Revista Iberoamericana de Automática e Informática Industrial

Revista Iberoamericana de Automática e Informática industrial 14 (2017) 279–287

www.elsevier.es/RIAI

Uso de Plataformas para el Desarrollo de Aplicaciones Virtuales en el Modelado de Robot Manipuladores

Róger E. Sánchez-Alonso[a], Jorge Ortega-Moody[b], José-Joel González-Barbosa[c,*], Guillermo Reyes-Morales[b]

[a] *Universidad Nacional de Ingeniería, Avenida Universitaria, Managua, Nicaragua.*

[b] *Instituto Tecnológico Superior de San Andrés Tuxtla, Carretera Costera del Golfo S/N, Km. 140+100, C.P. 95804. San Andrés Tuxtla, Veracruz, México.*

[c] *Centro de Investigación en Ciencia Aplicada y Tecnología Avanzada, Instituto Politécnico Nacional, Cerro Blanco, N° 141, Colinas del Cimatario, C.P. 76090. Querétaro, QRO, México.*

Resumen

En este trabajo se propone el uso de plataformas para el desarrollo de aplicaciones virtuales como herramientas para el modelado de robots manipuladores. La propuesta se basa en aprovechar el gran potencial que actualmente tienen estas plataformas para solucionar la dinámica de cuerpos rígidos, lo que permite modelar de forma sencilla los aspectos mecánicos del manipulador. Por otro lado, la posibilidad ofrecida por estas plataformas de incorporar código de programación en lenguajes convencionales, permite modelar el comportamiento dinámico de sistemas físicos reales, tales como sensores y actuadores, lo que hace posible la implementación de una etapa virtual de instrumentación y control tal y como se realiza en un robot real. El uso de estas plataformas permite modelar desde cero cualquier robot manipulador. El modelado de un robot paralelo reconfigurable es presentado como caso de estudio.

Palabras Clave: Modelado, Robots manipuladores, Realidad virtual, Sistemas dinámicos.

1. Introducción

El modelado es una etapa vital en el proceso de diseño de robots industriales. Un modelo que incorpore la mayoría de las características físicas de un robot y del ambiente que lo rodea permitirá verificar su funcionalidad real, por ejemplo, analizando su movilidad, estudiando sus configuraciones singulares, determinando su espacio de trabajo, visualizando detalles importantes para el ensamblaje, identificando posibles interferencias mecánicas o bien problemas de control.

Muchos investigadores dedicados a proponer robots manipuladores validan sus mecanismos y sus análisis cinemáticos o dinámicos a través de software de modelado mecánico tales como ADAMS© o SolidWorks® (Sánchez-Alonso *et al*., 2015; Gallardo-Alvarado *et al*., 2013; García-Murillo *et al*., 2013; Sam *et al*., 2012). Este tipo de software permite modelar cualquier robot manipulador incluyendo un sinnúmero de propiedades físicas, por ejemplo, materiales, límites articulares, coeficientes de fricción, fuerzas, momentos, modelos de colisión, entre muchas otras. Sin embargo, la limitante de estas plataformas es que están orientadas al modelado de aspectos mecánicos, lo que constituye un problema cuando se quiere estudiar el comportamiento de un manipulador bajo una acción de control.

Para incorporar aspectos de control al modelado de un robot manipulador es necesario auxiliarse de software más especializado. Matlab®, a través de su módulo Simscape Multibody™, provee una serie de herramientas para este fin. A continuación se presenta un conjunto de referencias para ejemplificar la aplicación de esta plataforma; en (Jamali y Shirazi, 2012) se usó para modelar un manipulador tipo SCARA de 3 GDL (grados de libertad), en (Dung *et al*., 2010) y (de Gea y Kirchner, 2008) para modelar un brazo robótico tipo planar de 2 GDL, en (Dang *et al*., 2013) y (Gao *et al*., 2014) se modela el manipulador paralelo 3-RRR, en (Dalay Udai *et al*., 2011) un manipulador KUKA KR5, y en (Fedák *et al*., 2014) un manipulador serial de 6 GDL. Por otro lado, LabWIEW, a través de su plataforma *Robot Simulation Model Builder* incluye ciertos aspectos de control al modelado de manipuladores industriales. El inconveniente con este tipo de software es que al momento de usar el modelo a través de una simulación, dicha simulación no es interactiva, pues generalmente se ejecuta considerando condiciones de operación y períodos de tiempo previamente definidos, lo cual impide al usuario manipular las variables del modelo durante el proceso de simulación.

Actualmente existe una gran disponibilidad de software para modelar de forma simple y simular interactivamente robots

* Autor en correspondencia.

Correos electrónicos: rogersan1984@hotmail.es (Róger E. Sánchez-Alonso), jorgemoody@gmail.com (Jorge Ortega-Moody), jgonzalezba@ipn.mx (José-Joel González-Barbosa), greyesm_13@hotmail.com (Guillermo Reyes-Morales)

URL: uni.edu.ni

http://dx.doi.org/10.1016/j.riai.2017.04.001

Disponible en www.sciencedirect.com

manipuladores, por ejemplo: RoboDK, Workspace, RobotStudio, WorkcellSimulator, Roboguide, 3DSimulate, RoboLogix, etc. Para fines prácticos del proceso de investigación y diseño de robots este tipo de software presenta dos problemas importantes. Primero, están diseñados para modelar ciertos robots comerciales, es decir no permiten la creación o importación de modelos 3D para que el usuario pueda modelar sus propios diseños. Segundo, este tipo de software permite el modelado para desarrollar básicamente animaciones 3D, en donde no siempre hay un motor de física involucrado para resolver la dinámica de cuerpos rígidos y en donde ninguno de los elementos del robot puede ser modelado como un sistema dinámico, lo que limita el nivel de detalle del modelo e impide el modelado de una etapa de instrumentación y control.

Como una alternativa a estas limitantes existen plataformas más especializadas como Gazebo (Koenig and Howard, 2004) y USARsim (Carpin *et al*., 2007), cuyas versiones más recientes, junto con V-Rep (Rohmer *et al*., 2013) y otros pocos, permiten modelar prácticamente cualquier robot manipulador, incorporan motores de física de alto desempeño, modelos básicos para actuadores y sensores, y la posibilidad de integrar código de programación para implementar y validar algoritmos de control.

Por otro lado, recientemente ha crecido el desarrollo de aplicaciones virtuales cuyo propósito, a diferencia de aplicaciones convencionales, va más allá del entretenimiento. Esta nueva tendencia conocida en inglés como "Serious Games" tiene entre sus principales aplicaciones el desarrollo de simuladores (Zyda, 2005), pues se toma ventaja de las bondades que herramientas como las plataformas para el desarrollo de aplicaciones virtuales, motores de física, procesadores de cómputo y procesadores gráficos ofrecen para desarrollar modelos más detallados. Este nuevo concepto se ha aplicado exitosamente en diferentes contextos, por ejemplo en la industria (Guo *et al.*, 2012; Brasil *et al.*, 2011), la salud (Lancaster, 2014; Erazo *et al.*, 2014; de O Andrade *et al.*, 2013), la educación (Schäfer *et al.*, 2013; Adamo-Villani *et al.*, 2013; Khayat *et al.*, 2012), lo militar (da Silva-Simones y Ferreira, 2011), la seguridad pública (García-García *et al.*, 2012; Backlund *et al.*, 2007), etc.

Tomando en cuenta lo anterior, es que surge el interés de utilizar las plataformas para el desarrollo de aplicaciones virtuales como plataformas para el modelado de manipuladores industriales. La idea general es incluir en el modelado no sólo las características físicas elementales de los componentes que integran a los robots y al ambiente que los rodea, sino también incluir el modelado dinámico de sensores, actuadores y algoritmos de control, lo cual vendría a fortalecer el realismo de las simulaciones derivadas de estos modelos y a abrir una serie de oportunidades al momento de evaluar el comportamiento de un manipulador industrial durante su etapa de diseño.

Este trabajo constituye un esfuerzo para generar alternativas más eficientes y simples de implementar para el modelado de manipuladores industriales desde cero. Las plataformas para el desarrollo de aplicaciones virtuales proveen muchas de las bondades ofrecidas por Gazebo, USARsim, V-Rep y los demás paquetes ya mencionados. Sin embargo, brindan otras ventajas para la navegación, interactividad, visualización 3D, multiplataforma y conectividad con otros dispositivos, lo que constituye aspectos muy importantes a la hora de realizar las simulaciones derivadas del modelado. Las aplicaciones resultantes de esta técnica de modelado pueden ser muchas. Estas pueden ir desde el desarrollo de pruebas pre-operativas de robots o de celdas completas de manufactura, hasta la implementación de laboratorios virtuales para la capacitación en materia de robótica y automatización, la cual es una temática que actualmente está despertando mucho interés (Cerezo y Sastrón, 2015; Candelas *et al*., 2013; Mateo-Sanguino y Andújar-Márquez, 2012; Jara *et al*., 2011; Torres *et al*., 2006). Sea cual fuera la aplicación particular seleccionada, la simulación basada en la técnica de modelado propuesta permitirá llevar la experiencia del usuario a un nivel de realismo superior al provisto por plataformas convencionales.

El resto del trabajo es organizado como sigue: en la Sección 2 la metodología propuesta es presentada; en la sección 3 se presenta el modelado de un robot paralelo como caso de estudio, la plataforma utilizada fue Unity 3D; y finalmente las conclusiones y trabajos futuros son expuestos en la sección 4 y 5 respectivamente.

2. Metodología de modelado propuesta

A finales de los años 90´s se introdujo el concepto de "Hardware in the loop" (Isermann *et al.*, 1999) como una técnica de simulación interactiva basada en el modelado matemático de sistemas complejos, sin embargo, debido a las limitantes tecnológicas de aquella época, su implementación no era muy factible. En la actualidad esta situación es diferente debido a la fácil disponibilidad de ordenadores con capacidad de cálculo y despliegue de gráficos suficiente para poder aplicar este tipo de conceptos, y es ahí donde recae una de las principales fortalezas de la técnica de modelado propuesta.

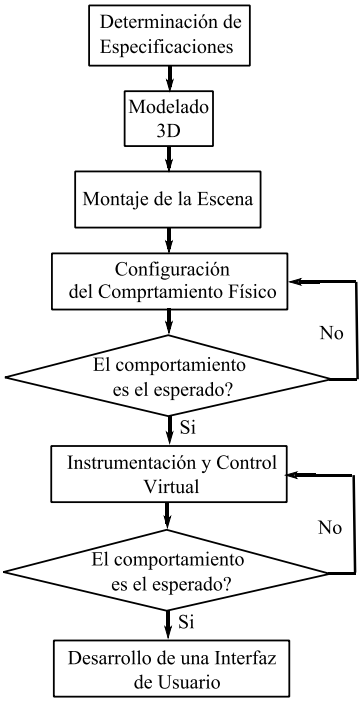

Figura 1: Metodología propuesta.

La técnica de modelado propuesta está pensada para implementarse sobre plataformas para el desarrollo de aplicaciones virtuales que incluyan motores de física capaces de modelar la dinámica de cuerpos rígidos. La plataforma debe

ofrecer además la posibilidad de implementar código de programación para incluir el modelo matemático de todos los sistemas dinámicos involucrados. A continuación se presenta las etapas que componen esta metodología (Figura 1).

2.1. Determinación de especificaciones

En esta etapa se definen todas las características que tendrá el modelo, desde la apariencia general de la escena donde se implementará, hasta las especificaciones técnicas de cada robot, componente y equipo a modelar.

2.2. Modelado 3D

En la actualidad se dispone de muchas herramientas de diseño CAD y modelado 3D que permiten obtener diseños robóticos con un elevado nivel de detalle. Sin embargo hay que tener presente que mientras mayor sea el nivel de detalle, mayor será el costo computacional del modelo al momento de ejecutar una simulación. Es por ello que en esta etapa se debe tener cuidado en modelar a detalle sólo aquellos componentes estrictamente necesarios.

2.3. Montaje de la escena dentro de la plataforma virtual

Todos los modelos 3D obtenidos en la etapa anterior son insertados dentro de la plataforma de desarrollo virtual hasta obtener la escena deseada. Los componentes que forman parte de la escena se pueden dividir en tres grupos; objetos no animados, objetos animados y objetos cuyo comportamiento físico debe ser modelado. Los objetos no animados y animados son esos elementos extra que decoran el ambiente y dan una vista profesional del entorno, la diferencia entre ellos es que unos se mueven y otros no. El montaje de este tipo de componentes es muy simple. Sin embargo en el caso de los robots, cuya movilidad debe ser modelada, se debe hacer mucho hincapié, pues errores en su ensamblaje generan errores en el posicionamiento de su efector final, tal y como sucede con un robot real.

2.4. Configuración del comportamiento físico

En esta etapa se configuran las características físicas elementales de los componentes del robot y el ambiente que lo rodea. Para el tipo de modelo que se propone, básicamente se requiere configurar la gravedad, la detección de colisiones entre cuerpos, la movilidad de las articulaciones que unen los eslabones del robot y algunas propiedades físicas como la masa, fuerzas, torques, etc. Esta etapa implica evaluar constantemente que se vaya obteniendo el comportamiento deseado en el modelo mecánico del robot.

2.5. Instrumentación virtual y control

En esta etapa se modelan los actuadores, sensores y algoritmos de control que permiten representar el movimiento automatizado de un mecanismo, diferenciándose así la simulación que puede arrojar este modelo de una simple animación 3D. Dependiendo del alcance deseado, algunos sensores y actuadores pueden ser modelados de forma simple, por ejemplo sensores de proximidad y efectores finales cuyo estado en el tiempo está asociado a una variable booleana. Sin embargo otros dispositivos deben ser modelados como sistemas dinámicos, por ejemplo los servomotores de los robots, los cuales son básicamente sistemas electromecánicos que incluyen un motor, un sensor de posición y una etapa de control PID. Lo anterior no significa que la metodología se limita al modelado de este tipo de actuadores, por el contrario, la metodología es tan abierta que se puede incluir cualquier otro tipo de sistema que pueda interactuar con un sistema robótico, por ejemplo sistemas térmicos o hidráulicos, los cuales son sistemas cuyo modelado ya ha sido muy estudiado. En (Palm, 1998; Ljung y Glad, 1994; Ogata, 2010) se puede encontrar mayor información acerca del modelado y control de sistemas dinámicos.

Adicionalmente, en esta etapa también se pueden programar rutinas o secuencias a realizar por el robot o por las celdas robóticas presentes en la escena. Esta etapa, al igual que la anterior, requiere de la evaluación exhaustiva de que el comportamiento obtenido corresponda al de los sistemas modelados, para ello se puede hacer uso de herramientas como Malbab y su módulo Simulink.

2.6. Desarrollo de una interfaz de usuario

El modelado de una interfaz de usuario permite interactuar con el robot y con todos los elementos disponibles en la escena. En dependencia de los controles e indicadores modelados, el usuario puede, por ejemplo, configurar la masa de cada elemento del robot para evaluar la capacidad de carga de un actuador en particular, puede modificar las características mecánicas y eléctricas de un motor para apreciar su comportamiento en el tiempo, puede sintonizar el controlador implementado bajo diferentes cargas, visualizar el status de todo el sistema, visualizar señales de control, errores de posicionamiento, configurar rutinas de trabajo, operar el robot, etc.

3. Caso de estudio: Modelado de un robot paralelo reconfigurable

Para este caso de estudio la plataforma de desarrollo virtual utilizada es Unity 3D en su versión 4.6, la cual cuenta con el motor de física PhysX^{TM}. Esta plataforma permite desarrollar código de programación en lenguajes de alto nivel como C# y Java, los cuales pueden ser elaborados en IDEs como Microsoft Visual Studio o MonoDevelop (Licencia Pública General). En esta sección se presentan los aspectos más importantes para la implementación de la metodología.

3.1. Determinación de especificaciones

Las especificaciones tomadas en cuenta para el desarrollo de este caso de estudio se pueden agrupar en dos categorías; las referidas estrictamente a la geometría del manipulador a modelar y las referidas al modelado físico de cada uno de los componentes del robot y su entorno.

Geometría del manipulador. El robot a modelar es un manipulador reconfigurable que está integrado por dos sub-manipuladores paralelos que comparten la plataforma móvil (Figura 2a). A menos que se especifique lo contrario, los subíndices $i = 1, 2, 3$ están asociados respectivamente a cada una de las tres cadenas cinemáticas que integran al sub-manipulador llamado M_1, de forma similar, los subíndices $i = 4, 5, 6$ están asociados a las tres cadenas cinemáticas que integran al sub-manipulador llamado M_2.

En cualquiera de las seis cadenas cinemáticas del robot están presentes los eslabones LA_i y LB_i, sin embargo, las cadenas cinemáticas del sub-manipulador M_1 cuentan con un eslabón adicional llamado Rr_i (Figura 2b).

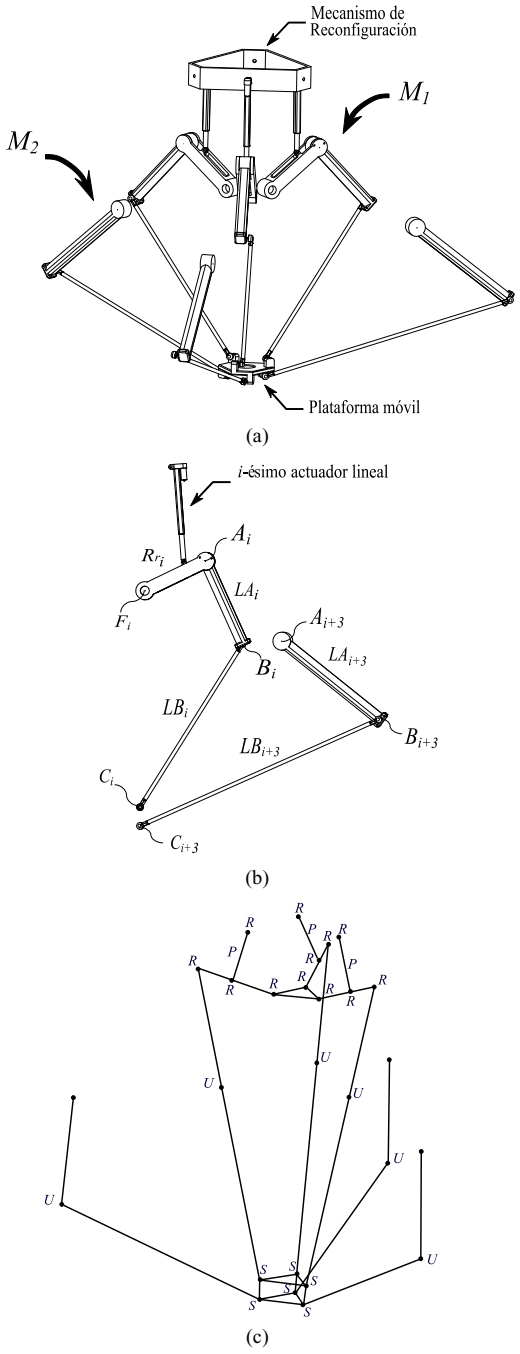

Figura 2: Manipulador paralelo a simular. (a) Vista general. (b) Estructura de la *i*-ésima y la *i+3*-ésima cadena cinemática. (c) Esquema articular del manipulador.

La movilidad de la plataforma móvil del robot, expresada en tres rotaciones y tres traslaciones, se genera por el movimiento angular independiente de los eslabones Rr_i y LA_i. En el caso de los eslabones Rr_i, su movilidad alrededor de las articulaciones pasivas de revoluta definidas en los puntos F_i es generada por la acción de tres actuadores lineales localizados en la parte superior del robot. Estos tres actuadores lineales forman el mecanismo que reconfigura al robot. Por otro lado, la movilidad de los eslabones LA_i alrededor de las articulaciones de revoluta definidas en los puntos A_i es generada por la acción de actuadores rotacionales. En lo que respecta al arreglo articular restante, los eslabones LA_i están conectados a los eslabones LB_i en los puntos B_i mediante articulaciones universales, mientras que los eslabones LB_i están conectados a la plataforma móvil en los puntos C_i mediante articulaciones esféricas. En la Figura 2c se presenta el esquema articular completo del robot (*R* representa articulaciones de revoluta, *P* articulaciones prismáticas, *U* articulaciones universales y *S* articulaciones esféricas).

Para mayor información acerca de este robot; su geometría, análisis cinemático, ventajas sobre mecanismos similares y oportunidades derivadas de su reconfiguración, revisar (Sánchez-Alonso *et al*., 2016).

Componentes del manipulador y su entorno. El modelo debe incluir las características físicas elementales para poder representar el mecanismo, es decir, el robot debe estar sujeto a la acción de la gravedad, sus componentes deben incluir todas propiedades inerciales de un cuerpo rígido, la movilidad de las articulaciones activas y pasivas debe ser modelada y cada componente debe ocupar un único espacio (detección de colisiones). Se incluirá una cámara virtual para visualizar interactivamente el movimiento del efector final del robot. El robot debe estar habilitado para manipular objetos dentro de su espacio de trabajo, para ello se debe modelar un efector final y un sensor de proximidad. Los servomotores del manipulador deben estar modelados como sistemas dinámicos y una etapa de control PID será implementada para controlar su posición angular. La movilidad del manipulador debe estar basada en su modelo cinemático inverso. Finalmente, se agregarán elementos extras como objetos no animados, colores y luces para dar un acabado profesional a la escena que envuelve al modelo final.

Con respecto a la interacción con el modelo, se desarrollará un interfaz de usuario que permita: (i) revisar el estatus general de los sensores y actuadores del robot, (ii) configurar la masa de los componentes principales del robot, (iii) configurar las propiedades eléctricas y mecánicas de los motores, (iv) sintonizar el controlador PID, deshabilitar la acción de control o controlar en lazo abierto cada motor, y (v) operar el robot.

3.2. Modelado 3D

Todos los modelos CAD utilizados fueron desarrollados a escala real en SolidWorks®, luego fueron exportados en formato STL (Stereo Lithography) hacia 3DS MAX, donde se trabajó en la reducción de los polígonos que forman el mallado de cada modelo 3D. El propósito de este tratamiento es disminuir el costo computacional del despliegue gráfico durante la simulación del modelo. Posteriormente cada modelo 3D es enviado en formato FBX (FilmBox) hacia Unity 3D.

3.3. Montaje y configuración del comportamiento físico

Cada uno de los modelos 3D desarrollados debe ser localizado dentro del área gráfica de Unity 3D tratando de recrear la escena deseada. En la Figura 3 se muestra el montaje de una de las tres cadenas cinemáticas del sub-manipulador M_1. Cada una de las partes de esta cadena cinemática debe ser configurada dentro de

Unity 3D con la opción "Rigid Body" (Cuerpo Rígido). De esta manera el motor de física puede tener efecto sobre cada cuerpo, lo cual significa que actuará la gravedad, se le podrá asignar masa, un material especifico, detección de colisiones, aplicar una fuerza, un torque, etc.

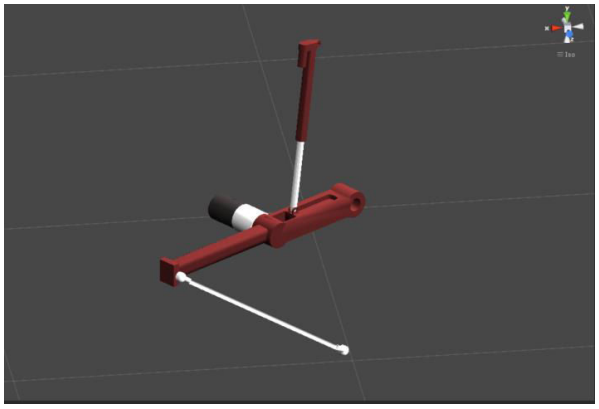

Figura 3: Montaje de una cadena cinemática del robot dentro de Unity 3D.

Para convertir el montaje mostrado en la Figura 3 en un sub-ensamble del robot se deben modelar las articulaciones que conectan a cada cuerpo con otro. En Unity 3D se puede modelar la movilidad y el comportamiento de prácticamente cualquier articulación comúnmente utilizada en robótica; articulaciones de revoluta, prismáticas, universales, esféricas, etc., y gran parte de ello se debe a la disponibilidad de la herramienta "Configurable Joint" (Articulación Configurable). En la Figura 4 se muestra una parte del menú de opciones para la configuración de una articulación que conecta uno de los eslabones LB_i con la plataforma móvil (llamada Pmov) en el punto de giro X = -0.0865 m, Y = -0.0446 m, Z = -0.5878 m de su sistema de referencia local. Dicha articulación representa una de tipo esférica, pues tal y como se observa el movimiento lineal sobre los ejes X, Y y Z fueron convenientemente bloqueados.

Figura 4: Ejemplo de la configuración de una articulación esférica en Unity 3D.

En el caso de las articulaciones activas, es decir aquellas que son actuadas por motores, aparte de definir el punto de giro y restringir los ejes de rotación y traslación requeridos, es necesario configurarlas en modo "Velocity". Lo anterior permite aplicar a un cuerpo una determinada velocidad angular, lo que básicamente simula la acción de un motor. Otra propiedad importante que se debe configurar en las articulaciones activas es el máximo torque que pueden generar.

Por otro lado, sobre cada cuerpo se debe configurar un "Collider" (Colisionador) para simular su interacción con otros cuerpos. Un collider es una región envolvente que delimita el espacio de contacto de un cuerpo. Los tipos de collider disponibles en Unity 3D son: Box Collider (tipo caja), Sphere Collider (tipo esfera), Capsule Collider (tipo cápsula) y Mesh Collider (tipo malla). En la Figura 5 se muestra la aplicación de un Mesh Collider a uno de los eslabones LB_i del robot, el cual es el tipo de collider disponible más preciso para la detección de colisiones, pues genera una malla limitada a 255 triángulos que se ajusta a la superficie del sólido de interés. En Unity 3D un Mesh Collider debe ser configurado como Convex (convexo) para poder interactuar con otro Mesh Collider.

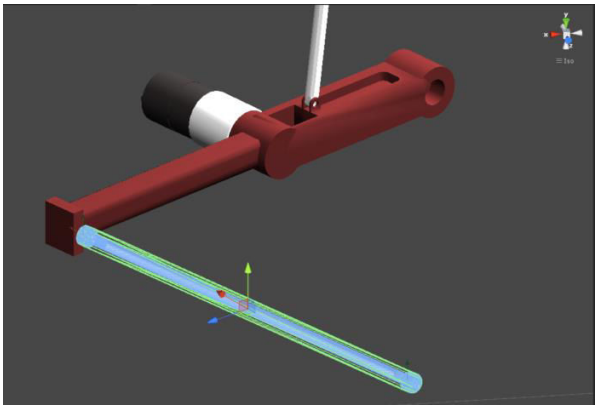

Figura 5: Aplicación de un Mesh Collider a uno de los eslabones del robot.

Cuando el procedimiento anterior se aplica a todos los componentes del robot se obtiene su modelo virtual. En la Figura 6 se muestra el robot prácticamente ensamblado. En esta figura se observan las cadenas cinemáticas del robot completamente sueltas, pendiendo de la base fija bajo únicamente la acción de la gravedad.

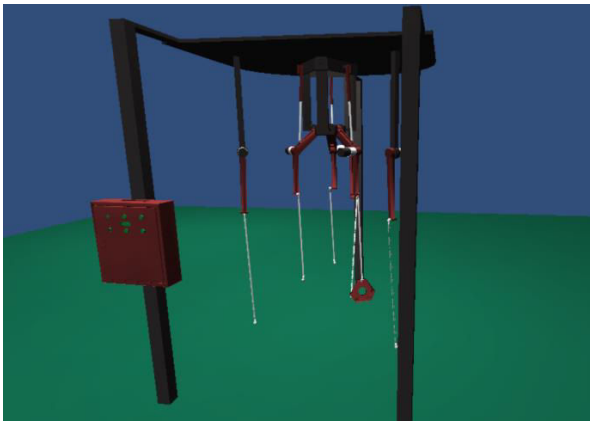

Figura 6: Modelo virtual del robot bajo la acción de la gravedad.

Finalmente se agregan algunos elementos para mejorar el aspecto visual del escenario. En la Figura 7 se muestra una vista del escenario completo, en donde se puede apreciar la inclusión de luces, colores y otros objetos, entre los que destaca un monitor que proyecta el streaming de video captado por una cámara virtual instalada en el robot. Además se aprecia la presencia de cinco cubos de colores dispuestos sobre una mesa para ser manipulados por el robot. Hasta este punto el modelo desarrollado representa las características de un mecanismo paralelo sin ningún tipo de sensor y actuador para controlar su movimiento.

Figura 7: Escenario de la simulación.

3.4. Instrumentación virtual y control

A continuación se presenta la forma en que se modeló esta etapa para el robot.

Sensor de proximidad y efector final. Ambos dispositivos se pueden modelar en Unity 3D a través de código de programación muy simple. En el caso del sensor de proximidad se utiliza la opción "Trigger" (Disparo) de los colliders, de manera que cuando el collider del efector final entre en contacto con el collider de otro objeto, una variable booleana previamente definida toma valor verdadero, misma variable que toma valor falso cuando los colliders no están en contacto. El rango de acción del sensor está íntimamente relacionado con la posición y tamaño del collider del efector final. Por otro lado, el efector final es simulado utilizando la opción "Parent" de los objetos definidos dentro de Unity 3D, de esta manera un objeto se puede adjuntar o separar jerárquicamente del efector final del robot mediante la acción de una variable booleana.

Servomotores DC. Estos dispositivos juegan un papel importante dentro del proceso de simulación, por ello deben ser modelados como sistemas dinámicos, es decir conviene no sólo conocer su respuesta en el estado final, también es necesario obtener su comportamiento en el estado transitorio. El modelado de un servomotor de corriente directa implica la representación de básicamente tres sub-sistemas: un motor de corriente directa, un sensor de posición angular y una etapa de control.

En lo que respecta al motor de corriente directa, éste puede ser representado a través de la siguiente función de transferencia de lazo abierto.

$$\frac{\Omega_{(s)}}{V_{(s)}} = \frac{k_t}{\left(Js + B_m\right)\left(Ls + R\right) + k_t k_m} \tag{1}$$

donde; $\Omega_{(s)}$ es la velocidad angular de la flecha del motor, $V_{(s)}$ el voltaje de alimentación, k_t y k_m son respectivamente la constante de par y la constante de voltaje del motor, J el momento de inercia del rotor, B_m la constante de fricción viscosa, L la inductancia eléctrica y R la resistencia eléctrica.

La expresión (1) es obtenida a través de las ecuaciones diferenciales que describen distribución del voltaje de alimentación del motor en sus elementos eléctricos y la distribución del par en sus elementos mecánicos, el cual es un procedimiento que puede ser verificado en (Ogata, 2010). Esta expresión representa un sistema continuo en el dominio de Laplace, y para poder implementarla como un sistema discreto en Unity 3D, se debe aplicar la transformada bilineal (método Tustin). En dicha transformada se especifica el período de muestreo *T*, el cual para el caso de Unity 3D está fijado en 0.02 s.

De lo anterior se puede calcular la velocidad angular de la flecha de un motor con determinadas características eléctricas y mecánicas dado un voltaje de entrada, lo cual junto con el hecho de que las articulaciones simuladas en Unity 3D pueden ser configuradas como activas e inducirles dicha velocidad angular, permite modelar el movimiento de un eslabón activo generado por un motor de corriente directa.

Por otro lado, el sensor de posición del servomotor se puede modelar a través de la lectura en cada instante de tiempo de la posición angular del eslabón activo respecto a su eje de rotación. La diferencia entre un SetPoint de posicionamiento angular (*SP*) y el valor actual (*PV*) registrado por el sensor de posición virtual permite conocer el error *E(n)* de posicionamiento en un instante *n* del tiempo, el cual es un dato muy importante para la implementación de la etapa de control del servomotor.

La etapa de control de posición implementada en los servomotores es de tipo PID, en donde la señal de control *Vin* se estima por la sumatoria de una acción de control proporcional (*P*), una integral (*I*) y una derivativa (*D*). Estas acciones de control fueron definidas en función del error *E(n)* tal y como se muestra:

$$P_{(n)} = K_p E_{(n)} \tag{2}$$

$$I_{(n)} = K_i T\left(E_{(n)} + I_{(n-1)}\right) \tag{3}$$

$$D_{(n)} = K_d\left(\frac{E_{(n)} - E_{(n-1)}}{T}\right) \tag{4}$$

donde K_p, K_i y K_d representan las ganancia proporcional, integral y derivativa que permiten sintonizar el controlador para obtener un comportamiento deseado.

3.5. Desarrollo de la interfaz de usuario

Para este caso de estudio se desarrolló una interfaz con cinco menús, los cuales se pueden observar en la Figura 8. La funcionalidad de cada menú se describe a continuación:

Menú Status: la función principal de este menú es mostrar el estado de algunas variables durante la simulación. Se puede leer la posición angular actual de cada motor y su máxima velocidad angular. En la parte inferior se encuentran tres indicadores: "Position Reached", el cual indica cuando los motores han llegado a su SetPoint, "Effector Sensor", que indica cuando alguna pieza que se desea manipular ya está dentro del campo de acción del efector final, y "Effector On/Off", que indica cuando el efector final fue activado o desactivado para manipular una pieza.

Menú Mass: Este menú permite configurar de forma interactiva la masa de los principales eslabones del robot y la plataforma móvil (M.P). Cuando la simulación del modelo es ejecutada, ésta alcanza un nivel de realismo tal que si se seleccionan masas que no pueden ser cargadas por los motores se verá cómo el robot o las cadenas cinemáticas con menos capacidad de carga no pueden sostenerse.

Menú Motor: Este menú permite configurar las características eléctricas y mecánicas de cada uno de los motores del robot. En la parte superior de este menú el usuario puede seleccionar el motor que desea configurar. En este punto es importante mencionar que el submenú llamado "Motor 0" configura los tres servomotores que forman el mecanismo de reconfiguración del robot. Adicional a todas las características que fueron tomadas en cuenta para el modelado dinámico de los motores, en este menú también se pueden configurar el máximo torque y la relación de reducción en los engranajes del motor. Recuérdese que la asignación del máximo torque es posible gracias a una de las opciones avanzadas de la "Configurable Joint" cuando es configurada como activa.

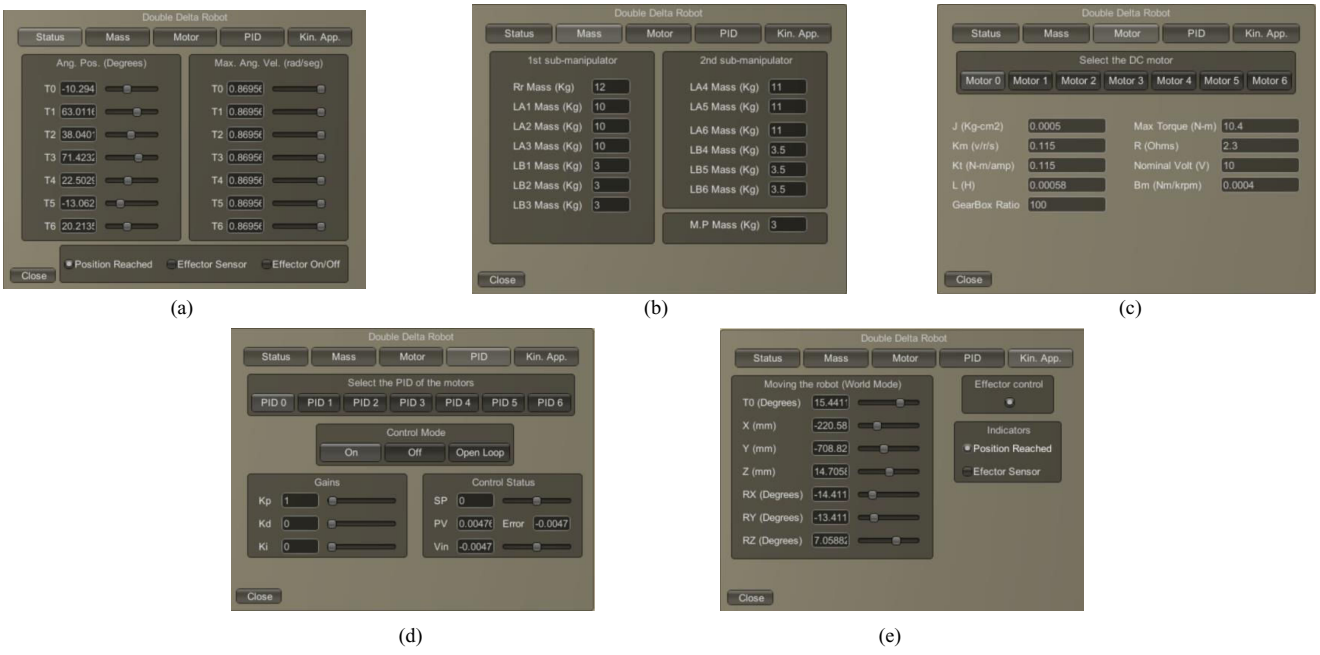

Fig. 8. Interfaz de usuario. (a) Menú Status. (b) Menú Mass. (c) Menú Motor. (d) Menú PID. (e) Menú Kin. App.

Menú PID: Este menú permite configurar algunos parámetros relacionados con el lazo de control PID de cada motor. En la parte superior de este menú el usuario puede seleccionar el PID que necesita configurar. Similar al menú anterior, el submenú llamado "PID 0" sirve para configurar el algoritmo de control de los tres servomotores que forman el mecanismo de reconfiguración del robot. A lo interno de cada submenú está la posibilidad de configurar el modo de operación del algoritmo de control, en este sentido se puede seleccionar entre las opciones de control: "On", "Off" y "Open Loop". La opción "On" habilita el lazo de control PID, lo cual permite al usuario sintonizar el controlador a través de las ganancias K_p , K_i y K_d mientras visualiza el estado de las variables de control; SetPoint (*SP*), valor actual del proceso (*PV*), error y señal de control (*Vin*). La opción "Off" permite generar una señal de control igual a cero, lo cual implica básicamente apagar ese motor. Finalmente la opción "Open Loop" permite configurar manualmente la señal de control.

Menú Kin. App: Este menú permite al usuario operar el robot, para lo cual es necesario embeber el modelo cinemático inverso del robot (Sánchez-Alonso *et al*., 2016). El usuario puede mover el efector final del robot a lo largo de los ejes coordenados *X*, *Y* y *Z*, además puede rotarlo alrededor de estos tres ejes. Adicionalmente se puede controlar de forma simultánea el movimiento angular (T0) de los eslabones *Rr* que generan la reconfiguración en el robot. En este menú el usuario tiene a su disposición el control del efector final del robot y los indicadores "Position Reached" y "Efector Sensor".

La interfaz de usuario desarrollada se despliega directamente sobre el área gráfica del usuario, tal y como se muestra en la Figura 9. En dicha figura se puede observar al robot manipulador apilando unos cubos pequeños dentro de su espacio de trabajo operable.

Una vez que se obtiene el modelo virtual del robot, este puede ser encapsulado y guardado como un archivo de Unity para posteriormente ser cargado en otros escenarios.

En la Figura 10 se muestra el robot modelado dentro de un escenario que representa un proceso de manufactura. Se puede observar la presencia de otros dos robots, un Delta y un Serial de 3 y 6 grados de libertad respectivamente. Ambos robots fueron modelados con la metodología que se propone en este trabajo.

Otro aspecto importante a resaltar es la facilidad con la que el usuario puede navegar dentro de la escena una vez que se ejecuta una simulación. El usuario puede escoger entre una navegación en primera o tercera persona, por ejemplo, en la Figura 10 se observa la inclusión de un "Avatar" inmerso en la escena para navegar en tercera persona.

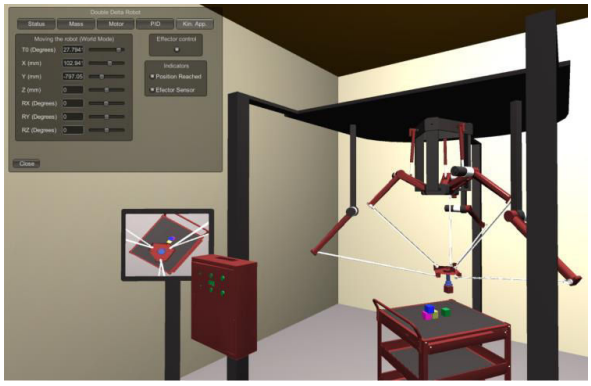

Figura 9: Vista del robot simulado apilando objetos en su espacio de trabajo.

Figura 10: Escenario con la inclusión de múltiples robots.

Por otro lado, no es necesario que el usuario navegue dentro de la escena para monitorear o controlar el proceso que está simulando. Lo anterior se puede hacer desde un cuarto de control, tal y como se hace en la vida real mediante un sistema SCADA.

Figura 11: Ejemplo de un cuarto de control.

4. Conclusiones

En este trabajo se propone el uso de plataformas para el desarrollo de aplicaciones virtuales como herramientas para el modelado robots industriales. Usar este tipo de plataformas permite modelar cualquier tipo de mecanismo, sin importar sus grados de libertad, estructura cinemática, tipo de actuadores, arreglo articular, etc., pues incluyen motores de física muy robustos que permiten solucionar la dinámica de cuerpos rígidos. Los aspectos específicos no soportados por estas plataformas, por ejemplo la implementación de sensores, actuadores y algoritmos de control pueden ser modelados por el usuario gracias a la disponibilidad de incluir código de programación.

Esta contribución pone a la disposición de la comunidad académica y tecnológica una alternativa útil para el modelado y la validación de la funcionalidad real de un diseño robótico o de una celda completa. De la misma forma provee una plataforma para la implementación de pruebas de control y automatización previas a la construcción de un prototipo real, lo que puede suponer un ahorro económico sustantivo en muchos casos.

El abanico de aplicaciones derivadas de esta propuesta es muy grande. Sobre todo en temáticas referentes a simulación, pues se aprovechan las ventajas para la navegación, interactividad, visualización 3D, multiplataforma y conectividad con otros dispositivos que este tipo de plataformas provee. Por ejemplo, a través de este tipo de plataformas se pueden simular plantas industriales completas, en donde el comportamiento físico de cada elemento en la escena esté caracterizado y modelado.

Las aplicaciones de esta metodología no se limitan a la simulación con fines de diseño. Las oportunidades derivadas de esta metodología son muchas, por ejemplo, se pueden desarrollar plataformas virtuales para el entrenamiento de personal en la operación de robots manipuladores. Por otro lado, desde el punto de vista académico, este desarrollo puede constituir una plataforma para la implementación y validación de nuevos algoritmos de control para robots industriales o de cualquier tipo. La técnica de modelado propuesta constituye en definitiva una alternativa con muchas ventajas y oportunidades que pueden ser explotadas para llevar la experiencia de modelado y simulación de robots a un nivel de realismo superior al que proveen plataformas convencionales.

5. Trabajo Futuro

Las aplicaciones virtuales derivadas de la metodología propuesta son completamente locales, por lo que definitivamente un aspecto que sería importante desarrollar es la comunicación de este tipo de aplicaciones virtuales con aplicaciones o dispositivos externos. En el caso particular de dispositivos externos, valdría mucho la pena invertir esfuerzos de investigación en implementar, por ejemplo, la comunicación de este tipo de escenarios virtuales con un PLC real.

Otra directriz de trabajo futuro está orientada hacia aumentar la experiencia realista del usuario al momento de ejecutar simulaciones, por ejemplo a través de cascos de realidad virtual o a través de nuevas técnicas como la realidad aumentada.

English Summary

Use of Platforms for the Development of Virtual Applications in the Modeling of Robot Manipulators

Abstract

This paper describes the use of platforms for the development of virtual applications as tools for modeling of robot manipulators. The proposal is based on take advantage of the

potential that these platforms currently have for solving the rigid body dynamics, which easily allows modeling the mechanical aspects of the manipulator. On the other hand, the possibility offered by these platforms of incorporate programming code in conventional languages allows to modeling the dynamic behavior of real physical systems, such as sensors and actuators, which allows implementing the development of the instrumentation and control stage of an industrial robot in the same way as a real one. Using these platforms allows the modeling from the bases of any manipulator robot. The modeling of a reconfigurable parallel robot is presented as a case study.

Keywords:
Modeling, Manipulator Robots, Virtual reality, Dynamic systems.

Referencias

Adamo-Villani, N., Haley-Hermiz, T., Cutler, R., 2013. Using a Serious Game Approach to Teach 'Operator Precedence' to Introductory Programming Students, In 17th International Conference Information Visualisation (IV), London, 523-526. IEEE.

Backlund, P., Engstrom, H., Hammar, C., Johannesson, M., Lebram, M., 2007. Sidh-a Game Based Firefighter Training Simulation, In 11th International Conference Information Visualization, Zurich, 899-907. IEEE.

Brasil, I., Neto, F., Chagas, J., Monteiro, R., Souza, D., Bonates, M., Dantas, A., 2011. An intelligent and persistent browser-based game for oil drilling operators training, In 2011 IEEE 1st International Conference on Serious Games and Applications for Health (SeGAH), Braga, 1-9. IEEE.

Candelas, F., Puente, S., Torres, F., Ortiz, F., Gil, P., Pomares, J., 2013. A Virtual Laboratory for Teaching Robotics, International Journal of Engineering Education, 19 (3), 363-370.

Carpin, S., Lewis, M., Wang, J., Balakirsky, S., Scrapper C., 2007. USARSim: a robot simulator for research and education, In Proceedings of the 2007 IEEE International Conference on Robotics and Automation, Roma, 1400-1405, IEEE.

Cerezo, F., Sastrón, F., 2015. Laboratorios virtuales y docencia de la automática en la formación tecnológica de base de alumnos preuniversitarios, Revista Iberoamericana de Automática e Informática Industrial, 12 (4), 419-431.

da Silva-Simones, P., Ferreira, C., 2011. Military war games edutainment, In IEEE 1st International Conference on Serious Games and Applications for Health (SeGAH), Braga, 1-7. IEEE.

Dalay Udai, A., Rajeevlochana, C. G., Kumar Saha, S., 2011. Dynamic Simulation of a KUKA KR5 Industrial Robot using MATLAB SimMechanics, In 15th National Conference on Machines and Mechanisms, India.

Dang, X. Z., Zhou, L. S., Liao, L. P., Liang, D., 2013. Modelling and Simulation of Forward Kinematics for Planar 3-DOF Parallel Robot Based on Simulink, Applied Mechanics and Materials, 397, 1552-1557.

de Gea, J., Kirchner, F., 2008. Modelling and Simulation of Robot Arm Interaction Forces Using Impedance Control, In Proceedings of the 17th World Congress, The International Federation of Automaic Control, Seoul, Korea.

de O. Andrade, K., Fernandes, G., Martins, J., Roma, V., Joaquim, R., Caurin, G., 2013. Rehabilitation robotics and serious games: An initial architecture for simultaneous players, In Biosignals and Biorobotics Conference (BRC), Rio de Janerio, 1-6. IEEE.

Dung, L. T., Kang, H. J., Ro, Y. S, 2010. Robot manipulator modelling in MatlabSimmechanics with PD control and online Gravity compensation, In 2010 International Forum on Strategic Technology (IFOST), Ulsan, 446-449. IEEE.

Erazo, O., Pino, J., Pino, R., Fernandez, C., 2014. Magic Mirror for Neurorehabilitation of People with Upper Limb Dysfunction Using Kinect, In 47th Hawaii International Conference on System Sciences (HICSS), Waikoloa, 2607-2615. IEEE.

Fedák, V., Ďurovský, F., Üveges, R., 2014. Analysis of Robotic System Motion in SimMechanics and MATLAB GUI Environment, MATLAB Applications for the Practical Engineer, Mr Kelly Bennett (Ed.), InTech.

Gallardo-Alvarado, J., García-Murillo, M., Castillo-Castañeda, E., 2013. A 2(3-RRPS) parallel manipulator inspired by Gough-Stewart platform, Robotica, 31 (3), 381-388.

Gao, J. R., Wang, Y. Z, Chen, Z. P., 2014. Modelling and Simulation of Inverse Kinematics for Planar 3-RRR Parallel Robot Based on SimMechanics, Advanced Materials Research, 898, 510-513.

García-García, C., Fernández-Robles, J., Larios-Rosillo, V., Luga, H., 2012. ALFIL: A Crowd Simulation Serious Game for Massive Evacuation Training and Awareness, International Journal of Game-Based Learning, 2 (3), 71-86.

García-Murillo, M., Castillo-Castañeda, E., Gallardo-Alvarado, J., 2013. Dynamics of a 2(3-RRPS) parallel manipulator, In 9th Workshop on Robot Motion and Control (RoMoCo), Kuslin, 270-275. IEEE.

Guo, H., Li, H., Chan, G., Skitmore, M., 2012. Using game technologies to improve the safety of construction plant operations. Accident Analysis & Prevention, 48, 204-213.

Isermann, R., Schaffnit, J., Sinsel, S., 1999. Hardware-in-the-loop simulation for the design and testing of engine-control systems, Control Engineering Practice, 7 (5), 643-653.

Jamali, P., Shirazi, K. H., 2012. Robot Manipulators: Modeling, Simulation and Optimal Multi-Variable Control, Applied Mechanics and Materials, 232, 383-387.

Jara, C., Candelas, F., Puente, S., Torres, F., 2011. Hands-on experience of undergraduate students in automatic and robotics using a virtual lab and remote laboratory. Computers & Education, 57 (4), 2451-2461.

Khayat, G., Mabrouk, T., Elmaghraby, A, 2012. Intelligent serious games system for children with learning disabilities, In 17th International Conference on Computer Games (CGAMES), Louisville, 30-34. IEEE.

Koenig, N., Howard, A., 2004. Design and use paradigms for Gazebo, an open-source multi-robot simulator, In Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, (3) 2149-2154, IEEE.

Lancaster, R., 2014. Serious Game Simulation as a Teaching Strategy in Pharmacology. Clinical Simulation in Nursing, 10 (3), 129-137.

Ljung, L., Glad, T., 1994. Modeling of Dynamic systems, PTR Prentice Hall.

Mateo-Sanguino, T., Andújar-Márquez, J., 2012. Simulation tool for teaching and learning 3D kinematics workspaces of serial robotic arms with up to 5-DOF. Computer Applications in Engineering Education, 20 (4), 750-761.

Ogata, K., 2010. Ingeniería de Control Moderna, PEARSON EDUCACIÓN, S.A., Madrid, 5ta ed.

Palm, W. J., 1998. Modeling, Analysis, and Control of Dynamic Systems, John Wiley & Sons, New York, NY, 2nd ed.

Rohmer, E., Singh, S. P. N., Freese, M., 2013. V-REP: A versatile and scalable robot simulation framework, In Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, 1321-1326, IEEE.

Sam, R., Arrifin, K., Buniyamin, N., 2012. Simulation of pick and place robotics system using Solidworks Softmotion, In 2012 International Conference on System Engineering and Technology (ICSET), Bandung, 1-6. IEEE.

Sánchez-Alonso, R., González-Barbosa , J., Castillo-Castaneda, E., Gallardo-Alvarado, J., 2015. Kinematic analysis of a novel 2(3-RUS) parallel manipulator, Robotica (First View Paper).

Sánchez-Alonso, R., González-Barbosa , J., Castillo-Castaneda, E., García-Murillo, M., 2016. Análisis Cinemático de un Novedoso Robot Paralelo Reconfigurable, Revista Iberoamericana de Automática e Informática Industrial, 13 (2) 247-257.

Schäfer, A., Holz, J., Leonhardt, T., Schroeder, U., Brauner, P., Ziefle, M., 2013. From boring to scoring-a collaborative serious game for learning and practicing mathematical logic for computer science education. Computer Science Education, 23 (2), 87-111.

Torres, F., Candelas, F., Puente, S., Pomares, J., Gil, P., Ortiz, F., 2006. Experiences with Virtual Environment and Remote Laboratory for Teaching and Learning Robotics at the University of Alicante, International Journal of Engineering Education, 22 (4), 766-776.

Zyda, M., 2005. From visual simulation to virtual reality to games. Computer, 38 (9), 25-32.