e-Book

Actas

Actas II Jornadas Argentinas de Didáctica de la Programación

Editores:

Araceli Acosta
Belén Bonello
Cecilia Martínez
Sonia Permigiani
Nicolás Wolovick

UNC FAMAF ffyh UNRC

ACTAS

II JORNADAS ARGENTINAS DE DIDÁCTICA DE LA PROGRAMACIÓN

Actas II Jornadas Argentinas de Didáctica de la Programación / Alejandro Iglesias... [et al.] ; editado por Araceli Acosta... [et al.].- 1a ed.- Córdoba : Universidad Nacional de Córdoba. Facultad de Filosofía y Humanidades, 2020.
Libro digital, PDF

Archivo Digital: descarga y online
ISBN 978-950-33-1600-9

1. Didáctica. 2. Lenguaje de Programación. 3. Formación Docente. I. Iglesias, Alejandro. II. Acosta, Araceli, ed.
CDD 004.071

COMITÉ ACADÉMICO

EDITORES

ILUSTRACIÓN DE TAPA

Manuel Coll – Área de Comunicación Institucional – FFyH – UNC

Robótica educativa con software libre y hardware de especificaciones abiertas para enseñanza de programación

Diego Letzen, Valentín Basel, Alba Massolo, Federico Ferrero

Resumen: El presente trabajo intenta dar cuenta de parte del trabajo desarrollado por los autores en la elaboración de una propuesta para la enseñanza de la programación en escuelas secundarias de la provincia de Córdoba. En el transcurso del proyecto de investigación, se evaluaron algunos de los paquetes de robótica educativa disponibles en Argentina a la fecha y se discutieron y diseñaron una serie de robots educativos basados en el hardware de desarrollo ICARO (placa NP07 con microcontroladores PIC 18F4550) diseñados con la finalidad de proponer intervenciones para la enseñanza de programación.

Palabras claves: Robótica educativa, software libre, hardware libre.

Introducción

A partir de las discusiones llevadas a cabo en el grupo de investigación se decidió seleccionar, para el desarrollo del robot, diferentes categorías de análisis que permitan dar cuenta de las distintas alternativas de desarrollo que actualmente se consiguen en el mercado, teniendo en cuenta las discusiones hechas sobre el uso de recursos educativos abiertos (Z. González & HernándezII, s. f.), software libre, hardware de especificaciones libres, soberanía tecnológica (Haché, 2014) y didáctica de la programación. Teniendo en cuenta esas dimensiones de análisis, se eligieron las siguientes categorías para seleccionar el software y hardware que usaría el equipo de investigación:

• Que sea desarrollado con software libre.
• Que las especificaciones del diseño de hardware sean libres.
• Que pueda ser fabricado de forma hogareña (DiY).
• Bajo costo de adquisición / fabricación.
• Buena documentación.
• Multiplataforma.
• Que tanto el hardware como el software permitan proyectar en complejidad de trabajo (sin encerrar en una "caja negra" partes del software y el hardware).
• La cantidad de propuestas educativas que pueden ser vinculadas al paquete.
• Es un proyecto recomendado para trabajar en el aula, en grupos.
• Es un proyecto recomendado para trabajar de forma individual.
• Los conocimientos necesarios para poder implementar su uso.
• Las comunidades de desarrollo de hardware y software que puedan dar soporte al proyecto.

Seleccionadas las categorías de análisis, se procedió a catalogar distintos proyectos comerciales y comunitarios que podrían servir para el desarrollo de la investigación. La necesidad de plantear un proyecto que se posicione desde una perspectiva de soberanía tecnológica, llevó a tomar como criterio principal el uso de software y hardware libre.

Software libre y hardware de especificaciones libres

El software libre es un movimiento que comenzó en el año 1983 cuando Richard Stallman anunció el proyecto GNU (Stallman, 2007) en contraposición a la aparición de monopolios artificiales en el desarrollo de software (Busaniche y otros, 2007). Se podría decir que la meta del movimiento fue dar libertad a los usuarios de programas de computadoras reemplazando el software con términos de licencias restrictivas (software privativo) por una alternativa libre que permitiera a los usuarios y desarrolladores de software contar con al menos cuatro libertades con respecto al uso del mismo, a saber:

• Libertad 0: la libertad de usar el programa, con cualquier propósito (uso).
• Libertad 1: la libertad de estudiar cómo funciona el programa y modificarlo, adaptándolo a las propias necesidades (estudio).

• Libertad 2: la libertad de distribuir copias del programa, con lo cual se puede ayudar a otros usuarios (distribución).
• Libertad 3: la libertad de mejorar el programa y hacer públicas esas mejoras a los demás, de modo que toda la comunidad se beneficie (mejora).

Por su parte, el hardware de especificaciones abiertas busca llevar el concepto del software libre (la libertad de usar, estudiar, distribuir o mejorar el software) al diseño de componentes físicos (Olivencia & Martínez, 2015), especificando una licencia que permite distribuir planos y código fuente de desarrollos de PCBs (Printed Circuit Board, por sus siglas en inglés) y hardware electrónico. Asimismo, se considera que un diseño de circuito (esquemático, diseño de PCB y archivos GERBER) debe ser desarrollado con software libre y usando formatos abiertos.

Se tomó una clasificación del hardware en función de su diseño y el software empleado para su creación (González, González, & Gómez-Arribas, 2003), definiendo una clasificación primaria de los tres tipos de archivos necesarios para la fabricación de un PCB, el esquemático, el archivo de pcb y el archivo GERBER. En función de esa clasificación, se puede separar al hardware en tres tipos:

• (P): Software de diseño propietario.
• (L): Software de diseño libre.
• (M): Software de diseño propietario pero multiplataforma (funciona en sistemas operativos libres y también en propietarios).

Si tenemos en cuenta los 3 planos electrónicos necesarios para fabricar un PCB podemos ver que necesitamos tener el archivo esquemático, el archivo de PCB y el fichero de fabricación. Por tanto, los tres archivos de construcción para el circuito electrónico pueden ser de clasificación como enteramente libres (tipo LLL) o completamente propietarios (PPX), y todas sus combinaciones posibles. Además, desde la perspectiva de la soberanía tecnológica (Haché, 2014), se decidió que la tecnología elegida para la construcción del hardware robot, fuera de fácil acceso, bajo costo y con materiales sencillos de trabajar sin necesidades de herramientas especiales.

Clasificando hardware

Para la elección del hardware de control se analizaron las dimensiones propuestas y sobre eso se armó una tabla de doble entrada que pusieran en comparación las distintas soluciones propuestas por el grupo.

Para clasificar el hardware que se podía obtener para la investigación se utilizó una métrica de 3 valores que permitieron seleccionar entre:

• Valor 0: No aplica al esquema de clasificación ponderado.

• Valor 1: Algunas partes del proyecto pueden ser consideradas en la categoría seleccionada.
• Valor 2: Aplica completamente a la categoría seleccionada.

	Lego mindstorms ev3	Robotgroup robot N6	Kit Arduino Electr. básica	Proyecto ICARO	Makeblock ultimate kit	Mis ladrillos probots R502
desarrollado con software libre.	0	1	2	2	1	0
Diseñado con Hardware libre	0	1	2	2	1	0
Capacidad Do It Yourself (DIY)	0	0	1	2	0	0
Bajo costo de adquisición / fabricación.	0	1	2	2	0	1
Documentación	2	1	2	1	1	1
Multiplataforma	0	0	2	2	1	0
Proyección de complejidad de trabajo (eliminar “cajas negras”)	1	1	2	2	1	0
propuestas educativas	2	1	2	1	1	1
Bajo Conocimientos necesarios para poder implementar su uso	2	1	1	1	1	2
comunidades de desarrollo de hardware y software	1	0	2	2	0	0
TOTAL	8	7	18	17	7	5

Tabla comparativa.

La confección de una tabla de doble entrada demostró rápidamente que los 2 únicos proyectos que cumplimentaban con la mayor cantidad de requerimientos eran el uso de robots basados en ARDUINO o el uso de hardware del proyecto ICARO (ver tabla comparativa).

Finalmente, se decidió trabajar usando el hardware y software del proyecto ICARO, dado que siendo la evaluación muy similar, en el caso de ICARO se tenía acceso a la comunidad de desarrollo del software, y la posibilidad de adecuar la electrónica y el software de control en función de las necesidades del grupo.

Tribot

En principio se optó por un esquema de robot tipo TRIBOT, con dos motores de corriente continua, 6 pilas AA, un sensor de ultrasonido (hc-sr04), un sensor infrarrojo (CNY70) y dos micro switch del tipo “final de carrera” como sensores digitales de contacto, como se puede ver en la figura 1. Para su programación se utilizó el software ICARO, creando el firmware de trabajo usando lenguaje ANSI C, con el compilador libre SDCC (Versión 3.7.0) y las bibliotecas del proyecto PINGUINO (bootloader v4).

Este robot TRIBOT (Zabala, 2007) se programa y carga su firmware mediante cable USB, siendo una mecánica de trabajo muy parecida a los esquemas propuestos por otros kits comerciales o diseñados por comunidades.

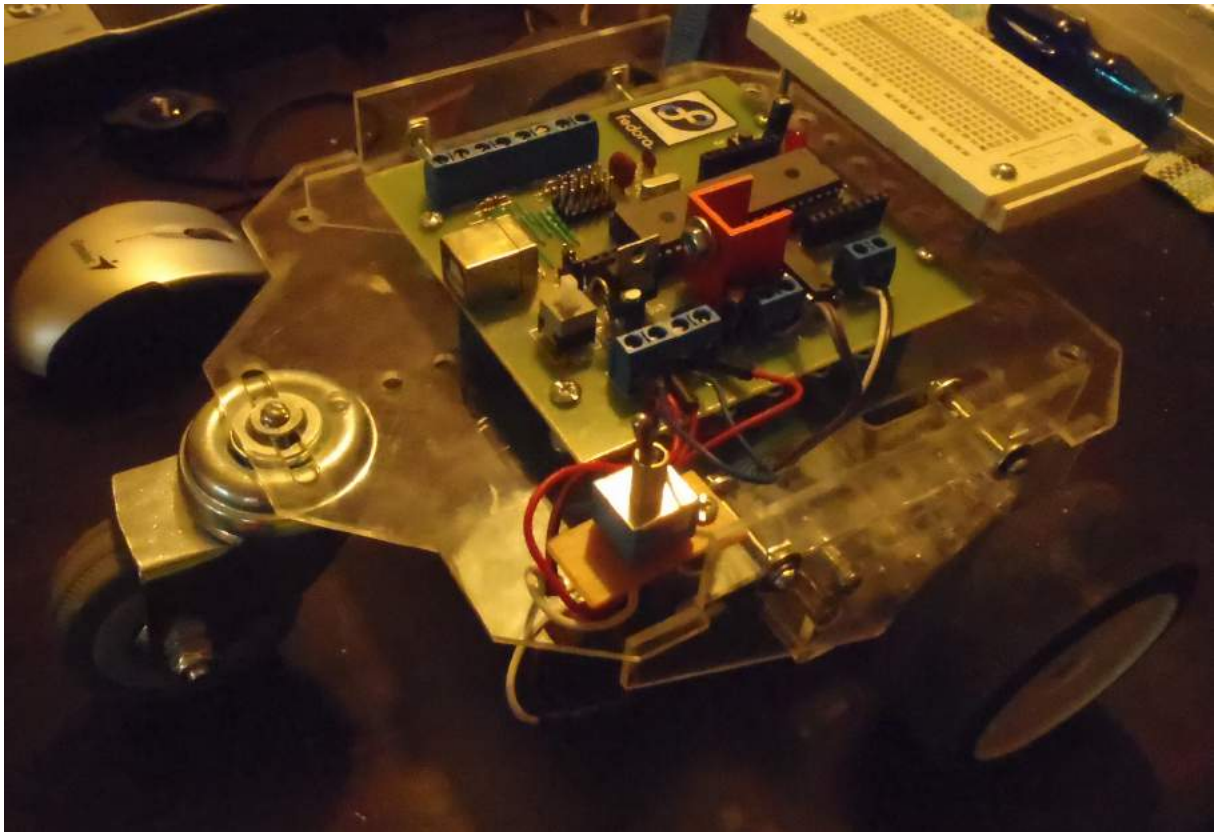

Figura 1. Prototipo de robot tipo TRIBOT.

La decisión de usar microcontroladores PICs en vez de hardware basado en ARDUINO (ATmega328), se debió principalmente a la ventaja de poder contar con un microcontrolador en formato THT (Through-Hole Technology) que tuviera una interface USB 2.0 por hardware y que fuera fácil de adquirir en pequeñas cantidades para prototipado. El formato THT permite trabajar con integrados grandes que se pueden manipular y soldar con un soldador de estaño hogareño y sin herramientas especiales. El prototipo funcionó satisfactoriamente pero con algunas limitaciones relacionadas con el uso del espacio en situaciones de aprendizaje grupal formal y la logística implicada en la dependencia de pilas para el funcionamiento. La observación que más peso tuvo a la hora de dejar de lado este tipo de

robots fue el requerimientos que los robots considerados pudieran dibujar con más o menos precisión para poder retomar algunos de los ejercicios propuestos por Papert y el sistema LOGO (Papert, 1980), como piso de la experiencia. El robot TRIBOT se mostró ineficiente en ese cometido, y por lo tanto se tomó la decisión de experimentar con plataformas fijas tipo PLOTTER o robots de tipo SCARA, que fueran lo más baratos y fáciles de fabricar y emplear en un aula de escuela secundaria de una escuela pública argentina.

Robot SCARA

En paralelo a las pruebas que se hicieron sobre el robot tribot, se tomó la decisión de fabricar un robot de tipo SCARA (acrónimo que responde por sus siglas en inglés a *Selective Compliant Assembly Robot Arm*) que permitiría trabajar con una plataforma conectada a la computadora y controlarlo mediante programación en PYTHON.

Los robots de tipo SCARA tienen una cinemática inversa del movimiento sencilla de calcular, y tienen un margen amplio de movimientos, aunque no tienen las capacidades de un brazo robot de 6 grados de libertad (Reyes Cortés, 2011), son más fáciles de fabricar y tienen mejor relación de fuerza. Además, los robots SCARA son generalmente más rápidos y sencillos que los sistemas comparables de robots cartesianos.

Para su fabricación, se eligió continuar con los componentes electrónicos, eléctricos y mecánicos de fácil adquisición que se usaron para el robot tribot, aunque los motores paso a paso 28BYJ-48 se mostraron demasiado lentos y de escasa fuerza como para poder mover los ejes del robot, por lo tanto se eligió trabajar con servomotores de aeromodelismo.

Los servomotores de aeromodelismo, se presentan como una alternativa interesante por su bajo costo, fácil control a través de PWM (siglas en inglés de *pulse-width modulation*), potencia y velocidad, además de poseer toda la electrónica de control dentro del mismo servomotor, lo que reduce mucho el desarrollo de electrónica de control.

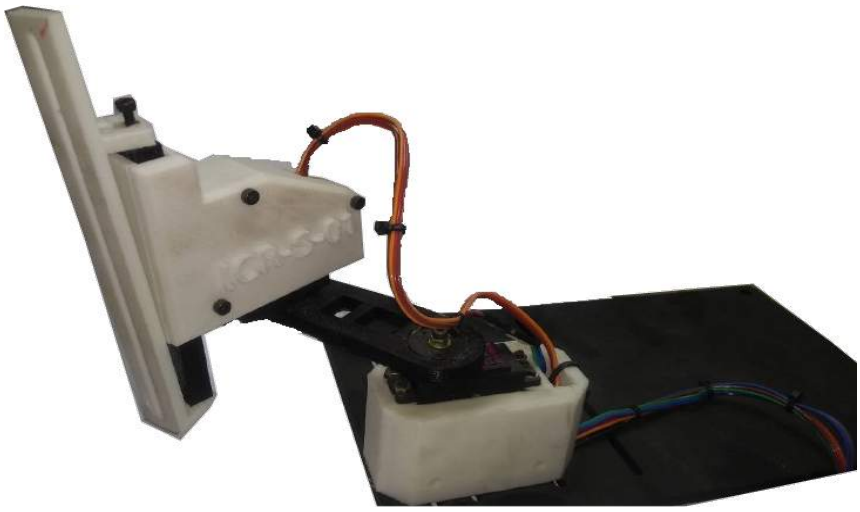

Figura 2. Robot tipo SCARA.

El robot SCARA mínimo y funcional necesita tener 3 servomotores, generalmente dos servos “grandes” (con un torque de 3.7 kg-cm) que controlan el “hombro” y “codo” del brazo robot, y un micro servo 9g para el actuador.

En las pruebas realizadas, los servomotores resultaron tener poca precisión como para dibujar sobre una superficie, aunque era muy veloces y con bastante potencia como para mover piezas pequeñas y trabajar con un sistema más parecido al uso industrial que generalmente se da con estas plataformas, por lo que se decidió considerar una nueva alternativa para solucionar estos inconvenientes.

V-plotter

Si bien los robots SCARA pueden hacer trazos, al usar servo motores de poca precisión, no se obtuvieron resultados que justificaran su uso para hacer dibujos, y por lo tanto se decidió desarrollar una plataforma basado en un sistema V-plotter.

Estos son uno de los modelos más sencillos de plotters para fabricar, en los que el cabezal de dibujo cuelga atado a dos cables que se enrollan en dos poleas a cada lado de la superficie de trabajo. Estas poleas son controladas por motores “paso a paso” que enrollando y desenrollando su cable, mueven el cabezal en los ejes X/Y.

Este modelo de plataforma robot se presenta como una alternativa fácil de fabricar respecto a plotters de tipo cartesiano (impresoras 3d por ejemplo), o robots de movimientos polares tipo SCARA, dado que solo tienen un cabezal colgando de dos poleas, sobre una plataforma vertical.

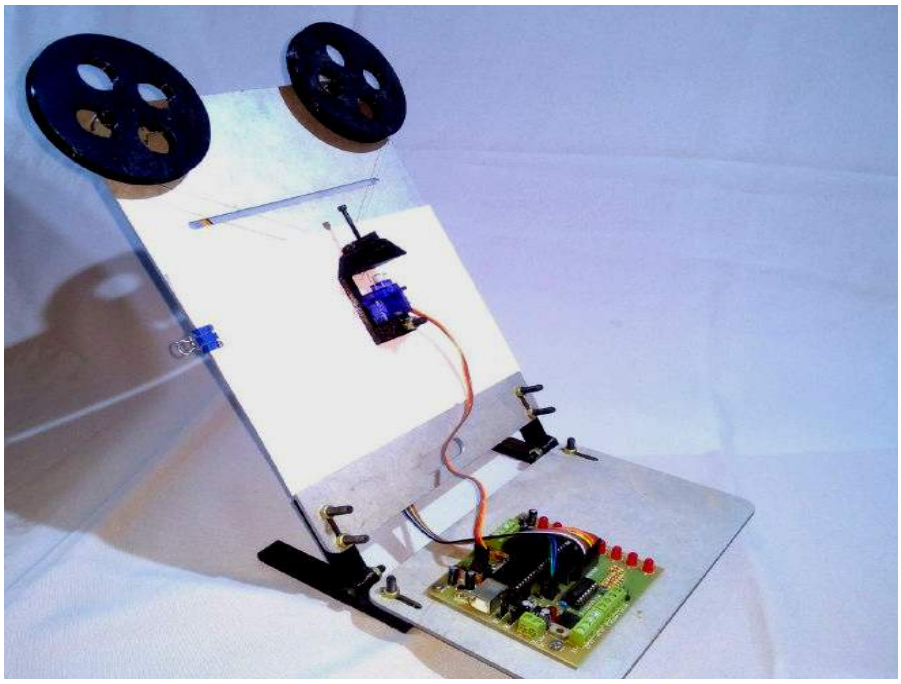

Figura 3. Plataforma Vplotter final.

Para el desarrollo del V-PLOTTER se decidió trabajar con 2 motores PAP (28BYJ-48) de bajo coste y micro servos 9g en el cabezal para poder levantar el lápiz y mover el cabezal sin dibujar. La particularidad de los motores 28BYJ-48 es su bajo torque y escasa velocidad de movimientos, sin embargo, al ser motores de 4 pasos (step) u 8 medio pasos (half steps) con caja reductora 1/64, dan como resultado que para hacer una vuelta entera, el motor debe recibir 512 impulsos (cada "paso" del motor equivale a 0,703125 grados) dando un buen margen de precisión de movimiento para la plataforma robot. Para el control de la plataforma se optó por trabajar con bibliotecas escritas en Python 2.7 que permitió el desarrollo de una API (*Application programming interface*) que implementa los algoritmos necesarios para poder controlar el movimiento de de los motores, y pudiera servir como un paralelismo entre el lenguaje LOGO y sus primitivas (adelante, atrás, izquierda, derecha), para permitir trabajar desde Python pero con instrucciones parecidas a LOGO.

Mecánica

Al ser una plataforma sencilla, para su diseño se decidió poder contar con piezas que pudieran ser fabricadas en pequeña escala, aprovechando las herramientas comunes en el movimiento "*makers*" como son las impresoras 3d y máquinas de corte láser. Para el desarrollo de las piezas 3d se utilizó software libre (FreeCad) y una impresora de construcción casera modelo PRUSA I3 que permitió ir probando distintas alternativas de piezas que podían usarse en la plataforma V-plotter. Para ahorrar costes y tiempo de impresión, se decidió que las piezas más grandes podían ser fabricadas usando madera MDF (*medium density fiberboard*) y cortadas con máquinas de corte láser.

API (*Application programming interface*)

Para el cálculo de posición de la plataforma se toma el teorema del COSENO como herramienta matemática para poder obtener los pares ordenados que representan la posición X,Y relativa al punto (0,0) del cabezal. El esquema de trabajo de la API es el siguiente: recibe un vector de movimiento y un ángulo (en el esquema de trabajo LOGO), convierte esos vectores en pares ordenados x-y, para luego mediante un algoritmo de bresenham calcular el trazado de la línea de dibujo que tiene que seguir el cabezal. Luego decodifica la cantidad de pasos que debe dar cada motor que mueve las poleas y envía esa información en formato byte a la placa ICARO, que tiene conectada los dos motores mediante un integrado UNL2803 controlado por el PORTB del microcontrolador 18F4550. Como cada motor tiene 4 bobinas que se prenden secuencialmente para hacer un "paso", por tanto, para mover los 2 motores se usa un byte entero (8 bits) donde cada motor es controlado por 1 nibble (4 bits) del PORTB. De esta forma cuando la placa pone en estado alto los pines del PORTB, alimenta las correspondientes bobinas de los motores PAP 28BYJ-48 haciendo que este mueva su eje una posición.

Por su naturaleza, los V-plotters tienen un área de dibujo delimitada por las tensiones ejercidas por el peso del cabezal sobre los hilos de donde cuelga, además del diámetro de las poleas que recogen o sueltan cada hilo. A medida que el dibujo se va alejando del centro de la plataforma se puede apreciar una cierta deformación de las líneas rectas creando un

curva de forma ascendente, sin embargo a los efectos pedagógicos de generar dibujos usando lenguaje PYTHON, resultaron satisfactorios, por su bajo costo de fabricación y por cubrir correctamente las expectativas planteadas para la propuesta de enseñanza.

Consideraciones finales

El uso de dispositivos electromecánicos, como plataformas robots (Builes et al., 2011), se presentan como una herramienta muy poderosa para poder enseñar conceptos de programación (uso de repeticiones, variables, estructuras condicionales entre otras) generando situaciones genuinas de programación para los estudiantes. La capacidad de "abrir la caja negra" permite que los estudiantes ahonden en las tecnologías de base (Montfort et al., 2012) que permiten a su vez comprender cabalmente cómo son las interacciones entre hardware y software.

El desarrollo de esta plataforma V-PLOTTER, permitió al grupo de investigación dar cuenta de la necesidad de contar con recursos educativos abiertos (REA/OER) específicos para la didáctica de la programación (Bağcı, Kamaşak, & Ince, 2017), así como la posibilidad de retomar una currícula construccionista (Papert, 1993) trabajando con un lenguaje moderno como PYTHON (González Duque, 2016).

Bibliografía

Bağcı B.B., Kamaşak M. & Ince G. (2018). The Effect of the Programming Interfaces of Robots in Teaching Computer Languages. En Lepuschitz W., Merdan M., Koppensteiner G., Balogh R., Obdržálek D. (Eds) *Robotics in Education. RiE 2017. Advances in Intelligent Systems and Computing, vol 630*. Springer, Cham, pp. 88–99.

Busaniche, B., Chaparro, E., Heinz, F., Ribeiro, S.,Rodríguez Cervantes, S. Fiorito, S. & Westermann, W. (2007). *Monopolios artificiales sobre bienes intangibles*. Córdoba, Ediciones Fundación vía libre.

González Duque, R. (2016). *Python para todos*. Recuperado el 3 de marzo de 2019 de http://www.utic.edu.py/citil/images/Manuales/Python_para_todos.pdf

González, I., González, J., & Gómez-Arribas, F. (2003). *Hardware libre: clasificación y desarrollo de hardware reconfigurable en entornos GNU/Linux*. Recuperado de http://ftp1.nluug.nl/ftp/pub/ftp/os/Linux/doc/LuCaS/Presentaciones/200309hispalinux/8/8.pdf

González, Z., & HernándezII, G. M. (s. f.). Recursos educativos abiertos. Recuperado de http://www.medigraphic.com/pdfs/educacion/cem-2013/cem133p.pdf

Haché, A. (2014). *Soberanía tecnológica*. Dossier sobre Soberanía Tecnológica, pp. 9–18.Recuperado de https://www.plateforme-echange.org/IMG/pdf/dossier-st-cast-2014-06-30.pdf.

Jiménez Builes, J. A., Ramírez Patiño, J. F., & González España, J. J. (2011). Collaborative robotics modular system used in education. *Revista Facultad de Ingeniería Universidad de Antioquia*, (58), 163-172

Montfort, N., Baudoin, P., Bell, J., Bogost, I., Douglass, J., Marino, M. C. & Vawter, N. (2012). *10 PRINT CHR$(205.5+RND(1));? GOTO 10*. Cambridge (MA), The MIT Press. Leiva Olivencia, J. & Moreno Martínez, N. (2015). Recursos y estrategias educativas basadas en el uso de hardware de bajo coste y software libre: una perspectiva pedagógica intercultural. *Revista científica electrónica de Educación y Comunicación en la Sociedad del Conocimiento*, 15(1), pp. 30-50.

Papert, S. (1980). *Mindstorms: Children, Computers, and Powerful Ideas*. New York, Basic Books, Inc., Publishers.

Papert, S. (1993). *The children's Machine: Rethinking School in the Age of the Computer*. New York, BasicBooks, Inc., Publishers.

Reyes Cortés, F. (2011). *Robótica: control de Robots manipuladores*. México, Alfaomega.

Stallman, R. M. (2007). *Software libre para una sociedad libre.* Madrid, Traficantes de Sueños.

Zabala, G. (2007). *Robótica*. Buenos Aires, Gradi.