APLICACIÓN DE UN ROBOT COMERCIAL DE BAJO COSTE EN TAREAS DE SEGUIMIENTO DE OBJETOS

APPLICATION OF A LOW COST COMMERCIAL ROBOT IN TASKS OF TRACKING OF OBJECTS

HÉCTOR QUINTIÁN PARDO
Universidad de La Coruña, La Coruña, España, Ingeniero Industrial, hector.quintian@udc.es

JOSÉ LUIS CALVO ROLLE
Universidad de La Coruña, La Coruña, España, Doctor Ingeniero Industrial, jlcalvo@udc.es

OSCAR FONTENLA ROMERO
Universidad de La Coruña, La Coruña, España, Doctor en Informática, ofontenla@udc.es

Recibido para revisar Diciembre 21 de 2011, aceptado Mayo 8 de 2012, versión final Mayo 14 de 2012

RESUMEN: La visión computacional desempeña un papel fundamental en el campo de la robótica móvil, dada la gran cantidad de información sobre el entorno que puede ser extraída de una imagen.
El objetivo del trabajo desarrollado es conseguir un comportamiento autónomo de un robot comercial de bajo coste denominado Rovio, en la tarea de seguir un objeto móvil de forma autónoma. Para lograrlo, ha sido necesario desarrollar un conjunto de funciones, unas de manejo del robot y otras para llevar a cabo la comunicación y transferencia de datos con él.
Se realizan pruebas para determinar la estabilidad de la propuesta implementada y para poder medir la precisión que se obtiene.

PALABRAS CLAVE: Seguimiento, ROVIO, OpenCV, detector de circularidad, teleoperación.

ABSTRACT: The computer vision plays a key role in the field of mobile robotics, given the large amount of information about the environment that can be extracted from an image.
The aim of the work developed is to get a robot autonomous behavior of a low cost commercial called Rovio on the task of following a moving object independently. To achieve this, it has been necessary to develop a set of functions, ones for robot management and others to carry out communication and data transfer with it.
Tests are performed to determine the stability of the proposed and implemented to measure the precision obtained.

KEYWORDS: Follow, ROVIO, OpenCV, circularity detector, teleoperation.

1. INTRODUCCIÓN

La robótica es una disciplina en la que se combinan diversas áreas de conocimiento como la mecánica, electrónica, informática, inteligencia artificial, ingeniería de control…, cuyo principal objetivo es la automatización de procesos para la reducción de costes y/o mejora de las condiciones laborales.

La robótica es capaz de aportar una mayor flexibilidad, una mejora de la calidad y un incremento de la seguridad laboral [1]. Por ello la robótica ha sido empleada en gran cantidad de procesos industriales como: manutención de productos, soldadura, aplicación de producto (pintura, pegado…), carga y descarga de máquinas, mecanización, procesos de corte, ensamblado de productos, paletización, medición e inspección [2].

Si bien en la actualidad los robots industriales realizan múltiples tareas con gran precisión y rapidez, el principal hándicap que presentan está en la falta de movilidad de los mismos, pudiendo sólo actuar dentro de la zona su trabajo, entendiendo esta como el volumen del espacio definido por los puntos accesibles por el elemento terminal del robot [1].

Dentro de la robótica se encuentra la robótica móvil, disciplina que se centra en el estudio de la movilidad de los robots. Dentro de la misma la mayor parte de

las aplicaciones se centran bien en la teleoperación de robots [3](movimientos controlados por un operador humano), o bien en el comportamiento autónomo de los mismos (sin ningún tipo de control por parte de los seres humanos).

En la robótica móvil, uno de los aspectos fundamentales es la obtención de información del entorno en el que el robot se desenvuelve. Para ello los robots móviles disponen de múltiples sensores, como pueden ser sensores de proximidad, de presión, de luz… Sin embargo la cantidad de información que se puede extraer de dichos sensores es limitada, sin embargo han sido utilizados en gran cantidad de trabajos como [4,5] debido a que la información procedente de dichos sensores es muy fácil de procesar e interpretar. Una de las opciones de sensorización que mayor importancia tiene en la actualidad, está en la visión computacional, ya que de a partir de una imagen es posible extraer gran cantidad de información como pueden ser distancias, ubicación, posición, formas y colores de objetos…

Los principales inconvenientes que plantea la visión computacional en robots son el coste computacional que requiere la extracción y el procesado de información de imágenes, así como la dependencia de las condiciones del entorno, en términos de iluminación, presencia de objetos, etc. Dicho coste no supondría un problema en otros ámbitos, sin embargo en la robótica móvil sí, ya que la mayoría de las operaciones sobre los robots deben realizarse en tiempo real.

En este trabajo se presentan un conjunto de funciones para el manejo de un robot móvil comercial denominado Rovio así como un algoritmo basado en reglas para el seguimiento de objetos que utiliza instrucciones de tipo giro incremental, disponibles en el robot, para conseguir realizar el seguimiento de un objeto. La principal ventaja de este sistema frente a otros, es su baja carga computacional, lo que constituye un aspecto fundamental a la hora de conseguir un control en tiempo real…

Muchos han sido los autores que han tratado tanto el seguimiento de objetos en tiempo real [6-9], así como el del desarrollo de controladores de gran precisión [10-12] o técnicas de control inteligente [13-15].

Sin embargo dichos estudios no son de aplicación en el presente trabajo debido a dos motivos:

- El robot empleado en este trabajo, dispone de un controlador interno con el que sólo es posible comunicar mediante instrucciones preestablecidas, permitiendo sólo órdenes de giro "incrementales", o ángulos de giro absolutos no inferiores a 20 grados. Por ello no es posible tener gran precisión en el control de los actuadores.
- El elevado coste temporal empleado en la transferencia de imágenes a la estación de trabajo imposibilita el uso de algoritmos de control de gran precisión que precisan de un elevado coste computacional.

Este robot también ha sido previamente usado en algunos trabajos como en [16], pero con una finalidad distinta a la del presente trabajo, siendo el objetivo de [16] el evitar obstáculos fijos en lugar de realizar el seguimiento de objetos como ocurre en el presente trabajo.

En los siguientes puntos se dará una visión de la arquitectura del robot y se describirán el conjunto de funciones desarrolladas para el control del robot, describiendo más en detalle el algoritmo de seguimiento empleado, para finalizar con un conjunto de experimentos encaminados a medir la estabilidad y precisión del sistema implementado.

2. COMPONENTES DEL SISTEMA

Para poder realizar la esta investigación han sido necesarios los siguientes elementos:

- Robot móvil comercial de bajo coste denominado Rovio.
- Estación de trabajo formada por un ordenador con conexión wifi.
- Router inalámbrico.

2.1. Robot móvil

Se ha utilizado un robot comercial denominado Rovio que está provisto de 3 ruedas omnidireccionales dispuestas en una configuración de triángulo, una de ellas trasera y las otras 2 a ambos lados del robot formando un ángulo de 60 grados con respecto a la trasera (ver figura 1). Las dimensiones: 34.29 x 30.48 x 35.56 cm y el peso de: 2.3 Kg [17].

El robot Rovio dispone de: cámara multiresolución, sensor de infrarrojos, micrófono, altavoz, leds en parte frontal y superior, antena de comunicación wifi, brazo orientable en eje vertical para el posicionamiento de la cámara (ver figura 2)…

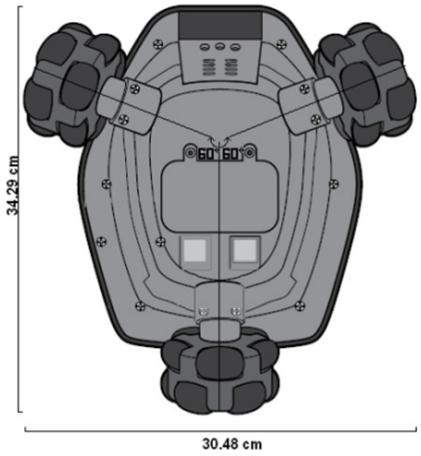

Figura 1. Dimensiones de Rovio [17]

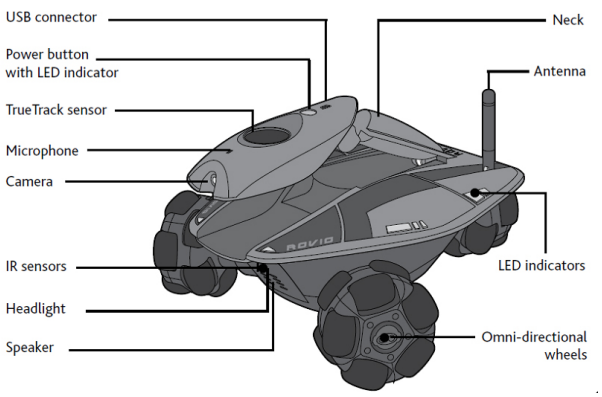

Figura 2. Componentes y sensores de Rovio [17]

Las principales ventajas del robot así como de la configuración empleada, son:

- El sistema de control de los actuadores, así como la comunicación con los sensores, está totalmente integrado en Rovio.
- Posee una gran movilidad, gracias a la disposición de sus ruedas.

Como inconvenientes principales:

- a pesar de que dispone de una interfaz gráfica para el manejo del robot, esta no es válida para el manejo autónomo del mismo, ya que es de código cerrado, lo que ha llevado a tener que establecer el protocolo de comunicación con el robot para el envío de órdenes y la lectura de datos.
- El tiempo de transferencia de imágenes es elevado, lo que provoca que se tenga que usar una resolución inferior a la máxima permitida por el robot.

2.2. Estación de trabajo

En el presente trabajo, Rovio será manejado por una estación de trabajo, consistente en un ordenador con conexión wifi, en la que se programarán los comportamientos autónomos del robot y comunicando con él a través de wifi.

El proyecto ha sido desarrollado en el sistema operativo Linux bajo Ubuntu 9.10, en un ordenador portátil con procesador AMD Athlon 2650e con 2 GB de memoria RAM, con adaptador wifi 802.11b/g.

Para el procesado y análisis de imágenes, se ha utilizado la librería de tratamiento de imágenes OpenCV [20], teniendo como principal ventaja frente a otros programas de tratamiento de imágenes que este es de distribución libre.

2.3. Router inalámbrico

El proceso de comunicación con Rovio se realiza a través de wifi, bien por medio de router o bien directamente con el robot.

En este proyecto la comunicación es realizada a través de un router inalámbrico Thomsom TCW710 (figura 3).

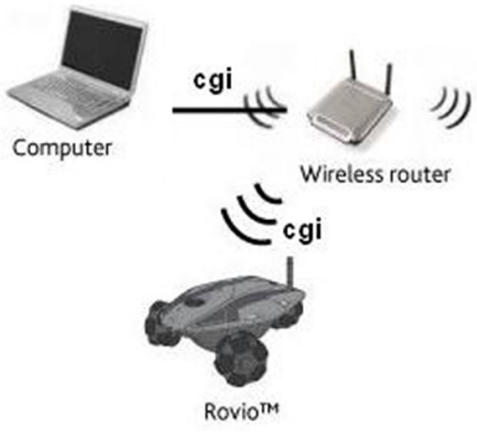

Figura 3. Estructura de comunicación con Rovio

La comunicación con Rovio se realiza a través de wifi empleando el protocolo HTTP (HyperText Transfer Protocol), donde las órdenes que son enviadas al robot, tienen que ser peticiones CGI (Common Gateway Interface) [18]. El protocolo HTTP es el protocolo usado en cada transacción de la web y cuyo objetivo es coordinar la transferencia de datos, en este caso, entre la estación de trabajo y el robot.

El uso del sistema operativo Linux, permite el uso de un comando denominado "CURL", con el que es posible la transferencia de datos entre la estación de trabajo y el robot sin preocuparse del proceso de establecimiento y finalización de la comunicación entre ambos. Sin embargo se ha preferido usar "SOCKETS" [19], con los que es necesario establecer la comunicación, solicitar el envío de datos y finalizar la comunicación. No obstante, con "SOCKETS", se consigue que el proceso de comunicación sea independiente del sistema operático empleado, al contrario que con el comando "CURL", el cual es propio de Linux.

3. TRABAJO DESARROLLADO

Para la consecución del objetivo global de la investigación, ha sido necesario realizar tres grandes tareas, que se indican a continuación:

- Desarrollo y programación de una librería de funciones para el control de Rovio desde el terminal de trabajo, en el cual se implementarán los comportamientos autónomos.
- Desarrollo e implementación de un algoritmo para el seguimiento de objetos, en concreto una pelota de color naranja.
- Creación de un API (Application Programming Interface) de la librería desarrollada para la documentación de las funciones.

En los siguientes subapartados se desarrolla cada una de ellas por separado.

3.1. Librería de funciones de control

La interfaz gráfica de que dispone Rovio es de código cerrado. Por ello, el primer paso para el manejo de Rovio, ha sido la realización de una librería de funciones básicas para el control de los movimientos, recepción de imágenes, manejo de luces y comprobación del estado de los distintos componentes: dirección de rotación de cada rueda, número de pulsos de los encoders de cada rueda desde la última lectura, nivel de batería, etc. También se implementa la función de detección de caras disponible en OpenCV, basada en el método de Viola and Johns [21]. Por último se desarrollan dos funciones de mayor complejidad, usando como base las anteriores para la teleoperación del robot y para el cálculo del centroide de una pelota de color naranja.

Teleoperación de Rovio

El objetivo de esta función, es conseguir que los movimientos del robot puedan ser totalmente controlados por un operador humano desde el teclado de la estación de trabajo. Además de un control total sobre los movimientos del robot también es necesaria la recepción de imágenes en tiempo real para que el operador humano pueda manejar el robot en el entorno en que este se encuentre, ya que el robot puede encontrarse en zonas donde el operador no esté presente.

Se han implementado los mismos movimientos disponibles en la interfaz gráfica. También se ha añadido la posibilidad de mover el "cuello" de Rovio con incrementos de 5 grados, lo cual no es posible en la interfaz gráfica, en la que el "cuello" (ver figura 1), sólo puede adoptar 3 posiciones, baja, media o alta. Esta nueva funcionalidad ha sido desarrollada con objeto de poder realizar el seguimiento de un objeto también en el eje vertical si fuese necesario.

Para la teleoperación, cada vez que se recibe una imagen, se leen los eventos producidos en el teclado vaciando el buffer del mismo, ya que al pulsar de manera continua, el número de pulsaciones almacenadas es dependiente de la configuración del teclado. Se asocia a cada evento de teclado una acción, enviando la instrucción correspondiente a Rovio mediante petición CGI.

Cálculo del centroide

El objetivo de esta función es calcular el centroide de una pelota de color naranja dentro de una imagen enviada por Rovio. Para ello se optó por dos alternativas:

- La primera se basa sólo en el color de los objetos para el cálculo del centroide.

- La segunda tiene en cuenta además la forma de los objetos.

La detección sólo por color tiene un coste computacional inferior a la detección por color y forma, sin embargo la detección mediante color y forma aporta una mayor robustez a la detección, permitiendo distinguir entre objetos de distintas formas y mismo color.

Basándose sólo en el color

Se calcula el centroide sólo del conjunto de puntos cuyos valores RGB (Red, Green, Blue) se encuentre próximos a los valores del naranja de la pelota buscada. Para dicho cálculo se siguen los siguientes pasos:

- Se recorre la imagen pixel a pixel, realizando la suma de las coordenadas de aquellos pixeles cuyo valor de RGB este dentro del rango de valores de RGB que se consideran como el color naranja de la pelota buscada.
- Se guarda el número total de pixeles que cumplen la condición anterior.
- Si el número total de pixeles del paso anterior es inferior a 20, se reporta como falso positivo, lo que implica que no hay ninguna pelota dentro de la imagen.
- Si el número total de pixeles del paso anterior es superior a 20, se divide la suma realizada en el primer paso, entre el número total de pixeles del segundo paso, obteniéndose así el centroide de la pelota buscada.

Basándose en el color y la forma

En este método, a la imagen lo primero que se le aplica es un detector de circularidad, de modo que se descarte cualquier objeto que no sea un círculo. Una vez identificado todo aquello que constituye un círculo, caben 3 posibilidades:

- si no hay ningún circulo, no habrá pelota,
- si hay un círculo, se comprobará si los valores medios de RGB en el interior del círculo, son próximos a los valores de referencia, en cuyo caso se considera que es el objeto buscado.
- Si hay más de un círculo, se comprobará cuál de ellos tiene unos valores de RGB más próximos a los de referencia, estableciendo el mismo como la pelota buscada.

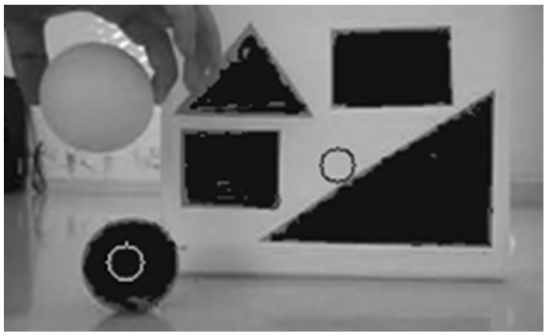

Figura 4. Con círculo rojo, el centroide usando sólo el filtro de color y con círculo verde, el centroide usando el filtro de color y el detector de circularidad.

De esta forma es posible discriminar objetos del mismo color que no sean la pelota y también se consigue ser menos restrictivo en cuanto a nivel de iluminación. La figura 4 muestra, con un círculo rojo el centroide obtenido usando solo el filtro de color, y con un círculo verde, el centroide usando el filtro de color más el detector de circularidad. Los pixels filtrados por el filtro de color han sido pintados en azul.

3.2. Algoritmo de seguimiento y acercamiento a pelota

El objetivo del algoritmo desarrollado es el realizar el seguimiento y acercamiento a un objeto mediante el robot Rovio de forma autónoma. En particular se ha implementado el seguimiento de una pelota de color naranja.

El seguimiento de objetos ha sido objeto de estudio de en múltiples trabajos [3-6], en los cuales se obtienen resultados muy precisos gracias a que se dispone del modelo cinemático del robot o bien mediante el uso de algoritmos de control complejos y con elevado coste computacional. Sin embargo, el robot empleado en este estudio no permite ninguna de las anteriores alternativas, ya que no se dispone del modelo cinemático del mismo y el coste computacional empleado en la transferencia de imágenes hace inviable el empleo de algoritmos con un alto coste computacional. Por ello el algoritmo desarrollado consiste en un sistema basado en reglas, con bajo coste computacional.

A continuación se presenta el pseudocódigo de dicho algoritmo basado en reglas:

```
if Objeto dentro de la imagen then
   if Objeto 20 pixel izq. -> direction=left.
   if Objeto 20 pixel derc.-> direction=right.

      if objeto fuera limites-> update the status
         if Objeto derch. y puede actuar-> turn_right.
         if Objeto izq. y puede actuar->  turn_left.
      else
         direccion=centro
        if Objeto lejos-> avanzar

else
         if direction es izq. -> update status2
%Status2 para cuando el obj.. está fuera de la imagen
                  if satus2==Action -> turn_left

         if direction=right -> update status2
          if status2=Action -> turn_right
```

Instrucciones para el movimiento de Rovio

Para realizar la implementación del algoritmo desarrollado, es necesario el uso de instrucciones de giro del robot. Para ello, Rovio dispone de dos tipos de instrucciones de giro distintas [22]:

- "Giro incremental".
- "Girar un ángulo"

La primera variante, son instrucciones de giro que por sí solas no son capaces de vencer las fuerzas internas del robot, por lo que deben de combinarse una sucesión de instrucciones en un corto espacio de tiempo, produciendo pequeños movimientos. El principal inconveniente es que no permite precisar con exactitud los grados que va a girar el robot, sin embargo, experimentalmente se ha comprobado que se puede conseguir un ángulo de giro de entre 3 y 6 grados, concatenando 3 instrucciones de giro.

La segunda variante las instrucciones de giro permiten que el robot gire unos determinados ángulos (20º, 30º, 45º, 55º, 65º, etc.). El principal problema que plantea es la imposibilidad de realizar giros de ángulo inferior a 20º. Como variante de esta opción estaría el enviar una instrucción de giro de 90º, seguida por una instrucción de parada con un espacio de tiempo para que gire el ángulo deseado, sin embargo los retardos producidos por el coste temporal de la recepción y procesado de imágenes imposibilitan dicha opción.

Algoritmo de seguimiento y acercamiento

El algoritmo desarrollado realiza el seguimiento y acercamiento a un objeto, en este estudio se trata de una pelota de color naranja, sin embargo este algoritmo sólo precisa las coordenadas del centroide del objeto a seguir, de modo que el proceso de detección y cálculo del centroide es competencia de la función explicada en el apartado anterior.

El algoritmo implementado es un sistema basado en reglas que lo que hace es determinar la posición del objeto respecto al centro de la imagen, para determinar en qué dirección debe de moverse Rovio. También se almacena la última posición del objeto, para poder determinar, por diferencia con la posición actual, la dirección que está siguiendo el objeto. De esta forma, en caso de que el objeto salga del campo de visión de Rovio, el robot seguirá girando en la misma dirección hasta encuadrarlo de nuevo dentro de su campo de visión.

Experimentalmente se ha comprobado que para poder obtener un adecuado refresco de imágenes (un refresco de unos 100ms) a la par de conseguir que el robot se mueva de forma fluida y no intermitente, era necesario enviar una instrucción de giro cada vez que se reciba una imagen, en lugar de enviar varias instrucciones de giro entre imágenes consecutivas. Además no se debe de realizar más de tres veces consecutivas el ciclo de recepción de imagen y envío de una misma instrucción. Es decir si se quiere que el robot gire a la derecha, recibiremos una imagen y enviaremos una instrucción de giro a derecha, recibiremos la siguiente y enviaremos una nueva instrucción, recibiremos la tercera imagen y enviaremos la tercera instrucción, recibiremos la cuarta y sin embargo no podremos enviar una cuarta instrucción igual a las demás, sino que tendremos que esperar a una sexta imagen. De esta forma evitamos que la inercias alcanzadas por Rovio sean excesivas (espera de 2 ciclos por cada 3 ciclos de movimiento), consiguiendo un mayor control sobre Rovio.

El número de ciclos de acción consecutivos y de espera, han sido calculados de forma experimental y sobre un mismo tipo de suelo (baldosa lisa), sin embargo para otros suelos con mayores coeficientes de fricción, esta relación podría aumentar, obteniéndose un movimiento más rápido, con el mismo ratio de error. Relaciones inferiores de 3 a 2 harían que Rovio se moviese de forma no continua o incluso que no llegase a moverse.

4. PRUEBAS REALIZADAS

Se realizan pruebas al algoritmo de seguimiento y acercamiento a pelota una vez implementado, utilizando en todas ellas la mínima velocidad de movimiento permitida por el robot, con objeto de ver los resultados que se obtienen con él. Las principales pruebas realizadas son las dos que se muestran a continuación:

1.) Se dispuso una pelota naranja de 6 cm de diámetro a una distancia de 92 cm de la rueda trasera de Rovio y 28 cm a la derecha de la misma (figura 5). El objetivo de esta prueba es que el robot mediante el algoritmo desarrollado centre la pelota lo más rápido que pueda así como con el menor error posible. Par ello se parte de la posición inicial mostrada en la figura 5 y se arranca el algoritmo para que centre la pelota, al mismo tiempo que se toman medidas de posición de la pelota en la imagen en distintos instantes temporales desde el inicio del algoritmo hasta la estabilización del robot.

Dicha prueba se repitió para distintos márgenes de error permitidos respecto al centro de la imagen (figura 6), para determinar qué valores de margen de error eran aceptables para obtener un control estable de Rovio.

En este experimento, sólo se tienen en consideración las coordenadas en el eje horizontal, ya que las verticales implicarían un movimiento del ángulo de la cámara, siendo el control mucho más complejo. Además, estas medidas se han elegido de forma que la pelota quede lo más a la derecha de la imagen, para que así Rovio adquiera la máxima inercia posible, que se correspondería con el peor caso.

Se realizaron 3 pruebas para márgenes de error respecto del centro de la imagen, del ±20% del ancho de la imagen, de ±10% y del ±5% (figura 6).

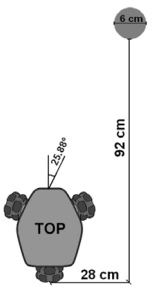

Figura 5. Posición de elementos en el primer experimento

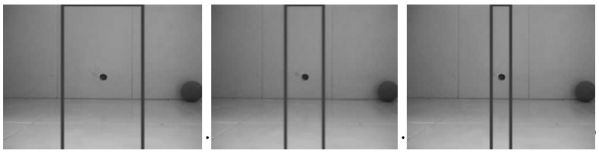

Figura 6. Márgenes de error ±20%, ±10%, ±5% (de izquierda a derecha)

1.) En el segundo experimento lo que se ha tratado es de medir el error producido usando las instrucciones de giro de que dispone Rovio. Estas medidas son dependientes del suelo sobre el que este Rovio por lo que en los experimentos se ha usado un suelo de baldosa lisa, ya que provoca los mayores errores por el bajo coeficiente de rozamiento de dicho suelo.

En este experimento, se han realizado 5 medidas para cada uno de los ángulos de giro disponibles en forma de instrucción, se ha promediado y calculado los errores y su desviación. Los ángulos disponibles en forma de instrucción son 20º, 30º, 45º, 55º, 65º, 80º, 90º, 100º, 110º, 125º, 135º, 150º, 160º, 170º y 180º. Todos ellos pueden realizarse tanto a la derecha como a la izquierda, cubriendo así un área de giro de 360 º.

Para realizar las medidas se ha utilizado una cámara ubicada en el techo, se ha tomado una imagen de referencia, y con cada instrucción de giro se toma una nueva imagen, de modo que, superponiendo las imágenes, se puede calcular el ángulo girado con cada instrucción.

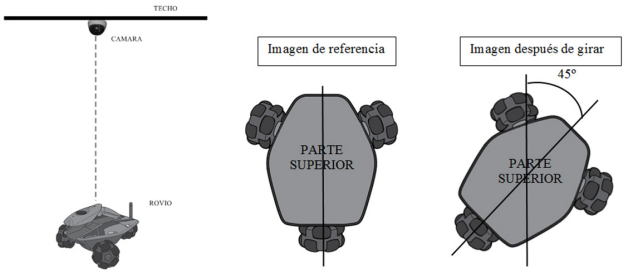

Figura 7. Segundo experimento

5. RESULTADOS

En este apartado se muestran los resultados de los 2 experimentos realizados, cuya metodología ha sido explicada en el apartado anterior.

Primer experimento

En el primer experimento se grafica la posición de la pelota en pixeles (eje vertical), en función del tiempo en milisegundos (eje horizontal) (figura 9) para distintos márgenes de error permitidos.

Para graficar la respuesta obtenida, se ha procedido a realizar un desplazamiento del eje de coordenadas de modo que el nuevo eje de coordenadas se sitúe sobre el centro de la pelota en su posición inicial (figura 8). El eje de coordenadas original se sitúa en la esquina superior izquierda de la imagen (imágenes de 320x240), por lo que para desplazarlo al centro de la pelota, la cual se encuentra a 300 pixeles a la derecha del borde izquierdo de la imagen, será necesario restar a las coordenadas originales 300 pixeles en el eje horizontal y cambiar el signo resultante, ya que el nuevo eje de coordenadas se incrementa hacia la izquierda. Así por ejemplo si el cálculo del centroide de la pelota da como resultado (280,y), al mover el eje de coordenadas, tendremos (280-300)*-1=+20 lo que quiere decir que la pelota se ha desplazado 20 pixeles a la izquierda de la posición inicial de la misma.

Mediante este cambio del eje de coordenadas, el centro de la imagen tendrá coordenadas (160,y), y un punto en el extremo derecho de la imagen tendrá (-20,y).

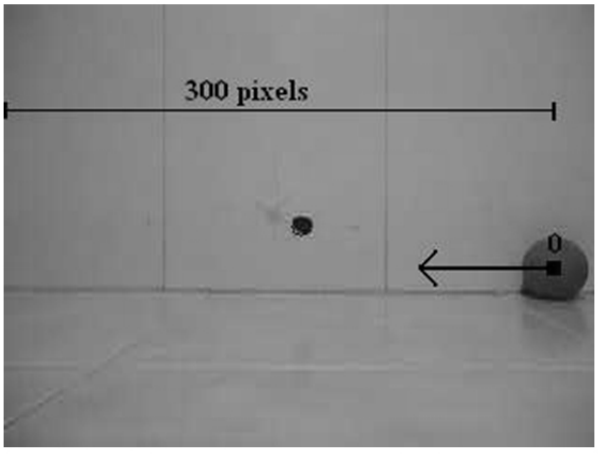

Figura 8. Desplazamiento del eje de coordenadas

Para el primer experimento, se obtuvieron los siguientes resultados:

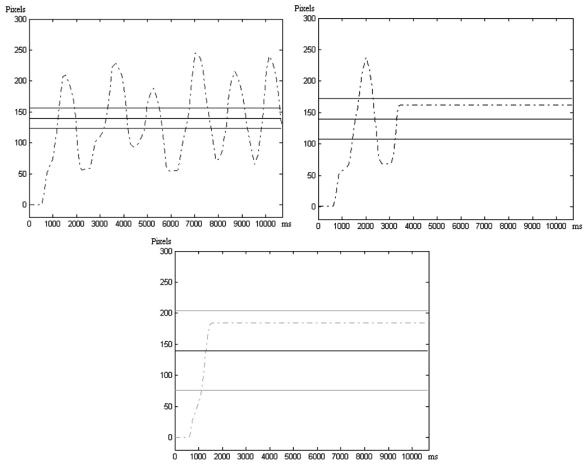

Figura 9. Respuesta con un margen de ±5% (rojo), ±10% (azul) y de ±20% (verde) del ancho de ventana

Las líneas continuas en color, representan los márgenes de % admitidos, por lo que para que el sistema sea estable la respuesta (línea discontinua) debe de permanecer, después del estado transitorio, dentro de los márgenes de error (líneas continuas). La línea continua negra representa las coordenadas del centro de la imagen (figuras 9 y 10).

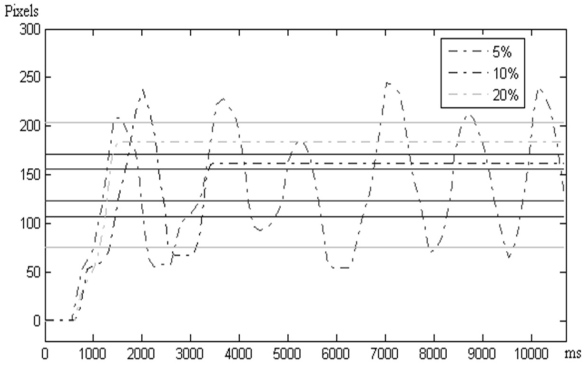

Figura 10. Comparativa de las 3 respuestas

Segundo experimento

En el segundo, se representa en una tabla el ángulo real girado por cada instrucción de "girar un ángulo" de las que dispone Rovio. Dichas medidas han sido realizadas 5 veces para cada ángulo, obteniéndose la media de los ángulos girados, el error medio y la desviación (tabla 1).

Para el segundo experimento, se obtuvieron los siguientes resultados:

Tabla 1. Resultados de mediciones de ángulos

ANGULO	MEDICIONES					MEDIA	s	ERROR MEDIO
	1	2	3	4	5			
20	23	21	21	22	21	21,6	0,894	1,60
30	30	30	29	30	28	29,4	0,894	0,60
45	44	43	43	43	44	43,4	0,548	1,60
55	54	54	54	54	54	54	0,000	1,00
65	64	62	64	63	62	63	1,000	2,00
80	78	79	79	79	77	78,4	0,894	1,60
90	89	90	89	89	89	89,2	0,447	0,80
100	103	101	102	103	103	102,4	0,894	2,40
110	113	110	112	111	113	111,8	1,304	1,80
125	127	127	127	126	126	126,6	0,548	1,60
135	133	132	133	132	134	132,8	0,837	2,20
150	151	150	150	152	151	150,8	0,837	0,80
160	161	162	161	161	160	161	0,707	1,00
170	169	168	169	170	170	169,2	0,837	0,80
180	179	179	179	179	179	179	0,000	1,00
Todos los datos están en grados								**1,39**

6. CONCLUSIONES Y DISCUSION

De los resultados anteriores se deduce que un margen igual o inferior al ±5% de error, da lugar a que la respuesta del sistema sea oscilante [23], [24] de modo que no será posible conseguir centrar la pelota dentro de ese margen.

Si se aumenta el margen de error permitido, se observa que se consigue estabilizar el sistema, obteniendo en el caso de ±10% una respuesta con mayor sobreoscilación [24], [25] que en el caso del ±20% donde la respuesta no presenta ningún tipo de sobreoscilación. Ello es debido a que al aumentar los márgenes, una vez que se entre dentro de la zona de error permitido, se dejará de enviar ordenes de movimiento al robot, teniendo un mayor margen de espacio para que actúe la inercia sin conseguir que el centro de la pelota salga de los márgenes establecidos. La opción más recomendable es la del ±20%, ya que si se efectúa el seguimiento de un objeto en movimiento con fuertes y constantes cambios de dirección, el margen del ±10% en ocasiones no es suficiente, provocando inestabilidades.

En cuanto a los resultados del segundo experimento, se deduce que el error medio cometido por las instrucciones de giro, es de 1.4º, sin embargo el error tiene una alta repetitividad. Esto se debe a que Rovio dispone de un encoder en cada una de las ruedas, el cual le aporta información precisa sobre el ángulo girado, sin embargo, el error no es cero, puesto que cuando se produce un deslizamiento de una de las ruedas, el robot no se mueve pero si se computa el ángulo giro por la rueda. Este deslizamiento es muy dependiente de la superficie sobre la que se mueva el robot, por ello en los experimentos realizados se ha elegido una superficie con poco agarre, como es el caso de la baldosa lisa.

El principal problema de estas instrucciones de "girar un ángulo", es que Rovio no dispone de ángulos de giro inferiores a 20º, por lo que este tipo de instrucciones no servirán para realizar el seguimiento de un objeto de forma estable.

REFERENCIAS

[1] Siegwart, R., Introduction to autonomous mobile robots. The MIT Press, 2004.

[2] Turiel, J., Fraile, J.C. and Peran, J. R., Aplicaciones de la Robótica: Últimas tendencias y nuevas perspectivas, Dyna, pp. 61-68, 2002.

[3] Santonja, R. A. and Sabater, J., Robots trepadores de estructura paralela Dyna, pp. 55-59, 2000.

[4] Floreano, D. and Mondada, F., Evolutionary neurocontrollers for autonomous mobile robots, Neural networks : the official journal of the International Neural Network Society, vol. 11, no. 7-8, pp. 1461-1478, Oct. 1998.

[5] Meeden, L., An incremental approach to developing intelligent neural network controllers for robots, IEEE transactions on systems, man, and cybernetics. Part B, vol. 26, no. 3, pp. 474-85, Jan. 1996.

[6] Brezak, M., and Petrovic, I., Global Vision Based Tracking of Multiple Mobile Robots in Subpixel Precision, 2007 5th IEEE International Conference on Industrial Informatics, pp. 473-478, Jul. 2007.

[7] Rui, L., Zhijiang, D. and LINING, S., Moving Object Tracking based on Mobile Robot Vision, Area, pp. 3625-3630, 2009.

[8] Rokunuzzaman, MD., Sekiyama, K. and Fukuda, T., Real Time Detection and Evaluation of Region Of Interest by Mobile Robot using Vision, 18th IEEE International Symposium on Robot and Human Interactive Communication, pp.1149 – 1154, 2009.

[9] Guo, X., Wang, C. and Qu, Z., Object Tracking

for Autonomous Mobile Robot based on Feedback of Monocular-vision 2007 2nd IEEE Conference on Industrial Electronics and Applications, pp. 467-470, May. 2007.

[10] Xu, D., Zhao, D., YI, J. and Tan, X., Trajectory Tracking Control of a Four-wheel Differentially Driven Mobile Robot, IEEE transactions on systems, man, and cybernetics. Part B, vol. 39, no. 3, pp. 788-99, Jul. 2009.

[11] Hwang, W., Park, J., Kwon, H. and Anjum, M., Vision tracking system for mobile robots using two Kalman filters and a slip detector, 2010 International Conference on Control Automation and Systems (ICCAS),, pp. 2041-2046, 2010.

[12] Fukao, T., Nakagawa, H. and Adachi, N., Adaptive tracking control of a nonholonomic mobile robot, IEEE Transactions on Robotics and Automation, vol. 16, no. 5, pp. 609-615, 2000.

[13] Machán-González, I., López-García, H. and Calvo-Rolle, J. L., Controlador Neuro-Robusto para sistemas no lineales, Dyna, vol. 86, pp. 308-317, 2011.

[14] Jiménez, J.A., Ovalle, D.A., Ochoa, J.F., SMART: Sistemas Multi-Agente Robótico, Dyna, 75 (154), pp. 179-186, 2008.

[15] Ramírez, J.F., Jiménez, J.A., Álvarez, J.S., Ciclo de diseño de un robot para el aprendizaje y desarrollo de la creatividad en ingeniería, Dyna, 170, pp. 51-58, 2011.

[16] Begum, A. and Lee, M., A Simple Visual Servoing and Navigation Algorithm for an Omnidirectional Robot, Human-Centric Computing (HumanCom), pp. 1-5, 2010.

[17] WOWWEE GROUP LIMITED, Rovio user manual, 2008.

[18] Mora, S. L., Programación de servidores web con CGI, SSI e IDC, Club Universitario, p. 1, 2001.

[19] Donahoo, M. J. and Calvert, K. L., TCP/IP sockets in C: practical guide for programmers, Morgan Kaufmann Pub, 2009.

[20] Bradski, G. and Kaehler, A., Learning OpenCV, O´Reilly, Sep. 2008.

[21] Viola, P. and Jones, M., Robust real-time object detection, International Journal of Computer Vision, vol. 57, no. 2, pp. 137–154, 2002.

[22] WOWWEE GROUP LIMITED, API Specification for Rovio, 2008.

[23] Dorsey, J., Sistemas de control continuos y discretos, McGraw-Hill, 2005.

[24] Ogata, K., Modern Control Engineering, Prince Hall, P. 965, 2002.

[25] Silva, C. W. D., Modeling and Control of Engineering Systems, CRC Press, 2009.