UNIVERSIDAD DE COSTA RICA

FACULTAD DE INGENIERÍA
ESCUELA DE CIENCIAS DE LA COMPUTACIÓN E INFORMÁTICA

CI2657- ROBÓTICA
Prof. Kryscia Daviana Ramírez Benavides

Tarea #4:
Introducción a la Robótica, Sensores, Actuadores y Introducción al Control de Robots

Elaborado por:
Emmanuel Arias B30640
Gloriana Garro B32831
Luis Diego Hernández B23188
Jose Víquez B37638
Geovanny Zúñiga B37822

19 de Septiembre del 2016

Tema 6: Programación de Robots

- Conexión de todas las piezas del robot está en el software
- complejidad para programar robots: impredictibilidad
- Necesidad de usar lenguajes para especificar tareas en términos entendible a humanos. Se usan para dos fines
    - Definir la tarea a realizar
    - Controlar al robot mientras la realiza
- Lenguaje para programar robot debe ser diferente a los convencionales
    - El entorno del robot debe usar relaciones espaciales, temporales, o de causalidad
    - robot opera en mundo real impreciso, modelo del mundo que el programa maneja puede ser impreciso
- Interrupciones en cualquier momento: señales sensoriales se producen de forma impredecible
- Lenguajes deben incorporar fácil manejo de transformaciones homogéneas

Requerimientos de los lenguajes de programación de robots

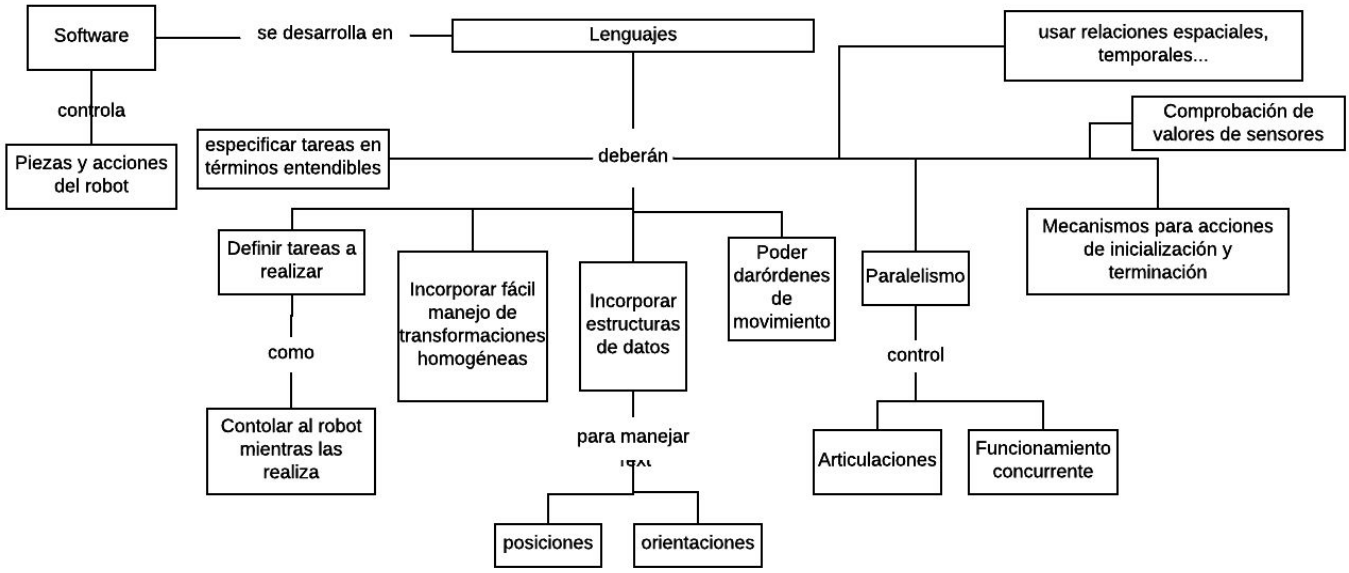
- El lenguaje deberá
    - incorporar estructuras de datos apropiadas para manejar posiciones, orientaciones, y transformaciones espaciales
        - matrices de transformación homogénea
        - operaciones de rotación y traslación
    - órdenes de movimiento
        - en espacio cartesiano: trayectoria
        - conocer posición actual
    - Permitir paralelismo:
        - control simultáneo de articulaciones y funcionamiento concurrentes y con sus propios sensores externos
    - Permitir la comprobación constante de valores de sensores
        - comprobación de eventos
        - mecanismo de prioridades
    - Proveer estructuras y acceso a variables sensoriales
        - no se inicializan explícitamente
        - alcance global
    - Mecanismos para especificar acciones de inicialización y terminación

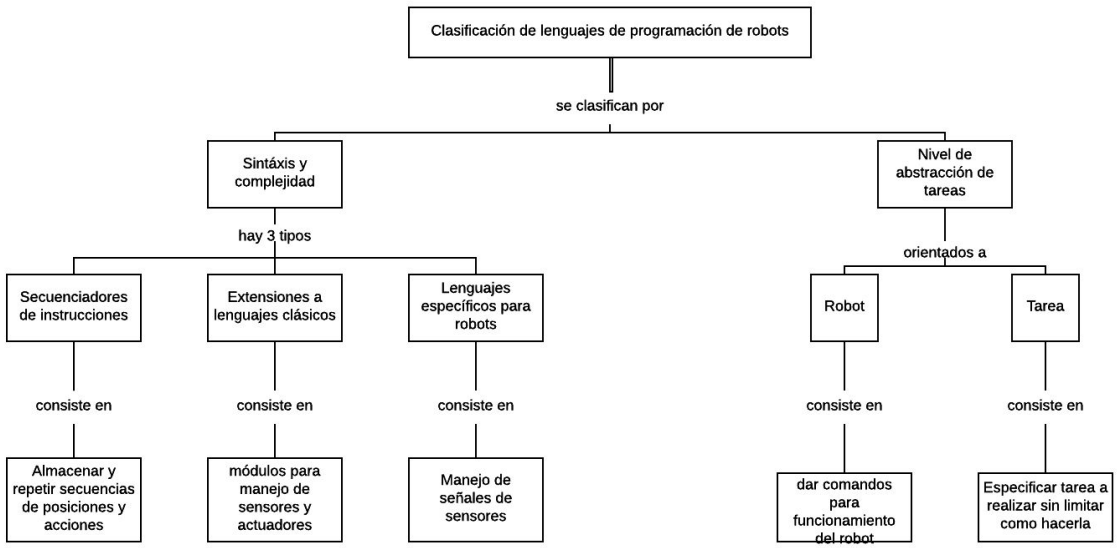
Sistemas operativos

- Los requerimientos de lenguaje pueden repartirse entre el sistema operativo y el lenguaje
    - depende de la velocidad del computador
- La característica más importante del sistema operativo: que pueda verificarse en tiempo real
    - Ej: Necesidad de mantener lazos cerrados de retroalimentación de los controladores de las articulaciones → muestreo de sensores en intervalo de tiempo, y el envío de acción de control
- Es importante el diseño cuidadosos de cada proceso para garantizar su terminación en un tiempo acotado

Clasificación de los lenguajes de programación de robots

1. Tres tipos de lenguajes, clasificados por sintaxis y complejidad
    a. Secuenciadores de instrucciones
        i. almacenan y repiten una secuencia de posiciones y acciones en un orden más o menos fijo
        ii. Aprende posiciones y acciones de varias maneras
            1. Movimiento del robot con el joystick, mouse, teclado...
            2. Movimiento manual y almacenamiento de posiciones
        iii. Incluyen órdenes especiales para retraso, especificación de acciones...
    b. Extensiones a lenguajes clásicos
        i. módulos para el manejo de sensores y actuadores
        ii. Pueden basar en varios lenguajes
        iii. Es necesario que corran sobre un sistema operativo en tiempo real
    c. Lenguajes Específicos para robots
        i. Diseñados con fines comerciales
        ii. incorporan manejo de señales de los sensores
        iii. Descripción y razonamiento en términos geométricos
2. Clasificación según el nivel de abstracción para especificar tareas
    a. Orientados al robot
        i. comandos para el movimiento del robot, lectura de sensores...
    b. Orientados a la tarea
        i. Especificación de qué tarea hacer sin limitación de como hacerlo





Niveles de programación:

La siguiente figura muestra las distintas capas de abstracción en un software para robots. Más adelante se entrará en detalles para cada una de estas.

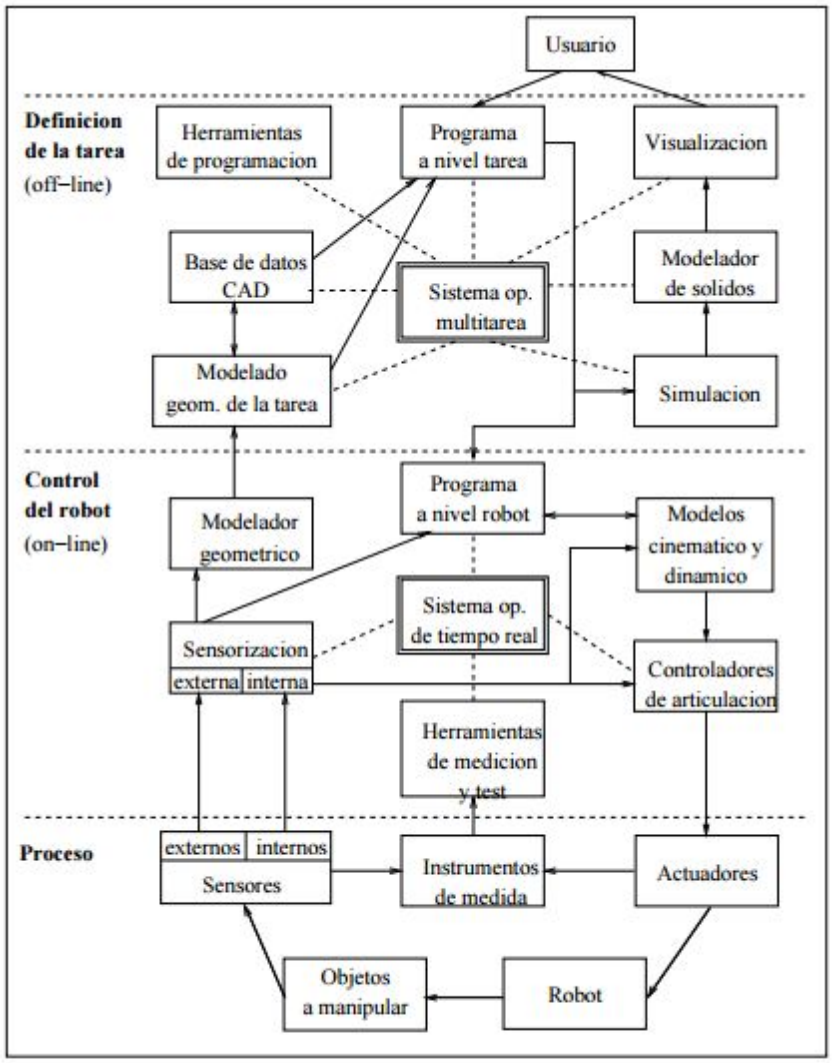


Figura 6.1: Cuadro resumen del software de un sistema robotizado

Proceso: se refiere al hardware actuando sobre el mundo físico. En este nivel se encuentran los sensores y los actuadores de un robot.

Control del robot: Contiene herramientas para instrumentos de medición y otros procesos básicos del robot. En este nivel se procesan las entradas de los sensores y se efectúan cambios en robot partir de estas señales. Además, se controlan las articulaciones del robot, enviando señales a los actuadores. Todo este procesamiento se hace en un sistema operativo de tiempo real.

Definición de la tarea: En este nivel hay dos ramas importantes: el modelador geométrico de la tarea y su simulador. El modelador determina los objetos y sus posiciones dentro del mundo del robot. Es con esto que se toma una decisión sobre qué tarea llevar a cabo. El simulador es opcional y ofrece una prueba de esta

decisión antes de realizarla con el fin de depurar errores. El simulador corre sus tareas de forma concurrente. A este nivel, el sistema operativo funciona con un modelo multitarea, por ejemplo un SO basado en UNIX.

Lenguajes orientados al robot:

En esta parte del documento se definen las especificaciones que debía tener un lenguaje orientado a robots en su momento. Estos tipos de lenguajes generan órdenes para el robot. Entre las órdenes más importantes para el robot están las de movimiento. Por eso, el lenguaje del robot deberá soportar movimientos de tipo libres, condicionados y restringidos.

Sobre la evolución de los lenguajes orientados al robot, el autor expone tres niveles de complejidad. Los detalles de cada uno se muestran el la siguiente lista:

1. En un principio los lenguajes orientados a robots generaban y ejecutaban secuencias de instrucciones. Permitían la lectura y modificación de variables de usuario. Por ejemplo, el lenguaje APT era capaz de generar instrucciones para máquinas CNC (control numérico por computador).
2. Luego se desarrollaron interfaces más amigables para definir posturas y orientación. Además hacían cálculos y comprobaciones de trayectorias. Estos lenguajes heredaron características de lenguajes como Pascal y Algol. Tenían la capacidad de ejecutar concurrentemente.
3. En una generación más avanzada, los lenguajes podían ser definidos por secuencias de instrucciones pero además podían actuar bajo estrategias basadas en la entrada de datos de los sensores.

Estudios de caso

AL: este fue desarrollado por la Universidad de Stanford para manipular su brazo robótico, su sintaxis y características son similares al de ALGOL y a PASCAL ya que permite ordenes a nivel del robot; algunos ejemplos de estas son: ROT(eje, ángulo), MOVE barm TO A(donde A es un MH y barm es el punto final del brazo).

Este lenguaje maneja concurrencia y necesita un sistema operativo que maneje tiempo real ya que este es sumamente necesario en el brazo mecánico. Un ejemplo de una función que se puede resolver en AL es que, el brazo tome un tornillo del alimentador y lo inserte en un orificio.

VAL-ll:este lenguaje fue desarrollado por Unimation para su robot PUMA, su sintaxis es similar a FORTRAN y a BASIC;algunos ejemplos de instrucciones en este lenguaje son: MOVE, REMARK, SET. Un problema que se puede resolver utilizando este lenguaje sería el de tomar cuatro bloques y apilarlos en otro lado formando una torre.

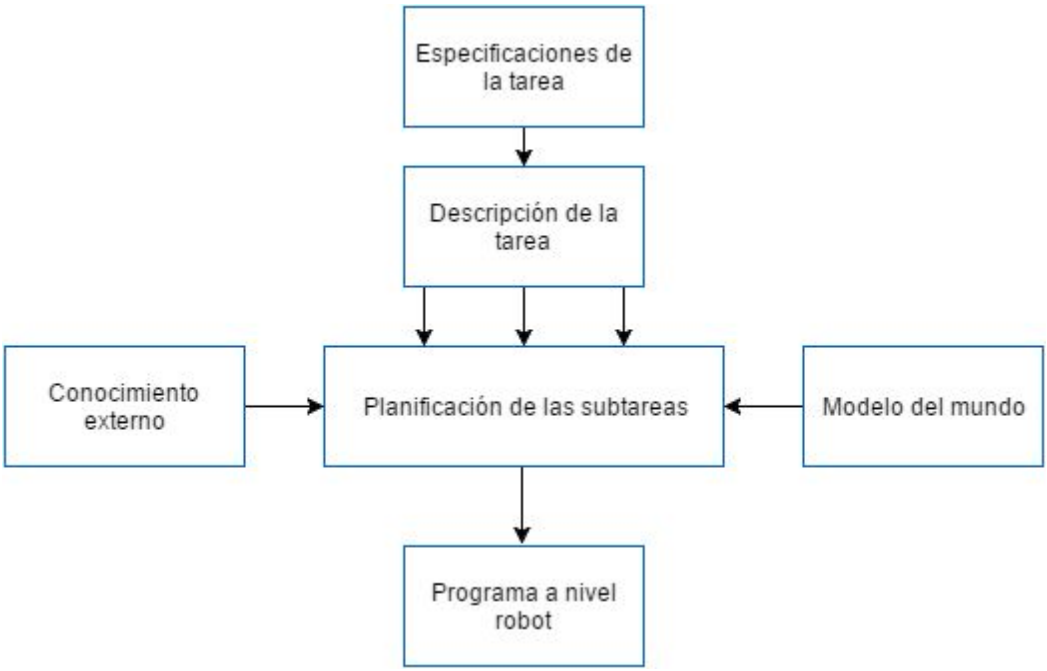
Lenguajes orientados a la tarea:

Como los lenguajes orientados al robot son un poco tediosos y además susceptibles al fallo ya que dependen totalmente de las partes mecánicas del robot, se crearon los lenguajes orientados a la tarea, en este el programador especifica que desea hacer y es el sistema el que decide qué sensores utilizar y qué movimientos realizar. Para esto se utilizan lenguajes de alto nivel y un módulo es el encargado de convertir estos comando en instrucciones para el robot.

En este campo los objetos que se utilizan comúnmente son los MMO(Man Made Objects) que se pueden definir discretamente, los que escenciales en términos de volumen CGS(Constructive Solid Geometry) y los esenciales en términos de límites BR(Boundary Representation). La mayoría de formalismos utilizan fronteras, superficies, cilindros generalizados y una descomposición en células.

En robótica la característica que más nos interesa es la instanciación, esta sirve para llenar una estructura vacía y compararla con una ya llena anteriormente, esto con cierto grado de tolerancia ya que son datos de sensores.

El siguiente es una ilustración simple de cómo se crea un programa para un robot con un lenguaje orientado a la tarea:



Estudios de caso:

AUTOPASS: Fue creado por IBM para usos robóticos y pretende ser el modelador geométrico que debería ser consultado por las rutinas de planeación de tareas. Cada objeto se define en AUTOPASS mediante un procedimiento que invoca otros,

bien previamente definidos, bien primitivas del lenguaje. Cada procedimiento admite parámetros que definen el objeto. Todos los objetos se consideran poliédricos y los cilindros se aproximan como prismas de base poligonal.

AUTOPASS usa operaciones de lógica y un encaje sintético que puede instanciar el dato en alguno de los procedimientos, haciendo así encajar la descripción proporcionada por alguno de los sensores con la descripción almacenada.

RAPT: Los objetos pueden ser modelados de un modo un poco más flexible que en AUTOPASS. Los objetos pueden ser cuerpos con caras planas o esféricas, o bien ser cilindros generalizado, o tener orificios cilíndricos. Las relaciones entre objetos se definen en una forma más natural hacia los humanos, por ejemplo las relaciones de contacto son AGAINST, FIT y COPLANAR. A RAPT se le deben dar las especificaciones de un conjunto de objetos en forma de descripciones de sus aspectos (caras y vértices), y declaración de las relaciones entre estos en un cierto instante, así como la descripción para el estado deseado. RAPT entonces genera un programa que pueda llevar los objetos desde el estado original al estado deseado. El programa es específico para las capacidades del robot y conociendo las características físicas de las figuras y del espacio es que logra elegir las acciones más óptimas para realizar la tarea.