(19) OFICINA ESPAÑOLA DE PATENTES Y MARCAS

ESPAÑA

(11) Número de publicación: **2 628 195**

(51) Int. CI.:

B25J 9/16 (2006.01)
G05B 19/409 (2006.01)

(12) TRADUCCIÓN DE PATENTE EUROPEA T3

(86) Fecha de presentación y número de la solicitud internacional: **25.05.2010 PCT/EP2010/057111**

(87) Fecha y número de publicación internacional: **02.12.2010 WO10136427**

(96) Fecha de presentación y número de la solicitud europea: **25.05.2010 E 10723567 (3)**

(97) Fecha y número de publicación de la concesión europea: **29.03.2017 EP 2435216**

(54) Título: **Sistema y procedimiento para editar y controlar los comportamientos de un robot móvil**

(30) Prioridad:

26.05.2009 FR 0953434

(45) Fecha de publicación y mención en BOPI de la traducción de la patente:
02.08.2017

(73) Titular/es:

SOFTBANK ROBOTICS EUROPE (100.0%)
43 rue du Colonel Pierre Avia
75015 Paris, FR

(72) Inventor/es:

MONCEAUX, JÉRÔME y
MAISONNIER, BRUNO

(74) Agente/Representante:

CARPINTERO LÓPEZ, Mario

DESCRIPCIÓN

Sistema y procedimiento para editar y controlar los comportamientos de un robot móvil

La presente invención pertenece al campo de los sistemas de programación de robots. Más precisamente, se aplica a la edición y el control de los comportamientos y movimientos de robots que se desplazan sobre o utilizan unos miembros articulados, principalmente de forma humana o animal. Un robot puede calificarse de humanoide a partir del momento en que posee ciertos atributos de aspecto humano: una cabeza, un tronco, dos brazos, dos manos, dos piernas, dos pies, etc. Un robot humanoide puede sin embargo estar más o menos evolucionado. Sus miembros pueden tener un número más o menos grande de articulaciones. Pueden gestionar por sí mismos el equilibrio estático y dinámico y marchar sobre dos miembros, eventualmente en tres dimensiones. Pueden captar unas señales del entorno (“escuchar”, “ver”, “tocar”, “oler” …) y actuar según unos comportamientos más o menos sofisticados, de modo que interactúe con otros robots o humanos, bien mediante la palabra o bien mediante el gesto. Para aprovechar lo mejor de estas posibilidades, se hace necesario proporcionar una unidad informática de control del robot para los controles de dichos comportamientos. Ciertos robots pueden buscar de manera autónoma unos controles necesarios para ejecutar unos comportamientos correspondientes a su perfil de utilización. Continúa siendo sin embargo necesario que un usuario pueda intervenir para crear nuevos comportamientos y movimientos. Una primera manera de programar los controles necesarios es hacer que se desarrollen según una secuencia temporal predeterminada. Esta lógica de programación presenta la ventaja de que el comportamiento del robot es predecible en el tiempo: los comportamientos tienen un inicio y un fin que se producen en unos momentos predecibles. Véase por ejemplo el procedimiento de edición y de programación temporal de los comportamientos de un robot divulgado por la Patente US 6.718.231. Sin embargo, esta lógica encuentra su límite en el hecho de que las interacciones del robot con su entorno son a la vez inevitables (encuentra obstáculos en el camino) y deseables (reacciones del robot a las solicitaciones del entorno o de su usuario), lo que interrumpe el curso de la secuencia de comportamientos preprogramados. Esto es por lo que se ha utilizado otra lógica de programación, de tipo por eventos, inspirada en la teoría de los autómatas tanto para los robots humanoides como para los robots industriales: en esta lógica de programación, los vínculos relacionan unos comportamientos, dependiendo el segundo en una cadena de las salidas del primero. Esta lógica permite gestionar las interacciones con el entorno. Véase por ejemplo el procedimiento de edición y de programación por eventos de un robot divulgado por la Patente US 6.470.235. Esta lógica presenta sin embargo el inconveniente de interrumpir una secuencia de comportamientos preprogramada en tanto que el evento esperado no se ha producido. Un ejemplo simplificado permitirá comprender fácilmente esta dificultad: el usuario desea que el robot pase de una primera posición a una segunda posición que debe esperar a un instante determinado. Por el camino, puede tener que efectuar unas acciones que dependen de interacciones con el entorno (decir “buenos días” a otro robot con el que se encuentra, recoger un objeto, etc.). Una programación puramente por eventos, no limitada por el tiempo, no puede resolver este problema.
Un procedimiento que yuxtapone las dos lógicas de programación por eventos y temporal se divulga por la Solicitud de Patente Japonesa JP2002/120174. En dicho procedimiento, sin embargo, el programador debe intervenir no obstante en uno de los dos bloques de programa para hacerlos compatibles. Este procedimiento no resuelve por tanto el problema que consiste en convertir a las dos lógicas de programación por eventos y temporal en compatibles sin intervención del programador. Igualmente, la Solicitud de Patente US2007/150102 divulga unas lógicas de programación por eventos y temporal que no se combinan en el seno de una misma Caja de control.

La presente invención resuelve este problema previendo un sistema y un procedimiento de edición y de programación de los comportamientos de un robot que combina las lógicas de programación temporal y por eventos sin intervención del programador en cada evento.
Con este fin, la presente invención divulga un procedimiento de edición y un programa de control según la reivindicación 1.

La invención divulga igualmente un programa de ordenador para realizar el procedimiento de edición según la reivindicación 11, así como un programa de control según la reivindicación 12 y un robot según la reivindicación 13.

Le invención proporciona igualmente la ventaja de permitir una edición amigable de los controles a ejecutar por el robot que permite a la vez la utilización de las Cajas de control de comportamientos y de movimientos preprogramados. Las Cajas se organizan en una estructura jerárquica que permite descomponer los comportamientos complejos en comportamientos elementales. Los comportamientos elementales pueden programarse en lenguaje de scripts directamente interpretable por el robot. Los scripts se organizan en módulos que utilizan unas estructuras de datos comunes y son por ello independientes entre sí. El usuario puede intervenir a todos los niveles del árbol jerárquico de los comportamientos. El usuario puede igualmente crear unas interacciones entre unos comportamientos de una rama del árbol y otros comportamientos que pertenecen a otra rama del árbol. Puede intervenir igualmente en los movimientos elementales del robot que determinarán un comportamiento dado. Además, el sistema es compatible con varios entornos de desarrollo (lenguajes Python, URBI, principalmente: Windows XP, Mac OS, Unix…).

La invención se comprenderá mejor y surgirán sus diferentes características y ventajas de la descripción que sigue de varios ejemplos de realización y de sus figuras adjuntas de las que:

- la figura 1 es un esquema de la arquitectura funcional de edición y de programación de los comportamientos de un robot en un modo de realización de la invención;
- la figura 2 es una vista de la pantalla principal del sistema de edición y de programación de los comportamientos de un robot en un modo de realización de la invención;
- la figura 3 es una vista de la pantalla que permite a un usuario editar las Capas de comportamiento y de movimiento de un robot en un modo de realización de la invención;
- la figura 4 es una vista de la pantalla que muestra las Cajas de control de comportamientos conectadas para editar y programar una Trama principal de comportamientos a ejecutar por un robot en un modo de realización de la invención;
- la figura 5 es un organigrama funcional de la arquitectura de una Caja de control del sistema de edición y de programación de los comportamientos de un robot en un modo de realización de la invención;
- la figura 6 es una vista que muestra los diferentes tipos de conexiones entre las Cajas de control de comportamientos conectadas para editar y programar una Trama principal de comportamientos a ejecutar por un robot en un modo de realización de la invención;
- la figura 7 es una vista que muestra un ejemplo de programación por acción directa sobre una representación gráfica de los motores de un robot virtual en un modo de realización de la invención;
- las figuras 8a y 8b son unas vistas de dos pantallas que ilustran un ejemplo de movimiento de un robot imposible de ejecutar en el tiempo concedido, en un modo de realización de la invención;
- las figuras 9a, 9b y 9c son unas vistas de pantallas que ilustran un ejemplo complejo de edición de comportamientos de un robot en un modo de realización de la invención;
- las figuras 10a, 10b y 10c son unas vistas de pantallas que ilustran un ejemplo de edición de un encadenamiento de comportamientos de un robot sincronizados con un eje temporal de Tramas o Timeline en un modo de realización de la invención;
- la figura 11 ilustra un ejemplo de script que permite la programación directa de la ejecución de un comportamiento de un robot en un modo de realización de la invención.

La figura 1 es un esquema de la arquitectura funcional de edición y de programación de los comportamientos de un robot en un modo de realización de la invención.

El robot controlado por el procedimiento y el sistema de la invención puede ser un robot humanoide que tenga una cabeza, un tronco y cuatro miembros, estando articulada cada una de las partes, estando controlada cada articulación por uno o varios motores. La invención permite a un usuario del sistema controlar un robot de ese tipo creando los comportamientos simulados sobre un robot virtual y ejecutados sobre el robot real unido al sistema mediante un enlace por cable o inalámbrico.

Se trata de visualizar, de simular y de hacer ejecutar unos comportamientos (tales como la marcha —en recto, a la derecha o a la izquierda en n pasos; un “hola”— movimientos de uno de los brazos por encima de la cabeza; la voz, etc.) y los movimientos (de la cabeza, de una parte del miembro, en un ángulo dado) sobre la pantalla de un ordenador programado para hacerlo.

La figura ilustra la articulación de los controles desencadenados por los eventos con su dimensión temporal. Los controles desencadenados por unos eventos se representan en la semántica de la invención por unas Boxes o “Cajas” o “Cajas de control” 10. Una Caja es una estructura de programación arborescente que puede comprender uno o varios de los elementos que siguen que se definen a continuación:

- Una “Timeline” o eje 20 temporal de Tramas;
- Un “Diagram” o Diagrama 70 de flujo
- Un Script 90

Las Cajas de control se unen normalmente entre sí mediante unas conexiones que transmiten frecuentemente una información de eventos de una Caja a la otra, como se detalla más adelante en la descripción. Toda Caja se une directamente o indirectamente a una “Caja raíz” o Raíz que inicializa el escenario de comportamiento/movimiento del robot.

Un eje 20 temporal de Tramas representa la restricción temporal a la que deben someterse los comportamientos y los movimientos del robot definidos en la Caja en la que se inserta dicho Eje temporal de Tramas. En lo que sigue de la descripción y de las reivindicaciones, se utilizará la denominación anglosajona de Timeline, comúnmente admitida con el mismo sentido en el mundo de la programación. La Timeline realiza así la sincronización de los comportamientos y movimientos de la Caja. Esta se descompone en frames (Tramas) a las que se asocia una velocidad de desarrollo definida el número de Tramas por segundo o Frames Per Second (FPS). Las FPS de cada Timeline es parametrizable por el usuario. Por de omisión, la FPS puede fijarse a un valor dado, por ejemplo 15 FPS.

Una Timeline puede comprender:

- una o varias Behaviour Layers o “Capas de comportamiento” 30, comprendiendo cada una uno o varios Behaviour Key Frames o “Tramas principales de comportamiento” 50, que pueden comprender a su vez uno o varios Diagrams o “Diagramas de flujo” 70, que son de hecho unos conjuntos de Cajas que pueden relacionarse igualmente directamente con una Caja de nivel superior, sin pasar por una Capa de comportamiento ni una Timeline;
- Una o varias Motion Layers o “Capas de movimiento” 40, comprendiendo cada una uno o varios Motion Key Frames o “Tramas principales de movimiento” 60 que pueden comprender una o varias Motion Screens o

“Pantallas de movimiento” 80.

Una Capa de comportamiento define un conjunto de comportamientos del robot o Tramas principales de comportamiento. Pueden definirse varias Capas de comportamiento en el seno de una misma Caja. Se programarán entonces para desarrollarse de manera sincronizada por la Timeline de la Caja.

Una Capa de comportamiento podrá comprender una o varias Tramas principales de comportamiento. Una Trama principal de comportamiento define un comportamiento del robot, tal como la marcha (“Marcha”), la palabra (“Decir”), el juego de música (“Música”), etc. Un cierto número de comportamientos se preprograman en el sistema de la invención para ser insertados directamente por el usuario en un simple “drag and drop” a partir de una librería tal como se detalla más adelante en la descripción. Cada Trama principal de comportamiento se define por un evento desencadenador que es el inicio de la Trama que se inserta en la Timeline. El final de la Trama principal de comportamiento no está definido más que en la medida en la que se inserta a continuación otra Trama principal de comportamiento, o si se define un evento de final.

Una Capa de movimiento define un conjunto de movimientos del robot que se programan mediante una o varias Tramas principales de movimiento sucesivas que reagrupan unos movimientos de los motores de las articulaciones del robot. Estos movimientos a ejecutar se definen por las posiciones angulares de llegada de dichos motores que pueden programarse mediante la acción sobre unas pantallas de movimiento, siendo detalladas dichas acciones más adelante en la descripción. Todas las Tramas principales de movimiento de una misma Caja se sincronizan por la Timeline de la Caja. Una Trama principal de movimiento se define por una Trama de llegada. La Trama de salida es la del final de la Trama principal de movimiento precedente o la del evento de inicio de la Caja.

Se designa bajo la denominación común de Trama principal de acción a las Tramas principales de comportamiento y a las Tramas principales de movimiento.

Es posible ejecutar en paralelo varias Tramas principales de acción (de comportamiento o de movimiento), con la condición de que se relacionen con la mismo Timeline.

Un Diagrama de flujo es un conjunto de Cajas conectadas entre sí, como se detalla más adelante. Cada una de las Cajas puede comprender a su vez otras Timeline con las que se relacionan nuevas Capas de comportamiento o de movimiento.

Un script es un programa directamente ejecutable por el robot. Una Caja que comprende un script no comprende otro elemento.

La figura 2 es una vista de la pantalla principal del sistema de edición y de programación de los comportamientos de un robot en un modo de realización de la invención.

La parte derecha 2.1 comprende una imagen virtual del robot a controlar que permitirá realizar unas simulaciones de la ejecución de los controles. El robot puede ser de cualquier tipo, pero la invención será particularmente ventajosa en el caso de un robot humanoide o de forma animal, que tenga una cabeza, un cuerpo y unos miembros. La parte de la izquierda 2.2 da acceso a una biblioteca de comportamientos que se almacena en el ordenador. Estos comportamientos son unos componentes reutilizables con los que pueden definirse unos comportamientos elementales utilizados frecuentemente como la marcha, la danza, el habla, la iluminación de los ojos o de otros LED, etc.

La parte central 2.3 permite ensamblar unos comportamientos extraídos en la biblioteca 2.2 para editarlos y ensamblarlos en secuencias de comportamientos bajo la forma de Diagramas de flujo, como se indica de manera detallada más adelante en la descripción.

La figura 3 es una vista de la pantalla que permite a un usuario editar las Capas de comportamiento y de movimiento de un robot en un modo de realización de la invención.

Como se ve en las partes 3.1 (Capas de comportamiento) y 3.2 (Capas de movimiento), pueden definirse varias Capas en el tipo de comportamiento y el tipo de movimiento y, en cada Capa, pueden crearse varias Tramas principales. Las Capas de tipo comportamiento se describirán en lo que sigue de la descripción.

La creación de las Tramas principales de las Capas de tipo movimiento se explica más adelante en la descripción. Se explican en este caso las modalidades de edición de las Tramas principales de movimiento. Es posible copiar o desplazar una secuencia en el seno de una Capa de movimiento o de una Capa de movimiento a otra. En este último caso, puede ser que el movimiento no pueda ejecutarse debido a los conflictos de órdenes dadas a un mismo motor.

Es posible alisar las transiciones entre secuencias de movimientos realizando una interpolación de tipo lineal o exponencial, por ejemplo. Es igualmente posible duplicar una secuencia definida para un miembro superior o inferior en una secuencia aplicable al miembro simétrico en donde los movimientos son en sí mismos simétricos.

Como se visualiza en la parte 3.3 de la figura, en una Capa de tipo movimiento, se puede visualizar igualmente la evolución de los motores de cada una de las partes del robot en un sistema de ejes cuya abscisa es la Timeline de las Capas de comportamiento y de movimiento y las ordenadas son el ángulo de giro de cada uno de los motores que puede ser positivo o negativo. En un ejemplo de realización de la invención, el robot puede incluir varios motores. Los movimientos pueden modificarse desplazando los puntos de parada del motor en abscisas y/o en ordenadas. Ciertos movimientos del motor no son compatibles entre sí y no podrán ejecutarse.

La figura 4 es una vista de una pantalla que muestra las Cajas de control de comportamiento conectadas para editar y programar una secuencia de comportamientos a ejecutar por un robot en un modo de realización de la invención.

La Caja de control de comportamiento es el ladrillo de base del sistema de edición y de control de un robot según la invención. Una Caja de control de comportamiento define una o varias acciones e incluye una o varias entradas y

una o varias salidas. En el ejemplo de la figura 4, la entrada es el evento “Inicio de la secuencia de comportamiento X” arriba y a la izquierda de la pantalla 2.3 de la figura 2. Este punto se une mediante un arco a una de las entradas situadas a la izquierda de la primera Caja “Movimiento”. La salida de dicha Caja se conecta a una de las entradas de la segunda Caja “Hola”. La salida de esta segunda Caja se conecta al evento “Final de la secuencia de comportamiento X” arriba y a la derecha de la pantalla. Las dos Cajas se han creado en la pantalla por copia de los elementos correspondientes de la biblioteca 2.2 de la figura 2 (operación de arrastrar y soltar). Las condiciones se han creado de manera gráfica uniendo los puntos de entrada/salida con la ayuda del ratón. Como se ha indicado en el comentario de la figura 1, en un modo de realización de la invención, las Cajas pueden ser de tres tipos:

- Caja de scripts, que comprende una serie de instrucciones en lenguaje informático; los lenguajes utilizados pueden ser unos lenguajes especializados en el control de robots tales como URBI™;
- Caja Diagrama de flujo, que comprende varias Cajas, de las que por ejemplo una Caja permite al usuario definir un parámetro utilizado por otra Caja, tal como, por ejemplo, la distancia de marcha;
- Caja que comprende una Timeline con una o varias Capas de movimiento y/o una o varias Capas de comportamiento.

La figura 5 es un organigrama funcional de la arquitectura de una Caja del sistema de edición y de programación de los comportamientos de un robot en un modo de realización de la invención.
El organigrama de la figura recapitula las diferentes posibilidades ofrecidas por las Cajas: scripts, Diagramas de flujo, Timeline con, por una parte, una Trama principal de comportamiento y por otra parte otro Diagrama de flujo compuesto de comportamientos.

La figura 6 es una vista que muestra diferentes tipos de enlaces entre unas Cajas de comportamiento conectadas para editar y programar una Trama principal de comportamiento a ejecutar por un robot en un modo de realización de la invención.
Son posibles varios tipos de entradas:

- OnStart: entrada de salida que lanza la ejecución de una Caja;
- OnStop: entrada, representada por una cruz, que detiene la ejecución de la Caja cuando se activa; la parada no interviene sin embargo más que al final de una acción cuya interrupción en el curso de ejecución pondría al robot en posición inestable (caso de la Caja “Andar” o “Marcha”, por ejemplo);
- OnStoped: salida que se activa cuando se finaliza la ejecución de la Caja; esta salida puede vincularse a las entradas de otras Cajas; estas entradas se activarán cuando los comportamientos de las Cajas anteriores que están vinculadas a esta entrada hayan sido ejecutados completamente;
- OnLoad: entrada oculta en la estructura de una Caja; esta entrada aparece cuando se abre una Caja de varios niveles; se activa cuando está cargada completamente en la memoria una Caja de varios niveles, no interviniendo dicha carga más que para la activación de la Caja de nivel superior;
- Entrada conectada a ALMemory: esta entrada se activa cada vez que cambia el valor calculado en ALMemory;

    ◦ La memoria ALMemory es una base de datos interna del robot; es una memoria a corto término: se inserta en ella el último valor de una variable dada (con posibilidad de tener un histórico corto, es decir algunos valores procedentes de la variable), y la memoria puede encargarse de notificar a los módulos interesados por el cambio de una o varias variables. Contiene por ejemplo las informaciones de todos los captadores, actualizados cada 20 ms: valores angulares de motores, corriente, temperaturas, etc. En el curso de la ejecución sobre un robot, esta memoria puede contener más de 4000 variables. Todos los módulos que tratan sobre el robot o que están conectados a él pueden comunicar con ALMemory, para insertar unos valores que desean publicar, o para inscribirse en unas variables que les interesan y reaccionar a su cambio (por ejemplo: reaccionar al valor de la batería, para que el robot busque su cargador cuando el nivel de batería está por debajo de un cierto umbral).

- Entrada evento: cuando esta entrada se activa, el comportamiento en el curso de ejecución en la Caja se modifica;

    ◦ A título de ejemplo ilustrativo, se toma el caso de una Caja “Temporizador” que simula una de sus salidas con una cadencia regular: cada n segundos: una entrada de tipo “entrada evento” espera un parámetro del tipo “Número”, y cambia la cadencia de la Caja si recibe una señal en esta entrada; de ese modo, si llega una señal de valor “3”, la salida no se estimulará más que cada 3 segundos;

- Salida Puntual: está salida puede activarse en no importa qué momento durante la ejecución de un comportamiento en la Caja;

    ◦ A título de ejemplo ilustrativo, tomemos el caso de una Caja de detección de caras; esta Caja tiene una salida puntual de tipo “Número”; cuando la Caja está en el curso de ejecución, se estimula esta salida cada vez que el robot identifica en su campo visual un número de caras diferente a la del instante anterior. Si no había ninguna persona, y si el robot ve una cara, estimula su salida con la información “1”; si llega una 2ª cara, envía la información “2”, y si desaparecen las 2 caras un poco más tarde, envía “0”, se recupera de ese modo el evento “cara(s) identificada(s)”, con el número de caras identificadas.

Las Cajas y eventos se conectan entre sí por unos enlaces, como se ha representado en la figura 6. Un enlace transporta una señal de evento entre entradas y salidas. Se puede tener en ella varios tipos de señales de eventos:

- Una señal de activación, sin otra información;
- Una señal portadora de uno o varios números (por ejemplo, un número de pasos o una distancia);
- Una señal portadora de una o varias cadenas de caracteres (por ejemplo, un texto que debe pasarse al módulo de síntesis de voz de un robot);
- Una señal compuesta, que puede comprender varias señales, cada una de uno de los tipos anteriores.

Ventajosamente, estos enlaces se representan de manera gráfica de diferentes maneras, mediante unos colores o unos trazados de texturas diferentes. Estas Cajas no admiten más que unas entradas de ciertos tipos definidos y no entradas de otros tipos.
Es posible establecer unas conexiones entre dos Cajas que pertenezcan a unas Tramas principales de comportamientos diferentes pero que tengan una intersección sobre una misma Timeline. Para ello, el usuario del sistema de la invención puede colocarse en un instante sobre una Timeline y editar todas las Tramas principales de comportamiento activas en dicho instante. De ese modo, si se carga una primera Trama principal en la Timeline y aparece una segunda Trama principal con sus Cajas y sus conexiones, el usuario puede desplazar o modificar las Cajas y las conexiones entre ellas y modificar de ese modo de manera dinámica los comportamientos que se ejecutarán por el robot.

La figura 7 es una vista que muestra un ejemplo de programación mediante acción directa sobre una representación gráfica de los motores de un robot virtual en un modo de realización de la invención.
La figura 7 representa el brazo izquierdo de un robot. En este ejemplo, el brazo incluye una articulación brazo-hombro con dos grados de libertad ortogonales accionados por dos motores y una articulación mano-brazo, igualmente con dos grados de libertad. Los ángulos de rotación de los 4 motores de las articulaciones pueden controlarse directamente haciendo deslizar con el ratón los cursores que corresponden a cada uno de estos motores. Los valores de los ángulos se presentan para cada motor. En un modo estándar de realización de la invención, la realización del control para el motor de la articulación del robot virtual puede llevar del orden de un segundo. En el caso de que el movimiento no se realice en este tiempo, esto puede querer decir que el movimiento no es realizable, principalmente porque existe un obstáculo que se opone a la realización del movimiento.
Un indicador de estado muestra si el movimiento controlado ha podido registrarse. Si el indicador de estado está en una posición “No registrado”, esto puede querer decir que el movimiento no es ejecutable, principalmente porque un motor solicitado en una Capa de movimiento que depende de una Timeline no puede estar en otra Capa de movimiento que dependa de la misma Timeline.
Es posible visualizar las líneas tiempos/ángulos de cada uno de los motores mostrados en la figura 3, individualmente o para una parte dada. Como ya se ha indicado, es igualmente posible en este modo de control transferir los movimientos programados para un miembro sobre el miembro simétrico. En este caso, los movimientos de los motores son simétricos.
Es posible programar sobre esta pantalla un modo maestro-esclavo en el que un robot físico conectado al sistema de edición y de programación de la invención está controlado por el sistema y el robot virtual se esclaviza al robot físico.

Las figuras 8a y 8b son unas vistas de dos pantallas que ilustran respectivamente un ejemplo de movimiento de un robot imposible de ejecutar tal como se ha ordenado y un ejemplo de ajuste del movimiento para hacerlo ejecutable, en un modo de realización de la invención.
En la figura 8a, se ve que la línea que representa el movimiento ordenado no se une a la posición de la secuencia de movimiento siguiente a la que debería normalmente unirse (materializada por un pequeño cuadrado). Entre la posición de salida y la posición de llegada, no hay más que 5 pasos de secuencia de movimiento. Teniendo en cuenta el límite de velocidad fijado en el motor (que puede variar por ejemplo entre 4 y 8 rad/s según los motores), la velocidad de marcha de las Tramas (30 FPS en este caso), y la amplitud angular a recorrer por el motor (de -52 a +37 grados, en el ejemplo de la figura), la orden no puede ejecutarse.
En la figura 8b, se ve que el motor se pasa a la ejecución de la orden siguiente a partir de que se haya alcanzado la Trama en la que está posicionada la Trama principal de movimiento no ejecutable.

Las condiciones de imposibilidad son, tal como se ve, dependientes del número de FPS elegido por el usuario y de las velocidades límite de los motores, habiéndose observado que este último parámetro no admite más que pocas variaciones y depende de las características de los motores.

Las figuras 9a, 9b y 9c son unas vistas de pantallas que ilustran un ejemplo complejo de edición de comportamientos de un robot en un modo de realización de la invención.
La figura 9a ilustra un caso en el que se genera una primera secuencia de comportamiento (tramaclave 1 sobre capa1_comportamiento) por la creación en una Capa de tipo comportamiento mediante adición en la ventana de edición de las Cajas de una primera Caja “Danza”, estando sincronizada dicha primera secuencia con la Timeline en la capa1_movimiento. Se añade una 2ª Caja “LedCara” (LED de la cara) en la ventana de edición de las Cajas en la 1ª secuencia tramaclave 1.
Las figuras 9b y 9c son dos variantes de la figura 9a en las que:

- en la figura 9b, se sustituyen otros LED en la LedCara y se unen todos a la entrada de la Caja, lo que quiere decir que los comportamientos correspondientes (iluminación de los LED) se ejecutarán en paralelo;
- en la figura 9c, se añaden en secuencia otros LED antes de LedCara, lo que quiere decir que su iluminación se efectuará sucesivamente antes de la iluminación de los LedCara.

Las figuras 10a, 10b y 10c son unas vistas de pantallas que ilustran un ejemplo de edición de un encadenamiento de comportamientos de un robot sincronizados con la línea de tiempos en un modo de realización de la invención.
La figura 10a muestra una Caja “irAYParar” parametrizada por un índice de Trama de la Timeline (100 en el ejemplo de la figura). Cuando esta Caja se activa, la ejecución salta al paso designado por el parámetro de la Caja y la ejecución del comportamiento en curso se detiene.
Como se indicado en la figura 10b, existe igualmente una Caja de comportamiento “irAYReprod”, que se distingue de la Caja de comportamiento “irAYParar” en que, en lugar de detener el desarrollo del tiempo en la Capa de comportamiento afectada, deja de desarrollarse a partir del paso especificado. En el caso ilustrado, se conectan dos Cajas “irAYReprod” y “irAYParar” a las salidas de una Caja “Parachoques” que comprende los captadores de choque respectivamente del pie izquierdo y del pie derecho del robot (en un modo de realización, el robot tiene dos captadores de choque en cada pie).
La figura 10c muestra una vista interna de la Caja Parachoques en la que se ilustran los dos captadores “izquierda” (captadores del pie izquierdo) y “derecha” (captadores del pie derecho).
En relación con la figura 10b, se ve que

- si al menos uno de los captadores del pie izquierdo se activa (choque del pie izquierdo sobre un obstáculo detectado por el captador), el comportamiento en curso (Danza en el ejemplo de la figura) se detiene, pero la Timeline de la Capa de comportamiento en la que está insertado el comportamiento Marcha continúa desarrollándose;
- si al menos uno de los captadores del pie derecho se activa (choque del pie derecho sobre un obstáculo de cesado por el captador), el comportamiento en curso (Marcha en el ejemplo de la figura) se detiene y la Timeline de la Capa de comportamiento en la que está insertado el comportamiento Marcha se pone en pausa.

La figura 11 ilustra un ejemplo de script que permite la programación directa de la ejecución de un comportamiento de un robot en un modo de realización de la invención.
El ejemplo de la figura es un script de iluminación de los LED de la oreja derecha del robot (LedOrejaIzd). Un script es un módulo, en este caso en lenguaje Python, que es un lenguaje interpretado. La creación de una Caja de script se realiza en 3 etapas:

- instanciación de la Caja;
- registro de la Caja como un módulo que puede llamarse por las otras Cajas a la que están vinculadas;
- inicialización de la Caja.

El usuario no interviene en las 2 primeras etapas. Controla la 3ª. Como es clásico en lenguajes de script, cada módulo está compuesto por procedimientos que son utilizables en una o varias clases. Se definen unos atributos (o variables) para uno o varios módulos. Los scripts son modificables totalmente o en parte por el usuario. Los módulos del sistema son directamente accesibles. Ciertos módulos gestionan las funciones de base del robot (ALMemory para su memoria, ALMotion para las funciones de movimiento, ALLeds para los LED, etc.). El módulo ALMotion propone una interfaz (en el sentido de interfaz de programación) para desplazar el robot. Se refieren a ella unos procedimientos de muy alto nivel, como “marcha de 1 m a la derecha adelante”, o más bajo nivel: “hacer moverse la articulación ‘CabecCabeza’ —“Cabeceo de la cabeza”— hasta 30 grados en 1,5 segundos, utilizando una interpolación para alisar el movimiento”. El módulo ALLeds da acceso a todos los LED del robot; se puede de ese modo jugar acerca de la intensidad, el color (cuando el LED puede cambiar de color) de cada LED o de un grupo de LED (todos ellos del ojo izquierdo de una vez, por ejemplo). Hay también unos procedimientos de más alto nivel, que permiten hacer girar los LED de un grupo (como los LED de las orejas, por ejemplo).

El sistema de edición y de control de los movimientos de un robot móvil de la invención puede implantarse sobre un ordenador personal comercial que dispone de un sistema operativo clásico (Windows, Mac, Unix). El o los lenguajes de script utilizados deben implantarse en el ordenador. Si el sistema se utiliza para controlar directamente un robot físico, es necesario un enlace o bien por radio de tipo Wi-Fi o bien por cable. Estos enlaces pueden utilizarse igualmente para recuperar en una red local o distante unos scripts de comportamiento intercambiados con otros usuarios. El robot virtual del sistema de edición y de controles es la imagen de los robots físicos a controlar por el sistema que deberá estar en condiciones de interpretar y de ejecutar los controles recibidos del sistema.
Un mismo sistema de edición y de control según la invención puede controlar varios robots. Ciertos de estos robots pueden tener unos aspectos físicos diferentes —humanoide, animal, etc.— e igualmente unas estructuras diferentes (número de grados de libertad diferentes; robots simples que no ejecutan más que unos movimientos y otros robots que pueden ejecutar movimientos y comportamientos). Los robots adecuados para ser controlados por el sistema de la invención deben disponer sin embargo de las interfaces de comunicación necesarias y de módulos de software adecuados para ejecutar los scripts de las Cajas del sistema de edición y de control según la invención. Los ejemplos descritos en el presente documento anteriormente se dan a título de ilustración de modos de realización de la invención. No limitan de ninguna forma el campo de la invención que se define por las reivindicaciones que siguen.

REIVINDICACIONES

1. Procedimiento de edición de un programa de control de una pluralidad de acciones de un robot móvil que pertenece al grupo de las acciones de tipo comportamiento y de las acciones de tipo movimiento, perteneciendo dicha pluralidad de acciones a al menos una Trama (50, 60) principal de acciones, comprendiendo dicho procedimiento:

- una etapa de conexión de dicha al menos una Trama principal de acciones a al menos un evento elegido en el grupo de los eventos antecedentes y de los eventos sucesores, perteneciendo dicha al menos una Trama principal de acciones a una Caja (10) de control; y
- una etapa (20) de definición de una Timeline de la Caja de control aplicable a dicha al menos una Trama principal de acciones, representando dicha Timeline una restricción temporal a la que se someten las acciones del robot definidas en la Caja de control en la que se inserta dicha Timeline y realizando una sincronización de las acciones de la Caja de control;

estando dicho procedimiento **caracterizado porque** comprende además una etapa de toma en consideración de un parámetro de velocidad de desarrollo de la Timeline definido por un número de Tramas por unidad de tiempo, siendo dicho parámetro o bien definido por un usuario o bien fijado a un valor dado.

2. Procedimiento de edición según la reivindicación 1, **caracterizado porque** dicha Trama principal de acciones pertenece a al menos una Capa (30, 40) de acción, perteneciendo dicha Capa de acción a dicha Caja de control.

3. Procedimiento de edición según una de las reivindicaciones 1 a 2, **caracterizado porque** dicha Trama principal de acciones es de tipo comportamiento y se descompone en comportamientos elementales, definido cada uno por al menos una Caja de control, definiendo dichas Cajas de control dichos comportamientos elementales y estando conectadas entre sí en el seno de al menos un Diagrama (70) de flujo.

4. Procedimiento de edición según la reivindicación 3, **caracterizado porque** se desarrollan al menos dos Tramas principales de comportamiento según la misma Timeline y comprendiendo cada una al menos una Caja de control de la primera Trama principal de comportamiento conectada a al menos una Caja de control de la segunda Trama principal de comportamiento.

5. Procedimiento de edición según la reivindicación 3, **caracterizado porque** al menos una de dichas Cajas de control comprende el menos una segunda Caja de control.

6. Procedimiento de edición según la reivindicación 3, **caracterizado porque** al menos una de las Cajas de control comprende un script (90) directamente interpretable por el robot, siendo dicho script modificable directamente por el usuario.

7. Procedimiento de edición según la reivindicación 6, **caracterizado porque** al menos una salida de al menos una primera Caja de control se une a al menos una entrada de al menos una segunda Caja de control mediante un conector que transporta una señal representativa de un evento.

8. Procedimiento de edición según una de las reivindicaciones 1 a 7, **caracterizado porque** el robot móvil comprende una cabeza y hasta cuatro miembros.

9. Procedimiento de edición según la reivindicación 8, **caracterizado porque** una de dichas acciones es un movimiento de al menos una parte del robot elegida entre el grupo que comprende al menos la cabeza y hasta cuatro miembros y definido por un giro de al menos un motor.

10. Procedimiento de edición según la reivindicación 9, **caracterizado porque** el al menos un giro del al menos un motor se define por una acción del usuario sobre un cursor posicionado sobre una pantalla de movimiento (80) que representa la parte del robot a poner en movimiento.

11. Programa de ordenador que comprende unas instrucciones de código de programa configurado para ejecutar el procedimiento según una de las reivindicaciones 1 a 10 cuando el programa se ejecuta en un ordenador, estando adaptado dicho programa para la edición en un lenguaje interpretable o compilable de una pluralidad de acciones de un robot móvil que pertenece al grupo de los comportamientos y de los movimientos.

12. Programa de ordenador que comprende unas instrucciones de código de programa, ejecutables dichas instrucciones por al menos un ordenador para simular una pluralidad de acciones de un robot móvil o mediante un robot móvil para controlar una pluralidad de acciones de dicho robot móvil, perteneciendo dichas acciones al grupo de las acciones de tipo comportamiento y de las acciones de tipo movimiento, perteneciendo dichas acciones a al menos una Trama (50, 60) principal de acciones, generado dicho programa de ordenador por un programa de ordenador en lenguaje interpretable o compilable según la reivindicación 11.

13. Robot móvil configurado para ejecutar un programa de ordenador según la reivindicación 12.

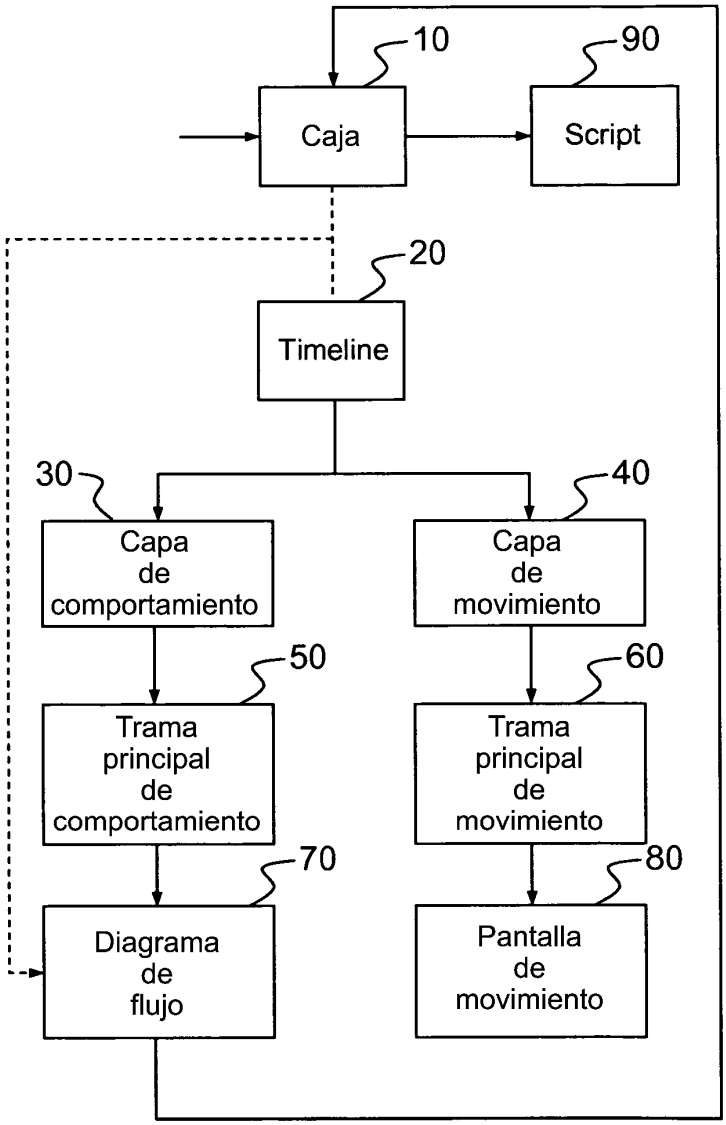
10
90
Caja
Script
20
Timeline
30
40
Capa de comportamiento
Capa de movimiento
50
60
Trama principal de comportamiento
Trama principal de movimiento
70
80
Diagrama de flujo
Pantalla de movimiento

FIG.1

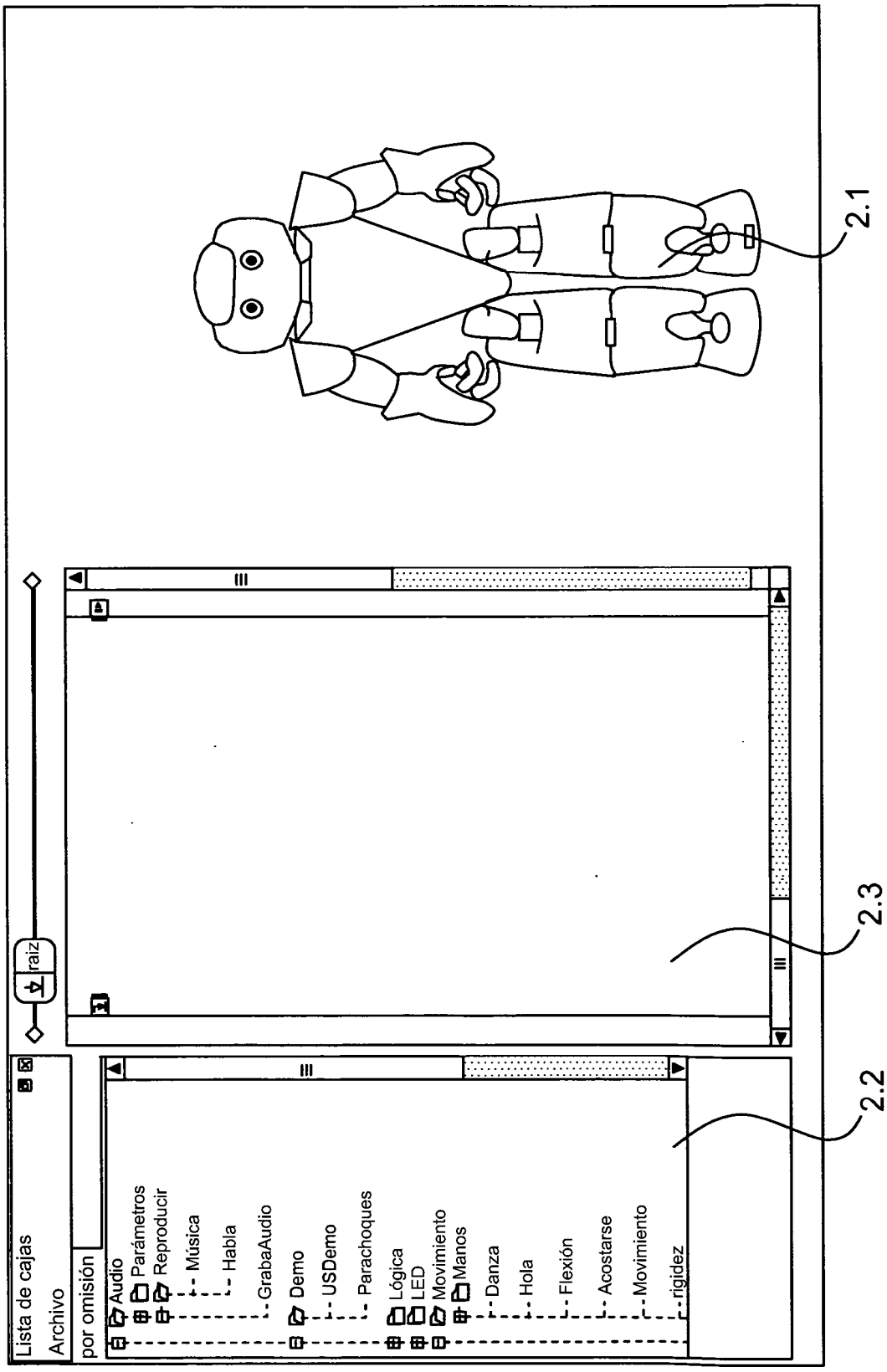
Lista de cajas
Archivo
raiz
por omisión
Audio
Parámetros
Reproducir
Música
Habla
GrabaAudio
Demo
USDemo
Parachoques
Lógica
LED
Movimiento
Manos
Danza
Hola
Flexión
Acostarse
Movimiento
rigidez
2.1
2.2
2.3

FIG.2

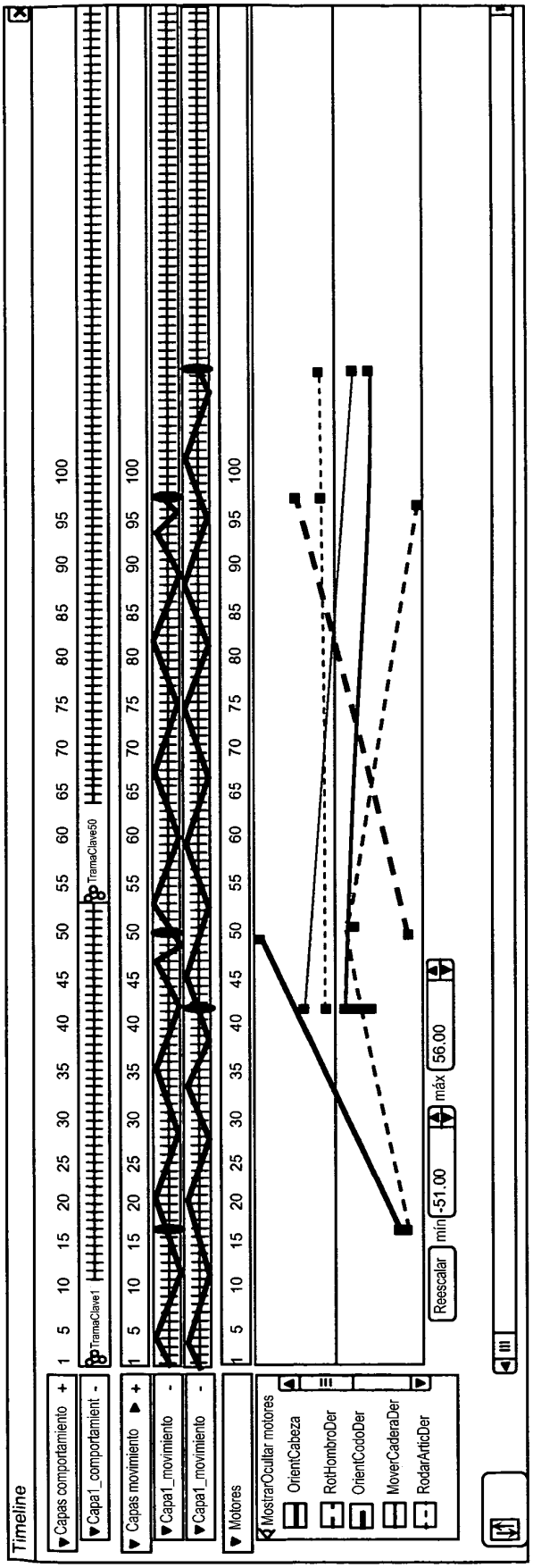
Timeline
Capas comportamiento
Capa1_comportamient
Capas movimiento
Capa1_movimiento
Capa1_movimiento
Motores
TramaClave1
TramaClave50
MostrarOcultar motores
OrientCabeza
RotHombroDer
OrientCodoDer
MoverCaderaDer
RodarArticDer
Reescalar
mín -51.00
máx 56.00

FIG.3

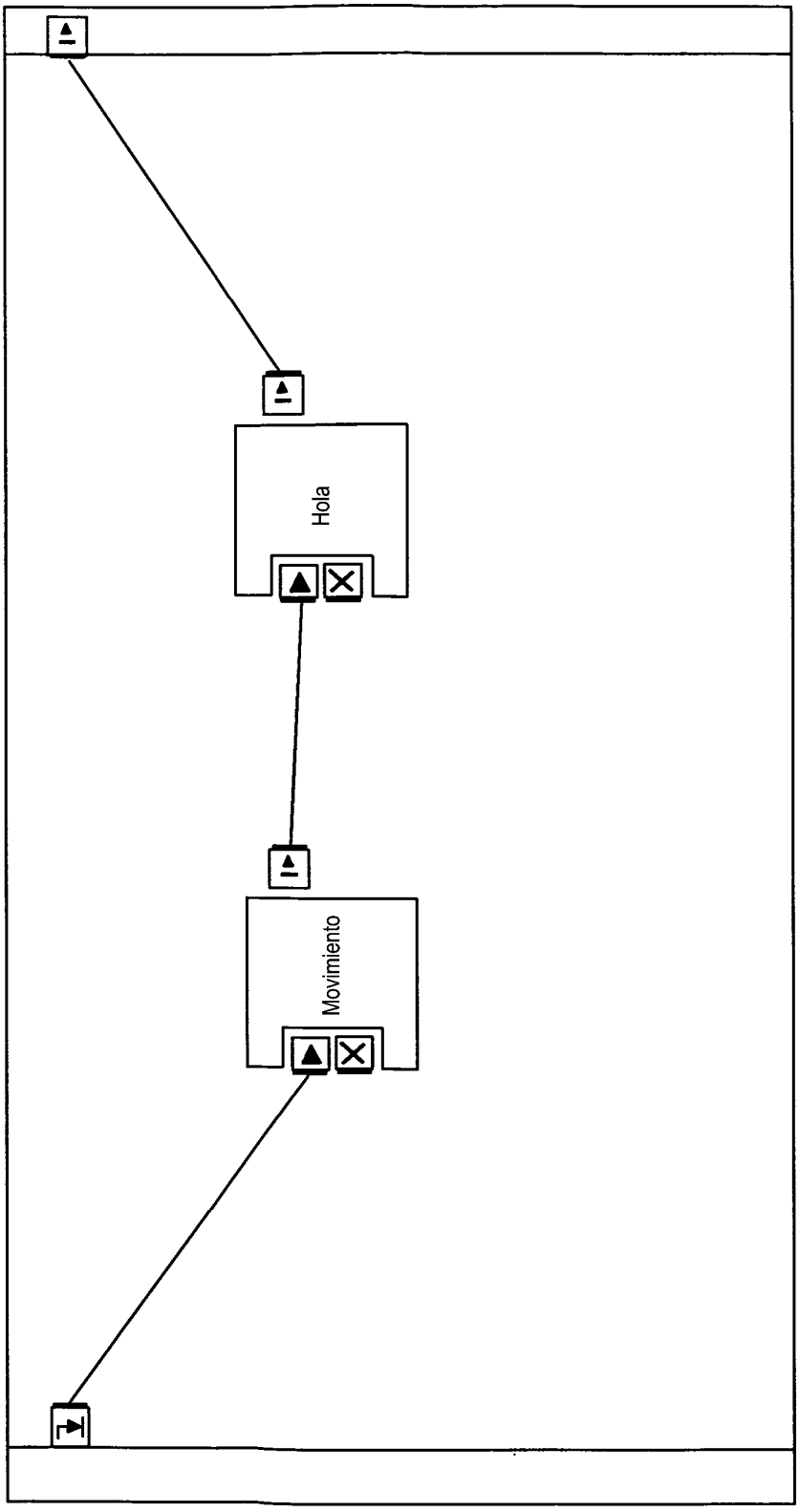

FIG.4

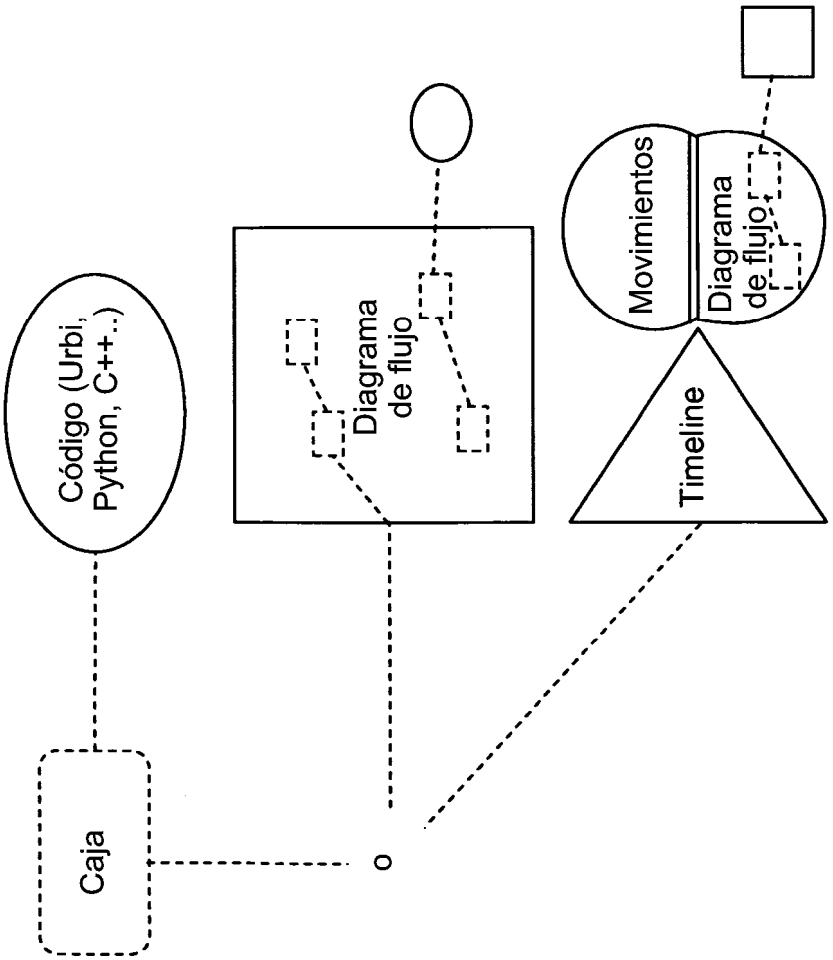
Código (Urbi, Python, C++..)
Caja
Diagrama de flujo
Movimientos
Diagrama de flujo
Timeline

FIG.5

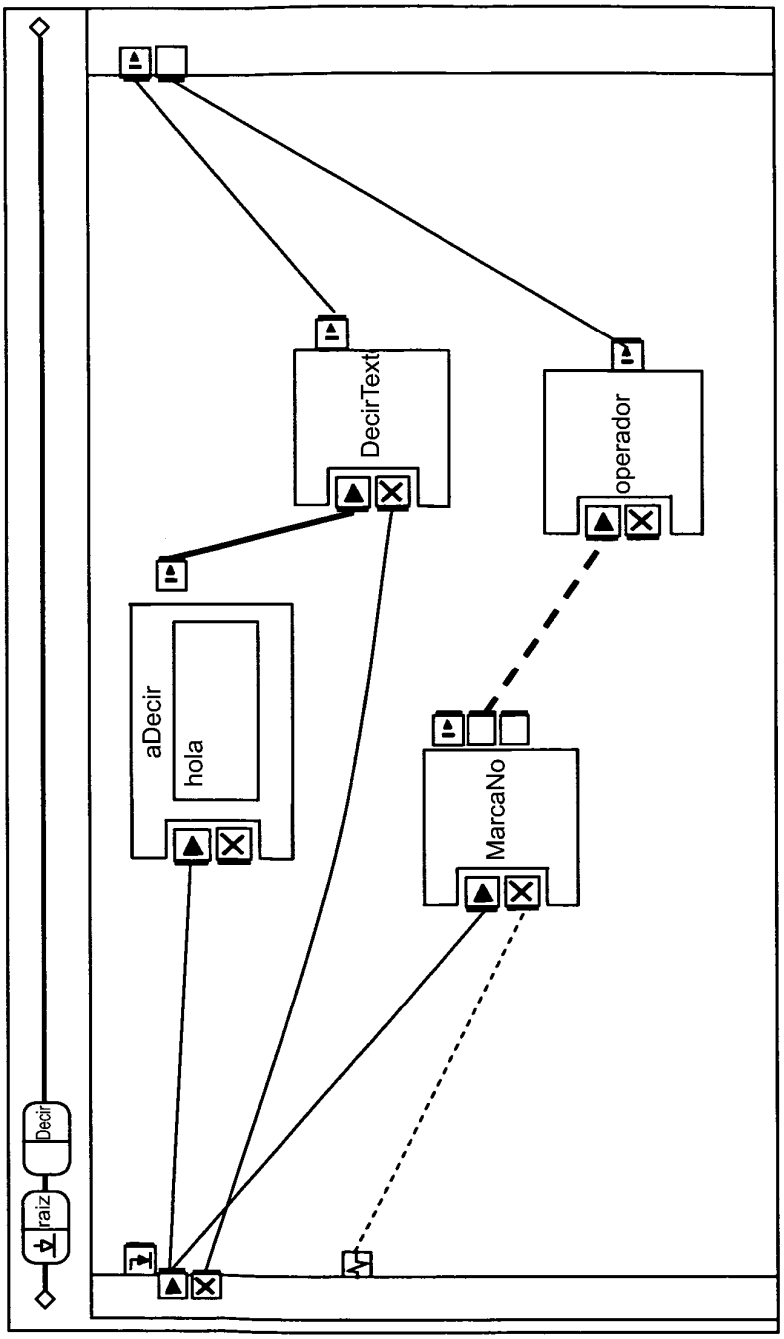

FIG.6

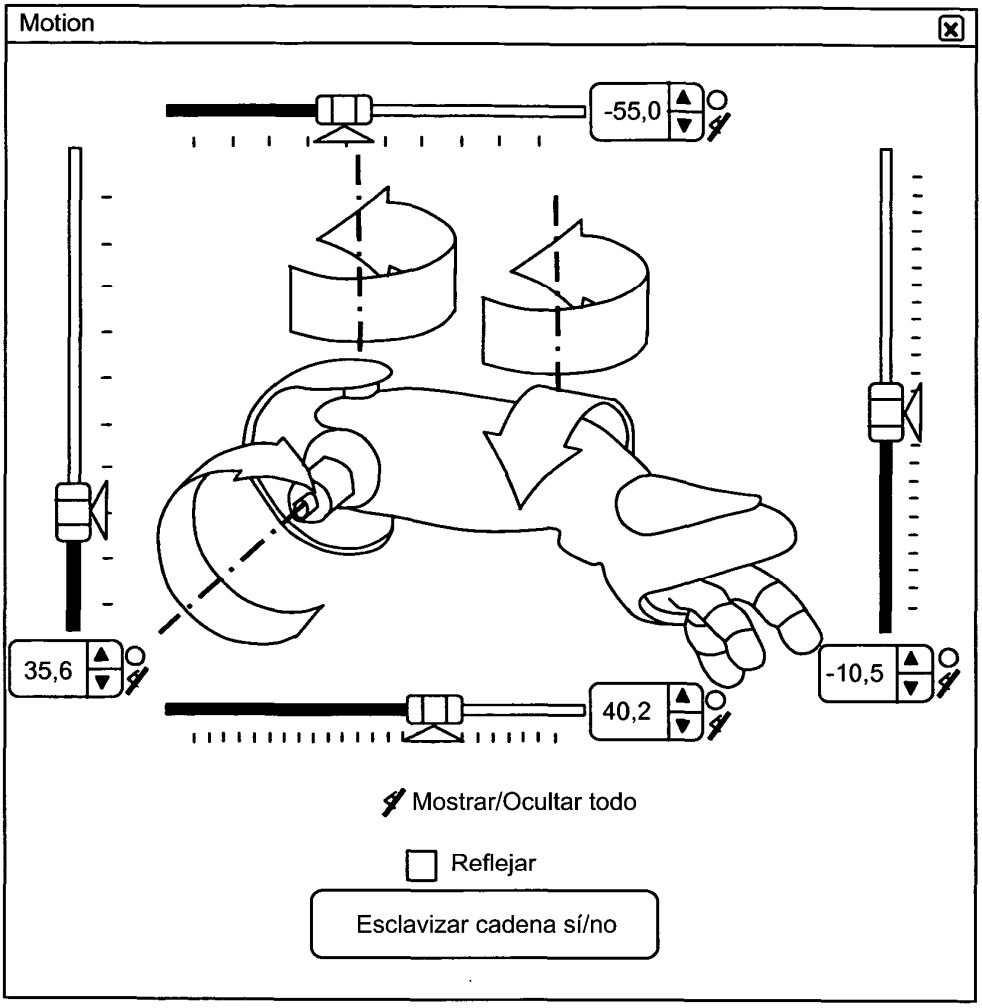
Motion
-55,0
35,6
-10,5
40,2
Mostrar/Ocultar todo
Reflejar
Esclavizar cadena sí/no

FIG.7

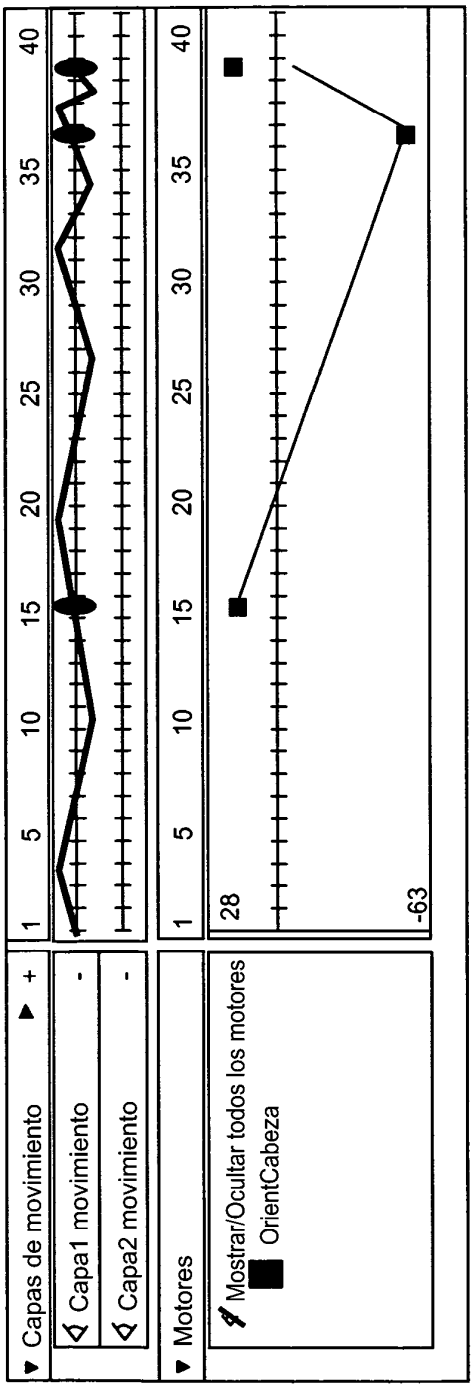
Capas de movimiento
Capa1 movimiento
Capa2 movimiento
Motores
Mostrar/Ocultar todos los motores
OrientCabeza
1 5 10 15 20 25 30 35 40
28
-63

FIG.8a

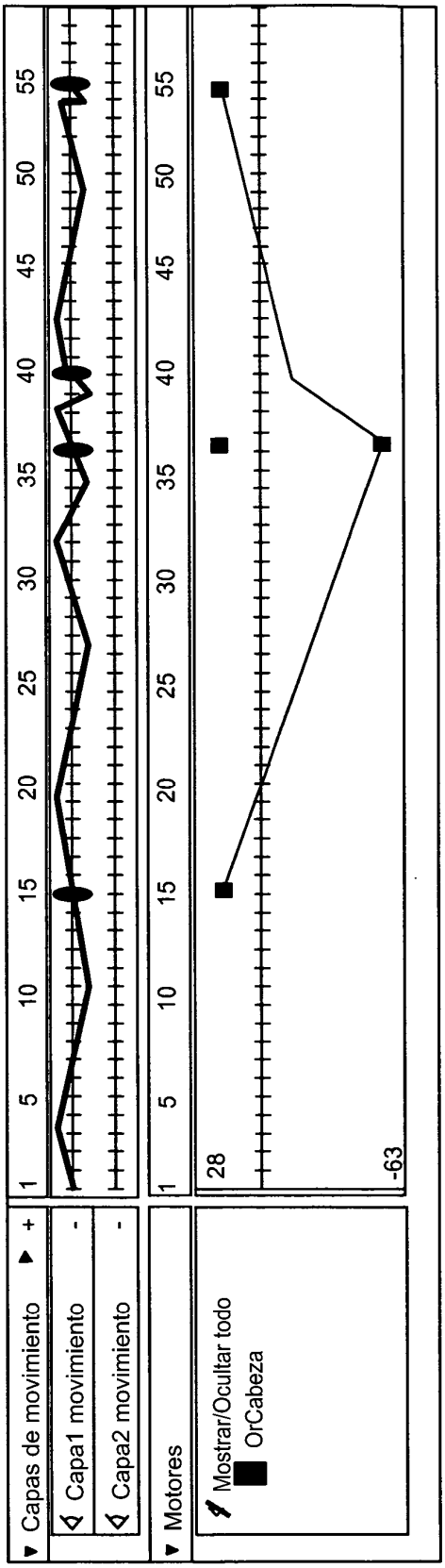
Capas de movimiento
Capa1 movimiento
Capa2 movimiento
Motores
Mostrar/Ocultar todo
OrCabeza
1 5 10 15 20 25 30 35 40 45 50 55
28
-63

FIG.8b

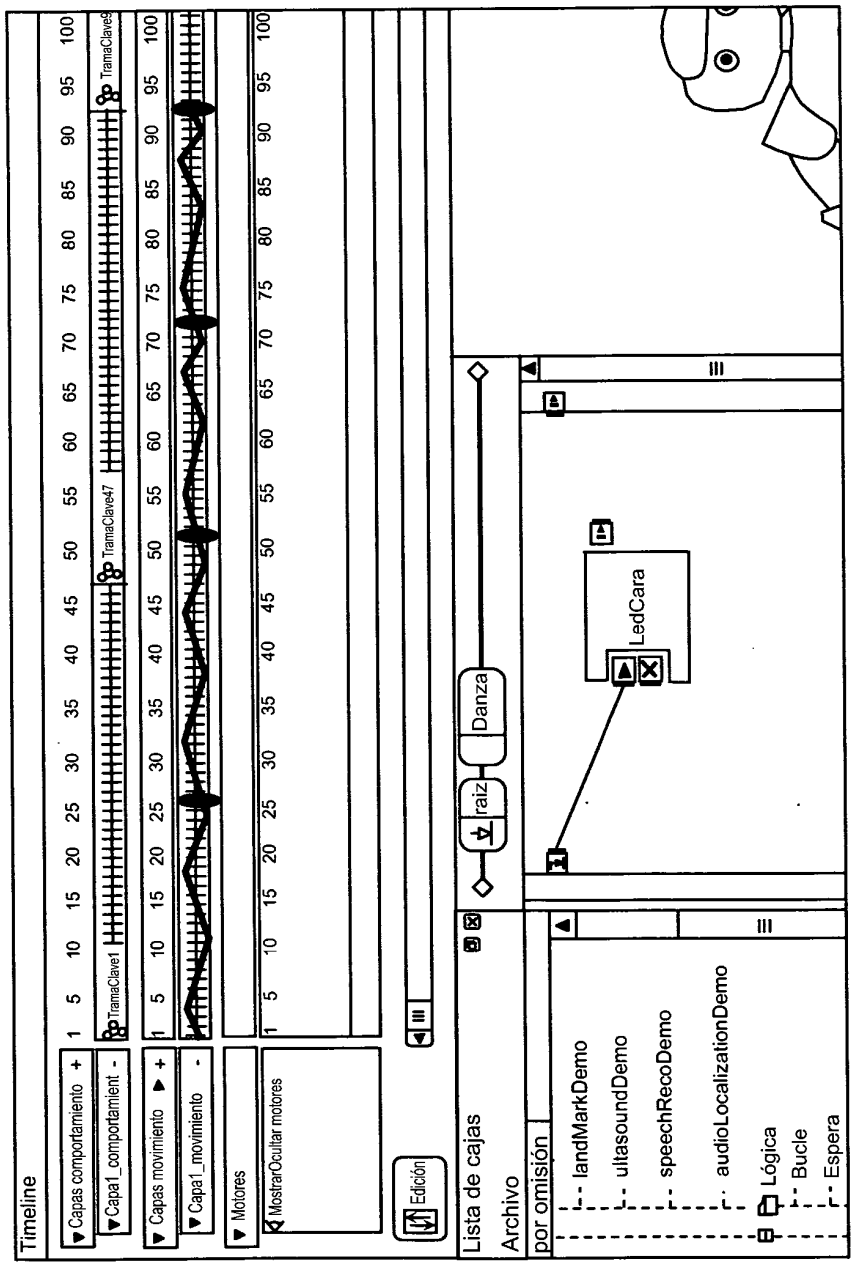
Timeline
Capas comportamiento
Capa1_comportamient
Capas movimiento
Capa1_movimiento
Motores
MostrarOcultar motores
TramaClave1
TramaClave47
TramaClave9
Edición
Lista de cajas
Archivo
por omisión
landMarkDemo
ultasoundDemo
speechRecoDemo
audioLocalizationDemo
Lógica
Bucle
Espera
raiz
Danza
LedCara

FIG.9a

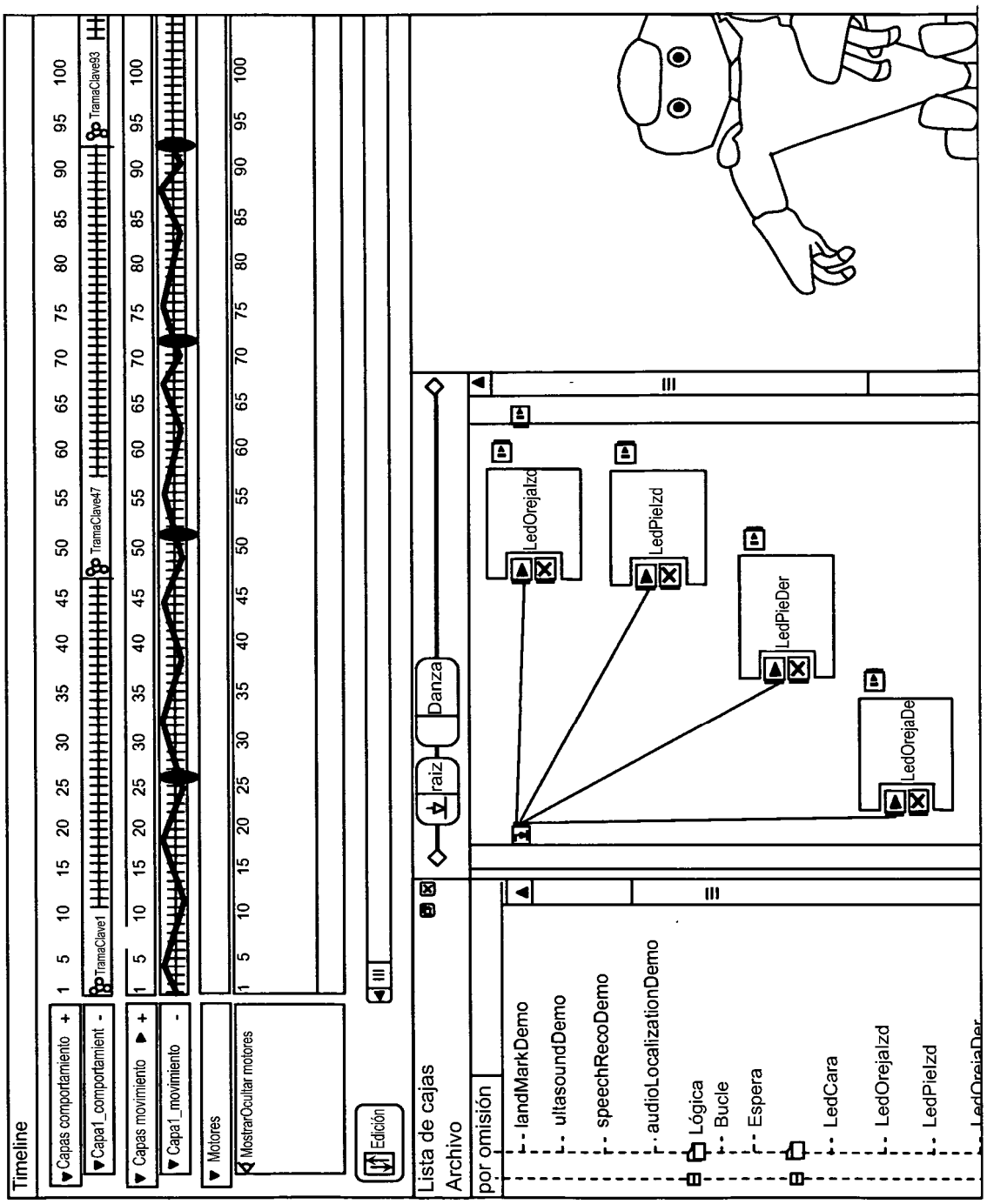
Timeline
Capas comportamiento
Capa1_comportamient
Capas movimiento
Capa1_movimiento
Motores
MostrarOcultar motores
Edición
Lista de cajas
Archivo
por omisión
raiz
Danza
landMarkDemo
ultasoundDemo
speechRecoDemo
audioLocalizationDemo
Lógica
Bucle
Espera
LedCara
LedOrejaIzd
LedPielzd
LedPieDer
LedOrejaDe

FIG.9b

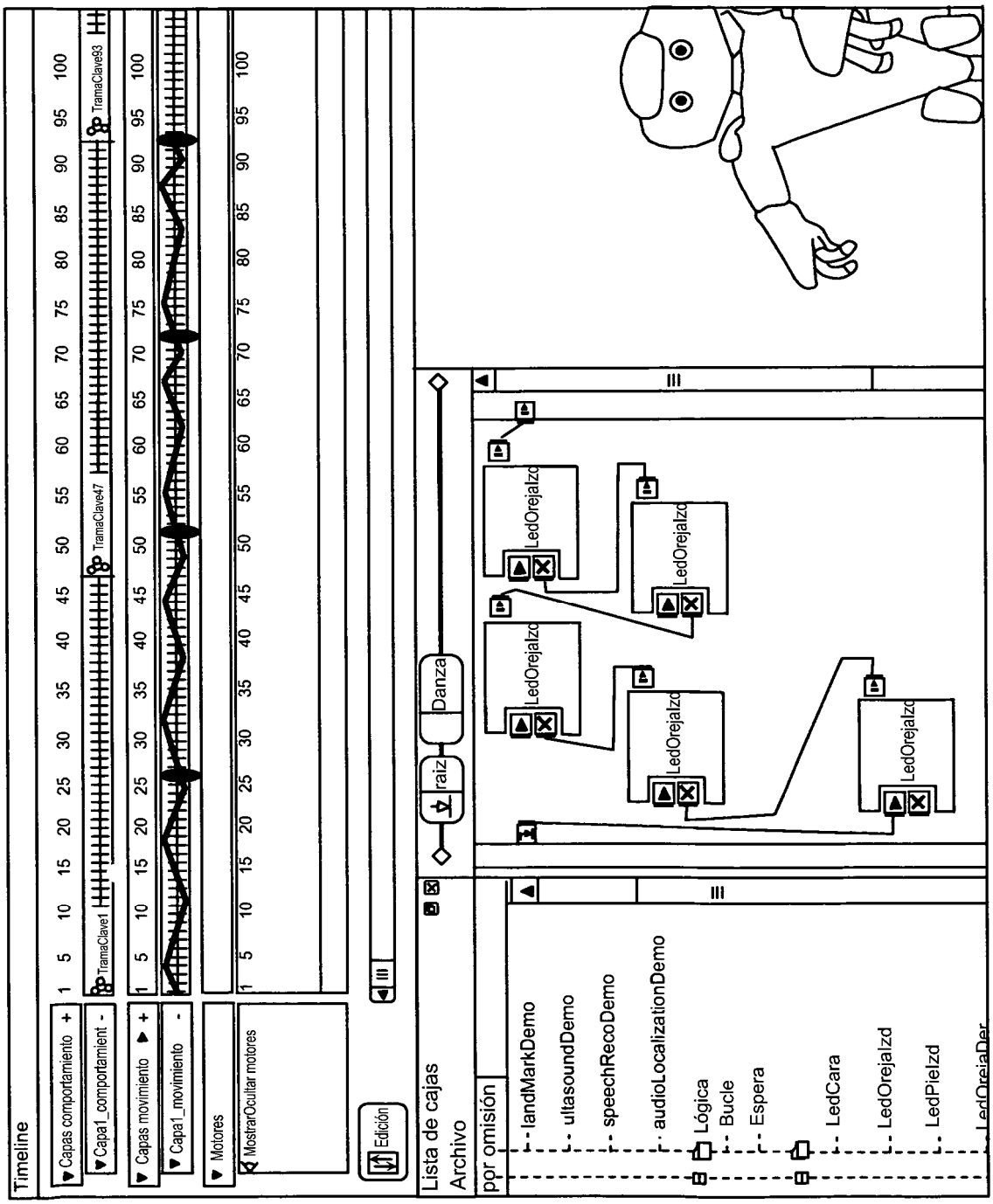
Timeline
Capas comportamiento
Capa1_comportamient
Capas movimiento
Capa1_movimiento
Motores
MostrarOcultar motores
TramaClave1
TramaClave47
TramaClave93
Edición
Lista de cajas
Archivo
por omisión
raiz
Danza
landMarkDemo
ultasoundDemo
speechRecoDemo
audioLocalizationDemo
Lógica
Bucle
Espera
LedCara
LedOrejaIzd
LedPielzd
LedOrejaDer
LedOrejaIzq

FIG.9c

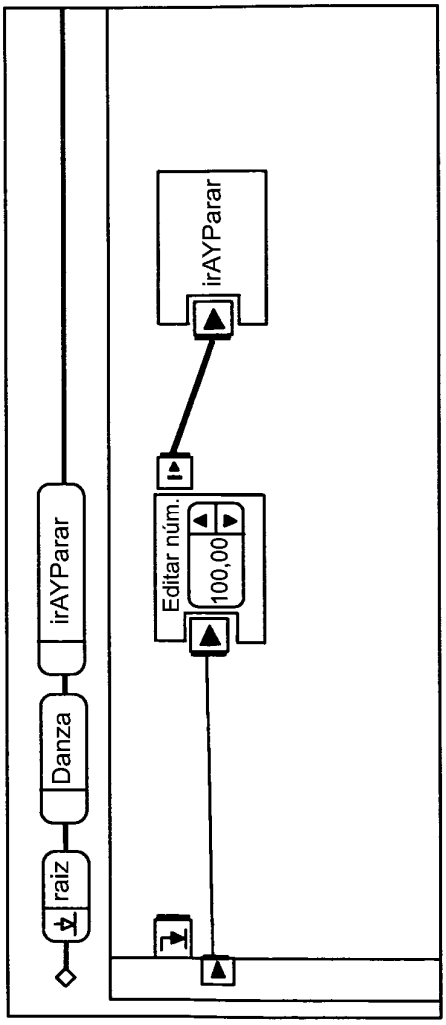

FIG.10a

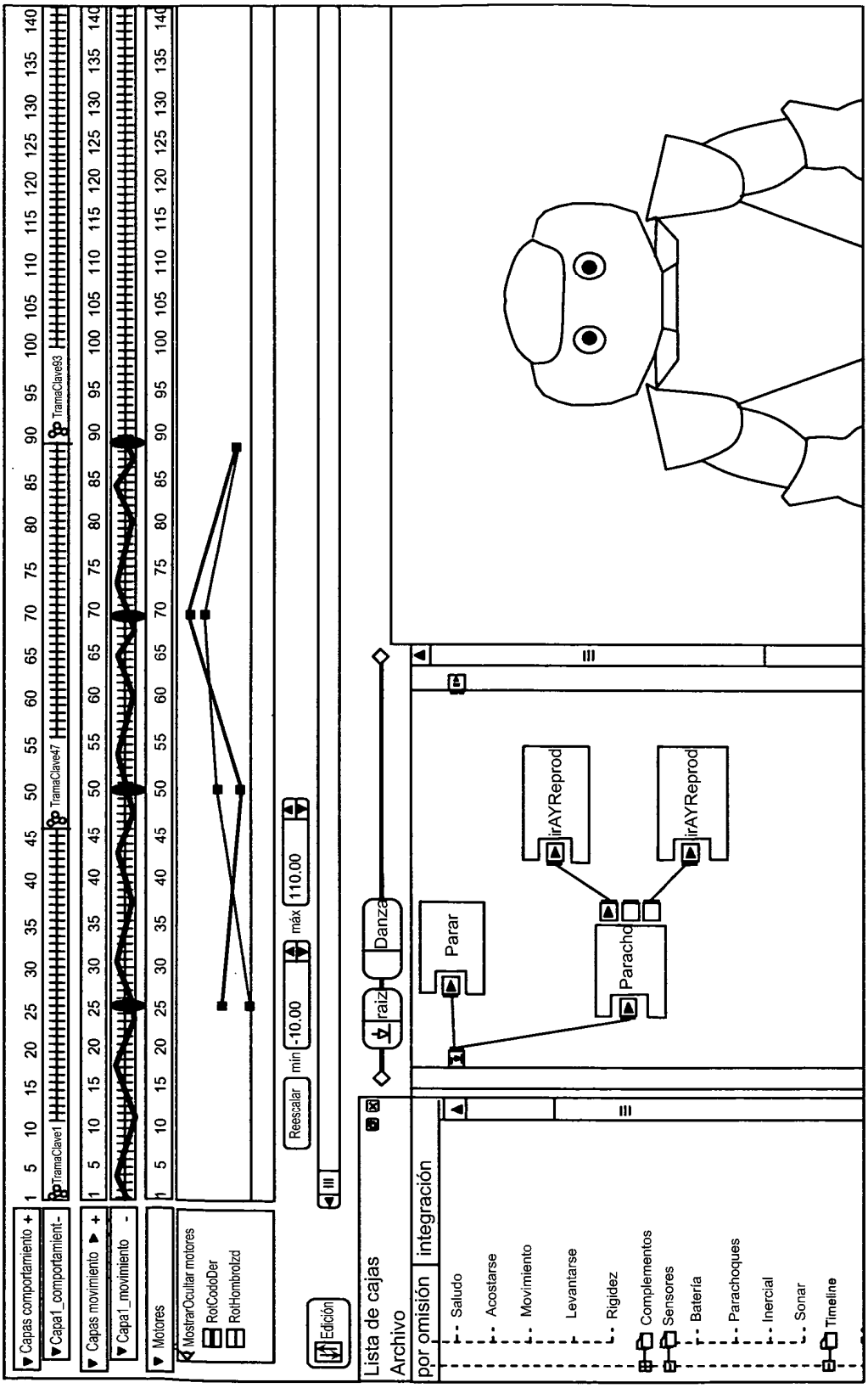
Capas comportamiento
Capa1_comportamient
Capas movimiento
Capa1_movimiento
Motores
MostrarOcultar motores
RotCodoDer
RotHombroIzd
TramaClave1
TramaClave47
TramaClave93
Reescalar
min
-10.00
máx
110.00
Edición
Lista de cajas
Archivo
raiz
Danza
Parar
Paracho
irAYReprod
irAYReprod
por omisión
integración
Saludo
Acostarse
Movimiento
Levantarse
Rigidez
Complementos
Sensores
Batería
Parachoques
Inercial
Sonar
Timeline

FIG.10b

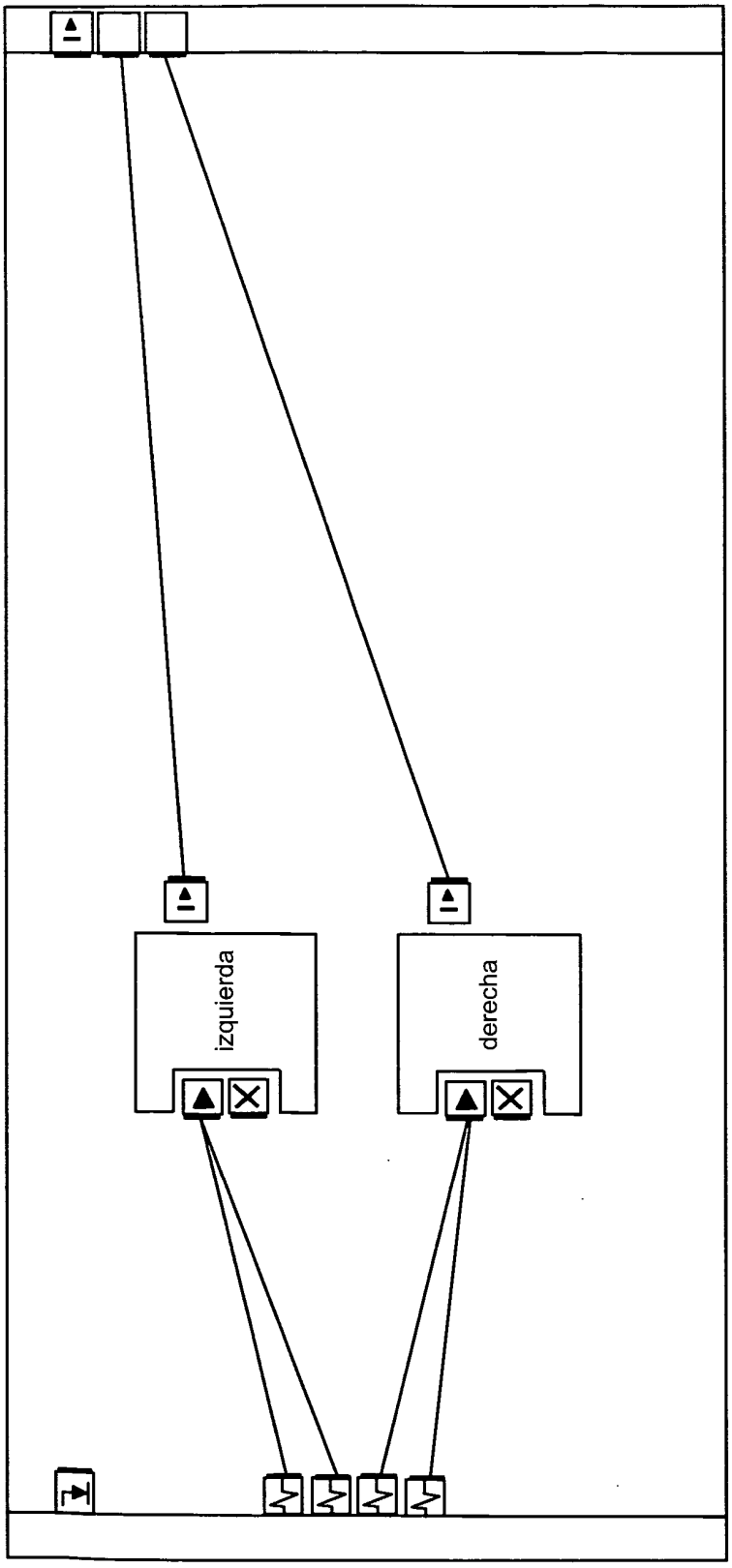

FIG.10c

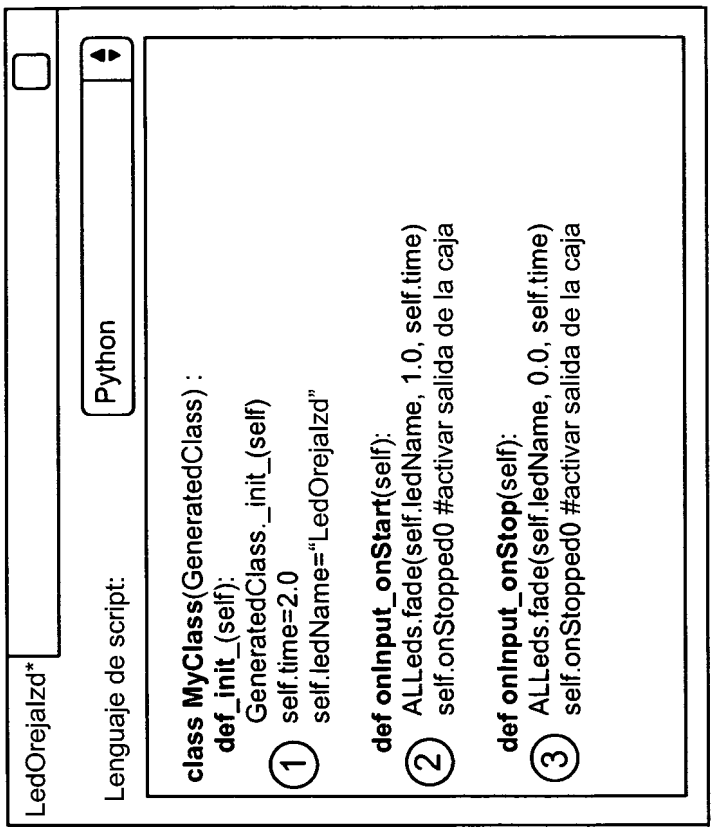
LedOrejalzd*
Lenguaje de script:
Python
class MyClass(GeneratedClass) :
def_init_(self):
GeneratedClass._init_(self)
1
self.time=2.0
self.ledName="LedOrejalzd"
def onInput_onStart(self):
2
ALLeds.fade(self.ledName, 1.0, self.time)
self.onStopped0 #activar salida de la caja
def onInput_onStop(self):
3
ALLeds.fade(self.ledName, 0.0, self.time)
self.onStopped0 #activar salida de la caja

FIG.11