Lenguajes de programación de los Robots

Samuel Candelas Rodríguez UNAM

INTRODUCCIÓN

En las máquinas controladas por sistemas informáticos, el lenguaje es el medio que utiliza el hombre para gobernar su funcionamiento, por lo que su correcta adaptación con la tarea a realizar y la sencillez de manejo, son factores determinantes del rendimiento obtenido en los robots industriales.

Hay varias maneras de comunicarse con un robot, y tres soluciones generales para lograrlo, que son reconocimiento de palabras separadas, enseñanza y repetición y lenguajes de programación de alto nivel.

Los sistemas de reconocimiento de la voz en la tecnología moderna son bastante primitivos y suelen depender de quien habla. Estos sistemas pueden reconocer un conjunto de palabras concretas de un vocabulario muy limitado y en general exigen al usuario una pausa entre las palabras, aunque en la actualidad es posible reconocer las palabras separadas en tiempo real debido a los cada vez más rápidos componentes de las computadoras y algoritmos de procesamiento más eficientes, la utilidad del reconocimiento de palabras separadas para describir la tarea de un robot es bastante limitada.

La enseñanza y repetición, también conocido como guiado, es la solución más común utilizada en el presente para los robots industriales. Este método implica enseñar al robot dirigiéndole los movimientos que el usuario desea que realice. La enseñanza y repetición se lleva a cabo normalmente con los siguientes pasos:

1) dirigiendo al robot con un movimiento lento utilizando el control manual para realizar la tarea completa y grabando los ángulos del movimiento del robot en los lugares adecuados para que vuelva a repetir el movimiento;
2) reproduciendo y repitiendo el movimiento enseñado;
3) si el movimiento enseñado es correcto, entonces se hace funcionar al robot a la velocidad correcta en el modo repetitivo.

Guiar al robot en movimiento lento, puede ser en general llevado a cabo de varias maneras: usando un joystick, un conjunto de botones (uno para cada movimiento) o un sistema de manipulación maestro-esclavo.
Los lenguajes de programación de alto nivel suministran una solución más general para resolver el problema de comunicación hombre-robot. En la década anterior, los robots fueron utilizados con éxito en áreas tales como soldadura por arco voltaico o pintura con spray utilizando el guiado (Engelberger [1980]). Estas tareas no requieren interacción entre el robot y su entorno y pueden ser programadas fácilmente por guiado. Sin embargo, la utilización de robots para llevar a cabo las tareas requieren técnicas de programación en lenguajes de alto nivel ya que el robot de la línea de producción suele confiar en la realimentación de los sensores y este tipo de interacción sólo puede solo puede ser mantenida por métodos de programación que contengan condiciones.

Los lenguajes clásicos empleados en informática, como el FORTRAN, BASIC, PASCAL, etc., no disponen de las instrucciones y comandos específicos que necesitan los robots, para aproximarse a su configuración y a los trabajos que han de realizar. Esta circunstancia, ha obligado a los constructores de robots e investigadores a diseñar lenguajes propios de la Robótica. Sin embargo, los lenguajes desarrollados hasta el momento, se han dirigido a un determinado modelo de manipulador y a una tarea concreta, lo que ha impedido la aparición de lenguajes transportables entre máquinas y por lo tanto de carácter universal.

La estructura del sistema informático del robot varía notablemente, según el nivel y complejidad del lenguaje y de la base de datos que requiera.

CLASIFICACIÓN DE LA PROGRAMACIÓN USADA EN ROBÓTICA

La programación empleada en Robótica puede tener un carácter explícito, en el que el operador es el responsable de las acciones de control y de las instrucciones adecuadas que las implementan, o estar basada en la modelación del mundo exterior, cuando se describe la tarea y el entorno y el propio sistema toma las decisiones.

La programación explícita es la utilizada en las aplicaciones industriales y consta de dos técnicas fundamentales:

1. Programación Gestual.
2. Programación Textual.

La programación gestual consiste en guiar el brazo del robot directamente a lo largo de la trayectoria que debe seguir. Los puntos del camino se graban en memoria y luego se repiten. Este tipo de programación, exige el empleo del manipulador en la fase de enseñanza, o sea, trabaja "on-line".

En la programación textual, las acciones que ha de realizar el brazo se especifican mediante las instrucciones de un lenguaje. En esta labor no participa la máquina (off-line). Las trayectorias del manipulador se calculan matemáticamente con gran precisión y se evita el posicionamiento a ojo, muy corriente en la programación gestual.

Los lenguajes de programación textual se encuadran en varios niveles, según se realice la descripción del trabajo del robot. Se relacionan a continuación, en orden creciente de complejidad:

- Lenguajes elementales, que controlan directamente el movimiento de las articulaciones del manipulador
- Lenguajes dirigidos a posicionar el elemento terminal del manipulador.
- Lenguajes orientados hacia el objeto sobre el que opera el sistema.
- Lenguajes enfocados a la tarea que realiza el robot.

PROGRAMACIÓN GESTUAL O DIRECTA

En este tipo de programación, el propio brazo interviene en el trazado del camino y en las acciones a desarrollar en la tarea de la aplicación. Esta característica determina, inexcusablemente, la programación "on-line".

La programación gestual se subdivide en dos clases:

- Programación por aprendizaje directo.
- Programación mediante un dispositivo de enseñanza.

En el aprendizaje directo, el punto final del brazo se traslada con ayuda de un dispositivo especial colocado en su muñeca, o utilizando un brazo maestro o maniquí, sobre el que se efectúan los desplazamientos que, tras ser memorizados, serán repetidos por el manipulador.

La técnica de aprendizaje directo se utiliza, extensamente, en labores de pintura. El operario conduce la muñeca del manipulador o del brazo maestro, determinando los tramos a recorrer y aquellos en los que la pistola debe expulsar una cierta cantidad de pintura. Con esta programación, los operarios sin conocimientos de "software", pero con experiencia en el trabajo a desarrollar, pueden preparar los programas eficazmente.
La programación por aprendizaje directo tiene pocas posibilidades de edición, ya que, para generar una trayectoria continua, es preciso almacenar o definir una gran cantidad de puntos, cuya reducción origina discontinuidades. El "software" se organiza, aquí, en forma de intérprete.

La programación, usando un dispositivo de enseñanza, consiste en determinar las acciones y movimientos del brazo manipulador, a través de un elemento especial para este cometido. En este caso, las operaciones ordenadas se sincronizan para conformar el programa de trabajo.

El dispositivo de enseñanza suele estar constituido por botones, teclas, pulsadores, luces indicadoras, ejes giratorios o "joystick".

Dependiendo del algoritmo de control que se utilice, el robot pasa por los puntos finales de la trayectoria enseñada. Hay que tener en cuenta que los dispositivos de enseñanza modernos no sólo permiten controlar los movimientos de las articulaciones del manipulador, sino que pueden, también, generar funciones auxiliares, como:

- Selección de velocidades
- Generación de retardos
- Señalización del estado de los sensores
- Borrado y modificación de los puntos de trabajo
- Funciones especiales

Al igual que con la programación directa, en la que se emplea un elemento de enseñanza, el usuario no necesita conocer ningún lenguaje de programación. Simplemente, debe habituarse al empleo de los elementos que constituyen el dispositivo de enseñanza. De esta forma, se pueden editar programas, aunque como es lógico, muy simples.

La estructura del "software" es del tipo intérprete; sin embargo, el sistema operativo que controla el procesador puede poseer rutinas específicas, que suponen la posibilidad de realizar operaciones muy eficientes.

Los lenguajes de programación gestual, además de necesitar al propio robot en la confección del programa, carecen de adaptabilidad en tiempo real con el entorno y no pueden tratar, con facilidad, interaciones de emergencia.

PROGRAMACIÓN TEXTUAL EXPLICITA

El programa queda constituido por un texto de instrucciones o sentencias, cuya confección no requiere de la intervención del robot; es decir, se efectúan "off-line". Con este tipo de programación, el operador no define, prácticamente, las acciones del brazo manipulado, sino que se calculan, en el programa, mediante el empleo de las instrucciones textuales adecuadas.

En una aplicación tal como el ensamblaje de piezas, en la que se requiere una gran precisión, los posicionamientos seleccionados mediante la programación gestual no son suficientes, debiendo ser sustituidos por cálculos más perfectos y por una comunicación con el entorno que rodea al sistema.

En la programación textual, la posibilidad de edición es total. El robot debe intervenir, sólo, en la puesta a punto final.

Según las características del lenguaje, pueden confeccionarse programas de trabajo complejos, con inclusión de saltos condicionales, empleo de bases de datos, posibilidad de creación de módulos operativos intercambiables, capacidad de adaptación a las condiciones del mundo exterior, etc.

Dentro de la programación textual, existen dos grandes grupos, de características netamente diferentes:

- Programación textual explícita.
- Programación textual especificativa.

En la programación textual explícita, el programa consta de una secuencia de órdenes o instrucciones concretas, que van definiendo con rigor las operaciones necesarias para llevar a cabo la aplicación. Se puede decir que la programación explícita engloba a los lenguajes que definen los movimientos punto por punto, similares a los de la programación gestual, pero bajo la forma de un lenguaje formal. Con este tipo de programación, la labor del tratamiento de las situaciones anormales, colisiones, etc., queda a cargo del programador.

Dentro de la programación explícita, hay dos niveles:

1. Nivel de movimiento elemental
2. Nivel estructurado

Nivel de movimiento elemental

Comprende los lenguajes dirigidos a controlar los movimientos del brazo manipulador. Existen dos tipos:

- Articular, cuando el lenguaje se dirige al control de los movimientos de las diversas articulaciones del brazo.
- Cartesiano, cuando el lenguaje define los movimientos relacionados con el sistema de manufactura, es decir, los del punto final del trabajo (TCP).

Los lenguajes del tipo cartesiano utilizan transformaciones homogéneas. Este hecho confiere "popularidad" al programa, independizando a la programación del modelo particular del robot, puesto que un programa confeccionado para uno, en coordenadas cartesianas, puede utilizarse en otro, con diferentes coordenadas, mediante el sistema de transformación correspondiente. Son lenguajes que se parecen al BASIC, sin poseer una unidad formal y careciendo de estructuras a nivel de datos y de control.

Por el contrario, los lenguajes del tipo articular indican los incrementos angulares de las articulaciones. Aunque esta acción es bastante simple para motores de paso a paso y corriente continua, al no tener una referencia general de la posición de las articulaciones con relación al entorno, es difícil relacionar al sistema con piezas móviles, obstáculos, cámaras de TV, etc.

Los lenguajes correspondientes al nivel de movimientos elementales aventaja, principalmente, a los de punto a punto, en la posibilidad de realizar bifurcaciones simples y saltos a subrutinas, así como de tratar informaciones sensoriales.

Nivel estructurado

Intenta introducir relaciones entre el objeto y el sistema del robot, para que los lenguajes se desarrollen sobre una estructura formal.

Se puede decir que los lenguajes correspondientes a este tipo de programación adoptan la filosofía del PASCAL. Describen objetos y transformaciones con objetos, disponiendo, muchos de ellos, de una estructura de datos arborescente.

El uso de lenguajes con programación explícita estructurada aumenta la comprensión del programa, reduce el tiempo de edición y simplifica las acciones encaminadas a la consecución de tareas determinadas.

En los lenguajes estructurados, es típico el empleo de las transformaciones de coordenadas, que exigen un cierto nivel de conocimientos. Por este motivo dichos lenguajes no son populares hoy en día.

PROGRAMACIÓN TEXTUAL ESPECIFICATIVA

Se trata de una programación del tipo no procesal, en la que el usuario describe las especificaciones de los productos mediante una modelización, al igual que las tareas que hay que realizar sobre ellos.

El sistema informático para la programación textual especificativa ha de disponer del modelo del universo, o mundo donde se encuentra el robot. Este modelo será, normalmente, una base de datos más o menos compleja, según la clase de aplicación, pero que requiere siempre, computadoras potentes para el procesado de una abundante información.

El trabajo de la programación consistirá, simplemente, en la descripción de las tareas a realizar, lo que supone poder llevar a cabo trabajos complicados.

Actualmente, los modelos del universo son del tipo geométrico, no físico.

Dentro de la programación textual especificativa, hay dos clases según que la orientación del modelo:

1. Referencia a los objetos
2. Referencia a los objetivos.

Nivel a los objetos

Si el modelo se orienta al nivel de los objetos, el lenguaje trabaja con ellos y establece las relaciones entre ellos. La programación se realiza "off-line" y la conexión CAM es posible.

Dada la inevitable imprecisión de los cálculos del ordenador y de las medidas de las piezas, se precisa de una ejecución previa, para ajustar el programa al entorno del robot.

Los lenguajes con un modelo del universo orientado a los objetos son de alto nivel, permitiendo expresar las sentencias en un lenguaje similar al usado comúnmente.

Nivel a los objetivos

Por otra parte, cuando el modelo se orienta hacia los objetivos, se define el producto final.

La creación de lenguajes de muy alto nivel transferirá una gran parte del trabajo de programación, desde el usuario hasta el sistema informático; éste resolverá la mayoría de los problemas, combinando la Automática y la Inteligencia Artificial.

LENGUAJES DE PROGRAMACIÓN GESTUAL PUNTO A PUNTO

Se aplican con el robot "in situ", recordando a las normas de funcionamiento de un magnetofón doméstico, ya que disponen de unas instrucciones similares: PLAY (reproducir), RECORD (grabar), FF (adelantar), FR (atrasar), PAUSE, STOP, etc. Además, puede disponer de instrucciones auxiliares, como INSERT (insertar un punto o una operación de trabajo) y DELETE (borrar).

Conceptualmente, el manipulador en línea funciona como un digitalizador de posiciones.

Los lenguajes más conocidos en programación gestual punto a punto son el FUNKY, creado por IBM para uno de sus robots, y el T3, original de CINCINNATI MILACROM para su robot T3.

En el lenguaje FUNKY se usa un mando del tipo "joystick" para el control de los movimientos, mientras que el T3 dispone de un dispositivo de enseñanza ("teach pendant").

Como en un grabador de cassettes, y en los dos lenguajes mencionados, los movimientos pueden tener lugar en sistemas de coordenadas cartesianas, cilíndricas o de unión, siendo posible insertar y borrar las instrucciones que se desee. Es posible, también, implementar funciones relacionadas con sensores externos, así como revisar el programa paso a paso, hacia delante y hacia atrás.

El lenguaje FUNKY dispone de un comando especial para centrar a la pinza sobre el objeto.

El procesador usado en T3 es el AMD 2900 ("bit slice"), mientras que en el FUNKY está constituido por el IBM SYSTEM-7.

LENGUAJES DE PROGRAMACIÓN A NIVEL DE MOVIMIENTOS ELEMENTALES.

Como ya menciono, se tratan los movimientos de punto a punto, expresados en forma de lenguaje. Se citan, entre los más importantes:

- ANORAD
- EMILY
- RCL
- RPL
- SIGLA
- VAL
- MAL

Todos ellos mantienen el énfasis en los movimientos primitivos, ya sea en coordenadas articulares, o cartesianas. En comparación, tienen, como ventajas destacables, los saltos condicionales y a subrutina, además de un aumento de las operaciones con sensores, aunque siguen manteniendo pocas posibilidades de programación "off-line".

Estos lenguajes son, por lo general, del tipo intérprete, con excepción del RPL, que tiene un compilador. La mayoría dispone de comandos de tratamiento a sensores básicos: tacto, fuerza, movimiento, proximidad y presencia. El RPL dispone de un sistema complejo de visión, capaz de seleccionar una pintura y reconocer objetos presentes en su base de datos.

Los lenguajes EMILY y SIGLA son transportables y admiten el proceso en paralelo simple.

Otros datos interesantes de este grupo de lenguajes son los siguientes:

ANORAD.- Se trata de una transformación de un lenguaje de control numérico de la casa ANORAD CORPORATION, utilizado para robot ANOMATIC. Utiliza, como procesador, al microprocesador 68000 de Motorola de 16/32 bits.

VAL.- Fue diseñado por UNIMATION INC para sus robots UNIMATE y PUMA. Emplea, como CPU, un LSI-II, que se comunica con procesadores individuales que regulan el servocontrol de cada articulación. Las instrucciones, en idioma inglés, son sencillas e intuitivas, como se puede apreciar por el programa siguiente:

```
LISPT
PROGRAM PICKUP
    1. APRO PART, 25.0
    2. MOVES PART
    3. CLOSE, 0.0.0
    4. APRO PART, -50.0
    5. APRO DROP, 100.0
    6. MOVES DROP
    7. OPEN, 0.0.0
    8. APRO DROP, -100.0
    .END
```

RPL.- Dotado con un LSI-II como procesador central, y aplicado a los robots PUMA, ha sido diseñado por SRI INTERNATIONAL.

EMILY.- Es un lenguaje creado por IBM para el control de uno de sus robots. Usa el procesador IBM 370/145 SYSTEM 7 y está escrito en Ensamblador.

SIGLA.- Desarrollado por OLIVETTI para su robot SUPER SIGMA, emplea un mini-ordenador con 8 K de memoria. Escrito en Ensamblador, es del tipo intérprete.

MAL.- Se ha creado en el Politécnico de Milán para el robot SIGMA, con un Mini-multiprocesador. Es un lenguaje del tipo intérprete, escrito en FORTRAN.

RCL.- Aplicado al robot PACS y desarrollado por RPI, emplea, como CPU, un PDP 11/03. Es del tipo intérprete y está escrito en Ensamblador.

LENGUAJES ESTRUCTURADOS DE PROGRAMACIÓN EXPLÍCITA

Teniendo en cuenta las importantísimas características que presenta este tipo de programación, merecen destacarse los siguientes lenguajes:

- AL
- HELP
- MAPLE
- PAL
- MCL
- MAL EXTENDIDO

Un sencillo ejemplo, de carácter didáctico, utilizando el lenguaje AL, puede mostrar el interés del control estructurado. Partiendo de la definición de unos objetos, se puede lograr una estructura superior que los relacione.

Supongamos que se dispone de los objetos 01 y 02, y se intenta colocar al primero encima del segundo.

Con referencia en la figura, 01T y 01B señalan, respectivamente, la parte superior e inferior del objeto 01, mientras que 01AS indica su posición de asimiento. Las partes del objeto 02 se denominan de la misma forma.

Un programa "orientativo", en AL, que coloque 01 sobre 02, podría ser:

MOVE ARM TO 01AS	El brazo se desplaza hasta la posición de asimiento de 01.
GRASP	Aprehende a 01.
AFFIX 01B TO ARM	Fija el sistema de coordenadas de 01 con el de la pinza del brazo.
MOVE 01B TO 02T	Mueve la parte inferior de 01 hasta la parte superior de 02.
RELEASE	Suelta 01 sobre 02.
UNIFIX 01	Destruye la relación entre el sistema de coordenadas del brazo y 01.

Con excepción de HELP, todos los lenguajes de este grupo están provistos de estructuras de datos del tipo complejo. Así, el AL utiliza vectores, posiciones y transformaciones; el PAL usa, fundamentalmente, transformaciones y el MAPLE permite la definición de puntos, líneas, planos y posiciones.

Sólo el PAL, y el HELP carecen de capacidad de adaptación sensorial. Los lenguajes AL, MAPLE y MCL, tienen comandos para el control de la sensibilidad del tacto de los dedos (fuerza, movimiento, proximidad, etc.). Además, el MCL posee comandos de visión para identificar e inspeccionar objetos.

A continuación, se exponen las características más representativas de los lenguajes dedicados a la programación estructurada.

AL.- Trata de proporcionar definiciones acerca de los movimientos relacionados con los elementos sobre los que el brazo trabaja. Fue diseñado por el laboratorio de Inteligencia Artificial de la Universidad de Stanford, con estructuras de bloques y de control similares al ALGOL, lenguaje en el que se escribió. Está dedicado al manipulador de Stanford, utilizando como procesadores centrales, a un PDP 11/45 y un PDP KL-10.

HELP.- Creado por GENERAL ELECTRIC para su robot ALLEGRO y escrito en PASCAL/FORTRAN, permite el movimiento simultáneo de varios brazos. Dispone, asimismo, de un conjunto especial de subrutinas para la ejecución de cualquier tarea. Utilizando como CPU, a un PDP 11.

MAPLE.- Escrito, como intérprete, en lenguaje PL-1, por IBM para el robot de la misma empresa, tiene capacidad para soportar informaciones de sensores externos. Utiliza, como CPU a un IBM 370/145 SYSTEM 7.

PAL.- Desarrollado por la Universidad de Purdure para el manipulador de Stanford, es un intérprete escrito en FORTRAN y Ensamblador, capaz de aceptar sensores de fuerza y de visión. Cada una de sus instrucciones, para mover el brazo del robot en coordenadas cartesianas, es procesada para que satisfaga la ecuación del procesamiento. Como CPU, usan un PDP 11/70.

MCL.- Lo creó la compañía MC DONALL DOUGLAS, como ampliación de su lenguaje de control numérico APT. Es un lenguaje compilable que se puede considerar apto para la programación de robots "off-line".

MAL EXTENDIDO.- Procede del Politécnico de Milán, al igual que el MAL, al que incorpora elementos de programación estructurada que lo potencian notablemente. Se aplica, también, al robot SIGMA.

LENGUAJES DE PROGRAMACIÓN ESPECIFICATIVA A NIVEL OBJETO

En este grupo se encuentran tres lenguajes interesantes:

- RAPT
- AUTOPASS
- LAMA

RAPT.- Su filosofía se basa en definir una serie de planos, cilindros y esferas, que dan lugar a otros cuerpos derivados. Para modelar a un cuerpo, se confecciona una biblioteca con sus rasgos más representativos. Seguidamente, se define los movimientos que ligan a los cuerpos a ensamblar (alinear planos, encajar cilindros, etc.).

Así, si se desea definir un cuerpo C1, se comienza definiendo sus puntos más importantes, por ejemplo:

P1 = < x, 0, 0 >
P2 = < 0, y, 0 >
P3 = < x/2, y, 0 >
P4 = < 0, 0, z >
Si, en el cuerpo, existen círculos de interés, se especifican seguidamente:
C1 = CIRCLE/P2, R;
C2 = CIRCLE/P4, R;
A continuación, se determinan sus aristas:
L1 = L/P1, P2;
L2 = L/P3, P4;

Si, análogamente al cuerpo C1, se define otro, como el C2, una acción entre ambos podría consistir en colocar la cara inferior de C1 alineada con la superior de C2. Esto se escribiría.

AGAINST / BOT / OF C1, TOP / OF C2;

El lenguaje RAPT fue creado en la Universidad de Edimburgo, departamento de Inteligencia Artificial; está orientado, en especial, al ensamblaje de piezas. Destinado al robot FREDY, utiliza, como procesador central, a un PDP 10. Es un intérprete y está escrito en lenguaje APT.

AUTOPASS.- Creado por IBM para el ensamblaje de piezas; utiliza instrucciones, muy comunes, en el idioma inglés. Precisa de un ordenador de varios Megabytes de capacidad de memoria y, además de indicar, como el RAPT, puntos específicos, prevé, también, colisiones y genera acciones a partir de las situaciones reales.

Un pequeño ejemplo, que puede proporcionar una idea de la facilidad de relacionar objetos, es el programa siguiente, que coloca la parte inferior del cuerpo C1 alineada con la parte superior del cuerpo C2. Asimismo, alinea los orificios A1 y A2 de C1, con los correspondientes de C2.

```
PLACE C1
SUCH THAT C1 BOT CONTACTS C2TOP
AND B1 A1 IS ALIGNED WITH C2A1
AND B1 A2 IS ALIGNED WITH C2A2
```

El AUTOPASS realiza todos sus cálculos sobre una base de datos, que define a los objetos como poliedros de un máximo de 20,000 caras. Está escrito en PL/1 y es intérprete y compilable.

LAMA.- Procede del laboratorio de Inteligencia Artificial del MIT, para el robot SILVER, orientándose hacia el ajuste de conjuntos mecánicos.

Aporta más inteligencia que el AUTOPASS y permite una buena adaptación al entorno.
La operatividad del LAMA se basa en tres funciones principales:

1º Creación de la función de trabajo. Operación inteligente.
2º Generación de la función de manipulación.
3º Interpretación y desarrollo, de una forma interactiva, de una estrategia de realimentación para la adaptación al entorno de trabajo.

LENGUAJES DE PROGRAMACIÓN EN FUNCIÓN DE LOS OBJETIVOS

La filosofía de estos lenguajes consiste en definir la situación final del producto a fabricar, a partir de la cual se generan los planes de acción tendentes a conseguirla, obteniéndose, finalmente, el programa de trabajo.

Estos lenguajes, de tipo natural, suponiendo una potenciación extraordinaria de la Inteligencia Artificial, para descargar al usuario de las labores de programación. Prevén, incluso, la comunicación hombre-máquina a través de la voz.

Los lenguajes más conocidos de este grupo son:

- STRIPS
- HILAIRE

STRIPS.- Fue diseñado, en la Universidad de Stanford, para el robot móvil SHAKEY. Se basa en un modelo del universo ligado a un conjunto de planteamientos aritmético-lógicos que se encargan de obtener las subrutinas que conforman el programa final.

Es intérprete y compilable, utilizando, como procesadores, a un PDP-10 y un PDP-15.
HILAIRE.- Procedente del laboratorio de Automática Y Análisis de Sistemas (LAAS) de Toulouse, está escrito en lenguaje LISP. Es uno de los lenguajes naturales más interesantes, por sus posibilidades de ampliación e investigación.

CARACTERÍSTICAS DE UN LENGUAJE IDEAL PARA LA ROBÓTICA

Las seis características básicas de un lenguaje ideal, expuestas por Pratt, son:

1. Claridad y sencillez.
2. Claridad de la estructura del programa.
3. Sencillez de aplicación.
4. Facilidad de ampliación.
5. Facilidad de corrección y mantenimiento.
6. Eficacia.

Estas características son insuficientes para la creación de un lenguaje "universal" de programación en la robótica, por lo que es preciso añadir las siguientes:

- Transportabilidad sobre cualquier equipo mecánico o informático.
- Adaptabilidad a sensores (tacto, visión, etc.).
- Posibilidad de descripción de todo tipo de herramientas acoplables al manipulador.
- Interacción con otros sistemas.

En el aspecto de claridad y sencillez, la programación gestual es la más eficaz, pero impide la confección de programas propiamente dichos. Los lenguajes a nivel de movimientos elementales, como el VAL, disponen de bastantes comandos para definir acciones muy parecidas que fueran surgiendo según las necesidades y que, en gran medida, oscurecen su comprensión y conocimiento.

Aunque, inicialmente, las técnicas de programación estructurada son más difíciles de dominar, facilitan, extraordinariamente, la comprensión y corrección de los programas.

Respecto a la sencillez de aplicación, hay algunos lenguajes (como el MCL) dedicados a las máquinas herramienta (APT), que pueden ser valorados, positivamente, por los usuarios conocedores de este campo. El PAL, estructurado sobre la matemática matricial, sólo es adecuado para quienes están familiarizados con el empleo de este tipo de transformaciones.

Uno de los lenguajes más fáciles de utilizar es el AUTOPASS, que posee un juego de comandos con una sintaxis similar a la del inglés corriente.

Es imprescindible que los lenguajes para los robots sean fácilmente ampliables, por lo que se les debe dotar de una estructura modular, con inclusión de subrutinas definidas por el mismo usuario.

La adaptabilidad a sensores externos implica la posibilidad de una toma de decisiones, algo muy interesante en las labores de ensamblaje. Esta facultad precisa de un modelo dinámico del entorno, así como de una buena dosis de Inteligencia Artificial, como es el caso del AUTOPASS.

Aunque los intérpretes son más lentos que los compiladores, a la hora de la ejecución de un programa, resultan más adecuados para las aplicaciones de la robótica. Las razones son las siguientes:

1. El intérprete ejecuta el código como lo encuentra, mientras que el compilador recorre el programa varias veces, antes de generar el código ejecutable.
2. Los intérpretes permiten una ejecución parcial del programa.
3. La modificación de alguna instrucción es más rápida con intérpretes, ya que un cambio en una de ellas no supone la compilación de las demás.

Finalmente, el camino para la superación de los problemas propios de los lenguajes actuales ha de pesar, necesariamente, por la potenciación de los modelos dinámicos del entorno que rodea al robot, acompañado de un aumento sustancial de la Inteligencia Artificial.

B I B L I O G R A F Í A

- Robótica: Control, Detección, Visión e Inteligencia
  K.S. FU, R.C González, C.S.G. LEE
  McGraw Hill
- Robótica Practica Tecnología y Aplicaciones
  José Ma. Angulo
  Ed. Paraninfo