Modelar o programar en prácticas de robótica

Antonio J. Muñoz-Ramírez, J. Manuel Gómez-de-Gabriel
Dto. Ingeniería de Sistemas y Automática
Universidad de Málaga
aj@uma.es

Resumen

En este trabajo se muestra la experiencia de usar una herramienta de desarrollo de ingeniería basada en modelos (MDE) frente a otra herramienta tradicional de programación de sistemas embebidos en lenguaje C, para la realización de clases prácticas de robótica. Se ha planteado una práctica basada en el control cinemático de un robot móvil en ambos entornos con similares bloques y funciones de partida. Los resultados muestran tanto la comparación de las evaluaciones objetivas realizadas a los dos grupos como los datos relativos a los tiempos requeridos para la realización de las diferentes partes de la práctica. Si bien, los resultados del aprendizaje son mayores en el caso del método basado en programación, la diferencia en tiempos invertidos y otras valoraciones hacen más adecuado el MDE.

Palabras clave: Educación, Robótica, Ingeniería basada en Modelos, Simulink.

1. INTRODUCCIÓN

Las prácticas docentes en asignaturas de robótica requieren de la implementación de sistemas informáticos de control sobre una planta física experimental. Los enfoques tradicionales de implementación de sistemas embebidos requieren del uso de un lenguaje de programación y de un entorno de desarrollo con unas bibliotecas adecuadas para codificar los modelos de control obtenidos en la fase de diseño. Sin embargo, las asignaturas relacionadas con la robótica no incluyen en su contenido la enseñanza de un lenguaje de programación y las sesiones de prácticas se encuentran cada día más restringidas y asociadas a un contenido teórico oficial que deja poco margen a unificar conocimientos prácticos de programación entre los alumnos. El tiempo invertido por el profesor en la realización de una práctica típica de robótica real se compone de una fase de diseño, en base a la teoría de la asignatura, de otra fase de implementación en la cual se codifican los métodos y algoritmos diseñados, y finalmente una fase de depuración y experimentación. Con el enfoque basado en programación, gran parte del tiempo del alumno se invierte en codificar los modelos diseñados. La calidad de los resultados depende por tanto de la calidad de la programación. Asimismo, el profesor invierte un tiempo en depurar los programas y encontrar los problemas de la implementación [1].

La Ingeniería basada en modelos (Model-Driven Engineering) ó MDE, en contraposición a la programación, tiene como misión permitir a los ingenieros desarrollar y analizar un sistema mediante abstracciones y formalismos más adecuados y parecidos a sus modelos mentales que los lenguajes de programación convencionales ([2], [3]). La principal característica a valorar de este enfoque es que se prescinde de la fase de programación, por lo que se puede pasar del diseño a la experimentación de manera automática permitiendo dedicar más tiempo a los objetivos del aprendizaje. Existen trabajos en los que se usa *Simulink* para MDE en robótica [4, 5], los haptics [6] y también, como en este caso a la educación [7], si bien no son muchos los estudios comparativos de herramientas como [8] o [9] y menos sobre la eficiencia de ambos enfoques en la educación superior.

Otro objetivo de este trabajo es el de comparar la eficiencia de la utilización de herramientas de desarrollo MDE (con generación automática de código, como *Simulink*) respecto a las basadas en programación de sistemas embebidos (como el lenguaje C del entorno *Arduino*) para la realización de prácticas docentes de control de robots.

El presente artículo se encuentra estructurado de la siguiente manera: en la siguiente sección se describe el contexto de los experimentos, describiendo brevemente el contexto docente y la plataforma robótica educativa utilizada. En la sección 3 se describe la práctica comparativa y los puntos de partida de ambos experimentos. En la sección 4 se muestran los resultados de las evaluaciones objetivas y datos de los tiempos dedicados a cada práctica. Finalmente, la sección 5 analiza los resultados y establece las conclusiones.

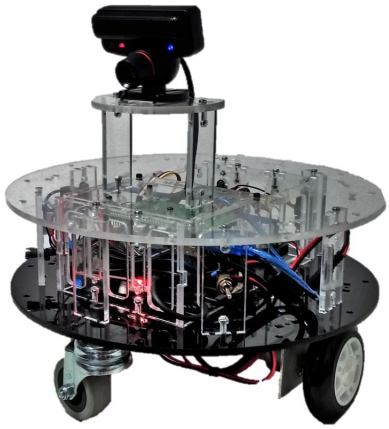


Figura 1: Plataforma educativa de Robótica Móvil PIERO 3 con el módulo de visión integrado

2. CONTEXTO

Este estudio es una continuación de los trabajos desarrollados en un proyecto de innovación educativa de la Universidad de Málaga con código PIE13-185, durante el cual se desarrolló y evaluó el uso de una plataforma robótica didáctica para su uso con herramientas MDE. Los experimentos que se presentan ahora se realizaron durante el curso 2015-16, con alumnos de titulaciones de Grado y Máster de la Escuela Técnica Superior de Ingeniería Industrial: Asignatura de "Laboratorio de Robótica", del grado de Electrónica Robótica y Mecatrónica (ERM) y la asignatura de "Modelado de Sistemas Mecatrónicos y Robots" del Máster de Ingeniería Mecatrónica. Asimismo, se han realizado seminarios ofertados públicamente a alumnos tanto de la Escuela Técnica de Ingeniería Industrial como de la Escuela Politécnica Superior, interesados en realizar prácticas con robots móviles fuera del programa oficial.

Las prácticas se han realizado utilizando las plataformas robóticas basadas en hardware de bajo coste de código abierto (*Arduino*) que pueden ser programadas utilizando tanto lenguaje C (con ligeras simplificaciones) como modelos de *Simulink*. Contamos con un conjunto de doce robots con lo que los grupos de prácticas son de uno o dos estudiantes por robot, según el número de alumnos en cada asignatura. Las plataformas robóticas PIERO ([10], [1]) consisten en un robot móvil con tracción diferencial (Véase Figura 1) y basadas en un popular microcontrolador que han demostrado una gran robustez y versatilidad para el empleo en diferentes asignaturas de robótica, control y mecatrónica.

Entre las características se encuentran su sistema modular para reconfigurar su sistema sensorial, un sistema de tracción diferencial fácilmente observable por el alumno, y un sistema de electrónico de control centralizado también visible. En la Figura 2(a) se pueden apreciar los siguientes componentes principales de su configuración más básica: dos servomotores motores provistos de codificadores angulares incrementales ópticos, un circuito de potencia basado en un puente-H, baterías de ión de litio y un *Arduino Mega* con una placa de conexiones (shield) para facilitar el cableado. Para facilitar su uso por parte de los alumnos, se han realizado modelos 3D para visualización y simulación dinámica (véase Figura 2(c)) que permiten al alumno realizar prácticas en casa sin necesidad de disponer de la plataforma.

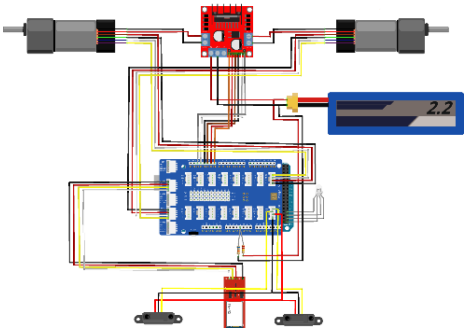
3. EXPERIMENTOS

Para comparar ambos métodos se ha realizado la misma práctica con los dos entornos: *Simulink* con generación de código de *Arduino* y programación en lenguaje C con el IDE original de *Arduino*. se han medido los resultados del aprendizaje en ambos grupos mediante un cuestionario y se han ajustado los tiempos invertidos en la realización de cada fase de la práctica, que incluye el planteamiento de la base teórica.

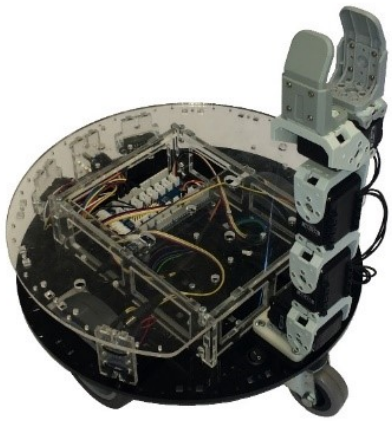
El objetivo docente de la practica consiste en controlar la trayectoria del robot mediante una velocidad lineal y angular local cartesiana fija , para lo cual hay que realizar transformaciones de coordenadas mediante la matriz Jacobiana directa e inversa, y finalmente obtener la localización cartesiana en coordenadas globales del robot mediante odometría (lo cual requiere de integración numérica y de control de los tiempos de ejecución del bucle). La comprobación de la calidad de la estimación de la posición del vehículo se verifica experimentalmente haciendo detener el vehículo cuando haya realizado un giro de 360 grados.

Para desarrollar las prácticas tanto en lenguaje C como en *Simulink* con igualdad de condiciones, se dispone de un conjunto de bloques básicos y funciones que realizan las funciones de control de bajo nivel de control del robot (Véanse Figura 3 y Figura 4). En concreto estos bloques y funciones realizan internamente el control PI de la velocidad de las ruedas ajustado previamente pero no se muestran en las figuras por simplicidad. En ambos casos se proporciona a los alumnos estos programas como plantilla sobre los que pueden trabajar ampliándolo y modificándolo.

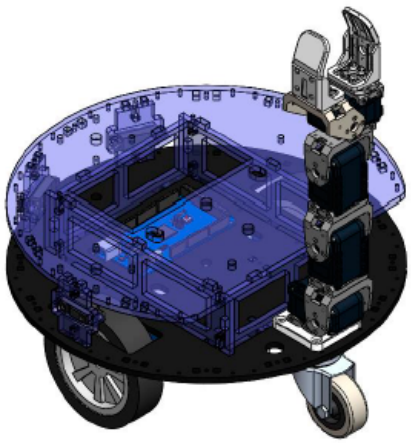
En ambas sesiones de prácticas se explican de manera teórica los conceptos de los diferentes espacios de coordenadas y las transformaciones entre ellos. Tras la parte de teoría, el alumno desarrolla el sistema de control para mover la plataforma con



(a) Esquema de control electrónico electrónicos



(b) Robot PIERO con manipulador



(c) Modelo 3D de PIERO en *Simscape*

Figura 2: Robot PIERO-2 mostrando su sistema electrónico de control, el brazo opcional montado y su modelo 3D para simulación gráfica y dinámica.

Figura 3: Bloques básicos para el desarrollo de las prácticas con *simulink*. Internamente implementa el control PI de la velocidad de las ruedas. Esta plantilla mueve las dos ruedas del robot con velocidades lineales de $0{,}4m/s$

```
//////////////////////////////////////////////////////////////
// Aquí empieza el código de aplicación
//////////////////////////////////////////////////////////////

void setup()
{
  Piero_Init();
}


float velocidad_actual[2];
float velocidad_deseada[2] = {0.3, -0.3};

void loop()
{
  Piero_SpeedControl(velocidad_deseada,velocidad_actual);
  delay(10);
}
```

Figura 4: Programa básico en lenguaje C para el desarrollo de las prácticas con *Arduino*. Nótese que se omite de la figura la implementación de las funciones que realizan el control realimentado de la velocidad lineal en la superficie de las ruedas. En este ejemplo la ruedas izquierda y derecha se moverían con velocidades lineales locales de $0{,}3m/s$ en direcciones opuestas.

una velocidad cartesiana determinadas y obtener las posiciones del vehículo. La complejidad del trabajo del alumno no es muy diferente en ambos casos. En la Figura 5 se muestra el modelo resultante realizado en *Simulink*, y en la Figura 6 se muestra el resultado esperado en la implementación mediante programación en lenguaje C. Ambos enfoques deben producir los mismos resultados, si bien en el caso de la programación en lenguaje C será necesario introducir elementos adicionales, que sin embargo son implícitos en los modelos de *Simulink*, como los relativos a la temporización del bucle de control y los de la integración numérica.

Las prácticas de comparación se han efectuado mediante grupos de alumnos de diversas asignaturas de los centros de la Escuela de Ingenierías, evaluándose mediante cuestionarios tras la realización de la práctica. Los cuestionarios están compuestos por las siguientes secciones:

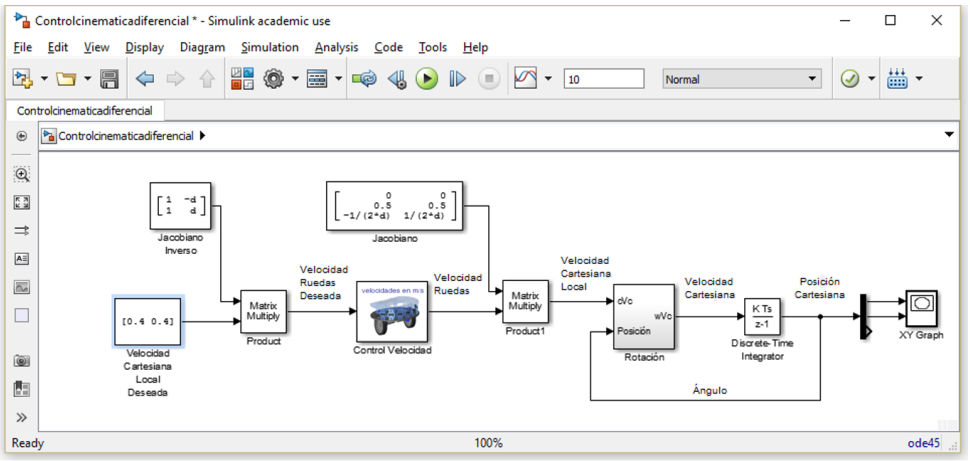


Figura 5: Aspecto final de la práctica realizada en el entorno de modelado *Simulink*, donde se muestra la claridad del diseño del sistema de control que permite una fácil revisión y donde no se observan elementos relativos a la implementación.

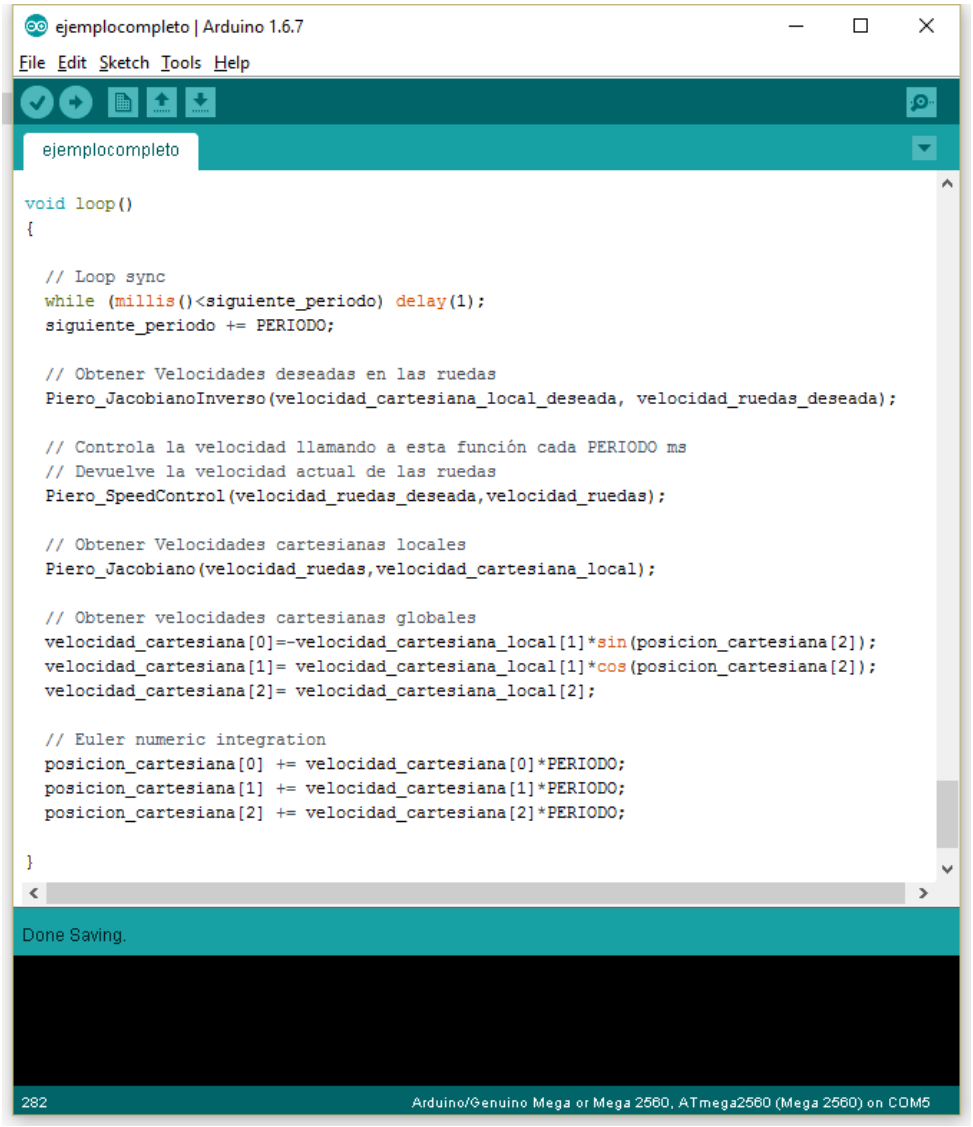


Figura 6: Aspecto final de la práctica realizada en el lenguaje de programación C, donde se observa que el código refleja el sistema de control de una manera menos clara y se incluyen aspectos relativos a la temporización y a la integración numérica.

- Información de identificación anónima del alumno (Titulación de origen, equipo utilizado, sesión, ...)
- Preguntas experiencia previa en el entorno de desarrollo en cada caso (*Arduino* o *Simulink*).
- Evaluación objetiva sobre conceptos de robótica presentados en la teoría.
- Evaluación objetiva sobre conceptos de mecatrónica y *Arduino* (No explicados en la teoría, pero usados indirectamente en las prácticas).
- Cuestionario de satisfacción sobre la práctica.

Las tres últimas secciones se han promediado y puntuado entre 0 y 10. El incremento esperado del aprendizaje debería producirse exclusivamente en la sección de Robótica, mientras que los conocimientos sobre la plataforma (que etiquetaremos como *Arduino*) deben permanecer constantes ya que no han necesitado aprenderlos y los profesores no se los han enseñado.

4. RESULTADOS

Para la comparación de los dos enfoques de desarrollo se han realizado dos sesiones piloto de prácticas con 10 y 8 alumnos respectivamente. Tras la realización de cada práctica se han realizado los correspondientes cuestionarios y se han observado los tiempos requeridos para la realización de las diferentes etapas. Los datos relativos a la experiencia previa en estos casos han sido prácticamente cero tanto en el caso de *Simulink* como de *Arduino*.

4.1. Calificaciones

Aunque los resultados subjetivos por parte de los profesores son claramente positivos en términos de ventajas aportadas y eficiencia del uso del tiempo del alumno y del profesor, es necesario examinar los datos de la evaluación objetiva realizada mediante los cuestionarios.

La Figura 7 muestra las calificaciones medias obtenidas en las tres últimas secciones de los cuestionarios para los dos tipos de prácticas: Aprendizaje sobre conceptos de robótica, aprendizaje de conceptos sobre mecatrónica y *Arduino*, y de la evaluación subjetiva consistente en la encuesta de satisfacción. Los resultados muestran una mejor calificación en todos los aspectos en el grupo de alumnos que realizaron la práctica mediante programación en lenguaje C, aunque con resultados muy similares, si bien el reducido tamaño de la muestra hace necesario seguir realizando experimentos durante el siguiente curso académico para tener datos más representativos.

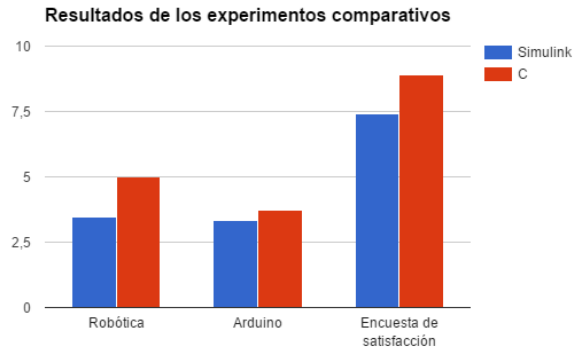


Figura 7: calificaciones medias obtenidas al final de las prácticas y encuesta de satisfacción con programación en C como en *Simulink*.

Tabla 1: Actividades desarrolladas en la sesión práctica

Actividades	**Simulink**	**Prog. C**
Teoría	20 min.	20 min.
Control articular	15 min.	20 min.
Control Cartesiano	15 min.	20 min.
Odometría	15 min.	40 min.
Revisión	5 min.	10 min.
Tiempo total	70 min.	110 min.

4.2. Tiempos

Se ha planificado el siguiente cronograma de desarrollo de la práctica destinada a comparar ambos métodos. Si bien, inicialmente, los tiempos previstos han sido iguales para los los dos, los tiempos finales, modificados para poder impartir la práctica satisfactoriamente, en función de los tiempos invertidos por los alumnos, han quedado según la tabla 1.

Si bien el número de muestras obtenidas no es aún significativo, se observa un mayor incremento en alumnos con un menor nivel de experiencia previa obtenido de la primera parte de cuestionario, en ambos grupos de conocimientos.

5. CONCLUSIONES

Una comparación cuantitativa sobre los tiempos de diseño y resolución de ambas versiones de la prácticas por parte del profesorado es dificil de estimar dado que se basa en años de experiencia en ambas metodologías si bien una percepción cualitativa daría ganadora a la práctica en *Simulink* por el empleo de elementos más generales y fáciles

de depurar.

En la experiencia por parte del profesor a hora de impartir las prácticas la comparación entre ambas mostró una clara diferencia objetiva a favor de la versión en *Simulink*: las prácticas en C requirieron en la mayoría de los casos de un apoyo individualizado casi constante hasta la resolución de la misma tanto en la programación como en la resolución de errores, mientras que en la versión *Simulink* el apoyo fue tan solo puntual en la resolución de errores y estos a su vez fueron fácilmente detectados por por parte del profesor muy al contrario que en C.

Las gráficas de aprendizaje denotan un sesgo a favor de la práctica en C, que puede deberse fundamentalmente a que el lenguaje C es el único lenguaje de programación estudiado oficialmente, en tan sólo una asignatura de la titulación, siendo por tanto la programación con el IDE de *Arduino* y su filosofía textual y secuencial muy familiar para el alumno. La programación en *Simulink*, su entorno y su enfoque de alto nivel basado en modelado se contrapone a su concepción de la programación y dificultaría la asimilación de los conceptos de la práctica. Es decir, la práctica en *Simulink* presenta una curva de aprendizaje superior al IDE de *Arduino* más parecido al entorno de programación clásico.

Sin embargo hay que hacer mención de la facilidad de depuración de los modelos de *Simulink*, que permite la detección rápida de errores de diseño, incluso sin la presencia del robot real gracias a su capacidad de simulación y gráfica. Elementos, estos, de los que carece la programación en C.

Asimismo, si se tienen en cuenta los tiempos dedicados a la realización de los dos tipos de prácticas, a pesar de tener un menor aprendizaje medio, la herramienta MDE proporciona una mayor eficiencia.

Dado los beneficios ampliamente reconocidos por la industria de la MBD como en [11] creemos que es necesario su inclusión en el curriculum del estudiante de ingeniería, en cuyo caso la práctica en *Simulink* saldría vencedora desde todos los puntos de vista.

Agradecimientos

Este trabajo han sido parcialmente financiado por el Proyecto de Innovación Educativa de la Universidad de Málaga PIE15-180. Los equipos y componentes utilizados en proyecto han sido patrocinados por Ingeniería UNO (http://www.ingenieriauno.com/) y Seeed Studio (http://seeedstudio.com/).

Referencias

[1] Muñoz-Ramirez, A.J., Gomez-De-Gabriel, J.M. y Fernandez-Lozano, J.J,(2015) Ingeniería Basada en Modelos en Prácticas de Robótica. Jornadas CEA de Automática 2015, Bilbao, España.

[2] Schmidt, D.C. (2006), Guest editor's introduction: Model-driven engineering. Computer, 39(2):0025–31, 2006.

[3] Harel, D. (1987), Statecharts: a visual formalism for complex systems. Science of Computer Programming, 8(3):231 – 274, 1987. ISSN 0167-6423.

[4] Barber, R., Crespo, M. H. J. (2013). Control Practices using Simulink with Arduino as Low Cost Hardware. In The 10th IFAC Symposium on Advances in Control Education.

[5] Gartseev, I. B., Lee, L. F., Krovi, V. N. (2011). A low-cost real-time mobile robot platform (ArEduBot) to support project-based learning in robotics & mechatronics. In Proceedings of 2nd International Conference on Robotics in Education (RiE 2011).

[6] Beni, N.; Grottoli, M.; Ferrise, F.; Bordegoni, M., (2014) Rapid prototyping of low cost 1 DOF haptic interfaces, Haptics Symposium (HAPTICS), 2014 IEEE , vol., no., pp.479,483, 23-26 Feb. 2014

[7] Sobota, J., Balda, P., Schlegel, M. (2013). Raspberry Pi and Arduino boards in control education. In Advances in Control Education (Vol. 10, No. 1, pp. 7-12).

[8] Kehtarnavaz, N. and Gope,C. (2006), DSP System Design Using Labview and Simulink: A Comparative Evaluation,"2006 IEEE International Conference on Acoustics Speech and Signal Processing Proceedings, Toulouse, 2006, pp. II-II.

[9] Wetter, M. and Haugstetter, C (2006), Modelica versus Trnsys - A comparison between and equation-base and a procedural modeling language for building energy simulation. Simbuild 2006, Second National IBPS-USA Conference, Cambridge, USA.

[10] Gil-Lozano, J.E. Muñoz-Ramírez, A.J., Torres Lopez, V.E. y Gómez-de-Gabriel, (2014), Uso de Simulink y Arduino para Prácticas de Robótica. Jornadas de Automática 2014, Valencia, España.

[11] Broy, M., Kirstan, S., Krcmar, H., Schätz, B.,(2011). What is the Benefit of a Model-Based Design of Embedded Software Systems in the Car Industry?. Jörg Rech, Christian Bunse (eds.): Emerging Technologies for the Evolution and Maintenance of Software Models. IGI Global.