INSTITUTO TECNOLÓGICO SUPERIOR DE LERDO

DIVISIÓN DE POSGRADO

PROTOTIPO PARA EL ESTUDIO DE ACABADOS POR MEDIO DE LASER EN DIFERENTES TIPOS DE MATERIALES

T E S I S

PARA OBTENER EL TÍTULO DE MAESTRÍA EN INGENIERÍA MECATRÓNICA

PRESENTA

ING.YASMIN LIZETH RODRÍGUEZ LUEVANOS

Asesor: M.C. Ernesto II Castro Juárez

La excelencia académica el servicio de la sociedad

Ciudad Lerdo, Durango. Junio del 2022.

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico Superior de Lerdo

" 2022: Año de Ricardo Flores Magón, Precursor de la Revolución Mexicana"

Cd. Lerdo, Durango, 6/Junio/2022
Oficio No. DA/xxx/2022

ING. YASMIN LIZETH RODRÍGUEZ LUÉVANOS
ALUMNO DE LA MAESTRÍA EN INGENIERÍA MECATRÓNICA
PRESENTE

Se le comunica que la comisión revisora integrada por los cuatro sinodales, revisó y aprobó en su totalidad el trabajo de tesis:

"PROTOTIPO PARA EL ESTUDIO DE ACABADOS POR MEDIO DE LASER EN DIFERENTES TIPOS DE MATERIALES"

Presentado por usted para obtener el título de:

MAESTRO EN INGENIERÍA MECATRÓNICA.

Por lo anterior y de acuerdo a los lineamientos profesionales se da el trámite legal para que proceda a la impresión del trabajo presentado.

ATENTAMENTE
"LA EXCELENCIA ACADÉMICA AL SERVICIO DE LA SOCIEDAD"

M.C. JOSÉ ÁNGEL MÉNDEZ ORTEGA
DIRECTOR ACADÉMICO

INSTITUTO TECNOLÓGICO SUPERIOR DE LERDO
DIRECCIÓN ACADÉMICA

C.p. Archivo
JAMO/CRM

R1/03/18

F-GE-01-006

Av. Tecnológico s/n, Col. Periférico, Cd. Lerdo, Dgo., C.P. 35150
Tels. (871) 725-23-71 725-57-79 725-58-02
www.tecnm.mx | www.lerdo.tecnm.mx

CACEI
Consejo de Acreditación de la Enseñanza de la Ingeniería, A.C.

EDUCACIÓN
SECRETARÍA DE EDUCACIÓN PÚBLICA

TECNOLÓGICO NACIONAL DE MÉXICO

Instituto Tecnológico Superior de Lerdo

" 2022: Año de Ricardo Flores Magón, Precursor de la Revolución Mexicana"

Cd. Lerdo, Durango, **03/Junio/2022**

M. C. CARLOS URIEL FLORES PRINCE
JEFE DE LA DIVISÓN DE POSGRADO
P R E S E N T E

Por este conducto hacemos de su conocimiento que después de haber procedido a la revisión rigurosa y detallada de la tesis del:

ING. YASMIN LIZETH RODRÍGUEZ LUÉVANOS

Cuyo título es:

"Prototipo para el estudio de acabados por medio de laser en diferentes tipos de materiales"

Este jurado considera su impresión.

A T E N T A M E N T E:

M. C. Ernesto II Castro Juárez
Presidente

Dr. Hesner Coto Fuentes
Secretario

M. I. M. Arturo Serrano Hernández
Vocal Propietario

Dr. Raymundo Juárez del Toro
Vocal Suplente

Av. Tecnológico s/n, Col. Periférico, Cd. Lerdo, Dgo., C.P. 35150
Tels. (871) 725-23-71 725-57-79 725-58-02
www.tecnm.mx I www.lerdo.tecnm.mx

CACEI
Consejo de Acreditación de la Enseñanza de la Ingeniería, A.C.

DEDICATORIA

La siguiente tesis la dedico a mi esposo Francisco. Quien desde que lo conocí ha sido un compañero incondicional en todo momento, me ha motivado cada día ser mejor persona en todos los aspectos de mi vida. Animandome a seguir avanzando por muy difícil que se visualice cualquier situación.

A mi hija Cibeles, quien sin darse cuenta se adaptó a nuevos horarios y situaciones. Por lo cual agradezco mucho siendo yo consciente lo difícil que fue para ella debido a su edad.

A mis padres Cresencio y Magdalena por su tiempo, esfuerzo y sobre todo por ser un gran apoyo en mi vida. Por lo cual no tengo palabras para agradecerles todo lo que han hecho por mi.

A mis hermanos Yuriana, Yasiel y Fabian por sus consejos y su incondicional cariño. Pero sobre todo a Dios por permitir tener en mi vida a estas maravillosas personas.

AGRADECIMIENTOS

Quiero agradecer a mi asesor el M.C Ernesto II Castro quien me apoyo en todo momento a la elaboración de mi tesis, por su tiempo, paciencia y conocimientos gracias infinitas.

A los docentes miembros de la división de posgrado quienes con infinita dedicación y profesionalismo compartieron su tiempo y conocimientos, muchas gracias.

A el M.S.C. César Ríos Marmolejo jefe de la división de posgrado quien en todo momento estuvo al pendiente de nuestra situación escolar muchas gracias.

Agradezco a el M.C. Luis Felipe Chairez Fernández jefe de división de Ingeniería en Sistemas Automotrices quien me brindo la oportunidad de ejercer como docente en la carrera.

Agradezco a mis compañeros que se convirtieron en amigos, conociendo a algunos de ellos en el aula de clase y a otros tantos fuera de ella. Siempre brindándome su apoyo en todo momento y de manera incondicional.

ÍNDICE GENERAL

íNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1: TEMAS PRELIMINARES

1.1 RESUMEN

Actualmente se utiliza el grabado de materiales con láser para fines estéticos o fabricación de piezas para diversas aplicaciones. La selección del tipo de material se realiza de manera empírica teniendo pérdidas económicas. Por ello, se planteó realizar un prototipo para el estudio de acabados de materiales utilizando diversos métodos de análisis de diversas propiedades como lo son difusividad, temperatura de ignición, etc.

El siguiente trabajo muestra la obtención de valores numéricos como lo son el número de Prandlt, Nusselt, Grashof y Rayleigh enfocados a mecanismos de transferencia de calor y aplicados a simulaciones térmicas. Asi mismo las pruebas físicas y las realizadas mediante interfaz a materiales como madera blanda, polipropileno y PVC.

El proceso se realiza mediante una interfaz visual capaz de controlar servomotores montados en un brazo robótico que controla el movimiento de un módulo láser. Así mismo se hace la adquisición de temperaturas y cambio de algunos parámetros como lo son tiempo, temperatura ambiente, temperatura inicial del material cuando se expone al haz de luz láser, difusividad térmica entre otros que permiten obtener un análisis de los materiales a los que se expone la luz.

1.2 ABSTRACT

Laser engraving of materials is currently used for aesthetic purposes or the manufacture of parts for various applications. The selection of the type of material is carried out empirically, which incurs economic losses. For this reason, it was proposed to make a prototype for the study of material finishes using various methods of analysis such as diffusivity, ignition temperature, etc.

The following work shows the obtaining of numerical values such as the Prandtl, Nusselt, Grashof and rayleigh numbers, which are measures related to heat transfer mechanisms, useful when applied in thermal simulations. Likewise, the physical tests and those carried out through interface to materials such as soft wood, polypropylene and pvc.

The process is carried out through a visual interface capable of controlling servomotors mounted on a robotic arm that controls the movement of a laser module. Additionally, the acquisition of data related to parameters such as ambient temperature, temperature of the material over time (e.g. before, during, and after exposure to the laser light beam), thermal diffusivity, and others will allow an analysis of how the materials respond when exposed to laser light.

1.3 INTRODUCCIÓN

En la búsqueda de facilitar su existencia, el hombre ha recabado varias alternativas para conocer el comportamiento de los materiales a través de diferentes dispositivos. Al hablar de corte o grabado lo más usual es el proceso corte por chorro de agua, sin embargo, no es el único. Existen diversas formas de trabajar al momento de proporcionar los acabados: Corte con láser, corte con arco de plasma, oxicorte, entre otros.

El presente documento se enfoca a la técnica requerida para dar acabados a los materiales. La mayoría de propiedades mecánicas y físicas de los materiales suelen mostrar dependencia con la temperatura. Los procesos industriales enfocados a tratamiento de materiales con láser más tradicionales son el corte, la soldadura.

Considerando al láser como una fuente de energía al cual se puede o no aplicar movimiento, un láser por sus siglas en el idioma inglés (light amplification by stimulated emission of radiation) genera un haz de luz. El color de la luz láser viene determinado por su longitud de onda**,** abarca desde el azul en la franja de las longitudes de onda inferiores hasta el rojo. Cada dispositivo láser opera dentro de una determinada longitud de onda, por ejemplo, ciertos láseres médicos tienen su campo de actividad en la región del espectro azul y los reproductores de discos compactos normalmente usan láseres infrarrojos.

En el Capítulo 1 se presentan temas preliminares, introducción al tema, descripción de la problemática y descripciones básicas de lo que se plantea realizar. De igual manera se presentan diversos proyectos con planteamientos parecidos brindando una idea del seguimiento al desarrollo. En cuanto al fundamento teórico aplicado para el desarrollo, aplicación y recopilación del proyecto.

En el Capítulo 2 se da la metodología propuesta para la realización de objetivos planteados a través de actividades que permitirá el desarrollo y detección de errores. Los métodos planteados se presentan de manera escrita o mediante diagramas de flujo.

El Capítulo 3 presenta la metodología comenzando por la creación de materiales en la biblioteca del software NX de SIEMENS. La etapa del desarrollo consta de la obtención de parámetros mediante una interfaz visual utilizados posteriormente en simulaciones térmicas. Mediante una interfaz visual de software especializado se adquieren y proporcionan parámetros para la simulación térmica mediante el método de nodos.

El Capítulo 4 da los resultados obtenidos en simulaciones implementadas en el prototipo necesarias para la validación de la investigación. Los objetivos específicos y general se cumplieron, dando pauta a la simulación del comportamiento del calor lo cual se puede enfocar en la industria y en el sector educativo.

1.4 ANTECEDENTES (ESTADO DEL ARTE)

Los cortadores láser operan al dirigir grandes cantidades de energía en una superficie muy pequeña. La potencia del láser se define como la velocidad a la que la energía es entregada por el rayo y se mide en unidades de Joules / segundo o vatios. Algunas aplicaciones actuales del cortador láser son:

- Corte por láser de metales, plásticos y tejidos.
- Cirugía láser de la piel.
- Cirugía láser en órganos internos y otros tejidos delicados.
- Grabado con láser de artesanías y litografía.

En un principio el agua a presión no se empleó para cortes, su principal función fue limpiar los almacenes de carbón y limpieza de fundiciones. Se planteó la idea de un intensificador de presión del agua y fue así como poco a poco y en base a experimentos se fue logrando llegar al primer corte por agua, cabe recalcar que debido a la baja presión no se obtuvieron resultados benéficos.

Al no obtener los resultados esperados se siguieron realizando más pruebas y se agregó un aditivo de partículas abrasivo al chorro de agua de esta manera se pudieron obtener más cortes en diferentes materiales no metálicos y blandos (como alimentos, papeles, plásticos y espuma). Para cortar piedra, vidrio, metales y otros materiales de alta dureza, FLOW inventó, desarrolló y patentó el corte con chorro de agua a alta.

"El proceso consiste en la focalización del haz láser en un punto del material que se desea tratar, para que éste funda y evapore lográndose así el corte con una determinada potencia procedente del generador y de un sistema de conducción, un grupo óptico se encarga de focalizar el haz con un diámetro determinado sobre un punto de interés del material a tratar. La gran desventaja que presenta el corte de chapa por láser frente a otros procedimientos reside principalmente en el espesor máximo que se puede cortar." (Ortega ,2007).

El avance de la tecnología láser enfocada al corte se utiliza para metales, goma, vidrio, cuero y madera los cuales son susceptibles de ser cortados, hay que tener en cuenta factores como la calidad del material o posibles recubrimientos. Una desventaja al momento de los cortes, es no contar con parámetros para la realización de cortes. No se cuenta con algún estudio que indique el comportamiento de los materiales al acabado del láser.

Un láser es un generador y amplificador de luz, la luz del láser es de una sola longitud de onda es decir no se desvía o amplia por lo cual el corte o grabado es más preciso. Las aplicaciones del láser no sólo son corte y grabado de materiales, se utiliza en la guía para el tendido de tuberías, construcción de edificios totalmente planos y medición de distancias.

El mecanizado láser se basa en la elaboración de un láser con gran potencia dirigido a la pieza, se podrían conseguir maquinados sumamente pequeños de acuerdo al haz del láser. El maquinado en superficies metálicas y cerámicas consiste en erosionar el material en varias capas obteniendo la forma geométrica deseada. La gran versatilidad del láser nos da como ventaja amplia variedad al cortar contornos de cualquier forma y complejidad.

“Los requisitos básicos en cualquier láser son idénticos. En primer lugar, se necesita un medio que presente la estructura de nivel de energía deseada para permitir el efecto láser. La tecnología del mecanizado láser se basa en la generación de un rayo láser de alta potencia que es dirigido contra la pieza mediante un sistema de espejos de alta precisión” (Díaz del Castillo, 2011).

En febrero del 2017, el dimensionamiento e implementación de una máquina CNC de corte por láser para optimizar la calidad de trabajos en acrílico de hasta 5 mm en varias industrias. Se planteó la necesidad de aumentar la producción y diferentes estilos de cortes. Originalmente los cortes sumamente rudimentarios daban este tipo de maquinados, como resultado se obtenía producción artesanal. El corte por láser es la aplicación industrial más común en el mecanizado de materiales. Los metales, cerámicas, polímeros y compuestos pueden cortarse o perforarse con láser independientemente de la dureza.

Las principales técnicas empleadas fueron: observación científica y experimentación de campo. Anteriormente no existió algún estudio respaldado donde se tengan las características de como actúan las propiedades de los materiales al verse afectados por algún tipo láser. Se recopilaron datos sobre la velocidad de avance a mayor espesor menor será la velocidad, los espesores deben ser mayores a 2 mm de no ser así sufren perforaciones debido a la potencia del láser, fue recomendado utilizar la máxima velocidad y reducir la resolución de grabado. En la máquina de corte por láser de CO_2, se realizaron varias pruebas en láminas de acrílico en las cuales se logró cortar, en una sola pasada, láminas de 2 mm. Y con tres pasadas láminas de 5 mm con una potencia máxima del 80% con una velocidad de 60 mm/min.

La obtención de resultados permitió aumentar la calidad del trabajo con cortes de 5mm de espesor. Los cortes implementados fueron circunferencias y cuadrados ambos de diferentes tamaños, estableciendo las medidas en valores máximos para circunferencias y cuadrados, la finalidad de establecer medidas fue el ahorro de material y disminución de gastos por sobrantes del mismo. (Cruz Carrillo, 2017)

“Entre las aplicaciones industriales del láser para procesado de materiales se calcula que en torno al 60% de la actividad está dedicada al corte. Una de las industrias que mayormente absorbe esta actividad es la industria del automóvil. Sin embargo, en la actualidad se aplica de manera creciente en la industria de la estructura metálica” (Mantilla, 2017).

Otro ejemplo de diseño y programación de un eje rotatorio para la máquina de corte láser NTC TLM 610 siendo requerido un proceso de fabricación nuevo o mejorar la implementación de los mismos. Fue necesario la actualización de la máquina de corte mediante la implementación de dos grados de libertad. De esta manera se adapta una boquilla para helio u oxígeno. Los gases como el helio u oxigeno el cual se encontraría en una boquilla coaxial al láser son los que se pueden utilizar para cortar el acero. Una gran desventaja se comentó el tamaño reducido y el corte de diámetro muy parecido a la boquilla.

A pesar de ser muy útil se encontró que tiene una gran desventaja ya que en este caso el espesor máximo es de 30 mm para madera, para metales y plásticos 40mm. Otros procedimientos como el oxicorte, corte por plasma, electroerosión o corte por chorro de agua permiten cortar espesores mayores que el láser. En cuanto a la selección del mecanismo se optó por uno de suma precisión, dicha selección establecida con cálculos como potencias y recorridos en 5 ejes (Mantilla López, 2017).

El trabajo con tecnología láser no es exclusivo de la industria, los escultores se han planteo la problemática constante, o saber el comportamiento de algunos metales con gran impacto en la elaboración de piezas de escultura. Demostrar que el trabajo en algunos metales puede ser sencillo en cuanto la finalidad del corte, la humanidad se ha desarrollado en cuanto sus descubrimientos, vinculándose con los metales y su comportamiento.

Las técnicas de modificación superficial con láser realizadas en superficies de aluminio sus características internas no se ven afectadas es una buena elección debido a sus propiedades. Algunos tipos de láser como CO_2 y Nd: YAG, el segundo mencionado es más apto para sistemas de fabricación. debido a su alta conductividad térmica se dificulta el proceso con aluminio. Lo deseable en cualquier proceso de recubrimiento es tener las características termo físicas de cualquier material. Algunos materiales frecuentes o para el recubrimiento láser son los cerámicos para mejorar su resistencia y evitar la corrosión su dificultad es el elevado punto de fusión. (Fernández Carrasquilla,2008).

De acuerdo con Lascano Román “La primera prueba de grabado se realizó en madera MDF utilizando la potencia máxima del láser con una velocidad de gravado de 1200 mm/min mediante el software LASER GBRL, como resultado se obtuvo que la superficie de la madera se quemaba intensamente y no se distinguía los detalles de la imagen.” (Lascano Román, 2018).

La implementación de láser para realizar diferentes actividades como corte o grabado tubo mejor resultado en materiales como MDF y cuero, el haz debe de estar cerca al material donde se desea trabajar. La realización de diferentes pruebas una

de ellas en cuero da como resultado (a una máxima potencia del láser en dicho material se obtuvo una intensidad de grabado alta). La tercera prueba de corte se realizó en fomi utilizando la potencia máxima del láser con una velocidad de 800mm/min como resultado se obtuvo un corte limpio sin sobrecalentamiento del láser. Se debe de mencionar que también se realizó una prueba de precisión para verificar que realmente se encontrara en la posición que se indicaba mediante la programación del software.

Sus principales beneficios fueron:

- Disminución de tiempos muertos debido a procesos artesanales.
- Cortes limpios y con mayor estética.
- Realizo grabado con diferentes tonalidades.

Se eligió aluminio para la elaboración de la estructura ya que cuenta con numerosas propiedades

- Material ligero.
- Resistente a la corrosión.
- Resiste a los productos químicos.
- No es toxico.
- Su precio en el mercado es accesible.
- Es el tercer elemento que tiene mayor abundancia en la corteza terrestre.

Con el inicio de la tecnología láser se comenzó a experimentar para ver si era factible en la industria esta hacia el año de 1993 con el corte de cuero el cual cabe mencionar que se realizaba para aquella fecha de manera manual, se menciona que el láser aumentara la producción y en menor tiempo debido a la facilidad con de corte por tratarse de un láser de CO_2. Otra aplicación del rayo láser de CO_2 fue el ajuste de la resistencia de un material conductor en forma de película delgada (Alfonso, 1993).

El láser se ha utilizado como lo mencionamos no solo para corte y grabado, existe la ablación láser técnica para obtener materiales en forma de película delgada que se han conocido desde los 60s, demostró que la radiación láser intensa se podría

utilizar en el crecimiento de materiales en forma de películas delgadas por consiguiente la técnica consiste en emplear pulsos láser al volumen y esta a su vez los absorbe en una área determinada del tal forma que en la superficie se crea calor y esta a su vez evapora el material superficial.

Bajo condiciones experimentales adecuadas, es posible conseguir que la película obtenida tenga la misma composición que el blanco. La transferencia congruente es una consecuencia de la alta rapidez inicial del calentamiento y evaporación del blanco. (Álvarez, 2018).

El diodo láser infrarrojo es un dispositivo que irradia calor, por este motivo se pretendió medir el calor en el dispositivo, cabe mencionar que la temperatura y calor no son lo mismo ya que la temperatura es una medida o indicación de que tan frio o caliente es un objeto, el calor es la energía transferida de un objeto a otro. Por lotanto el proceso de corte por láser es muy importante en la concentración de energía y con ello habrá una precisión en el perfil geométrico de la pieza. Se estableció que para el proceso de grabado el prototipo opera a velocidades altas y potencias bajas, se comentó entonces que el corte y grabado dichos procesos se realizan por medio de la modulación de ancho de pulso.

Durante el tiempo que el módulo láser se encuentra actuando sobre el material que procesa hay desprendimiento de gases. Al finalizar el trabajo los resultados cumplieron con su objetivo, mejoro los sistemas anteriores donde se utilizaba un láser de CO_2, el cambio permitió procesos más finos y de mayor precisión (Chávez, 2015).

En cuanto a los estudios realizados en las simulaciones “Los análisis térmicos son de gran importancia en una amplia gama de productos para su control. Concluyendo el ensamblaje de los sistemas se procede a realizar los cálculos térmicos considerando convección y conducción esto tomando en cuenta el coeficiente de transferencia térmica en algunos materiales y las temperaturas a las cuales se encontraba el sistema. El análisis térmico en un conjunto de técnicas analíticas que estudian el comportamiento térmico de los materiales. Cuando un material se

calienta o se enfría su estructura o composición química pueden sufrir cambios.” (Serrano, 2017).

Dentro del software utilizado el cual fue NX de SIEMENS se estableció una metodología para la elaboración del estudio de análisis térmico del elemento finito. Dentro del análisis térmico se aplicaron principios de conducción, destacando que los cambios en los materiales pueden ser variados y realizar cambios en sus propiedades.

1.5 DEFINICIÓN DEL PROBLEMA

La mayor parte de los fenómenos físicos se encuentran representados de forma matemática. Gracias a los avances en tecnologías de corte y softwares especializados en diseño, surge la necesidad de implementar nuevos procesos para la selección de parámetros a analizar en diferentes materiales.

Anteriormente los acabados de materiales se realizaban mediante chorro de agua u oxicorte, paulatinamente con el crecimiento de la industria surge la necesidad de acabados más precisos. Conforme el paso del tiempo aumenta la necesidad de implementar tecnología láser, mediante el corte o grabado de piezas. Ya que la versatilidad de acabados precisos requiere conocimientos en diferentes áreas como lo son transferencia de calor, diseño, programación y electrónica. Al volverse complicado realizar acabados con precisión, considerando el comportamientos y propiedades físicas de cada material.

En los últimos años la tecnología mediante el uso del láser ha aumentado de manera exponencial en diferentes ámbitos, particularmente en la industria debido a la versatilidad de la misma. Dentro del ramo textil comúnmente llamadas maquiladoras se emplean para el proceso de corte y acabado de telas. En la industria metal-mecánica con el corte de estructuras metálicas. Si bien no se cuenta con estudios que proporcionen parametros establecidos para el corte o grabado de piezas, se pueden presentar limitaciones para analizar determinados materiales debido al costo elevado de algunos como lo son el oro o plata.

1.6 JUSTIFICACIÓN

Debido a la importancia de conocer las propiedades de los acabados para materiales, es necesario analizar el comportamiento con la exposición a un haz de luz láser. Surge la necesidad de conocer la transferencia de calor a altas temperaturas todo esto es clave para su mejor manejo o mejoramiento del mismo.

En estudios previamente analizados la mayoría de las características se obtienen mediante prueba y error. Seleccionando empíricamente parámetros del láser. Algunos parámetros son la potencia, velocidad de avance, espesor del material.

Se desarrollará un prototipo para fines de investigación, el cual nos permita conocer propiedades de los materiales, es decir que sucede internamente con la estructura del material (cómo se comporta al aplicarle calor mediante un haz de luz).

Cabe mencionar que al ser para fines de investigación se pudiera dar la posibilidad de usarlo posteriormente para la mejor selección de un material a grabar o cortar, debido a que se contara con parámetros y espesores establecidos. La línea de investigación del programa de posgrado donde se desarrolla el presente proyecto corresponde a diseño e implementación para sistemas de control y adquisición de datos, desarrollando interfaces visuales y simulaciones en diferentes softwares.

1.7 OBJETIVOS: GENERAL Y ESPECÍFICOS

1.7.1 OBJETIVO GENERAL

Desarrollar un prototipo que permita el estudio de acabados de materiales con láser que permita arrojar resultados del comportamiento de los mismos, mediante análisis y simulaciones térmicas.

1.7.2 OBJETIVOS ESPECÍFICOS

- Recopilar información relacionada con los materiales a analizar.
- Recopilar información de corte y grabado con láser.
- Analizar la información recopilada.
- Simular análisis térmicos en NX.
- Desarrollar propuesta del prototipo para el movimiento del láser.
- Buscar proveedores y selección de equipo.
- Adquirir equipo.
- Instalar, configurar y programar prototipo.
- Desarrollar pruebas térmicas.

CAPÍTULO 2: FUNDAMENTO TEÓRICO

2.1 Historia del láser

La historia del láser más significativa se remonta al año de 1916, cuando Albert Einstein estudió y predijo el fenómeno de emisión estimulada en los átomos. Un átomo que recibe luz de la misma longitud de onda de la que puede emitir, es estimulado a emitirla en ese instante.

Theodore H. Maiman en 1960 consiguió la producción de luz láser utilizando un cristal de rubí, óxido de aluminio con pequeñas impurezas de cromo. El láser ha sido uno de los descubrimientos científicos que con mayor rapidez se ha introducido en la industria tecnológica en diversos ámbitos, lo que nos da una idea de la importancia del hallazgo. tan sólo un año más tarde se publicaron los primeros resultados sobre el uso del láser para coagular tejido de la retina, en contraste con el uso de lámparas de Xenón. En 1964, cuatro años después del descubrimiento del láser, se comenzó a investigar el uso del láser en odontología. Tan sólo nueve años después del descubrimiento del láser, en 1969, se construyó el primer prototipo de impresora láser en Xerox, que comenzó a comercializarse en 1975. Como último ejemplo del impacto tecnológico que significó el descubrimiento del láser podemos mencionar que, en 1971 once años después de la aparición del láser, se instaló en General Motors el primer escáner de códigos de barras basado en un láser de helio-neón, que había sido descubierto en 1961. Toda esta enorme influencia del láser en la industria tecnológica, y también en la investigación científica, e incluso en la misma vida cotidiana, es una de las razones por las que el láser se considera uno de los descubrimientos más importantes de la historia reciente. (Ibarra Villalon, 2017).

Un láser típico consta de los siguientes elementos:

- Cavidad óptica resonante: Medio en el que la luz puede circular, consta habitualmente de un par de espejos.
- Medio activo: Puede ser sólido, liquido o gas su función es amplificar la luz.
- Bombeo: Brinda un cierto aporte de energía para poder amplificar la luz.

A la hora de clasificar los distintos tipos de láser se pueden seguir muchos criterios distintos:

- Longitud de onda
- Potencia de emisión
- Tamaño

De acuerdo con estos criterios podemos dividir la mayoría de los láseres en tres familias:

- Gas
- Estado sólido
- Semiconductor

2.1.1 Láser estado sólido

El término estado sólido se refiere a láseres de estructura cristalina o de materiales vítreos dopados con ion apropiado, no incluye los láseres semiconductores. Este tipo de láser se caracteriza por tener como medio activo una varilla o una plancha sólida de aislante cristalino ligeramente impurificado. Es el constituyente impurificado lo que proporciona la estructura energética requerida para producir el efecto láser. La energía eléctrica se convierte en luz de banda ancha mediante una descarga de arco o un filamento caliente. Cuando el medio activo de láser está contenido en un resonador óptico, la ganancia resultante de la inversión de población causa oscilación y se emite radiación coherente intensa.

2.1.2 Diodo Láser

Se denomina así porque sus propiedades son características de los diodos eléctricos. También se conoce como láser semiconductor. Se utilizan en discos compactos, impresoras láser, lectores de códigos de barras, comunicación y óptica, además de contar con una potencia elevada. Cabe destacar que son muy accesibles tanto en el mercado como económicamente.

2.1.3 Láser de gas (helio-neón)

Este láser fue diseñado por William Bennett, Ali Javan y Donald Herriot en 1961 y se constituyó en el primer láser de gas de emisión continua. Los láseres de He-Ne tienen una vida media que puede alcanzar decenas de miles de horas de funcionamiento. Los láseres de helio-neón se han utilizado en impresión láser, lectores de códigos de barras, alineamiento, metrología. Es compacto, portátil y sencillo con multitud de aplicaciones que no requieren una potencia alta, su medio activo es una mezcla de helio 85% y neón 15%. (Valenzuela, 2011). Las partes principales se muestran en la Figura 1.

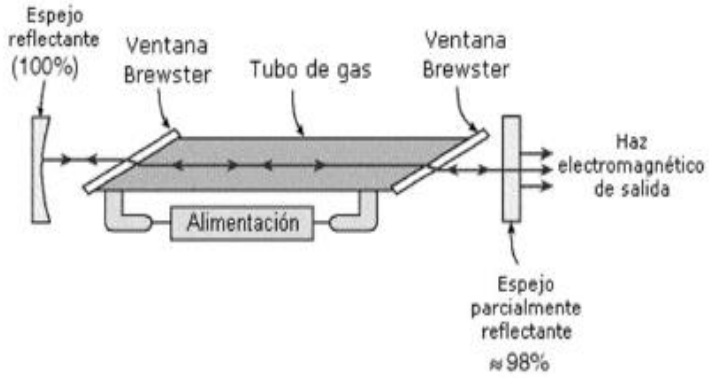


Figura 1 Estructura del láser Helio-Neón

2.2 Soldadura de materiales con láser

La soldadura con láser por conducción, se utiliza en aplicaciones relacionadas a la union de metales preciosos como los son el oreo y la plata. En cuanto a la potencia se destina una baja verificando las propiedades termicas del material seleccionado.Una vez fundiendo la superficie del material seleccionado.Asi mismo la transferencia de calor por conveción considera la geometria del material a soldar.Considerando que la conducción termica no es direccional influye en la penetracion de la soldadura siendo 2mm como máximo.(Valenzuela,2011):

2.3 Corte de materiales por láser

Las propiedades termicas, geometricas y mecanicas del propio material son los que determinan la aplicación del corte por láser. Uno de los procesos que se inducen termicamente es el corte por láser, elevando la temperatura del material. Ya que el haz de luz focalizadio incide en la cara a cortar, de la misma manera el gas expulsa el material a incidir.La calidad del haz de luz es sumamente imprtante de ello depende un buen resultado en el corte(Cruz Carrillo, 2018).

2.4 Arduino uno

Arduino es una plataforma enfocada al desarrollo de hardware para uso libre o código abierto (open –source), se basa en software y hardware flexibles fáciles de utilizar. Se produce en diferentes tipos de placas, varia en el microcontrolador, cantidad de entradas y salidas Figura 1. Arduino permite grabar instrucciones las cuales crean programas.

El microcontrolador posee una interfaz de entrada y una interfaz de salida como principal característica la sencillez tanto del software como del lenguaje de programación. Arduino se puede utilizar para la creación de proyectos y controlar dispositivos como lo son servomotores, sensores de humedad, temperatura entre otros. (Enríquez, 20219).

Figura 1 Placa Arduino Uno

2.5 Módulo Bluetooth HC-05

El módulo Bluetooth HC-05 (maestro/esclavo), se utiliza en aplicaciones de comunicación inalámbrica. Es capaz de recibir conexiones desde dispositivos como PC / tables y generar conexiones hacia otros dispositivos con bluetooth. El módulo hc-05 puede alimentarse con un voltaje de entrada entre 3.3 y 6V. La Figura 2 muestra la forma de conexión entre Arduino y modulo Bluetooth.

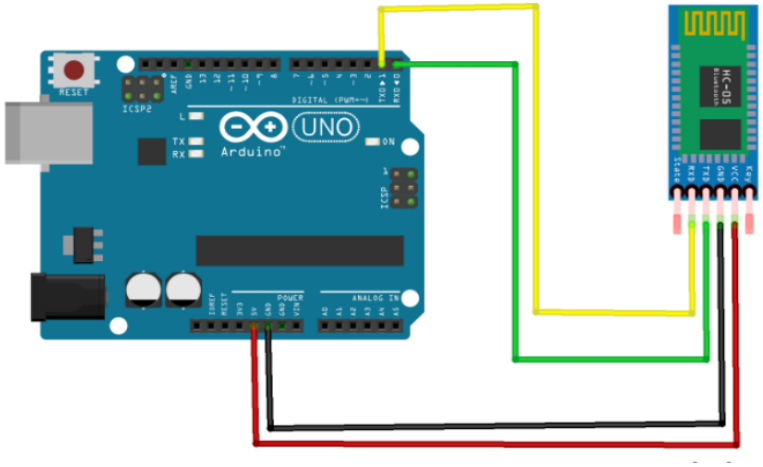


Figura 2 Conexión entre Arduino y módulo Bluetooth HC-05

2.6 Servomotor

Un servomotor es un actuador rotativo con componentes electromecánicos y electrónicos el cual consistes en un juego de engranes, un motor eléctrico y una tarjeta de control. Posee tres cables dos de ellos sirven para la alimentación (cable café y rojo), el restante para el control (cable amarillo). Controlado mediante el puerto PWM (modulación por ancho de pulso), la cual se obtiene de la tarjeta Arduino a través de la señal de control, Figura 3 .

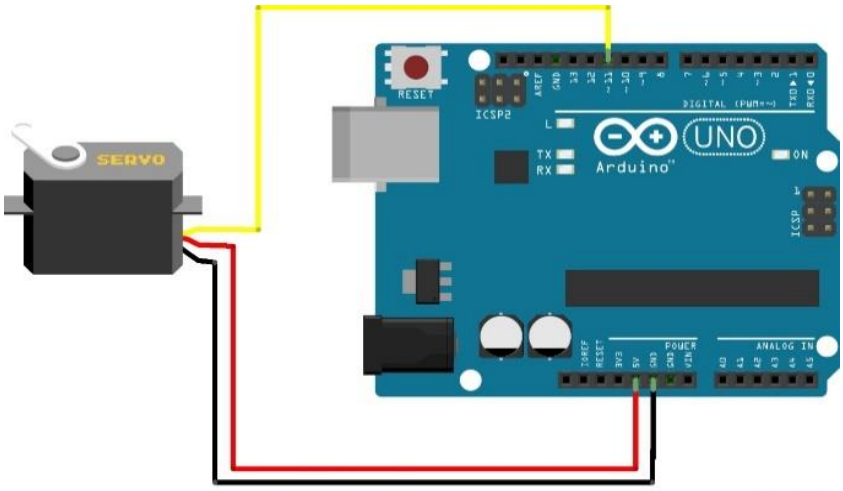


Figura 3 Conexión entre servomotor y Arduino

2.7 Sensor de temperatura LM35

El sensor LM35 permite medir temperatura, siendo la salida de voltaje proporcional a la temperatura. Realiza una conversión de análogo a digital, los LM35 no requieren ningún tipo de calibración externa. Proporciona precisión de $\pm\frac{1}{4}°C$ a temperatura ambiente -55°C a 150°C, en la Figura 4 se muestra la configuración del LM335. (Fernandez,2019)

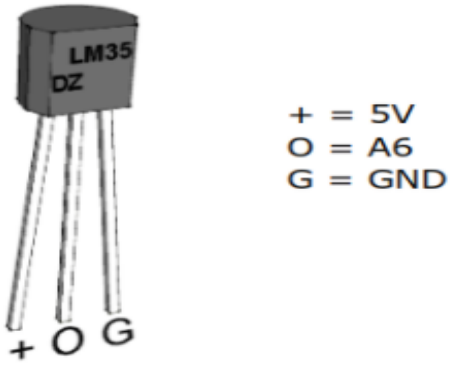


Figura 4 Sensor LM35

2.8 Cámara termográfica Ti32_Ti25_Ti10

La cámara termográfica muestra la imagen de como se distribuye el calor en una superficie determinada. El termograma permite analiza la distribución de las temperaturas en diferentes ubicaciones y seleccionar un punto en específico. Ya que adquirir un correcto termograma depende de la superficie libre a seleccionar.

La obtención del termograma es un método muy sencillo, no produce cambios en la superficie a analizar. Los ensayos termográficos se clasifican en el apartado de ensayos no destructivos. La cámara TI32_TI25_TI10 tiene una sensibilidad térmica de 0.045 °C a 30 °C (45 mK) y un rango de temperatura de entre -20 ° y 600 °C (entre -4 ° y 1112 °F), como se muestra en la Figura 5 .

Figura 5 Cámara termográfica

2.9 Brazo robótico

Un robot es una máquina controlada por ordenador y programada para moverse**,** manipular objetos y realizar trabajos a la vez que interacciona con su entorno. El término Robot, proviene de la palabra checa robota. Los parámetros para especificar el número de reacciones en una estructura denominados grados de libertad generalmente coinciden con el número de eslabones en cadena cinemática

Su constitución física es similar a la de un brazo humano, mientras que el sistema está elaborado por una estructura mecánica. (Concepto, 2022).

Las aplicaciones principales son las siguientes:

- Soldadura por arco
- Montaje
- Limpieza / Pulverización
- Máquina tendiendo
- Manejo de materiales
- Embalaje

2.10 MIT APP INVENTOR

App Inventor es un entorno para la creación de aplicaciones de desarrollo web, utilizada en sistemas operativos Android. Siendo una aplicación desarrollada por Google en conjunto con MIT (Instituto Tecnológico de Massachusetts), su creación se basa en la investigación informática educativa al ser muy versátil y de fácil acceso. La Figura 6 muestra el logotipo de MIT App Inventor.

Figura 6 Logotipo MIT APP INVENTOR

2.11 NX de SIEMENS

NX de Siemens es un software implementado para diseño industrial proporciona la oportunidad de crear diseños y simulaciones con mejor calidad. Como nombre previo fue unigraphics, creado por General Motors. La Figura 7 muestra el logotipo de NX.

Con el paso del tiempo este programa fue adquirido por Siemens PLM Software en donde se nombró como SIEMENS NX PL. La prioridad de NX es el diseño, implementa todos los pasos que conlleva la realización de un producto desde el concepto hasta la ingeniería. NX en el diseño industrial es la solución en el desarrollo, ya que se crean productos más innovadores y con muchas más ventajas

evitando costos innecesarios. En la industria manufacturera NX tiene un gran impacto mejorando gran variedad de operaciones. (NX Solución,2022)

Figura 7 Logotipo NX de SIEMENS

2.12 MATLAB

MATLAB por su abreviatura en inglés (matrix laboratory), es un software de alto nivel creado con la finalidad de realizar cálculos simbólicos y numéricos. Creado por Cleve Mole en los setentas y adquirido en los ochentas por Math Works, la Figura 8 muestra el logotipo de MATLAB. (Casado,2022).

MATLAB ofrece gran infinidad de funciones siendo las más comunes:

- Diseñar
- Modelar
- Simular
- Análisis de datos

Las aplicaciones más comunes en la industria son:

- Manipulación de matrices.
- Representación de datos y funciones.
- Implementación de algoritmos.
- Creación de interfaces de usuario (GUI).
- Comunicación con programas en otros lenguajes.

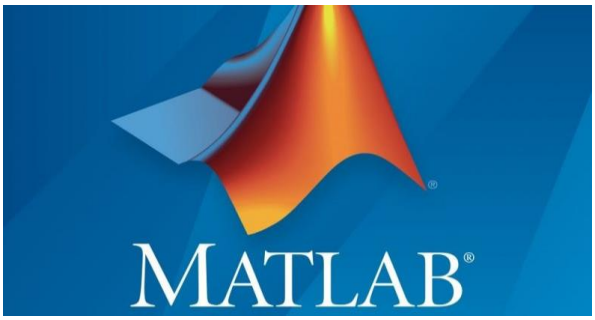


Figura 8 Logotipo MATLAB

CAPÍTULO 3: DESARROLLO DE LA INVESTIGACIÓN

3.1 Metodología

La presente metodología fue dividida en seis etapas. Las cuales se desarrollarán en la parte teórico-práctico. En la Figura 9 se muestra las etapas y una breve explicación de las actividades.

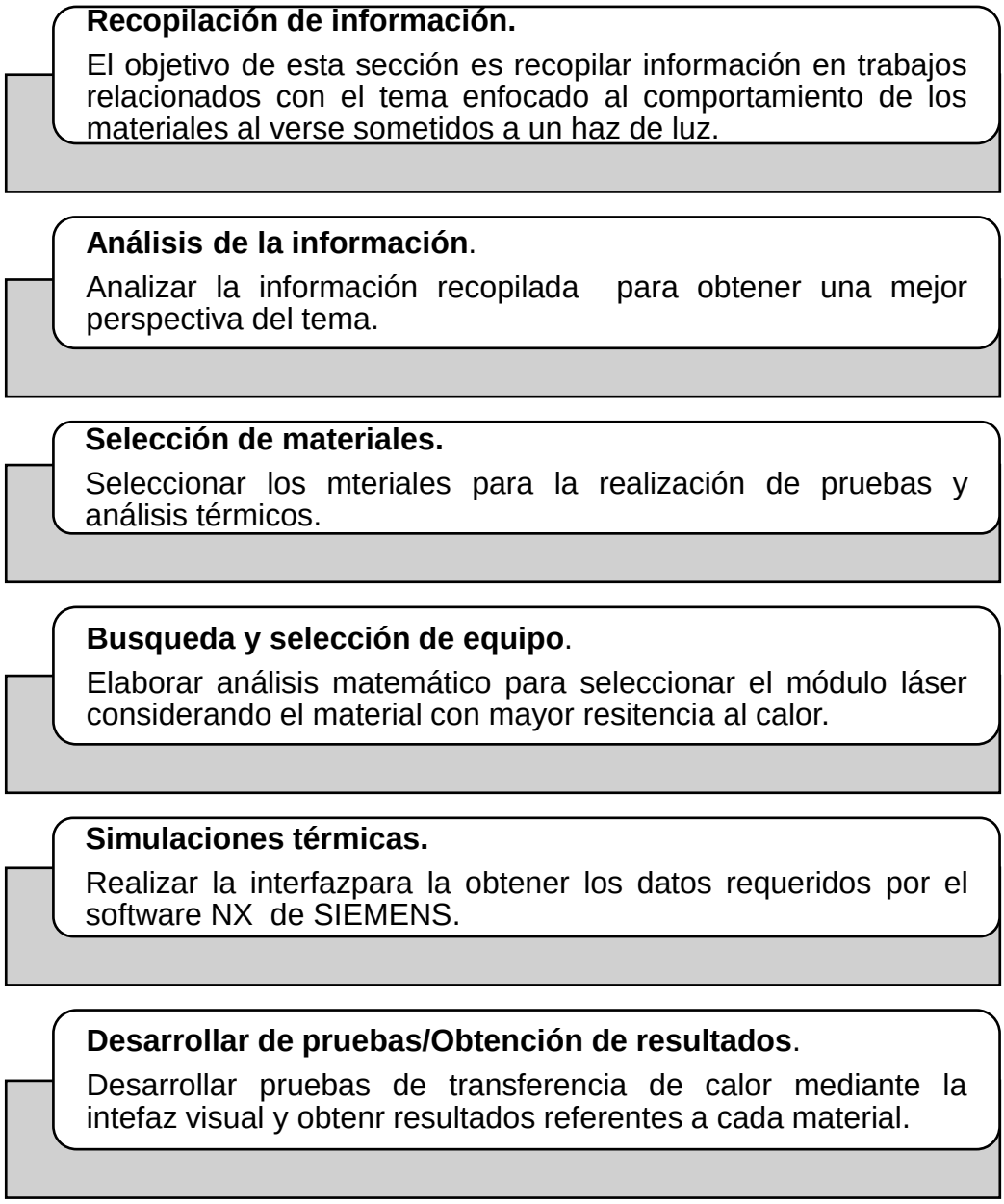


Figura 9 Metodología

Mediante el prototipo para el estudio de acabado de materiales con láser se analizaron tres materiales. La selección de los materiales analizados fue establecida por factibilidad económica y proveedores ubicados en la región. Los materiales analizados se utilizan en aplicaciones cotidianas. Por ejemplo, el grabado o corte de madera, tuberías de PVC y artículos decorativos elaborados con polipropileno.

- Madera blanda
- Policloruro de vinilo PVC
- Polipropileno

3.1.1 Creación de material en biblioteca NX de SIEMENS

Algunos materiales no se encontraron establecidos en la biblioteca NX de SIEMENS. Por lo cual fue necesario la creación del material madera, de ello depende la realización de posteriores análisis térmicos. Para ello se realizaron diseños del prototipo y recopilación de propiedades térmica y mecánicas. Geométricamente se diseñó el prototipo considerando las dimensiones de según Tabla 1Tabla 1 Dimensiones de pieza.. La caracterización de la madera como material establece clasificaciones (blanda y dura). De ambas maderas fue seleccionada la blanda, debido a su versatilidad y bajo costo.

- Blanda: referenciando a que fue obtenida de los árboles pino, cedro.
- Duras: referenciando a que fue obtenida de los árboles roble, haya, nogal, olmo o caoba. Para esta caracterización se optó la clasificación de madera blanda (pino).

Tabla 1 Dimensiones de pieza.

DIMENSIÓN	MEDIDA
Largo	100 mm
Ancho	100 mm
Espesor	5 mm

Los valores de la Tabla 2 se ingresaron en un apartado denominado biblioteca MATMI del sitio Figura 10 Introducción de valores . Fue necesario clasificar la madera como material isotrópico, ya que sus propiedades mecánicas son idénticas en todas las direcciones. Posteriormente se genera un archivo, ubicándose en la carpeta donde se encuentran los archivos. ptr correspondientes a los diseños.

El diseño de la placa fue con las dimensiones establecidas en la Tabla 1. Se decidió homologar las medidas para futuras pruebas físicas, mediante el prototipo de acabados de materiales con láser. El siguiente procedimiento fue realizado para el diseño de la placa.

- Trazar la figura con las dimensiones establecidas.
- Finalizar croquis.
- Extruir la pieza respetando el vector de orientación, el rango de extrusión corresponde al espesor de la placa.

Tabla 2 Propiedades mecánicas de madera blanda.

PROPIEDAD MECÁNICA	**VALOR**
Densidad de la masa	513 kg/m^3
Módulo de elasticidad	10395049000 Pa
Coeficiente de poison	0.25
Limite elástico	13729310 Pa
Tensión a la rotura	40 MPa
Conductividad térmica	0.115 Wm/ K
Calor especifico	1.38 KJ /Kg .K

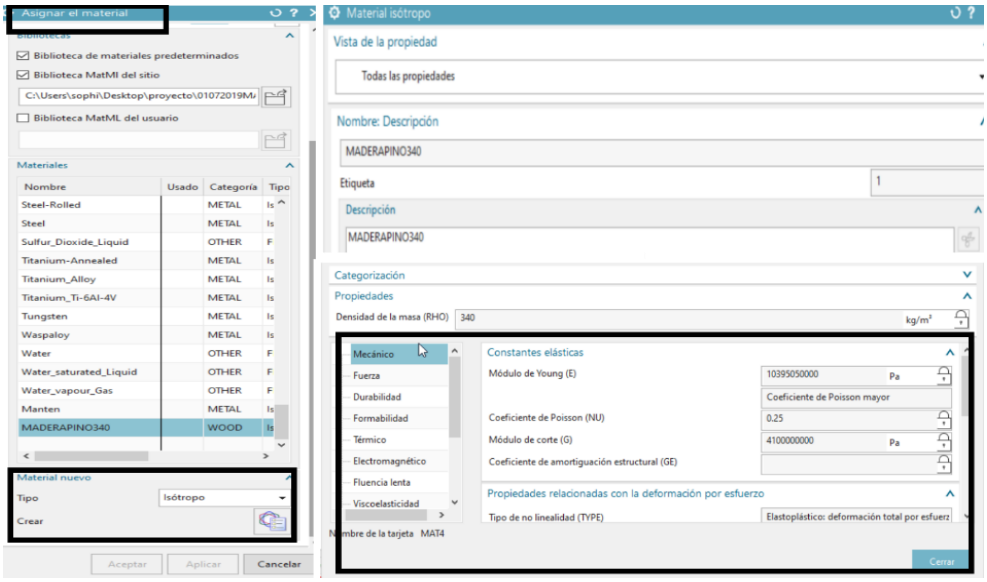


Figura 10 Introducción de valores

3.1.2 Selección de módulo láser.

El módulo láser fue seleccionado considerando pérdidas de convección y radiación. Con la finalidad de comprender mejor el procedmiento se segmentan las perdidas térmicas, al efectuarse la transferencia de calor entre el material analizado y medio ambiente. La selección del módulo láser considero geometría de placa, propiedades mecánicas y térmicas del material. Algunas propiedades térmicas son:

- Flujo de calor .
- Temperatura ambiente.
- Coeficiente de transferencia de calor por convección.

Los siguientes procedimientos muestran matemáticamente la obtención de perdidas por convección y radiación. Es necesario conocer las perdidas térmicas, considerando el flujo de calor como primer valor obtener. El flujo de calor es la transferencia de calor por unidad de tiempo y unidad de área. Para obtener el flujo de calor fue necesario considerar los datos mostrados en la Tabla 3.

Tabla 3 Datos a considerar para la obtención del flujo de calor

DATOS	VALOR
Temperatura ambiente	28°C
Densidad de la madera	513 Kg/m^3
Calor especifico	1.38 KJ /Kg .K
Temperatura de ignición de la madera blanda	70°C
Largo de la placa de madera	100mm
Ancho de la placa de madera	100mm
Espesor de la placa de madera	5mm
Volumen placa de madera	0.00005m^3
Ancho del spot donde se aplica el haz de láser	6mm
Largo del área donde se aplica el spot	10cm
Área placa de madera	0.01m^2

Al transferir energía desde una masa o hacia ésta mediante mecanismos como el calor, fue necesario conocer la cantidad de calor transferido. Siendo (Q) la cantidad de calor transferido durante el proceso, es necesario aplicar la siguiente fórmula.

$$Q = m \cdot Cp(T2 - T1) \quad (1)$$

Dónde:

$m =$ masa del material.

$Cp =$ Calor especifico.

$$(T2 - T1) = Diferencia\ de\ temperaturas \quad (2)$$

La obtención de la masa establecido por la ecuación:

$$m = \rho \cdot V \quad (3)$$

Dónde:

ρ= densidad del material.

v= volumen de placa.

Al realizar las operaciones y sustituir los valores de la tabla (3) se obtiene el siguiente resultado:

$m = (513kg/\ m^3)(0.0000007m^3)$

$m = 1.539x10^{-4}kg$

De la ecuación (1) al sustituir y realizar las operaciones se obtiene:

$Q = (1.539x10^{-4}Kg)(1.38\ KJ\ /Kg\ .K)(70°C - 28°C)$

$Q = 8.920044x10^{-3}KJ$

La cual se interpretó como la energía que necesito transferir al sistema para de 28°C aumentar a 70°C. La velocidad de transferencia de calor también conocida como razón de transferencia de calor involucra la derivada de (Q) por unidad de tiempo (t), por lo cual se denomina $(\dot{Q})$, la unidad es el J/s, equivalente a W, establecida por la ecuación:

$$\dot{Q}prom = \frac{Q}{\Delta t} \quad (4)$$

Por lo tanto, al haber realizado las operaciones y sustituir valores en la ecuación (4) se obtuvo el siguiente resultado:

$$\dot{Q}prom = \frac{8.920044x10^{-3}KJ}{10s}$$

$$\dot{Q}prom = \frac{8.920044x10^{-4}KJ}{s}\ \ o\ \ 8.92W$$

Cuando se cuenta con un área de transferencia de calor (A) y la velocidad de transferencia de calor, se podrá conocer el flujo de calor ($\dot{q}$). Siendo ($\dot{q}$) la cantidad de calor transferido por unidad de área y unidad de tiempo, establecido por la ecuación:

$$\dot{q} = \frac{\dot{Q}}{A} \tag{5}$$

Al realizar las operaciones y sustituir los valores de la ecuación (5) se obtuvo el siguiente resultado:

$$\dot{q} = \frac{8.92W}{(6x10^{-4}m^2)(0.1)}$$

$$\dot{q} = \frac{148.66W}{m^2}$$

El mecanismo de convección natural sobre una superficie depende de la orientación geométrica y variación de temperaturas. Si se desea expresar la transferencia de calor en convección natural, es necesario haber obtenido el número Rayleigh como se muestra en la ecuación (6). El número de Rayleigh es el producto de los números de Grashof y Prandlt, estableciendo la configuración geométrica del área superficial.

$$Ra_L = \frac{g\beta(T_s - T_\infty)L_c^3}{V^2} \quad (Pr) \tag{6}$$

Dónde:

g = aceleración gravitacional de $9.81m/s^2$.

β = coeficiente de expansión volumétrica, $\frac{1}{K}$ ($\beta = \frac{1}{T}$) para gases ideales.

T = temperatura de la superficie en °C.

$T\infty$ = temperatura del fluido suficientemente lejos de la superficie, °C.

Lc = longitud característica de la configuración geométrica (para placas horizontales).

V = viscosidad cinemática del fluido $\frac{m^2}{s}$.

$Pr =$ número de Prandlt.

Considerar la temperatura inicial de 28 ° C y final de 270 ° C para obtener el coeficiente de expansión.

$$\beta = \frac{1}{T_f} \tag{7}$$

$T_f =$ temperatura de la película en grados K.

$$T_f = \frac{(Ts+T\infty)}{2} \tag{8}$$

Al realizar las operaciones y sustituir los valores de la ecuación 8 se obtuvo el siguiente resultado:

$$T_f = \frac{(70°C + 28°C)}{2} = 49$$

$$T_f = 49 + 273 = 322°K$$

Obtener la longitud característica de la configuración geométrica (para placas horizontales)

$$Lc = \frac{L}{4} \tag{9}$$

$$L = longitud$$

$$Lc = \frac{0.01m}{4}$$

$$Lc = \ 0.025m$$

Todas las propiedades del aire fueron evaluadas a temperatura de la película. Aplicando la ecuación 8 se obtuvo un resultado de 49°C, el cual no apareció de manera directa. Por lo cual se estimaron los valores en un intervalo superior e inferior como se muestra en la Figura 11.

Temp., T, °C	Densidad, ρ, kg/m³	Calor específico, c_p, J/kg · K	Conductividad térmica, k, W/m · K	Difusividad térmica, α, m²/s	Viscosidad dinámica, μ, kg/m · s	Viscosidad cinemática, ν, m²/s	Número de Prandtl, Pr
−150	2.866	983	0.01171	4.158×10^{-6}	8.636×10^{-6}	3.013×10^{-6}	0.7246
−100	2.038	966	0.01582	8.036×10^{-6}	1.189×10^{-6}	5.837×10^{-6}	0.7263
−50	1.582	999	0.01979	1.252×10^{-5}	1.474×10^{-5}	9.319×10^{-6}	0.7440
−40	1.514	1 002	0.02057	1.356×10^{-5}	1.527×10^{-5}	1.008×10^{-5}	0.7436
−30	1.451	1 004	0.02134	1.465×10^{-5}	1.579×10^{-5}	1.087×10^{-5}	0.7425
−20	1.394	1 005	0.02211	1.578×10^{-5}	1.630×10^{-5}	1.169×10^{-5}	0.7408
−10	1.341	1 006	0.02288	1.696×10^{-5}	1.680×10^{-5}	1.252×10^{-5}	0.7387
0	1.292	1 006	0.02364	1.818×10^{-5}	1.729×10^{-5}	1.338×10^{-5}	0.7362
5	1.269	1 006	0.02401	1.880×10^{-5}	1.754×10^{-5}	1.382×10^{-5}	0.7350
10	1.246	1 006	0.02439	1.944×10^{-5}	1.778×10^{-5}	1.426×10^{-5}	0.7336
15	1.225	1 007	0.02476	2.009×10^{-5}	1.802×10^{-5}	1.470×10^{-5}	0.7323
20	1.204	1 007	0.02514	2.074×10^{-5}	1.825×10^{-5}	1.516×10^{-5}	0.7309
25	1.184	1 007	0.02551	2.141×10^{-5}	1.849×10^{-5}	1.562×10^{-5}	0.7296
30	1.164	1 007	0.02588	2.208×10^{-5}	1.872×10^{-5}	1.608×10^{-5}	0.7282
35	1.145	1 007	0.02625	2.277×10^{-5}	1.895×10^{-5}	1.655×10^{-5}	0.7268
40	1.127	1 007	0.02662	2.346×10^{-5}	1.918×10^{-5}	1.702×10^{-5}	0.7255
45	1.109	1 007	0.02699	2.416×10^{-5}	1.941×10^{-5}	1.750×10^{-5}	0.7241
50	1.092	1 007	0.02735	2.487×10^{-5}	1.963×10^{-5}	1.798×10^{-5}	0.7228
60	1.059	1 007	0.02808	2.632×10^{-5}	2.008×10^{-5}	1.896×10^{-5}	0.7202
70	1.028	1 007	0.02881	2.780×10^{-5}	2.052×10^{-5}	1.995×10^{-5}	0.7177
80	0.9994	1 008	0.02953	2.931×10^{-5}	2.096×10^{-5}	2.097×10^{-5}	0.7154
90	0.9718	1 008	0.03024	3.086×10^{-5}	2.139×10^{-5}	2.201×10^{-5}	0.7132
100	0.9458	1 009	0.03095	3.243×10^{-5}	2.181×10^{-5}	2.306×10^{-5}	0.7111
120	0.8977	1 011	0.03235	3.565×10^{-5}	2.264×10^{-5}	2.522×10^{-5}	0.7073
140	0.8542	1 013	0.03374	3.898×10^{-5}	2.345×10^{-5}	2.745×10^{-5}	0.7041
160	0.8148	1 016	0.03511	4.241×10^{-5}	2.420×10^{-5}	2.975×10^{-5}	0.7014
180	0.7788	1 019	0.03646	4.593×10^{-5}	2.504×10^{-5}	3.212×10^{-5}	0.6992
200	0.7459	1 023	0.03779	4.954×10^{-5}	2.577×10^{-5}	3.455×10^{-5}	0.6974
250	0.6746	1 033	0.04104	5.890×10^{-5}	2.760×10^{-5}	4.091×10^{-5}	0.6946
300	0.6158	1 044	0.04418	6.871×10^{-5}	2.934×10^{-5}	4.765×10^{-5}	0.6935
350	0.5664	1 056	0.04721	7.892×10^{-5}	3.101×10^{-5}	5.475×10^{-5}	0.6937
400	0.5243	1 069	0.05015	8.951×10^{-5}	3.261×10^{-5}	6.219×10^{-5}	0.6948
450	0.4880	1 081	0.05298	1.004×10^{-4}	3.415×10^{-5}	6.997×10^{-5}	0.6965
500	0.4565	1 093	0.05572	1.117×10^{-4}	3.563×10^{-5}	7.806×10^{-5}	0.6986
600	0.4042	1 115	0.06093	1.352×10^{-4}	3.846×10^{-5}	9.515×10^{-5}	0.7037
700	0.3627	1 135	0.06581	1.598×10^{-4}	4.111×10^{-5}	1.133×10^{-4}	0.7092
800	0.3289	1 153	0.07037	1.855×10^{-4}	4.362×10^{-5}	1.326×10^{-4}	0.7149
900	0.3008	1 169	0.07465	2.122×10^{-4}	4.600×10^{-5}	1.529×10^{-4}	0.7206
1 000	0.2772	1 184	0.07868	2.398×10^{-4}	4.826×10^{-5}	1.741×10^{-4}	0.7260
1 500	0.1990	1 234	0.09599	3.908×10^{-4}	5.817×10^{-5}	2.922×10^{-4}	0.7478
2 000	0.1553	1 264	0.11113	5.664×10^{-4}	6.630×10^{-5}	4.270×10^{-4}	0.7539

Figura 11 Propiedades del aire a presión de una atmosfera

Para obtener la viscosidad cinemática del fluido y el número de Prandlt se consideró los valores en el intervalo, mediante la fórmula de interpolación lineal. La fórmula de interpolación lineal puede ser aplicada cuando se cuenta con un valor a conocer y dos valores conocidos

$$Y = Y_1 + \frac{X - X_1}{X_2 - X_1}(Y_2 - Y_1) \tag{10}$$

Donde:

X_1 =valor superior del intervalo.

$\boldsymbol{X_1}$	45	$\boldsymbol{Y_1}$	1.750X10^{-5}$\boldsymbol{m^2/s}$
$\boldsymbol{X}$	49	$\boldsymbol{Y}$	
$\boldsymbol{X_2}$	50	$\boldsymbol{Y_2}$	1.798X10^{-5}$\boldsymbol{m^2/s}$

$Y = 1.7884X10^{-5}m^2/s$

Obtención de número de Prandlt considerando ecuación 10.

$\boldsymbol{X_1}$	45	$\boldsymbol{Y_1}$	0.7241
$\boldsymbol{X}$	49	$\boldsymbol{Y}$	
$\boldsymbol{X_2}$	50	$\boldsymbol{Y_2}$	0.7228

$$Y = 2.745X10^{-5} + \frac{49-45}{50-45}(0.7228 \ - \ 0.7241\ )$$

$$Y = 0.72306$$

Sustituir los valores obtenidos en la fórmula (6)

$$Ra_L = \frac{(9.81m/s^2)(^1/_{322}\,K)\ (70-28K)(0.025)^3}{(1.7884X10^{-5}\,\frac{m^2}{s})^2} \quad (0.72306)$$

$$Ra_L = 5.8591X10^4$$

Considerando las correlaciones empíricas para el número de Nusselt Figura 12 Correlaciones empíricas número de Nusselt convección natural, seleccionar la ecuación, verificando el intervalo correspondiente a número de Rayleigh. El número de Nusselt es la división del flujo de calor por convección sobre el flujo de calor por conducción, es decir el mejoramiento de transferencia de calor, a la obtención del

mismo. Si el valor obtenido es cercano a uno indica que la transferencia es por conducción.

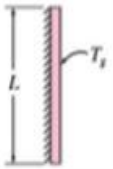
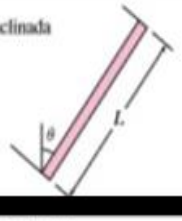
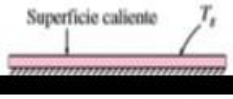
Configuración geométrica	Longitud característica L_c	Intervalo de Ra	Nu
Placa vertical 	L	$10^4 - 10^9$ $10^9 - 10^{13}$ Todo el intervalo	$Nu = 0.59Ra_L^{1/4}$ (9-19) $Nu = 0.1Ra_L^{1/3}$ (9-20) $Nu = \left\{0.825 + \frac{0.387Ra_L^{1/6}}{[1 + (0.492/Pr)^{9/16}]^{8/27}}\right\}^2$ (9-21) (compleja pero más exacta)
Placa inclinada 	L		Utilícense las ecuaciones de la placa vertical para la superficie superior de una placa fría y la superficie inferior de una placa caliente Reemplácese g por $g\cos\theta$ para $Ra < 10^9$
Plástico horizontal (Área superficial A y perímetro p) *a*) Superficie superior de una placa caliente (o superficie inferior de una placa fría) 	A_s/p	$10^4 - 10^7$ $10^7 - 10^{11}$	$Nu = 0.54Ra_L^{1/4}$ (9-22) $Nu = 0.15Ra_L^{1/3}$ (9-23)
b) Superficie inferior de una placa caliente (o superficie superior de una placa fría) 		$10^5 - 10^{11}$	$Nu = 0.27Ra_L^{1/4}$ (9-24)

Figura 12 Correlaciones empíricas número de Nusselt convección natural

Obtención del número de Nusselt.

$$Nu = 0.54\mathrm{Ra}_{\mathrm{L}^{1/4}} \tag{11}$$

Donde:

Ra_L =Numero de Rayleigh.

$Nu = 0.54(5.8591X10^4)^{1/4}$

$Nu = 8.401428$

Fue necesario obtener el coeficiente de transferencia de calor por convección. Siendo la transferencia de convección proporcional a la diferencia de temperatura. Lo antes mencionado se puede expresar por la ley del enfriamiento de Newton.

$$h = \frac{K}{L_c} Nu \tag{12}$$

Donde:

k =conductividad térmica.

Nu =número de Nusselt.

Lc = longitud característica de la configuración geométrica (para placas horizontales).

Siendo la capacidad de un material para conducir calor, la conductividad térmica debe evaluarse con las propiedades del aire a temperatura de la película. Al igual que viscosidad cinemática y número de Prandlt, se obtuvieron los valores mediante interpolación aplicados en ecuación10.

$\boldsymbol{X_1}$	45	$\boldsymbol{Y_1}$	0.026996 $\boldsymbol{W/m}$.°$\boldsymbol{C}$
$\boldsymbol{X}$	49	$\boldsymbol{Y}$	
$\boldsymbol{X_2}$	50	$\boldsymbol{Y_2}$	0.02735 $\boldsymbol{W/m}$.°$\boldsymbol{C}$

$$Y = 0.02699 + \frac{49 - 45}{50 - 45}(0.02735 - 0.026996)$$

$$Y = 0.027278$$

$$h = \frac{0.027278 W/m \ .°C}{0.025m} (8.401428)$$

$h = 9.1669\, {}^{W}/_{m}\ {}^{\circ}C$

$$\dot{Q} = h * As * (T_s\ - T_\infty\ ) \quad (13)$$

$\dot{Q}$ =índice de transferencia de calor.

h = coeficiente de transferencia de calor.

$A_{s=}$ área superficial de la placa.

$\dot{Q} = \left(9.1669\, {}^{W}/_{m}\ {}^{\circ}C \right)(6x10^{-4}m^2)(70°C - 28°C)$

$\dot{Q} = 0.23100588W$

$q = \frac{0.23100588W}{6x10^{-4}m^2}$

$q = 385\ {}^{W}/_{m^2}$

Para compensar pérdidas ocasionadas por el medio ambiente, fue considerada la razón de transferencia de calor promedio. La radiación es la energía emitida por la materia en forma de ondas electromagnéticas. A diferencia de la conducción y la convección, la transferencia de calor por radiación no requiere la presencia de un medio interventor, la radiación térmica, es la forma de radiación emitida por los cuerpos debido a su temperatura. La radiación depende de la emisividad térmica, mostrando algunos valores de emisividad en la Tabla 4.

Para obtener la transferencia de calor por radiación se consideraron los siguientes datos.

$$\dot{Q}radiación = \varepsilon * \sigma * As * Ts^4 \quad (14)$$

Donde:

σ = constante de Stefan-Boltzmann 5.67x10^8w/$m^2 \cdot K^4$.

ε = emisividadde la superficie.

T_s = temperatura termodinámica en grados kelvin.

Tabla 4 Emisividad de algunos materiales a 300 K

Material	Emisividad
Hoja de aluminio	0.07
Alumino anonizado	0.82
Cobre pulido	0.03
Plata pulida	0.02
Acero inoxidable pulido	0.17
Pintura negra	0.98
Pintura blanca	0.9
Papel blanco	0.92-0.97
Pavimento de asfalto	0.85-0.93
Ladrillo rojjo	0.93-0.96
Piel ahumada	0.95
Madera	0.82-0.92
Suelo	0.96
Agua	0.96

$$\dot{Q}radiación = (0.92)(5.67x10^{8}W/m^{2} \cdot K^{4})\ (6x10^{-4}m^{2})(343^{4} - 301^{4})$$

$$\dot{Q}radiación = 7.3223x10^{-4}W$$

$$\dot{Q}radiación = \frac{0.1762959W}{6x10^{-4}m^{2}}$$

$$\dot{Q}radiación = 293.82W/m^{2}.$$

La selección del módulo láser se realizó a partir de las perdidas por radiación y convección, dando un resultado 827.35 W/m^2, comercialmente el valor más cercano es 1 W/m^2. Sin embargo, como se desea realizar pruebas implicando variaciones de potencia, se optó por adquirir un módulo láser con potencia de 5.5 W. Considerando tiempo máximo de operación treinta minutos y vida útil aproximada de 80 horas siendo, datos según proveedor.

149.33 W/m^2 Flujo de calor

\+ 385.00 W/m^2 transferencia de calor por convección natural

293.82 W/m^2 transferencia de calor por radiación.

Total, de: 827.35 W/m^2

3.1.3 Diseño de interfaz visual

Guide es un entorno disponible en MATLAB, utilizado para realizar programación visual, el cual necesita un ingreso continuo de datos. Con la finalidad de obtener diversos parámetros se optó por la creación de una interfaz visual. Siendo capaz proporcionar parámetros numéricos obtenidos mediante operaciones matemáticas y utilizados en simulaciones térmicas.

Los parámetros son los siguientes:

- Flujo de calor
- Temperatura ambiente
- Coeficiente de transferencia de calor por convección.

Es necesario mencionar que al modificar alguna dimensión de la placa, los valores cambiarian por completo. Por tal motivo el procedimeirnto fue estandarizado, permitiendo introducir valores del material a analizar.

Una vez ingresando a MATLAB en la pestaña de home buscar la sección llamada New, mostrara la opción App. Aparecen dos opciones desplegadas de la cual se

seleccionó Guide. Posteriormente fue localizada la opción Blank GUI, la Figura 13 muestra los controladores que utilizaremos con mayor frecuencia.

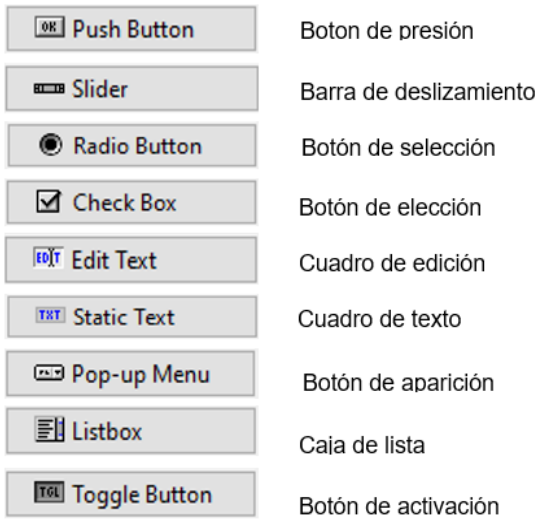


Figura 13 Algunos controladores de GUIDE

Las propiedades de los componentes se describen a continuación:

- Push button: Es capaz de generar una acción debido a que con solo presionar se ejecuta la rutina para la cual fue programado.
- Slider: Aceptan datos numéricos los cuales son de entrada al momento de mover la barra indica un valor numérico.
- Checkbox: Las casillas de verificación realizan operaciones encendido/apagado.
- Edith text: Permite editar texto mediante la introducción de una cadena de entrada.
- Static text: Muestra textos al igual que valores numéricos.

En cuanto a activar los objetos se deben configurar mediante el controlador callback. El controlador callback quien direcciona a la programación del código generada automáticamente Figura 14.

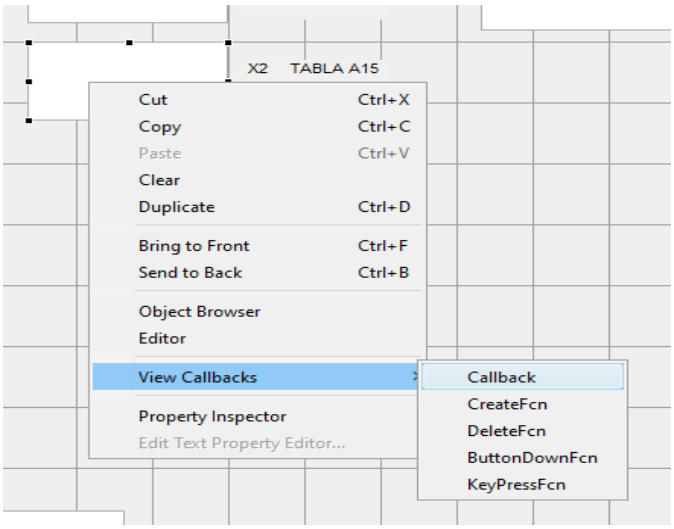


Figura 14 Ubicación del controlador callback

Mediante la selección de controladores se formuló cada una de las operaciones referentes a perdidas (flujo de calor y coeficiente de transferencia de calor por convección). Los primeros datos a ingresar fueron aquellos relacionados con la superficie del material. Posteriormente propiedades mecánicas como densidad y calor especifico Figura 15**¡Error! No se encuentra el origen de la referencia.**. Obtener los valores de viscosidad cinemática del fluido, número de Prandlt y se efectúa mediante interpolación.

- Seleccionar el componente.
- Activar los objetos, configurando mediante el controlador callback, quien a su vez direcciona a la programación del código generada automáticamente.
- Realizar la programación pertinente mediante las fórmulas utilizadas en el apartado selección del módulo láser.
- Mediante el componente axes programar la imagen de fondo correspondiente a una simulación térmica

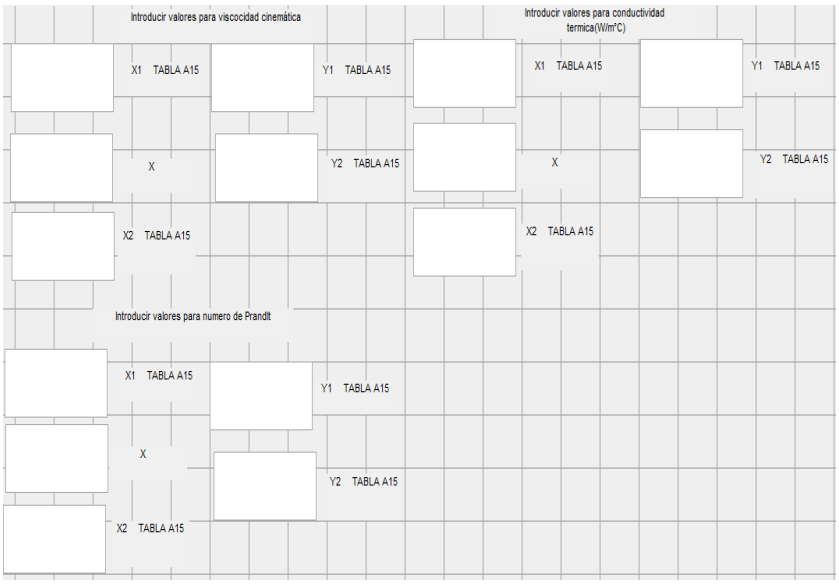


Figura 15 Introducción de datos para elaborar interpolación

Si bien se mencionó con anterioridad, la creación de la interfaz visual fue con la finalidad de ahorrar tiempo y procedimientos, los cuales se aplicaban en cada material. El resultado de la interfaz visual fue exitoso, de otra manera si el procedimiento se realizara manualmente por cada material se tardaría un aproximado de 60 minutos. Con la implementación de la interfaz visual solamente basta con presionar un botón y aparecerán los resultados, la interfaz visual se muestra en la Figura 16 .

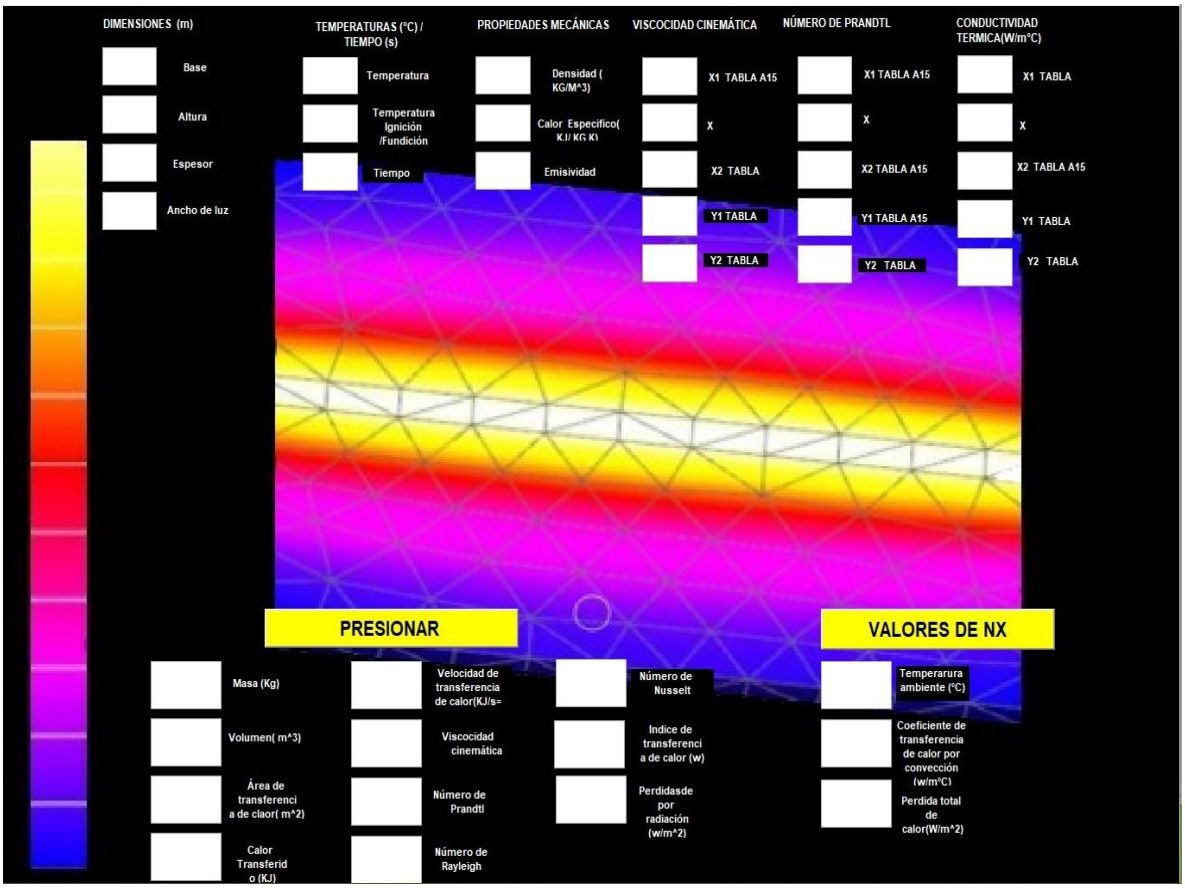


Figura 16 Interfaz visual

3.1.4 Elaboración de análisis térmico

Algunos diseños en ingeniería cuentan con condiciones geométricas o comportamiento poco usuales. Debido a ello se deben de establecer diferentes métodos para la resolución de simulaciones térmicas. Siendo uno de los métodos más utilizados el elemento finito. Aplicado en análisis térmicos, estructurales y de viento. El siguiente procedimiento fue el realizado para el análisis térmico mediante elemento finito.

Seleccionar el plano de orientación para elaborar el diseño.

- Trazar la figura con las dimensiones establecidas.
- Finalizar croquis.
- Extruir la pieza respetando el vector de orientación, rango de extrusión corresponde al espesor de la placa.

- Crear el archivo FEM y SIM Figura 17.
- En la pestaña de archivo seleccionar preprocesamiento / posprocesamiento
- Seleccionar FEM y SIM nuevos

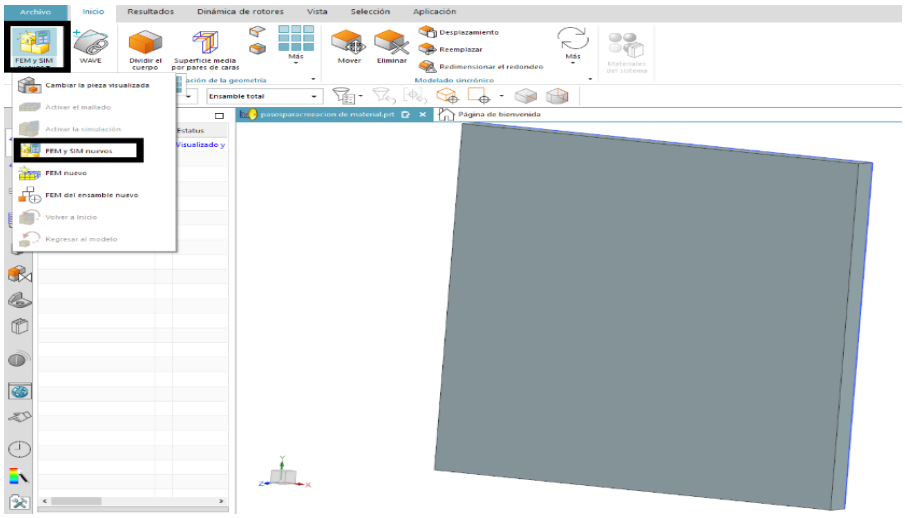


Figura 17 Creación de archivo FEM y SIM

- Seleccionar la casilla crear una pieza idealizada y resolución de cuerpo poligonal.
- Dar clic en aceptar parte del procedimiento anterior se muestra en la Figura 18.

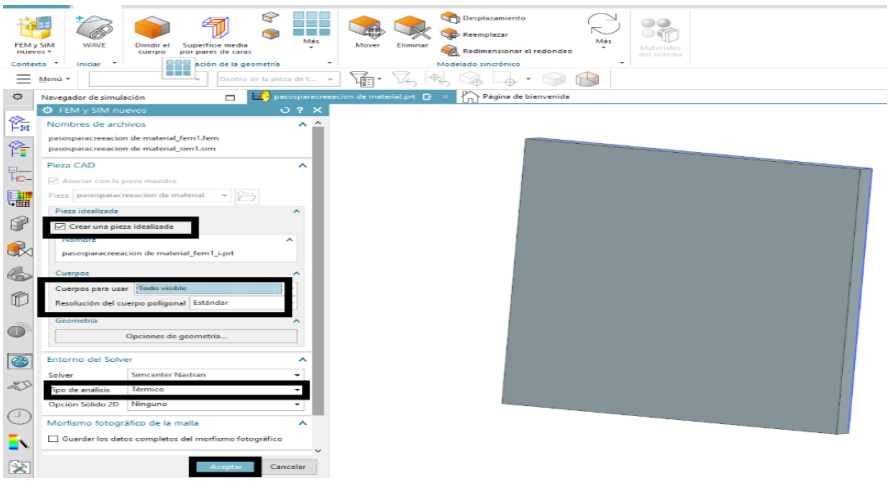


Figura 18 Selección de análisis térmico

La geometría WAVE permite segmentar un cuerpo en varios fragmentos y seleccionar el ideal, para posteriormente analizarlo. Utilizando una distribución de cargas térmicas en una longitud de 6mm, se implementaron las cargas térmicas obtenidas con anterioridad. La idealización de placa permitió indicar el vector de dirección. A continuación, se enlista detalladamente el procedimiento, mostrado en la Figura 19.

- Ascender el cuerpo.
- Dividir la cara a segmentar, indicando el desplazamiento.
- Seleccionar la opción geometría de enlaces modo WAVE.
- Seleccionar la cara a segmentar.
- Seleccionar la curva.

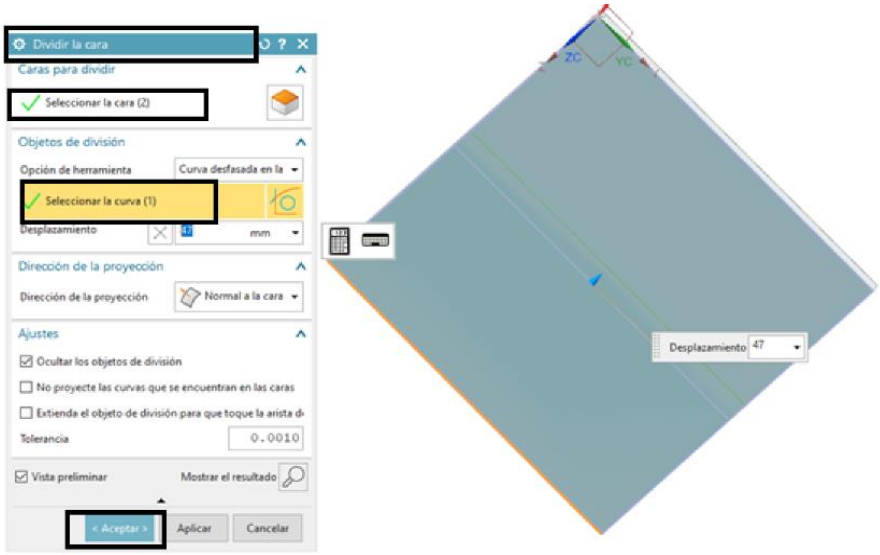


Figura 19 Selección de geometría WAVE

El entorno FEM y SIM contiene propiedades del material y de cargas térmicas respectivamente mostrado en la Figura 20. FEM contiene materiales, mallado, segmentaciones y geometría poligonal. SIM contiene propiedades del estudio como cargas y restricciones. La obtención de algunos parámetros antes mencionados se obtuvo habiendo realizado la interfaz visual. A continuación, se enlista detalladamente los procedimientos en los entornos FEM y SIM.

- Asignar el material.
- Seleccionar el cuerpo al cual se asignará el material.
- Asignar la malla y activar el parámetro de malla tamaño del elemento.
- Seleccionar tipo de carga correspondiente a temperatura.
- Seleccionar tipo de restricción.
- Proporcionar temperatura ambiente y coeficiente de convección.
- Proporcionar el flujo de calor.

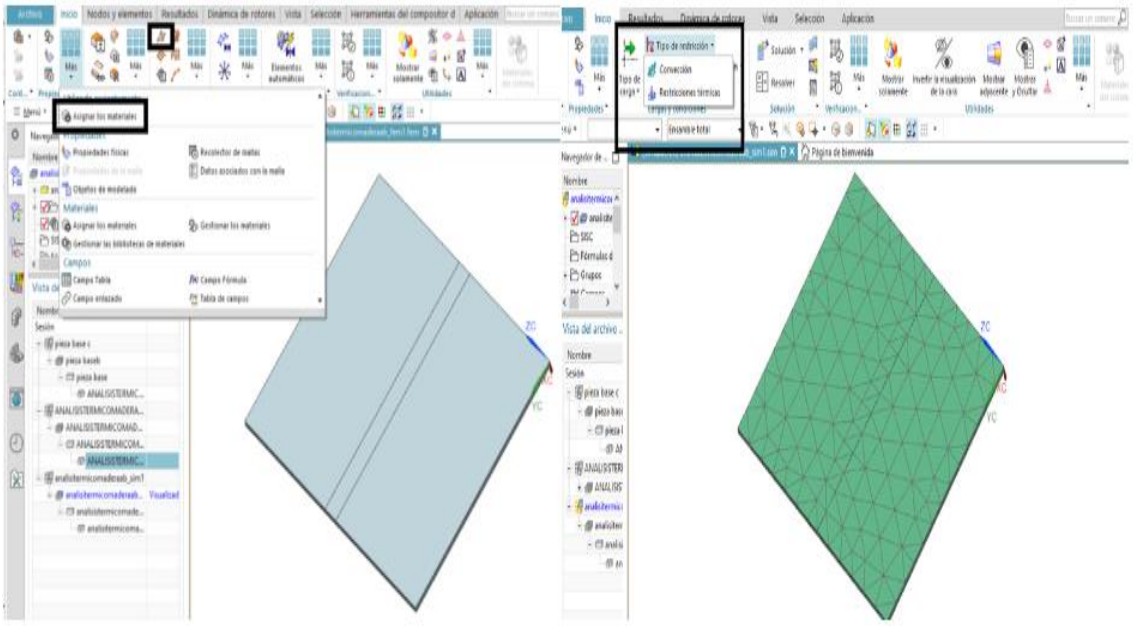


Figura 20 Entornos FEM y SIM

3.1.5 Diseño de aplicación control de potencia y brazo robótico con MITT APP INVENTOR

Mit App Inventor es un software de uso libre, siendo conveniente por no tener que adquirir licencias. Al ser un software de uso libre cuenta con tutoriales de fácil acceso, los cuales implementan programación mediante bloques.

Como primera opción se optó por la creación de una aplicación en Mit App Inventor, permitiendo el control de potencia del módulo láser. Gracias a la versatilidad del

software la aplicación se puede implementar en dispositivos móviles, utilizando un módulo bluetooth HC-05.

Es necesario agregar componentes mediante la pestaña, primeramente, en diseñador, posteriormente en la de bloques. A cada componente se asigna una instrucción. Cada pestaña cuenta con elementos propios descritos a continuación.

- Interfaz de usuario: Contiene elementos los cuales se agregan en la parte de visor.
- Visor: Muestra los elementos seleccionados y agregados de la interfaz de usuario.
- Componentes: Muestra el orden de aplicación de los elementos utilizados.
- Propiedades: Se visualiza y edita las características de cada elemento.
- Pestaña de bloques contiene los componentes integrados en el apartado diseñador, en cada componente se encuentran instrucciones en forma de bloque correspondientes a sus funciones.
- Visor: Contiene los bloques seleccionados los cuales pueden sufrir cambios de acuerdo a la necesidad del programador.

En la parte superior se selecciona el apartado de proyectos, posteriormente a comenzar un nuevo proyecto. En el apartado de disposiciones seleccionar horizontal. La sección de bloques utiliza el deslizador con la posición del pulgar indicada en la sección de propiedades obteniendo un dato. El dato obtenido mediante la pantalla del dispositivo móvil se almaceno en una lista denominada L, la variable L se encuentra declarada en un código de Arduino.

Adicionalmente se cuenta con tres botones diseñados para encender, apagar el módulo láser y desconectar la interfaz en cualquier momento de la trayectoria. En la parte de interfaz de usuario seleccionamos tres botones los cuales separamos con una disposición horizontal posteriormente editamos propiedades Figura 21.

Figura 21 Programación de interfaz en APP INVENTOR

Con la finalidad de comprobar el correcto funcionamiento de la aplicación se implementó un circuito, sustituyendo el módulo láser con un led. La Figura 22 muestra el circuito elaborado para realizar las pruebas iniciales, el led se conectó a PWM corresponde al pin once, mientras que el módulo bluetooth HC-05 Tx y Rx corresponden al pin uno y cero.

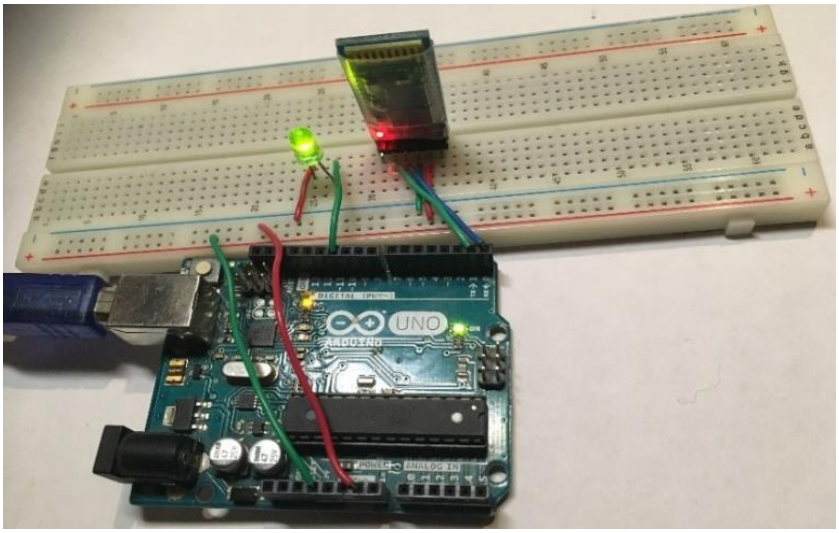


Figura 22 Prueba piloto control de potencia

Como se planteó desde un inicio el brazo robótico proporciono movimiento al módulo láser, la programación del brazo se realizó mediante Arduino uno. Los servomotores se programaron con trayectorias definidas de acuerdo a la altura deseada, partiendo de un home establecido, el siguiente fragmento de programación muestra las posiciones iniciales.

```
#include <Servo.h>

//Declaración de servomotores//
Servo servomotorbase;
Servo servomotorbrazo;
Servo servomotorhombro;
Servo   servomotorlaser;

//Declaración de posiciones iniciales//
int posbase = 90 ;
int posbrazo=180 ;
int poshombro= 150;
int poslaser =0;
int pos= 0;
```

Al realizar las primeras pruebas se observó que la aplicación para control de potencia con MIT APP INVENTOR no es adecuado, ya que la comunicación en cuanto al cambio de potencia no es fluida. Encontrando un área de oportunidad se decidio migrar a MATLAB.

3.1.6 Control movimiento brazo robótico

Con anterioridad se realizó una aplicación para controlar el movimiento del brazo robótico, la cual no conto con una comunicación estable. Por lo cual se decidió realizar un interfaz visual, capaz de controlar los servomotores que proporcionaron movimiento al brazo robótico. El control del brazo robótico se realizó mediante dos modelos de servomotores MG 995 y MG905 ambos compatibles con Arduino uno.

Los servomotores se controlaron mediante Arduino. Los pines tres, cinco y seis son los seleccionados para el control de los servomotores. Las siguientes indicaciones corresponden al segmento control de servomotores en brazo robótico:

- Seleccionar la opción Blank GUI (default) y aparecerá el cuadro de diálogo.

- Una vez que aparezca el área de trabajo insertar por cada puerto seleccionado los siguientes elementos un Push Button, un Pop-up Menu, un Edit text, los tres elementos anteriores se agruparan en un panel Figura 23.

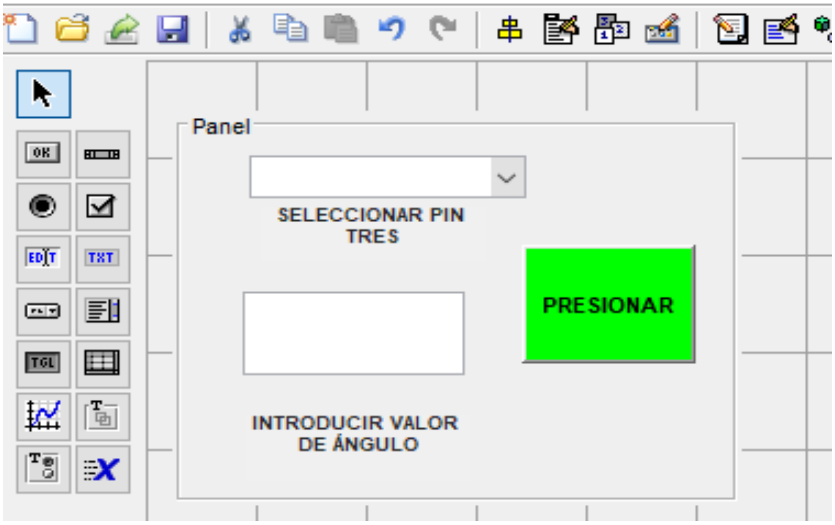


Figura 23 Elementos necesarios en control de servomotor

- Activar los objetos se deben configurar mediante el controlador callback, quien a su vez direccione a la programación del código generada automáticamente.
- Establecer la comunicación con Arduino mediante la siguiente línea de código handles.a =Arduino.

La comunicación para los elementos del archivo m. y fig. con Arduino se establece a continuación:

- Establecer el Pop-up Menu, debe de coincidir con el seleccionado en el archivo.
- Indicar la lista de pines a seleccionar el Pop-up Menu y posteriormente indicar cual va a ser el seleccionado.
- Agregar en la función Push Button el código para establecer el movimiento de servomotor, el código se modifica según el servomotor que se desea accionar.

```
% --- POSICIÓN LÁSER
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
escribirvalortres=str2num(get(handles.edit18,'String'));
for b =150:-0.5:escribirvalortres
%Ejecuta el movimiento inicando en cuarenta, con avancxe de 0.5y detente en
%el valor escrito en edit 4.
writePosition(handles.s3,((b)/180));
 %Escribe la posición en el servo 3  y divide b/180 para la conversión a
 %grados.
pos=readPosition(handles.s3)*180;
%Leer la posición  del servo s3 y multiplicarla 180
  pause(0.1);
 %Realice una pausa de 0.01
  disp(b);
end
```

Los servomotores tienen movimiento de cero a ciento ochenta grados, sin embargo, es necesario plantear valores para posicionar el brazo robótico en una misma trayectoria. La finalidad de ello es repetir el procedimiento con cada uno de los materiales seleccionados, para ello se realizaron pruebas de movimiento obteniendo la mejor posición en cada parte del brazo robótico. La trayectoria efectuada por el módulo láser corresponde a 10 cm. La Tabla 5 muestra los grados en los que se posiciono cada servomotor.

Tabla 5. Grados seleccionados en movimiento de brazo robótico.

ELEMENTO DE BRAZO RÓBOTICO	INICIO DE TRAYECTORIA	FIANAL DE TRAYECTORIA
HOMBRO	110°	120°
BRAZO	150°	130°
APOYO MÓDULO LÁSER	70°	135°

Al seleccionar los valores realizar los cambios e indicar visualmente en la interfaz el valor a introducir Figura 24.

Figura 24 Valores a introducir en la interfaz

3.1.7 Control de potencia en módulo láser

Al igual que los servomotores el módulo láser es controlado mediante PWM por sus siglas en inglés, Pulse Width Modulation este tipo de método se utiliza para transmitir señales de información en forma de pulsos utilizando esta para reducir la energía total entregada a una carga. (Chávez, 2015).

El pin seleccionado fue el número once de Arduino uno, el valor deseado se introduce en Edit text y al presionar un Push Button proporciona la potencia seleccionada. La potencia máxima del láser es de 5.5 W, la Tabla 4 proporciona porcentajes y valor correspondientes al módulo láser. Según indicaciones del fabricante el módulo láser no debe exceder dos horas encendido de manera continua, por lo cual es recomendable dejar de utilizar mínimo treinta minutos por cada hora y media.

Tabla 6 Valor de potencia y porcentaje correspondiente.

VALOR EN W	PORCENTAJE
5.5	100%
3.85	70%
2.75	50%
1.65	30%
0.55	10%

Las siguientes indicaciones corresponden al segmento control de potencia de módulo láser:

- Seleccionar la opción Blank GUI (default) y aparecerá el cuadro de diálogo mostrado en Figura 18.
- Una vez que aparezca el área de trabajo insertar un Push Button y un Edit text, los dos elementos anteriores se agruparan en un panel.
- Activar los objetos se deben configurar mediante el controlador callback, quien a su vez direccione a la programación del código generada automáticamente.

En la función relacionada con el Push Button se asignó la variable valor, siendo igual al valor número introducido en Edit text 19, el cual se divide entre cien como se indica en el siguiente código.

```
% CONTROL POTENCIA LÁSER
function pushbutton7_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton7 (see GCBO)
% eventdata  reserved - to be defined in a future versión of MATLAB
% handles    structure with handles and user data (see GUIDATA)
valor=str2num(get(handles.edit19,'String'));
%Obtener el valor del edit2 y convertirlo en valor número
valor = (valor/100);
%El valor  obtenido de edit2 dividirlo entre100
 writePWMDutyCycle(handles.a,'D11',valor);
 %Escribe en la salida PWM el valor
 guidata(hObject,handles);
```

Con la finalidad de comprobar el correcto funcionamiento se realizaron pruebas piloto. Correspondiente al control de potencia se implementó un led simulando el módulo láser**¡Error! No se encuentra el origen de la referencia.** Figura 25. Posteriormente se implementó el módulo láser, es necesario recalcar que cuando se utilizó el módulo láser se conto con medidas de seguridad, correspondientes a gafas de protección para láser y luz pulsante.

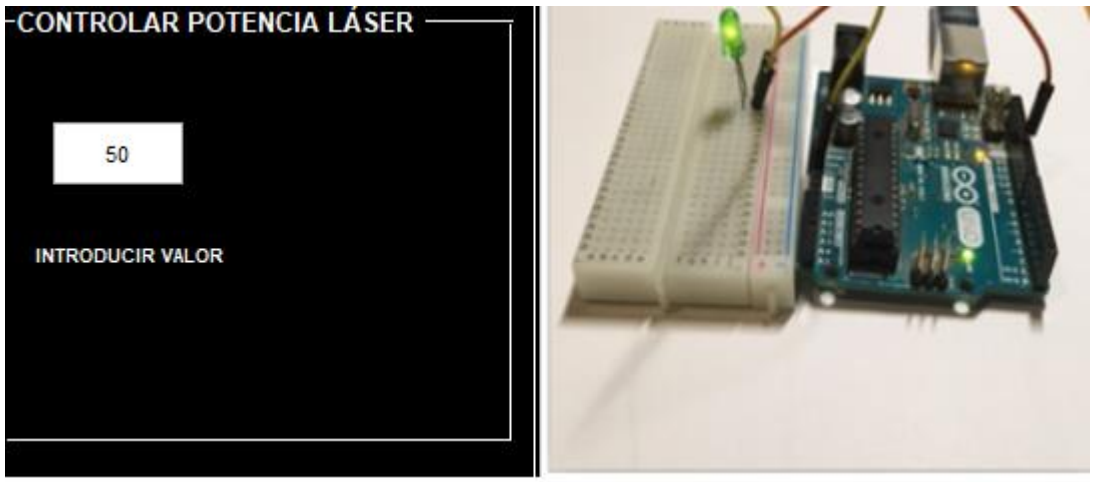


Figura 25 Prueba piloto control de potencia

3.1.8 Adquisición de temperaturas

La adquisición de temperaturas fue implementada de dos maneras, mediante sensores LM35 o cámara termográfica. Para la simulación en la interfaz visual fue

necesario adquirir cuatro temperaturas, por lo cual se implementaron cuatro sensores. Cada sensor se conecto con un Edit text, donde se visualizo la temperatura obtenida, adquiriendo los parámetros al presionar un Pusn Button Figura 26. Los sensores se encuentran conectados en los pines analógicos de Arduino uno, siendo colocados en los laterales de la placa y cerca del haz de luz láser.

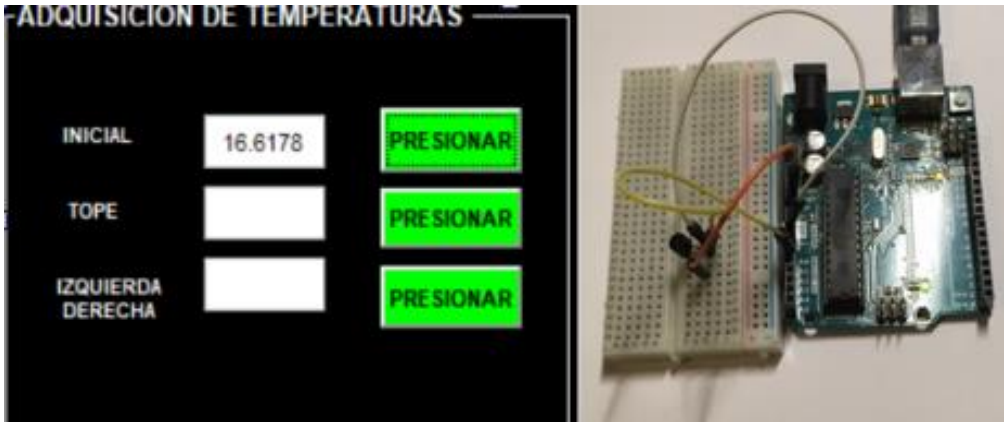


Figura 26 Prueba piloto adquisición de temperaturas

La otra manera de adquirir temperaturas fue mediante cámara termográfica, siendo una manera más precisa. Consiste en enfocar la cámara termográfica en la zona específica y obtener el termograma en tiempo real. Posteriormente se introducen las temperaturas en el apartado temperaturas en °C ubicado en la interfaz visual.

3.1.9 Simulación de temperaturas

La conducción de calor en régimen transitorio cuenta con dos métodos de análisis implícito y explícito, en cualquiera de ellos la figura se segmenta en partes llamadas nodos entre más nodos existan más exacto es el valor obtenido según la variación del tiempo, la diferencia radica en los intervalos de tiempo mientras al método implícito se asigna $i+1$, el explícito solamente consideramos i siendo la literal mencionada un contador para los intervalos de tiempo en régimen transitorio.

La conducción de calor se encuentra caracterizada mediante el número de Fourier (Fo), implicando la velocidad de conducción y el almacenamiento de energía el cual

debe cumplir con el criterio de estabilidad, de lo contrario no es válido este tipo de análisis $Fo \leq 0.5$. (A Cenguel,2007).

El procedimiento para simular transferencias de calor suele ser repetitivo, sin embargo, se debe mencionar la variación de temperaturas obtenidas y difusividad que implica propiedades físicas del material, entre ellas se encuentran densidad, calor específico y conductividad térmica. Basándose en el código de uso libre "Conducción de calor transitorio" (Ye Cheng,2022), se implementaron modificaciones de acuerdo a las necesidades actuales. Como se mencionó con anterioridad algo fundamental para la transferencia de calor es la difusividad térmica, las difusividades térmicas del material se encuentran en Figura 27, obtenida de la siguiente manera:

$$\alpha = \frac{k}{\rho * Cp} \tag{15}$$

Dónde:

k =conductividad térmica.

ρ= densidad del material.

Cp = Calor específico.

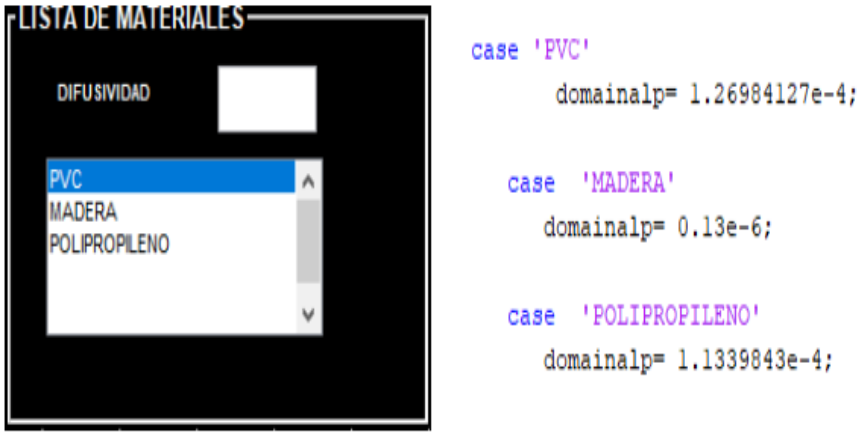


Figura 27 Materiales y conductividad térmica implementada en código

En transferencia de calor mediante régimen transitorio se considero el tiempo proporcionado por el recorrido total del módulo laser sobre la geometría asignada a

la placa, siendo 6 segundos. Los apartados referentes a temperaturas y geometría se muestran en la Figura 28

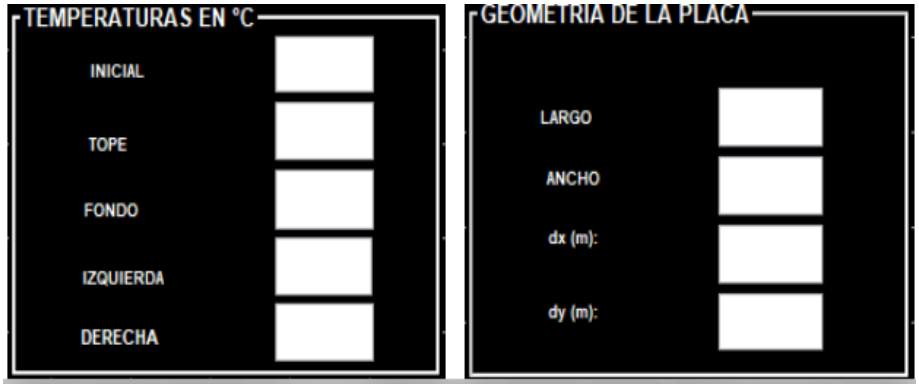


Figura 28 Temperaturas y geometría solicitadas en análisis térmicos

CAPÍTULO 4: RESULTADOS

Los resultados obtenidos implican análisis térmicos mediante el método de elemento finito, siendo un proceso que implica variación de potencia en materiales a analizar. La herramienta principal utilizada fue un módulo láser con potencia de 5.5 W.

Mediante la interfaz visual se obtuvieron los valores solicitados para las simulaciones térmicas Figura 29 Resultados obtenidos para simulación térmica a madera, se debe mencionar que los resultados de la interfaz son diferentes para cada material. Por ejemplo, la madera blanda cuenta con una temperatura de ignición de 70°C.

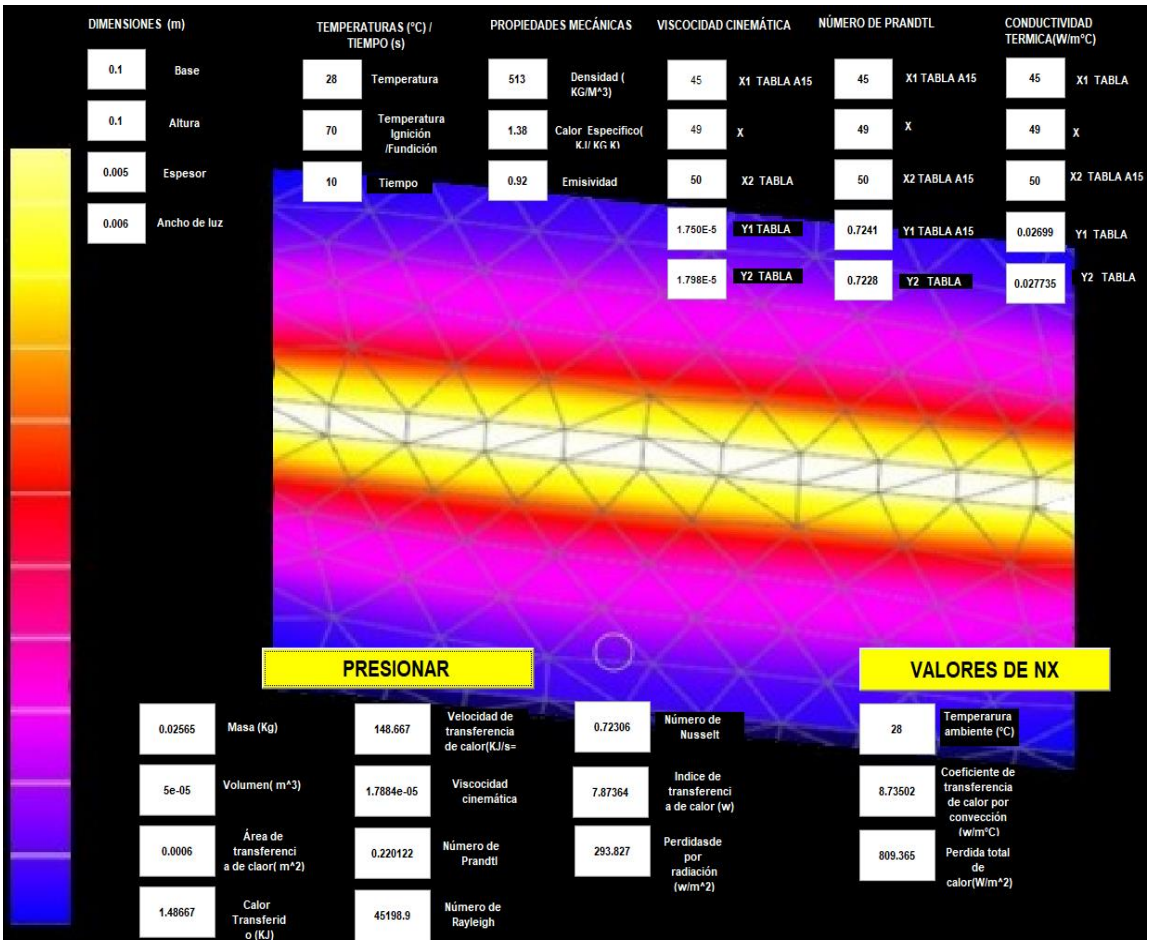


Figura 29 Resultados obtenidos para simulación térmica a madera

Los resultados obtenidos mediante análisis térmico, vistas frontal y posterior correspondiente a madera blanda, con temperaturas en el rango 28°C y 53°C. En

los análisis térmicos mediante NX de SIEMENS. La Figura 30 Análisis térmico madera vista frontal y posterior, no superando a la temperatura de ignición, ya que los análisis térmicos se realizaron con una potencia correspondiente a la de 1W.

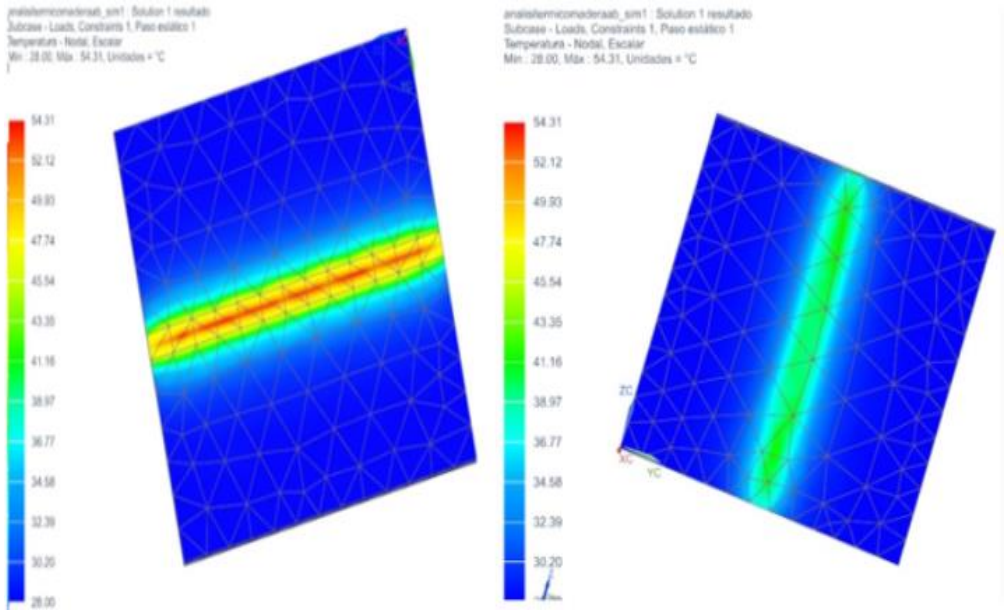


Figura 30 Análisis térmico madera vista frontal y posterior

La transferencia de calor mediante régimen transitorio, corresponde a la simulación del calor con diferentes potencias y alturas. La Tabla 7 indica el tiempo en segundos y la posición de los servomotores en grados con su correspondiente en centímetros, según la posición del brazo robótico. La sección de temperaturas en grados centígrados corresponde a los valores obtenidos en los termogramas.

Tabla 7: Resultado en madera 5mm de espesor

MADERA 5 mm DE ESPESOR							
POTENCIA	DISTANCIA	TEMPERATURAS EN GRADOS CENTIGRADOS					
POTENCIA	GRADOS/CM	INICIAL	TOPE	FONDO	D/I	TIEMPO	OBSERVACIONES
10%-0.55	130°-6	29	22	32	27	5	No presenta daño visible en el material, al avanzar el módulo láser existe rápidamente un equilibrio térmico.
	135°- 4	33	22	37	28	5	No presenta daño visible en el material, al avanzar el módulo láser existe rápidamente un equilibrio térmico.

	140- 2	35	22	39	30	5	No presenta daño visible en el material, al avanzar el módulo láser existe rápidamente un equilibrio térmico.
50%-2.75	130°-6	30	22	35	28	5	Presenta perforaciones. Tarda 15 segundos en llegar al equilibrio térmico.
	135°- 4	30	22	39	31	5	Tarda 25 segundos en llegar al equilibrio térmico.
	140- 2	33	22	47	33	5	Presenta grabado superficial el material presenta perforaciones. Tarda 49 segundos en llegar al equilibrio térmico.
90%-4.95	130°-6	36	22	53	34	5	Presento perforaciones. Tarda 30 segundos en llegar al equilibrio térmico.
	135°- 4	43	22	65	39	5	Presenta perforaciones. Tarda 40 segundos en llegar al equilibrio térmico.
	140- 2	47	22	69	50	5	Presenta perforaciones. Tarda 52 segundos en llegar al equilibrio térmico.

Con la interfaz visual, se realizaron las pruebas físicas correspondientes a la madera con 5mm de espesor. La potencia corresponde a 4.95W, distancia de 2 cm, los resultados obtenidos se muestran en la Figura 31.

Los resultados obtenidos mediante la simulación térmica en estado transitorio se observan en la parte central a la interfaz, mientras el funcionamiento del módulo láser de lado superior derecho. El funcionamiento del módulo láser corresponde al control de potencia, ubicado en la programación de la interfaz visual.

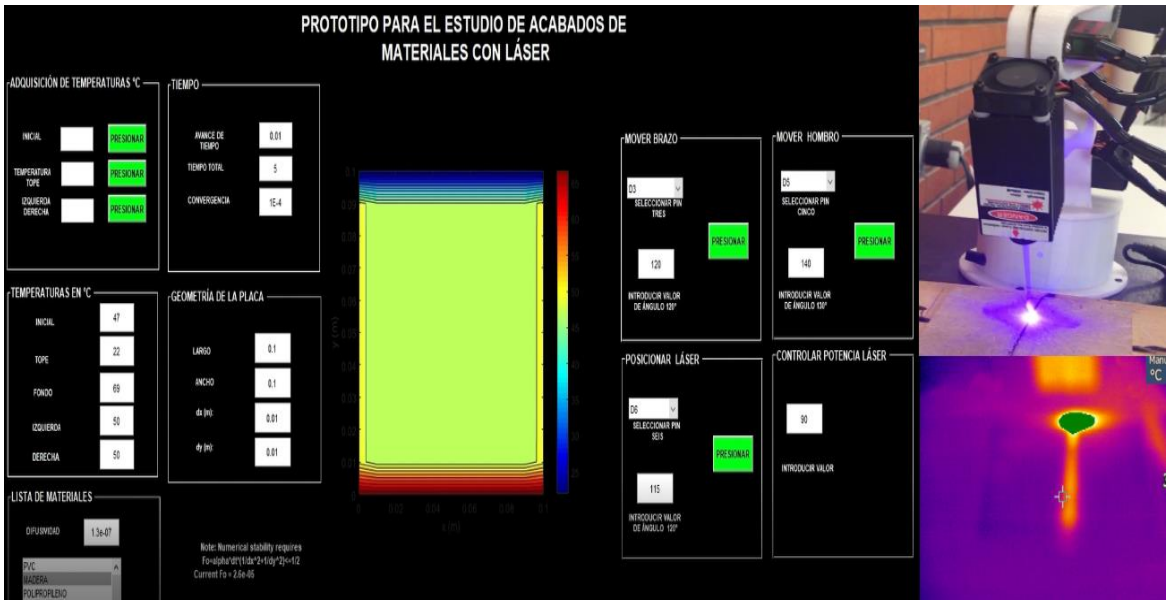


Figura 31 Resultados obtenidos madera 5mm de espesor

El segundo material a analizar fue policloruro de vinilo (PVC), mediante la interfaz visual se obtuvieron los valores solicitados en simulaciones térmicas, como referencia la temperatura de ignición del policloruro de vinilo es de 415°C. La **¡Error! No se encuentra el origen de la referencia.** muestra los resultados obtenidos para el análisis térmico en NX de SIEMENS.

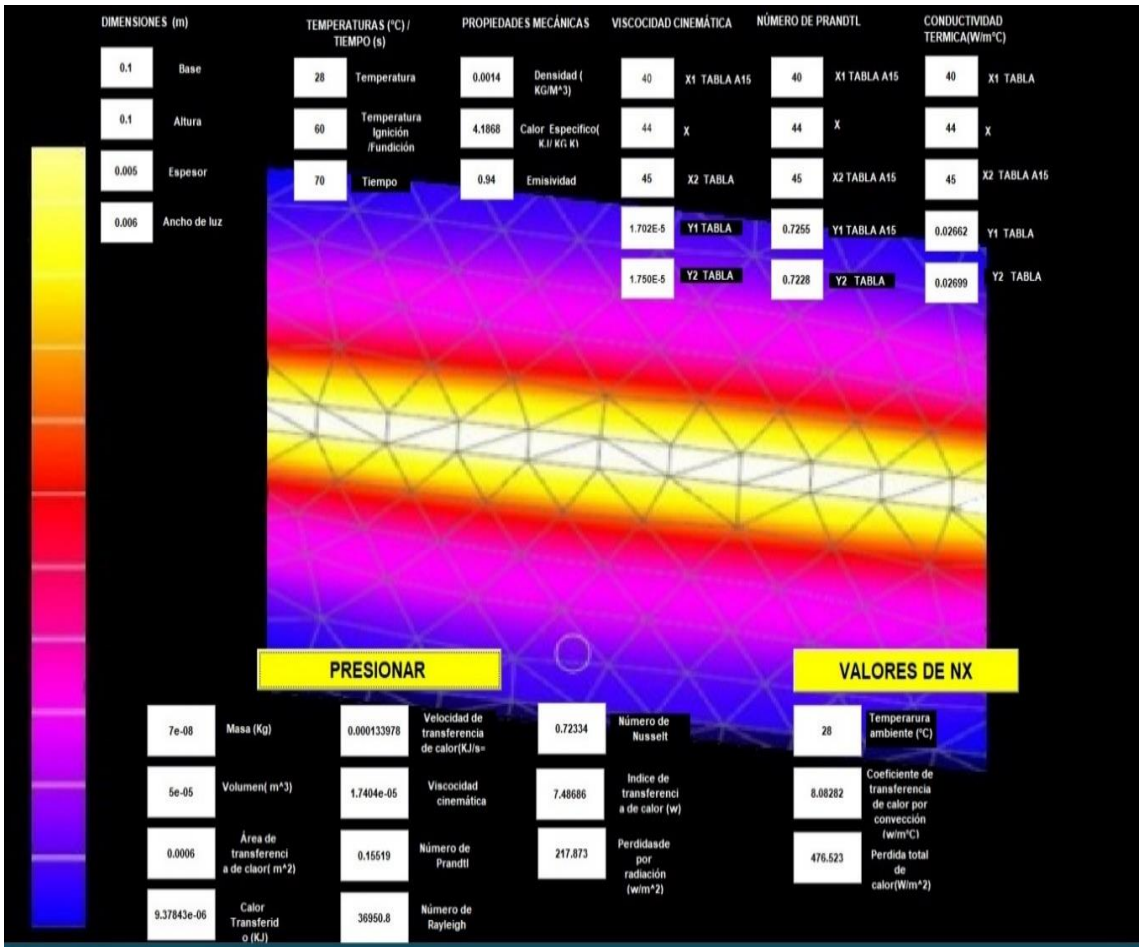


Figura 32 Resultados obtenidos para simulación térmica policloruro de vinilo.

Los resultados obtenidos mediante análisis térmico Figura 33 vista frontal y posterior con temperaturas en el rango 31.97°C y 28.69°C, corresponde a policloruro de vinilo (PVC). Por lo cual, los valores obtenidos en las simulaciones no superan la temperatura de ignición en 415°C, ya que los análisis térmicos se realizaron con una potencia correspondiente a la de 1W.

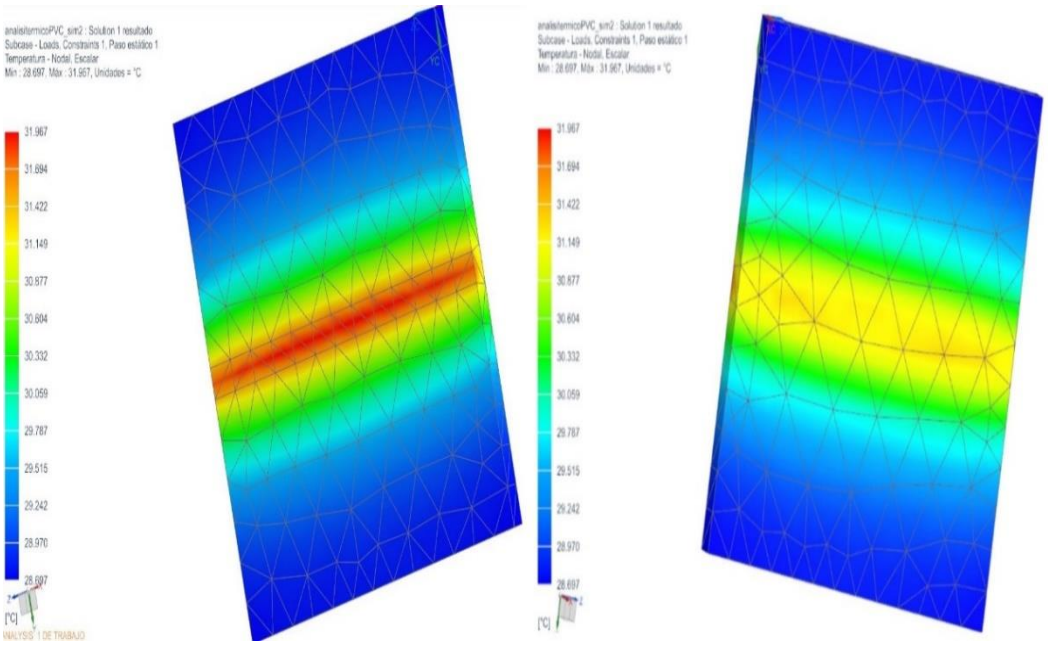


Figura 33 Análisis térmico policloruro de vinilo vista frontal y posterior

La transferencia de calor mediante régimen transitorio, corresponde a la simulación del calor con diferentes potencias y alturas. La Tabla 8 indica el tiempo en segundos y la posición de los servomotores con su correspondiente, según la posición del brazo robótico centímetros. Mie sección de temperaturas en grados centígrados corresponden a los termogramas obtenidos.

Tabla 8:Resultado en policloruro de vinilo 5mm de espesor

POLICLORURO DE VINILO (PVC) 5 mm DE ESPESOR							
POTENCIA	DISTANCIA	TEMPERATURAS EN GRADOS CENTIGRADOS					
POTENCIA	GRADOS/CM	INICIAL	TOPE	FONDO	D/I	TIEMPO	OBSERVACIONES
10%-0.55	130°-6	29	22	30	29	16	No presenta daño visible en el material.
	135°- 4	32	22	33.6	30	16	No presenta daño visible en el material.
	140- 2	33	22	32	32	16	No presenta daño visible en el material.
50%-2.75	130°-6	36	22	48	32	16	El material presenta daños visibles y tarda 10 segundos en alcanzar el equilibrio térmico.

	135°- 4	40	22	54	46	16	El material presenta daños visibles y tarda 16. segundos en alcanzar el equilibrio térmico.
	140- 2	50	22	58	53	16	Aunque la distancia entre el material y el láser es más cercana no presenta labores daños que a 130°C.
90%-4.95	130°-6	63	22	75	47.4	16	El material presenta daños visibles y tarda 9. segundos en alcanzar el equilibrio térmico.
	135°- 4	74	22	87	52	16	El material presenta daños visibles y tarda 33. segundos en alcanzar el equilibrio térmico.
	140- 2	96	22	100	72	16	El material presenta daños visibles y tarda 58. segundos en alcanzar el equilibrio térmico.

Mediante la Figura 34 se muestra el resultado de la interfaz para la simulación de policloruro de vinilo (PVC) de 5mm de espesor, al momento de variar las potencias se observó que el material sufre mayor daño físico cuando esta más alejado del haz de luz debido a el color blanco, suele ser un material reflejante.

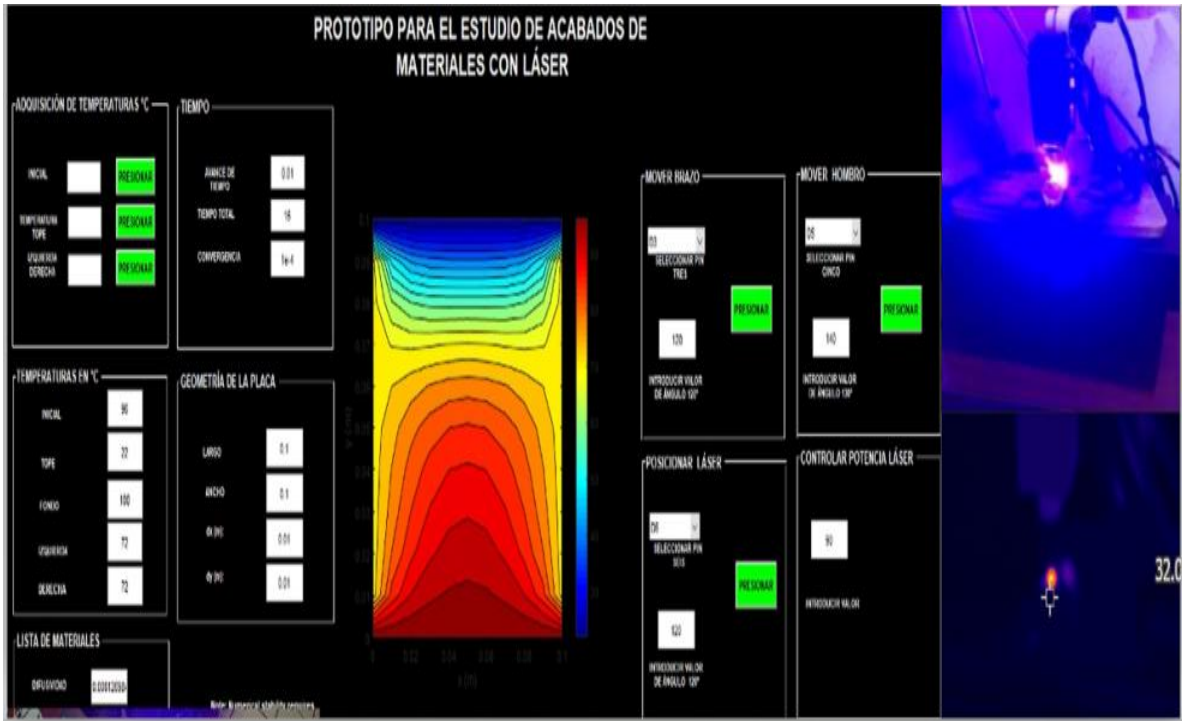


Figura 34 Resultados obtenidos polipropileno 5mm de espesor

Repitiendo las simulaciones en la interfaz visual, el tercer material a analizar es polipropileno. Asi mismo cuenta con una temperatura de ignición de 96°C, se estableció que los valores obtenidos en la interfaz denominados valores de NX, son aquellos que se introducen al realizar las simulaciones térmicas en el software NX de SIEMENS.

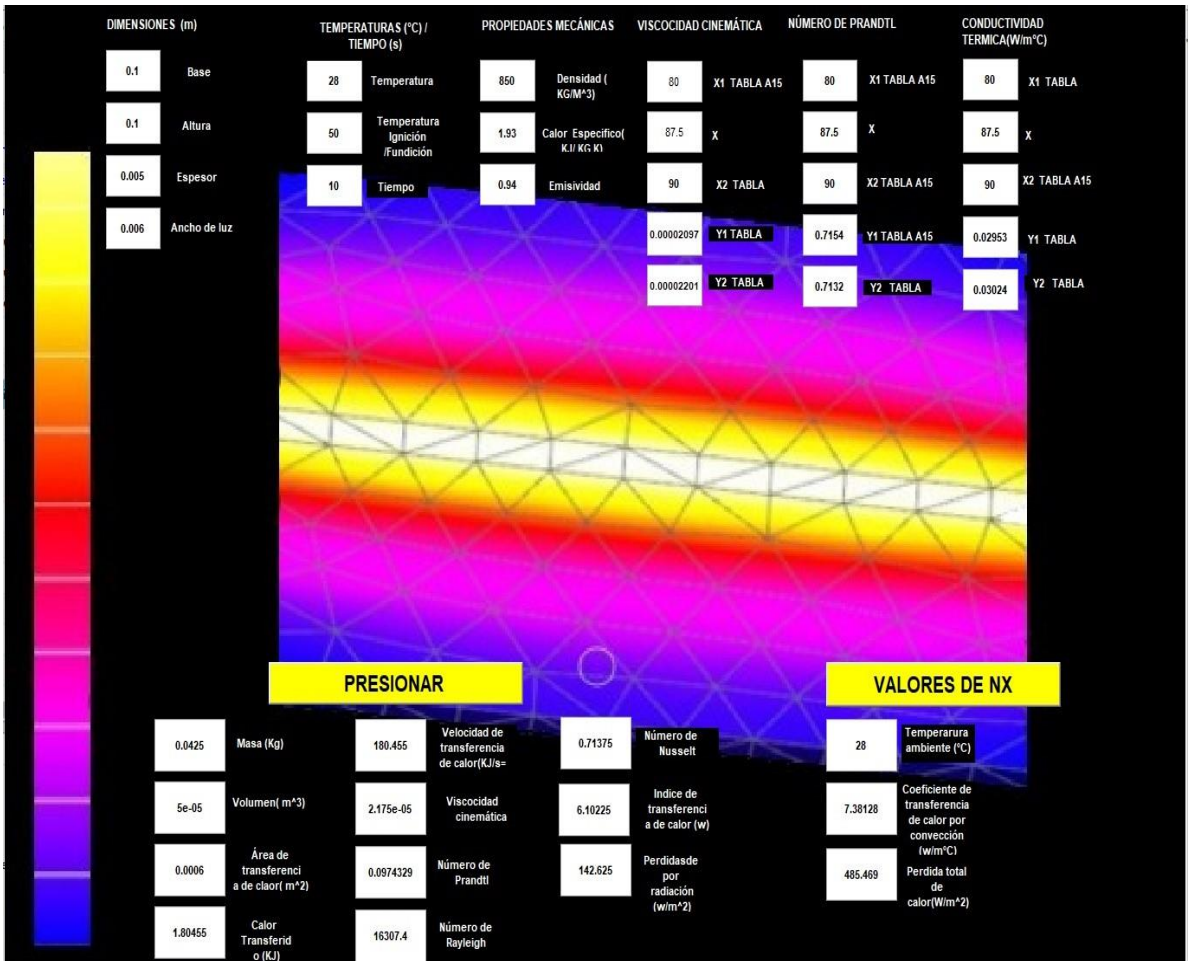


Figura: 2 Resultados obtenidos para simulación térmica policloruro de vinilo.

La Figura 35**¡Error! No se encuentra el origen de la referencia.** muestra los resultados obtenidos mediante análisis térmico. Mediante la vistas frontal y posterior correspondiente a el material polipropileno, se obtuvo la temperatura máxima de 46°C y mínima de 28°C. La línea central es la variación de temperatura, correspondiente al máximo valor el segmento de color rojo, siendo el color azul la

temperatura ambiente. Ya que existieron variaciones de temperaturas la perforación del material se realiza en la parte central.

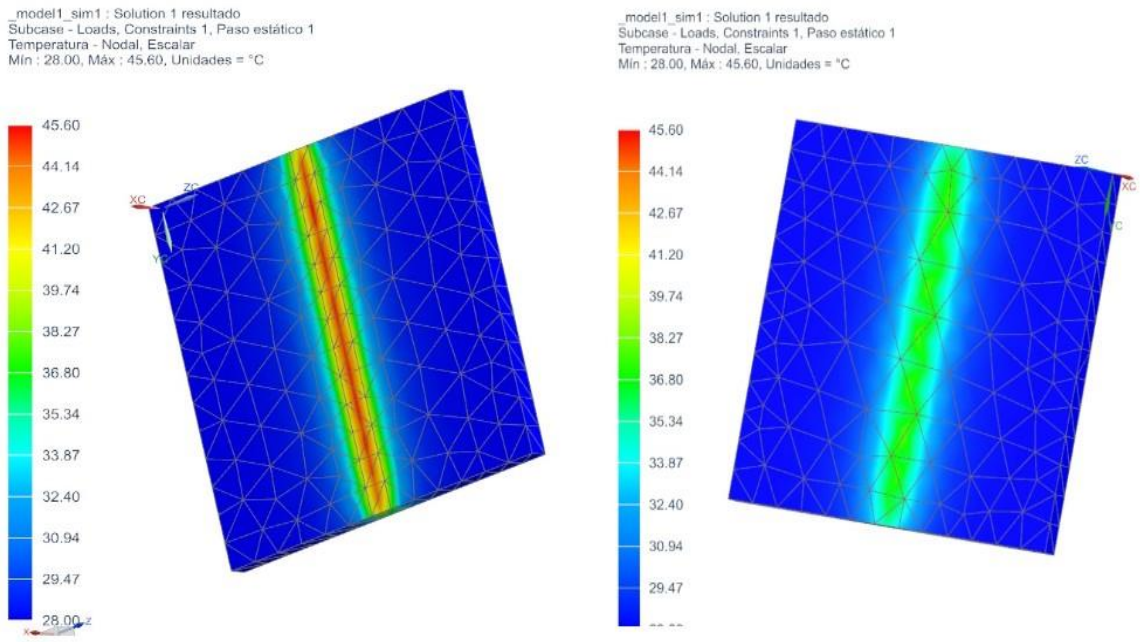


Figura 35 Análisis térmico polipropileno de vinilo vista frontal y posterior

.

La simulación de transferencia de calor mediante régimen transitorio para el polipropileno aplicó diferentes potencias y alturas. Se debe mencionar que en la Tabla 9, el tiempo es indicado en segundos y la posición de los servomotores en grados, con su correspondiente en centímetros. Por lo cual las temperaturas en grados centígrados corresponden a los termogramas obtenidos con anterioridad.

Tabla 9: Resultado polipropileno 5mm de espesor.

POLIPROPILENO 5 mm DE ESPESOR							
POTENCIA		TEMPERATURAS EN GRADOS CENTIGRADOS					
POTENCIA	DISTANCIA	INICIAL	TOPE	FONDO	D/I	TIEMPO	OBSERVACIONES
10%-0.55	130°-6	30	22	35	28	5	No presenta daño visible en el material.

	135°- 4	34	22	40	32	5	Presenta perforación de 2mm aproximadamente
	140- 2	38	22	42	33	5	Conforme avanza tarda 10 segundos en estar en equilibrio térmico, cuenta con perforaciones.
25%-1.35	130°-6	30	22	32	29	5	Conforme avanza tarda 40 segundos en estar en equilibrio térmico, cuenta con perforaciones.
	135°- 4	33	22	34	32	5	Conforme avanza tarda 47 segundos en estar en equilibrio térmico, cuenta con perforaciones.
	140- 2	35	22	35.5	32.3	5	Conforme avanza tarda 32 segundos en estar en equilibrio térmico, cuenta con perforaciones.
50%-2.75	130°-6	33	22	41	30	5	Conforme avanza tarda 55 segundos en estar en equilibrio térmico, cuenta con perforaciones de 2.5mm.
	135°- 4	40	22	44	33	5	Conforme avanza tarda 68 segundos en estar en equilibrio térmico, el material es perforado por completo.
	140°-2						El material es quemado por completo

Mediante la Figura 36 se muestra el resultado de la interfaz para la simulación de polipropileno con 5mm de espesor. Cuando se variaron las potencias se observó que el material es quemado por completo al aplicar una potencia 2.75W y distancia de 2 cm. Por lo cual el equilibrio térmico es alcanzado 68 segundos después de que el módulo láser realiza la trayectoria establecida.

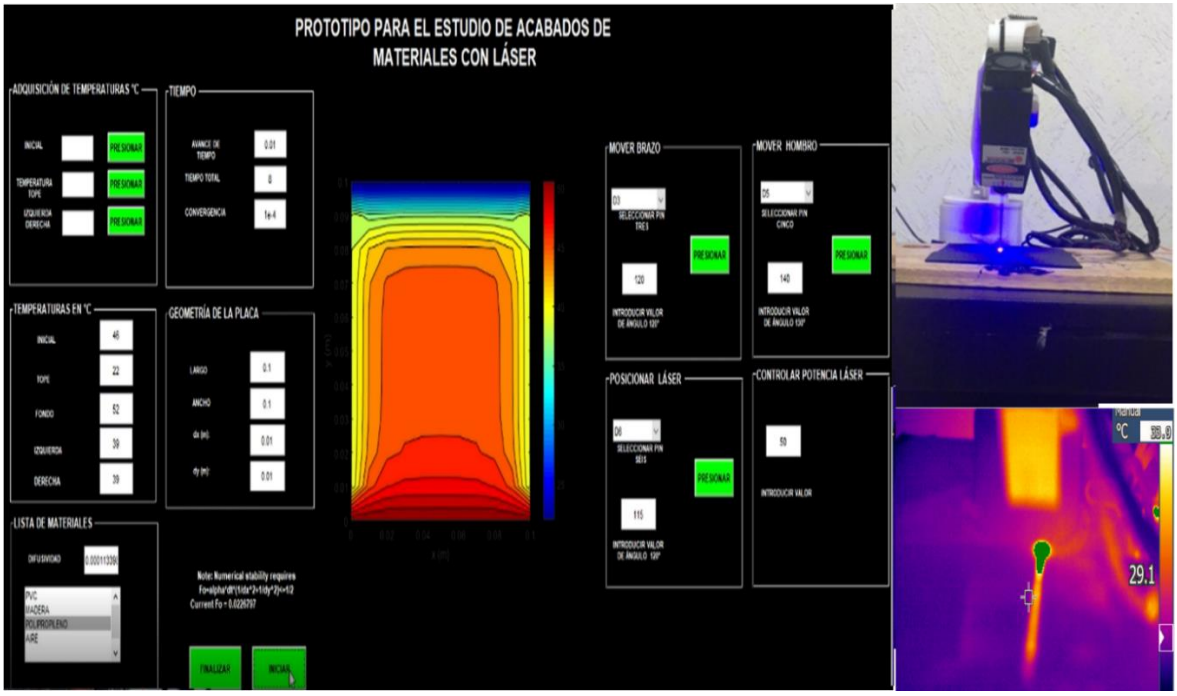


Figura 36 Resultados obtenidos polipropileno 5mm de espesor

CONCLUSIONES Y RECOMENDACIONES

CONCLUSIONES

Los acabados de materiales cada día son más comunes industria metal mecánica, maderera y textil. Es necesario el mejorar las herramientas para incrementar el número de producción sin desperdiciar material y eficientar costos. Para poder obtener las mejoras antes mencionadas, es la aplicación del procedimiento para analizar diferentes materiales.

En el presente proyecto se analizó de manera matemática y mediante simulaciones, como actúa el material aplicando una fuente de calor. Por lo cual en un inicio el proyecto se encontraba enfocado solamente en el sector industrial. Sin embargo, es un campo de posibilidades para el sector educativo especialmente en aquellas materias relacionadas con termodinámica y transferencia de calor.

La interfaz gráfica proporciona parámetros numéricos obtenidos mediante operaciones matemáticas utilizados en simulaciones térmicas. El proceso para la obtención puede ser replicado siempre y cuando coincida la misma forma de transferencia de calor y realizando las adecuaciones según la geometría de la pieza a analizar.

Se encontraron áreas de oportunidad como lo fue la adquisición de temperaturas mediante el sensor LM35, debido a su tamaño compacto y dimensiones de la placa. El sensor LM35 puede tener variaciones de posicionamiento referentes al área a sensar, ya que cada material tendría que ser adecuado para montar el sensor. Por lo cual se decidió adquirir las temperaturas mediante la cámara termográfica y el análisis de termogramas.

Otra área de oportunidad fue la creación de materiales no existentes dentro de la librería del software NX de SIEMENS. En cuanto al movimiento del brazo robótico se realizó en primera estancia una aplicación en MIT APP INVENTOR al momento de comunicar el módulo láser con el celular tardaba algunos segundos en realizar la trayectoria y en algunas ocasiones no la realizaba optando por realizar una interfaz en MATLAB.

Los análisis térmicos fueron muy cercanos a los obtenidos en temperaturas mediante la cámara termográfica. Al mencionar son muy cercanos debemos de tener en cuenta que el módulo láser no se utilizó con su potencia máxima por el sobrecalentamiento.

En conclusión, este proyecto puede implementarse en el sector educativo e industrial. En el sector educativo permite fortalecer las áreas de programación, transferencia de calor, termodinámica y diseño. Mientras que en el sector industrial puede reducir costos y eficiente los parámetros en cuanto al grabado de materiales, siendo un método aplicable al análisis de otros materiales por transferencia de calor en régimen transitorio.

RECOMENDACIONES

Las recomendaciones para investigaciones de análisis de materiales con láser son las siguientes:

Como primer punto se debe de tener en cuenta que al calcular matemáticamente el valor obtenido se debe de aproximar al de características comerciales más cercanas. El valor seleccionado referente a la potencia del láser se verá afectado de acuerdo al mecanismo de transferencia de calor analizado. Recordando que existen tres métodos de transferencia de calor, los cuales se pueden combinar entre ellos. Como segundo punto dependiendo el tipo de láser a utilizar son los materiales con los cuales se puede trabajar. Como tercer punto a considerar la vida útil del láser varía de acuerdo al modelo.

Al realizar la trayectoria del láser mediante el material, se consideró una potencia máxima de 4.95 W. De aplicar la potencia máxima según especificaciones del fabricante, el módulo láser tiende a no disipar el calor y apagarse repentinamente. Por lo cual es conveniente implementar un sistema de enfriamiento más eficiente.

BIBLIOGRAFÍA

A.Cenguel, Y. (2007). *Transferencia de calor y masa:Un enfoque práctico.* Mc Graw Hill.

Alfonso, J. (8 de Abril de 1993). Aplicaciones del láser de co2 de pequeña y mediana potencia. *bdigital*. doi:10.15446/mo

Alvarez, J. P. (s.f.). Peliculas delgadas de TiO2 Modificado con CO para aplicación en sistemas. *Repositorio institucional de universidad autonoma del estado de mexico*. Recuperado el 10 de Octubre de 2018, de http://ri.uaemex.mx/handle/20.500.11799/14994

Casado Fernandez, M. (28 de Enero de 2022). Obtenido de http://webs.ucm.es/centros/cont/descargas/documento11541.pdf

Chavez Talavera Ricardo Emanuel, C. U. (Julio de 2015). Obtenido de https://tesis.ipn.mx/jspui/bitstream/123456789/18755/1/Prototipo%20cortador%20y%20grabador%20l%C3%A1ser.pdf

Cheng, Y. (27 de Enero de 2022). *Math Works*. Obtenido de Math Works: Ye Cheng (2022). Conducción de https://www.mathworks.com/matlabcentral/fileexchange/40768-transient-heat-conduction

Concepto Definición;. (27 de Enero de 2022). Obtenido de https://conceptodefinicion.de/robot/

Cruz Carillo , N. (3 de Septiembre de 2018). Dimensionamiento e implementación de una máquina CNC de corte por láser para optimizar la calidad de trabajos en acrílico de hasta 5 mm de espesor. 5-50.

Enriquez Herrador Rafael. (13 de Noviembre de 2019). *Guía de usuario Arduino*. Obtenido de https://www.uco.es/aulasoftwarelibre/wp-content/uploads/2010/05/Arduino_user_manual_es.pdf

Fernandez Alzate, O. (5 de Noviembre de 2019). *Código Electrónica*. Obtenido de Código Electrónica: http://codigoelectronica.com/blog/lm35-datasheet

Fernandez Benjamin Alonso, A. T. (2010). *El láser la luz de nuestro tiempo.* Salamanca, España: Oxford. Recuperado el 2 de octubre de 2018, de http://laser.usal.es/posgrado/wp-content/uploads/2012/03/El_laser.pdf

Ibarra Villalon, G. V. (2017). El caminohacia la luz láser. *Revista Mexicana de Física*.

L. Bachs, J. C. (1988). *Aplicaciones industriales del láser* (Vol. 19). Marcombo. Recuperado el 05 de Noviembre de 2018

Lascano, R. R. (2018). Diseño y construcción de una máquina de grabado. *Repositorio digital de la universidad de Israel*. Recuperado el 8 de octubre de 2018, de http://repositorio.uisrael.edu.ec/handle/47000/1621

M.G.Pérez Artieda, F. C. (Julio-Agosto de 2008). Revisión sobre recubrimientos láser de aleaciones de aluminio. *Revista de metalurgia, 44*(4), 14. Recuperado el 5 de Septiembre de 2018, de revistademetalurgia.revistas.csic.es/index.php/revistademetalurgia/article/v

Mantilla Lopez , J. (13 de Enero de 2017). Diseño y progamación de un eje

rotatorio para la máquina corte de láser NTC TLM610. *Rwepositorio Institucional*, 1-8. Recuperado el 3 de septirmbre de 2018, de repository.udistrital.edu.co/handle /11349/6347

NX Solución CAE, C. (s.f.). Obtenido de https://www.3dcadportal.com/nx.html

Ortega Pozo, D. (11 de Enero de 2007). Proyecto de implantación de una empresa de corte por agua y láser. *Refseek*, 3-48. Recuperado el 03 de Septiembre de 2018, de https://upcommons.upc.edu/bitstream/handle/2099.1/3903/55856-5.pdf

Rodriguez, F. D. (2011). Proceso de maquinado sin arranque de viruta. *Toda la UNAM en linea*, 1-16. Recuperado el 5 de Septiembre de 2018, de olimpia.cuautitlan2.unam.mx/pagina_ingenieria/mecanica/.mat.mat_mec/m2 /proceso_maquinado.pdf

Serrano Hernandez Arturo, R. L. (2018). Análisis térmico por el metodo del elemento finito para determinar las temperaturas internas de las capas aislantes en el enclousurer de una turbina de gas. *Revista ciencia ,Ingenieria ydesarrollo tec lerdo*, 243. Recuperado el 10 de octubre de 2018

Valenzuela, J. E. (Marzo de 2011). *Repositorio institucional de documentos.* Recuperado el 1 de octubre de 2018, de http://zaguan.unizar.es/record/5646#

ANEXOS

Enseguida se muestra un fragmento del código de programación utilizado para la trayectoria establecida para el brazo robótico.

```
#include <Servo.h>

//Declaración de servomotores//
Servo servomotor base;
Servo servomotorbrazo;
Servo servomotorhombro;
Servo   servomotorlaser;

//Declaración de posiciones iniciales//
int posbase = 90 ;
int posbrazo=180 ;
int poshombro= 150;
int poslaser =0;
int pos= 0;


void setup() {
  servomotorbase.attach(2);
  servomotorbrazo.attach(3);
  servomotorhombro.attach(4);
  servomotorlaser.attach(5);

             }
void loop() {

//servomotorhombro.write(poshombro);
//delay(20);
//servomotorbase.write(posbase);
//delay(20);
//servomotorbrazo.write(posbrazo);
//delay(20);
//servomotorlaser.write(poslaser);
//delay(20);
  while(pos>=0){

//POSICIONAMIENTO DEL BRAZO

    for (pos = 180; pos >= 177; pos -= 1)
     {
     servomotorbrazo.write(pos) ;
      delay(100);
      }
       for (pos = 177; pos >= 174; pos -= 1)
     {
      servomotorbrazo.write(pos) ;
     delay(10);
     }



////////////////////////////
```

```
///POSICIONAMIENTO DEL HOMBRO
   for (pos = 180; pos >= 175; pos -= 1)
      {
       servomotorhombro.write(pos) ;
        delay(70);
       }
 for (pos = 175; pos >= 170; pos -= 1)
     {
      servomotorhombro.write(pos) ;
       delay(70);
     }
for (pos = 170; pos >= 165; pos -= 1)
    {
    servomotorhombro.write(pos) ;
    delay(70);
    }
for (pos = 165; pos >= 160; pos -= 1)
    {
    servomotorhombro.write(pos) ;
    delay(70);
    }
for (pos = 160; pos >= 155; pos -= 1)
    {
    servomotorhombro.write(pos) ;
     delay(70);
    }



////////////////////////////////
///INICIO DE SECUENCIA
 for (pos = 142; pos <= 144; pos += 1)
      {
       servomotorbrazo.write(pos) ;
       delay(100);
      }
 for (pos = 155; pos >= 150; pos -= 1)
    {
    servomotorhombro.write(pos) ;
     delay(100);
    }



////////////////////////////////
///SECUENCIA  DOS

 for (pos = 144; pos <= 146; pos += 1)
      {
      servomotorbrazo.write(pos) ;
       delay(100);
      }
for (pos = 150; pos >= 145; pos -= 1)
    {
   servomotorhombro.write(pos) ;
     delay(100);
    }
```

```
////////////////////////////////////
////SECUENCIA TRES

 for (pos = 146; pos <= 155; pos += 1)
      {
      servomotorbrazo.write(pos) ;
       delay(100);
      }

for (pos = 150; pos >= 145; pos -= 1)
    {
   servomotorhombro.write(pos) ;
     delay(100);
    }

////////////////////////////////////////
//SECUENCIA LÁSER
 for (pos =0; pos <=10; pos += 1)
       {
         servomotorlaser.write(pos) ;
         delay(450);
       }
   for (pos =10; pos <=20; pos += 1)
        {
        servomotorlaser.write(pos) ;
          delay(450);
        }

for (pos =20; pos <=30; pos += 1)
        {
        servomotorlaser.write(pos) ;
          delay(450);
        }

for (pos = 30; pos >= 20; pos -= 1)
     {
      servomotorlaser.write(pos) ;
      delay(450);
      }
for (pos = 20; pos >= 10; pos -= 1)
     {
      servomotorlaser.write(pos) ;
      delay(450);
      }

for (pos = 10; pos >= 0; pos -= 1)
     {
      servomotorlaser.write(pos) ;
      delay(450);
      }



pos=-1;
  }
```

```
}
```

El código de programación de interfaz visual GUI de MATLAB, utilizado para la obtención de valores solicitados por NX de SIMENS se muestra a continuación.

```
function varargout = INTERFAZMATLABPROYECTO(varargin)
gui_Singleton = 1;

gui_State = struct('gui_Name',       mfilename, ...
                   'gui_Singleton',  gui_Singleton, ...
                   'gui_OpeningFcn', @INTERFAZMATLABPROYECTO_OpeningFcn, ...
                   'gui_OutputFcn',  @INTERFAZMATLABPROYECTO_OutputFcn, ...
                   'gui_LayoutFcn',  [] , ...
                   'gui_Callbakc',   []);
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 INTERFAZMATLABPROYECTO is made visible.
function INTERFAZMATLABPROYECTO_OpeningFcn(hObject, eventdata, handles,
varargin)

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

% Update handles structure
guidata(hObject, handles);

function varargout = INTERFAZMATLABPROYECTO_OutputFcn(hObject, eventdata,
handles)
% Get default command line output from handles structure
varargout{1} = handles.output;
function Densidadtext_Callback(hObject, eventdata, handles)


% --- Executes during object creation, after setting all properties.
function Densidadtext_CreateFcn(hObject, eventdata, handles)

.
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Calorespecificotext_Callback(hObject, eventdata, handles)
% handles    structure with handles and user data (see GUIDATA)

% --- Executes during object creation, after setting all properties.
function Calorespecificotext_CreateFcn(hObject, eventdata, handles)
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end

function Temperaturainicialtext_Callback(hObject, eventdata, handles)
% hObject    handle to Temperaturainicialtext (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 Temperaturainicialtext as
text
%        str2double(get(hObject,'String')) returns contents of
Temperaturainicialtext as a double


% --- Executes during object creation, after setting all properties.
function Temperaturainicialtext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Temperaturainicialtext (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 Temperaturafinaltext_Callback(hObject, eventdata, handles)
% hObject    handle to Temperaturafinaltext (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- Executes during object creation, after setting all properties.
function Temperaturafinaltext_CreateFcn(hObject, eventdata, handles)
```

```
if ispc && isequal(get(hObject,'BackgroundColor'),
get(0,'defaultUicontrolBackgroundColor'))
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in Calcularvalorespushbutton.
function Calcularvalorespushbutton_Callback(hObject, eventdata, handles)

densidad =str2double(get(handles.Densidadtext,'String'));

calorespecifico=str2double(get(handles.Calorespecificotext,'String'));
temperaturainicial=str2double(get(handles.Temperaturainicialtext,'String'));
temperaturafinal=str2double(get(handles.Temperaturafinaltext,'String'));
tiempo=str2double(get(handles.Tiempotext,'String'));
largo=str2double(get(handles.Largotext,'String'));
ancho=str2double(get(handles.Anchotext,'String'));
espesor=str2double(get(handles.Espesortext,'String'));
ancholuz=str2double(get(handles.Ancholuztext,'String'));
x1=str2double(get(handles.X1text,'String'));
x=str2double(get(handles.Xtext,'String'));
x2=str2double(get(handles.XXX2text,'String'));
y1=str2double(get(handles.Y1text,'String'));
y2=str2double(get(handles.Y2text,'String'));
xx1=str2double(get(handles.XX1text,'String'));
xx=str2double(get(handles.XXtext,'String'));
xx2=str2double(get(handles.XX2text,'String'));
yy1=str2double(get(handles.YY1text,'String'));
yy2=str2double(get(handles.YY2text,'String'));
xxx1=str2double(get(handles.XXX1text,'String'));
xxx=str2double(get(handles.XXXtext,'String'));
xxx2=str2double(get(handles.XXX2text,'String'));
yyy1=str2double(get(handles.YYY1text,'String'));
yyy2=str2double(get(handles.YYY2text,'String'));
emisividad=str2double(get(handles.Emisividadtext,'String'));


masa=densidad*(largo*ancho*espesor);
resucalortransferido=masa * calorespecifico * (temperaturafinal-
temperaturainicial);
velocidaddetransferenciadecalor= resucalortransferido/tiempo*(1000);
Volumen=largo*ancho*espesor;
areadetransferenciadecalor=ancholuz*ancho;
viscocidadcinematica=y1+((x-x1)/(x2-x1))*(y2-y1);
numerodeprandtl=yy1+(((xx-xx1)/(xx2-xx1))*(yy2-yy1));
numeroderayleigh=(((9.81)*(1/((temperaturainicial+temperaturafinal)/2
+273))*(temperaturafinal-temperaturainicial)*((ancho/4)^3)
)/viscocidadcinematica^2)*numerodeprandtl;
numerodenusselt=0.54*((numeroderayleigh)^0.25);
coeficientedetransferenciadecalorporconveccion=(yyy1+((xxx-xxx1)/(xxx2-
xxx1))*(yyy2-yyy1))/(ancho/4)*numerodenusselt;
```

```
indicedetransferenciadecalor=(coeficientedetransferenciadecalorporconveccion)*(
ancholuz*ancho)*(temperaturafinal-temperaturainicial);
transferenciadecalorporradiacion=((emisividad)*(5.67e-
8)*(ancho*largo)*((temperaturafinal+273)^4-
(temperaturainicial+273)^4))/(ancho*largo);
peridatotaldecalor=transferenciadecalorporradiacion+velocidaddetransferenciadec
alor+(indicedetransferenciadecalor/areadetransferenciadecalor);

set(handles.Masa,'string',masa);
set(handles.Calortransferido,'string',resucalortransferido);
set(handles.Velocidadetransferencia,'string',velocidaddetransferenciadecalor);
set(handles.Volumenobtenido,'string',Volumen);
set(handles.Areadetransferenciacalor,'string',areadetransferenciadecalor);
set(handles.Viscocidadcinematica,'string',viscocidadcinematica);
set(handles.Numerodeprandlt,'string',numerodeprandtl);
set(handles.Numeroderayleigh,'string',numeroderayleigh);
set(handles.Numerodenusselt,'string',numerodenusselt);
set(handles.Coeficientedetransferenciadecalorporconveccion,'string',coeficiente
detransferenciadecalorporconveccion);
set(handles.Indicedetransferenciadecalor,'string',indicedetransferenciadecalor)
;
set(handles.Tansferenciadecalorporradiacion,'string',transferenciadecalorporrad
iacion);
set(handles.Perdidatotaldecalor,'string',peridatotaldecalor);


function Masa_Callback(hObject, eventdata, handles)
% hObject    handle to Masa (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 Masa as text
%        str2double(get(hObject,'String')) returns contents of Masa as a double


% --- Executes during object creation, after setting all properties.
function Masa_CreateFcn(hObject, eventdata, handles)
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 Calortransferido_Callback(hObject, eventdata, handles)
% hObject    handle to Calortransferido (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 Calortransferido as text
```

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


% --- Executes during object creation, after setting all properties.
function Calortransferido_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Calortransferido (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 Tiempotext_Callback(hObject, eventdata, handles)
% hObject    handle to Tiempotext (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 Tiempotext as text
%        str2double(get(hObject,'String')) returns contents of Tiempotext as a
double


% --- Executes during object creation, after setting all properties.
function Tiempotext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Tiempotext (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 Velocidadetransferencia_Callback(hObject, eventdata, handles)
% hObject    handle to Velocidadetransferencia (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 Velocidadetransferencia as
text
%        str2double(get(hObject,'String')) returns contents of
Velocidadetransferencia as a double
```

```
% --- Executes during object creation, after setting all properties.
function Velocidadetransferencia_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Velocidadetransferencia (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)

% 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 Largotext_Callback(hObject, eventdata, handles)
% hObject    handle to Largotext (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 Largotext as text
%        str2double(get(hObject,'String')) returns contents of Largotext as a
double

% --- Executes during object creation, after setting all properties.
function Largotext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Largotext (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 Anchotext_Callback(hObject, eventdata, handles)
% hObject    handle to Anchotext (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 Anchotext as text
%        str2double(get(hObject,'String')) returns contents of Anchotext as a
double


% --- Executes during object creation, after setting all properties.
function Anchotext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Anchotext (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 Espesortext_Callback(hObject, eventdata, handles)
% hObject    handle to Espesortext (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 Espesortext as text
%        str2double(get(hObject,'String')) returns contents of Espesortext as a
double


% --- Executes during object creation, after setting all properties.
function Espesortext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Espesortext (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)

% 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

function Volumenobtenido_Callback(hObject, eventdata, handles)
% hObject    handle to Volumenobtenido (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 Volumenobtenido as text
%        str2double(get(hObject,'String')) returns contents of Volumenobtenido
as a double


% --- Executes during object creation, after setting all properties.
function Volumenobtenido_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Volumenobtenido (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 Ancholuztext_Callback(hObject, eventdata, handles)
% hObject    handle to Ancholuztext (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 Ancholuztext as text
%        str2double(get(hObject,'String')) returns contents of Ancholuztext as
a double


% --- Executes during object creation, after setting all properties.
function Ancholuztext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ancholuztext (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 Areadetransferenciacalor_Callback(hObject, eventdata, handles)
% hObject    handle to Areadetransferenciacalor (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 Areadetransferenciacalor as
text
%        str2double(get(hObject,'String')) returns contents of
Areadetransferenciacalor as a double


% --- Executes during object creation, after setting all properties.
function Areadetransferenciacalor_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Areadetransferenciacalor (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 Numeroderayleigh_Callback(hObject, eventdata, handles)
% hObject    handle to Numeroderayleigh (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 Numeroderayleigh as text
%        str2double(get(hObject,'String')) returns contents of Numeroderayleigh
as a double


% --- Executes during object creation, after setting all properties.
function Numeroderayleigh_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Numeroderayleigh (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 X1text_Callback(hObject, eventdata, handles)
% hObject    handle to X1text (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 X1text as text
%        str2double(get(hObject,'String')) returns contents of X1text as a
double


% --- Executes during object creation, after setting all properties.
function X1text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to X1text (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 Xtext_Callback(hObject, eventdata, handles)
% hObject    handle to Xtext (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 Xtext as text
%        str2double(get(hObject,'String')) returns contents of Xtext as a
double


% --- Executes during object creation, after setting all properties.
function Xtext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Xtext (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 XXX2text_Callback(hObject, eventdata, handles)
% hObject    handle to XXX2text (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 XXX2text as text
%        str2double(get(hObject,'String')) returns contents of XXX2text as a
double


% --- Executes during object creation, after setting all properties.
function XXX2text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XXX2text (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 Y1text_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to Y1text (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 Y1text as text
%        str2double(get(hObject,'String')) returns contents of Y1text as a
double

% --- Executes during object creation, after setting all properties.
function Y1text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Y1text (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 Y2text_Callback(hObject, eventdata, handles)
% hObject    handle to Y2text (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 Y2text as text
%        str2double(get(hObject,'String')) returns contents of Y2text as a
double

% --- Executes during object creation, after setting all properties.
function Y2text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Y2text (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 Viscocidadcinematica_Callback(hObject, eventdata, handles)
% hObject    handle to Viscocidadcinematica (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 Viscocidadcinematica as text
```

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


% --- Executes during object creation, after setting all properties.
function Viscocidadcinematica_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Viscocidadcinematica (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 XX1text_Callback(hObject, eventdata, handles)
% hObject    handle to XX1text (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 XX1text as text
%        str2double(get(hObject,'String')) returns contents of XX1text as a
double


% --- Executes during object creation, after setting all properties.
function XX1text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XX1text (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 XXtext_Callback(hObject, eventdata, handles)
% hObject    handle to XXtext (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 XXtext as text
```

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

% --- Executes during object creation, after setting all properties.
function XXtext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XXtext (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 XX2text_Callback(hObject, eventdata, handles)
% hObject    handle to XX2text (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 XX2text as text
%        str2double(get(hObject,'String')) returns contents of XX2text as a
double


% --- Executes during object creation, after setting all properties.
function XX2text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XX2text (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 YY1text_Callback(hObject, eventdata, handles)
% hObject    handle to YY1text (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 YY1text as text
%        str2double(get(hObject,'String')) returns contents of YY1text as a
double
```

```
% --- Executes during object creation, after setting all properties.
function YY1text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YY1text (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 YY2text_Callback(hObject, eventdata, handles)
% hObject    handle to YY2text (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 YY2text as text
%        str2double(get(hObject,'String')) returns contents of YY2text as a
double


% --- Executes during object creation, after setting all properties.
function YY2text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YY2text (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 Numerodeprandlt_Callback(hObject, eventdata, handles)
% hObject    handle to Numerodeprandlt (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 Numerodeprandlt as text
%        str2double(get(hObject,'String')) returns contents of Numerodeprandlt
as a double


% --- Executes during object creation, after setting all properties.
function Numerodeprandlt_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Numerodeprandlt (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 Numerodenusselt_Callback(hObject, eventdata, handles)
% hObject    handle to Numerodenusselt (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 Numerodenusselt as text
%        str2double(get(hObject,'String')) returns contents of Numerodenusselt
as a double

% --- Executes during object creation, after setting all properties.
function Numerodenusselt_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Numerodenusselt (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 Coeficientedetransferenciadecalorporconveccion_Callback(hObject,
eventdata, handles)
% hObject    handle to Coeficientedetransferenciadecalorporconveccion (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
Coeficientedetransferenciadecalorporconveccion as text
%        str2double(get(hObject,'String')) returns contents of
Coeficientedetransferenciadecalorporconveccion as a double


% --- Executes during object creation, after setting all properties.
function Coeficientedetransferenciadecalorporconveccion_CreateFcn(hObject,
eventdata, handles)
```

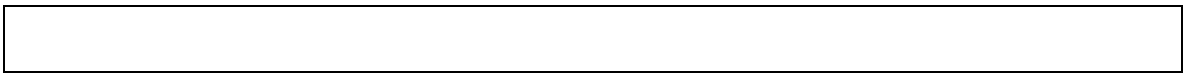
```
% hObject    handle to Coeficientedetransferenciadecalorporconveccion (see
GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% 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 XXX1text_Callback(hObject, eventdata, handles)
% hObject    handle to XXX1text (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 XXX1text as text
%        str2double(get(hObject,'String')) returns contents of XXX1text as a
double


% --- Executes during object creation, after setting all properties.
function XXX1text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XXX1text (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 XXXtext_Callback(hObject, eventdata, handles)
% hObject    handle to XXXtext (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 XXXtext as text
%        str2double(get(hObject,'String')) returns contents of XXXtext as a
double


% --- Executes during object creation, after setting all properties.
function XXXtext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XXXtext (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 XXX3text_Callback(hObject, eventdata, handles)
% hObject    handle to XXX3text (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 XXX3text as text
%        str2double(get(hObject,'String')) returns contents of XXX3text as a
double


% --- Executes during object creation, after setting all properties.
function XXX3text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to XXX3text (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 YYY1text_Callback(hObject, eventdata, handles)
% hObject    handle to YYY1text (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 YYY1text as text
%        str2double(get(hObject,'String')) returns contents of YYY1text as a
double

% --- Executes during object creation, after setting all properties.
function YYY1text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YYY1text (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 YYY2text_Callback(hObject, eventdata, handles)
% hObject    handle to YYY2text (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 YYY2text as text
%        str2double(get(hObject,'String')) returns contents of YYY2text as a
double


% --- Executes during object creation, after setting all properties.
function YYY2text_CreateFcn(hObject, eventdata, handles)
% hObject    handle to YYY2text (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 Indicedetransferenciadecalor_Callback(hObject, eventdata, handles)
% hObject    handle to Indicedetransferenciadecalor (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 Indicedetransferenciadecalor
as text
%        str2double(get(hObject,'String')) returns contents of
Indicedetransferenciadecalor as a double


% --- Executes during object creation, after setting all properties.
function Indicedetransferenciadecalor_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Indicedetransferenciadecalor (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 Tansferenciadecalorporradiacion_Callback(hObject, eventdata, handles)
% hObject    handle to Tansferenciadecalorporradiacion (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
Tansferenciadecalorporradiacion as text
%        str2double(get(hObject,'String')) returns contents of
Tansferenciadecalorporradiacion as a double


% --- Executes during object creation, after setting all properties.
function Tansferenciadecalorporradiacion_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Tansferenciadecalorporradiacion (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 Emisividadtext_Callback(hObject, eventdata, handles)
% hObject    handle to Emisividadtext (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 Emisividadtext as text
%        str2double(get(hObject,'String')) returns contents of Emisividadtext
as a double


% --- Executes during object creation, after setting all properties.
function Emisividadtext_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Emisividadtext (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 Perdidatotaldecalor_Callback(hObject, eventdata, handles)
% hObject    handle to Perdidatotaldecalor (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 Perdidatotaldecalor as text
%        str2double(get(hObject,'String')) returns contents of
Perdidatotaldecalor as a double


% --- Executes during object creation, after setting all properties.
function Perdidatotaldecalor_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Perdidatotaldecalor (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 edit70_Callback(hObject, eventdata, handles)
% hObject    handle to edit70 (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 edit70 as text
%        str2double(get(hObject,'String')) returns contents of edit70 as a
double


% --- Executes during object creation, after setting all properties.
function edit70_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit70 (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
```



Mediante el siguiente código la interfaz visual controla el brazo robótico, potencia de láser, adquisición de temperaturas y simulación del comportamiento del material

```
function varargout = TRANSFERENCIAB6(varargin)

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
    'gui_Singleton',  gui_Singleton, ...
    'gui_OpeningFcn', @TRANSFERENCIAB7_OpeningFcn, ...
    'gui_OutputFcn',  @TRANSFERENCIAB7_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

% --- Executes just before TRANSFERENCIAB7 is made visible.
function TRANSFERENCIAB7_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
handles.a=arduino;
%Establecer comunicación entre arduino y MATLAB

%POSICIONAMIENTO DE BRAZO
set(handles.popupmenu3,'String',strseq('D',2:6));
%Mostrar lista de pines digitales del  2 al 6
handles.seleccion='D3';
%Predeterminar  el pin digital  3
handles.s=servo(handles.a,handles.seleccion);
%El objeto s es igaul  a servo  y seleccionar el pin digital

%POSICIONAMIENTO DE HOMBRO
set(handles.popupmenu1,'String',strseq('D',2:6));
%Mostrar lista de pines digitales del  2 al 6
handles.seleccion='D5';
%Predeterminar  el pin digital 5
handles.s2=servo(handles.a,handles.seleccion);
%El objeto s2 es igaul  a servo  y seleccionar el pin digital

%POSICIONAMIENTO DE LASER
set(handles.popupmenu2,'String',strseq('D',2:6));
```

```
%Mostrar lista de pines digitales del  2 al 6
handles.seleccion='D6';
%Predeterminar  el pin digital 6
handles.s3=servo(handles.a,handles.seleccion);
%El objeto s3 es igaul  a servo  y seleccionar el pin digital
 handles.output = hObject;
% handles.output = hObject;
% Update handles structure
guidata(hObject, handles);



% UIWAIT makes TRANSFERENCIAB7 wait for user response (see UIRESUME)
% uiwait(handles.figure1);


% % --- Outputs from this function are returned to the command line.
% function varargout = TRANSFERENCIAB7_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;

function handles = getpara(handles)
% Obtain the input to show the current Fo Number
handles.data.T_inicial = str2double(get(handles.T_inicial,'String'));
%Aquirir la temperatura inicial del  cuadro de texto T_inicial  y convertirlo a
número
handles.data.T_tope = str2double(get(handles.T_tope,'String'));
%Aquirir la  del Temperaturatope cuadro de texto T_tope y convertirlo a número
handles.data.T_fondo = str2double(get(handles.T_fondo,'String'));
%Aquirir la Temperaturafondo inicial del  cuadro de texto T_fondo y convertirlo
a número
handles.data.T_izquierda = str2double(get(handles.T_izquierda,'String'));
%Aquirir la Temperaturaizquierda del  cuadro de texto T_izquierda  y
convertirlo a número
handles.data.T_derecha = str2double(get(handles.T_derecha,'String'));
%Aquirir la Temperaturaderecha del  cuadro de texto T_derecha y convertirlo a
número
handles.data.Largo = str2double(get(handles.Largo,'String'));
%Aquirir el largo de la pieza del  cuadro de texto Largo y convertirlo a número
handles.data.Ancho = str2double(get(handles.Ancho,'String'));
%Aquirir el largo de la pieza  del  cuadro de texto Ancho y convertirlo a
número
handles.data.dx = str2double(get(handles.dx,'String'));
% El valor del largo elevar al cuadrado adquirir el valor y convertirlo a
núnmero
handles.data.dy = str2double(get(handles.dy,'String'));
```

```
%El valor del ancho elevar al cuadrado adquirir el valor y convertirlo a
núnmero

handles.data.Tiempototal = str2double(get(handles.Tiempototal,'String'));
%Aquirir el Tiempomaximo o tiempo total del  cuadro de texto  Tiempototal y
convertirlo a número
handles.data.dt = str2double(get(handles.dt,'String'));
%Adquirir el valor de dt   del cuadro de texto  dt y convertirlo  a número
handles.data.epsilon = str2double(get(handles.epsilon,'String'));
%Adquirir el valor de la convergencia del cuadro de texto epsilon  y
convertirlo a número.
listStrings = get(handles.listalp,'String');
%Definir la lista de valores de acuerdo a los mariales en list alp
domaintype = listStrings{get(handles.listalp,'Value')};
%Definir el dominio de la lista de materiales e incluir su difusividad térmica
switch domaintype

 case 'PVC'
        domainalp= 1.26984127e-4;
%Si fuera el material seleccionado PVC tomar el siguiente valor
    case  'MADERA'
       domainalp= 0.13e-6;
%Si fuera el material seleccionado  MADREA tomar el siguiente valor
    case  'POLIPROPILENO'
       domainalp= 1.1339843e-4;
%Si fuera el material seleccionado POLIPROPILENO tomar el siguiente valor

end
handles.data.Difusividad = domainalp;
set(handles.Difusividad,'String',handles.data.Difusividad);
% Compute the Fo number
handles.data.r_x = handles.data.Difusividad*handles.data.dt/handles.data.dx^2;
handles.data.r_y = handles.data.Difusividad*handles.data.dt/handles.data.dy^2;
fo = handles.data.r_x + handles.data.r_y;
set(handles.rx,'String',...
    sprintf('Note: Numerical stability requires
Fo=alpha*dt*(1/dx^2+1/dy^2)<=1/2 \n Current Fo = %g',fo));
if fo > 1/2
    warndlg({'Numerical stability requires Fo <= 1/2';
        sprintf('Current Fo = %g',fo)},'Numerically Unstable');
end


function conduction(handles)

set(handles.stop,'UserData',0);
%si el boton de stop  es igual a cero

Largo = handles.data.Largo;
%Largo es igual a la variable  global largo
Ancho = handles.data.Ancho;
```

```
%Ancho es igual a la variable  global ancho
dx = handles.data.dx;
%dx es igual a la variable global dx
dy = handles.data.dy;
%dy es igual a la variable global dy
Tiempototal = handles.data.Tiempototal;
%Tiempototal es igual a la variable Tiempototal
dt = handles.data.dt;
%dt es igual a la variable dt
epsilon = handles.data.epsilon;
%Epsilon es igual a la variable epsilon
r_x = handles.data.r_x;
%r_x es igual a la variable  r_x
r_y = handles.data.r_y;
%r_y es igual a la variable  r_y
nx = uint32(Largo/dx + 1);
%nx es igual a la unidad e 32 bits por (Largo/dx + 1);
ny = uint32(Ancho/dy + 1);
%ny es igual a la unidad e 32 bits por (Largo/dy + 1);
[X,Y] = meshgrid(linspace(0,Largo,nx),linspace(0,Ancho,ny));
%Evaluar la función [X,Y]  en los
T = handles.data.T_inicial*ones(ny,nx);
%T es igual a la variable T_inicial *el arreglo (nx, ny)
T(:,1) = handles.data.T_izquierda;
% T(:,1) es igual a la variable T_izquierda
T(:,end) = handles.data.T_derecha;
% T(:,end) es igual a la variable T_derecha
T(1,:) = handles.data.T_fondo;
%T(1,:)  es igual a la variable T_fondo
T(end,:) = handles.data.T_tope;
% T(end,:) es igual a la variable T_tope


% iteration, march in time
n = 0;
%n=0
nmax = uint32(Tiempototal/dt);
%nmax=unidad de 32 bits(Tiempototal/dt)
while n < nmax
    %Mientras la variable n es menor a nmax
    if get(handles.stop,'UserData') == 1
        %si el boton stop toma el valor igual a uno
        break
        %romper
    end
    %fin
    n = n + 1;
    %n es igual a n mas uno
    T_n = T;
    %   T_n = T
    for j = 2:ny-1
```

```
        for i = 2:nx-1
            T(j,i) = T_n(j,i) + r_x*(T_n(j,i+1)-2*T_n(j,i)+T_n(j,i-1))...
                + r_y*(T_n(j+1,i)-2*T_n(j,i)+T_n(j-1,i));
        end
    end
    if uint16(n/50) == n/50 % refresh the plot every 50 time steps to save time
        % plot temperature Tcontour
        handles.fig.cont = contourf(handles.Tcontour,X,Y,T,20);
        title(handles.Tcontour,sprintf('Time = %g s',n*dt)),
        colorbar('peer',handles.Tcontour),
        xlabel(handles.Tcontour,'x (m)'),ylabel(handles.Tcontour,'y (m)')
        axis(handles.Tcontour,'equal','tight'),

        pause(0.01)
    end
    % check for convergence
    err = max(max(abs((T-T_n))));
    if err <= epsilon
        break
    end

end



% --- Executes on button press in start.
function start_Callback(hObject, eventdata, handles)
% hObject    handle to start (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% % cla(handles.Tplot);

% CONTROL POTENCIA LÁSER

valor=str2num(get(handles.edit19,'String'));
%Obtener el valor del edit2 y convertirlo en valor númerico
valor = (valor/100);
%El valor  obtenido de edit2 dividirlo entre100
 writePWMDutyCycle(handles.a,'D11',valor);
 %Escribe en la salida PWM el valor
 guidata(hObject,handles);
cla(handles.Tcontour,'reset');
handles = getpara(handles);
conduction(handles);
guidata(hObject,handles);

% --- Executes on button press in stop.
function stop_Callback(hObject, eventdata, handles)
% hObject    handle to stop (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
set(handles.stop,'UserData',1);


% --- Executes on selection change in listalp.
function listalp_Callback(hObject, eventdata, handles)
% hObject    handle to listalp (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
getpara(handles);
% guidata(hObject,handles);

% Hints: contents = cellstr(get(hObject,'String')) returns listalp contents as
cell array
%        contents{get(hObject,'Value')} returns selected item from listalp


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



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



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


% --- Executes on button press in mupad.
function mupad_Callback(hObject, eventdata, handles)
% hObject    handle to mupad (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
open('ss_conduction_analytical.mn');
```

```
% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over T_inicial.
function T_inicial_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to T_inicial (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over T_tope.
function T_tope_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to T_tope (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



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



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



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


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over Largo.


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


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over largo.
function Largo_ButtonDownFcn(hObject, eventdata, handles)
```

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



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



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



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


% --- Executes on selection change in popupmenu3.
function popupmenu3_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu3 contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu3


% --- Executes during object creation, after setting all properties.
function popupmenu3_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu3 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu 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)

% 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 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)

% 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


% --- Executes on selection change in popupmenu2.
```

```
function popupmenu2_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu2 contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu2


% --- Executes during object creation, after setting all properties.
function popupmenu2_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu2 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu 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)

% 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


% --- Executes on selection change in popupmenu1.
```

```
function popupmenu1_Callback(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% Hints: contents = cellstr(get(hObject,'String')) returns popupmenu1 contents
as cell array
%        contents{get(hObject,'Value')} returns selected item from popupmenu1


% --- Executes during object creation, after setting all properties.
function popupmenu1_CreateFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called

% Hint: popupmenu 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


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over popupmenu1.
function popupmenu1_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to popupmenu1 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

% --- POSICIÓN LÁSER
function pushbutton6_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton6 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
escribirvalortres=str2num(get(handles.edit18,'String'));
for b =160:-0.5:escribirvalortres;
%Ejecuta el movimiento inicando en cuarenta, con avancxe de 0.5y detente en
%el valor escrito en edit 4.
writePosition(handles.s3,((b)/180));
 %Escribe la posición en el servo 3  y divide b/180 para la conversión a
 %grados.
 pause(0.10);
pos=readPosition(handles.s3)*180;
%Leer la posición  del servo s3 y multiplicarla 180
 pause(0.01);
 %Realice una pausa de 0.01
  disp(b);
  %Mostrar los valores
end
```

```
% MOVER HOMBRO
function pushbutton5_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
escribirvalordos=str2num(get(handles.edit17,'String'));
for a =180:-0.25:escribirvalordos
%Ejecuta el movimiento inicando en cuarenta, con avancxe de 0.25 y detente en
%el valor escrito en edit 17
 writePosition(handles.s2,((a)/180));
%Escribe la posición en el servo s2  y divide b/180 para la conversión a
 %grados.
pos=readPosition(handles.s2)*180;
 %Leer la posición  del servo s3 y multiplicarla 180
 pause(0.01);
 %Realice una pausa de 0.01
  disp(a);
  %Mostrar los valores
end

% MOVER BRAZO
function pushbutton8_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton5 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
escribirvalor=str2num(get(handles.edit20,'String'));
for i =150 :-0.25:escribirvalor
%Ejecuta el movimiento inicando en cuarenta, con avancxe de 0.25 y detente en
%el valor escrito en edit 20
 writePosition(handles.s,((i)/180));
 %Escribe la posición en el servo 3  y divide b/180 para la conversión a
 %grados.
pos=readPosition(handles.s)*180;
%Leer la posición  del servo s y multiplicarla 180
 pause(0.01);
%Realice una pausa de 0.01
  disp(i);
  %Mostrar los valores
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)

% 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


% --- Executes during object creation, after setting all properties.
function uipanel7_CreateFcn(hObject, eventdata, handles)
% hObject    handle to uipanel7 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% --- Executes during object creation, after setting all properties.
function uipanel8_CreateFcn(hObject, eventdata, handles)
% hObject    handle to uipanel8 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called


% --- Executes during object creation, after setting all properties.
function uipanel9_CreateFcn(hObject, eventdata, handles)
% hObject    handle to uipanel9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    empty - handles not created until after all CreateFcns called



function T_inicial_Callback(hObject, eventdata, handles)
% hObject    handle to T_inicial (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 T_inicial as text
%        str2double(get(hObject,'String')) returns contents of T_inicial as a
double


% --- Executes during object creation, after setting all properties.
function T_inicial_CreateFcn(hObject, eventdata, handles)
% hObject    handle to T_inicial (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 T_tope_Callback(hObject, eventdata, handles)
% hObject    handle to T_tope (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 T_tope as text
%        str2double(get(hObject,'String')) returns contents of T_tope as a
double


% --- Executes during object creation, after setting all properties.
function T_tope_CreateFcn(hObject, eventdata, handles)
% hObject    handle to T_tope (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 T_izquierda_Callback(hObject, eventdata, handles)
% hObject    handle to T_izquierda (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 T_izquierda as text
%        str2double(get(hObject,'String')) returns contents of T_izquierda as a
double


% --- Executes during object creation, after setting all properties.
function T_izquierda_CreateFcn(hObject, eventdata, handles)
% hObject    handle to T_izquierda (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 T_fondo_Callback(hObject, eventdata, handles)
% hObject    handle to T_fondo (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 T_fondo as text
%        str2double(get(hObject,'String')) returns contents of T_fondo as a
double


% --- Executes during object creation, after setting all properties.
function T_fondo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to T_fondo (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 T_derecha_Callback(hObject, eventdata, handles)
% hObject    handle to T_derecha (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 T_derecha as text
%        str2double(get(hObject,'String')) returns contents of T_derecha as a
double


% --- Executes during object creation, after setting all properties.
function T_derecha_CreateFcn(hObject, eventdata, handles)
% hObject    handle to T_derecha (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 Largo_Callback(hObject, eventdata, handles)
% hObject    handle to Largo (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 Largo as text
%        str2double(get(hObject,'String')) returns contents of Largo as a
double


% --- Executes during object creation, after setting all properties.
function Largo_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Largo (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 Ancho_Callback(hObject, eventdata, handles)
% hObject    handle to Ancho (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 Ancho as text
%        str2double(get(hObject,'String')) returns contents of Ancho as a
double


% --- Executes during object creation, after setting all properties.
function Ancho_CreateFcn(hObject, eventdata, handles)
% hObject    handle to Ancho (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 dx_CreateFcn(hObject, eventdata, handles)
% hObject    handle to dx (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


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over dx.
function dx_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to dx (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes during object creation, after setting all properties.
function dy_CreateFcn(hObject, eventdata, handles)
% hObject    handle to dy (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


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over dy.
function dy_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to dy (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
```

```
function LM351_Callback(hObject, eventdata, handles)
% hObject    handle to LM351 (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 LM351 as text
%        str2double(get(hObject,'String')) returns contents of LM351 as a
double


% --- Executes during object creation, after setting all properties.
function LM351_CreateFcn(hObject, eventdata, handles)
% hObject    handle to LM351 (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


% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over LM351.
function LM351_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to LM351 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)


% --- Executes on button press in pushbutton9.
function pushbutton9_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



% --- If Enable == 'on', executes on mouse press in 5 pixel border.
% --- Otherwise, executes on mouse press in 5 pixel border or over pushbutton9.
function pushbutton9_ButtonDownFcn(hObject, eventdata, handles)
% hObject    handle to pushbutton9 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)



function edit32_Callback(hObject, eventdata, handles)
```

```
% hObject    handle to edit32 (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 edit32 as text
%        str2double(get(hObject,'String')) returns contents of edit32 as a
double


% --- Executes during object creation, after setting all properties.
function edit32_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit32 (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 on button press in pushbutton10.
% function pushbutton10_Callback(hObject, eventdata, handles)
% % hObject    handle to pushbutton10 (see GCBO)
% % eventdata  reserved - to be defined in a future version of MATLAB
% % handles    structure with handles and user data (see GUIDATA)



function edit33_Callback(hObject, eventdata, handles)
% hObject    handle to edit33 (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 edit33 as text
%        str2double(get(hObject,'String')) returns contents of edit33 as a
double


% --- Executes during object creation, after setting all properties.
function edit33_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit33 (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


%ADQUISICIÓN DE TEMPERATURA INICIAL
function pushbutton12_Callback(hObject, eventdata, handles)


 v = readVoltage(handles.a,'A0');
% Le el voltaje por el puerto A0
  analog =((v*5000)/1023);
%
 temp = analog/10;
 disp(analog);
  set(handles.edit33,'String',num2str(analog));
pause(1);

guidata(hObject,handles);




% --- Executes on button press in pushbutton13.
function pushbutton13_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton13 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)

 v1 = readVoltage(handles.a,'A1');
  analog1 = ((v1)/(5/1023));
  temp1 = ((analog1)*(5/1023*100));
%  disp(temp);
  set(handles.edit34,'String',num2str(temp1));
pause(1);

    guidata(hObject,handles);

function edit34_Callback(hObject, eventdata, handles)
% hObject    handle to edit34 (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 edit34 as text
%        str2double(get(hObject,'String')) returns contents of edit34 as a
double


% --- Executes during object creation, after setting all properties.
function edit34_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit34 (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');


% --- Executes on button press in pushbutton14.
function pushbutton14_Callback(hObject, eventdata, handles)
% hObject    handle to pushbutton14 (see GCBO)
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
v2 = readVoltage(handles.a,'A2');
  analog2 = ((v2)/(5/1023));
  temp2 = ((analog2)*(5/1023*100));
%  disp(temp);
  set(handles.edit35,'String',num2str(temp2));
pause(1);

    guidata(hObject,handles);


function edit35_Callback(hObject, eventdata, handles)
% hObject    handle to edit35 (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 edit35 as text
%        str2double(get(hObject,'String')) returns contents of edit35 as a
double


% --- Executes during object creation, after setting all properties.
function edit35_CreateFcn(hObject, eventdata, handles)
% hObject    handle to edit35 (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
```

En la Figura 37 se muestra el diagrama electrónico utilizado para las pruebas de alimentación y control del módulo láser aplicación para control de potencia con MIT APP INVENTOR.

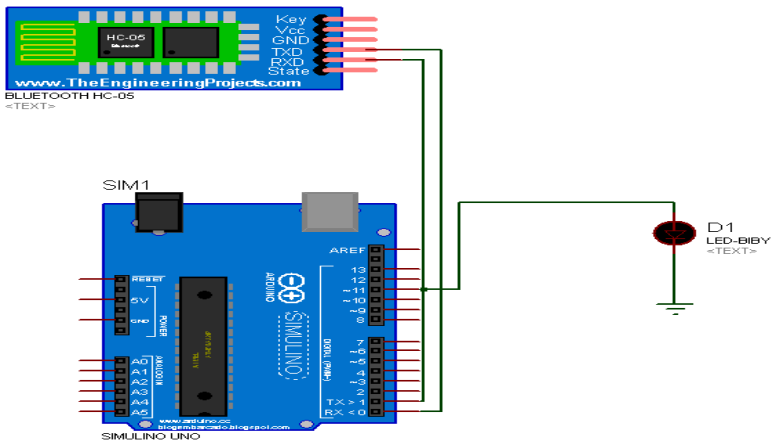


Figura 37 Control de potencia en módulo laser con Bluetoth

Diagrama de conexión para la interfaz utiliza que controla el brazo robótico, potencia de láser adquisición de temperaturas y simulación del comportamiento del material Figura 38.**¡Error! No se encuentra el origen de la referencia.**

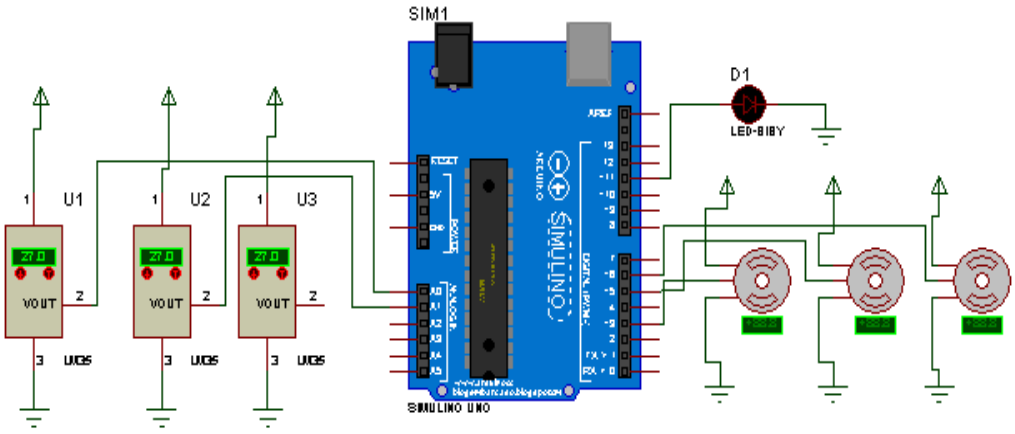


Figura 38 Diagrama conexión de interfaz

Las siguientes figuras muestran las partes que componen el brazo robótico utilizado en el movimiento del módulo láse. La Figura 39**¡Error! No se encuentra el origen de la referencia.** correspondiente al dimensionamiento de la base.

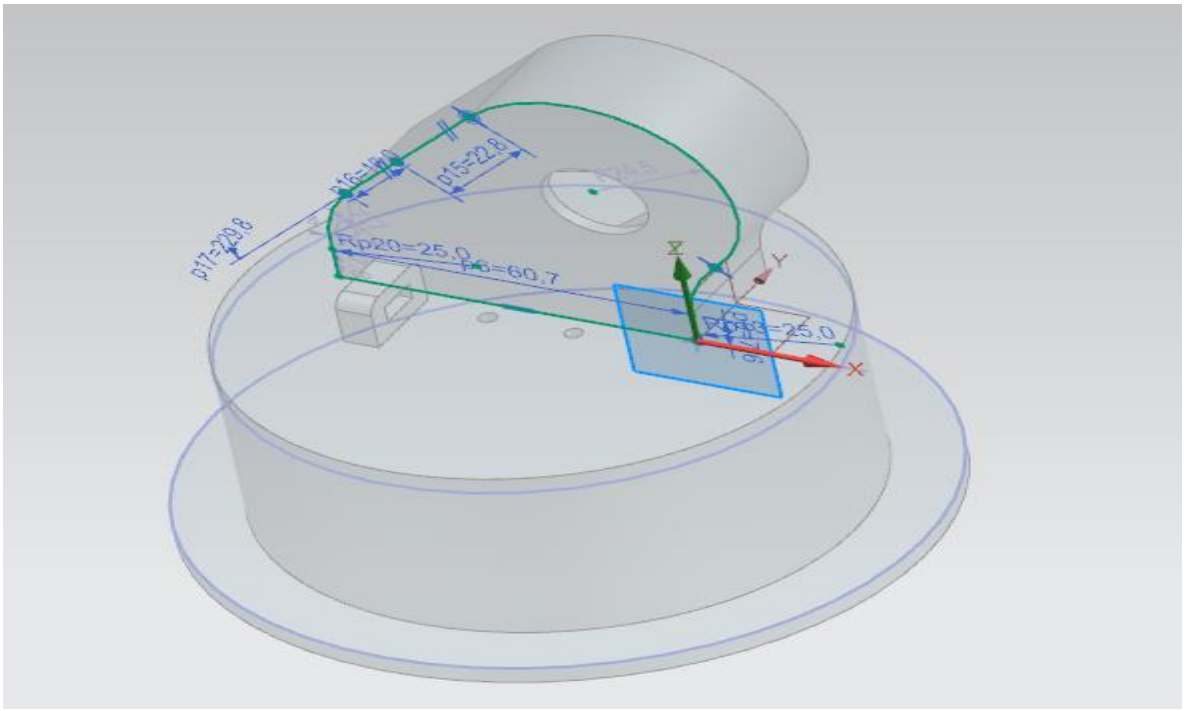


Figura 39 Parte superior de base

El módulo se apoya mediante un soporte de velcro, en parte del brazo robótico mostrando las medidas correspondientes de soporte y brazo Figura 40 y Figura 41.

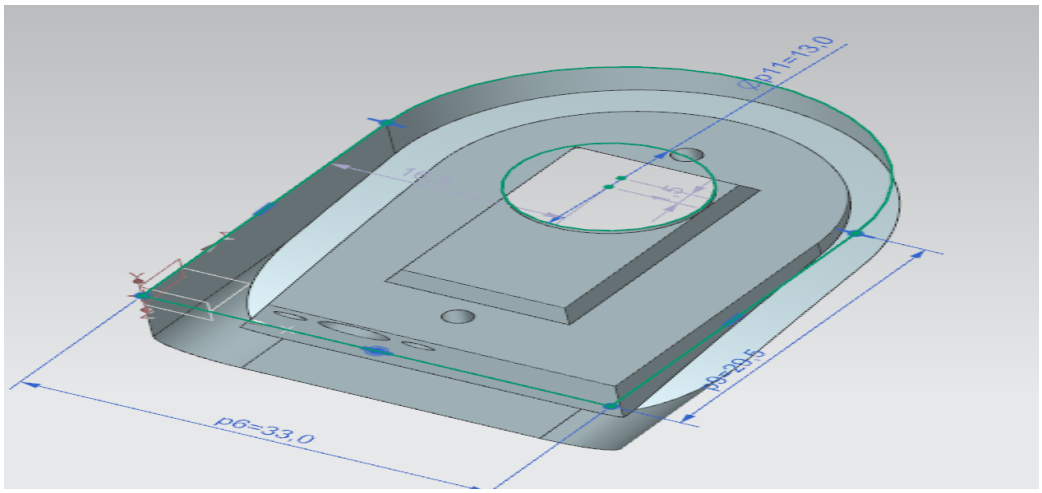


Figura 40 Soporte de módulo láser

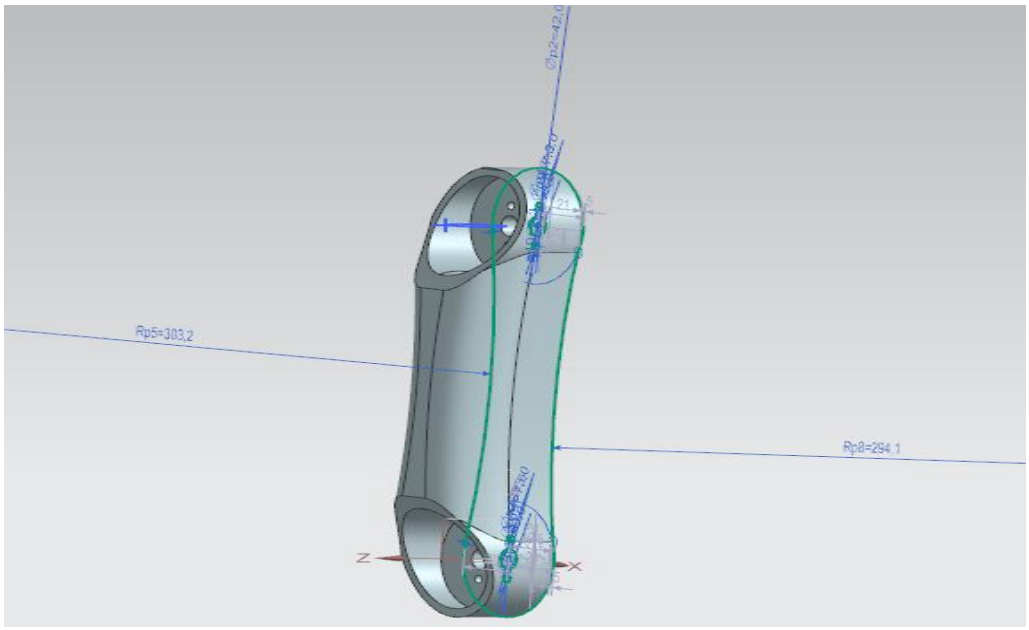


Figura 41 Brazo robótico

La parte del hombre es la que soporta a el brazo y módulo láser Figura 42**¡Error! No se encuentra el origen de la referencia.**.

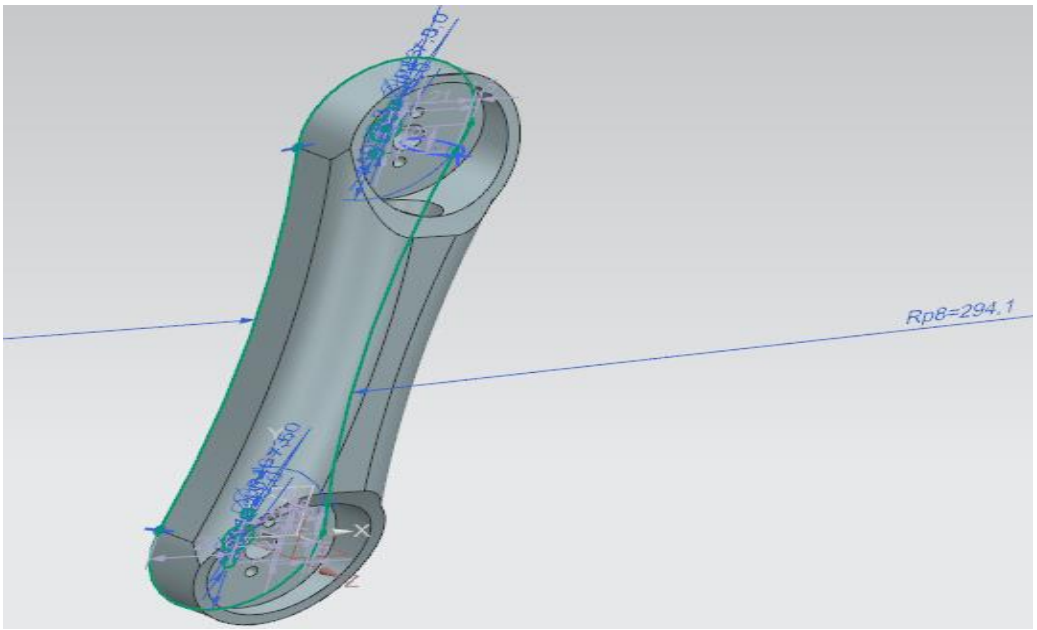


Figura 42 Hombro de brazo robótico