



**FACULTAD DE INGENIERÍA, ARQUITECTURA Y
URBANISMO**

**ESCUELA PROFESIONAL DE INGENIERÍA MECÁNICA
ELÉCTRICA.**

TESIS

**MODELAMIENTO Y SIMULACIÓN DE LOS
PARÁMETROS DE FUNCIONAMIENTO DE UNA
TURBINA HIDROCINÉTICA TIPO GORLOV
MEDIANTE EL USO DE UNA INTERFAZ GRÁFICA
DE USUARIO**

**PARA OPTAR TITULO PROFESIONAL DE INGENIERO
MECANICO ELECTRISISTA**

Autor:

Bach. Vasquez Acuña Jhon Anderson

<https://orcid.org/0000-0002-5552-7243>

Asesor:

Mg. Vives Garnique Juan Carlos

<https://orcid.org/0000-0003-0988-9881>

Línea de Investigación:

Infraestructura, Tecnología y Medio Ambiente

Pimentel – Perú

2020

**MODELAMIENTO Y SIMULACIÓN DE LOS
PARÁMETROS DE FUNCIONAMIENTO DE UNA
TURBINA HIDROCINÉTICA TIPO GORLOV MEDIANTE
EL USO DE UNA INTERFAZ GRÁFICA DE USUARIO**

Aprobación del Jurado

Mg. Ing. Alvarado Silva Carlos Alexis

Presidente del Jurado de Tesis

Mg. Ing. Gastiaburu Morales Silvia Yvone
Secretario del Jurado de Tesis

Mg. Ing. Vives Garnique Juan Carlos
Vocal del Jurado de Tesis

DEDICATORIA

Dedicarlo en primer lugar a Dios por ayudarme siempre en mi formación académica y guiarme por el camino correcto, a mis padres por tanto cariño por tanto amor por siempre apoyarme en mis proyectos, tan bien dedicarlo a mi hermana por todo su apoyo tanto emocional como económico, a mi novia por siempre estar a mi lado y apoyándome siempre y a mis familiares en especial a mis tíos Martiza Chamaya y Santos Carranza por siempre apoyarme y estar conmigo, que gracias a todos ellos he podido siempre superar los obstáculos que se presentan en la vida para poder terminar mi carrera profesional con éxito.

JHON ANDERSON VASQUEZ ACUÑA

AGRADECIMIENTO

Agradecer a Dios primeramente por cuidarme por darme todos los días las fuerzas necesarias para seguir adelante, también a todas aquellas personas que han influido de manera directa en mi desarrollo como persona moral, como ser humano, como estudiante, que, gracias a su apoyo, consejos, amigos, profesores, pero muy en especial a mi familia.

A mi padre Marco Vasquez Rojas y a mi madre María Acuña Castillo y a mi hermana Cecilia Vasquez Acuña estoy eternamente agradecido con ellos, que, con tanto esfuerzo, con tanta dedicación a base de tanto sacrificio me han proporcionado los medios para que yo pudiera estudiar una carrera profesional y que en estos momentos logre terminar mis estudios de nivel superior, gracias por todo.

Al Mg. Juan Carlos Vives Garnique, por su calidad de docente, asesor de este proyecto de investigación, gracias a su apoyo, colaboración y dedicación durante el desarrollo de este proyecto.

JHON ANDERSON VASQUEZ ACUÑA

RESUMEN

En la presente investigación se desarrolló una Interfaz Gráfica de Usuario (GUI), que permita el modelado y simulación de los parámetros y funcionamiento de una turbina hidrocinetica tipo Gorlov.

La investigación inicia con la identificación de las variables de entrada y de salida y los pasos necesarios para simular el funcionamiento de la turbina, reconociéndose los parámetros, fórmulas y procedimientos que se deben modelar y aplicar. La GUI se desarrolló mediante la herramienta de programación visual llamada GUIDE, disponible en MATLAB; para esto se ha diseñado en primer lugar el algoritmo de trabajo además de los modelos matemáticos necesarios. Finalmente se procedió a validar los resultados del programa, obteniendo como resultado las gráficas de la Potencia el Torque y los rpm, podemos

determinar que para valores de flujos de 4 m/s y con un valor de 100 RPM la turbina puede desarrollar una potencia aproximada de 5900 W. También para flujos de 3.5 m/s y a 100 RPM se logran potencias de 2000 W. En todos los casos se observa mejores desempeños de la turbina a altas velocidades de flujo obteniendo altas potencias, además también para la curva del torque nos indica que cuando la turbina gira a 90 rpm y con valores de flujo de 4 m/ se obtiene un torque aproximado de 80 N-m.

Palabras clave: GUI, Simulación, turbina hidrocínética Gorlov, Modelado, Software, interfaz gráfica, MATLAB.

ABSTRACT

In the present investigation, a Graphical User Interface (GUI) was developed, which allows the modelling and simulation of the parameters and operation of a Gorlov-type hydrokinetic turbine.

The investigation begins with the identification of the input and output variables and the steps necessary to simulate the operation of the turbine, recognizing the parameters, formulas and procedures that must be modelled and applied. The GUI was developed using the visual programming tool called GUIDE, available in MATLAB; For this, the working algorithm has been designed first, in addition to the necessary mathematical models. Finally, the results of the program were validated, obtaining as a result the graphs of Power, Torque and rpm.

determine that for flow values of 4 m / s and with a value of 100 RPM the turbine can develop an approximate power of 5900 W. Also for flows of 3.5 m / s and 100 RPM, powers of 2000 W are achieved. In all cases, observes better turbine performances at high flow speeds obtaining high powers, also for the torque curve it indicates that when the turbine rotates at 90 rpm and with flow values of 4 m / a torque of approximately 80 Nm is obtained.

Keywords: GUI, Simulation, Gorlov hydrokinetic turbine, Modeling, Software, graphical interface, MATLAB.

INDICE GENERAL

DEDICATORIA	III
AGRADECIMIENTO	IV
RESUMEN.....	V
ABSTRACT.....	VI
ÍNDICE DE TABLA	X
INDICE DE FIGURAS.....	XI
I. INTRODUCCIÓN	14
1.1. REALIDAD PROBLEMÁTICA.....	14
1.2. TRABAJOS PREVIOS	16
1.2.1. Nivel Internacional	16
1.2.2. Nivel Nacional	17
1.2.3. Nivel Local.....	18
1.3. TEORÍAS RELACIONADAS AL TEMA.....	19
1.3.1. Turbina Hidrocinética	19
1.3.2. Turbina Gorlov	20
1.3.3. Características de las Turbina Gorlov.	20
1.3.4. Bases teóricas.	25
1.3.5. Software para modelado y simulación en MATLAB	27
1.3.6. Descripción de la Interfaz.	27
1.4. FORMULACIÓN DEL PROBLEMA	28
1.5. JUSTIFICACIÓN E IMPORTANCIA DEL ESTUDIO	28
1.5.1. Justificación Tecnológica.....	28
1.5.2. Justificación Económica	29
1.6. HIPÓTESIS	29
1.7. OBJETIVOS	29
1.7.1. Objetivos Generales.....	29
1.7.2. Objetivos Específicos	29
II. MÉTODO	31
2.1. TIPO Y DISEÑO DE INVESTIGACIÓN.	31
2.1.1. Tipo De Investigación	31
2.1.2. Diseño de Investigación	31
2.2. VARIABLES, OPERACIONALIZACIÓN	31
2.2.1. Variable Independiente	31
2.2.2. Variable Dependiente	31
2.3. POBLACIÓN Y MUESTRA	32

2.4.	TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS, VALIDEZ Y	
	CONFIABILIDAD.....	33
2.4.1.	Técnicas de recolección de datos.....	33
2.4.2.	Validez y confiabilidad.	34
2.5.	PROCEDIMIENTOS DE ANÁLISIS DE DATOS.	34
2.6.	CRITERIOS ÉTICOS.....	35
III.	RESULTADOS	38
3.1.	FLUJO GRAMA DE LAS VARIABLES DE ENTRADA Y SALIDA DE LA TURBINA	
	HIDROCINETICA TIPO GORLOV.	38
3.2.	LISTADO DE FÓRMULAS UTILIZADAS	38
3.2.1.	Número de Reynolds.....	38
3.2.2.	Ángulo de Ataque (α).....	39
3.2.3.	Fuerza de sustentación (FL) o Lift (L).....	40
3.2.4.	Fuerza de arrastre (FD) o Drag (D)	41
3.2.5.	Torque	42
3.2.6.	Relación de velocidad de punta (TSR)	42
3.2.7.	Velocidad Angular	43
3.2.8.	Potencia del fluido.....	44
3.2.9.	Potencia de la turbina (mecánica)	44
3.2.10.	Área de Barrida	44
3.2.11.	Eficiencia o Coeficiente de potencia (Cp).....	45
3.2.12.	Relación de solidez o Blade Solidity (σ)	46
3.2.13.	Cuerda del álabes.....	46
3.2.14.	Angulo de Inclinación o de Hélice	47
1.1.1.	Perfil alar o perfil de álabes.....	48
3.2.15.1.	Perfil NACA.....	48
1.2.	ELABORACIÓN DEL ALGORITMO PARA EL FUNCIONAMIENTO DEL PROGRAMA	49
1.3.	PROGRAMACIÓN DEL MODELO MATEMÁTICO E INTERFAZ GRÁFICA DE USUARIO ...	51
1.3.1.	Herramienta GUIDE	51
1.3.2.	Función de la GUIDE	52
1.4.	OBTENCIÓN DE LAS GRÁFICAS EN LA GUI DE MATLAB.....	53
1.5.	VALIDACIÓN DE RESULTADOS.....	56
1.6.	DISCUSIÓN DE RESULTADOS	58
IV.	CONCLUSIONES Y RECOMENDACIONES.....	60
2.1.	CONCLUSIONES.....	60
2.2.	RECOMENDACIONES.....	61

REFERENCIAS	62
ANEXO N° 1.....	67
CÓDIGO DE ÉTICA DEL COLEGIO DE INGENIEROS DEL PERÚ (CIP).....	67
ANEXO N° 2.....	68
CÓDIGO DE ÉTICA DE INVESTIGACIÓN DE LA USS	68
ANEXO N° 3.....	69
CÓDIGO DE PROGRAMACIÓN PORTADA DE LA GUI.....	69

ÍNDICE DE TABLA

Tabla 1.Operacionalización de variable dependiente.....	32
Tabla 2. Análisis de documentos	33
Tabla 3. Parámetros de la turbina Gorlov	56
Tabla 4. Parámetros de la turbina Gorlov	56
Tabla 5. Parámetros de la turbina Gorlov	57

INDICE DE FIGURAS

Figura 1. Modelo experimental de la turbina Gorlov.....	20
Figura 2. Pruebas de turbina Gorlov en vertical y horizontal.	21
Figura 3 Descripción de la turbina Gorlov.....	21
Figura 4.Turbina de hélice triple.....	22
Figura 5.Rangos de presiones sobre la turbina tipo Gorlov.	23
Figura 6. Velocidad vs Potencia a diferentes áreas de barrido de la turbina hidrocinetica.	24
Figura 7.Representación Gráfica del Modelo 1.	25
Figura 8. Cilindro barrido por el agua.....	26
Figura 9. Logo Matlab.....	27
Figura 10. Interfaz turbina Gorlov.	28
Figura 11. Explicación del procedimiento del análisis de datos	34
Figura 12. Variables de entrada y salida de la turbina	38
Figura 13. Ángulo de Ataque a 5°	39
Figura 14. Ángulo de ataque a 10°.....	39
Figura 15. Relación entre L/D y el ángulo de ataque.....	40
Figura 16. Flujo laminar a través de un alabe	41
Figura 17. Álabe en perdida	42
Figura 18. Curva de Cp vs TSR	43
Figura 19. Representación gráfica del Área de Barrida	45
Figura 20. Curva de comportamiento para varios rotores.....	46
Figura 21. Longitud de la Cuerda	47
Figura 22. Angulo Hélice de la turbina Gorlov.....	47

Figura 23 Dimensiones de un perfil alar	48
Figura 24 Perfil NACA 0018.	49
Figura 25 Diagrama general de funcionamiento de la GUI para la simulación de la turbina Gorlov.....	50
Figura 26. Pantalla de editor GUIDE en MATLAB	51
Figura 27 Programación de la GUI	52
Figura 28, Programación de la GUI	53
Figura 29 Representación Gráfica de la potencia vs velocidad a diferentes áreas de barrido obtenidas en la GUI.....	53
Figura 30 Representación gráfica de la Potencia vs RPM obtenidas de la GUI	54
Figura 31 Representación gráfica del Torque vs RPM obtenidas de la GUI.	55
Figura 32. Curva del torque vs RPM	57
Figura 33. Curva de la potencia vs Velocidad	58

CAPITULO I

INTRODUCCIÓN

I. Introducción

1.1. Realidad Problemática

Ya desde ahora nos basamos por el interés de experimentar modelamientos asistidos por computadora llevando a cabo a aplicaciones de fluidos incluso para cursos como turbo maquinas. En el año 1998 Gorlov realiza una propuesta de un prototipo de turbina de tres hélices acto para convertir la potencia hidráulica de la corriente del agua en energía mecánica para aprovecharlo en la generación hidro-eléctrica, los usos de esta clase de turbina fueron llamadas turbinas hidrocineticas (Marturet, 2012).

Las turbinas hidrocinéticas son muy importantes dentro de la generación de electricidad con la ayuda de la corriente del agua, especialmente de un río de bajo caudal, donde se hace más accesible su instalación y puesta en marcha. Además, es necesario en muchas de las poblaciones que no tienen energía eléctrica, pero que están localizadas a orillas de un río, siendo así factible incorporar un sistema de generación de tipo hidrocinético, que a su vez puede ser interconectado a la red eléctrica nacional (Rivadeneira, 2015, pág. 2).

La energía es el motor impulsor del desarrollo del hombre y de las sociedades. Para su producción es imprescindible la utilización de recursos naturales, hecho que se ha puesto en evidencia especialmente a partir del siglo XX, por lo que se llevado acabo un modelo de producción basados en la energía por medio de la combustión de combustibles fósiles no renovables, una de las formas más viables son utilizar la infraestructura existen como ríos lagos, vientos, canales. Las primeras aplicaciones de esta tecnología se remontan sobre los griegos que utilizaban el potencial hidrocinético de las corrientes de los ríos para la molienda de granos y otras actividades como el bombeo. Además de ser una fuente renovable se ha despreciado con respecto a otras tecnologías lo largo de la historia por el alto costo de sus instalaciones (Leticia, 2015, pág. 1).

En estos tiempos de siglo XXI contar con energía eléctrica se ha convertido en un factor importante para tener una mejor calidad de vida. Sin embargo, aún existen familias que carecen de energía eléctrica a pesar de contar con recurso hídrico solo que esté en menor proporción (Quispe & Maquera, 2019, pág. 20).

El sistema interconectado de distribución eléctrica en el Perú no llega a zonas alejadas o aisladas, es por ello que las pequeñas centrales hidroeléctricas tiene importancia para así

satisfacer los requerimientos de energía eléctrica en zonas rurales o no interconectadas, siendo un factor para la economía y desarrollo de las zonas aledañas mejorando la calidad de vida (Quispe & Maquera, 2019, pág. 22).

Es así que la idea de aplicación de turbinas hidrocinéticas para generar electricidad de manera limpia y con un bajo casi despreciable impacto ambiental y social, nace con el fin de poder abastecer la demanda de electricidad en poblaciones aisladas, donde tengan cerca la corriente de un río, o a su vez poder inyectar energía eléctrica de manera continua al SIN (Rivadeneira, 2015).

Las energías renovables son vistas por muchos como el futuro energético para nuestro país y el mundo, mediante el presente informe de investigación se plantea realizar el modelamiento y simulación de los parámetros de una turbina hidrocinética tipo Gorlov que se recurrirá a la ayuda de una interfaz gráfica de usuario mediante métodos de análisis numéricos.

1.2. Trabajos previos

Por las décadas de los 80, el ingeniero Peter Garmarn diseña la primera turbina hidrocinetica, la cual es puesta en marcha en el rio Nilo para el riego de cultivos de arroz en Sudan. Al transcurrir los años se implementó y se mejoró a la turbina Garman, llegando a realizar una mejor eficiencia de un 7 %, con esta mejora de la turbina se dio a conocer que las turbinas hidrocineticas serían un factor importante para la ayuda de energía en los lugares que no contaban con energía eléctrica (Linares, 2019).

1.2.1. Nivel Internacional

(Marturet, 2012). Mediante su proyecto de investigación, “Simulación Fluidodinámica de un modelo de turbina hidrocínética tipo Gorlov”. Nos describe que mediante su proyecto de investigación se plantea la necesidad de obtener las potencialidades energéticas para poder suministrar mediante la ausencia de un modelo a escala, utilizando la herramienta computacional CDF. El proyecto se basa en un estudio de modelamiento matemático de turbina Gorlov y también realiza una simulación numérica por medio de métodos de volumen finitos con el propósito de alcanzar el comportamiento bidimensional y fluidodinámico del caudal, logrando un estudio de convergencia obteniendo unos modelos de volúmenes finitos teniendo un resultado del 0,5% de error obteniendo las gráficas de la turbina determinando el mayor punto de rendimiento con 5 m/s de velocidad de flujo, esto se logra por medio de la turbina cuando realiza un giro de 10 RPM y entregando 55,5 W de potencia con una eficiencia hidráulica 99%.

(García, Toicen, & Rojas, 2014). En su proyecto de investigación “Diseño de un banco de ensayos de turbina hidrocínética tipo Gorlov para el laboratorio de Termo-Fluidos del Instituto Universitario de tecnología del estado Bolívar, ciudad Bolívar, estado Bolívar”. Establece la necesidad de obtener los potenciales energéticos del comportamiento fluidodinámico, utilizando el CDF mediante métodos analíticos numéricos. El estudio planteado permite llevar a cabo las curvas características de torque y rpm con diferentes variables de flujo. Finaliza añadiendo precios para su elaboración y construcción.

(Rivadeneira, 2015). Mediante su proyecto de investigación “Modelamiento y Simulación de la operación de generadores que emplean turbinas hidrocineticas en ríos de bajo caudal”. Nos describe que utiliza herramientas computacionales basados en los modelos matemáticos para obtener las curvas características y realizar el estudio del análisis de la turbina hidrocineticas. También mediante esta herramienta computacional resuelve el análisis de la potencia

eléctrica que puede generar un río de bajo caudal, teniendo en cuenta los datos de la turbina y también los datos del río como: caudal, velocidad, área, profundidad, etc.

(Galpin & Bakker, 2008). Mediante su artículo sobre nueva tecnología, resaltan la importancia de utilizar la tecnología de herramientas matemáticas computacional Para llevar a cabo simulación teniendo en cuenta los modelos matemáticos. Algunas herramientas más conocidas como dinámica de fluidos computacional (CFD), utilizadas en fluidos y en la industria de turbo máquina.

1.2.2. Nivel Nacional

(Rodríguez, 2018). Mediante su proyecto de investigación “Diseño, Construcción Y Optimización de Turbinas Hidrocinéticas de ríos y Canales para Generación de Energía Eléctrica” ; nos describe mediante en su trabajo se elaboró el cálculo de un rodete Darrieus con tres álabes rectos y perfiles simétricos y realizo un modelo matemático de cálculo donde en primer lugar se obtuvieron valores de torque generado por la turbina con coeficiente experimentales obtenidos en condiciones estáticas del conjunto de perfil simétrico estudiados. Los coeficientes del perfil seleccionado se obtuvieron con un software “xflr5”, el perfil seleccionado fue el Naca 0021 que ofrece buenas características para el diseño, ya con los parámetros del diseño obtenido se construye el prototipo de la turbina hidrocinetica donde fue instalado en un canal de agua para evaluar su funcionamiento. Con los resultados obtenidos demuestra que los perfiles simétricos son los adecuados para el modelo de turbina a usar. Finaliza realizando un estudio técnico e económico sobre la factibilidad de construcción e instalación de este tipo de turbinas hidrocineticas en zonas rurales.

(Mendoza Y. P., 2017). En su proyecto de investigación “Diseño de Generador Hidroeléctrico Portable para zonas Rurales”; Empleando una turbina hidrocinetica que, mediante un gran movimiento giratorio gracias al fluido, logra hacer un cambio de energía de cinética a mecánica, gracias a un generador de imanes, que se ubica al eje de la turbina, puede generar corriente alterna, y gracias a un rectificar y un regulador de voltaje la corriente es acondicionada para los dispositivos electrónicos.

logra diseñar un sistema capaz de generar 10w de potencia para cargas de dispositivos eléctricos con puerto USB2.0, concluyendo que el sistema propuesto puede operar en cualquier hora del día.

(Peña, 2013). En su proyecto de investigación, “Diseño de una turbina hidro-cinética para aprovechamiento energético de ríos no caudalosos”. El punto más fundamental de este proyecto es diseñar de una turbina de menor potencial. Dicha turbina será diseñada por palas torsionales cuyo material será de fibra de vidrio con resina epoxi teniendo 1,58 m de radio, los engranajes serán rectos con un alternador de 24/48 V, brindando una potencia nominal de 1kW. Dicho perfil seleccionado es el SG6043 teniendo un óptimo desempeño con un régimen de bajo número de Reynolds. Nos aclara que debemos tener en cuenta algunos parámetros como son la velocidad del río y la profundidad para este diseño de turbina. Teniendo en cuenta que la turbina diseñada es de 1kW, se tiene que considerar que el río tiene que tener una profundidad de 3.8 m aproximadamente por el motivo que el radio es de 1.50 concluye que mediante la generación de energía de estos temas de turbinas se vuelven muy interesantes y nos obliga a buscar la forma de implementar algunos modelos de turbina para zonas rurales que beneficiará a los pobladores.

(Maldonado, 2005). Mediante su proyecto de investigación, “Diseño de una turbina de río para la generación de electricidad en el distrito de Mazan-Región Loreto” nos describe que realizó el diseño y el cálculo de un modelo de turbina de río, fabricado de fibra de vidrio con 2 m de diámetro con 250 W de potencia con 427 rpm de velocidad. Mediante un generador de imanes, va a generar corriente alterna que por medio un rectificador de diodos lo va a transformar a corriente continua. Los datos obtenidos de los cálculos del número de alabes y del perfil seleccionado se tabulan y esos datos van a servir para la fabricación de la turbina. La turbina después de ser instalado en el lugar correspondiente viene funcionando correctamente y la energía aprovechada gracias al diseño de la turbina es almacenada en baterías y utilizarlos en todas las casas domésticas y en otros equipos.

1.2.3. Nivel Local

(Yumpo, 2014). Mediante su proyecto de investigación “Diseño de una turbina hidrocinetica sumergible para la generación de energía eléctrica de 5kW de potencia en el caserío de Maino, distrito de San Isidro de Maino. Región Amazonas”, a quien denomina con las letras THS (Turbina hidrocinetica sumergible). Las datos necesarios para elaborar el diseño de la turbina se llevaron a cabo en el río Utcubamba, donde se realizaron el cálculo de la velocidad, potencia y también para el dimensionamiento de la turbina hidrocinetica sumergible, además se llevó a cabo el cálculo de la potencia mecánica para conocer el comportamiento de la turbina. Los datos obtenidos son; potencia útil 1,24 kW, velocidad de

1,85 m/s, la turbina hidrocínética con un diámetro de 1.5 m con una altura de 0,9 m, además para la elaboración de los alabes se considera un material de fibra de vidrio. Para el diseño de la turbina lo realizo en el software CAD Solidwork, también realizo simulaciones del rotor, concluye elaborando un plan de mantenimiento para turbina hidrocínética y para todos los accesorios además realiza la elaboración económica su metrado y su presupuesto.

1.3. Teorías relacionadas al tema

1.3.1. Turbina Hidrocinetica

Los términos utilizados como “turbinas hidrocínéticas” también se puede encontrar en la literatura de otras formas como: turbinas de corrientes de agua (WCT), turbina hidráulica de cabeza ultra baja, turbinas hidráulicas en la corriente, turbinas de corrientes de ríos (RCT) o métodos de cambios de energía de corrientes de río (RCECS). Por ende este trabajo se ha adoptado “turbinas hidrocínéticas” el cual va a permitir abarcar un aspecto más amplio, todas las turbinas usadas para la conversión de dicho tipo de energía, ya sean aquellas basadas en las corrientes marinas, corrientes de los ríos, o canales de aguas naturales y artificiales (Zubialde, 2016).

Esta turbina fue implementada con la finalidad de aprovechar los caudales de los ríos, la turbina será colocada o instalada en el agua y estará sujeta por medios de cables o cimientos en la profundidad del río. Las partes la cual conforma la turbina son; un generador, caja de transmisión de potencia, eje, hélices, y cada una de estas partes son elevadamente importantes para aprovechar los caudales del río o lago que llegan a impactar en los hélices (Linares, 2019).

Las turbinas hidrocínéticas se pueden utilizar tanto en ríos como en océanos (por marea o conversión de energía de corriente marina). Sin embargo, hay algunas diferencias sutiles entre estos dos campos de aplicación. Turbinas de marea son típicamente más grandes en tamaño (> 100 kW), mientras que las turbinas de río son generalmente en el rango de 1 kW a 10 kW. La mayoría de las turbinas marinas utilizan eje horizontal de anclaje rígido con configuración de generador sumergido. Por otro lado, el eje vertical o turbinas tipo Gorlov son comunes en aplicaciones de energía de río. La energía de las mareas y las turbinas de corriente marina trabajan bajo los acontecimientos naturales de flujo de marea diaria y variaciones estacionales oceánicas, respectivamente (Rivadeneira, 2015).

1.3.2. Turbina Gorlov

Las turbinas hidrocineéticas tales como Garman, Darrieus presentaron problemas tales como vibraciones y bajo eficiencia, por tal motivo fue desarrollada la famosa turbina Gorlov por parte del ingeniero Alexander Gorlov por los años 1995. Este nombre de turbina Gorlov es llamado de esta manera por su nombre del ingeniero, la cual se inspiró por medio de la turbina Darrieus (Marturet, 2012).

El primer ensayo se llevó a cabo en el año 1996, se demostró que con esta turbina la cual es de menor precio, más liviana, fácil de manejar, de reducidas dimensiones y mínimas vibraciones mecánicas, las cuales se practicaron en Massachusetts Helical Turbine Laboratory, obteniendo grandes resultados, la cual mediante las pruebas esta turbina puede generar una potencia de 5 KW, llegando a generar un 35 % de su rendimiento (Garcia, Toicen, & Rojas, 2014).

La figura 1 muestra un modelo experimental de dicha turbina

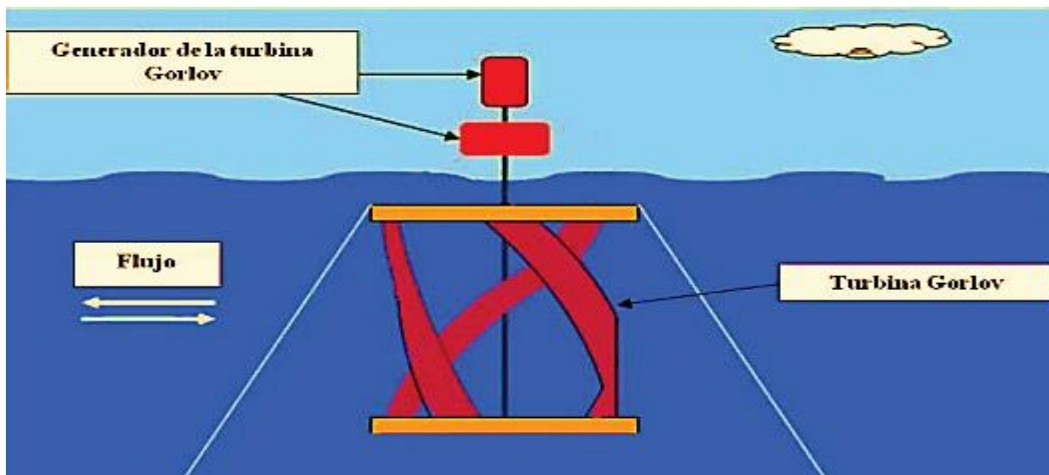


Figura 1. Modelo experimental de la turbina Gorlov.

Fuente: Alexander Gorlov (1998)

1.3.3. Características de las Turbina Gorlov.

- Pueden ser instaladas en forma horizontal o vertical, apilados en filas, como bobinas en una cuerda.
- Se pueden instalar en aguas tan bajas como de 0.91 m. (Las otras turbinas de flujo axial requieren 2 y 4 m o algo mas)

- El arranque es casi instantáneo. Dichas características se muestran de forma visual en la figura 2.



Figura 2. Pruebas de turbina Gorlov en vertical y horizontal.

Fuente. (Garcia, Toicen, & Rojas, 2014).

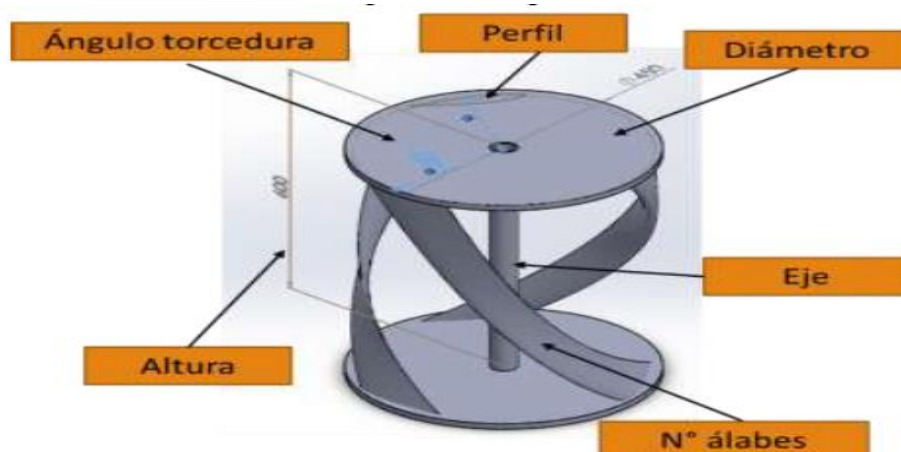


Figura 3 Descripción de la turbina Gorlov

Fuente: (Daniel Mendoza A; 2018)

En el año 1998 Alexander Gorlov, difunde un proyecto "DEVELOPMENT OF THE HELICAL REACTION HYDRAULIC TURBINE", muestra un molde de turbina que

mediante una corriente de flujo libre de ríos y océanos se podrá obtener energía hidráulica. Realizando propuestas en condiciones de: Eficiencia, mini estación de potencia, construcción, diseño y costos, del modelo de turbina tipo helicoidal que llega alcanzar una velocidad de flujo de 8 ft/s con una potencia de 2 KW y un giro de 100 y 300 RPM. Por medio de la siguiente figura se detalla el diseño de turbina de tres alabes para un tipo helicoidal utilizando un perfil de la siguiente descripción NACA 0020 (Marturet, 2012).

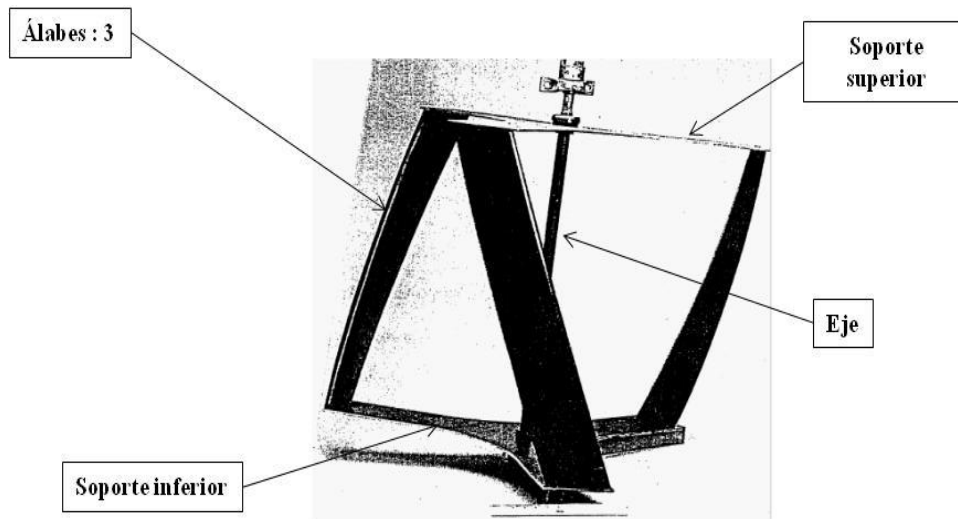


Figura 4. Turbina de hélice triple.
Fuente. Gorlov, Alexander (1998).

Serres realiza una investigación relacionado a la three-dimensional simulation of fluid flow in a Gorlov type turbine and structural design. Se llevó a cabo utilizando la dinámica de fluidos computacional (CFD), abordan elementos de interacción fluidos-estructural de una turbina como, por ejemplo: perfil de velocidades, deformaciones, gradientes de presión, campos de velocidades y esfuerzos.

Serres, también en su simulación empleo las mismas características y utilizo también los mismos planos de la turbina Gorlov que fue diseñada por Mata. Logrando el comportamiento estructural de una THC. Logra también llevar a cabo un estudio tridimensional del gradiente de presiones que efectúa por medio de las hélices de la turbina tipo Gorlov, que sus volúmenes alteran con un promedio de (0.294949 kPa hasta 0.399341 kPa), y una velocidad promedio de 0,74 m/s. En la siguiente figura N°4 se detalla cuyos rangos de presiones sobre la turbina obtenida por la simulación de serres (Marturet, 2012).

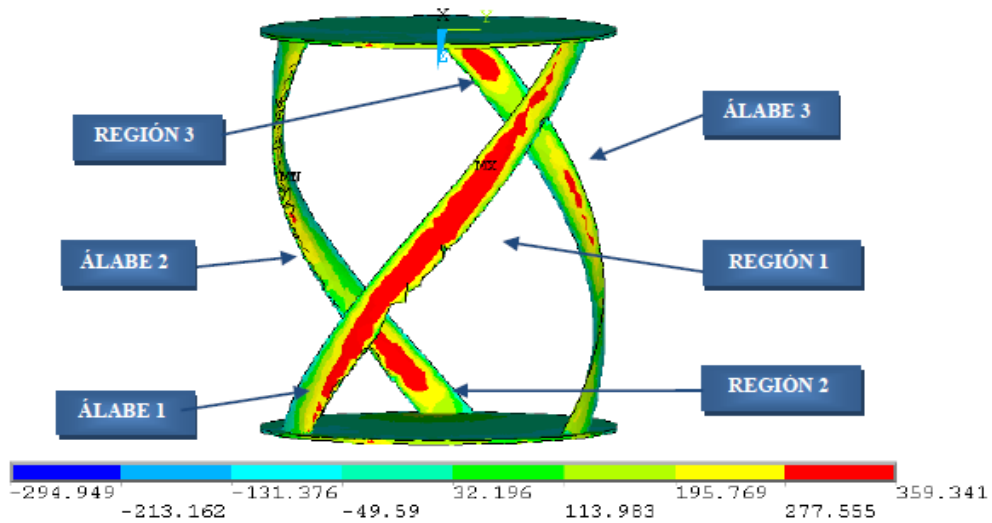


Figura 5. Rangos de presiones sobre la turbina tipo Gorlov.

Fuente: three-dimensional simulation of fluid flow in a Gorlov type turbine and structural design.

(Rivadeneira, 2015). En su proyecto de investigación nos describe la función de las gráficas realizadas en una simulación con la ayuda del software Matlab.

En la siguiente figura nos describe sobre gráfica de la velocidad vs la potencia. En color rojo con un valor de área de barrido de 0.5 m², se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 10 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 250 W. En color azul con un valor de área de barrido de 1 m², se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 10 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 500 W. En color negro con un valor de área de barrido de 1.5 m², se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 10 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 750 W. En color cian con un valor de área de barrido de 2 m², se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 10 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 1000 W. En color amarillo con un valor de área de barrido de 2.5 m², se observa que cuando

aumenta la velocidad de la corriente de agua desde 0 hasta 10 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 1250 W (Rivadeneira, 2015).

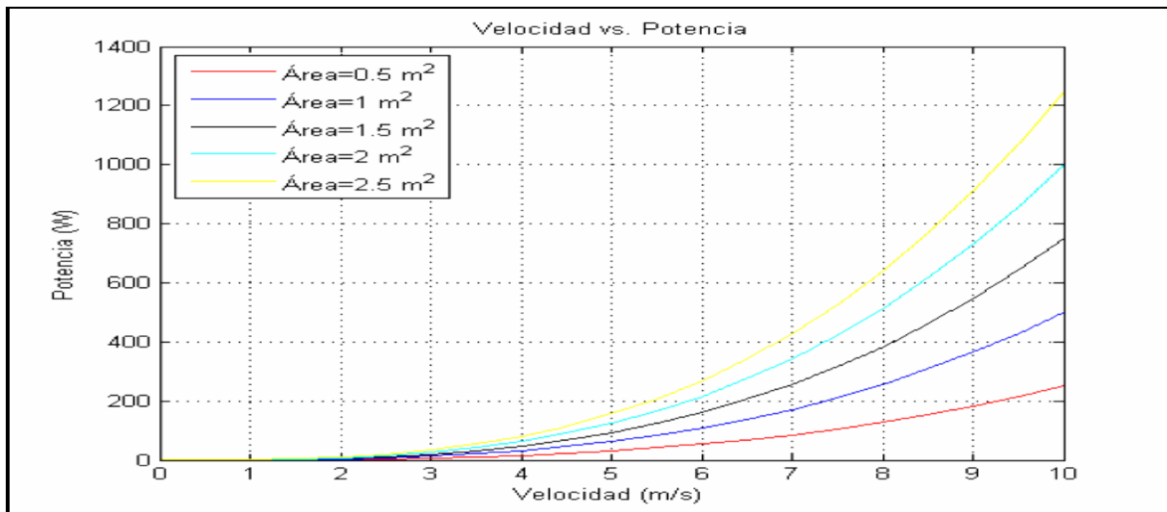


Figura 6. Velocidad vs Potencia a diferentes áreas de barrido de la turbina hidrocinetica.

Fuente: Rivadeneira (2015).

De igual manera en la siguiente figura nos describe la gráfica del modelo 1. Cada curva representa la potencia generada a cierta velocidad de corriente del caudal y a cierta área de barrido de la turbina. En el eje X se tiene la velocidad (m/s), en el eje Y se tiene la potencia (W). Claramente se puede observar que, a mayor intervalo de área de barrido y velocidad de corriente del caudal, la potencia disponible de la turbina también incrementa, se puede decir que el área y la velocidad son directamente proporcionales a la potencia, debido a que se tienen varios valores de intervalos para área y velocidad, existen varias curvas en la figura (Rivadeneira, 2015).

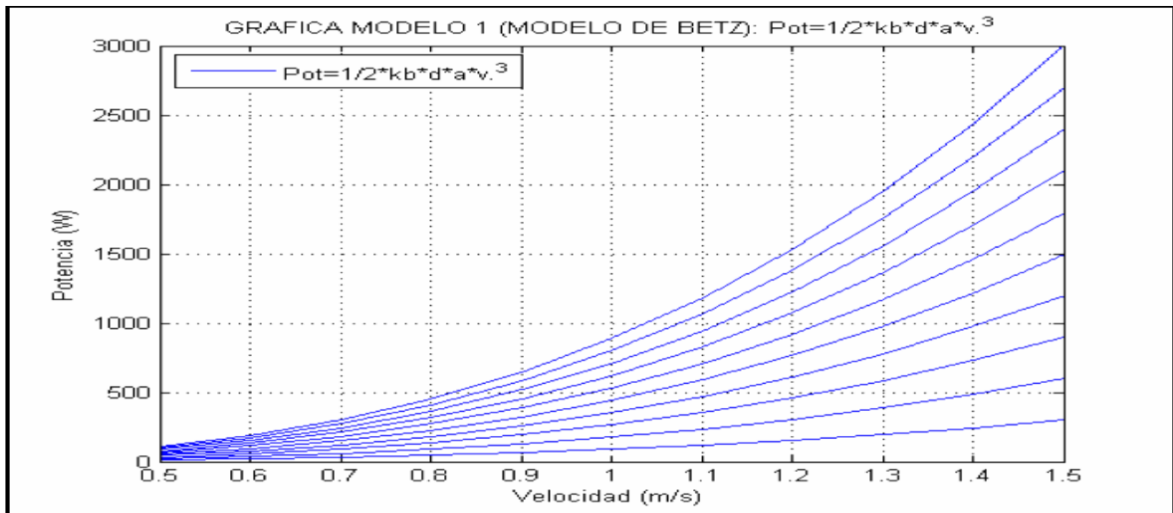


Figura 7. Representación Gráfica del Modelo 1.

Fuente 1Rivadeneira (2015)

1.3.4. Bases teóricas.

1.3.4.1 Turbinas Hidrocinéticas: Energía y Potencia Asociadas.

En la siguiente ecuación está dada por Bernoulli:

$$Z + \frac{P}{\rho g} + \frac{V^2}{2g} = H = \text{constante} \quad (1)$$

El término;

$$\frac{P}{\rho g} = \text{Carga de presión}$$

$$\frac{V^2}{2g} = \text{Carga de velocidades}$$

H = Carga del total de flujo

Z = Carga de elevación

Energía cinética, dada por:

$$E_c = \frac{1}{2} m V^2 \quad (2)$$

Donde:

m = Masa del agua

V = Velocidad del agua

La ecuación de la masa (m) de un cilindro barrido por el Caudal, está dada por:

$$m = \rho V_c \quad (3)$$

ρ = Densidad del agua

V_c = Volumen del cilindro barrido por el agua

La fórmula del volumen del cilindro por el agua está dada por:

$$V_c = A \cdot L \quad (4)$$

Donde:

A = Sección transversal del flujo

L = Longitud del cilindro

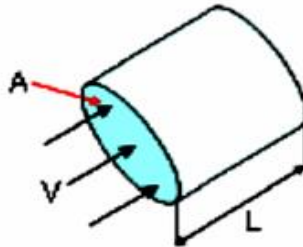


Figura 8. Cilindro barrido por el agua.

Fuente: Rivadeneira (2015)

“Teniendo en cuenta que, si la velocidad es constante del flujo del agua, la longitud (L) estará dada por la siguiente ecuación” (Marturet, 2012).

$$L = V \cdot T \quad (5)$$

Al sustituir (2.3), (2.4) y (2.5) en (2.2) se obtiene:

$$E_c = \frac{1}{2} + \rho AV^3 \quad (6)$$

1.3.5. Software para modelado y simulación en MATLAB

“MATrixs Laboratorys o conocido también como MATLAB es un software que nos permite llevar a cabo cualesquiera operaciones con matrices de una forma rápida y sencilla. Teniendo un lenguaje de programación propio, Lenguaje M” (Ramos, 2012).

“Matlab es un lenguaje de programación de alto rendimiento con un entorno interactivo para la visualización, programación y para los cálculos numéricos. Mediante este programa es posible generar un análisis de datos, crear modelos o aplicaciones y también desarrollar algoritmos” (Muñoz, 2013).

“Todas las funciones incorporas en él, como el lenguaje, herramientas y funciones matemáticas, nos permiten dar soluciones y explorar diversos enfoques, como puede ser con lenguajes de programación tradicionales, tales como; Fortran o C/C++” (Muñoz, 2013).



Figura 9. Logo Matlab.
Fuente: Wikipedia.

1.3.6. Descripción de la Interfaz.

La idea principal de la Interfaz realizada es que al seleccionar un modelo matemático de la potencia generada, se puedan ingresar los valores necesarios para cada uno de los modelos; y en otro caso las marcas de turbinas hidrocínéticas ya tendrán datos propios para luego poder graficar el comportamiento de la potencia frente a la velocidad del río, de esta manera se puede analizar el comportamiento de cualquier río de bajo caudal, frente a la producción de energía que se requiera, para lo cual se hacen necesarios los datos solicitados para cada modelo, ya sea que estos se obtengan por mediciones o historial del río, o por medio de los fabricantes o quien suministre los equipos ya sea los generadores (Rivadeneira, 2015).

Detallamos una gráfica en el software Matlab, para el rotor de eje vertical Gorlov (5 kW), en donde se tienen los datos propios para el modelamiento de esta turbina, los cuales son: Eficiencia del generador (0.66), eficiencia transmisión (0,85), coeficiente de potencia (0.35), ángulo de inclinación (270° debido a que la turbina está en posición vertical), para una velocidad de río de (0.5– 2.75 m/s), todos estos datos se grafican velocidad vs. potencia de acuerdo al modelo 3 (para rotores de eje vertical o inclinado) (Rivadeneira, 2015).

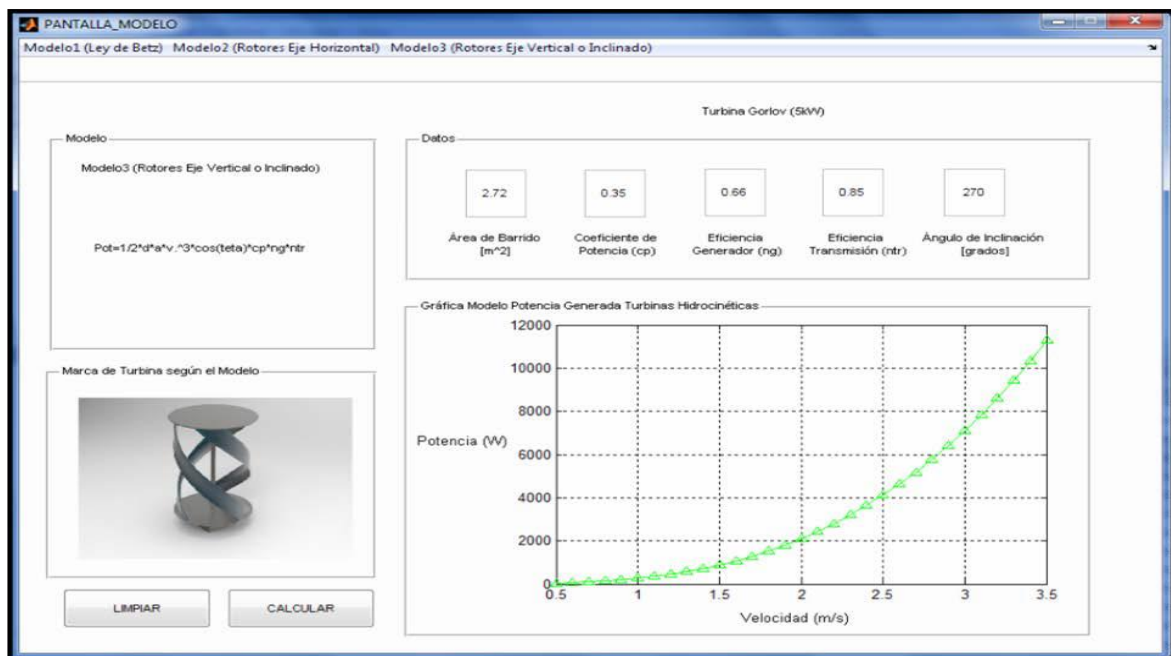


Figura 10. Interfaz turbina Gorlov.
Fuente: Rivadeneira (2015)

1.4. Formulación del Problema

¿Cuál será la secuencia de programación en una GUI basado en el modelamiento para simular los parámetros de funcionamiento de una turbina hidrocinética?

1.5. Justificación e importancia del estudio

El proyecto de investigación tiene como finalidad desarrollar el modelamiento y simulación de una turbina hidrocinética mediante una interfaz gráfica de usuario (GUI).

1.5.1. Justificación Tecnológica

GUI es un programa informático muy novedoso de trabajo y simulación teniendo como ventaja la capacidad de ejecutar cálculos, graficas. Además, se puede personalizar Apps mediante las herramientas de interfaces.

1.5.2. Justificación Económica

El uso de una GUI permitirá al usuario reducir tiempos en la etapa de selección de parámetros y concentrarse más en la aplicación de criterios económicos más adecuados para la reducción de costos en la instalación.

Una GUI que permitirá el modelamiento y simulación de una turbina hidrocínética podría utilizarse como curso de turbo maquinas, teniendo la capacidad de estudiar casos reales planteados por el docente mediante la utilización del GUI.

1.6. Hipótesis

No aplica

1.7. Objetivos

1.7.1. Objetivos Generales

Realizar el modelamiento y simulación de los parámetros de funcionamiento de una turbina hidrocínética tipo Gorlov mediante una interfaz gráfica de usuario.

1.7.2. Objetivos Específicos

- Determinar variables de entrada y salida de la turbina hidrocínética.
- Formular el modelo matemático para el funcionamiento de la turbina hidrocínética.
- Elaborar el algoritmo de trabajo para el funcionamiento del programa.
- Realizar la programación del modelo matemático del funcionamiento de la turbina en la interfaz gráfica de usuario por medio de la herramienta GUIDE de Mathworks Inc.
- Obtener las gráficas de rendimiento de Potencia, Torque y RPM mediante la simulación en la GUI.

CAPÍTULO II

MÉTODO

II. Método

2.1. Tipo Y Diseño De Investigación.

2.1.1. Tipo De Investigación

Aplicada, Cuantitativa

2.1.2. Diseño de Investigación

Investigación Experimental

2.2. Variables, Operacionalización

2.2.1. Variable Independiente

- ❖ Caudal
- ❖ N° Alabes
- ❖ Tipo de Perfil
- ❖ Coeficiente de sustentación
- ❖ Coeficiente de arrastre
- ❖ Viscosidad cinética del agua
- ❖ Longitud de la cuerda
- ❖ TSR
- ❖ Área de barrido

2.2.2. Variable Dependiente

Parámetros de funcionamientos

- Potencia
- Torque
- RPM
- Relación de solidez
- Ángulo hélice
- Coeficiente de potencia
- Fuerza de arrastre
- Fuerza de sustentación

Tabla 1.Operacionalización de variable dependiente

Variables	Dimensión	Indicador	Técnica e instrumento de recolección de datos
Independiente: Caudal N° Alabes Tipo de Perfil Coeficiente de sustentación Coeficiente de arrastre Viscosidad cinética del agua Longitud de la cuerda TSR Área de barrido	Q N - Cl Cd μ C λ A	Caudal N° de álabes - Coeficiente de sustentación Coeficiente de arrastre Viscosidad Longitud TSR Área de barrido	Información bibliográfica (internet, libros, tesis, etc.) Análisis de datos (Matlab) Consultas (anotaciones)
Dependiente: Parámetros de funcionamiento	P T - σ λ - -	Potencia Torque RPM Solidez Ángulo hélice Cp Fuerza de arrastre Fuerza de sustentación	Información bibliográfica (internet, libros, tesis, etc.) Análisis de datos (Matlab) Consultas (anotaciones)

Fuente: Elaboración

2.3.Población Y Muestra

Nuestro Proyecto de Investigación, el cual tenemos como objetivo principal realizar el modelo matemático y la simulación de una turbina tipo Gorlov, considerando el dominio computacional en su totalidad como población.

Mediante este software computacional se llevarán sucesivas pruebas por medio de la herramienta de simulación GUI, con el objetivo de obtener las gráficas características de una turbina tipo Gorlov.

2.4. Técnicas e instrumentos de recolección de datos, validez y confiabilidad.

2.4.1. Técnicas de recolección de datos.

La recolección de datos que se llevó la finalidad de que nos ayude a recolectar datos valiosos e importantes para poder realizar un buen proyecto de investigación.

a) Observación:

Esta técnica de observación es muy fundamental ya que nos didacta como por ejemplo que se va hacer, que se va a realizar, cuando se realizará, quien lo realiza, por qué y donde se va a realizar. Se observará las clases de turbo maquinas con el propósito de conocer el tema a desarrollar y realizar nuestro proyecto.

b) Información Bibliográfica:

La información bibliográfica es una técnica que nos va a servir para determinar una buena elección con respecto a nuestro tema elegido, dichas fuentes bibliográficas son por ejemplo libros, tesis, artículos, internet etc.

c) Análisis de documentos:

Esta técnica de análisis de documento nos sirve para llevar a cabo un análisis de los documentos recopilados ya sea por medio de medio de: tesis, artículos, libros, publicaciones, etc. Y tener una información veraz con respeto a nuestro tema.

Tabla 2. Análisis de documentos

Técnicas	Instrumentos	Validez
Observación	Cámara fotográfica	Especialista
Consultas	Anotaciones	Especialista
Información bibliográfica	Libros, Internet , tesis, etc.	Especialistas
Análisis de datos	Matlab	Especialista

Fuente: Elaboración propia.

2.4.2. Validez y confiabilidad.

La validez y confiabilidad se llevarán a cabo por un profesional capacitado (reconocido como investigador), teniendo un grado académico que lo respalda.

2.5. Procedimientos de análisis de datos.

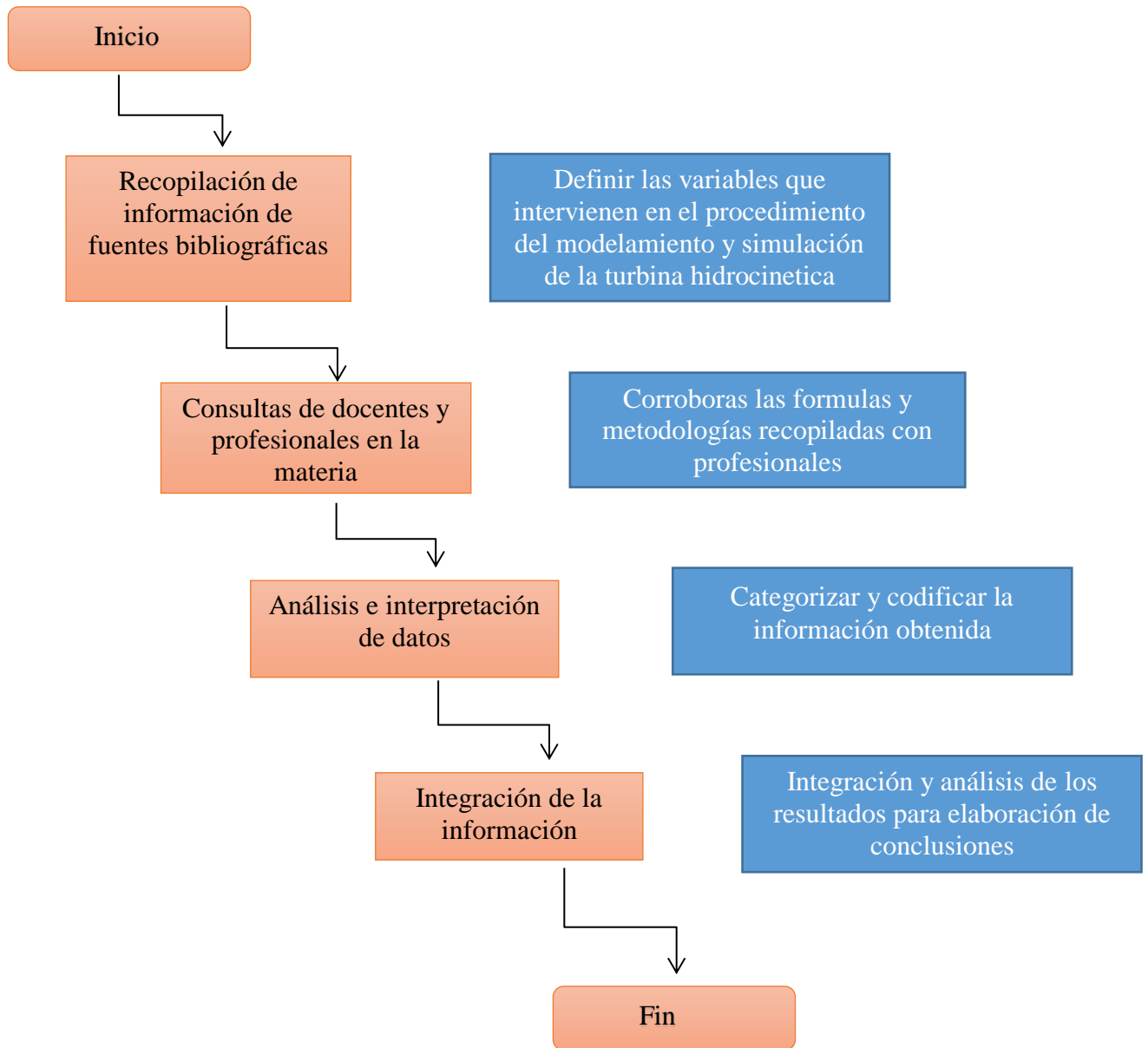


Figura 11. Explicación del procedimiento del análisis de datos

Fuente: Elaboración Propia.

2.6. Criterios éticos.

“Los criterios éticos se basarán en los términos establecidos por Colegio de Ingenieros del Perú (CIP, 1999) y el Código de Ética de Investigación de la Universidad Señor de Sipán (USS, 2017)”.

• Código de ética del Colegio de ingenieros del Perú (CIP, 1999)

Art. 1 – Los ingenieros están al servicio de la sociedad. Por consiguiente, tienen la obligación de contribuir al bienestar humano, dando importancia primordial a la seguridad y adecuada utilización de los recursos en el desempeño de sus tareas profesionales.

Art. 4 – Los ingenieros reconocerán que la seguridad de la vida, la salud, los bienes y el bienestar del país de la población y del público en general, así como el desarrollo tecnológico del país depende de los juicios, decisiones incorporados por ellos o por su consejo, en dispositivos, edificaciones, estructuras, máquinas, productos y procesos. Por ninguna razón pondrán sus conocimientos al servicio de todo aquello que afecte la paz y la salud.

Art. 5 - Los ingenieros cuidarán que los recursos humanos, económicos, naturales y materiales, sean racional y adecuadamente utilizados, evitando su abuso o dispendio, respetarán y harán respetar las disposiciones legales que garanticen la preservación del medio ambiente.

• Código de Ética de Investigación de la Universidad Señor de Sipán (USS, 2017)

Este código está enfocado en delimitar la ética y su aplicación en el quehacer de la investigación universitaria, En este documento se presentará los principios y deberes éticos, con el consentimiento para el uso de datos, las políticas anti plagio y los procedimientos de sanción. A continuación, un breve resumen de este documento.

ART. 1°: Finalidad

La finalidad del Código de Ética de investigación de la Universidad Señor de Sipán (USS), es proteger los derechos, la salud, la vida, la intimidad, el bienestar y la dignidad de la(s) persona(s) que participan en una actividad de investigación

Científica, Tecnológica e innovación, ciñéndose a los principios éticos acogidos por la normativa nacional e internacional, y los acuerdos suscritos por nuestro país en la materia.

ART. 2°: Objetivo

Es definir los principios éticos que orientan la actividad investigativa y su gestión, por parte las autoridades, docente, estudiantes, investigadores, y egresados de la USS. que es dada por el código de ética de investigación de la Universidad Señor de Sipán.

ART. 3°: Alcance

el alcance obligatorio del código ético de investigación es el cumplimiento por parte de los docentes, estudiantes, todas las autoridades académicas, administrativas, egresados y administrativo de la Universidad Señor de Sipán.

CAPÍTULO III

RESULTADOS

III. RESULTADOS

3.1. Flujo grama de las variables de entrada y salida de la turbina Hidrocinetica tipo Gorlov.

A continuación, se realiza un flujo grama de las variables de entrada y salida para la turbina hidrocinetica.

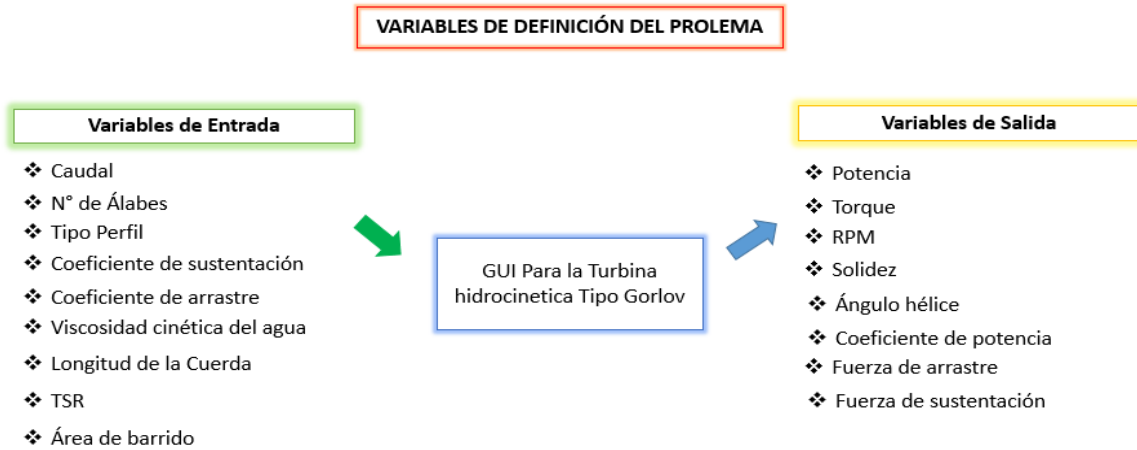


Figura 12. Variables de entrada y salida de la turbina

Fuente: Elaboración Propia

3.2. Listado de Fórmulas Utilizadas

El listado de fórmulas que se ha llevado a cabo, para realizar los cálculos y obtener los valores necesarios para ingresar a la GUIA para la turbina hidrocinetica tipo Gorlov se menciona a continuación.

3.2.1. Número de Reynolds

“Es utilizado para describir las características de un flujo, y es uno de los parámetros más importantes cuando se realiza un escalamiento, además es un parámetro adimensional” (LE-Quesne, 2018).

$$R = \frac{V_{\infty} * C}{\mu} \quad (7)$$

R = Numero de Reynolds

C = Longitud de la cuerda (m)

V_{∞} = Velocidad del flujo (m/s)

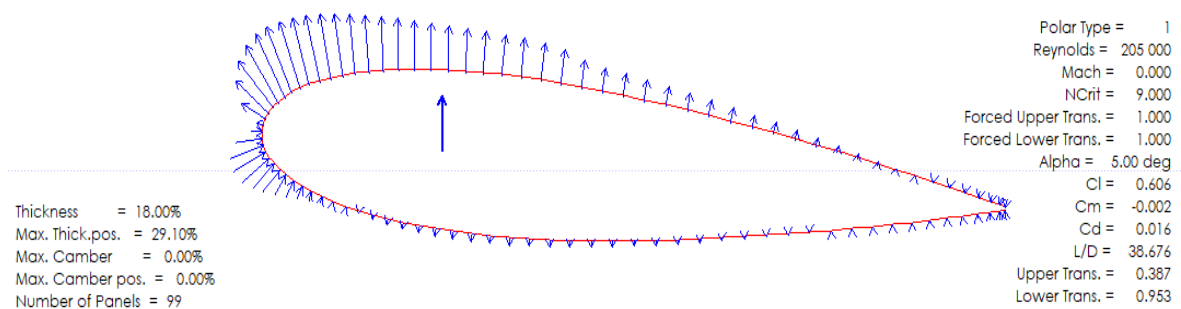
μ = Viscosidad cinética de agua m^2/s

El número de Reynolds es de gran relevancia y con la ayuda del software Q-Blade obtendremos los respectivos valores de ángulo de ataque y los coeficientes de arrastre (Cd) y sustentación (Cl), teniendo estipulado el perfil a utilizar.

3.2.2. Ángulo de Ataque (α)

Encontrando el número de Reynolds, procedemos a encontrar el (α) de ataque en el software Q-Blade, el cual a la vez nos encontramos con los Cl, Cd.

Figura 13. Ángulo de Ataque a 5°



Fuente: Elaboración Propia

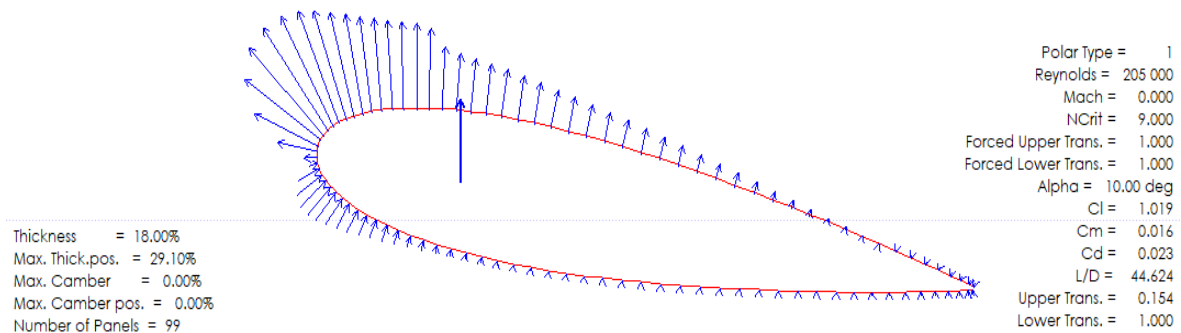


Figura 14. Ángulo de ataque a 10°

Fuente: Elaboración Propia

En la gráfica que podemos observar de la relación de L/D y a diferentes ángulos de ataque, el software Q-BladeDe, nos damos cuenta que el ángulo de mayor rendimiento es de 10°

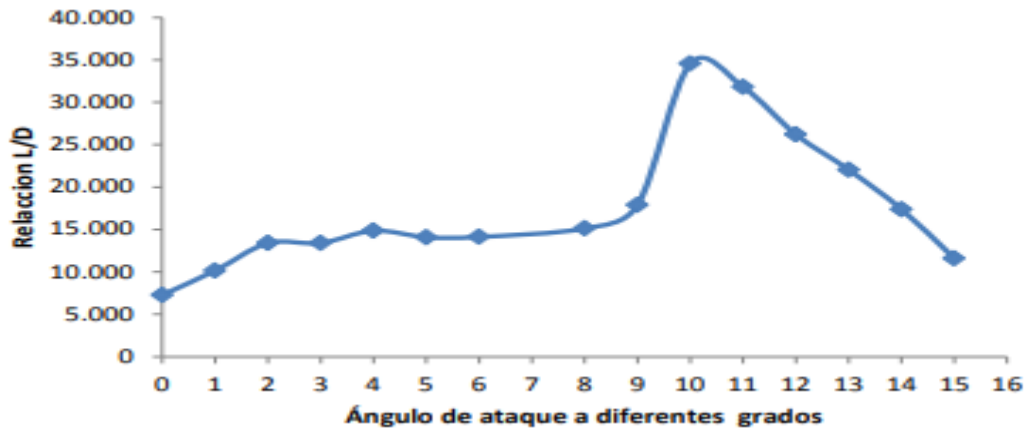


Figura 15. Relación entre L/D y el ángulo de ataque

Fuente: Elaboración Propia.

3.2.3. Fuerza de sustentación (F_L) o Lift (L)

“La fuerza de sustentación (F_L), es definida como la fuerzas que ejerce perpendicular a la dirección de la corriente de un fluido” (Bulla & Romero, 2018).

$$F_L = C_L * \frac{\rho}{2} * (D * H) * V_{\infty}^2 \quad (8)$$

F_L = Fuerza de sustentación (N)

C_L = Coeficiente de sustentación

ρ = Densidad (Kg/m^3)

D = Diámetro (m)

H = Altura (m)

V_{∞} = Velocidad del flujo (m/s)

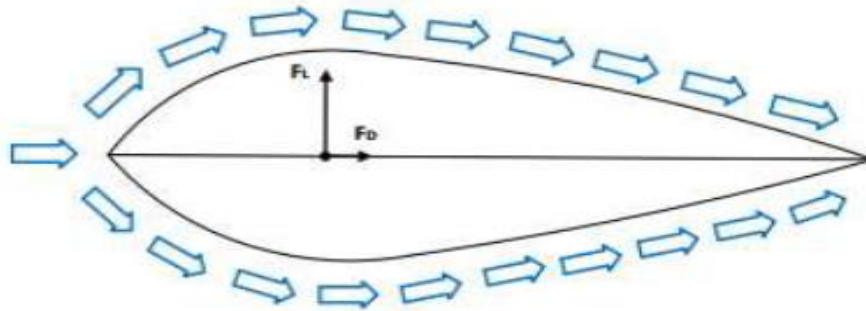


Figura 16. Flujo laminar a través de un alabe

Fuente: (Mendoza A. J., 2018)

3.2.4. Fuerza de arrastre (F_D) o Drag (D)

“Todo cuerpo que se mueve a través de una corriente será sometida a una fuerza, la cual esta fuerza es llamada F_D o D” (Mendoza A. J., 2018).

$$F_D = C_D * \frac{\rho}{2} * (D * H) * V_\infty^2 \quad (9)$$

F_D = Fuerza de arrastre (N)

C_D = Coeficiente de arrastre

ρ = Densidad (Kg/m^3)

D = Diámetro (m)

H = Altura (m)

V_∞ = Velocidad de flujo (m/s)

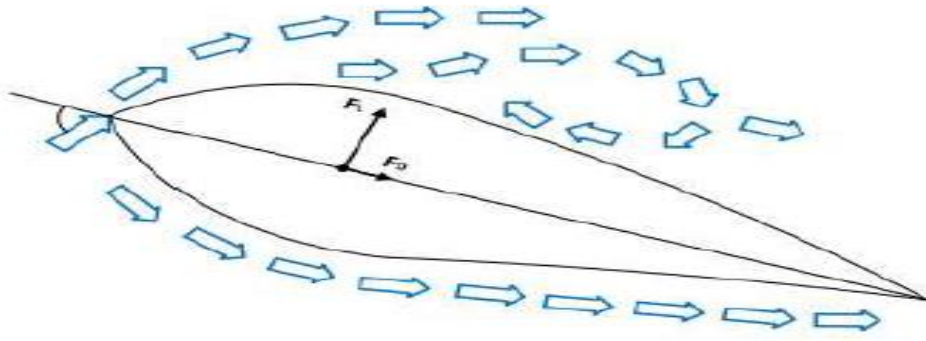


Figura 17. Álabe en pérdida

Fuente: (Mendoza A. J., 2018)

3.2.5. Torque

La ecuación del torque está definida por la siguiente manera:

$$T = D/2 [L \text{ sen } (\alpha) - D \text{ cos } (\alpha)] \quad (10)$$

T = Torque (N-m)

D = Diámetro (m)

L = Fuerza de sustentación

D = Fuerza de arrastre

α = Ángulo de ataque

3.2.6. Relación de velocidad de punta (TSR)

“La relación de velocidad de punta, es conocida también como λ , este es un número adimensional que corresponde a la razón entre dos velocidades, la velocidad tangencial de la turbina y la velocidad del flujo” (LE-Quesne, 2018).

$$\text{TSR} = \frac{\omega D/2}{V_{\infty}} \quad (11)$$

TSR = Relación de velocidad de punta

V_{∞} = Velocidad del flujo (m/s)

D = Diámetro (m)

ω = Velocidad angular (rad/s)

El TSR se puede obtener mediante el software Q-Blade, en la siguiente figura observamos la curva de Cp vs TSR. El TSR que tuvo mayor eficiencia fue de 1,2 con un Cp de 38,70 %.

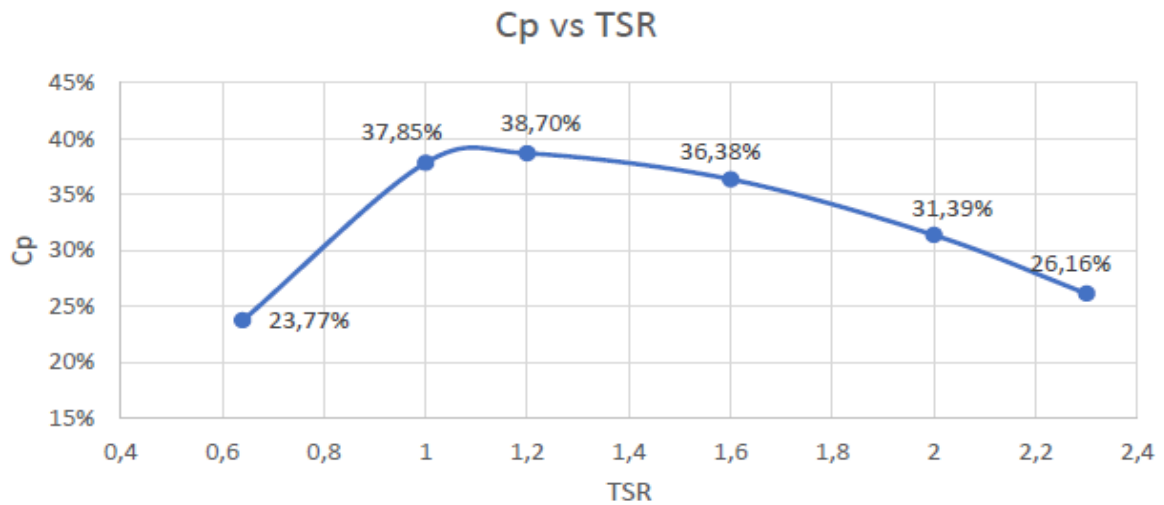


Figura 18. Curva de Cp vs TSR

Fuente: (LE-Quesne, 2018)

Observando la curva del Cp vs TSR, determinamos que el TSR óptimo para la turbina Gorlov es de 1,2 el cual utilizaremos como dato determinado para encontrar la velocidad angular.

3.2.7. Velocidad Angular

Para calcular la velocidad angular, utilizaremos un TSR = 1.2 óptimo ya mencionado anteriormente en la figura 16 y está dada por la siguiente ecuación:

$$\omega = \frac{60 * \lambda * V_{\infty}}{\pi * D} \quad (12)$$

ω = Velocidad angular (rpm)

λ = Relación d velocidad de punta

D = Diámetro (m)

H = Altura (m)

V_{∞} = Velocidad del flujo (m/s)

3.2.8. Potencia del fluido

la potencia del fluido está dada por la siguiente ecuación:

$$P = \frac{1}{2} \rho (D * H) V_{\infty}^3 \quad (13)$$

P = Potencia del flujo (W)

ρ = Densidad (Kg/m^3)

D = Diámetro (m)

H = Altura (m)

V_{∞} = Velocidad del flujo (m/s)

3.2.9. Potencia de la turbina (mecánica)

la potencia de la turbina o potencia mecánica está dada por la siguiente ecuación:

$$P_m = T * \omega \quad (14)$$

P_m = Potencia de la turbina (w)

T= Torque (N-m)

ω = Velocidad angular (rad/s)

3.2.10. Área de Barrida

El área de barrida de la turbina según Gorlov es considerada como el diámetro por y la altura de la turbina.

$$A = D * H \quad (15)$$

A = Área de Barrida (m^2)

D = Diámetro (m)

H = Altura (m)

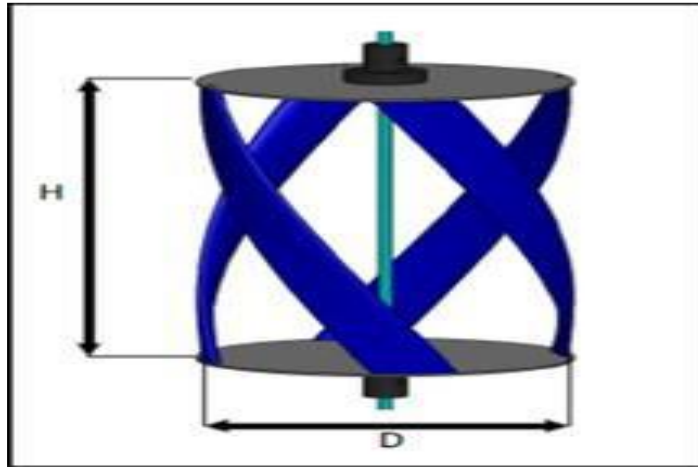


Figura 19. Representación gráfica del Área de Barrida

Fuente: (LE-Quesne, 2018)

3.2.11. Eficiencia o Coeficiente de potencia (C_p)

La fórmula de la eficiencia C_p o en inglés Power Coefficient, está determinada de la siguiente manera:

$$C_p = \frac{P_m}{P} \quad (16)$$

C_p = Coeficiente de potencia

P_m = Potencia mecánica (W)

P = Potencia del fluido (W)

Según Albert Betz una turbina puede alcanzar una eficiencia máxima del 59 % aprovechando netamente la energía cinética.

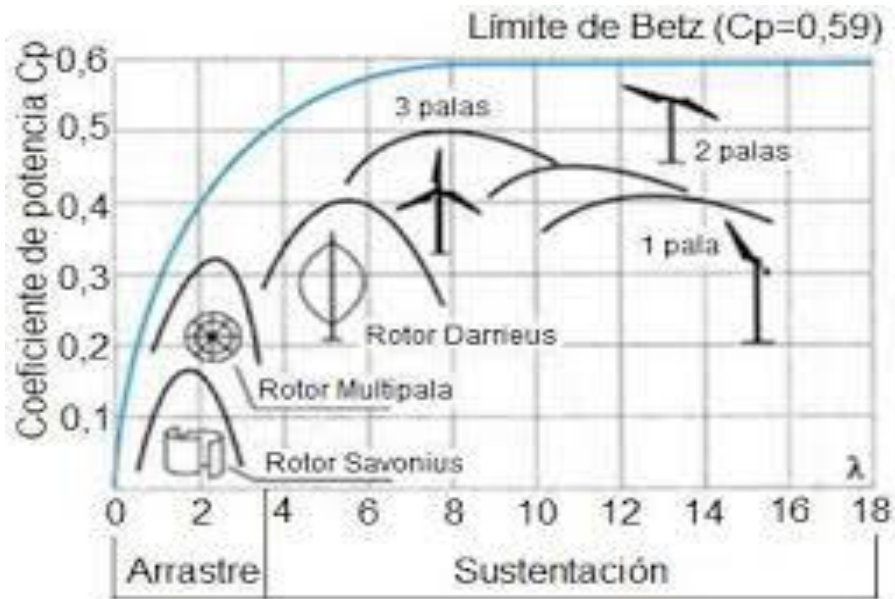


Figura 20. Curva de comportamiento para varios rotores

Fuente: (Bulla & Romero, 2018)

3.2.12. Relación de solidez o Blade Solidity (σ)

“Blade Solidity, es un número adimensional que representa la fracción del perímetro de la circunferencia de la turbina que está cubierta por álabes” (LE-Quesne, 2018).

$$\sigma = \frac{C B}{\pi D} \quad (17)$$

σ = Relación de solidez

C= Longitud de la cuerda

B = Numero de álabes

D = Diámetro de la turbina

3.2.13. Cuerda del álabe

La cuerda son variables definidas en los perfiles aerodinámicos. Esta cuerda es la distancia entre la punta hacia la cola del álabe.

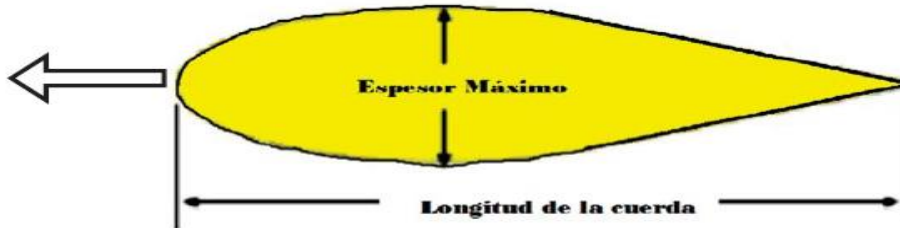


Figura 21. Longitud de la Cuerda

Fuente: (LE-Quesne, 2018)

3.2.14. Angulo de Inclinación o de Hélice

Angulo hélice, es el ángulo que forma la trayectoria de la hélice con la base de la turbina.

$$\delta = \arctan \left(\frac{B H}{\pi D} \right) \quad (18)$$

δ = Ángulo de hélice

B= Numero de álabes

H = Altura de la hélice

D = Diámetro de la turbina

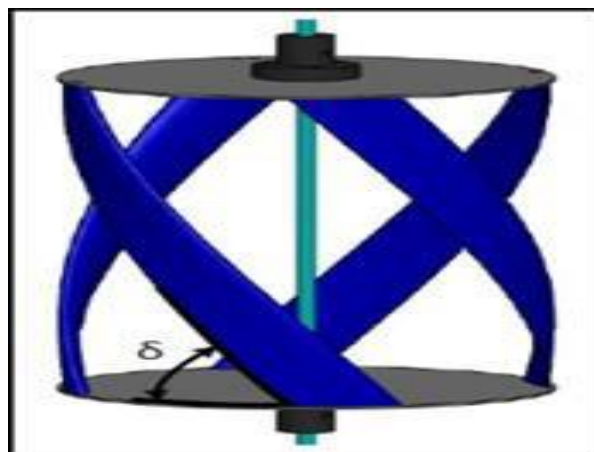


Figura 22. Angulo Hélice de la turbina Gorlov

Fuente: (Marturet, 2012)

1.1.1. Perfil alar o perfil de álabe

“El perfil alar o alabe , son conocidos por las geometrías utilizadas en hidrodinámicas e aerodinámica, las cuales generan fuerzas de sustentación” (LE-Quesne, 2018).

“Es necesario conocer ciertas dimensiones, para poder explicar correctamente el perfil del álabe” (LE-Quesne, 2018).

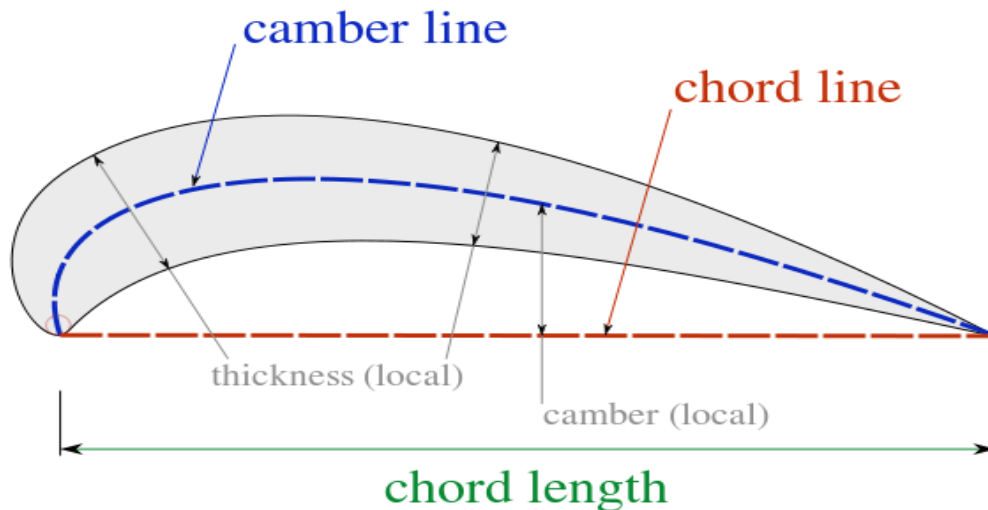


Figura 23 Dimensiones de un perfil alar

Fuente: (LE-Quesne, 2018)

De la figura 23 podemos determinar las partes del perfil

- Chord line: Es la línea recta que une ambos extremos del perfil, su longitud corresponde a chord length, c .
- Camber line: Es la línea que pasa por la mitad del perfil en todo momento.
- Thickness: Espesor del perfil en cierto punto de la camber line.
- Camber: Distancia entre la chord line y la camber line.

3.2.15.1. Perfil NACA

“Los perfiles NACA fueron concebidos National Advisory Committee for Aeronautics (NACA), con el fin de estandarizar los perfiles de acuerdo con su geometría y distribución diferenciándolos con nomenclatura” (LE-Quesne, 2018).

“Según la nomenclatura NACA, los perfiles que comienzan con dos dígitos iguales a cero son simétricos. Y los otros dígitos restantes nos indica el espesor con respecto a la longitud de la cuerda” (LE-Quesne, 2018).

La forma del perfil se describe con un número de 4 dígitos, estos corresponden a:

- Primer dígito: máximo camber, en porcentaje del chord length.
- Segundo dígito: indica la posición del máximo camber, en décimos del chord length.
- Tercer y cuarto dígitos: máximo espesor (thickness) en porcentaje del chord length.

Según el reporte 80-2114 de la Agencia Estadounidense SANDIA, el perfil NACA 0018 es un perfil aerodinámico óptimos para turbinas GORLOV, la cual utilizaremos para nuestra simulación.

A continuación, se muestran la representación gráfica del perfil NACA 0018.

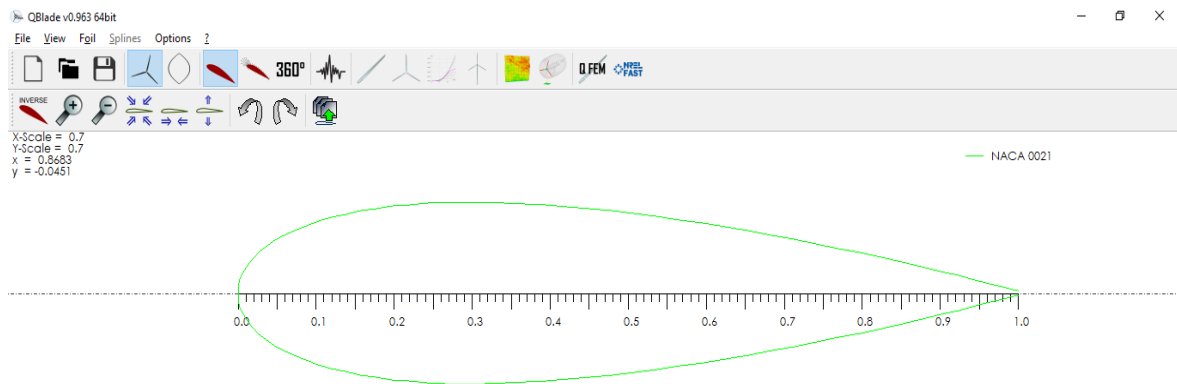


Figura 24 Perfil NACA 0018.

Fuente: Elaboración Propia

1.2. Elaboración del Algoritmo para el funcionamiento del programa

La elaboración del algoritmo tiene como objetivo que la GUI puede ser utilizado para cualquier caso de turbina hidrocinetica y no este limitado a casos particulares. El algoritmo de trabajo estará representado por un diagrama de flujo general, de cómo debería funcionar la interfaz.

Entonces, el diagrama general de trabajo de la GUI sería el siguiente:

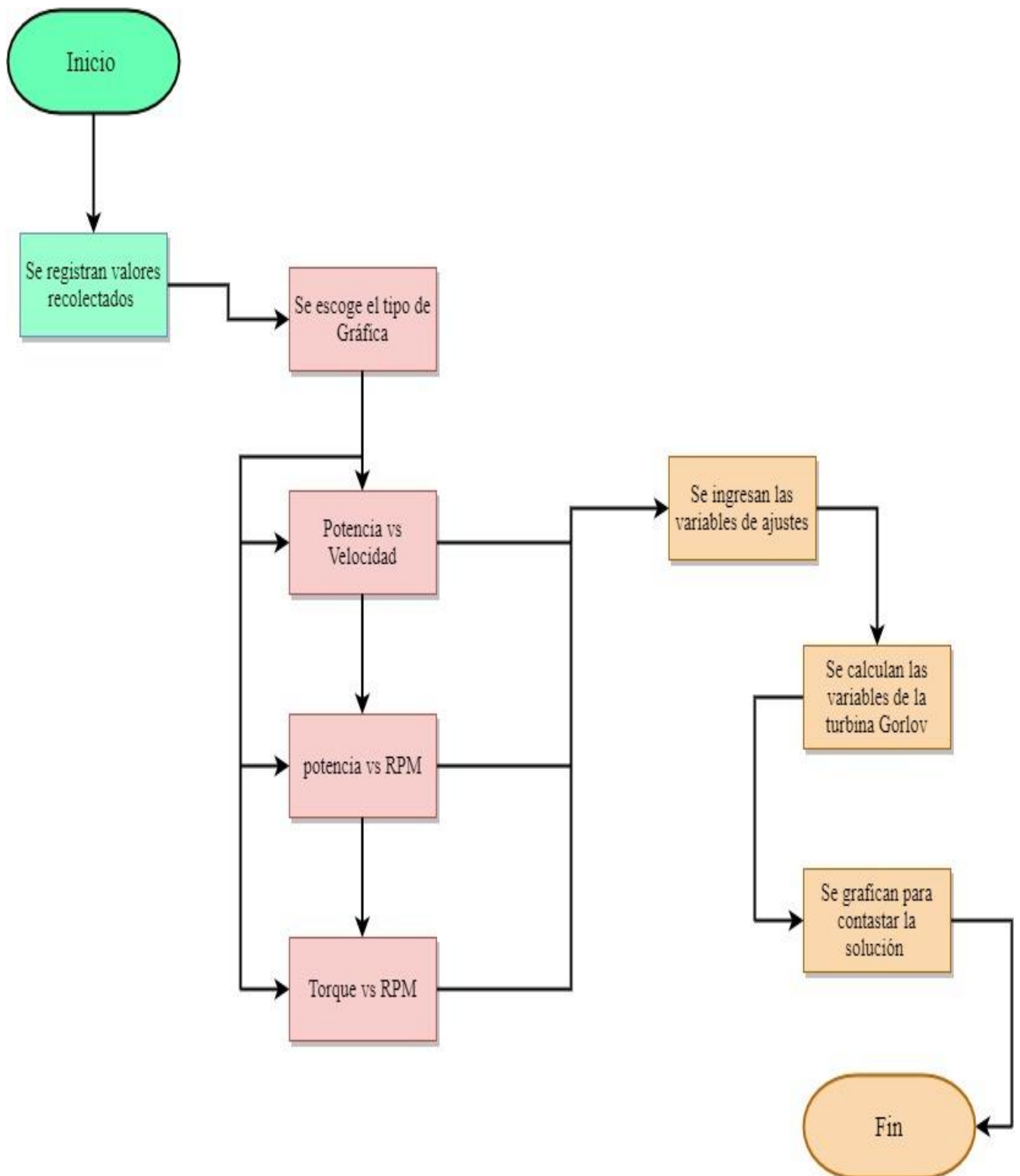


Figura 25 Diagrama general de funcionamiento de la GUI para la simulación de la turbina Gorlov

Fuente: Elaboración Propia.

1.3. Programación del modelo matemático e interfaz gráfica de usuario

1.3.1. Herramienta GUIDE

Graphical User Interface o (GUI), conocida como interfaz gráfica de usuario, esta interfaz permite realizar tareas interactivas, además se representa por una gráfica de una o varias ventanas que contienen controles o denominados componentes. La cual nos permite las creaciones de entornos gráficos denominados o llamados “Guide”, comportándose de una manera adecuada permitiendo a los usuarios las facilidades de ingresar datos y el análisis posterior a los resultados. Las diferentes funciones de la GUI, se detallan de la siguiente manera (Muñoz, 2013).

La interfaz gráfica de usuario o GUIDE, está formado por una pantalla en la cual se muestra una barra de herramientas, además una zona de diseño cuadrículada, una paleta con componentes, en la parte superior se muestra una barra de menú con funciones elementales de edición y por último el editor de código asociado a la GUI, como podemos observar en la imagen (Muñoz, 2013).

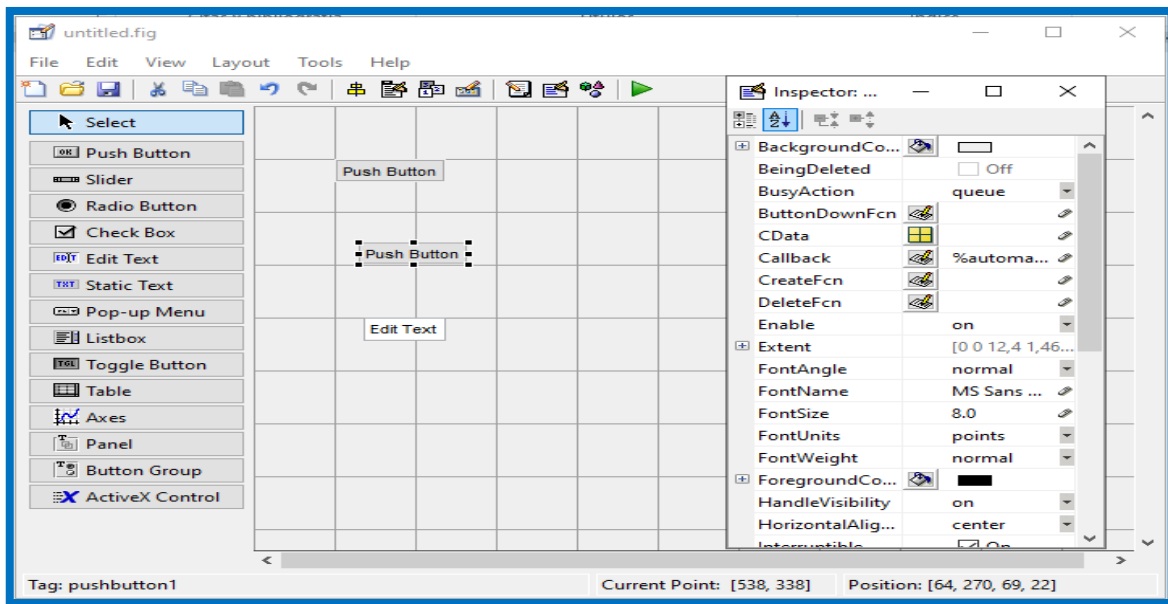


Figura 26. Pantalla de editor GUIDE en MATLAB

Fuente: Elaboración Propia.

1.3.2. Función de la GUIDE

Mediante esta herramienta de la Guide podemos diseñar interfaces de usuarios ya sea para Apps personalizadas, las cuales se lleva a cabo mediante el editor de diseño de Guide, con este editor es posible diseñar gráficamente la interfaz de usuario. La creación de toda esta herramienta Matlab, según la plataforma oficial de MathWorks en el 2018, la Guide, su objetivo de esta herramienta es permitir al usuario que sin tener ningún conocimiento del funcionamiento de este código pueda reconocer el funcionamiento de esta herramienta (Mejia, 2019).

La programación del modelamiento matemático se ha llevado a cabo dentro del programa MATLAB, el cual posee la herramienta de Interfaz Gráfica de Usuario (GUIDE), esta herramienta de la interfaz gráfica de usuario, está comprendido por una barra de herramientas como se muestra en la imagen, también por una zona de diseño cuadrículada y una paleta con componentes, además cuenta también con el editor de código asociado a la Guide, la cual se han ingresados los valores correspondientes para llevar a cabo la simulación y obtener las curvas características de la turbina hidrocinetica tipo Gorlov.



Figura 27 Programación de la GUI

Fuente: Elaboración Propia

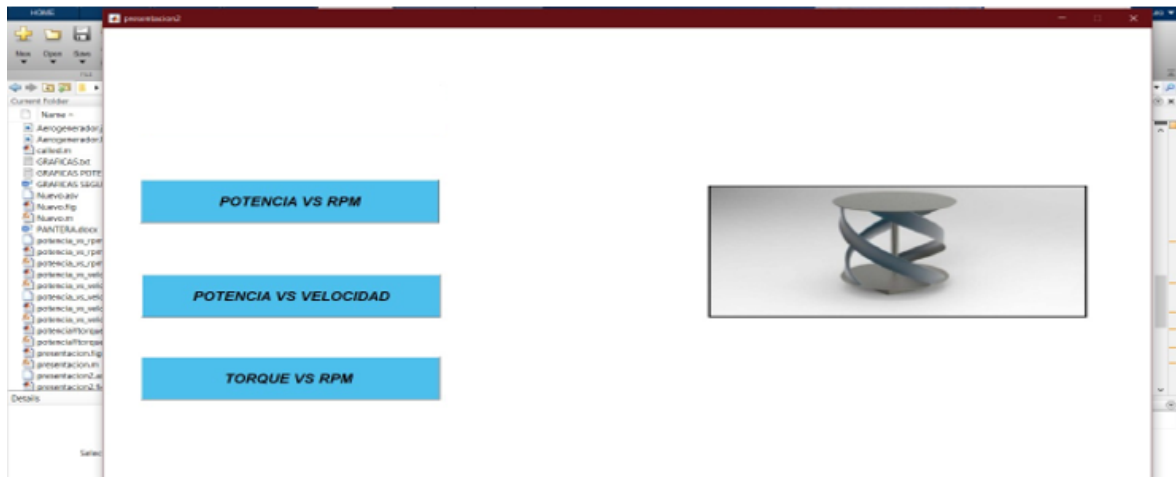


Figura 28, Programación de la GUI

Fuente: Elaboración Propia

1.4. Obtención de las Gráficas en la GUI de Matlab

Con el modelo matemático y la secuencia de nuestro algoritmo de trabajo encontraremos las gráficas.

En la figura 29, observamos la curva característica de la turbina hidrocinetica tipo Gorlov Potencia vs Velocidad.

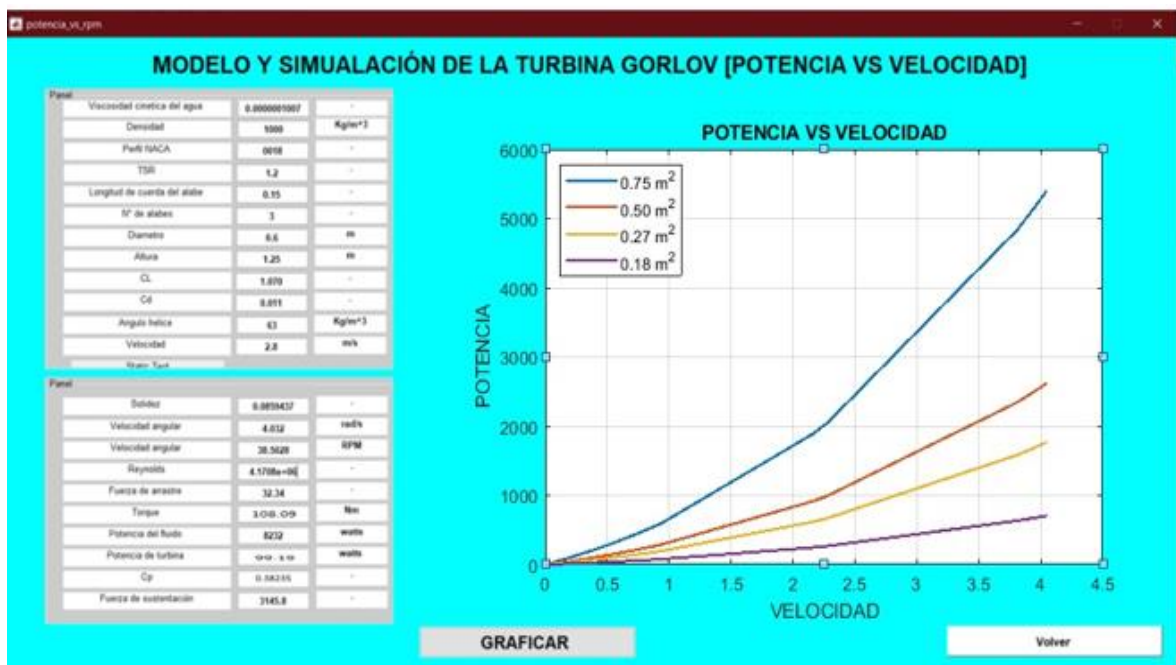


Figura 29 Representación Gráfica de la potencia vs velocidad a diferentes áreas de barrido obtenidas en la GUI.

Fuente: Elaboración Propia

En la figura 29, se detalla la representación gráfica Potencia vs Velocidad, y su creciente cuando se la referencia con el área de barrido de la Turbina Hidrocinética tipo Gorlov, para esto se utiliza la fórmula de la potencia del fluido de la ecuación (13), en el color azul con un valor de área de barrido de 0.75 m^2 , se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 4 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 5900 W. En el color rojo con un valor de área de barrido de 0.5 m^2 , se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 4 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 2500 W. En el color anaranjado con un valor de área de barrido de 0.27 m^2 , se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 4 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 1900 W. En el color morado con un valor de área de barrido de 0.18 m^2 , se observa que cuando aumenta la velocidad de la corriente de agua desde 0 hasta 4 m/s, la máxima potencia que puede generar la turbina es de aproximadamente 600 W.

Con todo esto queda claro que, a mayor área de barrido y mayores velocidades, la potencia generada por la Turbina Hidrocinética será mayor.

En la siguiente figura se representa la gráfica del Potencia vs RPM a diferentes diámetros y Ángulo de inclinación.

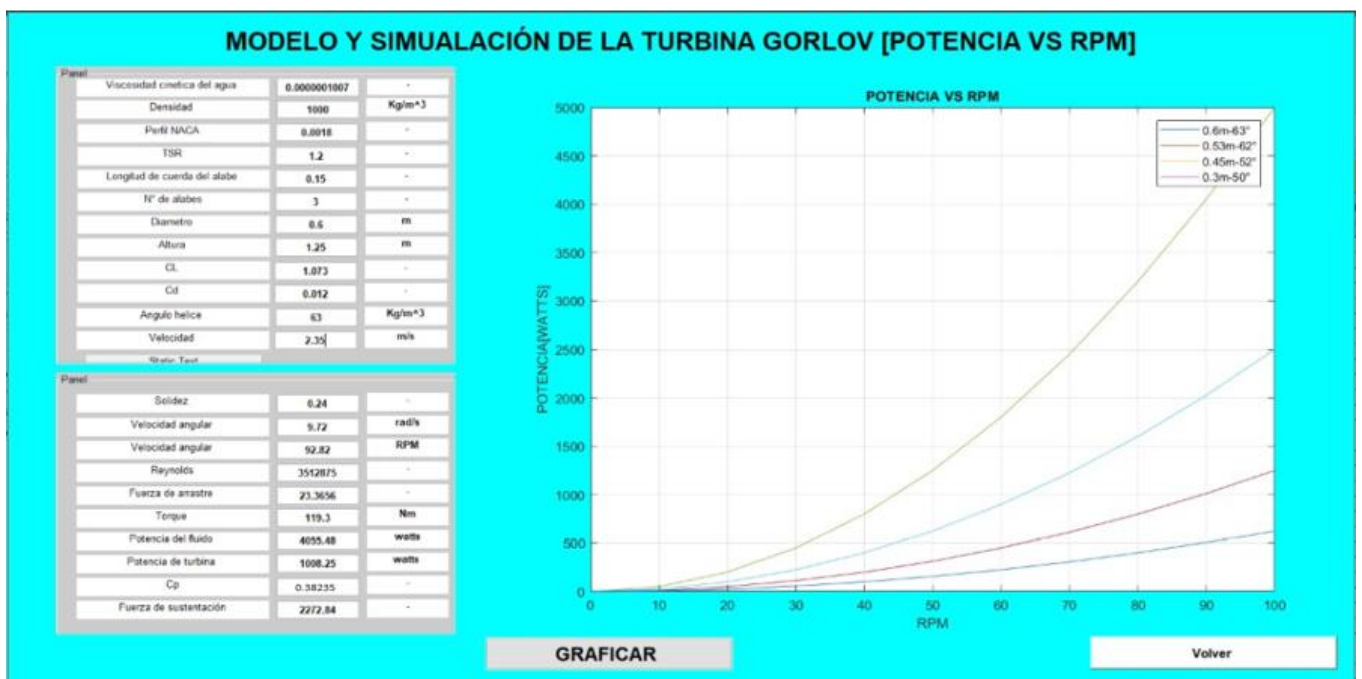


Figura 30 Representación gráfica de la Potencia vs RPM obtenidas de la GUI

Fuente: Elaboración Propia.

La Figura 30, observamos la representación gráfica Potencia vs. RPM entre diferentes diámetros y diferentes ángulos de inclinación. Para esto se utiliza la fórmula de la potencia del fluido de la ecuación (13) y la fórmula de velocidad angular de la ecuación (12). En la Figura podemos observar que la línea de color verde indica una potencia aproximada de 5000 W con un diámetro de 0.6 m y un ángulo de inclinación de 63°. Esto da por resultado obteniendo un logro de mayores potencias en la turbina hidrocinetica Gorlov.

En la siguiente figura se representa la gráfica del Torque vs RPM.

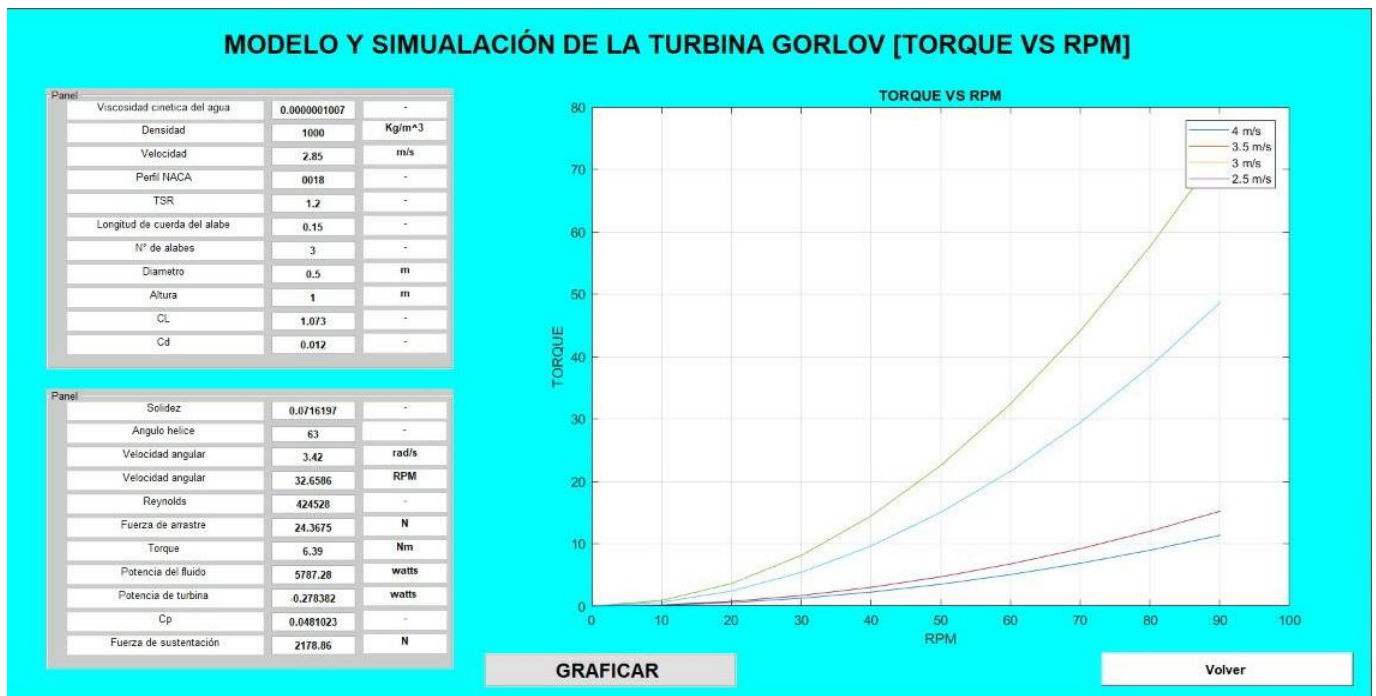


Figura 31 Representación gráfica del Torque vs RPM obtenidas de la GUI.

Fuente: Elaboración Propia.

En la figura 31 se muestra la representación gráfica del Torque vs RPM. Para esto se utiliza la fórmula del torque de la ecuación (10) y la fórmula de velocidad angular de la ecuación (12). Podemos observar que a diferentes velocidades y cuando la turbina gira a diferentes RPM, obtendremos un mejor torque, la cual, con una velocidad de 4 m/s y a 90 RPM, el torque obtenido será de un 70 N-m, condiciones de las cuales se puede obtener 5900 W de potencia. En esta condición se ubica operacionalmente la turbina en el mejor punto de funcionamiento o de mejor coeficiente de potencia, correspondiente a un 38,07%, teniendo en cuenta ya hecho mencionado en el punto 3.2.6.

1.5. Validación de resultados

Se representa a continuación la aplicación de la GUI sobre un proyecto de investigación, el cual llevan como título: “Simulación Fluidodinámica de un modelo de turbina hidrocínética tipo Gorlov”.

Se llevará a cabo mediante esta herramienta para determinar la simulación necesaria y obtener las gráficas con el fin de comparar dichos resultados con el de nuestro proyecto.

Señalamos los parámetros de investigación para obtener los resultados de dicha investigación las cuales tienen los siguientes valores:

Tabla 3. Parámetros de la turbina Gorlov

Parametros -NACA 0018													Resultados			Destacado
Viscosidad	densidad (kg / m3)	velocidad (m/s)	Longitud de cuerda del	Nº de alabes	TSR (λ)	Diametro (m)	Altura (m)	Reynolds	Cl	Cd	Fuerza de sustentación	Fuerza de arrastre	Torque	Potencia	RPM	
1.007E-07	1000	1.05	0.15	3	1	0.5	0.6	1564052	1.074	0.0121	177.61	2.00	7.22	173.64	40.11	simulación Fluidodinámica de un modelo de la turbina hidrocínética tipo Gorlov
1.007E-07	1000	1.4	0.15	3	1	0.5	0.6	2085402	1.065	0.011	313.11	3.23	12.80	411.60	53.48	
1.007E-07	1000	1.5	0.15	3	1	0.5	0.6	2234359	1.065	0.011	359.44	3.71	14.69	506.25	57.30	
1.007E-07	1000	1.54	0.15	3	1	0.5	0.6	2293942	1.065	0.011	378.86	3.91	15.48	547.84	58.82	
1.007E-07	1000	1.63	0.15	3	1	0.5	0.6	2428004	1.065	0.011	424.44	4.38	17.35	649.61	62.26	
1.007E-07	1000	1.7	0.15	3	1	0.5	0.6	2532274	1.065	0.011	461.68	4.77	18.87	736.95	64.94	
1.007E-07	1000	1.83	0.15	3	1	0.5	0.6	2725919	1.070	0.011	537.50	5.53	21.97	919.27	69.90	
1.007E-07	1000	2	0.15	3	1	0.5	0.6	2979146	1.075	0.011	645.00	6.60	26.38	1200.00	76.39	
1.007E-07	1000	2.5	0.15	3	1	0.5	0.6	3723932	1.085	0.010	1017.19	9.38	41.85	2343.75	95.49	

Fuente: (Marturet, 2012)

Tabla 4. Parámetros de la turbina Gorlov

Parametros -NACA 0018													Resultados			Destacado
Viscosidad	densidad (kg / m3)	velocidad (m/s)	Longitud de cuerda del	Nº de alabes	TSR (λ)	Diametro (m)	Altura (m)	Reynolds	Cl	Cd	Fuerza de sustentación (N)	Fuerza de arrastre	Torque	Potencia	RPM	
1.007E-07	1000	1.05	0.15	3	1	0.45	0.6	1564052	1.074	0.0121	159.85	1.80	5.85	156.28	44.56	simulación Fluidodinámica de un modelo de la turbina hidrocínética tipo Gorlov
1.007E-07	1000	1.4	0.15	3	1	0.45	0.6	2085402	1.065	0.011	281.80	2.91	10.37	370.44	59.42	
1.007E-07	1000	1.5	0.15	3	1	0.45	0.6	2234359	1.065	0.011	323.49	3.34	11.90	455.63	63.66	
1.007E-07	1000	1.54	0.15	3	1	0.45	0.6	2293942	1.065	0.011	340.98	3.52	12.54	493.06	65.36	
1.007E-07	1000	1.63	0.15	3	1	0.45	0.6	2428004	1.065	0.011	382.00	3.95	14.05	584.65	69.18	
1.007E-07	1000	1.7	0.15	3	1	0.45	0.6	2532274	1.065	0.011	415.51	4.29	15.28	663.26	72.15	
1.007E-07	1000	1.83	0.15	3	1	0.45	0.6	2725919	1.070	0.011	483.75	4.97	17.80	827.35	77.67	
1.007E-07	1000	2	0.15	3	1	0.45	0.6	2979146	1.075	0.011	580.50	5.94	21.36	1080.00	84.88	
1.007E-07	1000	2.5	0.15	3	1	0.45	0.6	3723932	1.085	0.010	915.47	8.44	33.90	2109.38	106.10	

Fuente: (Marturet, 2012)

Tabla 5. Parámetros de la turbina Gorlov

Parametros -NACA 0018													Resultados			Destacado
Viscosidad	densidad (kg / m ³)	velocidad (m/s)	Longitud de cuerda del	N° de alabes	TSR (λ)	Diametro (m)	Altura (m)	Reynolds	Cl	Cd	Fuerza de sustentación (N)	Fuerza de arrastre	Torque	Potencia	RPM	
1.007E-07	1000	1.05	0.15	3	1	0.2	0.2	1564052	1.074	0.0121	23.68	0.27	0.38	23.15	100.27	simulación Fluidodinámica de un modelo de la turbina hidrocínética tipo Gorlov
1.007E-07	1000	1.4	0.15	3	1	0.2	0.2	2085402	1.065	0.011	41.75	0.43	0.68	54.88	133.69	
1.007E-07	1000	1.5	0.15	3	1	0.2	0.2	2234359	1.065	0.011	47.93	0.50	0.78	67.50	143.24	
1.007E-07	1000	1.54	0.15	3	1	0.2	0.2	2293942	1.065	0.011	50.52	0.52	0.83	73.05	147.06	
1.007E-07	1000	1.63	0.15	3	1	0.2	0.2	2428004	1.065	0.011	56.59	0.58	0.93	86.61	155.65	
1.007E-07	1000	1.7	0.15	3	1	0.2	0.2	2532274	1.065	0.011	61.56	0.64	1.01	98.26	162.34	
1.007E-07	1000	1.83	0.15	3	1	0.2	0.2	2725919	1.070	0.011	71.67	0.74	1.17	122.57	174.75	
1.007E-07	1000	2	0.15	3	1	0.2	0.2	2979146	1.075	0.011	86.00	0.88	1.41	160.00	190.99	
1.007E-07	1000	2.5	0.15	3	1	0.2	0.2	3723932	1.085	0.010	135.63	1.25	2.23	312.50	238.73	

Fuente: (Marturet, 2012)

Con esta información del autor se ingresa en la GUI, se tienen los siguientes resultados:



Figura 32. Curva del torque vs RPM

Fuente: GUI para modelamiento y simulación de una turbina Gorlov

Podemos determinar que variando la velocidad de flujo obtenemos un mejor torque con respecto a las RPM, y también una mejor potencia como se puede observar a continuación en la siguiente gráfica, dando por resultado que mediante la GUI podemos determinar las potencias necesario la cual necesita para ser instalada la turbina hidrocinetica tipo Gorlov.

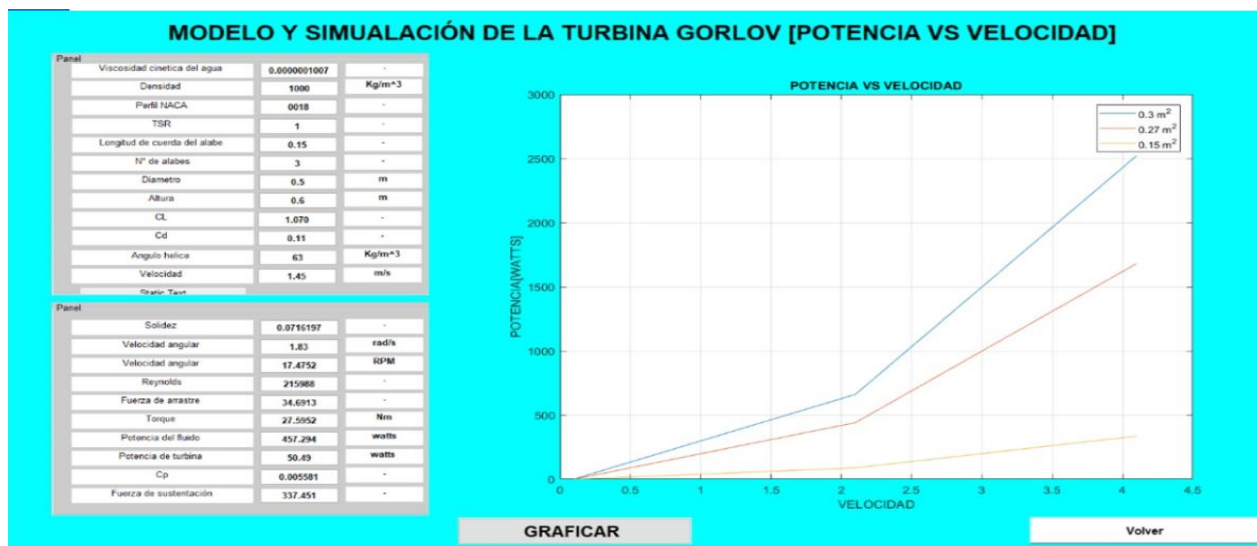


Figura 33. Curva de la potencia vs Velocidad

Fuente: GUI para modelamiento y simulación de una turbina Gorlov

1.6. Discusión de resultados

En el objetivo general que se planteó en la investigación fue realizar el modelamiento y simulación mediante software que permita obtener los parámetros de funcionamiento de la turbina hidrocinetica tipo Gorlov, y a la vez obtener los resultados para que con esta simulación sea empleado para el diseño de una turbina hidrocinetica tipo Gorlov.

Los valores obtenidos desarrollados al cálculo, coinciden con algunas investigaciones existentes en la biografía revisada, la cual la GUI sigue el algoritmo de trabajo desarrollado de manera adecuada.

Habiendo realizado el modelado y simulación se obtuvo resultados de las gráficas, obteniendo una mayor potencia, torque y RPM en cuento a su mayor área de barrida, de un diámetro $D = 0.6$ para una altura de 1.25 m, teniendo en cuenta, además la variación del ángulo helicoidal, En todos los casos estudiados se conserva el perfil del álabe tipo NACA 0018 de 0.15 m de cuerda.

Podemos verificar entonces que la GUI cumple con su objetivo, el de realizar el modelado y simulación de una turbina hidrocinetica tipo Gorlov, de esta manera se deduce satisfactorios el resultado de la investigación.

CAPÍTULO IV

CONCLUSIONES Y

RECOMENDACIONES

IV. CONCLUSIONES Y RECOMENDACIONES

2.1. CONCLUSIONES

Para realizar el análisis de comportamiento de la curva de potencia generada por la turbina hidrocínética, es necesario disponer de un gran historial de niveles, caudales, velocidades, lo que permitirá obtener un valor promedio de estos parámetros, para con la ayuda del modelo matemático escogido, se observa que mediante una gráfica la turbina hidrocínética tipo Gorlov, puede generar una potencia de hasta 5900 W, teniendo en cuenta también la velocidad del río.

Identificando todas las variables y las formulaciones matemáticas que influyeron en el proceso de la simulación de la turbina Gorlov, se llevó a cabo el desarrollo eficiente de la GUI, la cual fue presentada anteriormente, esto como punto de partida para las consideraciones de diseño y características del proyecto.

Para valores de flujos de 4 m/s y con un valor de 100 RPM la turbina puede desarrollar una potencia de 5900 W. También para flujos de 3,5 m/s y a 100 RPM se logran potencias de 2000 W. En todos los casos se observa mejores desempeños de la turbina a altas velocidades de flujo obteniendo altas potencias.

El torque generado por la turbina indica mejores desempeños de la turbina cuando gira 90 RPM y con velocidades de flujo 4 m/s., condiciones de las cuales se puede obtener 80 N-m de Torque. En esta condición se ubica operacionalmente la turbina en el mejor punto de funcionamiento.

El perfil seleccionado para el desarrollo del análisis en la GUI es el perfil NACA 0018 optimizado por tener una mayor eficiencia en turbinas helicoidales tipo Gorlov con 3 alabes en ángulos de 60°- 63°, obteniendo mayores resultados en cuanto a la potencia y al torque.

Para un ángulo helicoidal de 63° la turbina hidrocínética tipo Gorlov muestra mayores potencias, al mismo tiempo tendiendo el diámetro de la turbina de 0.6 m, lo cual también nos revela mayor torque en el eje.

Las turbinas hidrocínicas tipo Gorlov, se pueden instalar horizontal o verticalmente, a profundidades mínimas de 0.9 metros.

Realizado el análisis al caso de estudio, se puede notar que al trabajar con la turbina tipo Gorlov, se pueden obtener los mejores resultados, debido que la profundidad del río no

es un factor tan importante, es por eso que al trabajar con este tipo de turbinas se aprovecha de mejor manera la energía cinética del río.

2.2. RECOMENDACIONES

De los resultados obtenidos de esta investigación y en la prosecución para su desarrollo futuro se recomiendan las siguientes acciones:

Realizar estudios para otros perfiles de álabes a fin de obtener mayores y consecuentemente mayor potencia disponible en la turbina.

Realizar estudios variando el diámetro de giro de los álabes a fin de obtener mejores prestaciones en el desempeño de la Gorlov.

Elaborar modelos matemáticos para la simulación en el logro de determinar otras condiciones de trabajo de la turbina.

Para obtener datos de mayor confiabilidad es recomendable asistir a los puntos meteorológicos e hidrológicos cercanos al caso de aplicación, ya que de estos puntos parte la información hacia el Instituto Nacional de Meteorología e Hidrología (INAMHI).

Referencias

- Arbulú, V. J., & García, R. G. (2018). *Modelamiento y simulación del motor gasolinero Suzuki M16A mediante el uso de una interfaz gráfica de usuario en el taller de ciencias térmicas de la USS-Chiclayo*. Chiclayo: Universidad Señor de Sipan.
- Bulla, A. S., & Romero, S. N. (2018). *Fabricación de un Modelo de turbina Gorlov a escala de Laboratorio*. Bogota: Universidad Santo Tomas.
- Campos, M. R. (2017). *Análisis Técnico - Economico, diseño y evaluación experimental de la implantación de una turbina hidrocinetica para generación eléctrica*. Quito: Universidad Escuela Politécnica Nacional.
- Colina, A. O. (2018). *Diseño de un rotor hidrocínético tipo Gorlov para el suministro de energía eléctrica a una vivienda ubicada en una zona no interconectada del departamento de Casanare*. Bogota: Universidad Santo Tomas.
- Galpin, P., & Bakker, A. (2008). The new wave of fluids technology. *ANSYS ADVANTAGE, II(2)*, 17.
- Garcia, A., Toicen, c., & Rojas, D. (2014). *Diseño de un banco de enayos de turbina hidrocínética tipo Gorlov para el laboratorio de termo-fluido del instituto Universitario de Tecnología del estado Bolívar, estado Bolívar*. Bolívar: Instituto Universitario de tecnología del estado Bolívar.
- LE-Quesne, R. M. (2018). *"Diseño y ensayo de una turbina Gorlov para extracción de energía en canales de regadío"*. Chile: Universidad Técnica Federico Santa María .
- Leticia, L. R. (2015). *Procedimiento de Seleccion tecnológica para pequeñas instalaciones de aprovechamiento hidrocínético en canales hidráulicos*. Santa Clara: Universidad Central "Marta Abreu" de las Villas.

- Linares, c. k. (2019). *Diseño de una Turbina hidrocinetica para pruebas en el canal de ensayos hidrodinamicos de la Universidad Nacional* . Bogota: Universidad Santo Tomás .
- Maldonado, Q. f. (2005). *Diseño de una turbina de río para la generación de electricidad en el distrito de Mazán-Región loreto*. Lima: Universidad Nacional Mayor de San Marcos.
- Martínez, C. E. (2014). *Diseño y optimización por medio de analisis exergéticos de una turbina hidrocínética para generación de energía eléctrica a partir de fluidos de bajo caudal*. México: Universiad Nacional Autónoma de México .
- Marturet, P. G. (2012). *Simulación Fluidodinámica de un modelo de turbina hidrocínética tipo Gorlov*. Puerto Ordáz: Universidad Nacional Politécinca "Antonio José de Sucre".
- Mejia, V. O. (2019). *Algoritmos genéricos para el diseño de una minicentral eléctrica solar utilizando una interfaz de programación*. Chiclayo: Universidad Señor de Sipan.
- Mendoza, A. J. (2018). *Diseño de una turbina hidrocínética de eje vertical para el análisis de la dependencia de los principales parámetros geométricos que influyen en el coeficiente de potencia*. Ecuador: Universidad Técnica Estatal de Quevedo.
- Mendoza, Y. P. (2017). *Diseño de generador hidroeléctrico portable para zonas rurales* . Lima: Pontificia Universidad Catolica del Perú.
- Muñoz, Í. D. (2013). *Desarrollo de una interfaz gráfica de usuario en matlab para el diseño de ascensores eléctricos Disae 1.0*. Leganés: Universidad Carlos III de Madrid.

- Peña, G. V. (2013). *"Diseño de una turbina hidro-cinética para aprovechamiento energético de ríos no caudalosos"*. Piura: Universidad de Piura.
- Pérez, P. Á. (2019). *"Estudio y simulación de una turbina hidrocínética para generar energía eléctrica en la Universidad Politécnica Salesiana sede Guayaquil"*. Guayaquil: Universidad Politécnica Salesiana.
- Quispe, C. A., & Maquera, Q. J. (2019). *Diseño y construcción de un módulo de laboratorio para el análisis de los parámetros de una bomba centrífuga de 1 HP como turbina*. Puno: Universidad Nacional del Altiplano Puno.
- Ramos, F. s. (2012). *"Aplicación del programa MATLAB en la resolución de ecuaciones diferenciales aplicado a la materia de cálculo Tres"*. Guayaquil, Ecuador: Universidad Católica de Santiago de Guayaquil.
- Rivadeneira, M. D. (2015). *Modelamiento y simulación de la operación de generadores que emplean turbinas hidrocínicas en ríos de bajo caudal*. Quito: Universidad Politécnica Salesiana Sede Quito.
- Rodríguez, B. L. (2018). *Diseño, construcción y optimización de turbinas hidrocínicas de ríos y canales para generación de energía eléctrica*. Arequipa: Universidad Nacional San Agustín de Arequipa.
- Serres, P. (2010). *Simulación tridimensional de flujo del fluido en turbina tipo Gorlov y diseño estructural*. Puerto Ordáz: Universidad Nacional Experimental Politécnica "Antonio José de Sucre".
- Yumpo, B. J. (2014). *Diseño de una turbina hidrocínetica sumergible para la generación de energía eléctrica de 5 KW de potencia en el caserío de Maino, Distrito de San Isidro de Maino -Región Amazonas*. Chiclayo: Universidad Señor De Sipán.

Zubialde, G. I. (2016). *Diseño de una turbina hidrocínética tipo savonius*. Pinar del Río:
Universidad de Pinar del Río.

ANEXOS

ANEXO N° 1
Código de Ética del Colegio de Ingenieros del Perú (CIP)



CÓDIGO DE ETICA DEL CIP

**APROBADO EN LA III SESIÓN ORDINARIA DEL CONGRESO NACIONAL DE CONSEJOS
DEPARTAMENTALES DEL PERÍODO 1998 - 1999
EN LA CIUDAD DE TACNA 22, 23 Y 24 ABRIL 1999**

ANEXO N° 2

Código de Ética de Investigación de la USS



www.uss.edu.pe

**CÓDIGO DE ÉTICA DE
INVESTIGACIÓN DE LA USS**

VERSIÓN 03

RATIFICADO POR ACUERDO DE CONSEJO UNIVERSITARIO CON RESOLUCIÓN
RECTORAL N° 0851-2017/USS

CHICLAYO - PERÚ

Versión: 03	Código: VRI-CE	F. Implementación:	Página 1 de 29
Elaborado por: Dirección de Investigación	Revisado por: Planificación y Desarrollo Institucional – Asesoría Legal		Ratificado con Resolución Rectoral N° 0851- 2017/USS

ANEXO N° 3

Código de programación portada de la GUI

```
function edit56_Callback(hObject, eventdata, handles)
% hObject    handle to edit56 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global RPM;
RPM=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit56 as text
%        str2double(get(hObject,'String')) returns contents of edit56 as
a double

% --- Executes during object creation, after setting all properties.
function edit56_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit56 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn (hObject, eventdata, handles)
% hObject    handle to axes1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called
grid
xlabel('VELOCIDAD')
ylabel('TORQUE')
title ('TORQUE VS VELOCIDAD')

% Hint: place code in OpeningFcn to populate axes1

function edit57_Callback (hObject, eventdata, handles)
% hObject    handle to edit57 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global alfa;
alfa=str2double(get(hObject,'String'));

% Hints: get(hObject,'String') returns contents of edit57 as text
%        str2double(get(hObject,'String')) returns contents of edit57 as
a double
```

```

% --- Executes during object creation, after setting all properties.
function edit57_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit57 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit58_Callback (hObject, eventdata, handles)
% hObject    handle to edit58 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit58 as text
%         str2double(get(hObject,'String')) returns contents of edit58 as
a double

% --- Executes during object creation, after setting all properties.
function edit58_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit58 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit59_Callback (hObject, eventdata, handles)
% hObject    handle to edit59 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit59 as text
%         str2double(get(hObject,'String')) returns contents of edit59 as
a double

% --- Executes during object creation, after setting all properties.
function edit59_CreateFcn (hObject, eventdata, handles)
% hObject    handle to edit59 (see GCBO)
% eventdata reserved - to be defined in a future version of MATLAB

```

```

% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
function varargout = presentacion(varargin)
% PRESENTACION MATLAB code for presentacion.fig
%       PRESENTACION, by itself, creates a new PRESENTACION or raises the
existing
%       singleton*.
%
%       H = PRESENTACION returns the handle to a new PRESENTACION or the
handle to
%       the existing singleton*.
%
%       PRESENTACION ('CALLBACK', hObject, eventData, handles,) calls the
local
%       function named CALLBACK in PRESENTACION.M with the given input
arguments.
%
%       PRESENTACION('Property','Value',) creates a new PRESENTACION or
raises the
%       existing singleton*. Starting from the left, property value pairs
are
%       applied to the GUI before presentacion_OpeningFcn gets called. An
%       unrecognized property name or invalid value makes property
application
%       stop. All inputs are passed to presentacion_OpeningFcn via
varargin.
%
%       *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
%       instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help presentacion

% Last Modified by GUIDE v2.5 26-Jul-2020 18:53:58

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @presentacion_OpeningFcn, ...
                  'gui_OutputFcn',  @presentacion_OutputFcn, ...
                  'gui_LayoutFcn',  [], ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else

```

```

    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before presentation is made visible.
function presentacion_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to presentacion (see VARARGIN)

% Choose default command line output for presentacion
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes presentacion wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = presentacion_OutputFcn(hObject, eventdata, handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes during object creation, after setting all properties.
function edit1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close(presentacion);
presentacion2

```



```

unction varargout = potencia_vs_rpm(varargin)
% POTENCIA_VS_RPM MATLAB code for potencia_vs_rpm.fig
%     POTENCIA_VS_RPM, by itself, creates a new POTENCIA_VS_RPM or
raises the existing
%     singleton*.
%
%     H = POTENCIA_VS_RPM returns the handle to a new POTENCIA_VS_RPM or
the handle to
%     the existing singleton*.
%
%     POTENCIA_VS_RPM('CALLBACK',hObject,eventData,handles,...) calls
the local
%     function named CALLBACK in POTENCIA_VS_RPM.M with the given input
arguments.
%
%     POTENCIA_VS_RPM('Property','Value',...) creates a new
POTENCIA_VS_RPM or raises the
%     existing singleton*. Starting from the left, property value pairs
are
%     applied to the GUI before potencia_vs_rpm_OpeningFcn gets called.
An
%     unrecognized property name or invalid value makes property
application
%     stop. All inputs are passed to potencia_vs_rpm_OpeningFcn via
varargin.
%
%     *See GUI Options on GUIDE's Tools menu. Choose "GUI allows only
one
%     instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help potencia_vs_rpm

% Last Modified by GUIDE v2.5 28-Jul-2020 22:29:10

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @potencia_vs_rpm_OpeningFcn, ...
                  'gui_OutputFcn',  @potencia_vs_rpm_OutputFcn, ...
                  'gui_LayoutFcn',  [] , ...
                  'gui_Callback',    []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State, varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before potencia_vs_rpm is made visible.

```

```

function potencia_vs_rpm_OpeningFcn(hObject, eventdata, handles,
varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to potencia_vs_rpm (see VARARGIN)

% Choose default command line output for potencia_vs_rpm
handles.output = hObject;

% Update handles structure
guidata(hObject, handles);

% UIWAIT makes potencia_vs_rpm wait for user response (see UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command line.
function varargout = potencia_vs_rpm_OutputFcn(hObject, eventdata,
handles)
% varargout  cell array for returning output args (see VARARGOUT);
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in pushbutton1.
function pushbutton1_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
close(potencia_vs_rpm);
presentacion2

% --- Executes on button press in pushbutton2.
function pushbutton2_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
%
global velocidad;
global c;
global viscosidad;
global D;
global H;
global densidad;
global CL;
global Cd;
global alfa;
global FS;
global N;
global TSR;
global T;

```

```

global w;
global pt;
global pf;
global Wr;
%
ARRASTRE=Cd*(densidad/2)*(D*H)*velocidad^2
SUSTENTACION=CL*(densidad/2)*(D*H)*velocidad^2
R=(velocidad*c)/viscosidad
T=(D/2)*(sin(alfa)-D*cos(alfa))
pf=0.5*densidad*(D*H)*velocidad^3
solidez=(N*c)/pi*D
w=(60*TSR*velocidad)/pi*D
Wr=w*(pi*2/60)
pt=T*(Wr)
Cp=pt/pf

%
set(handles.edit51,'string',ARRASTRE)
set(handles.edit49,'string',SUSTENTACION)
set(handles.edit55,'string',R)
set(handles.edit47,'string',T)
set(handles.edit46,'string',pf)
set(handles.edit53,'string',solidez)
set(handles.edit56,'string',w)
set(handles.edit48,'string',pt)
set(handles.edit54,'string',Wr)
set(handles.edit50,'string',Cp)
legend('0.75 m^2','0.5 m^2','0.27m^2','0.18m^2')
xlabel('VELOCIDAD')
ylabel('POTENCIA[WATTS]')
title('POTENCIA VS VELOCIDAD')
function edit2_Callback(hObject, eventdata, handles)
% hObject    handle to Magnitud1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of Magnitud1 as text
%        str2double(get(hObject,'String')) returns contents of Magnitud1
%        as a double

% --- Executes during object creation, after setting all properties.
function Magnitud1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Magnitud1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
%            called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit3_Callback(hObject, eventdata, handles)

```

```

% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit3 as text
%         str2double(get(hObject,'String')) returns contents of edit3 as a
double

% --- Executes during object creation, after setting all properties.
function edit3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit4_Callback(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit4 as text
%         str2double(get(hObject,'String')) returns contents of edit4 as a
double

% --- Executes during object creation, after setting all properties.
function edit4_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit4 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit5_Callback(hObject, eventdata, handles)
% hObject    handle to edit5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit5 as text

```

```
%      str2double(get(hObject,'String')) returns contents of edit5 as a
double
```

```
% --- Executes during object creation, after setting all properties.
```

```
function edit5_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to edit5 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
```

```
%      See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
```

```
get(0,'defaultUiControlBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
```

```
end
```

```
function edit6_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit6 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit6 as text
```

```
%      str2double(get(hObject,'String')) returns contents of edit6 as a
double
```

```
% --- Executes during object creation, after setting all properties.
```

```
function edit6_CreateFcn(hObject, eventdata, handles)
```

```
% hObject    handle to edit6 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    empty - handles not created until after all CreateFcns
called
```

```
% Hint: edit controls usually have a white background on Windows.
```

```
%      See ISPC and COMPUTER.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
```

```
get(0,'defaultUiControlBackgroundColor'))
```

```
    set(hObject,'BackgroundColor','white');
```

```
end
```

```
function edit7_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit7 (see GCBO)
```

```
% eventdata  reserved - to be defined in a future version of MATLAB
```

```
% handles    structure with handles and user data (see GUIDATA)
```

```
% Hints: get(hObject,'String') returns contents of edit7 as text
```

```
%      str2double(get(hObject,'String')) returns contents of edit7 as a
double
```

```
% --- Executes during object creation, after setting all properties.
```

```

function edit7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit8_Callback(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit8 as text
%       str2double(get(hObject,'String')) returns contents of edit8 as a
double

% --- Executes during object creation, after setting all properties.
function edit8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit9_Callback(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit9 as text
%       str2double(get(hObject,'String')) returns contents of edit9 as a
double

% --- Executes during object creation, after setting all properties.
function edit9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit10_Callback(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit10 as text
%        str2double(get(hObject,'String')) returns contents of edit10 as
a double

% --- Executes during object creation, after setting all properties.
function edit10_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit10 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit11_Callback(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit11 as text
%        str2double(get(hObject,'String')) returns contents of edit11 as
a double

% --- Executes during object creation, after setting all properties.
function edit11_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit11 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

```

```

        set(hObject, 'BackgroundColor', 'white');
end

function edit12_Callback(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit12 as text
%        str2double(get(hObject, 'String')) returns contents of edit12 as
a double

% --- Executes during object creation, after setting all properties.
function edit12_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit12 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

function edit13_Callback(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject, 'String') returns contents of edit13 as text
%        str2double(get(hObject, 'String')) returns contents of edit13 as
a double

% --- Executes during object creation, after setting all properties.
function edit13_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject, 'BackgroundColor'),
get(0, 'defaultUiControlBackgroundColor'))
    set(hObject, 'BackgroundColor', 'white');
end

```



```

function edit14_Callback(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global D;
D=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit14 as text
%        str2double(get(hObject,'String')) returns contents of edit14 as
a double

% --- Executes during object creation, after setting all properties.
function edit14_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit15_Callback(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global N;
N=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit15 as text
%        str2double(get(hObject,'String')) returns contents of edit15 as
a double

% --- Executes during object creation, after setting all properties.
function edit15_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit15 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit16_Callback(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

global H;
H=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit16 as text
%       str2double(get(hObject,'String')) returns contents of edit16 as
a double

% --- Executes during object creation, after setting all properties.
function edit16_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit16 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit17_Callback(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Cd;
Cd=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit17 as text
%       str2double(get(hObject,'String')) returns contents of edit17 as
a double

% --- Executes during object creation, after setting all properties.
function edit17_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit17 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit18_Callback(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global CL;
CL=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit18 as text

```

```

%         str2double(get(hObject,'String')) returns contents of edit18 as
a double

% --- Executes during object creation, after setting all properties.
function edit18_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit18 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit19_Callback(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global c;
c=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit19 as text
%         str2double(get(hObject,'String')) returns contents of edit19 as
a double

% --- Executes during object creation, after setting all properties.
function edit19_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit19 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit20_Callback(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global densidad;
densidad=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit20 as text
%         str2double(get(hObject,'String')) returns contents of edit20 as
a double

```

```

% --- Executes during object creation, after setting all properties.
function edit20_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit20 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit21_Callback(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit21 as text
%         str2double(get(hObject,'String')) returns contents of edit21 as
a double
global viscosidad;
viscosidad=str2double(get(hObject,'String'));

% --- Executes during object creation, after setting all properties.
function edit21_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit21 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit22_Callback(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global velocidad;
velocidad=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit22 as text
%         str2double(get(hObject,'String')) returns contents of edit22 as
a double

% --- Executes during object creation, after setting all properties.

```

```

function edit22_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit22 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
cl

function edit23_Callback(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global TSR;
TSR=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit23 as text
%        str2double(get(hObject,'String')) returns contents of edit23 as
a double

% --- Executes during object creation, after setting all properties.
function edit23_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit23 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%         See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit24_Callback(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global perfil;
perfil=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit24 as text
%        str2double(get(hObject,'String')) returns contents of edit24 as
a double

% --- Executes during object creation, after setting all properties.
function edit24_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit24 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit46_Callback(hObject, eventdata, handles)
% hObject    handle to edit46 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Pf;
Pf=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit46 as text
%       str2double(get(hObject,'String')) returns contents of edit46 as
a double

% --- Executes during object creation, after setting all properties.
function edit46_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit46 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit47_Callback(hObject, eventdata, handles)
% hObject    handle to edit47 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global TORQUE;
TORQUE=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit47 as text
%       str2double(get(hObject,'String')) returns contents of edit47 as
a double

% --- Executes during object creation, after setting all properties.
function edit47_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit47 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit48_Callback(hObject, eventdata, handles)
% hObject    handle to edit48 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global Pt;
Pt=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit48 as text
%     str2double(get(hObject,'String')) returns contents of edit48 as
a double

% --- Executes during object creation, after setting all properties.
function edit48_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit48 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit49_Callback(hObject, eventdata, handles)
% hObject    handle to edit49 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global FS;
FS=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit49 as text
%     str2double(get(hObject,'String')) returns contents of edit49 as
a double

% --- Executes during object creation, after setting all properties.
function edit49_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit49 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))

```

```

        set(hObject,'BackgroundColor','white');
end

function edit50_Callback(hObject, eventdata, handles)
% hObject    handle to edit50 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global CP;
CP=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit50 as text
%        str2double(get(hObject,'String')) returns contents of edit50 as
a double

% --- Executes during object creation, after setting all properties.
function edit50_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit50 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit51_Callback(hObject, eventdata, handles)
% hObject    handle to edit51 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global ARRASTRE;
ARRASTRE=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit51 as text
%        str2double(get(hObject,'String')) returns contents of edit51 as
a double

% --- Executes during object creation, after setting all properties.
function edit51_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit51 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

```



```

function edit52_Callback(hObject, eventdata, handles)
% hObject    handle to edit52 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global alfa;
alfa=str2double(get(hObject,'String'));
% Hints: get(hObject,'String') returns contents of edit52 as text
%        str2double(get(hObject,'String')) returns contents of edit52 as
a double

% --- Executes during object creation, after setting all properties.
function edit52_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit52 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit53_Callback(hObject, eventdata, handles)
% hObject    handle to edit53 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit53 as text
%        str2double(get(hObject,'String')) returns contents of edit53 as
a double

% --- Executes during object creation, after setting all properties.
function edit53_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit53 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit54_Callback(hObject, eventdata, handles)

```

```

% hObject    handle to edit54 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global w;
w=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit54 as text
%        str2double(get(hObject,'String')) returns contents of edit54 as
a double

% --- Executes during object creation, after setting all properties.
function edit54_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit54 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit55_Callback(hObject, eventdata, handles)
% hObject    handle to edit55 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
global R;
R=str2double(get(hObject,'String'))

% Hints: get(hObject,'String') returns contents of edit55 as text
%        str2double(get(hObject,'String')) returns contents of edit55 as
a double

% --- Executes during object creation, after setting all properties.
function edit55_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit55 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%        See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit56_Callback(hObject, eventdata, handles)
% hObject    handle to edit56 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB

```

```

% handles      structure with handles and user data (see GUIDATA)
global RPM;
RPM=str2double(get(hObject,'String'))
% Hints: get(hObject,'String') returns contents of edit56 as text
%         str2double(get(hObject,'String')) returns contents of edit56 as
a double

% --- Executes during object creation, after setting all properties.
function edit56_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit56 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%           See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

% --- Executes during object creation, after setting all properties.
function axes1_CreateFcn(hObject, eventdata, handles)
% hObject      handle to axes1 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called
grid
xlabel('VELOCIDAD')
ylabel('TORQUE')
title('TORQUE VS VELOCIDAD')

% Hint: place code in OpeningFcn to populate axes1

function edit57_Callback(hObject, eventdata, handles)
% hObject      handle to edit57 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      structure with handles and user data (see GUIDATA)
global alfa;
alfa=str2double(get(hObject,'String'));

% Hints: get(hObject,'String') returns contents of edit57 as text
%         str2double(get(hObject,'String')) returns contents of edit57 as
a double

% --- Executes during object creation, after setting all properties.
function edit57_CreateFcn(hObject, eventdata, handles)
% hObject      handle to edit57 (see GCBO)
% eventdata    reserved - to be defined in a future version of MATLAB
% handles      empty - handles not created until after all CreateFcns
called

```

```

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit58_Callback(hObject, eventdata, handles)
% hObject     handle to edit58 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit58 as text
%     str2double(get(hObject,'String')) returns contents of edit58 as
a double

% --- Executes during object creation, after setting all properties.
function edit58_CreateFcn(hObject, eventdata, handles)
% hObject     handle to edit58 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function edit59_Callback(hObject, eventdata, handles)
% hObject     handle to edit59 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     structure with handles and user data (see GUIDATA)

% Hints: get(hObject,'String') returns contents of edit59 as text
%     str2double(get(hObject,'String')) returns contents of edit59 as
a double

% --- Executes during object creation, after setting all properties.
function edit59_CreateFcn(hObject, eventdata, handles)
% hObject     handle to edit59 (see GCBO)
% eventdata   reserved - to be defined in a future version of MATLAB
% handles     empty - handles not created until after all CreateFcns
called

% Hint: edit controls usually have a white background on Windows.
%     See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUiControlBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end
end

```