EL LENGUAJE V+

AUTORES: JAUME YEBRA PEREZ
NURIA LAGOS FERNÁNDEZ

PROFESOR: PERE PONSA

MANERAS DE COMUNICARSE CON EL ROBOT

- SISTEMAS DE RECONOCIMIENTO DE VOZ
- ENSEÑANZA Y REPETICION (GUIADO)
- LENGUAJES DE PROGRAMACION

TIPOS DE PROGRAMACIÓN

- Gestual o Directa (Guiado)
  - Programación por aprendizaje directo
  - Programación mediante un dispositivo de enseñanza (Botonera)
- Textual
  - Textual Explicita
  - Textual especificativa

LENGUAJE	UNIVERSIDAD /FABRICANTE	ORIGEN Y CARACTERÍSTICAS	APLICACIONES
WAVE (1973)	Stanford	- Nivel actuador - Compilador asociado al robot. - Sintaxis simple: MOVE, SEARCH, CENTER. - Objetos tratados: constantes, variables, vectores, contadores	- Manipulación de piezas mecánicas. - Montaje de bomba de agua. - Ensamblaje.
AL (Assembly Language) (1974)	Stanford	- Basado en PASCAL. - Nivel actuador. - Transformador de coordenadas. - Sintaxis compleja: MOVE TO, MOVE VIA, etc., SEARCH, ACROSS SEARCH, WITHOUT..., CENTER ON ... - Objetos tratados: los de Wave más planos, traslaciones, rotaciones. - Estructura Algol.	- Manipulación de piezas mecánicas. - Sistema de desarrollo de programas Pointy. - Robots Unimation.
RAPT (1978)	Sistemas de Toulouse (Universidad de Edimburgo)	- Interprete escrito en APT. - Nivel objeto.	- Montaje
VAL I (1979) VAL II (1983)	Unimation	- Se partió de AL, pero utilizando BASIC. - Nivel actuador. - Transformador de coordenadas. - Instrucciones de movimiento y de control (MOVE-, DEPART-). - instrucciones Aritméticas. - Estructura Algol.	- Robot Puma de Unimation
MCL	Cincinnati Milacron	- Desarrollo por ICAM y realizado en Fortram	
AML (1979) FUNKY MAPLE AUTOPASS (1977)	IBM	- Alto nivel interactivo estructurado. - Nivel objeto. - Compilador al nivel. - Sintaxis evolucionada, como colocar X sobre Y alineando Z y T.	- Corresponde al nivel de descripción de las gamas de montaje actuales.
LRP	ACMA	- Desarrollado en colaboración con la Universidad de Montpellier.	
V+ (1989)	ADEPT	- Lenguaje textual de alto nivel. - Ejecución de varios programas al mismo tiempo (multitarea). - Proceso asíncrono o ejecución de rutinas de reacción ante determinados eventos.	- Usado en Adept y Staübli
RAPID (1994)	ABB	- Lenguaje textual de alto nivel altamente estructurado. - Estructura modular del programa. - Uso de estructuras predefinidas para especificar la configuración del robot y las características de la herramienta.	
KAREL	FANUC	- Basado en PASCAL. - Soporta la multitarea. - Estructura de datos asociados modificable.	
KRL	KUKA	- Basado en el estándar IRL, descrito en la norma DIN 66312, lo que aporta transportabilidad.	

EL LENGUAJE V+

El lenguaje V+ aparece después de reemplazar el lenguaje VAL II, que a su vez, Val II reemplazó a VAL.
El lenguaje V+ proporciona una gran interacción entre el hombre y el robot.
Presenta las siguientes aportaciones más relevantes:

• Inteligibilidad: Nos puede proporcionar una buena documentación, además de un diseño ordenado y coherente del programa.

• Fiabilidad: Sobre todo en sistemas que deben responder a situaciones imprevistas.

• Adaptabilidad: Se pueden mejorar, ampliar y modificar los programas con poco esfuerzo.

• Transportabilidad: Los programas se pueden desarrollar en computadoras (ej: un PC) para poder introducirlos, posteriormente, en el controlador del sistema a través de un disquete.

LAS POSIBILIDADES DEL V+

- Programas de aplicación accesibles al operador mediante mouse, ventanas y gráficos en color.
- Modificación de la trayectoria en tiempo real,
- Acceso a comunicación con el BUS VME.
- Cálculos booleanos, funciones matemáticas, funciones geométricas, etc..
- Funciones gráficas para crear ventanas, iconos, gestión de eventos,...
- Construcción estructurada.
- Movimientos continuos.
-7 tareas paralelas (con posible extensión a 28 tareas).
- Acceso a *256* entradas/ *256* salidas digitales externas, a 28 entradas / 16 salidas analógicas, 4 (ó 5) puertos de comunicación serie.
- Gestión de ejes suplementarios y de encoders externos
- Matrices de hasta 3 dimensiones.
- Variables numéricas de 32 bits, coma flotante, simple y doble precisión.
- Manipulación de cadenas de caracteres. Herramientas de detección de errores.
- Los programas pueden ser escritos off-line en otros ordenadores

TIPOS DIFERENTES DE PROGRAMAS CON V+

•Programas de **control** del robot: Estos programas controlan directamente el robot y pueden contener cualquier instrucción, además de las instrucciones de movimiento del robot.

• Programas de control de propósito **general**: Estos programas no son controlados por el robot, y pueden ejecutarse al mismo tiempo. Pueden controlar procesos externos a través delineas binarias externas y comunicarse con el programa de control mediante variables compartidas y señales de software.

• Programas de comandos del **monitor**: Estos programas se componen de comandos de monitor, más que de instrucciones de programa y, se usan, para realizar secuencias de comandos del monitor.

TABLA DE OPERACIONES DE CONTROL DE PROGRAMAS:

Palabra Clave	Tipo	Función	Estructura
ABORT	Instrucción de Programa	Terminal de ejecución de programa	GOTO label
CALL	Instrucción de Programa	Suspende la ejecución del programa actual y continua la ejecución con un nuevo programa (es decir, un subprograma).	Ej: CALL check_data(locx, locy, length)
CALLS	Instrucción de Programa	Suspende la ejecución del programa actual y continua la ejecución con un nuevo programa	$program_name = $program_list[program_select]
CASE	Instrucción de Programa	Proceso iniciado con una estructura del CASE definiendo el valor de interés.	
CLEAR.EVENT	Instrucción de Programa	Despeja un acontecimiento asociado a la tarea especificada.	
CYCLE.END	Instrucción de Programa	Termina el programa de control especificado la próxima vez que ejecuta una instrucción del programa de PARADA	
DO	Instrucción de Programa	Introduce una estructura DO.	EJ: DO. code block . UNTIL expression
EXECUTE	Instrucción de Programa	Empieza la ejecución de un programa de control.	
EXECUTE	Comando de Monitor	Empieza la ejecución de un programa de control.	
EXIT	Instrucción de Programa	Sale de una estructura de control FOR, DO ...WHILE.	
FOR	Instrucción de Programa	Ejecuta un grupo de instrucciones de programa un cierto número de veces.	Ej: FOR index = start_val TO end_val STEP incr . code block . END
GET.EVENT	Función de valor Real	Devuelve acontecimientos que se fijan para la tarea	
GOTO	Instrucción de Programa	Realiza un parte del programa identificado por la etiqueta	IF logical_expression GOTO 100
HALT	Instrucción de Programa	Para la ejecución de programa	
IF...GOTO	Instrucción de Programa	Mira si el valor de la etiqueta especificada de una expresión lógica es TRUE	
IF...THEN	Instrucción de Programa	Ejecuta un grupo de instrucciones condicionales (o uno de dos grupos) dependiendo del resultado de una expresión lógica.	

TABLA DE OPERACIONES DE CONTROL DE PROGRAMAS:

INT.EVENT	Instrucción de Programa	Envía una instrucción de SET.EVENT a la tarea actual si una interrupción ocurre en un vector especificado del bus de VME.
LOCK	Instrucción de Programa	Fija la prioridad del cierre de la reacción del programa al valor dado.
MCS	Instrucción de Programa	Invoca un comando de control del monitor
NEXT	Instrucción de Programa	Rompe una estructura FOR, DO, o WHILE y comienza la iteración siguiente de la estructura del control.
PAUSE	Instrucción de Programa	Para la ejecución de programa y no permite que el programa sea reasumido.
PRIORITY	Función de valor Real	Devuelve la prioridad actual del cierre de la reacción hacia el programa.
REACT / REACTI	Instrucción de Programa	Inicia el control continuo de una señal numérica especifica y acciona automáticamente una llamada de subprograma si las transiciones de señal son correctas.
REACTE	Instrucción de Programa	Inicia la supervisión de los errores que ocurren durante la ejecución de la tarea del programa actual.
REALASE	Instrucción de Programa	Permite la tarea siguiente disponible del programa para funcionar.
RETRY	Instrucción de Programa	Controla si la PROGRAM START causa un resumens de programa .
RETURN	Instrucción de Programa	Termina la ejecución de la ejecución actual del subprograma y de la ultima ejecución del programa en el siguiente paso de la instruccion
RETURNE	Instrucción de Programa	Termina la ejecución de la ejecución actual del subprograma y de la ultima ejecución del programa en el siguiente paso de la instrucción que causa la subrutina invocada.
RUNSIG	Instrucción de Programa	Enciende o apaga la señal digital mientras la ejecución de la tarea invocada continua.
SET.EVENT	Instrucción de Programa	Fija un acontecimiento asociado a la tarea especificada.
STOP	Instrucción de Programa	Termina la ejecución del ciclo del programa
WAIT	Instrucción de Programa	Pone en el programa un lazo de espera hasta que la condición es TRUE.
WAIT.EVENT	Instrucción de Programa	Suspende la ejecución de un programa hasta que ha ocurrido un acontecimiento especificado, o hasta que ha transcurrido una cantidad de tiempo especificada.
WHILE	Interruptor de sistema	Inicia un proceso con la estructura WHILE si la condición es TRUE o salta la estructura del WHILE si la condición es inicialmente FALSA.

TABLA DE FUNCIONES PARA VALORES NUMÉRICOS

Palabra Clave	Función
ABS	Retorna un valor absoluto
ATAN2	Devuelve el tamaño del ángulo (grados) que tiene la tangente trigonometrica igual a value_1/value_2.
BCD	Convierte un valor real a formato de Código Decimal binario (BCD).
COS	Retorna el cosinos trigonométrico del ángulo dado.
DCB	Convierte digitos BCD a un número entero equivalente.
FRACT	Retorna la parte fraccionada de un argumento.
INT	Retorna la parte entera de un número.
INTB	Retorna el valor de dos bytes de un string interpretado
MAX	Devuelve el valor máximo contenido en la lista de valores.
MIN	Devuelve el valor mínimo contenido en la lista de valores.
OUTSIDE	Comprueba un valor para ver si es exterior al rango especificado.
PI	Devuelve el valor de la constante matemática pi (3,141593).
RANDOM	Devuelve un número pseudoaleatorio.
SIGN	Devuelve el valor 1 con la señal del parámetro del valor.
SIN	Devuelve el seno trigonométrico de un ángulo dado.
SQR	Devuelve el cuadrado del parámetro.
SQRT	Devuelve la raíz cuadrada del parámetro.

Tabla de funciones de sistemas de control:

Palabra clave	Función
DEFINED	Determina si se ha definido una variable.
ERROR	Devuelve el número del error de un error reciente que causó una parada mientras se ejecutaba el programa o causó una reacción de REACTE.
$ERROR	Retorna el mensaje de error asociado con el código de error.
FREE	Devuelve la cantidad de espacio de almacenaje libre inutilizado de la memoria.
GET.EVENT	Retorna acontecimientos que se fijan para la tarea especificada.
ID	Devuelve los que identifican la configuración del sistema actual.
$ID	Devuelve la fecha de creación del sistema y la información de edit/revision.
LAST	Devuelve el índice más alto usado para una matriz (dimensión).
PARAMETER	Devuelve el ajuste actual del parámetro nombrado del sistema.
PRIORITY	Devuelve la prioridad actual del cierre de la reacción para el programa.
SELECT	Devuelve el número de la unidad que es seleccionado actualmente por la tarea actual para el dispositivo nombrado.
STATUS	Retorna la información de estado para una aplicación de programa.
SWITCH	Devuelve una indicación del ajuste de un interruptor del sistema.
TAS	Devuelve el valor actual de una variable de valor real y le asigna un nuevo valor. Las dos acciones se hacen indivisiblemente, así que, ninguna otra tarea del programa puede modificar la variable en el mismo tiempo.
TASK	Retorna información sobre una tarea de la ejecución de programa.
TIME	Devuelve un valor del número entero que representa la fecha o la hora especificada en el parámetro dado, de la secuencia.
$TIME	Devuelve un valor de la secuencia que contiene la fecha actual del sistema y hora o la fecha y la hora especificadas.
TIMER	Devuelve el valor del tiempo actual (en segundos) del contador de tiempo especificado del sistema.
TPS	Devuelve el número de las señales del reloj del sistema que ocurren por segundo (señales por segundo).

APLICACIONES

- Aplicaciones en fundición.
- Aplicaciones de Soldadura
- Aplicación de Pintura, Esmalte, Partículas de metal, etc.
- Alimentación de máquinas.
- Corte.
- Montaje / Ensamblaje
- Paletización
- Pick and place

Aplicaciones en fundición.

La fundición de materiales por inyección fue el primer proceso robotizado en 1960.
Los robots, en estos procesos son utilizados para el transporte de las piezas a un lugar de enfriado y posteriormente a otro proceso (desbardado, corte, etc.).

Aplicaciones de Soldadura

La tarea más robotizada dentro de la fabricación de automóviles es la soldadura de carrocerías.
Los robots de soldadura por puntos precisan capacidad de cargas del orden de los 50-100 Kg. y estructura articular, con suficientes grados de libertad (5 o 6) como para posicionar y orientar la pinza de soldadura (o pieza según el caso) en lugares de difícil acceso.

Aplicación de Pintura, Esmalte, Partículas de metal, etc.

Los robots de pintura suelen ser robots articulares, ligeros, con 6 o más grados de libertad que les permiten proyectar pintura en todos los huecos de la pieza. Cuentan con protecciones especiales para defenderse de las partículas en suspensión dentro de la cabina de pintura y sus posibles consecuencias (explosiones, incendio, deterioro mecánico). También tienen una accionamiento hidráulico por el riesgo de incendio.

Alimentación de máquinas.

La utilización de robots para alimentar máquinas aparece por la peligrosidad y monotonía de las operaciones de carga y descarga de maquinas.

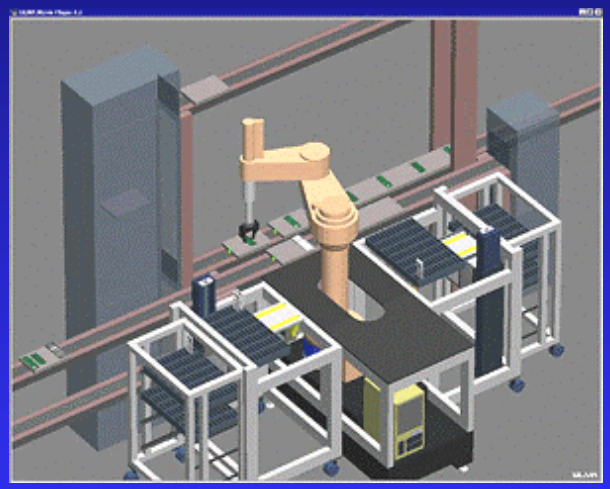
Los robots usados en estas tareas son de baja complejidad, precisión media, numero reducido de grados de libertad y un control sencillo, basado, en ocasiones, con manipuladores secuenciales y con un campo de acción grande.

Las estructuras mas frecuentemente utilizadas son la cilíndrica, esférica y articular. También la cartesiana puede aportar solución.

Corte.

El corte de materiales mediante el robot es una aplicación reciente, gracias a la capacidad de reprogramación del robot y su integración en un sistema que hace que sea el elemento ideal para transportar la herramienta de corte sobre la pieza, realizando con precisión un programa de corte desde un sistema de diseño asistido por computador (CAD).

Montaje / Ensamblaje.

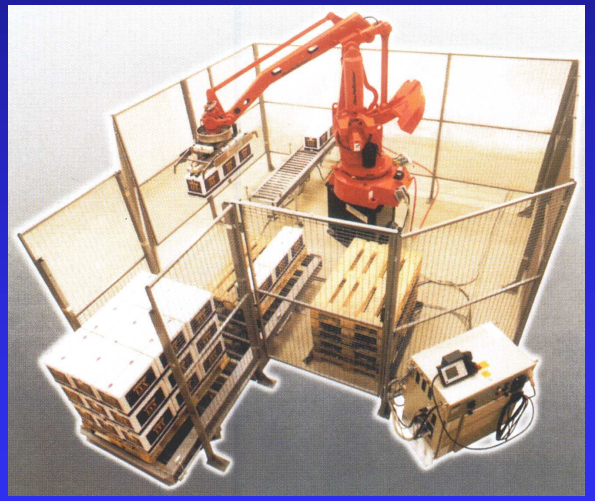


Las operaciones de montaje, por la gran precisión y habilidad que normalmente exigen, presentan grandes dificultades para su automatización flexible.

El tipo SCARA ha alcanzado gran popularidad en este tipo de tareas por su bajo coste y buenas características. Estas se consiguen por su adaptabilidad selectiva, presentando facilidad para desviarse, por una fuerza externa, en el plano horizontal y una gran rigidez para hacerlo en el eje vertical.

Paletización

La paletización es un proceso básicamente de manipulación, consistente en disponer de piezas sobre una plataforma o bandeja (palet). Generalmente, las tareas de paletización implican el manejo de grandes cargas, de peso y dimensiones elevadas. Por este motivo, los robots empleados en este tipo de aplicaciones acostumbran a ser robots de gran tamaño, con una capacidad de carga de 10 a 150 kg aproximadamente.

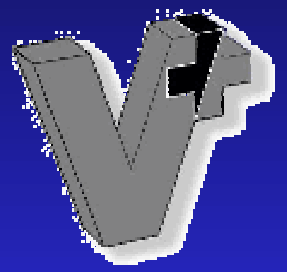


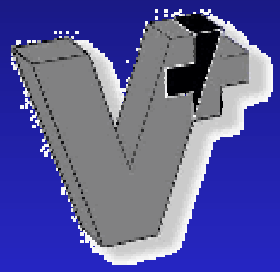
Pick and place

La misión de un robot trabajando en un proceso de pick and place consiste en recoger piezas de un lugar y depositarlas en otro
El propio robot gestiona las líneas de alimentación de las cajas y de palets, a la vez que toma las decisiones necesarias para situar la caja en el palet con la posición y orientación adecuadas de una manera flexible.

EJEMPLO DE MENU

```
.PROGRAM sub.menu()

; DESCRIPCIÓN: Este programa ofrece al operario un menú con las
; opciones que puede seleccionar, desde su terminal de usuario
; (botonera/ PC).
; Después de introducir una entrada desde el teclado, el programa
; ejecutará la operación seleccionada.
; El menú incluye la ejecución del programa de pick and place, mostrar
; los puntos, y volver al menu principal.
; SUPONEMOS: Las funciones pick.place() y teach() definidas.

AUTO opcion, salir, $frase

salir = FALSE

        DO
                TYPE /C2, "MENU DE OPERACIÓN PICK AND PLACE"
                TYPE /C1, " 1 => Inicializar pick and place"
                TYPE /C1, " 2 => Mostrar localización puntos"
                TYPE /C1, " 3 => Volver al menu principal", /C1

                PROMPT "Selecciona una opcion y presiona INTRO: ", $frase

                opcion = VAL($frase)        ;Convierte un string en un numero

                CASE opcion OF              ;entramos en el menu...
                        VALUE 1:            ;...seleccion 1
                        TYPE /C2, "Iniciando operacion..."
                        CALL move.parts()
                        VALUE 2:            ;...seleccion 2
                        CALL teach()
                        VALUE 3:            ;...seleccion 3
                        salir = TRUE
                        ANY             ;...si se presiona cualquier otra tecla
                        TYPE /B, /C1, "** ENTRADA NO VALIDA E R R O R **"
                END                         ;Fin del CASE
        UNTIL salir                 ;Fin del DO
.END
```





Este programa permite al operario grabar las posiciones de pick, place y start, con la paleta de control manual.

```
.PROGRAM teach(pick, place, start)

; ABSTRACT: Este programa se usa para grabar las posiciones
;"pick", "place", y "start" del "move.parts" programa.
; PARÁMETROS DE ENTRADA: Ninguno
; PARÁMETROS DE SALIDA: pick, place, y empezar
; EFECTOS SECUNDARIOS: El robot está inactivo mientras la rutina está
; activada.

AUTO $borrar.display          ;variable

$borrar.display = $CHR(12)+$CHR(7)

ATTACH (1)   ;Desactiva el control automatico del robot
DETACH (0)   ;Realiza el control manual del robot

; Salida en el display del control manual.

WRITE (1) $borrar.display, "Mueve el robot a la posicion 'START' & pulsa
RECORD"
WRITE (1) /X17, "RECORD", $CHR(5), /S
WRITE (1) $CHR(30), $CHR(3), /S  ;Parpadea el led de la paleta de control

WAIT PENDANT(3)   ;Esperar que una tecla sea pulsada

HERE start     ;Graba la posición "start"

; Aviso, en el display de la segunda localización

WRITE (1) $borrar.display, "Mueve el robot a la posicion  'PICK' y pulsa
RECORD"
WRITE (1) /X17, "RECORD", $CHR(5), /S

WAIT PENDANT(3)   ; Esperar que una tecla sea pulsada

HERE pick      ;Graba la posición "pick"

; Aviso, en el display de la tercera localización

WRITE (1) $borrar.display, "Mueve el robot a la posición  'PLACE' y pulsa
RECORD"
WRITE (1) /X17, "RECORD", $CHR(5), /S

WAIT PENDANT(3)   ; Esperar que una tecla sea pulsada

HERE place    ;Graba la posicion "place"

ATTACH (0)    ;Activa el control automatico del robot
DETACH (1)   ;Anula el control manual del robot

RETURN        ;Vuelve a llamar al programa
.END          ; Finaliza el programa
```

Ejemplo de Paletización

```
.PROGRAM move.parts()
;DESCRIPCION:    Este programa coge cajas en la localización “pick”
; y las deposita en “place”, incrementando la z de place, y así apilando
; las cajas en el palet.
        parts = 6                       ; nº de cajas a apilar
        height1 = 300                   ; altura de “approach/depart” en “pick”
        height2 = 500                   ; altura de “approach/depart” en “place”
        parameter HAND.TIME = 0.16      ; movimiento del brazo lento
        OPEN                                    ; apertura de pinza
        RIGHTY                          ; seleccionamos configuración derecha
        MOVE start                      ; mover a la localización segura de inicio
        FOR i = 1 TO parts          ; iniciar el apilado de cajas
                APPRO pick, height1     ; ir a “pick-up”
                MOVES pick                      ; mover hacia la caja
                CLOSEI                  ; cerrar la pinza
                DEPARTS height1         ; volver a la posición anterior
                APPRO place, height2    ; ir a “put-down”
                MOVES place             ; mover a la localización de destino
                OPENI                   ; abrir la pinza
                DEPARTS height2         ; volver a la posición anterior
                SHIFT heicht2 BY 0.00, 0.00, 300
        END
        TYPE “Fin de tarea. ”, /IO, parts, “ cajas apliadas.”
        RETURN                  ; fin del programa
.END
```