Desarrollo de un prototipo de robot móvil autónomo con capacidad de desplazamiento y transporte de paquetes.

Cortés García, Patricia Mercedes

2023

Desarrollo de un prototipo de robot móvil autónomo con capacidad de desplazamiento y transporte de paquetes

Cortés García Patricia Mercedes (décimo semestre en Ingeniería Mecatrónica)[1], Cruz López Enrique (décimo semestre en Ingeniería Mecatrónica)[1], Flores García Iván Gerardo (décimo semestre en Ingeniería Mecatrónica)[1], Mar Larios Rosa Elena (décimo semestre en Ingeniería Mecatrónica)[1], Lomas Montaudon Yvonne (profesora responsable)[1], Ochoa García Oliver (profesor asesor)[1], Girón Nieto Huber (profesor asesor)[1], González-Fernández Belinka (profesora asesora)
[1]*Universidad Iberoamericana Puebla, San Andrés Cholula, Puebla, México*

Resumen

En el presente artículo se describe el desarrollo de un prototipo de robot móvil diferencial autónomo, con capacidad de desplazamiento y transporte de paquetes de un máximo de 3 kg, mediante la integración de una cámara de control de posicionamiento y una estación de carga automatizada. Se diseñó el robot en CATIA y se realizaron pruebas de elemento finito, con base en las cuales se determinó que la construcción se haría en madera de pino de 5 mm, con esquineros impresos en 3D. Se generaron códigos en Arduino, para el control de movimiento, y Python, para la detección de objetos a través de una cámara de navegación autónoma, ubicada en la parte superior del área de trabajo. Una vez construido el robot, se hicieron pruebas de carga y trayectoria, para concluir que su resistencia era suficiente y que con este sistema se puede guiar robots e identificar la posición de paquetes, con lo cual viable aplicarlo a la industria.

Palabras clave: AGB, cámara, paquetería, paquetes, robot autónomo.
[*]**Autora Corresponsal:** patricia.cortes@iberopuebla.mx

Introducción

Las ventas en línea han ganado gran popularidad a raíz de la pandemia, tanto que en 2019 se registraron más de 44 millones de compras, la cifra más alta hasta a la fecha [1]. A pesar de que se han renovado las actividades comerciales de manera física, en el 2022 se estimó que el 61.35% de las compras se realizaron por internet y desde entonces este porcentaje ha ido en aumento [2].

Debido a esto, las empresas de paquetería se encuentran en constante competencia para liderar el mercado. Los tiempos de entrega son clave a la hora de elegir una compañía de distribución y éstos dependen de diferentes factores. Estudios revelaron que el 25% de los retrasos en las entregas se debe a errores de embalaje y etiquetado, mientras que otro 15%, a problemas de carga y descarga de los paquetes [3]. Una mala técnica en este rubro, por ejemplo, la manual, no sólo ocasiona retrasos en las entregas, sino que puede causar lesiones y fatiga en los trabajadores [4]–[6].

Otra de las razones por las que existen retrasos en las entregas es la pérdida de los paquetes dentro de la misma planta de distribución: debido a la gran cantidad de destinos que manejan, un error en el acomodo de un paquete pequeño puede hacer imposible su rastreo dentro de una planta grande.

El uso de un robot móvil autónomo para la carga y transporte de paquetes no sólo mejoraría la eficiencia en el proceso de organización y almacenamiento, sino que protegería la salud de los trabajadores, dándoles mejores condiciones laborales y aumentando la satisfacción de los clientes con el servicio de entrega.

La industria 4.0, también conocida como "fábrica inteligente", le apuesta a un modelo de control con la mayor automatización posible, enfocándose en las tecnologías de la información, es decir, se busca un ciclo de producción continuo con un registro de datos a lo largo de todo el proceso [7]. La robótica es una herramienta indispensable para el desarrollo de esta propuesta, debido a que con ella es posible integrar diferentes robots autónomos en distintos momentos dentro de un proceso, como los AGB (Automated Guided Vehicle) [8]–[10].

Una problemática observada dentro de la industria es que el transporte de materiales interno es realizado por personal, produciendo un costo significativo [11]. Un robot móvil autónomo no solo reduciría gastos, sino que evitaría cuellos de botella en la distribución interna.

El desarrollo de robots móviles autónomos es una herramienta innovadora que ha avanzado con los años desde diferentes enfoques, tanto para uso industrial como cotidiano. La integración de esta tecnología, acompañada de sistemas de control, ha permitido la optimización de diferentes procesos industriales, debido a su precisión y adaptación para la ayuda o ejecución total de tareas eficientemente.

En este proyecto se propone la elaboración de dos prototipos de robot móvil diferencial autónomos, con capacidad de desplazamiento alrededor de un área delimitada de trabajo, transportando paquetes de una carga máxima de 3 kg, que serán introducidos en ellos de manera automatizada.

El presente trabajo se organiza como sigue. En la metodología se discute cómo se eligió y diseñó cada parte de los prototipos y el código. En la sección de resultados y discusión se plantean los datos obtenidos con las pruebas realizadas. Finalmente, se exponen las conclusiones, perspectivas y recomendaciones, explicando que el sistema puede representar beneficios con respecto a procesos manuales y hablando de sus posibles áreas de mejora.

Metodología

En primer lugar, se determinaron las necesidades que el transporte de paquetes dentro de una planta de distribución

debe cubrir. Con base en su entorno y los tipos de paquetes que transportará, considerando las dimensiones y pesos de las cajas comerciales más utilizadas dentro de la industria [12], se establecieron las especificaciones del robot y se delimitaron las características que debía tener la banda, de acuerdo con los sistemas que se suelen utilizar dentro de este tipo de industria para poder apagarse más a la realidad.

Se determinó hacer el sistema de control a través de una cámara colocada en la parte superior del área de trabajo (a diferencia de la mayoría de los sistemas existentes, cuyas cámaras se encuentran en los robots [13]–[15]), debido a que ésta permitiría aprovechar la infraestructura presente en la mayoría de las plantas. Dicha cámara, mediante un código Arduino, tiene la función identificar los QR ubicados en la parte superior de los robots, para conocer su posición en todo momento y evitar que éstos choquen entre sí; además, leerá los QR ubicados en cada paquete, para conocer su destino de almacenamiento antes de ser introducidos en cualquiera de los robots.

Para la carga de los paquetes, de manera automatizada se fabricará una banda transportadora de rodillos que, con ayuda de sensores, activará el motor cuando se le envíe una señal, indicando que ya hay un paquete sobre ella.

Para poder integrar un AGB de manera exitosa, es necesario adaptar el diseño de acuerdo con las tareas que realizará y el entorno en el que trabajará, sin dejar de lado las características que lo definen. Con el fin de identificar estas últimas y alcanzar la adaptación con su entorno, se revisaron diferentes patentes y proyectos con elementos que aportaran información para el desarrollo de nuestra propuesta [9], [12]–[20]. Además, hay que considerar las necesidades de las empresas que se dedican a la distribución de paquetes para poder realizar el diseño, tanto del robot como de la banda transportadora.

Una vez establecidas las especificaciones de los diseños, se procedió a su fabricación, seleccionando los materiales que otorgarían una estructura firme, para después programar la cámara y la banda transportadora. Con el prototipo fabricado, se realizaron pruebas, tanto de simulación como físicas, para verificar y evaluar su funcionamiento, comprobando que cumpliera con el objetivo de la reducción de los tiempos de distribución de los paquetes, disminuyendo los errores de distribución y mejorando las condiciones laborales de los trabajadores encargados de estas tareas.

La selección de la banda se realizó con base en la seguridad de los paquetes, buscando evitar daños sobre el robot. Para ello, se revisaron diferentes métodos y sistemas de transporte, concluyendo que una banda de rodillos se adaptaría mejor a las necesidades del proyecto, debido a la seguridad que presenta por ser programada con una velocidad constante, ideal para las diferentes dimensiones y pesos de las cajas que se establecieron para el proyecto [21]–[24].

Entonces, se procedió con el diseño del robot, para lo que se creó cada pieza en el software CATIA, donde posteriormente se ensambló el chasis antes de su fabricación. De esta manera, se evitó el desperdicio de material, asegurando que las piezas tuvieran el tamaño ideal para encajar unas con otras, fig. 1.

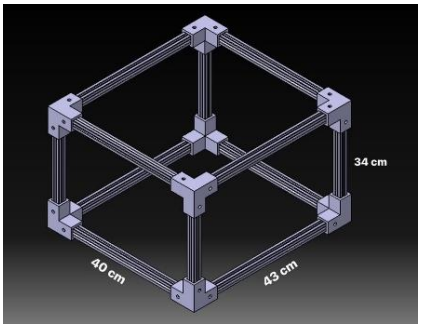

Fig. 1. Chasis del robot.

Para la estructura del robot se decidió usar un perfil de 20mm, ya que su forma facilita trabajar con él e implementarlo con otras piezas [25]. Para la unión de los perfiles se realizaron impresiones 3D de esquineros, lo que permitió darle más soporte a la estructura, al estar hechos a la medida. De igual manera, se imprimieron tuercas T para ser introducidas en el perfil y poder atornillar cada lado de los esquineros. Tanto las tuercas como los esquineros fueron diseñados en CATIA y después introducidos al software Cura, que convierte el diseño a un código G, el cual finalmente se introdujo en la impresora 3D, para obtener las piezas que se muestran en las figuras 2 y 3.

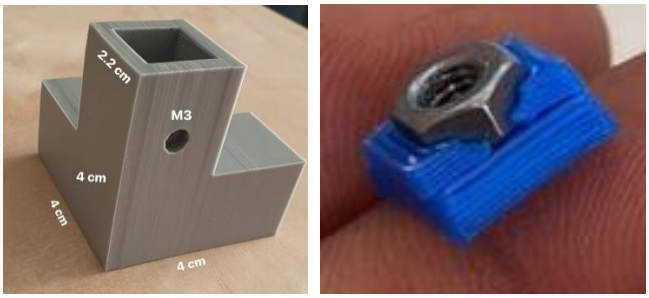

Fig. 2 y 3. Impresiones 3D esquineros y tuercas T.

Una vez obtenidas estas últimas, se cortaron los perfiles con las medidas establecidas y se fijaron los esquineros con los perfiles, usando las tuercas T.

Para el movimiento del robot, se seleccionaron dos motores JGA25-371, ya que trabajan a 12V y pueden cargar alrededor de 6kg, manteniendo una velocidad constante dentro de un rango de 0 a 140 rpm. Éstos se unieron a neumáticos de goma para el movimiento del robot, y se implementaron 4 ruedas locas en cada eje para un desplazamiento más fluido, como se puede observar en la figura 4.

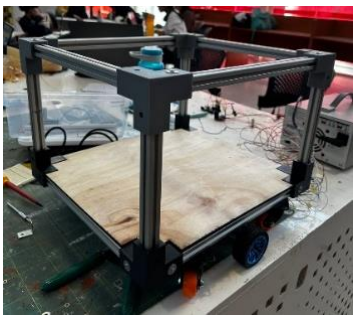

Fig. 4. Estructura del robot construido.

Para integrar el circuito electrónico del robot, se diseñó una estructura superior en forma de caja, debido a que la batería utilizada es grande y a que, por seguridad, el circuito debe estar cubierto, pero accesible en caso de alguna emergencia (fig. 5).

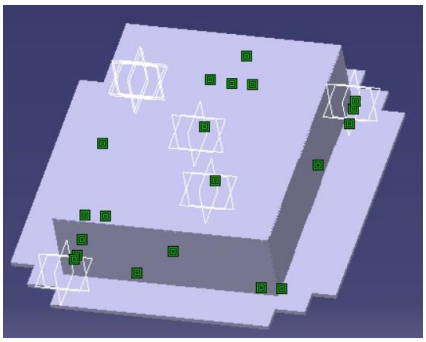

Fig. 5. Estructura superior del robot.

Para verificar que la estructura del robot pudiera soportar su peso total (tomando en cuenta el circuito electrónico y el peso del paquete), con ayuda de CATIA se realizaron pruebas de elemento finito, que permitieron observar el comportamiento de los materiales ante cargas simuladas para determinar el punto de quiebre, conociendo su capacidad de carga máxima y el punto en el que la estructura puede llegar a quebrarse [26] (fig. 6).

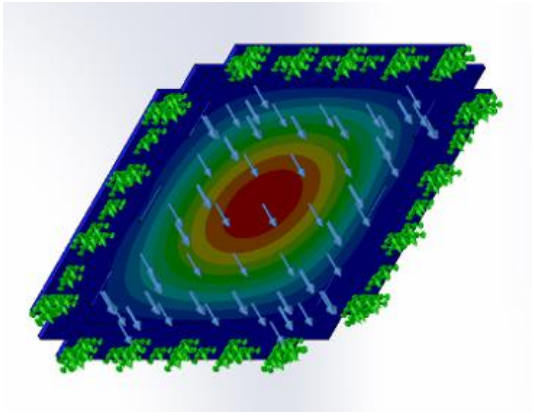

Fig. 6. Análisis de elemento finito.

Con el análisis obtenido, se decidió que no era necesario usar aluminio para la estructura, debido a que su resistencia es muy superior a la requerida; entonces se optó por madera para la base superior del robot donde caería el peso del circuito. Con la seguridad de que la estructura soportaría el peso deseado con un 50% margen de seguridad (1.5 kg más), se procedió a su elaboración completa. El mismo proceso se repitió para la construcción del segundo robot (figura 7).

Fig. 7. Robots terminados.

A continuación, se diseñó en CATIA la banda de rodillos, que transportaría los paquetes hacia el robot de manera automatizada (fig. 8).

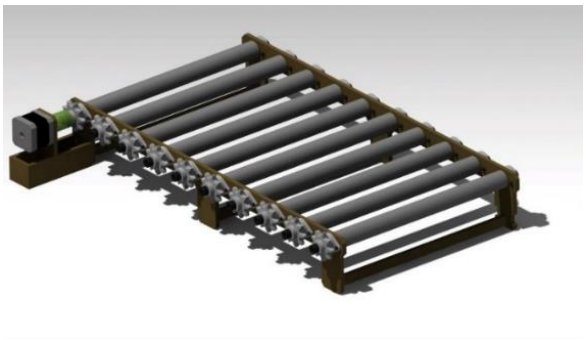

Fig. 8. Ensamble de banda transportadora.

Para fabricar la banda, primero se realizaron los cortes de la estructura en madera de 12mm de ancho, que se muestran en la figura 9.

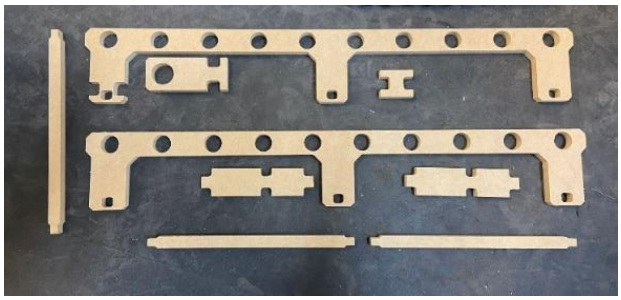

Fig.9. Cortes de la estructura de la banda.

Posteriormente se trabajó en los rodillos; para su construcción se usó un redondo de aluminio de una pulgada de grosor y de 2.4m de largo. Para obtener los 10 redondos que se utilizaron en la banda, se seccionó la pieza con una sierra de disco en pedazos de 23cm cada uno. Una vez obtenidos, se introdujeron en el torno para realizarles 2 procesos: el primero fue un desbaste de 1 cm con un inserto de trenzado, para obtener redondos de 22 cm; el segundo, fue un careado de 1mm en ambas caras, dejando los redondos en 21.8cm (fig. 10).

Fig. 10. Desbaste de redondo.

A continuación, se realizaron marcas de 5mm de profundidad en ambas caras con ayuda de la broca de centro, para posteriormente hacer una perforación de 30mm de profundidad, con ayuda de una broca de 7mm (fig. 11).

Fig. 11. Perforación del centro con broca de 7mm.

Una vez obtenida la profundidad deseada, con un machuelo M8x1.25 se procedió a darle cuerda a los 30mm de cada perforación (fig. 12).

Fig.12. Machuelado de los rodillos.

Desarrollada la parte mecánica, cuya imagen se muestra en la figura 13, se procedió con la programación de control del robot y sus conexiones electrónicas.

Fig.13. Banda de rodillos terminada.

El reconocimiento de los códigos QR se programó en Python, siguiendo el diagrama de flujo mostrado en la fig. 14. Sus detalles se encuentran en la primera sección del Anexo I.

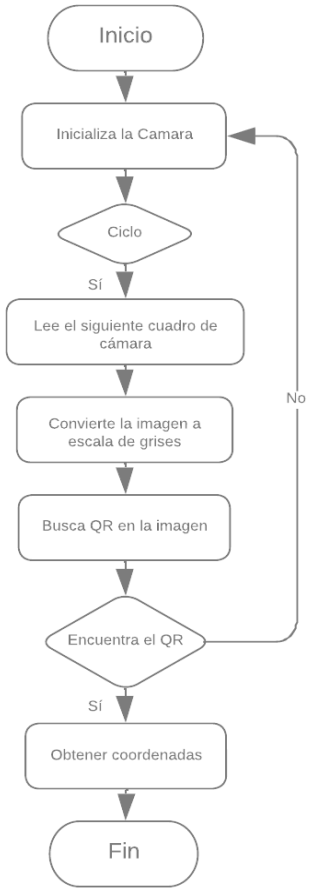

Fig.14. Diagrama de flujo del programa de control.

Para la recepción de datos se creó un programa con el mismo lenguaje, consistente en un sistema de seguimiento de códigos QR mediante una cámara conectada al equipo, que se ejecuta a través de una interfaz gráfica de usuario (GUI), fig. 15, la cual se creó con la biblioteca Tkinter y cuyas particularidades se presentan en la primera sección del mismo anexo.

Fig. 15. Interfaz GUI.

Para el movimiento de las 2 ruedas se realizó un código Arduino, cuyas especificaciones están en la Sección II del Anexo I.
Una vez integradas todas las partes del proyecto, se realizaron pruebas de seguimiento de trayectoria y de peso, con cajas de diferentes dimensiones, para poder verificar el correcto funcionamiento del robot.

Resultados y Discusión

Para la caracterización del robot se determinó que los paquetes más recurrentes en la industria de nuestro interés tienen las siguientes dimensiones: 15-20cm de ancho, 15-40cm de largo y 15-25cm de alto (con un peso promedio de 0.5–3kg) [12]; por ello se determinó que las del robot fueran 43cm de largo, 40cm de ancho y 34cm de alto, de manera que todas las cajas dentro del rango pudieran ser introducidas en él. Además, se diseñó el robot en forma de caja abierta, para simplificar el proceso de carga.
La selección del material para el soporte del robot y los esquineros que unen el chasis se hicieron con base en pruebas de simulación de elemento finito.
Como resultado de ellas, se encontró que el punto de estrés máximo de la base, hecha en madera de pino de 5mm de grosor, al aplicarle una fuerza vertical hacia debajo de 100N, fue de 2.533 N/mm^2, como se puede observar en la figura 16.

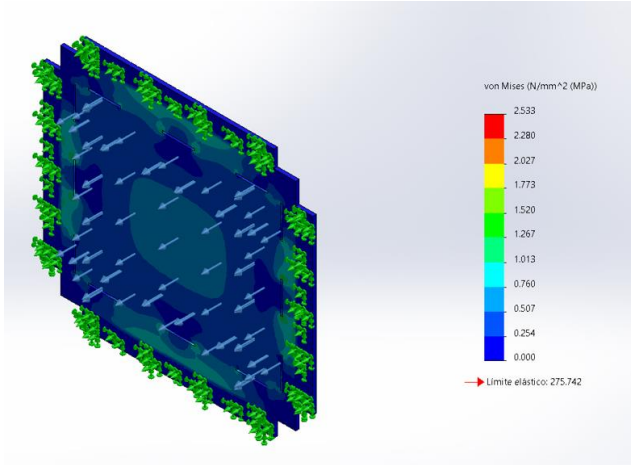

Fig.16. Punto de estrés máximo base del robot.

Con la misma simulación, se calculó el punto de desplazamiento máximo, que se encontró en el centro de la tabla, con una magnitud de 0.171mm, figura 17.

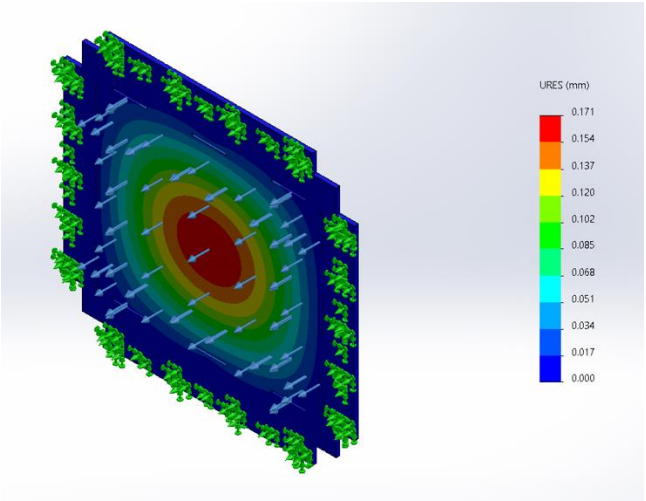

Fig.17. Punto de desplazamiento máximo base del robot aluminio.

Además, se calculó el factor de seguridad (FDS) con un peso de 10kg que, como se puede observar en la figura 18, resultó tener una magnitud de 2.002, lo que significa que resiste el doble del peso.

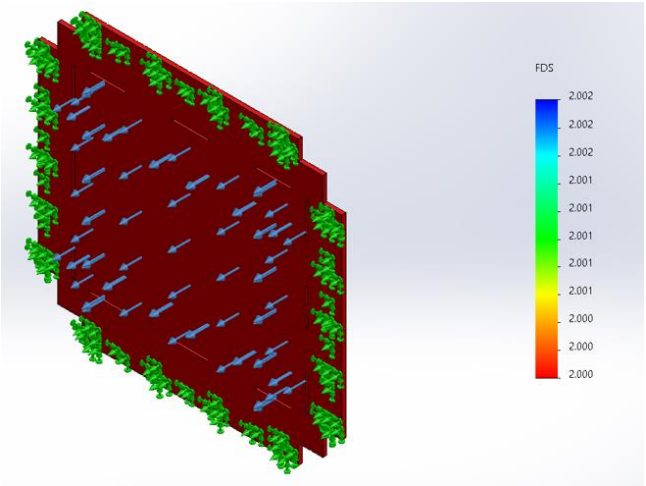

Fig.18. Factor de seguridad de la base del robot.

Con base en estos datos, se decidió utilizar ese material, por cumplir con la resistencia requerida y tener un costo menor al de cualquier metal.
Para evaluar la resistencia de los esquineros de unión del chasis, se realizaron también pruebas de elemento finito con PLA, que es el tipo de filamento utilizado en las impresiones 3D. Como se puede observar en la figura 20, el punto de estrés máximo de este material, con una carga de 100N (10.1972Kgf), se ubicó en la parte interna del esquinero.

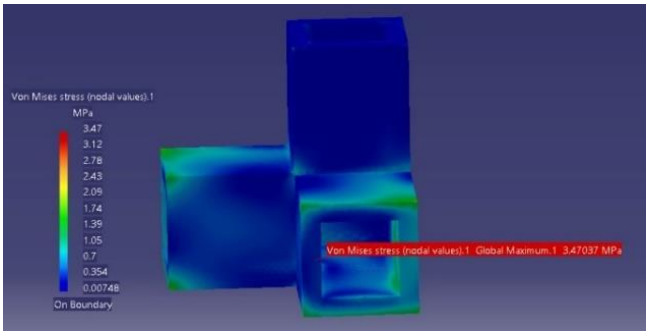

Fig.19. Punto de estrés máximo del esquinero.

Para poder visualizarlo más claramente, se realizó un corte vertical a la pieza, encontrando el punto máximo de estrés a en la esquina de unión de los perfiles que se introducen en el esquinero. Éste tiene una magnitud de 3.47MPa, que es mucho menor que el Módulo de Young de este material ($3.2x10^9\ N/m^2$) [27], figura 21.

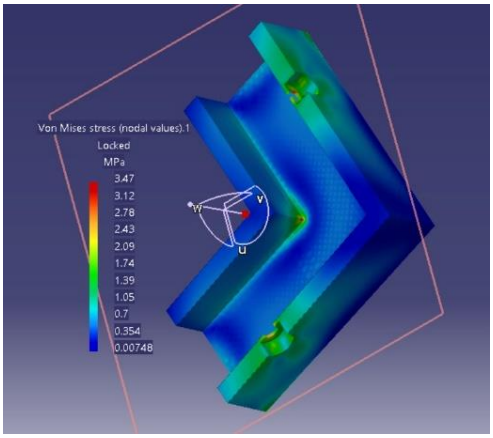

Fig.20. Punto de estrés máximo del esquinero.

Después de las pruebas de estrés, se realizó un análisis para conocer el desplazamiento sufrido por una carga de la misma magnitud. En la figura 22, podemos observar que el punto de desplazamiento máximo se encontró en la parte superior del esquinero, donde caía el peso de la estructura; este valor de desplazamiento es de 0.0588mm, y resulta despreciable de acuerdo con la literatura [27].

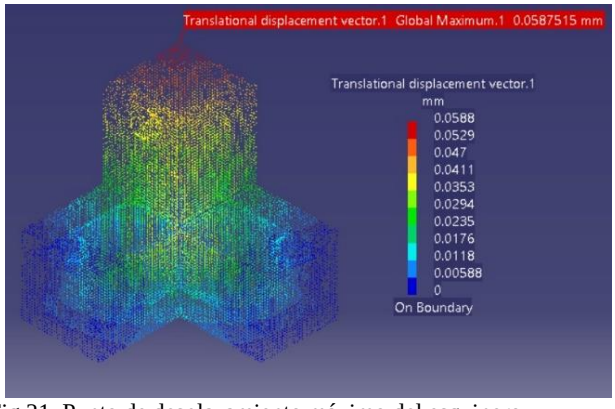

Fig.21. Punto de desplazamiento máximo del esquinero.

Con estos datos se decidió realizar la impresión del esquinero en 3D, por presentar suficiente precisión y seguridad, además de ser económico.
Una vez seleccionado el material y construido el robot, se procedió a hacer pruebas físicas con cajas de diferentes dimensiones y pesos. Es importante tener en cuenta que, además del de las cajas, la estructura y circuito del robot tienen un peso de 1kg aproximadamente, por lo que para calcular el peso total que carga del robot se debe sumar un kilo al peso del paquete. En la tabla 1 se presentan los datos de las cajas utilizadas para las pruebas, así como el peso total en cada caso y un afirmativo o negativo, para revelar si la estructura resistió o no.

Tabla 1: Pruebas de peso del robot.

Número de prueba	Dimensiones de la caja (cm)	Peso de la caja (kg)	Peso Total (kg)	Validación de resistencia
1	15x15x15	1	2	Sí
2	15x15x15	1.5	2.5	Sí
3	15x15x15	2	3	Sí
4	18x23x18.6	2.5	3.5	Sí
5	18x23x18.6	3	4	Sí
6	20x30x12.6	3.5	4.5	Sí
7	20x30x12.6	4	5	Sí
8	20x30x12.6	4.5	5.5	Sí
9	20x40x19	5	6	Sí
10	20x40x19	5.5	6.5	Sí
11	20x40x19	6	7	Sí

En las imágenes 22 y 23 se muestran imágenes de los robots cargados.

Fig.22 y 23. Prueba 3 y 4 de peso.

Con dichas pruebas, se concluyó que la estructura resistía más que el peso propuesto (que era de 3 kg) y podía cargar el doble de éste (al soportar un rebase de 3 kg).
Para las pruebas con cámara, se seleccionó la C920 PRO-HD WEBCAM, que tiene un sistema similar al de las cámaras de seguridad Dahua, muy comunes dentro de la industria.
Para iniciar con las pruebas de identificación de los códigos QR, se clasificaron los cuadros de reconocimiento por color de acuerdo con el objeto registrado de la siguiente manera: el robot-1, rojo; el robot-2, azul; la zona de carga de paquetes, color verde y los códigos para los paquetes, morado. Se imprimieron los QR que irían sobre los paquetes con una dimensión de 10x10 cm^2; en la figura 25 podemos observar que éstos no fueron reconocidos por la cámara, por lo que se hicieron ajustes al tamaño de la impresión, ya que al elevar dichos códigos sobre una mesa, sí eran detectados, pero al nivel del piso, donde estaría la banda, no.

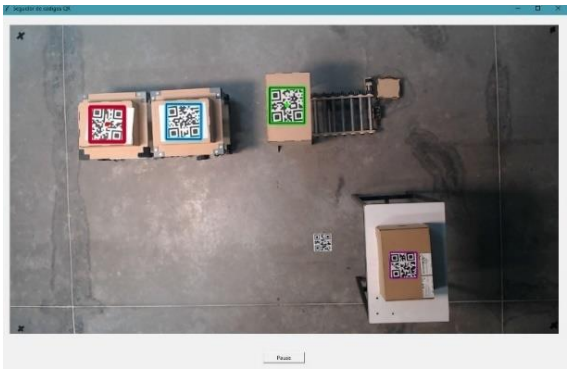

Fig.24. Primera prueba de reconocimiento QR. El código que se encontraba en el piso no fue detectado por la cámara.

Para solucionar este problema se podría haber bajado la cámara, pero esto hubiera reducido el área de trabajo, por lo que se optó por imprimir los códigos QR en un tamaño 80% mayor que el original, figura 33.

Fig.25. Segunda prueba de reconocimiento QR. En este caso, sí fue detectado el código cuyo tamaño fue mayor.

Cuando todos los códigos fueron detectados, se realizaron pruebas de desplazamiento para verificar el funcionamiento de los motores, las ruedas, el circuito electrónico, y la comunicación entre la cámara y el robot a través de ellos.
En estas pruebas se definió una trayectoria que el robot debía seguir; en la figura 27 se puede observar en color azul la trayectoria trazada para el robot y en color rojo la trayectoria que siguió.

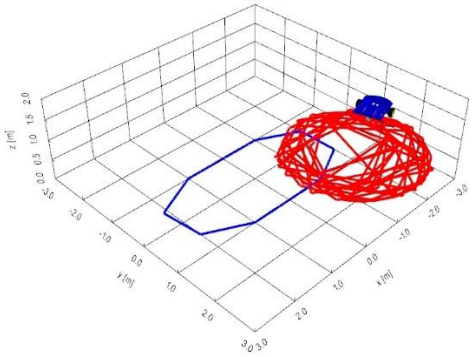

Fig.26. Prueba de seguimiento 1.

Se puede ver que el robot no siguió la trayectoria trazada y se mantuvo dando vueltas sobre el mismo punto de referencia. En la figura 28 se observa de manera gráfica la diferencia entre las velocidades deseadas y la velocidad real del robot. La gráfica superior hace referencia a la velocidad lineal y la inferior, a la velocidad angular.

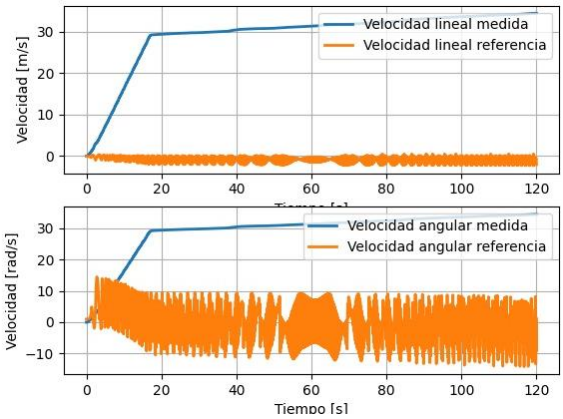

Fig.27. Gráficas de velocidad de seguimiento 1.

Después de una profunda revisión del código, se descubrió que el problema se encontraba en que éste no almacenaba los movimientos del robot, por lo que repetía el mismo movimiento y no pasaba al siguiente. Una vez que se cambió el código, se realizó la misma prueba; en la figura 29 se observa que en ese caso el robot sí siguió la trayectoria trazada.

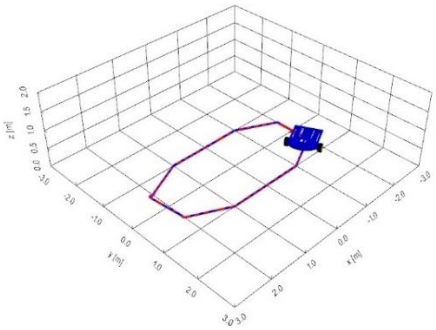

Fig.28. Prueba de seguimiento 2.

En la figura 30 se muestran las gráficas con las velocidades marcadas y las velocidades reales de esta segunda prueba.

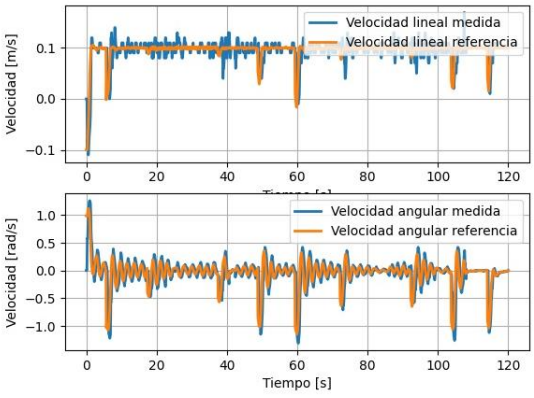

Fig.29. Gráficas de velocidad de seguimiento 2.

Se nota una pequeña variación en la velocidad lineal, debida a que el piso era irregular; sin embargo, como se puede apreciar en la figura 28, esto no afectó en el seguimiento de la trayectoria.

Por último, se realizaron las pruebas de carga de paquetes con diferentes dimensiones y pesos, para poder determinar la velocidad ideal de la banda.

El movimiento de esta última es generado por un motor de 12V que se activa a través de un sensor ultrasónico, y manda una señal cuando el robot se encuentra en la posición de carga. La alimentación del motor se manipula a través de un controlador conectado a una fuente de voltaje, que permite regular el torque y el paso.

Con el objetivo de conocer la configuración ideal para el desplazamiento de los paquetes, se realizaron varias pruebas. La primera, se hizo con una configuración de microstep en 8 y 1600 pulsos sobre revoluciones, y una corriente de 1A; con esta prueba, se observó que los paquetes iban a una velocidad baja, que evitaba que entraran por completo dentro del robot. La segunda prueba se realizó con una configuración de microstep en 1 y 200 pulsos sobre revolución, con una corriente de 3A; con esta configuración conseguimos que los paquetes entraran por completo en la banda, pero con un movimiento inestable sobre ella. La tercera prueba se realizó con parámetros intermedios entre las dos últimas: microstep en 2/A, con 400 pulsos sobre revolución y una corriente de 2A; con esta configuración, los paquetes cayeron de manera segura dentro del robot, por lo que fue la elegida para el transporte de las muestras descritas en la Tabla 1.

Por último, se evaluaron los registros de coordenadas de los paquetes a través de los códigos QR, para lo que se utilizó un programa Arduino (Anexo I, sección II), que permitió guardar las coordenadas en un archivo txt.

Se trazaron diferentes rutas de desplazamiento, para que el robot llevara distintos paquetes a diversos puntos, y se verificó que las coordenadas registradas en el archivo coincidieran con la trayectoria trazada (fig. 30).

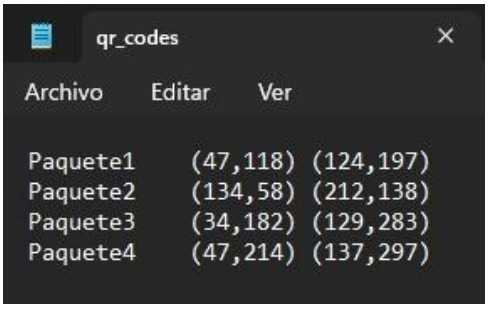

Fig.30. Archivo txt con coordenadas.

Al observar los resultados del archivo, pudimos confirmar que las coordenadas coincidían con las establecidas; de esta manera, se reduciría el porcentaje de pérdida de los paquetes dentro de la planta, al tener un registro de la última ubicación de cada caja.

Conclusiones, perspectivas y recomendaciones

Se partió del supuesto de que un robot autónomo diferencial mejoraría el transporte interno de una planta distribuidora de paquetes, debido a su fácil implementación, eficiencia y la reducción de costos que implicaría su uso. Además, mejoraría las condiciones de trabajo de los encargados de esta tarea, al reducir los riesgos de sufrir fatiga o, incluso, una lesión física, y permitiría tener un mejor control de la ubicación de los paquetes en todo momento, disminuyendo su extravío dentro de la planta.

A partir de ello, propuso un prototipo de robot móvil diferencial autónomo, con capacidad de desplazamiento y transporte de paquetes de un máximo de 3 kg, mediante la integración de una cámara de control de posicionamiento y una estación de carga automatizada. Se establecieron las dimensiones del robot en 43cm de largo, 40cm de ancho y 34cm de alto, de manera que las cajas comúnmente utilizadas en esta industria cupieran en él, y de caja abierta, por cuestiones de seguridad.

Posteriormente, se diseñó en CATIA, donde se llevaron a cabo pruebas de elemento finito con un peso de 10 kg, para determinar la resistencia de los materiales y los puntos de estrés y desplazamiento máximo, dando como resultado un FDS de 2 con madera de pino de 5mm. También se testó un esquinero impreso en 3D, obteniendo un punto de estrés máximo de 3.47MPa (menor a su número de Young).

Con base en los resultados de estas pruebas, se procedió con la fabricación del prototipo y la programación de los códigos QR que guiarían sus trayectorias. Se hicieron pruebas sobre la lectura de dichos códigos, y se determinó que éstos deben ser por lo menos de $18x18cm^2$, para garantizar su visualización. Además, se realizaron pruebas de peso físicas donde se comprobó el FDS obtenido en las pruebas de elemento finito, debido a que el robot soporta 6kg (el doble del peso planteado inicialmente).

De igual manera se realizaron pruebas de vacío de datos de las primeras y ultimas coordenadas de los paquetes, dando una precisión del 100% ya que en todas las pruebas los datos y las coordenadas coincidieron.

En futuros trabajos, se sugiere adaptar las dimensiones y velocidad del robot a las cajas utilizadas, el área de trabajo y el sistema de banda que ya tengan establecido en cada empresa. También sería conveniente desarrollar una interfaz para el vaciado de datos de la ubicación de las cajas, que se adaptara a las necesidades de la empresa. Es importante tener en consideración la iluminación del área de trabajo donde se implemente el robot, para garantizar una correcta lectura de los códigos QR.

Referencias

[1] M. Castells, “Internet y la sociedad RED,” 2022.

[2] A. F. Palomino Pita, M. V. Carolina, and J. F. Oblitas Cruz, “E-commerce and its importance in times of covid-19 in Northern Peru,” *Revista Venezolana de Gerencia*, vol. 25, no. 3, pp. 253–266, 2020, doi: 10.37960/rvg.v25i3.33367.

[3] D. R. Durán, G. Municio, and Á. Manuel, “Estudio de la compañía DHL,” 2019.

[4] K. Garcés Tabares, “Trastornos musculoesqueléticos (TME) por manipulación de cargas,” 2019.

[5] L. V. Revelo Torres, “Inadecuados métodos de manipulación de carga y descarga de mercancías en el área logística de la empresa Bidtrans Cía. Ltda. de la ciudad de Quito Provincia de Pichicha.,” 2020.

[6] Y. W. Chan, T. H. Huang, Y. T. Tsan, W. C. Chan, C. H. Chang, and Y. Te Tsai, “The Risk Classification of Ergonomic Musculoskeletal Disorders in Work-related Repetitive Manual Handling Operations with Deep Learning Approaches,” in *Proceedings - 2020 International Conference on Pervasive Artificial Intelligence, ICPAI 2020*, Institute of Electrical and Electronics Engineers Inc., Dec. 2020, pp. 268–271. doi: 10.1109/ICPAI51961.2020.00057.

[7] J. Luis and D. V. Román, “Industria 4.0: la transformación digital de la industria,” Bilbo, España, 2021.

[8] J. F. Nethery and M. W. Spong, “Robotica: A Mathematica Package for Robot Analysis,” *IEEE Robot Autom Mag*, vol. 1, no. 1, pp. 13–20, 1994, doi: 10.1109/100.296449.

[9] T. Ribeiro, I. Garcia, D. Pereira, J. Ribeiro, G. Lopes, and A. F. Ribeiro, “Development of a prototype robot for transportation within industrial environments,” in *2017 IEEE International Conference on Autonomous Robot Systems and Competitions, ICARSC 2017*, Institute of Electrical and Electronics Engineers Inc., Jun. 2017, pp. 192–197. doi: 10.1109/ICARSC.2017.7964074.

[10] H. Sasamoto, R. Velazquez, S. Gutierrez, M. Cardona, A. A. Ghavifekr, and P. Visconti, “Modeling and Prototype Implementation of an Automated Guided Vehicle for Smart Factories,” in *Proceedings of the 2021 IEEE International Conference on Machine Learning and Applied Network Technologies, ICMLANT 2021*, Institute of Electrical and Electronics Engineers Inc., 2021. doi: 10.1109/ICMLANT53170.2021.9690543.

[11] H. Giron Nieto, “Vigilancia tecnológica de robots AGB,” Universidad Iberoamericana Puebla, Puebla, 2022.

[12] U. Abelleyra Martínez and G. B. Jiménez Martínez, “Diseño y construcción de un sistema mecatrónico para el doblado y pegado de cajas de cartón de diferentes dimensiones,” 2012.

[13] Y. Wu and B. Yao, “Logistic Vehicule ,” US D888,791 S, Jun. 30, 2020

[14] T. Scott, B. Ilhan, V. Sood, and C. Andersen, “Collaborative Autonomous Ground,” US 2022/0024486A1, Jan. 27, 2022

[15] C. Guillermo, M. Machado, I. Oristela Benítez González, A. Manuel, R. Rivera, and V. Moreno Vega, “Implementación de sistema operativo robótico en una plataforma de robot móvil,” 2020.

[16] A. E. Afrouzi, S. Mehrnia, and L. Robinson, “Obstacle Recognition Method for Autonomous Robots,” US 11,449,063B1, Sep. 20, 2022

[17] T. Moore and B. Powers, “Enfoque dinámico de ventana usando un punto crítico de pérdida óptimo de evitación de colisiones recíprocas,” US 201715712256, 2022

[18] J. E. Caicedo Martínes and B. Bacca Cortes, *Autonavi3at Software Interface to Autonomously Navigate on Urban Roads Using Omnidirectional Vision and a Mobile Robot*, 1st ed., vol. 32. Ciencia e Ingeniería Neogranadina, 2022.

[19] E. Hernández Herrera and O. Lira Hernández, “Prototipo de robot de servicio doméstico conectado a internet,” 2018. [Online]. Available: https://www.globalsign.com/es/blog/desafios-y-

[20] G. Suarez-Rivera, N. D. Muñoz-Ceballos, and H. M. Vásquez-Carvajal, “Development of an Adaptive Trajectory Tracking Control of Wheeled Mobile Robot,” *Revista Facultad de Ingeniería*, vol. 30, no. 55, p. e12022, Feb. 2021, doi: 10.19053/01211129.v30.n55.2021.12022.

[21] F. Ernesto, A. Maldonado, A. Por, I. Milton, and A. Fuentes Orozco, “TRANSPORTADORA DE RODILLOS PARA PIEDRÍN,” Feb. 2016.

[22] A. Vázquez Hernández, I. Richard Sosa López Instructor, and D. Eduardo Izaguirre Castellanos Auxiliar, “Actuador lineal TRITEX TLM30 en aplicaciones de control.,” 2013.

[23] D. Mario and S. Gervaso, “DISEÑO DE UNA BANDA TRANSPORTADORA MEDIANTE GUIDE DE MATLAB,” Madrid, Sep. 2013.

[24] M. F. Caldas Ochoa, “DISEÑO MECÁNICO DE UN TRANSPORTADOR POR BANDA SOBRE RODILLOS PARA APILAMIENTO DE CALIZA Y ARCILLA,” Santiago de Cali, 2013.

[25] item Industrietechnik GmbH, “Perfil 5 20x20, natural,” 2023.

[26] M. Cañones Castellano, “Estudio comparativo entre el módulo de Elementos Finitos de CATIA V5 y ANSYS Workbench,” Sevilla, 2019.

[27] R. L. Mott, *Resistencia de los Materiales*, 5th ed., vol. I. México: Pearson Educación, 2009.

ANEXO I

I. Código Python lecturas QR

Para el desarrollo de este código se importaron las librerías *Pillow* y *Open CV*, que ayudaron con el procesamiento de imágenes, *PyArduino* para la comunicación con la placa Arduino y *Pyzard* para la decodificación de los códigos QR. La primera función *toggle* modifica el botón inicio al botón de pausa, dependiendo de si el robot esta en movimiento o no. La siguiente función *onClosing* cierra la comunicación con la placa Arduino y las cámaras cuando la interfaz GUI se cierra. La función *qrDetection* detecta los códigos QR en la imagen capturada por la cámara y la devuelve con los códigos marcados, así como las coordenadas de los códigos detectados (fig.).

```python
from tkinter import *
from PIL import Image, ImageTk
import numpy as np
from pyArduino import *
from pyzbar import pyzbar

import cv2
import sys


def toggle():
    btn.config(text=btnVar.get())


def onClosing():
    arduino.sendData([0,0])
    arduino.close()
    cap.release()
    print("Cámara desconectada")
    root.destroy()


def qrDetection(frame):

    cx = 0
    cy = 0
    cxd = 0
    cyd = 0

    isRobot = False
    isObject = False

    barcodes = pyzbar.decode(frame)

    for barcode in barcodes:
        (x, y, w, h) = barcode.rect
        barcodeData = barcode.data.decode("utf-8")

        if(barcodeData=="PIER1"):
            isRobot = True
            cx = x + (w)//2
            cy = y + (h)//2
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 0, 255), 2)

        if(barcodeData=="META"):
            isObject = True
            cxd = x + (w)//2
            cyd = y + (h)//2
            cv2.rectangle(frame, (x, y), (x + w, y + h), (0, 255, 0), 2)
```

Fig. 31. Primera parte código Python.

Posteriormente la función *callback* se encarga de ejecutar un bucle que realiza el seguimiento del robot: primero, lee la imagen de la cámara y la procesa para detectar los códigos QR, después realiza el control del robot mediante un algoritmo que, utilizando la información de la posición del destino y del robot, calcula la velocidad necesaria de las ruedas para que el robot siga el objeto.

```python
def callback():
    ret, frame = cap.read()

    if ret:

        frame,cxd,cyd,cx,cy,isRobot,isObject = qrDetection(frame)
        cv2.circle(frame,(cxd,cyd), 5,(0,255,0),-1)
        cv2.circle(frame,(cx,cy), 5,(0,0,255),-1)

        if(isRobot and isObject):

            ###################### Conversion coordenadas ###########
            hxd = cxd-frame.shape[1]/2
            hyd = frame.shape[0]/2 - cyd
            hx = cx - frame.shape[1]/2
            hy = frame.shape[0]/2 - cy

            ###################### Algoritmo de control ##############
            a = 0.215
            phi.set(arduino.rawData[6])

            # Errores
            hxe = hxd - hx
            hye = hyd - hy
            he = np.array([[hxe],[hye]])
            K = np.diag([0.001,0.001])
            J = np.array([[-np.sin(phi.get()),-a*np.cos(phi.get())],
                          [np.cos(phi.get()),-a*np.sin(phi.get())]])

            # Ley de control

            qpRef = np.linalg.inv(J)@(K@he)
            uRef.set(round(qpRef[0][0],3))
            wRef.set(round(qpRef[1][0],3))

        else:
            uRef.set(0)
            wRef.set(0)

        if btnVar.get() == 'Start':
            arduino.sendData([uRef.get(),wRef.get()])
```

Fig. 32. Segunda parte código Python.

```python
            else:
                arduino.sendData([0,0])

            img = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            img = Image.fromarray(img)
            img.thumbnail((1240,720))
            tkimage = ImageTk.PhotoImage(img)
            label1.configure(image = tkimage)
            label1.image = tkimage

    else:
        onClosing()

    root.after(100,callback)
```

Fig. 33. Tercera parte código Python.

Finalmente, se crea la interfaz gráfica de usuario y se ejecuta la función *caljback* en un bucle infinito para realizar el seguimiento en tiempo real.

```python
cap = cv2.VideoCapture(1)
cap.set(cv2.CAP_PROP_FRAME_WIDTH, 1240)
cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 720)

if cap.isOpened():
    print("Cámara inicializada")
else:
    sys.exit("Cámara desconectada")

########## Arduino ########

port = 'COM8'
arduino = serialArduino(port,baud=9600,sizeData=7)
arduino.readSerialStart()

######## HMI ########

root = Tk()
root.protocol("WM_DELETE_WINDOW",onClosing)
root.title("Seguidor de codigos QR")


uRef = DoubleVar(root,0)
wRef = DoubleVar(root,0)
phi = DoubleVar(root,0)

label1 = Label(root)
label1.grid(row=0,padx=20,pady=20)


btnVar = StringVar(root,'Pause')
btn = Checkbutton(root, text=btnVar.get(), width=12, variable=btnVar,
                  offvalue='Pause', onvalue='Start', indicator=False,
                  command=toggle)

btn.grid(row=1, padx = 20, pady = 20)


root.after(100,callback)
root.mainloop()
```

Fig. 34. Cuarta parte código Python.

II. Código Arduino movimiento del robot.

Para el movimiento de las 2 ruedas se realizó un código Arduino, el cual utiliza dos librerías; la primera *PinChangeInterrupt* permite manipular las interrupciones de los *encoders* para medir la velocidad angular de las ruedas; la segunda, *motor Control*, permite implementar y manipular un controlador PID para ajustar la velocidad y dirección para el movimiento del robot.
El programa comienza con la declaración de los pines para el manejo de los *encoders* y de los motores. Una vez declaradas las variables se declara la primera función *setup*, donde se establece la velocidad de comunicación serie y se configuran los parámetros de sintonía del controlador PID, fig___.

```arduino
#include "PinChangeInterrupt.h"
#include "motorControl.h"

//////////////////////////// CONTROLADOR PID ///////////////////
unsigned long lastTime, sampleTime = 100;

motorControl motorR(sampleTime);
motorControl motorL(sampleTime);

//////////////////// COMUNICACION SERIAL ////////////////
String inputString = "";
bool stringComplete = false;
const char separator = ',';
const int dataLength = 2;
double data[dataLength];

//////////////////////MOTOR DERECHO//////////////////////////
//// Ojo se ha invertido canales////////
const int    C1R = 2;    // Entrada de la señal A del encoder.
const int    C2R = 3;    // Entrada de la señal B del encoder.
int cvR = 0;

//// Puente H L298N ////
const int    in1 = 8;
const int    in2 = 7;
const int    ena = 6;

volatile int nR = 0;
volatile int antR      = 0;
volatile int actR      = 0;

double w1Ref = 0;
double wR = 0;
```

Fig.35. Declaración de librerías y variables.

```
void setup()
{
  Serial.begin(9600);

  ////////////////// SINTONIA FINA PID //////////////////

  motorR.setGains(0.28, 0.05, 0.05); // (Kc,Ti,Td)
  motorL.setGains(0.28, 0.05, 0.05); // (Kc,Ti,Td)

  ////////////////// Limites de señales //////////////////
  motorR.setCvLimits(255,30);
  motorR.setPvLimits(10,0);

  motorL.setCvLimits(255,30);
  motorL.setPvLimits(10,0);

  pinMode(C1R, INPUT);
  pinMode(C2R, INPUT);
  pinMode(C1L, INPUT);
  pinMode(C2L, INPUT);

  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);


  digitalWrite(in1, false);
  digitalWrite(in2, false);
  digitalWrite(in3, false);
  digitalWrite(in4, false);

  attachInterrupt(digitalPinToInterrupt(C1R), encoderR, CHANGE);
  attachInterrupt(digitalPinToInterrupt(C2R), encoderR, CHANGE);

  attachPinChangeInterrupt(digitalPinToPinChangeInterrupt(C1L), encoderL, CHANGE);
  attachPinChangeInterrupt(digitalPinToPinChangeInterrupt(C2L), encoderL, CHANGE);

  lastTime = millis();

}
```

Fig.36. Función setup.

La función *loop* es el corazón del código, ya que es la que se encarga de recibir y procesar los datos serie, mediante una cadena de caracteres que contienen las velocidades del motor derecho e izquierdo. Una vez recibidos estos valores, se llama a las funciones *velocityMotor* y *velocityRobot* (las cuales se encuentran dentro de las librerías antes mencionadas) que son las encargadas de establecer las velocidades de referencia de ambos motores, para después actualizarlas de manera cíclica hasta que el robot llega a su destino o deja de detectar el código QR que lo direcciona a él, fig….

```
void loop()
{
  /SI RECIBE DATOS/
  if (stringComplete)
  {
    for (int i = 0; i < dataLength ; i++)
    {
      int index = inputString.indexOf(separator);
      data[i] = inputString.substring(0, index).toFloat();
      inputString = inputString.substring(index + 1);
     }

     velocityMotor(data[0],data[1]);
     inputString = "";
     stringComplete = false;
  }
```

Fig.37. Función loop.

Para el controlador PID, se mide la velocidad angular de cada rueda por medio de los *encoders*, se calcula el error entre la velocidad de referencia y la velocidad real, y se utiliza el controlador PID para calcular la señal de control que se enviará a los motores. La función *compute* de la biblioteca "*motorControl*" realiza el cálculo del controlador PID y devuelve la señal

de control correspondiente. Finalmente, la dirección y velocidad de los motores se ajustan en función de la señal de control calculada y se actualiza la velocidad y dirección del movimiento del robot.

```
/ CONTROLADOR PID /
if(millis()-lastTime >= sampleTime)
{
  wR = constValue*nR/(millis()-lastTime);
  wL = constValue*nL/(millis()-lastTime);
  lastTime = millis();
  nR = 0;
  nL = 0;

  cvR = motorR.compute(w1Ref,wR);
  cvL = motorL.compute(w2Ref,wL);

  if (cvR > 0) clockwise(in2,in1,ena,cvR); else anticlockwise(in2,in1,ena,abs(cvR));
  if (cvL > 0) anticlockwise(in3,in4,enb,cvL); else clockwise(in3,in4,enb,abs(cvL));

  velocityRobot(wR,wL);

  phi =phi+wRobot*0.1;

  Serial.println(uRobot);
  Serial.println(wRobot);
  //Serial.println(phi);

}
```

Fig.38. Controlador PID.