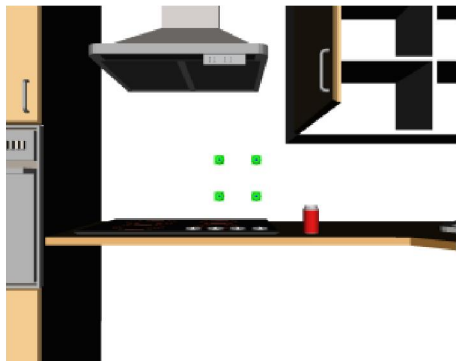


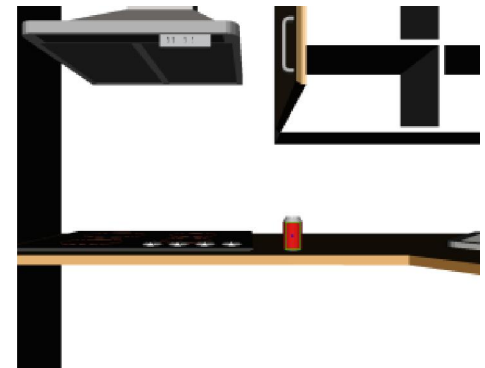


Universidad
Carlos III de Madrid

DESARROLLO DE MÓDULO DE VISUAL SERVOING PARA EL REPOSITORIO OPEN SOURCE ASIBOT



ÁLVARO MARTÍNEZ ESTRADÉ
TUTOR: ALBERTO JARDÓN HUETE
DIRECTOR: JUAN GONZÁLEZ VÍCTORES

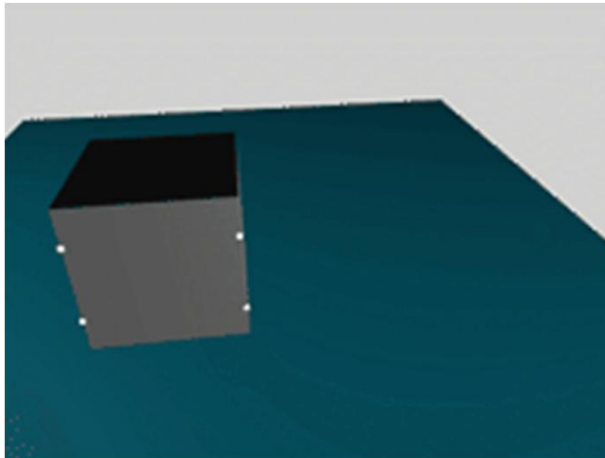




- Objetivos
- Estado del arte
- Libro de Peter Corke
- Sistema de simulación
- Desarrollo del módulo de Visual Servoing
- Funcionamiento y resultados
- Conclusiones y presupuesto



Objetivos específicos



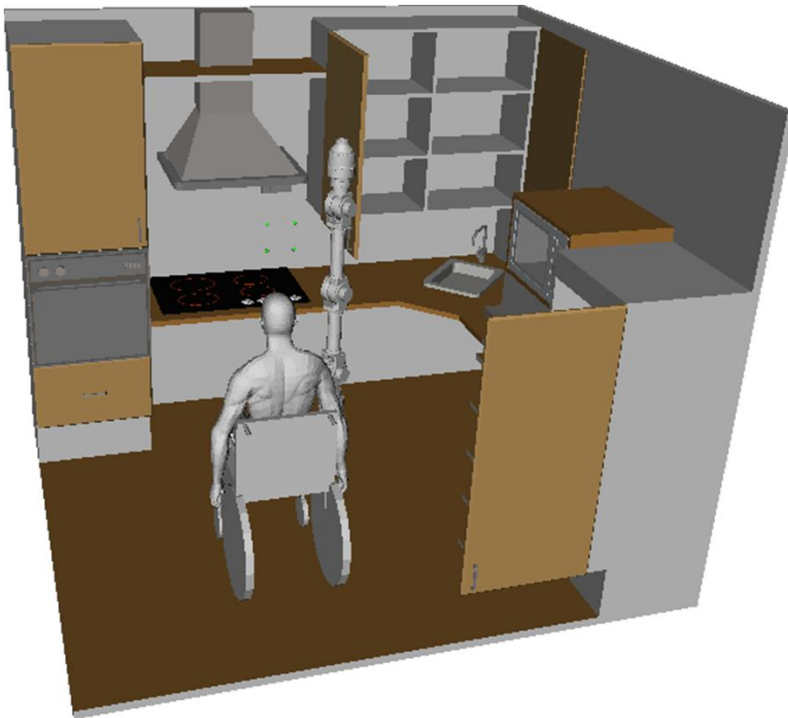
- Conexión MATLAB – OpenRAVE
- Desarrollo de entorno de pruebas en OpenRAVE y robot cartesiano
- Desarrollo sistema de programas
- Corrección de sistema
- Implementación del sistema entorno de la cocina ASIBOT con detección de lata o esferas.

Robot ASIBOT



- Brazo robótico asistencial
- 5 grados de libertad
- Alcance de 1.3 m
- Peso reducido (unos 10 Kg)
- Sistema de control y electrónica a bordo


Repositorio Open Source ASIBOT



- Simulador OpenRAVE
- Control cartesiano mediante terminal en Linux
- Diferentes conexiones preparadas mediante YARP
- Diferentes módulos de visión. OpenCV



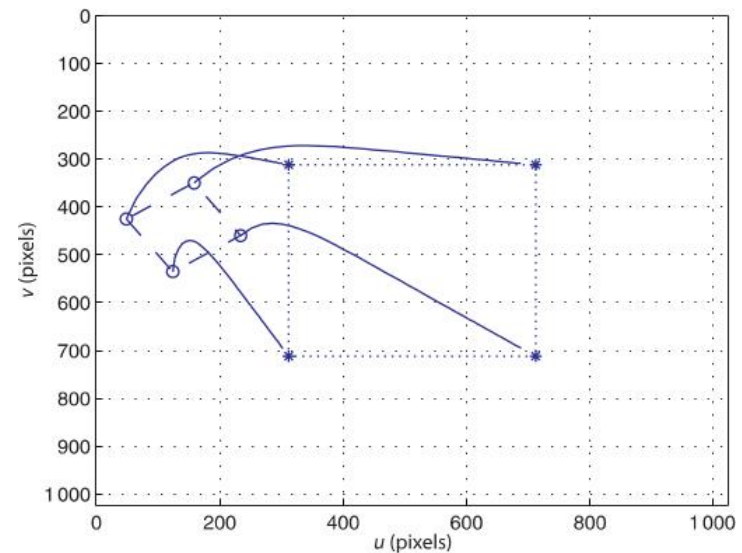
Sistemas de control en robótica

- Sistema de control en lazo abierto
 - Sistema de control en lazo cerrado
- 
- Control de posición
 - Control de velocidad
 - Control fuerza - posición
 - Control fuerza - velocidad
 - Control por impedancia
 - Control visual feedback
 - Control Visual Servoing



Sistemas de control Visual Servoing

- Sistema de control mediante la realimentación de imágenes proporcionadas por una cámara
- Identificación del objetivo buscado en la imagen y marcado del mismo mediante puntos feature o caja envolvente
- Encuadre de las features dentro de la imagen mediante el movimiento del robot



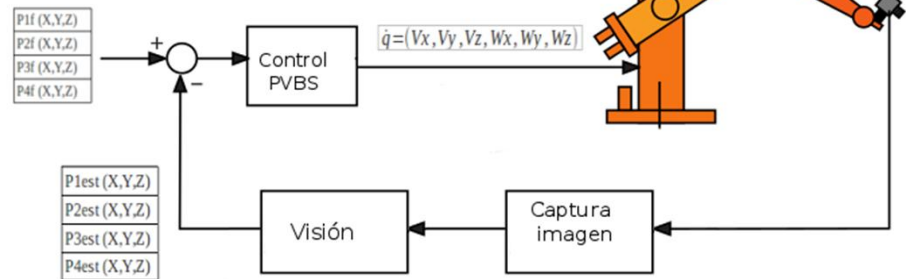
Dos opciones preliminares:

- Cámara montada en el robot
- Cámara fijada en el mundo

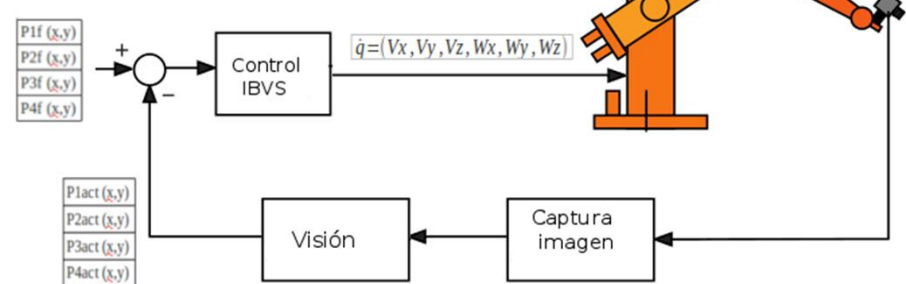
Dos tipos básicos de Visual Servoing:

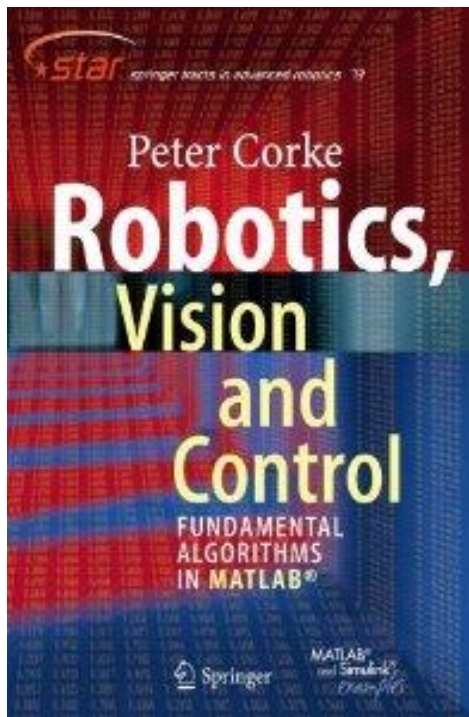
- Visual Servoing basado en posición (PBVS)
- Visual Servoing basado en imagen (IBVS)

a Position-based visual servo



b Image-based visual servo

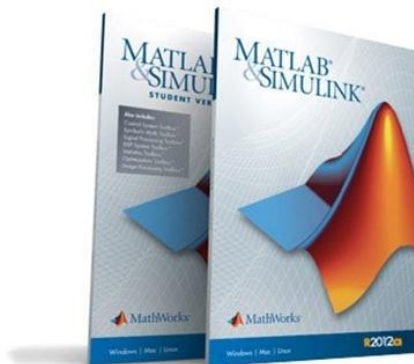




- Libro de robótica general con amplia máquina de visión
- Formación y tratamiento de imágenes
- Diferentes clases para la simulación de cámaras
- Simulación de Visual Servoing basado en posición o en imagen



MATLAB

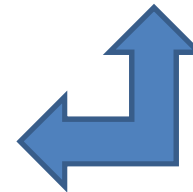
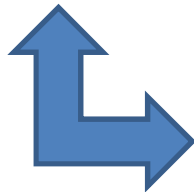


OpenRAVE



YARP

```
zuplent@zuplent:~$ yarp
Call with --help for information on available options.
Options can be set on command line or in /home/zuplent/.yarp/conf/yarpserver.conf
Using port database: ports.db
Using subscription database: subs.db
If you ever need to clear the name server's state, just delete those files.
IP address: default
Port number: 10000
yarp: Port /root active at tcp://127.0.0.1:10000
Registering name server with itself:
  * register "/root" tcp "127.0.0.1" 10000
  * set "/root" ips "127.0.0.1" "172.27.2.12" "::1" "fe80::17a92:9c9f:fe4f:edf0"
  * set "/root" process 3638
  * register fallback mcast "224.2.1.1" 10000
  * set fallback ips "127.0.0.1" "172.27.2.12" "::1" "fe80::17a92:9c9f:fe4f:edf0"
  * set fallback process 3638
Name server can be browsed at http://127.0.0.1:10000/
Ok. Ready!
```





Simulación pura (MATLAB) vs simulación con robot (OpenRAVE)

MATLAB

- Conocimiento a priori de todo el entorno incluyendo posiciones
- Cámara enfocando objeto de interés
- No existen errores de posicionamiento ni existe segmentación
- Cálculo y aplicación de velocidades de manera unitaria (pasos)

OpenRAVE

- Robot en alguna posición del entorno
- Cámara enfocando objeto de interés
- Existe cierto error de posicionamiento de los motores, además de tener segmentación con su error
- Cálculo y aplicación de velocidades de manera continua. Velocidad constante hasta el siguiente cálculo



Modelado de la cámara

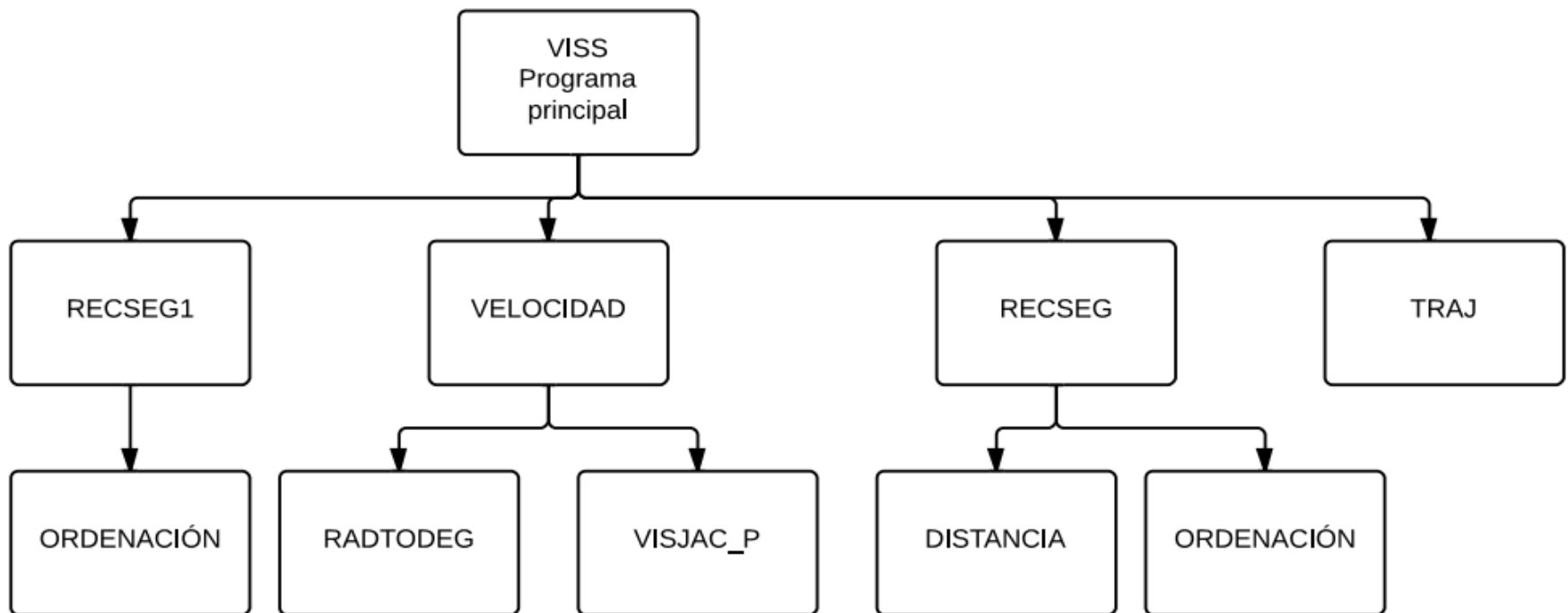
$$KK = \begin{pmatrix} \frac{cam.f}{\rho_x} & 0 & u_c \\ 0 & \frac{cam.f}{\rho_y} & v_c \\ 0 & 0 & 1 \end{pmatrix}$$

$$KK = \begin{pmatrix} 1000 & 0 & 640 \\ 0 & 1000 & 512 \\ 0 & 0 & 1 \end{pmatrix}$$

- Aproximación del modelado de una cámara usada en la simulación
- Obtener datos del fabricante para correcto modelo de cámara
- Mismo modelado en MATLAB y en fichero de entorno de OpenRAVE

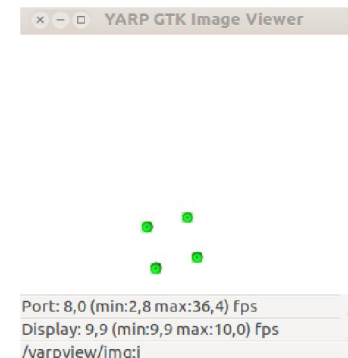
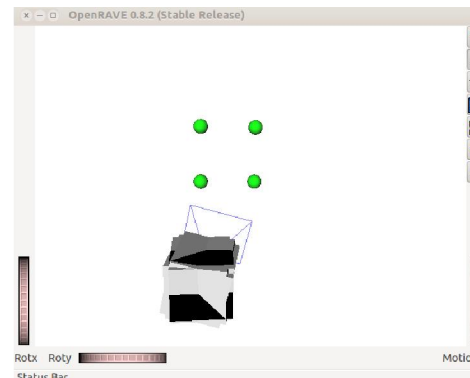
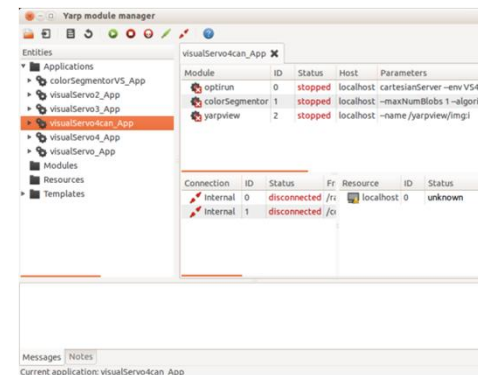


Sistema de programas



Funcionamiento

1. Lanzar OpenRAVE con un entorno determinado (gyarmanager) y conectar cámara
2. Posicionar robot en lugar deseado (viendo features o lata)
3. Lanzar MATLAB. Moverse a la carpeta correspondiente del repositorio





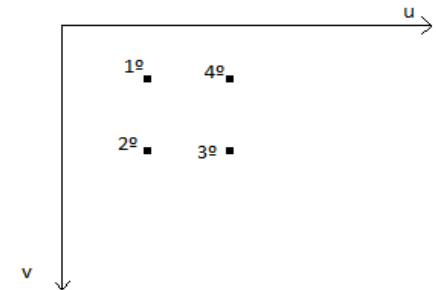
Funcionamiento

4. Definir matriz de coordenadas deseadas en píxeles
(manualmente, precargando, adquiriéndolas)

$$pStar = \begin{pmatrix} x1 & x2 & x3 & x4 \\ y1 & y2 & y3 & y4 \end{pmatrix}$$

5. Iniciar acción de control :

- Finalizar por error umbral
>>[a, b, c, d, e] = viss (pStar, profundidad)
- Finalizar por número de iteraciones
>>[a, b, c, d, e] = viss (pStar, profundidad, nº iter.)





Resultados

Salida por MATLAB una vez finalizada la acción de control

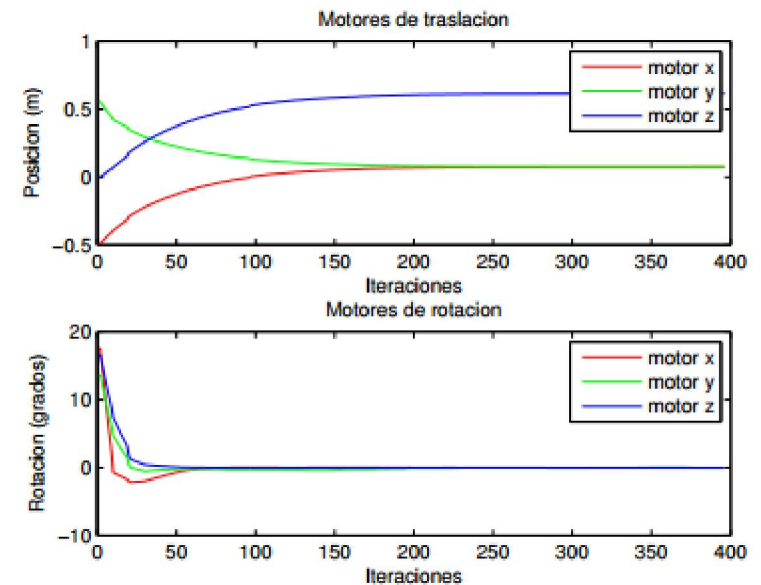
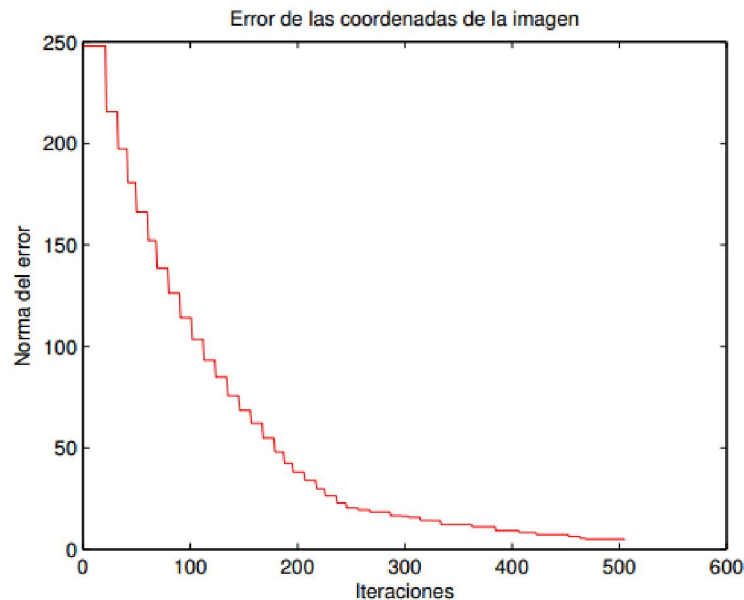
```
>> [a,b,c,d,e] = viss (pStar1,1);  
Yarp library already loaded and initialized, doing nothing  
[success] port connected from /colorSegmentor/state:o  
[success] port connected from /ravebot/rpc:i  
[success] port connected from /ravebot/state:o  
Elapsed time is 46.308809 seconds.  
Completado en el error requerido  
Cerrando puertos...
```

Cinco valores de retorno, matriz o vector (a, b, c, d, e), además del tiempo

Resultados

Ganancias de motores de rotación << Ganancias motores de traslación

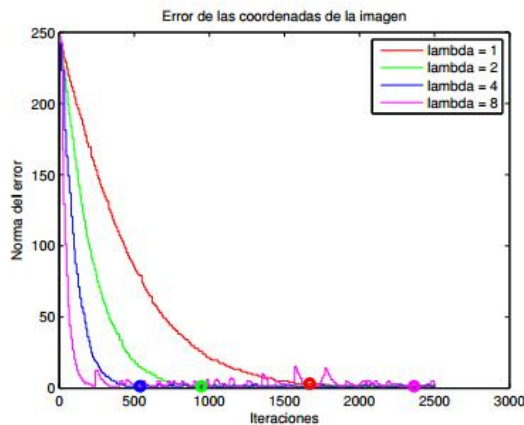
Graficas de error y posición



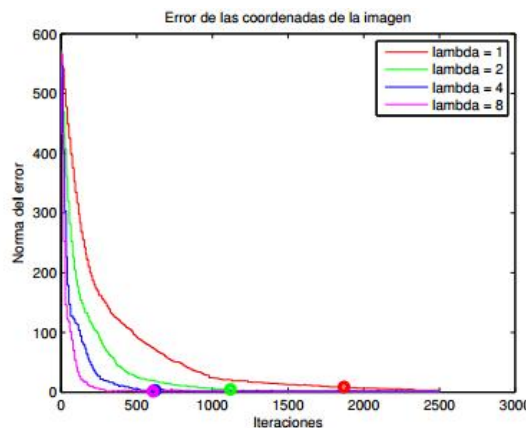
Resultados

Comparación de ganancias

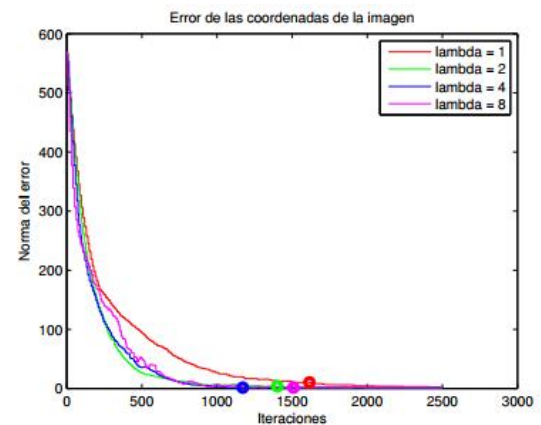
Ganancias independientes



Ganancias ligadas



Ganancias independientes



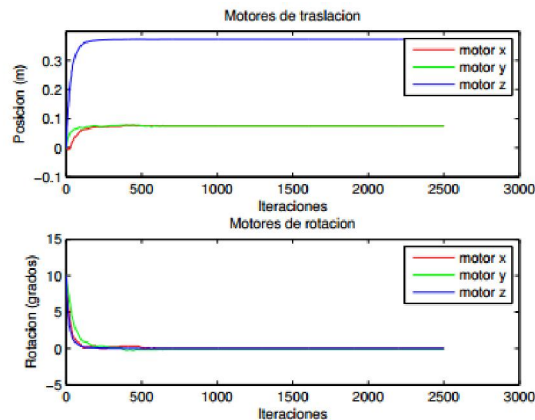
Posición sin rotar sin umbral

Posición rotada con umbral de reducción

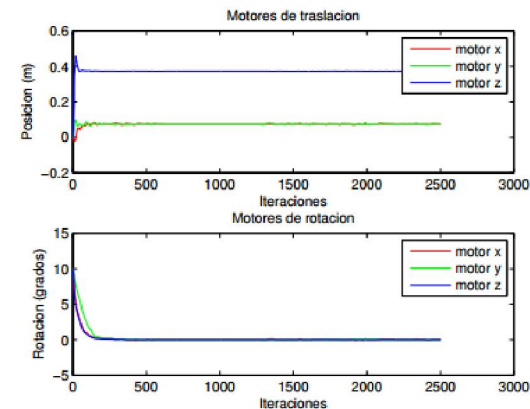
Resultados

Respuesta de los motores según la profundidad aplicada

Profundidad similar a la final



Profundidad mucho mayor a la final





Conclusiones

- Se ha conseguido con éxito la conexión entre módulos
- Desarrollo de entorno de pruebas en el que se ha comprobado el correcto funcionamiento del sistema de control, llevando al robot siempre a la misma posición relativa respecto de las features
- Implantación del sistema de control dentro del entorno de la cocina de ASIBOT con detección de esferas o de lata
- Sistema preparado para implantarlo en otros entornos/robots mediante la adaptación de éstos



Presupuesto

• Ordenador portátil	84 €
• Costes laborales	
• Alberto Jardón	240 €
• Juan González	1200 €
• Álvaro Martínez	5000 €
• Costes indirectos	1304.8 €
• Coste total	<hr/> 7828.8 €