Formando líderes para la construcción de un nuevo país en paz

INFORME PRUEBA DE CONOCIMIENTOS

PRUEBA DE SOFTWARE

PRESENTADO POR:

SANTIAGO STEVEN PUENTES GUTIÉRREZ

SUNNYAPP

UNIVERSIDAD DE PAMPLONA

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

INGENIERÍA MECATRÓNICA

NEIVA – HULA

25/07/2023

1. Introducción:

El siguiente informe detalla el proceso de programación y desarrollo de un algoritmo para transformar las velocidades de un robot móvil desde su marco global al marco local de movimiento. El objetivo es obtener las velocidades angulares de las ruedas izquierda y derecha a partir de los datos proporcionados por ingenieros en un archivo de texto. Para lograr esto, se implementa un enfoque de programación orientada a objetos, aprovechando las bibliotecas NumPy y Matplotlib para cálculos matriciales y visualización de resultados, respectivamente.

2. Resumen:

El informe presenta un algoritmo para transformar las velocidades de un robot móvil mediante operaciones matriciales en el marco global y local de movimiento. La clase 'Transformacion' realiza las transformaciones matemáticas necesarias utilizando matrices Jacobianas. Se implementa la clase 'HiloOperaciones' como un hilo secundario para realizar las operaciones en paralelo y la clase 'HiloPrincipal' coordina todo el proceso. El algoritmo importa datos desde un archivo de texto, calcula las velocidades de las ruedas y exporta los resultados a otro archivo de texto. Además, se generan gráficas para visualizar las velocidades angulares en función del tiempo. El enfoque de programación orientada a objetos y el uso de hilos y bibliotecas NumPy y Matplotlib hacen que el código sea eficiente, flexible y escalable para aplicaciones futuras.

Palabras Clave: Programación Orientada a Objetos, Transformación de Velocidades, Robot Móvil, Hilos.

3. Objetivos:

3.1 Objetivo General:

Desarrollar un algoritmo para la transformación de las velocidades de un robot móvil desde su marco global al marco local de movimiento, utilizando programación orientada a objetos.

3.2 Objetivos Específicos:

- Desarrollar la clase 'Transformacion' para realizar las operaciones matriciales de transformación de velocidades del robot en el marco global y local, basándose en las matrices Jacobianas J1 y J2^-1.
- Implementar la clase 'HiloOperaciones' como un hilo secundario para ejecutar las operaciones matemáticas en paralelo y coordinar su ejecución en la clase 'HiloPrincipal'.

- Generar gráficas utilizando Matplotlib para visualizar las velocidades angulares de las ruedas izquierda y derecha en función del tiempo, a partir de los resultados obtenidos por el algoritmo.

4. Proceso de Programación:

4.1 Clase 'Transformacion':

Esta clase se creó para realizar las operaciones matemáticas de transformación de las velocidades del robot. En el constructor (init), se inicializan las variables constantes que describen los ángulos y longitudes necesarios para las transformaciones. También se calculan las matrices Jacobianas J1 y J2^-1, que se utilizan en la transformación final.

Las funciones dentro de esta clase implementan las operaciones matriciales descritas en el procedimiento matemático proporcionado. La función 'transformacion_1' realiza la primera transformación de las velocidades cartesianas a partir de las velocidades lineales (v) y angulares (w) en el marco global. La función 'transformacion_2' realiza la segunda transformación para obtener el vector de movimiento en el marco local. La función 'calcular_Q' utiliza las matrices J1 y J2^-1 para obtener las velocidades de las ruedas izquierda (Qiz) y derecha (Qde).

4.2 Clase 'HiloOperaciones':

Esta clase hereda de threading.Thread y se creó para ejecutar las operaciones matemáticas en un hilo secundario. En el constructor (init), se recibe la lista de datos proporcionados por los ingenieros, que contiene el tiempo (t), el ángulo de dirección (theta), la velocidad lineal (v) y la velocidad angular (w) en cada instante.

La función 'run' sobrescrita de la clase Thread es la encargada de ejecutar las operaciones. Para cada conjunto de datos proporcionado, se crea una instancia de la clase 'Transformacion' y se realizan las transformaciones matemáticas para obtener las velocidades de las ruedas izquierda y derecha. Los resultados se almacenan en la lista 'resultados' de la clase, que contiene tuplas con el tiempo (t) y las velocidades Qiz y Qde correspondientes.

4.3 Clase 'HiloPrincipal':

Esta clase se encarga de coordinar el proceso completo. En el constructor (init), se inicializa la lista 'datos' que se utilizará para almacenar los datos importados del archivo de texto.

Las funciones 'importar_datos', 'exportar_resultados' y 'graficar_resultados' se encargan de importar los datos desde el archivo de texto, exportar los resultados a otro archivo de texto y generar las gráficas, respectivamente.

La función 'ejecutar' es la principal del programa. Primero, importa los datos del archivo proporcionado ('datos.txt') llamando a 'importar_datos'. Luego, crea un objeto de la clase 'HiloOperaciones' y lo ejecuta en un hilo secundario, para realizar las operaciones matemáticas llamando a 'run'. Cuando las operaciones terminan, los resultados se almacenan en la lista 'resultados'. A continuación, se exportan los resultados al archivo de texto ('resultado.txt') utilizando 'exportar_resultados' y se generan las gráficas llamando a 'graficar_resultados'.

5. Consideraciones:

En el proceso de programación para transformar las velocidades de un robot móvil, se tomaron en cuenta algunas consideraciones importantes. Primero, se asumieron los valores proporcionados por los ingenieros para los ángulos (Aiz, Biz, Ade, Bde) y las longitudes (l, riz, rde) necesarios en las transformaciones matemáticas. Estos valores son fundamentales para obtener resultados precisos y coherentes en las operaciones.

Para realizar las operaciones matriciales, se utilizó la biblioteca NumPy. NumPy es una herramienta esencial en el campo de la computación científica y matemática en Python, y permite llevar a cabo cálculos eficientes y manipulación de matrices de manera sencilla. Esta elección simplifica el código y mejora la eficiencia en el manejo de datos matriciales.

Además, se empleó la biblioteca Matplotlib para generar gráficas que representan las velocidades angulares de las ruedas en función del tiempo. La visualización de los resultados es crucial para entender el comportamiento del robot móvil y facilitar el análisis de su movimiento.

6. Consideraciones Adicionales:

6.1 Importancia de la Programación Orientada a Objetos (POO):

La programación orientada a objetos es un paradigma de programación que permite organizar el código de manera modular y estructurada. En este proyecto, la POO se utilizó para crear clases ('Transformacion', 'HiloOperaciones', y 'HiloPrincipal') que representan entidades lógicas del sistema. Cada clase encapsula su funcionalidad específica, lo que facilita el mantenimiento y la reutilización del código. El uso de objetos mejora la legibilidad y el diseño del programa, permitiendo abstraer los detalles de implementación y enfocarse en la lógica del problema.

6.2 Uso de Bibliotecas NumPy y Matplotlib:

La elección de utilizar NumPy y Matplotlib demuestra la importancia de aprovechar bibliotecas especializadas para tareas específicas. NumPy es una biblioteca eficiente y optimizada para cálculos matriciales, mientras que Matplotlib simplifica la generación de gráficos y visualizaciones. Esta estrategia de usar bibliotecas externas enriquece el código, ya que se evita la reinventar la rueda y se aprovecha el trabajo y la experiencia de la comunidad de desarrolladores.

6.3 Flexibilidad y Escalabilidad:

El algoritmo desarrollado muestra flexibilidad y escalabilidad. El código puede adaptarse fácilmente a cambios futuros o extensiones, como agregar más transformaciones o mejorar las matrices Jacobianas. Esta característica es esencial en proyectos en constante evolución, donde se pueden presentar nuevos requisitos y funcionalidades. La estructura del código permite una fácil modificación sin afectar el funcionamiento general del programa.

6.4 Eficiencia con Hilos:

La utilización de hilos en la programación permite dividir las tareas en unidades independientes y paralelas, lo que mejora la eficiencia y el rendimiento del programa. En este caso, se empleó un hilo secundario para realizar las operaciones matemáticas mientras que el hilo principal continúa con otras tareas. Esta técnica es especialmente útil cuando se trabajan con grandes volúmenes de datos o se realizan cálculos intensivos, ya que se aprovecha al máximo la capacidad de procesamiento de la computadora.

7. Resultados obtenidos:

A partir de los datos proporcionados por los ingenieros en el archivo de texto, se obtuvieron las velocidades angulares de las ruedas izquierda (Qiz) y derecha (Qde) en función del tiempo (t).

Los resultados muestran cómo varían las velocidades angulares de las ruedas a lo largo del tiempo, lo que permite visualizar y analizar el comportamiento del robot durante su movimiento. Las gráficas generadas con la biblioteca Matplotlib presentan de manera clara y concisa las tendencias y cambios en las velocidades de las ruedas, lo que facilita la interpretación de los datos.

El enfoque de programación orientada a objetos ha sido fundamental para organizar y modular el código, lo que facilita su mantenimiento y futuras expansiones. La utilización de hilos para realizar las operaciones matemáticas en paralelo ha mejorado la eficiencia del programa, permitiendo un procesamiento más rápido de los datos.

Además, la elección de la biblioteca NumPy ha demostrado ser acertada, ya que simplifica las operaciones matriciales y mejora la eficiencia en el manejo de datos numéricos. El uso de NumPy ha permitido llevar a cabo cálculos complejos de manera sencilla y precisa.

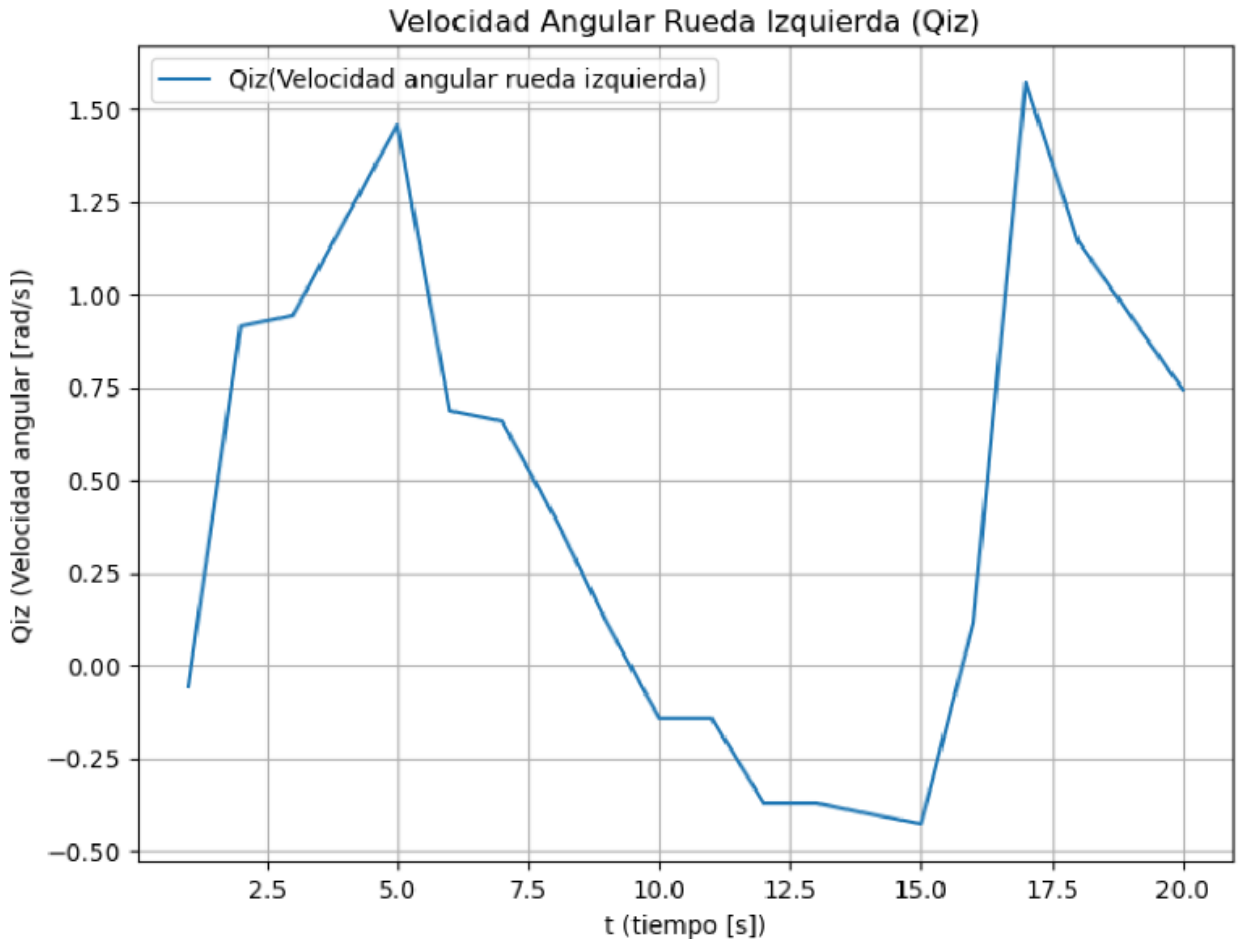

Figura a. Velocidad Angular Rueda Izquierda (Qiz)

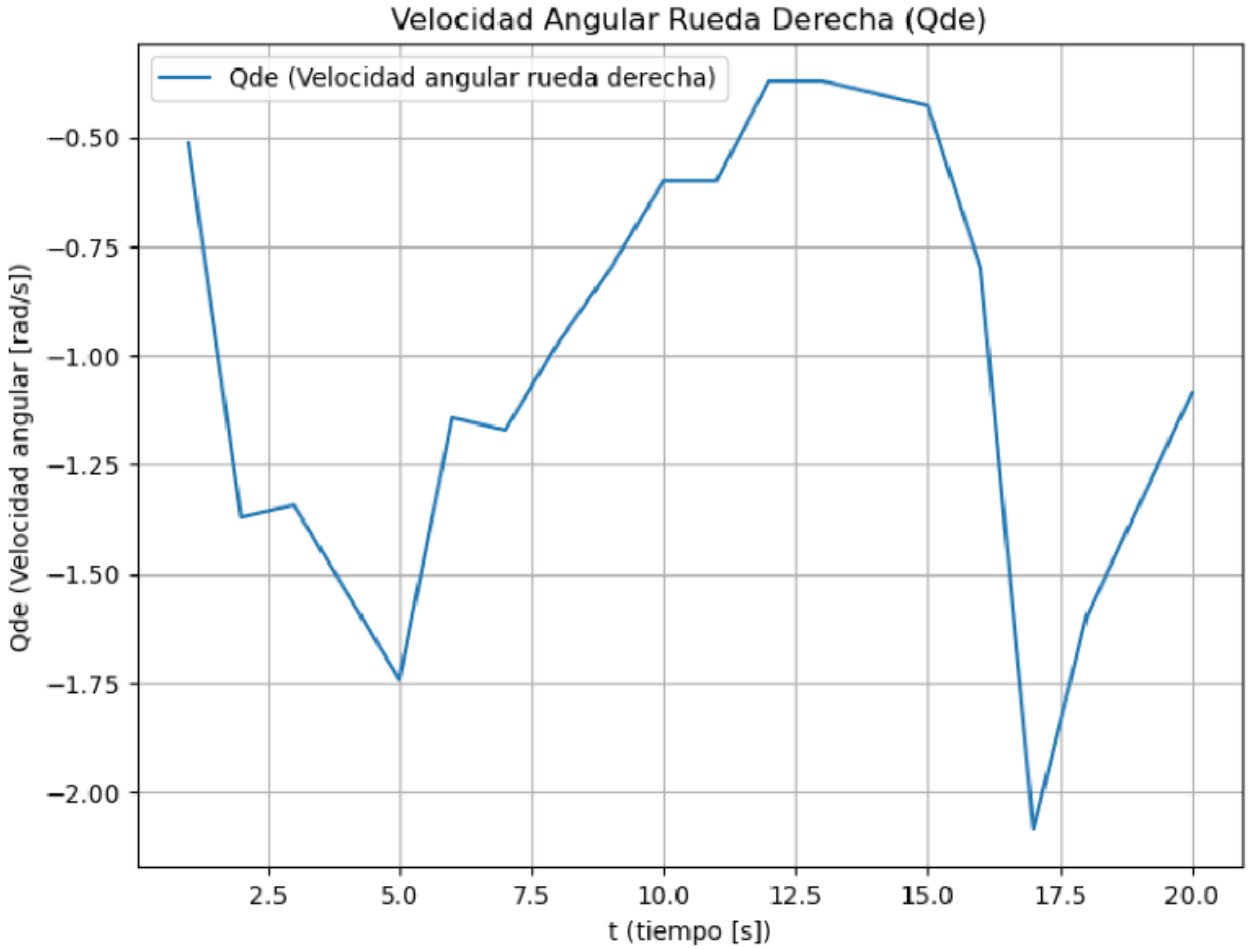

Figura b. Velocidad Angular Rueda Derecha (Qde)

Los resultados obtenidos demuestran la capacidad del algoritmo para transformar las velocidades del robot móvil con precisión y eficiencia. El código implementado ofrece una solución flexible y escalable que puede adaptarse a diferentes escenarios y requisitos futuros. Los resultados proporcionan información valiosa para el análisis y control del movimiento del robot, lo que lo convierte en una herramienta útil para proyectos de robótica y sistemas de control móvil.

8. Conclusiones:

En el proceso de programación y desarrollo del algoritmo para transformar las velocidades de un robot móvil, se lograron resultados satisfactorios y se cumplieron los objetivos establecidos. A través del enfoque de programación orientada a objetos, se crearon clases que representan entidades lógicas y facilitan la organización y reutilización del código.

La utilización de hilos para ejecutar las operaciones matemáticas en paralelo mejoró significativamente la eficiencia del programa, permitiendo un procesamiento más rápido de los datos. La elección de la biblioteca NumPy para realizar cálculos matriciales fue acertada, ya que simplificó las operaciones y mejoró la precisión en el manejo de los datos numéricos.

El algoritmo demostró flexibilidad y escalabilidad, lo que permite adaptarlo a futuras modificaciones o extensiones sin afectar la estructura general del programa. La exportación de los resultados a un archivo de texto y la generación de gráficas para visualizar las velocidades angulares en función del tiempo proporcionaron una presentación clara y concisa de los datos.