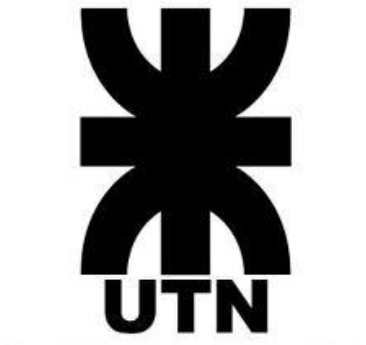
Universidad Tecnológica Nacional

FACULTAD REGIONAL BUENOS AIRES

Trabajo Práctico

Robot Móvil: Cinemática, dinámica y compilador del N6



Materia: Robótica

Profesor: Ing. Gianetta, Hernán

Ayudante: Ing. Granzella, Eduardo Damián

Alumnos: Paglia, Lucas 143.421-4
Sankowicz, Javier 143.556-5

Estudio cinemático para un robot móvil diferencial de dos ruedas

Introducción

Haciendo base en el modelo para robot diferencial de dos ruedas visto en clase, se pretende realizar una serie de pruebas y mediciones según las consignas planteadas a continuación, que permitan la validación parcial o total de este modelo.
Una vez realizadas las mediciones de campo, el siguiente trabajo pretende analizar la información obtenida mediante herramientas computacionales, y finalmente mostrar resultados y extraer conclusiones.

A continuación, se exponen las consignas para el trabajo:

1. Calcular en Matlab la trayectoria.
2. Programar el movimiento de Cinemática directa del Robot diferencial.
3. Realizar un ploteo del encoder o trayectoria medida.
4. Comparar El modelo Matlab con la programación real y extraer conclusiones y gráficos.

Materiales y Métodos

Para el trabajo se utilizó el robot diferencial de dos ruedas "Múltiplo N6", que cuenta con dos motores de corriente continua de 200 RPM con 12V de tensión nominal, ruedas de 6cm de diámetro separadas 13cm una de otra y encoders de 16 pasos con sus sensores infrarrojos pertinentes.

El modelo cinemático directo del Robot N6 es el siguiente:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} \cos\omega\delta t & -\sin\omega\delta t & 0 \\ \sin\omega\delta t & \cos\omega\delta t & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} x - ICC_x \\ y - ICC_y \\ \theta \end{bmatrix} + \begin{bmatrix} ICC_x \\ ICC_y \\ \omega\delta t \end{bmatrix}$$

Siendo:

$$ICC = [ICC_x \quad ICC_y] = [x - R\sin\theta \quad y + R\cos\theta]$$

$$\omega = \frac{(v_R - v_L)}{l}$$

$$R = \frac{l}{2} * \frac{(v_R + v_L)}{(v_R - v_L)}$$

Para el caso en que $v_R = v_L$, el modelo queda:

$$\begin{bmatrix} x' \\ y' \\ \theta' \end{bmatrix} = \begin{bmatrix} x + v \sin \delta t \\ y + v \cos \delta t \\ \theta \end{bmatrix}$$

En el presente trabajo se planteó 3 trayectorias para estudiar los errores que se cometen al medirla con los encoders ópticos del robot. Estas trayectorias son:

1. Movimiento Rectilíneo
2. Movimiento con pequeñas variaciones de velocidades entre rueda derecha e izquierda
3. Movimiento con grandes variaciones de velocidades entre rueda derecha e izquierda

Para estos 3 casos se midió con el robot dado vuelta, es decir, el mejor caso posible de análisis donde el movimiento del mismo no se ve afectado por el resbalamiento de las ruedas con el piso ni con la variación del terreno (tanto en tipo como en pendientes). Además, se tomaron una serie 10 velocidades distintas (para ambas ruedas) con distintos valores de δt.

Para la adquisición de la información de los encoders se utilizaron sensores ópticos analógicos CNY70, los cuales se encontraban conectados cada uno a un canal del conversor analógico digital de la plataforma. El algoritmo desarrollado detectaba los flancos de cambio entre el color blanco y el color negro de los encoders, disponiendo para ello de niveles analógicos definidos para cada color, los cuales fueron obtenidos mediante prueba y error.
Al ser posible detectar los flancos de cambio, el sistema podía entonces acumular la cantidad de pasos que daba el encoder y de esa manera calcular la distancia recorrida y la velocidad a partir de cálculos simples en los cuales estaba involucrada la geometría de las ruedas (se supusieron ruedas de sección perfectamente circular).

Resultados

Para propósitos gráficos y de visualización, se tomó un ángulo inicial de 45°. Este podría haber sido 0° y el estudio habría sido el mismo.

Movimiento Rectilíneo

En este caso, se esperaba obtener una trayectoria como la mostrada en la figura 1. Una vez hechas las mediciones obtuvimos una trayectoria "medida" por el robot como la mostrada en la figura 2. En la figura 3 se muestra la comparación de ambos gráficos.

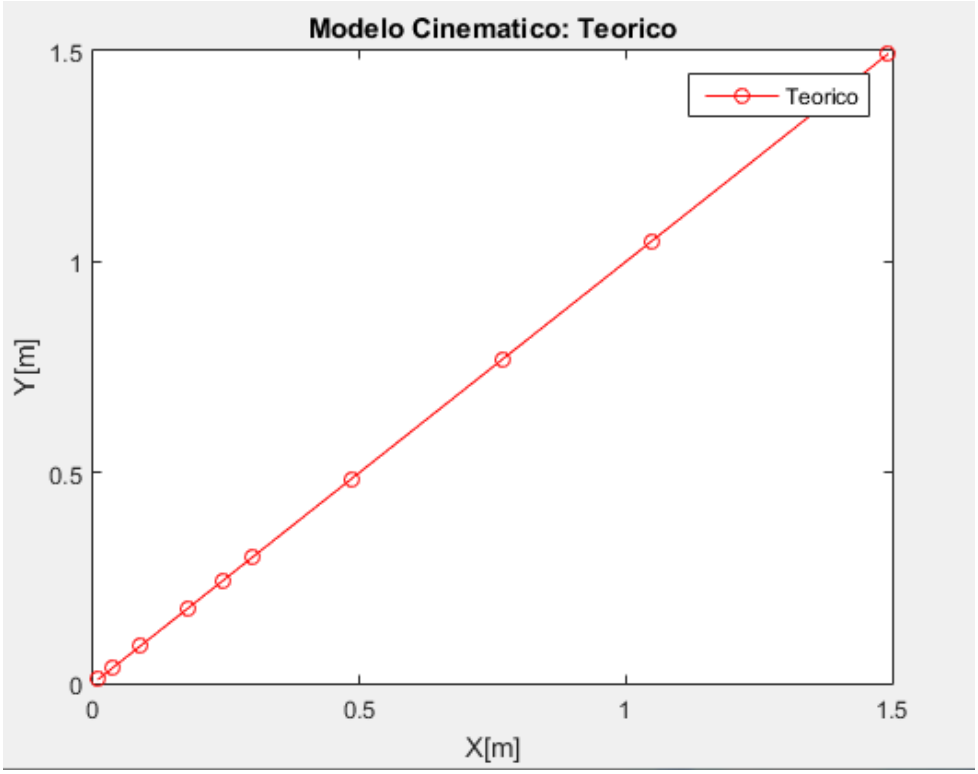


Figura 1: Movimiento Rectilíneo Teórico

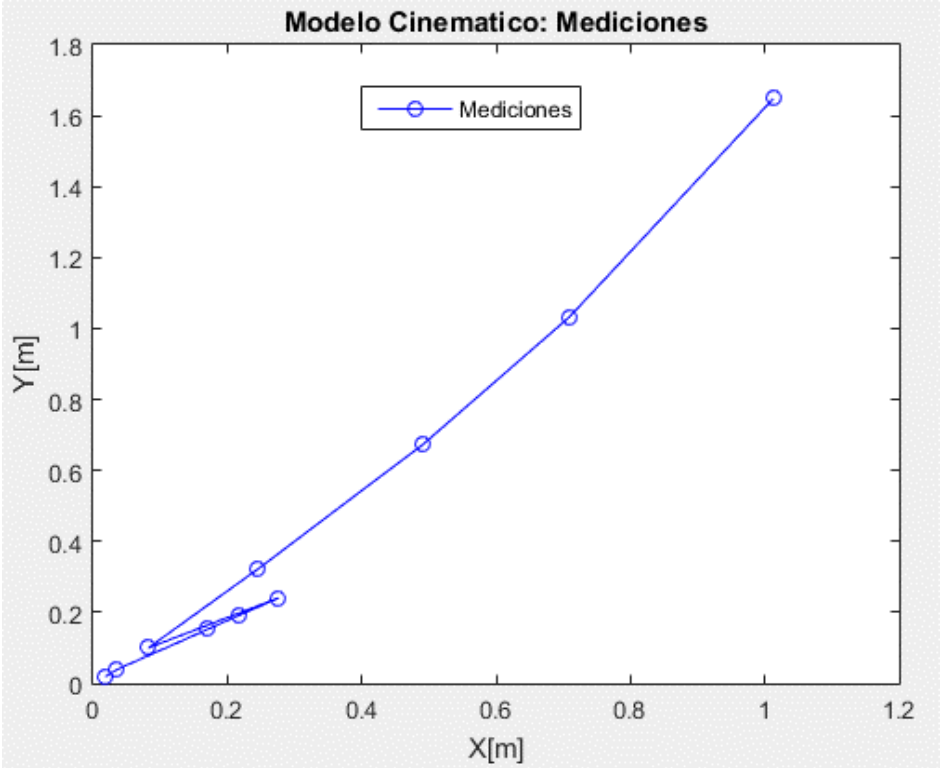


Figura 2: Movimiento Rectilíneo Medido

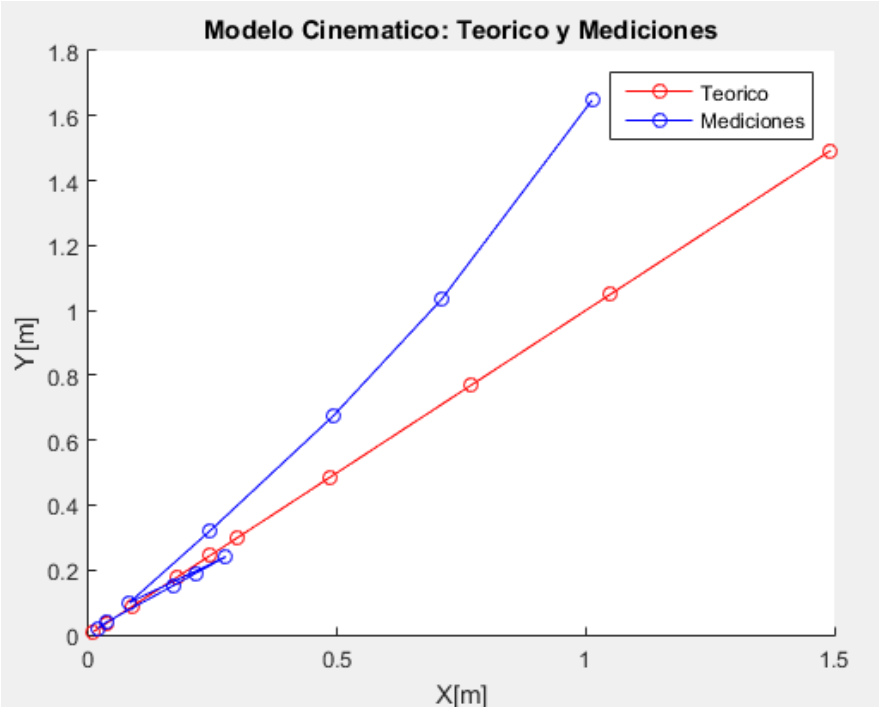


Figura 3: Movimiento Rectilíneo Comparación

Se puede observar en la figura 2 cómo se comete un error en la medición de la velocidad en la sexta medición. En ese tramo el robot mide que está retrocediendo. Esto se debe a una diferencia entre las ruedas que causan un giro. Pero al ver que luego de esto el robot “corrige” y sigue con la trayectoria normal, podemos concluir que la medición no es en sí errónea, sino que son rastros de una medición anterior.

Para analizar el giro mencionado, se graficó en la figura 4 las velocidades teóricas y medidas en cada rueda. Se puede observar cómo la velocidad de la rueda derecha está por encima de la teórica, mientras que la izquierda se encuentra por debajo. Esta medición genera el “giro” indeseado en el robot que luego es “corregido” por obtener las siguientes mediciones de forma correcta.

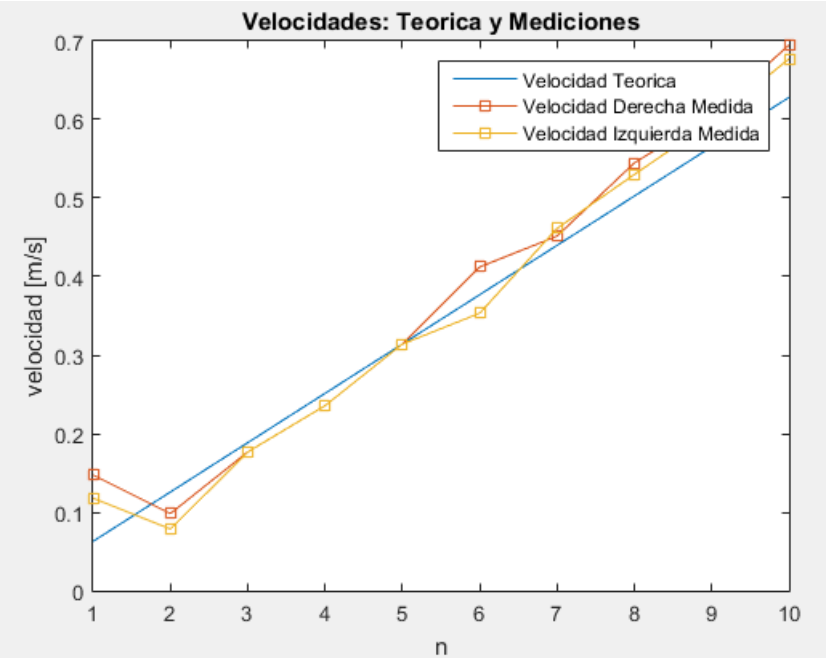


Figura 4: Comparación velocidades en las ruedas

Aplicando los factores de corrección calculados, se obtiene las siguientes velocidades derecha e izquierda en las figuras 5 y 6 respectivamente, así como también la gráfica de la trayectoria en la figura 7.

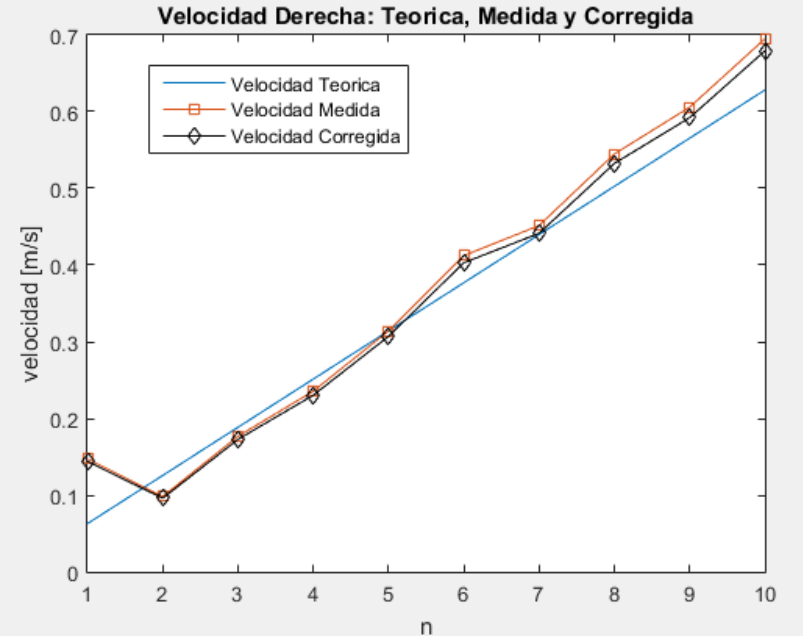


Figura 5: Velocidad Derecha Teórica, Medida y Corregida

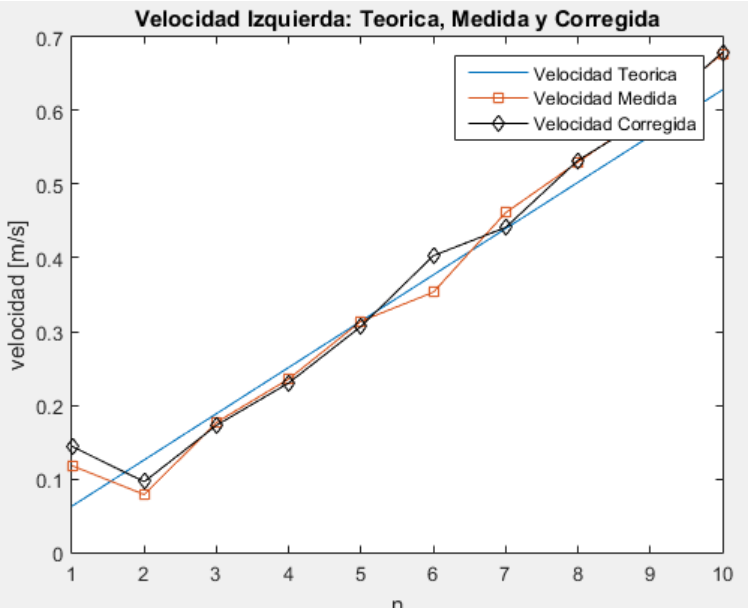


Figura 6: Velocidad Izquierda Teórica, Medida y Corregida

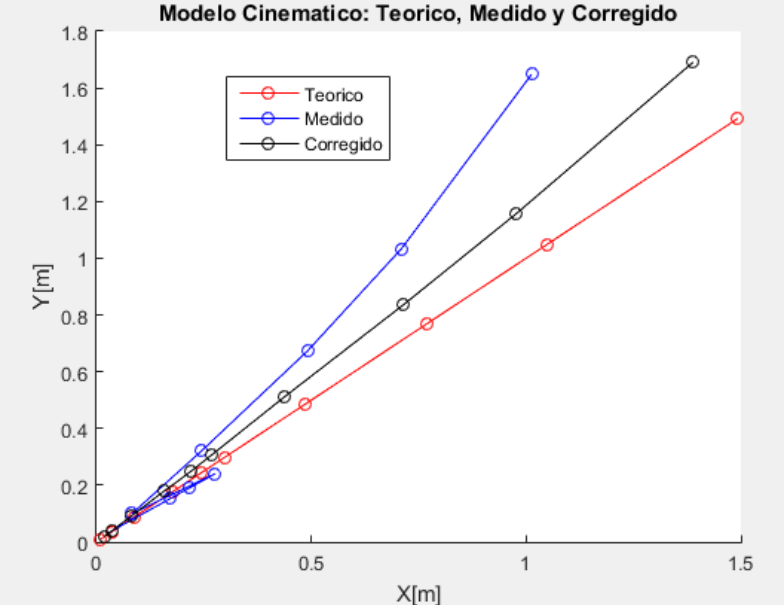


Figura 7: Trayectorias Teórica, Medida y Corregida

Movimiento con pequeñas variaciones de velocidad

A continuación, se muestra en la figura 8 las trayectorias teórica y medida del robot para pequeñas variaciones de velocidad entre ambas ruedas, y en la figura 9 las velocidades teóricas y medidas de cada rueda.
Además, se graficó el error en la medición de velocidad punto a punto en la figura 10.

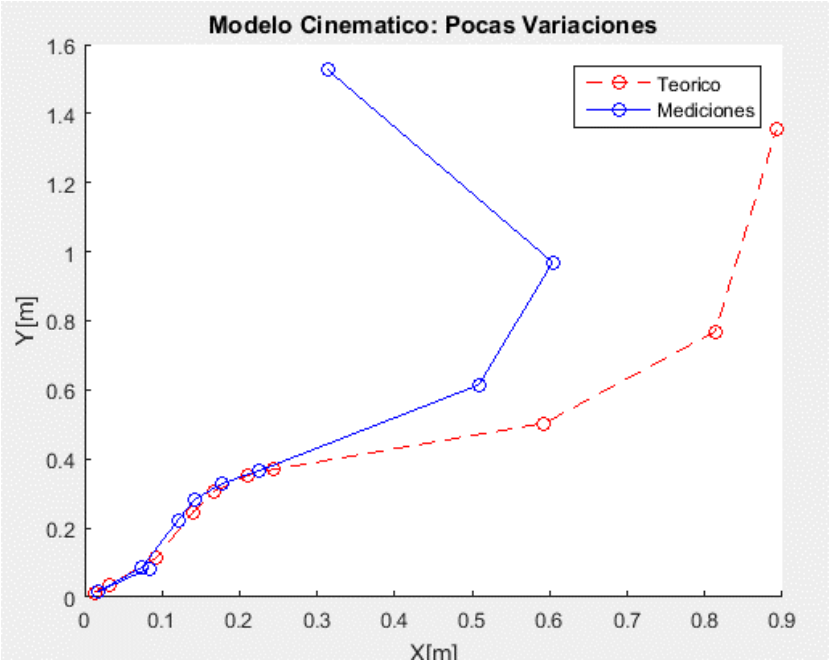


Figura 8: Pequeñas Variaciones, Teórico vs. Medido

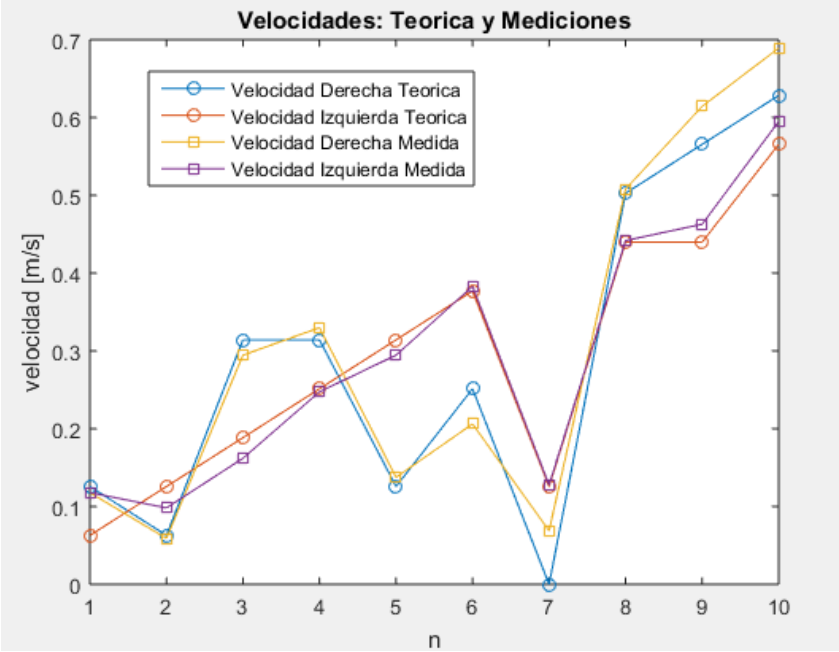


Figura 9: Pequeñas Variaciones, Velocidad Teórica vs. Medidas

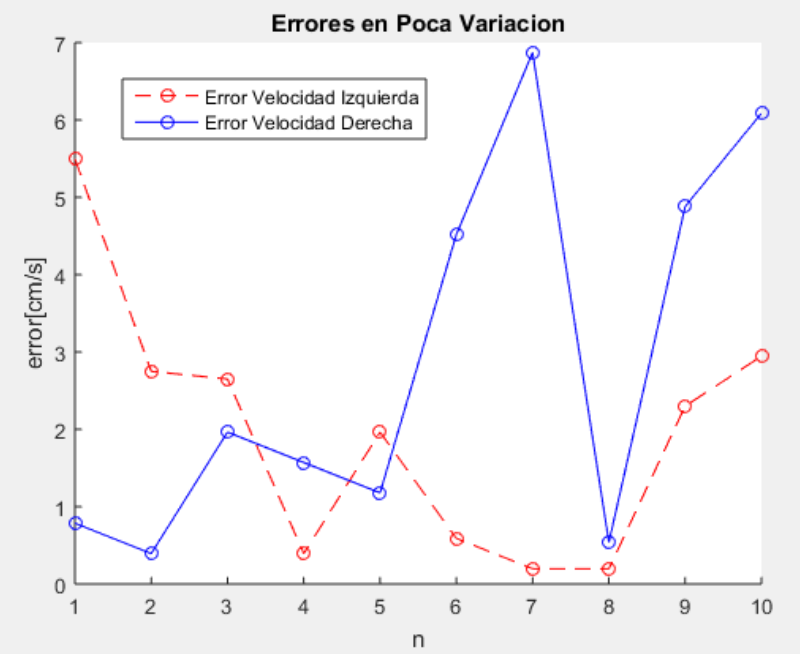


Figura 10: Errores en la Medición de Velocidad

Aplicando los factores de corrección calculados, se obtiene las siguientes velocidades derecha e izquierda en las figuras 11 y 12 respectivamente, así como también la gráfica de la trayectoria en la figura 13.

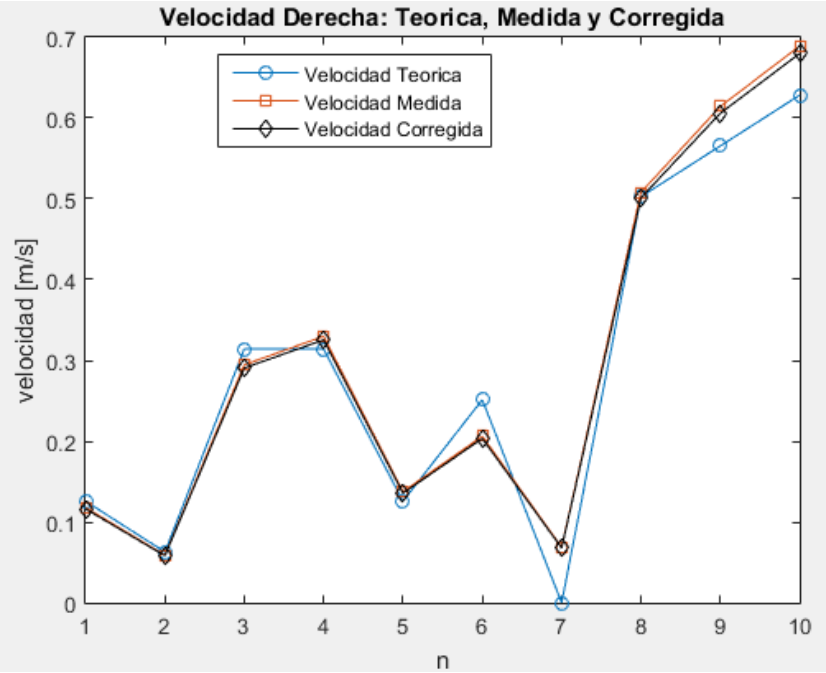


Figura 11: Velocidad Derecha Teórica, Medida y Corregida

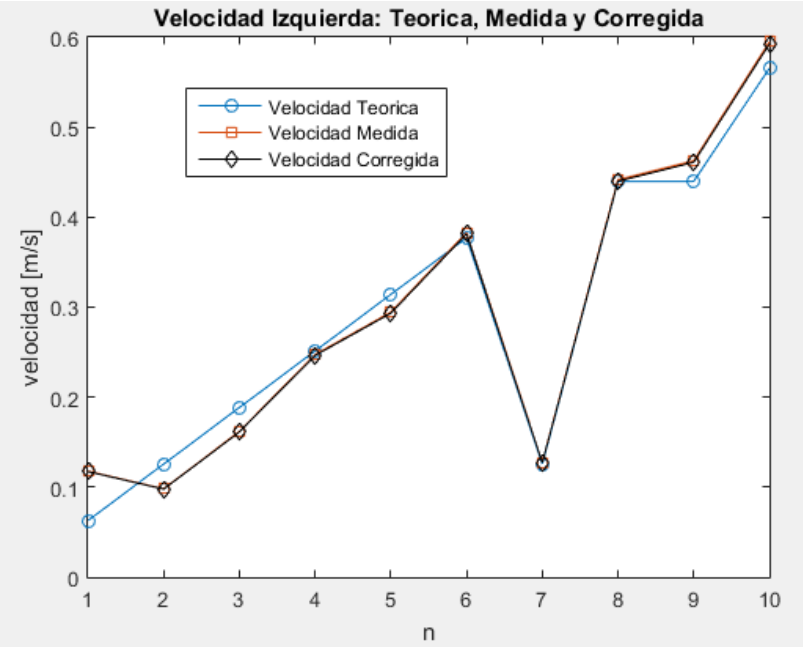


Figura 12: Velocidad Izquierda Teórica, Medida y Corregida

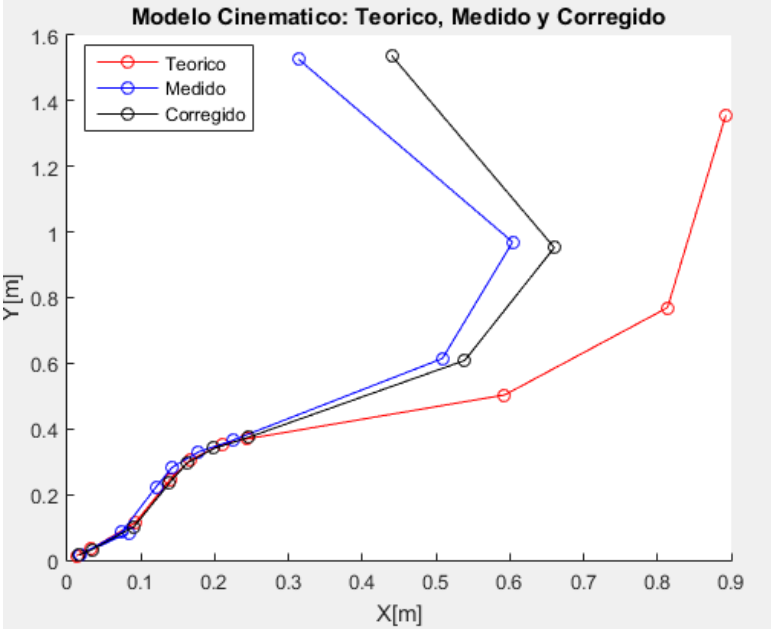


Figura 13: Trayectorias Teórica, Medida y Corregida

Movimiento con grandes variaciones de velocidad

A continuación, se muestra en la figura 14 las trayectorias teórica y medida del robot para pequeñas variaciones de velocidad entre ambas ruedas, y en la figura 15, las velocidades teóricas y medidas de cada rueda.
Además, se graficó el error en la medición de velocidad punto a punto en la figura 16.

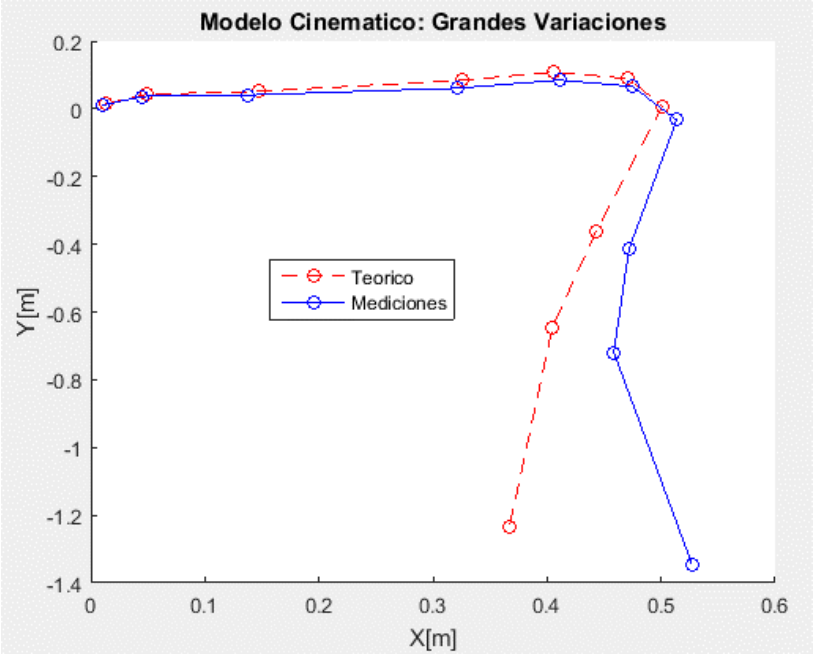


Figura 14: Grandes Variaciones, Teórico vs. Medido

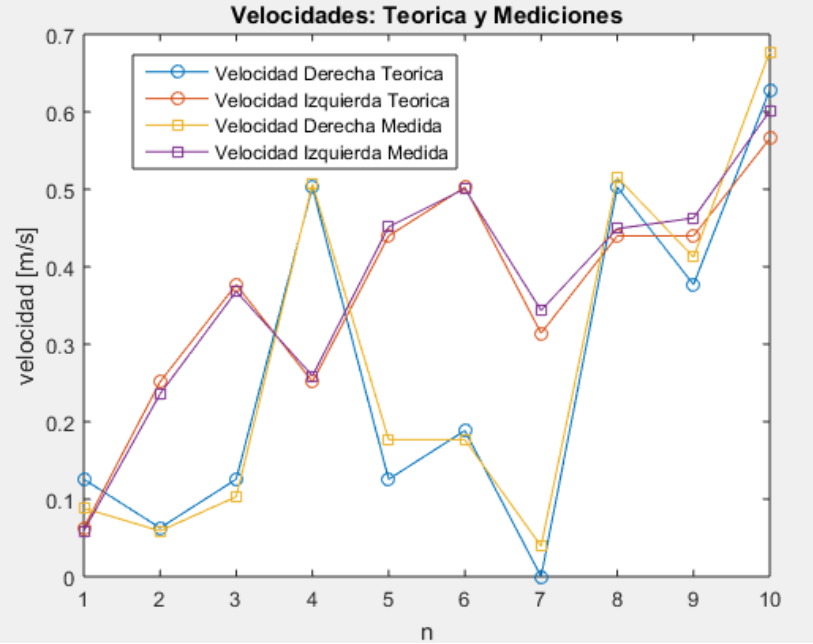


Figura 15: Grandes Variaciones, Velocidades Teóricas vs. Medidas

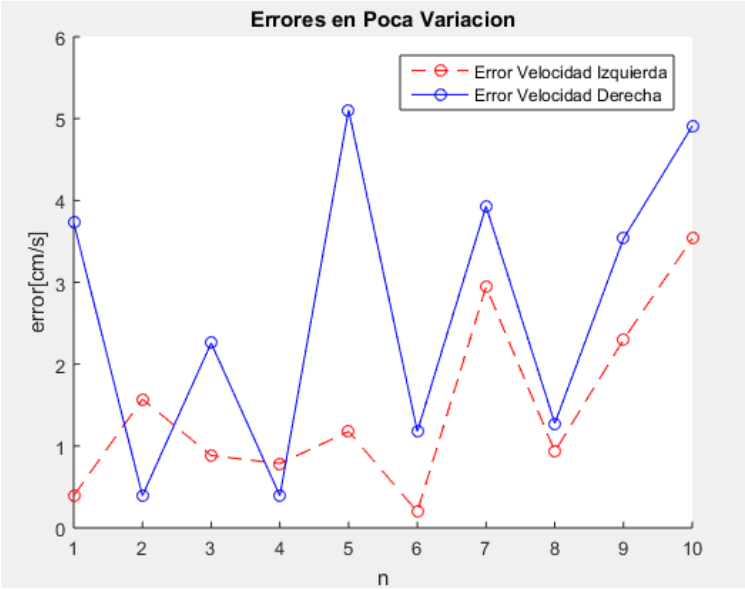


Figura 16: Errores en la Medición de Velocidad

Aplicando los factores de corrección calculados, se obtiene las siguientes velocidades derecha e izquierda en las figuras 17 y 18 respectivamente, así como también la gráfica de la trayectoria en la figura 19.

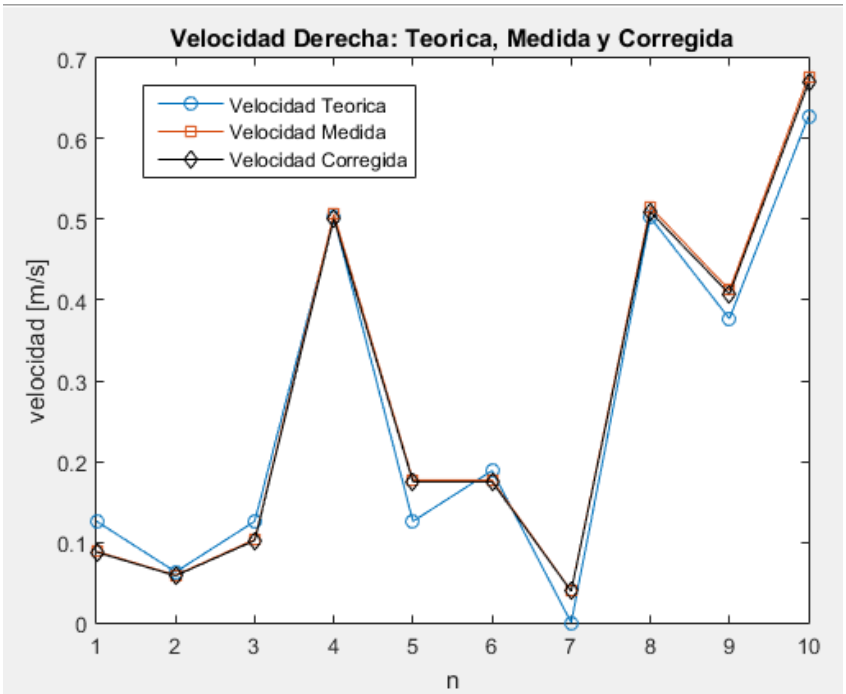


Figura 17: Velocidad Derecha Teórica, Medida y Corregida

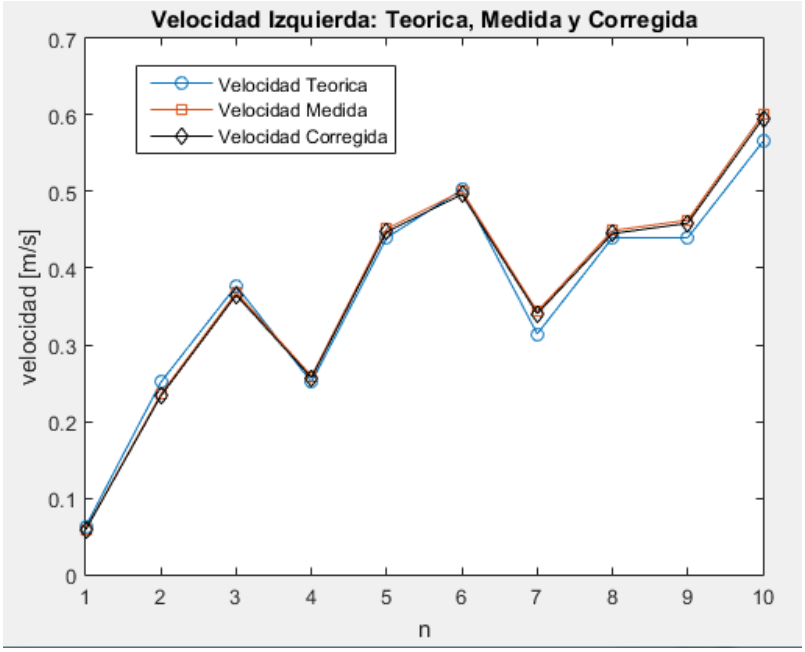


Figura 18: Velocidad Izquierda Teórica, Medida y Corregida

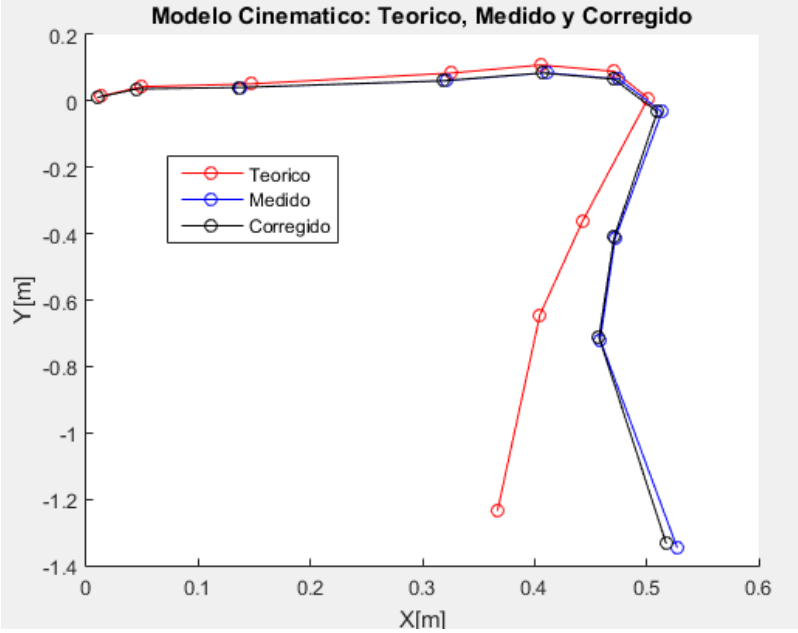


Figura 19: Trayectorias Teórica, Medida y Corregida

Conclusiones

Como fue mencionado anteriormente, todas las mediciones fueron llevadas a cabo con el robot dado vuelta, es decir, con la menor cantidad de errores introducidos por el medio ambiente (terreno rugoso, trayectoria sobre pendiente, resbalamiento de las ruedas, etc). Incluso en el mejor caso de medición, se ven errores en las mediciones por los encoders, ya sean los errores del propio sensor, o bien, los atribuibles a diferencias de velocidad entre los motores (diferencias constructivas, etc.).

En los gráficos de las trayectorias teóricas y obtenidas en el robot, vemos un bajo nivel de error (de aproximadamente 15-20 cm.) para la aplicación elegida. Si se estuviera analizando errores para un auto inteligente, los errores deberían estar entre los 2-10 cm, caso para el cual los errores obtenidos serían muy altos.

Para la mejora de la aplicación, proponemos dos soluciones:
La implementación de encoders ópticos con mayor cantidad de pasos para reducir el error en la cuenta y posterior medición de velocidad (limitación física-mecánica).
La introducción de un factor de corrección en la medición de velocidad derecha e izquierda:

$$vLcorregida = vL_m * [1 - E(vL)]$$

Donde:

$$KvL = 1 - E(vL) = 0.9913$$

$$vRcorregida = vR_m * [1 - E(vR)]$$

Donde:

$$KvR = 1 - E(vR) = 0.9884$$

Este factor de corrección fue calculado por medio del error para la velocidad de cada rueda. Dicho error es la media de las diferencias entre el valor teórico y el valor real medido con los encoders.

Referencias

[Dudek]: G.Dudek; M. Jenkin. Computational principles of mobile robotics. CAMBRIDGE UNIVERSITY PRESS, 2010, 2ed.

Estudio dinámico para un robot móvil diferencial de dos ruedas

Introducción

Haciendo base en el modelo dinámico para robot diferencial de dos ruedas visto en clase, se pretende realizar una serie de pruebas y simulaciones según las consignas planteadas a continuación, que permitan la validación parcial o total de este modelo.
Una vez realizadas las mismas, se pretende sacar conclusiones sobre las trayectorias, velocidades y aceleraciones más adecuadas para el robot, y en particular, para el conjunto rueda-motor del mismo.

A continuación, se exponen las consignas para el trabajo:

1. Calcular el modelo dinámico en Matlab para una trayectoria establecida.
2. Implementar dicho modelo en el N6.
3. Comparar El modelo Matlab con la programación real y extraer conclusiones y gráficos.

Materiales y Métodos

Para el trabajo se utilizó el robot diferencial de dos ruedas "Múltiplo N6", que cuenta con dos motores de corriente continua de 200 RPM con 12V de tensión nominal, ruedas de 6cm de diámetro separadas 13cm una de otra y encoders de 16 pasos con sus sensores infrarrojos pertinentes.

El modelo dinámico del Robot N6 es el siguiente:

$$D_{11}\theta_R'' + D_{12}\theta_L'' + \beta\theta_R' = \tau_R$$

$$D_{21}\theta_R'' + D_{22}\theta_L'' + \beta\theta_L' = \tau_L$$

$$D_{11} = D_{22} = \left[\frac{mr^2}{4} + \frac{\left(I_Q + mb^2\right)r^2}{8a^2} + I_o\right]$$

$$D_{12} = D_{21} = \left[\frac{mr^2}{4} - \frac{\left(I_Q + mb^2\right)r^2}{8a^2}\right]$$

En el presente trabajo se tomó la trayectoria del robot de la figura 20, analizada con el modelo cinemático directo del N6. Se pretende realizar un análisis de las velocidades, aceleraciones y torques necesarios para la trayectoria pedida y comparar con los valores de torque nominal y a rotor bloqueado de los motores del robot. Se plantearán diferentes interpolaciones para mejorar las funciones de aceleración, así como también realizar un análisis con la variación de los parámetros m (masa) y β (coeficiente de fricción).

Se realizó el modelo del robot N6 en solidworks, del cual se extrajeron los valores del centro de gravedad y momentos de inercia del robot y del conjunto ruedas-motores. Estos valores obtenidos son la primera fuente de error para el modelo dinámico, puesto que no se incluyeron todos los componentes del N6 en el modelado 3D. A modo de ejemplo, no se introdujeron los tornillos, tuercas y sensores. Además, las ruedas utilizadas en el modelo no son las mismas que posee dicho robot.

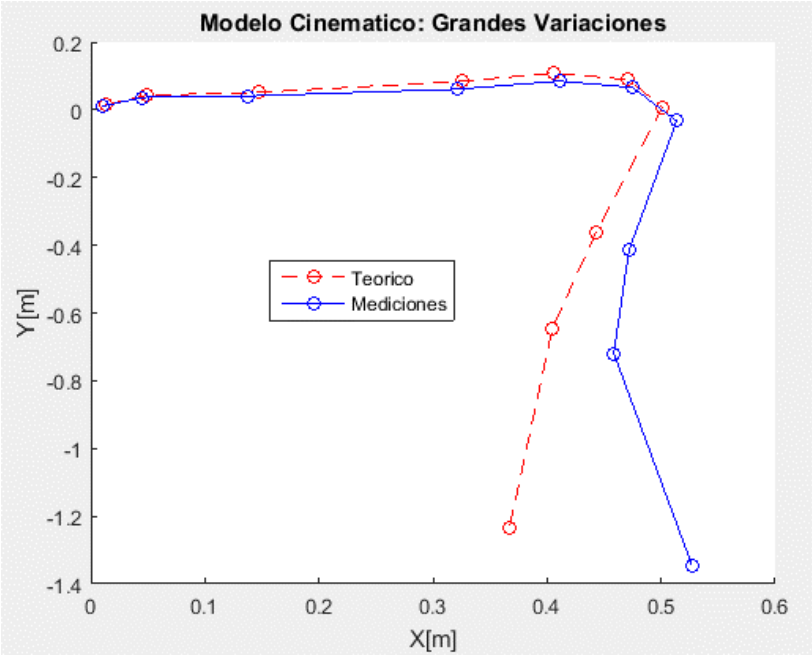


Figura 20: Trayectoria analizada para el modelo dinámico del Robot N6

En las figuras 21 a 24 se muestra el modelo de Solidworks utilizado y en la figura 25 se muestra los parámetros físicos obtenidos.

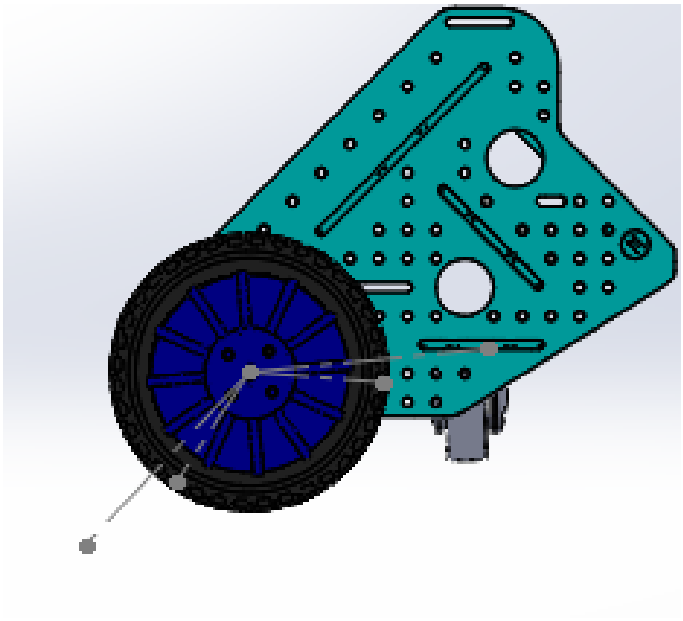


Figura 21: Vista lateral del modelo del N6

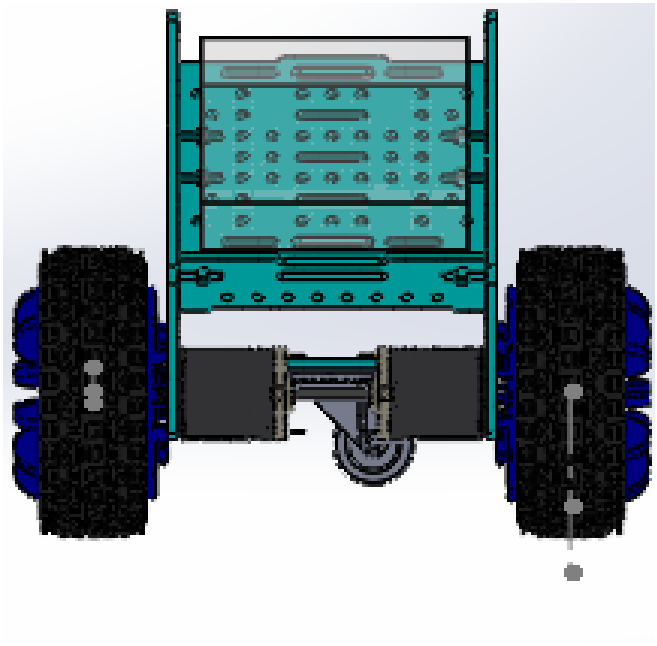


Figura 22: Vista frontal del modelo del N6

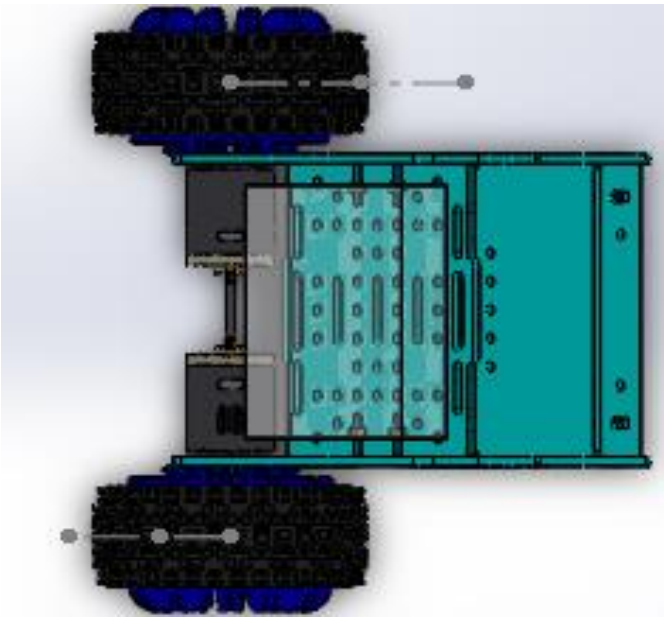


Figura 23: Vista inferior del modelo del N6

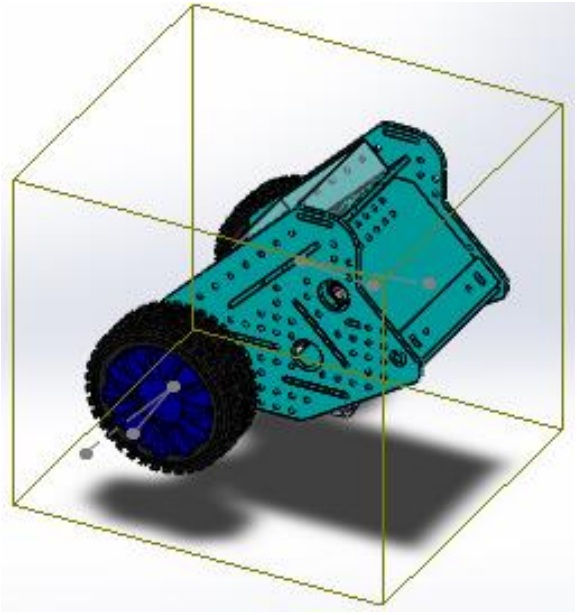


Figura 24: Perspectiva del modelo del N6

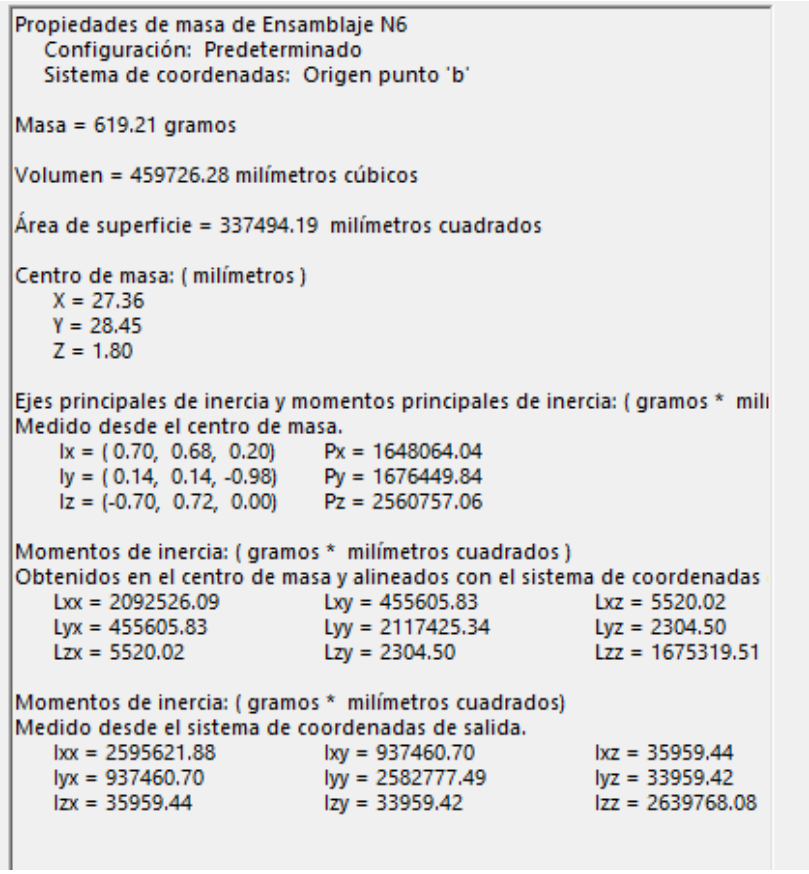


Figura 25: Propiedades físicas del ensamblaje del N6

Procedimiento y Resultados

Para la trayectoria deseada, las correspondientes velocidades de los motores derecho e izquierdo del motor deberían ser, de forma ideal, las graficadas en la figura 26.

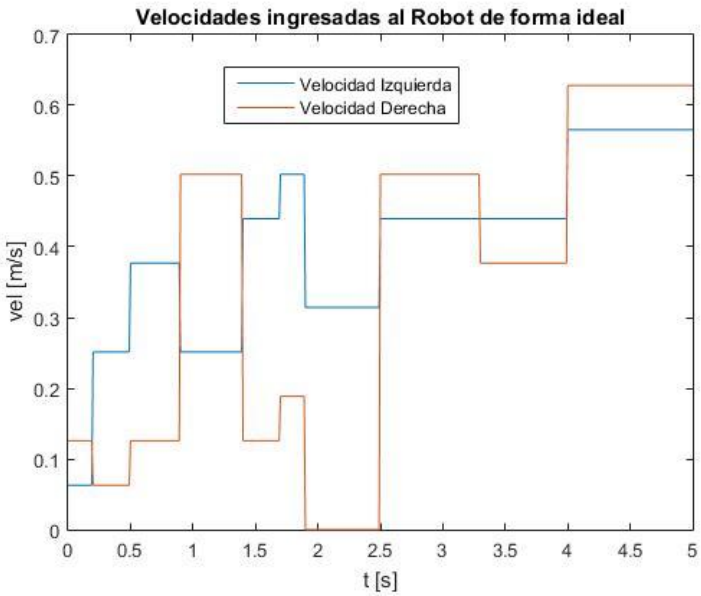


Figura 26: Velocidades ideales para el Robot

Estos valores son teóricos, no es posible realizar saltos de velocidad instantáneos. Por lo tanto, se plantearon las velocidades graficadas en la figura 27. Junto con esto, se extraen las aceleraciones en la figura 28 y los torques obtenidos del modelo en la figura 29. En la tabla 1, se pueden visualizar los parámetros utilizados en dicho modelo. Para ilustrar lo ya mencionado sobre el modelado del robot de forma incompleta, se menciona sobre el parámetro real y obtenido de la masa. Mientras que la masa del motor es de 720 gramos, se obtuvo una masa de 600 gramos.

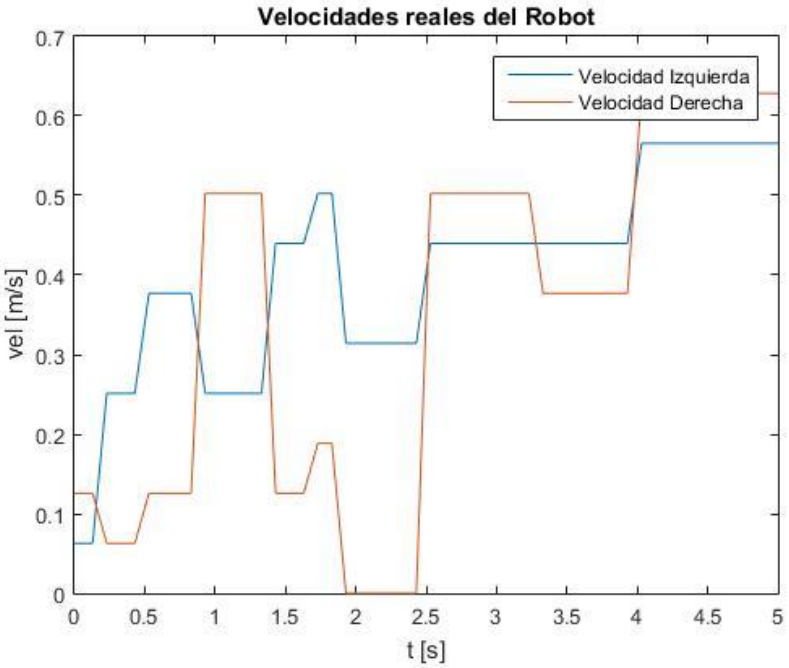


Figura 27: Velocidades reales del Robot

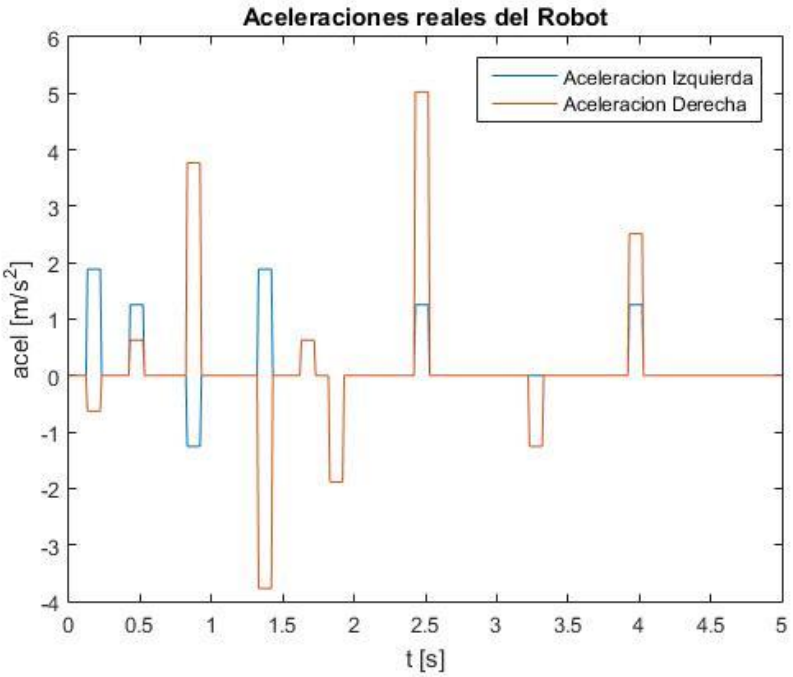


Figura 28: Aceleraciones reales del Robot

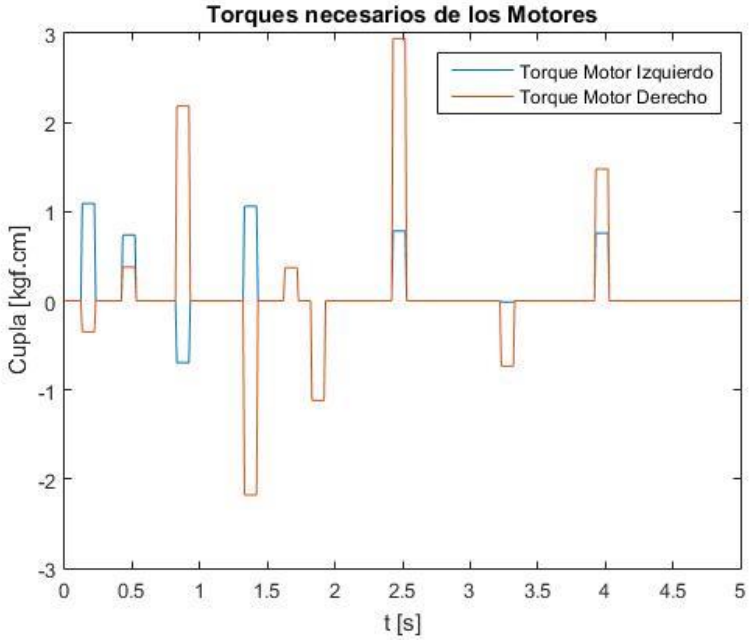


Figura 29: Torques reales del Robot

Parámetro	Valor	Unidad
m (masa)	0.6	Kg
r (radio ruedas)	30	mm
b (distancia al COG)	27.36	mm
2*a (dist. entre ruedas)	130	mm

I_Q (inercia del robot)	$3.47*10^6$	$Kgmm^2$
I_O (inercia ruedas-motores)	$1.5*10^6$	$Kgmm^2$
β (friccion)	$35*10^{-7}$	Nms/rad

Tabla 1: Parámetros del Robot

En cuanto al motor se tienen los siguientes datos:

- Cupla Nominal=0.58 kgf.cm
- Cupla de Arranque / Bloqueo = Cupla Nominal * 4 = 2.32 kgf.cm

De la figura 29 se extrajeron los siguientes valores de torque:

$$\tau_1 = 2.184\ kgf.cm$$

$$\tau_2 = -2.178\ kgf.cm$$

$$\tau_3 = 2.48\ kgf.cm$$

Para los valores τ_1 y τ_2 no es necesario realizar modificaciones. Pero para el valor τ_3 se debe realizar alguna modificación en la trayectoria para que cambien las velocidades y aceleraciones en ese tramo y que el torque necesario no supere al de arranque del motor utilizado. Esta es una de las posibles fuentes de error que no se visualiza en el modelo cinemático, por lo cual las grandes diferencias en la trayectoria del robot que fue analizada con el modelo cinemático y graficada en la figura 20, pueden deberse en gran parte a este motivo.

Variación de la masa y del coeficiente de fricción

A continuación, se experimentó con la variación del coeficiente del rozamiento y de la masa del robot, realizando el mismo análisis, y con el fin de verificar el modelo dinámico y la respuesta de torque.
Los resultados fueron los siguientes:
En las figuras 30 y 31 se graficó la variación de la masa en una constante k. Se puede observar que se requiere un torque cada vez más significativo para mover masas cada vez más grandes.

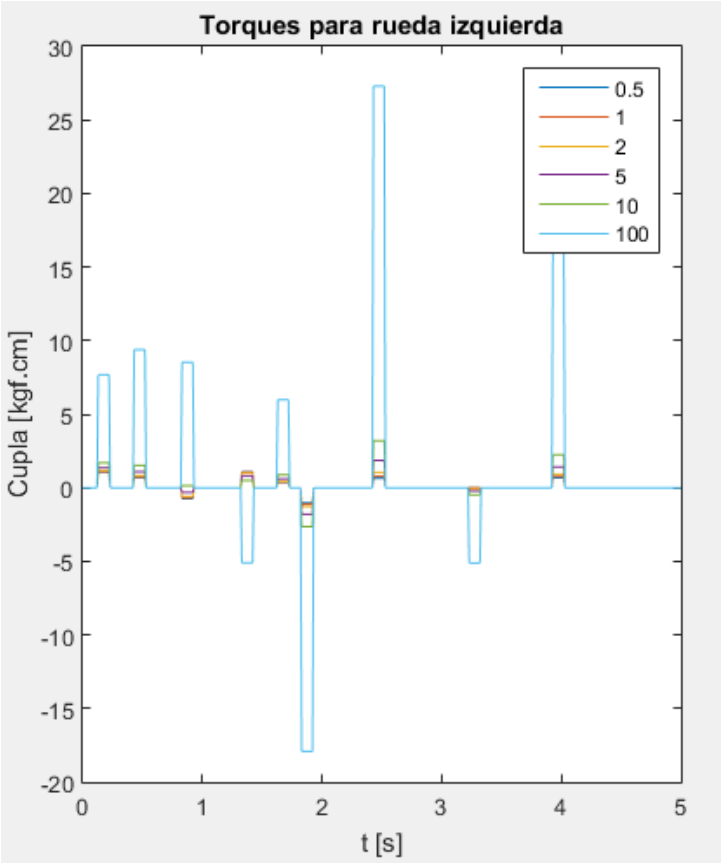


Figura 30: Torques para rueda izquierda con parametrización de la masa

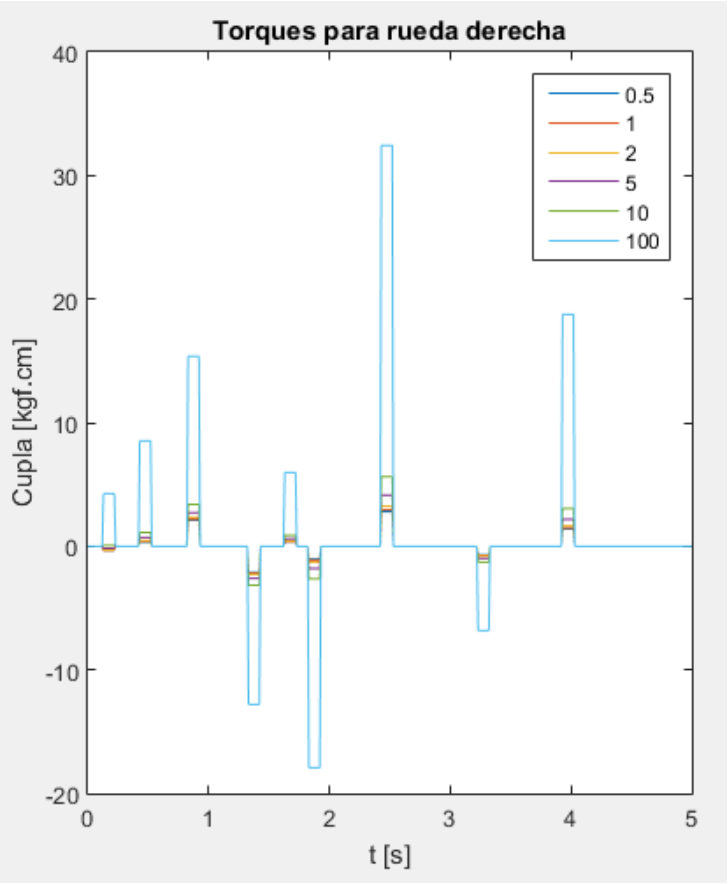


Figura 31: Torques para rueda derecha con parametrización de la masa

En las figuras 32 y 33 se graficó la variación de la fricción en una constante k. Se puede observar que se necesitan grandes variaciones de la fricción (valores muy grandes en proporción al utilizado como referencia) para obtener un cambio significativo en el torque necesario. Para valores mayores a 10000 veces el original, el torque necesario aumenta en valor y se deforma.

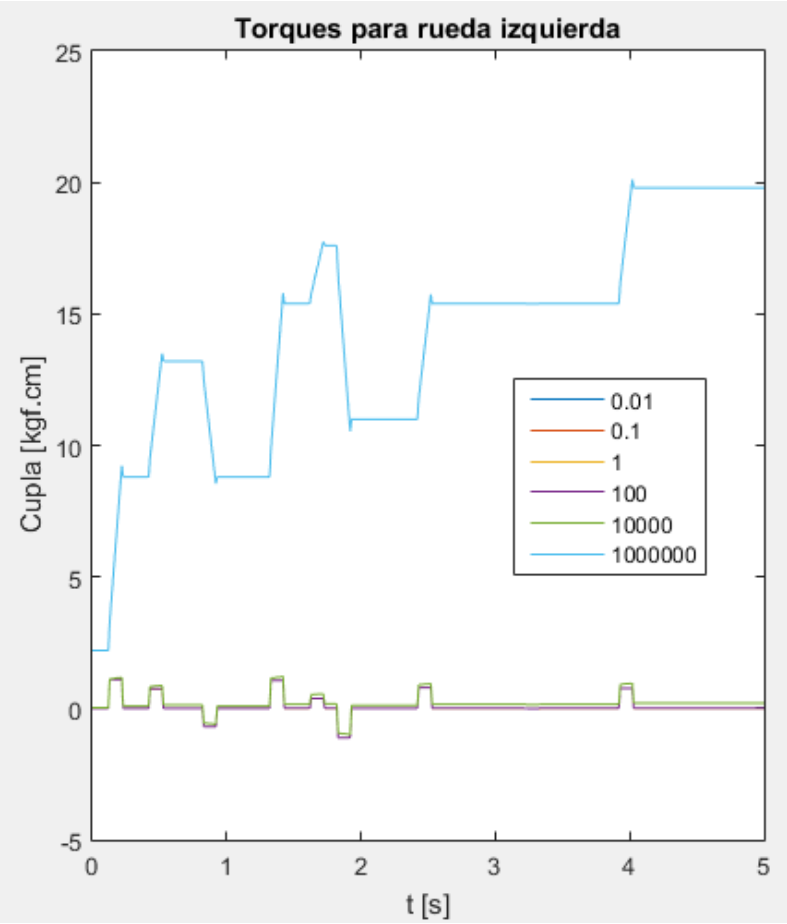


Figura 32: Torques para rueda izquierda con parametrización de la fricción

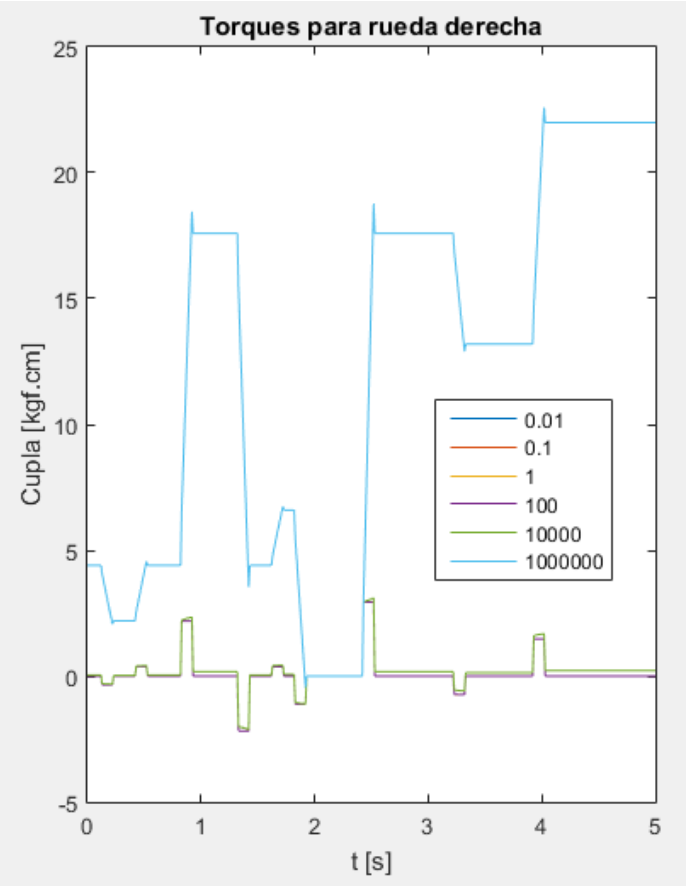


Figura 33: Torques para rueda derecha con parametrización de la fricción

Conclusiones

Mediante el presente trabajo hemos demostrado la utilidad de los modelos dinámicos matriciales aplicados a robótica, y al mismo tiempo hemos podido caracterizar el robot bajo ensayo, pudiendo de esta manera determinar aproximadamente las aceleraciones máximas para las cuales el motor puede responder con el torque del que dispone.

Existen diversas fuentes de error que condicionan tanto el análisis como el funcionamiento del robot. Entre estas se encuentran el modelado mecánico incompleto hecho en solidworks, el desconocimiento del coeficiente de rozamiento del robot y el uso de uno comparable, así como también, la posible variación del mismo en un terreno irregular.

Se espera que a futuro este trabajo forme parte de un análisis de mayor envergadura en donde pueda complementarse con el análisis cinemático del mismo y culminar el proceso con la síntesis de un compilador para que el usuario lo pueda programar.

Referencias

[Tzafestas]: Introduction to Mobile Robot Control, Spyros G Tzafestas, Elsevier, 2013, ISBN 0124171036, 9780124171039.

[Ivanjko]: Ivanjko, E., Petrinić, T., Petrović, I., Modelling of Mobile Robot Dynamics Proceedings of the 7th EUROSIM Congrees on Modelling and Simulation Vol. 2.

[Dhaouadi] Dynamic Modelling of Differential-Drive Mobile Robots using Lagrange and Newton-Euler Methodologies: A Unified Framework. Rached Dhaouadi, Ahmad Abu Hatab. 2013.

Desarrollo de un Compilador para un robot móvil diferencial de dos ruedas

Resumen— En el presente trabajo se presenta el desarrollo de un compilador para un robot móvil con configuración diferencial de dos ruedas. El mismo genera código interpretable por la plataforma del robot a partir de un pseudocódigo amigable al usuario.

Palabras Clave--- Robótica, compilador, parser, lexer, ANTLR

INTRODUCCIÓN

El uso de compiladores se ha vuelto imprescindible para la programación tanto de sistemas embebidos como de sistemas informáticos en general. Es por eso que el uso de herramientas como parsers y lexers es de suma importancia para poder simplificar la programación a futuro.

Particularmente, en el ámbito de la robótica, tener acceso a un compilador permite extender la cantidad de profesionales capacitados para utilizar la herramienta, de esta manera haciéndola más masiva, ya que provee una abstracción en el código para quien se encuentra interesado en el desarrollo de la aplicación.

En el presente trabajo se muestra el proceso para la creación de un compilador para un robot diferencial de dos ruedas. El proceso consiste en la creación de un parser y un lexer con reglas bien definidas y formas controladas de subsanar errores en el código de entrada tanto léxicos como gramáticos.

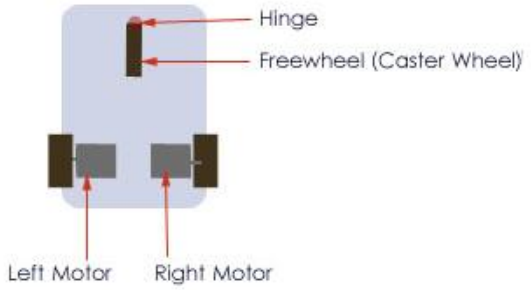


Ilustración 1: Robot diferencial de dos ruedas

MATERIALES Y MÉTODOS

La herramienta principal utilizada en el presente trabajo se trata del ANTLR, en conjunto con su interfaz gráfica ANTLRWorks. La misma proporciona un entorno de desarrollo integrado escrito en lenguaje Java, el cual permite de manera sencilla poder definir reglas tanto para el parser como para el lexer. En conjunto con el editor, esta herramienta provee un depurador de código (debugger) que permite verificar la integridad del parser y el lexer a través de un código de entrada.

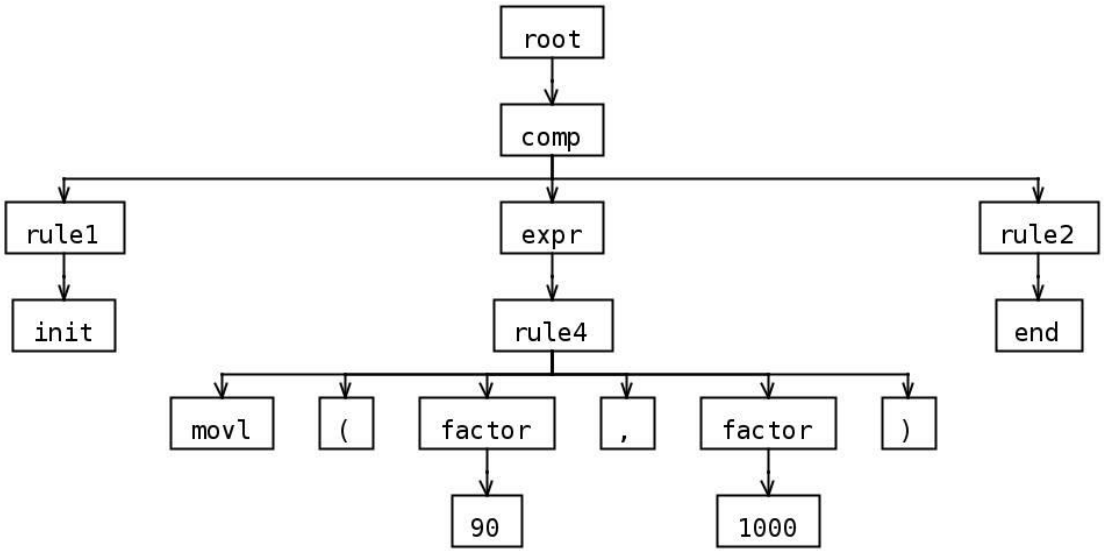


Ilustración 2: Plataforma ANTLRWorks

El robot a programar en este trabajo se trata de un Múltiplo N6. Dicha máquina diferencial fue creada con fines educativos y la placa de control cuenta con un microcontrolador Atmel que posee instalado el bootloader de Arduino. De esta manera, conociendo la plataforma objetivo, queda determinada la salida del compilador, que será código interpretable por el sistema Arduino.

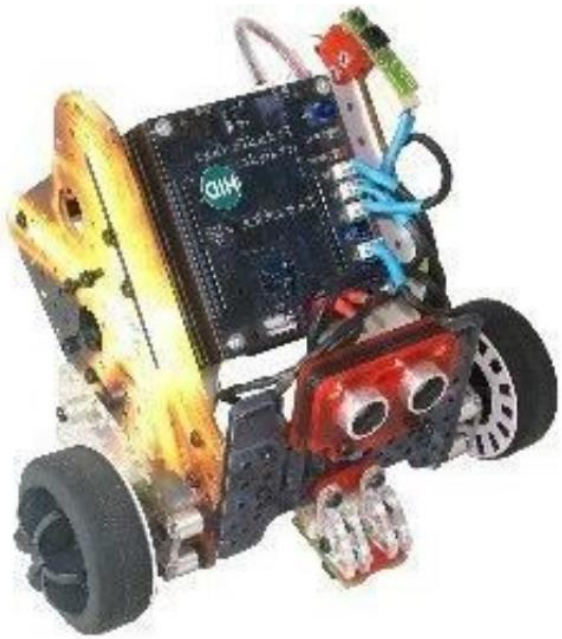


Ilustración 3: Robot diferencial Múltiplo N6

La plataforma de trabajo para la utilización del ANTRL utilizada fue una PC con sistema operativo Linux, más precisamente la distribución Ubuntu 14.04 LTS.

DESARROLLO

La primera acción que se realizó fue definir las palabras clave o “tokens” para el código de entrada. Al ser un compilador sencillo, las instrucciones también lo son. Por ende, se definen dos tipos de movimientos para el robot: lineal y circular.
Las palabras clave utilizadas, que se corresponden con las posibles instrucciones del código de entrada fueron las siguientes:

Init: realiza las rutinas de inicialización.
End: Cierra el programa de salida.
Movc: Realiza un movimiento circular.
Movl: realiza un movimiento lineal.
Wait: Espera sin moverse.

Cabe destacar que este compilador no es “case sensitive”, por lo tanto, interpreta dichas instrucciones tanto en minúscula como en mayúscula.

Luego fue necesario definir directivas para el lexer, de esta manera el compilador entiende y diferencia números enteros de caracteres o números en punto flotante.
Asimismo, se definió una función llamada factor, que se encarga de interpretar números de más de un dígito, de manera que el compilador pueda convertir una cadena de caracteres en ASCII en un número entero.

Para el manejo de variables, se utiliza una funcionalidad ya incorporada en Java, las hashtables. De esta manera fue posible desligarse completamente del manejo de memoria.

Para el parser, al ser imprescindible que el código posea la inicialización al principio y el cierre al final, se definió la regla inicial de tal manera que no pueda existir un código de entrada que no comience con la “init” y termine con la instrucción “end”. Si esto llegara a pasar, el interpretador genera un error.

Entre las instrucciones “init” y “end” pueden apilarse cuantas instrucciones se quiera. Cada una generará su código de salida y de esta manera se logrará la secuencia de movimientos deseada por el usuario.

Para definir cada regla del parser se utiliza la directiva “rule:” provista por el ANTLR, y luego se utiliza código Java para imprimir el código de salida y tomar decisiones lógicas en caso de ser necesario.
Para las instrucciones que requieren argumentos, fue necesario generar una sintaxis particular para cada regla y especificarla en el ANTLR. La sintaxis utilizada está tomada del lenguaje C, donde los argumentos se ingresan entre paréntesis luego del nombre de la función separados por comas en caso de ser más de uno, y a diferencia del lenguaje C, en este caso las instrucciones no terminan con punto y coma sino con un fin de línea generado por la tecla ENTER. (Caracter \n).

Los argumentos tomados para cada instrucción fueron:

wait(time)
time: [int] milisegundos

movl (velocidad, tiempo)
velocidad: [int] 0-100
tiempo: [int] milisegundos.

movc(velocidad, tiempo, delta_v, clockwise)
velocidad: [int] 0-100
tiempo: [int] milisegundos
delta_v: [int] 0-velocidad
clockwise: [int] 1=clockwise, else anticlockwise.

Las instrucciones *init* y *end* no llevan argumentos.

En la instrucción movc, para el parámetro delta_v se tomó en cuenta los resultados obtenidos en los trabajos anteriores de cinemática y dinámica, particularmente, el factor de corrección propuesto basado en los momentos donde las velocidades ingresadas al robot tenían grandes variaciones entre sí (grandes valores de delta_v). Los valores máximos que permite el compilador contempla esto.

A continuación, se presenta un ejemplo de código de entrada:

```
init                    //realiza la inicialización.
movl(90, 1000)          //movimiento lineal al 90% de la velocidad
                        //máxima durante 1 segundo.
wait(500)               //espera quieto 500ms
movc(90, 1000, 40, 1)   //movimiento circular al 90% de la vel.
                        //máxima durante un segundo, con un radio
                        //de apertura del 40% y en sentido horario.
end                     //rutina de cierre
```

CONCLUSIONES

El presente trabajo ha resultado muy útil dentro del marco de formación para la carrera de Ingeniería Electrónica de la UTN. El mismo ha permitido interiorizar una herramienta de trabajo nueva como es el ANTLR y haberle dado una aplicación concreta de manera tal de asentar el conocimiento de la misma.
Se logró incorporar al compilador los resultados obtenidos en los experimentos realizados anteriormente dentro del estudio de la cinemática y la dinámica del robot.
Al mismo tiempo, se espera a un futuro poder extender el presente trabajo agregando instrucciones más complejas para el mismo robot, extenderlo a otras configuraciones robóticas y poder utilizar la misma herramienta para la realización de compiladores en cualquier otro ámbito.

REFERENCIAS

[Barrientos]: Barrientos, Peñin, Balaguer y Aracil, Fundamentos de robótica, Mc Graw Hill. 2001.

[Seattlerobotics]: http://www.seattlerobotics.org/encoder/aug97/basics.html

[Terrence Parr]: The Definitive ANTLR 4 Reference

[Aguirre-Carlucci]: Desarrollo de un compilador y simulación para el control de un robot manipulador de 2 brazos. 2014.

APÉNDICE: EJEMPLO DE CÓDIGO DE ENTRADA Y DE SALIDA

ENTRADA

```
init
movl(90, 1000)
wait(500)
movc(90, 1000, 40, 1)
movc(90, 1000, 40, 0)
movc(90, 1000, 100, 1)
end
```

SALIDA

```
/*
 --ROBOTICA 2016--
 JAVIER SANKOWICZ-LUCAS PAGLIA
*/


#include <DCMotor.h>
DCMotor motor1(M1_EN, M1_D0, M1_D1);
DCMotor motor0(M0_EN, M0_D0, M0_D1);
void setup() {

//movimiento lineal
motor0.setSpeed(-90);
motor1.setSpeed(90);
delay(1000);

//espera n milisegundos
motor0.setSpeed(0);
motor1.setSpeed(0);
delay(500);

//movimiento circular
motor0.setSpeed(-90);
motor1.setSpeed(50);
delay(1000);

//movimiento circular
motor1.setSpeed(90);
motor0.setSpeed(-50);
delay(1000);

//movimiento circular
motor0.setSpeed(-90);
motor1.setSpeed(0)
delay(1000);
}
motor0.setSpeed(-0.0);
motor1.setSpeed(0.0);

//--fin del programa--

void loop() {
}
```