

Mendoza - Argentina



UNCUYO
UNIVERSIDAD
NACIONAL DE CUYO



FACULTAD
DE INGENIERÍA

ROBOT ASISTENTE DE OPERARIOS DE COSECHA - **RASOC**

PROYECTO FINAL DE ESTUDIOS

Ingeniería en Mecatrónica

Director: Mgtr. Ing. Eduardo Iriarte

Autores:

Cantú Tsallis, Maximiliano - 11294
Lage Tejo, Joaquín - 11495

Agosto de 2024

1. Resumen

En este proyecto se describe el desarrollo y la simulación de la navegación autónoma de un robot móvil asistente de operarios de cosecha, del tipo terrestre y configuración diferencial, con tracción a orugas y punto de control desplazado. Para lograrlo, se implementa un algoritmo de Inteligencia Artificial para la planificación de trayectoria y de movimiento entre 2 puntos pertenecientes al mapa de un terreno o finca real. Se diseña e implementa un Control por Modelo Dinámico Inverso del sistema, combinado con un Controlador PI para las velocidades angulares de cada rueda motriz, y un Controlador P en Lazo Interno, para las corrientes de cada motor, logrando que el robot siga la trayectoria de forma autónoma, cumpliendo con restricciones cinemáticas y dinámicas. Para la correcta visualización del movimiento, se realiza una simulación representativa de un robot con las características mencionadas y diseñado exclusivamente para el presente proyecto.

Índice

1. Resumen	1
2. Introducción	4
3. Aplicación	5
4. Modelado del sistema físico	6
4.1. Modelo Cinemático	6
4.2. Modelo Dinámico	9
5. Selección de Componentes	12
5.1. Requerimientos de Sensores	12
5.2. Requerimientos de Torque en Actuadores	13
5.3. Sensores	14
5.3.1. Encoder	14
5.3.2. Sensor Diferencial de Posicionamiento Global	14
5.3.3. Sensor LiDAR	15
5.3.4. Sensor Inercial - IMU	15
5.3.5. Sistema de Báscula	15
5.3.6. Sensor de Ultrasonido	16
5.4. Actuadores	16
5.4.1. Motor Brushless DC	16
5.4.2. Controlador de motores brushless	17
5.4.3. Caja Reductora Planetaria	18
5.4.4. Sistema de Orugas	19
5.5. Otros elementos	20
5.5.1. Batería	20
6. Dimensiones Físicas y CAD	22
7. Algoritmo de Control por Modelo Inverso	26
7.1. Planificación de Trayectoria - Path Planning	26
7.1.1. Pre-procesamiento del entorno	26
7.1.2. Planificador A* (estrella) Híbrido	30
7.2. Planificación de Movimiento - Motion Planning	33
7.3. Controlador	34
7.3.1. Control Dinámico	34
7.3.2. Controlador a Lazo Cerrado en Cascada	35
7.3.3. Modulador de Torque	36
7.3.4. Desacople de Variables Físicas	37
7.3.5. Lazo de Control de Corriente	38
7.3.6. Lazo de Control PI con Modelo Dinámico Inverso	39
7.3.7. Modulador de Tensión y Modelado de Motores	40
7.4. Simulación Cinemática	42
7.5. Percepción	43
7.5.1. Fusión de Sensores	44

8. Perturbaciones	47
8.1. Cambio de Pendiente del Terreno	47
8.2. Masa Agregada	47
8.3. Efectos Disipativos por Fricción	48
9. Resultados	49
9.1. Primera trayectoria	51
9.2. Segunda trayectoria	54
9.3. Tercera trayectoria	56
9.4. Trayectoria con obstáculos dinámicos	59
10. Costos de Fabricación	61
11. Conclusiones	62
Bibliografía y Referencias	63

2. Introducción

La visión principal de este proyecto es facilitar la tarea de cosecha de vid, colaborando con la labor realizada por el operario/cosechador. El robot podrá seguir al operario de cerca durante el proceso de recolección, evitándole así tener que levantar y mover repetitivamente el cajón de uvas al ir avanzando por la hilera de vid. A través de un sistema de báscula, podrá indicarle al operario cuando haya alcanzado el peso predeterminado para cada cajón, y a través de un pulsador, podrá enviarse al robot hasta la zona de descarga o centro de recolección de la viña. Este movimiento lo realizará de forma autónoma, a través de un sistema de navegación compuesto por sistemas de localización local y global, y el sensado de su entorno.

Para ello, se determina su forma y estructura geométrica y, se dimensiona acorde a la tarea a desarrollar y la carga a trasladar. A continuación, se desarrollan los modelos cinemático y dinámico, derivando las ecuaciones correspondientes. De acuerdo a lo obtenido, se seleccionan los actuadores adecuados para realizar los esfuerzos debidos según la tarea a efectuar, así como los sensores necesarios para la misma. En este caso se consideran, además de los inherentes a la dinámica del robot, aquellos que permitirán realizar las funciones de posicionamiento local, y los necesarios para la funcionalidad de báscula del robot.

Definida la totalidad de las características del robot, se emplea el modelo dinámico para generar el controlador, junto con la aplicación de moduladores de torque con desacople de variables físicas por realimentación. Por último, se añaden perturbaciones (tanto sensadas por el controlador, como desconocidas) en las distintas variables del sistema. Con ello, se simula el mismo para evaluar su desempeño.

Por último, para evaluar y comprobar el correcto funcionamiento de todo lo mencionado anteriormente, se realiza una simulación de movimiento a través de un mapa obtenido de una imagen de una finca virtual. Se calcula una trayectoria idónea de un punto a otro dentro del mapa mediante un algoritmo de Inteligencia Artificial, generando una serie de coordenadas intermedias para que el controlador efectúe el movimiento de forma efectiva y eficiente.

3. Aplicación

Se plantea como aplicación específica, implementar un robot móvil terrestre de orugas con tracción diferencial, debido a que su aplicación sería principalmente en terrenos irregulares y a la intemperie, sumado a las variaciones que el tiempo realiza sobre la superficie de movimiento del robot. Además, el terreno puede presentar pendientes, zonas de alto agarre y zonas donde el robot posiblemente “patine” o pierda pasos en su referencia. El robot debe ser capaz de cargar cajones de uvas de hasta 25kg y de avanzar, retroceder y girar en movimiento.

Su funcionamiento en el proceso de cosecha se basa en seguir de cerca al operario, tarea que se logrará mediante un dispositivo de tamaño reducido colocado en el operario (como tobillera, pulsera, u otro formato) que contiene un módulo DGPS de precisión centimétrica, el cual actualiza constantemente la posición del operario en coordenadas de latitud y longitud. El controlador del robot sería capaz de conocer la posición del operario al que se le ha asignado su tarea de seguidor, y podría realizar el movimiento necesario para mantenerse cerca del mismo en un rango de distancia prudente. De esta forma, se podría optar como opción la utilización del robot como seguidor de una pequeña cuadrilla o grupo de 2 o 3 operarios, y que el robot mantenga la distancia al centro geométrico calculado entre las posiciones reales instantáneas de cada uno de los operarios de la cuadrilla.

El sistema de navegación autónoma que implementa para posicionarse y trasladarse por las parcelas, consistirá de un modelo redundante constituido por elementos de tecnología GNSS (*Global Navigation Satellite System*, sistema global de navegación por satélite). Para asegurar la correcta navegación del robot, y evitar que este colisione con las plantas de vid encontradas en la parcela, se agregaría una fusión de sensores compuesta por un sensor LIDAR, una IMU y encoders en ambas ruedas motrices; que permitiría detectar el camino a seguir, corrigiendo la posición del robot en tiempo real, eliminando posibles errores debido a la perdida de paso por irregularidades en el terreno o resbalamientos de las orugas; o bien, esquivar obstáculos que se encuentre en el camino, desconocidos inicialmente por el algoritmo de navegación.

Por último, el robot posee un sistema de báscula que le permite determinar el peso total que está cargando, con lo cual puede alertar al operario cuando se alcanza el peso determinado por cajón. Este se emplea a través de un transductor de fuerza (celda de carga) colocado debajo del soporte del cajón.

4. Modelado del sistema físico

Como se mencionó previamente, el robot posee orugas gracias a las cuales tracciona en las irregularidades del terreno, no obstante, se lo asume como un robot diferencial con solo 2 ruedas motoras, una de cada lado; para obtener el modelo cinemático de movimiento. Luego, sí se considera el peso total del robot, la cantidad de ruedas en cada lado, sus inercias rotacionales y demás características físicas, al momento de obtener el modelo dinámico.

4.1. Modelo Cinemático

Se procede a realizar el análisis y cálculo cinemático de un robot móvil diferencial, el cual tiene 2 ruedas conductoras colocadas en el eje perpendicular a la dirección del movimiento del robot, es decir, 2 motores conductores. Cada rueda será controlada de manera independiente una de la otra, permitiéndole al robot realizar giros hacia un lado o hacia el otro, dependiendo de la diferencia en las velocidades angulares de cada una de las ruedas.

La cinemática de un robot móvil describe la evolución de la posición/orientación del mismo en función de las variables de actuación, y para calcularla se tienen en cuenta las siguientes hipótesis:

- No hay deslizamiento entre las ruedas y el suelo.
- Se considera al robot como un sólido rígido, es decir, no sufre flexión.

Para realizar los cálculos, se toma al punto dado por la intersección entre el eje longitudinal del robot y el eje de las ruedas como punto de control, el cual se encuentra a una distancia de $\frac{L}{2}$ de cada una de las ruedas. Como se puede ver en la figura 1, los parámetros del robot diferencial son los siguientes:

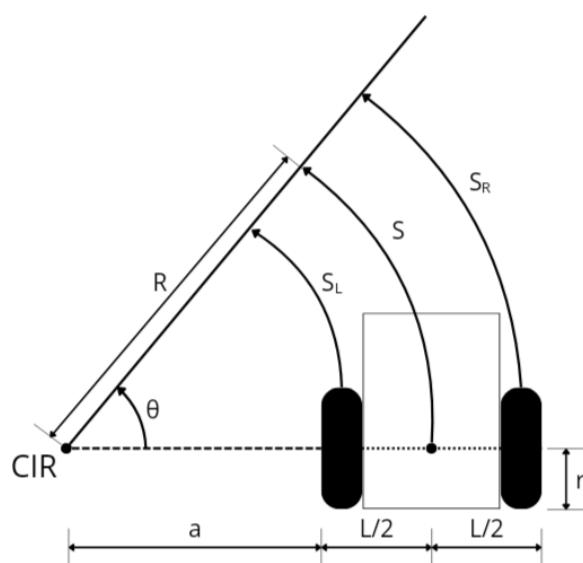


Figura 1: Configuración Cinemática. Fuente propia

- CIR: Centro Instantáneo de Rotación.

- R: Radio de giro.
- L: Distancia entre las ruedas.
- a: Distancia entre el CIR y la rueda más cercana.
- v_L : Velocidad lineal rueda izquierda.
- v_R : Velocidad lineal rueda derecha.
- r: Radio de las ruedas.
- ω_L : Velocidad angular rueda izquierda.
- ω_R : Velocidad angular rueda derecha.

La velocidad lineal de cada una de las ruedas está dada por:

$$v_L = r \cdot \omega_L \quad ; \quad v_R = r \cdot \omega_R \quad (4.1)$$

La distancia recorrida por la rueda izquierda es S_L , por la rueda derecha es S_R y por el robot referenciado al punto de control es S ; y están dadas por:

$$S_L = a \cdot \theta \quad (4.2)$$

$$S_R = (a + L) \cdot \theta \quad (4.3)$$

$$S = R \cdot \theta \quad (4.4)$$

Operando en 4.3 y por 4.2 se obtiene:

$$S_R = a \cdot \theta + L \cdot \theta$$

$$S_R = S_L + L \cdot \theta \quad (4.5)$$

Derivando 4.5 respecto al tiempo para obtener la velocidad angular:

$$\begin{aligned} \dot{S}_R &= \dot{S}_L + L \cdot \dot{\theta} \\ v_R &= v_L + L \cdot \omega \\ \omega &= \frac{v_R - v_L}{L} \end{aligned} \quad (4.6)$$

Sumando las distancias recorridas 4.2 y 4.3; y sabiendo que $R = a + \frac{L}{2}$, se obtiene:

$$S_R + S_L = a\theta + (a + L)\theta = 2a\theta + L\theta = 2\theta \left(a + \frac{L}{2} \right) = 2\theta R$$

Por 4.4:

$$\begin{aligned} S_R + S_L &= 2S \\ S &= \frac{S_R + S_L}{2} \end{aligned} \quad (4.7)$$

Derivando 4.7 respecto al tiempo para obtener la velocidad lineal del punto de control del robot:

$$\dot{S} = \frac{\dot{S}_R + \dot{S}_L}{2}$$

$$v = \frac{v_R + v_L}{2} \quad (4.8)$$

Sabiendo que, en términos generales, $v = \omega * r$; se puede calcular el radio de curvatura 'R' con 4.6 y 4.8:

$$\begin{aligned} R &= \frac{v}{\omega} = \frac{\frac{(v_R + v_L)}{2}}{\frac{(v_R - v_L)}{L}} \\ R &= \frac{L(v_R + v_L)}{2(v_R - v_L)} \end{aligned} \quad (4.9)$$

Analizando 4.9, se concluye lo siguiente:

- Si $v_R = v_L \Rightarrow R = \infty \rightarrow$ El CIR se encuentra en el infinito, el robot se moverá en línea recta.
- Si $v_R = -v_L \Rightarrow R = 0 \rightarrow$ El CIR se encuentra en el punto de control, el robot girará sobre su propio centro.
- Si $v_R \neq v_L \Rightarrow R = x \rightarrow$ El CIR se encuentra a una distancia 'x' del punto de control ($x \neq 0$), el robot girará para un sentido o para el otro.

Habiendo establecido la relación entre las velocidades lineales y angulares de las ruedas y del robot, su respuesta respecto a un punto de control y conocer la distancia a la que se encuentra de su CIR; se procede a analizar la cinemática respecto a un sistema de referencia desplazado respecto al eje de giro de cada rueda.

En la figura 2 se puede observar que el punto de control se encuentra en el punto (x_h, y_h) y el robot se orienta un ángulo θ respecto a la horizontal. Se aclara, entonces, que el eje motriz del robot (aquel que pasa por el eje de ambas ruedas motrices, una de cada lado del robot y en su respectiva oruga) se encuentra en la parte trasera del mismo. Entiéndase por 'trasera' aquella que posee los botones para que el usuario (cosechador) interactúe, dado que desde un punto de vista constructivo no posee parte delantera ni trasera.

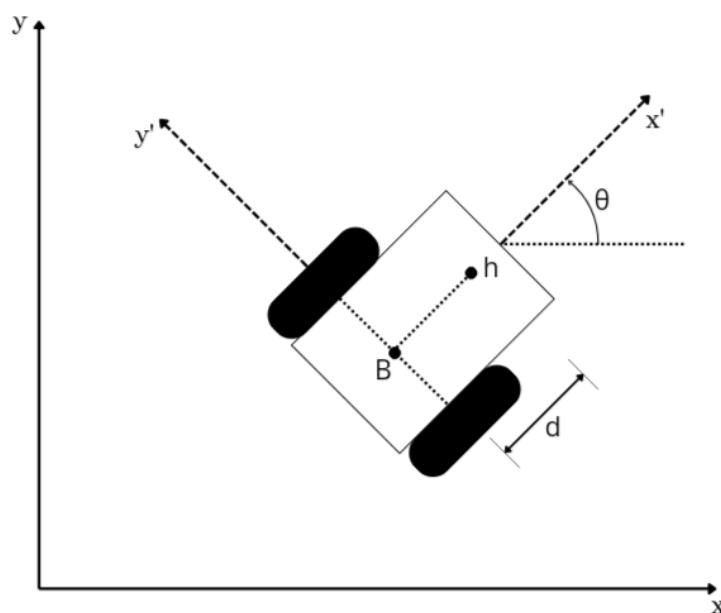


Figura 2: Punto de Control Desplazado. Fuente propia

Siendo (x_B, y_B) las coordenadas del punto medio entre las 2 ruedas (B), ‘d’ la distancia entre el punto B y el punto de control ‘h’, se buscan calcular las velocidades (\dot{x}, \dot{y}) del robot respecto al sistema de referencia fijo:

$$x_h = x_B + d \cos(\theta) \quad (4.10)$$

$$y_h = y_B + d \sin(\theta) \quad (4.11)$$

Derivando 4.10 y 4.11:

$$\dot{x}_h = \dot{x}_B - d \sin(\theta)\dot{\theta} \quad (4.12)$$

$$\dot{y}_h = \dot{y}_B + d \cos(\theta)\dot{\theta} \quad (4.13)$$

Sabiendo que:

$$\dot{x}_B = v \cos(\theta) \quad (4.14)$$

$$\dot{y}_B = v \sin(\theta) \quad (4.15)$$

$$\dot{\theta} = \omega \quad (4.16)$$

Siendo $v = \omega R$, obtenemos las ecuaciones cinemáticas del robot, con su punto de control fuera el eje de las ruedas:

$$\dot{x} = v \cos(\theta) - d \sin(\theta)\dot{\theta} \quad (4.17)$$

$$\dot{y} = v \sin(\theta) + d \cos(\theta)\dot{\theta} \quad (4.18)$$

$$\dot{\theta} = \omega \quad (4.19)$$

Donde, según 4.6 y 4.8, las velocidades v y ω , son:

$$v = \frac{(v_R + v_L)}{2} ; \quad \omega = \frac{(v_R - v_L)}{L} \quad (4.20)$$

Matricialmente queda:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -d \sin(\theta) \\ \sin(\theta) & d \cos(\theta) \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \omega \end{bmatrix} \quad (4.21)$$

4.2. Modelo Dinámico

La dinámica considera la evolución de la posición, velocidad y aceleración del robot en respuesta a los pares de actuación de las ruedas; y, se considera como punto de partida la dinámica del vehículo obtenida con la formulación de Lagrange:

$$\frac{d}{dt} \left(\frac{\partial T}{\partial \dot{q}} \right) - \frac{\partial T}{\partial q} = Q \quad (4.22)$$

Donde:

- q : vector de coordenadas generalizadas
- Q : vector de fuerzas generalizadas
- T : energía cinética del sistema

Cabe aclarar que no se tiene en cuenta la energía potencial, es decir, los pares gravitacionales, debido a que se supone que el vehículo se desplaza por una superficie plana, horizontal y sin variación de altura respecto al suelo. Luego, se realizará una corrección debida a las variaciones de pendiente del terreno, para tener en cuenta este fenómeno en el dimensionamiento dinámico del robot.

Por otro lado, la energía cinética (T) considera la suma de las energía cinética del cuerpo del robot (T_{robot}) y las energías de cada rueda (T_{r_i}).

El modelo dinámico genérico del robot móvil terrestre de orugas, va a depender de las orugas seleccionadas que, en este caso, cada una de ellas cuenta con 3 ruedas pequeñas y 2 ruedas grandes. Se asume una distribución uniforme (disco sólido) de la masa de las ruedas. Con todo esto, se plantea lo siguiente:

$$T = T_{robot} + T_{r_{izq}} + T_{r_{der}} \quad (4.23)$$

$$T_{robot} = \frac{1}{2}Mv^2 + \frac{1}{2}I_T\omega^2 + Mv\omega d \quad (4.24)$$

$$I_T = \frac{M}{12} (b^2 + l^2) \quad (4.25)$$

$$T_{r_{der}} = 3 \left[\frac{m_c}{2} r_c^2 \dot{\theta}_{cR}^2 \right] + 2 \left[\frac{m_g}{2} r_g^2 \dot{\theta}_{gR}^2 \right] \quad (4.26)$$

$$T_{r_{izq}} = 3 \left[\frac{m_c}{2} r_c^2 \dot{\theta}_{cL}^2 \right] + 2 \left[\frac{m_g}{2} r_g^2 \dot{\theta}_{gL}^2 \right] \quad (4.27)$$

Donde:

- M : masa total
- d : distancia del centro de masa al punto de control
- m_g : masa rueda grande
- m_c : masa rueda chica
- r_g : radio rueda grande
- r_c : radio rueda chica
- b : ancho del robot
- l : longitud del robot

Se calculan los cuadrados de las velocidades:

$$\omega^2 = \dot{\theta}^2 = \frac{r_g^2}{L^2} (\dot{\theta}_{gR}^2 - 2\dot{\theta}_{gR}\dot{\theta}_{gL} + \dot{\theta}_{gL}^2) \quad (4.28)$$

$$v^2 = \frac{r_g^2}{4} (\dot{\theta}_{gR}^2 + 2\dot{\theta}_{gR}\dot{\theta}_{gL} + \dot{\theta}_{gL}^2) \quad (4.29)$$

Reemplazando 4.28, 4.29, 4.6 y 4.8 en 4.23:

$$T = \frac{M}{2} \frac{r_g^2}{4} (\dot{\theta}_{gR}^2 + 2\dot{\theta}_{gR}\dot{\theta}_{gL} + \dot{\theta}_{gL}^2) + \frac{I_T r_g^2}{2L^2} (\dot{\theta}_{gR}^2 - 2\dot{\theta}_{gR}\dot{\theta}_{gL} + \dot{\theta}_{gL}^2) + \frac{Mr_g^2 d}{2L} (\dot{\theta}_{gR}^2 - \dot{\theta}_{gL}^2) + T_{r_{izq}} + T_{r_{der}}$$

Teniendo en cuenta que $\dot{\theta}_c = \frac{r_g}{r_c} \dot{\theta}_g$ es la relación entre las ruedas en 4.26 y 4.27; y que a partir de ahora $\theta_{gR} = \theta_R$, $\theta_{gL} = \theta_L$ y $r_g = r$, y recordando 4.25:

$$T = \frac{Mr^2}{2} \left[\frac{1}{4} (\dot{\theta}_{gR}^2 + 2\dot{\theta}_{gR}\dot{\theta}_{gL} + \dot{\theta}_{gL}^2) + \frac{d}{L} (\dot{\theta}_{gR}^2 - \dot{\theta}_{gL}^2) + \frac{I_T}{ML^2} (\dot{\theta}_R^2 - 2\dot{\theta}_R\dot{\theta}_L + \dot{\theta}_L^2) \right] + \frac{3}{2}m_cr^2 (\dot{\theta}_R^2 + \dot{\theta}_L^2) + m_gr^2 (\dot{\theta}_R^2 + \dot{\theta}_L^2)$$

Aplicando derivada parcial de T respecto $\dot{\theta}_R$ y $\dot{\theta}_L$, derivando respecto al tiempo; y expresando de forma matricial:

$$\begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} = Mr^2 \begin{bmatrix} \frac{1}{4} + \frac{d}{L} + \frac{I_T}{ML^2} + \frac{(3m_c+2m_g)}{M} & \frac{1}{4} - \frac{I_T}{ML^2} \\ \frac{1}{4} - \frac{I_T}{ML^2} & \frac{1}{4} + \frac{d}{L} + \frac{I_T}{ML^2} + \frac{(3m_c+2m_g)}{M} \end{bmatrix} \begin{bmatrix} \ddot{\theta}_R \\ \ddot{\theta}_L \end{bmatrix}$$

La ecuación anterior representa el Modelo Dinámico Inverso del Sistema, luego para obtener el Directo, solo debemos invertir la matriz (D) que pre-multiplica a las aceleraciones angulares:

$$\begin{bmatrix} \ddot{\theta}_R \\ \ddot{\theta}_L \end{bmatrix} = \frac{1}{Mr^2} D^{-1} \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix}$$

Se considera un Torque de fricción del 30 % del ideal obtenido en las simulaciones, para considerar posibles efectos de resbalamiento de las orugas en terrenos de muy bajo agarre. Además, agregamos un Torque de pendiente para tener en cuenta los cambios de elevación del terreno de trabajo (se asume un máximo de 30°). De esta forma queda:

$$\begin{aligned} \tau_{friccion} &= 0,3 \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} \\ \tau_{pendiente} &= \frac{r}{2} Mg \sin(\alpha) \begin{bmatrix} 1 \\ 1 \end{bmatrix} ; \quad \alpha = 30^\circ \end{aligned}$$

Se obtiene que el cálculo del torque total τ es:

$$\tau_{Total} = \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} + \tau_{friccion} + \tau_{pendiente} \quad (4.30)$$

5. Selección de Componentes

5.1. Requerimientos de Sensores

Se considera de suma importancia seleccionar adecuadamente los sensores que permitan lograr una navegación autónoma óptima, eficiente y segura del robot; y para ello se implementa una matriz de decisión, tabla 1.

En las filas de la misma, se observan las diferentes alternativas a considerar como posibles sensores, y son las siguientes:

- **Encoder incremental:** Sensor de velocidad angular de cada una de las ruedas, a partir de odometría.
- **GPS:** Sistema de Posicionamiento Global, por sus siglas en inglés.
- **DGPS:** GPS Diferencial, por sus siglas en inglés.
- **LiDAR:** Detección y medición de distancia, por sus siglas en inglés.
- **IMU:** Unidad de Medida inercial, por sus siglas en inglés.

Además, en las columnas se observan las consideraciones importantes a tener en cuenta de cada uno de los sensores:

- **Costo:** Valor monetario total, considerando la cantidad necesaria.
- **Confiabilidad:** Grado de precisión de los datos otorgados por el sensor en la aplicación en cuestión.
- **Frecuencia de muestreo:** Tasa de actualización del dato medido, en Hz
- **Adaptación al entorno:** Necesidad de modificar o no el entorno de trabajo del robot.

	Costo	Confiabilidad	Frecuencia de muestreo	Adaptación al entorno	TOTAL
Encoder Incremental	7	2	10	10	29
GPS	7	3	3	10	23
DGPS	5	9	3	7	24
LPS	0	10	4	0	14
LiDAR	8	8	8	10	36
IMU	10	7	6	10	33

Tabla 1: Matriz de Decisión. Fuente propia

Luego de plantear y analizar cada una de las consideraciones para cada una de las alternativas, se decide fusionar los 4 sensores con mayor puntaje: Encoder incremental, DGPS, LiDAR e IMU.

5.2. Requerimientos de Torque en Actuadores

Para realizar una aproximación inicial a las aceleraciones angulares de cada motor, con el fin de obtener los valores de torque necesarios en cada articulación, se deben definir los siguientes parámetros del robot, referentes a su comportamiento dinámico:

PARÁMETRO	DESCRIPCIÓN	VALOR
M	Masa total	100kg
L	Separación entre ruedas	55cm
m_g	Masa de la rueda grande	1,5kg
m_c	Masa de la rueda chica	1kg
r_g	Radio de la rueda grande	12,5cm
r_c	Radio de la rueda chica	8cm
a	Distancia al punto de control	44,5cm
I_T	Inercia total	7, 5kg.m ²
α	Pendiente máx. del suelo	30°

Tabla 2: Parámetros del robot móvil. Fuente propia

Para completar la tabla anterior, se considera un peso total del robot de 30kg, una carga de 25kg más un número conservador de sobrecarga de 10kg (contemplando lluvias que podrían llenar el canasto con hasta 10L de agua, u otros). Además, se considera al robot como un prisma de cara rectangular (recordar 4.25), que rota sobre su eje, para el cálculo de su inercia total. La masa de las orugas se obtiene del fabricante y es de 17,5kg. Con todo esto, se define el peso total en 100kg.

Habiendo definido la trayectoria y los parámetros del robot, se realiza el cálculo de torque necesario. Se asume una aceleración desde parada total (velocidad 0 m/s) a la velocidad máxima de trabajo (1 m/s), en 1 segundo. Recordando 4.8, y de acuerdo al Modelo Dinámico Inverso obtenido en 4.2, tenemos un torque nominal de:

$$\begin{aligned} \frac{v}{1s} &= \frac{r}{2s^2} (\dot{\theta}_R + \dot{\theta}_L) \\ \ddot{\theta}_R = \ddot{\theta}_L &= \frac{1m}{r2s^2} = 8 \frac{\text{rad}}{\text{s}^2} \\ \begin{bmatrix} \tau_R \\ \tau_L \end{bmatrix} &= Mr^2 D \begin{bmatrix} \ddot{\theta}_R \\ \ddot{\theta}_L \end{bmatrix} = \begin{bmatrix} 27,9119 \text{Nm} \\ 27,9119 \text{Nm} \end{bmatrix} \end{aligned}$$

Para el cálculo del torque total se considera, de forma conservadora, sumar un 30 % al torque calculado como teórico, en forma de Torque de Fricción o Deslizamiento (como se mencionó en el apartado del Modelo Dinámico).

- Torque Teórico en cada articulación: 27,9119Nm
- Torque debido a los efectos Gravitacionales: 30,66Nm
- Torque de Deslizamiento y Fricción (se considera un 30 %): 8,3736Nm

Toque total = 66,95Nm. Se considera un Torque necesario en cada rueda de 70 Nm.

5.3. Sensores

La selección de los sensores que el robot utilizará para orientarse en su entorno, está pensada desde un punto de vista de la funcionalidad de los mismos aplicadas al proyecto en cuestión. Se considera como característica ideal que todos los sensores cuenten con un grado de protección IP65 o superior, conforme la norma IEC 60529 [1]; pero en algunos casos, para reducir costos, se seleccionan componentes con grado de protección IP inferior, pero implementados dentro de una estructura que cumpla con las características de protección deseadas.

5.3.1. Encoder

Con el objeto de medir la velocidad de las ruedas grandes traseras del robot, es decir, las ruedas motrices (para obtener el mayor torque posible), se selecciona el siguiente producto, el cual sería necesario montarlo dentro de una caja estanca o similar para asegurar la impermeabilidad deseada:

- Encoder incremental AMT 103-V de CUI Devices con una resolución de hasta 2048 pulsos por revolución y frecuencia de muestreo de 100kHz [2].



Figura 3: Encoder AMT 103-V. Fuente CUI Devices

5.3.2. Sensor Diferencial de Posicionamiento Global

Se selecciona un sistema GPS Diferencial ya que corrige las medidas de posicionamiento del robot en función del error dado por la posición fija y conocida de la base de referencia, con una precisión submétrica.

- Kit DGPS TOP682-6043 de TopGNSS con doble antena, con frecuencia de muestreo de 20Hz [23].



Figura 4: DGPS TOP682-6043. Fuente TOPGNSS

5.3.3. Sensor LiDAR

El sensor LiDAR es un sensor de medición de distancia dentro de un rango angular específico, que permite detectar posibles obstáculos estáticos y/o dinámicos, y así esquivarlos logrando una navegación local segura.

- LiDAR kit D500 de LDROBOT, con frecuencia muestreo de 5000Hz [9].



Figura 5: LiDAR D500. Fuente LDROBOT

5.3.4. Sensor Inercial - IMU

La IMU es un sensor compuesto por acelerómetro, giróscopo y magnetómetro, lo cual nos proporciona datos de aceleraciones, velocidad y posición angular respecto a los 3 ejes (X, Y y Z).

- IMU WT901BLEVL de Wit Motion, con frecuencia de muestreo de 200Hz [25].



Figura 6: IMU WT901BLEV. Fuente Wit Motion

5.3.5. Sistema de Báscula

Con el fin de detectar cuándo se han cargado 25kg de uvas en el cajón, aproximadamente (considerando un margen para tierra, ramas, etc.), se selecciona un dispositivo capaz de traducir el peso de la masa apoyada sobre el robot, en una señal medible por el mismo. Para ello se selecciona:

- Módulo celda de carga AD-HX711 con 4 sensores de 50 kg c/u de Si Tai&SH [21].



Figura 7: Módulo celda de carga AD-HX711. Fuente Si Tai&SH

5.3.6. Sensor de Ultrasonido

De forma redundante a los sensores previamente mencionados y seleccionados, se agregan sensores ultrasonidos en la parte delantera y trasera del robot, con el fin de establecer un perímetro último de seguridad en cualquiera de sus modos de funcionamiento. Se selecciona:

- Sensor ultrasónico Telemecanique XX630A1KAM12 con distancia de 1m regulable o no regulable con pulsador de aprendizaje [22].



Figura 8: Sensor ultrasónico. Fuente Telemecanique

5.4. Actuadores

5.4.1. Motor Brushless DC

De acuerdo al Torque calculado, asumiendo una velocidad máxima de $1\frac{m}{s}$ que equivalen a $8\frac{\text{rad}}{\text{s}}$ y tomando como base una velocidad nominal de 2.800rpm en el motor, se calcula el torque de motor necesario (T_e) en cada rueda, y así poder seleccionar los actuadores.

$$i = \frac{n_s}{n_e} = \frac{8 \frac{\text{rad}}{\text{s}}}{2.800 \text{rpm} \left(\frac{\pi}{30} \right)} = 0,0273 \quad (5.1)$$

$$T_e = T_{si} = 1,91 \text{ Nm} \quad (5.2)$$

Teniendo en cuenta lo calculado en 5.2 y los parámetros indicados en la Tabla 2, se selecciona el siguiente motor sin escobillas de corriente continua (BLDC):

BM1418HQF - 48V 650W [20]



Figura 9: Motor Brushless DC. Fuente Tuoxiang Store

BLDC Motor
BM1418HQF
350-750W *MOTOR*

> Parameters

Specification	BM1418HQF(BLDC)			
Rated Output Power	350W	500W	650W	750W
Rated Voltage	48V DC	48/60V DC	48/60V DC	48/60V DC
Rated speed	2800RPM	2800RPM	2800RPM	2800RPM
No load speed	3100RPM	3100RPM	3100RPM	3100RPM
Full load Current	≤ 9.4A	≤ 13.4/10.8A	≤ 18.0/14.5A	≤ 20.0/16.0A
No load Current	≤ 2.8A	≤ 3.2/2.8A	≤ 4.0/3.5A	≤ 5.0/4.5A
Rated Torque	1.19 N.m	1.7 N.m	2.20 N.m	2.56 N.m
Efficiency	≥ 75 %	≥ 75 %	≥ 75%	≥ 75%
Gear ratio	1:10.4			
Application	Small and Medium size E-Tricycle			

Figura 10: Parámetros de Motor BLDC BM1418HQF

5.4.2. Controlador de motores brushless

Para lograr un control adecuado de ambos motores BLDC, se selecciona un controlador capaz de soportar las corrientes pico, la potencia y la alimentación del motor, contemplando un controlador por cada uno de los dos motores.

- Controlador BLDC UNITEMOTOR para BM1418HQF de 48V, 38A y 750W [19].



Figura 11: Controlador BLDC UNITEMOTORL. Fuente AMG Power Solutions

El rango de alimentación es de 10 a 55 VCC, por lo tanto no es necesario el uso de un inversor de tensión.

5.4.3. Caja Reductora Planetaria

Se seleccionó una caja reductora planetaria, teniendo en cuenta una relación de transmisión de 40, cuyo rendimiento a par nominal es como mínimo del 94 %:

iHF PRF80 Series [10].

Specifications		PRF40	PRF60	PRF80	PRF90	PRF120	PRF160
Technal Parameters							
Max. Torque	Nm			1.5times rated torque			
Emergency Stop Torque	Nm			2.5times rated torque			
Max. Radial Load	N	185	240	400	450	1240	2250
Max. Axial Load	N	150	220	420	430	1000	1500
Torsional Rigidity	Nm/arcmín	0.7	1.8	4.7	4.85	11	35
Max.Input Speed	rpm	8000	8000	6000	6000	6000	4000
Rated Input Speed	rpm	4500	4000	3500	3500	3500	3000
Noise	dB	≤55	≤58	≤60	≤60	≤65	≤70
Average Life Time	h			20000			
Efficiency Of Full Load	%			L1≥96%	L2≥94%		

Figura 12: Parámetros técnicos 1 caja reductora. Fuente Hefa Gear Machinery

Technical Parameter	Level	Ratio	PRF40	PRF60	PRF80	PRF90	PRF120	PRF160	
Rated Torque	L1	3 Nm	/	27	50	96	161	384	
		4 Nm	16	40	90	122	210	423	
		5 Nm	15	40	90	122	210	423	
		7 Nm	12	34	48	95	170	358	
		10 Nm	10	16	22	56	86	210	
	L2	12 Nm	/	27	50	95	161	364	
		15 Nm	/	27	50	96	161	364	
		16 Nm	16	40	90	122	210	423	
		20 Nm	15	40	90	122	210	423	
		25 Nm	16	40	90	122	210	423	
	L2	28 Nm	16	40	90	122	210	423	
		30 Nm	/	27	50	96	161	364	
		35 Nm	12	40	90	122	210	423	
		40 Nm	16	40	90	122	210	423	
		50 Nm	15	40	90	122	210	423	
	L2	70 Nm	12	34	48	95	170	358	
		100 Nm	10	16	22	56	80	210	
Degree Of Protection			IP65						
Operation Temperature			- 10°C to -90°C						
Weight	L1	kg	0.43	0.98	2.3	3.12	7.08	15.5	
	L2	kg	0.65	1.26	2.97	3.82	8.7	17	

Figura 13: Parámetros técnicos 2 caja reductora. Fuente Hefa Gear Machinery

Para verificar los cálculos, comenzamos desde el Motor entregando 2,56N.m a una eficiencia mínima del 75 %, a 2.800rpm; luego la relación de transmisión de la caja reductora (40) y su eficiencia mínima (94 %):

$$T_{\text{salida}} = 2,2 \text{Nm} * 40 * 0,94 = 82,72 \text{Nm} \quad (5.3)$$

Esto comprueba satisfactoriamente los cálculos de dimensionamiento y la selección de los elementos del Robot.

5.4.4. Sistema de Orugas

Por último se seleccionaron los sistemas de orugas que se utilizarán en el robot. Dichos sistemas cuentan con una disposición lineal, cuentan con la rueda motriz en la parte trasera y son de longitud modificable según la necesidad. Se seleccionó la oruga DP-PY-118B de la empresa Shanghai Puyi Industrial Co. Ltd [11], cuyos parámetros (figura 15) corresponden a un solo tren de orugas, es decir, se seleccionan dos del mismo modelo. Se aclara que la figura 14 no corresponde al modelo específico seleccionado, es la imagen brindada por el fabricante debido a que se fabrican a pedido según dimensiones y formatos.



Figura 14: Tren de Orugas. Fuente Shanghai Puyi Industrial Co. Ltd.

Tipo: PY-118B
Enlace nº20
Pitch: 60mm
Width: 118mm
El total de 1.200 mm de longitud:

Figura 15: Parámetros orugas. Fuente Shanghai Puyi Industrial Co. Ltd.

5.5. Otros elementos

En esta sección, además de seleccionar la batería que alimentará a todo el robot, se mencionan otros componentes de sensado del entorno (ultrasonido, y cámara), pero su implementación no está desarrollada en la simulación. Esto se debe a que su aplicación no forma parte del algoritmo de navegación y control del robot; representan elementos de seguridad última (ultrasonido) y son más aplicables al modo de funcionamiento de seguimiento de operario (cámara).

5.5.1. Batería

Por último, se selecciona el elemento encargado de alimentar de energía a todo el robot, contemplando una batería de ciclo profundo (80 %) de bajo peso. Para ello, se elige la UPower UE-12Li50BL:

- Batería de Litio UPower UE-12Li50BL, de 12V y 50 Ah, de ciclo profundo [24].



Figura 16: Batería UE-12Li50BL. Fuente UPower

Para evaluar la selección de esta batería, calculamos la energía total disponible, contemplando una descarga del 80 % y la eficiencia mínima de los motores de 75 %:

$$50\text{Ah} * 12\text{V} * 0,8 * 0,75 = 360\text{Wh} \equiv 1.296.000\text{J} \quad (5.4)$$

Luego, se realiza el ejercicio de suponer un funcionamiento con un torque promedio en requerimiento constante (50 Nm). Esto es equivalente a considerar el peso máximo estipulado (100 kg) moviéndose de forma alternada entre una pendiente máxima considerada

(30°), un plano horizontal y un plano inclinado descendente. Con ello, y el radio de las ruedas motrices (0,125m), se obtiene la ‘distancia’ máxima a recorrer en estas condiciones:

$$\frac{1.296.000J * 0,125m}{50Nm} = 3.240m \quad (5.5)$$

Luego, si se toma un camino promedio como 75m, una jornada de trabajo de 8 horas por día, y estimando que en dicha jornada, un cosechador puede llenar aproximadamente 16 cajones de buena calidad de uva y libre de palos, ramas u hojas, tenemos 32 viajes del robot por día (ida y vuelta). De esta forma, aproximadamente, se tiene la cantidad de ‘días’ de autonomía que ofrece esta opción:

$$\frac{3.240m}{75\frac{m}{viaje} * 32\frac{viajes}{dia}} = 1,35 dias \quad (5.6)$$

De esta forma, en condiciones de funcionamiento con un elevado grado de complejidad, el robot posee una autonomía superior a un día de trabajo. Por lo tanto, se acepta esta batería como una opción viable para alimentar a todo el sistema ya que, de forma preventiva permite ser utilizado en una jornada laboral completo y recargar sus baterías durante la noche.

6. Dimensiones Físicas y CAD

En esta sección se exponen las dimensiones físicas reales que tendría el robot, observables en un diseño CAD simplificado, realizado en el software Solid Edge. Al realizar dicho diseño se buscó dar una noción más realista de como podría ser el robot finalmente construido, esto implica que, de efectuarse dicha fabricación en un futuro, podría estar sujeto a modificaciones físicas de ser necesario.

En las figuras 17, 18 y 19, se puede observar el diseño realizado, en el cuál se pueden observar sus dimensiones finales, en milímetros.

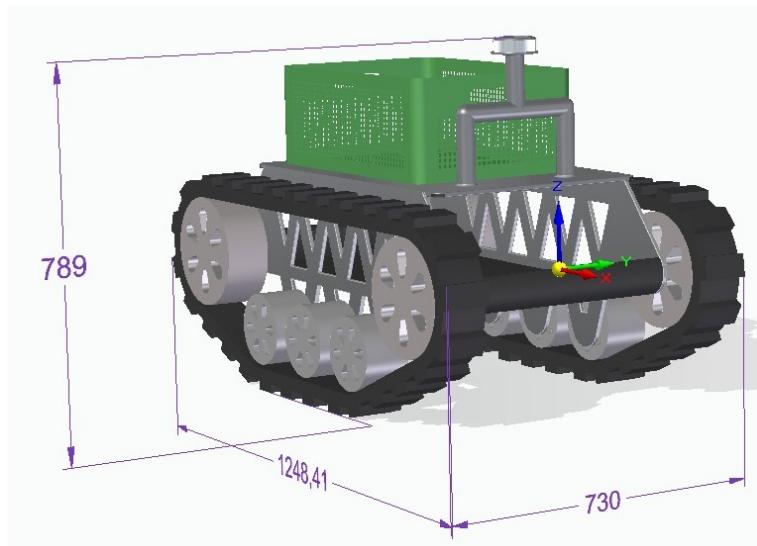


Figura 17: Perspectiva. Fuente propia

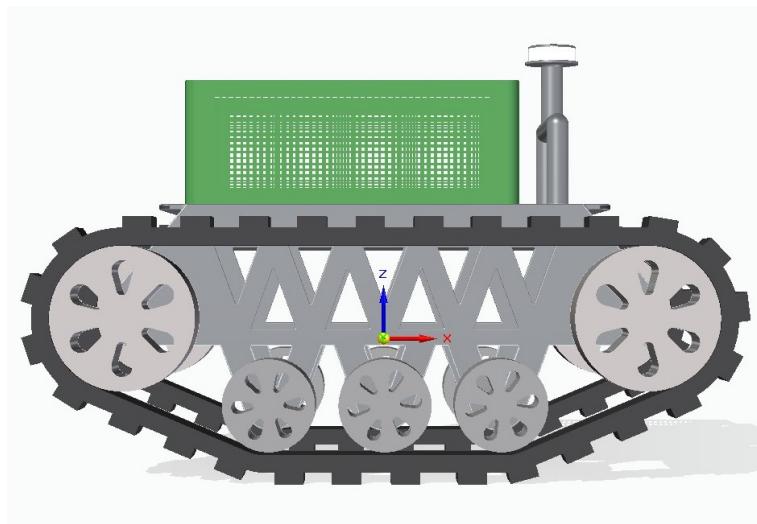


Figura 18: Vista Lateral. Fuente propia

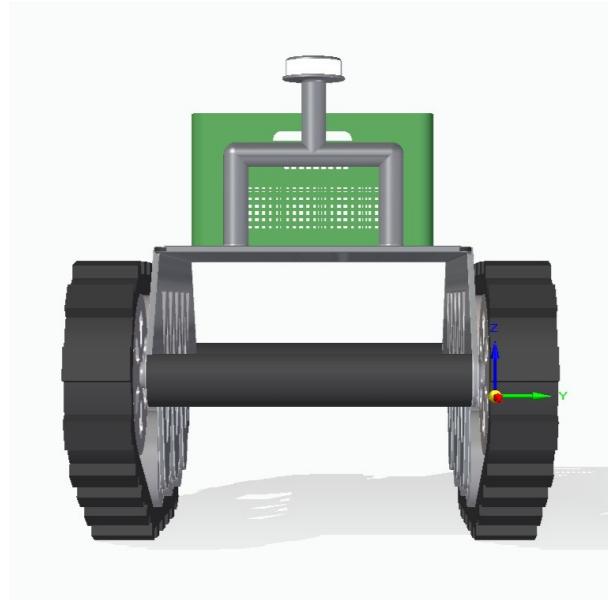


Figura 19: Vista Frontal. Fuente propia

En el modelo del robot RASOC, se puede observar que, por debajo de donde se encuentra el cajón requerido para realizar la cosecha, se ha dispuesto una estructura estilo bandeja, en la cual se dispondrían los diferentes elementos de hardware ya seleccionados. Dicha disposición se distribuye de tal manera que el centro de masa del robot se encuentre situado lo más próximo al centro geométrico del mismo, evitando así complicaciones dinámicas y/o cinemáticas durante el funcionamiento del robot. Teniendo en cuenta que, de los elementos de hardware necesarios, la batería y los motores serían los de mayor peso y los que mayor influencia tendrían en el desplazamiento del centro de masa del robot, se buscaría que dicho elementos se dispongan en extremos opuestos, por lo tanto, los motores se encontrarían en la parte trasera de la mencionada bandeja y la batería en la parte delantera. De esta manera, para la disposición de los elementos restantes, no sería necesario respetar una ubicación específica sino que se puede dar según practicidad y facilidad del cableado, siempre respetando la primicia de mantener un centro de masa total centrado.

A continuación se encuentran imágenes obtenidas del mundo virtual creado en el software SketchUp, del cual se obtuvo la imagen superior de la finca para implementar en el algoritmo de planificación de trayectoria. De esta forma, se expone visualmente un entorno virtual semejante a una realidad en la cuál se encontraría el robot fabricado, en funcionamiento con otros robots similares y con personas a las cuales asistirían durante la temporada de cosecha de una finca.



Figura 20: Mundo virtual - Seguidor de operario. Fuente propia



Figura 21: Mundo virtual - Seguidor de operario 2. Fuente propia



Figura 22: Mundo virtual - Navegación autónoma. Fuente propia



Figura 23: Mundo virtual - Navegación autónoma 2. Fuente propia

7. Algoritmo de Control por Modelo Inverso

El robot asistente de cosecha, debe seguir al operario de cerca durante la recolección, a través de sus sensores y ajustando el torque en las ruedas. Luego, cuando se le indique volver a la base, deberá moverse de forma autónoma por el viñedo, a través de las parcelas, hasta llegar a la posición objetivo. Con el fin de lograr dicho objetivo, se diseña un sistema de control basado en una trayectoria dada, y la acción sobre el motor de cada rueda motriz.

En esta sección se expone el desarrollo completo de la implementación en Matlab [13], Simulink [14] y Gazebo[3] con comunicación mediante ROS2 [18]; con lo cual se logra la navegación autónoma del robot en una finca virtual, trasladando el cajón de uva lleno desde una posición inicial hasta una posición final. En la figura 24 se puede observar un esquema general de lo mencionado anteriormente y en las secciones siguientes se detallará cada una de ellas.

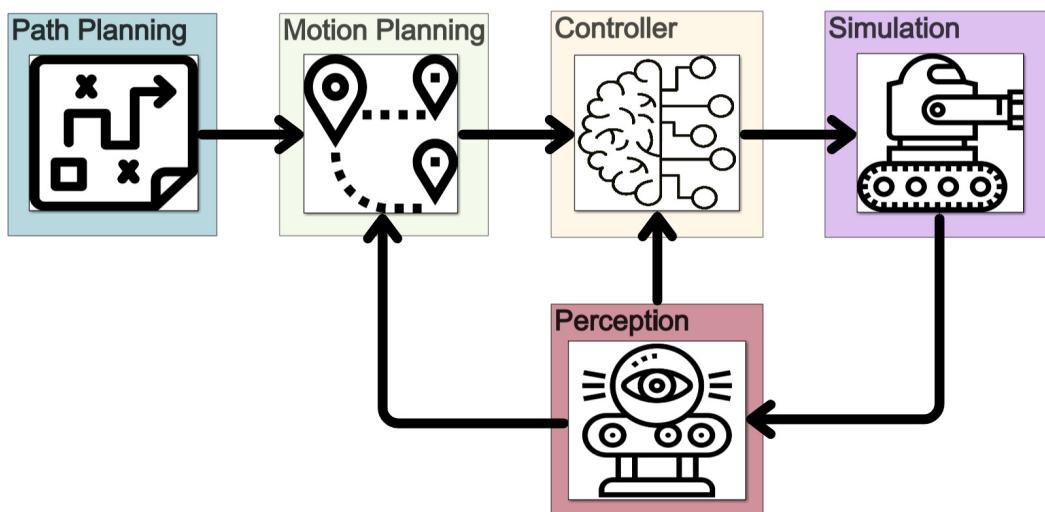


Figura 24: Diagrama de bloques completo. Fuente propia

7.1. Planificación de Trayectoria - Path Planning

Se debe planificar una trayectoria acorde a la tarea del robot, en la que debe recorrer las hileras en modo autónomo, ya sea para trasladar el cajón de uvas lleno hacia un punto de acopio de los mismo o para regresar con un cajón vacío para continuar con la recolección del fruto por parte del operario. Para lograr esto, es necesario que el robot conozca el mapa del terreno en el cual deberá desplazarse y, para lograrlo, se implementan las siguientes etapas:

- Tomar una foto área de la finca en la cual se realizará.
- Procesar la imagen para obtener un mapa binario de ocupación.
- Generar un mapa inflado de colisiones según dimensiones del robot.
- Implementar un algoritmo de IA para obtener la trayectoria más óptima.

7.1.1. Pre-procesamiento del entorno

Para el caso de estudio, no se cuentan con las herramientas que permitan obtener una imagen aérea real de una finca, por lo que se diseño en el software SketchUp, una finca

virtual de dimensiones similares a una real. Teniendo el mundo virtual, se obtiene una fotografía superior de la misma (figura 25), del mismo modo que se realizaría con un drone y una finca real.

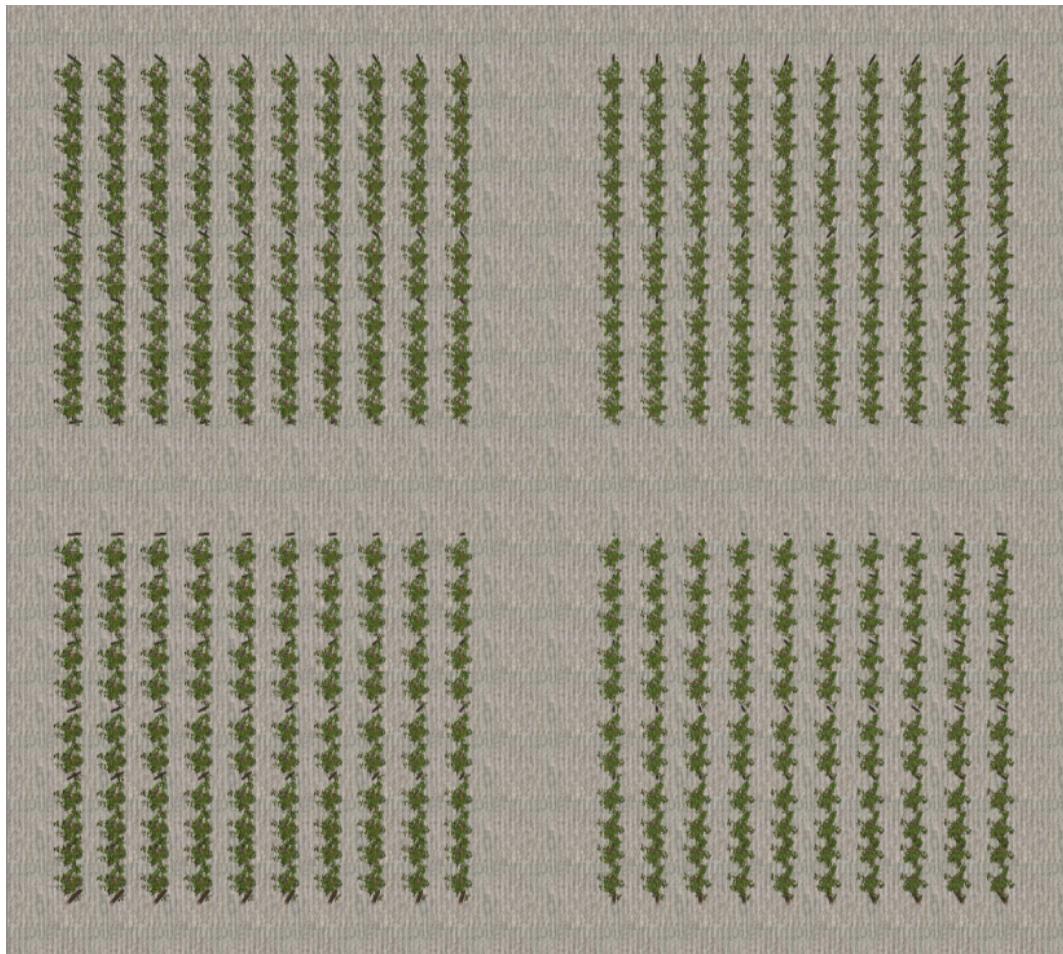


Figura 25: Imagen aérea de la finca virtual. Fuente propia

Al obtener la imagen aérea del terreno, se procede a procesar dicha imagen para obtener un mapa binario de ocupación. Para lograr esto se utilizó la herramienta de MATLAB llamada ‘Image Labeler’, la cual permite seleccionar regiones ocupadas, de forma manual o automática. Para realizar dicha selección de forma correcta y consistente con las dimensiones reales de una finca, de los pasillos entre las vid y del robot, se estableció como parámetro una resolución de aproximadamente 16 píxeles por metro. Esto permitió definir que, cada una de las hileras tenga un ancho de unos 20 centímetros y que la separación entre ellas ronde los 1.8 metros.

En la figura 26 se puede observar una captura de pantalla de dicha herramienta con los sectores ocupados por hilera de vid seleccionados por regiones y, en la figura 27 se visualiza el mapa obtenido, donde la superficie del terreno en cuestión es de 50x45 metros, aproximadamente. Los sectores ocupados se observan en negro y tienen un valor de 1 y, las regiones libres de obstáculos se observan en blanco y valen 0, es por eso que se lo denomina mapa binario de ocupación.

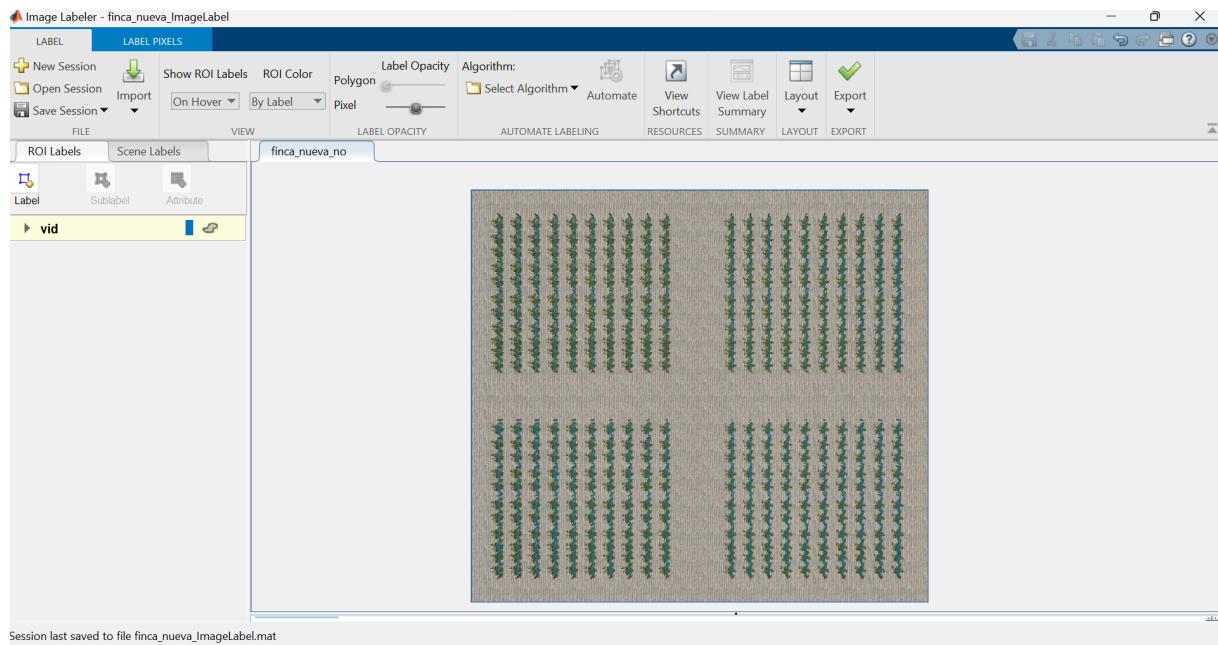


Figura 26: Image Labeler de Matlab. Fuente propia

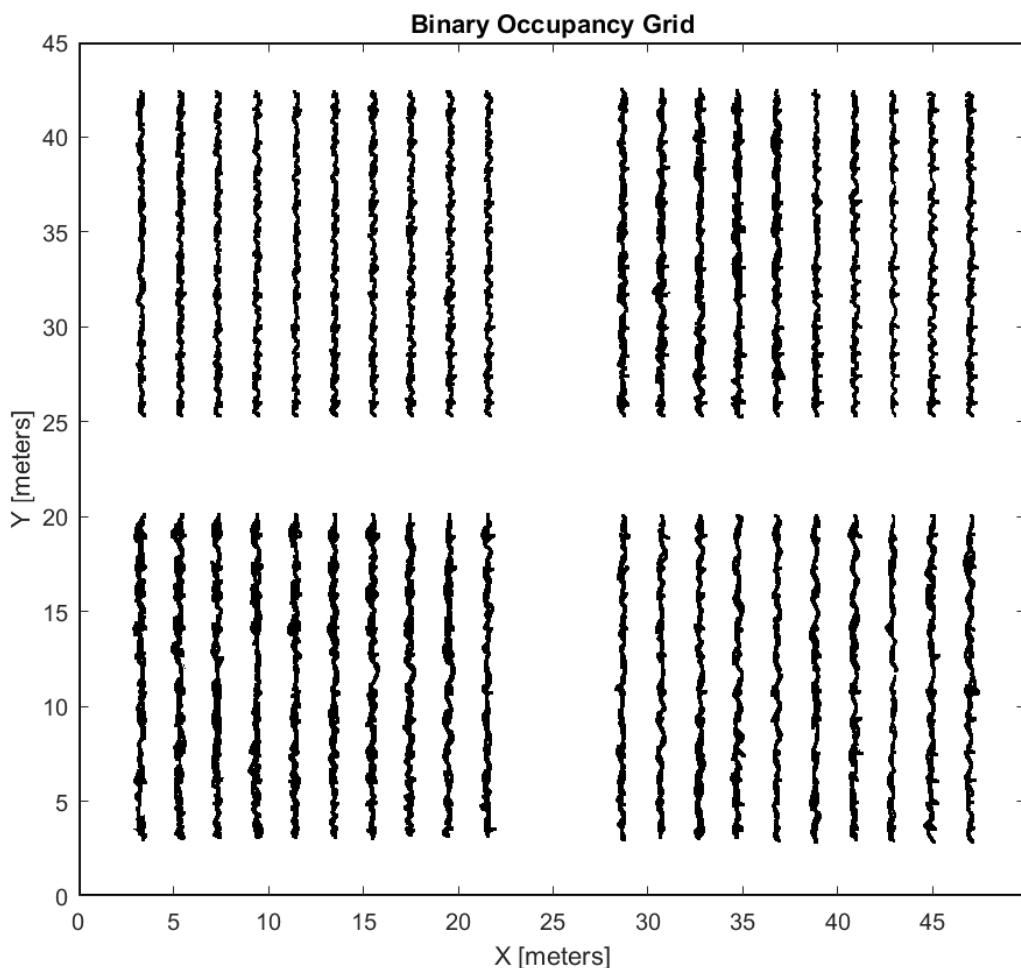


Figura 27: Mapa binario de ocupación. Fuente propia

Una vez obtenido el mapa del terreno como un mapa binario de ocupación, se procede a

contemplar las dimensiones reales del robot para verificar las regiones por las cuales podría transitar de acuerdo a su configuración y a sus restricciones cinemáticas. Se implementa una función que permite validar colisiones en función del largo y ancho del robot, así como también de la separación entre ruedas y del radio mínimo de giro (el cual, a pesar de tratarse de un robot diferencial, no se considera 0 ya que demandaría una elevada potencia debido a las orugas y al terreno).

Con el objetivo de que el robot transite por los pasillos principales del terreno y no entre dos hileras de vid, se implementó la planificación de trayectoria en 2 etapas:

- **1er Etapa:** Se desarrolla un algoritmo que, en función de la ubicación del robot en el mapa y de su orientación al momento de iniciar el modo de navegación autónoma, encuentre las coordenadas del punto más cercano por fuera del pasillo interno entre dos hileras de vid, considerando las restricciones de movimiento previamente mencionadas.
- **2da Etapa:** Se implementa la planificación de trayectoria con algoritmo de IA, pero tomando como coordenadas iniciales las encontradas previamente, y como coordenadas finales las de la posición pre-establecida del centro de acopio o recolección de la finca.

De esta forma se logra asegurar que el robot sea capaz de salir de un pasillo interno al iniciar el modo autónomo, pero que durante todo el movimiento no vuelva a ingresar a ninguno de los demás pasillos internos que se pudiese encontrar durante la navegación. Esto permite destacar una planificación de trayectoria robusta y aplicable a cualquier mapa que se implemente. Cabe destacar que, para asegurar que el robot no ingrese a los pasillos internos entre las hileras de vid, se sobre-dimensiona en un 20 % aproximadamente las dimensiones del robot, garantizando así que el mapa de costos final restrinja por completo la posibilidad de circulación en dichas regiones.

En la figura 28 se puede observar el mapa de costos que se obtiene al implementar todo lo mencionado anteriormente, donde se puede observar en color rojo las regiones prohibidas (que en el mapa binario anterior eran negras), en color rosa las regiones prohibidas debido a las restricciones geométricas y cinemáticas del robot, y en blanco las regiones de circulación permitidas.

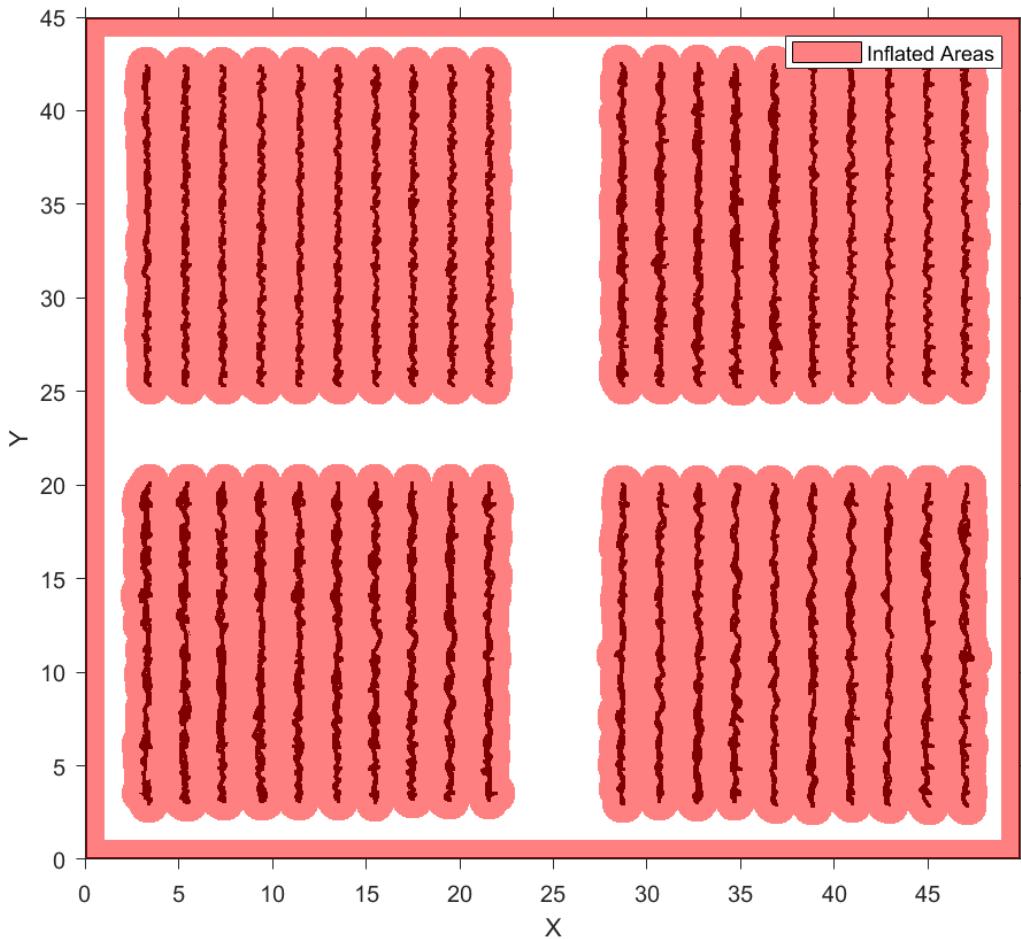


Figura 28: Mapa de costos con áreas infladas. Fuente propia

7.1.2. Planificador A* (estrella) Híbrido

Habiendo cumplido todas las etapas de procesamiento de la imagen del terreno, se procedió a implementar la planificación de la trayectoria que deberá seguir el robot, de forma autónoma, desde una posición a otra. Para esto, se utilizó el algoritmo de optimización A* híbrido, basado en Inteligencia Artificial, mediante la función de MATLAB ‘*plannerHybridAStar()*’ [7]. Dicha función recibe como parámetro el mapa de costos y las coordenadas de las posiciones iniciales y finales (siendo la posición inicial la obtenida en la 1er etapa de planificación y no la real del robot), en formato ‘ (x, y, θ) ’, donde ‘ θ ’ representa el ángulo de orientación. Finalmente, se obtiene una serie de coordenadas o ‘waypoints’ los cuales indican todos los puntos por los que el robot debe pasar para lograr la trayectoria deseada.

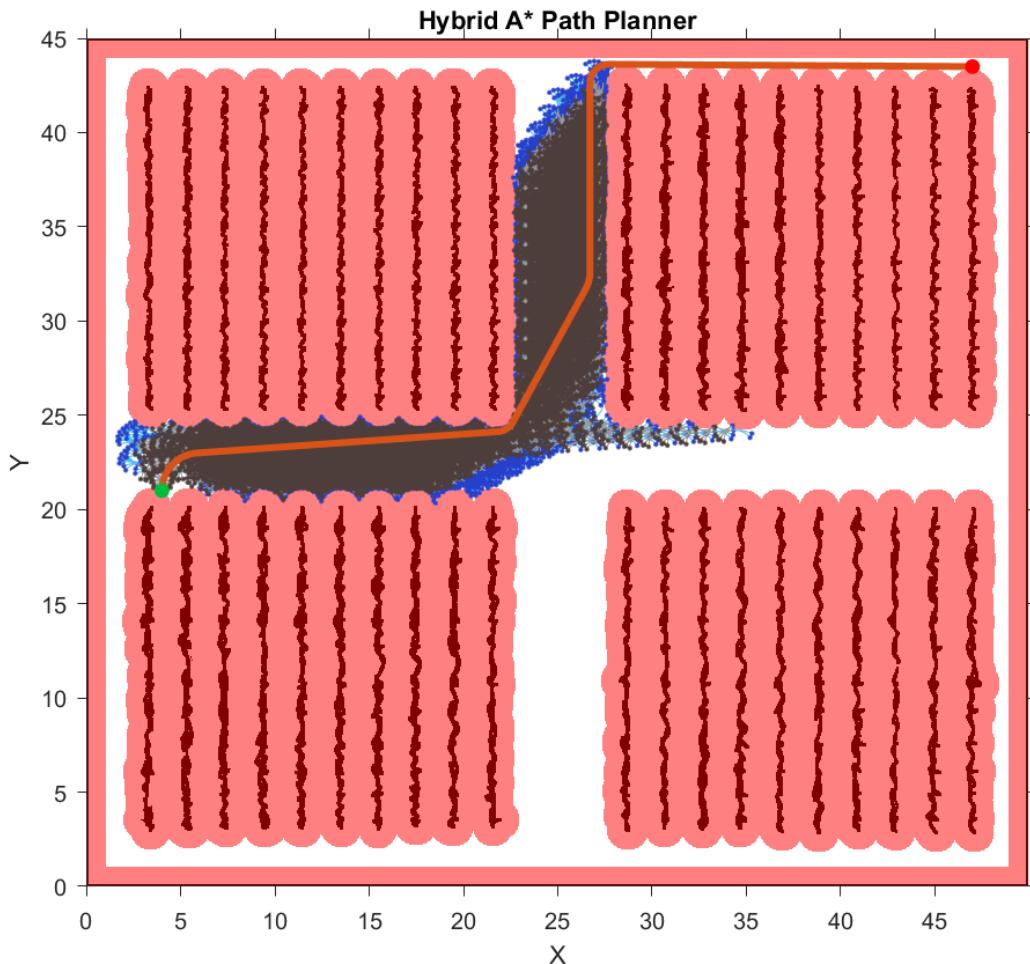


Figura 29: Trayectoria dada por en planificador. Fuente propia

En la figura 30 se observa una ampliación de uno de los sectores de la figura 29, en el que se visualiza con mayor detallamiento y claridad el comportamiento del algoritmo de planificación A* Híbrido implementado. El algoritmo evalúa el costo de movimiento de cada uno de los nodos mediante una expansión analítica hacia la ubicación de la coordenada objetivo o final. Dicha expansión se realiza 5 veces por nodo, a una distancia de 1 metro y en diferentes ángulos, donde los límites de cada ángulo de búsqueda está dado por el radio mínimo de giro del robot, según restricciones no-holonómicas dadas por la cinemática del mismo.

Luego de realizar la expansión de todos los posibles nodos en la dirección indicada y desde la posición inicial del robot, el algoritmo selecciona la mejor trayectoria según valores de costo de cada tramo, los cuales a su vez se determinan, entre otras cosas, en función de la dirección de movimiento (hacia adelante o hacia atrás) y de la cantidad de cambios de dirección de una trayectoria completa.

Cabe aclarar que la expansión de todos los nodos del algoritmo se realiza evaluando primordialmente el valor de cada posición en la matriz binaria de ocupación, es decir que, si la ubicación del nodo en dicha matriz corresponde a una posición ocupada por algún obstáculo, se lo descarta como opción a analizar y expandir dentro del árbol de búsqueda.

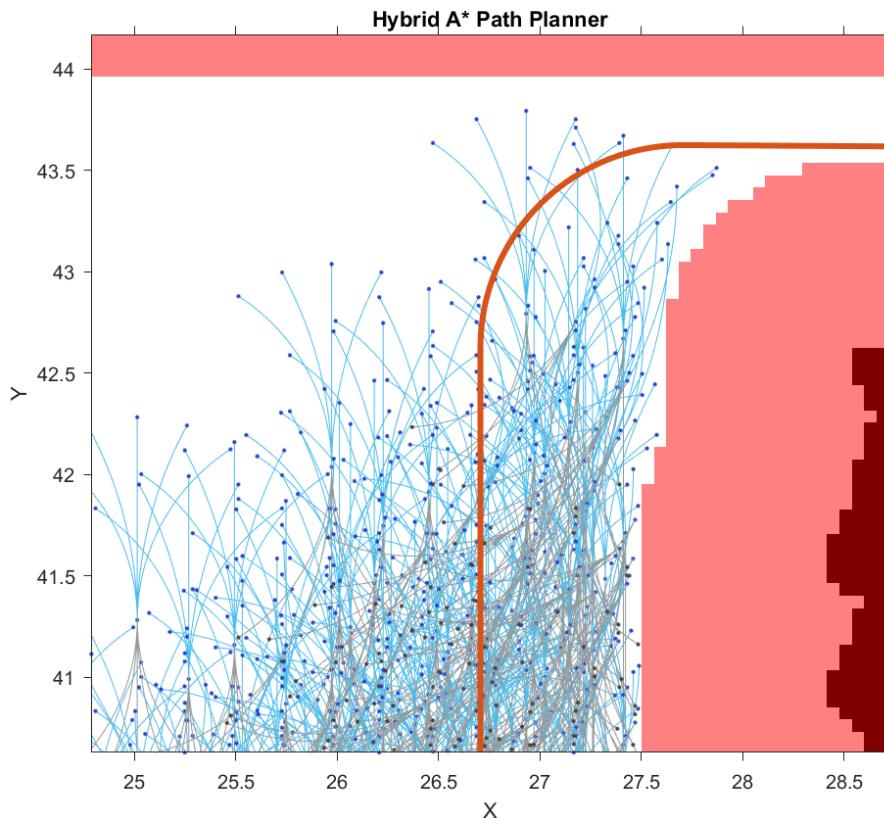


Figura 30: Imagen ampliada del algoritmo de planificación. Fuente propia

Considerando las etapas mencionadas en el apartado anterior, al vector de ‘waypoints’ que se obtiene, se le agrega como inicio, las coordenadas de posición correspondientes a la posición real del robot dentro de uno de los pasillos internos de las hileras de vid, obteniendo así un vector completo de trayectoria.

En la figura 31 se puede observar la trayectoria que deberá realizar el robot en la simulación alrededor del mapa de la finca, cumpliendo con cada uno de los ‘waypoints’ especificados. El camino desde el punto A al punto B es el obtenido en la 1er Etapa de la planificación de trayectoria, y el camino desde el punto B al punto final o punto C es el obtenido en la 2da Etapa.

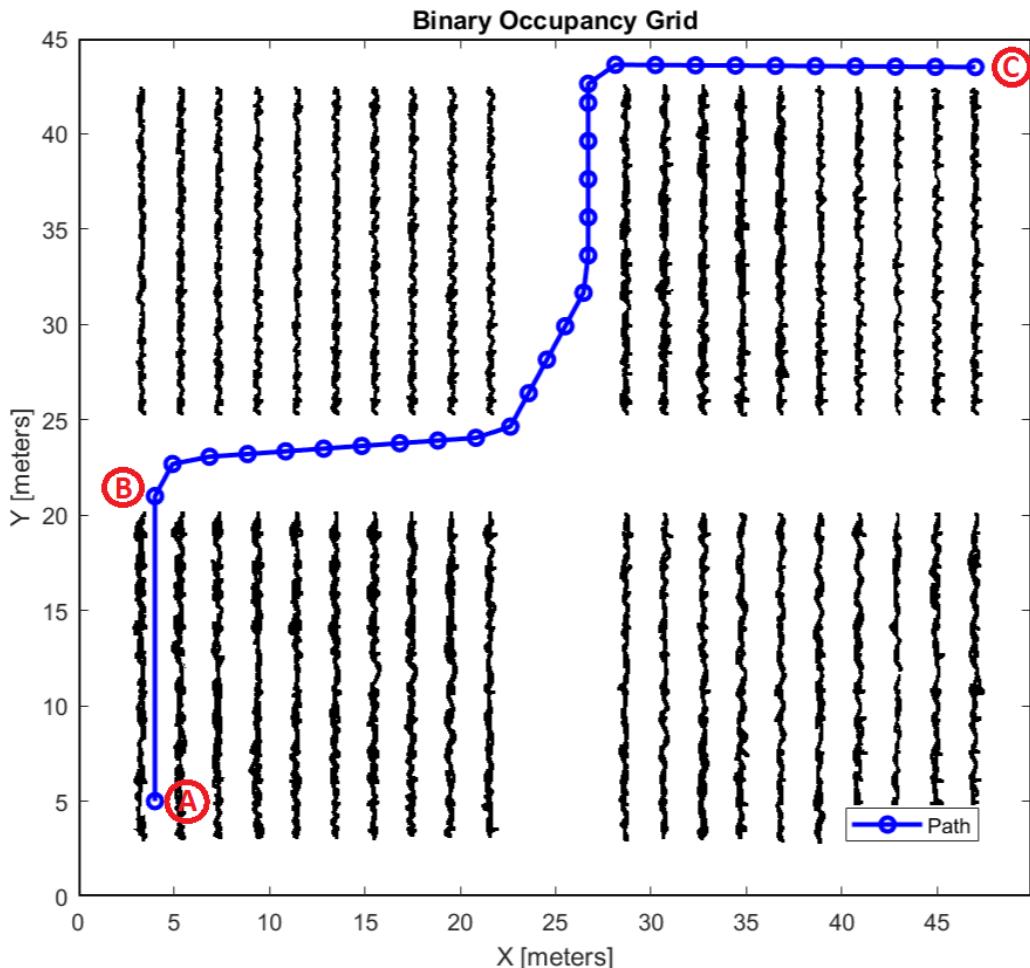


Figura 31: Trayectoria planificada. Fuente propia

7.2. Planificación de Movimiento - Motion Planning

En términos generales, en la etapa de Planificación de Movimiento se busca que el robot siga de forma armónica, suave y sucesiva las diferentes coordenadas de cada uno de los 'waypoints' que componen la trayectoria completa. Para lograrlo, se compara cíclicamente la posición actual del robot con la posición del próximo punto del camino a satisfacer, con un margen de distancia de 1 metro, y de esta forma se determina la velocidad lineal ' v ' y la velocidad angular ' ω ' del robot para lograr recorrer cada una de dichas coordenadas. Este ciclo finaliza una vez que la posición del robot coincida o se aproxime a las coordenadas objetivo de la Planificación de Trayectoria.

Considerando lo mencionado, es importante destacar que para ejecutar una correcta Planificación de Movimiento es de suma importancia implementar un algoritmo más específico que permita descomponer dicha planificación en dos métodos que se complementan entre sí:

- **Planificación Global:** Corresponde a lo mencionado anteriormente sobre el seguimiento sucesivo de cada una de las coordenadas de la trayectoria dada por los 'waypoints'. Sólo se consideran obstáculos estáticos conocidos a partir del preprocesamiento del mapa de la finca o del terreno, tal como se detalla en la sección 7.1.1.

- **Planificación Local:** Consiste en evitar obstáculos dinámicos que puedan aparecer y estorbar la traslación del robot dada por la planificación global.

En la figura 32 se observan los bloques implementados en Simulink para lograr complementar ambos métodos de planificación o navegación de forma eficiente.

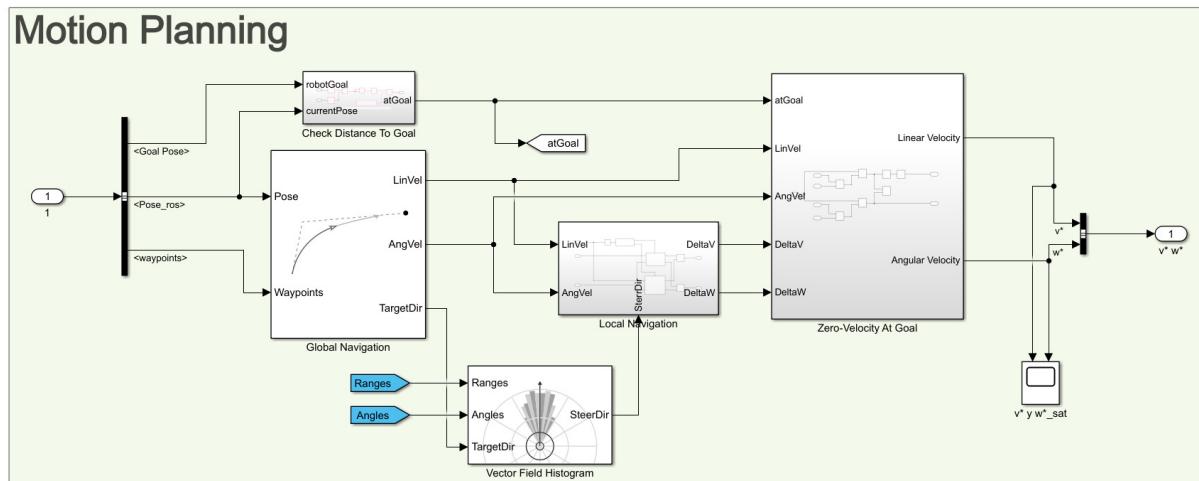


Figura 32: Planificación de Movimiento. Fuente propia

Luego de ejecutar la planificación global, tal como se explicó al comienzo de esta sección, se utiliza la información proporcionada por el sensor LiDAR, configurado con una amplitud de escaneo de 260° , con un intervalo de 2° y un rango de medición de distancia desde 10cm a 3m; y se genera un mapa local de densidad de obstáculos denominado *VFH - Histograma de Campo Vectorial* [15], por sus siglas en inglés. A partir de dicho mapa y de la dirección objetivo hacia la cual debe dirigirse el robot, dada por la navegación global, el algoritmo es capaz de calcular una nueva dirección libre de obstáculos y las nuevas velocidades lineal ' v ' y angular ' ω ' del robot, necesarias para esquivar el obstáculo satisfactoriamente y retomar su trayectoria global.

Los bloques de *Check Distance to Goal* y *Zero-Velocity At Goal* en conjunto realizan una comparación de la posición actual del robot y la final a la que debe llegar. Al coincidir dichas coordenadas y siendo la velocidad del robot prácticamente nula, los bloques permiten detener el robot y la ejecución del algoritmo de forma segura.

7.3. Controlador

7.3.1. Control Dinámico

Esta etapa constituye el Sistema de Control de los Actuadores, en sí mismo. Se comienza con una comparación entre la velocidad angular derecha e izquierda deseada, y aquella realimentada por los sensores (encoders) correspondiente. El error de esta comparación se introduce en un bloque de control mono-articular PI (Proporcional Integrador).

En la figura 33 se puede ver el bloque completo de Simulink del comportamiento Dinámico del sistema. Se diferencian las secciones del Controlador, y del Sistema Dinámico (simulado). Este último, en la aplicación real, estaría reemplazado por el robot en sí mismo.

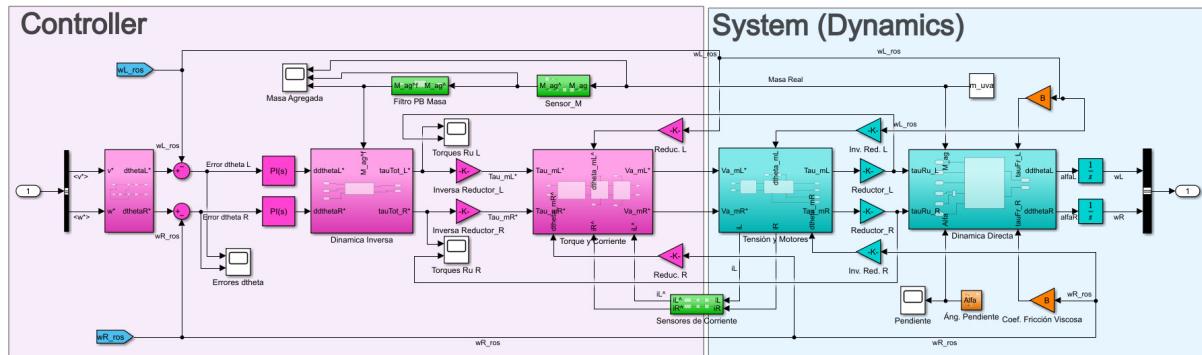


Figura 33: Bloques de Simulación Dinámica. Fuente propia

Se propone, entonces, un Control Mono-articular para el robot. Este método se justifica ya que solo se tienen dos actuadores independientes uno del otro, tanto en función como en referencia. Además, la presencia de reductores en ambas ruedas disminuye el acoplamiento entre ejes, con lo cuál es posible el control diferenciado de cada uno a través de su propio lazo cerrado.

El error en las consignas de velocidad entra a un control Proporcional-Integral, y su salida generará los valores de aceleración a producir en cada rueda, con los que se ingresa al bloque de Dinámica Inversa. Esto es lo que se conoce como Control por Modelo Inverso, ya que se utiliza la Dinámica propia del sistema para generar una señal de control mejor y más acorde al comportamiento (no solo cinemático, sino dinámico) del robot.

De dicho bloque, se obtendrán los valores de Torque necesarios en cada motor, y luego de multiplicarlos por la inversa de la constante de Reducción de la Caja (Inversa Reductor) entraremos al bloque de Torque y Corriente. Este, detallado en secciones siguientes, propone un Modulador de Torque encargado de asegurar la corriente necesaria en cada motor, para producir las aceleraciones angulares buscadas.

A su salida, vemos el ingreso al Sistema (Planta), iniciando con el bloque de Tensión y Motores. Como su nombre lo indica, este incluye el comportamiento (no ideal) del Modulador de Tensión, y la transformación de dicha Tensión en Torque en cada motor [17]. A su salida, y a través del bloque de reducción de las cajas, ingresamos al Modelo Dinámico Directo del Sistema, este entregará las aceleraciones reales que sufren las ruedas, y luego de integrarlas, obtendremos las velocidades angulares que serán realimentadas (por los sensores), al inicio del Controlador.

7.3.2. Controlador a Lazo Cerrado en Cascada

En esta parte del proyecto se presenta una estrategia de control a lazo cerrada denominada control en cascada. Consiste en implementar dos lazos de control, uno interno y otro externo. El primero es un lazo rápido de control de corriente y torque y el segundo es un lazo de velocidad angular. Se propone este tipo de controlador en contraposición al control por realimentación completa de estado.

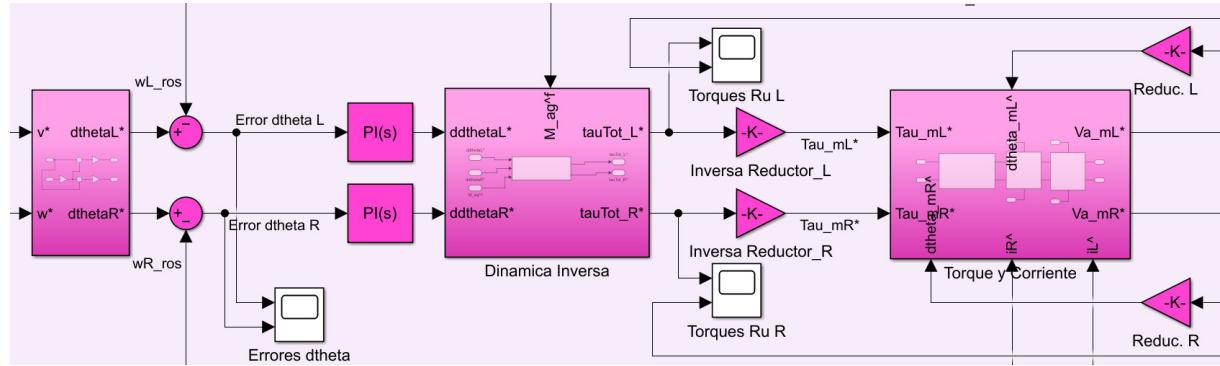


Figura 34: Bloques del Control Dinámico. Fuente propia

El controlador posee varias partes: un modulador de torque, que transforma las consignas de torque en tensiones; un modulador de tensión, que transforma las consignas (señales) de tensión en tensiones efectivas en los bornes de la máquina; y un controlador de velocidad PI que recibe un error de velocidad y genera las correspondientes consignas de aceleración. El modulador de torque se obtendrá al compensar las realimentaciones naturales del modelo físico y plantear un lazo interno de corriente.

Es importante remarcar que, si bien en el presente desarrollo se plantean los mencionados Moduladores de Torque, de Tensión, y el Lazo rápido de Corriente; en la práctica, los mismos no serían sintonizados ni implementados, ya que se encontrarían incluidos en el Controlador de los motores BLDC.

7.3.3. Modulador de Torque

En esta sección se implementará un modulador de torque que permita controlar el sistema mediante el uso de consignas de torque (τ^*) como parámetro de entrada, y que provenga del lazo externo de control, para obtener la señal de tensión a la salida.

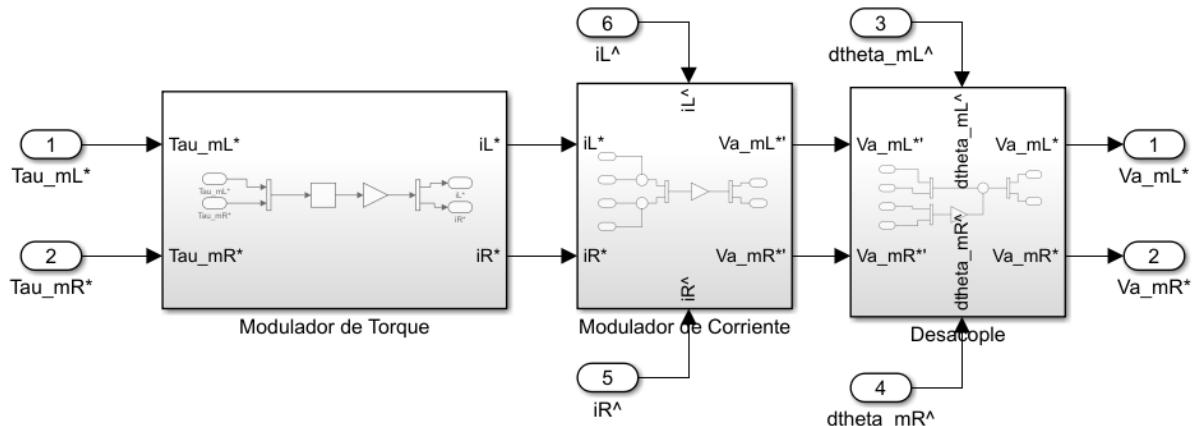


Figura 35: Bloque de Moduladores de Torque y Corriente. Fuente propia

Para ello, debemos expresar las ecuaciones dadas en [17] que gobiernan el motor seleccionado, asumiendo un comportamiento como motor de Corriente Continua por simplicidad, despreciando los efectos inductivos por falta de datos, y los efectos disipativos

por estar contemplados en el Torque de fricción estimado en un 30%; tenemos:

$$-\tau(t) = \tau(t)* \quad (7.1)$$

$$\tau(t) = k_t i_a(t) \quad (7.2)$$

$$V_a(t) = R_a i_a(t) + \dot{\theta}_m(t) k_b \quad (7.3)$$

Con las variables y constantes:

- $k_t = 0,128 \text{ Nm/A}$ (Constante de Torque)
- $i_a(t)$: Corriente del Motor
- $V_a(t)$: Tensión del Motor
- $\dot{\theta}_m(\text{rad/s})$: Velocidad Angular del Motor
- $R_a = 2,4 \Omega$ (Resistencia del Devanado del Motor)
- $k_b = 0,0793 \text{ V/rad/s}$ (Constante de Voltaje Contra-Electromotriz)

De 7.2 vemos que el bloque de Modulador de Torque, el cual define las corrientes consigna del Controlador y al cual se le agrega un limitador de Torque para proteger a los motores, se compone de la siguiente manera:

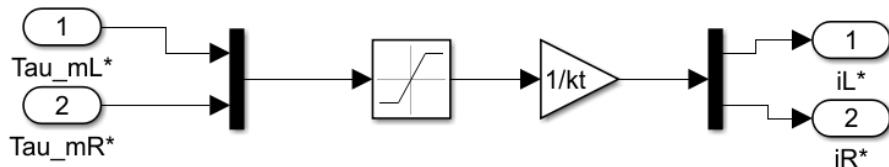


Figura 36: Modulador de Torque. Fuente propia

7.3.4. Desacople de Variables Físicas

En esta sección, se desea desacoplar el aporte de las variables físicas (en este caso la velocidad angular) de las ecuaciones vistas anteriormente, más específicamente, en la ecuación 7.1. Se busca obtener una ley de control que relacione directamente la tensión de alimentación con la corriente de los motores, de modo que, resulta evidente realizar la siguiente realimentación:

$$V_a^* = V_a^{*''} + \dot{\theta}_m k_b \quad (7.4)$$

Con:

- V_a^* : Tensión Consigna
- $V_a^{*'}$: Tensión Consigna Desacoplada

Realizando estas compensaciones en cada motor, tenemos acceso directo a manipular el torque electromagnético, sin los efectos de las realimentaciones físicas (velocidad angular). Al sustraer estos términos de las ecuaciones dinámicas y agregarlos al control, estamos realizando una técnica llamada linealización por realimentación (feedback linearization), ya que, de existir, este método permite eliminar los términos no lineales de la Ley de

control, en efecto, linealizándola. Esto se puede realizar siempre que el modelo del sistema físico sea suficientemente bueno, así como las salidas de los sensores, que el controlador pueda calcular los términos no lineales lo suficientemente rápido y que la acción resultante no cause la saturación del actuador.

Vemos entonces cómo queda el bloque de Desacople:

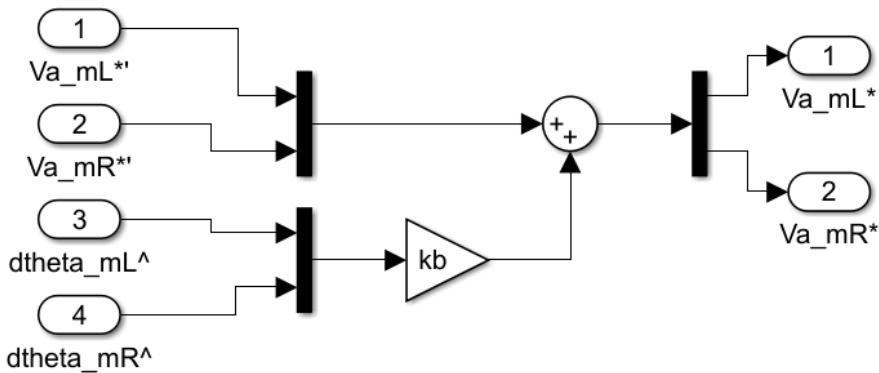


Figura 37: Desacople por Realimentación de Variables Físicas. Fuente propia

7.3.5. Lazo de Control de Corriente

Buscamos realizar el control de la tensión de alimentación de los motores ($V_a^{*'}$), para lo cual debemos realimentar el sistema con un lazo de corriente. Podemos observar la relación entre la tensión y la corriente, de la ecuación 7.3, y teniendo en cuenta la realimentación vista en 7.4, se busca una ley Proporcional que relacione la Tensión de Alimentación Desacoplada ($V_a^{*'}$) con el error en la consigna de corriente (i_a^*):

$$R_a i_a = V_a^{*' \prime} = R_p' (i_a^* - i_a) \quad (7.5)$$

Despejando y buscando la función de transferencia con Laplace, sabiendo que R_p' será la ganancia del lazo interno de control, tenemos:

$$\frac{I_a}{I_a^*} = \frac{1}{\frac{R_a}{R_p'} + 1} \quad (7.6)$$

Se refleja que, idealmente, la ganancia del controlador debería ser tan grande como se pueda. No obstante, por 7.5, vemos que grandes cambios generarían consignas de tensión demasiado grandes y bruscas. Por ello, se toma un valor de R_p' de 20 veces R_a , con lo cuál el error es menor al 5 %:

$$R_p' = 20R_a \quad (7.7)$$

$$\frac{I_a}{I_a^*} = 0,952 \quad (7.8)$$

Vemos entonces, el bloque del Modulador de Corriente, recordando que se alimentan las corrientes consigna (i_a^*) y las corrientes medidas (\hat{i}_a):

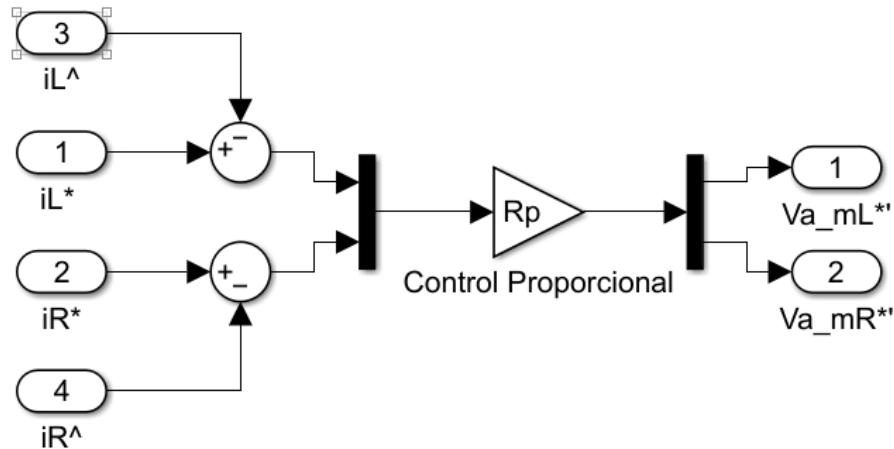


Figura 38: Lazo Proporcional en Modulador de Corriente. Fuente propia

7.3.6. Lazo de Control PI con Modelo Dinámico Inverso

El lazo de control cerrado principal, es un Proporcional Integrador que genera las consignas de aceleración a partir del error de las velocidades angulares respecto a las consignas propias del Planificador de Trayectorias. La selección de la parte Proporcional cumple con el estándar en control, siendo esta corrección acorde al error cometido; y por otro lado, el Integrador se encarga de corregir el error de estado estacionario presente en el sistema. Finalmente, no se incluye la parte Derivativa (que compondría un control PID), ya que, si bien esta mejora (en teoría) la respuesta del sistema y reduce considerablemente el sobre-pico en la corrección, es muy sensible al ruido de alta frecuencia. Esto es de interés, ya que los encoders en cada rueda estarán midiendo velocidades de giro, y las irregularidades del terreno generaran grandes variaciones repentinas en dichas medidas; si se incluye un control Derivativo en el modelo, estas variaciones se verían amplificadas y la respuesta total empeoraría considerablemente.

En las simulaciones realizadas, se encontró que es mejor el comportamiento al no utilizar un bloque Derivativo.

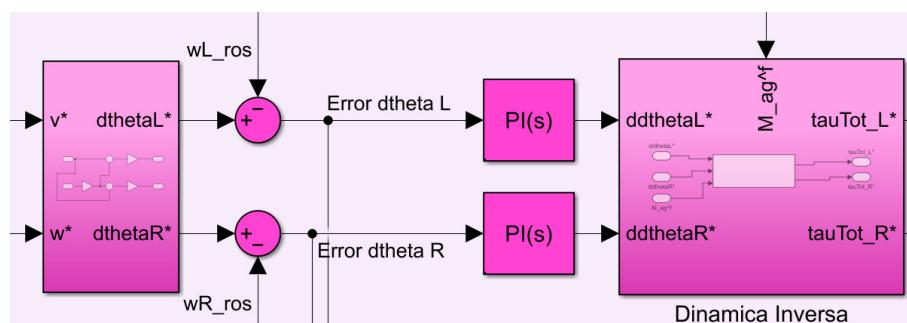


Figura 39: Control Proporcional Integral y Modelo Dinámico Inverso. Fuente propia

Para la sintonización de las ganancias P e I, se utilizó la herramienta de *PID Tuner* de MATLAB, enfocando la optimización en mejorar el seguimiento de consignas, que demostró tener mejor comportamiento que aquel optimizado para rechazo de perturbaciones. Se

obtuvieron, entonces, las siguientes constantes:

$$P = 56,215 \frac{1}{s} \quad (7.9)$$

$$I = 55,318 \frac{1}{s^2} \quad (7.10)$$

Estas consignas, ingresan al Modelo Dinámico Inverso del sistema, como se mencionó anteriormente, generando así el Control por Modelo Dinámico Inverso. El objetivo es generar un modelo lo más cercano posible a la realidad, disminuyendo al mínimo las incertidumbres (que se aceptan como errores de modelado).

Esto es una variación de un Control por Modelo Interno (IMC, en inglés), que es obtenible cuando el sistema es Lineal (como es este caso). Un IMC propone que solo se debe realimentar el error entre la salida estimada y la real. Por lo tanto, en una situación de modelado perfecto, no sería necesario cerrar el lazo de control, y el error realimentado sería de 0.

7.3.7. Modulador de Tensión y Modelado de Motores

Una vez que la señal de tensión (consigna) sale del controlador, ingresa al Sistema Real (Simulado).

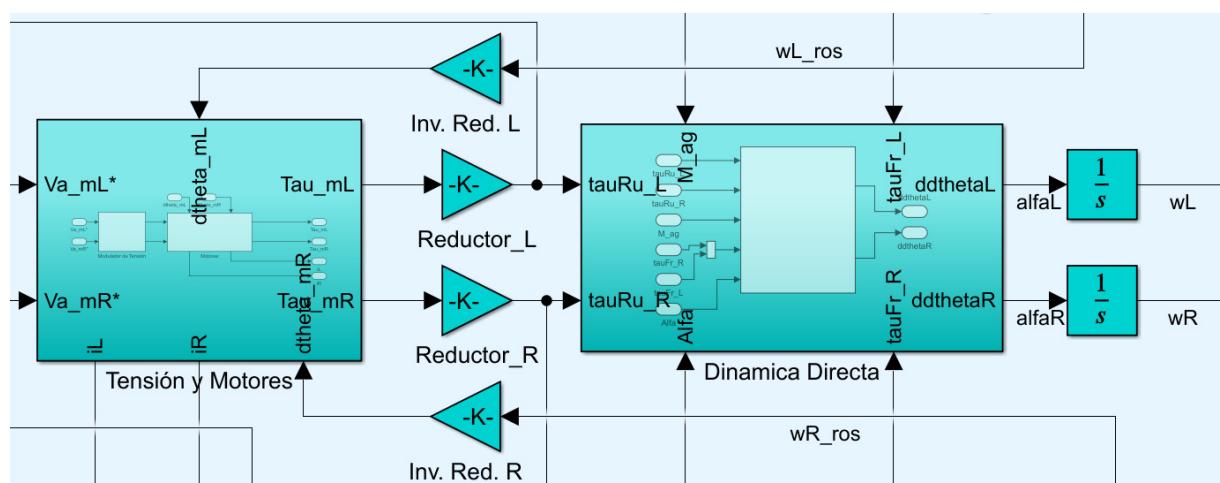


Figura 40: Sistema Real Simulado. Fuente propia

En la figura 41, se observa cómo las señales de tensión ingresan al bloque Tensión y Motores, dónde se consideran efectos no ideales del Controlador de los motores BLDC (Modulador de Tensión) y los efectos electromagnéticos de los mismos, y se terminan por obtener los valores de torque reales producidos por ambos motores.

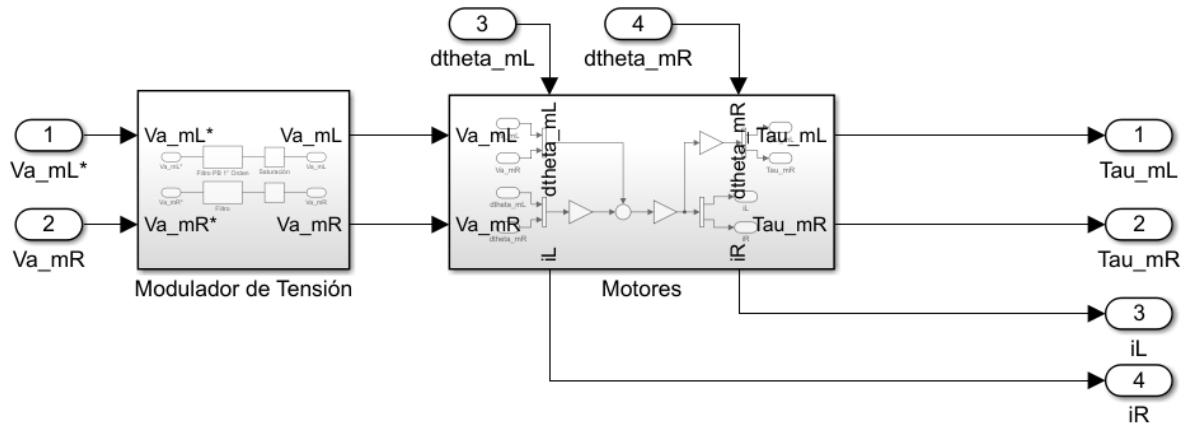


Figura 41: Bloque de Tensión y Motores. Fuente propia

Modulador de Tensión

En este apartado, se simula el funcionamiento y respuesta de un dispositivo electrónico encargado de generar la tensión en los bobinados de los motores (Controlador de motores BLDC). Para ello, se impone una limitación de tensión máxima igual a 48 V (tensión nominal de los motores, por protección) y un comportamiento como filtro pasa bajo de primer orden. Este último, se asume con una constante de tiempo de 0,005s. La función de transferencia de un filtro pasa bajo es:

$$G(s) = \frac{1}{1 + \tau s} \quad (7.11)$$

Se puede demostrar que la función de transferencia de un modelo en espacio de estados se calcula con la ecuación:

$$G(s) = C(sI - A)^{-1} B + D \quad (7.12)$$

Con ello, la representación en el espacio de estados del filtro queda:

$$A = \begin{bmatrix} -1 \\ \tau \end{bmatrix} \quad (7.13)$$

$$B = [1] \quad (7.14)$$

$$C = \left[\begin{array}{c} 1 \\ 0 \end{array} \right] \quad (7.15)$$

$$D = [0] \quad (7.16)$$

Vemos entonces el modulador de tensión propuesto:

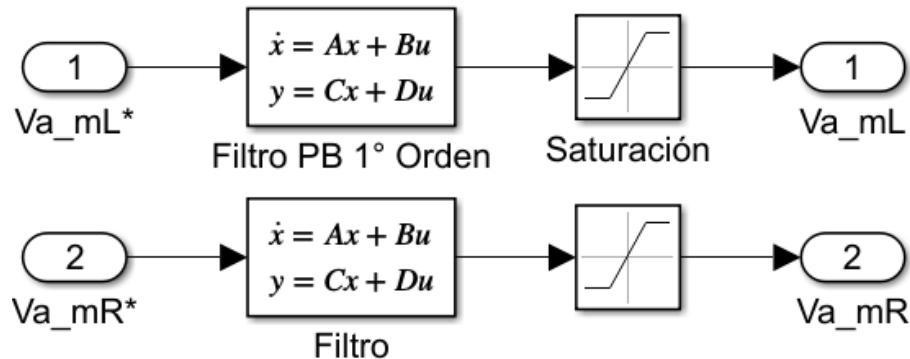


Figura 42: Modulador de Tensión. Fuente propia

Modelado de Motores

De acuerdo a las ecuaciones 7.1, 7.2 y 7.3; podemos obtener las corrientes reales de los motores, y los torques entregados a su salida. Vemos entonces su diagrama en bloques:

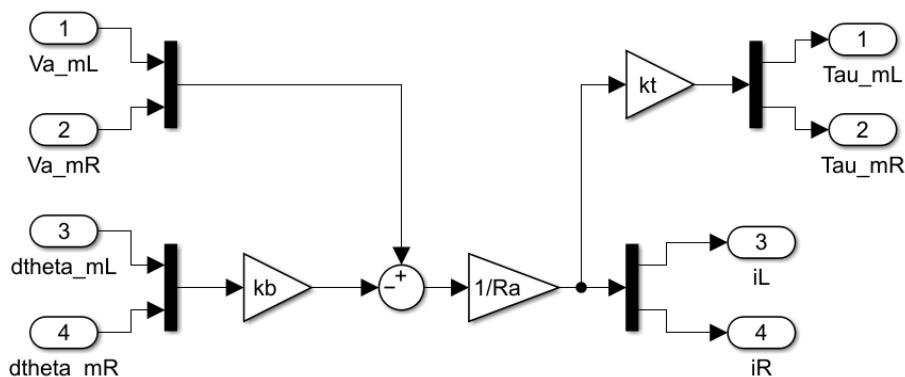


Figura 43: Bloque de Modelado de los Motores. Fuente propia

7.4. Simulación Cinemática

Se busca lograr una simulación cinemática correcta y acorde a todas las secciones mencionadas anteriormente, es por eso que se utiliza el entorno de simulación 3D llamado Gazebo en un sistema operativo Linux/Ubuntu 22.04 que se ejecuta en una máquina virtual. La comunicación entre la CPU con Simulink en Windows y el entorno Gazebo en la máquina virtual con Linux, se logra gracias a la implementación de ROS (Sistema Operativo de Robótica, por sus siglas en inglés) [12]. A pesar de lo que indica su nombre, ROS no es en sí mismo un Sistema Operativo, más bien se lo define más bien como un ‘middleware robótico’, es decir, una colección de frameworks para el desarrollo de software de robots.

La comunicación entre Simulink y Gazebo a través de la red creada por ROS2, se logra gracias al toolbox llamado *ROS Toolbox* [16], figura 44. Cabe destacar que ROS no solo es útil para simulaciones, sino que también es apto para implementar el hardware y lograr la construcción efectiva de robots.

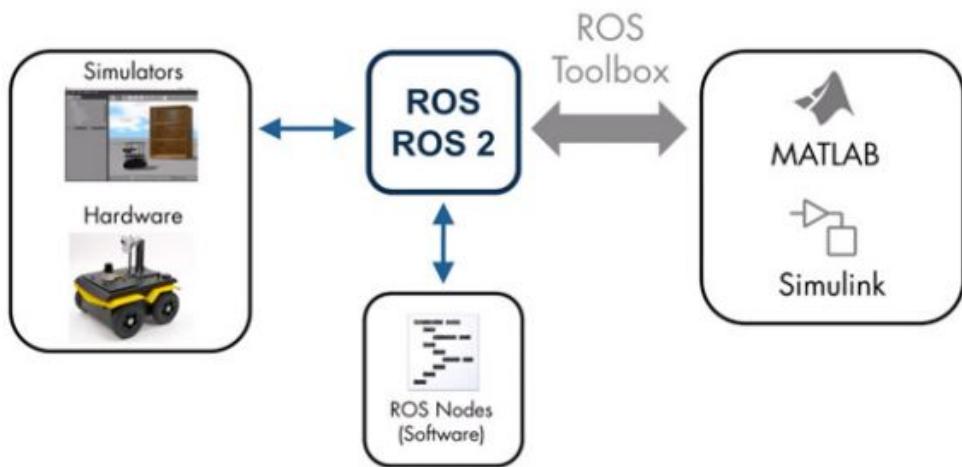


Figura 44: Comunicación entre Simulink y Gazebo mediante ROS2. Fuente Matlab

Una vez obtenidas las velocidades angulares de cada rueda ω_L y ω_R por el controlador mono-articular, y a partir de las ecuaciones 4.1 y 4.20, se obtiene la velocidad lineal ' v ' y la velocidad angular ' ω ' del robot. Dichas velocidades se ingresan al entorno de simulación 3D utilizando los bloques de ROS2 en Simulink que permiten establecer la comunicación creando nodos *Publicadores* a través de un *Tópico* llamado '/cmd_vel', donde el Robot es otro nodo del tipo *Suscriptor* al mismo tópico.

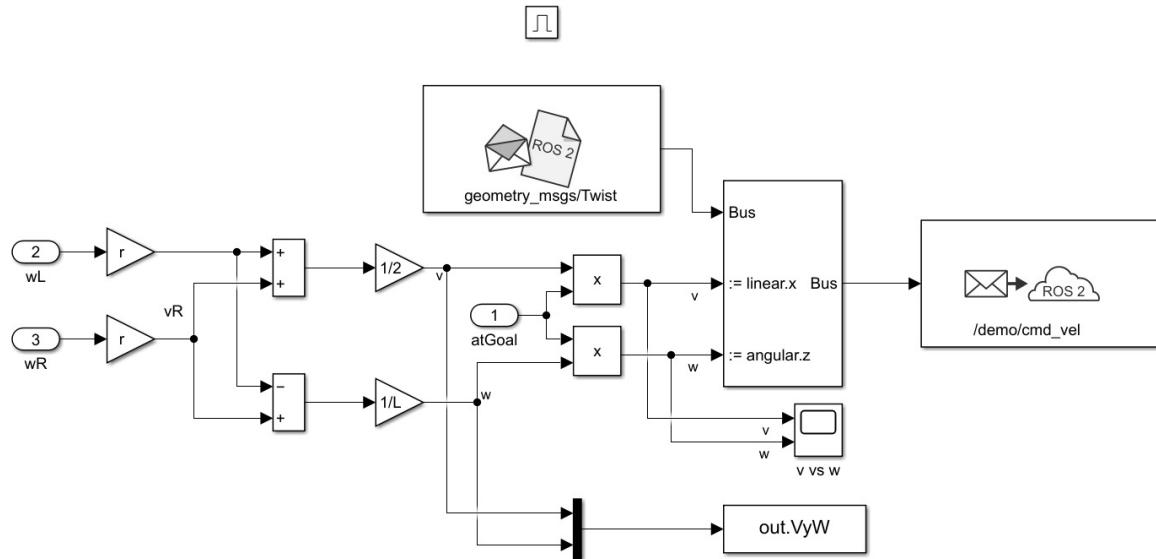


Figura 45: Bloques de salida de v y ω a ROS2. Fuente propia

7.5. Percepción

De la misma manera que se logró la comunicación entre las consignas del controlador y el robot en la simulación mediante ROS2, se obtienen los datos de los sensores seleccionados en el apartado 5.3 e implementados en el robot del mundo virtual simulado en el entorno Gazebo. En la figura 46 se pueden observar detalladamente cada uno de los

bloques y cada variable que se recibe con la información de cada uno de los sensores: Odometría(Encoder), DGPS (Robot y Base), IMU y LiDAR.

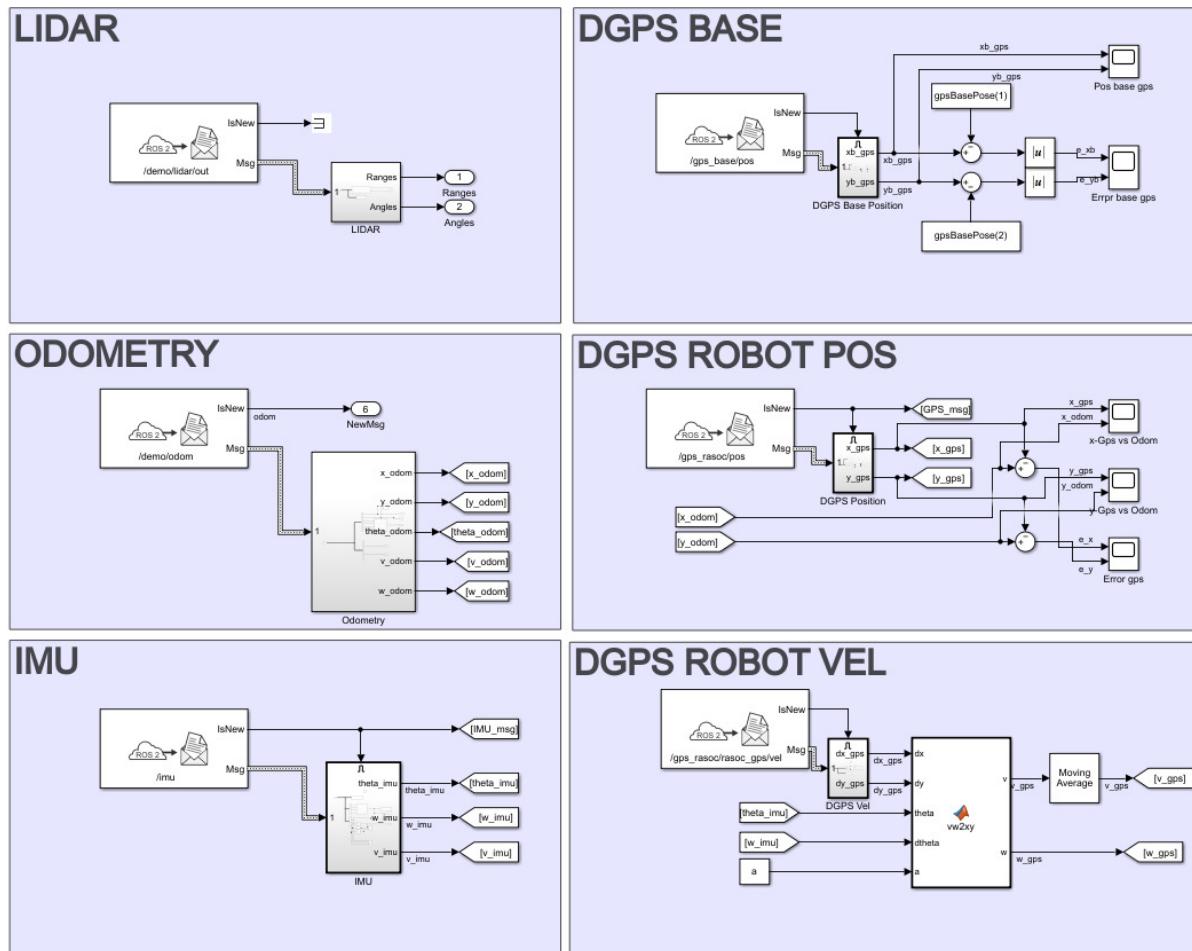


Figura 46: Bloques de salida de v y ω a ROS2. Fuente propia

Es importante destacar que los sensores agregados en el modelo del robot, se los configuro con parámetros de frecuencia de muestreo real según la información provista por los fabricantes de cada uno de los respectivos sensores seleccionados; así como también con ruido en la lectura a modo de asemejar lo más posible la simulación a la realidad, complejizando la adquisición de datos en condiciones ideales que suele darse en simulaciones computacionales.

7.5.1. Fusión de Sensores

Debido a la redundancia de sensores seleccionados e implementados para lograr una navegación autónoma óptima y eficiente, se debe implementar una técnica denominada *Fusión de Sensores*. Dicho metodología consiste en la combinación de la información proporcionada por cada uno de los sensores para reducir la incertidumbre que brinda cada uno por separado (puede darse por baja precisión, baja frecuencia de muestreo, bajo rechazo a perturbaciones, entre otras); logrando conseguir mayor precisión y fidelidad en la información adquirida respecto al sistema completo.

En el caso del robot asistente de cosecha, se optó por analizar ventajas y desventajas de cada uno de los sensores seleccionados para analizar e implementar una correcta fusión

de sensores. En la tabla 3 se detallan los aspectos analizados:

SENSOR	VENTAJAS	DESVENTAJAS
Encoder	Muy alta frecuencia de muestreo (100kHz)	Altamente sensible a perturbaciones dadas por el terreno
DGPS	Precisión submétrica	Muy baja frecuencia de muestreo (20Hz)
IMU	Muchas información relevante (vel, pos y acel angular). Alta precisión (0.1°)	Baja frecuencia de muestreo (200Hz) en comparación a encoder
LiDAR	Alta frecuencia de muestreo (5kHz). Gran utilidad en navegación local	Necesidad de algoritmo de localización y mapeo para referenciar al robot en el entorno (Ej: SLAM con filtro de partículas)

Tabla 3: Ventajas y Desventajas de Sensores. Fuente propia

Las variables que se buscan realimentar al controlador y a la planificación de movimiento son:

- Pose: $[x, y, \theta]$
- Velocidades angulares de cada rueda $[\omega_L, \omega_R]$ calculadas a partir de velocidad lineal v y angular ω del robot.

En la figura 47 se observa la lógica del algoritmo de fusión de sensores implementado en diagrama de bloques:

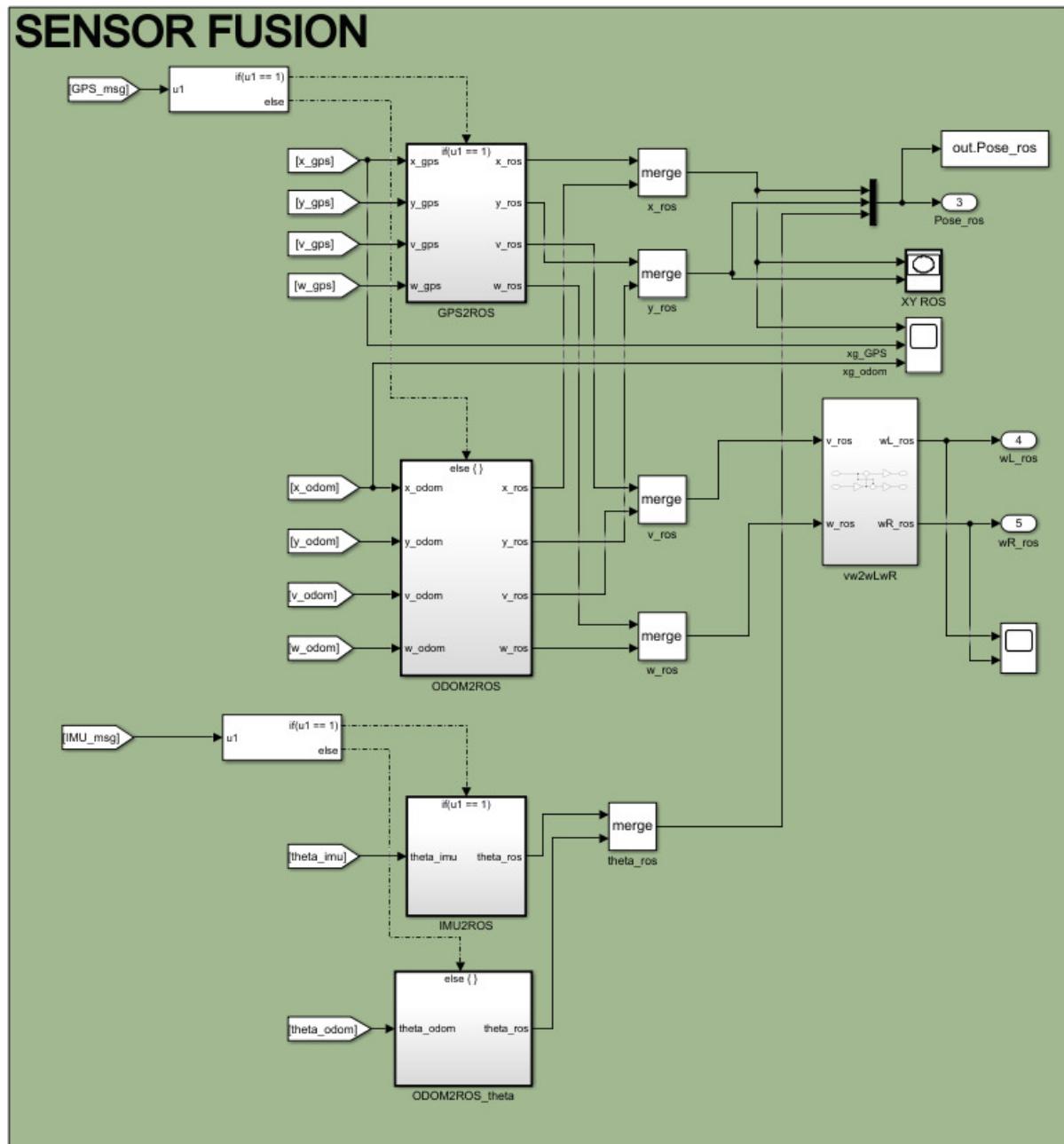


Figura 47: Fusión de sensores. Fuente propia

Debido a la elevada frecuencia de actualización del encoder, las variables de posición (x , y) se obtienen a partir del cálculo de odometría, pero considerando la acumulación de error en la lectura debido al terreno y la configuración de orugas del robot, se actualiza el valor de dichas variables cada vez que se obtienen datos del DGPS del robot, que cuenta con precisión submétrica pero baja frecuencia de muestreo; restableciendo así el valor de posición (x , y). La odometría también proporciona el dato del ángulo de orientación θ del robot respecto a la finca, y se lo fusiona con el obtenido de la IMU, asegurando la precisión del ángulo.

Por otro lado, los datos de v y ω se calculan (según 4.21) a partir de las velocidades del robot respecto a cada eje global de la finca (\dot{x} , \dot{y}) proporcionadas por el mismo DGPS; y la IMU brinda el dato de $\dot{\theta} = \omega$. De esta manera, solo queda obtener las velocidades angulares de cada rueda (ω_L , ω_R), las cuales se calculan según 4.1 y 4.20.

8. Perturbaciones

8.1. Cambio de Pendiente del Terreno

Con el objetivo de simular las variaciones de inclinación en el terreno, se inserta un valor de ángulo ‘ α ’ al Modelo Dinámico Directo (pero no al Inverso), para generar Torques de Pendiente que resten al torque entregado por los motores.

Vemos la variación de la pendiente, dónde se simula una pendiente positiva inicial, un plano en altura, una inclinación descendiente y nuevamente plano para finalizar:

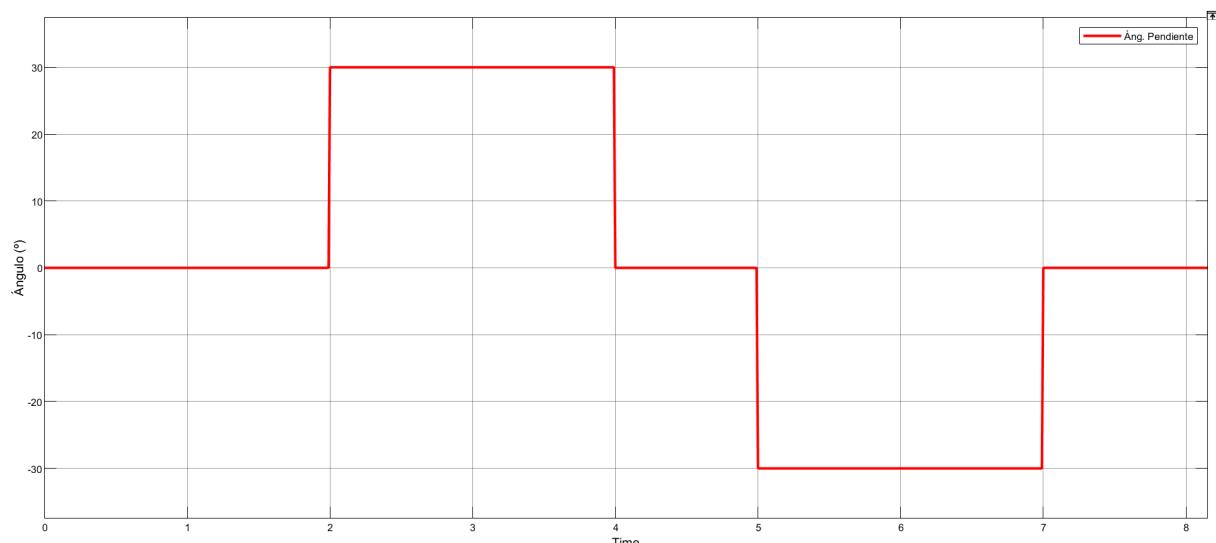


Figura 48: Perturbación (ángulo) en Inclinación de Terreno. Fuente propia

8.2. Masa Agregada

Para considerar en la simulación dinámica el efecto de la carga del canasto de uvas, se agrega una masa de 20kg considerando que dicho canasto se encuentra lleno y es por eso que el robot debe trasladarse al centro de recolección de la viña. Se implementa un sensor de masa que posee un ruido (Gaussiano) que simula el comportamiento inercial de la carga sobre el robot. La misma, se moverá a medida que el robot recorre el terreno irregular, generando variaciones abruptas en la medida. Con ello, el bloque del sensor de masa:

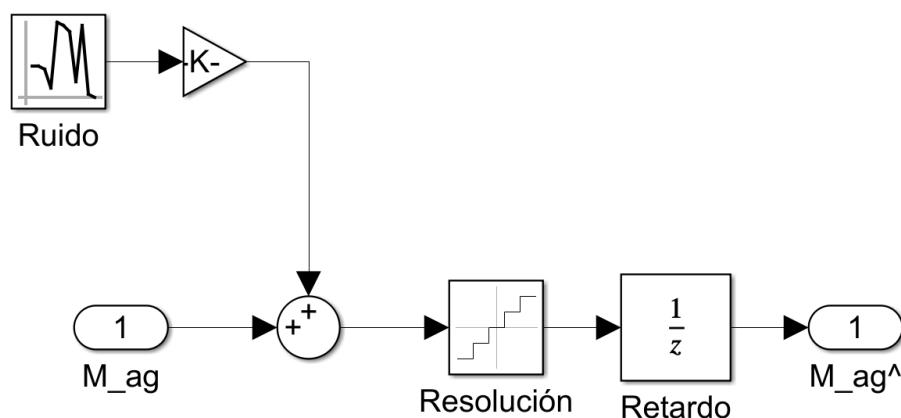


Figura 49: Bloque del Sensor de Masa. Fuente propia

El dato de la masa real es alimentado al Modelo Directo del sistema, pero al Controlador (modelo inverso) se alimenta al señal medida y luego filtrada. Se utiliza un filtro pasa bajo de constante de tiempo muy grande (2s), para remover el ruido de alta frecuencia de la medición. De manera análoga al filtro del Modulador de Tensión, podemos definir el actual:

$$G(s) = \frac{1}{1 + \tau s} \quad (8.1)$$

8.3. Efectos Disipativos por Fricción

De acuerdo a lo mencionado previamente, se simula un torque genérico de fricción viscosa (proporcional a la velocidad angular de cada rueda). Este coeficiente está calculado de manera que, cuando en ambas ruedas se produzca una velocidad de 8rad/s, la fuerza de fricción sea del 30% de la nominal. Véase la figura 33 para notar como esto está realimentado al Modelo Directo pero no al Inverso.

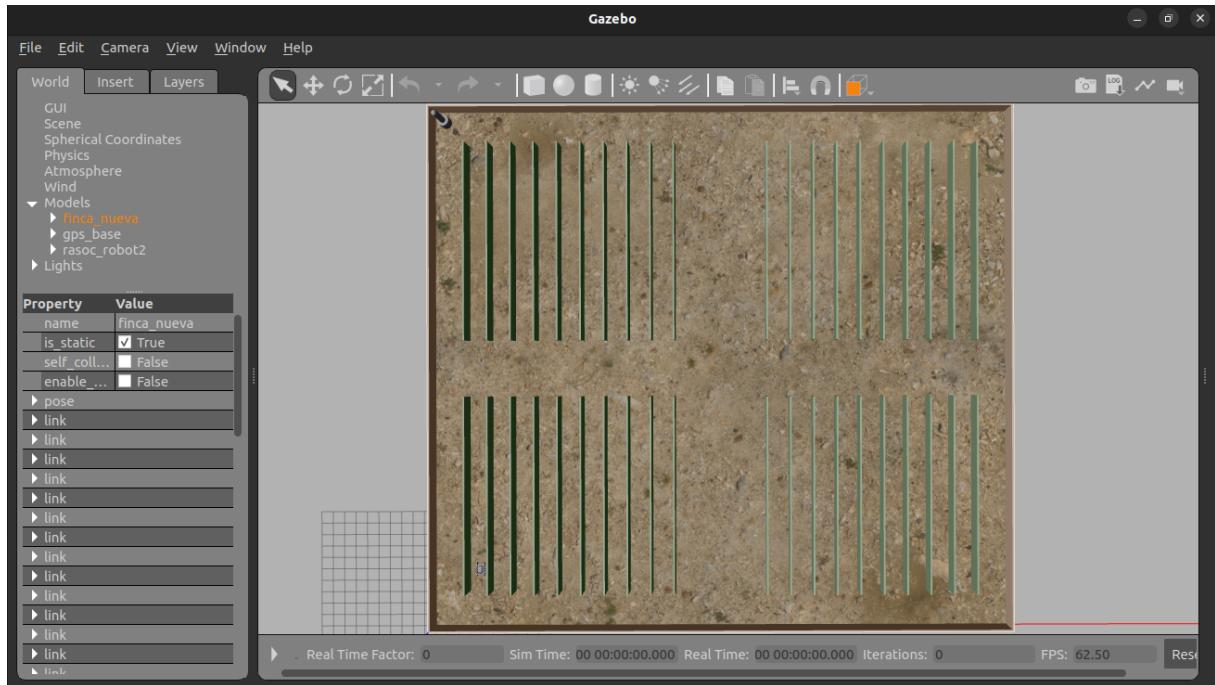
9. Resultados

En esta sección se presentan la gráficas y curvas obtenidas al ejecutar las simulaciones de las diferentes trayectorias planteadas y así evaluar el funcionamiento de todas las etapas del algoritmo mencionado en las secciones anteriores. Se plantean 3 casos diferentes considerando coordenadas de inicio y finales lejanas entre sí, contemplando el cajón a trasladar completamente lleno y pendientes elevadas, con el fin de evaluar el funcionamiento del sistema en las condiciones más desfavorables y/o complejas posibles.

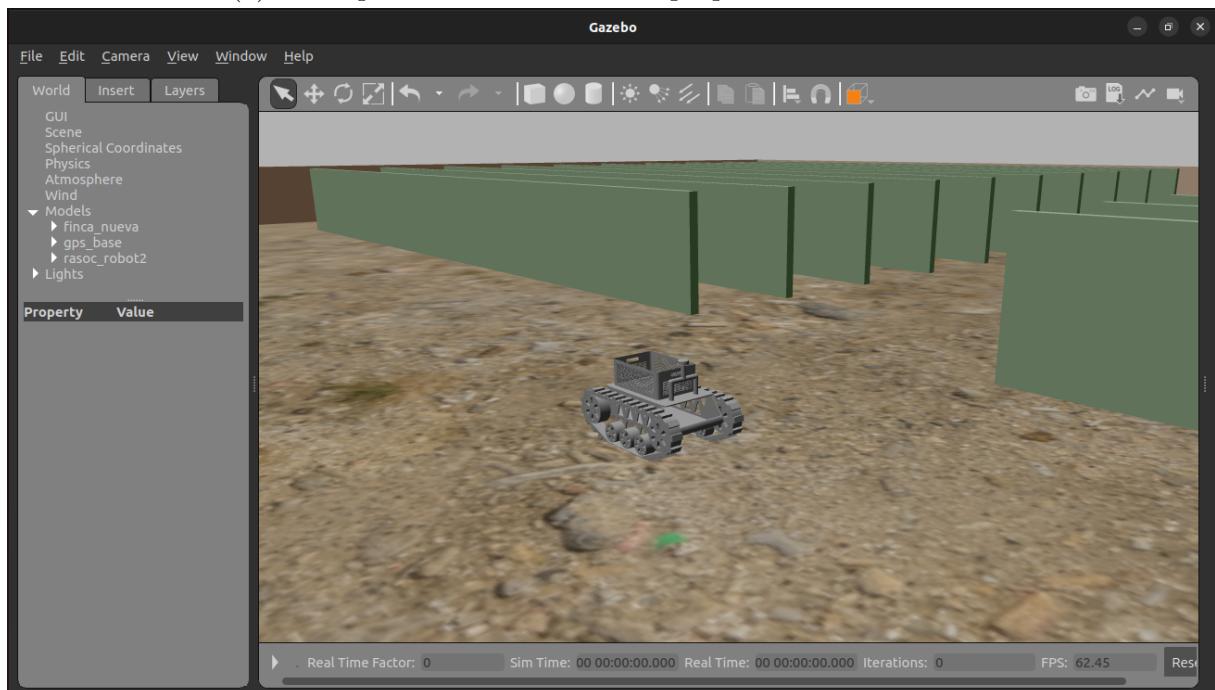
Es importante aclarar que los tiempos de simulación que se observan en cada una de las gráficas temporales obtenidas de Simulink, no corresponden al tiempo real de simulación del entorno Gazebo. Esto sucede ya que, al ejecutar en tiempo real dos simuladores independientes entre sí, pero comunicados mediante la red generada con ROS2, no fue posible sincronizar los tiempos de ejecución de cada uno de los softwares. Por lo tanto, luego de múltiples ejecuciones, se estimó que 1 segundo de simulación en Simulink corresponden a aproximadamente 13 segundos de simulación en Gazebo, entorno en el cual sí se apreciaba una simulación en tiempo real.

En las figuras 50a y 50b se observa el mundo virtual creado en Gazebo a partir del diseñado en SketchUp, manteniendo las dimensiones, y el robot integrado en dicho mundo. Debido a las limitaciones de diseño propias del entorno Gazebo, se simplificaron las viñas como paredes, de forma que el sensor LiDAR del robot las pueda detectar mientras se desarrolla la simulación y las contemple como obstáculos en la navegación local.

Se debe considerar que en todas las gráficas que se comparan parámetros ‘consigna’ vs ‘reales’, los primeros serán de color negro, para diferenciarlos fácilmente.



(a) Finca para simulación. Fuente propia



(b) Robot en finca simulada. Fuente propia

Figura 50: Mundo virtual desarrollado en Gazebo

Las trayectorias a simular corresponden a los siguientes vectores de ‘Pose’ [x y θ] iniciales y finales; siendo (x,y) las coordenadas planares de posición del robot en el mapa, en metros, y θ el ángulo de orientación del robot respecto al semi-eje positivo del eje X global, en radianes positivos en sentido anti-horario:

9.1. Primera trayectoria

- **Pose inicial:** [4 5 90°]
- **Pose final:** [47 43,5 0°]

En la figura 51 se puede observar la trayectoria planificada por el algoritmo, subfigura 51a, y a su lado la trayectoria realizada por el robot al ejecutar la simulación, figura 51b. Se destaca la correcta ejecución y seguimiento de cada uno de los diferentes ‘waypoints’ durante todo el recorrido de traslación del robot, evitando todos los obstáculos estáticos presentes en la finca y logrando alcanzar la posición final.

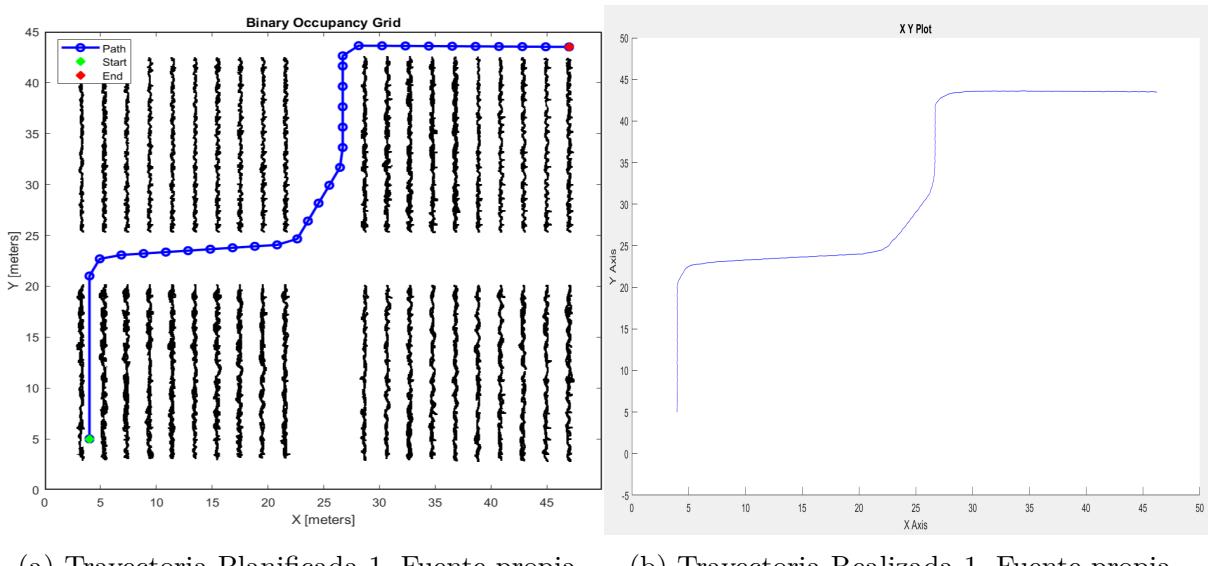


Figura 51: Primera Trayectoria

Continuando con el análisis de las curvas de interés obtenidas de la simulación de la trayectoria mencionada, se obtiene la gráfica 52, donde se compara la velocidad lineal consigna o referencia, del robot, que ingresa al controlador; con la velocidad lineal efectiva medida desde la simulación en Gazebo y la comunicación mediante ROS2. Se analiza que se satisface la rampa de aceleración ingresada y que no supera la velocidad límite máxima impuesta de 1m/s; así como también el ruido ingresado debido a la lectura de los valores de velocidad en tiempo real, configurados en Gazebo con ruido Gaussiano para asemejar lo más posible dicha simulación a una situación real.

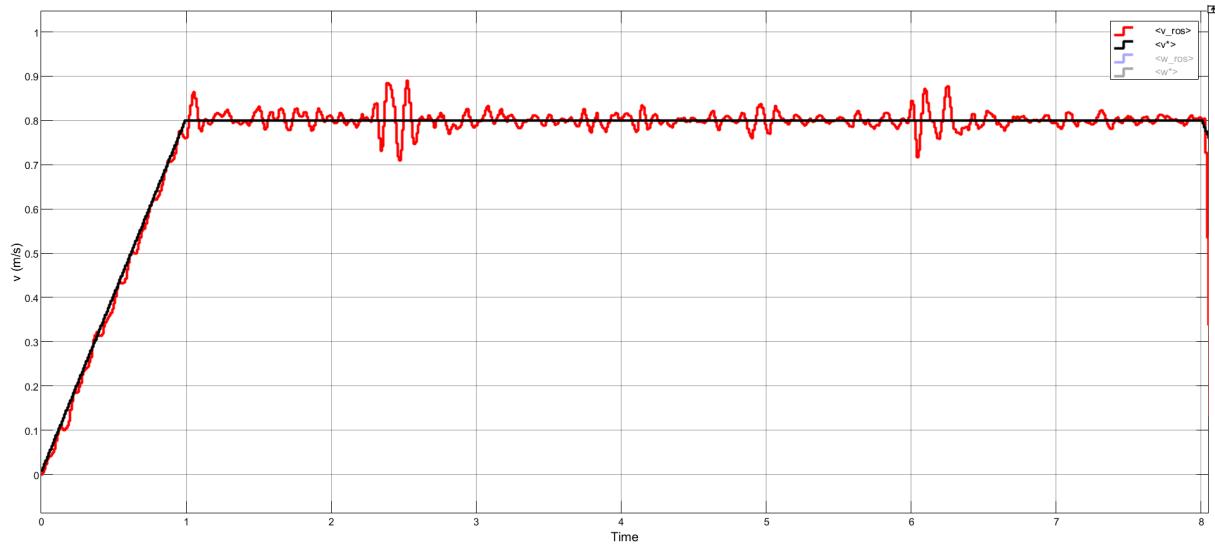


Figura 52: Velocidad Lineal v - Trayectoria 1. Fuente propia

En la figura 53, se compara el comportamiento de la velocidad angular consigna del robot con la real o efectiva durante todo el movimiento y, tal como se mencionó en el análisis anterior, se destaca notablemente el ruido dado por las lecturas de los sensores. Además, se visualiza con facilidad que, dada la configuración de la tracción diferencial, el robot no avanza siguiendo una velocidad angular constante, o nula al desplazarse en línea recta, sino que lo logra gracias al movimiento de ‘cabeceo’. Esto quiere decir que, logra el avance en función de la diferencia de las velocidades angulares (o lineales) de cada una de las ruedas, por ende de cada uno de los motores. Esto se puede observar muy levemente en la figura 51b, donde en los sectores de movimiento en línea recta, el robot realiza un movimiento bascular muy pequeño.

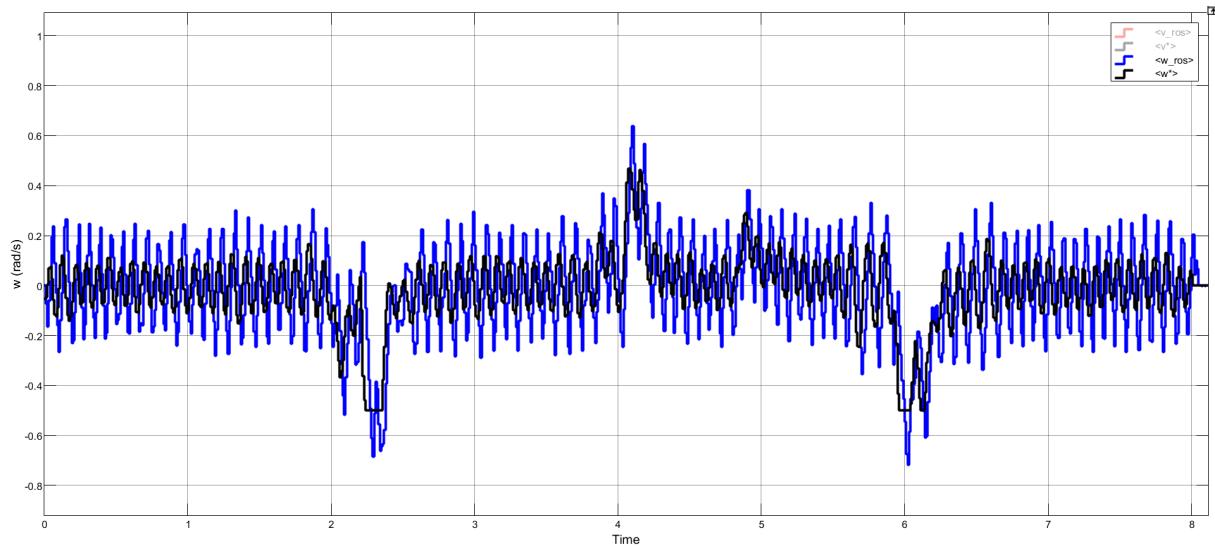


Figura 53: Velocidad Angular w - Trayectoria 1. Fuente propia

Casi finalizando con el análisis de curvas de la primera trayectoria, se visualizan en las figuras 54 y 55 los torques consigna y efectivos, en la rueda izquierda y en la derecha, respectivamente. Los valores medidos y expuestos en cada una de las curvas, corresponden a los torques a la salida de la caja reductora, es decir, están afectados por la relación

de transmisión de dicha reducción seleccionada. Observando con atención los torques efectivos de cada rueda, se observa, en ambos casos, la saturación máxima y mínima impuesta en el controlador a modo de proteger los motores.

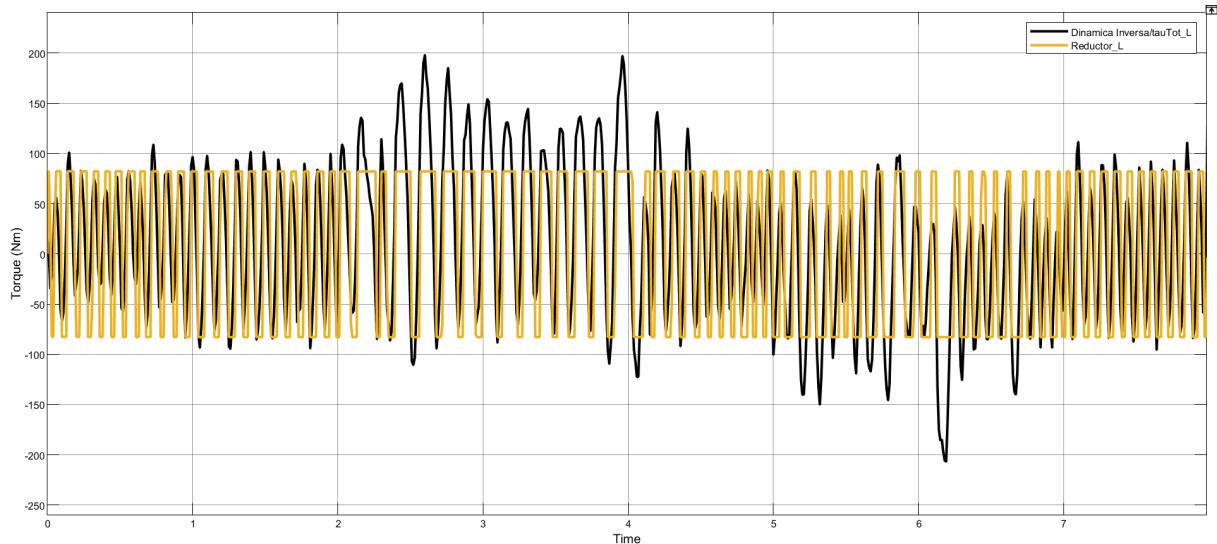


Figura 54: Torque Rueda Izquierda - Trayectoria 1. Fuente propia

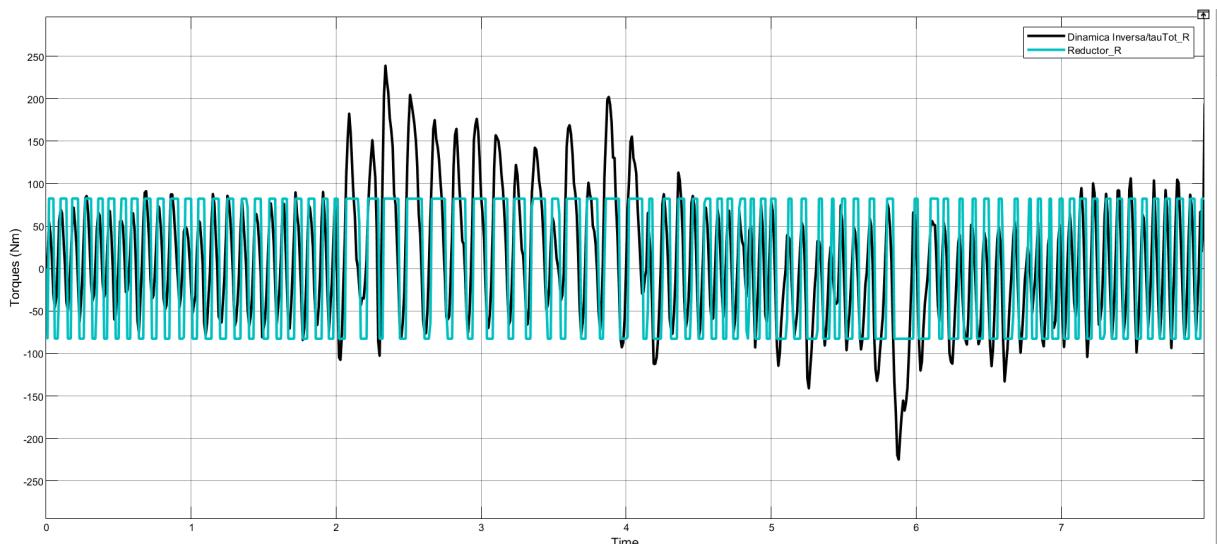


Figura 55: Torque Rueda Derecha - Trayectoria 1. Fuente propia

Por último, contemplando lo detallado al inicio de la presente sección sobre los tiempos de simulación de cada uno de los entornos implementados, se observa que las curvas obtenidas de Simulink en esta primer trayectoria muestran un tiempo de ejecución de aproximadamente 8 segundos, lo que implica que el tiempo real fue de 104 segundos (tiempo = 8s * 13), es decir, el robot realizó la navegación autónoma entre las posiciones inicial y final en 01:44 minutos.

Considerando una distancia recorrida de aproximadamente 80m, vemos que la velocidad promedio de movimiento es de:

$$V_{\text{prom}} = \frac{80\text{m}}{104\text{s}} = 0,769 \frac{\text{m}}{\text{s}} \quad (9.1)$$

Dicho resultado (consistente con la figura 52) muestra que se cumple con un correcto funcionamiento del robot durante esta trayectoria, aún considerando las diferencias en tiempo de simulación entre ambos software. Esto demuestra la robustez del diseño y los algoritmos implementados.

9.2. Segunda trayectoria

- **Pose inicial:** [4 5 -90°]
- **Pose final:** [47 43,5 0°]

Continuando con los resultados de las simulaciones, se procede a exponer y analizar las gráficas de la segunda trayectoria simulada. Dados los vectores de pose indicados, se tiene en cuenta que las coordenadas de inicio y fin son las mismas que los correspondientes a la primer trayectoria, pero la diferencia radica en que la orientación inicial se modifica 180°, es decir, en esta segunda trayectoria el robot inicia con una orientación de -90° respecto al semi-eje positivo del eje X global del mapa.

De la misma manera que se mencionó en la trayectoria anterior, en la figura 56 se observan las subfiguras 56a y 56b correspondientes a la segunda trayectoria planificada y a la segunda trayectoria realizada, respectivamente, donde el robot describe una trayectoria prácticamente idéntica a la dada por los ‘waypoints’ de la planificación.

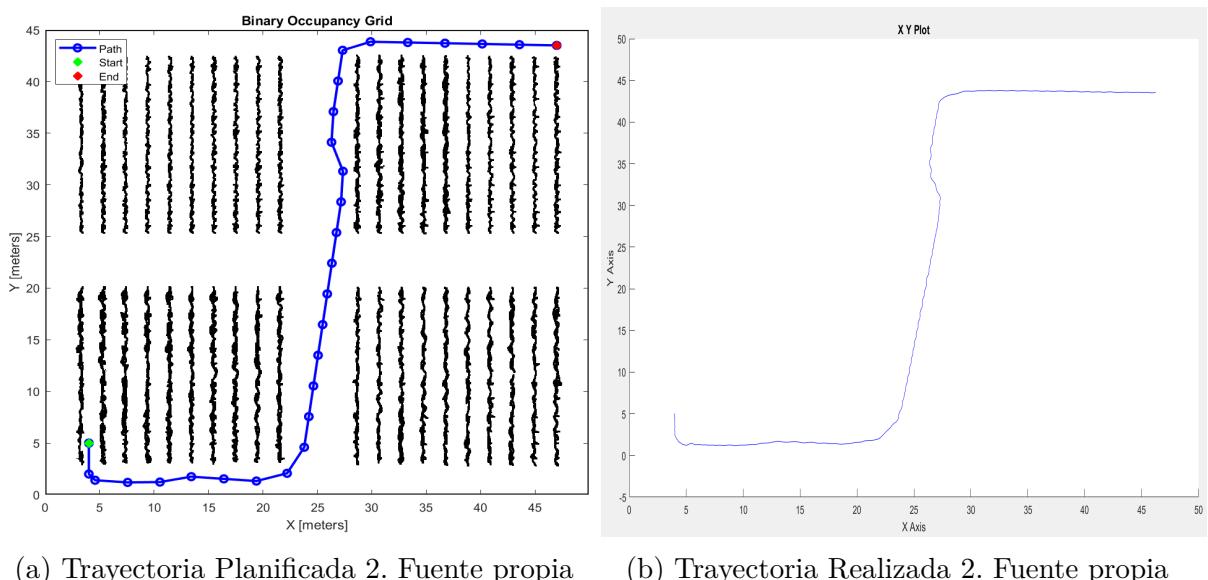
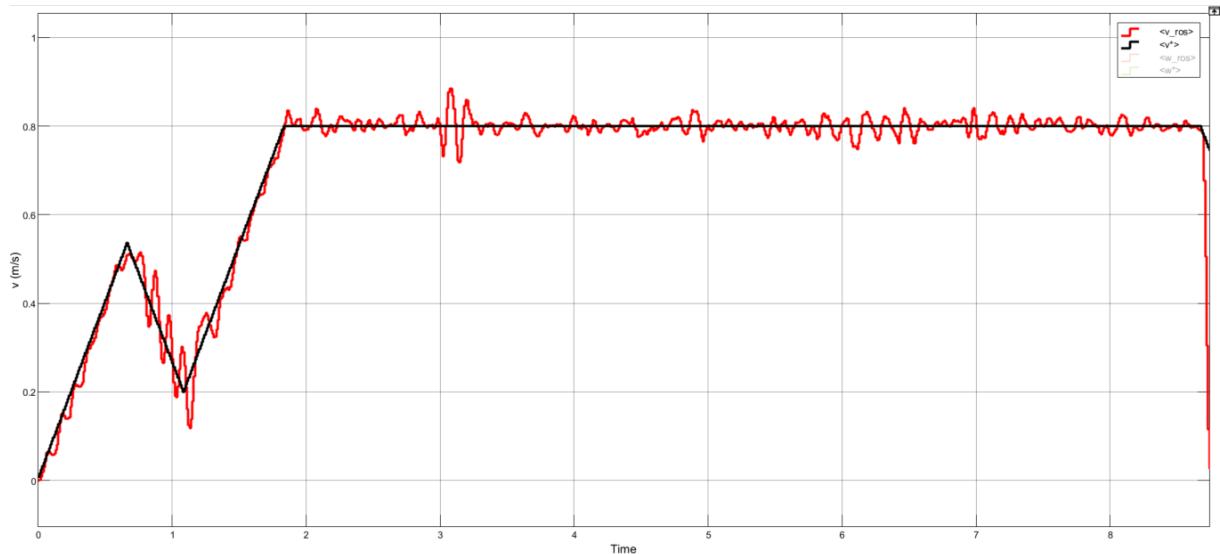
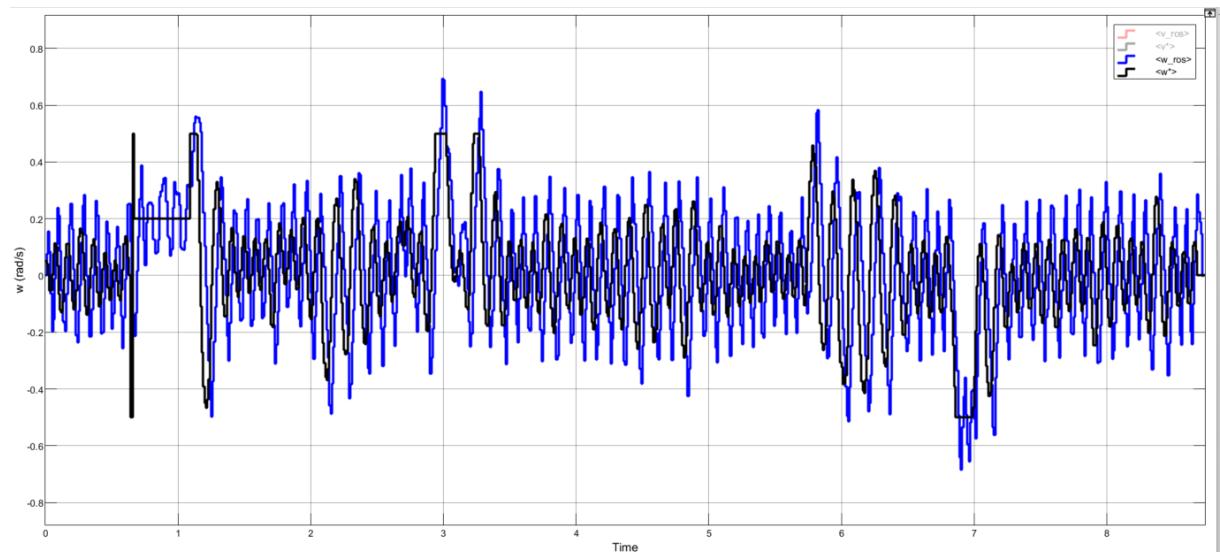


Figura 56: Segunda Trayectoria

En las figuras 57 y 58, donde se grafican la comparación de velocidad lineal y velocidad angular consignas con reales, se observan comportamientos y respuestas similares a las mencionadas en el apartado anterior, con la diferencia en algunos tramos debido a la diferencia de la trayectoria descripta por el robot, especialmente al inicio de la navegación.

Figura 57: Velocidad Lineal v - Trayectoria 2. Fuente propiaFigura 58: Velocidad Angular ω - Trayectoria 2. Fuente propia

De forma similar a las curvas de torques de la primer trayectoria, se observan en cada una de las figuras 59 y 60 los torques consigna y efectivos de las ruedas izquierda y derecha, respectivamente; y se contemplan los mismos análisis mencionados en la trayectoria anterior.

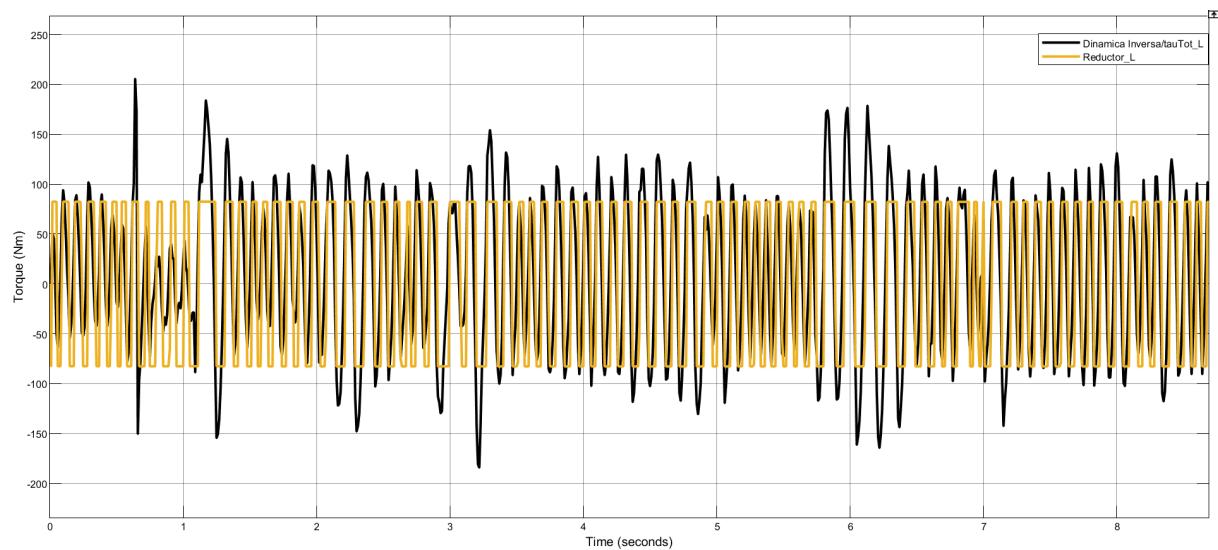


Figura 59: Torque Rueda Izquierda - Trayectoria 2. Fuente propia

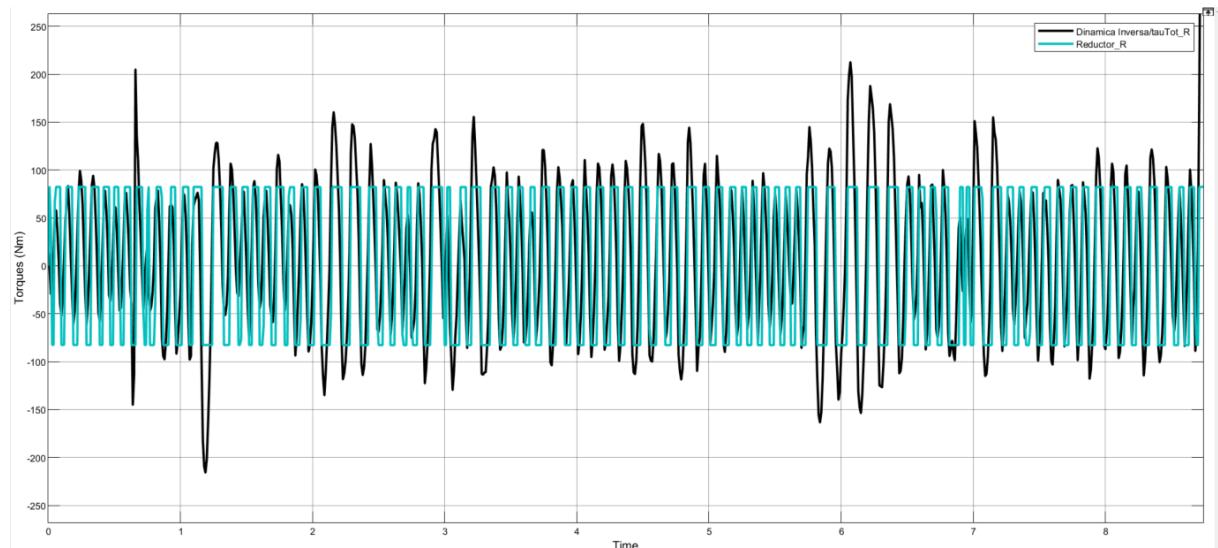


Figura 60: Torque Rueda Derecha - Trayectoria 2. Fuente propia

Finalizando con las gráficas de la presente trayectoria, se destaca que el tiempo de simulación es ligeramente mayor que el obtenido en la trayectoria anterior (13 segundos reales aproximadamente), lo que indica un tiempo aproximado de 2 minutos en efectuar satisfactoriamente, a velocidad máxima, la navegación autónoma de la segunda trayectoria.

9.3. Tercera trayectoria

- **Pose inicial:** [47 43,5 0°]
- **Pose final:** [1,5 5 90°]

Como tercera y última trayectoria de simulación, se exponen los resultados de las gráficas obtenidas al simular la trayectoria entre los puntos inicial y final dados en los

vectores de Pose indicados. En esta tercera trayectoria, se busca simular y analizar un comportamiento diferente respecto al descripto por las trayectorias anteriores, y para ellos se impone la situación en la que el robot regresa con un cajón vacío hacia una nueva posición del operario, para continuar con el proceso de recolección de uva.

En las figuras 61 se observan las subfiguras 61a y 61b correspondientes a la trayectoria planificada y a la realizada, respectivamente, en este tercer caso de simulación. Se observa que la posición y orientación inicial del robot es la misma que las finales de la primer trayectoria y que, las coordenadas de posición final corresponden a la hilera de vid contigua a la del inicio de la primera trayectoria; y con una orientación tal que el robot se posiciona listo para retomar el modo seguidor de operario durante la cosecha de dicha hilera.

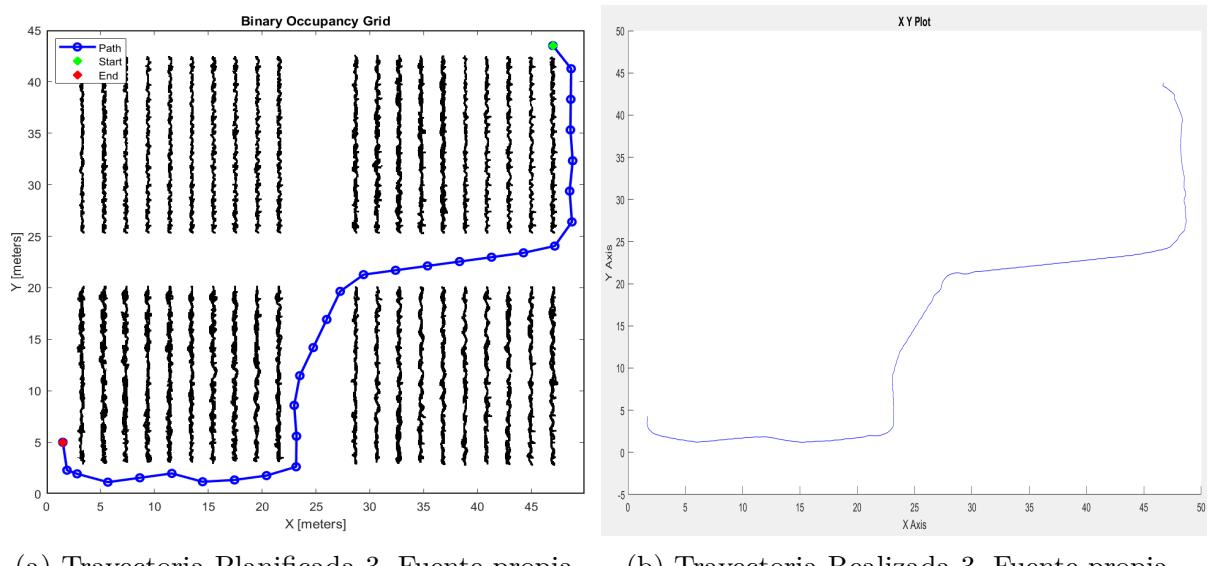


Figura 61: Tercera Trayectoria

De la misma manera que se mencionó en las trayectorias expuestas anteriormente, se observan en las figuras 62 y 63 las curvas de comparación de velocidad lineal y velocidad angular consignas con ambas velocidades reales, respectivamente.

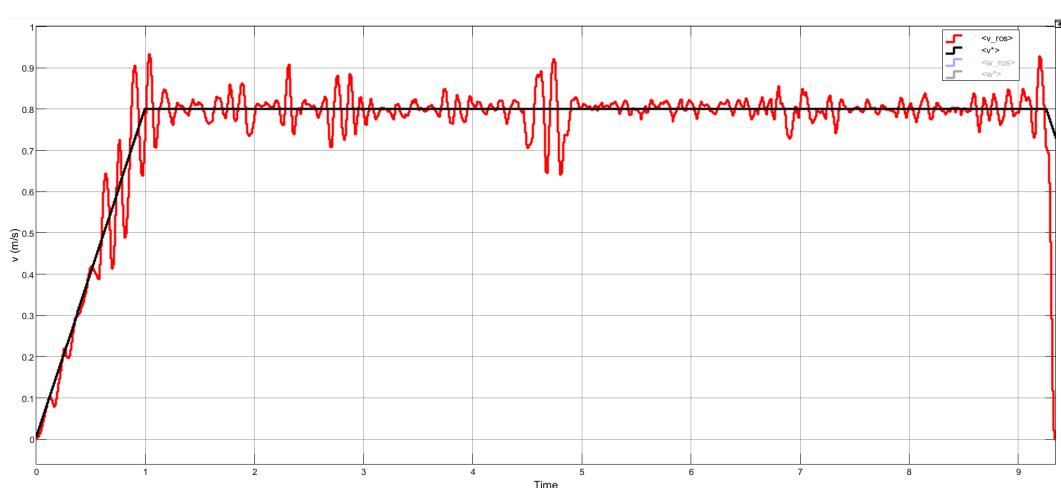
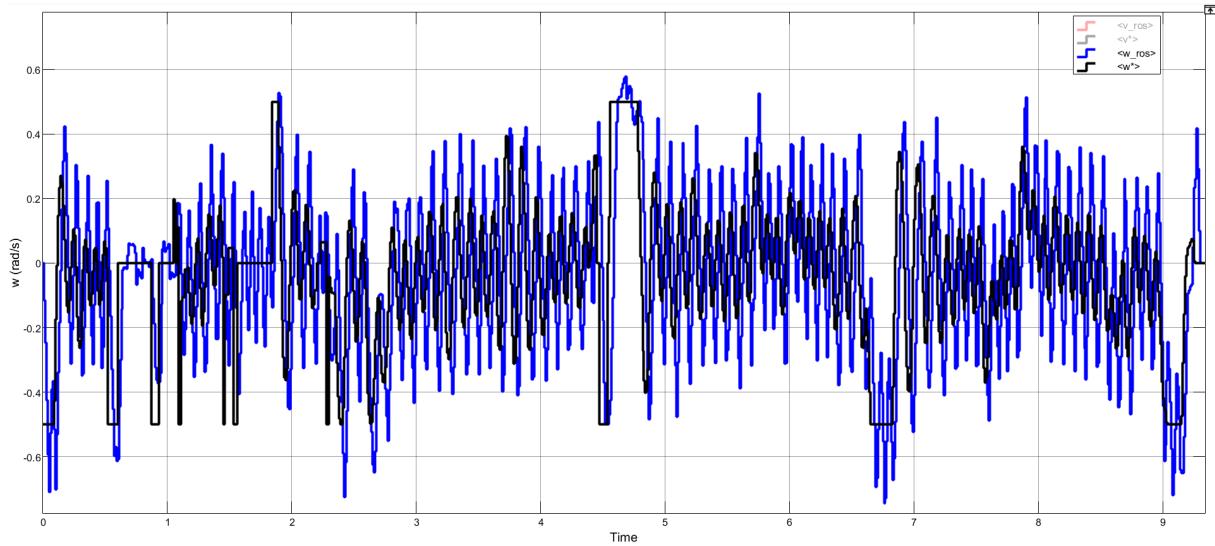


Figura 62: Velocidad Lineal v - Trayectoria 3. Fuente propia

Figura 63: Velocidad Angular ω - Trayectoria 3. Fuente propia

En el caso de las curvas de torque consigna y efectivo de la rueda izquierda, figura 64, y de la rueda derecha, figura 65, se observan comportamientos muy similares a los analizados en las simulaciones de la primer y segunda trayectoria.

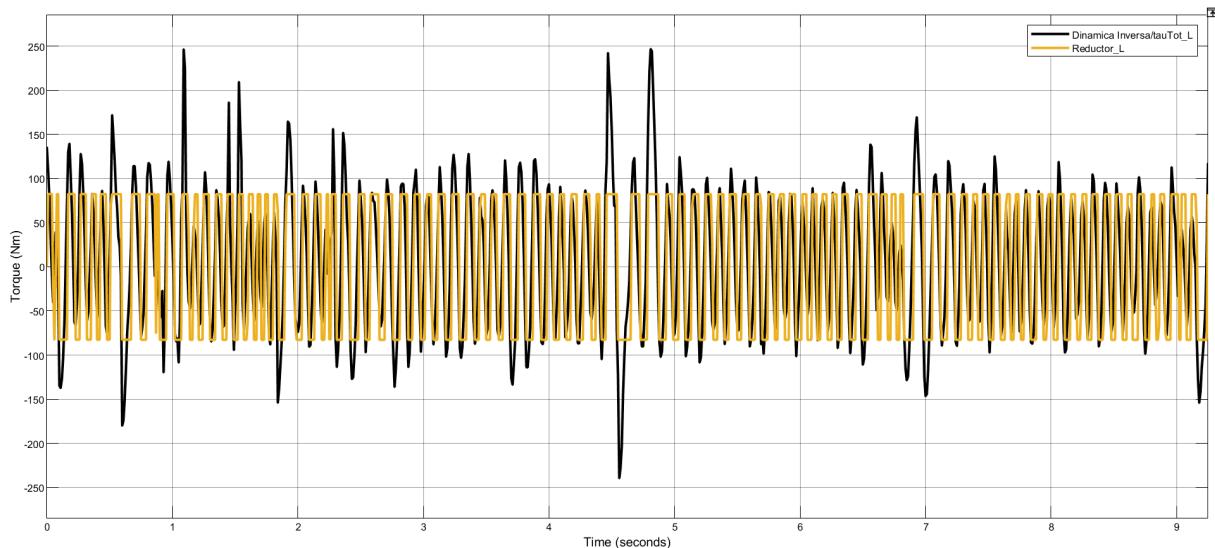


Figura 64: Torque Rueda Izquierda - Trayectoria 3. Fuente propia

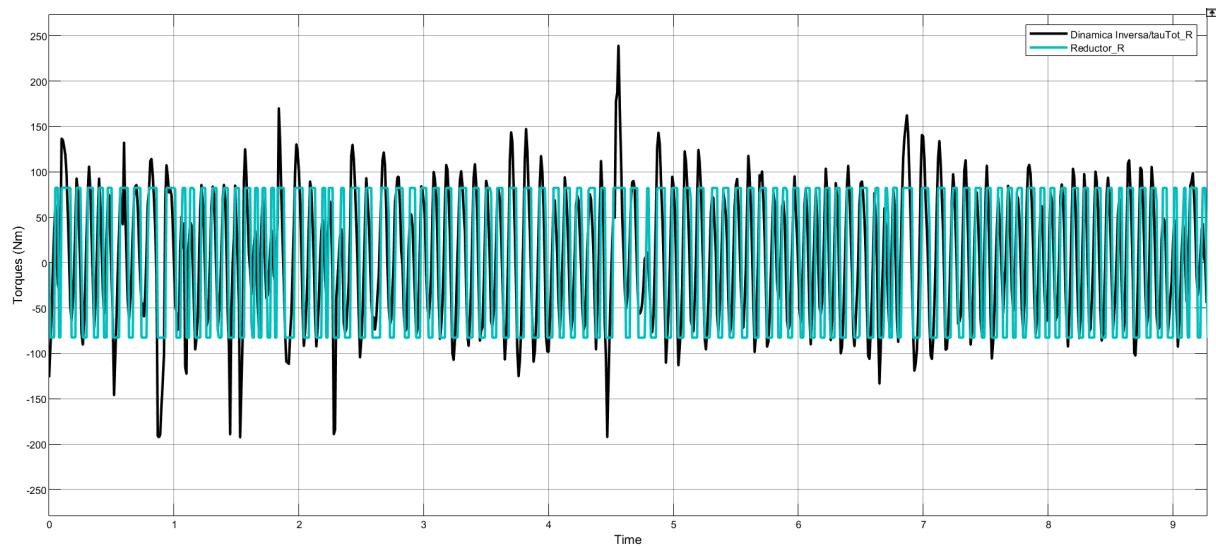


Figura 65: Torque Rueda Derecha - Trayectoria 3. Fuente propia

Por último, y de forma análoga a los análisis previos, se observa que el tiempo de ejecución de la tercer trayectoria es poco mayor a las trayectorias anteriores y esto es debido a q la distancia a recorrer es uno metros mayor. Observando las gráficas de esta última trayectoria simulada, el tiempo en Simulink es de aproximadamente 9,3 segundos, lo que equivale a un tiempo real de 121 segundos ($9,3\text{s} * 13 = 120,9\text{s}$), es decir, 2 minutos y 1 segundo (02:01 min); valor de tiempo por demás aceptable.

9.4. Trayectoria con obstáculos dinámicos

Se simula nuevamente la trayectoria 9.2 con los mismos vectores de Pose inicial y final pero con la particularidad de que, durante la simulación en tiempo real, se agregan obstáculos en sectores del mapa donde debe pasar el robot al seguir la trayectoria planificada. Esto se logra ya que el simulador Gazebo permite agregar objetos durante la ejecución, por lo tanto, se agregan 2 cubos de 1 metro de lado cada uno, en diferentes posiciones del mapa.

Considerando que la trayectoria global planificada es la misma que la de la figura 56a, se observa en la figura 66, en color rojo, la posición de los objetos integrados en el mundo virtual durante la ejecución de la simulación y, en la figura 67, la trayectoria descripta por el robot luego de finalizar el movimiento de traslación y eludir satisfactoriamente los obstáculos:

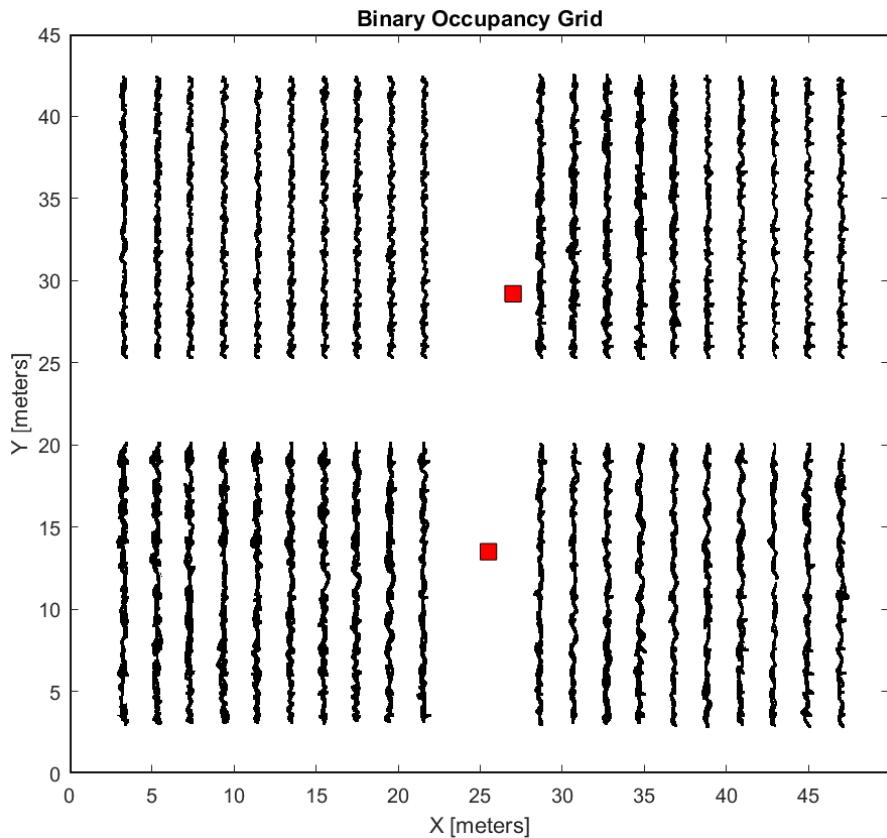


Figura 66: Mapa binario de finca con obstáculos. Fuente propia

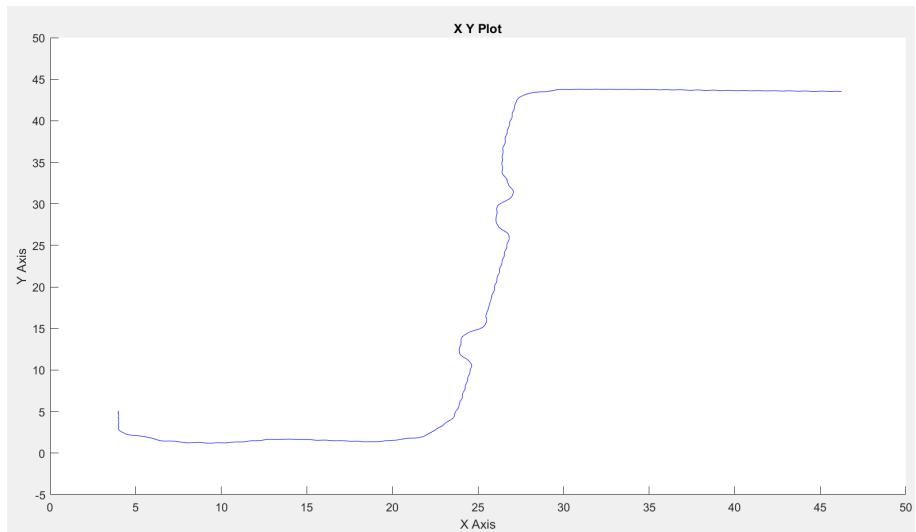


Figura 67: Trayectoria con obstáculos realizada. Fuente propia

10. Costos de Fabricación

En el siguiente cuadro, se observa el presupuesto estimado para la construcción efectiva del robot. Se tuvieron en cuenta los datos y costos expresados por los fabricantes, y se estimaron otros de acuerdo a investigaciones de mercado (como la Electrónica y Control, y la Estructura Mecánica).

COSTOS DE FABRICACIÓN				
Producto	Cantidad	Precio unitario	Precio	
Encoder AMT 103-V	2	US\$ 23,62	US\$ 47,24	
Kit DGPS TOP682-6043	2	US\$ 615,00	US\$ 1.230,00	
LiDAR kit D500	1	US\$ 120,99	US\$ 120,99	
IMU WT901BLEVL	1	US\$ 17,00	US\$ 17,00	
Kit celda de carga AD-HX711 + 4 sensores	1	US\$ 4,33	US\$ 4,33	
Sensor ultrasónico XX630A1KAM12	6	US\$ 290,00	US\$ 1.740,00	
Motor DC Brushless BM1418HQF	2	US\$ 71,60	US\$ 143,20	
Controlador BLDC	2	US\$ 69,00	US\$ 138,00	
Caja Reductora iHF	2	US\$ 192,00	US\$ 384,00	
Orugas DP-PY-148	1	US\$ 1.300,00	US\$ 1.300,00	
Batería UE-12Li50BL	1	US\$ 248,39	US\$ 248,39	
Electrónica y control	1	US\$ 500,00	US\$ 500,00	
Estructura mecánica	1	US\$ 500,00	US\$ 500,00	
Ingeniería y desarrollo	1	US\$ 5.000,00	US\$ 5.000,00	
TOTAL			US\$11.373,15	

Figura 68: Costos totales del fabricación del robot RASOC. Fuente propia

11. Conclusiones

En el presente trabajo se diseñó, analizó, modeló y simuló dinámica y cinemáticamente el robot móvil asistente de operarios de cosecha propuesto (RASOC), satisfactoriamente; haciendo especial énfasis en su funcionamiento con navegación autónoma.

Se implementaron exitosamente algoritmos de Inteligencia Artificial para la planificación de trayectoria y movimiento, observándose la generación de un camino válido para cada trayectoria, y la correcta navegación del robot, demostrable al comparar las gráficas de caminos generados y recorridos.

La simulación cinemática resultó altamente útil para visualizar y analizar el movimiento del robot en su espacio de trabajo, contemplando sus características físicas, y las restricciones del entorno.

Se logró desarrollar satisfactoriamente un algoritmo de control dinámico que responde bien al seguimiento de distintas trayectorias, y además resulta muy robusto frente a diferentes perturbaciones. Si consideramos, como la peor de ellas, los retardos de comunicación entre los softwares utilizados y sus ejecuciones asíncronas (resultado inevitable de la simulación computacional); podemos asegurar un desempeño estable del sistema en una aplicación real con tiempos de procesamiento mucho más rápidos que los estudiados. Esto se evidencia al analizar las curvas obtenidas de velocidad lineal, angular y torque en cada rueda motriz.

Se obtuvo un modelo completo y cercano a la realidad, y se evidencia la importancia de seleccionar correctamente el hardware a utilizar y el software que se incorporará al mismo, aplicando una matriz de decisión como estrategia útil para tal fin. Sin embargo, se evidenciaron posibles mejoras a realizar como desarrollos futuros, tales como:

- Se propone un estudio más profundo de las perturbaciones para asegurar el comportamiento del sistema ante posibles alteraciones de terreno, carga, etc.
- Se deja abierta la posibilidad de modelar el movimiento vertical de robot (vibraciones en Z), ya que estas permitirán una mejor estimación de la carga que se transporta, e inclusive, podría realizarse un relevamiento de las condiciones del terreno.
- Sería conveniente estudiar la posibilidad de implementar un algoritmo de SLAM (Mapeo y Localización Simultánea), para realizar el reconocimiento de la viña con el mismo RASOC, en lugar de necesitar una foto aérea.
- Es posible agregar la dinámica completa de los sensores y actuadores, al modelo del sistema, para corregir las inestabilidades que estos causan.

En este último párrafo, y a modo de reflexión personal como autores, nos permitimos cambiar la redacción a primera persona, ya que nos parece importante recordar que el presente trabajo comenzó hace años, dentro de los proyectos integradores de diversas materias. Fue durante ellos, que este robot fue tomando forma de Proyecto Final de Estudios, y por ello creemos que es una representación fiel del proceso de aprendizaje que transitamos a lo largo de la carrera de Ingeniería Mecatrónica, plasmándose en un desarrollo integral, englobando conceptos estudiados en muchas cátedras distintas ([4], [5], [6] y [8]). Esto sentó las bases para la investigación necesaria para completar el desarrollo. Es por esto, que consideramos que es un resultado muy satisfactorio y que deja planteada toda la ingeniería general para el desarrollo del robot, y creemos que es una posibilidad que puede ser implementada en un futuro, logrando ser un producto comercializable que cumpla con la tarea para la cuál fue diseñado.

Bibliografía y Referencias

- [1] IEC (International Electrotechnical Commission). *Clasificación de Protección de Ingresos (IP)*. URL: <https://www.iec.ch/ip-ratings>.
- [2] CUI Devices. *Sensor Codificador Incremental*. URL: <https://ar.mouser.com/ProductDetail/CUI-Devices/AMT103-V?qs=Wyj1AZoYn51X2GCrrvGQTg%3D%3D>.
- [3] Open Source Robotics Foundation. *Documentación de Gazebo*. URL: https://classic.gazebosim.org/tutorials?tut=install_ubuntu.
- [4] Dr. Ing. Rodrigo Gonzalez. *Material de cátedra y apuntes de clase - Control y Sistemas*.
- [5] Ing. Roberto Haarth. *Material de cátedra y apuntes de clase - Robótica II*.
- [6] Mgtr. Ing. Eduardo Iriarte. *Material de cátedra y apuntes de clase - Microcontroladores y Electrónica de Potencia*.
- [7] Felipe Jeon. *Algoritmo de Planificación A* Híbrido*. URL: <https://medium.com/@junbs95/gentle-introduction-to-hybrid-a-star-9ce93c0d7869>.
- [8] Ing. Gabriel L. Julián. *Material de cátedra y apuntes de clase - Automática y Máquinas Eléctricas*.
- [9] LDROBOT. *Sensor LiDAR*. URL: https://www.ldrobot.com/ProductDetails?sensor_name=D500+Kit.
- [10] Shenzhen Hefa Gear Machinery Co. Ltd. *Caja reductora planetaria*. URL: <https://cnhfgeart.en.made-in-china.com/product/wNpnXfclkZkB/China-Best-Price-Prf90-Series-Planetary-Gearbox-for-750W-Servo-Motor.html>.
- [11] Shanghai Puyi Industrial Co. Ltd. *Tren de orugas*. URL: https://es.made-in-china.com/co_puyitracks/.
- [12] MathWorks. *Comunicación entre Simulink y Gazebo a través de ROS2*. URL: <https://la.mathworks.com/help/ros/ug/connect-to-a-ros-enabled-robot-from-simulink-over-ros2.html>.
- [13] MathWorks. *Documentación de MATLAB*. URL: https://la.mathworks.com/help/matlab/index.html?s_tid=hc_panel.
- [14] MathWorks. *Documentación de Simulink*. URL: https://la.mathworks.com/help/simulink/index.html?s_tid=hc_panel.
- [15] MathWorks. *Histograma de campo vectorial*. URL: <https://la.mathworks.com/help/nav/ug/vector-field-histograms.html>.
- [16] MathWorks. *ROS Toolbox*. URL: <https://la.mathworks.com/products/ros.html>.
- [17] Prof. Larry Francis Obando. *Función de transferencia del Motor DC y su carga*. URL: <https://dademuchconnection.wordpress.com/2019/04/24/funcion-de-transferencia-del-motor-dc-y-su-carga/>.
- [18] ROS. *Documentación de ROS2*. URL: <https://docs.ros.org/en/humble/>.
- [19] AMG Power Solutions. *Controlador para motor BLDC*. URL: <https://es.aliexpress.com/item/4000006303860.html>.
- [20] Tuoxiang Store. *Motor Brushless DC*. URL: <https://es.aliexpress.com/item/1005007222602874.html>.

- [21] Si Tai&SH. *Módulo celda de carga.* URL: <https://es.aliexpress.com/item/1005005455218016.html>.
- [22] Telemecanique. *Sensor de Ultrasonido.* URL: <https://eref.se.com/ar/es/web-product-data-sheet/product-pdf/XX630A1KAM12>.
- [23] TOPGNSS. *Sensor módulo GPS Diferencial.* URL: <https://es.aliexpress.com/item/1005003485861159.html>.
- [24] UPower. *Batería.* URL: <https://autosolar.es/baterias-litio-12v/bateria-litio-12v-50ah-upower-ecoline>.
- [25] WitMotion. *Sensor de Unidad de Medición Inercial (IMU).* URL: <https://witmotion-sensor.com/products/bluetooth-5-0-accelerometer-inclinometer-wt901blecl-mpu9250-high-precision-9-axis-gyroscope-anglexy-0-05-accuracy-magnetometer-with-kalman-filter-low-power-3-axis-ahrs-imu-sensor-for-arduino>.