UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Diseño y fabricación de un robot capaz de saltar y controlar su orientación mediante una rueda de inercia

Trabajo de graduación presentado por Diego José Castro Armas para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

2021

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

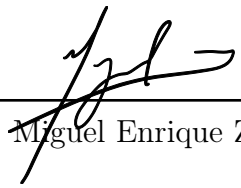
Diseño y fabricación de un robot capaz de saltar y controlar su orientación mediante una rueda de inercia

Trabajo de graduación presentado por Diego José Castro Armas para optar al grado académico de Licenciado en Ingeniería Mecatrónica

Guatemala,

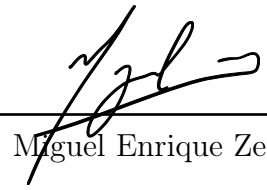
2021

Vo.Bo.:

(f) 

MSc. Miguel Enrique Zea Arenales

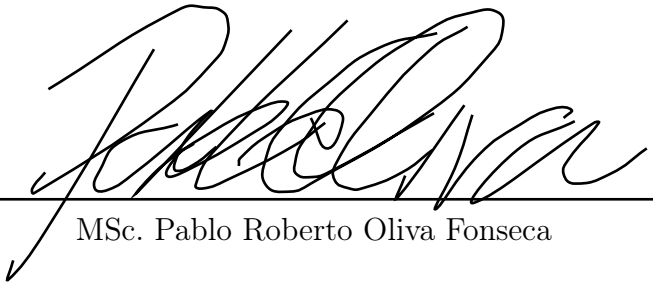
Tribunal Examinador:

(f) 

MSc. Miguel Enrique Zea Arenales

(f) 

MSc. Renato José Conedera Navas

(f) 

MSc. Pablo Roberto Oliva Fonseca

Fecha de aprobación: Guatemala, 14 de enero de 2021.

Prefacio

Principalmento doy gracias a Dios por darme la oportunidad de realizar este proyecto, ya que sin su sabiduría no hubiera podido lograrlo. En sengudo lugar agradezco a la Universidad del Valle de Guatemala la formación académica y profesional que recibí a lo largo de mi carrera. A cada uno de los catedráticos que me guió en cada curso y dio lo mejor de sí para transmitir sus conocimientos. A mi asesor, Msc. Miguel Zea por su colaboración y disposición en todo momento para la realización de este proyecto.

En lo personal, agradezco a mi familia por luchar a mi lado y esforzarse cada día por darme lo mejor y ayudarme a dar lo mejor de mí. Agradezco a cada persona que me dio ánimos para seguir adelante hacia la meta y no rendirme en ningún momento.

Dedico este documento a todas las personas que estuvieron a mi lado en este duro, pero satisfactorio camino y que no dudaron de mí por ninguna razón. De igual forma se lo dedico a mi padre, quien ya no se encuentra conmigo, y a pesar de su pérdida logré cumplir uno de los sueños que compartíamos juntos.

Índice

Lista de figuras

Lista de cuadros

Resumen

En este documento se muestra información a cerca de los conocimientos que se deben tener para poder diseñar y fabricar un robot capaz de saltar y a su vez controlar su orientación utilizando una rueda de inercia. Se muestra un análisis de varios robots con la habilidad de saltar con el fin de seleccionar el mecanismo de salto con las características que se acomplen más a lo requerido. Por otro lado el motivo por el cual se requiere realizar un robot de este tipo, es lograr implementarlo en áreas de exploración en terrenos donde a las personas se les dificulta movilizarse. Se detalla información sobre sistemas dinámicos, los tipos que existen, como modelar un sistema dinámico y a su vez encontrar la solución del mismo.

Se explican diferentes métodos para poder implementar un sistema de control a un modelo dinámico, ya que se requiere que el robot logre controlar su orientación mientras se encuentra en el aire haciendo uso de una rueda inercia como compensador del torque que se genera en el instante que el robot salta, este salto es generado por una fuerza aplicada a cierta distancia del centroide del robot. De igual manera se explica el funcionamiento de una rueda de inercia, para comprender el fin por el cual se utilizará este elemento mecánico, de igual manera se detallará su modelo dinámico para poderlo añadir al modelo del robot. Por otro lado se muestra el procedimiento que se debe seguir para poder diseñar y validar el comportamiento del robot utilizando Autodesk Inventor como software de diseño 3D y RecurDyn con Matlab para la validación del robot en simulaciones, con el diseño del mecanismo se logró que el robot salte una altura de 1.45m con lo que el robot logra saltar 10 veces su estatura. De igual forma se muestran dos variantes de control con los que se puede estabilizar la orientación del robot, un control PID con el cual se obtiene que el robot se estabiliza en 0.45 segundos y un control LQI con el que se obtiene que se estabiliza en 0.4 segundos. Se selecciona como mejor resultado el control LQI ya que es el que logra mantener estable la orientación sin oscilaciones.

Abstract

This document shows information about the knowledge that must be had to be able to design and manufacture a robot capable of jumping and in turn controlling its orientation using an inertia wheel. An analysis of several robots with the ability to jump is shown in order to select the jump mechanism with the characteristics that are more suited to what is required. On the other hand, the reason why a robot of this type is required is to implement it in exploration areas on land where people find it difficult to move. Information about dynamic systems is detailed, the types that exist, how to model a dynamic system and in turn find its solution.

Different methods are explained to be able to implement a control system to a dynamic model, since it is required that the robot manages to control its orientation while it is in the air using an inertia wheel as a compensator of the torque that is generated at the moment that the robot jumps, this jump is generated by a force applied at a certain distance from the centroid of the robot. In the same way, the operation of an inertia wheel is explained, in order to understand the purpose for which this mechanical element will be used, in the same way its dynamic model will be detailed in order to add it to the robot model. On the other hand, the procedure that must be followed to be able to design and validate the behavior of the robot is shown using Autodesk Inventor as 3D design software and RecurDyn with Matlab for the validation of the robot in simulations, with the design of the mechanism it was achieved that the robot jump a height of 1.45m with which the robot manages to jump 10 times its height. In the same way, two control variants are shown with which the orientation of the robot can be stabilized, a PID control with which it is obtained that the robot stabilizes in 0.45 seconds and a LQI control with which it is obtained that it stabilizes in 0.4 seconds. The LQI control is selected as the best result since it is the one that manages to keep the orientation stable without oscillations.

CAPÍTULO 1

Introducción

El proyecto que se presenta a continuación nace de la investigación realizada en un proyecto anterior, donde se estudió el comportamiento de insectos con la habilidad de saltar y escapar de sus presas, como lo son los escarabajos clic. Se tomó la decisión de realizar modificaciones en el proyecto y tomar como objetivo principal el desarrollo de un robot capaz de saltar y controlar su posición angular mientras se encuentra en el aire, esto con el fin de replicar el comportamiento de algunos insectos que huyen de sus presas utilizando sus patas para saltar, pero con la diferencia que los insectos caen en cual quier posición angular mientras que el robot tendrá la capacidad de controlar su orientación mediante una rueda de inercia o una barra de inercia y de esta forma tener un aterrizaje controlado.

A diferencia del trabajo de graduación "Diseño e implementación de un robot bio-inspirado con mecanismo de salto y control de orietanción por medio de una palanca de inercia", en este documento se presenta la situación en la cual el robot cuenta con la rueda de inercia para poder controlar la orientación. Se inició con la investigación de modelos similares al robot con el fin de analizar su mecanismo de salto y poder aplicar uno a este proyecto. Con este mecanismo se diseñó el robot en Autodesk Inventor, continuando con la exportación de dicho diseño al software de RecurDyn donde se seleccionaron los grados de libertad del robot. Con este diseño se procedió a realizar el modelado del sistema, extrayendo datos de RecurDyn para importarlos en Matlab y poder realizar el análisis del modelo lineal del robot. Contando con el modelo lineal se realizaron pruebas para poder aplicar el control PID y en base a la investigación previa para ruedas de inercia, se implementó el modelo dinámico de la rueda de inercia y se aplicó un control LQI al sistema.

Con los dos métodos de control aplicados se procedió a realizar pruebas y analizar que control es más eficiente y así concluir el mejor control para el robot con rueda de inercia.

CAPÍTULO 2

Antecedentes

Se han realizado diversas investigaciones que han basado sus resultados en el mecanismo de salto empleado por las pulgas como [1], otras en el mecanismo de salto de los grillos como [2] y algunos otros no muestran un parecido en su construcción a uno de estos y hacen uso de distintos tipos de resortes y motores para el almacenaje de energía potencial y su posterior liberación, como en [3] y [4]. Uno de los proyectos que cabe mencionar es la primera fase de este proyecto [5], que su investigación se basó en el salto que genera un escarabajo clic al momento de querer voltear su cuerpo al encontrarse boca arriba, pero ya no se decidió continuar con este proyecto porque se observó que era una mejor idea crear un robot con la capacidad de saltar y permanecer en equilibrio para poder aterrizar adecuadamente.

En el Cuadro 1 se puede observar una comparación entre robots ya existentes y sus características. Este cuadro resulta ser de gran utilidad para poder seleccionar el mecanismo que mejor se adapte a las necesidades de esta investigación, considerando que se debe alcanzar la mayor cantidad de desplazamiento horizontal posible, por otro lado tanto el tamaño como el peso deben de ser reducidos.

Basado en estos cuadros y en otras referencias como [6], se encontró que los dos mecanismos que mejor cumplen con los requerimientos anteriormente expuestos son los utilizados en [1] y en [2]. En el caso de [1] se tiene que su alcance horizontal es de los más altos y

Robot	**Masa [g]**	**Tamaño [cm]**	**Tiempo de carga [s]**	**Altura de salto [cm]**	**Distancia [cm]**	**Estabilidad aérea**
Flea robot	1.1	2	15	64	70	No
Frogbot	1300	15	30	90	200	Sí
Jollbot	465	30	1.44	18.4	0	Sí
Grillo III	22	5	8	10	20	No
MIT microbot	46	10	x	38	0	No
MSU jumper	23.5	6.5	10	87.2	89.8	Sí

Cuadro 1: Comparación entre robots saltarines existentes.

que es el mecanismo con menor peso y tamaño de todos. Sin embargo, su pequeño tamaño es debido a que carece de actuadores eléctricos como motores y utiliza únicamente resortes (SMA), por lo que en el caso de incluirlos su tamaño incrementaría significativamente y su estructura y funcionamiento cambiarían. Respecto a [3], se puede observar que también cuenta con uno de los alcances horizontales más grandes, así como un bajo peso y tamaño, además de contar con un solo motor como actuador del mecanismo (lo cual lo vuelve más económico), por lo que se considera a éste como la mejor opción para alcanzar los objetivos de dicha investigación. Adicionalmente se considera también que el mejor método de almacenaje de energía potencial para este mecanismo es por medio de resortes de torsión como los utilizados en [3] y en [6].

Por otro lado, es necesario implementar un sistema de estabilización aérea para el robot, donde se utilizará una rueda de inercia para compensar el movimiento angular del robot al realizar el salto, este sistema es muy común encontrarlo en los sistemas de estabilización para los satélites como se puede observar en [7] con la diferencia que aquí se implementará únicamente para el movimiento rotacional en un solo eje. Una mejor visualización de la implementación de una rueda de inercia como sistema de control de la postura de un robot que salta se puede encontrar en [8].

CAPÍTULO 3

Justificación

A lo largo de la historia, el ser humano se ha catalogado como el ser más versátil en el desarrollo de habilidades. A pesar de esta versatilidad, existen organismos que logran superar al humano al desarrollar habilidades específicas. En la naturaleza podemos encontrar diferentes animales con capacidades específicas que han logrado desarrollar a causa de su evolución y desempeño en su hábitat natural.

Por esta razón, el ser humano se ha dedicado por varios años a replicar el desempeño de estos animales mediante robótica bio-inspirada. Los mecanismos bio-inspirados son la tecnología que ha resultado tener más avances en la robótica ya que tanto universidades como científicos han realizado diversas investigaciones con el fin de recrear los comportamientos que tiene la naturaleza y de esta manera implementar robots en diferentes disciplinas específicas.

En este documento se hablará acerca de un mecanismo capaz de replicar el salto de algunos insectos al momento de escapar de sus presas y lograr estabilizar su posición angular en el aire mediante una rueda de inercia para que de esta manera el robot aterrice de manera horizontal. Este robot puede llegar a utilizarse en diferentes áreas, algunas universidades lo están implementando para realizar exploración en terrenos donde se les dificulta caminar a los exploradores o alcanzar muros de gran altura con el fin de realizar vigilancia en los alrededores del muro y observar lo que sucede.

CAPÍTULO 4

Objetivos

4.1. Objetivo general

Diseñar y fabricar un robot bio-inspirado con la habilidad de saltar y controlar su orientación implementando una rueda de inercia.

4.2. Objetivos específicos

- Diseñar un mecanismo de salto eficiente que logre trazar una trayectoria que maximice el desplazamiento horizontal y a su vez, sea capaz de alcanzar una altura mayor a 5 veces la altura del robot.
- Diseñar un sistema capaz de estabilizar la orientación del robot mediante una rueda de inercia acoplada al robot.
- Verificar y validar el funcionamiento del robot en terrenos irregulares.

CAPÍTULO 5

Alcance

El objetivo principal establecido fue diseñar y verificar el funcionamiento de un robot bio-inspirado, con mecanismo de salto y control de orientación mediante una rueda de inercia. Al inicio de este proyecto se había planteado la idea de diseñar y manufacturar cada pieza, tanto del robot como del mecanismo de salto y de la rueda de inercia. De igual forma se contemplaba diseñar la parte electrónica del robot, ya que tanto el mecanismo de salto como la rueda de incercia cuentan con un motor DC y su respectivo driver, con base en un circuito elétrico con un microcontrolador con el fin de poder aplicar un sistema de control para la orientación y controlar la energía de cada motor. Todo esto conectado a una batería que cumpliría el funcionamiento de la fuente de alimentación y contando con cada una de estas partes mencionadas, lograr ensamblar el robot y físico y realizar las pruebas en "vida real".

Lamentablemente a raíz de la pandemia por el virus COVID-19 del año 2020, la situación de Guatemala y el mundo limitó los objetivos iniciales. Las medidas de distanciamiento social implementadas, empleadas con el fin de prevenir la propagación del virus, forzaron a realizar cambios en los objetivos del proyecto, específicamente los relacionados a la manufactura del robot. A pesar que existió la posibilidad de poder realizar cada diseño CAD para el ensamblaje, el tiempo que se requiere para poder manufacturar y verificar el funcionamiento de cada diseño, unido con las restricciones de horario en el país, generaron una limitante de tiempo en el proyecto ya que fue prácticamente imposible llevar a cabo la investigación con esta limitante.

Debido a esta situación, se estableció la idea de realizar un diseño lo más apegado a la realidad y de esta manera llevar a cabo simulaciones del funcionamiento en diversos terrenos simulados para poder verificar y validar el funcionamiento del robot. A pesar que, de igual forma existió una limitante con muchos componentes necesarios para el robot, se utilizaron los aspectos más importantes de cada uno de ellos, como el peso y el espacio que ocuparían.

En el caso de la programación del robot, una vez más, debido a que no se contó con el robot físico la programación se realizó en el software de Matlab y no en un lenguaje especifíco para un microcontrolador. Sin embargo, se provee una sugerencia de entorno de programación, al igual que de algún controlador que se pueda implementar, basado en la experiencia

del autor. De igual forma se realizan sugerencias para los componenetes electrónicos, como los motores con sus drivers respectivos y la batería que se podría utilizar, con el fin de poder colaborar con la siguiente fase de esta investigación y así cuenten con un punto de partida para la manufactura del robot.

CAPÍTULO 6

Marco teórico

6.1. Sistemas dinámicos

En [9] definen a un sistema dinámico como una combinación de elementos o componentes relacionados entre sí y que actúan para alcanzar un objetivo. El estudio de un proceso dinámico implica el estudio de la variación a lo largo del tiempo de las variables de salida a partir de las variables de entrada, descrito de otra forma, entre las variables medibles y observables que alteran el comportamiento del sistema. Desde un punto de vista matemático, un sistema dinámico se representa por medio de una relación entre las variables de entrada y salida. Estos sistemas pueden clasificarse de acuerdo a sus características como:

- **Sistemas dinámicos continuos**, donde las variables de estado cambian de forma continua a lo largo del tiempo.
- **Sistemas dinámicos discretos**, donde las variables de estado cambian de valor en ciertos instantes de tiempo.
- **Sistemas dinámicos híbridos**, [10] explica que este tipo de sistema es aquel que cuya entrada es una secuencia discreta y su salida es una señal continua. El término sistema dinámico híbrido se utiliza para definir sistemas que tienen variables de carácter discreto y continuo en un conjunto finito de posibles estados.

En [11] se menciona que el propósito de crear un modelo dinámico es poder observar el comportamiento del sistema de forma aislada o con interacciones en su entorno, sin necesidad de manufacturar el sistema físico, sino poder observar lo mencionado mediante simulaciones. Como se mencióno con atnerioridad estos modelos se representan mediante análisis matemático, logrando relacionar sus variables de entrada y de salida. Existen diversas formas en las que se pueden representar estos modelos, por ejemplo, ecuaciones diferenciales, funciones de transferencia y variables de estado (método que se utiliza para este proyecto).

Los sistemas sin importar el método matématico que se utilice pueden obtener una solución cerrada o una solución abierta, para ambos casos se pueden implementar dos métodos de solución:

- **Solución análitica**, donde se logra encontrar la solución como una expresión matemática. Sustituyendo valores para la variable independiente se encuentran los valores para la variable dependendiente.
- **Solución numérica**, para este método se implementan algoritmos de métodos numéricos para llegar a la solución del modelo que describe al sistema.

El desarrollo matemático para lo sistemas conmúnmente es complejo ya que regularmente solo los sistemas LTI (lineares e invariantes en el tiempo) presentan solución análitica cerrada, por lo que se opta por utilizar el método de solución numérica haciendo uso de dispositivos de cómputo. Con el fin de optimizar el costo computacional que realizan estos dispositivos al calcular la solución del sistema se prefiere modelar al mismo en variables de estado, de esta manera es más sencillo para los ordenadores manipular al sistema. El sistema se puede modelar de la siguiente manera:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t)), \tag{1}$$

donde $\mathbf{x}$ es el vector de variables de estado y $\mathbf{u}$ es el vector de variables de entrada, o se puede escribir de manera matricial (únicamente si el sistema es LTI), mediante este sistema se puede encontar un modelo matricial el cual es más sencillo de manipular mediante álgebra lineal:

$$\begin{aligned} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}, \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}. \end{aligned} \tag{2}$$

Como se mencionó anteriormete, la solución de este sistema se puede calcular de forma numérica, donde existen diferentes métodos de calcular dicha solución recursiva como se puede observar en [12], entre los cuales se encuentran el método de bisección, el método de Newton, el método de Euler y el método más utilizado, por su mejor aproximación a la solución y fácil implementación, Runge-Kutta de orden cuatro.

Conociendo la solución del sistema se puede realizar una simulación del mismo donde se puede observar su comportamiento y decidir si se requiere controlar al sistema de alguna forma, lo cual es lo recomendable para garantizar un comportamiento estable.

6.2. Control de sistemas dinámicos

Como se establece en [9], el control de sistemas habitualmente requiere que la(s) salida(s) siga(n) una dirección o trayectoria deseada o permanezca(n) en un punto específico. Es necesario manipular las variables de entrada (únicamente aquellas que puedan ser manipuladas), para lograr regular el comportamiento de las variables de salida y lograr el objetivo deseado. Dentro de la teoría de control se conocen dos grandes ramas:

- Control clásico.
- Control moderno.

6.2.1. Control clásico

Dentro del control clásico se conocen varios métodos para controlar a los sistemas dinámicos, uno de los principales y más conocido es el control proporcional, integral y derivativo (PID por sus siglas en inglés). Según [13], el controlador PID es un mecanismo de control simultáneo por realimentación ampliamente usado en sistemas de control industrial y electrónico. Este calcula la desviación o error entre un valor medido y un valor deseado. Como cualquier método de control, el PID se puede implementar mediante software, donde el algoritmo de este controlador consta de tres parámentros: el valor proporcional que depende del error actual, el valor integral que depende de los errores pasados y el valor derivativo que es una predicción de los errores futuros. Al sumar estos tres parámetros se logra ajustar el proceso a controlar. Matemáticamente la ecuación del PID se observa de la siguiente forma:

$$u(t) = K_P e(t) + \frac{K_P}{T_I} \int_0^t e(\tau) d\tau + K_P T_D \frac{de(t)}{dt}. \tag{3}$$

Donde $u(t)$ es la entrada a la planta del sistema dinámico. K_P es la constante propocional, T_I tiempo de integración y T_D el tiempo de derivación. Esta ecuación se puede rescribir de la siguiente manera:

$$u(t) = K_P e(t) + K_I \int_0^t e(\tau) d\tau + K_D \frac{de(t)}{dt}. \tag{4}$$

Donde K_I es la constante propocional y K_D la constante derivativa. Para que el sistema logre controlarse es necesario ajustar las consntantes del PID, las cuales se pueden ajustar utilizando diferentes métodos, entre los principales se encuentran:

6.2.1.1. Ajuste manual

, con este ajuste se puede iniciar colocando las constantes integral y derivativa en cero y variar únicamente la constante proporcional hasta que la salida del lazo oscile, al ver esta oscilación se debe colocar esta constante a la mitad del valor encontrado. Para la constante

Parámetro	**Tiempo de subida**	**Sobre-elevación y oscilaciones**	**Tiempo de asentamiento**	**Error en estado estable**	**Estabilidad**
K_P	Decrece	Crece	Cambio pequeño	Decrece	Degrada
K_I	Decrece	Crece	Crece	Elimina	Degrada
K_D	Cambio menor	Decrece	Decrece	No tiene efecto (en teoría)	Mejora si K_D pequeño

Cuadro 2: Ajuste de constantes del PID.

K_P	T_I	T_D
$0.6K_u$	$0.5T_u$	$0.12T_u$

Cuadro 3: Constantes de Ziegler-Nichols en lazo cerrado.

integral se debe incrementar hasta que la planta se ajuste al tiempo requerido (aumentar mucho K_I puede provocar inestabilidad en el sistema). Por último se debe incrementar la constante derivativa, si es necesario, hasta que el lazo sea lo suficientemente rápido para alcanzar su referencia al variar bruscamente la carga. También se puede utilizar el Cuadro 2 para ajustar estas constantes:

6.2.1.2. Método de Ziegler-Nichols en lazo cerrado

, en [14] se menciona que este método es una regla simple para elgir los parámetros de los reguladores PID, está idealmente emparejada a la determinación de K_u y T_u por el método del relé. Los valores del controlador se pueden observar en el Cuadro 3. Estos parámetros dan un sistema en lazo cerrado con amortiguamiento bajo. Sistemas con mejor amortiguamiento se pueden obtener por ligeras modificaciones en los valores de la tabla.

6.2.2. Control moderno

En control moderno se conocen distintos métodos para realizar control, en este documento se explicarán dos métodos de ellos.

6.2.2.1. Regulador Cuadrático Lineal (LQR)

En [15] se tiene que el control LQR es una acción de control moderno multivariable que se caracteriza por su robustez tanto tiempo continuo como en tiempo discreto. Esta ley de control responde en esencia a una acción de control lineal como se ve en la siguiente ecuación:

$$\mathbf{u} = -\mathbf{K}\mathbf{x}. \tag{5}$$

Esta ecuación se puede modificar al utilizar un punto de operación específico ($\mathbf{x}_{ss}$, $\mathbf{u}_{ss}$) como:

$$\mathbf{u} = -\mathbf{K}(\mathbf{x}_{ss} - \mathbf{x}) + \mathbf{u}_{ss}. \tag{6}$$

El control LQR se calcula al momento de minimizar la siguiente función de costo:

$$\mathbf{J} = \int_0^\infty (\mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{u}^T\mathbf{R}\mathbf{u})dt, \tag{7}$$

donde $\mathbf{Q}$ y $\mathbf{R}$ son matrices de penalización para las variables de estado y para las variables de entrada respectivamente. Estas matrices típicamente son diagonales con dimensión de la cantidad de variables de estado para $\mathbf{Q}$ y de las variables de entrada para $\mathbf{R}$, al igual que los valores deben ser positivos. Mientras más grandes son los valores de $\mathbf{Q}$ la exigencia al controlador es mayor y mientras los valores de $\mathbf{R}$ aumenten más, se incrementa la influencia de las acciones del control a la planta.

Algo importante que mencionar es que para poder hacer uso de este método de control, previamente se debe verificar que el sistema sea completamente controlable. Según [15] un sistema completamente controlable es aquel que converge a una referencia en un tiempo finito, es decir que sin importar cuál sea su estado actual en algún momento la respuesta se estabilizará en un punto definido.

Para comprobar la controlabilidad se puede establecer que el rango de la matriz de controlabilidad coincida con el orden del sistema como se muestra a continuación:

$$\text{rank}[\mathbf{B}, \mathbf{AB}, \mathbf{A}^2\mathbf{B}, ..., \mathbf{A}^{n-1}\mathbf{B}] = n. \tag{8}$$

Donde n es el orden del sistema. Con esto se logra garantizar que en el comportamiento dinámico del sistema no exista un punto en el que se torne inestable.

6.2.2.2. Regulador Lineal Cuadrático Integral (LQI)

En [16] definen a los controladores LQI como controladores de retroalimentación estática basados en los estados aumentados (los estados de la planta más los errores integrales de seguimiento). Estos controladores estabilizan las plantas aumentadas y minimizan los costos de funcionamiento cuadráticos asociados con entradas exógenas escalonadas (entradas de referencia escalonadas y perturbaciones escalonadas).

En el control LQI se define una integral para el error del rastreo de referencia, ya que la parte integral se asemeja a la parte integral del controlador PID, esta integral se muestra a continuación:

$$\boldsymbol{\xi} = \int_0^t (\mathbf{r} - \mathbf{y}(\tau))d\tau. \tag{9}$$

De la ecuación anterior se puede encontrar:

$$\dot{\boldsymbol{\xi}} = \mathbf{r} - \mathbf{C}\mathbf{x}(t). \tag{10}$$

A partir de la ecuación anterior se puede encontrar el modelo aumentado del sistema:

$$\begin{bmatrix} \dot{\mathbf{x}}(t) \\ \dot{\boldsymbol{\xi}} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & 0_{n\times p} \\ \mathbf{C} & 0_{p\times p} \end{bmatrix} \begin{bmatrix} \mathbf{x}(t) \\ \boldsymbol{\xi}(t) \end{bmatrix} + \begin{bmatrix} \mathbf{B} \\ 0 \end{bmatrix} \mathbf{u}(t). \tag{11}$$

Donde p es el número de referencias rastredas. De este modelo se toman las matrices aumentadas y se reescribe el modelo como un modelo dinámico normal.

$$\dot{\mathbf{z}}(t) = \widetilde{\mathbf{A}}\mathbf{z}(t) + \widetilde{\mathbf{B}}\mathbf{u}(t), \tag{12}$$

$$\mathbf{w}(t) = \widetilde{\mathbf{C}}\mathbf{z}(t).$$

Conociendo este sistema se puede aplicar la misma teoría para el control LQR al igual que su implementación, con la única diferencia que en lugar de utilizar las matrices de estado, se utilizan las matrices aumentadas del sistema.

6.3. Ruedas de inercia

En [17] se describen como un elemento mecánico en forma de disco utilizado la mayoría de veces en vehículos espaciales con el fin de poder cambiar su momento angular sin consumir combustible, ya que las ruedas de inercia son impulsadas por un motor eléctrico que se encuentra acoplado a su eje. Este tipo de ruedas son útiles cuando una nave espacial debe ser girada en pequeñas cantidades y ser orientada en un ángulo específico.

Por la conservación de momento angular la nave gira proporcionalmente en la dirección opuesta a la rotación del motor. Este tipo de rueda sólo permite la rotación de la nave espacial alrededor de su centro de masa pero en ningún caso genera un movimiento de traslación. Regularmente las ruedas de inercia en las naves espaciales trabajan a revoluciones nominales de cero, pero las naves al encontrarse sometidas a torques externos en el espacio necesitan que las ruedas de inercia giren para poder mantener a la nave en una orientación fija, como se desea que permanezca el robot al momento de encontrarse en el aire.

Al implementar la rueda de inercia en el modelo del robot, el comportamiento de la orientación según [8] se vería de la siguiente manera:

$$\left(J_b + J_m + J_w + \frac{m_b l_g^2 (m_m + m_w)}{m_b + m_m + m_w}\right)\ddot{\theta} + (J_w + n_g J_m)\ddot{\theta}_w + c_b\dot{\theta} = 0, \tag{13}$$

$$(J_w + n_g J_m)\ddot{\theta} + (J_w + n_g^2 J_m)\ddot{\theta}_w + c_w\dot{\theta}_w = \tau. \tag{14}$$

Donde J_b es la inercia del robot, J_m la inercia del motor, J_w la inercia de la rueda, m_b la masa del robot, m_m la masa del motor, m_w la masa de la rueda, n_g la relación de la caja reductora del motor, l_g la distancia del centro de masa del robot al centro de masa de la rueda, c_b la constante de amortiguamiento del cuerpo en rotación y c_w la constante de amortiguamiento de la rueda en rotación. El modelo se puede reescribir en forma matricial de la siguiente manera:

$$\mathbf{M}\begin{bmatrix}\ddot{\theta}\\ \ddot{\theta}_w\end{bmatrix} + \begin{bmatrix}c_b\dot{\theta}\\ c_w\dot{\theta}_w\end{bmatrix} = \begin{bmatrix}0\\ \tau.\end{bmatrix} \tag{15}$$

Donde

$$\mathbf{M} = \begin{bmatrix}\mathbf{M_{11}} & \mathbf{M_{12}}\\ \mathbf{M_{21}} & \mathbf{M_{22}},\end{bmatrix} \tag{16}$$

$$\mathbf{M_{11}} = \left(J_b + J_m + J_w + \frac{m_b l_g^2 (m_m + m_w)}{m_b + m_m + m_w}\right),$$

$$\mathbf{M_{12}} = \mathbf{M_{21}} = J_w + n_g J_m,$$

$$\mathbf{M_{22}} = J_w + n_g^2 J_m.$$

Si se toma el vector de estado como $\mathbf{x} = [\theta, \dot{\theta}, \dot{\theta}_w]^T$, las matrices de estados se obtienen de la siguiente manera:

$$\mathbf{A} = \begin{bmatrix}0 & 1 & 0\\ 0 & \frac{a_{22}}{\Delta} & \frac{a_{23}}{\Delta}\\ 0 & \frac{a_{32}}{\Delta} & \frac{a_{33}}{\Delta},\end{bmatrix} \tag{17}$$

$$\mathbf{B} = \begin{bmatrix} 0 \\ \frac{b_{21}}{\Delta} \\ \frac{b_{31}}{\Delta}, \end{bmatrix}$$

$$\mathbf{C} = \begin{bmatrix} 1 & 0 & 0, \end{bmatrix}$$

$$\mathbf{D} = 0.$$

Y los elementos de las matrices son dados por:

$$\Delta = \mathbf{M_{11}M_{22}} - \mathbf{M_{12}M_{21}}, \tag{18}$$

$$a_{22} = -c_b(J_w + n_g^2 J_m), \quad a_{23} = c_w(J_w + n_g J_m), \tag{19}$$

$$a_{32} = c_b(J_w + n_g J_m), \quad a_{33} = -c_w \left(J_b + J_m + J_w + \frac{m_b l_g^2 (m_m + m_w)}{m_b + m_m + m_w} \right),$$

$$b_{21} = -(J_w + n_g J_m), \quad b_{31} = J_b + J_m + J_w + \frac{m_b l_g^2 (m_m + m_w)}{m_b + m_m + m_w}. \tag{20}$$

6.4. Software de simulación RecurDyn

RecurDyn es un software de ingeniería asistida por computadora (CAE, por sus siglas en inglés) capaz de realizar simulaciones muy apegadas a la realidad de sistemas de multiples cuerpos (MBD, por sus siglas en inglés). Este software realiza simulaciones de dinámica de cuerpos rígidos y flexibles combinando MBD rígido tradicional con tecnología de vanguardia de elementos finitos para modelar cuerpos flexibles.

Para poder utilizar RecurDyn es necesario crear un diseño asistido por computadora (CAD, por sus siglas en inglés) con algún sofware de diseño, como por ejemplo Autodesk Inventor. Luego de realizar el CAD, se debe exportar como un archivo .step, este archivo se importa en el sofware de RecurDyn donde se inicia con colocar las restricciones necesarias del diseño, entre estas restricciones se encuentran los tipos de juntas entre los eslabones, restricciones de contacto y restricciones de movimiento. De igual forma se pueden colocar diversos tipos de fuerzas, como fuerzas axiales, rotacionales, de tipo resorte tanto axiales como rotacionales.

Cuando el diseño contenga todos los parámetros necesarios, se pueden realizar simulaciones para observar el comportamiento, en este caso, del robot. De igual forma se puede realizar una conexión con Simulink de Matlab, para poder realizar pruebas, mediciones, entre otras cosas. Esta conexión es de suma importancia, ya que se mezcla la facilidad de implementar sistemas de control en Simulink y la visualización de una simulación lo más real posible de RecurDyn. Esta conexión se puede realizar colocando a RecurDyn como *Host* o como *Client*.

CAPÍTULO 7

Metodología

7.1. Diseño mecánico

Utilizando como base la información mencionada en la sección de antecedentes y mecanismos en aplicaciones similares a este proyecto, se procedió a diseñar al robot tal que cumpliera con los requerimientos propuestos, que sea liviano, pequeño (tanto como lo permitan los componentes), con una rueda de inercia para controlar la orientación y dos motores, uno para la rueda de inercia y otro para el mecanismo de salto.

7.2. Entorno de simulación

Con el diseño establecido del robot, se procedió a desarrollar el modelo CAD del mismo utilizando Autodesk Inventor, donde el diseño realizado fue lo más real posible. Luego de tener este diseño CAD realizado se importó en el software de simulación RecurDyn como un archivo tipo .step, en este software se colocaron las juntas del robot y los grados de libertad del mismo para poder apreciar el movimiento real en las pruebas de las simulaciones.

7.3. Mediciones del sistema

Para poder realizar las mediciones con el robot era necesario contar con una unidad de medición inercial (IMU, por sus siglas en inglés), pero al no contar con dicho sensor en RecurDyn fue necesario realizar una conexión entre este software y Simulink de Matlab, esto con el fin de poder medir los ángulos de roll (rotación sobre el eje x), pitch (rotación sobre el eje y) y yaw (rotación sobre el eje z) del robot, donde el ángulo de interés en este caso es

el de yaw, ya que este ángulo es el que interesa controlar en RecurDyn para mantener una orientación controlada.

7.4. Modelado del sistema

Utilizando la conexión entre RecurDyn y Simulink de Matlab, se utilizó la función de Simulink, *Linear Analysis Tool*, con la cual se logró obtener el modelo dinámico más cercano al sistema original, este modelo se obtubo con el sistema en estado estacionario.

7.5. Implementación del control

Con el modelo del sistema implementado, fue posible implementar métodos de control para poder controlar la orientación del robot. Se realizon dos métodos de control con el fin de observar cuál era el más conveniente para esta aplicación. Uno de ellos fue un control clásico, control PID; y un control moderno, control LQI.

7.6. Pruebas y validación del robot

Conociendo que método de control fue más eficiente para esta aplicación se procedió a realizar las pruebas en el software de RecurDyn en diferentes terrenos para poder validar de mejor manera el funcionamiento del robot.

CAPÍTULO 8

Diseño mecánico

8.1. Diseño del mecanismo

El diseño del mecanismo del robot es muy parecido al de un triquet hidráulico, este se muestra en la Figura 1. Se tomó este diseño ya que en varios documentos de investigación el diseño del robot que utilizan es similar y por facilidad de implementar un mecanismo de salto, ya que cuenta con resortes torsionales en las juntas A, C, D y F con las que al aplicarle una fuerza de compresión F_J (esta fuerza es generada por uno de los motores) los resortes se torsionan, y al quitar la fuerza los resortes regresan a su estado inicial provocando el salto del robot. Las dimensiones del robot se consideran en base a los componentes que se colocaran en él, como, el microcontrolador en su PCB, los motores, los drivers y la batería. Conociendo esto se define $L_1 = L_2 = L_3 = L_4$. El ángulo θ es medido respecto de la horizontal.

8.2. Parámetros del motor del mecanismo de salto

Para poder definir el motor que se desea utilizar se requiere de ciertos parámetros que deben cumplir con los requerimientos de su funcionamiento, entre dichos parámetros se encuentran: el torque, el radio del eje y la fuerza que este motor puede ejercer. Tanto el torque como el radio del eje se pueden conocer mediante la hoja de datos del dispositivo, mientras que la fuerza del motor es fácil calcularla utilizando el torque y el radio con la siguiente expresión:

$$\tau = F_J r. \tag{21}$$

Donde τ es el torque, F_J la fuerza y r el radio del eje. Despejando para F_J, la expresión resultante es:

$$F_J = \frac{\tau}{r}. \tag{22}$$

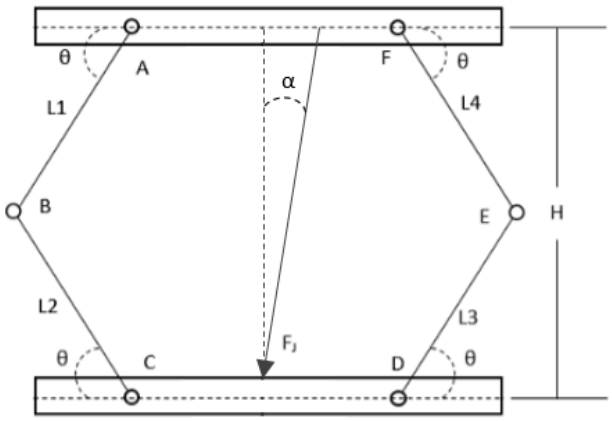


Figura 1: Mecanismo del robot.

Cabe mencionar que la fuerza efectiva para generar el salto es la componente vertical de la fuerza F_J, por lo que es necesario calcularla como se muestra en la ecuación 23.

$$F_{J_y} = F_J \cos \alpha. \tag{23}$$

8.3. Definición de los resortes

Con el mecanismo previamente definido se procedió a establecer y definir los resortes torsionales a utilizar para el mecanismo de salto. Para esto fue necesario calcular las fuerzas y los torques que experimenta el mecanismo considerando el diagrama de cuerpo libre de la Figura 2. En donde f son las fuerzas que se necesitan para torsionar los resortes, y f_x y f_y sus componentes horizontal y vertical respectivamente. En el DCL se muestran únicamente dos eslabones, pero el procedimiento es el mismo para los otros dos con la diferencia en que las direcciones son contrarias.

Considerando que todos los resortes utilizados son iguales, se calculó que la componente vertical de la fuerza F_J debe ser n veces f_y para lograr torsionar los resortes, donde n es el número de resortes utilizados.

$$F_{J_y} = n f_y. \tag{24}$$

Conociendo F_{J_y} de (23) fue posible calcular la fuerza f_y despejando (24). Esta fuerza es la componente vertical de la fuerza neta que debe resistir cada resorte. Conociendo f_y y el ángulo θ de los eslabones, se logró calcular la fuerza neta aplicada sobre los resortes de la siguiente manera:

$$f = \frac{f_y}{\cos \theta}. \tag{25}$$

De igual forma fue posible calcular el torque máximo τ_{max} de cada resorte mediante f y la

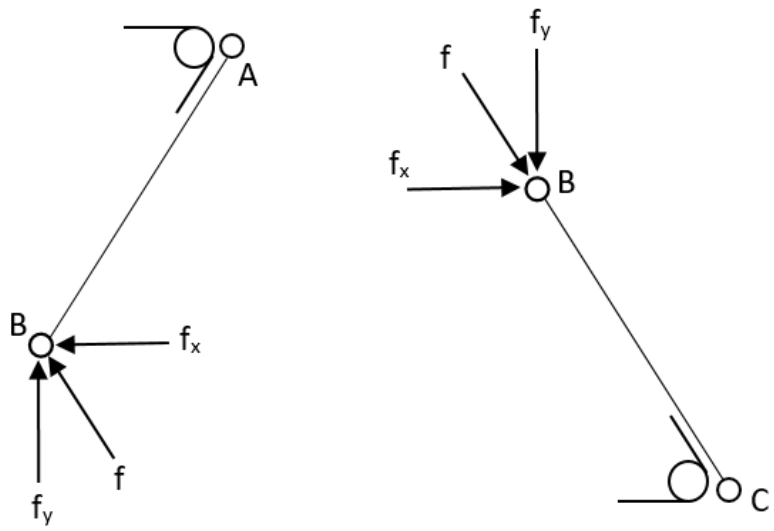


Figura 2: DCL de los eslabones.

longitud de los eslabones L_n.

$$\tau_{max} = fL_n. \tag{26}$$

Por otro lado, es importante mencionar que para evaluar que los resortes elegidos no se fracturen o se deformen plasticamente, pero a su vez el motor pueda deformarlos elásticamente, se utilizó la Ley de Hooke en forma angular (27).

$$\theta_{max} = \tau_{max}/k. \tag{27}$$

Esta ley indica a qué ángulo ocurre cierto torque dependiendo de la constante elástica torsional del elemento mecánico. Por lo que primero es necesario conocer el torque máximo que puede generar el motor sobre los resortes para luego poder seleccionar un resorte y verificar si este resorte puede ser torsionado por el motor al ángulo θ requerido y que a su vez el torque máximo que el motor genera sea menor al torque máximo que soporta el resorte.

Al conocer cada valor mencionado anteriormente, se eligió el modelo del resorte a emplear. Se buscó que fuera prefabricado, ya que es más sencillo comprar uno que diseñarlo y manufacturarlo. Se decidió utilizar un resorte de la marca *Vanel* ya que este provee el modelo CAD de sus productos. En este catálogo se desea encontrar un resorte con un ángulo de 90^o y con un torque máximo mayor al torque aplicado, este tipo de resorte se puede buscar en [18].

8.4. Análisis físico

Para estimar la altura máxima que el robot puede alcanzar, se utilizó el concepto de conservación de energía mecánica, en donde se consideraron la energía almacenada por lo

resortes y la energía potencial del sistema en el aire. Cabe mencionar que para realizar estos cálculos se modeló al robot como una partícula.

$$E = E_k + E_P. \tag{28}$$

En donde E es la energía mecánica del sistema, E_k la energía potencial de los resortes y E_P la energía potencial gravitacional. Asumiendo que la energía se conserva en el sistema se encontró la siguiente igualdad:

$$E_k = -E_P. \tag{29}$$

Donde la energía potencial de los resortes de describe como:

$$E_k = n\frac{1}{2}k\theta^2, \tag{30}$$

donde: n es la cantidad de resortes, k la constante del resorte y θ el ángulo de los eslabones con respecto de la horizontal. Por otro lado, la energía potencial gravitacional se describe de la siguiente manera:

$$E_P = -mgh, \tag{31}$$

donde m es la masa total del robot y esta se puede calcular utilizando el modelo CAD y los datos de las hojas de datos de los demás componenetes, g es la constante gravitacional que tiene un valor de $9.81m/s$ y h es la altura que el robot alcanzará con el salto. El signo menos es por la convención que se estableció.

Utilizando la igualdad de (29) y las definiciones de energía en (30) y (31) se logró encontrar una expresión para h:

$$h = \frac{nk\theta^2}{2mg}. \tag{32}$$

8.5. Componentes seleccionados

En el Cuadro 4 se muestran los componentes que se proponen utilizar al momento de realizar el diseño físico del robot. Cabe mencionar que estos elementos fueron elegidos en base a los requerimientos que se necesitan cumplir y de igual forma por la facilidad de adquirirlos. Los motores que se proponen son de la marca Pololu, esta es una empresa que fabrica componentes electrónicos y cuenta con una diversidad de motores, entre los cuales se encuentran los RB-Pol-4xx [19], de los cuales se seleccionaron 4 motores para realizar los cálculos como se observa en el Cuadro 5. De igual forma el Driver MC33926 [20] se encuentra disponible en Pololu, este driver es Dual-Motor, significa que puede alimentar dos motores al mismo tiempo recibiendo señales de un microcontrolador. Para poder medir el ángulo de pitch (el ángulo que se desea controlar), se propuso un acelerómetro y giroscopio de tres ejes MPU6050 [21]. Para alimentar el sistema completo se recomienda utilizar dos baterías Li-Po LP602040 [22] que cuentan con 450mAh y 3.7V, en serie para lograr un voltaje de 7.4V y así alcanzar el torque máximo de los motores. En el caso del microcontrolador se proponen dos, el microcontrolador TM4C123H6PM que es el que utiliza la board Tiva C, se consideró este por su facilidad de programar tanto en su software de Energia como en los softwares de CodeComposer o *µVision*. Por otro lado el microcontrolador ATmega128RFA1 tiene como ventaja la facilidad de la implementación del módulo MPU6050, ya que cuenta con librerías

para este sensor. Ambos microcontroladores tienen ventajas y desventajas. Se consideraron estos ya que esta aplicación de controlador no necesita de un período de muestreo muy pequeño, por lo que dichos microcontroladores puede alcanzar esta velocidad.

8.6. Resultados

Para realizar los cálculos y obtener los resultados requeridos se tomaron en cuenta algunos parámetros iniciales donde se realizaron modificaciones únicamente en el ángulo donde se aplica la fuerza de los resortes, estos parámetros se pueden observar en el Cuadro 6.

Tomando en consideración un ángulo de 45^o se obtuvieron los resultados tanto para los resortes como para la relación de altura del salto y la altura del robot en los Cuadros 7 y 8, respectivamente.

Tomando en consideración un ángulo de 60^o se obtuvieron los resultados tanto para los resortes como para la relación de altura del salto y la altura del robot en los Cuadros 9 y 10, respectivamente.

Algo importante que mencionar es que en la columna de "Ley de Hooke"de los Cuadros 8 y 10 se calculó el torque mínimo que se necesita para poder torsionar los resortes al ángulo especificado. Este valor se compara con el torque máximo que puede ejercer el motor sobre los resortes, con el fin de saber si este torque máximo del motor es mayor que el torque mínimo que necesita el resorte para ser torsionado y de esta manera saber si cumple con dicho requerimiento y poder implementarlo.

De los resultados que se obtuvieron realizando los cálculos mostrados en las tablas anteriores, se tomó la decisión de realizar la implementación del motor RB-Pol-456, ya que con este motor se lograba generar la mayor fuerza y se alcanzó la mayor altura del salto, teniendo un salto de 2.133m. De igual forma se utilizaron los resortes con código D.182.220.0425. A pesar que este resorte cuenta con el radio externo mayor a los demás resortes, este diámetro no es grande en comparación con el robot y de igual forma cumple con el requerimiento de la Ley de Hooke, por lo que la mejor opción es el motor con caja reductora de 298:1 en conjunto con ocho resortes D.182.220.0425 del catálogo de Vanel [18] donde se logra una relación de alturas de 14.81.

Por otro lado, es importante mencionar que para la fabricación de la rueda de inercia se recomienda utilizar ácido poliláctico (PLA por sus siglas en inglés) como primera iteración,

Componente	Propuesta
Motor	RB-Pol-4xx
Driver	Dual MC33926 Motor Driver
Sensor	Acelerómetro y Giroscopio MPU6050
Batería	Li-Po LP602040
Microcontrolador	TM4C123H6PM o ATmega128RFA1

Cuadro 4: Componentes recomendados.

Motor	Reducción	Torque [Nm]
RB-Pol-446	1:50	0.0431
RB-Pol-447	1:75	0.0628
RB-Pol-448	1:100	0.0726
RB-Pol-452	1:298	0.1961

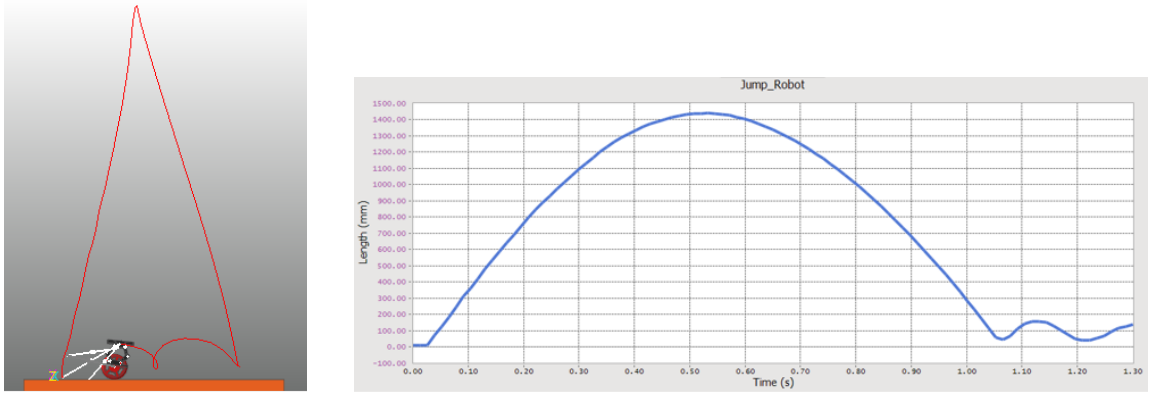
Cuadro 5: Motores seleccionados.

Parámetro	Valor inicial
r [m]	0.0015
n	8
θ	60^o y 45^o
L_n [m]	0.0372
m [kg]	0.181
h_{Robot} [m]	0.144

Cuadro 6: Parámetros iniciales.

de igual forma que la estructura del mecanismo del robot se puede fabricar con el mismo material. En la Figura 4 se puede observar el diseño propuesto para el robot.

De igual forma se observó el funcionamiento del robot considerando los valores del resorte seleccionado y de la fuerza aplicada (que genera el motor seleccionado) para el salto en el software de RecurDyn para poder verificar el funcionamiento del robot y verificar que se cumplieran los cálculos realizados. Los resultados de esta prueba se pueden obseravar en la Figura 3.



(a) Trayectoria del salto del robot.

(b) Movimiento en y del robot.

Figura 3: Simulación del salto en RecurDyn.

Algo importante que se debe mencionar es que en las simulaciones de RecurDyn se eliminaron los componentes electrónicos del robot como el motor que ejerce la fuerza de salto, las baterías, el sensor de orientación, la placa del microcontrolador, los resortes y los pines de unión de los eslabones, ya que estos por cuestiones de recursos computacionales provocaban que las simulaciones tardaran mucho más en compilar. De igual forma, para

Reducción motor / Parámetro	298:1	100:1	75:1	50:1
τ [Nm]	0.1961	0.0726	0.0628	0.0431
F_J [N]	130.73	48.40	41.87	28.73
F_{J_y} [N]	126.60	46.87	40.54	27.82
f_y [N]	15.82	5.86	5.07	3.48
f [N] (m)	22.38	8.29	7.17	4.92
τ_{max} [Nm]	0.833	0.308	0.267	0.183
Código Vanel del resorte	D.150.200.0225	D.104.130.0425	D.095.120.0425	D.110.125.0525
Diámetro externo del resorte [mm]	15	10.4	9.5	11

Cuadro 7: Cálculos de resortes para un ángulo de 45^o.

Código Vanel del resorte	Ley de Hooke	¿Cumple?	k [Nm/rad]	Energía del resorte [J]	h [m]	Razón de h
D.110.125.0525	0.092	Cumple	0.1176	0.369	0.134	0.93
D.095.120.0425	0.141	Cumple	0.1795	0.564	0.204	1.42
D.104.130.0425	0.164	Cumple	0.2088	0.656	0.238	1.65
D.150.200.0225	1.241	No cumple	1.5798	4.963	1.798	12.49

Cuadro 8: Cálculo de relación de altura con ángulo de 45^o.

compensar el peso de cada uno de estos componentes se modificó la densidad del material con el cual se diseñó el robot. Por otro lado, en la simulación se puede observar que la altura máxima que el robot alcanza es de aproximadamente 1.5m, mientras que el valor calculado con anterioridad es de 2.133m. Este margen de error se debe a que en los cálculos se modela al robot como una partícula, mientras que el software realiza los cálculos con un modelo más detallado con el fin de poder observar simulaciones apegadas a la realidad.

La bioinspiración del robot viene de los insectos que pueden alcanzar alturas que rebasan su estatura hasta 7 veces más como es el caso de las langostas de desierto que tienen la capacidad de saltar hasta 9 veces más su altura. Del resultado de salto se puede observar que el robot es capaz de saltar aproximadamente 10 veces más su altura por lo que el mecanismo de salto cumple con lo requerido de la bioinspiración.

Para comprobar que las piezas y los resortes no se romperán al aplicarle la fuerza para generar el salto, se realizó una prueba de esfuerzos al aplicarle dicha fuerza. El resultado de esta prueba se puede observar en la Figura 5 donde se muestra que el esfuerzo máximo fue de 99.82MPa y es aplicado en el resorte frontal del eslabón izquierdo superior. Ya que los resortes están fabricados con acero inoxidable, el cual soporta un esfuerzo máximo de 250MPa, se concluyó que el diseño soportará los esfuerzos aplicados.

Reducción motor / Parámetro	298:1	100:1	75:1	50:1
τ [Nm]	0.1961	0.0726	0.0628	0.0431
F_J [N]	130.73	48.40	41.87	28.73
F_{J_y} [N]	126.60	46.87	40.54	27.82
f_y [N]	15.82	5.86	5.07	3.48
f [N] (m)	31.65	11.72	10.14	6.96
τ_{max} [Nm]	1.177	0.436	0.377	0.259
Código Vanel del resorte	D.182.220.0425	D.127.150.0425	D.127.150.0525	D.095.120.0425
Diámetro externo del resorte [mm]	18.2	12.7	12.7	9.5

Cuadro 9: Cálculos de resortes para un ángulo de 60^o.

8.7. Diseño final del robot

Ya que al momento de realizar las pruebas del modelado y del control del sistema se contaron con algunos problemas de diseño se tomó la decisión de cambiar algunas caracte-

Código Vanel del resorte	Ley de Hooke	¿Cumple?	k [Nm/rad]	Energía del resorte [J]	h [m]	Razón de h
D.095.120.0425	0.188	Cumple	0.1795	0.752	0.363	2.52
D.127.150.0525	0.256	Cumple	0.2447	1.025	0.495	3.44
D.127.150.0425	0.317	Cumple	0.3024	1.267	0.612	4.25
D.182.220.0425	0.904	Cumple	0.8633	3.616	1.747	12.13

Cuadro 10: Cálculo de relación de altura con ángulo de 60^o.

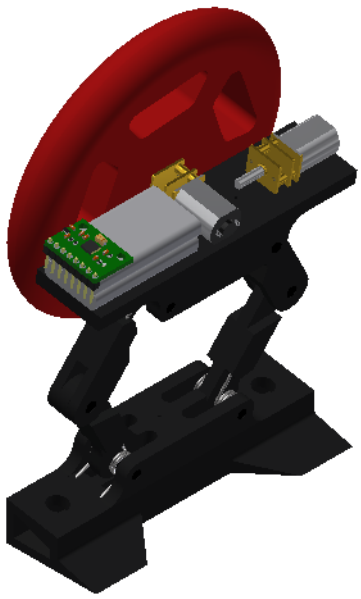


Figura 4: Diseño propuesto para el robot.

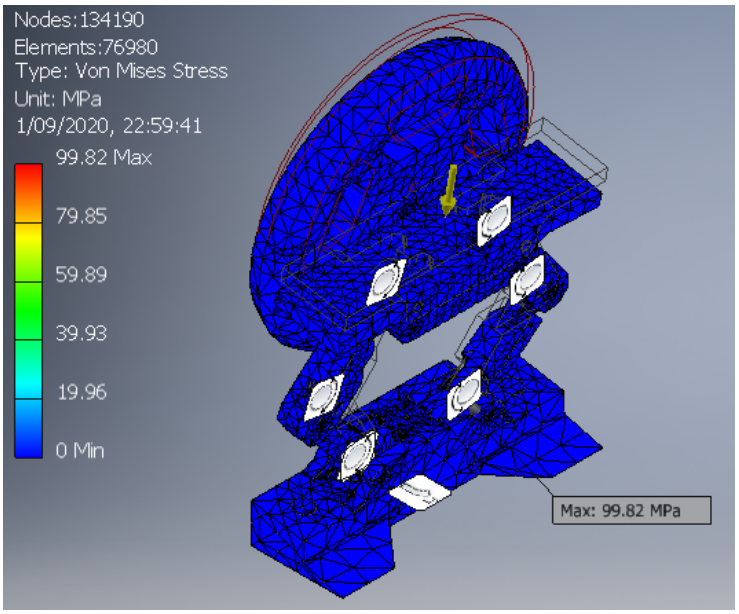


Figura 5: Análisis de esfuerzos para una fuerza puntual.

rísticas del robot. Estas características fueron que el ángulo con el que contaba el robot para compensar el peso de la rueda de inercia fue eliminado, al igual que la rueda de inercia fue modificada, se redujo de diámetro y de grosor al igual que se cambió su material de PLA a acero. Las razones del por qué se realizaron estas modificaciones se explican en el capítulo de control.

El nuevo diseño se puede apreciar en la Figura 6, donde se observa que la rueda de inercia es más pequeña y más delgada. Ya que para este robot la masa disminuyó, se realizó un nuevo cálculo de la altura máxima que puede alcanzar, donde se encontró que para una masa de 0.181 kg (masa del robot), la altura máxima que puede alcanzar el robot, modelándolo como partícula, es de 2.13m, logrando que salte aproximadamente 15 veces su altura. Cabe mencionar que para este cálculo se utilizaron los resortes y el motor seleccionados en el diseño anterior.

Por otro lado, ya que el diseño fue modificado, se realizó un nuevo análisis de esfuerzos, el cual se logra apreciar en la Figura 7, donde se observa que el esfuerzo máximo es de 138.6 MPa aplicado en el resorte frontal del eslabón inferior derecho. Ya que para los resortes el esfuerzo máximo que pueden soportar es de 250 MPa, se concluyó que el diseño es capaz de soportar la fuerza que aplica el motor sobre el robot.

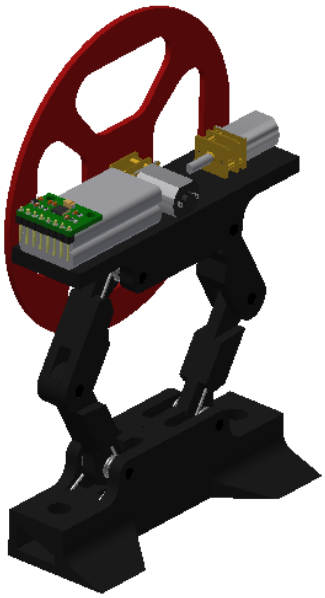


Figura 6: Diseño final del robot.

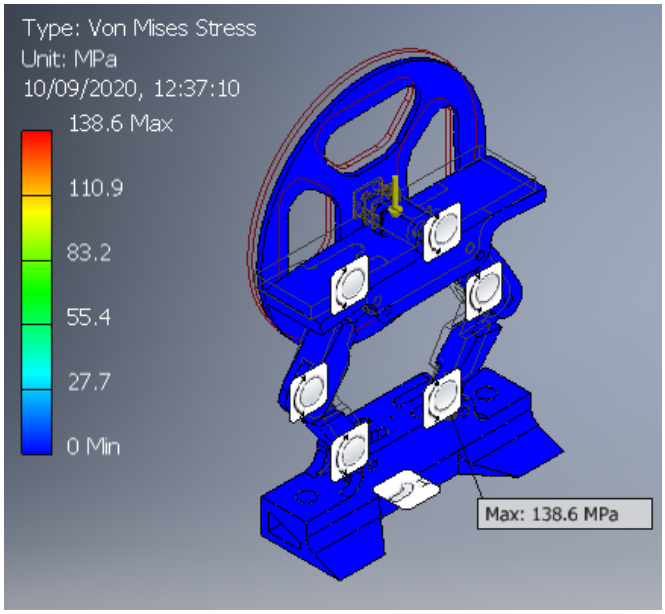


Figura 7: Análisis de esfuerzos para una fuerza puntual.

CAPÍTULO 9

Modelado del sistema

9.1. Modelo inicial

Para el análisis del sistema se inició con un modelo simple, donde se modela al robot como un bloque rectangular como se observa en la Figura 8 con un ancho con valor a, altura b, grosor c, masa m y una inercia I_{zz}; esta inercia se calculó mediante la ecuación (33).

$$I_{zz} = \frac{1}{2}m(a^2 + b^2) \tag{33}$$

De igual forma se realizó un análisis físico de este robot al encontrarse en el aire como se muestra en la Figura 9, donde se puede encontrar un vector de variables de estado $\mathbf{q}$, donde sus componentes q_1, q_2 y q_3 son la ubicación del robot respecto al marco inercial x_I, y_I y θ respectivamente. En este análisis se encontró su modelo dinámico, el cuál se muestra a continuación.

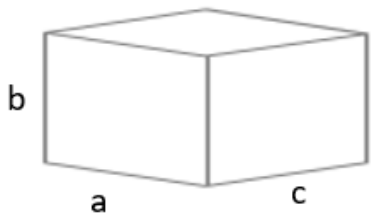


Figura 8: Robot inicial.

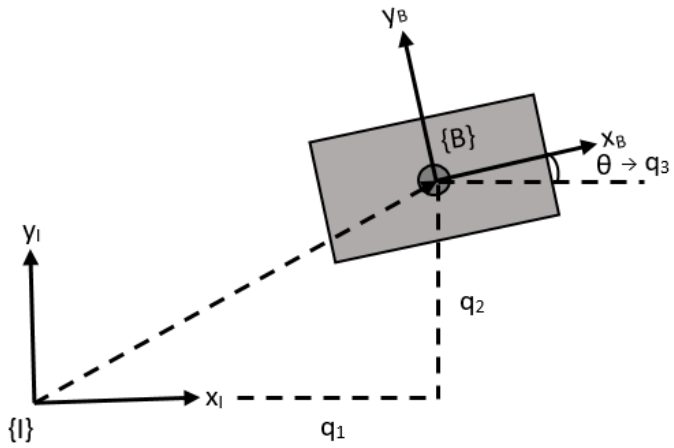


Figura 9: Análisis del modelo inicial.

$$\sum f_{x_I} = m\ddot{q_1} \rightarrow f_{x_I} = m\ddot{q_1} \rightarrow \ddot{q_1} = \frac{f_{x_I}}{m}, \tag{34}$$

$$\sum f_{y_I} = m\ddot{q_2} \rightarrow f_{y_I} - mg = m\ddot{q_2} \rightarrow \ddot{q_2} = \frac{f_{x_I}}{m} - g, \tag{35}$$

$$\sum \tau_z = I_{zz}\ddot{q_3} \rightarrow \tau_z = I_{zz}\ddot{q_3} \rightarrow \ddot{q_3} = \frac{\tau_z}{I_{zz}}. \tag{36}$$

donde f_{x_I}, f_{y_I} y τ_z se calculan utilizando la fuerza impulsional aplicada. Para el análisis con la fuerza impulsional aplicada se puede utilizar la Figura 10, donde $\mathbf{u}$ es la fuerza impulsional, y $\mathbf{r}$ es la posición donde se aplica dicha fuerza con respecto del centro de masa. Realizando nuevamente un análisis físico se encuentran las siguientes ecuaciones para las fuerzas y el torque aplicados al robot.

$$f_{x_I} = u_1 \cos(q_3) - u_2 \sin(q_3), \tag{37}$$

$$f_{y_I} = u_1 \sin(q_3) + u_2 \cos(q_3), \tag{38}$$

$$\tau_z = r_x u_2 - r_y u_1. \tag{39}$$

Conociendo estas expresiones se puede rescribir el vector $\mathbf{q}$ en forma matricial para conocer el modelo que describe al sistema, como se observa en la ecuación (40).

$$\begin{bmatrix} \ddot{q_1} \\ \ddot{q_2} \\ \ddot{q_3} \end{bmatrix} = \begin{bmatrix} 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} \frac{1}{m} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} \begin{bmatrix} \cos(q_3) & -\sin(q_3) \\ \sin(q_3) & \cos(q_3) \\ -r_y & r_x \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{40}$$

De igual forma si se define que:

$$\mathbf{x} = \begin{bmatrix} \mathbf{q} \\ \dot{\mathbf{q}} \end{bmatrix} \rightarrow \dot{\mathbf{x}} = \begin{bmatrix} \dot{\mathbf{q}} \\ \ddot{\mathbf{q}} \end{bmatrix} \tag{41}$$

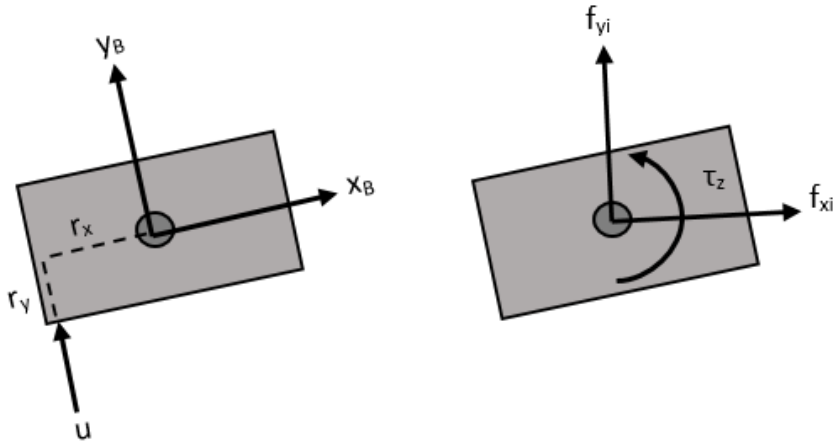


Figura 10: Análisis del modelo inicial con la fuerza impulsional.

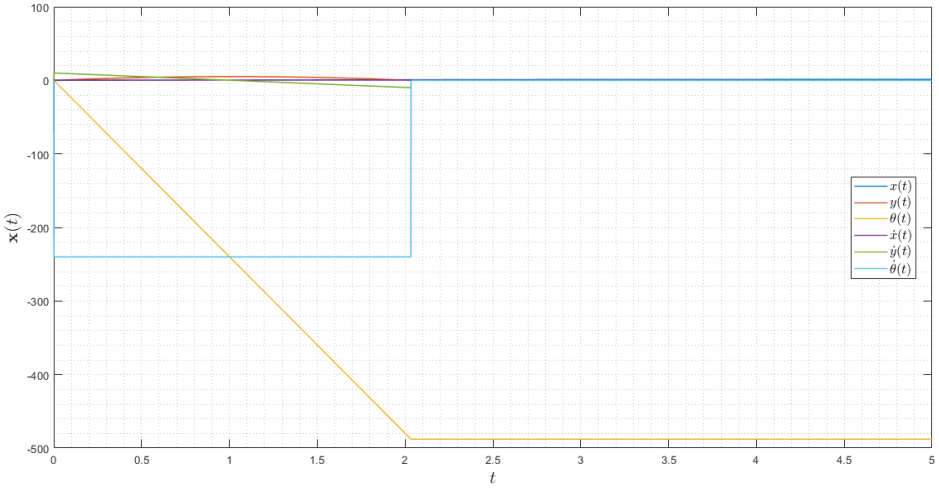
Se encuentra el sistema dinámico no lineal que describe al robot en tiempo continuo en la ecuación (42).

$$\dot{\mathbf{x}} \begin{bmatrix} x_4 \\ x_5 \\ x_6 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -g \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ \begin{bmatrix} \frac{1}{m} & 0 & 0 \\ 0 & \frac{1}{m} & 0 \\ 0 & 0 & \frac{1}{I_{zz}} \end{bmatrix} & \begin{bmatrix} \cos(q_3) & -\sin(q_3) \\ \sin(q_3) & \cos(q_3) \\ -r_y & r_x \end{bmatrix} \end{bmatrix} \begin{bmatrix} u_1 \\ u_2 \end{bmatrix}. \tag{42}$$

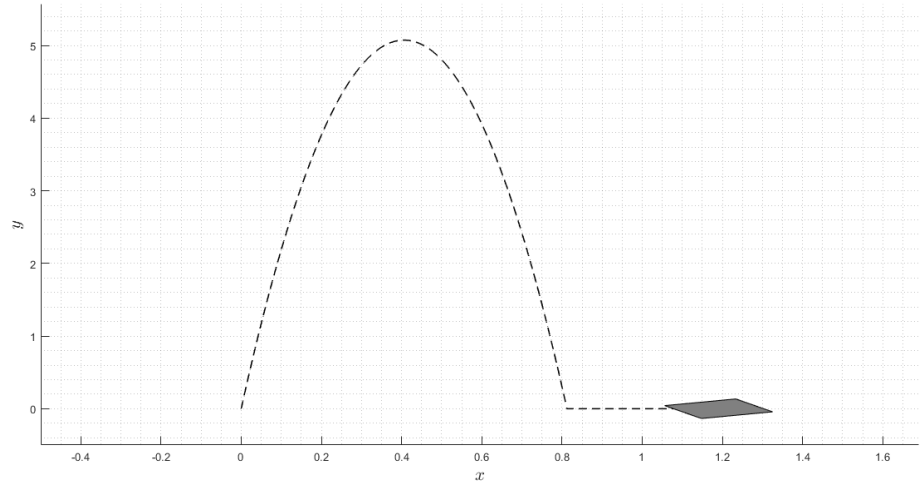
Obteniendo este modelo dinámico, se utilizó como base para realizar las primeras pruebas de este robot, cuyas simulaciones se realizaron en el software de Matlab. Cabe mencionar que este modelo en Matlab se tuvo que modificar, ya que se comportó como un modelo híbrido ya que tiene dos modelos que describen su dinámica. El primer modelo lo describe la dinámica mostrada en la ecuación (42), mientras que el segundo modelo lo describe tanto la primera dinámica como la restricción de movimiento mostrada en la ecuación (43).

$$\begin{cases} x_2 = 0 \\ x_4 = x_4 - bx_4 \\ x_5 = 0 \\ x_6 = 0 \end{cases}. \tag{43}$$

En donde la constante b es un coeficiente de fricción lineal en la dirección de movimiento horizontal. Esta restricción mostrada ayuda a que el robot no caiga infinitamente, sino que se crea un "suelo"simulado donde el robot pueda caer. Inicialmente se realizó la simulación sin un control de orientación; esta simulación se puede observar en la Figura 11 donde se puede apreciar el comportamiento de las variables de estado del modelo dinámico y la trayectoria que realiza el robot al aplicarle una fuerza de impulso en una posición respecto a su centro de masa.



(a) Variables de estado del sistema dinámico.



(b) Trayectoria del robot.

Figura 11: Simulación del robot sin control en Matlab.

De igual forma se realizó una segunda simulación donde se aplicaba un control de orientación mediante un controlador PID. Para poder aplicar este controlador se realizó un análisis donde se podía observar que para poder controlar al robot era necesario aplicar un torque externo para compensar el torque generado por la fuerza impulsional, modificando de esta manera la dinámica del sistema, específicamente la ecuación (36). Esta modificación se muestra en la ecuación (44).

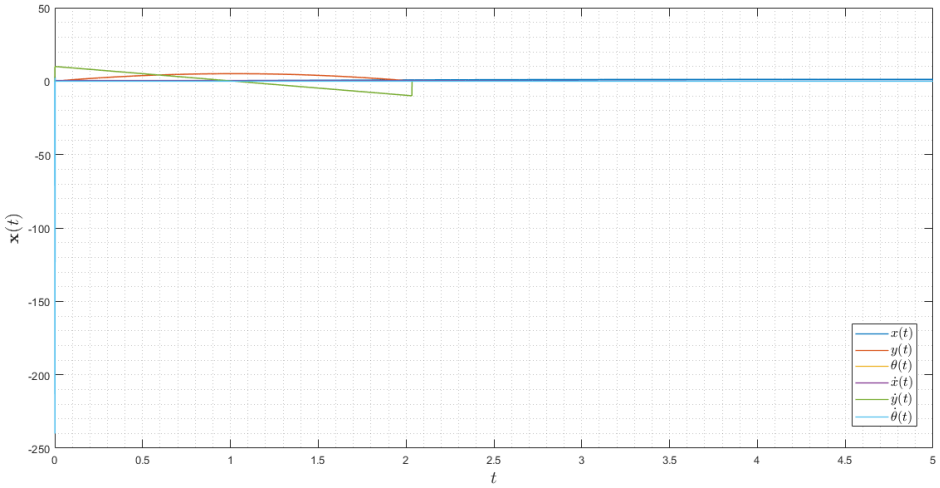
$$\sum \tau_z = I_{zz}\ddot{q}_3 \rightarrow \tau_z - \tau_w = I_{zz}\ddot{q}_3 \rightarrow \ddot{q}_3 = \frac{\tau_z - \tau_w}{I_{zz}}. \tag{44}$$

En esta ecuación se puede observar que se agrega un nuevo torque τ_w, el cuál estaría

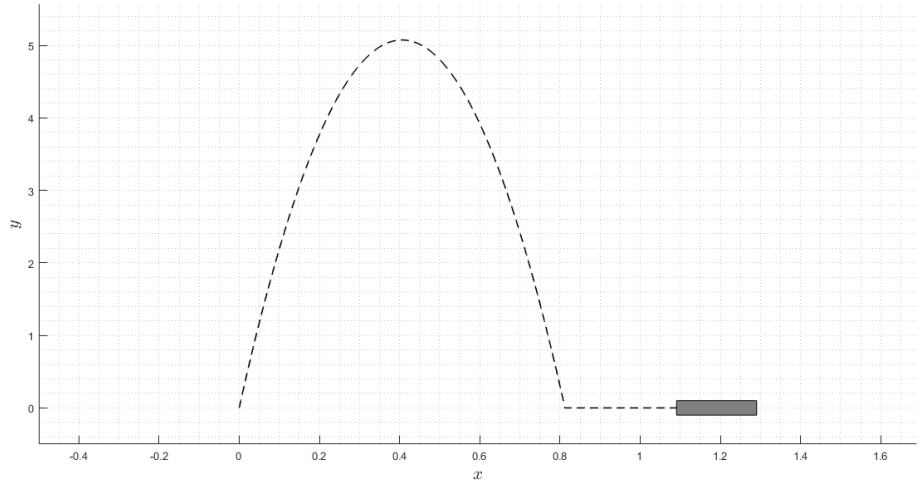
compensando el torque generado por la fuerza impulsional. Este torque se genera aplicando un control PID con el cual se logra corregir la orientación y lograr que tenga una referencia 0. Este resultado se puede observar en la Figura 12.

Para el control PID se utilizó la herramienta *PID Tuner* de Matlab, con la cual se encontraron las constantes del PID. Estas constantes se muestran a continuación:

$$\begin{aligned} K_P &= 0.002, \\ K_I &= 0.0013, \\ K_D &= 7.9039e^{-4}. \end{aligned} \tag{45}$$



(a) Variables de estado del sistema dinámico controlado.



(b) Trayectoria del robot controlado.

Figura 12: Simulación del robot controlado mediante un PID en Matlab.

Observando la Figura 12.a se puede apreciar como un controlador sencillo como lo es el PID fue capaz de llevar la orientación del robot a una referencia de 0. Con este resultado se pudo verificar que el comportamiento del sistema deducido cumple con el comportamiento deseado.

9.2. Modelo inicial con rueda de inercia

En el caso del modelo inicial con la rueda de inercia, se realizó el mismo análisis que se utilizó para el modelo inicial con la diferencia que para este diseño se modeló al robot rectangular con una rueda de inercia; estos dos componentes se comportan como dos cuerpos rígidos unidos. El comportamiento de la rueda de inercia se toma como un torque externo que es capaz de corregir la orientación del robot (parecido al modelo inicial con un control PID). Este nuevo modelo se analizó físicamente utilizando un nuevo diagrama, el cual se muestra en la Figura 13.

En esta figura se puede observar que el centro de masa del robot ahora está definido por los dos cuerpos rígidos, de igual forma se observa un nuevo parámetro, la posición angular de la rueda de inercia, este ángulo está medido con respecto del eje vertical del cuerpo del robot.

Como se mencionó anteriormente para la dinámica de este modelo se utilizó como base el modelo inicial, con diferencia que en este análisis se encuentra un segundo modelo que sería el comportamiendo del robot al encontrarse en el aire. Este comportamiento es el que tiene mayor cambio ya que se está incorporando la rueda de inercia, por otro lado en el instante de aplicar la fuerza impulsional el robot se comporta como el modelo inicial, pero con modificaciones en la masa y la inercia ya que se está agregando la masa y la incercia de la rueda y el motor que realiza el giro de la rueda. Se incluyó el motor para tener un modelo mejor aproximado y obtener mejores resultados.

$$M = m_b + m_w + m_m. \tag{46}$$

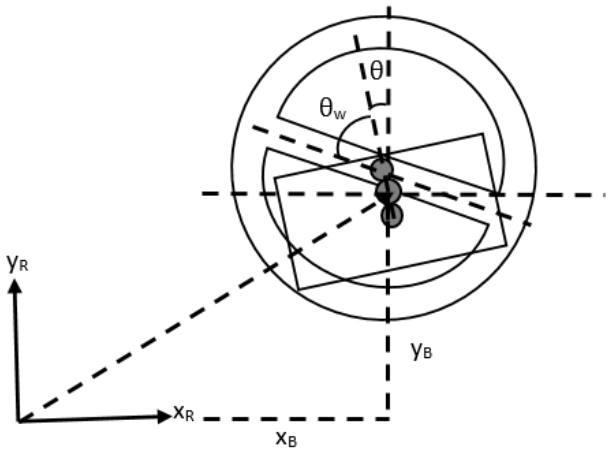


Figura 13: Análisis físico del robot con rueda de inercia.

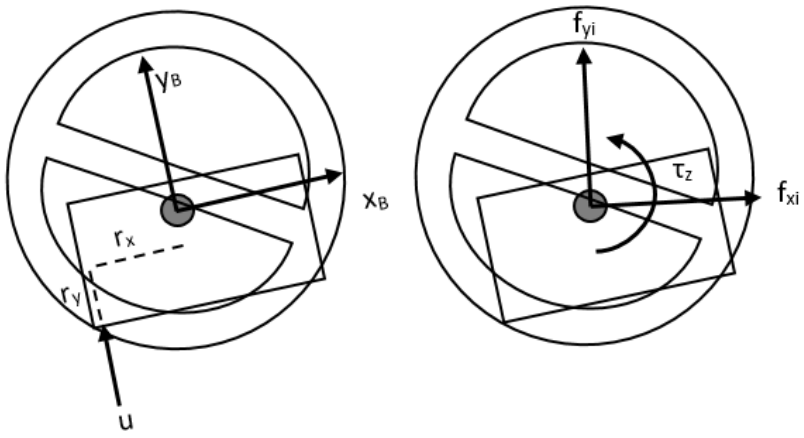


Figura 14: Aplicación de la fuerza impulsional al modelo con rueda de inercia.

$$I = I_b + I_w + I_m. \tag{47}$$

Denotando M como la suma de la masa del cuerpo del robot m_b, la masa de la rueda de inercia m_w y la masa del motor de la rueda de inercia m_m, al igual que I como la suma de la inercia del cuerpo del robot I_b, la inercia de la rueda de incercia I_w y la inercia del robot de la rueda de inercia I_m. De igual forma considerando la Figura 14 y las ecuaciones (34), (35) y (36), las ecuaciones que describen la dinámica del sistema antes y durante la aplicación de la fuerza impulsional son:

$$\ddot{q_1} = \frac{1}{M}(u_1 \cos(q_3) - u_2 \sin(q_3)), \tag{48}$$

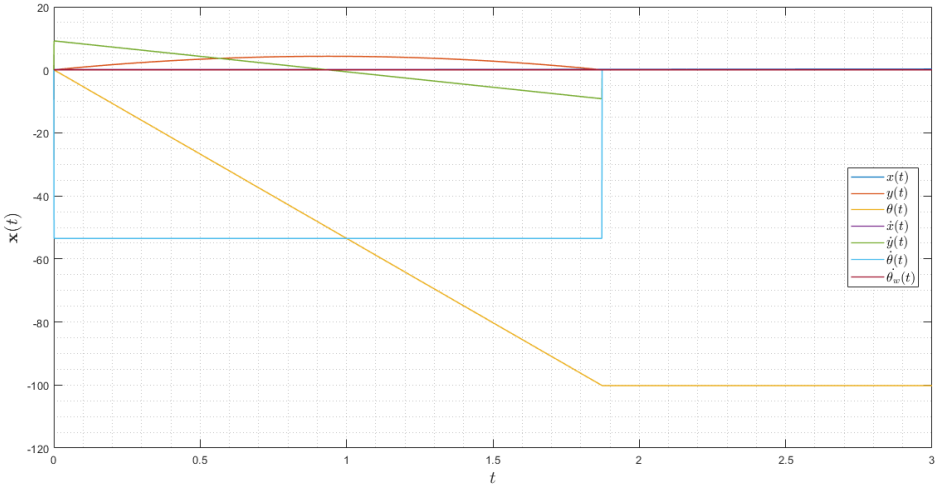
$$\ddot{q_2} = \frac{1}{M}(u_1 \sin(q_3) + u_2 \cos(q_3)) - g, \tag{49}$$

$$\ddot{q_3} = \frac{1}{I}(r_x u_2 - r_y u_1). \tag{50}$$

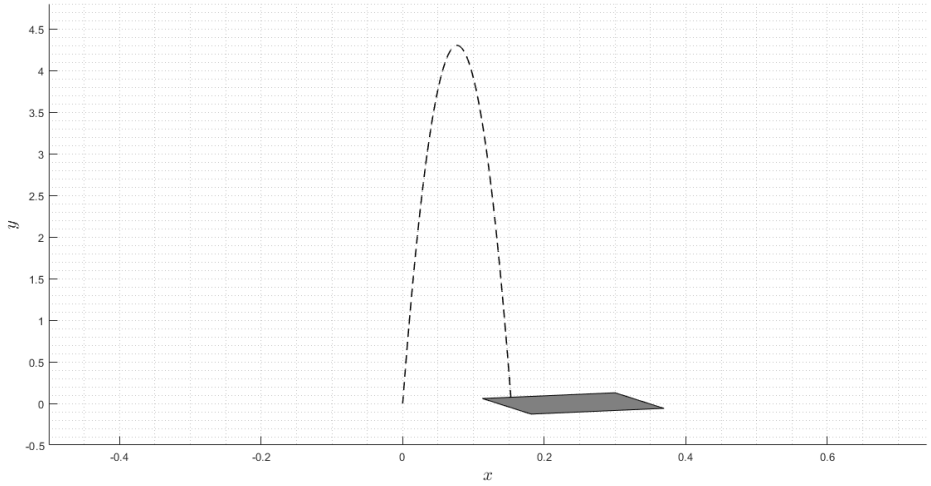
Para poder encontrar el comportamiento del robot en el aire se utilizó la Figura 13 como base, de donde se pueden definir ecuaciones que describan su movimiento angular (el cual se requiere para poder controlarlo). Cabe mencionar que para realizar un modelo mejor aproximado se incluyó el motor que hará girar la rueda de inercia como se mencionó anteriormente. Este análisis se puede observar en las ecuaciones (13) a la (20) en el capítulo de marco teórico.

l_g es la distancia entre el centro de masa del cuerpo del robot y el centro de masa de la rueda de inercia, n_g la relación de transmisión de la caja reductora del motor, c_b el coeficiente de amortiguamiento de la rotación del cuerpo del robot, c_w el coeficiente de amortiguamiento de la rotación de la rueda de inercia y τ es el torque generado por la rueda de inercia. El torque generado por la fuerza impulsional no se toma en cuenta en este análisis ya que este torque afecta al sistema en un instante de tiempo nada más (el instante en el que la fuerza impulsional es aplicada).

Con este modelo encontrado se procedió a realizar pruebas para el sistema utilizando el mismo código de Matlab que se implementó para el modelo inicial, con diferencia en la



(a) Variables de estado del sistema dinámico con rueda de inercia sin control.



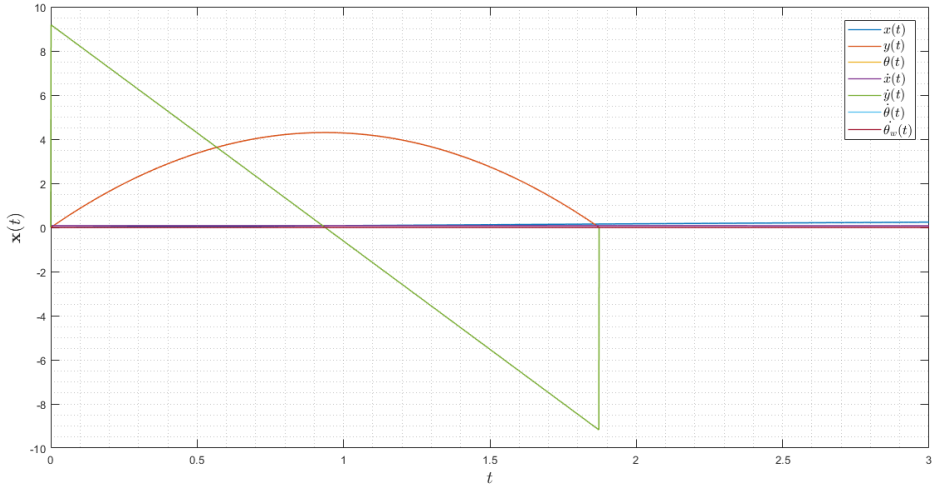
(b) Trayectoria del robot con rueda de inercia sin control.

Figura 15: Simulación del robot con rueda de inercia sin control.

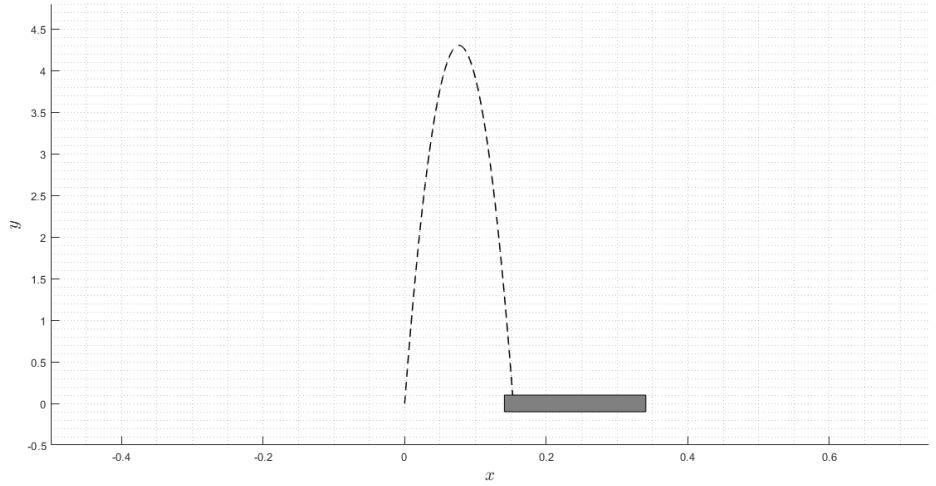
masa del robot y en la inercia del robot para el primer modelo y la implementación del segundo modelo que describe el comportamiento del robot en el aire, cabe mencionar que este segundo modelo se utiliza para actualizar el valor de la orientación del primer modelo del robot.

Como prueba inicial se realizó una prueba sin control, esto se puede visulizar en la Figura 15, donde se observa como el sistema sigue cambiando su orientación en el aire ya que aún no se implementa el torque de la rueda de inercia, el cual únicamente es implementado al momento de querer controlar la orientación del robot.

De igual forma que para el modelo inicial, para este modelo también se realizó una prueba aplicando el control de orientación. Cabe mencionar que al observar el modelo con rueda



(a) Variables de estado del sistema dinámico con rueda de inercia controlado.



(b) Trayectoria del robot con rueda de inercia controlado.

Figura 16: Simulación del robot con rueda de inercia controlado.

de inercia en el aire se puede ver que se cuenta con la matrices de variables de estado, por lo que se aprovechó esta forma del modelado de la dinámica para realizar un control LQR, para el cual como se menciona en el capítulo de marco teórico, se necesitan las matrices **A** y **B** del sistema dinámico, además de las matrices **Q** y **R** para realizar la optimización de la matriz **K**. Para encontrar la matriz **K** de este control LQR se utilizaron los siguientes valores para las matrices **Q** y **R**:

$$\mathbf{Q} = \begin{bmatrix} 100 & 0 & 0 \\ 0 & 100 & 0 \\ 0 & 0 & 100 \end{bmatrix}, \tag{51}$$

$$\mathbf{R} = 0.01. \tag{52}$$

Algo importante que mencionar es que estos valores se encontraron realizando pruebas y ajustando parámetros. Con estos valores conocidos se procedió a utilizar la función **lqr** de Matlab con la cual se encontró la matriz $\mathbf{K_{lqr}}$ (53) y de esta forma se aplicó el control utilizando la ecuación (6) considerando el punto de operación ($\mathbf{x}_{ss}$, $\mathbf{u}_{ss}$) como se muestra en (54).

$$\mathbf{K_{lqr}} = \begin{bmatrix} -61.7 & -1881 & 78.7, \end{bmatrix} \tag{53}$$

$$\mathbf{x}_{ss} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}; \mathbf{u}_{ss} = 0. \tag{54}$$

Para encontrar el valor de $\mathbf{u}_{ss}$ se utilizó la función fsolve de Matlab, donde se resuelve la función que describe la dinámica del sistema evaluada en $\mathbf{x}_{ss}$.

En la Figura 16 se puede apreciar como la orientación logra estabilizarse, ya que implementando el control LQR y la rueda de inercia se logra llegar a la referencia de cero en la orientación, con este primer resultado de control se cuenta con un pequeño indicio que nos demuestra que efectivamente el implementar una rueda de inercia como método de control en un robot que se encuentra en el aire luego de haber saltado, logra estabilizar su orientación.

Algo importante que mencionar es que tanto para el caso del modelo inicial como para el modelo inicial con rueda de inercia, la simulación en Matlab consideró que cuando el robot cae al suelo, tiene un choque inelástico, por lo que cuando cae se desliza horizontalmente hasta que la fricción detenga su movimiento.

9.3. Modelo basado en RecurDyn

Tomando en consideración que las ecuaciones que describen la dinámica del sistema real eran muy complejas, se tomó la decisión de utilizar un software capaz de simular el comportamiento del robot y extraer datos de dicho software con el fin de realizar una identificación de sistemas en Matlab y de esta manera determinar la función de transferencia y las matrices de variables de estado que describan el movimiento del robot.

Para este modelo, se utilizó el diseño CAD mostrado y explicado en el capítulo de diseño. Con este diseño 3D del robot se exportó un archivo .step de Inventor del mismo diseño, con las modificaciones mencionadas de igual forma en el capítulo de diseño, donde se menciona que se eliminan ciertos componentes por dificultades de implementación en RecurDyn. Este archivo .step se importó en el software de RecurDyn, donde se procedió a establecer las restricciones de movimiento y las juntas del robot.

En la Figura 17 se puede apreciar la barra de herramientas del software de RecurDyn, al igual que se encuentran enmarcadas las pestañas que se utilizaron para restringir al robot y obtener un resultado de simulación lo más real posible. Se inició restringiendo el movimiento del robot (movimiento en 2D) para este paso utilizando la pestaña de restricción de movimiento (Figura 19.a), para más información sobre estas restricciones se puede consultar el manual de RecurDyn [23]. Otro procedimiento importante que se realizó fue la relación en el contacto de las piezas con respecto al bloque de tierra de la simulación, en la pestaña de contacto entre elementos (Figura 19.d). Algo que mencionar es que cada vez que se presiona algún botón de esta barra de herramientas el cuadro de *Entity* (Figura 18) cambia y muestra los elementos que se deben seleccionar para colocar el elemento seleccionado. En las Figuras 20.a y 20.b se pueden apreciar las ventanas que aparecen al seleccionar el contacto entre superficies y el contacto entre sólidos, respectivamente. En estas ventanas lo único que se modificó fue el coeficiente de fricción dinámica, donde se investigó la fricción para un suelo de pavimento el cual cuenta con un coeficiente de 0.8 a 1, por lo que se seleccionó un coeficiente de fricción dinámica de 0.8 para considerar el caso donde se cuenta con menos fricción.

Luego de restringir el movimiento del robot se colocaron las juntas revolutas del robot y así generar el movimiento entre eslabones, para esto se seleccionó el botón de *Revolute* y se eligió la posición en la que se quiere que se encuentre cada una (siguiendo lo que pide el cuadro de *Entity*).

Después de colocar las juntas del robot, se procedió a colocar las fuerzas y resortes que van a realizar el movimiento del robot, para esto se selecciona la pestaña de fuerzas rotacionales (Figura 19.c) para colocar los 8 resortes torsionales (o rotacionales en este caso) en la posición deseada al igual que la fuerza rotacional (torque) que ejerce la rueda de inercia. De igual forma se colocó la fuerza axial que ejerce el motor sobre el robot, esta fuerza se selecciona en la pestaña de fuerzas axiales (Figura 19.b).

Para las propiedades que se deben modificar para los resortes rotacionales se puede observar la Figura 21 en la que se modificó únicamente el coeficiente del resorte (se coloca el valor de los resortes seleccionados), al igual que se modificó el coeficiente de amortiguamiento el cual indica a que velocidad se desplaza el resorte por lo que se eligió un amortiguamiento medio para que no fuese tan lento ni tan rápido el movimiento de los resortes (este coeficiente se modificó realizando pruebas ya que no se encontró información de dicha constante).

De igual forma para los exponentes de amortiguamiento y de rigidez se fueron realizando pruebas hasta lograr resultados aproximados a los calculados, para los valores iniciales de estas constantes se consultaron videos en los cuales utilizaban estos tipos de resortes en RecurDyn.

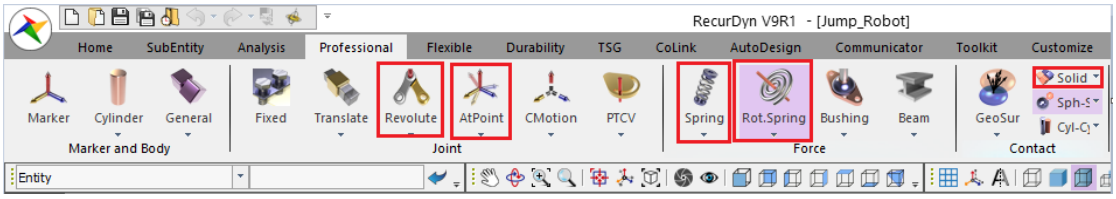


Figura 17: Pestañas utilizadas de la barra de herramientas *Professional* de RecurDyn.

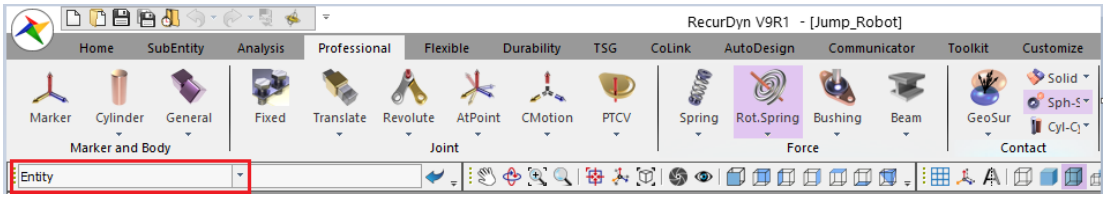
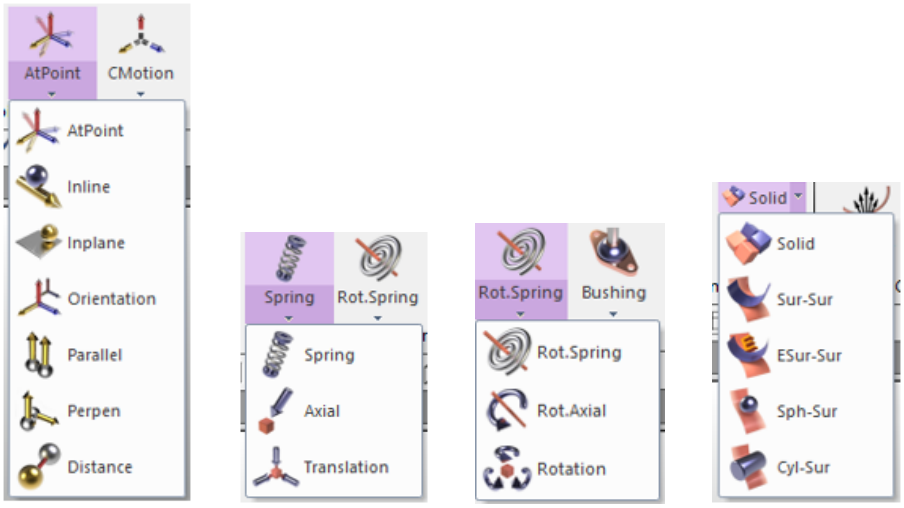


Figura 18: Cuadro de *Entity*.

Para el caso de la fuerza rotacional y la fuerza axial, en la Figura 22 se muestra la ventana de propiedades para ambas fuerzas en la cual al seleccionar el botón de *EL* se despliega una nueva ventana (Figura 23), en esta ventana se pueden agregar expresiones nuevas para cada una de estas fuerzas. En este caso como se realizó conexión con Simulink se debía crear una expresión para cada fuerza como las que se muestran en el cuadro de expresiones de la Figura 22.

Para poder realizar la conexión entre RecurDyn y Simulink fue necesario desplazarse a la pestaña de *Communicator* en donde se muestra una nueva barra de herramientas como la que muestra en la Figura 24. En esta barra de herramientas se inició estableciendo las entradas y las salidas de nuestro sistema, por lo que es necesario presionar el botón de *Plant In* y luego el de *Plant Out*. Al momento de presionarlos se desplegó una ventana como las de las Figuras 25.a y 25.b respectivamente. Para el caso de las entradas sólo fue necesario crear variables con el nombre que se quiera, en el caso de este proyecto se crearon dos variables, una llamada *Fuerza Salto* y la otra *Torque Rueda* estas son la fuerza axial y rotacional respectivamente. Luego de crear estas variables se procedió a realizar el procedimiento descrito para las fuerzas y se seleccionó la función *PIN* y se escribió el nombre de cada variable como argumento.

En el caso de la salida de la planta, se debe escribir una expresión como la que se muestra en la Figura 26, aquí se presionó la pestaña de *velocity* y se seleccionó la funcion *WZ*, ya



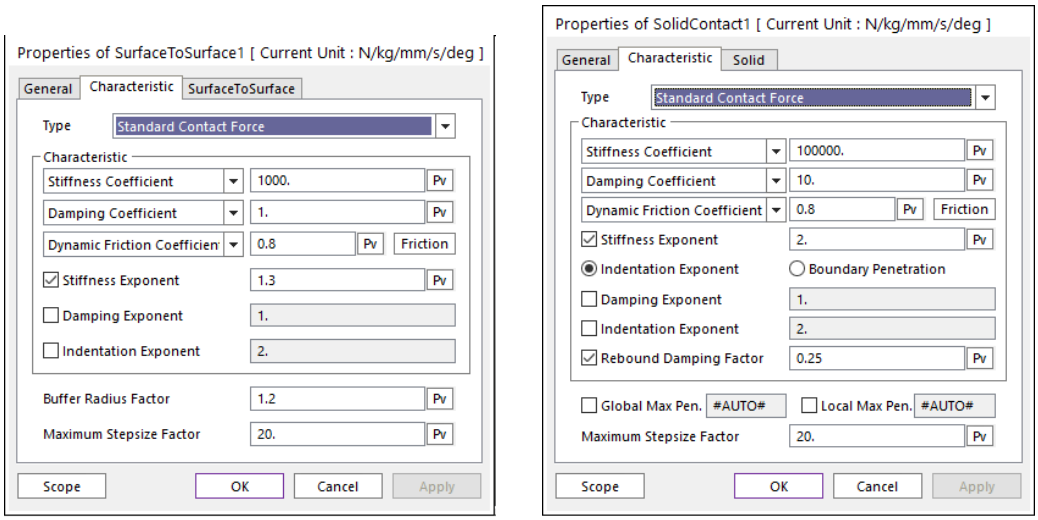
(a) Pestaña de restricción de movimiento.

(b) Pestaña de fuerzas axiales.

(c) Pestaña de fuerzas rotacionales.

(d) Pestaña de contacto entre los elementos.

Figura 19: Menú de cada pestaña utilizada.



(a) Propiedades de contacto entre superficies. (b) Propiedades de contacto entre sólidos.

Figura 20: Propiedades de contacto.

que se requiere como salida la velocidad angular angular del robot con respecto del eje z, y como argumento se seleccionó el punto en el cual se quiere medir la velocidad, para el caso de este proyecto se consideró el centro de masa de la pieza inferior del robot.

Ya que se cuenta con toda esta información se puede procecer a presionar el botón de *Simulink* de la Figura 24, al presionar este botón se despliega una nueva ventana (Figura 27), en esta ventana se debe seleccionar como *Host Program* a Simulink, para poder realizar las simulaciones desde este software y de esta manera poder utilizar la data recibida en Matlab.

El tiempo de muestreo se deja el que aparece por default, se elige un nombre para el archivo .m que genera RecurDyn y se presiona el botón de *Export*. Al presionar este botón se

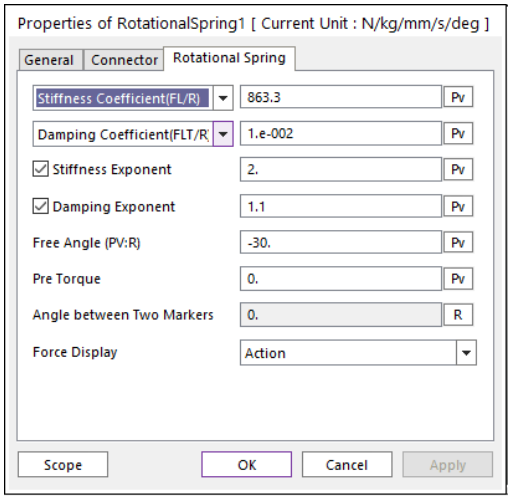
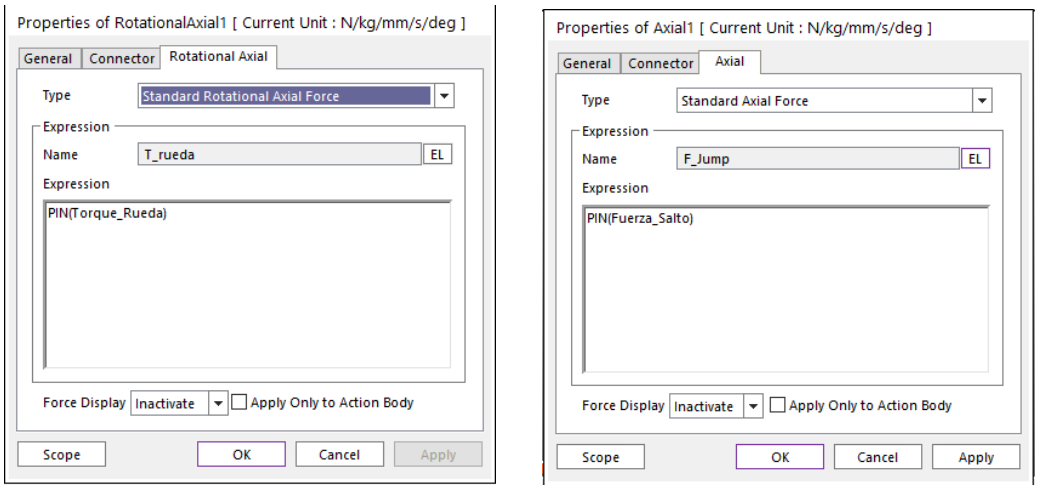


Figura 21: Propiedades de resortes rotacionales.



(a) Propiedades de fuerzas rotacionales.

(b) Propiedades de fuerzas axiales.

Figura 22: Propiedades de fuerzas rotacionales.

crea el archivo en la carpeta en la cual se enuentra el proyecto de RecurDyn. Se presionó *OK* y se pudo procecer a cerrar el sofrware de RecurDyn para iniciar a trabajar desde Simulink, cabe mencionar que cada vez que se realice una simulación desde Simulink, este abre el programa de RecurDyn para poder observar la simulación y luego cierra el programa, luego de esto se puede observar la información extraída y graficarla en un Scope.

En el programa de Matlab se inició colocando en la consola el nombre del archivo .m que generó RecurDyn ya que en él se encontró toda la información del sistema. Al correr este programa se generaron variables en el Workspace de Matlab. Luego de observar que estas variables se encuentran en el Workspace se coloca en la consola la función *rdlib*, la cual abre un archivo .slx (un archivo de Simulink), en el cual se encuentra la planta del sistema

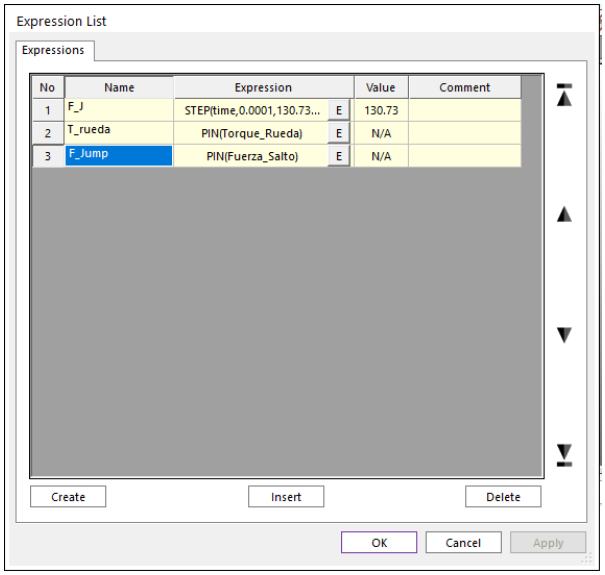


Figura 23: Lista de expresiones para fuerzas.

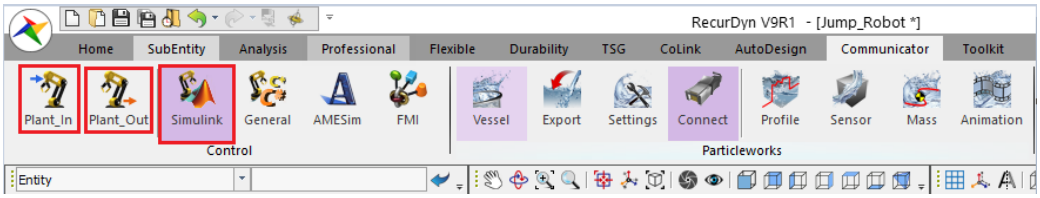


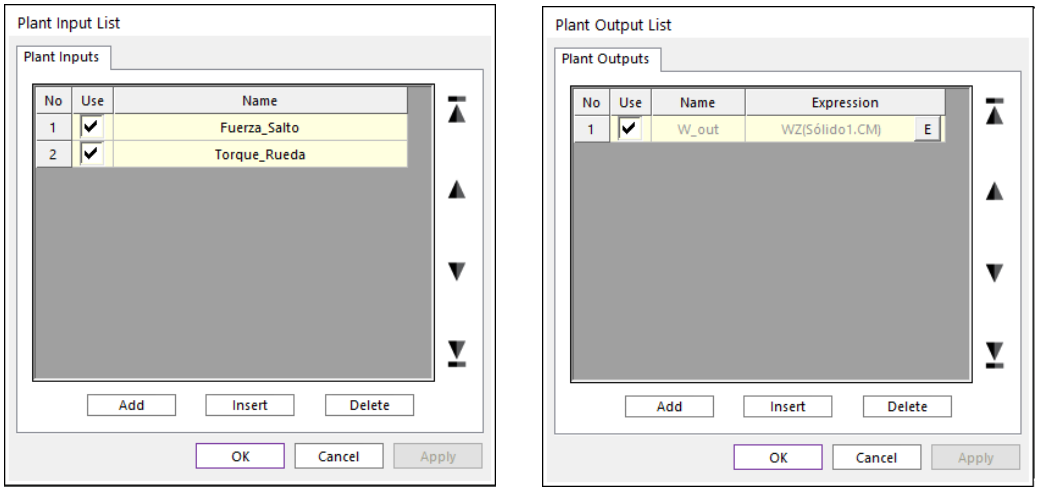
Figura 24: Pestañas utilizadas de la barra de herramientas *Communicator* de RecurDyn.

generado por RecurDyn. Con este bloque se procedió a crear un nuevo modelo de Simulink en el cual se realizó un diagrama para poder simular el robot y extraer la data necesaria para poder generar nuestra función de transferencia. Este diagrama se muestra en la Figura 28.

En este diagrama se puede observar la planta que genera RecudDyn, dos señales, una es el step para la fuerza impulsional y el otro step es un torque continuo para poder generar la función de transferencia. De igual forma se muestra un scope para poder observar las gráficas que genera el step y la velocidad angular (Figura 29) y se colocaron dos bloques que generan dos archivos .mat en donde se guarda una variable en cada uno con la data de la señal de entrada (torque continuo) y la señal de salida (velocidad angular del robot).

Con estas dos variables mencionadas anteriormente se procedió a utilizar la *Sytem Identification Toolbox* de Matlab, donde se importó la data en dominio del tiempo y se realizó una estimación para la función de transferencia (Ecuación (55)), especificando que se requiere un modelo sencillo de dos polos y cero ceros ya que para un sistema lineal que logre describir la velocidad del robot adecuadamente, es necesario tener una función de transferencia de orden 2.

$$G_{vel}(s) = \frac{4.203 \times 10^5}{s^2 + 15.42s + 2.415 \times 10^4} \tag{55}$$



(a) Propiedades de las entradas de la planta. (b) Propiedades de las salidas de la planta.

Figura 25: Lista de entradas y salidas para la planta del sistema.

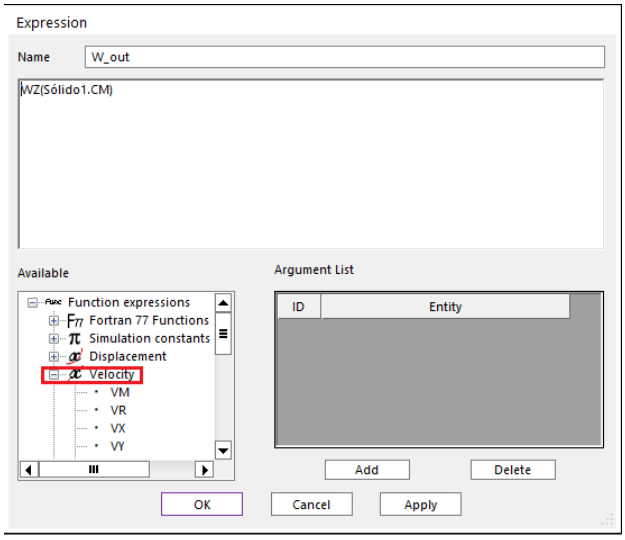


Figura 26: Expresión para la salida del sistema.

Cabe mencionar que esta función de transferencia es para la velocidad angular del robot, mientras que lo que se necesita es una función de transferencia para la posición, por lo que se procedió a integrar en el dominio de la frecuencia y tener como resultado la ecuación (56).

$$G_{pos}(s) = \frac{4.203 \times 10^6}{s^3 + 15.42s^2 + 2.415 \times 10^4 s} \tag{56}$$

En la Figura 30 se pueden observar los resultados de aplicar un escalón a las funciones

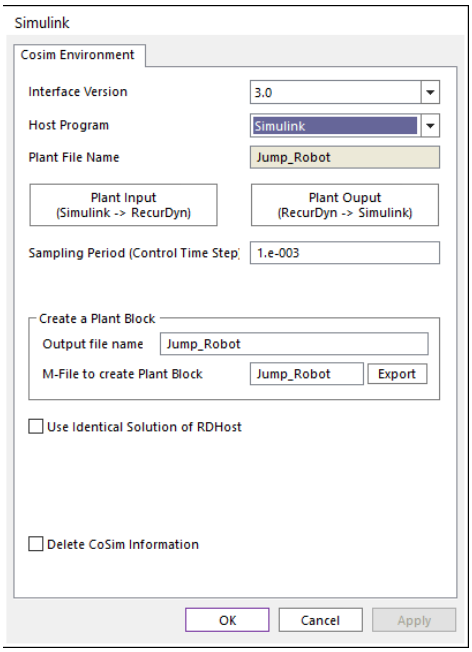


Figura 27: Propiedades para la conexión con Simulink

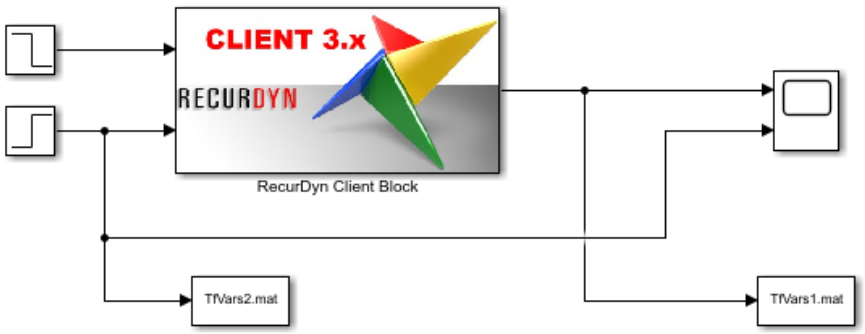


Figura 28: Diagrama de bloques en Simulink.

de transferencia G_{vel} y G_{pos} donde se puede observar como la velocidad tiende a un valor constante, mientras que la posición crece infinitamente, por lo que la integral se realizó bien ya que la posición tiene una pendiente constante y la velocidad tiende a un valor constante.

Con el fin de verificar que se puede aplicar un controlador sencillo, como lo es el controlador PID, a la función de transferencia G_{pos}, se realizó un análisis con la aplicación *PID Tuner* de Matlab, con la cual modificando algunos parámetros se obtuvo el resultado de la Figura 31. Observando este resultado se pudo concluir que un controlador PID era capaz de controlar la orientación del robot, por lo que esta fue la base para iniciar a diseñar el controlador para el robot.

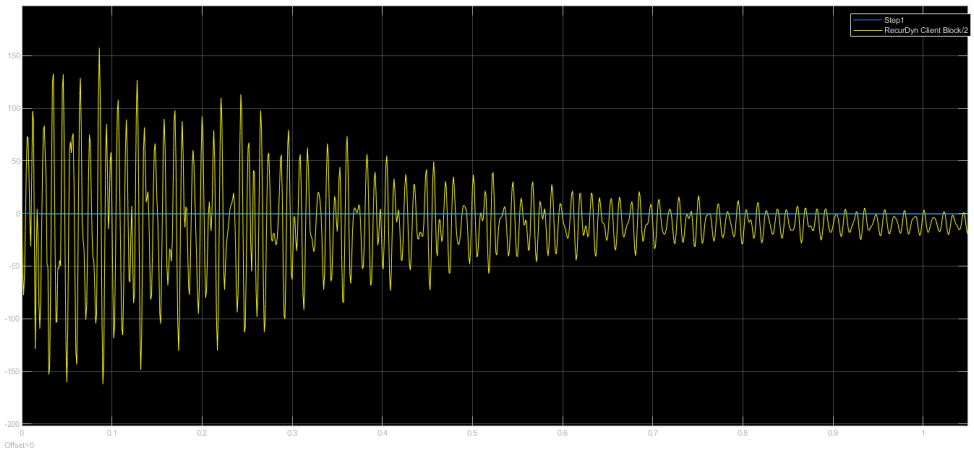
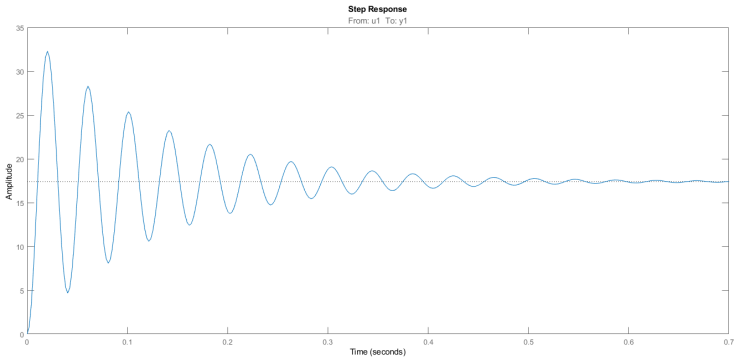
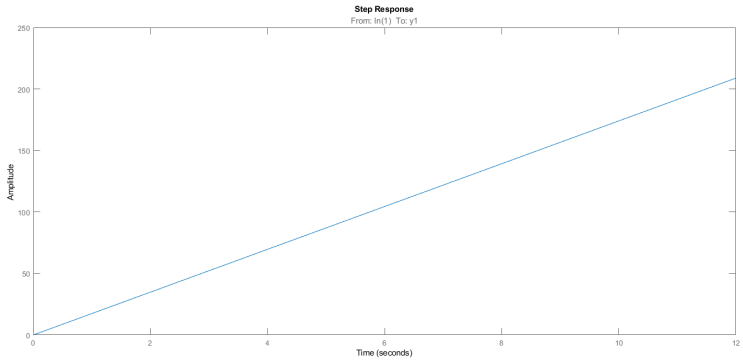


Figura 29: Respuesta al escalón en Simulink.



(a) Respuesta al escalón de G_{vel}.



(b) Respuesta al escalón de G_{pos}.

Figura 30: Respuesta al escalón para las funciones de trasferencia generadas.

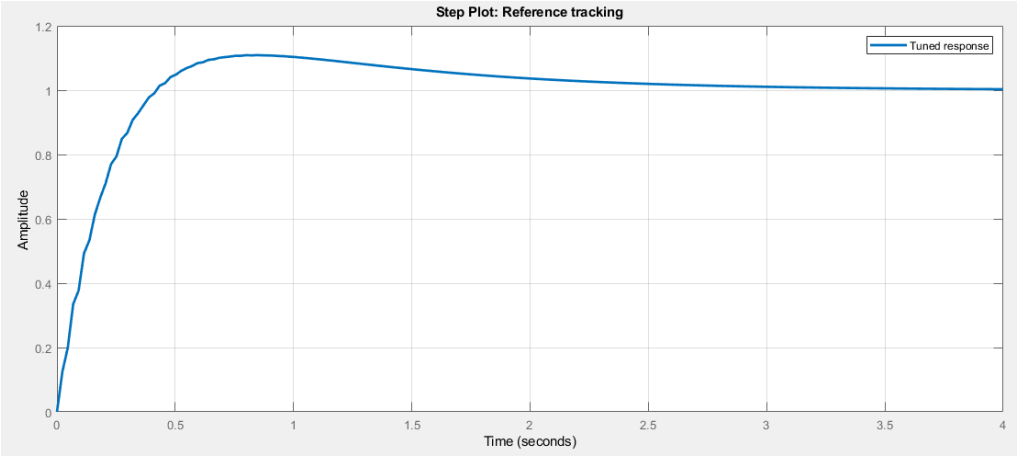


Figura 31: Controlador PID para la función de transferencia G_{pos}.

CAPÍTULO 10

Control del sistema

10.1. Diseño de la rueda de inercia

Para el diseño de la rueda de inercia se había tomado la decisión de utilizar una rueda de 12cm de diámetro con agujeros que disminuyeran el peso de la misma y un grosor redondeado de 2cm fabricada de PLA, el problema con esta rueda de inercia se presentó al momento de querer realizar las simulaciones en RecurDyn ya que al observar el comportamiento del robot, la rueda de inercia no lograba girar de forma correcta. Al realizar las primeras pruebas se observó que el primer problema se encontraba en el ángulo con el que contaba el robot, ya que la gravedad provocaba que la rueda de inercia se colocará de forma vertical y no girara correctamente. Por esta razón se tomó la decisión de modificar el diseño del robot y quitarle el ángulo, por lo que el robot es completamente vertical.

Con la modificación de colcocar al robot completamente vertical se resolvió el problema del giro de la rueda, pero se presentó un nuevo problema el cual era que al girar la rueda de inercia al máximo de velocidad que podía entregar el motor, esta no era capaz de vencer el peso del robot y hacerlo girar, por lo que se realizó una nueva modificación. Esta modificación consistió en cambiar de material la rueda de inercia, con base a lo que se puede observar en [8] la rueda de inercia está fabricada de un metal, por lo que se decidió utilizar acero ya que es un metal fácil de conseguir en Guatemala. Por otro lado se decidió reducir el grosor de dicha rueda a 2mm y dejar el diámetro de 12cm, al realizar pruebas se observó que efectivamente al hacer girar la rueda con un torque de 50Nmm se lograba hacer girar al robot estando parado. Con este primer resultado se decidió realizar una segunda prueba con una rueda de diámetro de 10cm y un grosor de 2mm con el fin de lograr que el robot se comprimiera lo mayor posible sin que la rueda topara con el suelo. En esta nueva prueba se utilizó un torque de 80Nmm donde se lograba hacer girar al robot, por lo que se decidió quedarse con este diseño de la rueda de inercia. La rueda se puede apreciar en la Figura 32.

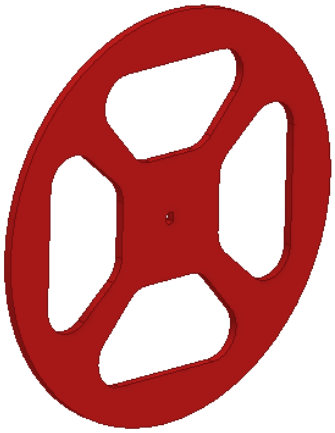


Figura 32: Diseño de la rueda de inercia.

10.2. Control PID

Al utilizar la aplicación *PID Tuner* se logró el resultado de la Figura 31 donde se aprecia que efectivamente se puede controlar al sistema, el incoveniente de utilizar las constantes que brinda esta aplicacion fue que el tiempo de asentamiento fue mayor al tiempo que tarda el robot en aterrizar que es de 1.05 segundos, esto implicó que el robot no se lograra controlar a tiempo, por esta razón se tomó la decisión de utilizar nuevas constantes para mejorar el tiempo de subida y el tiempo de asentamiento del robot y lograr que este se estabilizara antes de llegar a la altura máxima del salto, en aproximadamente 0.47 segundos.

Para realizar el diseño del controlador PID se realizaron pruebas y observaciones de la simulación al aplicarle diferentes valores para las constantes del PID basadas en el Cuadro 2, ya que no hay una forma para realizar el cálculo de estas constantes. Donde se observó el mejor resultado para el siguiente valor de constantes:

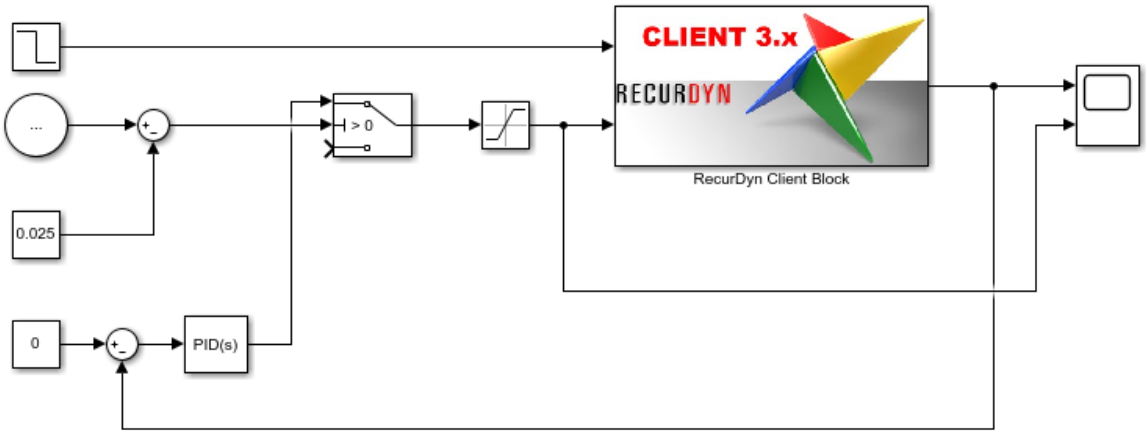


Figura 33: Diagrama de bloques del sistema con control PID.

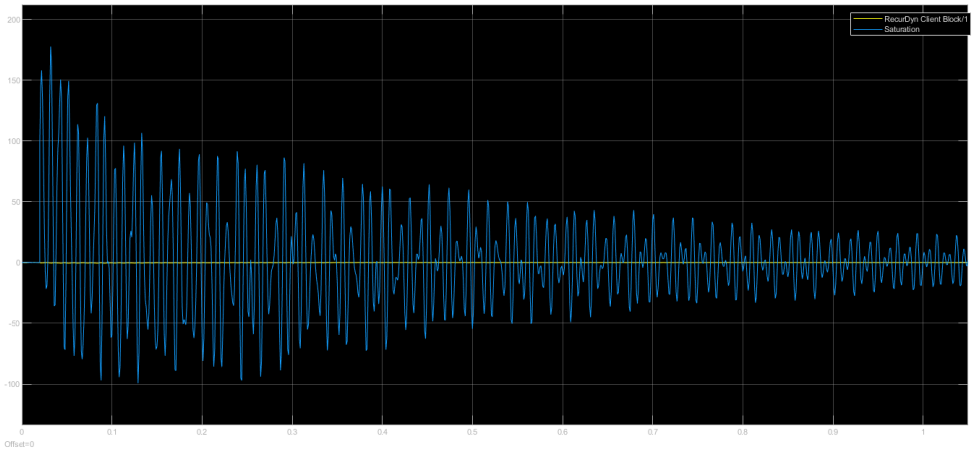


Figura 34: Orientación del robot y torque de control.

$$K_P = 35, \qquad K_I = 0.8, \qquad K_D = 5.$$

De igual forma en la Figura 33 se puede observar el diagrama de bloques utilizado en Simulink para realizar las simulaciones y pruebas del controlador. En este diagrama se colocó un bloque de tiempo para poder aplicar el controlador en el instante que el robot despega del suelo, al igual que se colocó un bloque de límites de saturación, este bloque se colocó con el fin de evitar que el torque que necesite el sistema exceda el que puede brindar el motor de la rueda de inercia.

Como resultado del controlador se obtuvo lo mostrado en la Figura 34 donde se puede apreciar como la orientación (señal en color amarillo) logra controlarse y el robot se mantiene con un ángulo aproximado a cero, de igual forma se muestra el torque suministrado por el control (señal de color azul) el cual no excedió el que el motor puede brindar. Con el fin de comprobar que el punto de operación utilizado para realizar la linealización era válido, se extrajeron los datos del torque generado por el PID en donde se observara que permaneciera

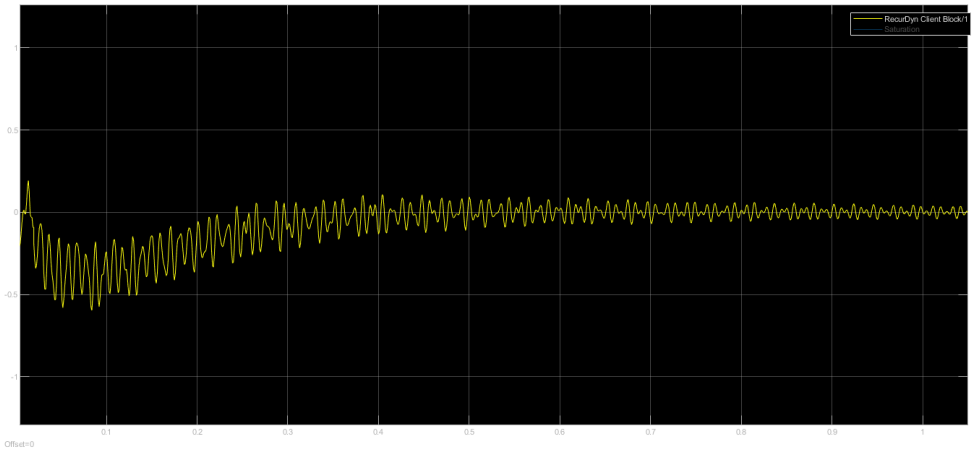


Figura 35: Orientación del robot controlada utilizando control PID.

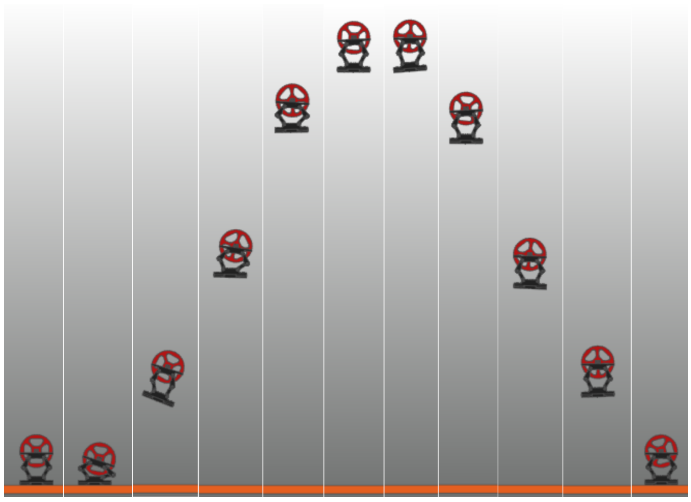


Figura 36: Fotogramas de la trayectoria del robot con control PID.

estable, en la Figura 34 se puede observar que esto sucedió entre aproximadamente 0.25 segundos y 1.05 segundos. Con estos datos se calculó la media para el cual el resultado fue que el torque en estado estable era de -0.6Nmm, con lo cual se comprobó que el haber utilizado un punto de operación de -0.5Nmm nos aseguró una linealización efectiva del sistema.

En la Figura 35 se puede observar más de cerca la posición angular controlada del robot, donde se aprecia que cuenta con un tiempo de asentamiento de 0.4 segundos, lo que significa que logra estabilizarse poco antes de llegar a la altura máxima. Con este resultado se pudo decir que un control PID era efectivo para controlar la orientación del robot luego de saltar.

Por otro lado, en la Figura 36 se observa la trayectoria que realiza el robot en la simulación al aplicarle un control PID, al igual que se observa como logra estabilizar su orientación al encontrarse en el aire.

10.3. Control LQI

Como segunda prueba de control se realizó un control moderno, utilizando el control Lineal Cudrático Integral (LQI por sus siglas en inglés). Como se describe en la sección del marco teórico, para este controlador se necesitan las matrices de variables de estado y las matrices $\mathbf{Q}$ y $\mathbf{R}$ y así determinar el valor de la matriz $\mathbf{K}_{lqi}$. Para encontrar las matrices de variables de estado se tomó como primera opción trasformar la función de transferencia a un sistema de espacio de estados, el inconveniente con esta solución era que al momento de tener el sistema en espacio de estados no se conoce el orden de las variables de estado, y ya que es necesario conocer cual es la variable de estado de la posición angular para poder contrarla esto era un problema. Por lo que se decidió utilizar las matrices de la ecuación (17) que describen el comportamiento de un robot que salta con rueda de inercia. Con estas matrices de variables de estado y utilizando las matrices $\mathbf{Q}$ y $\mathbf{R}$ de la ecuación (57) se encontró la matriz $\mathbf{K}_{lqi}$ de la ecuación (58).

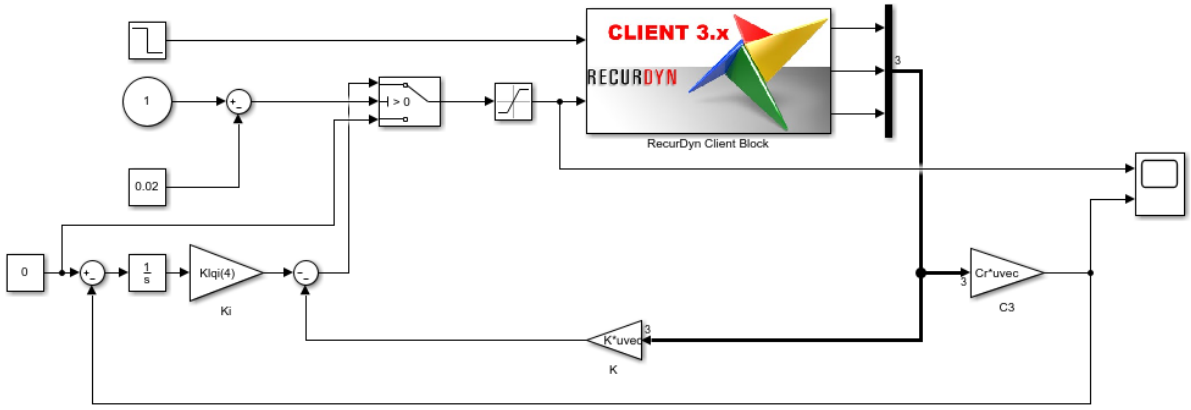


Figura 37: Diagrama de bloques del sistema con control LQI.

$$\mathbf{Q} = \begin{bmatrix} 810 & 0 & 0 & 0 \\ 0 & 0.7 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 2 \end{bmatrix}, \qquad \mathbf{R} = 0.1, \tag{57}$$

$$\mathbf{K}_{lqi} = \begin{bmatrix} 90.7357 & 8.0977 & 0.0014 & -4.4721. \end{bmatrix} \tag{58}$$

Con esta matriz conocida se procedió a realizar un diagrama de bloques en Simulink (Figura 37) para realizar las simulaciones y las pruebas del controlador. Ya que para este control es necesario conocer tres variables de estado: la posición y velocidad angular del robot y la velocidad angular de la rueda de inercia, en RecurDyn se colocaron como salidas de la planta estas tres variables. En el diagrama se puede observar que la parte inferior es la que describe al controlador LQI. De igual forma que para el PID se utilizó un bloque de saturación para evitar que el torque suministrado al sistema fuese mayor al que el motor puede entregar.

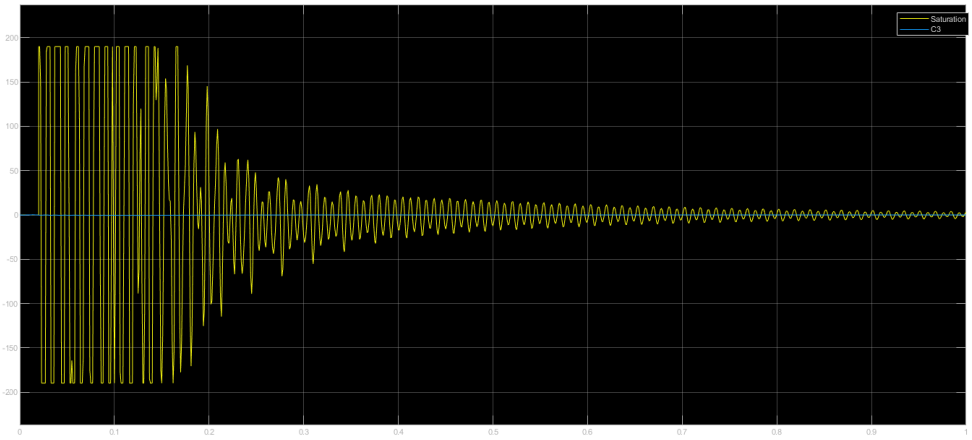


Figura 38: Orientación del robot y torque de control.

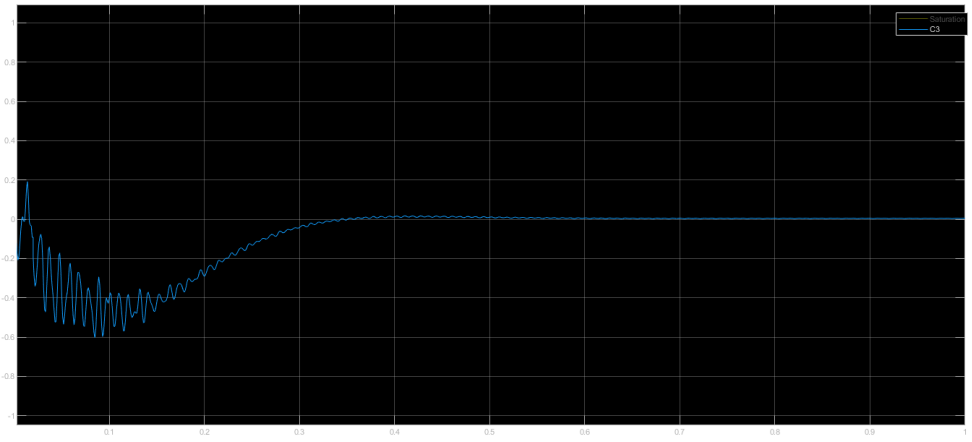


Figura 39: Orientación del robot controlada utilizando control LQI.

Como resultado del controlador se obtuvo lo que se muestra en la Figura 38, donde se puede apreciar cómo la orientación (señal en color azul) logra controlarse y el robot se mantiene con un ángulo aproximado a cero, de igual forma se muestra el torque suministrado por el control (señal de color amarillo). Como se observa al inicio el torque se satura, por lo que fue una buena idea el haber colocado el bloque de saturación y así evitar que el motor se sobre esfuerce.

Por otro lado en la Figura 39 se puede apreciar como la orientación logra ser cero en un tiempo aproximado de 0.45 segundos, por lo que utilizar un control LQI es otra buena opción para el robot ya que este control es capaz de controlar la orientación del robot y logra que este aterrice de pie.

De igual forma en la Figura 40 se observa una secuencia de imágenes que muestran la trayectoria que realiza el robot y cómo logra estabilizar su orientación utilizando el control LQI.

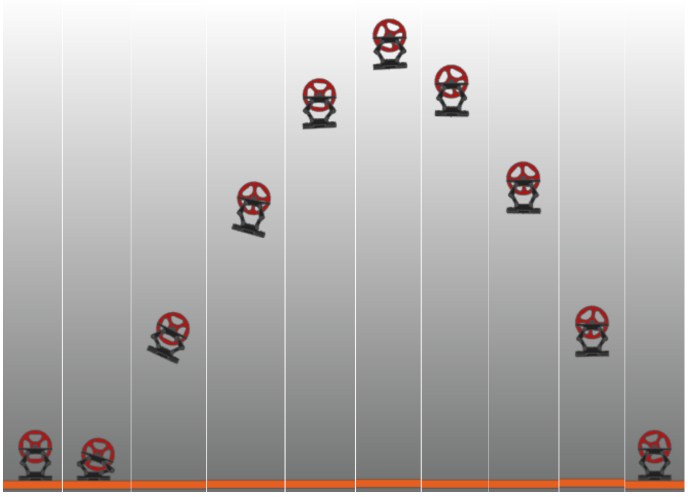


Figura 40: Fotogramas de la trayecctoria del robot con control LQI.

Observando y comparando las Figuras 35 (control PID) y 39 (control LQI), se puede apreciar que el control PID logró llegar a la posición cero antes que el control LQI, la diferencia es que el control LQI es capaz de estabilizar al robot de una mejor manera que el control PID ya que este no presentó las oscilaciones cerca del punto de referencia, por lo que se podría decir que la mejor forma de controlar al sistema es un control LQI. De igual forma se puede discutir el recurso de software, ya que un control LQI requiere de una mayor cantidad de recursos computacionales que un control PID, aunque el tema de recursos computacionales no es tan relevante en esta implementación por lo que en cuanto a recursos computacionales ambos controladores son útiles y eficientes. Con estas dos características tomadas en cuenta se pudo decidir que el controlador LQI es una mejor opción para la implementación.

10.4. Pruebas en terreno irregular

Para realizar las pruebas en un terreno irregular, se modificó el suelo en el cual se encontraba el robot en RecurDyn. Se inició realizando un *Outline* con triángulos de altura de 10mm y base de 200mm. Con este *Outline* se creó una superficie para luego extruirla y poder crear el suelo del robot. Para estas pruebas se pretendió verificar el funcionamiento tanto del robot como de los controladores y así poder validarlo.

Se inició realizando pruebas con el control PID para el cual se puede observar en la Figura 41 como el controlador fue capaz de estabilizar la orientación incluso en los últimos dos saltos que son donde el robot cae en el terreno irregular y vuelve a saltar. Para una mejor visualización de esta simulación se puede consultar el siguiente enlace (https://youtu.be/W5M7zyR0f7s), este enlace dirige a un vídeo en el cual se puede observar la simulación del robot en un terreno irregular con control PID.

Luego de esto se procedió a realizar una prueba para el control LQI de igual forma en un terreno irregular. Para esta prueba se obtuvo el resultado de la Figura 42. Al igual que para el controlador PID se puede observar como el control es capaz de controlar la posición angular incluso en los últimos dos ciclos, que son donde el robot aterriza y salta nuevamente. Para una

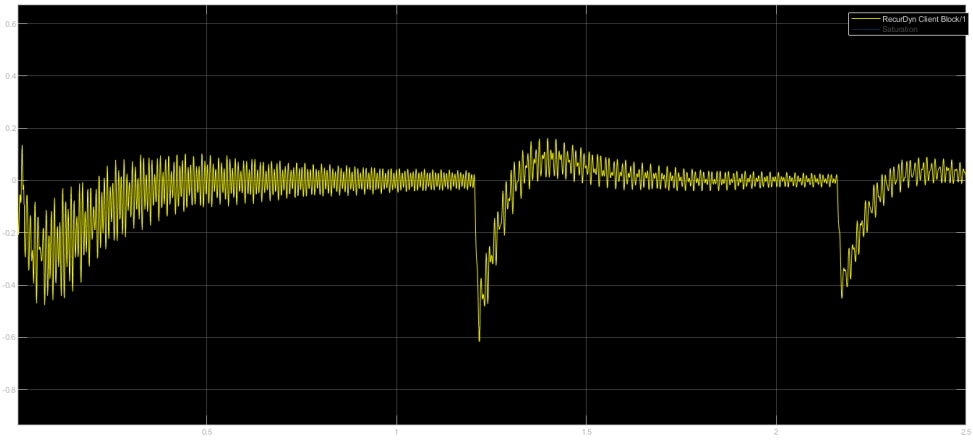


Figura 41: Posición angular controlada con un PID en terreno irregular.

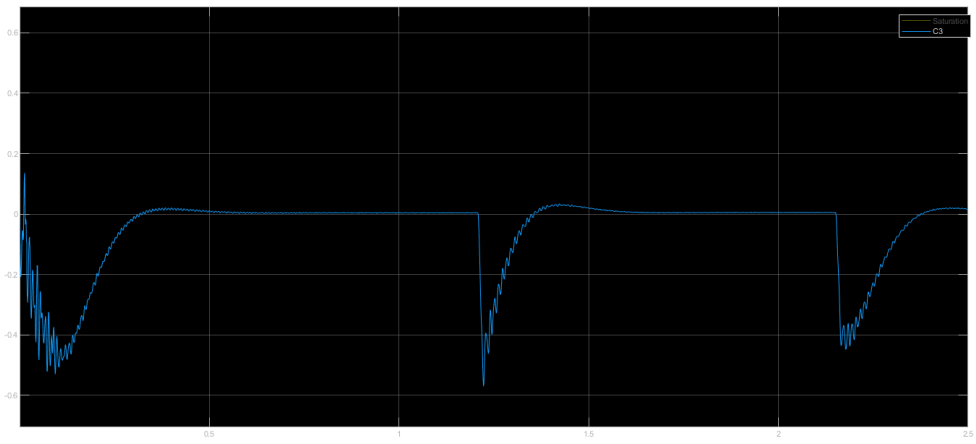


Figura 42: Posición angular controlada con un LQI en terreno irregular.

mejor visualización se puede observar el siguiente enlace (https://youtu.be/dd19kVyJfh4) que dirige al vídeo de la simulación del robot con control LQI en un terreno irregular.

Algo importante que mencionar es que en base a lo observado en las simulaciones se puede concluir que no importa en que parte del suelo aterrice el robot, al volver a saltar el control es capaz de controlar la orientación angular del robot, la única diferencia que existe del lugar en el que caiga el robot es la dirrección en la cual realizará el siguiente salto.

CAPÍTULO 11

Conclusiones

- El robot es capaz de alcazar una desplazamiento horizontal de 1.65m y a su vez una altura de 1.45m, lo que a su vez muestra que es capaz de saltar 10 veces su altura.
- Una rueda de inercia de acero de 10cm de diámetro y 2mm de espesor, controlada por un control PID, es capaz de compensar el torque generado por la fuerza de salto y a su vez controlar la posición angular del robot para lograr un aterrizaje de pie.
- Al encontrarse en un terreno irregular el robot aterriza, salta en dirección perpendicular al suelo y es capaz de controlar su orientación nuevamente.
- Según los resultados, se puede observar que un control LQI es más preciso al momento de estabilizar la orientación del robot.
- Ya que el sistema es capaz de controlarse se puede decir que el modelo lineal que se encontró utilizando RecurDyn y Matlab para el control PID y el sistema de un robot con rueda de inercia del capítulo de marco teórico para el control LQI se apega al modelo no lineal del robot.

CAPÍTULO 12

Recomendaciones

- Manufacturar y armar al robot, realizar la programación con uno de los microcontroladores mencionados y realizar pruebas con el robot en físico, tanto pruebas de esfuerzo como observaciones del mecanismo de salto y el movimiento del robot. A pesar que las simulaciones son muy cercanas a la realidad, no es lo mismo que observar el comportamiento del robot en físico.
- Contar con mejores recursos en software tanto computacionales como programas con los cuales se pueda generar un modelo no lineal del sistema, y de esta forma encontrar un modelo más certero del robot ya que con este se podría colocar el diseño del robot completo y no quitar algunas piezas como se hizo para este proyeto. Con este modelo más apegado al sistema del robot se pueden realizar diferentes pruebas con el robot mientras se encuentra en el aire y observar diferentes comportamientos del mismo.
- Con el robot manufacturado, se puede investigar más a cerca de los dispositivos electrónicos que se pueden implmentar, este es el caso de las baterías ya que las seleccionadas cumplen con el amperaje y voltaje que necesita el robot pero podría diseñarse algún circuito que pueda eficientar la energía eléctrica del robot.

CAPÍTULO 13

Bibliografía

[1] J. Koh, S. Jung, M. Noh, S. Kim y K. Cho, "Flea inspired catapult mechanism with active energy storage and release for small scale jumping robot", en *2013 IEEE International Conference on Robotics and Automation*, 2013, págs. 26-31.

[2] W. Liu, F. Li, X. Fu, C. Stefanini, G. Bonsignori, U. Scarfogliero y P. Dario, "Jumping Like an Insect: From Biomimetic Inspiration to a Jumping Minirobot Design", en. ene. de 2013, págs. 207-221. DOI: 10.1007/978-1-4419-9985-6_11.

[3] J. Zhao, J. Xu, B. Gao, N. Xi, F. J. Cintrón, M. W. Mutka y L. Xiao, "MSU Jumper: A Single-Motor-Actuated Miniature Steerable Jumping Robot", *IEEE Transactions on Robotics*, vol. 29, n.º 3, págs. 602-614, 2013.

[4] G. Jung, C. S. Casarez, J. Lee, S. Baek, S. Yim, S. Chae, R. S. Fearing y K. Cho, "JumpRoACH: A Trajectory-Adjustable Integrated Jumping–Crawling Robot", *IEEE/ASME Transactions on Mechatronics*, vol. 24, n.º 3, págs. 947-958, 2019.

[5] P. Castillo, *Construcción de un Mecanismo Bio Inspirado Para Auto Volteo de Vehículos Terrestres*, 2019.

[6] J. Zhao, T. Zhao, N. Xi, F. J. Cintrón, M. W. Mutka y L. Xiao, "Controlling aerial maneuvering of a miniature jumping robot using its tail", en *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2013, págs. 3802-3807.

[7] K. D. Kumar, N. Abreu, M. Sinha y col., "Fault-tolerant attitude control of miniature satellites using reaction wheels", *Acta Astronautica*, vol. 151, págs. 206-216, 2018.

[8] Y. Nomura y J. Ishikawa, "Aerial attitude control of hopping robots using reaction wheels: evaluation of prototype II in the air", en *International Conference on Advanced Engineering Theory and Applications*, Springer, 2017, págs. 361-372.

[9] J. Vagas y W. A. Pinzón, "Implementación de Sistemas de Control a partir de Modelos Sistemo-Dinámicos Implementation of Control Systems from Dynamics Models",

[10] C. García Benítez GermánVera Estrada, "Un modelo de sistema dinámico híbrido utilizando el enfoque de la lógica difusa", Español, *CIENCIA ergo-sum, Revista Científica Multidisciplinaria de Prospectiva*, 2010, ISSN: 1405-0269. dirección: https://www.redalyc.org/articulo.oa?id=10413200007.

[11] M. B. O. Moctezuma, *Sistemas dinámicos en tiempo continuo: Modelado y simulación*. OmniaScience, 2015.

[12] J. Cánovas, *Métodos numéricos para las ecuaciones diferenciales*. dirección: http://www.dmae.upct.es/~jose/metodos/numer4.pdf.

[13] M. H. Jonhson Micheal A. y Moradi, *PID Control: New Identification and Design Methods"*. Springer-Verlag London, 2005.

[14] M. J. Rodríguez Francisco y López, *Control Adaptativo y Robusto*. Secretariado de Publicaciones de la Universidad de Sevilla, 1996.

[15] J. D. Martinez Velasco, L. T. Poveda Galvis y col., "Diseño e Implementación de un Control Òptimo LQR con la Tarjeta Raspberry Pi",

[16] Z. Feng, J. Zhu y R. Allen, "Design of LQI Control Systems with Stable Inner Loops", *Journal of Shanghai Jiao-tong University (Science), Vol. E*, págs. 2-12, 2007.

[17] P. Poubeau, *Inertia wheel*, US Patent 4,211,452, jul. de 1980.

[18] Vanel, *Catalogue of Torsional Springs*. dirección: https://www.vanel.com/_download/torsion-eng.pdf.

[19] Pololu, *Micro Metal Gearmotors*. dirección: https://www.pololu.com/category/60/micro-metal-gearmotors.

[20] ——, *Dual MC33926 Motor Driver Carrier*. dirección: https://www.pololu.com/product/1213/resources.

[21] E. DIY, *Acelerómetro MPU 6050*. dirección: https://www.electronicadiy.com/products/acelerometro-mpu-6050.

[22] Fullwat, *Batería Li-Po estándar 3.7V - 450 mAh*. dirección: http://fullwat.com/LP602040-bateria-li-po-estandar-37v-450-mah/.

[23] F. Bay, *V9R1 RecurDyn Manual*. Function Bay, Inc, 2017.

CAPÍTULO 14

Anexos

14.1. Planos de construcción

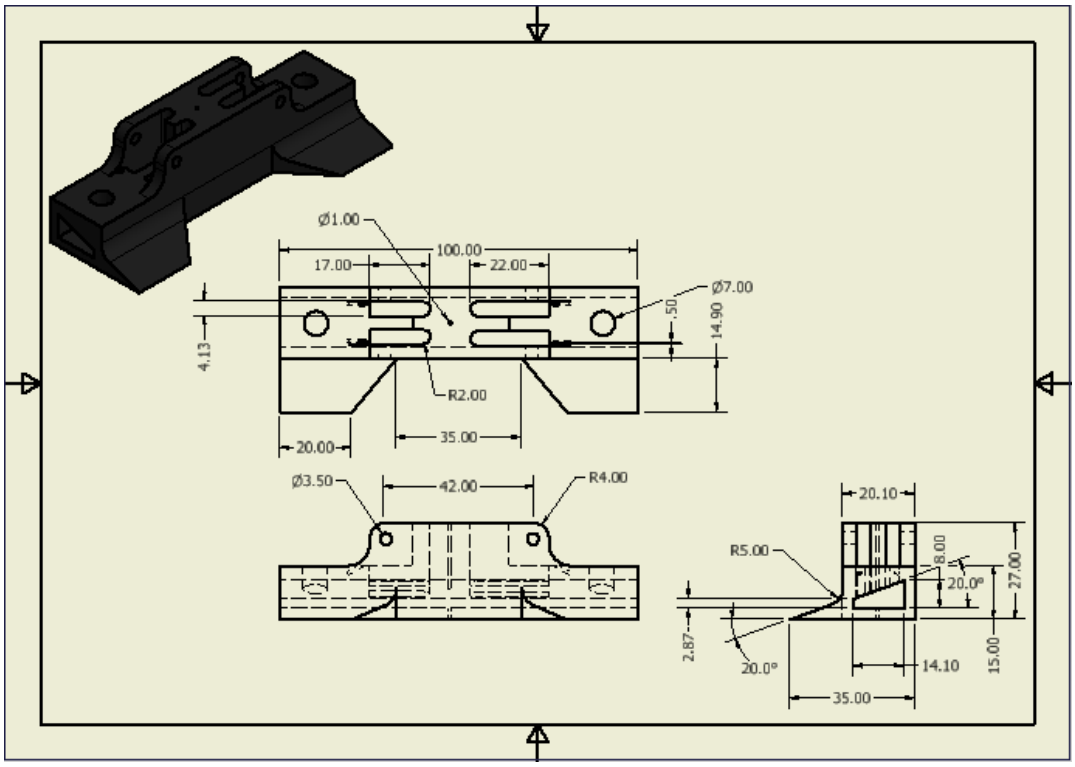


Figura 43: Plano de las patas del robot.

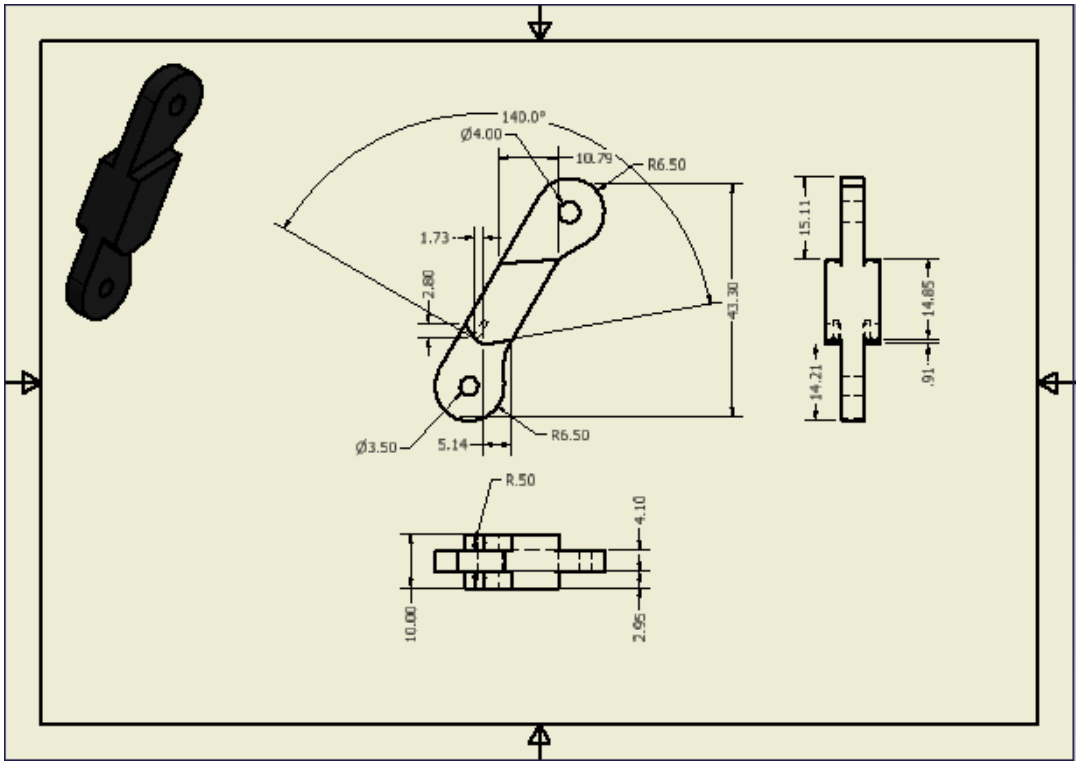


Figura 44: Plano de los eslabones inferiores del robot.

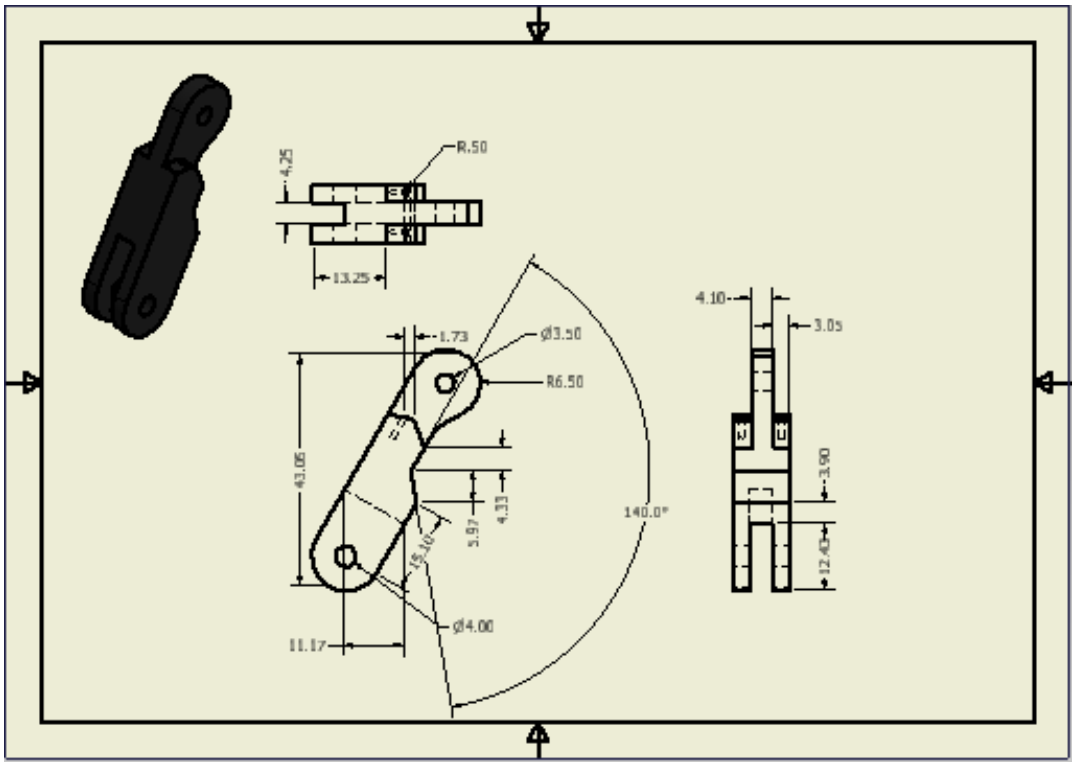


Figura 45: Plano de los eslabones superiores del robot.

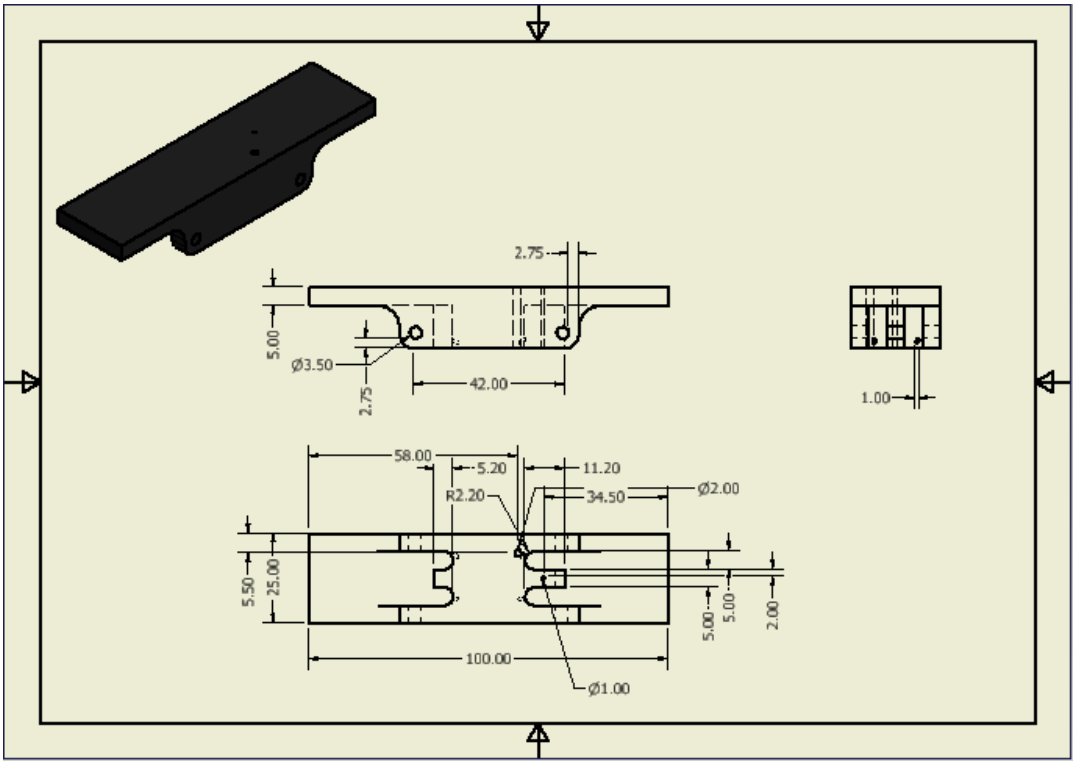


Figura 46: Plano de la parte superior del robot.

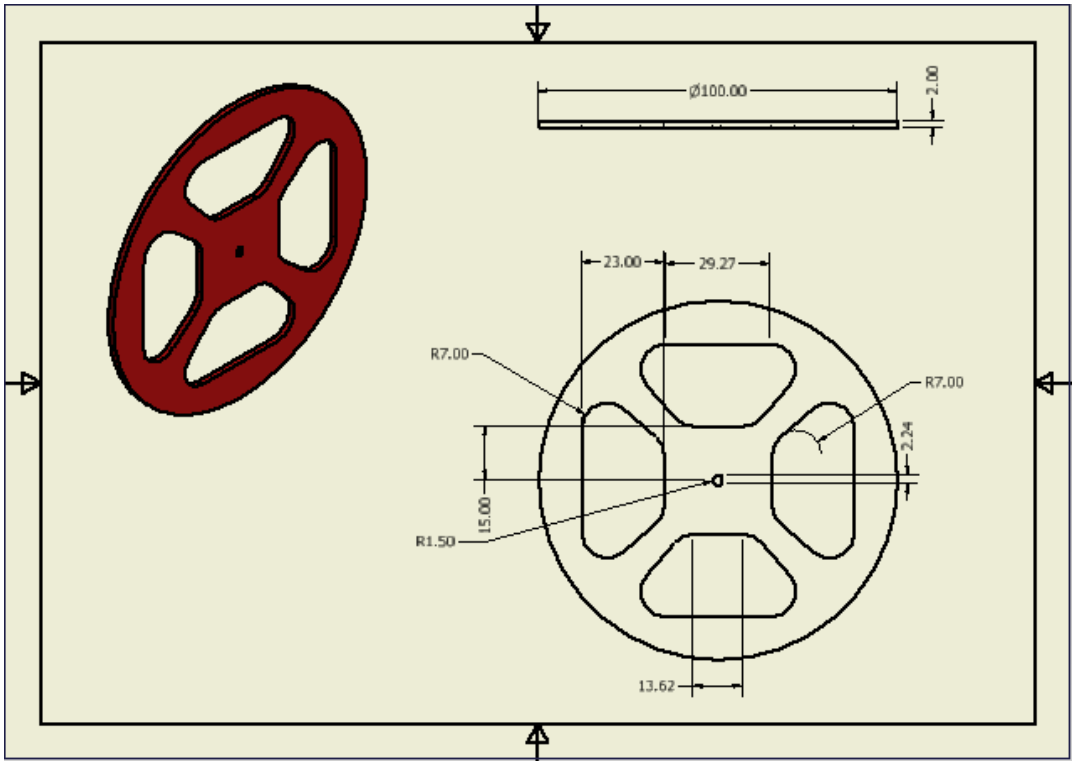


Figura 47: Plano de la rueda de inercia.

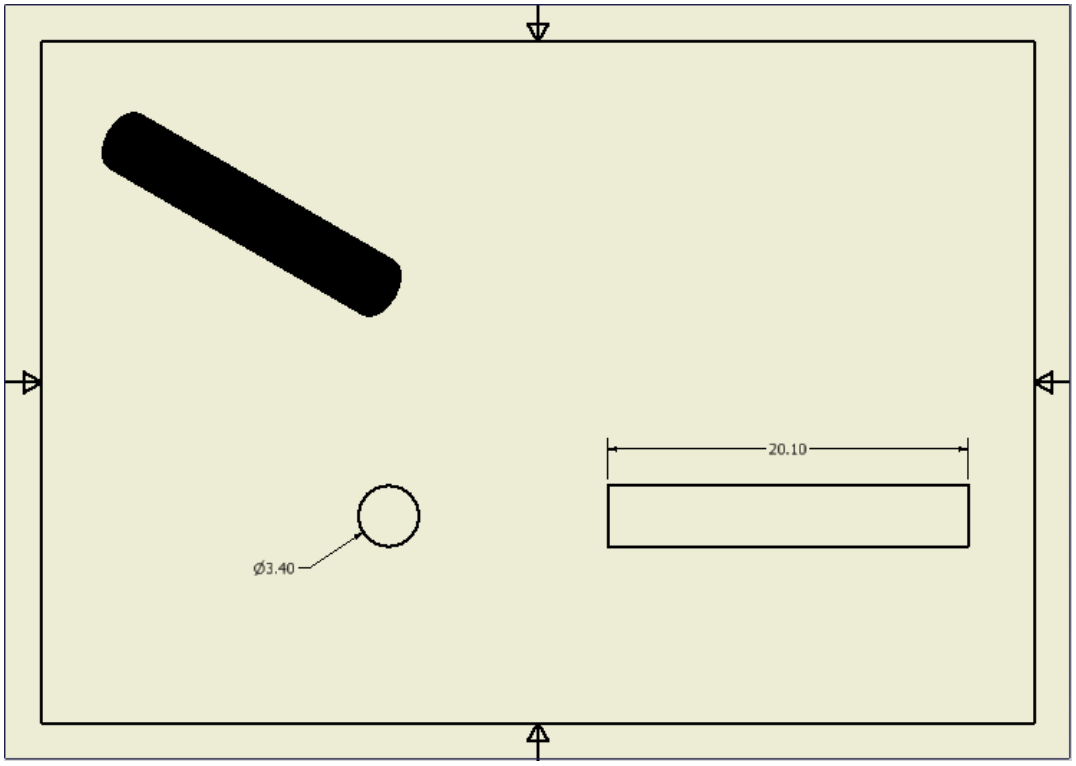


Figura 48: Plano del pin de sujeción de las patas con los eslabones inferiores.

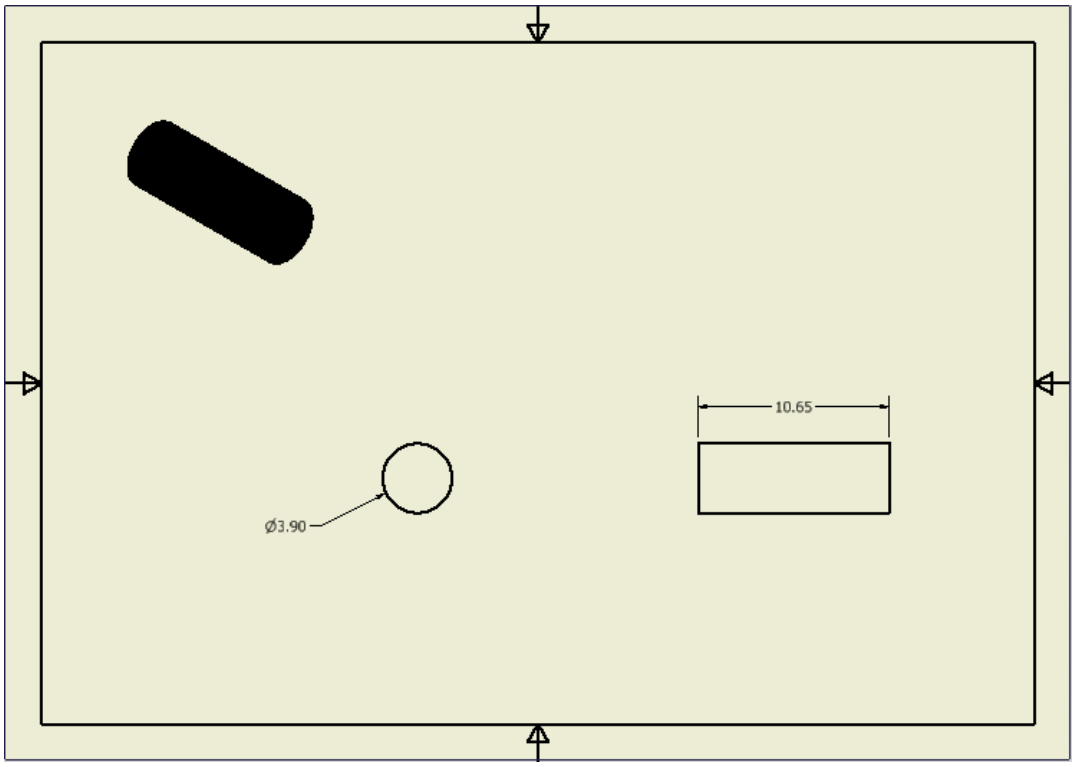


Figura 49: Plano del pin de sujeción de los eslabones inferiores con los eslabones superiores.

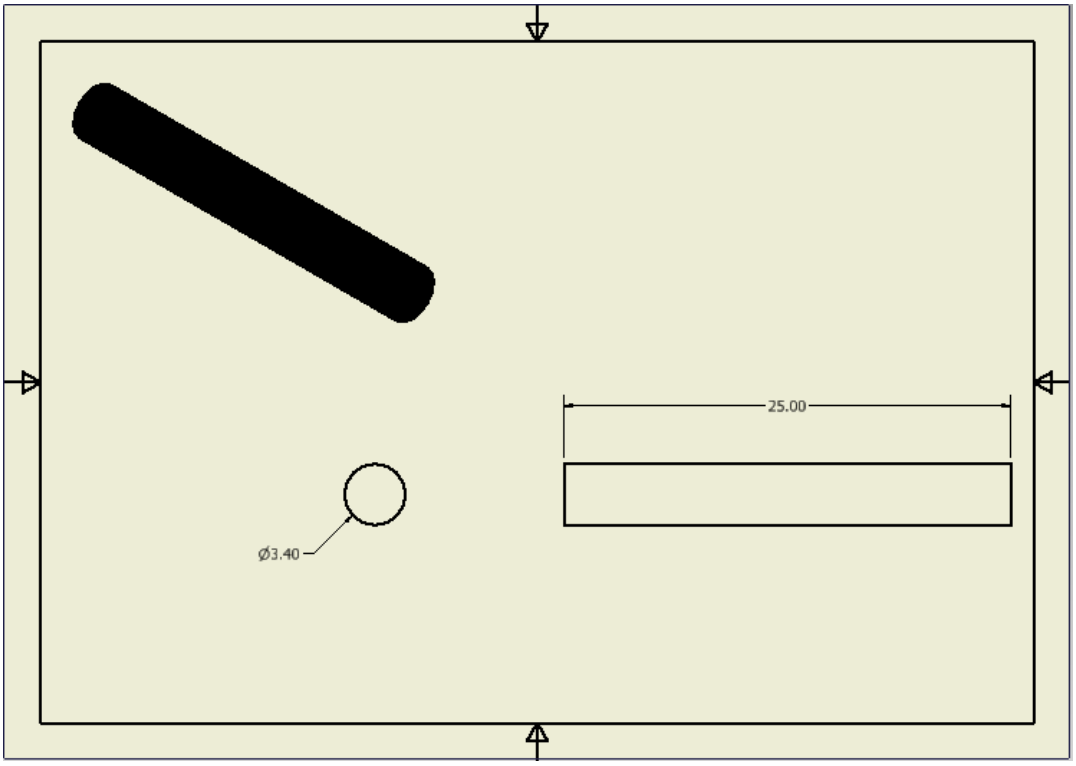


Figura 50: Plano del pin de sujeción de la parte superior del robot con los eslabones superiores.

14.2. Enlace para repositorio de GitHub

https://github.com/DiegoCastroA/Jumping_Robot_Inertia_Wheel

CAPÍTULO 15

Glosario

ATmega128RFA1: Modelo de microcontrolador de la empresa Microchip 24

Autodesk Inventor: Es un paquete de modelado paramétrico de sólidos en 3D producido por la empresa de software autoDesk 1

Matlab: Abreviatura de MATrix LABoratory (laboratorio de matrices en español) es un sistema de cómputo numérico que ofrece un entorno de desarrollo integrado (IDE) con un lenguaje de programación propio (lenguaje M) 1

Outline: Contorno de líneas rectas 55

PCB: Printed Circuit Board, por sus siglas en inglés. Una placa de circuito impreso es una superficie constituida por caminos, pistas o buses de material conductor laminadas sobre una base no conductora. Se utiliza para conectar eléctricamente a través de las pistas conductoras, y sostener mecánicamente, por medio de la base, un conjunto de componentes electrónicos 21

Scope: Ventana se Simscape que despliega las variables que se desean graficar en tiempo real 44

SMA: Shape Memory Alloy, por sus siglas en inglés, que traducido significa efecto térmico de memoria 4

Tiva C: Es una plataforma de prototipos electrónicos basados en una familia de microcontroladores creada por Texas Instruments 24

TM4C123H6PM: Modelo del microcontrolador que utiliza la tarjeta Tiva C 24