

# ALCABOT 2002 - ROBOT: AURELIO

Bartolomé Gómez Pérez

[tolygp@teleline.es](mailto:tolygp@teleline.es)

Francisco Javier Martínez García.

[j\\_martinez\\_garcia@hotmail.com](mailto:j_martinez_garcia@hotmail.com)

Andrés González Andino

[andres.gonzalez@attglobal.net](mailto:andres.gonzalez@attglobal.net)

## Resumen

AURELIO es el producto de un año de trabajo dedicado al conocimiento de las técnicas asociadas al mundo de la microbótica. Se ha tratado de depurar los sistemas y la mecánica en el transcurso de las anteriores ediciones de ALCABOT. El equipo está formado por tres estudiantes: dos ingenieros de telecomunicaciones y un ingeniero industrial. Se ha tratado de dotar al microbot de la versatilidad suficiente para ser utilizado en la implementación de diferentes tareas.

## 1. Introducción

El trabajo realizado para el diseño y construcción de AURELIO incluye a muchas y muy diversas fuentes de información, desde fabricantes de automatismos industriales, hasta libros y publicaciones especializadas en el campo de la inteligencia artificial.

AURELIO es un intento por combinar, de forma sencilla y accesible, la mecánica y robótica con los algoritmos de inteligencia artificial utilizados para la resolución de laberintos y la navegación completa por entornos desconocidos.

## 2. Plataforma mecánica usada

La base sobre la que se ha construido todo el robot consiste en una superficie circular de Policarbonato de 140mm a la que se le han extraído dos secciones laterales destinadas a las ruedas.

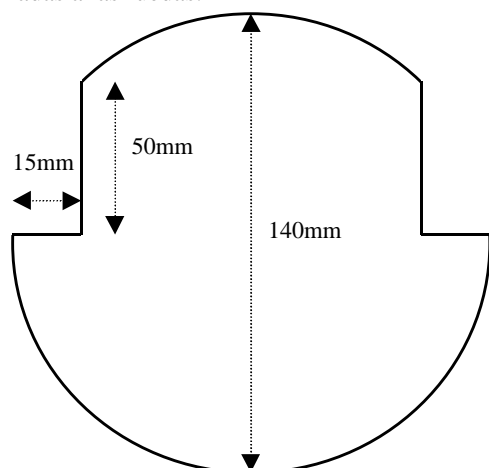


Figura 1. Plataforma utilizada.

La figura 1 muestra una vista cenital de la base utilizada como chasis del robot.

Se eligió el uso de policarbonato pues este material es resistente, a la par que ligero, permite ser cortado y troquelado sin presentar problemas, con lo que resulta ideal para este tipo de aplicaciones.

Los motores utilizados para generar el movimiento del robot se han dispuesto en una configuración de tracción diferencial en triciclo mostrada en la figura 2, que consiste en dos motores independientes formando un eje común. Esta configuración permite al robot avanzar en línea recta, girar sobre si mismo y describir arcos. El punto de apoyo del triciclo consiste en un rodamiento trasero obtenido en una ferretería industrial. Los motores utilizados son servos de aeromodelismo de la marca Futaba [1], mostrados en la figura 3 y a los que se les ha eliminado toda la electrónica interna, así como las limitaciones físicas que solo permiten un giro de 60°, de forma que se ha obtenido unos motores DC de giro continuo con su propia caja reductora [2]. Los motores se alimentan de forma independiente al resto del sistema con dos baterías de 9V. La excitación de los mismos puede realizarse con una tensión continua o bien con una señal modulada en anchura de pulsos o PWM (Pulse Width Modulation).

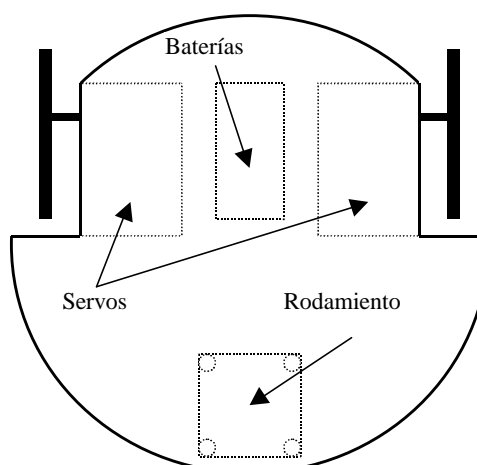


Figura 2. Configuración de los motores.

Una ventaja añadida a la utilización de servos reconvertidos es la cantidad de accesorios que incluyen, y que permiten unir todo tipo de ruedas, poleas y palancas al eje del motor. En el caso de AURELIO las ruedas se han fabricado usando una plancha de aluminio, a las que

se les ha añadido una pieza de separación y una rueda interna con taladros, que permiten cuantificar el giro. En el extremo de la rueda que hace contacto con el terreno, se ha adjuntado una junta de tubería industrial de 120mm de diámetro, que hace las veces de neumático. Dichas ruedas se encuentran representadas en la figura 4. Con esta configuración se obtiene una rueda ligera y con un alto agarre a todo tipo de material.

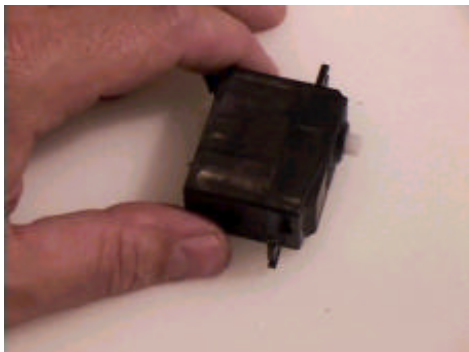


Figura 3. Servos trucados de Futaba.

Cuando se utiliza una configuración en triciclo es preciso seguir una regla fundamental: el rodamiento debe soportar sobre sí la menor cantidad de peso posible [3]. Si esta regla no se cumpliera dicha rueda podría quedarse atascada en un giro e incluso producir un salto del robot. Por ello mismo se ha intentado que todo el peso de la electrónica y de las baterías de alimentación, tanto para la lógica, como para los motores se sitúe sobre el eje común creado por los dos motores.



Figura 4. Ruedas de METABOT.

El sistema de control microprocesador, las baterías para la alimentación del mismo, así como los sensores de distancia se han situado sobre la base de Policarbonato.

### 3. Hardware.

El elemento microprocesador para el control del sistema está formado por la tarjeta de desarrollo CT6811 de la empresa Microbótica S.L. [4], dicha tarjeta está basada en el microcontrolador MC68HC11A8 de Motorola.

El sistema cuenta además con la tarjeta de expansión CT293+, capaz de excitar dos motores de continua con el

punteo en H integrado L293D [5] y de trabajar con los sensores fotoreflexivos de infrarrojos CNY70

El microcontrolador MC68HC11A1 funciona con una señal de reloj de sistema interna de 2MHz, y dispone únicamente de 256 bytes de RAM interna y 512 bytes de memoria EEPROM interna, lo que resulta insuficiente para el programa a cargar en el robot, por ello se ha diseñado e implementado una placa de expansión de memoria que aumenta la capacidad del sistema a 32Kbytes de memoria RAM no volátil y una memoria Flash de la misma capacidad. Esta placa permite además la conexión de un LCD de 2x16 líneas para la depuración del programa y la visualización de resultados.

Sensor	Función	Cantidad	Lectura
GP2D12	Medida de distancia	5	Analógica
OPB804	Encoder	2	Digital

Tabla 1. Sensores utilizados por el Robot.

Un robot destinado a resolver laberintos debe considerar dos problemas fundamentales el primero es la detección de paredes o espacios vacíos y el segundo su posición relativa o absoluta dentro del laberinto.

La tabla 1 muestra los sensores utilizados para resolver los problemas antes mencionados. La medida de distancias y detección de paredes se realiza a través de tres sensores GP2D12 [6] [7] de Sharp, estos dispositivos son capaces de entregar una medida de tensión entre 3.1V y 0.6V cuando detectan una distancia ente 10cm y 80cm respectivamente, lo que los hace ideales para la medida de pequeñas distancias como las del pasillo del laberinto.

Estos sensores incorporan toda la electrónica necesaria para que el usuario solo tenga que conectarlos a su dispositivo de medida y comenzar a tomar muestras. Además existe la posibilidad de ajustar la distancia focal de las lentes para aumentar o reducir, en algunos centímetros, la distancia máxima a detectar.

Los sensores se han conectado al puerto E del microcontrolador, ya que este es el puerto donde se sitúa el convertidor Analógico/Digital del mismo. De esta forma es posible tomar medidas de los sensores con una frecuencia máxima de 20KHz.

Los sensores se han colocado en la parte delantera y lateral del robot, de forma que apuntan a izquierda, derecha y hacia delante con una diferencia de 90°. Lo que permite la detección de camino libre o paredes hacia adelante o a los lados.

La odometría del robot se realiza con dos encoders en cuadratura, situado uno en cada rueda. Cada uno de estos encoders se ha realizado haciendo uso de una pareja de foto-reflectores infrarrojos OPB804 [8] que apunta a un disco de comacem taladrado, donde se ha hecho 50 agujeros equiespaciados. La resolución del sistema resulta por tanto:

$$\text{Resolución} = \frac{360^\circ}{100} = 3.6^\circ$$

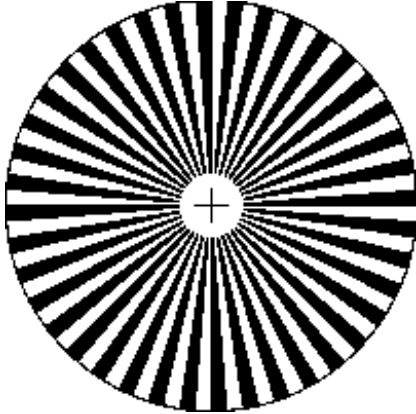


Figura 5. Disco de bandas del Encoder.

En un encoder en cuadratura el giro de la rueda produce dos señales con un desfase de 90°, denominadas usualmente canal A y canal B y representadas en la figura 6. Dichas señales permiten conocer la dirección de giro de la rueda, pero sobre todo duplicar la resolución de la medida, de forma que puede llegarse hasta los 1,8°.

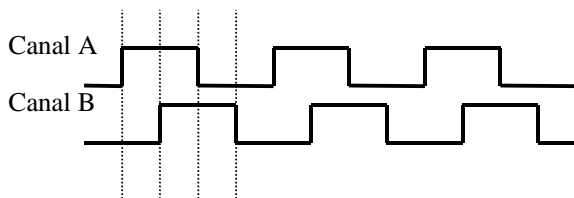


Figura 6. Señales del Encoder en cuadratura.

La conexión de los sensores OPB804 puede realizarse directamente a la tarjeta CT293+, que incorpora los conectores y la electrónica necesaria para la polarización y acondicionamiento de los foto-reflectores. Así, una simple lectura del puerto A del microcontrolador nos permite detectar si los sensores están sobre comacem o leyendo un agujero, al leer un '0' o un '1' lógico, respectivamente.

Más aún, el microcontrolador puede ser interrumpido cada vez que el sensor detecte un paso de una banda de un agujero a comacem, o viceversa. Con este método el sistema puede estar actualizando continuamente las medidas relativas a la posición y velocidad del robot, de forma independiente al programa principal.

#### 4. Software y estrategias de control

El código diseñado para el robot se puede separar en dos grandes bloques, el primero formado por el programa

principal, que consiste en un bucle de control que se ejecuta de forma periódica e infinita, y el segundo formado por las interrupciones que permiten realizar la excitación de los motores y la interacción con los elementos externos, en este caso los encoders. Vamos a ver cada uno de ellos en más detalle.

#### 4.1 Interrupciones.

Dos son las tareas que deben cumplir las interrupciones, como ya se ha comentado, la primera es la generación de una señal PWM para la excitación de cada uno de los motores, y la segunda la detección de la posición relativa del robot, así como su velocidad y sentido de giro de los motores.

##### 4.1.1 PWM

La generación de esta señal puede realizarse de dos formas, la primera denominada signo-magnitud, en la que se ha de suministrar al sistema una línea que indique la dirección de giro del motor, y otra que sea la PWM en si, y que indique la velocidad de giro. La segunda denominada anti-fase y representada en la figura 7, en la que la velocidad y dirección de giro se controlan con una sola línea. Con este segundo método una PWM del 50% implica que el motor se encuentra parado, mientras que ciclos de trabajo superiores supondrán que el motor comience a girar en una determinada dirección. Naturalmente ciclos de trabajo inferiores harán girar al motor en una dirección opuesta.

La señal PWM se ha implementado con el método de anti-fase, pues simplifica la cantidad de líneas de control a utilizar, y permite mayor control del motor.

El MC68HC11A1 dispone de diferentes elementos temporizadores que permiten realizar medida de tiempo, generación de señales de reloj, detección y cuenta de eventos, etc. Para la generación de la PWM necesaria para excitar los motores se utilizan los capturadores de salida, a los cuales solo es preciso suministrarles la cantidad de ciclos de reloj del sistema que la PWM ha de estar a nivel alto y a nivel bajo. En este caso se ha optado generar una PWM de 2KHz, lo que supone que un periodo de la misma se ha de separar en 10000 ciclos de la señal de reloj del sistema, en alguno de los cuales la señal de salida (PWM) estará a nivel alto y otros en los que estará a nivel bajo.

Así por ejemplo si deseamos parar el motor sería preciso programar 5000 ciclos en el tiempo a nivel alto y 5000 ciclos en el tiempo a nivel bajo. Obteniendo una señal cuadrada en la salida.

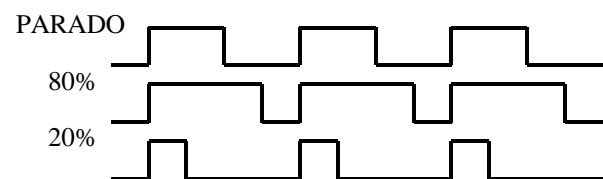


Figura 7. Señales PWM.

#### 4.2.2. Encoders.

Los dos sensores OPB804 que forman los encoders se han conectado a dos captadores de entrada y al acumulador de pulsos del microcontrolador. Estos son elementos capaces de detectar flancos de subida y de bajada en las entradas e interrumpir al sistema cada vez que esto ocurre. Utilizando esta capacidad puede implementarse la máquina de estados representada en la figura 8 que aumenta la resolución de la medida, ya que en cada cambio de estado hacia uno superior el contador se incrementa ya que la dirección de giro es hacia delante, y sino se decrementa, determinando que la dirección de giro es hacia atrás.

Este sistema es utilizado por múltiples dispositivos comerciales orientados al control de motores, como es el caso del HCTL1000 de Hewlett Packard.

embargo resulta la única opción, aparte de la posibilidad de “vagar” por el laberinto, siguiendo una de las paredes del mismo.

El sistema puede pasar por varios estados dependiendo de su posición y de la lectura de sus sensores. Así, existe un estado “Busqueda” en el cual el sistema se encuentra de partida y siempre que el robot tenga camino para recorrer hacia delante o hacia los lados. En este estado el robot avanza almacenando posibles caminos laterales, detectados con sus sensores derecho e izquierdo, un camino se considera como una bifurcación válida una vez se ha detectado el final de la pared y la continuación de la misma, esto se hace ante la posible existencia de espacios abiertos. Esta idea puede quedar más clara observando la figura 9, en ella el robot detecta inicialmente la falta de pared (Situación 1), pero no marca el camino como:

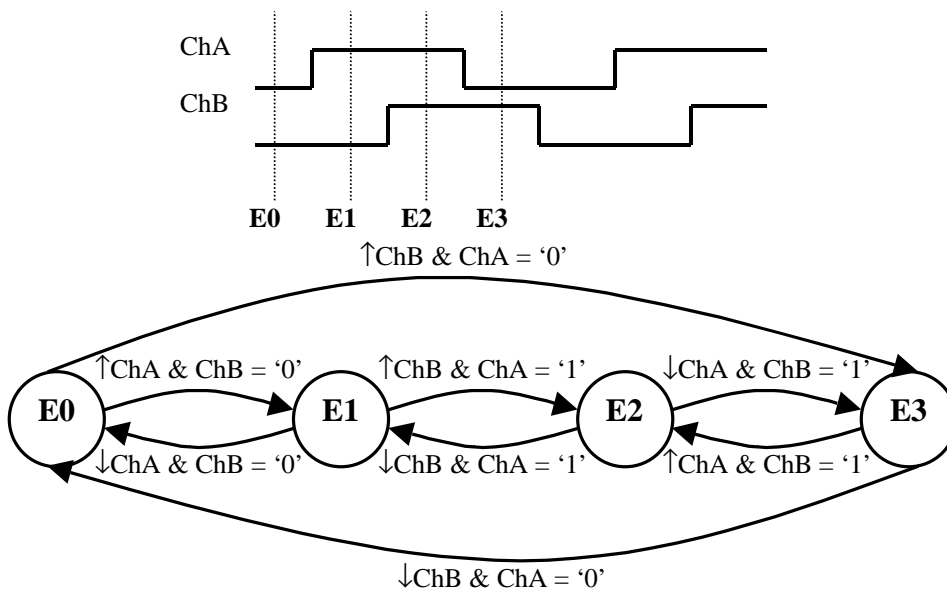


Figura 8. Máquina de estados utilizada con los encoders.

#### 4.2 Programa Principal.

El programa principal representa el mayor nivel de abstracción del sistema, en él se implementan los lazos cerrados para el control de la velocidad y la posición del robot, así como el nivel de “inteligencia” del mismo.

Para asegurar que el robot avanzará de forma correcta se ha implementado un lazo Proporcional-Integral-Derivativo [9] entre los dos motores, tal y como se recomienda en el libro “Mobile Robots: Inspiration to Implementation” [10]. Para conseguir que las dos ruedas avancen a la misma velocidad. Este lazo se repite periódicamente cada 200ms.

El algoritmo utilizado para la resolución del laberinto es el de “Backtrack” [11] o búsqueda y marcha atrás, según el cual el robot avanza siempre que tenga vía libre hacia delante, y se da la vuelta y analiza las posibles alternativas que dejó a derecha e izquierda cuando no le es imposible seguir avanzando. Es un método completo pero no óptimo pues no garantiza la mejor solución. Sin

“válido y aun por recorrer”, hasta que no encuentra la continuación de la pared (Situación 2). En la figura 10 el robot detecta el espacio abierto (Situación 1) pero no lo incluye en su memoria como una posible bifurcación.

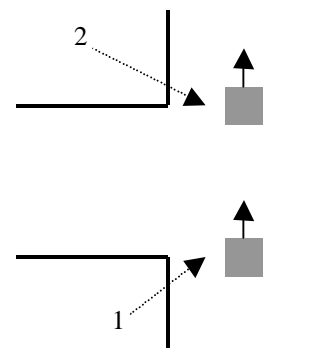


Figura 9. Detección de posibles caminos.

En este modo los pasillos se utilizan como referencia para centrar al robot, que puede haberse descolocado tras varios giros sucesivos, evitando el uso de “Bumpers” para detectar el contacto con una pared.

Cuando el robot detecta que es imposible seguir avanzando, tanto hacia delante, como hacia los lados se entra en el modo “Vuelta”, el robot gira 180° y se dispone a estudiar caminos laterales descartados durante el avance en modo Búsqueda. Con un buen criterio de almacenamiento de caminos se puede recorrer el laberinto completo encontrando la salida.

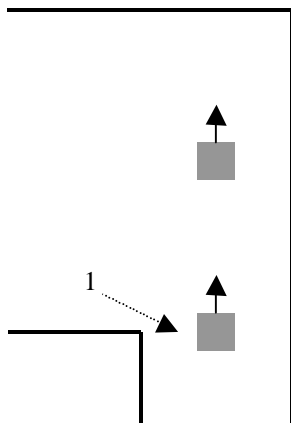


Figura 10. Navegación en un espacio abierto.

Conociendo que la salida está en el extremo opuesto a la entrada una buena política de maniobra es elegir siempre caminos distintos en cada giro, así si la última vez se giró a la izquierda esta vez será a la derecha, con lo que el robot puede avanzar zigzagueando. Este método permite además evitar posibles lazos en los que el robot se quedaría dando vueltas eternamente.

Por último el robot dispone de un modo “Maniobra” en el que se incluyen las maniobras necesarias para girar a derecha e izquierda y dar media vuelta.

El giro de 90 grados se realiza frenando una de las ruedas y moviendo la otra, con lo que el robot describe un arco, de esta forma el error de giro depende únicamente de uno de los encoders y es 1.8°, además resulta un proceso menos brusco que poner las ruedas a girar en sentido contrario, tal y como lo hacen los tanques.

El giro de 180° se realiza cuando el robot se ha quedado encerrado, en este caso si es necesario poner las ruedas a girar en sentido contrario para rotar sobre el eje del robot. El sistema puede centrarse con respecto a las paredes haciendo uso de los sensores de distancia y aprovechando que ha quedado encajonado.

## 5. Especificaciones

Físicas	
Dimensiones	140 x 140 x 120 (mm)
Peso (Sin Baterías)	999 g
Separación del Eje.	135 mm
Diámetro de la Rueda.	85 mm ( $\pi \cdot d = 263\text{mm}$ )
Velocidad Máxima	0,195 m/s

Tabla 2. Especificaciones Físicas.

Eléctricas	
Alimentación de la lógica.	5V
Alimentación de los motores.	9V
Tensión de referencia del ADC	2,5V
Consumo (Lógica y Sensores)	150mA

Tabla 2. Especificaciones Eléctricas.

## 6. Conclusiones

A la vista de las pruebas realizadas AURELIO parece una buena plataforma para el desarrollo de un sistema de resolución de laberintos, siendo además un robot para múltiples propósitos, que puede convertirse fácilmente en un rastreador de líneas en el suelo o en un seguidor de paredes.

Sin embargo durante la realización del robot y su programación se han encontrado varios aspectos que han dificultado el trabajo y que conviene comentar. El primero es la baja capacidad de cálculo del microcontrolador utilizado, lo que lleva al desbordamiento del sistema cuando se trabaja con aritmética de punto flotante, algo necesario para el control PI de los motores. Por ello es posible que, en futuras ampliaciones, se sustituya la placa de desarrollo por una con un microprocesador más potente.

El encoder en cuadratura también supuso un problema pues las señales no estaban desfasadas 90° exactamente, debido a la colocación de los sensores y el ancho de las bandas, con lo que algunos estados permanecían más tiempo que otros. De esta forma la medida de velocidad no era nada precisa aun girando la rueda a una velocidad constante.

Al margen de estos inconvenientes técnicos los resultados han sido satisfactorios y AURELIO resulta una buena plataforma para el desarrollo y prueba de algoritmos de Inteligencia Artificial, ya sean clásicos como la resolución de laberintos mediante búsqueda y marcha atrás o más modernos como los obtenidos mediante programación genética [12].

## 7. Agradecimientos

Quisiéramos aprovechar la oportunidad para agradecer su ayuda a Sebastián González Andino, por sus continuas sugerencias en cuanto a la programación del robot, y en general a nuestras familias que nos han facilitado el trabajo y que en algunos casos nos han aportado ideas o material para el diseño y construcción del robot.

También queremos agradecer al comité organizador de Alcabot 2002 y al Departamento de electrónica de la U.A.H. por la ayuda prestada a los participantes, bien fuera mediante información, seminarios, cursos, o simplemente cediendo laboratorios para la realización de pruebas.

## 8. Referencias

- [1] FutabaHomePage:  
[www.futabarc.com](http://www.futabarc.com)
- [2] Seattle Robotics Society Homepage:  
[www.seattlerobotics.com](http://www.seattlerobotics.com)
- [3] Karl Lunt, “*Build Your Own Robot!*”, A K Peters Ltd; ISBN: 1568811020
- [4] Página oficial de la empresa Microbótica S.L.:  
[www.microbotica.com](http://www.microbotica.com)
- [5] Información sobre el puente en H L293D:  
[www.st.com](http://www.st.com)
- [6] Información sobre el sensor GP2D12:  
[www.sharp.co.jp/ecg/opto/products/f-osd-e.html](http://www.sharp.co.jp/ecg/opto/products/f-osd-e.html)
- [7] H.R. Everett, “*Sensors for Mobile Robots : Theory and Application*”, A K Peters Ltd; ISBN: 1568810482
- [8] Información sobre el sensor CNY70:  
[www.vishay.com](http://www.vishay.com)
- [9] Fco. Javier Rodríguez Sánchez, Felipe Espinosa Zapata y otros, “*Análisis, Diseño Y Realización De Sistemas Electrónicos De Control Discreto*”, Servicio de Publicaciones de la Universidad de Alcalá, ISBN: 84-8138-357-0
- [10] Joseph L. Jones, Anita M. Flynn, Bruce A. Seiger, “*Mobile Robots : Inspiration to Implementation*”, A K Peters Ltd; ISBN: 1568810970.
- [11] Ronald C. Arkin “*Behavior-Based Robotics (Intelligent Robots and Autonomous Agents)*”, MIT Press; ISBN: 0262011654
- [12] J.R. Koza, “*Genetic programming: On the programming of computers by means of natural selection*”, MIT Press, Cambridge, MA, 1992.