

# **INTRODUCCIÓN A LA ROBÓTICA INTELIGENTE**

**\***

***Pedro José Sanz Valero***

**sanzp@icc.uji.es**

## ÍNDICE

PRÓLOGO	“LA ERA DE LA ROBÓTICA”
TEMA I	PRELIMINARES
TEMA II	EL ROBOT MANIPULADOR
	2.1. Localización espacial
	2.2. Cinemática de posición
	2.3. Cinemática del movimiento
	2.4. Control del movimiento
TEMA III	SISTEMAS DE PERCEPCIÓN ROBÓTICA
	3.1. Generalidades del proceso de percepción
	3.2. Introducción a la percepción artificial
	3.3. La percepción visual
	3.4. La visión artificial
	3.5. Introducción al reconocimiento de formas
TEMA IV	COMUNICACIÓN HOMBRE-ROBOT
	4.1 Aspectos diferenciales de la programación de robots
	4.2 Clasificación de los lenguajes de programación
	4.3 Lenguajes de programación a nivel tarea
	4.4 Interfaces avanzadas
BIBLIOGRAFÍA	
APÉNDICE 1	“LA ROBÓTICA EN EL CIBERESPACIO”
APÉNDICE 2	“LA FUNCIÓN <i>atan2</i> ”

## PRÓLOGO “LA ERA DE LA ROBÓTICA”

Aunque los antecedentes de la robótica se remontan muy atrás en el tiempo, Grecia clásica e incluso antes, su verdadero motor de desarrollo no llega hasta la aparición de los microprocesadores, que en definitiva acabarán constituyendo su cerebro, ya en la década de los setenta. Así pues, la aparición de los primeros robots industriales comerciales dista tan solo una treintena de años en el tiempo. Siendo el reto actual, el mismo que potenció su origen, acercar el robot al hombre, para hacerle más grata su existencia. Marcando su evolución, la tecnología disponible en cada momento de la historia.

Sea como fuere, parece ser que el destino de la humanidad está cada vez más ligado a la robótica, y por tanto, nada ni nadie puede inhibirse de esta realidad social, de ahí la proliferación de congresos, cursos y, en general, planes de estudio de Universidades en los que, cada vez más, se incluyen asignaturas relacionadas directa o indirectamente con la robótica. Seguramente en un futuro cercano hará falta, incluso, una especialización mayor que puede dar lugar a titulaciones específicas en esta materia, como en su día ocurriera con otras disciplinas (e.g. mecánica, electrónica, informática, etc.).

Para entender mejor los intereses que la sociedad demanda en lo referente al campo de la Robótica, y, por tanto, las complejidades inherentes a dicha disciplina, será conveniente que hagamos una pequeña reflexión histórica de su evolución.

### Perspectiva Histórica

En el desarrollo de la robótica tal y como la entendemos hoy, existen una pluralidad de intereses y tecnologías diferentes que hace que sea muy difícil establecer una secuencia lógica y consistente de eventos ó descubrimientos en los que apoyarse. No obstante, se pueden establecer algunas líneas maestras para esclarecer el progreso de este campo, teniendo en cuenta que las incorporaciones científico-tecnológicas se suceden a cada momento.

Ya en la antigua Grecia, con *Herón* de Alejandría (85 d.C.), aparecen los primeros automatismos dirigidos a ahorrar esfuerzo en diferentes tareas. De hecho, la palabra autómatas deriva del término griego “automatos”. La tecnología empleada era hidráulica y mecánica, fundamentalmente. A medida que los medios tecnológicos fueron progresando, mejoraron igualmente las posibilidades y prestaciones, desde los mecánicos, en los ingenios de *L. Da Vinci* (1499), pasando por los mecanismos de relojería, incorporados por *J. De Vaucanson* (1738), llegando finalmente, al célebre telar de *Jacquard* (1801), que admitía programación mediante tarjetas perforadas, fraguando de manera incipiente, la génesis de la revolución informática que habría de venir.

Ya en nuestro siglo, *N. Wiener*, acuñó el término “cibernética” [Wiener, 48], para designar el estudio unificado del control y la comunicación en los animales y las máquinas. Siendo la famosa “tortuga” de *W.G. Walter*, uno de los primeros ingenios en utilizar esta tecnología analógica, denominada cibernética.

Conforme la industria nuclear fue expandiéndose, a raíz de la segunda guerra mundial, surge otro campo impulsor de la robótica moderna, sin duda, el de los telemanipuladores, iniciado por *R.C. Goertz*, en 1948, en el “Argonne National Laboratory” (EEUU) que, mediante una arquitectura maestro-exclavo, permitían manipular a distancia materiales radiactivos.

Casi al mismo tiempo, para automatizar determinadas tareas en la industria, aparece la primera máquina de control numérico, desarrollada en el MIT<sup>1</sup> (EEUU), durante 1952.

Y con este panorama, tremendamente simplificado, se llega a la aparición del primer robot industrial, por *G.C. Devol*, en 1954. [Fu et al., 88]:

“La clave de este dispositivo (Robot) era el uso de un ordenador en conjunción con un manipulador para producir una máquina que podía ser enseñada para realizar una variedad de tareas de forma automática”.

La primera empresa productora de robots fue Unimation (Universal Automation, *Devol* y *Engelberger*, 1958) que instaló su primer robot en la General Motors, incorporándolo a su sistema de producción, en 1961.

Casi desde su inicio, los ingenieros e investigadores en el campo de la robótica, fueron conscientes de que la flexibilidad, conducta adaptativa, capacidad de reacción ante situaciones imprevistas, en definitiva, el trabajo en entornos denominados “no-estructurados”, estaba ligado a capacidades perceptivas que posibilitaran la realimentación sensorial en los movimientos del robot. Ello produjo un ingente esfuerzo investigador en este sentido, desde los sensores táctiles (*Ernst*, 1962) y de presión (*Tomovic* y *Boni*, 1962), hasta el denominado “Remote Center Compliance” (RCC, *Nevins* et al., 1974), pasando por la visión (e.g. “sistema ojo-mano” de *M. Minsky*, en el MIT, 1965). Puntualicemos, que algunos trabajos de esta época, como el denominado “SRI<sup>2</sup> vision system”, desarrollados en Stanford, en la década de los setenta [Agin & Duda, 75], han sido fuentes de inspiración de sistemas industriales actuales.

En 1960, *J. McCarthy*, en el MIT, acuña el término “Inteligencia Artificial”, encargándose sus discípulos más sobresalientes, *Minsky*, *Newell* y *Simon*, de difundir los beneplácitos que esta nueva rama del saber deparaba. Incluyendo, el razonamiento automático, los sistemas expertos, y, como no, la Robótica.

Entusiasmados con las nuevas herramientas de planificación y razonamiento automático que la IA promovía, se diseña en la Universidad de Stanford (1969), el primer robot móvil controlado por visión, “Shakey”. En realidad, la característica más singular de este robot era que su control se realizaba fundamentalmente por medio de programas de razonamiento (de hecho, el planificador era STRIPS, desarrollado por *Nilsson* et al.). En palabras de *H. Moravec* [Moravec, 93]:

“Shakey era impresionante como concepto pero digno de compasión cuando se movía”

El estrepitoso fracaso de “Shakey”, puso de manifiesto que trasladar a una máquina el razonamiento lógico de alto nivel era más sencillo, que incorporar habilidades sensomotoras [Churchland & Sejnowski, 92], en contra de lo que se pensaba inicialmente. Siguiendo un diseño de “arriba hacia abajo”, se le había dado prioridad al razonamiento sobre la percepción y la movilidad. Precisamente este fracaso, y otros que le siguieron, obligaron a volver la vista atrás y retomar viejos conceptos provenientes de la cibernética, que seguía el enfoque contrario de “abajo hacia arriba”, apareciendo un primer intento, proveniente del MIT, en los

---

<sup>1</sup> Massachusetts Institute of Technology

<sup>2</sup> Stanford Research Institute

denominados “robots insectoides” de *R. A. Brooks* [Brooks, 86], y su “arquitectura de subsumción”.

Finalmente, ya prácticamente en nuestros días, se hace patente el principio de la denominada “robótica inteligente”, preconizado por investigadores como *M. Brady* [Brady, 85]:

“Conexión inteligente entre percepción y acción”

La lección que parece desprenderse de todo lo anterior es que si bien, en un principio la robótica presagiaba un rápido progreso, haciendo un uso creciente de realimentación sensorial, para el control de sus acciones, e incorporando los nuevos avances en la teoría del control, en el diseño mecánico, en la electrónica, en la tecnología informática y, sobre todo, en las potentes herramientas (al menos, potencialmente) que ofrecía la IA, esto no ha sido así. De hecho, la Robótica está constituyendo un auténtico reto para todas estas áreas y tecnologías. Lo cuál, tiene como contrapartida, un mayor esfuerzo investigador y, en general, de recursos humanos y materiales, hacia líneas de investigación que logren hacer avanzar la Robótica, como la sociedad demanda.

## **Situación Actual**

Como ya se apuntó, al principio de éste capítulo, el mejor cerebro que en la actualidad puede incorporar un robot está basado en la tecnología digital y, normalmente está constituido por un conjunto de microprocesadores que ayudan a paralelizar los procesos que de forma concurrente requiere llevar a cabo un robot actual en el transcurso de su tarea.

Si el primer robot industrial se incorporó a procesos reales de producción, a principios de los sesenta, no fue hasta finales de los setenta y principios de los ochenta, cuando los robots industriales se consolidaron como herramienta significativa en la industria. Y esto ha sido así, fundamentalmente por dos razones [Regh, 97]: fue en dicho periodo cuando su rendimiento laboral (cociente entre el coste económico y la productividad) se hizo asequible, y además, se incorporó el “microprocesador”, desarrollado por *Intel Corporation* en 1971. Siendo el robot “Cincinnati Milacron T3 (*The Tomorrow Tool*)”, diseñado por *R. Hohn* en 1973, el primero que hace uso de dicha tecnología.

Es paradójico constatar, siguiendo a *Regh* [Regh, 97], que aunque inicialmente la génesis de los robots industriales comerciales se debió a empresas de EEUU, en la actualidad entre las empresas más relevantes la única no japonesa es ABB, de Suecia. Por tanto la hegemonía absoluta en este terreno de la robótica industrial la ostenta Japón, muy por delante de cualquier otro país en el mundo, incluyendo EEUU. Y esto es así, no sólo en producción, sino también en explotación. Según datos de la CEE a partir del estudio “*World Industrial Robots*” de 1997, dado a conocer en Ginebra, Japón posee la mitad del parque mundial de robots industriales, siendo el incremento en todo el mundo del 11%, y en España del 37%.

Seguramente, lo anterior, está relacionado con el hecho de que, ya en 1971, la Robótica, se declara de interés nacional en Japón, creándose al mismo tiempo, la primera asociación al respecto (JIRA), por delante de los EEUU, que lo hicieron hacia 1975, fundando el denominado “Robot Institute of America”. Japón se convirtió desde este momento en el principal receptor de la tecnología Robótica y, ha sido sólo cuestión de tiempo, el que se haya hecho con el parque mundial de robots, implantados y, de fabricación de los mismos.

Relativo al impacto de los robots en el mercado laboral, conviene observar las conclusiones del estudio citado anteriormente (“*World Industrial Robots*”), que apunta la idea de la necesidad de incrementar la tecnología de la robotización en cualquier proceso de producción para poder competir, rechazando de esta manera la relación causa-efecto entre el incremento de la robotización y el despido de los trabajadores. No obstante, conviene recordar algunas otras ideas al respecto para no precipitar las conclusiones. ¿Cuándo se justifica la implantación de robots?:

N. W. Clapp, (1982) *“únicamente en aquellas tareas donde el hombre actúa como un robot y no el robot como un hombre”.*

M. Minsky, (1985) *“(a menudo) sobre la base de conseguir una ganancia neta en productividad que redunde en beneficio de toda la sociedad, aunque esto signifique la pérdida de empleo y dignidad para unos pocos”.*

Este, en resumen, es un tema muy serio y, sin embargo, en el que los investigadores en cualquiera de los aspectos que aglutina la robótica, no suelen reparar. Pero alguien tendrá que plantear este asunto con seriedad, sino se quiere que la sociedad, afectada cada vez en mayor medida con la lacra del paro, ejerza un movimiento involucionista sobre esta nueva tecnología que hay que poner al servicio de la humanidad, y no de unos pocos.

Por otra parte, los dominios o campos de aplicación donde existen robots implantados actualmente, son cada vez más numerosos:

- Conquista del Espacio
- Industria en General
- Laboratorios
- Manipulación en Entornos Peligrosos
- Agricultura
- Educación
- Asistencia a Discapacitados
- Seguridad y Defensa
- Sector Servicios

Pudiendo clasificarse, de forma general en [Barrientos et al., 97]:

Consolidados. Donde los robots están implantados hace años y forman parte intrínseca de los procesos que automatizan. Básicamente, aplicaciones industriales (e.g. soldadura, ensamblado, paletización, pintura, alimentación de máquinas, carga y descarga, etc.).

No-Consolidados. Son los dominios de reciente aplicación de los robots. Algunos como el sector espacial, con más experiencia que otros. No obstante, los más novedosos constituyen el incipiente sector de servicios (e.g. cirugía, ayuda a discapacitados, etc.).

## **Prospección de Futuro**

En el ámbito industrial, se camina hacia soluciones integrales [Rehg, 97], véase CAD/CAM/CAE/CIM, etc., donde el robot no se plantea nunca como solución

aislada, sino que forma parte de un proceso global, donde intervienen la gestión, el diseño, la fabricación, el control de calidad, etc., de forma interconectada.

Por otra parte, los nuevos ámbitos de implantación de robots (espacial, servicios, etc.), constituyen un reto permanente, y cualquier avance sustancial en estos nuevos dominios revierte sobre los demás.

Conforme la tecnología avanza, se incorporan al robot nuevas posibilidades, fundamentalmente a partir de la percepción de su entorno, cada vez más potente y sofisticada, favoreciendo e impulsando nuevas capacidades, como la toma de decisiones de alto nivel, propiciada por técnicas propias de la IA. La prueba más evidente del enorme progreso de la robótica, reside en el hecho de que se está implantando robots en áreas absolutamente impensables hace bien poco, como por ejemplo, para asistencia en operaciones quirúrgicas, incluso a distancia (“telesurgering”), o su uso, cada vez mayor, en nuevos sectores de la industria alimentaria, farmacológica, etc.

Es interesante, comentar la existencia de algunos proyectos de investigación que están, actualmente, en vanguardia, y que preparan una generación de robots y sistemas robotizados, en general, muy por delante de lo conseguido hasta ahora. Por ejemplo, el denominado proyecto ISAC (“Intelligent Soft Arm Control”) [ISAC, 97], véase la figura 1, desarrollado en la Universidad de Vanderbilt (EEUU), y dirigido al sector de servicios (hospitales, doméstico, asistencia a discapacitados etc.). Estos proyectos tienen un interés “*per se*”, en el sentido de que ponen a prueba la tecnología punta existente, integrándola en un sistema único, para alcanzar un objetivo realmente complejo. En el caso particular de ISAC, el objetivo puede ser dar de comer a alguien, mediante un brazo manipulador “antropomorfo” y dotado de visión, y que se comunica con el robot mediante una interfaz basada en voz.

Sí en un principio el entorno se adaptaba al robot, estructurándolo de forma que facilitara su implantación para resolver una tarea, ahora la tendencia es la contraria, es decir, que el robot se adapte al entorno, en la medida que la tecnología lo permita.



**Figura 1.** Imagen extraída de “shogun.vuse.vanderbilt.edu”, mostrando el denominado “Soft Arm” en acción. El usuario interactúa con dicho sistema a partir de órdenes verbales.

Esta filosofía de trabajo, llevada al extremo, retoma el antiguo espíritu renacentista (Leonardo y otros) del robot “antropomorfo”. Pero, surge aquí una cuestión:

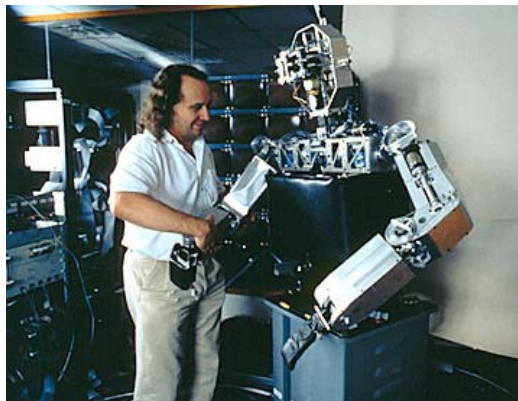
¿Existe realmente necesidad de que un robot adopte nuestra apariencia?. O, dicho de otro modo:

¿Cuál es la auténtica razón del empeño en construir un robot humanoide?. Es decir, no sólo queremos una máquina capaz de imitar el comportamiento humano, sino además, que se le parezca fielmente: ¿Por qué?.

La respuesta es compleja, porque son complejos los intereses que mueven la robótica desde su inicio. Y, no solamente existen distintas respuestas que defienden este intento, sino que cuenta con auténticos detractores también. Veamos dos posibles razones a favor del robot humanoide:

- *Brooks*, que dirige en la actualidad el denominado proyecto “Cog” [Brooks, 97], en el MIT (EEUU), defiende la tesis de que hasta el momento, en la comunicación hombre-robot, no se han tenido en cuenta las necesidades psicológicas del hombre, que permitan la integración del robot, como uno más en su entorno emocional. De tal suerte, que sino se considera al robot como a un “igual”, este se verá imposibilitado de progresar en sus capacidades de comunicación pseudo-humanas, quedando, en consecuencia, muy limitadas sus posibilidades de aprendizaje.
- *Engelberger*, retoma la idea de *Asimov*, de conseguir un robot plenamente autónomo al servicio del hombre [Engelberger, 95]. Defendiendo la idea de que los robots del futuro serán por necesidad antropomórficos. El mundo del hombre está diseñado por y para el hombre, si queremos que una máquina lo comparta, necesitará desplazarse, percibir, comunicarse y, en definitiva, comportarse como un ser humano.

El proyecto “Cog” (véase la figura 2), no busca una solución a un problema concreto, sino que intenta integrar toda la tecnología punta disponible actualmente, para ver hasta donde somos capaces de llegar en el desarrollo de un robot humanoide. Y, como es lógico, existen otros proyectos similares en el mundo (e.g. “*The Japanese Humanoid Project*”, Universidad de Waseda, en Japón), persiguiendo la misma idea. Incluso empresas como Honda, han lanzado ya un primer prototipo de robot humanoide, véase la figura 3, realmente espectacular, capaz de andar con sus dos piernas, subir y bajar escaleras, manipular con sus dos brazos, etc.



**Figura 2.** Imagen extraída de [www.ai.mit.edu](http://www.ai.mit.edu) El responsable del proyecto Cog, Rodney Brooks y su obra.



No obstante, en honor a la verdad lo anterior es más un golpe de efecto que otra cosa, e incluso, hay quien opina que estos intentos más que adelantar, atrasan el progreso auténtico y bien consolidado de la robótica, pues (afirman sus detractores), conviene seguir “paso a paso”, no haciendo uso de “atajos” ni derroches humanos y materiales que pueden suponer una pérdida de credibilidad en la sociedad, al no conseguir los ambiciosos resultados propuestos.

Lo que parece estar fuera de toda duda es que si bien en un principio la industria ha sido el principal motor de desarrollo de la Robótica, y hasta la fecha, la mayoría de robots implantados en diferentes sectores, son de este tipo, existen otras áreas como la espacial, militar, minería, etc., que están potenciando robots autónomos móviles, muy sofisticados.

Ahora bien, por desgracia para los robots humanoides, en palabras de *Moravec* [Moravec, 93], lo que peor hacen los ordenadores y, por ende los robots, son las cosas más naturales para los seres humanos: ver, oír, manipular objetos, aprender idiomas y razonar con sentido común. Esta dicotomía – las máquinas hacen bien cosas que a los humanos nos resultan muy difíciles, mientras que hacen mal las cosas que nos resultan más fáciles – es una ayuda enorme para resolver el problema de construir una máquina inteligente.

Uno de los grandes inconvenientes para el progreso eficaz de la robótica, parece ser la falta de comunicación entre los diferentes grupos que investigan y desarrollan en los diferentes campos de la Robótica. Precisamente el carácter interdisciplinar “*per se*”, inherente a esta ciencia, está impidiendo que avance más deprisa. Las nuevas ciencias de la complejidad [Pagels, 91], constituyen un primer paso hacia la resolución de problemas interdisciplinares que afectan a los sistemas complejos, como el cerebro humano o la predicción meteorológica. Un esfuerzo análogo debiera emprenderse con relación a la robótica inteligente.



**Figura 3.** Imagen extraída de “[www.honda.co.jp](http://www.honda.co.jp)”. Prototipo del robot humanoide, P2, fabricado por Honda.

Según *Moravec* [Moravec, 93], El auténtico avance de los robots se producirá cuando se consiga crear un robot multiuso, válido tanto en la fábrica como en el hogar. Digamos que lo vaticinado por *Moravec*, que afirmaba que en el plazo de diez años existiría un robot multiuso para el hogar, ayudando en las tareas domésticas, no se ha cumplido. Pero, seguramente no le faltaba razón al afirmar que el mercado potencial de los robots crecería enormemente si se alcanzara un cierto nivel de utilidad general.

Finalmente, podemos concluir con las ideas de *Minsky* [Minsky, 94], al respecto del futuro de la robótica, quien afirma que análogamente a como ha ocurrido en el campo de los ordenadores, que su entrada en las casas ha sido decisiva para su increíble desarrollo, una incipiente entrada del robot en el ámbito doméstico sería igualmente determinante para su impulso definitivo. En cierta medida lo que realmente está ocurriendo hasta ahora es que se están “automatizando” determinadas tareas (fregaplatos, lavadora, etc. ), e incluso se construyen “casas inteligentes”, pero en realidad no existe un agente autónomo que realice dichas tareas de forma completa. Este agente será el robot doméstico.

Quizás no nos damos cuenta pero el robot móvil de juguete de nuestros hijos pueda ser un precursor de lo que se avecina. No hace mucho, algunas estrategias que incorporan estos juguetes eran piezas preciadas en los laboratorios de vanguardia de todo el mundo, e incluso se han creado nuevas arquitecturas como la de Sony, denominada “OPEN-R”, incorporada a su robot “mascota”, véase la figura 4, que pueden significar un paso acertado para el futuro robot de servicios.



**Figura 4.** Imagen extraída de “[www.sony.co.jp](http://www.sony.co.jp)”. Prototipo del robot perro “AIBO” de Sony.

### **Resumiendo: ¿Cuál es el objetivo de esta introducción a la denominada “Robótica Inteligente”?**

Con los pies en la tierra, después de la perspectiva de futuro abierta en este prólogo, el autor, desde una cierta experiencia en la materia que nos ocupa, pretende ofrecer, al futuro ingeniero informático, los conocimientos básicos que le permitan abordar proyectos avanzados en el campo de la Robótica, tales como la percepción artificial o la comunicación hombre-robot, fundamentalmente.

*Pedro J Sanz*

*Campus de Riu Sec, Castellón*

*Diciembre de 2006*

## TEMA I PRELIMINARES

### Objetivos:

- *Introducir al alumno en el campo de la Robótica*  
Definiciones Previas  
Perspectiva Histórica
- *Dar una visión de conjunto a partir de aspectos claves como:*  
Campos del Conocimiento con los que se relaciona  
Su Interacción con la Sociedad Actual y Tendencias Futuras

### 1.1. ¿Qué es un Robot?

**Definición 1:** Un aparato mecánico que se *parece* y *hace* el trabajo de un ser humano. (Según el *Oxford English Dictionary*).

**Definición 2:** Un manipulador *reprogramable* y multifuncional concebido para transportar materiales, piezas, herramientas, o dispositivos especiales a través de movimientos programados variables para llevar a cabo tareas diversas. (Según el *Robot Institute of America* [Schlussel, 85]). Mediatizado por antecesores como Teleoperadores y CNC<sup>3</sup>.

**Definición 3:** Una máquina que puede ser programada para realizar una gran variedad de tareas, del mismo modo que un ordenador es un circuito electrónico que puede ser programado para llevar a cabo diferentes tareas. (Según [McKerrow, 86]).

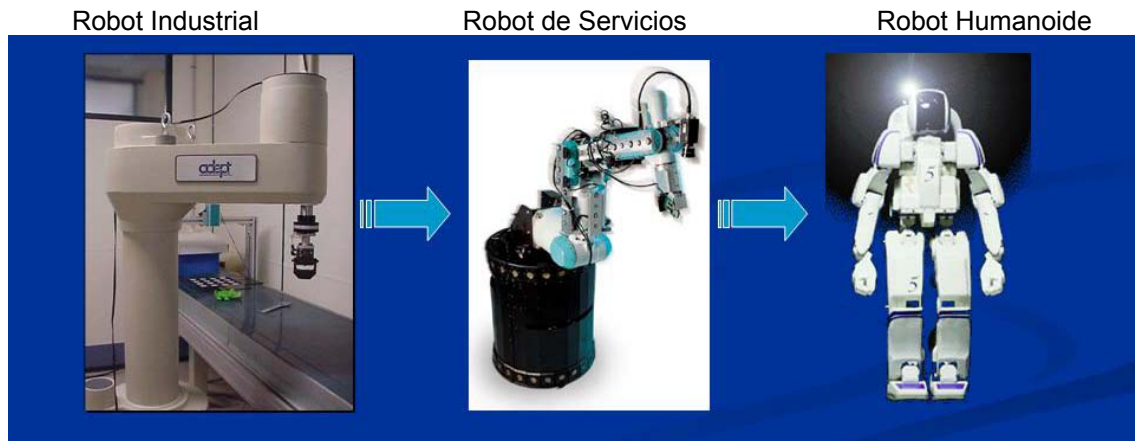
**Definición 4:** Un agente artificial, activo, cuyo entorno es el mundo físico. (Según [Russell & Norvig, 96]).

**Definición 5:** Un “robot inteligente” es una criatura mecánica que puede funcionar de manera autónoma. (Según [Murphy, 2000]).

Compárense las definiciones anteriores con una pequeña muestra de distintos robots actuales, cada uno adaptado al contexto de la aplicación para la que ha sido diseñado, en las tablas 1.1, y 1.2.

Seguramente, sería más acertado definir el robot con arreglo a las especificaciones para las que ha sido diseñado. Por ejemplo, hoy en día, se trabaja en un tipo de robot llamado “robot de servicios”, el cual puede considerarse como un estadio intermedio en la evolución desde el robot industrial hasta el futuro robot humanoide [ECAI2000], véase la tabla 1.1. Dichos robots han de ser manipuladores móviles, dotados de interfaces avanzadas hombre-robot, capaces de soportar comunicación próxima al lenguaje natural, y de desarrollar tareas complejas de manera autónoma en entornos reales (i.e., 3D, dinámicos, no estructurados, etc., es decir, con incertidumbre máxima).

<sup>3</sup>máquinas de control numérico programables



**Tabla 1.1.** Tres estadios fundamentales en la evolución de la Robótica actual.



**Tabla 1.2.** Tres aplicaciones no industriales de nuestros días.

### 1.2. ¿Qué es la Robótica?

**Definición 1:** Es la conexión *inteligente* entre percepción y acción. [Brady, 85].

**Definición 2:** Es la disciplina que involucra [McKerrow, 86]:

- a) el diseño, fabricación, control, y programación de robots;
- b) el uso de robots para resolver problemas;
- c) el estudio de los procesos de control, sensores y algoritmos usados en humanos, animales y máquinas; y
- d) la aplicación de estos procesos de control y algoritmos al diseño de robots.

### 1.3. Campos de Estudio de la Robótica

Cinemática	Control	Percepción
Dinámica	Sensores	Arquitectura
Planificación de Sistemas	Lenguajes de Programación	Inteligencia

## 1.4. Componentes y Estructura de los Robots

### Observación:

En lo que sigue centraremos nuestro estudio en aquellos robots dotados de capacidad de manipulación de su entorno, es decir, en el denominado “robot manipulador”, con independencia de si son móviles o estáticos.

Básicamente, podemos hablar de cuatro componentes:

- El **Manipulador**
  - Los **Sensores**
  - La **Unidad de Control**
  - La **Unidad de Conversión de Potencia**
- El Manipulador
    - Constituye la **estructura mecánica** del robot. Conjunto de barras conectadas por pares cinemáticos de modo que constituyan una **cadena cinemática abierta**.
  - Los Sensores
    - Internos**  
Informan a la unidad de control acerca del estado en el que se encuentra el manipulador, relativo al sistema de referencia del robot tales como los ángulos de sus articulaciones, fuerza o presión en el elemento terminal, etc. **Información Propioceptiva.**
    - Externos**  
Informan a la unidad de control acerca del entorno del robot y de la posición del robot relativa al entorno. **Información Exteroceptiva.**
  - La Unidad de Control
    - Funciones básicas:**
      - Iniciación y finalización del movimiento de los componentes individuales del manipulador en una secuencia de puntos especificados.
      - Almacenamiento en su memoria de datos acerca de la posición y secuencia de movimientos
      - Permite al robot interactuar con el entorno por medio de sensores.
    - Tipos** (por orden creciente de complejidad):
      - Secuenciadores.
      - Sistemas Lógicos Neumáticos.
      - Secuenciadores Electrónicos.
      - Microordenadores.
  - La Unidad de Conversión de Potencia
    - Proporciona la **Energía** necesaria a los **Actuadores** del Manipulador

## 1.5. Especificaciones de un Robot

- Capacidad de Carga.  
*Máxima masa que puede movilizar bajo cualquier configuración de su estructura mecánica.*
- Movilidad.  
*Se determina por el nº de **Grados de Libertad** (o ejes) del Robot.*
- Espacio de Trabajo.  
*Región del Espacio compuesta por todos los puntos que pueden ser alcanzados por el final del brazo del robot (brida o TCP 0), sin considerar el elemento terminal.*  
*Un parámetro a tener en cuenta será el Grado de Redundancia (o accesibilidad).*
- Agilidad.  
*Se mide, fundamentalmente, por dos parámetros: la velocidad máxima y las aceleraciones y deceleraciones máximas.*
- Precisión y Repetitividad.  
*La **precisión** de un robot es su capacidad para posicionar el elemento terminal en un punto arbitrario predeterminado de su espacio de trabajo.*  
*La **repetitividad** es la medida de la habilidad del robot para situar su elemento terminal en un mismo punto repetidas veces.*
- Entorno de Trabajo.  
*La aplicación concreta o el entorno de trabajo al que se destine un robot, influirá en las características de su propio **diseño**.*

## 1.6. Perspectiva Histórica

Se incluye a continuación una breve cronología, donde el autor recoge algunos de los momentos considerados de mayor impacto sobre la materia que nos ocupa:

- 1497** *Leonardo da Vinci*. Primer estudio serio sobre un posible robot antropomorfo.
- 1801** Telar de *J. Jacquard*. Origen de las máquinas programables.
- 1818** “*Frankenstein*“ de *M. Shelley*. Análisis de posibles consecuencias de una máquina construída a semejanza del hombre.
- 1921** “Robot“, *Karel Capek*. Introduce en su famosa obra de teatro dicho concepto.
- 1942** “Robótica“, *Isaac Asimov*. Introduce esta palabra, así como las tres famosas leyes.
- 1948** “Cybernetics“, *Wiener* (MIT). Acuña este concepto de “cibernética”, que analiza las causas y los efectos del control en el hombre y en la máquina.
- 1954** Era de la Robótica, *Devol*.
- 1956** UNIMATION, *Engelberger*. Creador de la primera empresa de robots industriales.
- 1960** “I. A.“, *McCarthy* (MIT). Acuña dicho término (Inteligencia Artificial).
- 1965** Coordinación “Ojo-Mano“, *Minsky* (MIT). Desarrolla el primer sistema completo.
- 1968** Robot móvil, (Stanford). Primer robot autónomo (“Shakey”).
- 1970** Stanford Robot Arm.
- 1971** Robótica declarada de interés nacional (Japón, JIRA)
- 1975** Robot Institute of America
- 1976** NASA Viking 1 y 2. Sondas americanas que alcanzan la superficie de Marte.
- 1977** ASEA B. B. Robotics (ABB)
- 1978** PUMA (Programmable Universal Assembly), de Unimation, es desarrollado para la General Motors.

- 1980's** Robots móviles, se potencia la comunicación hombre-robot mediante visión, voz, etc.
- 1990's** Espacio (e.g. NASA Sojourner), hacia el robot de servicios (e.g. proyecto “robot humanoide“), etc.
- 2000's** 1er Congreso Internacional sobre “Robots humanoides“ [Humanoids'2000, MIT], Creación de EURON (Red Europea de Robótica), “Web Robotics”, etc.

## 1.7. Generaciones de Robots

- **Primera Generación**

Los robots industriales, llamados clásicos. Utilizan un lenguaje de programación por guiado/gestual/textual. Funcionan en modo autómatas programables. No utilizan información externa que les permita conocer su entorno o sus movimientos relativos a dicho entorno.

- **Segunda Generación**

Estos robots están dotados de *sensores* que les permiten obtener información de sí mismos y de su entorno. Por ej., serán capaces de localizar y reconocer piezas gracias a la visión, o bien detectar un obstáculo gracias a detectores de proximidad, o bien realizar un ensamblado midiendo las fuerzas que se desarrollan durante su ejecución e intentando anularlas mediante movimientos adecuados, etc.

- **Tercera Generación**

Está en fase de investigación. Intenta potenciar al máximo la capacidad de percepción, utilizando para ello los logros conseguidos por los diferentes paradigmas de la *Inteligencia Artificial*. Pretende hacer uso extensivo de la *comprensión del lenguaje natural*, de la *interpretación visual del entorno*, mediante *visión 3D*, *color*, *texturas* etc., de la *capacidad de aprendizaje* y de una completa interrelación que le permita elaborar sus propios *planes* para alcanzar los objetivos que el mismo construirá a partir de ciertas directrices iniciales, en un mundo totalmente dinámico y cambiante.

## 1.8. Expansión Industrial de los Robots

Extraído de AER (Asociación Española de Robótica, 1996). Se muestra a continuación la “densidad de robots”, es decir, el nº de robots por cada 10.000 trabajadores de la industria, y el “índice de paro”, de los países más significativos al respecto.

País	Densidad de robots	Índice de Paro	Nº de Robots
------	--------------------	----------------	--------------

Japón	250	3%	413.578
Suecia	60	9%	5.911
Alemania	58	9%	56.175
Italia	55	12%	25.096
EEUU	35	7%	65.198
Francia	33	12%	14.376
España	22	22%	5.346

### RESUMEN T-I. Aspectos Clave:

- Un robot puede descomponerse en un conjunto de subsistemas funcionales: ***mecánicos, eléctricos, sensores, software, procesos, planificación y control.***
- El subsistema de ***software*** permite ***integrar*** todos los subsistemas del robot.
- Un brazo articulado emula las principales características de un brazo humano pero, a diferencia del brazo humano, ***cada articulación posee solo un grado de libertad.***
- Los manipuladores pueden clasificarse conforme a la ***geometría de su espacio de trabajo***, el cual es consecuencia de las ***articulaciones*** usadas en su construcción: ***Cartesianas, cilíndricas, polares*** o de ***revolución.***
- El movimiento de un robot puede describirse en cuatro sistemas de coordenadas diferentes: ***motor, articulaciones, universal y herramienta.***
- Los ***grados de libertad*** de un objeto son el ***nº de movimientos independientes*** que puede realizar respecto a un sistema de referencia. Un max. de 6 para un sist. Cartesiano: ***3 traslacionales y 3 rotacionales.***
- El ***grado de movilidad*** de un robot es el ***nº de articulaciones*** independientes que posee.
- En el desarrollo de cualquier aplicación real con robots, una parte de tiempo importante se destina a la tarea del diseño de ***garras específicas*** para dicha aplicación.
- El mejor software no puede impedir los posibles problemas causados por un ***diseño mecánico inadecuado.***



## TEMA II EL ROBOT MANIPULADOR

### Objetivos:

- *Desarrollar Herramientas para Modelar las Relaciones Geométricas entre Objetos usando Sistemas de Coordenadas y Transformaciones Homogéneas.*
- *Usar dichas herramientas para Modelar las Relaciones Espaciales entre las Ligaduras del Manipulador.*
- *Calcular Posición y Orientación del elemento terminal a partir de valores medidos de los ángulos de las articulaciones (modelo Directo) y estimar los ángulos de las articulaciones requeridos para situar al elemento terminal en la localización deseada del espacio (modelo Inverso).*
- *Extender el modelo Cinemático del Manipulador para incluir el tiempo.*

### 2.1 LOCALIZACIÓN ESPACIAL

#### 2.1.1. Preliminares

**Manipulación** es la habilidad en el manejo y el tratamiento de objetos: levantarlos, moverlos, fijar unos a otros y trabajarlos con herramientas (taladros, fresas, etc.).

Se requiere un **método** para especificar **donde está el objeto relativo a la mano del robot**, y un **modo de controlar el movimiento** de dicha **mano**.

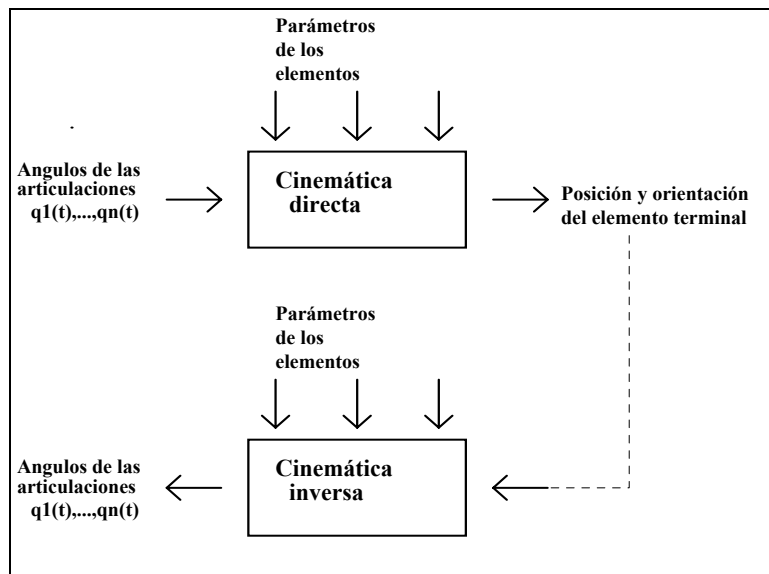
La **gente** usa una combinación de **realimentación visual, táctil y de posicionamiento del brazo** cuando coge un objeto.

La mayoría de **robots** calculan la posición de su mano usando un **modelo cinemático** de su brazo.

En la figura 2.1 se muestra la problemática relativa a la solución de los problemas cinemático directo e inverso, respectivamente.

Abordaremos esta problemática haciendo uso de:

*Álgebra Matricial y Razonamiento Geométrico*



**Figura 2.1.** Representación esquemática de los problemas cinemáticos directo e inverso.

### 2.1.2. Coordenadas Homogéneas y Matriz de Transformación

En un espacio tridimensional, un vector de posición  $p = (p_x, p_y, p_z)^T$  se representa por un vector ampliado  $p = (wp_x, wp_y, wp_z, w)^T$  en la representación en coord. homogéneas. Las coord. físicas se relacionan con las homogéneas como:

$$p_x = wp_x / w, p_y = wp_y / w, p_z = wp_z / w$$

La matriz de transformación homogénea es una matriz 4 x 4 que transforma un vector de posición expresado en coord. homogéneas desde un sistema de coord. a otro. En general se representa:

$$T = \begin{bmatrix} R_{3 \times 3} & P_{3 \times 1} \\ F_{1 \times 3} & 1 \times 1 \end{bmatrix} = \begin{bmatrix} \text{rotación} & \text{traslación} \\ \text{perspectiva} & \text{escalado} \end{bmatrix}$$

Ejemplo de rotación pura:	Ejemplo de traslación pura:
$T_{x,\alpha} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \alpha & -\text{sen } \alpha & 0 \\ 0 & \text{sen } \alpha & \cos \alpha & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$	$T_{\text{tras}} = \begin{bmatrix} 1 & 0 & 0 & px \\ 0 & 1 & 0 & py \\ 0 & 0 & 1 & pz \\ 0 & 0 & 0 & 1 \end{bmatrix}$

Los elementos de la diagonal principal de una matriz de transf. homogénea producen escalado local y global.

### 2.1.3. Interpretación Geométrica de las M. de T.Homogéneas

En general, una M.de T. Homogénea para un espacio tridimensional y en el contexto de robótica, se representará:

$$\vec{p}_{xyz} = T\vec{p}_{uvw}$$

$$T = \begin{bmatrix} n_x & s_x & a_x & p_x \\ n_y & s_y & a_y & p_y \\ n_z & s_z & a_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} \vec{n} & \vec{s} & \vec{a} & \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Una matriz de transf. homogénea geoméricamente representa la *localización*<sup>4</sup> de un sistema de coordenadas ligado al cuerpo, con respecto a un sistema de coord. de referencia.

En general la inversa de una matriz de transformación homogénea se puede representar como:

$$T^{-1} = \begin{bmatrix} n_x & n_y & n_z & -n^T \vec{p} \\ s_x & s_y & s_z & -s^T \vec{p} \\ a_x & a_y & a_z & -a^T \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} & & & -n^T \vec{p} \\ & R_{3 \times 3}^T & & -s^T \vec{p} \\ & & & -a^T \vec{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

#### 2.1.4. Transformación Homogénea Compuesta

Reglas a seguir:

- 1º) Inicialmente ambos sistemas de coordenadas son coincidentes, lo que implica que la matriz de transformación homogénea será la identidad,  $I_4$ , de orden 4 x 4.
- 2º) Si el sistema de coord. rotante  $OUVW$  está rotando/trasladándose respecto de uno de los ejes principales del sistema  $OXYZ$ , entonces premultiplicar la matriz de trans. homogénea previa (resultante) por una matriz de traslación/rotación básica apropiada
- 3º) Si el sistema de coord. rotante  $OUVW$  está rotando/trasladándose respecto de su propio eje principal, entonces postmultiplicar la matriz de trans. homogénea previa (resultante) por una matriz de rotación/traslación básica apropiada

#### 2.1.5. Grafo de una Transformación

El problema de localizar un objeto se puede descomponer en dos etapas:

- 1ª) Describir el objeto respecto de un sist. de coordenadas solidario con él.
- 2ª) Describir el sist. de coordenadas del objeto mediante una transformación del sist. de referencia.

Estas transformaciones pueden visualizarse a partir del llamado *grafo de la transformación*, ideado por Paul [Paul, 81]. Se trata de un grafo cerrado, lo que implica que cualquier transformación puede describirse en términos del resto de transformaciones del grafo.

<sup>4</sup> Localización = posición + orientación

## 2.2 CINEMÁTICA DEL MANIPULADOR

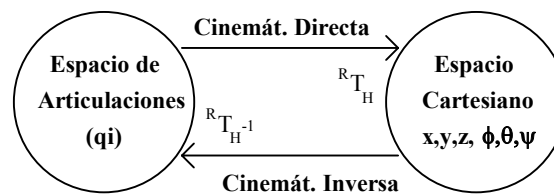
**Cinemática:** Establece las relaciones entre las **posiciones, velocidades y aceleraciones** de las ligaduras de un manipulador.

### Observación (Clasificación de los Manipuladores):

Partimos de la hipótesis de que la conexión entre elementos (las articulaciones) tienen solamente un grado de libertad. Con esta restricción son de interés dos tipos de articulaciones: De Revolución (permiten la rotación respecto de un eje) y Prismáticas (permiten el deslizamiento a lo largo de un eje). Un manipulador, considerado como una combinación de elementos y articulaciones, con el primer elemento conectado a la base y el último conteniendo la "mano", se puede clasificar por el tipo de articulaciones y su orden (desde la base hasta la mano). Ej. el robot PUMA será del tipo 6R, y el brazo de Stanford será 2R-P-3R.

### 2.2.1 CINEMÁTICA DE POSICIÓN

Básicamente, los dos problemas a resolver en la cinemática de posición se pueden resumir en la figura siguiente:



#### 2.2.1.1. La Representación de Denavit-Hartenberg [1955]

La representación de D-H de un elemento rígido depende de cuatro parámetros geométricos asociados con cada elemento (véase la figura 2.2):

$a_i$  ( $l_i$ ): Longitud de la normal común  $H_iO_i$   
 $d_i$ : Distancia del origen  $O_{i-1}$  al punto  $H_i$   
 $\alpha_i$ : Ángulo entre los ejes  $Z_{i-1}$  y  $Z_i$ , medido alrededor de  $X_i$  en sentido positivo  
 $\theta_i$ : Ángulo entre el eje  $X_{i-1}$  y la normal común  $H_iO_i$ , medido alrededor de  $Z_{i-1}$  en sentido positivo

Una vez establecido el sist. de coord. de D-H para cada elemento se procede a encontrar la transformación homogénea:  ${}^{i-1}A_i$ , que representa la relación existente entre dos articulaciones consecutivas del manipulador. De la figura 2.2, empleando  $\{H_i-X' Y' Z'\}$  como sistema de referencia intermedio, se llega a:

$${}^{i-1}A_i = \text{Trans}(z', d_i) \text{Rot}(z', \theta_i) \text{Trans}(x_i, a_i) \text{Rot}(x_i, \alpha_i)$$

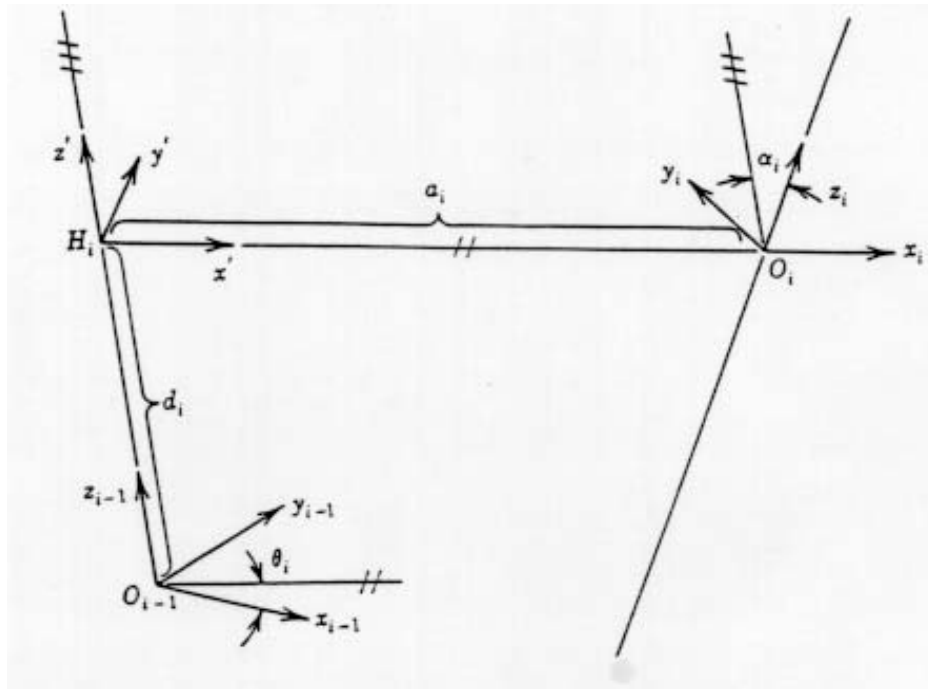


Figura 2.2. Relación entre sistemas de referencia adyacentes.

Un manipulador con  $n$  grados de libertad ( $n$  articulaciones), se modelará con  $n+1$  sistemas de referencia. Considerando las  $n$  transformaciones consecutivas sobre la secuencia de barras adyacentes, se podrá obtener la localización del elemento terminal con respecto a la base del robot, es decir:

$${}^R T_H = {}^0 A_1 {}^1 A_2, \dots, {}^{n-1} A_n$$

donde la matriz  ${}^R T_H$ , describe la posición y orientación del elemento terminal (H) con respecto a un sistema de referencia ligado a la base del manipulador (R). En el contexto de la robótica industrial lo habitual serán dos tipos básicos de articulaciones, cada una asociada con un sólo grado de libertad, o bien de rotación o bien prismática. En el primer caso (rotación), la matriz  ${}^R T_H$  correspondiente será:

$$\begin{bmatrix} \cos\theta_i & -\cos\alpha_i \operatorname{sen}\theta_i & \operatorname{sen}\alpha_i \operatorname{sen}\theta_i & a_i \cos\theta_i \\ \operatorname{sen}\theta_i & \cos\alpha_i \cos\theta_i & -\operatorname{sen}\alpha_i \cos\theta_i & a_i \operatorname{sen}\theta_i \\ 0 & \operatorname{sen}\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Y, para las prismáticas:

$$\begin{bmatrix} \cos\theta_i & -\cos\alpha_i \operatorname{sen}\theta_i & \operatorname{sen}\alpha_i \operatorname{sen}\theta_i & 0 \\ \operatorname{sen}\theta_i & \cos\alpha_i \cos\theta_i & -\operatorname{sen}\alpha_i \cos\theta_i & 0 \\ 0 & \operatorname{sen}\alpha_i & \cos\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

### 2.2.1.2 Problema Cinemático Directo

Algoritmo<sup>5</sup> de Resolución del Problema Cinemático Directo

- 1.- Mover el manipulador a su posición cero.
- 2.- Asignar un Sist. de Coord. a cada elemento ("link").
- 3.- Describir las posiciones y orientaciones entre elementos con los parámetros de D-H correspondientes.
- 4.- Construir las matrices A que relacionan dichos elementos.
- 5.- Calcular a partir de aquí la matriz asociada a la transformación del manipulador:  ${}^R T_H$ .

$$6.- \text{Plantear la Ecuación: } {}^R T_H = \begin{bmatrix} \bar{x} & \bar{y} & \bar{z} & \bar{p} \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Para obtener coordenadas cartesianas en términos de coordenadas de articulación. En general, se cumplirá:  ${}^R T_H = T_{\text{posición}} T_{\text{rotación}}$

### Transformación de Orientación General

Problemas que plantea el movimiento de un manipulador en cualquier dirección del espacio tridimensional:

- 1.- Los ángulos de orientación son difíciles de evaluar, debido a que los elementos de la matriz contendrán términos con múltiples ángulos.
- 2.- No resulta fácil visualizar la orientación de un vector general. Lo que implica que el proceso de descomposición en las tres rotaciones básicas será difícil.

De 1 y 2, se plantea resolver la Transformación de Orientación "RPY" (*Roll, Pitch, Yaw*) General:

$${}^R T_H = \text{Trans}(p_x, p_y, p_z) \text{RPY}(\phi, \theta, \psi) = \text{Trans}(p_x, p_y, p_z) \text{Rot}(z, \phi) \text{Rot}(y, \theta) \text{Rot}(x, \psi)$$

Lo que implica,

$$\theta = -\sin^{-1}(x_z), \psi = \sin^{-1}(y_z/\cos(\theta)), \phi = \sin^{-1}(x_y/\cos(\theta))$$

Estas soluciones plantean problemas debido a imprecisiones en el cálculo de las funciones transcendentales *sin* y *cos*. Por lo que introduciremos la función: **atan2**, (véase *Apéndice 2*).

Así pues, para obtener las ecuaciones anteriores en términos de la función *atan2*, hacemos:

$${}^R T_H = \text{Trans}(p_x, p_y, p_z) \text{Rot}(z, \phi) \text{Rot}(y, \theta) \text{Rot}(x, \psi)$$

$$\text{Rot}^{-1}(z, \phi) \text{Trans}^{-1}(p_x, p_y, p_z) {}^R T_H = \text{Rot}(y, \theta) \text{Rot}(x, \psi)$$

Y se llega a la ecuación matricial siguiente,

<sup>5</sup> Adaptado de [McKerrow, 91]

$$\begin{bmatrix} Xx\cos(\phi) + Xy\sin(\phi) & Yx\cos(\phi) + Yy\sin(\phi) & Zx\cos(\phi) + Zy\sin(\phi) & 0 \\ Xy\cos(\phi) - Xx\sin(\phi) & Yy\cos(\phi) - Yx\sin(\phi) & Zy\cos(\phi) - Zx\sin(\phi) & 0 \\ Xz & Yz & Zz & pz \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
 = \begin{bmatrix} C(\theta) & S(\theta)S(\psi) & S(\theta)C(\psi) & 0 \\ 0 & C(\psi) & -S(\psi) & 0 \\ -S(\theta) & C(\theta)S(\psi) & C(\theta)C(\psi) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

donde igualando términos se obtiene:

$$\begin{cases} \phi = \text{atan2}(Xy, Xx) \\ \theta = \text{atan2}(-Xz, Xx\cos(\phi) + Xy\sin(\phi)) \\ \psi = \text{atan2}(Yz, Zz) \end{cases}$$

**Observación:**

Se observa que la suma  $C(\phi) + S(\phi)$ , elimina la división por cero y minimiza imprecisiones. Esta solución es menos eficiente que la anterior, pero es más *ROBUSTA*.

### 2.2.1.3. Problema Cinemático Inverso

Se trata de resolver un sistema de ecuaciones no lineales, y habrá que plantearse:

- 1.- La existencia o no de soluciones
- 2.- El que existan soluciones múltiples
- 3.- Un método general para buscar la solución

#### Existencia de soluciones

Está directamente relacionada con el Espacio de Trabajo del manipulador. Podemos distinguir entre:

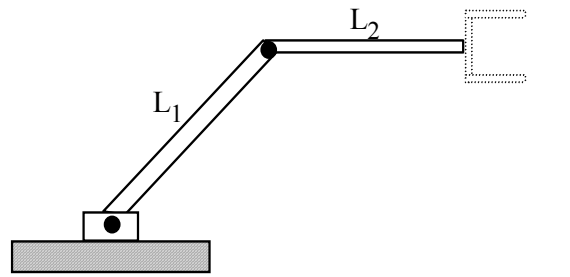
- a) Espacio de Trabajo Diestro. Volumen del espacio que el robot puede alcanzar con su TCP 0, en cualquier orientación.
- b) Espacio de Trabajo Alcanzable. Volumen del espacio que el robot puede alcanzar con su TCP 0, en, al menos, una orientación.

Lo anterior implica que el Espacio de Trabajo Diestro es un subconjunto del Espacio de Trabajo Alcanzable.

*Ejemplo*<sup>6</sup>.- Supongamos un manipulador de dos articulaciones, 2R planar, caracterizado por  $l_1$  y  $l_2$ .

Si  $l_1 = l_2$  Implica que el espacio de trabajo alcanzable será un disco de radio  $2l_1$ . El espacio de trabajo diestro constará de un único punto, el origen.

Si  $l_1 \neq l_2$  Implica que el espacio de trabajo alcanzable será un disco anular de radio exterior  $l_1 + l_2$  y radio interior  $|l_1 - l_2|$ . Y el espacio de trabajo diestro no existirá.



En el interior del espacio de trabajo alcanzable habrá dos posibles orientaciones del TCP 0. Mientras que en los bordes de dicho espacio solo existirá una determinada orientación.

Observación: Cuando un manipulador posee menos de 6 grados de libertad, no puede alcanzar determinados puntos de su entorno, que requieren posiciones y orientaciones imposibles para él. Especificado el sist. de coord. asociado a un punto "objetivo", un problema interesante a resolver por estos manipuladores será: ¿Cuál es el punto alcanzable más cercano al objetivo?. O ¿Está dicho punto en el E. de T. Alcanzable?. Sí se encuentra que el sist. de coord. del TCP-0, con la posición y orientación deseadas está dentro del E. de T. Alcanzable implica que existe al menos una solución.

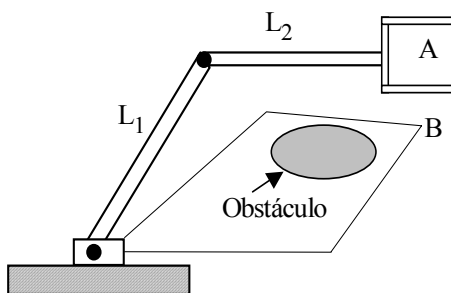
#### Soluciones Múltiples

El hecho de que un manipulador posea múltiples soluciones puede causar problemas debido a que el sistema tiene que ser capaz de elegir una.

*Ejemplo*.- Suponiendo el manipulador del ejemplo anterior, se observa el problema de la solución múltiple., en este caso, “codo arriba” ó “codo abajo”, (véase la figura siguiente).

<sup>6</sup> Adaptado de [Craig, 89]





(1) Un posible criterio de elección sería el de la solución más próxima. En la figura, para pasar del pto. A al B, una buena elección (en ausencia de obstáculos), sería la solución que minimiza los valores de las articulaciones que es preciso mover. Se elegiría la configuración superior (“codo arriba”). Esto sugiere que un argumento de entrada a nuestro Prob. Cinemático Inverso, podría ser la posición actual del manipulador.

(2) Pero, y si hay que evitar obstáculos?. En el ejemplo anterior implicaría la elección de la configuración inferior (“codo abajo”).

(1) y (2) implican que, en general, se necesita calcular todas las soluciones posibles.

El nº de soluciones depende del nº de articulaciones del manipulador, pero teniendo en cuenta que es función también de los parámetros de D-H asociados ( $\alpha_i$ ,  $a_i$ , y  $d_i$  para una articulación de revolución), y del rango permitido de movimiento de las articulaciones.

### Un método de solución

No existe un algoritmo general que pueda emplearse para resolver un sistema de ecuaciones no lineales.

Lo primero será definir QUE constituye la "solución" de un manipulador dado.

DEF.- Un manipulador se considerará *resoluble* SI las variables de articulación pueden determinarse mediante un algoritmo que permite determinar todos los conjuntos de variables de articulación asociados con una posición y orientación dadas [Roth, 75].

Podemos dividir todas las posibles estrategias propuestas para encontrar solución en:

- 3 Soluciones en Forma Cerrada
- 4 Soluciones Numéricas

Nos centraremos exclusivamente en el primer tipo:

- Soluciones en Forma Cerrada
  - { Métodos Algebráico s
  - { Métodos Geométrico s

En este contexto "forma cerrada" significa un método de solución basado en expresiones analíticas o en la solución de un polinomio de grado  $\leq 4$ , de manera que cálculos no iterativos sean suficientes para llegar a una solución.

### Observación:

Un resultado reciente en cinemática es: Todos los sistemas con articulaciones prismáticas y de revolución que tengan un total de seis grados de libertad a lo largo de una cadena de elementos simples en serie es *resoluble*.

Pero CUIDADO!, esto es un resultado numérico. Sólo en casos especiales se podrá resolver analíticamente para los robots de seis articulaciones

La gran mayoría de los robots comerciales tienen una u otra de las siguientes condiciones suficientes que hacen posible la solución del brazo en forma cerrada:

- 1.- Tres ejes de articulación adyacentes se intersectan en un punto. (e.g., PUMA, Stanford).
- 2.- Tres ejes de articulación adyacentes son paralelos entre sí. (e.g., ASEA, MINIMOVER).

Una Aproximación Heurística para buscar la solución en Forma Cerrada

Recordemos Qué es lo que se requiere para abordar la construcción de un modelo cinemático de un manipulador cualquiera:

Se trata de expresar las variables de articulación en términos de las variables que describen el espacio cartesiano. Vamos a dar un método heurístico para buscar soluciones a partir de la ecuación general de transformación del manipulador. Pero

**CUIDADO!!**, dicho método no garantiza la existencia de solución, y además algunas soluciones encontradas pueden ser redundantes.

Para ver en que consiste, veamos primero un ejemplo.

**Ejemplo.**- Vamos a ilustrar la heurística de la cinemática inversa buscando la solución general para el manipulador de 2 articulaciones (no plano) definido por la tabla de parámetros de D-H:

$\theta$	$\alpha$	$a$	$d$
$\theta_1$	$\pi/2$	0	$d_1$
$\theta_2$	0	$a_2$	0

$${}^R T_H = \begin{bmatrix} C_1 C_2 & -C_1 S_2 & S_1 & l_2 C_1 C_2 \\ S_1 C_2 & -S_1 S_2 & -C_1 & l_2 S_1 C_2 \\ S_2 & C_2 & 0 & d_1 + l_2 S_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} X_x & Y_x & Z_x & p_x \\ X_y & Y_y & Z_y & p_y \\ X_z & Y_z & Z_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

A partir de aquí podemos encontrar  $\theta_1$  dividiendo las expresiones para  $p_y$  y  $p_x$ . Observemos que podríamos calcular  $\theta_1$  en términos de  $X_y$  y  $X_x$ , pero esto ha de evitarse siempre que sea posible, pues siempre será menos eficiente que buscar soluciones en términos de  $p_j$ . Se llega a :

$$\frac{p_y}{p_x} = \frac{l_2 S_1 C_2}{l_2 C_1 C_2} \Rightarrow \theta_1 = \text{atan2}(p_y, p_x)$$

Para resolver  $\theta_2$  podríamos utilizar la expresión para  $p_z$ , pero el resultado incluiría un *arcsin*, y esto origina imprecisiones. Hay que buscar una solución más robusta en términos de *atan2*. Para ello hacemos:

$$A^{-1} {}^R T_H = A_2$$

$$\begin{bmatrix} Xx C_1 + Xy S_1 & Yx C_1 + Yy S_1 & Zx C_1 + Zy S_1 & px C_1 + py S_1 \\ Xz & Yz & Zz & pz - d_1 \\ Xx S_1 - Xy C_1 & Yx S_1 - Yy C_1 & Zx S_1 - Zy C_1 & px S_1 - py C_1 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} C_2 & -S_2 & 0 & l_2 C_2 \\ S_2 & C_2 & 0 & l_2 S_2 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$px C_1 + py S_1 = l_2 C_2 \wedge pz - d_1 = l_2 S_2 \Rightarrow \theta_2 = \text{atan2}(pz - d_1, px C_1 + py S_1)$$

**Algoritmo<sup>7</sup>.** - *Enfoque Heurístico para la Resolución del Problema Cinemático Inverso:*

- 1.- Igualar la Matriz de Transformación General a la Matriz de Transformación del Manipulador.
- 2.- Examinar ambas matrices con vistas a encontrar:
  - elementos que contengan sólo una variable de articulación.
  - pares de elementos que al ser divididos llevarán a una expresión que sea función de una sola variable de articulación. En particular, buscar divisiones cuyo resultado sea función de atan2.
  - elementos, o combinaciones de elementos, que puedan simplificarse usando identidades trigonométricas.
- 3.- Habiendo seleccionado un elemento, igualarlo al correspondiente elemento en la otra matriz para construir una ecuación. Resolver esta ecuación para encontrar una descripción de una variable de articulación en términos de los elementos de la matriz de transformación general.
- 4.- Repetir el paso 3 hasta que todos los elementos identificados en el paso 2 hayan sido usados.
- 5.- Sí cualquiera de las soluciones encontradas posee imprecisiones, resultados indefinidos o redundantes (debido a términos en senos, cosenos etc.), guardarlas aparte y buscar mejores soluciones. Soluciones en términos del vector p conducirán a soluciones más eficientes que en términos de los vectores x, y, z, porque encontrar los términos asociados a dichos vectores puede involucrar la resolución de ecuaciones complejas, mientras que la posición deseada en el espacio del manipulador es conocida.
- 6.- Sí restan por encontrar más variables de articulación, premultiplicando ambos lados de la ecuación matricial por la inversa de la matriz A para la primera variable de articulación, se obtiene un nuevo conjunto de elementos matriciales equivalente.
- 7.- Repetir los pasos 2 al 6 hasta encontrar solución para todas las variables de articulación, o hasta acabar de premultiplicar (o postmultiplicar) las matrices A requeridas.
- 8.- Sí no se logra encontrar una solución adecuada para una variable de articulación, elegir una de las descartadas en el paso 5, tomando buena nota de las regiones que puedan darnos problemas.
- 9.- Sí no puede encontrarse una solución para una variable de articulación en términos de los elementos de la matriz de transf. del manipulador, puede ser que el manipulador no pueda alcanzar la posición y orientación especificadas, lo que implica que la posición está fuera del espacio de trabajo del manipulador. Además, soluciones teóricas pueden no ser físicamente alcanzables a causa de límites mecánicos en el rango de variación asociado a las variables de articulación.

<sup>7</sup> Adaptado de [McKerrow,91]

Vamos a ver un caso práctico de uso del algoritmo ant. para encontrar la solución general de una manipulador de dos articulaciones 2R (plano), definido por la tabla de parámetros de D-H:

$\theta$	$\alpha$	$a$	$d$
$\theta_1$	0	$a_1$	0
$\theta_2$	0	$a_2$	0

El primer paso, es plantear la ecuación:

$${}^R T_H = \begin{bmatrix} C_{12} & -S_{12} & 0 & l_1 C_1 + l_2 C_{12} \\ S_{12} & C_{12} & 0 & l_1 S_1 + l_2 S_{12} \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} X_x & Y_x & Z_x & p_x \\ X_y & Y_y & Z_y & p_y \\ X_z & Y_z & Z_z & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

En éste caso, no hay elementos con una sola variable de articulación, y premultiplicar o postmultiplicar no servirá de nada. Así que habrá que escoger dos expresiones y utilizar identidades trigonométricas para simplificarlas:

$$\begin{aligned} p_x &= l_1 C_1 + l_2 C_{12} \\ p_y &= l_1 S_1 + l_2 S_{12} \\ p_z &= 0 \end{aligned}$$

Las ecs. para  $p_x$  y  $p_y$ , son del tipo descrito como "class5"<sup>8</sup>. Obraremos en consecuencia, y operando se llega a:

$$p_x^2 + p_y^2 = l_1^2 + l_2^2 + 2l_1 l_2 C_2 \Rightarrow C_2 = \frac{p_x^2 + p_y^2 - l_1^2 - l_2^2}{2l_1 l_2}$$

Y de aquí obtenemos el seno y el ángulo:

$$S_2 = \pm \sqrt{1 - C_2^2} \Rightarrow \theta_2 = \text{atan2}(S_2, C_2)$$

Hemos encontrado una solución REDUNDANTE!.

Para encontrar  $\theta_1$  hacemos:

$$\frac{p_y}{p_x} = \frac{(l_1 + l_2 C_2)S_1 + l_2 S_2 C_1}{(l_1 + l_2 C_2)C_1 - l_2 S_2 S_1} = \frac{\tan(\theta_1) + l_2 S_2 / (l_1 + l_2 C_2)}{1 - \tan(\theta_1)(l_2 S_2 / (l_1 + l_2 C_2))}$$

Y operando se llega a:

$$\theta_1 = \text{atan2}(p_y, p_x) - \text{atan2}(l_2 S_2, l_1 + l_2 C_2)$$

Que sigue siendo una solución redundante.

<sup>8</sup> Véase [McKerrow, 91]

### 2.2.2 CINEMÁTICA DEL MOVIMIENTO

Def.- *Camino* ("path") es la secuencia de puntos en el espacio que describe por donde tiene que ir el elemento terminal del manipulador, fijados los ptos. inicial y final.

Def.- *Trayectoria* es un camino con restricciones temporales, esto es, incluye velocidad y aceleración, así como la localización de cada punto a lo largo del camino.

#### Observación:

Una estrategia de control usual, es calcular el *cambio en las variables de articulación* requerido para alcanzar la localización final, controlando los motores de las articulaciones de manera que todas las articulaciones alcancen sus valores finales al mismo tiempo ("Joint Interpolated Motion").

#### Objetivo:

Calcular una trayectoria en el *Espacio de Articulaciones* a partir de la trayectoria (localización, velocidad y aceleración) en el *Espacio Cartesiano*. Así pues, se requieren *Transformaciones Cinemáticas Inversas* para computar velocidades y aceleraciones de las articulaciones.

#### Un paralelismo: *Movimiento de Acomodación vs Movimiento Diferencial*

Los *Movimientos de Acomodación*, en el hombre, están relacionados con el uso de realimentación de la información sensorial (vista, tacto...) para realizar movimientos precisos y tienen un paralelismo con el llamado *Movimiento Diferencial* en robótica. Por ej., cuando se utiliza un sistema de visión para monitorizar la localización del elemento terminal. Para usar esta información, debemos de ser capaces de controlar el *movimiento diferencial* del robot.

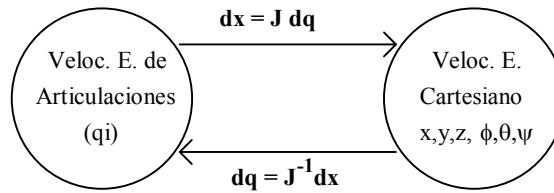
#### Observación:

Cuando se controla el movimiento con un ordenador, las señales de realimentación son leídas a intervalos discretos de tiempo. Sí se lee una señal de velocidad, se asume que ésta permanece cte. hasta la próxima lectura. Si se lee un valor para la posición o el desplazamiento, entonces la velocidad media es el desplazamiento durante el intervalo de muestreo. Por lo que "la medida y control del movimiento diferencial es análogo a la medida y control de la velocidad".

### Jacobiano del Manipulador

Def.- Dado un manipulador de  $n$  articulaciones, el Jacobiano asociado a dicho manipulador, representa la transformación instantánea entre el vector  $n$ -dimensional de velocidades de las articulaciones y el vector de 6 dimensiones conteniendo las velocidades lineales y angulares del elemento terminal. El jacobiano será por tanto una matriz  $6 \times n$ .

Análogamente al esquema visto para la cinemática de posición, veamos ahora, gráficamente, el esquema asociado a la cinemática del movimiento:



**Nota:** El mismo enfoque puede utilizarse para calcular la transformación entre las velocidades de articulación y las velocidades lineales y angulares de cualquier punto del manipulador.

En particular, veamos la representación para un manipulador de 6 articulaciones:

$$D = JD_q = (J_1, J_2, J_3, J_4, J_5, J_6) \cdot D_q$$

$$\begin{bmatrix} d_x \\ d_y \\ d_z \\ \delta_x \\ \delta_y \\ \delta_z \end{bmatrix} = \begin{bmatrix} d_{x1} & d_{x2} & d_{x3} & d_{x4} & d_{x5} & d_{x6} \\ d_{y1} & d_{y2} & d_{y3} & d_{y4} & d_{y5} & d_{y6} \\ d_{z1} & d_{z2} & d_{z3} & d_{z4} & d_{z5} & d_{z6} \\ \delta_{x1} & \delta_{x2} & \delta_{x3} & \delta_{x4} & \delta_{x5} & \delta_{x6} \\ \delta_{y1} & \delta_{y2} & \delta_{y3} & \delta_{y4} & \delta_{y5} & \delta_{y6} \\ \delta_{z1} & \delta_{z2} & \delta_{z3} & \delta_{z4} & \delta_{z5} & \delta_{z6} \end{bmatrix} \cdot \begin{bmatrix} dq_1 \\ dq_2 \\ dq_3 \\ dq_4 \\ dq_5 \\ dq_6 \end{bmatrix} = \begin{bmatrix} J_{11} & J_{12} & J_{13} & J_{14} & J_{15} & J_{16} \\ J_{21} & J_{22} & J_{23} & J_{24} & J_{25} & J_{26} \\ J_{31} & J_{32} & J_{33} & J_{34} & J_{35} & J_{36} \\ J_{41} & J_{42} & J_{43} & J_{44} & J_{45} & J_{46} \\ J_{51} & J_{52} & J_{53} & J_{54} & J_{55} & J_{56} \\ J_{61} & J_{62} & J_{63} & J_{64} & J_{65} & J_{66} \end{bmatrix} \cdot \begin{bmatrix} dq_1 \\ dq_2 \\ dq_3 \\ dq_4 \\ dq_5 \\ dq_6 \end{bmatrix}$$

Observaciones:

$$d_x = d_{x1} dq_1 + d_{x2} dq_2 + d_{x3} dq_3 + d_{x4} dq_4 + d_{x5} dq_5 + d_{x6} dq_6$$

es la componente x del movimiento diferencial del elemento terminal en función del movimiento diferencial de las articulaciones,

$\delta_y$  es la componente y del movimiento angular del elemento terminal en función del movimiento diferencial de las articulaciones,

$$d_{x1} = J_{11} = \partial x / \partial q_1$$

es la derivada parcial de la componente x de la posición del elemento terminal con respecto a la variable de articulación 1,

$$\delta_{y6} = J_{56}$$

es la derivada parcial de la componente y de la orientación del elemento terminal con respecto a la variable de articulación 6,

Veamos a continuación dos estrategias de evaluación del Jacobiano:

- *Cálculo por Diferenciación*

El Jacobiano de cualquier manipulador puede calcularse por diferenciación de su transformación cinemática directa. Este método puede resultar dificultoso para un manipulador con 6 grados de libertad, pero para uno de dos resulta bastante simple:

Ejemplo.- Sup. el manipulador planar de dos articulaciones, se obtiene:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} l_1 C_1 + l_2 C_{12} \\ l_1 S_1 + l_2 S_{12} \end{bmatrix} \Rightarrow \begin{bmatrix} dx \\ dy \end{bmatrix} = \begin{bmatrix} -l_1 S_1 - l_2 S_{12} & -l_2 S_{12} \\ l_1 C_1 + l_2 C_{12} & l_2 C_{12} \end{bmatrix} \cdot \begin{bmatrix} d\theta_1 \\ d\theta_2 \end{bmatrix}$$

**OJO!**, el Jacobiano obtenido se define respecto al sistema de referencia, "frame 0". Para hallar el Jacobiano con respecto al elemento terminal, haremos:

$${}^0J = {}^0Rot_3 \cdot {}^3J \Rightarrow {}^3J = {}^0Rot_3^{-1} \cdot {}^0J = \begin{bmatrix} C_{12} & S_{12} \\ -S_{12} & C_{12} \end{bmatrix} \cdot \begin{bmatrix} -l_1S_1 - l_2S_{12} & -l_2S_{12} \\ l_1C_1 + l_2C_{12} & l_2C_{12} \end{bmatrix} = \begin{bmatrix} l_1S_2 & 0 \\ l_1C_2 + l_2 & l_2 \end{bmatrix}$$

Análogamente podemos calcular  $J^{-1}$ , y se obtiene:

$$\begin{bmatrix} d\theta_1 \\ d\theta_2 \end{bmatrix} = \frac{1}{l_1l_2S_2} \cdot \begin{bmatrix} l_2C_{12} & l_2S_{12} \\ -l_1C_1 - l_2C_{12} & -l_1S_1 - l_2S_{12} \end{bmatrix} \cdot \begin{bmatrix} dx \\ dy \end{bmatrix}$$

- *Cálculo mediante Coordenadas Homogéneas*

Vamos a estudiar el algoritmo desarrollado por [Paul, 81], no por ser el más eficiente, sino porque hace uso de las coord. homogéneas y por tanto es consistente con la formulación matemática que hemos utilizado.

En un manipulador, un cambio diferencial en la localización del elemento terminal es una función de la transf. del movimiento diferencial con respecto a él:  $d^R T_E = {}^R T_E \cdot {}^E \Delta$  ♣

Para simplificar el análisis, consideremos el movimiento diferencial de sólo una articulación. La transformación del movimiento diferencial ( ${}^E \Delta$ ) se divide en dos partes:

${}^E \Delta = {}^E \Delta_n dq_n$ , un movimiento diferencial en coord. de articulación ( $dq_n$ ) multiplicado por una transformación diferencial desde la articulación n al sistema de coord. del elemento terminal ( ${}^E \Delta_n$ )

Para transformar el movimiento de articulación desde el sist. de ref. ligado a la misma ("frame n-1") al asociado con el elemento terminal (E), reescribimos la ec. ant. como:

${}^E \Delta_n = {}^{n-1} T_E^{-1} \cdot {}^{n-1} \Delta_n \cdot {}^{n-1} T_E$ , donde  ${}^{n-1} \Delta_n$  es una transf. de movimiento diferencial, correspondiente a una unidad de rotación diferencial alrededor del eje z, si la articulación n es de revolución, y a una unidad diferencial de traslación a lo largo del eje z, si la articulación n es prismática.

Y sustituyendo en ♣, se obtiene:

$$\begin{aligned} d^R T_E &= {}^R T_E \cdot {}^{n-1} T_E^{-1} \cdot {}^{n-1} \Delta_n \cdot {}^{n-1} T_E \cdot dq_n = {}^R T_{n-1} \cdot {}^{n-1} \Delta_n \cdot {}^{n-1} T_E \cdot dq_n = \\ &= A_1 L A_{n-1} \cdot {}^{n-1} \Delta_n \cdot A_n L A_E \cdot dq_n \end{aligned}$$

Esta ecuación describe el movimiento cartesiano del elemento terminal como una función del movimiento diferencial de la articulación n.

**Algoritmo** [Paul,81], para encontrar el Jacobiano del Manipulador con respecto al Sist. de Ref. del Elemento Terminal.

1.- Empezar en la articulación 1, donde n=1,

$$\begin{aligned} {}^E \Delta_1 &= {}^R T_E^{-1} \cdot {}^0 \Delta_1 \cdot {}^R T_E \\ d^0 T_E &= {}^R T_{n-1} \cdot {}^{n-1} \Delta_n \cdot {}^{n-1} T_E \cdot dq_n = {}^R T_R \cdot {}^0 \Delta_1 \cdot {}^R T_E \cdot dq_1 \\ &= {}^0 \Delta_1 \cdot A_1 L A_E \cdot dq_n \end{aligned}$$

$$\text{y, } {}^{n-1} T_E = {}^R T_E = A_1 L A_E$$

2.- Obtener la 1ª columna del Jacobiano, el vector asociado al movimiento diferencial para la articulación 1, (n=1); los elementos de  ${}^0 \Delta_1$ .

Caso 1: Articulación de Revolución: (rotación diferencial alrededor del eje z)

$$J_1 = [x \cdot \delta \times p \quad y \cdot \delta \times p \quad z \cdot \delta \times p \quad \delta \cdot x \quad \delta \cdot y \quad \delta \cdot z]$$

Donde, debido a que  $\delta x = \delta y = 0$ , se obtiene,

$$\begin{aligned}
 x \cdot \delta \times p &= \frac{{}^E \partial x}{\partial \theta_n} = J_{1n} = {}^{n-1}x_y \cdot {}^{n-1}p_x - {}^{n-1}x_x \cdot {}^{n-1}p_y = {}^0x_y \cdot {}^0p_x - {}^0x_x \cdot {}^0p_y \\
 y \cdot \delta \times p &= \frac{{}^E \partial y}{\partial \theta_n} = J_{2n} = {}^{n-1}y_y \cdot {}^{n-1}p_x - {}^{n-1}y_x \cdot {}^{n-1}p_y = {}^0y_y \cdot {}^0p_x - {}^0y_x \cdot {}^0p_y \\
 z \cdot \delta \times p &= \frac{{}^E \partial z}{\partial \theta_n} = J_{3n} = {}^{n-1}z_y \cdot {}^{n-1}p_x - {}^{n-1}z_x \cdot {}^{n-1}p_y = {}^0z_y \cdot {}^0p_x - {}^0z_x \cdot {}^0p_y \\
 \delta \cdot x &= \frac{{}^E \partial \delta x}{\partial \theta_n} = J_{4n} = {}^{n-1}x_z; \delta \cdot y = \frac{{}^E \partial \delta y}{\partial \theta_n} = J_{5n} = {}^{n-1}y_z; \delta \cdot z = \frac{{}^E \partial \delta z}{\partial \theta_n} = J_{6n} = {}^{n-1}z_z
 \end{aligned}$$

Caso 2: Articulación Prismática: (traslación diferencial a lo largo del eje z)

$$J_1 = [d \cdot x \quad d \cdot y \quad d \cdot z \quad 0 \quad 0 \quad 0]$$

Donde, debido a que  $dx=dy=0$ , y los ángulos entre los ejes son fijos,

$$d \cdot x = \frac{{}^E \partial x}{\partial d_n} = J_{1n} = {}^{n-1}x_z = {}^0x_z$$

$$d \cdot y = \frac{{}^E \partial y}{\partial d_n} = J_{2n} = {}^{n-1}y_z = {}^0y_z$$

$$d \cdot z = \frac{{}^E \partial z}{\partial d_n} = J_{3n} = {}^{n-1}z_z = {}^0z_z$$

$$\frac{{}^E \partial \delta x}{\partial d_n} = \frac{{}^E \partial \delta y}{\partial d_n} = \frac{{}^E \partial \delta z}{\partial d_n} = 0 = J_{4n} = J_{5n} = J_{6n}$$

3.- A continuación, para calcular el vector asociado al movimiento diferencial para la 2ª articulación, ( $n=2$ ) obtenemos la transf. diferencial de coord. para la articulación n, premultiplicando la transf. diferencial de coord. para la articulación n-1 (la articulación anterior) por la inversa de  $A_{n-1}$ .

$${}^{n-1}T_E = A_{n-1}^{-1} \cdot {}^{n-2}T_E = A_n \mathbf{L} \quad A_E \Rightarrow {}^1T_E = A_1^{-1} \cdot {}^0T_E = A_2 \mathbf{L} \quad A_E$$

Ahora calcularemos la segunda columna del Jacobiano sustituyendo los elementos de la transf. diferencial de coord. en las ecs. vistas en el pto. 2, según se trate de una articulación prismática o de revolución.

4.- Repetir el paso 3 hasta calcular todas las columnas del Jacobiano, una para cada articulación.

Ejemplo 1.- Apliquemos el algoritmo al manipulador planar de dos articulaciones (de revolución):

A partir de la Transf. cinemática directa,

$$\begin{aligned}
 {}^0T_E &= \begin{bmatrix} C_{12} & -S_{12} & l_1 \cdot C_1 + l_2 \cdot C_{12} \\ S_{12} & C_{12} & l_1 \cdot S_1 + l_2 \cdot S_{12} \\ 0 & 0 & 1 \end{bmatrix}; \quad {}^1T_E = \begin{bmatrix} C_2 & -S_2 & l_2 \cdot C_2 \\ S_2 & C_2 & l_2 \cdot S_2 \\ 0 & 0 & 1 \end{bmatrix} \\
 {}^E J &= \begin{bmatrix} {}^0x_y \cdot {}^0p_x - {}^0x_x \cdot {}^0p_y & {}^1x_y \cdot {}^1p_x - {}^1x_x \cdot {}^1p_y \\ {}^0y_y \cdot {}^0p_x - {}^0y_x \cdot {}^0p_y & {}^1y_y \cdot {}^1p_x - {}^1y_x \cdot {}^1p_y \end{bmatrix} = \begin{bmatrix} l_1 \cdot S_2 & 0 \\ l_1 \cdot C_2 + l_2 & l_2 \end{bmatrix}
 \end{aligned}$$



Este resultado es el mismo que el Jacobiano con referencia al elemento terminal, obtenido aplicando cálculo diferencial.

*Ejemplo2.-* Apliquemos el algoritmo, ahora, al manipulador planar de dos articulaciones (prismáticas):

$${}^0T_E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & -1 & -d_2 \\ 0 & 1 & 0 & d_1 \\ 0 & 0 & 0 & 1 \end{bmatrix}; \quad {}^1T_E = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & d_2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \Rightarrow {}^EJ = \begin{bmatrix} {}^0x_z & {}^1x_z \\ {}^0y_z & {}^1y_z \\ {}^0z_z & {}^1z_z \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

### Singularidades

La mayoría de manipuladores poseen configuraciones donde el Jacobiano es singular, esto es, no tiene inversa. Cuando un manipulador está en una configuración singular, pierde uno o más grados de movilidad, lo cual, para un manip. ≤6 grados de movilidad corresponde a una pérdida de grados de libertad, lo que supone que existirán ciertas direcciones cartesianas inalcanzables por el elemento terminal.

- Tipos:
- (1) Singularidad interna al espacio de trabajo.
  - (2) Singularidad en la frontera del espacio de trabajo.

*Ejemplo.-* Manipulador de dos articulaciones 2R planar, visto en la sección 2.2.1.3.

Como el  $\det(J) = l_1 l_2 \sin\theta_2 = 0$  implica que  $\theta_2 = 0$  ó  $\pi$ , es decir, exhibe dos singularidades en la frontera de su espacio de trabajo: Cuando el brazo está completamente retraído (0) y cuando se encuentra completamente extendido ( $\pi$ ).

## 2.3 CONTROL DEL MOVIMIENTO: CONCEPTOS BÁSICOS

En el movimiento de un manipulador interesa conocer la posible existencia de obstáculos en su camino (ligaduras de obstáculo) y si el elemento terminal debe recorrer un camino especificado (ligaduras del camino). La combinación de éstas dos ligaduras combinadas dan lugar a cuatro modelos de control posibles:

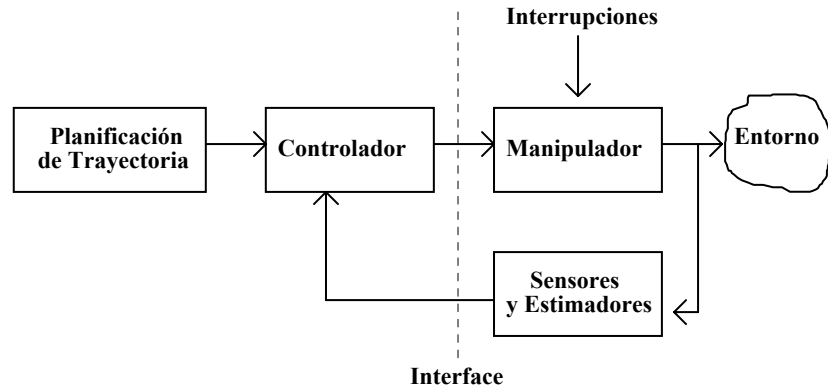
		Ligaduras de Obstáculo	
		Si	No
Ligaduras del Camino	Si	Fuera de Línea Libre de Colisión Planificación del Camino en Línea [ + ] Seguimiento del Camino	Fuera de Línea Planificación del Camino en Línea [ + ] Seguimiento del Camino
	No	Control Posicional [ + ] Obstáculo en Línea Detección y Evitación	Control Posicional

Así pues, el problema de Control de un Manipulador se puede dividir en dos subproblemas (1) *Planificación del Movimiento* (o trayectoria) y, (2) *Control del Movimiento*. Nos centraremos exclusivamente en (2).

**Observación:** Desde el pto. de vista del análisis de Control, el movimiento de un brazo se suele efectuar en dos fases de control distintas:

- I).- *Control del Movimiento de Aproximación.* El brazo se mueve desde una localización inicial, hasta la localización final deseada, a lo largo de una trayectoria planificada.
- II).- *Control del Movimiento Fino.* El elemento terminal interactúa dinámicamente con el objeto utilizando información de la realimentación sensorial para completar la tarea.

Diagrama de Bloques del Control básico del Robot:



Considerando el control del robot como un problema de seguimiento de trayectoria, el control del movimiento se puede clasificar en 3 grandes categorías:

- 1.- Controles de Movimiento de la Articulación
- 2.- Controles con Movimiento resuelto (control en el espacio cartesiano)
- 3.- Controles Adaptativos

- *Control del Movimiento en el Espacio de Articulaciones*

Algoritmo básico para generar puntos de consigna de la trayectoria de la articulación:

$$t = t_0;$$

bucle : Esperar hasta el próximo intervalo de control;

$$t = t + \Delta t;$$

$h(t)$  = donde debería estar en el instante  $t$  la posición de la articulación del manipulador;

Sí  $t = t_f$ , entonces salir;

ir a bucle;

donde  $\Delta t$  es el período de muestreo de control para el manipulador. Por lo tanto, el cálculo consiste en una función de trayectoria (o planificación de trayectoria),  $h(t)$  que se modifica en cada intervalo de control.

- *Control del Camino Cartesiano*

Algoritmo básico para generar puntos de consigna de la trayectoria del camino cartesiano:

$$t = t_0;$$

bucle : Esperar hasta el próximo intervalo de control;

$t = t + \Delta t$ ;  
 $H(t)$  = donde debería estar en el instante  $t$  la mano del manipulador;  
 $Q[H(t)]$  = solución correspondiente de la articulación para  $H(t)$ ;  
 Sí  $t = t_f$ , entonces salir;  
 ir a bucle;

Por lo que además del cálculo de la función de trayectoria de la mano del manipulador  $H(t)$  en cada intervalo de control, necesitamos convertir las posiciones cartesianas en sus correspondientes soluciones de articulación,  $Q[H(t)]$ . La función  $H(t)$  indica la posición deseada de la mano del manipulador en el instante  $t$ .

## RESUMEN T-II. Aspectos Clave:

- Redundancia y Degeneración

Def.- La *redundancia* aparece cuando existe más de una solución a la Transformación Cinemática Inversa. El n° de soluciones posibles es  $2^n$ , siendo  $n$  el n° de redundancias.

Observación: Sí el n° de soluciones es mayor que el n° de variables de articulación implica que el manipulador puede no tener solución utilizando el Algoritmo dado anteriormente, por lo que deberán imponerse “restricciones”.

Observación: La existencia de ciertas funciones matemáticas en la solución a la Transformación Cinemática Inversa, indica la existencia de redundancias en un manipulador. (e.g., la *raíz cuadrada* y el *coseno*, pues no discriminan el "signo")

Def.- La *degeneración* aparece cuando existen un n° infinito de configuraciones para alcanzar una posición. Una configuración degenerada es cualquier configuración del manipulador donde el control sobre uno o más grados de libertad se pierde. (e.g., Muñeca donde existen articulaciones con al menos dos ejes colineales)

- Precisión del Modelo Cinemático

Depende de la precisión, repetibilidad, y resolución

Def.- La *Precisión* es la diferencia entre posición real actual y la requerida.

La *Repetibilidad* es la variación en la localización real actual cuando el manipulador mueve repetidamente el TCP a la misma localización deseada.

La *Resolución* es la mínima distancia garantizada para el movimiento del TCP.

Observación: Las causas de imprecisión se agrupan en dos tipos de parámetros: 1) *No-Geométricos* (e.g., resolución de encoders, errores de transmisión en engranajes, etc.). 2) *Del Modelo Geométrico* (parámetros de D-H del modelo cinemático).

- Eficiencia de las Soluciones Cinemáticas

La *eficiencia* del modelo cinemático de un manipulador se mide usualmente en función del n° de operaciones de adición, producto y llamadas a funciones transcendentales requeridas para resolver el modelo.

- Un camino es una secuencia de puntos en el espacio.
- Una trayectoria es un camino con restricciones temporales.
- El Jacobiano del manipulador es una matriz de elementos diferenciales, utilizada para transformar velocidades desde el espacio de articulaciones al cartesiano.
- Cuando el determinante del Jacobiano es cero, el Jacobiano no posee inversa, y el manipulador se encuentra en una configuración singular.
- En una singularidad, el control del movimiento en alguna dirección cartesiana es físicamente imposible. Las singularidades aparecen cuando existe alineación de ejes de articulaciones, y en la frontera del espacio de trabajo del manipulador.

## TEMA III SISTEMAS DE PERCEPCIÓN ROBÓTICA

### Objetivos:

- *Introducir al alumno en el campo de la sensorización de robots, comprendiendo los principios físicos subyacentes.*
- *Estudiar los dos tipos de información sensorial existentes: la interna y la externa.*
- *Describir algoritmos capaces de procesar, con un determinado fin, los datos de entrada de un sistema de visión.*
- *Enlazar brevemente con la problemática asociada al reconocimiento de formas.*
- *Describir su problemática en el campo de la Visión Artificial, mediante casos prácticos.*
- *Poner de manifiesto las complejidades que entraña la construcción de un Clasificador Euclídeo.*

### 3.1 GENERALIDADES DEL PROCESO DE PERCEPCIÓN

Investigadores como Dario [Dario, 89] plantearon en su día un dilema que sigue siendo actual en la automatización industrial de nuestros días: ¿cuándo se debería incorporar información sensorial a los sistemas robotizados reduciendo, en consecuencia, el grado de estructuración del entorno de trabajo?, y ¿cuándo se debería prescindir de dicha información, incrementando la estructuración del campo de operaciones?.

Dario responde sugiriendo que podemos distinguir dos extremos en el campo de aplicación o dominio de los robots industriales:

- A.** Tareas cuyas prioridades son: alta velocidad y precisión absoluta en la posición. Características típicas de las cadenas de producción o ensamblado a gran escala.
- B.** Tareas más sofisticadas, necesitadas de información sensorial. Típica de la producción o ensamblado de pequeños lotes.

En **A** interesa estructurar al máximo el entorno, evitando el uso de manos sofisticadas y sensores. Por contra, **B** justifica el coste adicional que supone el uso de sensores y la pérdida de estructuración. En conclusión, el uso de sensores estará justificado en aquellas aplicaciones en las que la calidad prime sobre la velocidad como único parámetro a contemplar.

Al margen de justificaciones como la anterior para el empleo o no de sensores, digamos que éstos tienen interés *per se*, en el sentido de que son ingredientes intrínsecos del proceso de percepción, siendo ésta la vía obligada para alcanzar la conexión inteligente entre percepción y acción, que es como se ha definido la denominada "robótica inteligente" [Brady, 84].

En general, la percepción artificial, según Mackerrow [Mackerrow, 91], intenta capacitar al robot para realizar tareas complejas, adaptarse a posibles variaciones de su entorno (ambiente no estructurado) y afrontar situaciones imprevistas. Se trata de un problema difícil, cuya

solución requiere la aportación de diferentes campos del conocimiento y, en su forma general, exigirá la utilización de herramientas tecnológicas aún no disponibles.

Para comprender mejor donde residen las dificultades podemos buscar un paralelismo con la percepción visual humana. Para una revisión bibliográfica sobre el tema ver el trabajo de Zucker [Zucker, 81]. El mecanismo de percepción visual humano funciona como una cadena en la cual, a partir de un cierto estímulo (la luz), capturado por un receptor (la retina), se produce una determinada respuesta. Dicha respuesta, generalmente, requiere una interpretación previa de la escena iluminada que provocó el estímulo, y ésta a su vez puede depender del contexto, de la experiencia cognoscitiva anterior y de los objetivos que se persigan. Por tanto, la interpretación de una sensación no puede basarse únicamente en la información contenida en la misma. Requiere la cooperación de otras fuentes de información (experiencia, contexto, objetivos, etc.).

Llegado este punto interesa recordar algunas reflexiones de Russell [Russell, 48] al hilo de la percepción y el ordenador. Este plantea preguntas como: ¿podría un ordenador, con suficiente conocimiento de la estructura de determinado cerebro, predecir la respuesta muscular a un estímulo dado utilizando las leyes de la física y la química?; ¿O la intervención de la mente es un eslabón esencial para conectar un antecedente físico (el estímulo) con una consecuencia física (un movimiento corporal)? La experiencia demuestra que existen reflejos en los que la respuesta es automática y no controlada por la volición.

Subyacente en las observaciones de Russell está la idea del movimiento de acomodación *versus* el balístico. En el ser humano, el primero hace uso de cierto mecanismo de realimentación sensorial, mientras que el segundo se relaciona con los movimientos reflejos. El origen de la traslación de estos conceptos, en especial el de realimentación, a los sistemas de control de los robots, se debe a Wiener [Wiener, 48]. La realimentación sensorial en los sistemas robotizados, permite cerrar el bucle de control, dando lugar a los denominados sistemas servocontrolados, si la información sensorial utilizada es propioceptiva. Si además, se utiliza realimentación de información exteroceptiva (e.g. visual o táctil), se conseguirá una adaptabilidad respecto a la posición y una potencial capacidad de reacción, logrando que el sistema robotizado sea mucho más robusto<sup>9</sup> e inteligente<sup>10</sup>.

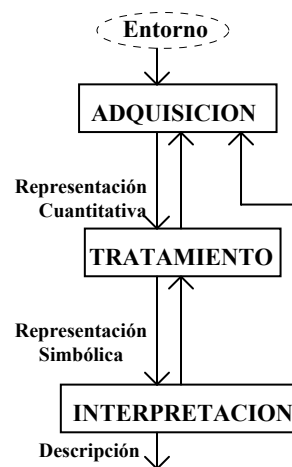
De modo esquemático (véase la figura 3.1) el proceso de percepción se puede dividir en tres fases [Mackerrow, 91]:

- Fase de Adquisición.
  - Depende del tipo de percepción de que se trate.
  - Las disciplinas relevantes aquí serán la electrónica y ciertos campos específicos de la Física.
  - Su aportación a la siguiente fase será, en general, una representación matricial cuantitativa y digitalizada de la información ambiental adquirida.
- Fase de Tratamiento.
  - Abarca aspectos muy diversos (corrección de distorsiones, eliminación de ruido, segmentación y extracción de características etc. ) de importancia variable en función de: (1º) Tipo, cantidad y calidad de la información suministrada por la 1ª fase, y (2º) Requisitos de la 3ª fase.

<sup>9</sup> Es decir, menos vulnerable a posibles errores de posicionamiento, derivados del calibrado, modelo cinemático, etc.

<sup>10</sup> Suponiendo que la capacidad de adaptación sea un indicador de un cierto grado de inteligencia.

- Las técnicas de base utilizadas son, hasta cierto punto, independientes de la naturaleza de la información procesada. En general corresponden al conjunto de métodos reunidos bajo el nombre de "*procesado digital de señales*".
  - La salida de ésta fase será una representación simbólica de la información.
- Fase de Interpretación.
    - Es la de mayor dificultad del proceso.
    - Representa, fundamentalmente, la aportación de la I.A. y de las técnicas del Reconocimiento de Formas en su doble vertiente de decisión teórica y sintáctica.
    - El resultado será una descripción precisa y completa del entorno de interés.



**Figura 3.1.** Diagrama de bloques del proceso de percepción

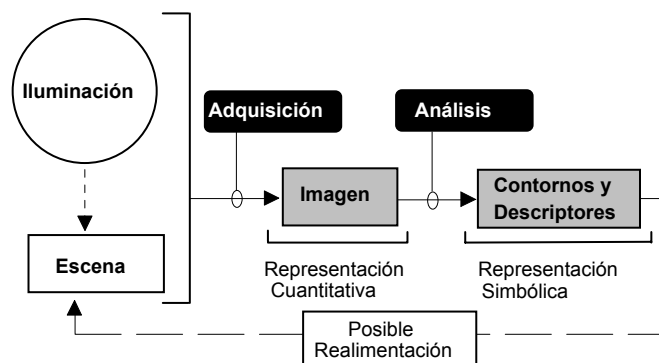
## 3.2 VISION ARTIFICIAL: ADQUISICIÓN Y ANÁLISIS

### 3.2.1. Preliminares

Como ya se comentó en el punto 3.1, al hilo de la percepción visual, hay que resaltar la importancia de la experiencia cognoscitiva previa en la interpretación de una imagen [Ferraté et al., 86]. Una persona no ve niveles de gris caprichosamente repartidos en una superficie bidimensional, ni tampoco ve contornos, ni saltos bruscos de intensidad, ni ninguna otra característica definida numéricamente sobre una matriz. Lo que una persona realmente ve son objetos tridimensionales con los que ha sido familiarizado en el transcurso de su vida. He aquí la diferencia entre sensación y percepción. La sensación es la información captada por nuestros receptores; la percepción incluye además la interpretación dada a esa información gracias a la experiencia previa obtenida en situaciones semejantes. Además, el contexto ayuda a eliminar posibles ambigüedades. Otro factor a tener en cuenta es el principio gestáltico de que "el todo es más que la suma de las partes". Un ejemplo evidente de ello se observa a partir de la existencia de propiedades como la simetría, que sólo pueden ser captadas a nivel global y desaparecen en cuanto se analiza localmente la imagen. Por último, destacar que los objetivos que se persigan tendrán una influencia decisiva en el diseño del mecanismo de percepción implementado. Por ejemplo, si se requiere razonar a partir de la forma de los

contornos de los objetos, prescindiremos de toda aquella información que no sea relevante al respecto (e.g. texturas, color, etc.).

De acuerdo con otros investigadores [Horn, 86], podemos afirmar que estamos todavía muy lejos de poder conseguir un sistema de visión "universal", que funcione bien con independencia del contexto y de los objetivos a alcanzar. Por el contrario, la mayor parte de los sistemas de visión tienen un alcance bastante limitado, resolviendo únicamente aquella aplicación concreta para la que fueron diseñados bajo unas determinadas condiciones iniciales. No obstante, se están empezando a crear las bases necesarias para construir módulos que solucionen áreas acotadas del proceso de visión con la idea de lograr una aproximación cada vez mayor a un sistema de visión "universal" (o de propósito general). Estos módulos exigirán adaptaciones mínimas a problemas específicos cualesquiera. Un módulo de visión, típico en robótica, habrá de capturar la escena y entregar a la salida una representación simbólica formada por cada uno de los contornos de los objetos que en ella aparezcan así como una serie de descriptores asociados (véase la figura 3.2). En dicha figura se representan dos módulos básicos: *adquisición* y *análisis*. En *adquisición* englobaremos la problemática de la iluminación, la captura por el sistema de visión utilizado (cámara CCD más tarjeta digitalizadora). Mientras que en *análisis* abarcaremos las etapas de segmentación, extracción de contornos, cálculo de momentos y extracción de características basadas en los momentos. Dejando el preprocesado (filtros, etc.), para un nivel intermedio entre ambas etapas.



**Figura 3.2.** Adaptada de [Horn, 86]. Diagrama de bloques de un módulo de visión genérico.

Puntualicemos:

- La visión, se puede definir como la habilidad de reconocer objetos a partir de la luz reflejada por los mismos en una imagen, mediante el consiguiente procesado de ésta.
- El ojo humano focaliza la luz en la retina, donde los elementos receptores (conos y bastones) recogen la intensidad y longitud de onda de la luz. A continuación ésta información se pasa al cerebro, el cual lo interpreta como una escena procediendo a identificar los objetos de la misma.
- Una imagen digitalizada es el resultado de realizar un muestreo equidistanciado y una cuantificación (sampling & quantization) a una imagen real o continua.

### 3.2.2. Adquisición de la imagen

Este problema es de importancia crucial para poder garantizar que el procesamiento de la imagen, a realizar posteriormente, será satisfactorio. Podemos considerar tres aspectos fundamentales [Davies, 90]:

- (1) El esquema de iluminación empleado.
- (2) La tecnología utilizada en la captura y digitalización de las imágenes.
- (3) La teoría básica subyacente al proceso de muestreo (e.g. el teorema de muestreo [Aström & Wittenmark, 88]).

El punto (2) está relacionado, básicamente, con los dispositivos físicos necesarios para capturar la imagen utilizando una cámara; digitalizarla a continuación mediante la tarjeta diseñada al efecto; y ubicarla, finalmente, en la memoria del ordenador, como una matriz de píxeles. Las cámaras se encargan de enfocar y depositar la imagen de la escena tomada sobre un sensor electrónico, que transforma las variaciones luminosas en eléctricas para transmitirlos, posteriormente al ordenador.

Elementos fundamentales:

- 1) Objetivo.- Sistema óptico, formado por un conjunto de lentes, que proyecta la imagen sobre el sensor de forma adecuada.
- 2) Sensor.- Transforma las señales luminosas en eléctricas.

Según el sensor existen, básicamente dos tipos de cámaras:

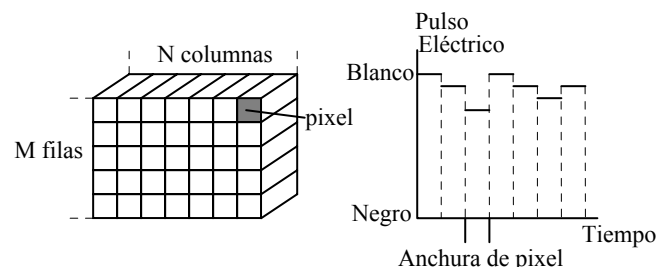
- 1.- De tubo (e.g. Vidicón)
- 2.- De estado sólido (e.g. CCD)

Elementos básicos de una cámara de Estado Sólido:

- 1.- Fotodiodos
- 2.- Dispositivos de Acoplamiento de Carga CCD
- 3.- Dispositivos de Inyección de Carga CID

2.- Dispositivos de Acoplamiento de Carga CCD (véase la fig. 3.3)

Producen una carga eléctrica proporcional a la intensidad luminosa que incide sobre ellos. Cada célula física proporciona una información puntual de la luz que recibe (el elemento básico de la imagen se denomina *pixel* ó *picture element*). Cada pixel proporcionará una señal eléctrica comprendida entre un máximo (pto. blanco en la imagen), y un mínimo (pto. negro ó no iluminado). Entre estos dos extremos existirán infinitos valores que representan los infinitos niveles de gris.



**Figura 3.3.** Las células de una fila de la matriz del sensor proporcionan una información eléctrica correspondiente a una línea de la imagen:

Una línea de la imagen se compondrá de un nº de señales eléctricas equivalentes al nº de columnas de la matriz. Relativo a la exploración de la imagen, en una cámara de Estado



Sólido, conformada por una matriz de elementos fotosensibles, la intensidad de cada uno de ellos se obtiene mediante un circuito electrónico que los va explorando, de izquierda a derecha y de arriba abajo. La señal obtenida es la SEÑAL DE VIDEO!.

### 3.2.3. Análisis de la imagen

Def.-*Imagen bidimensional*,  $f(x,y)$ , que representa el resultado de un muestreo más una cuantización de intensidad o nivel de gris.

El resultado de digitalizar la imagen continua  $f(x,y)$ , espacialmente y en amplitud, será lo que llamaremos una *Imagen Digital*:

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(1,0) & \cdots & f(N-1,0) \\ f(0,1) & f(1,1) & \cdots & f(N-1,1) \\ \vdots & \vdots & \vdots & \vdots \\ f(0,N-1) & f(1,N-1) & \cdots & f(N-1,N-1) \end{bmatrix}$$

Normalmente se definen:  $N=2n$ , y  $G=2m$  ( $n^\circ$  de niveles de gris). Luego, el  $n^\circ$ ,  $b$ , de bits requeridos para contener una imagen digitalizada será:  $b=N \times N \times m$

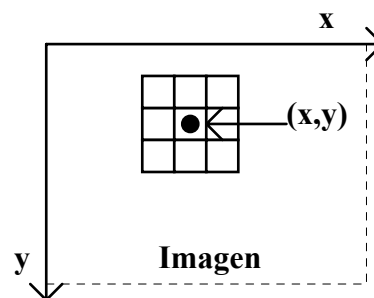
- Relaciones Básicas entre Pixels: Conectividad

Dado un pixel,  $p$ , de coord.  $(x,y)$  se tiene:

Def.-  $N_4(p)$  ó 4-vecinos de  $p \equiv \{(x+1,y),(x-1,y),(x,y+1),(x,y-1)\}$

Def.-  $N_8(p)$  u 8-vecinos de  $p \equiv \{N_4(p)+[(x+1,y+1),(x+1,y-1),(x-1,y+1),(x-1,y-1)]\}$

*Observación:* Se utiliza el convenio ,



Def.- Conectividad-4  $\equiv$  Dos píxeles  $p$  y  $q$  están 4-conectados si  $q$  está en el conjunto  $N_4(p)$ .

Def.- Conectividad-8  $\equiv$  Dos píxeles  $p$  y  $q$  están 8-conectados si  $q$  está en el conjunto  $N_8(p)$ .

Def.- Conectividad- $m$  (conectividad mixta)  $\equiv$  Dos píxeles  $p$  y  $q$  están  $m$ -conectados si:  
1)  $q$  está en el conjunto  $N_4(p)$ , ó

2)  $q$  está en el conjunto  $N_D(p)$  y  $N_4(p) \equiv N_4(q)$  es el conjunto vacío. (Este es el conjunto de los pixels que son 4-vecinos de  $p$  y  $q$ )

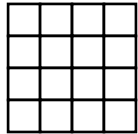
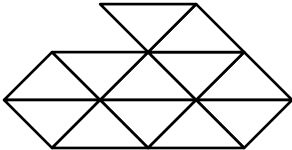
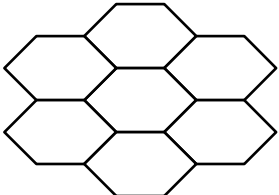
*Def.- Camino* desde el pixel  $p$  de coord.  $(x_0, y_0)$ , hasta el pixel  $q$ , de coord  $(x_n, y_n) \equiv (x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)$ . Donde los  $(x_i, y_i)$  son adyacentes a  $(x_{i-1}, y_{i-1})$ , con  $1 \leq i \leq n$ , y  $n$  es la *longitud* del camino. Se pueden definir 4-caminos, 8-caminos etc., dependiendo del tipo de adyacencia usada.

La def. anterior constituye también la def. de conectividad: Diremos que  $p$  y  $q$  están conectados si  $\exists$  un camino (cadena de puntos) de  $p$  a  $q$ , en el sentido anterior.

- Teselaciones y Métricas

*Def.- Teselación*  $\equiv$  Patrón en el que se divide el plano

Ejemplos de diferentes Teselaciones del plano:

Rectangular	Triangular	Hexagonal
		

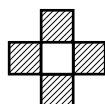
Observación: Aunque las teselaciones rectangulares son universalmente aceptadas en visión artificial, poseen un problema estructural conocido como *PARADOJA DE LA CONECTIVIDAD*:

Dado un pixel en una teselación rectángular, ¿Cómo deberíamos definir los pixels a los que está conectado?

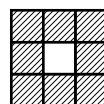
Hemos visto que dos métodos comunes serán la 4-conectividad y la 8-conectividad.

Visualización de la denominada “paradoja de la conectividad” para teselaciones rectangulares:

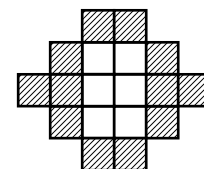
a) Un pixel central y sus 4-vecinos



b) Un pixel central y sus 8-vecinos



c) Conectividad ambigua



La fig. c) muestra un objeto obscuro con un agujero, sobre un fondo blanco.

Si usamos 4-conectividad, la fig. consistirá en 4 piezas desconectadas  $\Rightarrow$  El agujero queda separado del objeto.

Si usamos 8-conectividad, la fig. quedará conectada en una pieza  $\Rightarrow$  El agujero ahora formará parte del objeto (conectado al borde).

Esta paradoja conlleva complicaciones para muchos algoritmos geométricos.

**Nota:** Las teselaciones triangular y hexagonal no presentan dificultades en cuanto a conectividad (e.g., la 3-conectividad para triángulos). Sin embargo la *DISTANCIA* puede ser más difícil de calcular en éstas que en las matrices de rectángulos!!.

**Def.- Distancia.** En general una distancia  $d$  es una métrica:

- (1)  $d(\bar{x}, \bar{y}) = 0 \Leftrightarrow \bar{x} = \bar{y}$
- (2)  $d(\bar{x}, \bar{y}) = d(\bar{y}, \bar{x})$
- (3)  $d(\bar{x}, \bar{y}) + d(\bar{y}, \bar{z}) \geq d(\bar{x}, \bar{z})$

Para matrices cuadradas con espaciado unidad entre pixels, podemos usar cualquiera de las métricas siguientes para 2 pixels,  $(x_1, y_1)$ ,  $(x_2, y_2)$ :

$$\text{Euclídea: } d_e(\bar{x}, \bar{y}) = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

$$4\text{-Distancia: } d_4(\bar{x}, \bar{y}) = |x_1 - x_2| + |y_1 - y_2|$$

$$8\text{-Distancia: } d_8(\bar{x}, \bar{y}) = \max\{|x_1 - x_2|, |y_1 - y_2|\}$$

4-Distancia: Los pixels cuya distancia  $D_4$  a  $(x, y)$  sea menor o igual que un valor dado  $r$  forman un rombo centrado en  $(x, y)$ . Ejemplo,  $D_4 \leq 2$  a  $(x, y)$ , da lugar a la siguiente configuración de distancias constante:

$$\begin{array}{ccccc} & & 2 & & \\ & & 2 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ & & 2 & 1 & 2 \\ & & & & 2 \end{array}$$

8-Distancia: Los pixels cuya distancia  $D_8$  a  $(x, y)$  sea menor o igual que un valor dado  $r$  forman un cuadrado centrado en  $(x, y)$ . Ejemplo,  $D_8 \leq 2$  a  $(x, y)$ , da lugar a la siguiente config. de dist. cte:

$$\begin{array}{ccccc} 2 & 2 & 2 & 2 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 1 & 0 & 1 & 2 \\ 2 & 1 & 1 & 1 & 2 \\ 2 & 2 & 2 & 2 & 2 \end{array}$$

**Observación:** Las distancias  $D_4$  y  $D_8$  pueden considerarse en términos de la existencia o no de un camino de conexión entre los dos pixels implicados, ya que la definición de éstas distancias involucra sólo a las coordenadas de éstos puntos.

- Preprocesados Básicos de la Imagen: Filtros  
Básicamente existen dos métodos de tratar/filtrar una imagen:

- (1) Métodos en el Dominio de la Frecuencia
- (2) Métodos en el Dominio del Espacio

## (1) Métodos en el Dominio de la Frecuencia

Estas técnicas conllevan un alto coste computacional, por lo que su uso es todavía infrecuente en el campo de la robótica, por su necesidad de trabajar en tiempo real.

## (2) Métodos en el Dominio del Espacio

*Def.-* Las funciones de Preprocesamiento en el dominio espacial se pueden expresar como:

$$g(x,y) = h[f(x,y)]$$

donde,  $f(x,y)$ ≡imagen de entrada;  $g(x,y)$ ≡imagen obtenida (preprocesada);  $h$ ≡operador en  $f$ , def. sobre el entorno de vecindad de  $(x,y)$

**Nota:** Se puede hacer también que  $h$  opere sobre un conjunto de imágenes (e.g., la suma pixel a pixel de  $k$ -imágenes para la reducción de ruido).

Las técnicas en el dominio espacial usadas más frecuentemente:

Máscaras de Convolución  
(plantillas, ventanas ó filtros)

*Def.-* Una máscara es una matriz bidimensional cuyos coeficientes se eligen de forma que podamos detectar una propiedad dada para una imagen.



Se utiliza para las llamadas "operaciones" en entorno de vecindad:

- Reducción de Ruido
- Para obtener niveles de imagen variables
- Para hacer apreciaciones de textura
- Para obtener el esqueleto de un objeto

*Fig.-* Una Máscara 3 x 3 General:

<b>w1</b> <b>(x-1,y-1)</b>	<b>w2</b> <b>(x-1,y)</b>	<b>w3</b> <b>(x-1,y+1)</b>
<b>w4</b> <b>(x,y-1)</b>	<b>w5</b> <b>(x,y)</b>	<b>w6</b> <b>(x,y+1)</b>
<b>w7</b> <b>(x+1,y-1)</b>	<b>w8</b> <b>(x+1,y)</b>	<b>w9</b> <b>(x+1,y+1)</b>

A partir de aquí, se obtiene:

$$h[f(x,y)] = w1 f(x-1,y-1)+w2 f(x-1,y)+w3 f(x-1,y+1)+w4 f(x,y-1)+w5 f(x,y)+w6 f(x,y+1)+w7 f(x+1,y-1)+w8 f(x+1,y)+w9 f(x+1,y+1)$$

Ejemplo.- Máscara para detectar puntos aislados:

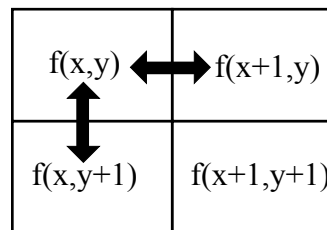
-1	-1	-1
-1	8	-1
-1	-1	-1

**Observación:** En General, el Preproceso debe de cumplir 2 funciones:

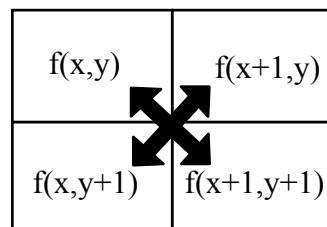
- a) *Codificación y Aproximación*; para reducir la cantidad de información.  
 b) *Filtrado, Restauración y/o Realce*; para eliminar ruidos, compensar la degradación debida a la compresión de la información y/o mejorar la calidad de la representación.

- Filtros PASA-ALTO

Consisten en hacer resaltar el perímetro de los distintos objetos que componen la imagen. Se basan en el cálculo del *gradiente*. En esencia, se trata de comparar los pixels de una imagen con sus vecinos:



Gradiente de Roberts.- Se trata de una mejora del anterior:

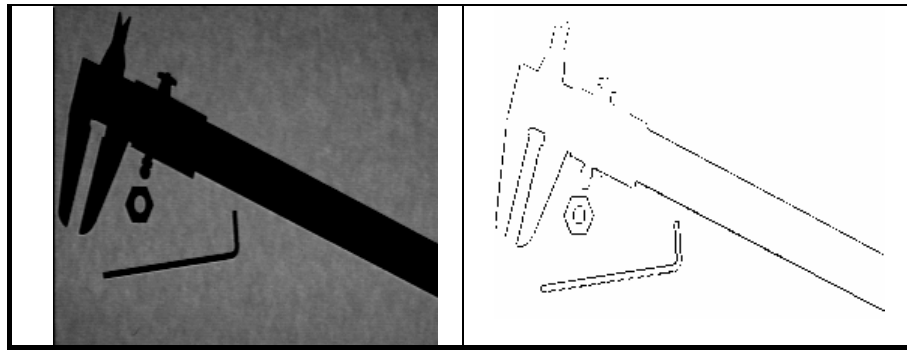


Observación:

*En ambos casos, el cálculo del gradiente es proporcional a la diferencia entre los niveles de gris de pixels adyacentes.### Un valor alto de gradiente en pixels asociados a las aristas de los objetos, y valores bajos en zonas con niveles de gris próximos.*

*Ejemplo.-* Resultados de aplicar el gradiente de Roberts:

Imagen Original	Imagen Resultante
-----------------	-------------------



- Filtros PASA-BAJO

Objetivo: Eliminación de Ruido

Las dos Técnicas más utilizadas son:

- 1).- Filtro de MEDIA

Un pixel en la nueva imagen es el resultado de promediar los valores de los pixels vecinos al pixel homólogo de la imagen original:

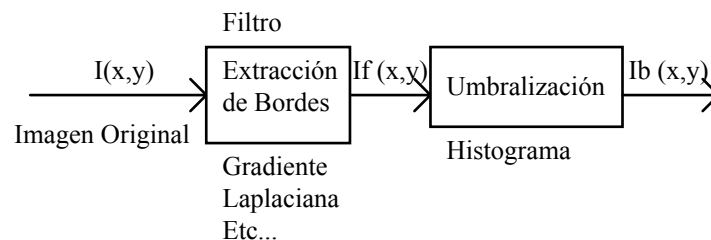
$$g(x,y) = [f(x-1,y)+f(x+1,y)+f(x,y-1)+f(x,y+1)]/4$$

- 2).- Filtro de MEDIANA (*Pasa Banda*)

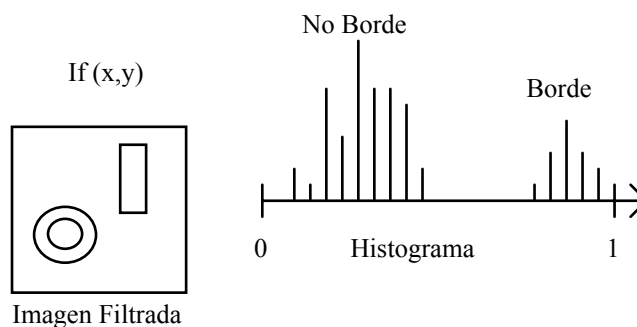
Aquí se reemplaza el valor de gris de cada pixel por el valor de *mediana* (No media!) de los valores de gris de los pixeles vecinos.

- **Binarización mediante Umbral**

En síntesis, un esquema del proceso de segmentación de una imagen,  $I(x,y)$ , basado en umbralizar el histograma de la imagen original filtrada,  $I_f(x,y)$ , puede seguirse en el esquema siguiente:



*Ejemplo.* Se representa, a continuación, una imagen filtrada,  $I_f(x,y)$ , en la que aparecen claramente resaltados los bordes de los objetos. El histograma asociado presenta dos lóbulos, uno correspondiente a los píxeles de borde (menor) y el otro relativo al resto de píxeles de la imagen.:



La umbralización del histograma de la imagen filtrada producirá una imagen binaria  $I_b(x,y)$  tal que:

$$I_b(x,y) = \begin{cases} 1 & \text{sii } (x,y) \in \text{borde} \\ 0 & \text{sii } (x,y) \in \text{resto} \end{cases}$$

- **Procesado de Imágenes a Nivel Intermedio**

- 1.- **Introducción.**

- 2.- **Segmentación mediante Seguimiento de Contornos.**

- 3.- **Descriptores: Invariantes Geométricos.**

### Introducción

Una vez segmentada la imagen digital de partida (etiquetados sus píxeles), el siguiente paso es calcular un vector de características discriminantes asociadas con el contorno.

Existen dos procedimientos clásicos a la hora de calcular rasgos discriminantes de contornos cerrados:

- || [1].- Basado en el cálculo de momentos de una función bidimensional  $f(x,y)$  acotada y cerrada en el plano  $x,y$ .
- || [2].- Basado en la Transformada de Fourier.

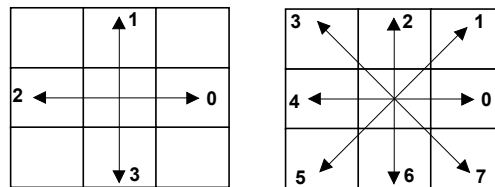
Como paso previo a la aplicación de alguno de estos dos procedimientos es preciso codificar el contorno, cerrado, de cada objeto.

### Segmentación mediante Seguimiento de Contornos.

El algoritmo utilizado es una variante del algoritmo de codificación de contornos mediante códigos de cadena descrito en [Gonzalez & Wintz, 87], el cual reduce la imagen pseudobinaria, obtenida en el paso anterior (segmentación), a una *lista* de contornos, unívocamente determinados mediante la especificación de:

- 1 Su nivel de gris.
- 2 La posición (fila y columna) de un píxel en su contorno, llamado punto inicial (PI).
- 3 Una secuencia de direcciones que permiten trazar el borde externo del contorno.

Un código de cadena puede representar un contorno mediante una secuencia de segmentos rectos de longitud y dirección específica, como por ejemplo el código 4 u 8-direccional de la figura 4.6. Supondremos en lo que sigue una teselación rectangular del espacio, por otra parte la más utilizada con diferencia, en el procesamiento digital de imágenes. Es importante puntualizar que los códigos de direcciones representados en la figura 3.4. definen una relación de vecindad, es decir, el código 4-direccional da lugar a la 4-conectividad, y el 8-direccional a la 8-conectividad. Para evitar ambigüedades hay que fijar a priori uno de los dos. En nuestro caso, elegimos 8-conectividad por ser la que mejor se adapta a las representaciones de los contornos digitalizados y ser más compacta que la 4-conectividad. Tengamos en cuenta que la consideración de un mayor número de direcciones no mejora la precisión de la cuantización de una curva [Saghri & Freeman, 81].



**Figura.3.4.** Representación gráfica del código 4-direccional (izquierda) y 8-direccional (derecha), de Freeman.

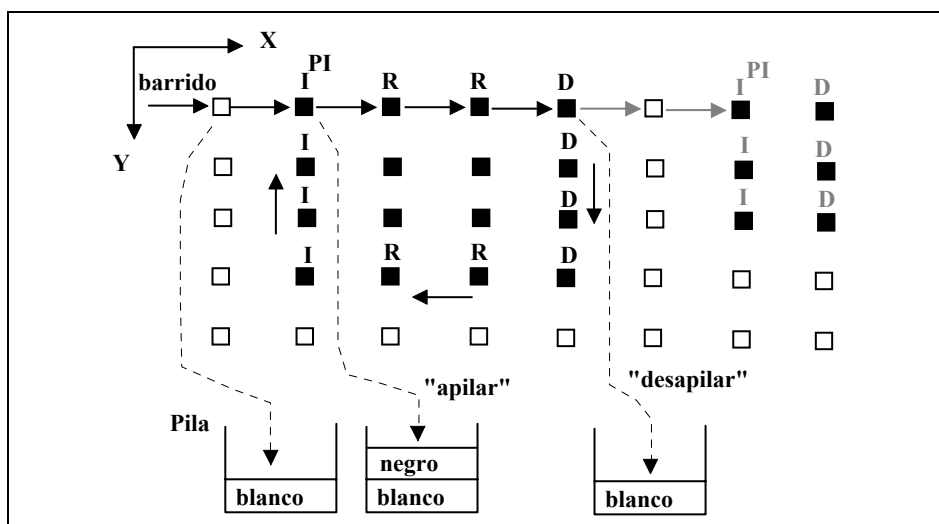
El algoritmo utilizado se basa en dos subalgoritmos:

(1)Subalgoritmo de búsqueda de puntos iniciales (**PI**).

Definido un nivel de gris para el fondo de la imagen (Blanco ó 63, en nuestro caso), el algoritmo hace un barrido por líneas de toda la imagen buscando puntos con un nivel de gris distinto del indicado. Una vez encontrado tal punto (si existe), llama al algoritmo de trazado de contorno. Cuando este termina, se continúa con la búsqueda de puntos iniciales evitándose que cualquier punto del objeto ya contorneado sea considerado como punto inicial mediante un proceso de marcado de los puntos del contorno y un inteligente manejo de una pila de niveles de gris (véase la figura 3.5). Nótese que el píxel alcanzado será un punto inicial si su atributo es izquierda (I, en la figura 3.5) y su nivel de gris no está en la misma mitad del rango de grises que el de la última entrada en la pila.

(2)Subalgoritmo de trazado de un contorno (**T**).

Desde el punto inicial de un contorno (que establecemos como el de mínima coordenada "y" del objeto) se utiliza la regla clásica de "atravesar un laberinto girando siempre en el mismo sentido" (supondremos sentido horario, en nuestro caso).

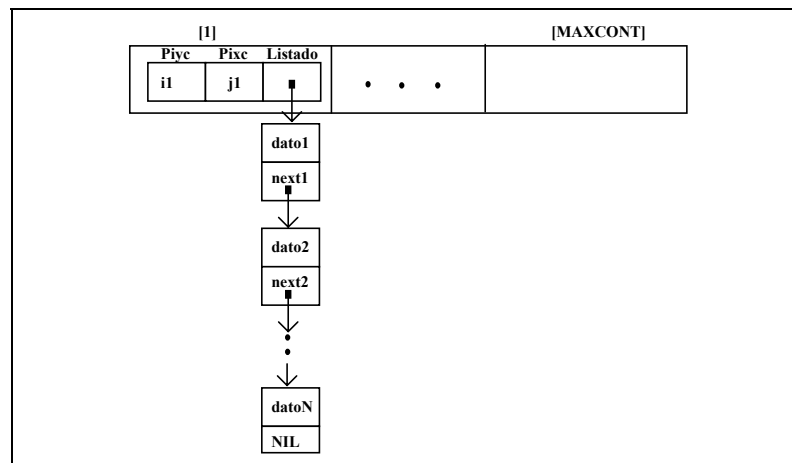


**Figura.3.5.** Representación gráfica del esquema de trabajo del algoritmo de "búsqueda de puntos iniciales". Se observa la detección de dos puntos iniciales, uno para cada contorno, cumpliendo la caracterización dada por el subalgoritmo correspondiente (1).



Cuestión-1: ¿Cómo podemos almacenar los contornos? (véase la figura 3.6).

Una primera idea puede ser almacenar cada una de las coordenadas de los puntos (*pixels*) que forman parte de cada contorno. Nosotros seguiremos la idea propuesta en [Gonzalez & Wintz, 87], y guardaremos únicamente las coordenadas de los puntos iniciales, PI, de cada uno, con la secuencia correspondiente de direcciones asociadas a los puntos que forman parte de dicho contorno. La implementación de esta estructura puede realizarse utilizando un vector cuyos elementos serán registros con tres campos: *Pixc*, *Piyc* (de tipo entero), que representarán las coordenadas del PI de cada contorno; y un campo de tipo puntero, *listado*, que apuntará a una lista cuyos nodos tendrán un único campo de información, *dato* (de tipo *byte*) que contendrá el código del segmento de la cadena, además del puntero al siguiente elemento (*next*). Los sucesivos contornos que se vayan detectando se almacenarán siguiendo la estructura de la figura 3.6. Posteriormente, con el objetivo de facilitar el razonamiento geométrico necesario en las fases posteriores del algoritmo, esta estructura de datos inicial del contorno, la lista dinámica, se trasvasa a un vector dinámico<sup>11</sup>, donde se guardan las coordenadas de los puntos del contorno.



**Figura 3.6.** Cada contorno se almacena, guardando las coordenadas de su punto inicial, PI, y creando una Lista a partir de éste que guarde las direcciones de los sucesivos puntos del contorno que encontramos a partir de dicho punto inicial.

Cuestión-2: ¿Cómo procederemos a una búsqueda exhaustiva de todos los contornos?:

Representaremos (en el programa), la **A** por un cero, la **I** por un uno, la **D** por un dos y la **R** por un tres. Empezaremos creando una matriz de atributos de las mismas dimensiones que la imagen. Se inicializa a cero (**A**). Y la recorreremos punto a punto. Al comienzo estaremos en un área de unos, es decir fuera de todo objeto. Cuando encontramos un **cero** éste será un punto inicial (**PI**). Entonces llamamos al procedimiento **T** que busca el contorno a partir de dicho punto. Cuando acaba lo recorremos de principio a fin aplicando la siguiente regla:

<sup>11</sup> Véase el apéndice D.

Sí el punto tiene un atributo A (no se ha pasado anteriormente por él) aplicamos la **regla 1<sup>a</sup>** (vista posteriormente). Sí por el contrario el punto ya tiene un atributo aplicaremos la **2<sup>a</sup> regla** (vista posteriormente).

Cada vez que pasemos de una zona a otra (de nivel de gris 0 a 1 o viceversa) lo indicaremos con una variable Booleana (en general, se utilizará una pila, véase la figura 3.5). Los puntos iniciales (**PI**) coinciden con el paso de un nivel al siguiente. El algoritmo termina cuando se han utilizado todos los puntos de la imagen.

A continuación se expone la estrategia seguida para implementar el algoritmo de seguimiento de contornos, teniendo en cuenta 8-conectividad:

**Entrada:** Array de niveles de gris (imagen pseudo-binarizada). Umbral:  $T_0$

**Salida:** Lista de contornos (PI, Codificación de Freeman y atributo de objeto o hueco)

**Pasos:**

1. Partir del punto superior izquierda de la imagen y suponer a éste perteneciente al fondo ( $z > T_0$ ).
2. Barrer la imagen de izquierda a derecha y de arriba a abajo hasta encontrar un punto en el que se cambie de zona de gris y con atributo **0**. Marcar este punto como **PI** de este contorno.
3. Trazar el contorno de la región detectada:
  - Inicializar valores del contorno y empezar por **PI**
  - repetir**
  - repetir**
  - mirar el píxel correspondiente según la regla de la figura 3-7
  - Si** z está al otro lado de  $T_0$  en el rango de grises **entonces**
  - avanzar a la siguiente dirección de salida posible según la figura 3-7
  - hasta** encontrar un píxel con z en el mismo lado del rango de grises
  - O** volver a la dirección de llegada sin haberlo conseguido.
  - si** se encuentra un píxel con z en el mismo lado del rango de grises **entonces**
  - Asignar la dirección de salida que se dirige hacia él desde el punto de entrada
  - si no**
  - Se trata de un contorno de un solo punto
  - hasta** Llegar de nuevo a **PI**
4. Marcar el contorno en la matriz de atributos recorriendo el contorno en sentido horario aplicando las reglas pertinentes para la asignación de atributos a cada píxel del contorno.
5. Repetir 2, 3 y 4 hasta llegar a la esquina inferior derecha de la pantalla.

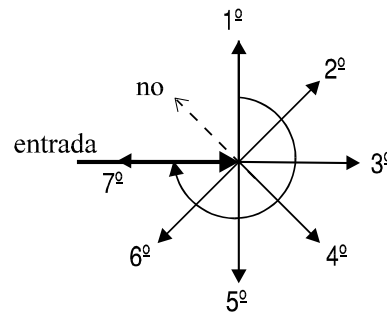
Finalmente, lo relativo a las Reglas de Asignación de Atributos explícitamente sería:

```

regla1 :
    r1 = ((3,3,3,0,2,2,2,2), (1,1,1,1,0,3,3,3),
          (1,1,1,1,3,0,3,3), (1,1,1,1,3,3,0,3),
          (1,1,1,1,3,3,3,0), (0,3,3,3,2,2,2,2),
          (3,0,3,3,2,2,2,2), (3,3,0,3,2,2,2,2));

regla2 :
    r2 = ((1,3,1), (3,2,2), (1,2,3));
  
```

Donde, r1, representa las 64 posibilidades asociadas con las ocho direcciones de entrada y salida (véase la figura 3.7), contempladas por el subalgoritmo de trazado de contornos, y r2, las nueve asociadas con los tres atributos posibles (I, D y R), cuando dicho algoritmo pasa por segunda vez por el mismo píxel del contorno.



**Figura 3.7.** Regla de salida en función de la entrada a cada píxel.

En resumen, una curva digital cerrada  $C$  queda representada por una secuencia de  $N$  puntos de coordenadas enteras:

$$C = \{ p_i = (x_i, y_i), \quad i = 1, \dots, N \}$$

donde  $p_{i+1}$  es un 8-vecino de  $p_i$  (módulo  $N$ ), puede expresarse mediante un punto inicial  $PI = (x_0, y_0)$  y un código de cadena de Freeman que se explicita como una sucesión de  $N$  vectores  $\{c_i, i=1, \dots, N\}$ , donde

$$c_i = \overline{p_{i-1}p_i}$$

y cada vector  $c_i$  es codificado mediante el valor de un entero  $c_i \in [0,7]$  (siendo  $c_i(\pi/4)$  el ángulo entre cada vector y el eje  $x$ ).

### Descriptores: Invariantes Geométricos.

En general, dada una imagen representada por una función arbitraria representativa de los niveles de gris en cada imagen,  $f(x,y)$ , se define el momento de orden  $(p,q)$ ,  $m_{pq}$  como

$$m_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} x^p y^q f(x, y) dx dy \quad p, q \in \{0, 1, 2, \dots, \infty\}$$

Si dicha función,  $f(x,y)$ , es acotada (como siempre ocurre en el caso de imágenes digitales) existirá un conjunto único de momentos generales asociados que la describen y viceversa [Duda & Hart, 73], [Maravall, 93]. Este resultado explica el porqué de la importancia de los descriptores basados en el cálculo de momentos en el área de reconocimiento de formas, con vistas a construir vectores de características, etc. En el presente trabajo, estos momentos geométricos serán utilizados para calcular el centroide, la orientación de la forma estudiada y las magnitudes de los semiejes principales de la elipse de mejor ajuste de los objetos de interés.

Teniendo en cuenta que las coordenadas del centroide de la función  $f(x,y)$  se definen como:

$$\bar{x} = \frac{m_{10}}{m_{00}} \quad ; \quad \bar{y} = \frac{m_{01}}{m_{00}}$$

podremos definir los denominados momentos centrales:

$$\mu_{pq} = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} (x - \bar{x})^p (y - \bar{y})^q f(x, y) dx dy \quad p, q \in \{0, 1, 2, \dots, \infty\}$$

En nuestro caso, la función  $f(x,y)$  representa la intensidad de cada uno de los puntos  $(x,y)$  de una imagen digital; por tanto, necesitamos la representación discreta de las definiciones anteriores [Fu et al., 88], [Maravall, 93]. Es decir,

$$m_{pq} = \sum_{x=0}^{N_x} \sum_{y=0}^{N_y} x^p y^q f(x, y) \quad p, q \in \{0, 1, 2, \dots, \infty\}$$

para los momentos geométricos, y

$$\mu_{pq} = \sum_{x=0}^{N_x} \sum_{y=0}^{N_y} (x - \bar{x})^p (y - \bar{y})^q f(x, y) \quad p, q \in \{0, 1, 2, \dots, \infty\}$$

para los centrales.

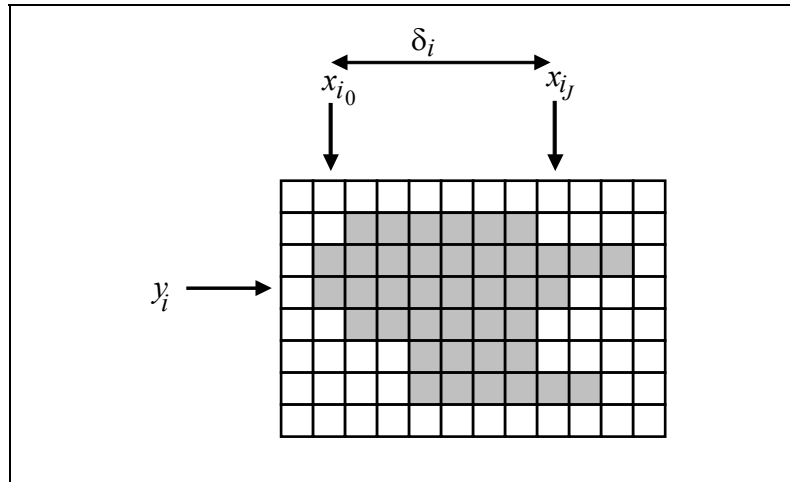
En el caso de una imagen binaria, la función  $f(x,y)$  es una función acotada que toma valores distintos de cero únicamente en el contorno de los objetos presentes en ella y en su interior. Para caracterizar una forma mediante un conjunto de momentos se puede optar por dos aproximaciones: (1) Considerar la función definida exclusivamente en el conjunto de puntos que definen su contorno; (2) Establecer el campo de existencia de dicha función en el conjunto de puntos del contorno del objeto más su interior.

Los momentos geométricos que se obtienen a partir del contorno, aunque lo caracterizan adecuadamente, sufren el defecto de ser muy sensibles al ruido y a pequeñas variaciones en la forma de un contorno. Por contra, los basados en el contorno más su interior presentan una mayor robustez [Maravall, 93]. Nosotros seguiremos este segundo enfoque.

Zakaria [Zakaria et al., 87] propuso un método llamado *regla delta* en el que los momentos geométricos totales se calculan sumando las contribuciones de cada una de las líneas de las que está compuesto el objeto sometido a estudio. Para ello basta con conocer la posición de los puntos del contorno, lo que reduce la complejidad a un orden lineal con el número de puntos del mismo, ver figura 3.8.

Este método será el utilizado cuando calculemos los  $m_{pq}$ , pues es sencillo de implementar y fácilmente generalizable para tratar objetos con agujeros, concavidades arbitrarias e imágenes no binarias, aspectos estos no contemplados en el trabajo original [Zakaria et al., 87]. El método integral [Dai et al., 92] también es eficiente y sencillo, pero sus ventajas respecto a la *regla delta* residen fundamentalmente en su implementación para el cálculo a partir del código de cadena de un contorno y este aspecto no contempla la posibilidad de objetos con agujeros internos.

Se ha desarrollado un algoritmo basado en la *regla delta* que lo extiende para alcanzar estos objetivos de generalización necesarios para nuestras formas (con agujeros y concavidades arbitrarias) [Sanz et al., 94].



**Figura 3.8.** Interpretación geométrica de la regla delta original. Se observa la contribución de la fila  $y_i$ , correspondiente a  $\delta_i = x_{i_j} - x_{i_0} + 1$ .

A partir de la definición para la regla delta original:

$$m_{pq} = \sum_{i=0}^{N_y} m_{pq,i} \quad p, q \in \{0, 1, 2, \dots\}$$

se obtienen los siguientes momentos de hasta orden dos:

$$m_{00,i} = \delta_i$$

$$m_{01,i} = \delta_i y_i = m_{00,i} y_i$$

$$m_{02,i} = \delta_i y_i^2 = m_{01,i} y_i$$

$$m_{10,i} = \delta_i x_{i_0} + (\delta_i^2 - \delta_i)/2$$

$$m_{11,i} = m_{10,i} y_i$$

$$m_{12,i} = m_{11,i} y_i$$

$$m_{20,i} = \delta_i x_{i_0}^2 + 2x_{i_0}(\delta_i^2 - \delta_i)/2 + \delta_i^3/3 - \delta_i^2/2 + \delta_i/6$$

$$m_{21,i} = m_{20,i} y_i$$

y sólo resta sumar las contribuciones de las  $N_y$  filas de cada momento de orden  $(p, q)$ , para obtener los momentos totales.

Como interesa trabajar con objetos sin restricciones (cóncavos, con huecos etc.), utilizaremos la regla delta extendida, la cual considera la posibilidad de que las contribuciones por filas de los píxeles de la imagen esté particionada en  $J_i$  secciones para cada línea  $i$ ,

$$m_{pq,i} = \sum_{j=1}^{J_i} m_{pq,ij}$$

De esta manera, los  $m_{pq}$  calculados con este método se obtienen en un tiempo del orden de 30 veces menor que los calculados a partir de la definición aplicada sobre la imagen binaria  $I(x, y)$  del objeto,

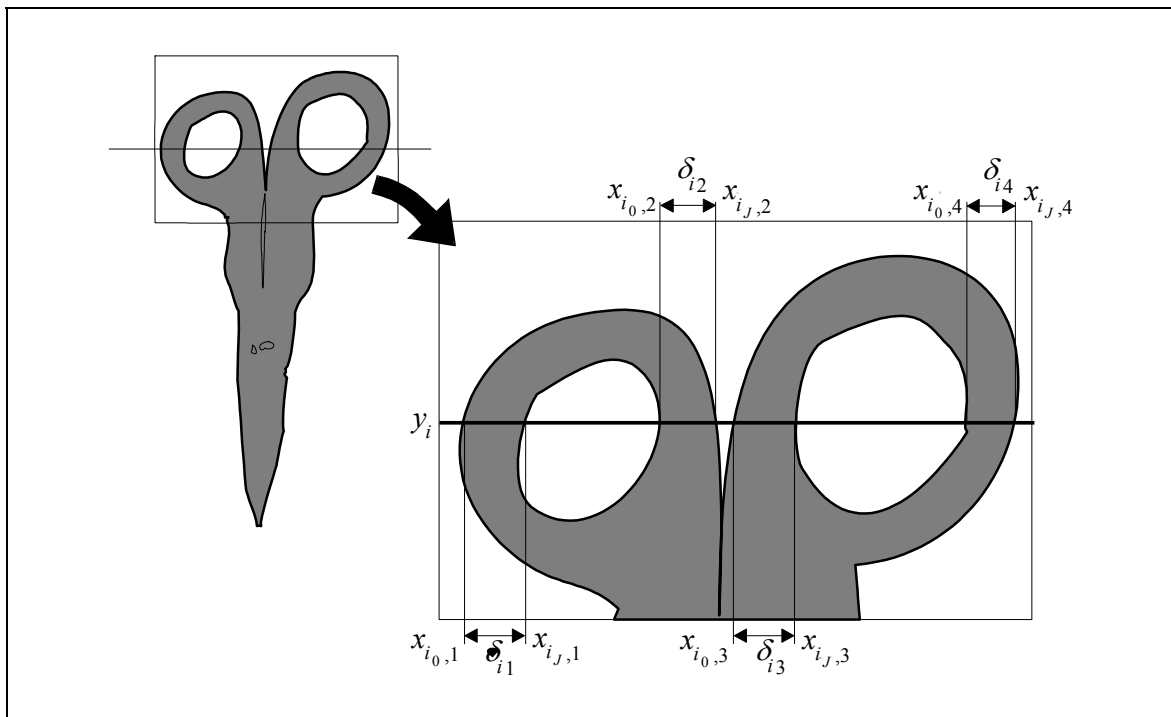
$$m_{pq} = \sum_{j=0}^{N_x} \sum_{i=0}^{N_y} x_j^p y_i^q I(x_j y_i)$$

que exhibe complejidad  $O(N_x \times N_y) \cong O(N^2)$ ; lo cual es muy conveniente para nuestros propósitos. Podemos ver su significado en el ejemplo de la figura 3.9.

A partir de estos  $m_{pq}$  se definen los correspondientes momentos centrales [Gonzalez & Wintz, 87]:

$$\begin{aligned} \mu_{00} &= m_{00} ; \quad \mu_{10} = \mu_{01} = 0 \\ \mu_{02} &= m_{02} - \bar{y}m_{01} \\ \mu_{11} &= m_{11} - \bar{y}m_{10} \\ \mu_{12} &= m_{12} - 2\bar{y}m_{11} - \bar{x}m_{02} + 2\bar{y}^2m_{10} \\ \mu_{20} &= m_{20} - \bar{x}m_{10} \\ \mu_{21} &= m_{21} - 2\bar{x}m_{11} - \bar{y}m_{20} + 2\bar{x}^2m_{01} \end{aligned}$$

Puntalicemos que los momentos centrales serán invariantes a traslaciones, al definirse relativos al centroide de la forma y no a coordenadas absolutas. Es interesante recordar que existen un conjunto de momentos, denominados "invariantes de Hu" [Hu, 62] que son invariantes a traslaciones, rotaciones y cambios de escala. Estas combinaciones no lineales de los momentos  $\mu_{pq}$  son de importancia crucial como descriptores en el área de reconocimiento de formas. No obstante, no se requiere su uso para los objetivos aquí planteados.



**Figura 3.9.** Interpretación geométrica de la regla delta extendida. Se observa la contribución de la fila  $y_i$ , correspondiente a  $\delta_{i1} + \delta_{i2} + \delta_{i3} + \delta_{i4}$ .

### Extracción de características basadas en los momentos.

Dos descriptores fundamentales serán:

1º El centro de masas o *Centroide* de la función de distribución de intensidades:

$$\bar{x} = \frac{m_{10}}{m_{00}}; \bar{y} = \frac{m_{01}}{m_{00}}$$

2º La orientación del *Eje Principal*, que coincide con la del eje de mínima inercia del objeto ( $I_{min}$ ):

$$\theta = \frac{1}{2} \operatorname{atan} \left[ \frac{2\mu_{11}}{\mu_{20} - \mu_{02}} \right]$$

dicho eje pasa por el centroide del objeto y, a partir del mismo, es trivial calcular el eje ortogonal a él pasando por el centroide: dirección del eje de máxima inercia ( $I_{max}$ ). Para evitar posibles ambigüedades en el cálculo de la orientación anterior, se hará uso de la función "atan2" (introducida en el Tema II), que es una función de dos argumentos que permite discriminar cualquier región angular sin posibilidad de error, teniendo en cuenta el seno y el coseno simultáneamente:

$$\theta = \frac{1}{2} \operatorname{atan} 2(2\mu_{11}, \mu_{20} - \mu_{02})$$

Una visualización gráfica de estos dos descriptores puede verse en la figura 3.10.

Estos dos ejes representan dos direcciones privilegiadas en el plano, de gran ayuda para automatizar la localización de los objetos en una escena. Dichas direcciones pueden visualizarse mediante la denominada "elipse de mejor ajuste" [Jain, 89], cuyos semiejes (mayor,  $a$ , y menor,  $b$ ), coinciden en dirección con los ejes de mínima y máxima inercia ( $I_{min}$ ,  $I_{max}$ ) respectivamente:

$$I_{min} = \frac{\mu_{20} + \mu_{02} - \sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{2}$$

$$I_{max} = \frac{\mu_{20} + \mu_{02} + \sqrt{4\mu_{11}^2 + (\mu_{20} - \mu_{02})^2}}{2}$$

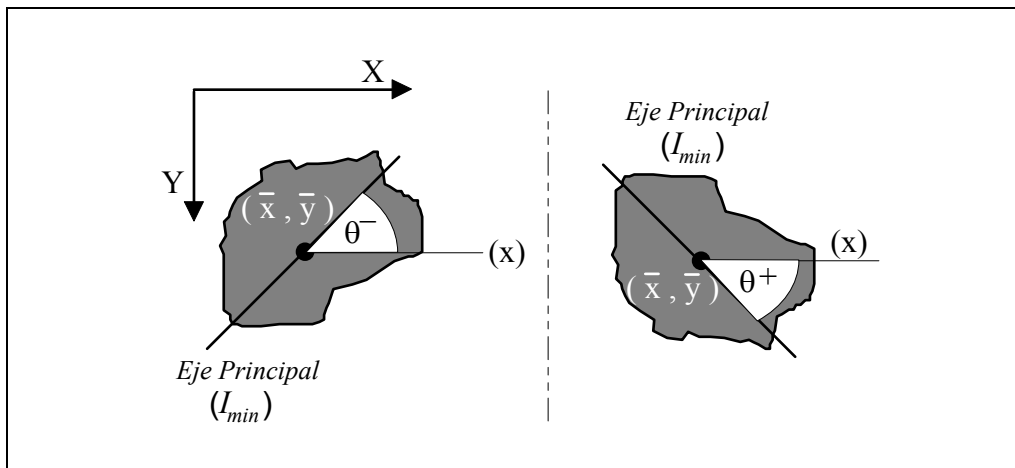
calculándose el módulo de dichos semiejes a partir de:

$$a = \left(\frac{4}{\pi}\right)^{1/4} \left[ \frac{I_{max}^3}{I_{min}} \right]^{1/8}; \quad b = \left(\frac{4}{\pi}\right)^{1/4} \left[ \frac{I_{min}^3}{I_{max}} \right]^{1/8}$$

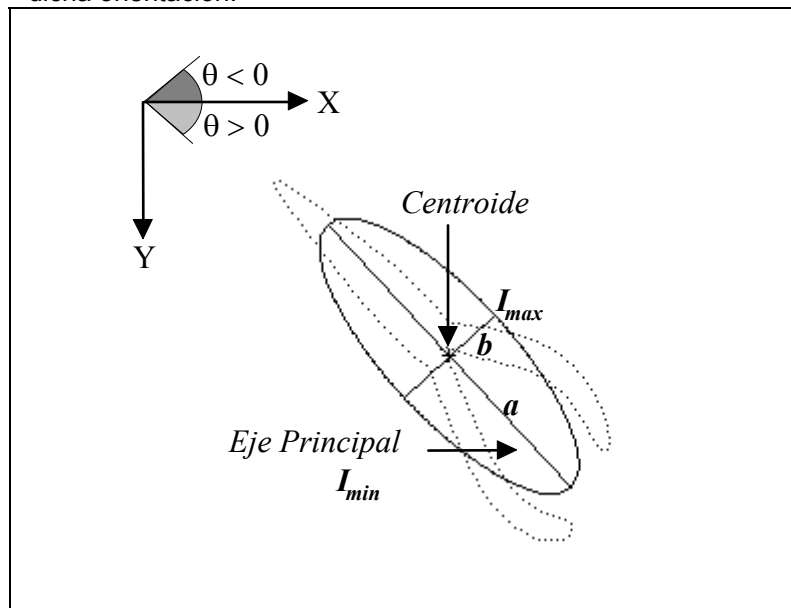
Se observa que ambas características se definen en base a los momentos centrales de segundo orden. Puntalicemos que es además en los momentos de menor orden donde se concentran las principales características de la forma de un contorno cerrado [Maravall, 93].

En la figura 3.11, se visualiza un ejemplo de una imagen real (alicates) donde se muestran los descriptores calculados, representándose además la elipse de mejor ajuste, de forma que se evidencia la bondad de los mismos. Obsérvese que la dirección del eje principal coincide con

la del momento de inercia mínimo (semieje mayor de la elipse), y la del momento de inercia máximo, con la dirección de la recta que pasando por el centroide es ortogonal al eje principal (semieje menor de la elipse). También se observa el convenio seguido en la orientación del eje principal ( $I_{max}$ ), en la esquina superior izquierda de dicha figura.



**Figura 3.10.** Interpretación geométrica de los descriptores utilizados: centroide y orientación ( $\theta$ ) del *Eje Principal*. Obsérvese el convenio seguido para dicha orientación.



**Figura 3.11.** Visualización sobre una imagen real (alicates) de la elipse de mejor ajuste, con sus semiejes mayor,  $a$  ( $I_{min}$ ) y menor,  $b$  ( $I_{max}$ ) y de los descriptores utilizados. En este caso el valor de la orientación de  $I_{min}$  es de  $47^\circ$ , siguiendo el convenio visualizado .

Con la sintaxis de Pascal, lo visto para la elipse de mejor ajuste se escribe:

$$IMin := ( (MC[2,0] + MC[0,2] - \text{Sqrt}(4 * \text{Sqr}(MC[1,1]) + \text{Sqr}(MC[2,0] - MC[0,2]))) / 2) ;$$



```

IMax:=( (MC[2,0]+MC[0,2]+Sqrt(4*Sqr(MC[1,1])+Sqr(MC[2,0]-MC[0,2]))) /2);
Ejea:=R4_pi_4*Potencia(IMax*IMax*IMax/IMin,0.125);
Ejeb:=R4_pi_4*Potencia(IMin*IMin*IMin/IMax,0.125);

```

Podemos completar éste conjunto de descriptores geométricos, considerando el denominado "esparcimiento":

```
Spread:=(IMin+IMax)/Sqr(mpq[0,0]);
```

Y la elongación:

```
Elong:=IMin/IMax;
```

El paso siguiente será normalizar los momentos centrales, de manera que además de ser invariantes frente a traslaciones lo sean frente a cambios de escala:

$$\eta_{p,q} = \frac{\mu_{p,q}}{m_{0,0}^{\frac{p+q}{2}}}; \quad p,q \in \{0,1,2,\dots\}; \quad p+q \in \{2,3,\dots\}$$

Con la sintaxis de Pascal:

```

MC[0,2]:=MC[0,2]/Sqr(mpq[0,0]);
MC[0,3]:=MC[0,3]/Potencia(mpq[0,0],5/2);
MC[1,1]:=MC[1,1]/Sqr(mpq[0,0]);
MC[1,2]:=MC[1,2]/Potencia(mpq[0,0],5/2);
MC[2,0]:=MC[2,0]/Sqr(mpq[0,0]);
MC[2,1]:=MC[2,1]/Potencia(mpq[0,0],5/2);
MC[3,0]:=MC[3,0]/Potencia(mpq[0,0],5/2);

```

Teniendo en cuenta lo anterior, se demuestra<sup>12</sup> la existencia de siete momentos invariantes ("de Hu"),  $\phi_i$ , a traslaciones, rotaciones y cambios de escala, definidos a partir de los momentos centrales normalizados:(Sintaxis de Pascal)

```

MomInvar[1]:=MC[2,0]+MC[0,2];
MomInvar[2]:=Sqr(MC[2,0]-MC[0,2])+4*Sqr(MC[1,1]);
MomInvar[3]:=Sqr(MC[3,0]-3*MC[1,2])+Sqr(3*MC[2,1]-MC[0,3]);
MomInvar[4]:=Sqr(MC[3,0]+MC[1,2])+Sqr(MC[2,1]+MC[0,3]);
MomInvar[5]:=(MC[3,0]-3*MC[1,2])*(MC[3,0]+MC[1,2])*(Sqr(MC[3,0]+MC[1,2])-
3*Sqr(MC[2,1]+MC[0,3]))+(3*MC[2,1]-
MC[0,3])*(MC[2,1]+MC[0,3])*(3*Sqr(MC[3,0]+MC[1,2])-
Sqr(MC[2,1]+MC[0,3]));
MomInvar[6]:=(MC[2,0]-MC[0,2])*(Sqr(MC[3,0]+MC[1,2])-
Sqr(MC[2,1]+MC[0,3]))+4*MC[1,1]*(MC[3,0]+MC[1,2])*(MC[2,1]+MC[0,3]);
MomInvar[7]:=(3*MC[2,1]-MC[0,3])*(MC[3,0]+MC[1,2])*(Sqr(MC[3,0]+MC[1,2])-
3*Sqr(MC[2,1]+MC[0,3]))+(3*MC[1,2]-
MC[3,0])*(MC[2,1]+MC[0,3])*(3*Sqr(MC[3,0]+MC[1,2])-
Sqr(MC[2,1]+MC[0,3]));

```

*Observaciones finales:*

- \* El objetivo perseguido por nosotros al calcular los momentos está más dirigido a la discriminación que a la reconstrucción de los contornos de los objetos.
- \* Podemos resumir lo visto hasta aquí diciendo que dado un objeto en forma de función bidimensional  $f(x,y)$  que toma el valor unidad en su contorno y en el interior del mismo, y el valor nulo fuera de dicha región, se obtendrá un conjunto finito de características invariantes a traslaciones, giros y cambios de escala (u homotecias) según el siguiente proceso secuencial:

$$m_{p,q} \rightarrow \mu_{p,q} \rightarrow \eta_{p,q} \rightarrow \phi_{p,q}$$

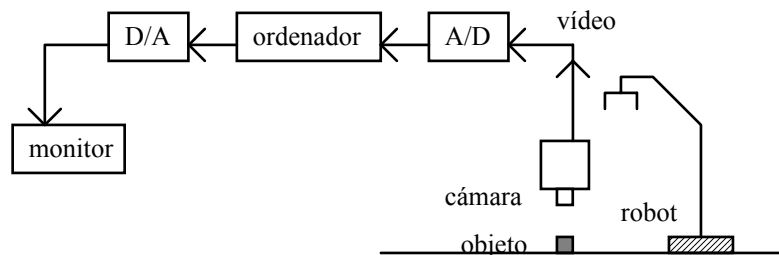
<sup>12</sup>[Hu MK, 62]

Digamos para finalizar que,

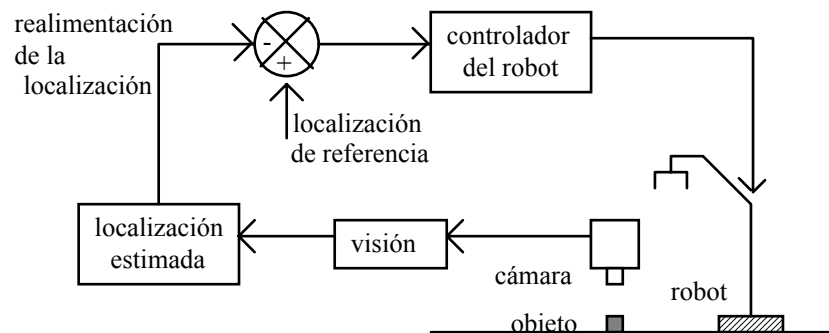
- 1º) Será un problema práctico a resolver en cada aplicación específica el nº de estos invariantes a considerar como elementos finales del vector de características. Y
- 2º) En la obtención del subconjunto de momentos invariantes discriminantes se comienza siempre desde los momentos de menor a mayor orden. Esto es así porque es en los momentos de menor orden donde están concentradas las principales características de la forma de un contorno cerrado.

### Ejemplo práctico: "Sistema de Coordinación Ojo-Mano".-

Desde el punto de vista de la **visión** por ordenador:



Desde el punto de vista del **control del manipulador**, mediante realimentación visual:

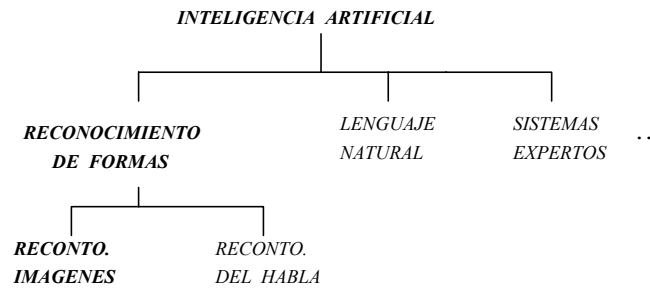


En general, digamos que el problema planteado de "coordinación ojo-mano", sigue siendo en la actualidad una línea abierta de investigación activa. Tengamos en cuenta que entran en juego muchos factores como son:

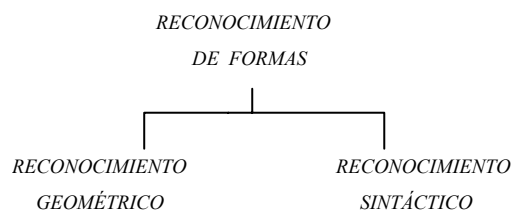
- 1).- Los relacionados con el calibrado del conjunto del sistema.
- 2).- Los asociados al control del manipulador (posicional, de trayectorias, etc.) y su comunicación con los módulos de visión.
- 3).- La estrategia seguida por los algoritmos de visión para extraer la información necesaria en tiempo real.
- 4).- Recuperación de la profundidad del objeto (mediante laser, luz estructurada, etc.)

### 3.3 RECONOCIMIENTO DE FORMAS: INTRODUCCIÓN

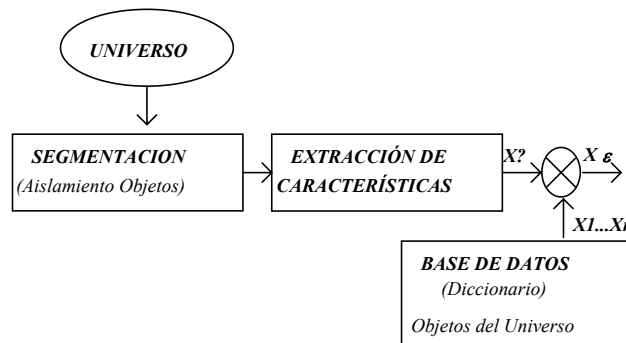
A continuación representamos, de modo gráfico, la ubicación en el contexto de la Inteligencia Artificial (IA), de la disciplina conocida como "Reconocimiento de Formas":



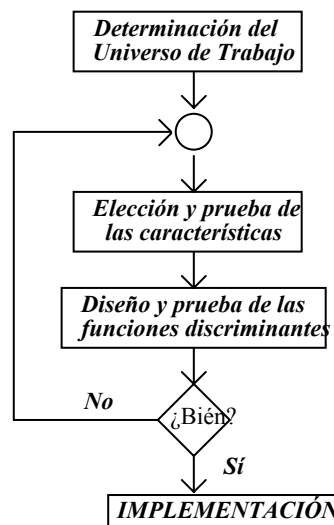
La cual a su vez, se puede descomponer en dos vertientes:



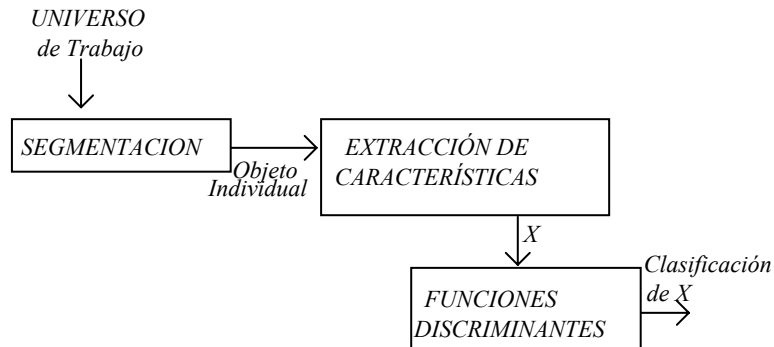
En general, un posible diagrama de bloques de un Sistema de Reconocimiento Automático de Formas, sería:



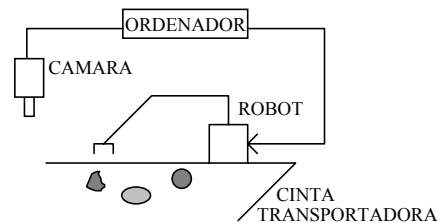
De modo esquemático, las etapas del diseño de un Sistema de Reconocimiento Automático de Formas serán:



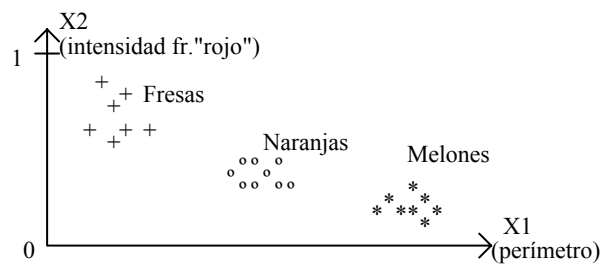
Una vez superada la etapa de diseño anterior, se obtiene el siguiente diagrama de bloques de un Sistema de Reconocimiento Automático de Formas en la fase operativa:



Ejemplo<sup>13</sup>. - Un posible sistema robotizado para la selección de frutas.



La distribución en el plano de las tres *clases* del Ejemplo anterior:



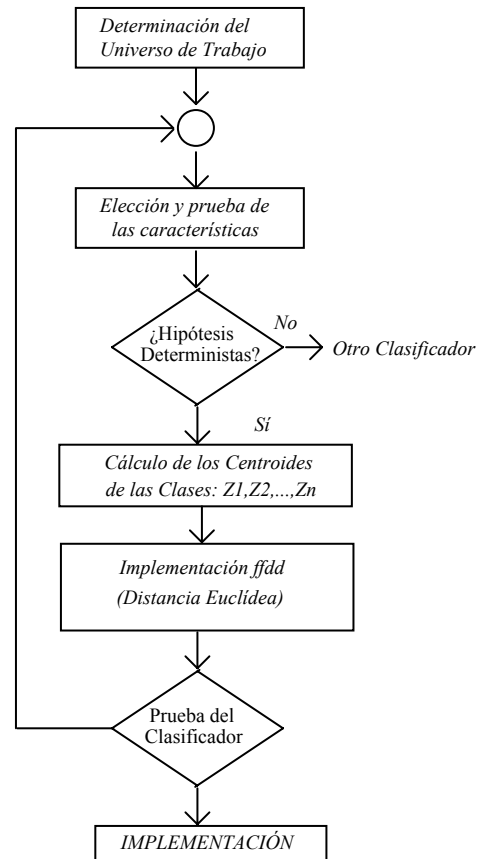
Propiedades a tener en cuenta en la *elección y prueba* de las *características*:

- 1<sup>a</sup>. - Poder discriminante
- 2<sup>a</sup>. - Fiabilidad
- 3<sup>a</sup>. - Incorrelación
- 4<sup>a</sup>. - Tiempo de computo aceptable
- 5<sup>a</sup>. - Economía de medios

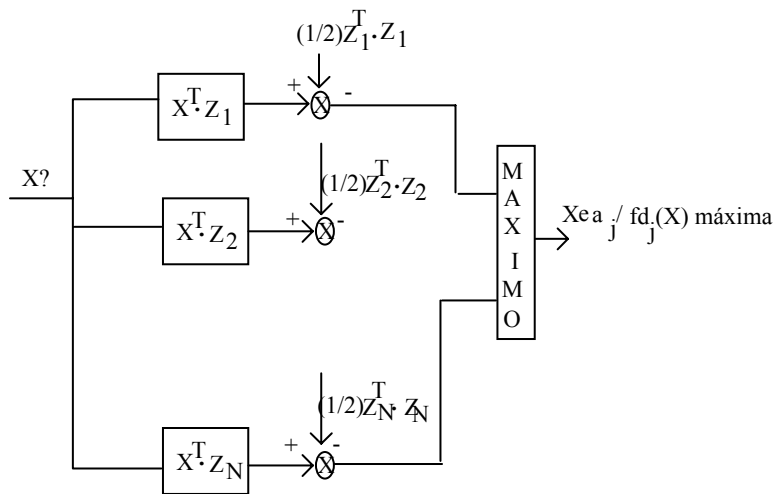
### 3.3.1 El Clasificador Euclídeo

Fase de *diseño* del Clasificador Euclídeo:

<sup>13</sup> Ver [Maravall, 93]



Fase operativa del Clasificador Euclídeo:

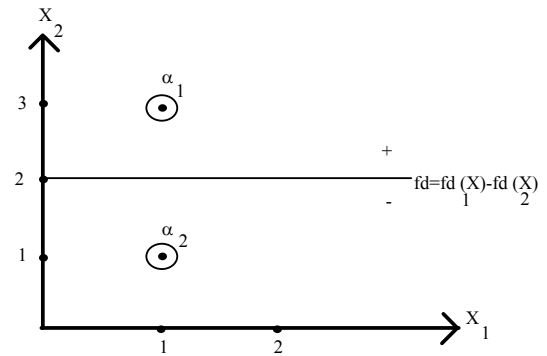


**Observaciones:**

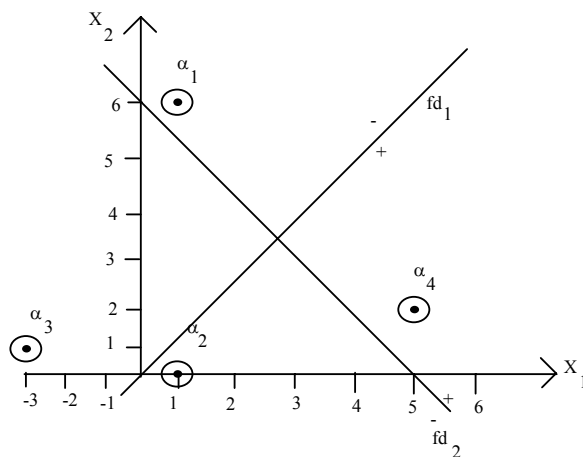
El vector  $X$  a clasificar es operado en paralelo por las  $N$  funciones discriminantes, de modo que la *carga computacional* de éste clasificador es igual<sup>14</sup> a  $N \cdot n$  productos reales y  $N$  sumas.

Ejercicio 1: Demostrar que  $fd(X) = X_2 - 2$ , para la distribución biclase siguiente,

<sup>14</sup> Suponiendo que la dimensión del vector de características es  $n$  y que existen  $N$  clases.



**Ejercicio 2:** Sea la distribución multiclase siguiente,



a).-Encontrar las funciones de decisión asociadas al clasificador Euclídeo.

b).-Razonar los pros y los contras del clasificador Euclídeo frente al clasificador por regiones.

**Clasificación (Estrategia de la Distancia Mínima)**

Regla de Decisión:

---

Dada una muestra (o patrón)  $X_0$  decidir  $\alpha_i$  ( $X_0 \text{ ### } w_i$ ) sii  $D_{0i} \leq D_{0j} \quad \text{### } j = 1, 2, \dots$

---

Procedimiento operativo clasificador *Distancia Mínima* [UPV, 91]:

```

INICIO programa
  inicializar y leer las listas de objetos de muestra (aprendidos)
  seleccionar el nivel de umbralización
  capturar una imagen
  umbralizar la imagen
  MIENTRAS (hay objetos en la imagen)
    trazar el contorno
    calcular momentos geométricos de la superficie del objeto
    calcular momentos geométricos del contorno del objeto
    calcular 14 invariantes (X vector de características)
  
```

```
    escalar el vector  $X_0$ 
    calcular la distancia a todas las muestras de todas las clases
    SI (existe un empate)
        asignar  $X_0$  aleatoriamente a alguna de las clases del empate
    SINO
        asignar  $X_0$  a la clase de mínima distancia
    FIN SI
FIN MIENTRAS
FIN del programa
```

### ***Clasificación (Estrategia de los K Vecinos más Próximos)***

Regla de Decisión:(K-Nearest Neighbor)

---

Dada una muestra (o patrón)  $X_0$  decidir  $\alpha_i$  ( $X_0$  ###  $w_i$ ) sii la clase  $i$  es la más numerosa entre el conjunto de  $K$  muestras más próximas a  $X_0$

---

Procedimiento operativo clasificador "K-Vecinos más Próximos" [UPV, 91]:

```

INICIO programa
  inicializar y leer las listas de objetos de muestra (aprendidos)
  seleccionar el nivel de umbralización
  capturar una imagen
  umbralizar la imagen
  MIENTRAS (hay objetos en la imagen)
    trazar el contorno
    calcular momentos geométricos de la superficie del objeto
    calcular momentos geométricos del contorno del objeto
    calcular 14 invariantes (X vector de características)
    escalar el vector  $X_0$ 
    inicializar  $i = 1$ 
    HACER HASTA (encontrar los K-vecinos más próximos)
      calcular la distancia de  $X_0$  a  $X_i$ 
      SI1 ( $i \leq K$ )
        incluir  $X_i$  en la lista de los K-vecinos más próximos
      (KNN)
      SINO
        SI2 ( $X_i$  es más próximo que alguno de los KNN anteriores)
          eliminar el vecino más lejano de la lista KNN
          incluir  $X_i$  en la lista KNN
        FIN SI2
      FIN SI1
      incrementar  $i$ 
    FIN HACER
    determinar la clase más votada en la lista KNN
    SI3 (existe un empate)
      calcular la suma de las distancias a los vecinos dentro
      de cada clase
      SI4 (hay otro empate)
        clasificar  $X_0$  aleatoriamente en alguna de las clases del
        empate
      SINO
        clasificar  $X_0$  en la clase con menor suma
      FIN SI4
    SINO
      clasificar  $X_0$  en la clase más votada de la lista KNN
    FIN SI3
  FIN MIENTRAS
FIN del programa

```

### **3.3.1 Aplicación Práctica: Sistema de Visión Industrial SRI<sup>15</sup>.**

<sup>15</sup> Ver [Klafter et al. 89]



Se trata de un sistema de visión desarrollado en la década de los 70 en el "Stanford Research Institute" (EEUU). Ha tenido una importancia crucial en la implantación de sistemas de visión en diferentes entornos industriales, así como una gran influencia en el campo de investigación en robótica, e.g. diseño del lenguaje RAIL<sup>16</sup>. Los algoritmos SRI tenían como objetivos fundamentales:

- La clasificación de objetos.
- La manipulación robotizada de piezas y materiales.
- La inspección visual de piezas.

Entre los diferentes aspectos a tener en cuenta para la consecución de los objetivos anteriores están:

- Técnicas de iluminación y preprocesamiento de imágenes.
- Hardware específico para análisis de imágenes.
- Extracción de características.
- Reconocimiento automático de piezas.

### **RESUMEN T-III. Aspectos Clave:**

- La Percepción Artificial constituye un elemento crucial en el desarrollo de la Robótica actual. Imprescindible para llevar a cabo Tareas Complejas en Entornos No-Estructurados.
- Aún cuando para los seres vivos, incluso los no inteligentes, la percepción es algo inmediato y aparentemente sencillo, resulta extraordinariamente complicada y difícil mediante ordenador.
- Tanto la Electrónica y el Procesado Digital de Imágenes, como un conjunto de Técnicas agrupadas bajo el nombre genérico de I.A., suponen una aportación decisiva a la resolución de éste problema que, sin embargo, dista mucho de estar resuelto de una forma General.

---

<sup>16</sup> Desarrollado por Automatrix, Inc (Massachusetts)

## TEMA IV COMUNICACIÓN HOMBRE-ROBOT

### Objetivos:

- Introducir al alumno en el contexto de la programación de robots.
- Estudiar los diferentes tipos de lenguajes existentes.
- Poner de manifiesto las dificultades inherentes a este tipo de programación.
- Vislumbrar las metas que se pretenden alcanzar en un futuro próximo. Una primera aproximación a los lenguajes de programación avanzados.

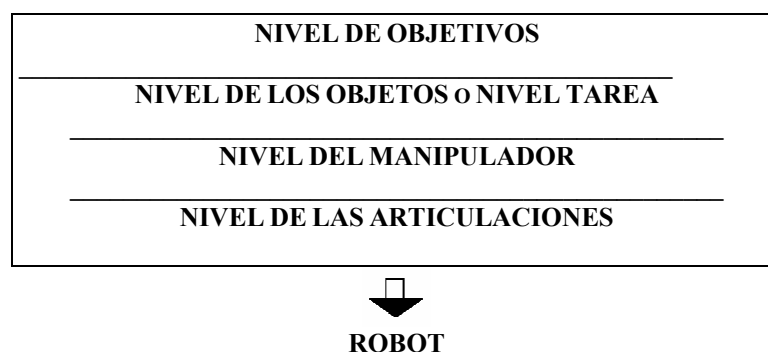
### 4.1 ASPECTOS DIFERENCIALES DE LA PROGRAMACIÓN DE ROBOTS

¿Qué circunstancias hacen diferentes los lenguajes de programación de robots del resto?

- El entorno en que actúa el robot generalmente no puede describirse en términos puramente cuantitativos.
- Se necesita poder incluir condiciones no usuales, tales como la prevención de colisiones.
- Las acciones del robot están sujetas a imprecisiones que pueden dar lugar a incidentes que el programa debe ser capaz de tratar.
- Ciertas informaciones sensoriales del sistema pueden ser no sólo difíciles de procesar en tiempo real sino también ambiguas.

### 4.2 CLASIFICACIÓN DE LOS LENGUAJES DE PROGRAMACIÓN DE ROBOTS

Una posible clasificación de los lenguajes de programación en robótica, puede hacerse estableciendo niveles, según se adapten en mayor o menor medida al lenguaje natural del hombre o al robot. Esto se observa mejor en la figura 4.1.



**Figura 4.1.** Adaptada de [Gini et al. 85]. Descripción esquemática, por niveles, de los lenguajes de programación de robots.

En el primer nivel, de las articulaciones, existen lenguajes como ARMBASIC y MAL que llegan a especificar incluso los pasos de los motores internos del robot para ejecutar un determinado movimiento.

A nivel del manipulador, los lenguajes incorporan ya la conversión entre el espacio cartesiano y el de las articulaciones, así como también la posibilidad de generar determinadas trayectorias entre puntos del espacio. Estos lenguajes son los que en la actualidad están más extendidos en los robots industriales y un representante característico sería VAL, desarrollado por Unimation en la década de los 70, para controlar su robot PUMA.

En el siguiente nivel es donde existe un mayor esfuerzo investigador por parte de la comunidad científica. Aquí se intersecta con otro tópico de gran importancia por su repercusión en la industria: "*planificación del ensamblado*". Básicamente se pretende: a partir de un modelo del mundo (representación geométrica del entorno del robot, incluido el robot), mediante la construcción de un "planificador de tareas", traducir automáticamente las especificaciones "estado inicial/estado objetivo" (*especificaciones a nivel de tarea*), al código de un manipulador específico (*especificaciones a nivel de manipulador*).

Fruto del esfuerzo investigador en esta línea son los lenguajes: RAPT, desarrollado en la Universidad de Edimburgo [Poplestone et al., 78], LAMA, desarrollado en el M.I.T. [Lozano-Perez et al., 77], AUTOPASS, desarrollado por IBM [Lieberman et al., 77]. En la frontera entre el nivel del manipulador y el de los objetos estaría el lenguaje AL, desarrollado en la Universidad de Stanford [Finkel et al., 74], del cual arrancan algunas ideas incorporadas en los anteriores lenguajes.

En el nivel más alto planteado, se sitúan los lenguajes a nivel de objetivos, donde se pretende que sea el propio robot el que cree sus planes para cada tarea que se le ordene mediante lenguaje natural. Esto requiere un sistema capaz de: razonamiento geométrico, comprensión del lenguaje natural, generación de planes, interpretación de imágenes etc. Todo lo cual lo hace inaccesible por el momento. Los esfuerzos investigadores se centran en la etapa anterior, pues está claro que hasta que aquella no esté resuelta de modo satisfactorio difícilmente podremos pasar a la siguiente.

### **4.3 LENGUAJES DE PROGRAMACIÓN A NIVEL ROBOT**

Los lenguajes de programación a nivel manipulador o nivel robot constituyen los denominados lenguajes industriales, puesto que es para dichos robots para lo que se han diseñado. Veremos solo dos ejemplos recientes de dichos lenguajes: RAPID, desarrollado por la multinacional sueca ABB a partir de 1994, y V+, desarrollado por Adept Technology en 1989.

#### **4.3.1 Lenguaje de Programación RAPID**

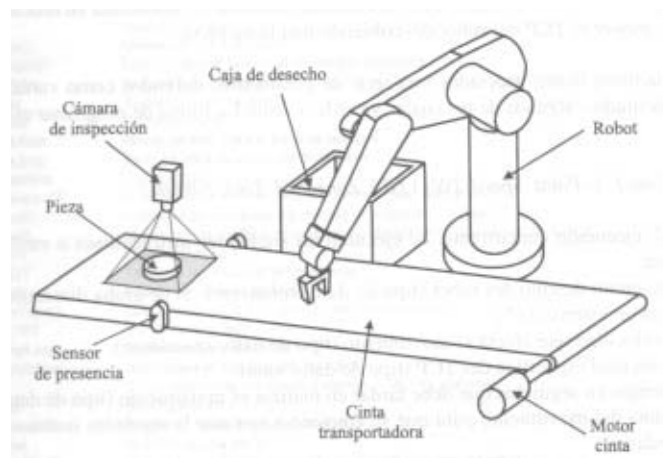
Los programas desarrollados en RAPID se denominan tareas e incluyen el programa en sí junto con varios módulos de sistema, que contienen rutinas y datos de tipo general, independientes del programa pero que pueden ser utilizados por él en cualquier momento. A su vez, el programa puede ser dividido en varios módulos, uno de los cuales ha de ser el principal. Cada uno de estos módulos contiene submódulos de datos, además de diversas rutinas de ejecución.

La definición de módulos se realiza especificando su nombre y sus atributos, como por ejemplo que se trata de un módulo del sistema (SYSMODE) o que no se puede modificar (VIEWONLY).

Existen tres tipos diferentes de rutinas o subprogramas posibles:

- **Procedimiento:** Rutina que no devuelve ningún valor y se utiliza como una instrucción.
- **Función:** Rutina que devuelve un dato de tipo específico y que se utiliza como una expresión.
- **Rutina TRAP:** Rutinas que se asocian a interrupciones y se ejecutan cuando éstas se activan. No pueden llamarse nunca de forma explícita.

*Ejemplo* [Barrientos et al., 97] “Selección de piezas defectuosas”



El robot retira de una cinta transportadora aquellas piezas identificadas como defectuosas. El robot se encuentra en espera hasta la llegada de una señal indicando la existencia de una pieza defectuosa sobre la cinta transportadora. El robot procede entonces a parar la cinta y a coger la pieza, depositandola en un almacén de piezas defectuosas. El propio robot se encarga de activar de nuevo el movimiento de la cinta, una vez que la pieza ha sido cogida. Tras la operación, el robot vuelve a su posición inicial y se repite de nuevo el ciclo. El programa cuenta con una rutina principal junto con varias subrutinas específicas, además de la definición de las variables correspondientes.

Programa principal (PROC main())

Comprende todas las instrucciones a ejecutarse para la realización de la tarea.

Primero se va a la posición inicial de espera y se aguarda a la indicación de que existe una pieza defectuosa. Tras parar la cinta y coger la pieza, el robot lleva la cinta al almacén de desechos y la deposita allí. Este proceso se repite hasta que se presione un botón asociado a la entrada digita **terminar**.

Rutina de ir a la posición de espera

Se mueve al robot desde la posición en la que se encuentre hasta la posición de espera.

Rutina de coger la pieza de la cinta

Se coge una pieza de la cinta transportadora. Se realiza primero una aproximación en coordenadas angulares, movimiento coordinado en el espacio de articulaciones (trayectoria isocrona), para más tarde acercarse en línea recta y con precisión a coger la pieza.

**OJO**-> Existen tres modos de instrucciones de movimiento: MoveC (TCP describe un círculo), MoveJ (TCP con trayectoria articular), y MoveL(TCP describe una línea recta).

Formato:

## MoveJ [\Conc] ToPoint Speed [\V] \ [T] Zone [\Z] Tool [\WObj]

**PROC main()**

```

  Ir_posicion_espera;
  WHILE DInput(terminar)=0 DO
    IF DInput(pieza_defectuosa)=1 THEN
      SetDO activar_cinta,0;
      Coger_pieza
      SetDO activar_cinta,1;
      Dejar_pieza
      Ir_posicion_espera;
    ENDIF
  ENDWHILE
ENDPROC

```

**PROC Ir\_posicion\_espera()**

```

  MOVEJ conf_espera, VMAX,z30,herramienta
ENDPROC

```

**PROC Coger\_pieza()**

```

  MOVEJ*, VMAX,z60,herramienta
  MOVEJ*, V500,z20,herramienta
  MOVEJ*, V150,FINE,herramienta
  Coger
  MOVEJ*, V200,z20,herramienta
ENDPROC

```

**PROC Dejar\_pieza()**

```

  MOVEJ*, VMAX,z30,herramienta
  MOVEJ*, V300,z30,herramienta
  Dejar
ENDPROC

```

## Definición de Variables

```

PERS tooldata herramienta := [FALSE, [[97,0,223], [0.924,0,0.383,0] ],
                               [5[-23,0,75], [1,0,0,0],0,0,0] ]
PERS loaddata carga := [5[50,0,50], [1,0,0,0],0,0,0]

```

```

VAR signaldo pinza ;señal de activación de pinza
VAR signaldo activar_cinta ;señal de activación de cinta
VAR signaldi pieza_defectuosa ;señal de activación de pieza defectuosa
VAR signaldi terminar ;señal de terminar programa

```

```

PERS robtarjet conf_espera := [[600,500,225], [1,0,0,0], [1,1,0,0] ,
                               [9E9, 9E9, 9E9, 9E9, 9E9, 9E9] ]

```

## Rutinas de control de la pinza

**PROC Coger()**

```

  Set pinza ; cerrar pinza
  WaitTime 0.3 ; esperar 0.3 segundos
  GripLoad carga ; señal de pieza cogida
ENDPROC

```

**PROC Dejar()**

```

  Reset pinza ; abrir pinza
  WaitTime 0.3 ; esperar 0.3 segundos
  GripLoad LOAD0 ; ausencia de carga
ENDPROC

```

Tipos de datos: **atómicos** y **registros**.

Declaración de datos: **Constantes** (CONS), **Variables** (VAR) y **Persistentes** (PERS)-> se trata de variables en las que cada vez que se cambia su valor durante la ejecución del programa, también se cambia el valor de su inicialización.

Estructuras predefinidas: (ejemplos)

**Tooldata:** Especifica las características de una herramienta.

Campos: *robhold*: define si el robot tiene la herramienta o no.

*tframe*: sistema de coordenadas de la herramienta.

*tload*: dato tipo *loaddata*.

**Loaddata:** Especifica la carga colocada en la muñeca del robot.

Campos: *mass*: peso de la carga en kilogramos.

*cog*: centro de gravedad de la carga.

*aom*: orientación de los ejes de inercia en el centro de gravedad expresada como cuaternios.

*ix, iy, iz*: momentos de inercia de la carga ( $\text{kg m}^2$ ).

**Robtarget:** define la localización del robot y de los ejes externos.

Campos: *trans*: desplazamiento en x,y,z del sistema de coordenadas.

*rot*: rotación del sistema de coordenadas como cuaternios.

*robconf*: configuración del robot (*cf1, cf4, cf6* y *cfx*).

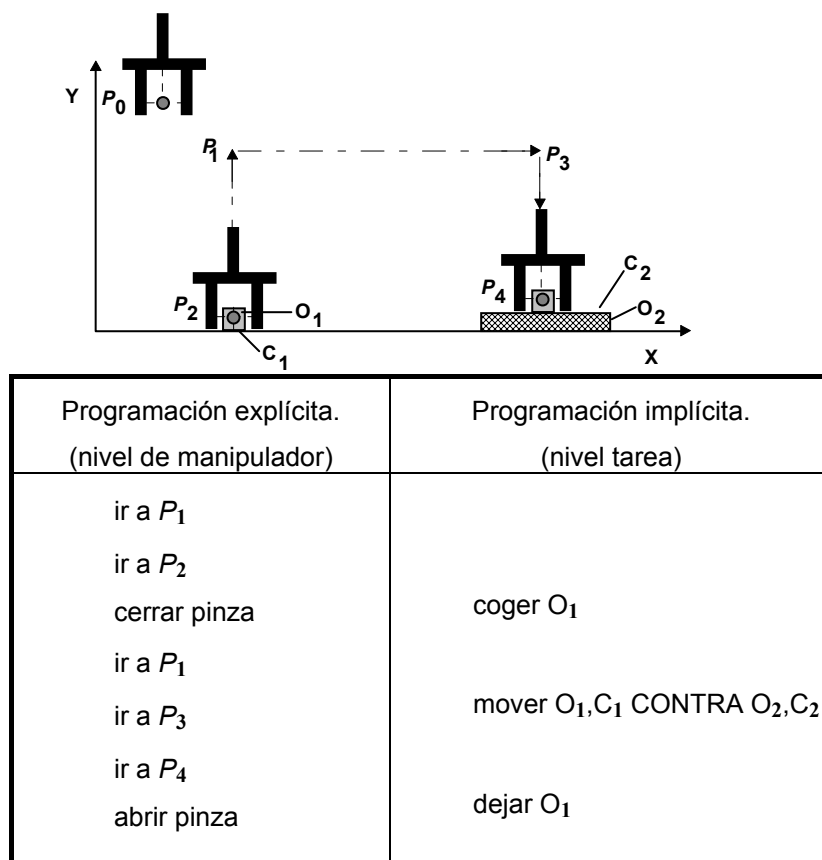
*extax*: posición de los ejes externos.

### 4.4 LENGUAJES DE PROGRAMACIÓN A NIVEL TAREA

El concepto de robot inteligente incluye la capacidad de recibir instrucciones de alto nivel expresadas como "comandos" para realizar una tarea general, trasladando dichas instrucciones a un conjunto de acciones que deben ejecutarse para llevar a cabo dicha tarea. Será consciente de su entorno y capaz de tomar decisiones acerca de sus acciones basadas, en parte, en la interpretación de dicho entorno.

Un ejemplo sencillo que pone de manifiesto el significado de lo dicho, puede observarse en la figura 4.2, donde se representan dos alternativas para programar una tarea sencilla de coge y deja. La primera, a la izquierda, representa la programación usual, denominada de "nivel robot", que requiere especificar cada una de las acciones del mismo (programación explícita), e.g. cada posición distinta a alcanzar es una acción diferente a tener en cuenta. Si se requiere encapsular diferentes movimientos en unas pocas acciones, estamos obligados a integrar información sensorial que permita disparar de forma automática dichas acciones, en nuestro caso, de manipulación (parte derecha de la figura), lo cual necesita, además, de una representación del modelo del mundo (programación orientada al modelo o implícita). El tipo de declaraciones simbólicas:

*mover* objeto 1, característica 1 *contra* objeto 2, característica 2.



**Figura 4.2.** Adaptada de [Koutsou 81]. Alternativas en la programación de una tarea simple de "coge y deja". En la parte izquierda se observan las instrucciones necesarias para ejecutar la tarea a nivel de manipulador, orientada a las

acciones. En la parte derecha, su equivalente en versión de nivel tarea, orientada al objeto (modelo del mundo).

donde aparecen relaciones que implican restricciones espaciales (e.g. *contra*, *coplanar*, etc.) es típico de los lenguajes de programación a nivel tarea como RAPT ([Poplestone et al., 78], [Poplestone et al., 80]) o LM-GEO ([Mazer, 83]), una de cuyas metas es facilitar la especificación de tareas al usuario. La idea es que a partir de, únicamente, ciertas declaraciones simbólicas que definen unos estados objetivos se pueda desencadenar automáticamente la correspondiente secuencia de operaciones a nivel del manipulador.

De las tres fases básicas de la planificación: Modelado del Mundo, Especificación de Tarea y Síntesis del Programa de Manipulador, nos centraremos en la de Especificación de Tarea. Aquí existen tres métodos básicos de describir la tarea:

- Gestual, utilizando el propio efector final del robot para grabar las posiciones
- Por sistemas CAD
- Mediante Lenguajes formales

Donde existen a su vez dos líneas básicas de ataque del problema:

- Especificación por Secuencia de Estados
- Especificación por Secuencia de Operaciones

RAPT utiliza la especificación de tarea por secuencia de estados del modelo del mundo. Mediante Relaciones Espaciales Simbólicas, entre características de los objetos, limita sus posibles configuraciones, intentando conseguir eliminar cualquier tipo de ambigüedad.

Una limitación importante para éste método es la de que no especifica toda la información necesaria para describir una operación. Por ejemplo, [Fu et al., 88], el par necesario para apretar un tornillo no puede ser incluido en la descripción de estados. Una solución alternativa es escribir la tarea como una secuencia de operaciones simbólicas con los objetos: Especificación de Tareas por Secuencia de Operaciones. En ello se basa AUTOPASS, que admite sentencias como la siguiente:

```
DRIVE IN tornillo AT cabeza_tornillo SUCH THAT TORQUE IS EQ 12,0 IN-
LBS USING destornillador
```

Que se emplearían para describir la operación de apretar un tornillo, con un determinado "torque" ó par, para el caso del ejemplo planteado anteriormente.

Tanto en RAPT como en AUTOPASS se hace uso de las llamadas Relaciones Espaciales Simbólicas, pero es en el primer lenguaje en el que se lleva su filosofía hasta sus últimas consecuencias. La cual consiste básicamente en:

- 1) Definir un sistema de coordenadas para los objetos y sus características.
- 2) Definir las ecuaciones de los parámetros de configuración del objeto para cada una de las relaciones espaciales entre características.
- 3) Combinar las ecuaciones para cada objeto.
- 4) Resolver las ecuaciones para la configuración de parámetros de cada objeto.



La filosofía subyacente en lenguajes dirigidos a objetos, como RAPT, es conseguir que la persona que programa el robot sea capaz de describir lo que quiere realizar, en términos que sean naturales para él, en lugar de los naturales al robot. Por tanto, en este tipo de lenguajes, él describirá como los diferentes objetos se relacionan unos con otros, en varias situaciones y no necesitará describir explícitamente el camino que el manipulador habrá de seguir.

Los objetos son modelados en términos de las características de su superficie (planos, cilindros, agujeros, esferas, aristas, esquinas y puntos) y de aquí los distintos pasos en un proceso de ensamblado son programados como Relaciones Espaciales, que son creadas para ser sostenidas entre características de los objetos, y como movimientos de objetos relativos a características. Las relaciones espaciales son conceptos como "against", "coplanar", "aligned", "fits", "parallel". Los movimientos pueden ser cualificados —como movimiento perpendicular a cierta superficie, o giro alrededor de un eje—. Ya que la programación se realiza simbólicamente con referencia a características de cuerpos, y no explícitamente usando la geometría real de los cuerpos, es posible escribir programas generales que podrán ser usados en gran variedad de formas y tamaños de los objetos. Por ejemplo, uno puede describir un programa general que permita a un manipulador coger un objeto de cualquier sitio y ponerlo donde sea, en una relación particular con otro objeto. Esta acción de "coge" y "deja", puede generalizarse a una macro generalizada. Esto quiere decir, que un programador puede describir una librería de macros, que pueden ser reutilizadas en una gran variedad de contextos. Una vez que el trabajo duro de describir los objetos, el espacio de trabajo y el manipulador son dados, y una vez que la librería de macros de manejo y ensamblado han sido escritas, la descripción real de una tarea, puede ser muy simple. Ya que las manipulaciones han sido descritas simbólicamente, correcciones, extensiones y revisiones son tarea fácil para el programador.

El esfuerzo investigador realizado en el campo de la robótica para crear modelos computacionales que permitan simular los conceptos geométricos relativos a la planificación del ensamblado, derivan en dos grandes grupos: aquellos que tienen una representación interna del modelo del mundo en el que existen, y los que no la tienen. Actualmente la mayoría de los lenguajes utilizados en la práctica, corresponden a la segunda categoría ("nivel del manipulador"). Por otra parte, se están desarrollando en la actualidad varios sistemas que admiten representación interna, como los ya mencionados RAPT y AUTOPASS.

Una cierta ventaja de RAPT sobre AUTOPASS, es que el primero solo necesita modelos parciales de los cuerpos, de forma que en las descripciones de los objetos solo se describen aquellas características que son relevantes para el manejo y posicionamiento de los objetos. Esto es una ventaja obvia para aquellos cuerpos que poseen una geometría compleja. Por contra, la descripción completa de los objetos puede ser una parte tediosa en la descripción del ensamblado (e.g. descripción de partes en AUTOPASS). Sin embargo tiene la desventaja de que al ser descripciones parciales de los objetos, no permiten la incorporación de aspectos referidos a la planificación de trayectorias libres de colisiones, y comprobación de errores en los planes de ensamblado.

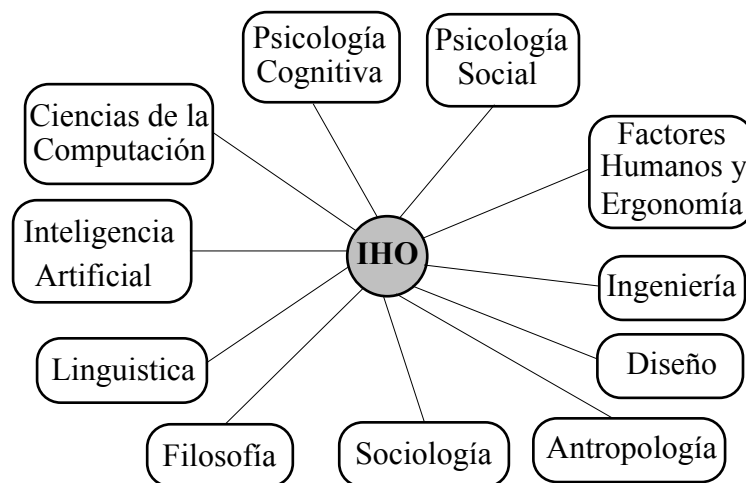
Resumiendo, quedan por resolver muchos problemas, entre los más importantes: la planificación del agarre, la del movimiento y además, la planificación en la recuperación de errores, debidos a imprecisiones en los modelos cinemáticos, etc. que conllevan incertidumbre en el posicionamiento, etc. En la actualidad no existe ningún lenguaje a nivel de tarea que no esté todavía en fase de investigación. Por ejemplo, se está trabajando en la creación de una interfaz entre RAPT y un sistema de modelado sólido. Esto significa que la descripción inicial de los cuerpos será mucho más fácil y que RAPT podrá integrarse en sistemas de fabricación que usen el modelador para descripción de partes y fabricación. Así mismo, se está

incrementando el uso de sensores como visión, o de fuerza, con el objetivo de dotar al robot de la percepción necesaria para evitar errores de posicionamiento y, además, a partir de su integración sensomotora, conseguir cierta capacidad reactiva y autónoma. Dejando a parte consideraciones técnicas, digamos para terminar que, la filosofía que subyace en los lenguajes de programación a nivel de tarea es que sean capaces de acercarse al máximo a la forma de razonar del hombre, y en este sentido el hecho de utilizar relaciones espaciales simbólicas, o mecanismos similares, supone un claro progreso.

#### 4.4 INTERFACES AVANZADAS

El cerebro del robot sabemos que es un ordenador, y por tanto la problemática de la interfaz con el robot está muy ligada a la de aquél.

Durante los años setenta surgió el concepto de "interfaz de usuario", también denominado "interfaz hombre-máquina". Dicho término considera los aspectos cruciales del sistema (ordenador), para el contacto requerido con el usuario, lo cual supone: un lenguaje de entrada para el usuario, un lenguaje de salida para la máquina, y un protocolo para la interacción. Más recientemente, a mediados de los ochenta, aparece el concepto de "interacción hombre ordenador" (IHO), el cual va más allá de la sola consideración del diseño del interfaz, teniendo en cuenta todos aquellos aspectos que intervienen, directa o indirectamente, en la interacción entre usuarios y ordenadores. La complejidad asociada al concepto IHO puede seguirse en la figura 4.3.



**Figura 4.3.** Adaptada de [Preece et al., 94]. Disciplinas que contribuyen a la IHO.

Una de las claves del éxito en la creación de una interfaz concreta, en un determinado contexto y para un tipo específico de usuarios, radica en la generación de una realimentación adecuada, que facilite y potencie la comunicación, en nuestro caso: hombre-robot. Esto pasa por la incorporación de dispositivos de entrada/salida y un determinado estilo de interacción, que aproveche de forma óptima, tanto los recursos del usuario (sus habilidades en cuanto a percepción/comunicación se refiere), como de la máquina: el robot, dotado o no de algún tipo de información sensorial.

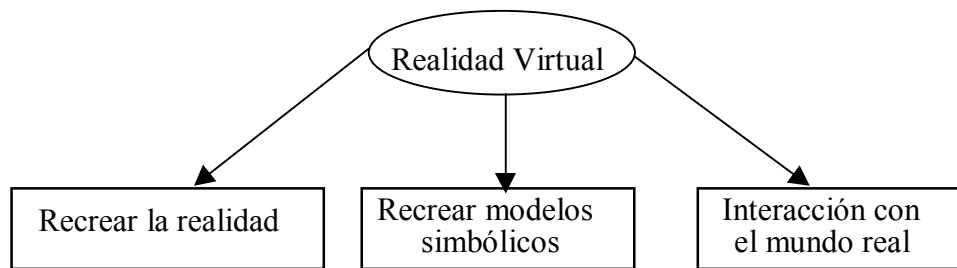
Algunas definiciones útiles son:

- Dispositivo de entrada: Aquél que junto con un software adecuado, transforma información del usuario en datos que una aplicación del ordenador puede procesar.
- Dispositivo de salida: Aquél que facilita información o realimentación en forma perceptible para el hombre.

Si nos restringimos al caso de usuarios sin discapacidades de ningún tipo, parece que un interfaz adecuado, hombre-robot, pasa por el procesamiento del lenguaje natural que, debido a las complejidades que presenta, representa todavía una línea abierta de investigación. No

obstante, una primera aproximación al problema lo constituyen sistemas de reconocimiento del habla, restringidos a una serie determinada de palabras y, normalmente, con entrenamiento previo de cada sujeto que las emite. De hecho, están apareciendo ya algunos paquetes de software comercial que incorporan estas posibilidades.

Otra gran línea de actuación se abre a través de un nuevo concepto denominado "realidad virtual", véase la figura 4.4. Digamos para simplificar que la diferencia crucial entre realidad virtual y síntesis de imagen sin más, radica en la posibilidad de "inmersión" que brinda una interfaz generada con esta nueva tecnología. Lo cuál, obviamente, requiere dispositivos especiales, más o menos sofisticados (cascos, guantes, etc.).



**Figura 4.4.** Posibles dimensiones o ejes de la Realidad Virtual.

El control de robots a distancia constituye el medio privilegiado de intervención en los medios llamados hostiles (interior de una central nuclear, fondos marinos, etc.): es lo que habitualmente se denomina "telecomando". En la actualidad, las técnicas de realidad virtual vienen a completar el arsenal de las que se utilizan en telecomando.

Con las técnicas de telecomando un operador humano situado en una sala de control dirige un robot esclavo, más o menos alejado, según las informaciones tomadas por distintos sensores (cámara, telémetros, sensores de esfuerzo, etc.). Mediante la manipulación de un órgano de generación y de realimentación cinestésica, llamado brazo de control. Las técnicas de realidad virtual permiten, entre otras cosas, preparar por simulación un encadenamiento de las tareas que hay que efectuar antes de realizarlas efectivamente. Puesto que al poner en marcha procesos complejos, las acciones del robot ya están en gran parte preprogramadas, se integran fases automáticas durante las cuales el operador supervisa la acción del robot, y otras en las que el control está compartido. Se habla entonces de "teleasistencia". Este concepto fue introducido a principios de los 80 por los equipos de investigación del programa nacional francés "Automatismo y robótica avanzada" (ARA)-Telemando [André & Fournier, 87]. Con él, se desmarcaban de otros conceptos: el de "telepresencia", explotado principalmente por los japoneses (bajo el que subyace únicamente la inmersión del operador en el entorno esclavo), y el de "telesupervisión" utilizado por los norteamericanos (sólo relacionado con el encadenamiento de acciones automáticas).

Con el nuevo enfoque introducido por la teleasistencia, una misión consiste en una sucesión rápida de secuencias cortas que se componen de tres fases:

- Modelización del robot y de su entorno a partir de los datos recogidos del mundo real.
- Programación de las acciones del robot en el mundo virtual.
- Ejecución de la tarea por el robot real y control de esta ejecución (las ordenes de ejecución se dan a través del robot virtual).

De esta forma se evita que se produzcan diferencias muy grandes entre lo que se está programando y lo que realmente se ejecuta. Incluso mediante la inclusión de sensores de realimentación táctil y cinestésicos, se puede permitir al operador humano experimentar las mismas sensaciones que produce esta realimentación en el mundo virtual.

En concreto, en el contexto de la robótica, las técnicas de realidad virtual son susceptibles de cumplir dos funciones principales:

- *Simulación.* Mediante esta función se ofrece la posibilidad de desconectar el robot distante y sustituirlo por un simulador. Lo que permite programar una tarea, ensayar nuevas técnicas de control, y también entrenar a los operadores. Un simulador de este tipo debe incluir elementos geométricos (forma y configuración del robot), cinemáticos (velocidad del robot), y dinámicos (inercia del robot), con el fin de aproximar todo lo posible el comportamiento del simulador al de un robot real. Se habla en este caso de telecomando virtual.
- *Interfaz.* Un entorno virtual puede servir como interfaz entre el operador y el robot durante la ejecución de una tarea para manipular el robot y supervisar sus acciones.

Resumiendo, en la actualidad los primeros simuladores dinámicos ya permiten simular en tiempo real ciertos tipos de interacción entre un robot distante y su entorno. Sin embargo, estos instrumentos, generalmente muy simples, no pueden todavía pretender simular de manera realista fenómenos físicos. Con el tiempo, el desarrollo de arquitecturas paralelas de más rendimiento y un análisis en profundidad de las necesidades de modelos dinámicos permitirán que en un futuro no lejano puedan realizarse verdaderos telemandos virtuales que ofrecerán al hombre la posibilidad de explorar entornos cada vez más lejanos.

## BIBLIOGRAFÍA

- [Barrientos et al., 97] Barrientos A, Peñin LF, Balaguer C, Aracil R. *Fundamentos de Robótica*. McGraw-Hill. 1997.
- [Dai et al., 92] Dai M, Baylou P, Najim M. *An Efficient Algorithm for Computation of Shape Moments from Run-Length Codes or Chain Codes*. Pattern Recognition, vol. 25(10), pp. 1119-1128. 1992.
- [Davies, 90] Davies ER. *Machine vision: theory, algorithms, practicalities*. Academic Press Ltd. 1990.
- [[Duda & Hart, 73] Duda RO, Hart PE. *Pattern, clasiffication and scene analysis*. John Wiley & Sons. 1973.
- [Ferraté et al., 86] Ferraté G et al. *Robótica Industrial*. Marcombo, S.A. 1986.
- [Freeman, 61] Freeman H. *On the encoding of arbitrary geometric configurations*. IRE Trans. Electronic Computers, vol. EC-10(2): 260-268. 1961.
- [Fu et al., 88] Fu, González & Lee. "Robótica: Control, Detección, Visión e Inteligencia", McGraw Hill Ed. 1988.
- [Gini et al., 85] Gini & Gini. "ROBOTIQUE, contrôle, programmation, interaction avec l'environnement", Ed Masson, Paris. 1985.
- [Gonzalez & Wintz, 87] Gonzalez RC, Wintz P. *Digital Image Processing*. 2ª Ed., Addison-Wesley, Reading, Massachussets. 1987.
- [Groan & Verbeek, 78] Groan FCA, Verbeek PW. *Freeman-code probabilities of object boundary quantized contours*. Comput. Vision, Graphics, Image Processing, vol. 7. pp. 301-402. 1978.
- [Horn, 86] Horn BKP. *Robot Vision*. The MIT Press. 1986.
- [Hu, 62] Hu MK. *Visual Pattern Recognition by Moment Invariants*. IRE Trans. Information Theory, vol. IT-8: 179-87. 1962.
- [Imai & Iri, 86] Imai H, Iri M. *Computational Geometric Methods for Polygonal Approximations of a Curve*. Computer Vision, Graphics and Image Processing, vol. 36: 31-41. 1986.
- [Jain, 89] Jain AK. *Fundamentals of Digital Image Processing*. Prentice Hall, Englewood Cliff. pp. 391-394. 1989.
- [Klafter et al., 89] Klafter, Chmielewski & Negin. "Robotic engineering: an integrated approach". Prentice Hall. 1989.
- [Maravall, 93] Maravall. "Reconocimiento de Formas y Visión Artificial", RA-MA. 1993.
- [McKerrow, 91] McKerrow. "Introduction To Robotics", Addison\_Wesley P.C. 1991.
- [Pal & Pal, 93] Pal NK, Pal SK. *A review on image segmentation techniques*. Pattern Recognition, vol. 26(9): 1277-1294. 1993.
- [Pavlidis, 78] Pavlidis T. *A review of algorithms for shape analysis*. Comput. Graph. Imag. Processing, vol. 7: 243-258. 1978.
- [Pratt, 91] Pratt WK. *Digital Image Processing*. J. Wiley and Sons, New York. 1991.
- [Preece et al., 94] Preece J. et al. *Human-Computer Interaction*. Addison-Wesley, Reading (GB), 1994.
- [Proffit & Rosen, 79] Proffit D, Rosen D. *Metrication errors and coding efficiency of chain-encoding schemes for the representation of lines and edges*. Comp. Graph. Image Processing, 10: 318-32. 1979.
- [Saghri & Freeman, 81] Saghri JA, Freeman H. *Analysis of the Precision of Generalized Chain Codes for the Representation of Planar Curves*. IEEE Trans. Pattern Anal. Mach. Intell. vol. PAMI-3(5), pp. 533-539. 1981.
- [Sanz et al., 94] Sanz PJ, Iñesta JM, Buendia M, Sarti MA. *A Fast and Precise Way for Computation of Moments for Morphometry in Medical Images*. Proceedings of the V Int. Symposium on Biomedical Engineering, pp. 99-100. Santiago de Compostela, Spain, Sept. 1994.
- [UPV, 91] "Apuntes MASTER CAD/CAM", ETSII, U.Politécnica de Valencia. 1991.
- [Zakaria et al., 87] Zakaria MF, Vroomen LI, Zsombor-Murray PJA, van Kessel JMHM. *Fast Algorithm for the Computation of Moment Invariants*. Pattern Recog., 20(6): 639-643. 1987.

## APÉNDICE 1. “La Robótica en el Ciberespacio”

En este anexo se le ofrecen al lector algunas direcciones de Web en relación con el proyecto aquí presentado, las cuales llevan incorporadas además, *links* que le permitirán ir a otras páginas dedicadas a la robótica.

<http://www.frc.ir.cmu.edu/robotics-faq>

Como su nombre indica esta dirección se trata de un FAQ de robótica (*Frequently Asked Questions*), es decir, un documento donde está toda la información en modo preguntas-respuestas sobre los robots, su manejo y elementos necesarios para su construcción.

<http://telerobot.mech.uwa.edu.au/java3.htm>

Esta es una de las páginas más curiosas y a la vez más entretenidas que hay sobre la robótica en Internet. Con nuestro ratón podemos manejar un robot, que se encuentra en Australia, a través del lenguaje Java. Pulsando con el ratón el brazo mecánico y moviéndolo podremos observar como el robot sigue nuestros pasos.

<http://ranier.oact.hq.nasa.gov/telerobotics.html>

Si hay alguna institución que necesite robots es sin duda la NASA. Dentro de su estructura tiene creado un programa donde se estudian las necesidades de los viajes espaciales para crear nuevos ingenios mecánicos que ayuden a las misiones a alcanzar los objetivos.

[http://www.tiac.net/users/jfrancis/dino\\_top.html](http://www.tiac.net/users/jfrancis/dino_top.html)

Presenta un robot realizado por un estudiante. Además de ver las fotografías nos explica cómo lo ha hecho y qué es lo que ha necesitado, así como los requerimientos de tensión y el procesador que controla todas las funciones. También se expone cómo se ha desarrollado la aplicación de control. Es prácticamente una guía de cómo construirse un robot paso a paso.

<http://www.ai.mit.edu/projects>

Uno de los centros más emblemáticos sobre la tecnología como el MIT (Massachusetts Institute Technology) también está presente en el mundo de la robótica. Desde microrobots que podemos cogerlos con la palma de la mano hasta robots que pueden desenvolverse sin problema en cualquier terreno, son mostrados en éstas páginas por sus mismo creadores. En especial, se presenta el proyecto denominado “Cog” sobre un “Robot Humanoide” dirigido por Rodney Brooks, toda una autoridad en el mundo de la robótica.

<http://www.euron.org>

Este es el web-site de la red europea de robótica. Dicha red funciona desde el 2001 como red de excelencia, bajo el V Programa Marco de la Unión Europea, y nuestra Universidad pertenece a ella desde sus inicios.

## APÉNDICE 2. “La función *atan2*”

La función *atan2* fue introducida por primera vez en el contexto de la robótica por Paul [Paul, 81] en relación con el problema del modelado cinemático de los manipuladores. Dichos modelos cinemáticos, en especial el inverso, requieren la evaluación de funciones trascendentes (senos, cosenos, etc.) en sus ecuaciones correspondientes, las cuales pueden dar lugar a imprecisiones e indeterminaciones. La causa principal, tratada en detalle por algunos autores [Mckerrow, 91], reside en el hecho de que al evaluar una determinada orientación,  $\phi$ , en el espacio, se utiliza una función con un único argumento (e.g.  $\cos(-\phi) = \cos(\phi)$ ). Sin embargo, considerando la región:  $-\pi \leq \phi < \pi$ , a partir del conocimiento de  $\sin(\phi)$  y  $\cos(\phi)$ , es obvio que existe una solución única para el valor del ángulo  $\phi$  asociado. Precisamente la función *atan2* provee un medio de expresar dicha solución extendiendo la función  $\text{atan}(\phi)$  a dos argumentos de entrada. En consecuencia, la función *atan2* es una función escalar de dos argumentos escalares tal que [Yoshikawa, 90],

$$\text{atan2}(a, b) = \arg(b + j \cdot a)$$

donde  $a$  y  $b$  son dos números reales,  $j$  es la unidad imaginaria, y  $\arg$  representa el argumento de un número complejo.

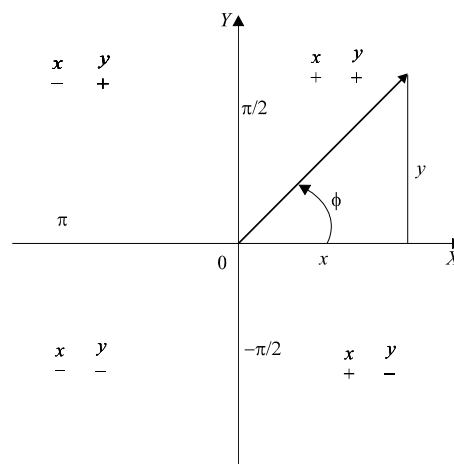


Figura 2.3. Valores posibles del argumento para  $\text{atan2}(y,x)$ .

Es obvio que,  $\text{atan2}(\sin(\phi), \cos(\phi)) = \phi$ , y que  $\text{atan2}(k \cdot \sin(\phi), k \cdot \cos(\phi)) = \phi$  para cualquier escalar  $k$ . Pueden seguirse una serie de propiedades interesantes de esta función en el apéndice 1 de [Yoshikawa, 90] dedicado enteramente a la misma. Esta función devuelve valores angulares  $\phi$  univaluados, tal que  $-\pi \leq \phi < \pi$  (véase la figura 2.3). A partir de las consideraciones anteriores se ha construido el siguiente algoritmo:

```

Función atan2(y,x: Real): Real;
Variables
  aux : Real;
Inicio
  Si (x = 0) ^ (y = 0) Entonces atan2 ← 0
  Sino
    Si (abs(x) < epsilon) Entonces
      Si (y ≥ 0.0) Entonces atan2 ← π / 2.0
      Sino atan2 ← -π / 2.0
    Sino
      aux ← arctan(y/x);
      Si x < 0.0 Entonces
        Si y ≥ 0.0 Entonces aux ← aux + π
        Sino aux ← aux - π;
      FinSi;
      atan2 ← aux
    FinSino;
  Fin;

```