**PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ**

**FACULTAD DE CIENCIAS E INGENIERÍA**

**ROBÓTICA E INTELIGENCIA ARTIFICIAL (1MTR53)**

**Laboratorio 6**

(Semestre 2024 – 2)

**Indicaciones generales:**

* **Importante:** Revisar las **observaciones** en cada ejercicio.
* Considerar que el **radio de las ruedas**, **ancho** y **largo** de la plataforma omnidireccional son 0.04 m, 0.21 m y 0.195 m, respectivamente.
* En el nombre de cada archivo agregar el número de grupo siguiendo el formato***PX\_GrupoY*.py** o ***PX\_GrupoY.xls***, agregar en un archivo de texto los integrantes del grupo con el formato ***Integrantes\_GrupoY.txt*** y comprimir todos los archivos en una carpeta con el formato de nombre ***Lab6\_GrupoY.zip*** y subirlo a Paideia.

**Pregunta 1 (8 puntos)**

Realizar la programación en **Arduino UNO** de la plataforma omnidireccional para que pueda desplazarse de todas las formas presentadas en la Figura 1.

|  |  |  |  |
| --- | --- | --- | --- |
| (a) | (b) | (c) | (d) |
|  |  |  |  |
| (e) | (f) | (g) | (h) |
|  |  |  |  |
|  | (i) | (j) |  |

Figura 1: Movimientos del módulo omnidireccional.

La plataforma debe moverse en la secuencia desde (a) hasta (j) durante 1 s en cada caso y esperar 0.5 s (velocidad a cero) entre secuencias. Para ello, revisar el código P1.ino donde se ejecutan las secuencias a partir de un arreglo de las velocidades absolutas respecto al eje fijo de la plataforma. Las velocidades en cada eje se guardan en los arreglos correspondientes y en cada iteración se calculan las velocidades de las ruedas requeridas para que la plataforma se mueva de acuerdo a la secuencia actual, luego se pasa a la siguiente secuencia. Considerar la distribución de ruedas, y los ejes **X (verde)**, **Y (rojo)** y **Z (azul)** fijos en la plataforma presentados en la Figura 2.

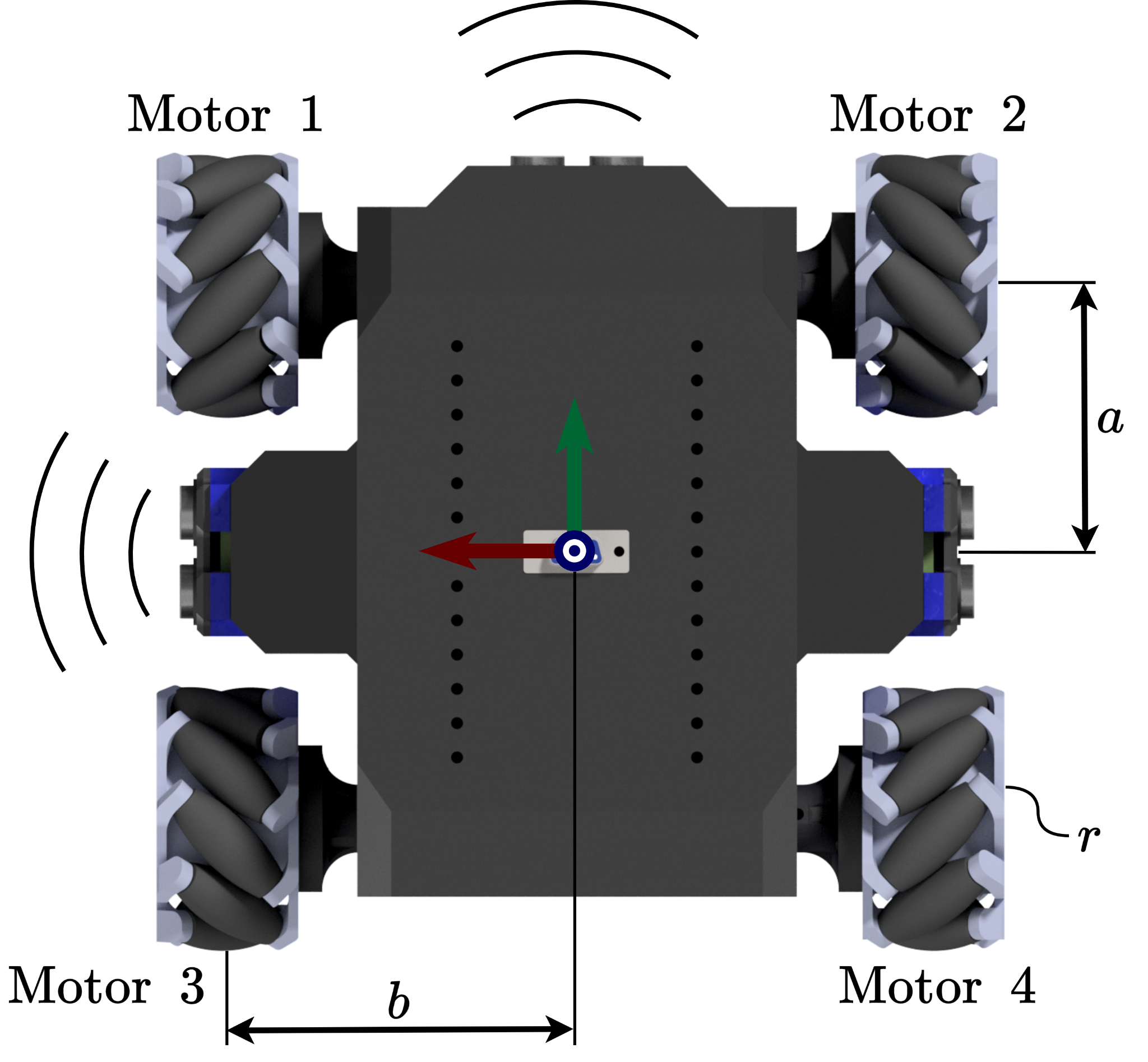
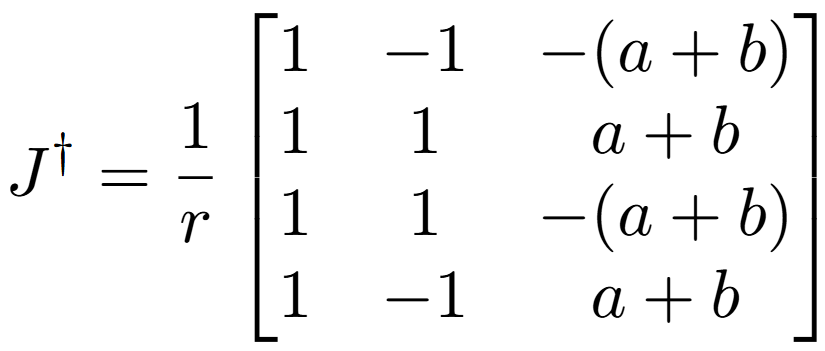


Figura 2: Motores y eje fijo de la plataforma omnidireccional.

Además, considerar la pseudoinversa de la matriz Jacobiana de velocidad que permite obtener las velocidades de las ruedas a partir de la velocidad en eje fijo al robot como:



Donde:

* : Longitud de la plataforma dividido entre dos, en m.
* : Ancho de la plataforma dividido entre dos, en m.
* : Radio de la rueda, en m.

**Procedimiento:**

1. Defina las velocidades absolutas respecto al eje fijo en el robot de acuerdo a las secuencias que se desean realizar, para ello modificar las variables vxSeq[], vySeq[] y vwSeq[]. (2 puntos)
2. Defina las dimensiones del robot, el valor máximo de control del driver y la máxima velocidad angular en RPM en las variables correspondientes. (2 puntos)
3. Complete la función CalculateVelAng() a partir de la pseudoinversa de la matriz Jacobina para obtener la velocidad requerida de las ruedas a partir de la velocidad absoluta respecto al eje fijo en la plataforma, donde la variable w[] contiene la velocidad de cada rueda en rad/s. (2 puntos)
4. Complete el código en loop() tal que en cada iteración se ejecute una nueva secuencia hasta que el robot realice las diez secuencias solicitadas. (2 puntos)

**Observaciones:**

* Se les exige utilizar la cinemática inversa del módulo omnidireccional.
* **90 RPM equivale a 100% de Duty Cycle** (0-255), aproximadamente.
* Utilizar una velocidad controlable de 0.1 - 0.2 m/s en los ejes en X e Y, y una velocidad angular máxima absoluta de π/5 rad/s.

**Pregunta 2 (6 puntos)**

Para garantizar la correcta velocidad de giro en todos los motores de la plataforma, se les pide realizar el control PID de cada uno de los motores. Para ello se les pide utilizar el método Ziegler-Nichols y ajustar los parámetros , y de cada motor.

**Procedimiento:**

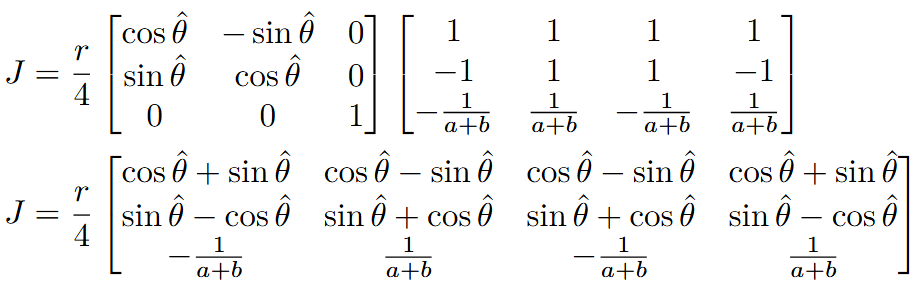
1. Utilizar el archivo Zieger.ino y cargarlo en el microcontrolador para mover un motor independiente, modificando la variable motor, y enviar por comunicación serial el tiempo en μs y la velocidad del motor en RPM. Analizar el código y modificar las variables waitT y totalT. Si lo considera conveniente, modificar el tiempo de muestreo de la variable sampleT para obtener mayor cantidad de muestras.
2. Con la plataforma al aire evitando que la ruedas toquen alguna superficie, conectar el arduino a un dispositivo y abrir el archivo ReadMotorData.ipynb, colocar el puerto serial COM respectivo en el código al cual está conectado el microcontrolador. Al ejecutar la primera parte del código se obtendrá la información del tiempo y velocidad del ensayo. La segunda parte del código mostrará la respuesta en una gráfica la cual puede ser utilizada para el análisis. Finalmente, la tercera parte exportará los valores a un archivo de texto lo cual también permitirá realizar el análisis en otro software como *Excel*.
3. Obtener los parámetros del controlador PID para el motor a partir de la gráfica o utilizando otro software y repetir para cada uno. **Documentar lo realizado en la primera hoja del archivo *ControlAndPoseEstimationAnalysis.xls* (agregar ganancias y salida del *Serial Plotter*).**
4. Completar el código del archivo P2.ino agregando los valores de las ganancias obtenidas para poder controlar el movimiento del carrito con una velocidad de 0.15 m/s en el eje X fijo al robot y con un tiempo determinado, con el fin de probar que la plataforma se mueva en línea recta una distancia esperada (1 m por ejemplo) modificando las variables vx, vy, vw y time.
5. En caso no llegue a la distancia deseada, ajustar los parámetros o determinar una forma de reducir el error.
6. Análisis: ¿Qué valores se han modificado con respecto a los obtenidos por el método y por qué?

**Observaciones:**

* Recordar que vel[] es la velocidad medida del encoder en RPM y vt[] es la velocidad deseada en cada rueda en RPM.
* Antes de encender la plataforma, **AVISAR AL JEFE DE PRÁCTICA** para que realice las validaciones correspondientes.
* Antes de probar el robot en el suelo con el archivo P2.ino se recomienda probar el control con las ruedas al aire y verificar que la sintonización logra alcanzar el valor deseado, lo cual se puede visualizar con la herramienta *Serial Plotter* del *Arduino IDE*.
* Si la respuesta es muy lenta, es decir que llega al valor deseado en un tiempo prolongado se recomienda **aumentar el valor de la ganancia integrativa**. Si se presentan muchas oscilaciones o cambios en la velocidad muy variables se recomienda **reducir la ganancia derivativa al mínimo**. Si la velocidad aumenta muy rápidamente al inicio se recomienda **disminuir la ganancia proporcional**.

**Pregunta 3 (6 puntos)**

Para estimar la posición y orientación de la plataforma se pueden utilizar la información del sensores de cada rueda. En este sentido, si los movimientos de la plataforma se simplifican a secuencias consecutivas de rotación y traslación, es decir sin movimientos combinados, la matriz Jacobiana de velocidad puede considerarse constante en cada secuencia de movimiento:



Donde:

* : Ángulo de orientación actual del robot, en rad/s.

Se le pide: (1) aplicar una velocidad angular de -π/6 rad/s, (2) detener el vehículo y (3) aplicar una velocidad lineal de 0.15 m/s en el eje X fijo al robot, de forma consecutiva (ver Figura 3). En este sentido, se debe implementar un código en la función BilinealEstimation() que calcule y muestre la posición cartesiana y orientación relativas de la plataforma utilizando el método bilineal. Además, se deben **realizar tres (03) ensayos con DIFERENTES TIEMPOS en cada secuencia** (giro, espera y desplazamiento) y anotar en la hoja de cálculo compartido en PAIDEIA el valor final esperado, valor final real y error de las tres variables. Además, no se deben modificar los valores de las ganancias del controlador PID obtenidas del ejercicio anterior en todas las pruebas de este ejercicio.

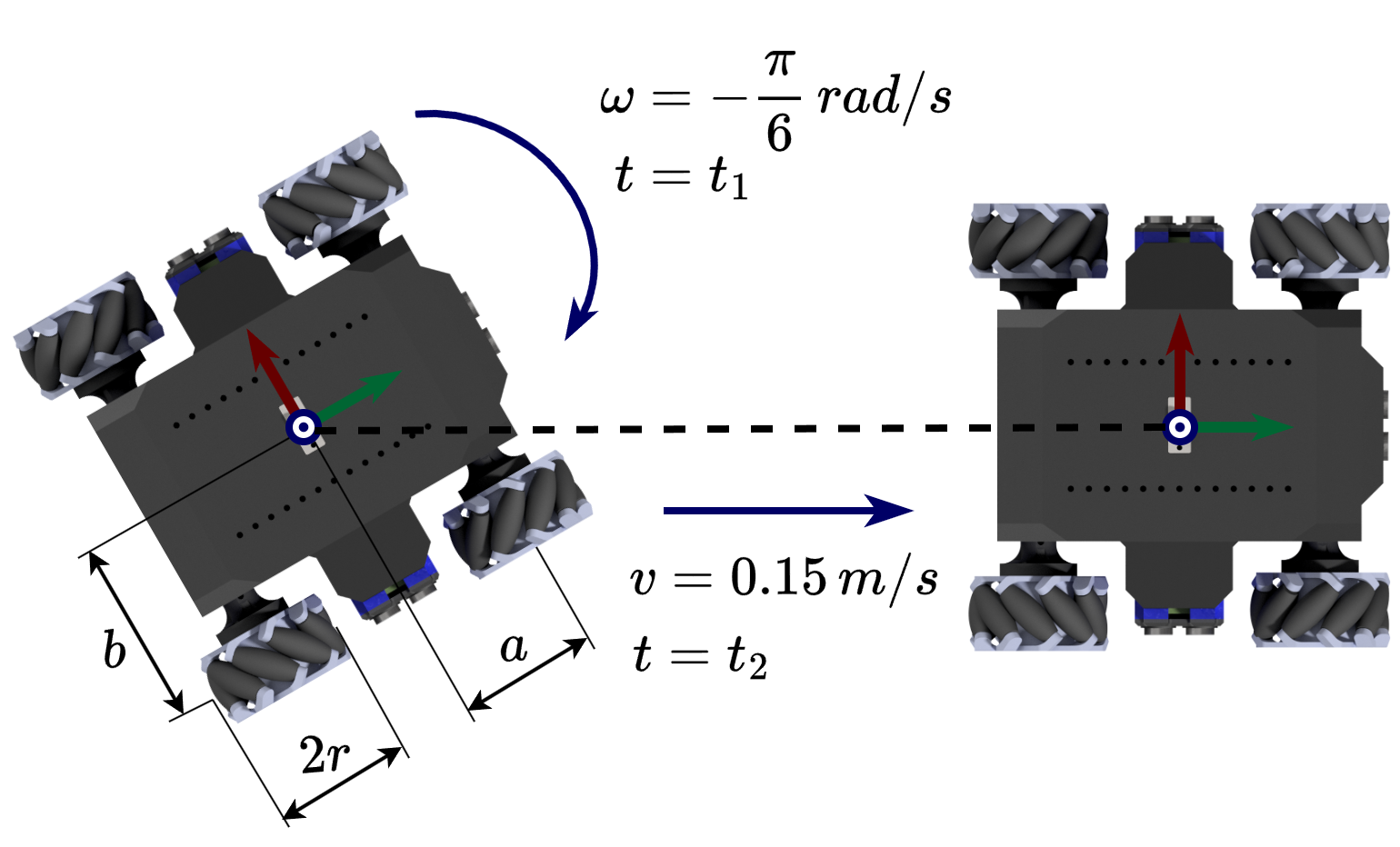


Figura 3: Secuencia de movimiento de la plataforma omnidireccional.

**Procedimiento:**

1. Completar el archivo P3.ino con las variables y código obtenido de las experiencias anteriores (cinemática inversa, dimensiones, ganancias del controlador, etc.)
2. Completar las secuencias de movimiento en las variables correspondientes y definir los tiempos para cada secuencia en la variable T[].
3. Completar el código para computar la variable meanVelAng[], esta variable contiene el **promedio de las velocidades angulares en rad/s** de cada rueda del instante actual y anterior, para ello se debe considerar sus signos registrados en las variables sgn[] y sgnPrev[]. (1 punto)
4. Completar la función BilinealEstimation() utilizando para estimar la posición en las variables theta, x e y utilizando sus valores previos guardados en las variables thetaPrev, xPrev e yPrev, la variable meanVelAng[] y la diferencia temporal entre muestras de la variable deltaT (tiempo de muestreo). (2 puntos)
5. Debido a que el microcontrolador se reinicia al conectarlo a un COM serial, al finalizar un ensayo las posiciones lineales y angulares se guardarán en la memoria EEPROM del microcontrolador y estas se mostrarán por comunicación serial al inicio del siguiente reset de la plataforma. Por ello, **luego de finalizar un ensayo se debe reconectar el microcontrolador a un dispositivo y anotar los valores mostrados al inicio.**
6. Llenar la segunda hoja de cálculo de ***ControlAndPoseEstimationAnalysis.xls*** con los valores de las posiciones finales esperadas después de cada ensayo, calculadas en función a los tiempos y velocidad de las secuencias, y la obtenida por el algoritmo de estimación. **Se otorgará 1 punto por cada ensayo realizado y registrado**.
7. Análisis: ¿El resultado final es el esperado?¿Cuáles son las causas de desviación entre los valores reales y esperados?¿Qué propondría implementar para disminuir el error y mejorar el control de la plataforma?

**Observaciones:**

* Recordar que velAng[] es la velocidad medida por los encoders en rad/s.
* El control PID implementado se ejecuta con valores de velocidad absoluta y se considera que las ruedas tienen la misma velocidad que la controlada (ya que no se cuenta con el segundo canal del encoder). A partir de ello, se ha guardado en las variables sgn[] y sgnPrev[] los signos de las velocidades de cada rueda en el tiempo actual y un instante anterior. Entonces para obtener la velocidad real de cada rueda en el instante actual y pasado, se debe multiplicar la velocidad absoluta por su signo correspondiente.
* Para reducir el tiempo de cálculo, se ha realizado el cálculo del seno y coseno del ángulo de orientación para utilizarlos en la integración numérica.
* Para este ensayo particular, las posiciones lineales y angulares finales esperadas se pueden obtener con las siguientes expresiones que dependen de la **velocidad angular ω** en el primer ensayo, la **velocidad lineal Vx** del robot en el segundo ensayo, y los **periodos de duración t1 y t3** del la primera y tercera secuencia, respectivamente:

