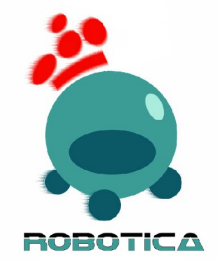
Programación de robots con la plataforma Jderobot

José María Cañas Plaza

http://gsyc.es/jmplaza

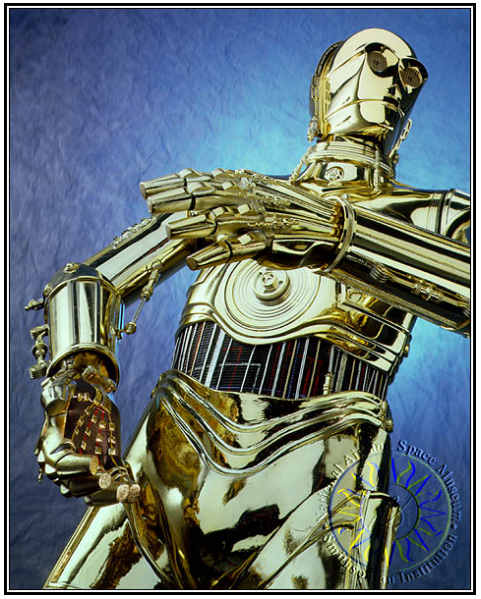


Universidad de Málaga, 17 abril 2009

Contenidos

1. Introducción

Robótica ficción vs Robótica real



Aplicaciones reales

- *Dull, Dirty, Dangerous*
- Industria automovilística: brazos robotizados para pintar, soldar, mover piezas, etc.
- Gestión de almacenes: KIVA, Cofares
- Espacio: Spirit, Opportunity
- Entretenimiento: Aibo, NXT
- Usos militares, desactivación explosivos: PackBot
- Medicina: DaVinci
- Hogar: Roomba
- Prestige, limpieza centrales nucleares

Investigación en robótica

- Generar comportamiento autónomo (inteligencia) en robots móviles
- A más autonomía más aplicaciones
- Multidisciplinar: electrónica, informática, psicología, etología...
- Un robot en cada casa, paralelismo con PC
- Los deseos (y las películas) van por delante de la realidad, pero hay progreso real
- Humanoides
- RoboCup (liga estandard), UrbanChallenge, etc.
- Mapas, localización y navegación
- Interacción con personas
- Prototipos, robustez

¿Qué es un robot?

Sistema informático con:

- Sensores
- Actuadores
- Computador

Hay que programarlo para que consiga sus objetivos y sea sensible a la situación

¿Por qué no tenemos robots que hagan las tareas domésticas?

- Falta flexibilidad
- Tareas complejas
- ¿Problema tecnológico o teórico?
- No es sólo un problema de programación

Arquitectura cognitiva

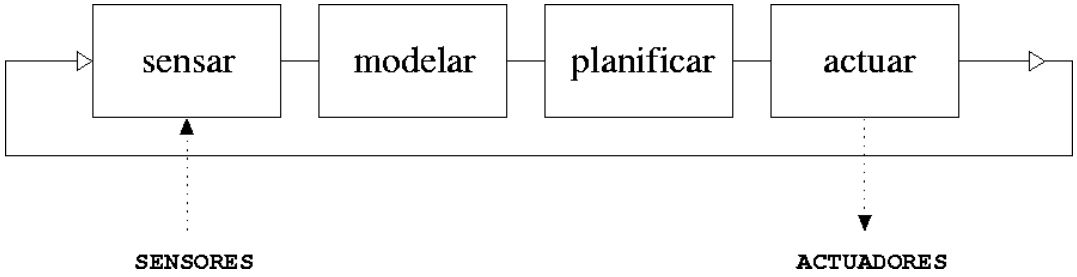
La arquitectura de un robot es la organización de sus capacidades sensoriales, de procesamiento y de acción para conseguir un repertorio de comportamientos inteligentes interactuando con cierto entorno

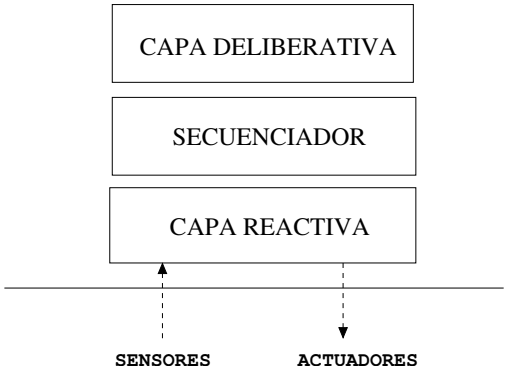
- La arquitectura determina el comportamiento observable
- Un robot móvil es un sistema (muy) complejo
- Capacidades perceptivas, capacidades de actuación
- ¿Cuándo?
- Para comportamientos sencillos, casi cualquier organización vale

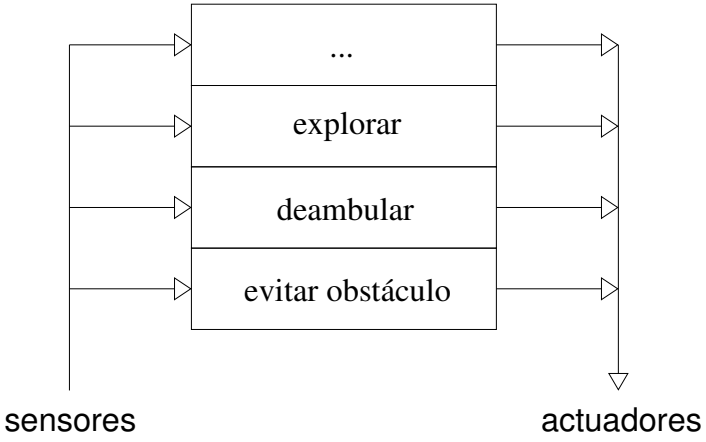
Uno, varios, muchos

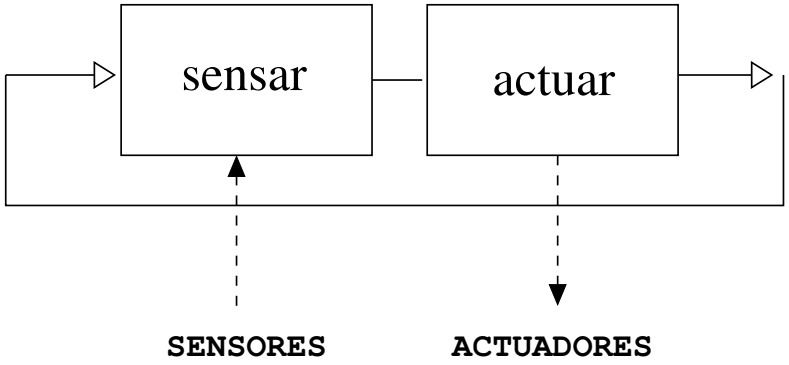
- Termostato, Roomba
- Repertorio de comportamientos
- Del cómo al cuándo
- Selección de acción
- Información desbordante, incierta
- Atención
- Visión computacional es complicada y potente

Paradigmas cognitivos









Paradigmas deliberativo, reactivo, basado en comportamientos, híbrido

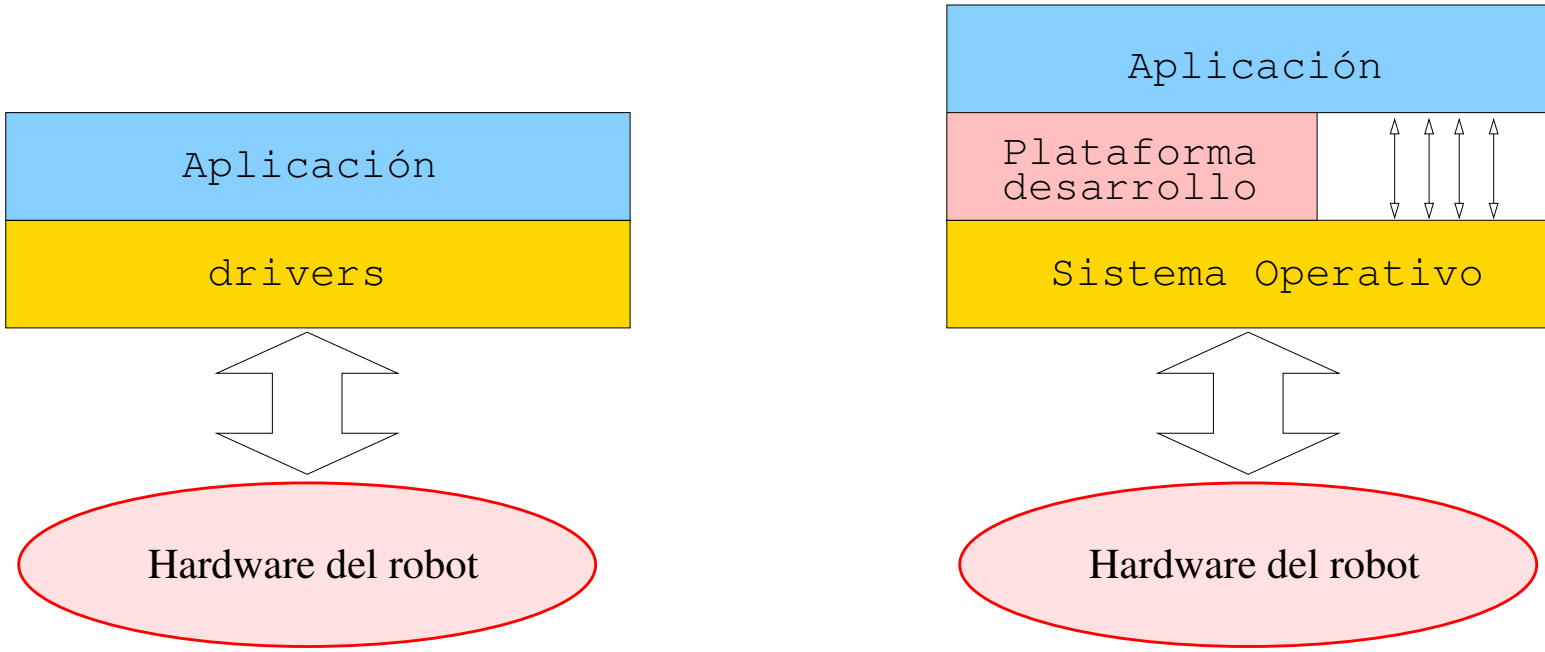
2. Software para robots

- La inteligencia de un robot reside en su software
- No hay una manera universalmente aceptada de programarlos
- Lenguajes: ensamblador, C, C++
- Heterogeneidad
  - Dispositivos hardware
  - Encapsular funcionalidad
- Empezar de cero con cada robot
- Requisitos específicos
- Sistemas operativos y plataformas
- Simuladores

Requisitos específicos

- Vivacidad, agilidad
- Multitarea
- Distribuido, comunicaciones
- Interfaz gráfica, depuración
- Expandible

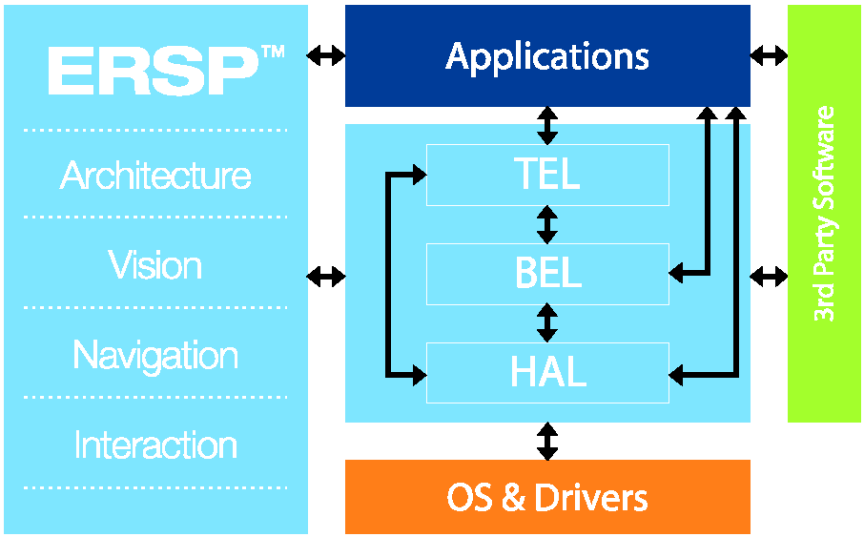
Sistemas operativos y plataformas



- Procesadores empotrados (robots pequeños) o PC (medianos-grandes).
- Sistemas operativos: dedicados o generalistas
- *Middleware* para simplificar la creación de aplicaciones robóticas

¿Qué proporciona una plataforma sw para robots?

- Abstracción del hardware (HAL)
- Arquitectura software
- Funcionalidades de uso común
- Arquitectura cognitiva

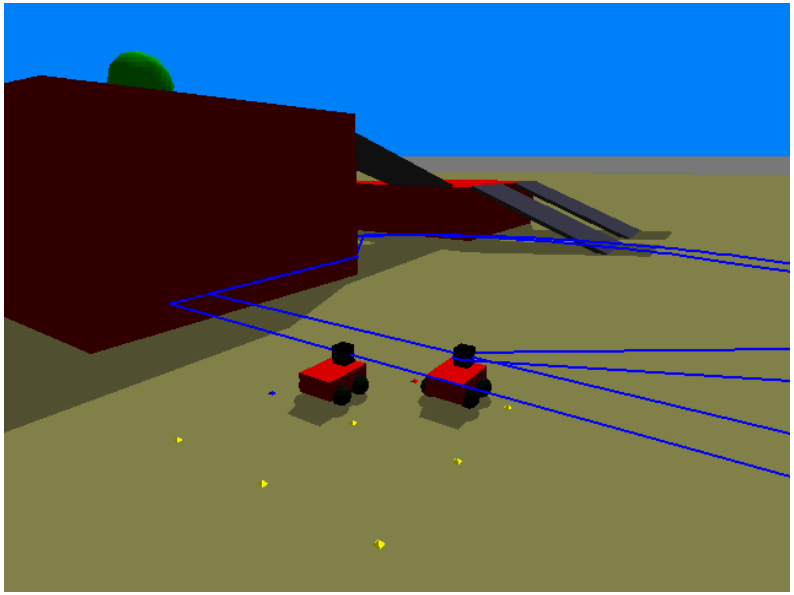


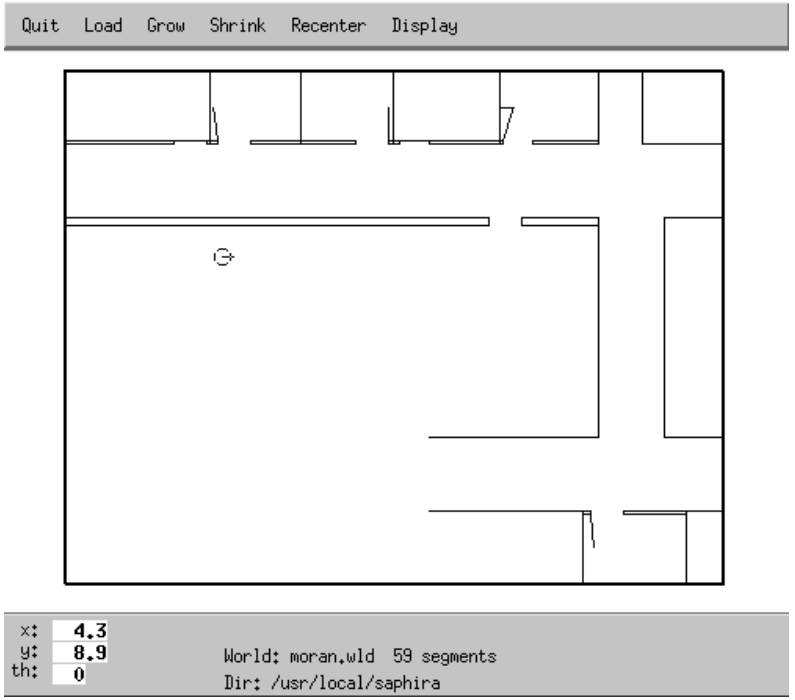
- Comerciales, investigación, software libre
- Ingeniería software: orientación a objetos, distribución
- Carmen, OROCOS, ERSP, Player/Stage, Miro, Orca, etc.

Simuladores

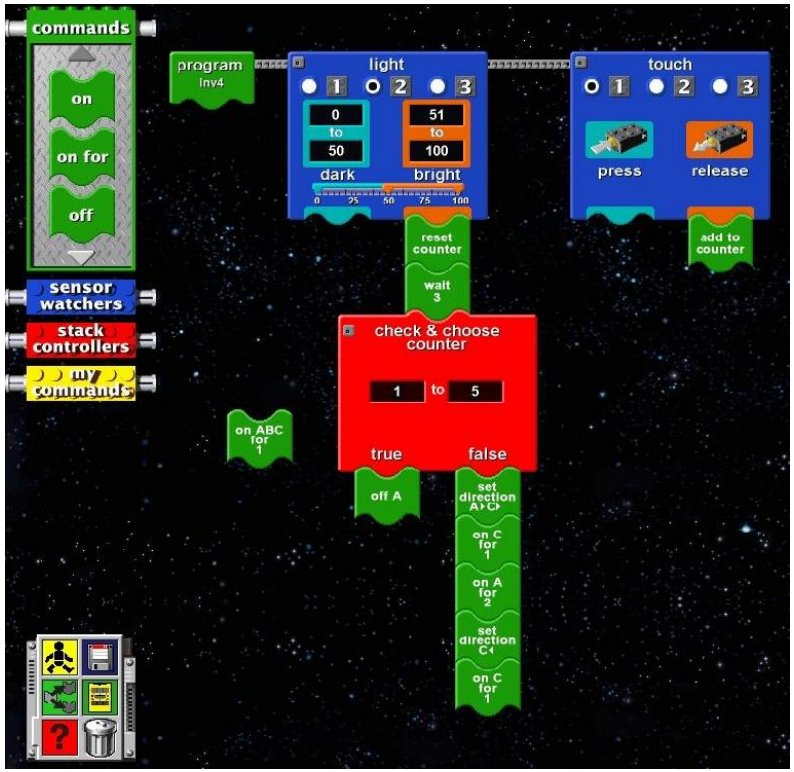
- Madurar algoritmos
- Comodidad trabajar sin robot
- Las caídas no duelen
- Mundo, sensores y actuadores
- OpenGl (OGRE) para imágenes
- Motor físico: ODE (*Open Dynamics Engine*)

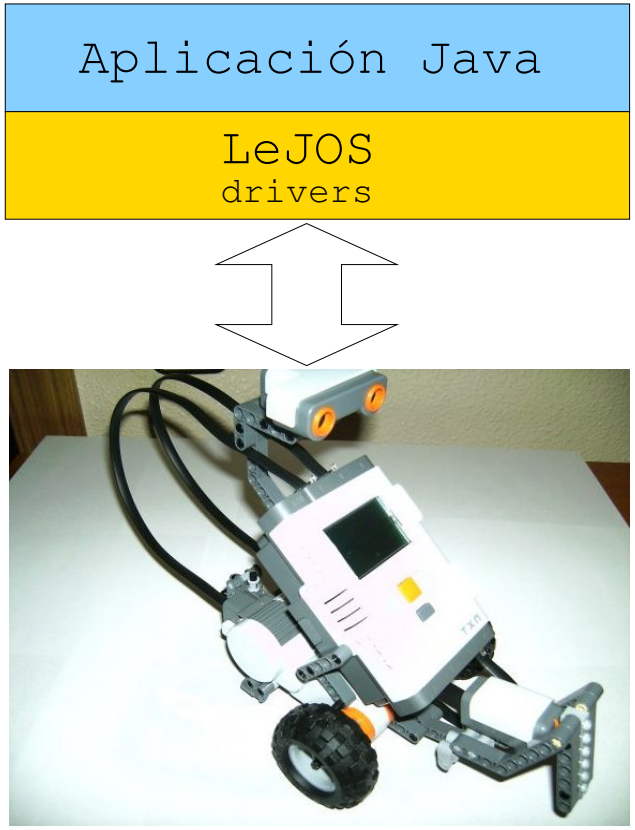
- Gazebo, Stage, Webots, Microsoft Robotics Studio




Quit
Load
Grow
Shrink
Recenter
Display
x: 4.3
y: 8.9
th: 0
World: moran.wld 59 segments
Dir: /usr/local/saphira

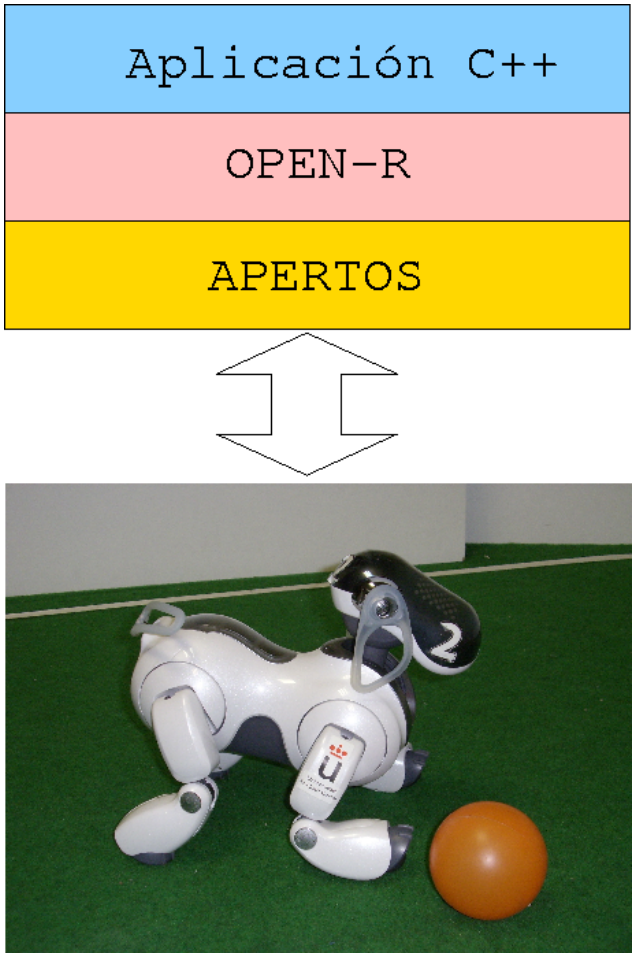
Programación del robot LEGO NXT



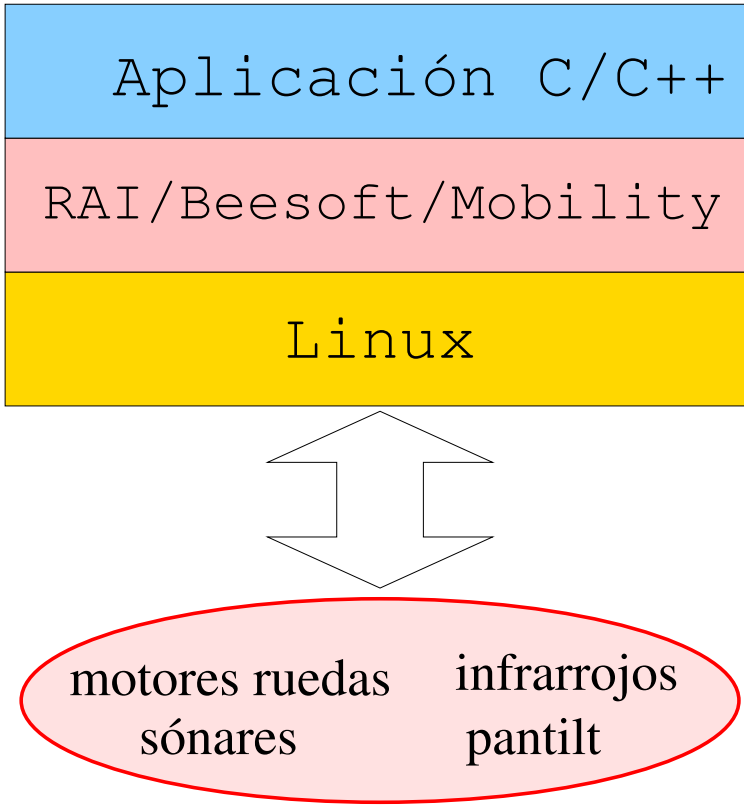


Programación del robot Aibo

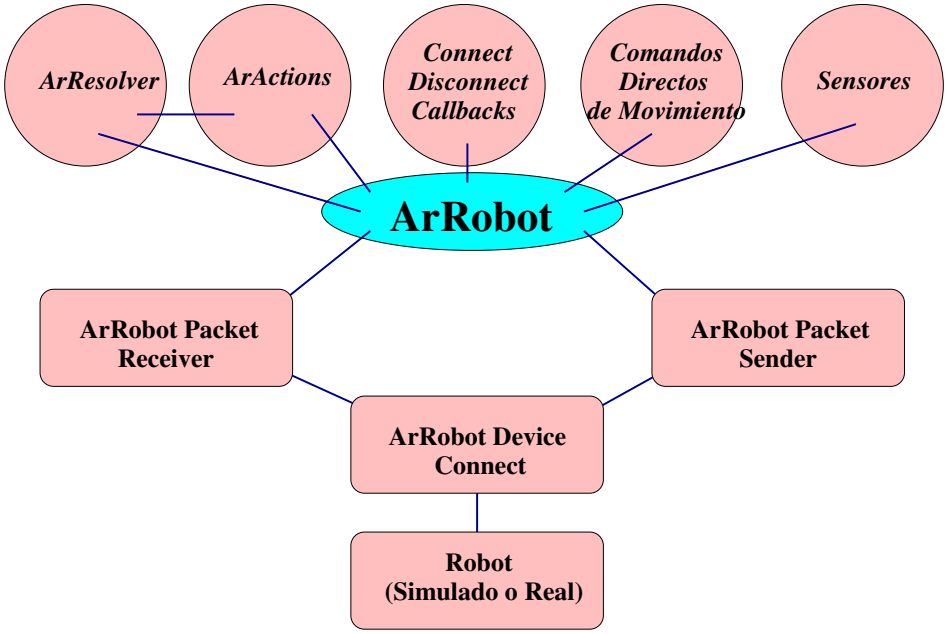
- MIPS 450Mhz
- Objetos monohilo
- Se comunican vía mensajes
- Objetos básicos
  - OVirtualRobotComm: Effector, Sensor, OFbkImageSensor
  - OVirtualAudioComm: Mic, Speaker
  - ANT Aibo Network Tool

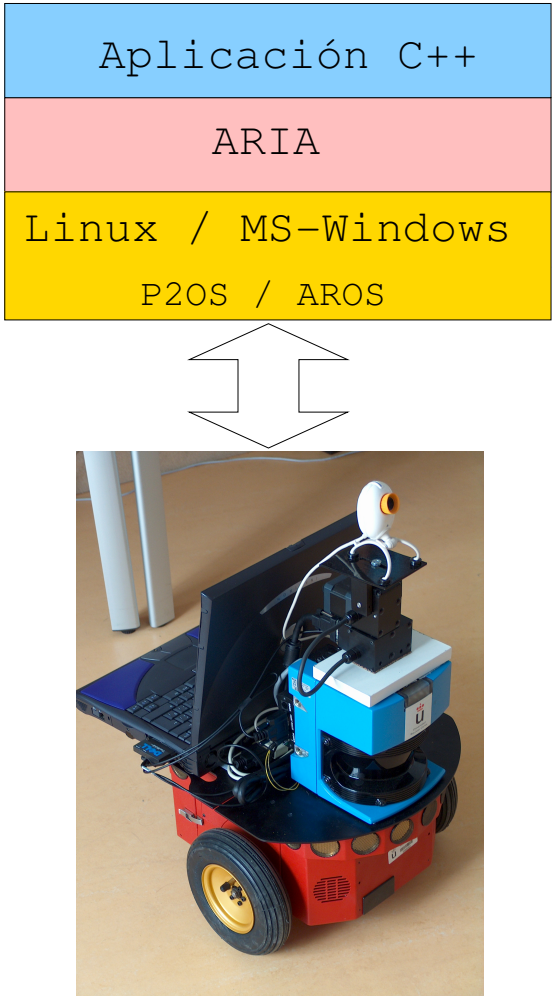


Programación del robot B21



Programación del robot Pioneer

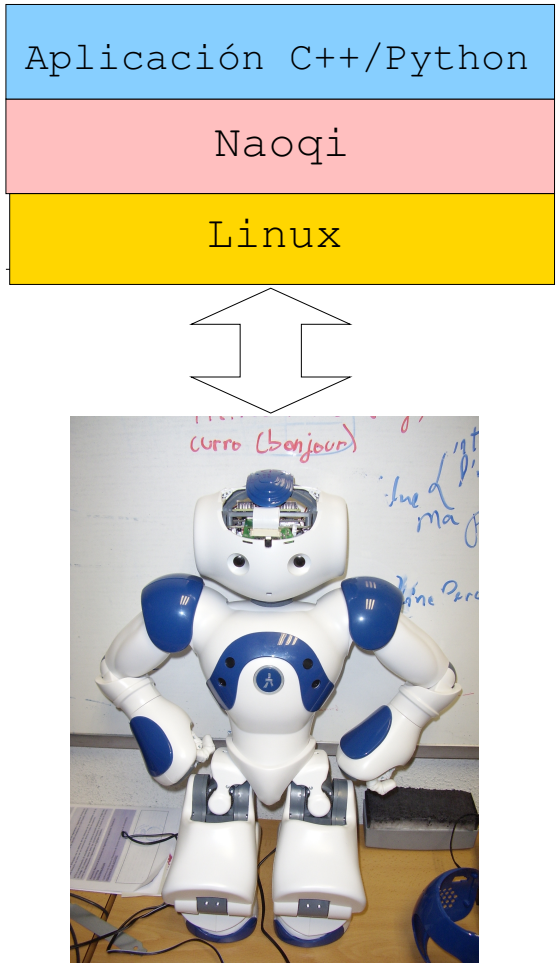




- Acceso a los sensores y actuadores
- Librerías, objetos
- ActivMedia: Saphira, ARIA (C++)

Programación del robot Nao

- AMD Geode 500Mhz
- Objetos distribuidos, invocación de métodos remotos
  - ALMotion
  - ALVision
  - ...
- Módulos, *brokers*



3. Plataforma software Jderobot

Middleware para la programación de aplicaciones robóticas, domóticas y de visión computacional

- Implementación de la arquitectura cognitiva JDE
- Nace con una tesis doctoral y la mantiene un grupo de desarrolladores
- Prácticas de varias asignaturas de robótica en la URJC
- Aplicaciones robóticas (guiderobot, PFCs) y de visión computacional (eldercare, carspeed)
- Licencia GPLv3, paquete debian

1. Arquitectura cognitiva
2. Modelo de programación
3. Acceso a los dispositivos
4. Servicios
5. Bibliotecas
6. Proyecto de software libre

Arquitectura cognitiva JDE

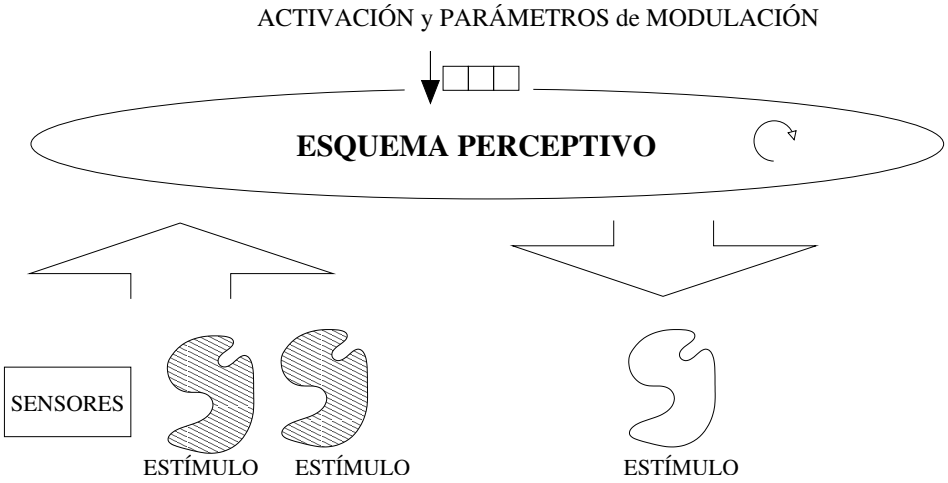
- Comportamiento = percepción y control
- Fragmentación en unidades (esquemas) asíncronas concurrentes
  - de percepción elaboran estímulos
  - de actuación toman decisiones
- La colección de esquemas se organiza en jerarquía dinámica (JDE)
- Sigue el paradigma basado en comportamientos

Esquemas

Un esquema es un flujo de ejecución independiente, con un objetivo

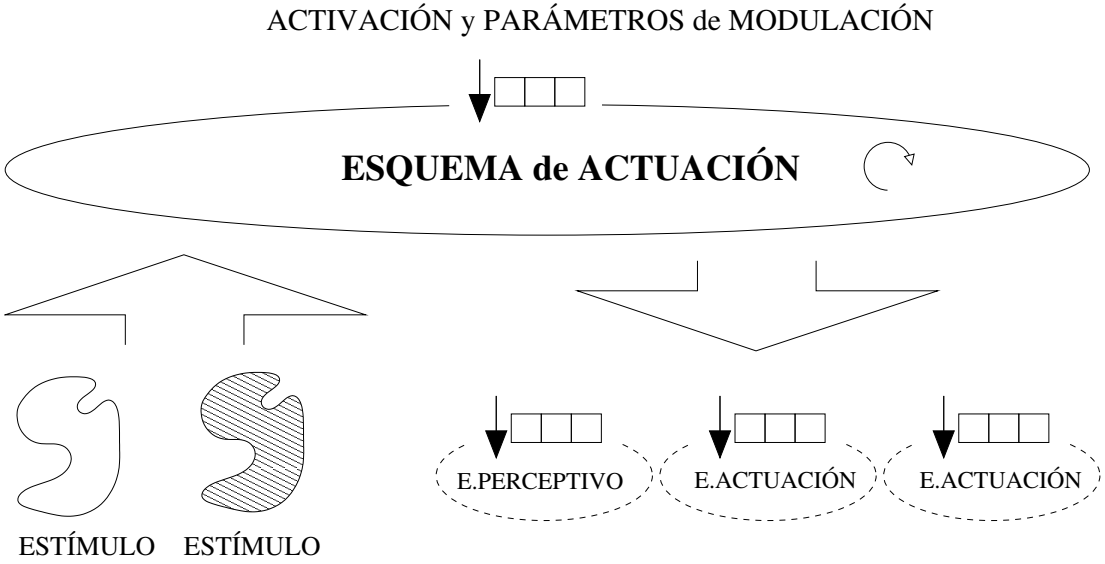
- Funcionamiento continuo
- Se puede activar y desactivar a voluntad
- Modulable a través de parámetros
- Esquemas perceptivos y de actuación
- Su funcionalidad se usa despertándolo y modulándolo

Esquemas perceptivos



- Un estímulo es *una pieza de información que al menos un esquema de actuación necesita para tomar sus decisiones*
- Producen estímulos y los mantienen actualizados (anclados)
- Lecturas sensoriales, transformaciones más elaboradas
- Estados: DORMIDO o ACTIVO

Esquemas de actuación



- Toman decisiones de actuación para conseguir o mantener un objetivo
- Su salida pueden ser comandos a los actuadores o la activación y modulación de otros esquemas hijos
- Estados: DORMIDO, ALERTA, PREPARADO o ACTIVO

¿Qué software hay en jderobot?

- Esquemas
- Drivers
- Servicios
- *Plugins* que se cargan (o no) en una ejecución concreta
- Diseño modular
- Bibliotecas auxiliares

Modelo de programación: esquemas

- Cada esquema se materializa en un componente software
- Los componentes son la unidad básica de la aplicación programada
- Cada esquema se compila por separado, es un *plugin*
- Cada esquema se implementa como una hebra (`pthreads`)
- El funcionamiento continuo como ejecución iterativa
- Iteraciones periódicas a cierto ritmo (motor temporal), para percibir o para actuar

- Interfaz funcional para activar y desactivar a otros esquemas
- Los estímulos perceptivos se implementan como variables compartidas
- Los parámetros de modulación son variables compartidas
- Comunicación con otros componentes por memoria compartida, variables tipadas en interfaces de datos explícitos
- Una aplicación robótica es un conjunto de esquemas concurrentes
- Varias hebras en la misma máquina
- Lenguajes C, C++, (python)

Interfaz funcional de un esquema

- Permiten a otros esquemas activar la ejecución de otros o detenerla
- schema_init, schema_terminate
- schema_run, schema_stop
- schema_iteration
- schema_show, schema_hide

Motor temporal de un esquema

- schema_cycle
- gettimeofday
- schema_iteration
- usleep
- conditional_wait

Interfaz de datos de un esquema

- Comunicación entre componentes, conexión con otros esquemas
- Interfaces fijan la estructura de los datos que un esquema publica o necesita
- Importación y exportación de variables
  - Importa las variables que necesita
  - Exporta las variables que produce
  - Censo: qué esquema exporta qué interfaces (nombre)
- Leer y escribir variables
- Cada esquema mantiene su espacio de nombres

Interfaz gráfico de un esquema

- Es opcional
- Activable y desactivable a voluntad, cuando interesa
- Visualización, interacción
- Depuración, útil en desarrollo
- Modular: cada esquema tiene su GUI
- Cada esquema incluye el código de su interfaz gráfica
- Ejecución centralizada (conexión con XWindow)
- GTK, XForms

Acceso a los dispositivos

- El acceso al hardware se ofrece con *drivers*
- Los drivers son plugins que cargan uno o varios esquemas básicos
- Los esquemas básicos proporcionan interfaces de datos para sensores y actuadores
- Los esquemas básicos no tienen GUI
- El mismo interfaz de datos lo puede proporcionar varios drivers
- Mismo programa funciona sobre robot real o simulador.
- Independencia de la fuente de imágenes

Servicios

- Shell para activar/desactivar esquemas e interfaces gráficas desde consola de texto
- MasterGUI para activar/desactivar esquemas e interfaces gráficas visualmente
- Graphics_gtk, glade
- Graphics_xforms
- Networkserver y networkclient
  - Un jderobot como proxy para otro
  - Protocolo de comunicación: suscripción, comandos, imágenes (llega una y pide la siguiente)
- Webservice

MasterGui

- Permite activar/desactivar esquemas y su interfaz gráfico
- Muestra el ritmo de cada esquema

Consola de texto

- Permite activar/desactivar esquemas y su interfaz gráfico
- Muestra el ritmo de cada esquema
- ls, ps, quit, help
- show, hide
- run, stop

Fichero de configuración

- Fichero de configuración (jde.conf):
- Cada esquema o driver puede tener su propio fichero de configuración o empotrar su configuración en el fichero general
- Cuál es la fuente de cada dispositivo (local o remota)
- Qué drivers se van a usar, y qué esquemas básicos carga cada uno
- Qué esquemas del usuario se van a cargar
- Cuáles arrancar al principio

Bibliotecas

- Colorspaces: espacios de color para imágenes
- Fuzzylib: control borroso
- Progeo: relacionar las imágenes con 3D, cámaras calibradas

Jderobot, proyecto de software libre

- 44000 líneas de código
- Comunidad de usuarios y desarrolladores
- ¿Dónde conseguirla? ¿Cómo preguntar dudas?
- Página web: jde.gsyc.es
- Listas de correo: jde-users@gsyc.es y jde-developers@gsyc.es
- Svn, trac, blog, mediawiki
- Paquete debian: `apt-get install jderobot`

¿Por qué software libre en robótica?

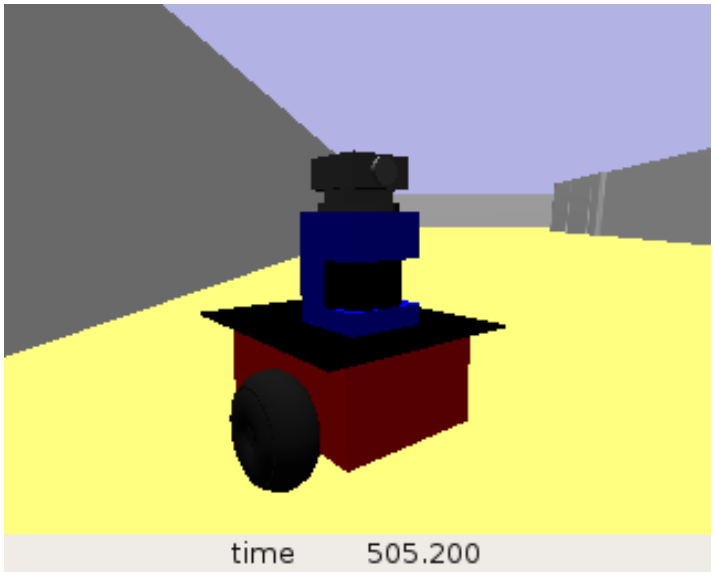
- Independencia de fabricantes
- Varios modelos de robots
- Libertad para modificar código fuente
- Aumentar la calidad del software
- "Devolver el favor": Player, Stage, Gazebo, Opencv, GTK,...

4. Dispositivos soportados

- Soporte de muchos sensores y actuadores
- Player
- Robot Pioneer
- Laser: Hokuyo, Sick
- Cuellos mecánicos: Directed Perception, Sony Evi
- Simuladores: Stage, Gazebo
- Cámaras: usb, firewire, digitalizadoras, videos
- Wiimote
- Robot Nao

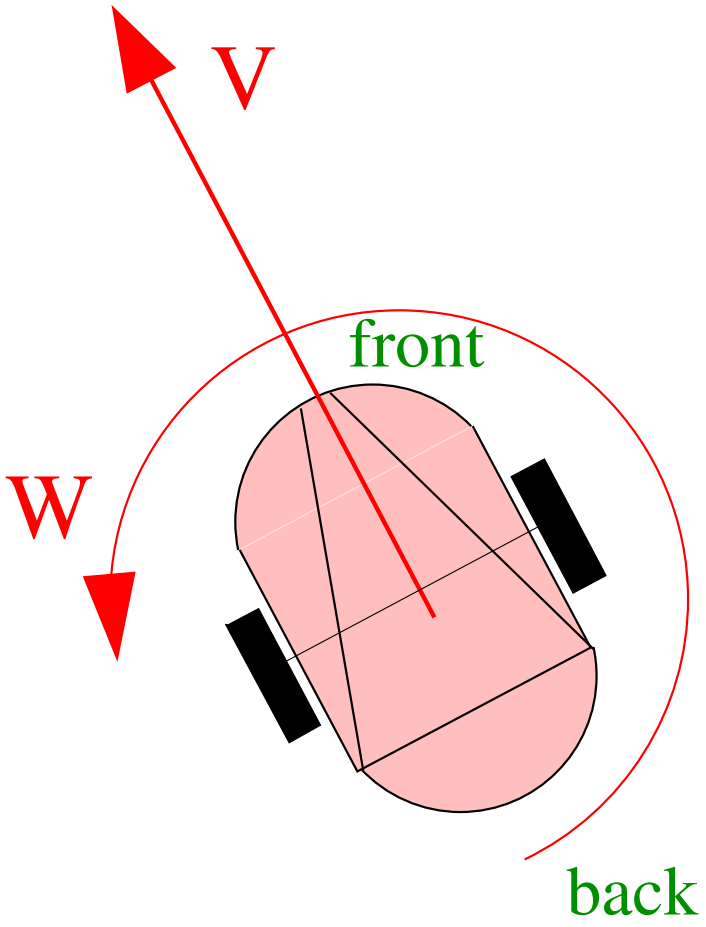
Interfaces para el robot Pioneer

- Base: motores, sónares, odometría
- Drivers: player, gazebo



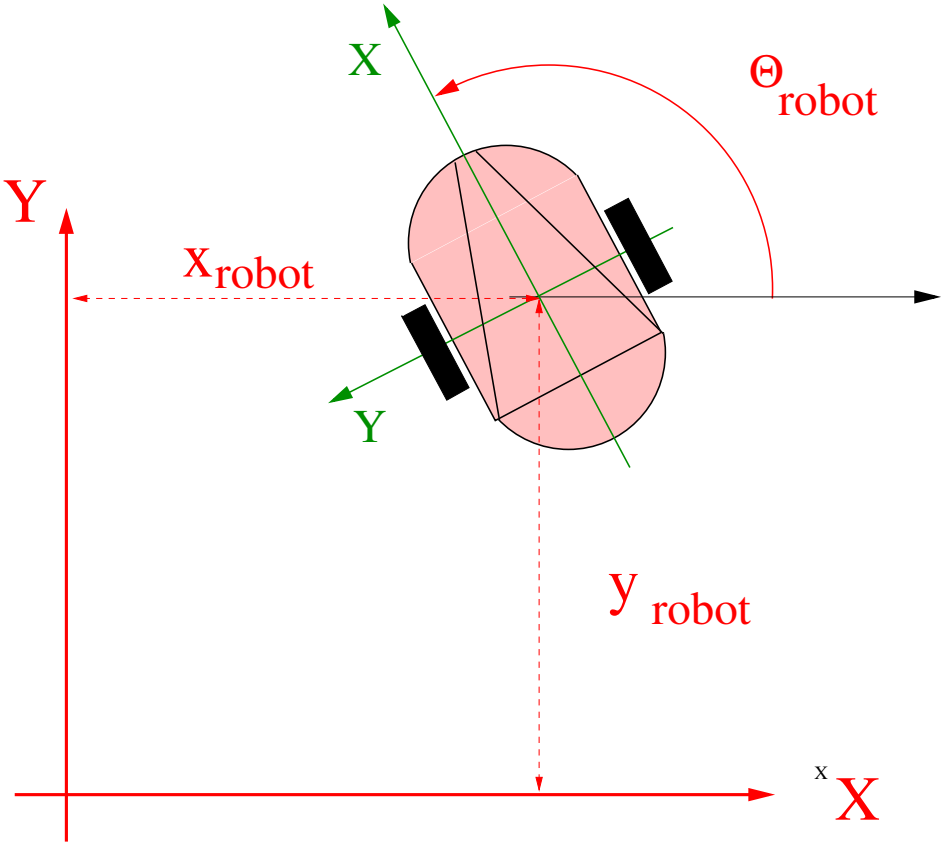
Interfaz para motores

- Esquema `motors`
- Datos: v,w
- Drivers player, gazebo



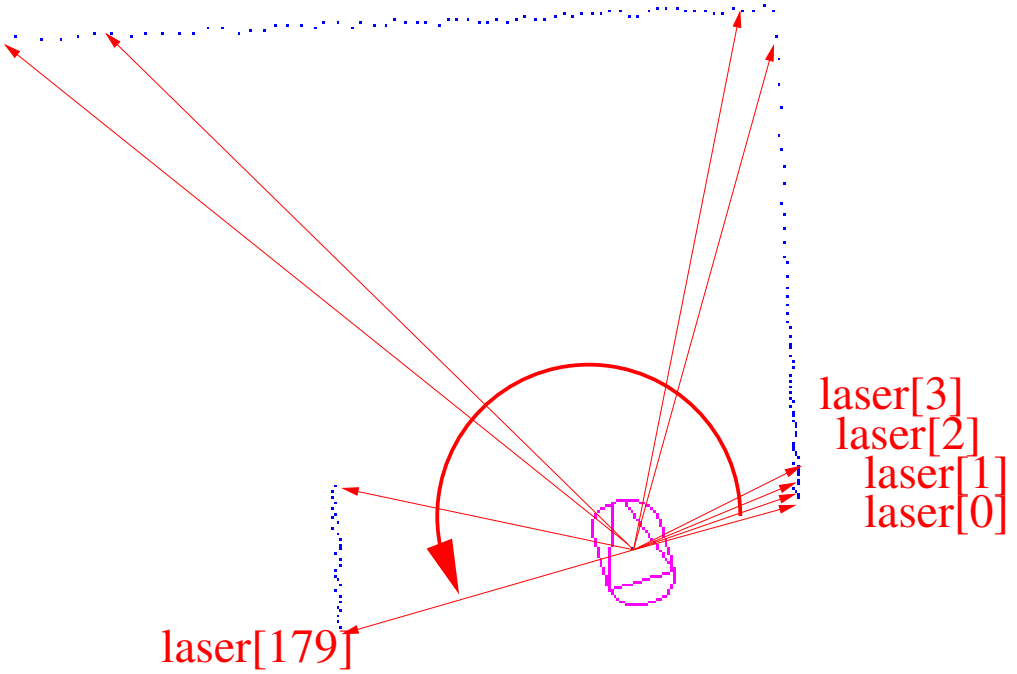
Interfaz de posición

- Esquema `encoders`
- Datos: x, y, theta (mm,radianes)
- Drivers `player` y `gazebo`



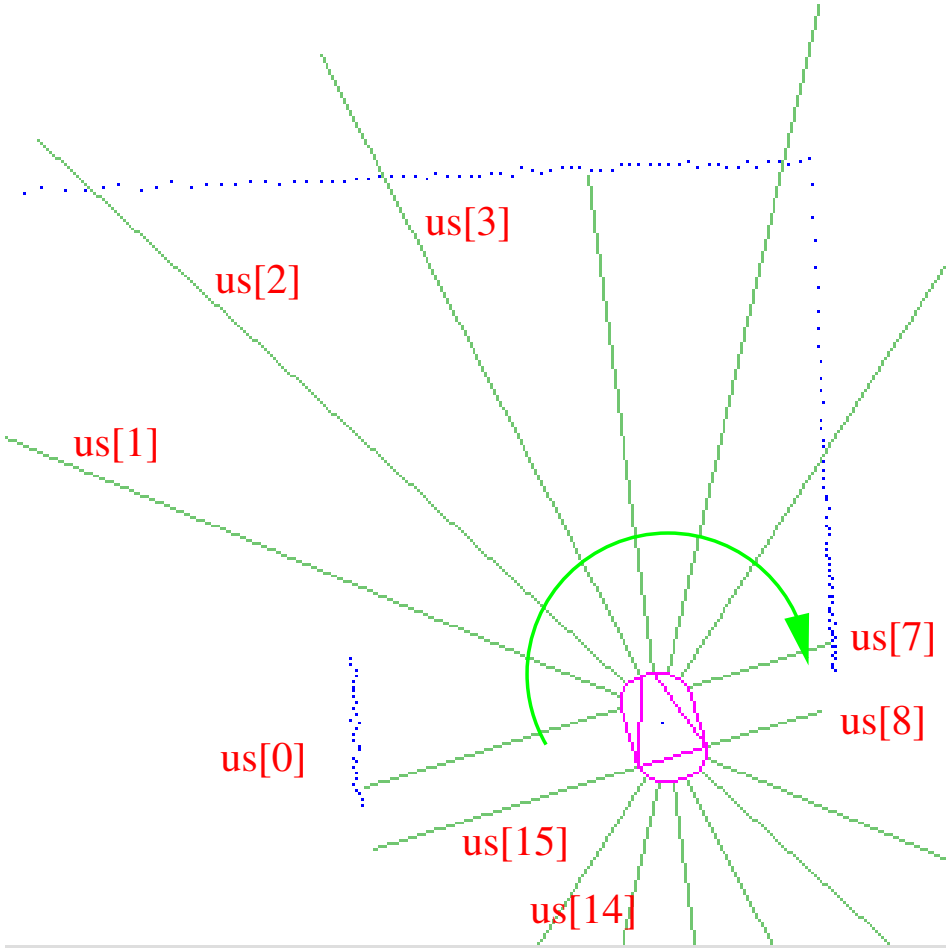
Interfaz para laser

- Esquema `laser`
- Datos: laser
- Drivers `player` y `gazebo`



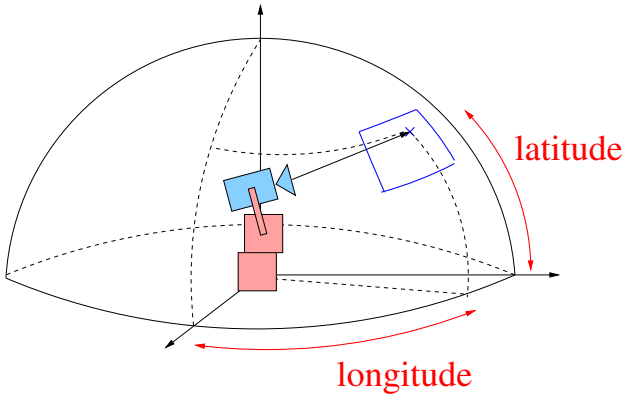
Interfaz para ultrasonidos

- Esquema `sonars`
- Datos: us
- Drivers `player` y `gazebo`

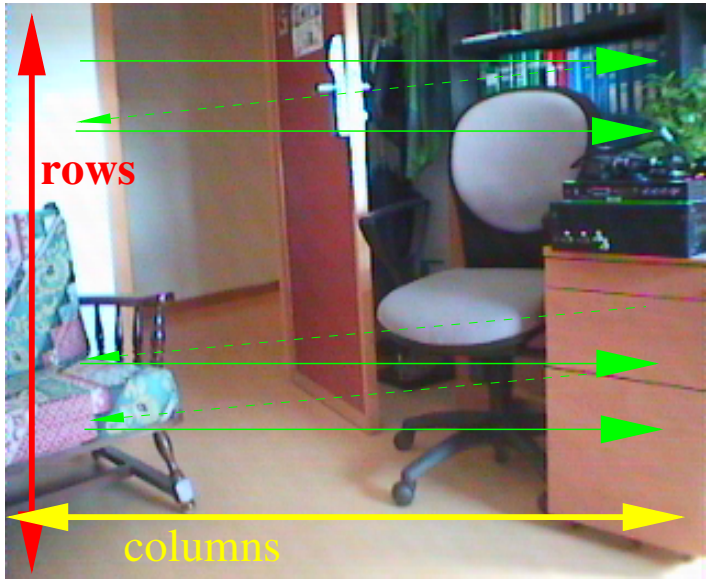


Interfaces para el cuello mecánico

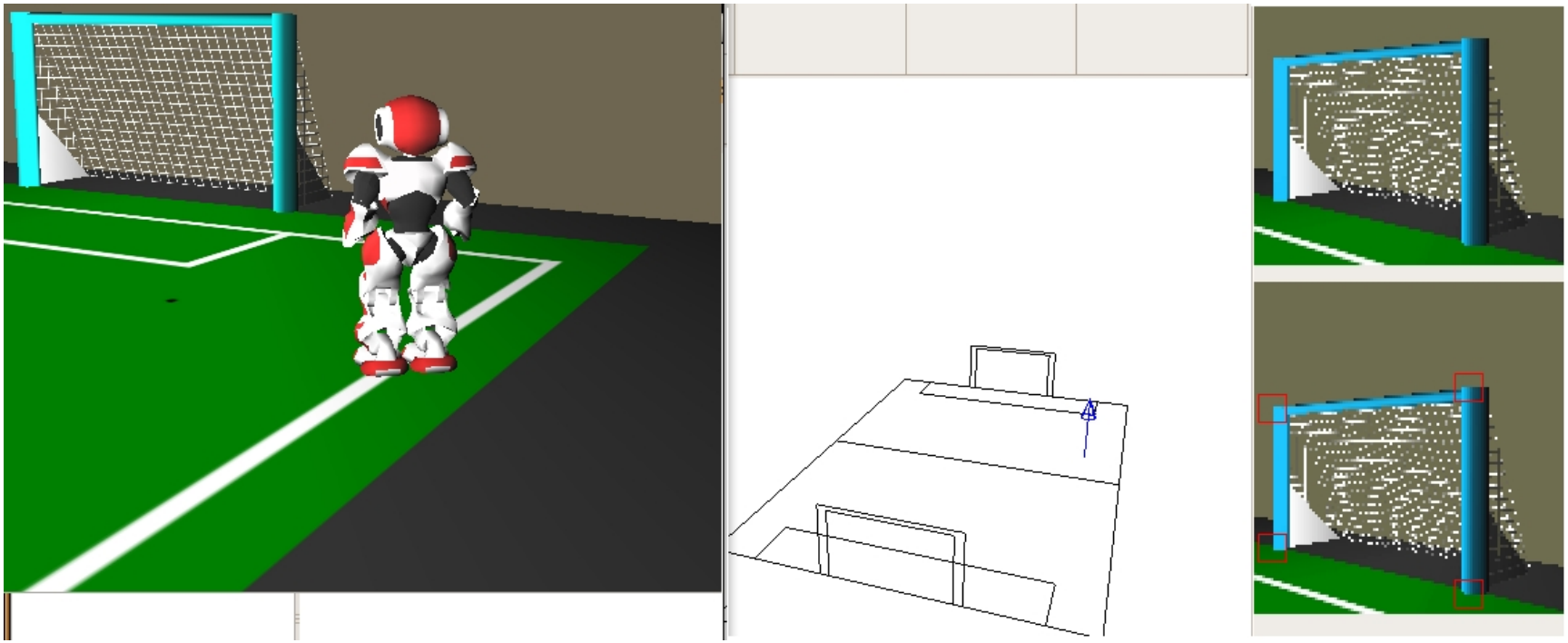
- Drivers `pantilt` y `evi`
- Esquema pantiltencoders:
  pan, tilt
- Esquema pantiltmotors:
  latitude, longitude



Interfaces para imágenes



- Cámaras webcam USB, firewire, en red, digitalizadoras, ficheros
- Esquemas varcolorA, varcolorB, varcolorC, varcolorD
- Datos: varcolor
- Drivers: imagefile, mplayer, firewire, gazebo, v4l, video4linux2, naoqi



Interfaces para wiimote

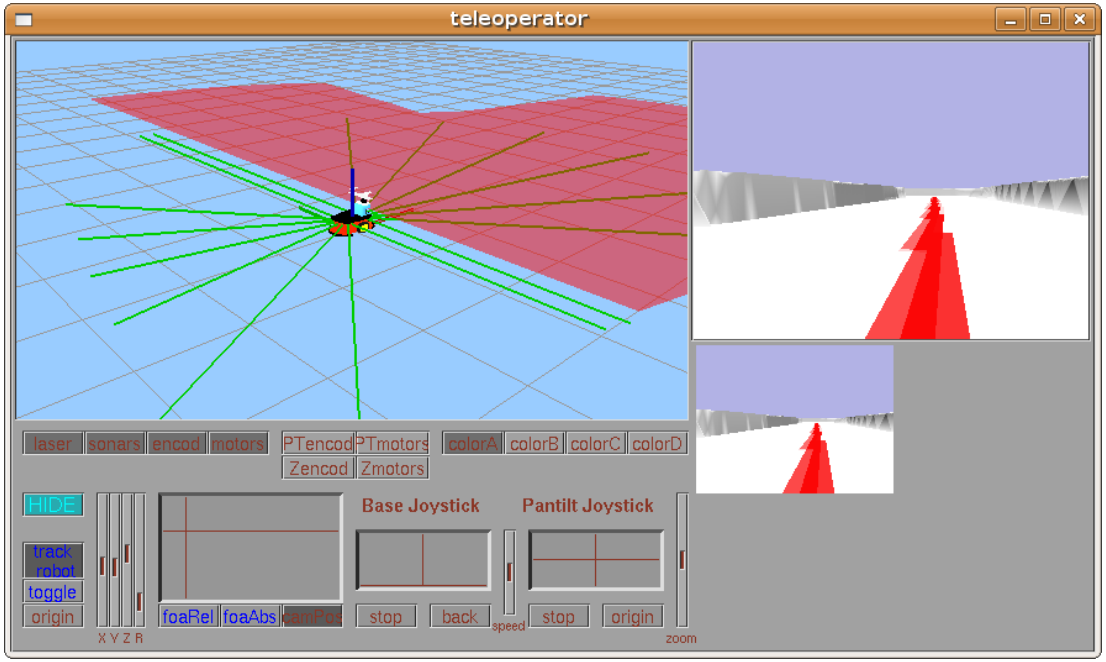
- Esquemas: buttons, accel, ir, nunchuk
- Datos: buttons, accel, ir, nunchuk
- Driver wiimote

5. Componentes desarrollados

- Aplicaciones y herramientas
- Conjunto de esquemas
- Materializan las capacidades perceptivas y de actuación
- Prácticas de robótica
- Investigación en robótica
- Aplicaciones domóticas
- Aplicaciones visión computacional

Teleoperator

- Visualiza los sensores, cámaras, etc.
- Permite teleoperar los motores (base y cuello)

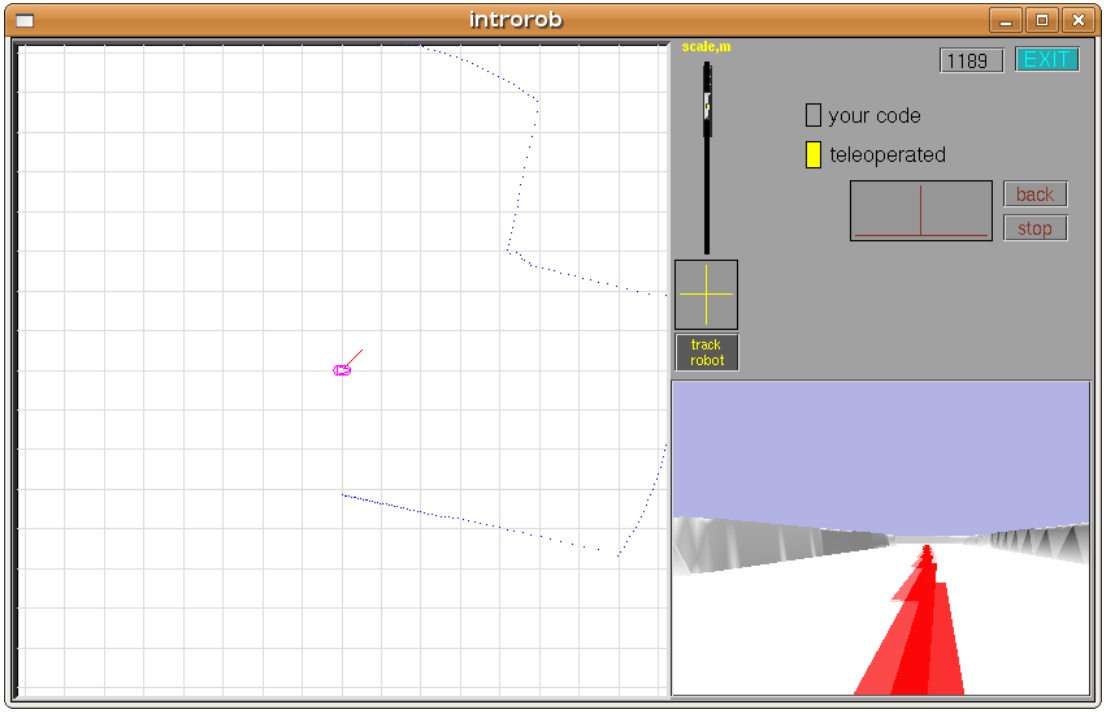


Prácticas de robótica

- Introrob
- Choca-gira, control reactivo, autómata
- Navegación VFF
- Carreras, navegación híbrida
- Sigue líneas visual
- El ratón y el gato

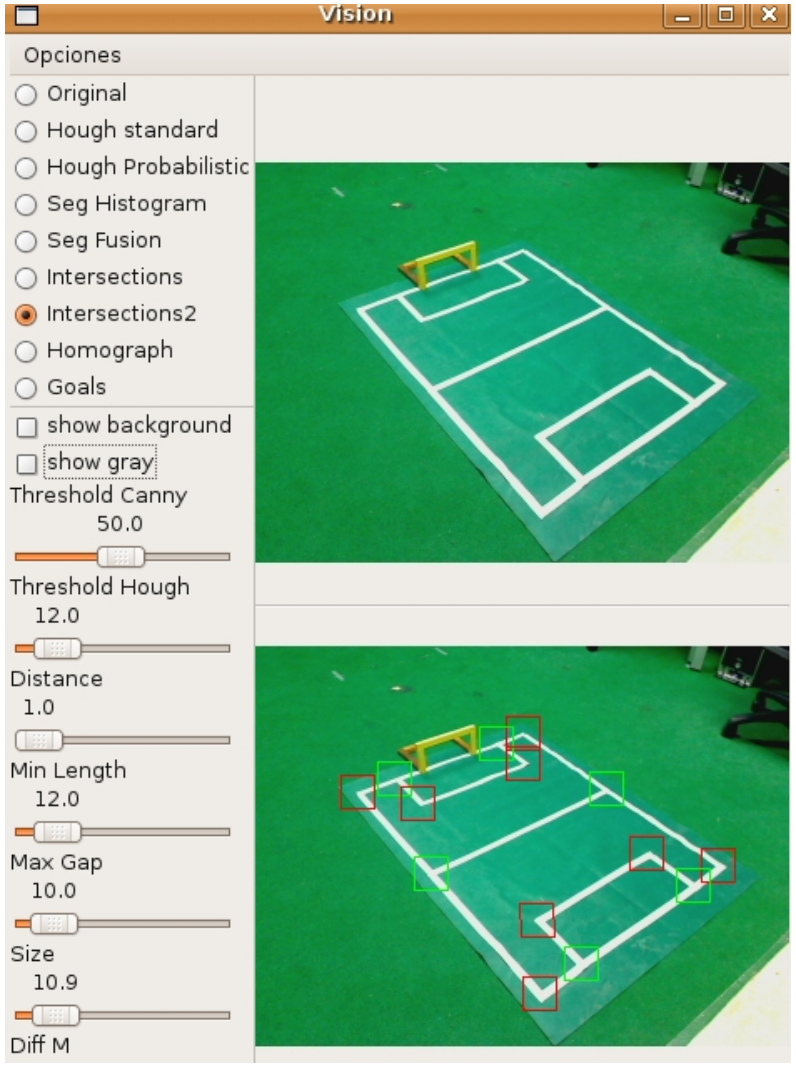
Introrob

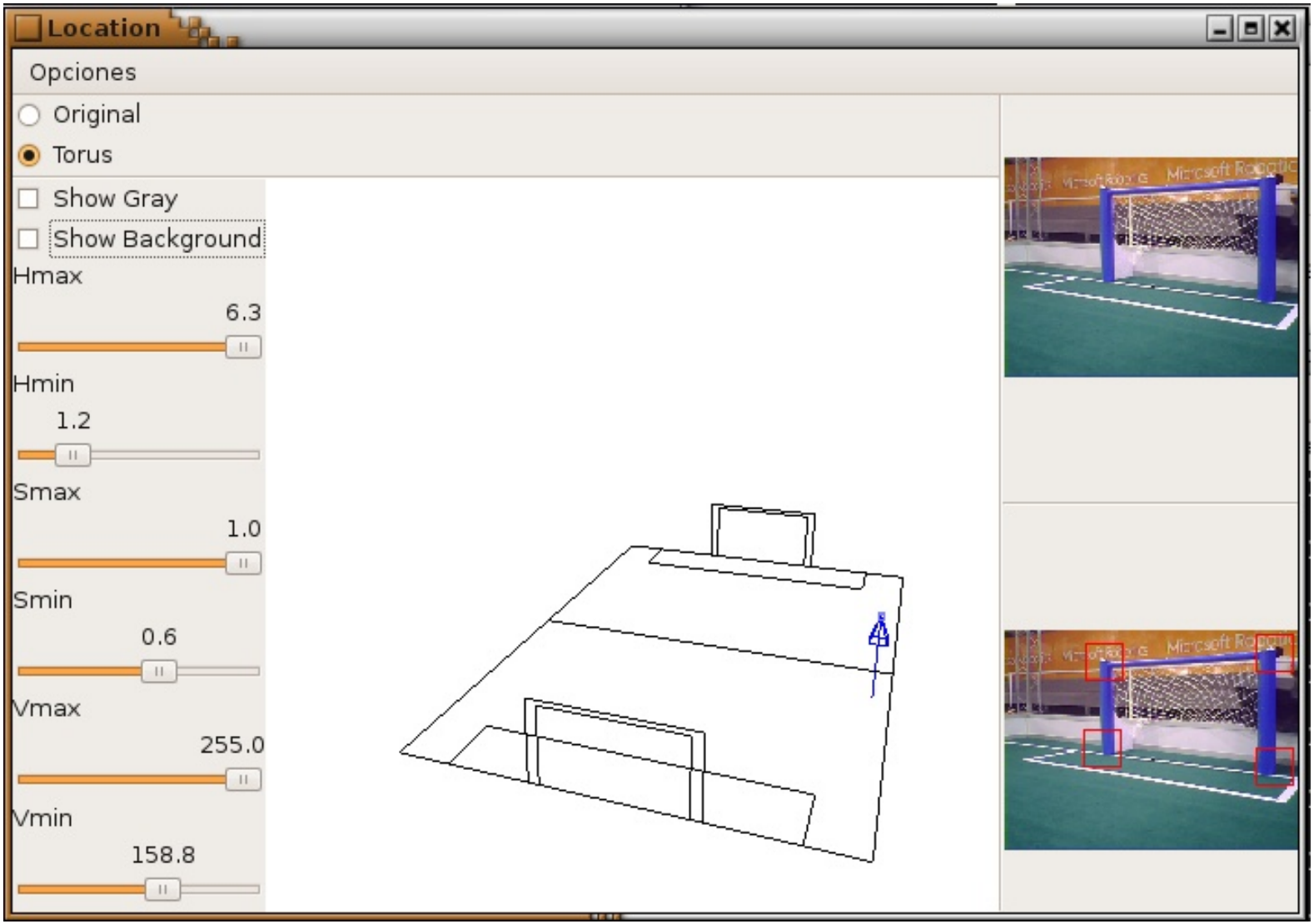
- Pioneer con cámara, láser, cuello mecánico, us, odometría
- Prácticas de robótica



Investigación en robótica

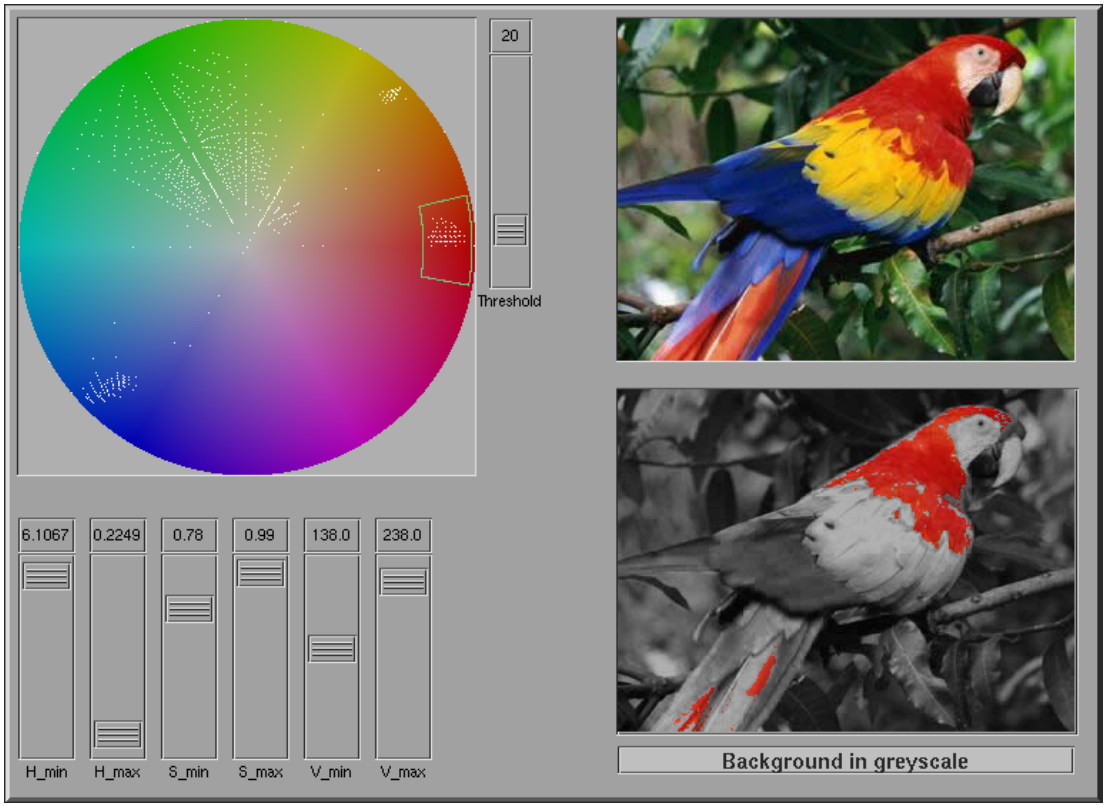
- Taxias: FollowBall
- Seguir a una persona
- Buscar y seguir a un congénere
- Navegación local VFF
- Navegación global GPP
- Autolocalización


Vision
Opciones
Original
Hough standard
Hough Probabilistic
Seg Histogram
Seg Fusion
Intersections
Intersections2
Homograph
Goals
show background
show gray
Threshold Canny
50.0
Threshold Hough
12.0
Distance
1.0
Min Length
12.0
Max Gap
10.0
Size
10.9
Diff M


Location
Opciones
Original
Torus
Show Gray
Show Background
Hmax
6.3
Hmin
1.2
Smax
1.0
Smin
0.6
Vmax
255.0
Vmin
158.8

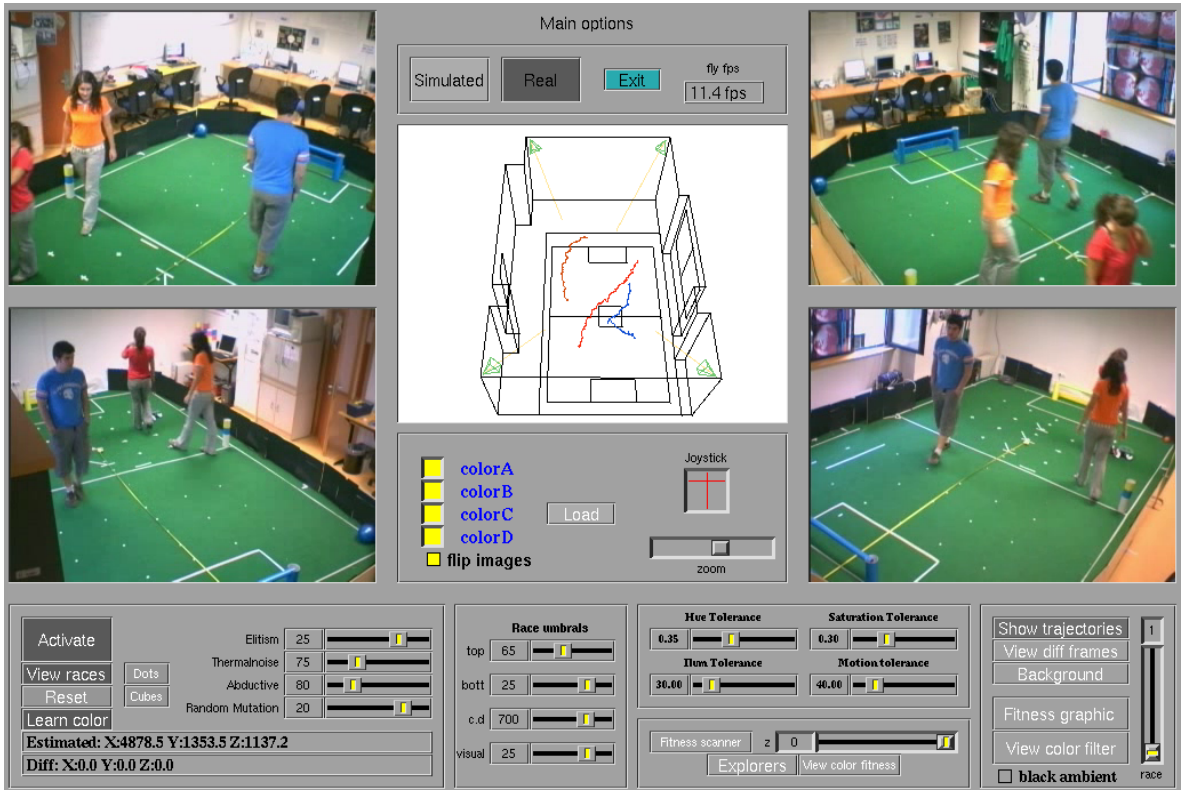
Herramientas de visión

- Filtros de color
- Cámaras en 3D
- Hsvtuner
- Calibrator
- Extrinsics
- OpenCVdemo



Aplicaciones de visión

- CarSpeed
- ElderCare
- Visualizador de esqueletos



6. Conclusiones

- Plataforma para programar robots y aplicaciones domóticas o de visión computacional
- Software libre, comunidad alrededor
- Relajar la arquitectura cognitiva
- Simplifica las prácticas de robótica
- Está en crecimiento

- Distribución intermáquina (ICE)
- Interoperabilidad lenguajes