Instituto Tecnológico Autónomo de México
Depto. Académico de Sistemas Digitales
Robótica

Proyecto 2
Generación de trayectorias en ROS

Edgar A. Granados Osegueda- 129565
Mariana R. Hernández Rocha- 150845
Jorge Isaac Chang Ortega - 149961
Humberto Isaac Téllez Benítez - 151887

***Resumen*— En este primer proyecto, se aprendieron los conceptos básicos para trabajar con el Robotic Operating System (ROS) y se desarrollaron dos programas para dicha plataforma utilizando los conceptos aprendidos y algunos recursos que ofrece este sistema como lo es turtlesim.**

Además de tener un primer acercamiento con ROS, la práctica también sirvió para dar los primeros pasos en un sistema operativo muy popular para aplicaciones tecnológicas, como lo es Linux en su versión Ubuntu 14.

I. INTRODUCCIÓN

La rueda es una máquina simple que ha sido de gran utilidad en la historia de la humandad. Su principal uso es para transportar y su implementación en vehículos forma parte de nuestro día a día.

Sin embargo, encontraremos que vehículos como el automóvil se encuentran muy limitados en su movimiento. Por ejemplo, sabemos que un coche no se puede mover lateralmente y tampoco puede girar sobre su propio eje.

Además, existe la restricción no holónoma, que forman las restricciones en las configuraciones de un sistema. En el caso del automóvil, ocurre cuando el objetivo al que queremos llevar el auto, está demasiado cerca de él y para llegar sería necesario dar un giro muy cerrado, lo que imposibilita llegar directamente a dicha pose.

El modelo cinemático que se utiliza para los automóviles comunes es tipo bicicleta, en el cuál las variables que conocemos son la distancia entre el eje trasero y el frontal, el radio de las ruedas y la distancia entre las ruedas.

Conociendo éstas variables y las restricciones del modelo, podemos conseguir que el automóvil llegue a una pose deseada por otros medios, ya sea dando giros más amplios o en varios movimientos. La acción a realizar la determinará el controlador.

II. MARCO TEÓRICO

II-A. Modelado cinemático y control de robots móviles con ruedas

II-B. Robot diferencial

El modelado de sistemas consiste en el uso de las ecuaciones que rigen la cinemática del robot móvil para determinar la posición de este en su entorno coordenado. La posición se calcula a partir de los cambios que se generen en los motores de locomoción del robot.

El robot diferencial, representado con una plataforma móvil de tracción diferencial, cuenta con dos pares de ruedas: dos ruedas de tracción y dos ruedas de estabilización que mantienen el balance del vehículo. Para su desarrollo se asume que las ruedas no tienen deslizamiento, de esta manera, el movimiento de las llantas del robot se convierte totalmente en un movimiento de traslación y/o rotación, movimientos independientes en cada una de las ruedas de tracción.

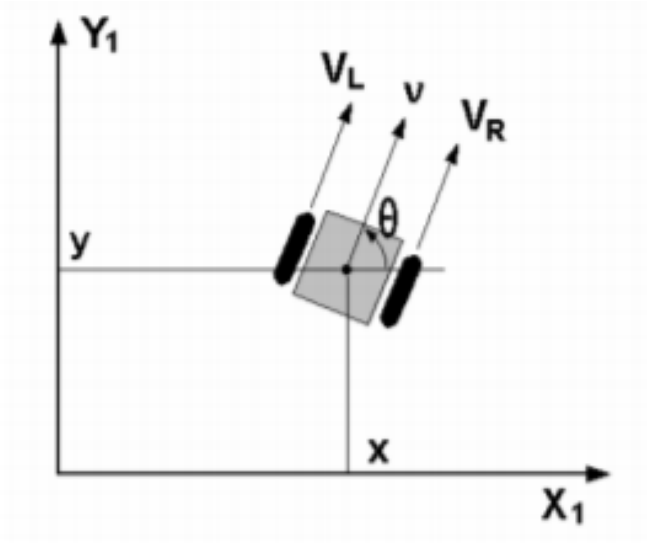

Fig. 1. modelo cartesiano del robot diferencial.

Se considera un vehículo diferencial: robot con dos ruedas independientes con un eje común a una distancia 2b de cada una y el modelo cinemático

$$u = 1/2(Vr + Vl)$$

$$w = 1/(2b)(Vr - Vt)$$

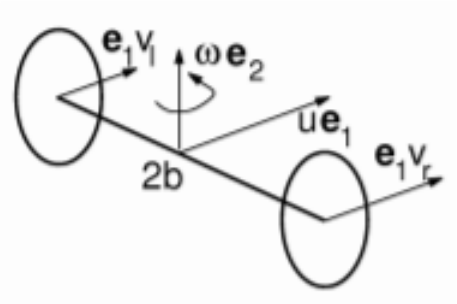

Fig. 2. robot diferencial.

II-C. *Modelo de Ackermann*

Se puede utilizar el modelo cinemático de la bicicleta para simplificar la estructura Ackermann. El modelo de la bicicleta simplifica las ruedas derecha e izquierda en dos ruedas situadas en la mitad de los ejes delantero y trasero como se muestra en la siguiente figura.

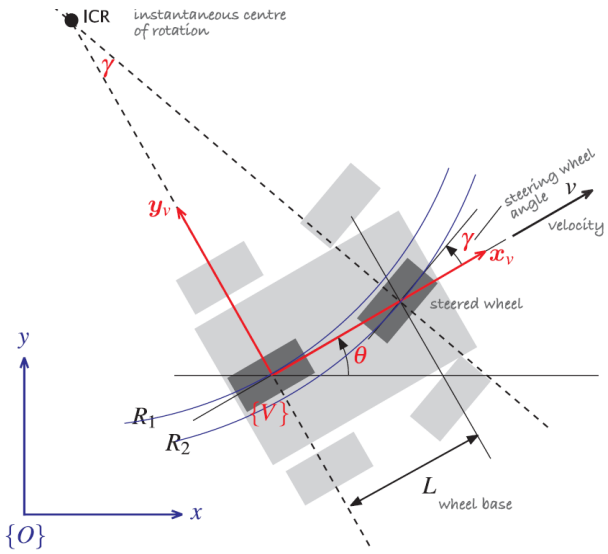

Fig. 3. modelo de la bicicleta.

La pose del vehículo está representada por el marco de coordenadas V, con su eje x en la dirección de avance del vehículo y su origen en el centro de la parte trasera eje. La configuración del vehículo está representada por las coordenadas generalizadas

$$q = (x, y, theta)$$

la velocidad del vehículo es v, la dirección x, y cero en la dirección y, ya que las ruedas no pueden deslizarse hacia los lados. El punto de referencia del vehículo sigue una trayectoria circular y su velocidad angular es $theta = v/R1$. Por su geometría, el radio de giro es $R1 = L/tan(lambda)$ donde L es la longitud de el vehículo o la base de la rueda, es importante resaltar que el ángulo de dirección lambda está limitado mecánicamente y su valor máximo lo dicta el valor mínimo de R1.

II-D. *Controlador para cambio de poses*

Para controlar la orientación final, se utilizó la siguiente matriz junto con transformaciones de las ecuaciones a coordenadas polares y cambios de variables. Podemos resumir el control proporcional y modelo cinemático con lo siguiente.

$$\begin{pmatrix}\dot{\rho}\\ \dot{\alpha}\\ \dot{\beta}\end{pmatrix} = \begin{pmatrix}-\cos\alpha & 0\\ \frac{\sin\alpha}{\rho} & -1\\ -\frac{\sin\alpha}{\rho} & 0\end{pmatrix}\begin{pmatrix}\nu\\ \gamma\end{pmatrix}, \quad \text{if } \alpha \in \left(-\frac{\pi}{2}, \frac{\pi}{2}\right]$$

Fig. 4. modelo cinemático.

$$\nu = k_\rho \rho$$

$$\gamma = k_\alpha \alpha + k_\beta \beta$$

Fig. 5. control proporcional.

II-E. *Gazebo*

Gazebo es un simulador de entornos 3D que posibilita evaluar el comportamiento de un robot en un mundo virtual. Permite diseñar robots de forma personalizada, crear mundos virtuales usando sencillas herramientas CAD e importar modelos ya creados.Además, es posible sincronizarlo con ROS de forma que los robots emulados publiquen la información de sus sensores en nodos, así como implementar una lógica y un control que dé ordenes al robot.

III. DESCRIPCIÓN DE LA SOLUCIÓN

Se implementó el control lineal proporcional presentado en el libro de Peter Corke, el cual hace que un automóvil con modelo cinemático tipo bicicleta llegue desde una pose inicial hasta alguna pose deseada establecida por el usuario. Este tipo de control tiene limitaciones, no sólo por la naturaleza del modelo cinemático sino por ser altamente propenso a oscilar, así como su mayor tiempo de estabilización respecto a controladores más complejos.

Para este propósito se creó un nodo llamado ¨robot_control", desde el cual se publicarán mensajes al tópico */AutoNOMOS_mini/manual_control/steering* y */AutoNOMOS_mini/manual_control/velocity*. Los datos requeridos por el controlador propuesto se obtienen de los tópicos */robot/pose* para los valores de la pose actual

del robot y */robot/next_pose* para los valores de la pose deseada.

III-A. Geometría de la solución y controladores

Utilizando el modelo cinemático de la bicicleta presentado en la sección anterior, el problema se divide en dos partes: el control de la velocidad del robot y el control de su orientación.

Para el primer caso, se utilizó un control que relacionara de manera proporcional la distancia entre el robot y el punto objetivo con la velocidad. Esto logra que entre más se acerque el robot a su destino, más se reduzca su velocidad, para permitir una desaceleración gradual y evitar un overshoot en la posición final del robot.

$$\upsilon = k_\rho \rho$$

Donde ρ representa precisamente la distancia entre la pose actual *(x*, y*, θ*)* y la pose deseada *(x*,y*,θ*)*.

$$\rho = \sqrt{(x^* - x)^2 + (y^* - y)^2}$$

Un aspecto importante de la solución presentada es la necesidad de determinar si la posición final deseada se encuentra en frente del robot o detrás del mismo. La posición deseada determina el signo que tendrá la velocidad, siendo positiva si el punto se encuentra en frente del robot con respecto a su eje 'Y' y negativa de otra forma. Esta cualidad de la solución se implementó directamente en el código de C++ con el uso de una estructura condicional *if*.

```
double x_pt_rob = new_pt.point.x;
double y_pt_rob = new_pt.point.y;

if (x_pt_rob < 0)
{
        vel.data = -kp * ro;
        ste.data = -(ka * alpha
        + kb * beta) * 180 / M_PI;
} else {
        vel.data = kp * ro;
        ste.data = (ka * alpha
        + kb * beta) * 180 / M_PI;
}
```

Las primeras dos líneas representan la transformación necesaria para ver la posición deseada con respecto al robot, pues el tópico */robot/next_pose* da la posición deseada con respecto al mundo (${}^r\xi_o$).

La segunda parte del problema, el control de la orientación, se compone de dos partes. Para implementarlo, hace falta calcular dos ángulos, α y β, ilustrados en la siguiente figura.

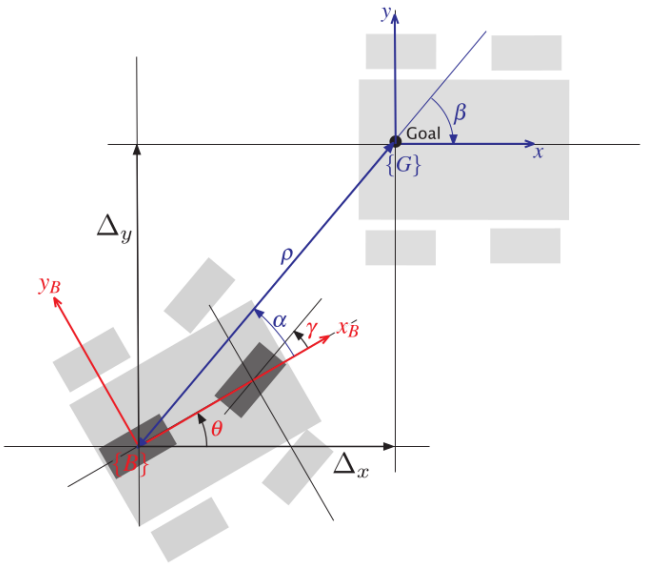

Fig. 6. Representación de los componentes del control de orientación.

En la Fig. 4, ρ es la distancia hasta la posición deseada explicada anteriormente, α representa el ángulo del vector destino respecto al robot y β es el ángulo del vector destino respecto al mundo. Estos dos últimos valores se usan en un control proporcional para ponderar el peso de cada uno y controlar la orientación de manera que la orientación final del vehículo coincida con la destino. Una vez más, la relación entre ambas es lineal y se expresa de la siguiente forma.

$$\omega = k_\alpha \alpha + k_\beta \beta$$

III-B. Valores para el control

Para los valores utilizados en cada control existen restricciones mínimas. En el caso del control de velocidad sólo existe una y tiene que ver con el hecho de que si no existiera, el control propuesto no cumpliría ningún propósito pues el vehículo aceleraría mientras más cerca se encontrara del objetivo.

$$k_\rho > 0$$

Para el control de dirección, existen dos restricciones, las cuales tienen que ver con la prioridad que tiene la rotación del robot según su propia percepción de orientación respecto al objetivo sobre la percepción global del mismo.

1. $k_\beta < 0$
2. $k_\alpha - k_\rho > 0$

Respetando estos valores en la implementación del código fuente y durante las pruebas es como se logró obtener resultados favorables, los cuales serán expuestos en la siguiente sección.

IV. EXPERIMENTOS

En la Fig. 7 se muestra la trayectoria del robot entre los puntos $(0, 0)$ y los puntos $(5, 1)$. En la Fig. 8 se muestra la velocidad en el tiempo seguida por el robot. Al tratarse de un control proporcional, la velocidad inicial es alta, disminuyendo conforme la distancia (el error) al punto meta disminuye.

Una consecuencia de usar un control proporcional es que el error no necesariamente será cero, aunque si será muy cercano a cero. Para obtener un error de cero (o más cercano a cero), sería necesario utilizar un control PI o PID. Con la implementación de un control PI o PID, se podría llegar a la posición deseada más rápido. Otro aspecto importante es que la velocidad llega al valor deseado rápidamente, lo cual no necesariamente pasaría en un robot real ya que el tiempo de respuesta de un motor suele ser menor. En la Fig. 9 se muestra el ángulo en radianes del robot respectos al mundo en el tiempo. Debido a que el ángulo inicial es igual al final, primero crece para poder orientarse y llegar al punto deseado. Conforme se acerca al punto final, la orientación del robot también se acerca a la deseada, aunque termina teniendo un error. Éste error está relacionado con las constantes de control elegidas. Se utilizó $k_\rho = 1$, $k_\alpha = .1$ y $k_\beta = -0.1$, obtenidas experimentalmente. Aunque con éstas constantes se obtuvieron buenos resultados, es posible que se puedan determinar mejores valores analizando el comportamiento de los actuadores (articulaciones en el simulador).

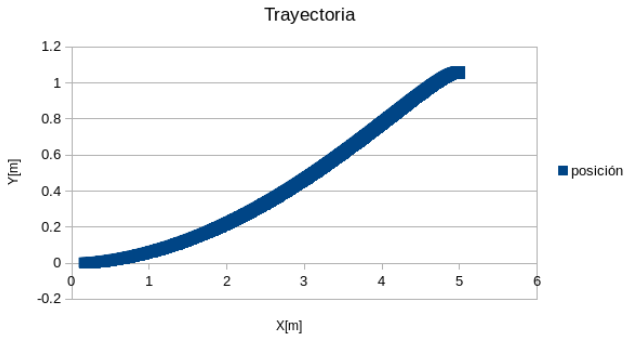

Fig. 7. Trayectoria 1 del Robot

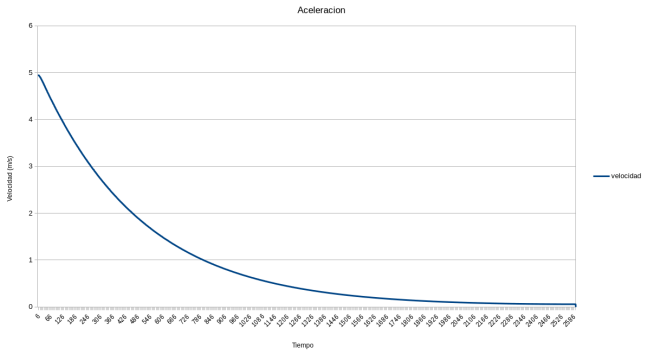

Fig. 8. Aceleración durante la Trayectoria 1 del Robot

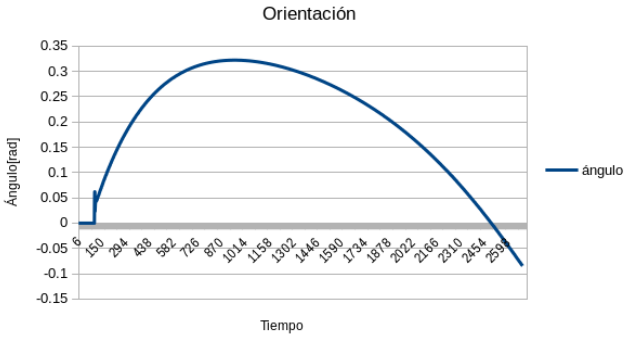

Fig. 9. Ángulo del Robot respecto al Mundo en el Tiempo

En la Fig. 10 se muestra una segunda prueba realizada que consiste en realizar la trayectoria entre la pose inicial $(0, 0, 0)$ y la pose final $(-5, -1, 0)$. La velocidad del robot en el tiempo se muestra en la Fig. 11. Ésta prueba representa un caso especial ya que el punto meta se encuentra atrás del robot. El robot llega correctamente al punto, aunque la velocidad comienza a oscilar al acercarse a la meta. Ésta oscilación está relacionada a los valores elegidos para las constantes k de control.

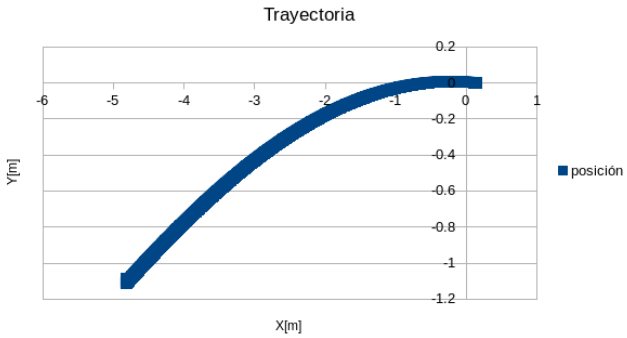

Fig. 10. Trayectoria 2 del Robot

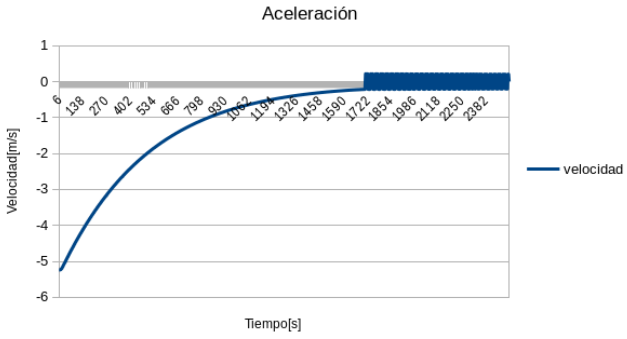

Fig. 11. Aceleración durante la Trayectoria 2 del Robot

V. CONCLUSIONES

Basándonos en los conocimientos adquiridos en el primer proyecto sobre generación de trayectorias, el objetivo del

presente proyecto fue aplicarlos al problema del control cinemático de un coche autónomo.

El control se vale de los parámetros de velocidad y ángulo del volante para generar la trayectoria que llevará al coche de una pose actual a una pose deseada, por lo que el coche no sólo llega al punto cartesiano deseado sino que además debe estar orientado correctamente.

Las complicaciones que se presentaron durante la realización están relacionadas con la selección de los valores adecuados para las constantes k ya que son éstas las que determinan al sistema haciéndolo oscilar al incrementar o decrementar su valor.

Además, se podía presentar el caso en el que la posición final deseada estuviera detrás del robot, al cual la solución fué implementada directamente en el código, dividiendo el problema en casos.

Los resultados obtenidos tras la realización de varias pruebas fueron satisfactorios, sin embargo se siguen presentando pequeñas oscilaciones. Un área de oportunidad de nuestro proyecto sería refinar el control para minimizar dichas variaciones y conseguir una convergencia óptima.

REFERENCIAS

[1] Jason M. O'Kane, "A Gentle Introduction to ROS", Independently Published, 2013.

[2] Corke Peter, "Robotics, Vision and Control", Springer, Second Edition, pg.99.