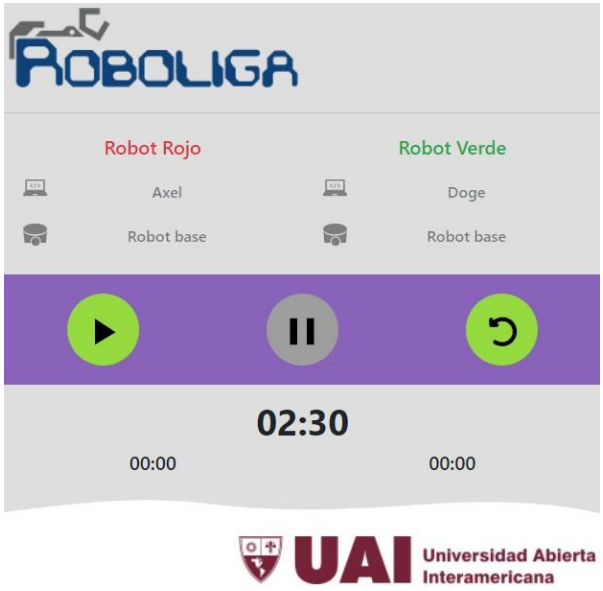

Figura 5 - Sistema de arbitraje

Este sistema de arbitraje está incorporado al modelo de simulación, utilizando un objeto Robot abstracto al que se le suma el código controlador de supervisión (Supervisor). Dicho código está realizado en Python, con una interfaz en web desarrollada en Javascript, HTML y CSS. Cuando se desarrolló el sistema, se abría automáticamente en la misma ventana de simulación, pero a partir de Webots R2022, por cuestiones de seguridad, abre en una ventana aparte en un navegador.

Programación del robot

La programación del robot se realiza en lenguaje Python. Para simplificar esta tarea, y para ocultar algunos aspectos de bajo nivel del funcionamiento de la clase, se decidió hacer un *wrapper* con métodos para:

- Asignar y obtener las velocidades de las ruedas izquierda y derecha.
- Esperar una determinada cantidad de tiempo, manteniendo al robot en el estado en el que se encontraba al comenzar la espera.
- Obtener el tiempo actual.
- Obtener el valor de brillo del sensor que apunta al piso en un rango de 0 a 100.
- Obtener los valores de rojo, verde y azul del sensor que apunta al piso.
- Obtener la distancia de cada uno de los sensores izquierdo y derecho.
- Obtener el estado de los paragolpes izquierdo y derecho.

Para poder realizar tareas de depuración, el comando ***print*** escribe sobre la consola que tiene la interfaz de Webots.

Por otra parte, para simplificar aún más la entrada de nuevos participantes al mundo de la simulación, se desarrolló un lenguaje de programación visual de tipo Blockly, que posee los mismos métodos y estructuras de control y decisión que los disponibles en Python.

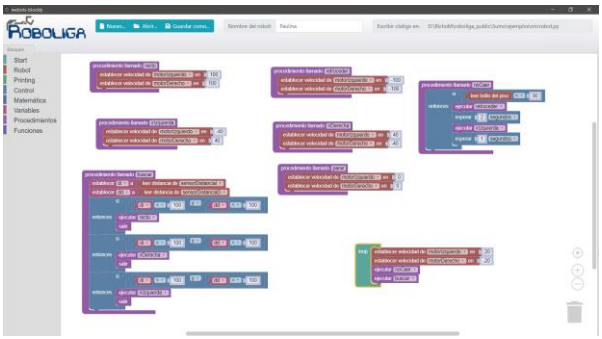

Figura 6 - Entorno gráfico de programación

Herramientas para la Roboliga Virtual

A partir de la experiencia de la Roboliga 2020, que se describe más adelante, se tomó la decisión de realizar un campeonato latinoamericano denominado Roboliga Virtual (https://virtual.roboliga.ar). Para automatizar la ejecución de las luchas se construyó un nuevo Supervisor, que no sólo arbitra las peleas, sino que lleva a cabo la ejecución de las zonas en las que se divide el campeonato. De esta manera, y mediante flags de configuración, se generan un conjunto de luchas que completan una o más rondas de dichas zonas.

Además, se incluyó código para vincular todo el sistema a OBS, un software de grabación de video en tiempo real. De esta manera, mientras se ejecutan las luchas, se generan videos que incluyen placas de presentación, música, exhibición de resultados y de las tablas de cada zona. Con esta herramienta, los campeonatos se pueden ejecutar en forma dinámica, y sin supervisión humana, subiendo luego los videos generados a un servidor para que los participantes puedan analizar su desempeño y prepararse para la siguiente ronda. Esta cualidad simplifica la organización de torneos de muchos equipos.

Personalización del robot

Finalmente, y con el objetivo de enriquecer la diversidad de robots en la competencia, a partir de la Roboliga 2021 se agregó una herramienta para personalizar algunos aspectos del robot. De esta manera, se puede modificar la posición y rotación de los sensores de distancia y de color. Estas pequeñas modificaciones transforman profundamente el comportamiento del robot, y permiten jugar con nuevas variables para mejorar su desempeño.

Figura 7 - Personalizador de los sensores del robot

Resultados del uso del simulador de Sumo en las competencias

A pesar de que la robótica física presenta un conjunto de variables que se acotan drásticamente en una experiencia simulada, el ambiente desarrollado permitió llevar a cabo competencias de Sumo con el mismo nivel de indeterminismo que en el mundo real. Las peleas siguen siendo divertidas y apasionantes para los estudiantes.

Uno de los problemas de las competencias físicas es que, por cuestiones presupuestarias, se desarrollan en uno o dos días. Esto trae aparejado que cualquier problema que pueda surgir en un robot en ese período, deja fuera de competencia a su equipo. Además, el lapso entre competencias no fomenta que los estudiantes reflexionen sobre las virtudes y errores de sus diseños.

En contraposición, las competencias virtuales facilitan la participación de equipos de cualquier lugar del mundo con un costo nulo. Además, se pueden organizar torneos de largo plazo, con competencias por zonas y llaves finales. Esto posibilita el análisis profundo de los participantes en relación a los problemas de sus robots, así como las virtudes y defectos de los robots con los que van a competir. A pesar de las limitaciones inherentes a la simulación, esta experiencia enriquece los aprendizajes obtenidos, proporcionando aspectos que no son posibles en una competencia física.

Equipos participantes en las competencias

La Roboliga es una competencia donde participan estudiantes primarios y secundarios de la República Argentina. Se lleva a cabo desde el año 2000, siendo la más antigua de Latinoamérica.

A pesar de que la pandemia se presentó en un primer momento como un gran limitante, el surgimiento del Sumo Virtual permitió mantener un muy buen número de participantes durante el 2020 y el 2021:

Roboliga 2020: 49 equipos participantes.
Roboliga 2021: 26 equipos participantes.

Durante el 2021, ante el éxito del simulador en el 2020, y con el objetivo de poder realizar una competencia de largo plazo, se decidió organizar un campeonato latinoamericano con una duración de 6 semanas aproximadamente. Dado el éxito en la primera edición, se mantiene su organización anual. Los equipos participantes son:

Roboliga Virtual 2021: 165 equipos de Argentina, Panamá, México, Cuba y Colombia.

Roboliga Virtual 2022: 83 equipos de Argentina, Colombia, Panamá y Ecuador.

Roboliga Virtual 2023: 72 equipos de Argentina, Panamá, Bolivia y Brasil.

En el año 2022 se sumó a la Roboliga Virtual la competencia de rescate Rescue Sim (Erebus) de la Robocup.

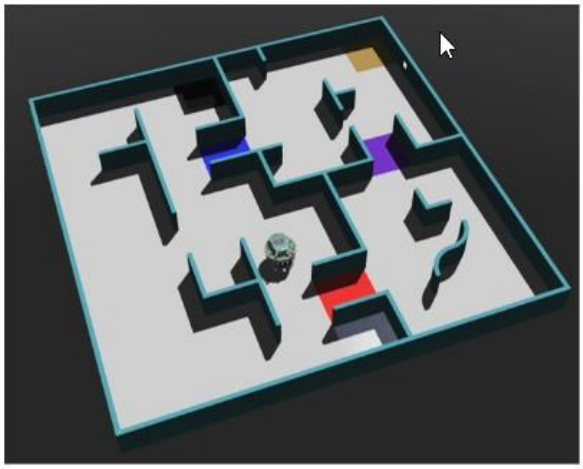

Figura 8 - Rescue Sim (Erebus)

Por último, en 2023 se incorporó una categoría de fútbol de robots, también desarrollado por Robocup pero que aún no entró dentro de la competencia oficial. Sobre el desarrollo realizamos modificaciones para permitir la programación en cualquier lenguaje de programación que implemente comunicación por sockets.

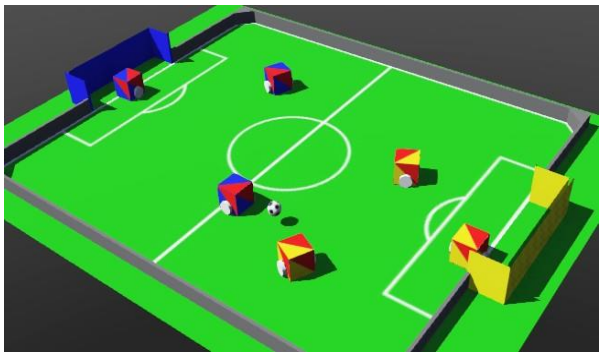

Figura 9 - Fútbol de Robots

Trabajos futuros

A partir de la experiencia con el uso del simulador, más allá de superar los problemas que ocasionó el aislamiento de la pandemia, se vislumbraron algunas características que podrían complementar o superar dificultades habituales de la robótica física. Es por ello que se comenzó el desarrollo de un entorno de aprendizaje que simula los desafíos habituales de la robótica terrestre móvil, como el seguimiento de línea, la evasión de obstáculos, y otros. Para poder comparar el mundo físico y el virtual, se diseñaron robots físicos con características similares al simulado, y se elaboraron un conjunto de actividades con un grupo de control y otro experimental. Estos experimentos serán llevados a cabo en breve.

También se encuentran en producción nuevos mundos y diseños de robots, para poder construir un conjunto de secuencias didácticas que permita una introducción a la robótica complementaria a la actividad con el mundo físico.

Referencias

Bräunl, T. (2023). *Mobile Robot Programming: Adventures in Python and C* (2nd ed. 2023 edition). Springer.

CoderZ Homepage—CoderZ. (2019). https://gocoderz.com/

Eguchi, A. (2017). Bringing robotics in classrooms. En *Robotics in STEM education* (pp. 3-31). Springer.

Gazebo. (2014). http://gazebosim.org/

Hughes, J., Shimizu, M., & Visser, A. (2019). A Review of Robot Rescue Simulation Platforms for Robotics Education. En S. Chalup, T. Niemueller, J. Suthakorn, & M.-A. Williams (Eds.), *RoboCup 2019: Robot World Cup XXIII* (pp. 86-98). Springer International Publishing.

https://doi.org/10.1007/978-3-030-35699-6_7

Michel, O. (2004). Cyberbotics Ltd. Webots™: Professional mobile robot simulation. *International Journal of Advanced Robotic Systems*, *1*(1), 5.

Miller, D. P., & Nourbakhsh, I. (2016). Robotics for education. En *Springer handbook of robotics* (pp. 2115-2134). Springer.

Reich-Stiebert, N., & Eyssel, F. (2016). Robots in the classroom: What teachers think about teaching and learning with education robots. *International Conference on Social Robotics*, 671-680.

Robot simulator CoppeliaSim: Create, compose, simulate, any robot—Coppelia Robotics. (2020). https://www.coppeliarobotics.com/

Simulate FIRST LEGO League & WRO | Virtual Robotics Toolkit. (2020). https://www.virtualroboticstoolkit.com/

Tosello, E., Castaman, N., & Menegatti, E. (2019). Using robotics to train students for Industry 4.0. *IFAC-PapersOnLine*, *52*(9), 153-158. https://doi.org/10.1016/j.ifacol.2019.08.185

Zhao, W., Queralta, J. P., & Westerlund, T. (2020). Sim-to-Real Transfer in Deep Reinforcement Learning for Robotics: A Survey. *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, 737-744. https://doi.org/10.1109/SSCI47803.2020.9308468