IMPLEMENTACIÓN DE UN PROCEDIMIENTO DE NAVEGACIÓN EN FORMA GUIADA O AUTÓNOMA EN UNA PLATAFORMA MÓVIL EMPLEANDO FUSIÓN SENSORÍSTICA Y SOFTWARE LIBRE

Vazquez Raimundo[1], Demartino Bruno[1], Carpintero Diana[1], Colcombet Esteban[1], Fernandez Fernanda[1], Canali Luis[2].

GUDA-Grupo Universitario de Automatización[1]
Universidad Tecnológica Nacional - Facultad Regional Resistencia
French 414. 3500 Resistencia-Chaco. República Argentina
Tel: +54 362 4432928. / Fax: +54 362 4432683/
e-mail: ray_vazquez_2005@hotmail.com/ ray_vazquez_2005@yahoo.com.ar

UTN Facultad Regional Cordoba[2]
Maestro M. Lopez esq, Cruz Roja Argentina, Ciudad Universitaria, 5016
Córdoba, Argentina.
e-mail: luis.canali@gmail.com

IMPLEMENTACIÓN DE UN PROCEDIMIENTO DE NAVEGACIÓN EN FORMA GUIADA O AUTÓNOMA EN UNA PLATAFORMA MÓVIL EMPLEANDO FUSIÓN SENSORÍSTICA Y SOFTWARE LIBRE

Resumen: Utilizando la fusión de sensores de posición, efecto hall y una cámara de video se desarrolló un procedimiento para implementar algoritmos de navegación guiada o autónoma. El software desarrollado permitió a una plataforma de robot móvil trasladarse en un ambiente dinámico evitando colisiones y posibles accidentes con operarios. Se utilizaron bibliotecas de visión artificial y técnicas de control automático en tareas de navegación. El software de navegación guiada funcionó en forma satisfactoria en superficies de 80 m^2. Un GPS permitió implementar un programa de navegación autónoma en ambientes de trabajo superiores a 2500 m^2 debido a la baja exactitud del instrumento. El software desarrollado no requiere instalación y arranca desde un DVD. Los algoritmos desarrollados en la experiencia son de uso libre y código fuente abierto.

INTRODUCCIÓN

La teoría del control automático y la tecnología de los robots, permite combinar diversas disciplinas como por ejemplo mecánica, informática y electrónica con la finalidad de diseñar y construir aplicaciones, facilitando la experimentación en entornos de robots móviles. La integración de la arquitectura de control con las funciones de un sistema de medida [1] permite evaluar tareas de fusión sensorial. Normalmente un único sensor cubre un rango espacial reducido y fusionando los datos de varios transductores se puede conseguir una cobertura mayor. Debido a que un sensor necesita un tiempo determinado para obtener y transmitir una medida determinada, la fusión de manera eficiente de la información de varios sensores reduce su número.

Otro problema en el campo de la robótica es la incertidumbre. La incertidumbre depende del objeto que se está observando en vez del sensor. Se da cuando el sensor no puede medir todos los atributos relevantes para la percepción o cuando la observación es ambigua. Un único sensor es incapaz de reducir la incertidumbre en su percepción debido a su visión limitada del objeto. Al fusionar la información se reduce el conjunto de interpretaciones de una determinada medida. Cuando medidas independientes de la misma propiedad se fusionan midiendo una cualidad determinada con dos tipos de sensores, el sistema se vuelve menos vulnerable a interferencias [2]. En el siguiente trabajo se implementa procedimientos de navegación en forma guiada o autónoma a una plataforma de robot móvil

denominado RoMAA II mediante la fusión de la información proveniente de sensores de posición, magnéticos y una cámara de video.

ESPECIFICACIONES DE LA PLATAFORMA MÓVIL

El dispositivo denominado RoMAA II fue desarrollado en el Centro de Investigación en Informática para la Ingeniería [3], surge como respuesta a la necesidad de disponer de una plataforma sobre la cual ensayar y validar las investigaciones llevadas a cabo. Desde el inicio fue concebido con arquitectura abierta, pudiendo acceder a cada nivel jerárquico de su estructura para modificar o adaptar a los diferentes experimentos. La estructura es de tipo uniciclo con dos ruedas motoras y una rueda castor atrás. Las dimensiones son adecuadas para ambientes interiores de laboratorio; con posibilidad de montar diferentes tipos de sensores, actuadores y cámaras de video de manera sencilla en la parte frontal. El robot móvil se complementa con las herramientas de desarrollo denominados Stage-Player [4] y OpenCV [5].

HERRAMIENTAS DE DESARROLLO

El programa denominado Player-Stage es un esfuerzo internacional en producir herramientas de software libre para investigación y desarrollo en el mundo de la robótica. Se utiliza el lenguaje de programación C++ [6]. La elección de la plataforma Player facilita la forma de gestionar los controladores de hardware y la comunicación. Posee un servidor de dispositivos y un sockets TCP/IP, su estructura es multihilos, la lectura de sensores es transparente y contempla librerías oficiales en C y C++. Incluye drivers de los sensores, actuadores y robots comerciales como por ejemplo: Robots Pioneer, iRobot Roomba, Segway RMP y cámaras de video.
El Stage simula entornos virtuales con robots complementándose con el Player. Es decir, el Stage permite evaluar la eficiencia del algoritmo de navegación sin conectar la plataforma móvil en el ambiente real. El Stage dispone de muchos modelos de sensores y actuadores. El Player contiene el código fuente de las funciones encargadas de manejar y controlar el hardware del RoMAA II.
Un complemento indispensable del Player-Stage es la librería de visión artificial denominada OpenCV originalmente desarrollada por Intel. Se lo utiliza desde sistemas de seguridad con detección de movimiento, hasta aplicativos de control de procesos donde se requiere reconocimiento de objetos. Su licencia permite su uso en forma libre para propósitos comerciales o de investigación. El OpenCV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos, calibración de cámaras, visión estéreo y visión robótica. El proyecto encargado de desarrollar OpenCV pretende proporcionar un entorno fácil de utilizar y altamente eficiente. Para realizar tareas de acondicionamiento de señales y transmisión de información entre los sensores y la

notebook se emplea un dispositivo genérico denominado Student Full [7][8]. El núcleo del Student Full es un microcontrolador diseñado para realizar tareas de control y mediciones analógicas empleando un programa escrito en lenguaje C [9][10]. Finalmente se utiliza un framework informático denominado QT. El mismo es un desarrollo para aplicaciones en C++ y otros lenguajes de programación. El QT no sólo incluye un conjunto de librerías y herramientas, si no también una IDE específica que facilita el trabajo sobre dicho framework. Se lo configura para manejar interrupciones en el puerto de comunicaciones USB, facilitando el envío de información de los sensores al programa.

HIPÓTESIS DE TRABAJO

Es posible implementar algoritmos de navegación guiada o autónoma en una plataforma móvil articulando un software de robótica, visión artificial y sensores de proximidad y de efecto hall.

METODOLOGÍA

La teoría del control automático estudia la estabilidad de los sistemas y proporciona una metodología que permite definir criterios de seguimientos. En la figura 1 se visualiza un modelo simplificado encargado de seguir una banda magnética.

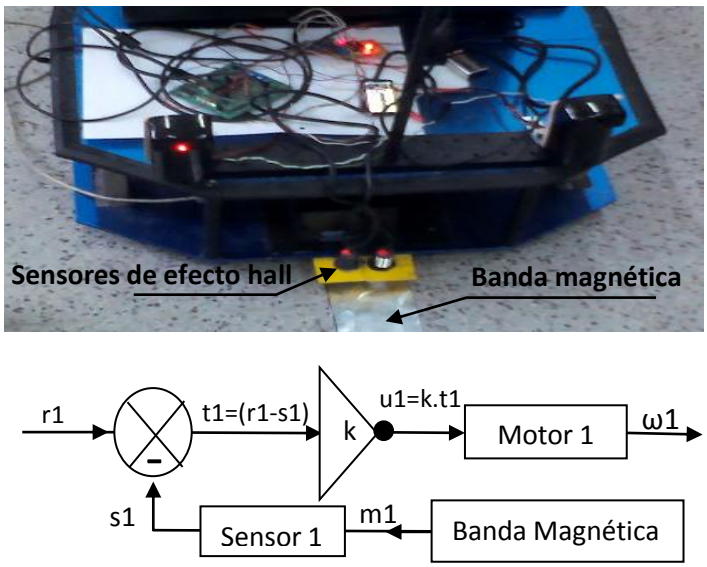

Figura 1. La plataforma móvil sigue el camino magnético empleando técnicas de retroalimentación.

El bloque denominado Motor 1 representa el motor izquierdo de la plataforma móvil. La variable ω1 es la velocidad angular de la rueda del robot. El bloque denominado Banda Magnética representa la cinta de aluminio colocado en el piso. El bloque Sensor 1 transforma la señal eléctrica m1 en digital s1. La variable t1 se obtiene sumando la constante de referencia r1 con la salida s1. El triangulo representa un amplificado de ganancia k negado. Finalmente u1 excita el motor de corriente continua, permitiendo el giro de la rueda izquierda cuando el sensor de efecto hall esta excitado. El motor derecho del robot es controlado bajo el mismo principio.
Es decir, si los dos sensores de efecto hall detectan la lámina metálica, el RoMAA II sigue la guía magnética. En la parte frontal superior de la plataforma se colocan dos sensores de proximidad denominados SP1 y SP2. Los mismos detectan objetos en frente del robot a una distancia no mayor de 1.5 metros.
La información obtenida de los sensores de efecto Hall y de proximidad es codificada y transmitida a la notebook vía USB empleando el Student Full. La librería de comunicación serial recibe los códigos y mediante una interrupción notifica al programa desarrollado en QT. El programa acepta la información disponible, interpreta los códigos y, mediante un algoritmo de navegación guiada, envía al Player las órdenes necesarias para llevar al RoMAA II a la posición deseada. El programa se detalla de la figura 2.

```
1   if (SP1 = 0 and SP2 = 0) then
2       if (s1 = 1 and s2 = 1) then
3           velocidadLineal = 0.07
4           velocidadAngular = 0
5       else if (s1 = 0 and s2 = 1) then
6           velocidadLineal = 0
7           velocidadAngular = -0.05
8       else if (s1 = 1 and s2 = 0) then
9           velocidadLineal = 0
10          velocidadAngular = +0.05
11      end if
12  else
13      velocidadLineal = 0
14      velocidadAngular = 0
15  end if
```

Figura 2. Los sensores reciben los posibles estado lógico.

De esta manera, si alguno de los sensores de proximidad detecta un objeto, el robot se detiene. En caso contrario avanza sobre la línea magnética. El algoritmo controla la velocidad lineal y angular de los motores de corriente continua. Detecta cuando los sensores de efecto hall salen de la banda magnética. En ese caso el programa realiza en forma automática la acción de giro necesaria para encontrar el camino.

NAVEGACIÓN GUIADA

El algoritmo descripto anteriormente no detecta un operario u objetos fuera del alcance de los sensores de proximidad. Se utiliza entonces visión artificial para detectar un cuerpo específico y su posición.
En una primera fase de prueba se experimenta con un chaleco fosforescente. El código fuente encargado de realizar las tareas de detección se obtuvo de las librerías del OpenCV[11]. Luego se articula el algoritmo de seguimiento desarrollado anteriormente con las funciones de detección de objetos de la librería de visón artificial como se muestra en la figura 3. La obtención y el procesamiento del video para detectar el chaleco se ejecutan concurrentemente con el programa de control principal mediante hilos e interrupciones. Así, se evita que el procesamiento de video aumente el tiempo de respuesta a los eventos externos, tanto de sensores como de la cámara.

Figura 3. Se configura el OpenCV para detectar el chaleco y su posición en la pantalla

De esta manera el sistema queda configurado para detectar el chaleco como se visualiza en la figura 4, es decir, las librerías de visión artificial permiten generar una condición de stop en la plataforma móvil.

Figura 4. El operario se interpone en la trayectoria del robot fuera del alcance de los sensores de posición.

Seguidamente se plantea una segunda condición en las tareas de navegación guiada. Se interpone en la trayectoria del robot una barrea móvil que detecta la condición de accidente entre el operario y el robot. Es decir, si una persona atraviesa la trayectoria guiada del RoMAA II y a la vez es detectado por la barrera móvil como se muestra en la figura 5, el dispositivo de control baja la barrera para interrumpir la movilidad del robot.

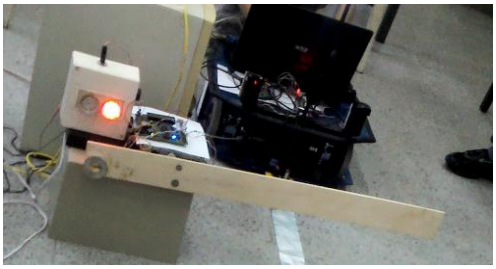

Figura 5. La barrera móvil detecta una persona que se desplaza perpendicular a la trayectoria del robot.

Como consecuencia, se enciende una luz roja en forma automática para indicar peligro de accidente. Los sensores frontales del RoMAA II detectan la barrera y el robot detiene su camino como se visualiza en la figura 6.

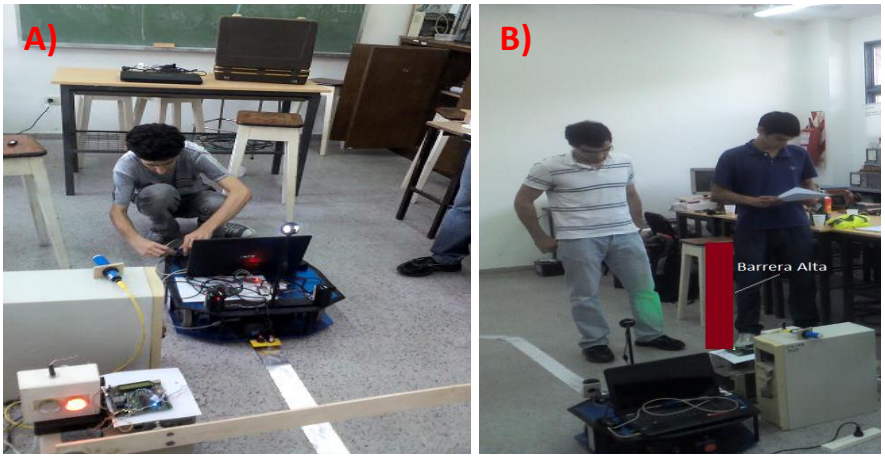

Figura 6. A) El sensor de la barrera móvil detecta al operario y baja la barrera, luego prende la luz roja y el robot se detiene. B) El sensor de la barrera móvil al no detectar al operario sube la barrera, prende la luz verde y el robot sigue su trayectoria.

El siguiente ejemplo el RoMAA II es simulado en el Stage esquivando un objeto situado en la vía magnética. En la simulación se necesitaron cuatro sensores de

proximidad, dos en el frente y uno en cada lado del RoMAA II. El sensor que detecte primero el objeto tiene prioridad en la dirección de giro. En la figura 7 el sensor conectado a la derecha detecta primero el obstáculo. Luego se activa el sensor lateral [12] que mide la distancia del robot a la superficie del objeto.

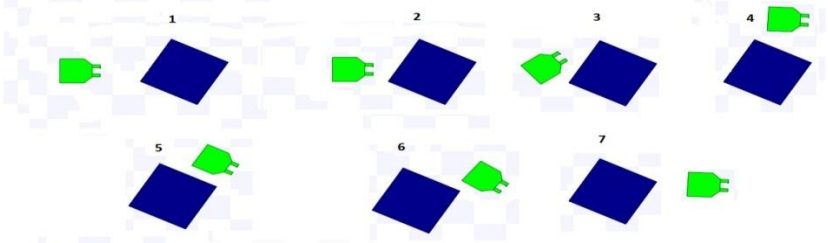

Figura 7. El simulador Stage utiliza el algoritmo de navegación para esquivar el objeto en su trayectoria.

Finalmente la plataforma móvil encuentra la vía magnética para continuar la navegación guiada.

NAVEGACIÓN AUTÓNOMA

El Stage tiene el instrumento de navegación denominado GPS de alta exactitud. De esta manera es posible establecer un punto de partida y otro de llegada mediante un algoritmo. El RoMAA II es representado de color verde y se dirige desde la posición de partida hacia la llegada a través de la trayectoria mostrada en la figura 8.

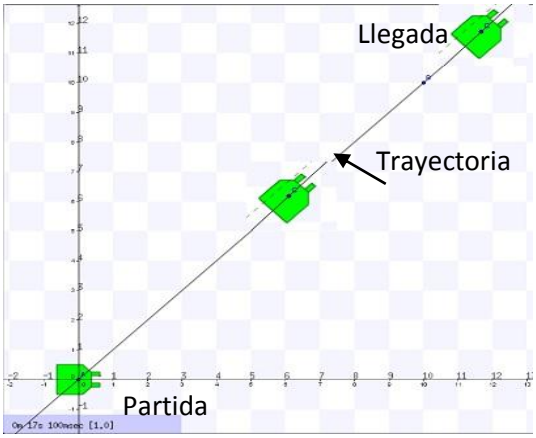

Figura 8. La plataforma móvil navega en forma autónoma mediante el instrumento de navegación denominado GPS.

El algoritmo de navegación es desarrollado teniendo en cuenta que el GPS disponible en el Stage da la ubicación instantánea del robot, pero no dice nada sobre la dirección del movimiento ni la forma de llegar al destino.

ALGORITMO DE NAVEGACIÓN AUTÓNOMA

El algoritmo consiste en aproximar la dirección real del robot a partir de la ubicación actual con la anterior. Con los datos obtenidos se puede establecer la dirección aproximada para guiar al robot a la meta.
El algoritmo se basó en geometría euclidiana, ya que dadas las condiciones del experimento no ofrece errores significativos. Los ángulos de las direcciones se obtienen utilizando la función arco tangente corregido, para contemplar el ángulo con el cuadrante correcto. Los ángulos necesarios se obtienen de la siguiente ecuación:

double λ = atan2(y - ultimaY, x - ultimaX);
double μ = atan2(destinoY - y, destinoX - x);

Donde λ es la dirección actual y μ la dirección que debería tener el robot para llegar al destino desde la posición actual. Luego se calcula el ángulo de rotación para alcanzar la dirección deseada mediante la resta $\theta = \lambda - \mu$.
El algoritmo descripto se implementa en C++ como se visualiza en la figura 9.

```
void RomaaController::procesarPosicion(double x, double y){
    if(!rotando){
        double lambda = atan2(y - ultimaY, x - ultimaX);
        double mu = atan2(destinoY - y, destinoX - x);
        double tita = lambda - mu;
        if(tita < -3.1416) {
            tita += 2 * 3.1416;
        }else if(tita > 3.1416){
            tita -= 2 * 3.1416;
        }
        qDebug() << "ANGULO: " << QString::number(tita, 'g', 9);
        if(tita > 0.1 || tita < -0.1){
            ultimaX = x;
            ultimaY = y;
            rotar(tita);
        }
    }
}
```

Figura 9. Código fuente encargado de direccionar al robot hacia la meta.

En la figura 10 se observa el ángulo de giro y la dirección de movimiento de la plataforma móvil empleando el algoritmo de la figura 9.

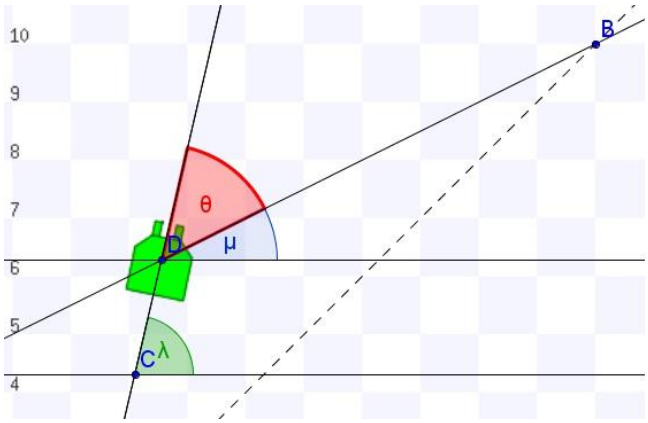

Figura 10. Ángulo de giro y dirección del movimiento del robot.

Al implementar el algoritmo en el Stage, tomado como partida el punto (0,0) y destino al punto (10,10), el robot describe la trayectoria observada en la figura 11.

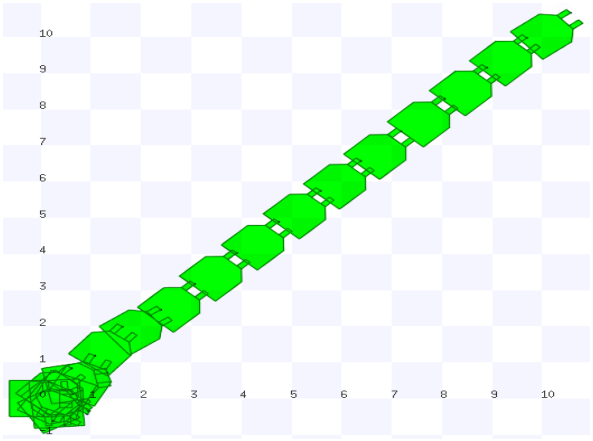

Figura 11. El robot al iniciar la navegación toma un camino aleatoria hasta encontrar la dirección correcta.

Como puede verse en un principio el robot toma posiciones arbitrarias, hasta que encuentra la dirección correcta para alcanzar el destino.
La ventaja en las tareas de depuración y puesta a punto de los algoritmos de navegación en Stage y Player es que los movimientos visualizados en la simulación es el obtenido en la plataforma móvil en forma real. El Stage posee dispositivos GPS con una exactitud elevada y la escala de la imagen en la experiencia fue de 80 m^2

representando el interior del laboratorio. En este trabajo se utilizo un GPS comercial con una exactitud del orden de dos metros. Por ese motivo las tareas de navegación en el laboratorio no son válidas debido al alto error. Para implementar el algoritmo en el GPS real es necesario agrandar la escala de la imagen de 80 m^2 a 2500 m^2. En esta condición el RoMAA II no puede seguir directamente la trayectoria ideal trazada en el ejemplo anterior debido al error del instrumento. En la figura 12 se observa un error entre la trayectoria y la navegación del robot debido a la exactitud de GPS real.

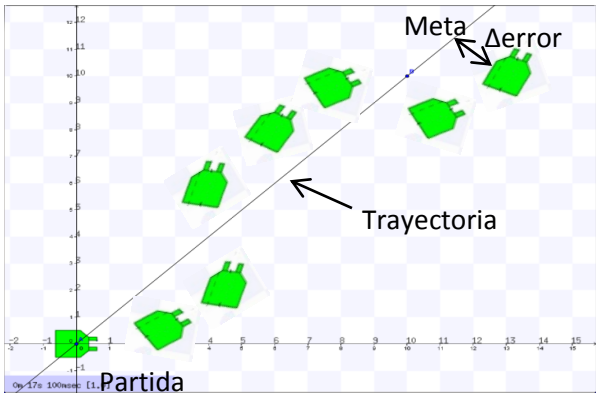

Figura 11. La trayectoria de navegación no coincide con el ideal debido al error del GPS real.

La ventaja que se obtiene al utilizar este tipo de dispositivo es que la trama de comunicación que transmite el GPS real al Player tiene un byte que identifica el sentido del movimiento. De esta manera cuando el robot arranca y se traslada en una dirección aleatoria el GPS real detecta el sentido del movimiento, permitiendo de esta manera corregir la orientación y ubicarlo hacia la meta.

CONCLUSIÓN

Utilizando la fusión de sensores de posición, efecto hall y una cámara de video se desarrolló un procedimiento para implementar algoritmos de navegación guiada y autónoma. De ésta manera se programó la plataforma móvil para que pueda trasladarse en un ambiente dinámico evitando colisiones y posibles accidentes con operarios. Existe una amplia disponibilidad de librerías y frameworks de software libre que gracias a las libertades que este otorga han sido ampliamente probados y mejorados, teniendo además una gran comunidad detrás. Esto hace que implementar nuevos sistemas implique simplemente buscar, comprender y ensamblar los distintos bloques disponibles. Así, los algoritmos desarrollados en este trabajo son concretizados y probados de una manera sencilla, sin requerir conocimientos elevados de los lenguajes C o C++ y abstrayéndose del hardware subyacente. El GPS utilizado

permitió implementar un algoritmo de navegación autónoma, pero teniendo un elevado margen de error entre la trayectoria real y la ideal. El algoritmo de navegación guiada funciona en forma satisfactoria en ambientes de trabajo superiores a 2500 m^2.

CONSIDERACIONES FINALES

Los programas desarrollados para realizar las tareas de navegación guiada o autónoma en este trabajo son de uso libre bajo la licencia GPLv3. Los mismos se pueden encontrar en [13], donde además se pueden hacer consultas o contribuciones.

REFERENCIA

1.Garcia, A,. Antón Álvarez, C.,:
Instrumentación Electrónica. Editorial Thomson. Cap 1. pp 10. España. Madrid 2011.
2.Baturone, A.,:
Robótica Manipuladores y Robot Móviles. Editorial Marcombo. Cap 1. pp 6. Tercera edición. España 2010. Madrid.
3. http://ciii.frc.utn.edu.ar/Robotica/RobotsRoboticaWeb
4. www.playerstage.sourceforge.net/
5. www.es.wikipedia.org/wiki/OpenCV
6. Garay, J., Bronson S.:
C++ para Ingeniería y Ciencia. Editorial Thomsom 2002. Madrid. España.
7.Vazquez R,. Robledo A,. Toledo P,. Mason L,. Mariguetti J,. Canali Luis.
Desarrollo de un Procedimiento para Diseñar Dispositivos Genéricos Multipropósitos. Jornadas de investigación en ingeniería del NEA y países limítrofes, ISBN: 978-950-42-0142-7
8.Saravia, B., Coria J., Fiadino, G,. Arroldi A.,:
MPlab X y Técnicas de Programación en Librerías de Microchip. Cap 3. pp 55. Primera Edición 2011. Buenos Aires. Argentina.
9.Santos M,. Patiño I,. Carrasco R,. :
Fundamento de programación. Primera Edición 2006. Cap 5. p.97. Editorial Alfaomega Rama. México. México.
10.Usategui, J., Ruiz Etebarria A., Martínez Angulo, I., Parra Trueva, I.,:
dsPIC Diseño prácticos de aplicaciones. Editorial Mc Graw Hill. Aplicación 0. pp 175. España 2006. Madrid.
11. https://github.com/Itseez/opencv/blob/master/samples/cpp/camshiftdemo.cpp
12. Palacios, E., Remiro F., López, J.,:
Microcontroladores PIC 16f84. Desarrollo de Proyectos. Cap 32. pp530. Alfaomega Ra-Ma. Segunda Edición. España 2006.Madrid.
13. http://sourceforge.net/projects/romaa/