Artículo **Revista de Ingeniería Eléctrica**
Diciembre 2017 Vol.1 No.1, 27-40

Seguimiento de muros con robot Koala usando algoritmo Bugs

GIL-VÁZQUEZ, Alejandro †*, GARCÍA-RAMÍREZ, Rubén Senén y MARTÍNEZ-RAMÍREZ, Violeta

Recibido 2 de Agosto, 2017; Aceptado 28 de Septiembre, 2017

Resumen

El presente artículo describe los temas considerados para realizar el seguimiento de muros con autonomía por el robot Koala (robot móvil de ruedas) mediante la implementación de un software utilizando algoritmos de evasión de obstáculos (Bugs). La evitación de obstáculos se enfoca en cambiar la trayectoria del robot, según lo informen sus sensores durante el desplazamiento del mismo. En los algoritmos de evitación de obstáculos las lecturas de los sensores del robot juegan un papel importante en la trayectoria futura. El algoritmo Bug representa una técnica la cual utiliza los valores más recientes de los sensores del robot, que lo acercan a la posición objetivo.Se expresan algunos resultados de pruebas realizadas con el robot Koala, se anexa un fragmento de los valores de los sensores del robot junto con la gráfica del trayecto correspondiente obtenida mediante el software implementado con los cuales se demuestra la nitidez del seguimiento del muro.

Robot Diferencial, Seguimiento de muros

Abstract

The present article describes the topics considered for the monitoring of walls with autonomy by the Koala robot (mobile robot of wheels) by means of the implementation of a software using algorithms of evasion of obstacles (Bugs). The avoidance of obstacles focuses on change the path of the robot, as reported by its sensors during the movement of the same. In obstacle avoidance algorithms the readings of the robot sensors play a role important in the future. The Bug algorithm represents a technique which uses the most recent values of the robot's sensors, which bring it closer to the target position. Some results of tests performed with the Koala robot are expressed, a fragment of the values of the sensors of the robot together with the graph of the corresponding path obtained through the implemented software with which the clearness of the wall follow-up is demonstrated.

Differential Robot, Monitoring of walls

Citación: GIL-VÁZQUEZ, Alejandro, GARCÍA-RAMÍREZ, Rubén Senén, MARTÍNEZ-RAMÍREZ, Violeta. Seguimiento de muros con robot Koala usando algoritmo Bugs.Revista de Ingeniería Eléctrica. 2017. 1-3:27-40.

† Investigador contribuyendo como primer autor.
*Correspondencia al Autor Correo Electrónico: ing.gil@ittlahuac.edu.mx

Introducción

Un robot es un mecanismo constituido por dispositivos electrónicos, mecánicos y de computación, está formado por: software y hardware. El primero da autonomía al robot y controla el mecanismo. Esta autonomía es la que distingue a un robot de otras formas de automatización. Existen básicamente dos tipos de robots: los fijos, su giro es principalmente industrial y realizan tareas tales como: ensamble de coches, pintura, embarque, entre otras, trabajando en ambientes altamente controlados para los cuales fueron diseñados. El otro tipo de robots se refieren al caso de la robótica móvil, los robots se desplazan en su entorno. Uno de los retos consiste en caracterizar dicho entorno a través de sensores, identificar obstáculos y zonas de paso, e incluso ubicarse con la mayor precisión posible con respecto a un sistema de referencia dado (López García, 2011).

La robótica autónoma móvil se ha desarrollado principalmente para hacer posible la realización de tareas de alto nivel por parte de máquinas sin la necesidad del control humano. A tales efectos, los robots deberían tener la habilidad de moverse apropiadamente en un mundo real. Por lo tanto, planificar sus propios movimientos se vuelve uno de los problemas más importantes a ser resuelto en el diseño de robots autónomos móviles (Benavides F., 2012). Él enfoca la planificación de movimientos para robots móviles con dos ruedas y control diferencial, que se desempeñan en entornos estáticos bidimensionales a partir de la construcción de un mapa de ruta empleando diagramas de Voronoi. Su objeto de pruebas, fue un robot móvil de ruedas el Khepera III de la empresa K-Team, con sensores IR, ultrasónicos y puertos serial, USB y Bluetooth. El entorno que utilizó fue:

Mono-agente :	Solo hay un robot moviéndose en el entorno.
Parcialmente observable:	Utiliza un sistema de visión global, que brinda información con un nivel no despreciable de incertidumbre con algo de ruido en la información, la cual puede afectar el rendimiento del robot.
Estático:	Dado que el robot es el único objeto móvil el entorno no cambia mientras se elige la otra acción.

En la prueba de su propuesta, la planificación de movimientos se realiza una única vez, a partir de la primera imagen asumiendo que los obstáculos son estáticos. Las siguientes imágenes solo se procesan para obtener la ubicación del robot. El sistema de navegación (sensores) del robot no fue explotado mínimamente.

Quintero P., et al., (2010) aborda el problema de la planificación de trayectorias de robots móviles no holonómicos en ambientes congestionados de objetos. Se basa en una representación de los objetos en el espacio de velocidades del robot, llamada Polígono de Velocidades Admisibles (PVA) del robot y una ley de control para limitar las velocidades que el robot puede alcanzar. Los resultados presentados todos fueron simulados.

Ying L., (2016) propone una nueva estrategia de seguimiento de muros para robots móviles. Esta estrategia establece el modelo matemático de auto-convergencia que logra ejecutar la actividad de seguimiento de la pared con sólo un único conmutador de proximidad de distancia.

ISSN-2523-2517

Sus ventajas sobre las anteriores son: la evitación de la interferencia mutua entre los sensores y la reducción del coste alto del hardware. Muestra resultados experimentales y sus posibles aplicaciones.

El proyecto propuesto en el presente documento, considera al robot Koala RMR (robot móvil de ruedas) que se probará en entornos de trabajo *desconocidos*, es decir, se omite cómo estarán dispuestos los obstáculos, el espacio de configuraciones será *estático*, puede cambiar entre ejecuciones. Siendo su principal herramienta de navegación los sensores *infrarrojos*, quienes serán los que suministren la información a la aplicación que se desarrollará y tendrá como función principal dar autonomía al RMR.

Se desempeñará en entornos *reales* no simulados, en el que se basan prioritariamente, los trabajos de Quintero P., et al., (2010) y Ying L., (2016) respectivamente; se harán más consideraciones físicas que beneficiarán directamente los movimientos precisos del robot. Además, este proyecto se diferencia respecto de Benavides F., (2012) en donde las rutas son calculadas previo al desplazamiento del robot gracias a la correcta captura de imagen del entorno, que deberá ser, obligatoriamente, en condiciones de luminosidad suficientes; esa condición de luminosidad, no es imprescindible en el Koala ya que, bastará solo con el 10% de ella para que los sensores del koala trabajen correctamente.

El Koala es un robot móvil de ruedas (RMR), con tres ruedas por lado, de dimensiones: Largo 32 cm. x Ancho 30 cm. x Alto 20 cm., con sencillez de uso, mayor potencia computacional y una mejor comunicación con el procesador.

En general, es un robot dotado de bondades las cuales necesitan explorarse, dominarse y utilizarse a través de aplicaciones que interactúen con su plataforma y obtener dividendos que faciliten: el trabajo, objetivos o retos de los usuarios (K-Team S.A., 2001).

El presente documento describe el diseño e implementación del software de control que integra bondades de la técnica de los algoritmos Bugs para permitir el proceso de navegación del robot Koala en entornos de trabajo cerrados de manera autónoma. En las siguientes secciones se describen los sensores incorporados al robot Koala, la cinemática del robot móvil de ruedas, técnicas de los algoritmos Bugs, así como la solución desarrollada y las Conclusiones.

Se presentan los sistemas más simples de evitación de obstáculos que son utilizados con éxito en robótica móvil. El algoritmo Bug representa una técnica que utiliza los valores más recientes de los sensores del robot, que lo acercan a la posición objetivo.

Sensores

Una de las tareas más importantes de un sistema autónomo de cualquier tipo es adquirir conocimientos sobre su entorno. Esto se hace tomando mediciones utilizando diversos sensores y luego extraer información significativa de esas mediciones.

Detalles sobre los sensores para robots

Existe una amplia variedad de dispositivos diseñados para percibir la información externa de una magnitud física y transformarla en un valor electrónico que sea posible introducir al circuito de control, de modo que el robot sea capaz de cuantificarla y reaccionar en consecuencia.

ISSN-2523-2517

Un sensor consta de algún elemento sensible a una magnitud física —ejemplo: la intensidad o color de la luz, temperatura, humedad, etc. — y debe ser capaz, por sus características propias, o por medio de dispositivos intermedios, de transformarla en un cambio eléctrico que pueda utilizarse, finalmente, para el control del robot.

Rayos infrarrojos

Los Rayos Infrarrojos (IR) tienen una cara frontal en la cual se encuentran tanto la Luz Emitida por un Diodo (LED) como el fototransistor. Debido a que no están colocados en forma enfrentada, la única forma posible para que la luz generada por el LED active el fototransistor es haciendo reflejar esta luz en una superficie reflectiva. Estos sensores son muy útiles para detectar, por ejemplo, una línea negra sobre una superficie blanca o viceversa (Figuras 1 y 2).

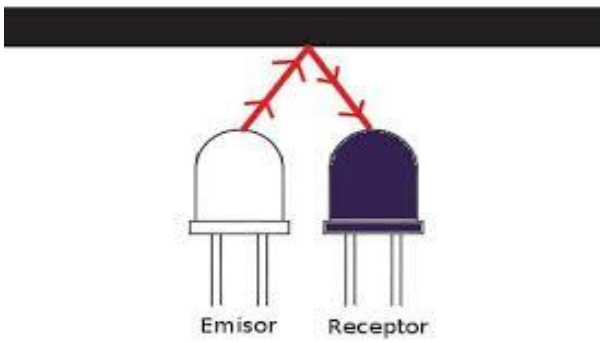

Figura 1 Sensor infrarrojo en configuración réflex

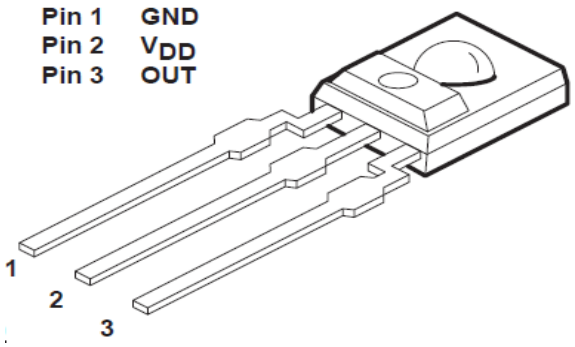

Figura 2 Sensor instalado en el robot Koala, TSL252 de la Texas Instruments

Encoders

Los encoders son codificadores angulares de posición utilizados para determinar la posición (angular) de las articulaciones de los robots. Principalmente se usan dos tipos: el incremental y el absoluto (Barrientos, 2007).

Encoders incrementales

Los codificadores ópticos o encoders incrementales constan, en su forma más simple, de un disco transparente con marcas opacas colocadas radialmente y equidistantes entre sí; de un sistema de iluminación en el que la luz es colimada correctamente, y de un elemento fotorreceptor. El eje cuya posición se quiere medir va acoplado al disco transparente (Figura 3).

Figura 3 Encoder incremental

ISSN-2523-2517

Con esta disposición, a medida que el eje gire, se irán generando pulsos en el receptor cada vez que la luz atraviese cada marca, y llevando la cuenta de estos pulsos es posible conocer la posición del eje. Existe, sin embargo, el problema de no saber si en un momento dado se está realizando un giro en un sentido o en otro con el peligro que supone no estar contando adecuadamente (Barrientos, 2007).

Encoders absolutos

El funcionamiento básico de los encoders absolutos es muy similar al de los incrementales. Se tiene una fuente de luz con sus lentes de adaptación correspondientes, un disco graduado y unos fotorreceptores. En este caso, el disco transparente se divide en un número determinado de sectores (potencia de 2), codificándose cada uno de ellos según un código binario cíclico (normalmente código Gray) que queda representado por zonas transparentes y opacas dispuestas radialmente tal y como se aprecia en la Figura 4.

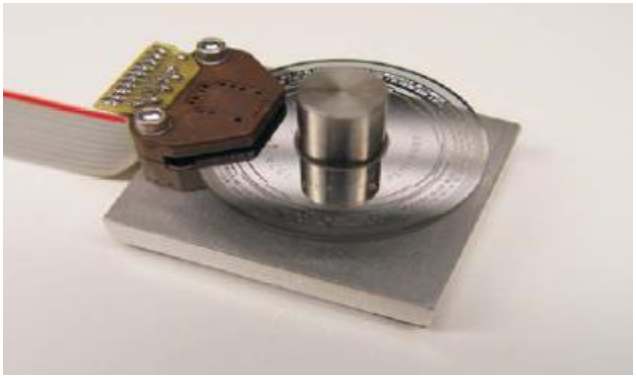

Figura 4 Encoder Absoluto

No es necesario ningún contador o electrónica adicional para detectar el sentido de giro, pues cada posición (sector) es codificado de forma absoluta. Su resolución es fija, y vendrá dada por el número de anillos que posea el disco graduado.

Resoluciones habituales van desde 2^8 a 2^{19} bits (desde 256 a 524,288 posiciones distintas).Los encoders pueden presentar problemas mecánicos debidos a la gran precisión que se debe tener en su fabricación. Son dispositivos particularmente sensibles a golpes y vibraciones, estando su margen de temperatura de trabajo limitado por la presencia de componentes electrónicos (Barrientos, 2007).

La cinemática del robot móvil de ruedas

Un robot móvil es un sistema en el que cabe identificar diversos subsistemas de percepción, planificación, control de movimientos y locomoción que interaccionan entre sí. El sistema de percepción permite que el robot sea capaz de hacer frente a situaciones cambiantes del entorno, así como a reaccionar ante posibles eventos inesperados mientras navega, lo que exige la utilización de un sistema sensorial que suministre la información del entorno. Esta información requerida debe permitir al robot realizar tres tareas fundamentales: estimar su posición y orientación, mantener actualizado el mapa del entorno y detectar los posibles obstáculos.

Para que un robot móvil pueda satisfactoriamente afrontar tareas como generar trayectorias, evitar obstáculos, monitorizar la ejecución, etc. se requiere que éste sea capaz de determinar su localización (posición y orientación) con respecto a un sistema de referencia absoluto. De forma general, determinar la localización de un robot móvil equivale a encontrar las componentes de translación (tx, ty, tz) y de rotación (θx, θy, θz) del sistema de coordenadas solidario al robot (y por tanto móvil) con respecto a un sistema absoluto. En particular, en este trabajo se considera el caso bidimensional (con mucho el más común en las aplicaciones actuales de los robots móviles), donde el robot se mueve con tres posibles grados de libertad.

ISSN-2523-2517

Así, el problema se reduce a encontrar la terna (tx, ty, θ) asociada al sistema móvil del vehículo, donde (tx, ty) representa su posición y θ representa su orientación.

Robots con ruedas

Las ruedas del robot móvil se mueven por el contacto superficial (o fricción con la superficie), idealmente, se desplaza $2\pi r$ por vuelta (Figura 5) (Ruiz del Solar, 2006).

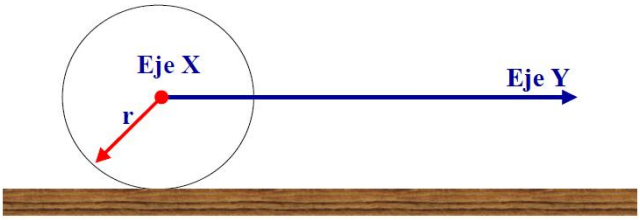

Figura 5 Esquema básico de giro de la rueda

Los robots móviles con ruedas pueden tener varias de éstas, lo que implica distintos modos de girar. Las formas más conocidas de disponer las ruedas en los robots móviles:

– Diferencial
– Síncrona
– Tipo triciclo
– Tipo carro

Configuración Diferencial

Este esquema es uno de los más sencillos, consiste de dos ruedas en un eje común, donde cada una es controlada por un motor, de tal forma que el giro del robot queda determinado por la diferencia de velocidad de las ruedas. Sus movimientos son:

– Línea Recta
– Cambio en la orientación(ángulo)
– Vuelta sobre su propio eje

Para que el robot se desplace en línea recta conservando su orientación (ángulo) las ruedas deben girar a la misma velocidad. Para que el robot cambie su orientación debe existir una diferencia de velocidades en las ruedas, mientras más grande sea la diferencia de velocidades en las ruedas más grande será el cambio en la orientación del robot.

Este esquema utiliza una o dos ruedas adicionales para mantener el balance, estas formas tienen diferentes nombres dependiendo de las ruedas por ejemplo: con 3 ruedas se denominan triángulo, el cual puede presentar problemas de estabilidad; con 4 ruedas se conoce como diamante la pérdida de contacto de las ruedas de tracción hace requerir de un sistema de suspensión (Figura 6). Para que el movimiento sea recto se requiere que las ruedas vayan a la misma velocidad (Ruiz del Solar, 2006).

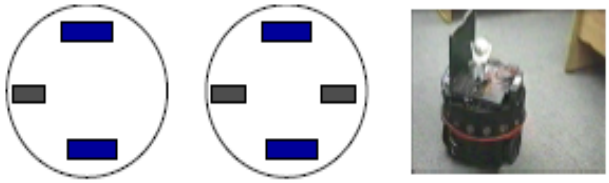

Figura 6 Esquema diferencial con 3 y 4 ruedas

La conducción diferencial es la elección cinemática más común, en la cual, la diferencia en las velocidades de las ruedas "determina su ángulo de giro". Sin embargo, la cinemática se plantea un par de preguntas básicas, teniendo en cuenta las velocidades o posiciones de la rueda: ¿cuál es la velocidad/posición del robot?, ¿Existen limitantes inherentes del sistema? (Siegwart, 2004).El centro de curvatura instantáneo (CCI) es el punto alrededor del cual el robot gira (Figura 7). Cuando ocurren giros se presentan deslizamientos en las ruedas, para minimizarlos, este punto (CCI) debe estar en la intersección de los ejes de las ruedas.

ISSN-2523-2517

Cada rueda debe viajar a la misma velocidad angular (ω) alrededor de CCI.

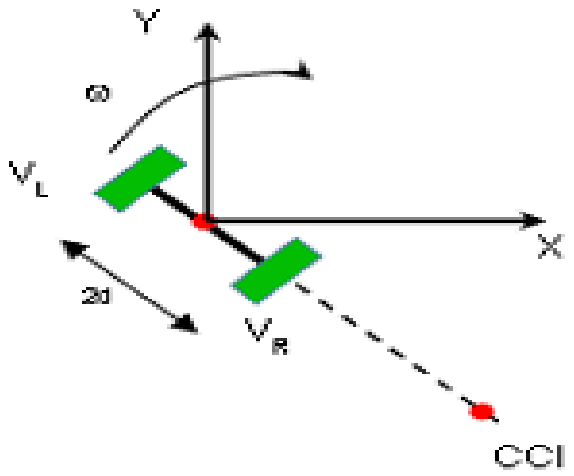

Figura 7 Centro de curvatura instantáneo

Para determinar la velocidad del robot alrededor de CCI y su velocidad lineal (Figura 8), se tiene:

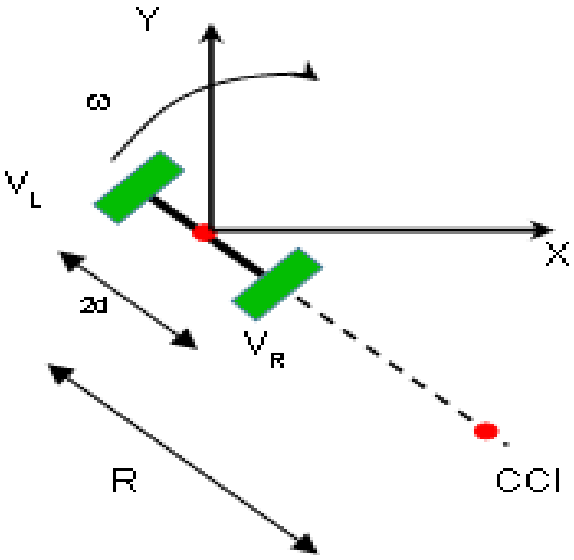

Figura 8 Esquema general del vehículo

$$\omega (R + d) = V_L \quad (1)$$
$$\omega (R - d) = V_R \quad (2)$$

Así,

$$\omega = (V_R - V_L) / 2d \quad (3)$$
$$R = d (V_R + V_L)/(V_R - V_L) \quad (4)$$

Así, la velocidad del robot es:

$$V = \omega R = (V_R + V_L)/2 \quad (5)$$

Para conocer la posición del robot en un instante de tiempo t (Figura 9), se integran las componentes de la velocidad:

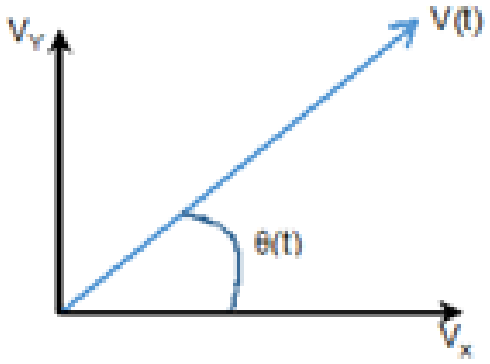

Figura 9 Componentes de la velocidad.

$$V_X = V(t) \cos(\theta(t)) \quad (6)$$
$$V_Y = V(t) \operatorname{sen}(\theta(t)) \quad (7)$$

Integrando, se obtiene:

$$x(t) = \int V(t) \cos(\theta(t))\, dt \quad (8)$$
$$y(t) = \int V(t) \operatorname{sen}(\theta(t))\, dt \quad (9)$$
$$\theta(t) = \int \omega(t)\, dt \quad (10)$$

Ecuaciones cinemáticas:

$$\omega = (V_R - V_L) / 2d$$
$$R = d (V_R + V_L) / (V_R - V_L)$$
$$V = \omega R = (V_R + V_L) / 2$$

ISSN-2523-2517

Evasión de obstáculos

La evitación de obstáculos se enfoca en cambiar la trayectoria del robot, según lo informen sus sensores durante el desplazamiento del mismo. El movimiento resultante está en función de las lecturas recientes de los sensores y su *posición meta* y la ubicación con respecto a ésta posición.

Los algoritmos de evitación de obstáculos dependen en mayor o menor grado de la existencia de un mapa global y el conocimiento preciso del robot de su ubicación con respecto al mapa. El algoritmo Bug representa una técnica tal que sólo se utilizan los valores más recientes de los sensores del robot, que lo aproximan a la *posición meta*.

Algoritmos Bugs

Muchos algoritmos de planificación asumen el conocimiento global del ambiente. Los algoritmos Bug asumen sólo el conocimiento local del medio ambiente y un objetivo global. Los comportamientos de este tipo de algoritmos son simples:

- Seguir un muro (derecha o izquierda).
- Moverse en línea recta hacia la meta.

Los algoritmos Bug consideran sensores táctiles (de contacto).

El problema básico de Planificación de Movimientos (PM) es calcular un camino libre de colisiones desde una posición inicial hasta una final.

En algoritmos Bug: no existe un modelo global del espacio de trabajo, los obstáculos son desconocidos, sólo la información local se adquiere a través del proceso de sensado (Figura 10).

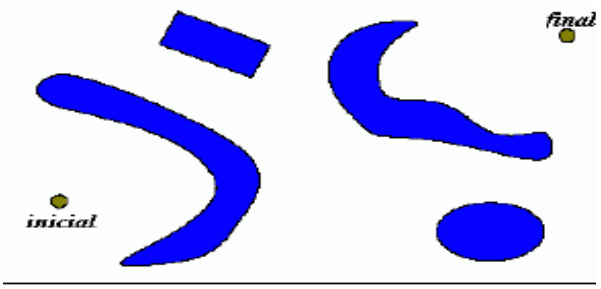

Figura 10 Espacio de trabajo

Procedimiento: El algoritmo Bug1 es ejecutado en algún punto de una trayectoria continua, la Figura 11 demuestra el comportamiento de un autómata móvil. Genera una trayectoria desde el inicio(S) hasta la meta (T). Cuando un i-ésimo obstáculo es encontrado, el autómata móvil define un punto de llegada H_i, i=1,2,…, cuando se sale del i-ésimo obstáculo, continua el viaje hacia la meta, el autómata móvil define un punto de salida L_i; inicialmente, i=1, L_0= Inicio (S). El procedimiento utiliza tres registros, R_1, R_2, R_3, para almacenar información intermedia. Los tres son inicializados a cero cuando un nuevo punto de llegada, H_i, es encontrado.

Específicamente, R_1 es utilizado para almacenar las coordenadas del punto actual, Q_m, de la mínima distancia entre el borde del obstáculo y la meta; R_2 integra la longitud del borde del obstáculo iniciando en H_i; y R_3 integra la longitud del borde del obstáculo iniciando en Q_m (en el caso de existir muchas opciones para Q_m cualquiera de ellas puede tomarse). El procedimiento consiste de los siguientes tres pasos:

1) Desde el punto L_{i-1}, desplazarse en línea recta hacia la meta (T) hasta que uno de lo siguiente ocurra:

a) La meta (T) sea alcanzada. El procedimiento termina.

ISSN-2523-2517

b) Un obstáculo es encontrado y un punto de llegada, H_i, es definido. Ir al paso 2).

2) Utilizando la dirección local, seguir el borde del obstáculo. Si se alcanza el objetivo, detenerse. Después de haber recorrido todo el obstáculo y haber regresado a H_i, definir un nuevo punto de salida $L_i = Q_m$. Ir al paso 3).

3) Verificar la accesibilidad de la meta (T). Si la meta no es accesible el procedimiento termina. De lo contrario, utilizar el contenido de los registros R_2 *y* R_3, para determinar el camino más corto a lo largo del borde hacia L_i, y utilizar esto para obtener la nueva L_i; hacer i = i +1 e ir al paso 1).

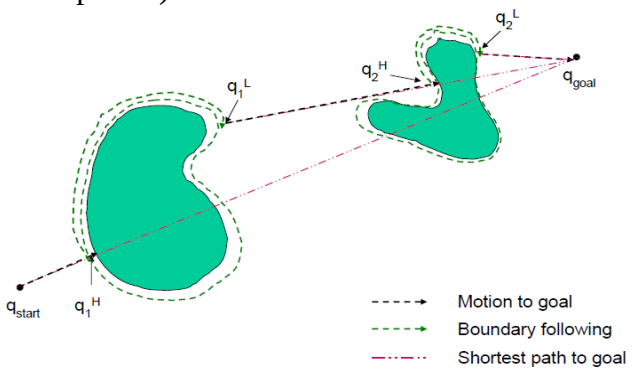

Figura 11 Trayectoria del autómata móvil (línea punteada), obstáculos (ob1, ob2), puntos de llegada (H1, H2), puntos de salida (L1, L2)

Tangent bugs

Este algoritmo presenta dos comportamientos básicos:

– Movimiento hacia el Objetivo
– Obstáculo límite-siguiente

En cualquier caso, el robot construye el Grafo Local Tangente (GLT) basado en el rango de lecturas actuales, y usa el GLT como sigue. Durante el movimiento hacia el objetivo el robot se mueve en la Dirección Optima Local, la cual es la dirección del camino más corto al objetivo. Sea la función d (w, T) medida de la distancia euclidiana de un punto w de T, de tal manera que los puntos w pertenecen al espacio libre. El robot se mueve hacia el objetivo hasta que se encuentra atrapado en la cuenca de atracción de un mínimo local de d (w, T). Entonces eso cambia el comportamiento límite-siguiente.

El robot elige una dirección límite-siguiente, y se mueve a lo largo del contorno del obstáculo mientras usa el GLT para hacer atajos locales. Pero el robot no puede dejar el contorno antes de que se cumpla la siguiente condición de salida. Mientras el robot sigue el contorno, registra la distancia mínima hacia el objetivo $d_{min}(t)$ observado hasta el momento a lo largo del contorno del obstáculo.

El algoritmo Tangent Bug trabaja con información limitada acerca del entorno. En este algoritmo, el robot solamente conoce acerca del inicio y la posición objetivo. No tiene información acerca de donde están los obstáculos. Solamente información extra proviene de los sensores de rango los cuales dan los puntos de los obstáculos que están dentro del rango del sensor.

El algoritmo Tangent Bug consiste de dos diferentes modos o acciones: 1) Movimiento hacia el objetivo, 2) Siguiendo muro (u obstáculo).

ISSN-2523-2517

Modo Movimiento hacia el objetivo.

En este modo el robot puede moverse sobre la M-línea, la cual es una línea virtual entre la posición del robot y el objetivo. Este modo se activa bajo 2 condiciones: La primera es que no debe haber ningún obstáculo en el rango del sensor entre la posición del robot y el objetivo. El robot puede moverse libremente a lo largo de la dirección hacia el objetivo. En la Figura 12, el movimiento entre (2,3) y (4, 3.5) es el ejemplo del modo "Movimiento hacia la meta". Entre estos puntos, no hay obstáculo entre el robot y la meta.

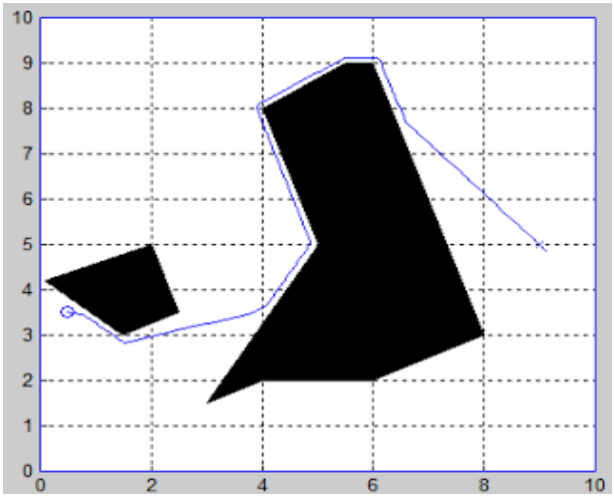

Figura 12 Plan de la ruta

La segunda condición ocurre cuando el robot encuentra un obstáculo. Para este caso, el robot obtiene información de la distancia desde el sensor. El sensor rastrea el ángulo especificado y decide si hay un punto del obstáculo. Si el sensor es capaz de rastrear, por ejemplo 180 grados, entonces más de un punto del obstáculo podría encontrarse dentro del rango del sensor. Para decidir hacia donde ir, la distancia estimada de estos puntos se calculan por la fórmula (11).

Después de encontrar todos los valores de los puntos del obstáculo, el robot se mueve hacia el punto del obstáculo con el valor más pequeño estimado.

Valor Estimado = Distancia (Posición del robot, Meta) + Distancia (Punto del Obstáculo, Meta)

Modo Siguiendo un Muro

Este es un modo excepcional. Cuando el robot se mueve "siguiendo un límite", en un punto donde se dan cuenta de que el valor estimado comienza a aumentar después de disminuir. Esto demuestra que en algún momento hay un mínimo local. Con el fin de evitar la oscilación alrededor de mínimos locales, Este modo se activa. El robot comienza a seguir el límite del obstáculo (Figura 13).

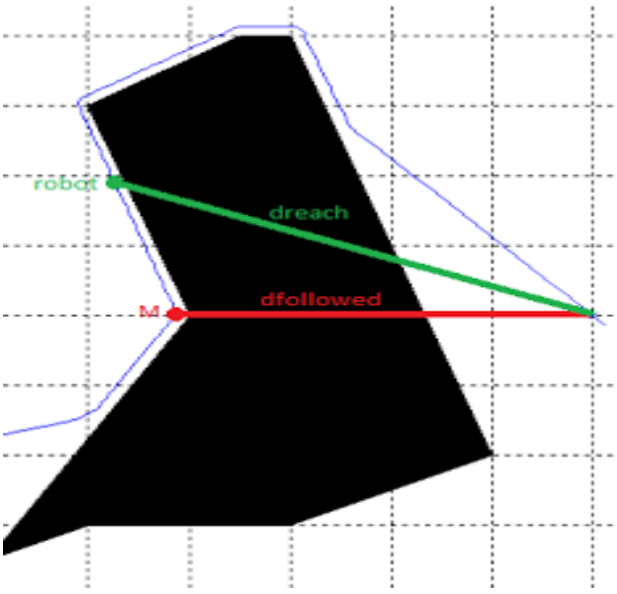

Figura 13 Modo Siguiendo un Muro

La condición de paro para este modo, está dada por la relación:

dreach < *dfollowed*

dreach	:	Distancia más pequeña entre el robot y la meta.
dfollowed	:	Distancia entre el punto M (mínimo local) y la meta.

Solución

¿Cómo se hace el movimiento?

El Robot Koala se desplaza de manera autónoma gracias al software de control que se implementó para darle funcionalidad. Se distinguen los siguientes módulos diseñados que forman parte fundamental del software de control:

1) Comunicaciones (E/S a través del puerto serie de la computadora): en éste, se configuran los puertos de la computadora y del Robot Koala, estableciendo el canal de comunicación entre los dispositivos.

2) Conversión e interpretación de la información del exterior: La información de los valores de proximidad proporcionada por los sensores IR del robot Koala durante su traslado en el entorno, es tratada para su respectivo análisis, interpretación y almacenamiento.

3) Reacción y Movimiento del robot: Generar instrucciones de acción basadas en la información del entorno que el robot Koala va reportando, permiten definir los movimientos de desplazamiento del móvil que lo hacen capaz de evitar obstáculos que encuentre en su recorrido.

4) Trazo de la trayectoria del robot Koala: Construcción de la Gráfica del trayecto, utilizando los valores de trazado del robot sobre la ruta en el entorno de prueba.

El tercer módulo está enfocado completamente en hacer que el Robot Koala "encuentre y siga un muro" desplazándose a lo largo de este mientras no descubra obstáculos, los cuales evitará en cuanto los detecte a través de sus sensores IR.

Códigos

El módulo de Control de Reacción y Movimiento ocupa las lecturas obtenidas del exterior por los sensores, para obtener: Posición y Orientación, mediante la aplicación de las ecuaciones correspondientes, para determinar si en el entorno hay algún objeto próximo al robot, que tan cerca, cuál es su orientación, etc. Si se encuentra cerca la orden para el Koala será disminuir la velocidad, si el objeto está por enfrente, por un lado o por el otro, la orden será ir hacia el sentido opuesto al objeto, siempre y cuando el sentido no esté bloqueado.

De ser afirmativo, nuevamente se verifica velocidad del Koala y orientación respecto del objeto. Al final, el software de control estructura la instrucción adecuada que se enviará a través del puerto y se ejecute en el Koala. En la figura 14, se muestra el algoritmo correspondiente.

```
Inicio
  Leer buffer desde puerto serie
  Con la lectura de proximidades del Koala:
     Obtener la proximidad mayor
  Calcular Dirección, donde se estima, existe un obstáculo (mediante Formula Lebetel)
  Si está próximo a algún obstáculo
  Entonces: Modificar velocidad del Koala según la Dirección/Orientación del obstáculo
  Sino:        // Verificar que mantiene su distancia respecto del muro por la Derecha
```

```
Si  Separación del Koala con muro Derecho está dentro de los límites establecidos
    Entonces:
        Asignar velocidad igual a motores del Koala manteniendo Dirección/Orientación
    Sino:
        Si  es muy cerca del muro Derecho
        Entonces:  Modificar velocidad en motores para separarlo del muro
        Sino:
            Si se está alejando del muro Derecho
            Entonces:  Modificar velocidad en motores para acercarlo al muro
            Sino:  Calcular…
                Velocidad de Traslación según proximidad de obstáculo
                Velocidad Angular ocupando Dirección/Orientación del obstáculo
                Velocidades de motores Derecho e Izquierdo en términos de la Velocidad de
                    Traslación y Angular.
            Fin_Si
        Fin_Si
    Fin_Si
Fin_Si
Validar velocidades de motores, no rebasen el máximo permitido
Vel_Traslacion  = ( Vel_DER + Vel_IZQ ) / 2
Vel_Angular  = ( Vel_DER - Vel_IZQ ) / LONG_EJES
Almacenar lecturas de proximidades en archivo
Generar con las velocidades Derecha e Izquierda el comando para robot Koala
Enviar comando, a través de Puerto Serie, para ser ejecutado por Koala.
Calculo de diferenciales por cada concepto (Tiempo, Ángulos, DistanciaX, DistanciaY)
Fin
```

Figura 14 Algoritmo para el Módulo de Control de Reacción y Movimiento del robot Koala

La información generada por el módulo anterior, respecto de la Posición del robot al momento, se almacena en un dispositivo secundario. Al finalizar su recorrido el Koala, todos estos puntos de posiciones visitadas se procesan con el algoritmo de la figura 15, obteniendo de la conjunción de cada uno de ellos la gráfica del seguimiento del muro.

```
Inicio
  x0=0,  y0=0,  x1=0,  y1=0
  AbrirArchivo(Datos1.Dat)
  Mientras  NO SEA FIN ARCHIVO (Datos1.Dat) hacer:
      LeerArchivo(Datos1.Dat, Registro)
      Si  NO ES ( El primer registro  )
      Entonces:
          AsignarValorX1Y1 ( Registro )
          TrazarLínea ( x0, y0, x1, y1 )
          ActualizarX0Y0 ( x0, y0, x1, y1 )
      Fin_Si
  Fin_Mientras
  Cerrar Archivo ( Datos1.Dat )
Fin
```

Figura 15 Algoritmo para la gráfica del seguimiento del muro del robot Koala

En la Figura 15 se visualiza la gráfica del seguimiento del muro del robot Koala, producto de la ejecución del software de control diseñado para el mismo.

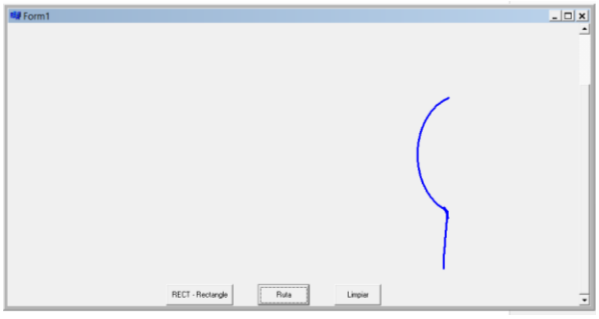

Figura 15 Gráfica seguimiento del muro del robot Koala

ISSN-2523-2517

Resultados

Gracias al software implementado y probado en este proyecto, el Koala tuvo movilidad. En las primeras pruebas, los resultados obtenidos no fueron los más halagadores, el robot colisionaba con los elementos del entorno, sus movimientos no eran finos. Sin embargo, se desplazaba con cierto control.

Entre otras, se ajustó: la frecuencia de lectura en los sensores infrarrojos para contribuir con movimientos finos; se actualizó los valores constantes y condiciones de referencia de seguimiento de muros en el *Módulo de Control de Reacción y Movimiento del robot Koala.* Con ello y, después de varias pruebas, observaciones y ajustes sobre los valores de: 1) luminosidad, 2) superficie y 3) espacio dentro de la configuración del software desarrollado, respondió favorablemente al entorno, quedando con ello, demostrado la autonomía de movimiento en el robot Koala.

Agradecimiento

Los autores agradecen infinitamente el apoyo invaluable de: Secretaría de Educación Pública, CONACYT, Tecnológico Nacional de México (TecNM), Instituto Tecnológico de Tláhuac y al Instituto Tecnológico de Puebla por auspiciar el proyecto.

Conclusión

La aplicación desarrollada en lenguaje C/C++ de Borland fue ejecutada para prueba en el puerto serie del robot Koala con los métodos y fundamentación de la planeación de trayectorias con algoritmos Bug.

Gracias a las pruebas del software ejecutadas con el Koala en entornos reales donde factores como la luminosidad, medio ambiente, superficie, tipo de obstáculo son cambiantes con respecto al tiempo y de ejecución a ejecución, permitieron incorporar estos aspectos al software implementado haciéndolo más "robusto" beneficiando directamente la autonomía del robot. A diferencia de las simulaciones, las cuales solo representan una situación particular. En la práctica y aplicando "ensayo y error" se mejoró el rendimiento del robot y se acrecentó el conocimiento del dispositivo.

Finalmente, las comprobaciones realizadas en el RMR indican que el software implementado mejora las soluciones alcanzadas; así como, el desempeño en relación al tiempo invertido para el cálculo de dichas soluciones.

Referencias

Barrientos, A., Peñín L.F., Balaguer C., Aracil R. (2007). *Fundamentos de Robótica.* México, D.F., México: McGraw-Hill Interamericana.

K-Team S.A., (2001). Koala User Manual Versión 2.0 (silver edition). Préverenges, Switzerland.

Benavides F., (2012). *Planificación de movimientos aplicada en robótica autónoma móvil*. Tesis de maestría ISSN 0797-6410. Universidad de la República, Montevideo. Uruguay.

López García, D.A. (2011). *Nuevas aportaciones en algoritmos de planificación para la ejecución de maniobras en robots autónomos no holónomos.* Tesis de maestría no publicada. Universidad de Huelva, Huelva, España.

Siegwart, R., Nourbakhsh, I. (2004). *Introduction to Autonomous Mobile Robots.* Cambridge, Massachusetts: The MIT Press.

Quintero P., et al., (2010). *Técnicas para evasión de obstáculos en Robótica Móvil*. IEEE Publications. Orlando, Florida, USA.

Ying L., Ruiqing F., Jiping W., (2016) *Una estrategia de seguimiento de muros para robots móviles basada en la autoconvergencia*. Shenzhen Institutes of Advanced Technology, Chinese Academy Science.

ISSN-2523-2517