ESCUELA POLITÉCNICA NACIONAL

INFORME DE PROYECTO FINAL DE ELECTRÓNICA

DEPARTAMENTO DE FÍSICA

Montaje experimental para el análisis de un sistema angular para excitación de plasmones

Autores:
Nicolás A. Noriega C.
Juan J. Naranjo N.

Curso Dirigido por:
Eliana Acurio

Supervisor de Laboratorio:
Ricardo Araguillin

24 de enero de 2024

Índice

Índice de figuras

Índice de cuadros

Símbolo	Definición	Unidades
R	Resistencia	[Ω]
I	Corriente	[A]
V	Voltaje	[V]
C	Capacitancia	[F]
λ	Longitud de onda	[m]
c	Velocidad de la luz	[$m \cdot s^{-1}$]
f	Frecuencia	[Hz]
W	Potencia	[W]
S	Responsividad	[A/W]
T	Temperatura	[K]

Cuadro 1. *Tabla de nomenclaturas.*

1. Introducción

Mediante la colaboración entre el departamento de espectroscopía, se propuso colaborar con el magister y estudiante de doctorado Ricardo Araguillin, quien tiene el proyecto de análisis de un sistema angular para excitación de plasmones. Se informó que se necesitaba la colaboración con:

1. El control por comunicación serial del motor que determina el movimiento angular del prisma, y el registro de los datos con el fin de realizar un análisis posterior.

2. Adaptación del desplazamiento angular del prima en conjunto con el LED incidente.

3. Acondicionamiento del sensor mediante amplificadores de transimpedancia.

La importancia de este montaje experimental reside en la información que produce durante el desarrollo del mismo, y del alcance que puede tener el proyecto general de análisis de un sistema angular para excitación de plasmones.

1.1. Motivación

- Experimentar la utilidad de la materia Electrónica en el campo de la física, durante un proyecto real de relevancia académica.

- Explorar el alcance de los conocimientos adquiridos en el transcurso de la materia, mediante la investigación y resolución de problemas.

- Colaborar con miembros de la comunicad politécnica, específicamente con quienes poseen proyectos en el laboratorio de espectroscopia.

2. Descripción del proyecto

2.1. Surface plasmon resonance (SPR)

El SPR, o Resonancia Plasmónica de Superficie en español, es un fenómeno físico que ocurre cuando la luz incide en una superficie metálica y excita los electrones libres en la superficie del metal, generando una onda electromagnética conocida como plasmón de superficie.

Esta onda se propaga a lo largo de la interfaz entre el metal y el medio adyacente (como un líquido o un gas) y su frecuencia de resonancia depende de las propiedades del metal y del medio cercano. Al monitorear cambios en la frecuencia de resonancia del plasmón de superficie debido a la interacción de moléculas en el medio cercano, es posible detectar y medir la presencia y concentración de estas moléculas.

La técnica de SPR se ha utilizado ampliamente en diversas aplicaciones, incluyendo la investigación biomédica y la industria de los biosensores, para detectar la unión de moléculas y la interacción de proteínas, entre otras cosas. La técnica es no destructiva y altamente sensible, lo que la convierte en una herramienta valiosa para la investigación y el desarrollo de nuevos tratamientos y diagnósticos. [1–3]

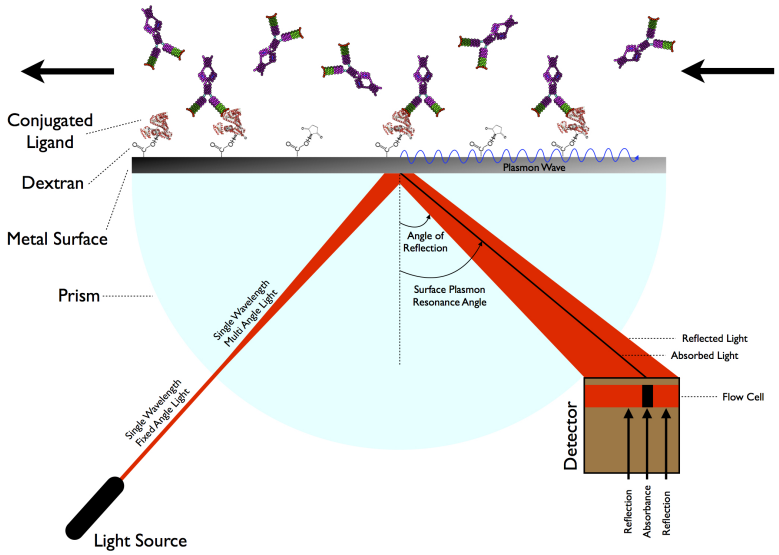


Figura 1. *Surface plasmon resonance*

2.2. Amplificadores de transimpedancia.

Un amplificador de transimpedancia es un tipo de amplificador electrónico que convierte una corriente en una señal de voltaje. Está diseñado para medir corrientes muy pequeñas, del orden de los picoamperios o nanoamperios, y se utiliza en aplicaciones como fotodetectores y células fotoeléctricas.

El amplificador de transimpedancia utiliza un amplificador operacional y una resistencia de realimentación para convertir la corriente de entrada en una señal de voltaje amplificada y de polaridad opuesta. La ganancia del amplificador se define por el valor de la resistencia de realimentación, y la corriente de entrada fluye a través de esta resistencia.

Los amplificadores de transimpedancia son ampliamente utilizados en sistemas de comunicaciones ópticas de alta velocidad, ya que son capaces de detectar señales ópticas muy débiles y convertirlas en señales eléctricas amplificadas para su procesamiento posterior. También se utilizan en la industria de los sensores para medir pequeñas corrientes generadas por sensores de baja corriente, como sensores de pH o sensores de gas. [4, 5]

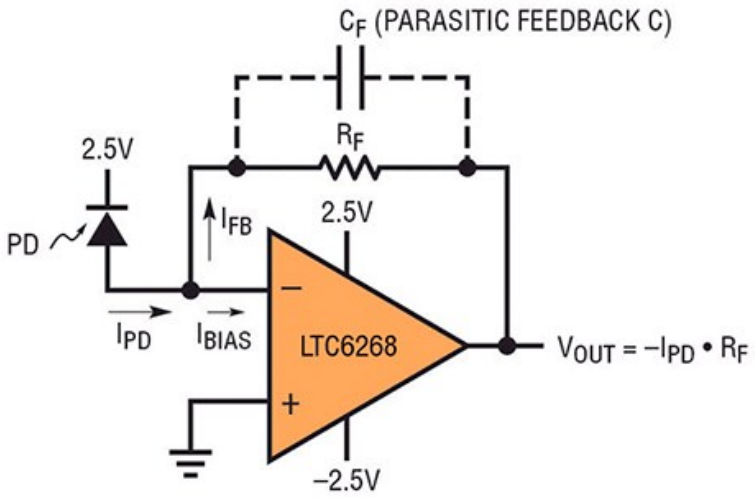


Figura 2. *Diagrama del amplificador de transimpedancia*

2.3. Estación de rotación con motores piezoeléctricos resonantes Thorlabs' Elliptec ELL18.

El motor Thorlabs' Elliptec ELL18 es un motor de paso a paso sin núcleo que proporciona una alta resolución y precisión de movimiento en aplicaciones ópticas y de posicionamiento. El motor cuenta con una interfaz de control de microcontrolador y un amplificador de corriente integrado que permite una fácil integración en sistemas de control automatizados.

El motor tiene una alta relación de engranajes de $288 : 1$, lo que permite un posicionamiento preciso y suave con una resolución de movimiento de $43{,}0 \mu rad$. El motor es compatible con una amplia variedad de controladores de motor y se puede utilizar en sistemas de posicionamiento de precisión, como microscopios y sistemas de microscopía de fluorescencia.

Además, el motor ELL18 tiene un diseño compacto y liviano que lo hace fácil de integrar en aplicaciones de espacio limitado. Su construcción sin núcleo permite un alto rendimiento con bajo calentamiento y baja vibración, lo que resulta en un movimiento suave y estable. En resumen, el motor Thorlabs' Elliptec ELL18 es una solución confiable y precisa para aplicaciones de posicionamiento y automatización ópticas.

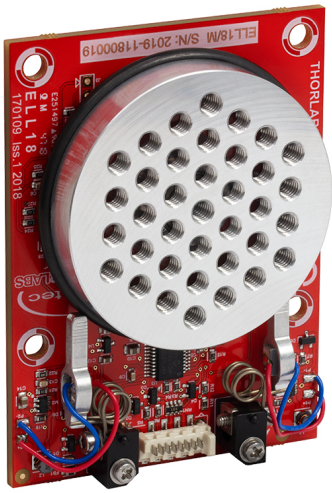


Figura 3. *Motor THOR ELL18*

2.4. Montaje experimental.

El montaje consiste en un espejo conectado a un motor que controla su ángulo de desplazamiento para reflejar la luz de una fuente. La luz reflejada pasa a través de un lente y luego incide en un prisma y una lámina de oro, para finalmente reflejarse y pasar por un segundo lente que desvía el haz hacia un detector, un fotodiodo. La señal del fotodiodo es acondicionada por un circuito de transimpedancia y los datos son adquiridos por un microcontrolador Arduino Uno.

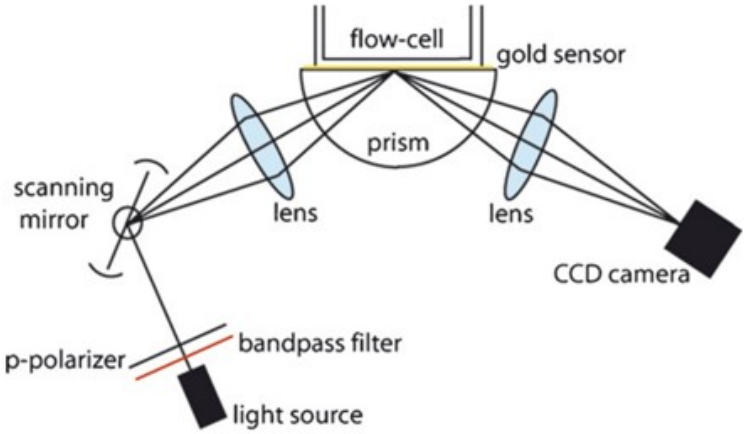


Figura 4. *Diagrama del montaje a aplicar*

3. Desarrollo

3.1. Evolución del código.

El motor THOR ELL18 es controlado por el software disponible en la página ELL mediante instrucciones en formato hexadecimal. Después de ejecutar cada instrucción, el motor devuelve su posición en un código hexadecimal que debe ser traducido para entender su significado.

Después de recibir los códigos hexadecimales a través de la comunicación serial, se pueden convertir a decimal y obtener un valor numérico representado como $\theta/(\alpha) = \#decimal$. Se realizaron varios ajustes para encontrar el valor de "α", utilizando 25 datos de diversas escalas de magnitud, y el valor óptimo obtenido fue de $4{,}38282x10 - 5rad$

Se realizó un código en Python utilizando la librería pyserial para establecer la comunicación serial con el motor THOR ELL18, basado en la investigación de los diversos ID de los comandos que se pueden ingresar en el protocolo de comunicación del motor. Además, se creó un código para transformar los datos de ángulo a decimal y hexadecimal, lo que permite al usuario ingresar directamente los ángulos sin necesidad de conocer los valores hexadecimales y guardarlos en una hoja de datos. Para ingresar datos mediante la librería pyserial, es necesario que los datos se encuentren codificados en formato UTF-8, por lo que se incluyó dicha codificación en el programa.

Posteriormente, luego de resolver errores del código y obtener una versión preliminar se inició la discusión para la consideración de diferentes programas para desarrollar la interfaz que permitirá controlar el motor, recolectar datos de posición angular y del fotodiodo, y graficar. Se planteó la posibilidad de unificar el programa en Python con el programa en ARDUINO para evitar problemas de concatenación. Se investigaron diversas plataformas, como GTK, MegunoLink, Tkinter y ECLIPSE, pero finalmente se decidió usar MATLAB para diseñar la interfaz.

Se mudo el código de Python a MATLAB, lo cual permitió la finalización de la interfaz gráfica, sin embargo, se planean distintas mejoras que permitirán optimizar el código y una mayor calidad de la interfaz y las gráficas que otorgue esta. Los códigos finales de Python y MATLAB, que son funcionales con el motor y su fin, se encuentran en los apéndices.

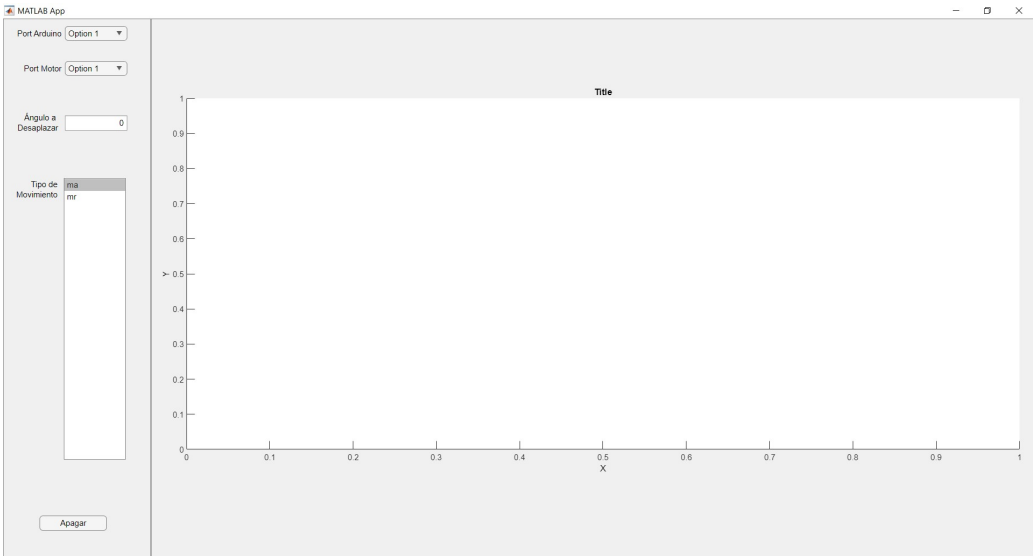


Figura 5. *Interfaz final en MATLAB.*

3.2. Circuito en protoboard y montado en una placa.

El primer prototipo del circuito de acondicionamiento para el sensor fue realizado en una protoboard, como se observa en la Fig.(6). Éste se basó en el circuito indicado en [4]. Entonces, se buscó la configuración de elementos capacitivos y resistivos que mejor amplificaban la señal de entrada. Conseguir una buena amplificación es importante porque la señal de entrada es considerablemente pequeña, y está contaminada con señales parásitas de luz incidente no deseada. Finalmente, se trabajó con una resistencia de $10\,\Omega$ y un capacitor de 100. Para estos valores, se trabajó y hallo los siguientes valores, según lo indicado en [5]:

$$
\begin{aligned}
Potencia &= 25mW \\
\lambda &= 645\ nm \\
S(\lambda) &= 0{,}3A/W \\
Ancho\ de\ banda &= 150\ kHz \\
Relacion\ seal/ruido &= 20\ dB \\
C_{termica\ tipica} &= 700\ pF \\
Area\ efectiva &= 9\ mm^2 \\
C_F &= 100\ nF \\
R_F &= 10\ \Omega \\
Corriente\ de\ Oscuridad &= 520\ mA \\
Corriente\ de\ base &= 23\ mA \\
Corriente\ potencia\ maxima &= 35\ mA \\
I_{termica} &= 0{,}16\ pA \\
R_{SN} = 100\ G\Omega \qquad & A\ 20°C \\
I_{Ruido\ electronico} &= 16{,}1\ nA \\
I_{SN} &= 23\ mA
\end{aligned}
$$

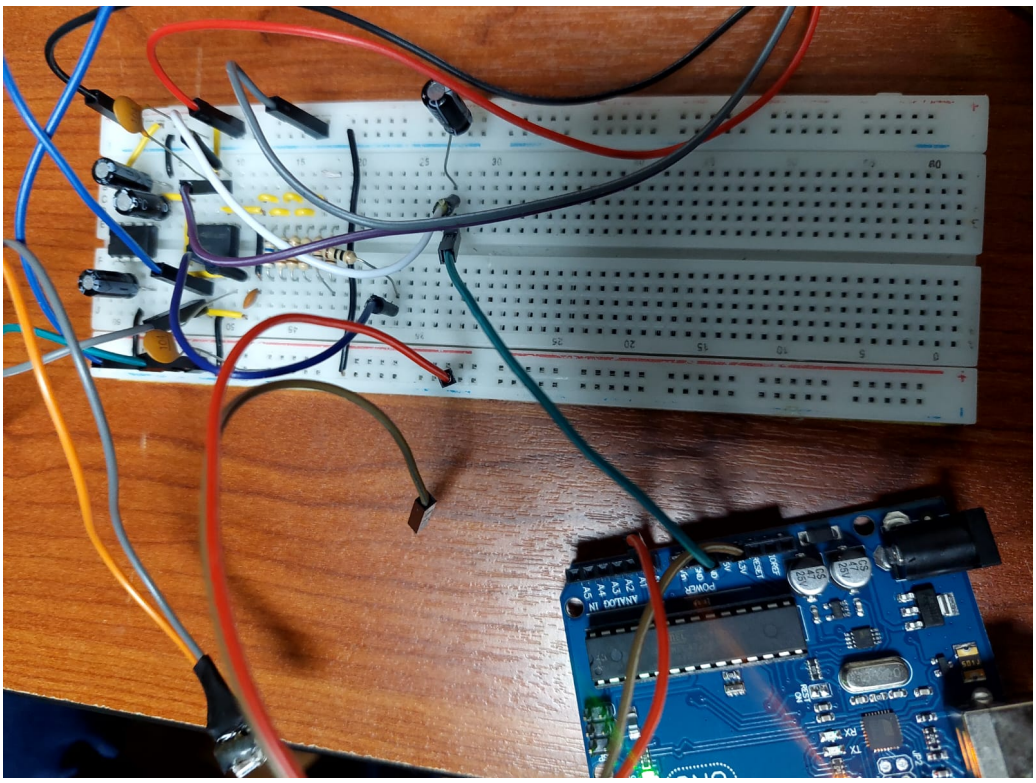


Figura 6. *Circuito de acondicionamiento en protoboard.*

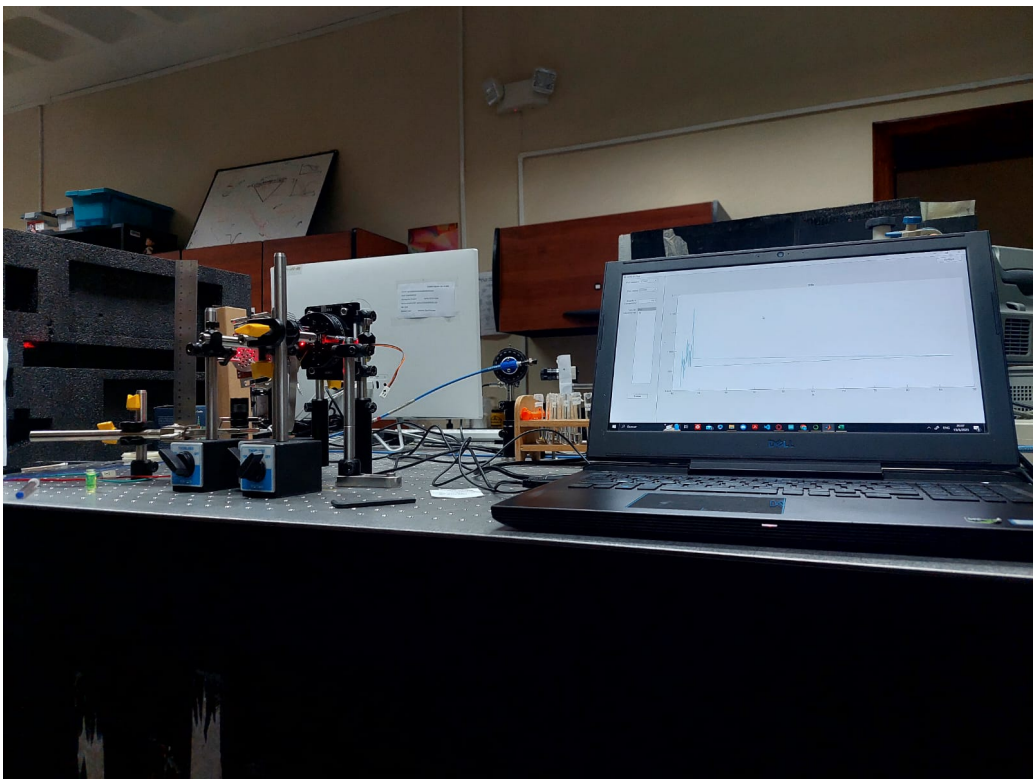
Una vez comprobado el funcionamiento del circuito, se le optimizó fabricando una placa que implemente el circuito en una placa más pequeña y con mayor control de los elementos. Un cambio razonable que hubo fue la eliminación de una capacitancia debida a la existente como parásita en el ground de la placa. Esta capacitancia fue usada como atenuador de ruido. Véase Fig.(7).

Figura 7. *Simulación de la placa que se fabricó e implementa el circuito de la Fig.(6).*

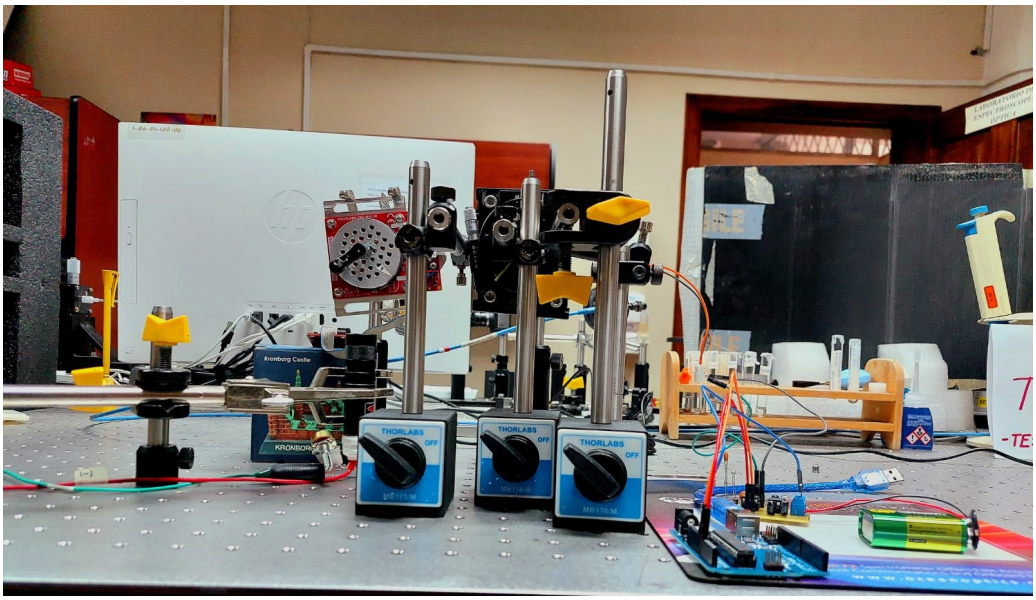
3.3. Setup Final

Finalmente, se trató con una etapa demorosa del proyecto: alinear los elementos. Una vez con el código y la interfaz en funcionamiento óptimo, se colocó los elementos del proyecto sobre la mesa y se hizo un proceso iterativo para coordinar ángulos, posiciones, y rotaciones de los elementos, de tal

forma que se reproduzca el esquema experimental. Se logró tener un intervalo de medición entre $41{,}5°$ y $43{,}6°$, suficiente para obtener gráficas útiles de las cuales se puede extraer información relevante para los fines del proyecto general.



(a) *Imagen del setup final, junto con la interfaz realizada.*



(b) *Imagen de únicamente el setup, donde observan el láser, el motor, el armado de lentes que incluye el repositorio de la lámina de oro y el sensor fotoeléctrico junto a la etapa de acondicionamiento.*

Figura 8. *Resultado final del proyecto.*

4. Conclusiones

Tras realizar el proyecto, se puede asegurar la utilidad y valor de los conocimientos adquiridos en la materia, y cómo los conceptos desarrollados en clase cobran relevancia dentro de un proyecto académico de impacto significativo en la comunidad de investigadores.

Los temas introducidos en clase llegan a ser las bases de los circuitos y programas con los que nos relacionamos diariamente. Nos permiten expandir nuestro conocimiento dentro del área de la electrónica para buscar e implementar las aplicaciones de circuitos más complejos o programas diseñados en otros lenguajes de comunicación además de Arduino.

Investigar sobre la aplicación de nuestros conocimientos de la materia, nos permite analizar nuevos proyectos y buscar formas de optimizarlos. Desde cambiar la posición física de un elemento del proyecto hasta programar completamente un nuevo código a usar, el proceso de analizar y mejorar viene acompañado del conocimiento previo del tema y de la capacidad de aprendizaje y comprensión de información nueva de los participantes del proyecto.

4.1. Nota final y agradecimiento.

El proyecto ha finalizado con resultados satisfactorios. Sin embargo, es posible realizar mejoras, tanto en la interfaz realizada como en el montaje experimental. Por ello, con la guía de Ricardo, se ha propuesto implementar estas mejoras para dejarle al laboratorio un montaje experimental de mayor calidad. Lo previsto es que las mejoras se realicen en el interciclo próximo.

Agradecemos el acompañamiento y paciencia de Ricardo y de los demás miembros del Laboratorio de Espectroscopia, y la guía y apoyo de nuestra Profesora Eliana Acurio. Ahora, reconocemos el aprendizaje como la oportunidad de generar cambio, y la experiencia obtenida como la garantía de la gran capacidad que poseemos de realizar lo necesario para producir el cambio. Queremos seguir aprendiendo y obteniendo experiencia, y lo haremos.

5. Apéndices

5.1. Apéndice A

Listing 1. *Código final implementado en MATLAB.*

```
classdef SPR_ELECTRONIC_PROYECT < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                matlab.ui.Figure
        GridLayout              matlab.ui.container.GridLayout
        LeftPanel               matlab.ui.container.Panel
        EnviarButton            matlab.ui.control.Button
        TipodeMovimientoListBox matlab.ui.control.ListBox
        TipodeMovimientoListBoxLabel matlab.ui.control.Label
        PortArduinoDropDown     matlab.ui.control.DropDown
        PortArduinoDropDownLabel matlab.ui.control.Label
        PortMotorDropDown       matlab.ui.control.DropDown
        PortMotorDropDownLabel  matlab.ui.control.Label
        nguloaDesaplazarEditField matlab.ui.control.NumericEditField
```

```
    nguloaDesaplazarEditFieldLabel matlab.ui.control.Label
    RightPanel              matlab.ui.container.Panel
    ApagarButton            matlab.ui.control.StateButton
    UIAxes                  matlab.ui.control.UIAxes
end

% Properties that correspond to apps with auto-reflow
properties (Access = private)
    onePanelWidth = 576;
end

     methods (Access = private)
       % Detecta los puertos
       function lista = deteccion_de_puertos(app)
          % Obtener una lista de puertos seriales disponibles
          ports = serialportlist;

          % Imprimir la lista de puertos seriales disponibles
          lista = {}; % Inicializar lista vaca

          for i = 1:numel(ports)
             lista{end+1} = ports{i}; % Agregar elemento a la lista
          end

          %Concatenacin con el menu
          app.PortArduinoDropDown.Items = lista;
          app.PortMotorDropDown.Items = lista;
          % Devolver la lista de puertos seriales
       end


    function promedio = Data_prom(~, port)
       % Creamos una lista vaca
       lista = [];

       % Pedimos al usuario que ingrese num datos y los agregamos a la
          lista
       for i = 1:10
          while true
             dato = readline(port);
             try
                valor = str2double(dato);
                lista = [lista, valor];
                break
             catch
                disp("Ingrese un valor numrico");
             end
          end
       end

       % Calculamos el promedio de los valores en la lista
       if numel(lista) > 0
          promedio = mean(lista);
```

```
            disp("La lista es:");
            disp(lista);
            disp("El promedio de los valores en la lista es:");
            disp(promedio);
        else
            disp("La lista est vaca");
        end
    end
     end

% Callbacks that handle component events
methods (Access = private)

    % Drop down opening function: PortArduinoDropDown
    function PortArduinoDropDownOpening(app, event)
        puertos = deteccion_de_puertos(app);
        disp(puertos)
        if isempty(puertos)
            app.PortArduinoDropDown.Enable = 'off';
            app.PortMotorDropDown.Enable = 'off';
            app.ApagarButton.Enable = 'off';
            uialert(app.UIFigure, "No se detectaron puertos", "Error")
            return
        end

        % Find the index of the selected port in the list of available
            ports
        motor_index = find(strcmp(puertos,
            app.PortMotorDropDown.Value));

        % Remove the selected port from the list of available ports
        ardu_index = setdiff(puertos, puertos{motor_index});

        % Update the options for the second drop-down
        app.PortMotorDropDown.Items = ardu_index;

        % Select the first option in the second drop-down
        app.PortMotorDropDown.Value = ardu_index(1);

        if isempty(ardu_index) || isempty(motor_index)
            app.PortArduinoDropDown.Enable = 'off';
            app.PortMotorDropDown.Enable = 'off';
            app.ConectarButton.Enable = 'off';
            uialert(app.UIFigure, "El puerto seleccionado no est
                disponible", "Error")
            return
        end

        motor = puertos{motor_index};

        % Asignar el puerto serial seleccionado a diferentes funciones
        % Configuracin del puerto
```

```
    port2 = serialport(motor, 9600, "DataBits", 8, "Parity",
        "none", "StopBits", 1);

    %Inicializacion
    configureTerminator(port2,"CR/LF");

    fopen(port2);

    init = strcat('0ho0');

    % Codificacin a cadena de bytes
    str_encoded = uint8(init);

    % Enviar cadena de bytes
    fwrite(port2,str_encoded );

    fclose(port2);
end

% Value changed function: ApagarButton
function ApagarButtonValueChanged(app, event)
    value = app.ApagarButton.Value;
    if value  % Si el valor es verdadero
        delete(gcf);  % Cerrar la figura actual
        delete(app);  % Cerrar la aplicacin
        % Verificar si los puertos estn abiertos
        if strcmp(port1.Status, 'open')
            % Cerrar el puerto 1
            fclose(port1);
        end
        if strcmp(port2.Status, 'open')
            % Cerrar el puerto 2
            fclose(port2);
        end

    end
end

% Button pushed function: EnviarButton
function EnviarButtonPushed(app, event)
    % Detecta los puertos
    puertos = deteccion_de_puertos(app);
    puertos2 = flip(puertos);

    % Asignacin de puertos
    ardu_index = strcmp(puertos2, app.PortArduinoDropDown.Value);
    motor_index = strcmp(puertos, app.PortMotorDropDown.Value);

    ardu = puertos2{ardu_index};
    motor = puertos{motor_index};

    % Asignar el puerto serial seleccionado a diferentes funciones
    % Configuracin del puerto
```

```
port2 = serialport(motor, 9600, "DataBits", 8, "Parity",
    "none", "StopBits", 1);

port1 = serialport(ardu, 9600, "DataBits", 8, "Parity", "none",
    "StopBits", 1);


%Apertura puerto
configureTerminator(port2,"CR/LF");

fopen(port2);

%configureTerminator(port1,"CR/LF");

fopen(port1);

% Llamamos a la funcin Data_prom y pasamos el objeto serialport
    como parmetro
promedio = Data_prom(app, port1);
% Conversin a hexadecimal y codificacin
des = app.nguloaDesaplazarEditField.Value; % Leer el valor del
    EditField
dec_num = (des * pi) / (4.38282 * 10^-5 * 180);
hexa = dec2hex(int32(dec_num),8);

% Tipo de movimiento
tipo = app.TipodeMovimientoListBox.Value;
hexa = strcat('0',tipo, hexa);

% Codificacin a cadena de bytes
str_encoded = uint8(hexa);

% Enviar cadena de bytes
fwrite(port2,hexa);

% Leer una lnea de respuesta
response = fread(port2, 11, 'uint8');
disp(response)
res = response(4:11);

% Convert the uint8 values to a binary string
binary_str = char(res);

% Concatenate all the binary characters into a single binary
    string
hexade = binary_str(:)';

% Conversin de respuesta a decimal
res_des = int32(hex2dec(hexade));
%disp(hex2dec(hexade))
%disp(res_des)
```

```
        % Conversin a grados de la posicin del motor
        save_des = (double(res_des)*180/pi)*(4.38282 * 10^-5) ;
        disp(save_des)

        % Open the text file for writing
        fileID = fopen('output.txt', 'a');

        % Write the variables to the file in the desired format
        fprintf(fileID, '%f %f\n', save_des, promedio);

        % Close the text file
        fclose(fileID);

        %     fclose(port1);
        fclose(port2);
    end

    % Button down function: UIAxes
    function UIAxesButtonDown(app, event)

        % Open the output.txt file
        fid = fopen('output.txt', 'r');
        if fid == -1
            uialert(app.UIFigure, 'Could not open output.txt', 'Error');
            return
        end

        % Read the data from the file
        data = textscan(fid, '%f %f', 'Delimiter', ' ',
            'MultipleDelimsAsOne', true);
        x = data{1};
        y = data{2};

        % Close the file
        fclose(fid);

        % Plot the data on the UIAxes
        plot(app.UIAxes, x, y);

        %Definicin de los lmites de los ejes
        xlim(app.UIAxes, [0 90]);
        ylim(app.UIAxes, [0 1]);
    end

    % Changes arrangement of the app based on UIFigure width
    function updateAppLayout(app, event)
        currentFigureWidth = app.UIFigure.Position(3);
        if(currentFigureWidth <= app.onePanelWidth)
            % Change to a 2x1 grid
            app.GridLayout.RowHeight = {480, 480};
            app.GridLayout.ColumnWidth = {'1x'};
            app.RightPanel.Layout.Row = 2;
            app.RightPanel.Layout.Column = 1;
```

```
            else
                % Change to a 1x2 grid
                app.GridLayout.RowHeight = {'1x'};
                app.GridLayout.ColumnWidth = {220, '1x'};
                app.RightPanel.Layout.Row = 1;
                app.RightPanel.Layout.Column = 2;
            end
        end
    end

    % Component initialization
    methods (Access = private)

        % Create UIFigure and components
        function createComponents(app)

            % Create UIFigure and hide until all components are created
            app.UIFigure = uifigure('Visible', 'off');
            app.UIFigure.AutoResizeChildren = 'off';
            app.UIFigure.Position = [100 100 640 480];
            app.UIFigure.Name = 'MATLAB App';
            app.UIFigure.SizeChangedFcn = createCallbackFcn(app,
                @updateAppLayout, true);

            % Create GridLayout
            app.GridLayout = uigridlayout(app.UIFigure);
            app.GridLayout.ColumnWidth = {220, '1x'};
            app.GridLayout.RowHeight = {'1x'};
            app.GridLayout.ColumnSpacing = 0;
            app.GridLayout.RowSpacing = 0;
            app.GridLayout.Padding = [0 0 0 0];
            app.GridLayout.Scrollable = 'on';

            % Create LeftPanel
            app.LeftPanel = uipanel(app.GridLayout);
            app.LeftPanel.Layout.Row = 1;
            app.LeftPanel.Layout.Column = 1;

            % Create nguloaDesaplazarEditFieldLabel
            app.nguloaDesaplazarEditFieldLabel = uilabel(app.LeftPanel);
            app.nguloaDesaplazarEditFieldLabel.HorizontalAlignment =
                'right';
            app.nguloaDesaplazarEditFieldLabel.Position = [25 274 66 30];
            app.nguloaDesaplazarEditFieldLabel.Text = {'ngulo a ';
                'Desaplazar'};

            % Create nguloaDesaplazarEditField
            app.nguloaDesaplazarEditField = uieditfield(app.LeftPanel,
                'numeric');
            app.nguloaDesaplazarEditField.Position = [107 278 100 22];

            % Create PortMotorDropDownLabel
            app.PortMotorDropDownLabel = uilabel(app.LeftPanel);
```

```
app.PortMotorDropDownLabel.HorizontalAlignment = 'right';
app.PortMotorDropDownLabel.Position = [38 343 61 22];
app.PortMotorDropDownLabel.Text = 'Port Motor';

% Create PortMotorDropDown
app.PortMotorDropDown = uidropdown(app.LeftPanel);
app.PortMotorDropDown.Items = {'Option 1', 'Option 2', 'Option
   3'};
app.PortMotorDropDown.Position = [104 343 100 22];

% Create PortArduinoDropDownLabel
app.PortArduinoDropDownLabel = uilabel(app.LeftPanel);
app.PortArduinoDropDownLabel.HorizontalAlignment = 'right';
app.PortArduinoDropDownLabel.Position = [25 395 71 22];
app.PortArduinoDropDownLabel.Text = 'Port Arduino';

% Create PortArduinoDropDown
app.PortArduinoDropDown = uidropdown(app.LeftPanel);
app.PortArduinoDropDown.Items = {'Option 2', 'Option 3',
   'Option 4'};
app.PortArduinoDropDown.DropDownOpeningFcn =
   createCallbackFcn(app, @PortArduinoDropDownOpening, true);
app.PortArduinoDropDown.Position = [104 395 100 22];
app.PortArduinoDropDown.Value = 'Option 2';

% Create TipodeMovimientoListBoxLabel
app.TipodeMovimientoListBoxLabel = uilabel(app.LeftPanel);
app.TipodeMovimientoListBoxLabel.HorizontalAlignment = 'right';
app.TipodeMovimientoListBoxLabel.Position = [22 219 70 30];
app.TipodeMovimientoListBoxLabel.Text = {'Tipo de'; '
   Movimiento'};

% Create TipodeMovimientoListBox
app.TipodeMovimientoListBox = uilistbox(app.LeftPanel);
app.TipodeMovimientoListBox.Items = {'ma', 'mr'};
app.TipodeMovimientoListBox.Position = [107 206 100 45];
app.TipodeMovimientoListBox.Value = 'ma';

% Create EnviarButton
app.EnviarButton = uibutton(app.LeftPanel, 'push');
app.EnviarButton.ButtonPushedFcn = createCallbackFcn(app,
   @EnviarButtonPushed, true);
app.EnviarButton.Position = [98 125 100 23];

% Create RightPanel
app.RightPanel = uipanel(app.GridLayout);
app.RightPanel.Layout.Row = 1;
app.RightPanel.Layout.Column = 2;

% Create UIAxes
app.UIAxes = uiaxes(app.RightPanel);
title(app.UIAxes, 'SPR')
xlabel(app.UIAxes, 'Angle ')
```

```
            ylabel(app.UIAxes, 'Intensity [V]')
            zlabel(app.UIAxes, 'Z')
            app.UIAxes.ButtonDownFcn = createCallbackFcn(app,
               @UIAxesButtonDown, true);
            app.UIAxes.Position = [14 125 390 256];

            % Create ApagarButton
            app.ApagarButton = uibutton(app.RightPanel, 'state');
            app.ApagarButton.ValueChangedFcn = createCallbackFcn(app,
               @ApagarButtonValueChanged, true);
            app.ApagarButton.Text = 'Apagar';
            app.ApagarButton.Position = [304 446 100 23];

            % Show the figure after all components are created
            app.UIFigure.Visible = 'on';
        end
    end

    % App creation and deletion
    methods (Access = public)

        % Construct app
        function app = SPR_ELECTRONIC_PROYECT

            % Create UIFigure and components
            createComponents(app)

            % Register the app with App Designer
            registerApp(app, app.UIFigure)

            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

5.2. Apéndice B

Listing 2. *Código final implementado, desarrollado en Python.*

```
#Librerias
import serial.tools.list_ports
import serial
import numpy as np
```

```
import pandas as pd
import matplotlib.pyplot as plt
from scipy.interpolate import make_interp_spline

   # Funcin que regresa el verdadero valor hexadecimal.
   # Por ejemplo, si recibe un 15 devuelve f, y si recibe un nmero menor
      a 10, devuelve el nmero sin modificarlo
   def obtener_caracter_hexadecimal(valor):
      # Lo necesitamos como cadena
      valor = str(valor)
      equivalencias = {
         "10": "A",
         "11": "B",
         "12": "C",
         "13": "D",
         "14": "E",
         "15": "F",
      }
      if valor in equivalencias:
         return equivalencias[valor]
      else:
         return valor


   def decimal_a_hexadecimal(decimal):
      hexadecimal = ""
      while decimal > 0:
         residuo = decimal % 16
         verdadero_caracter = obtener_caracter_hexadecimal(residuo)
         hexadecimal = verdadero_caracter + hexadecimal
         decimal = int(decimal / 16)
      return hexadecimal

   #tipo de movimiento
   def Tipo_de_movimiento(lower_bound, upper_bound):
      while True:
         try:
            number = int(input("Tipo de desplazamiento:Mov absoluto (1)
               o Mov Relativo (2)"))

            if number == lower_bound:
               tip = str("0ma")
               return tip
            if number == upper_bound:
               tip = str("0mr")
               return tip
         except ValueError:
            pass
         print(f"Por favor ingresa un nmero ({lower_bound} o
            {upper_bound})")

   #Detecta los puertos
   def Deteccin_de_puertos(puertos):
```

```
    # Obtener una lista de puertos seriales disponibles
    ports = serial.tools.list_ports.comports()

    # Imprimir la lista de puertos seriales disponibles
    print("Puertos seriales disponibles:")
    for i, port in enumerate(ports):
        print(f"{i+1}. {port.device}")

    # Permitir al usuario seleccionar un puerto serial
    selected_port = int(input("Seleccione el puerto para el " +
        puertos))

    # Asignar el puerto serial seleccionado a diferentes funciones
    #Configuracin del puerto
    port = serial.Serial(ports[selected_port-1].device, baudrate=9600,
        bytesize=8, parity='N', stopbits=1, timeout=None, xonxoff=0,
        rtscts=0)

    #Apertura del puerto

    #port.open()
    a = port.is_open

    #Comprobacin del estado del puerto e inicializacin

    if a:
        print("El puerto se encuentra abierto" + "\n")

    else:
        print("Puerto no disponible" + "\n")

    return port

#Fucin que que toma datos de un cierto perido y luego devuelve el
    promedio de ellos
def Data_prom(ser, num):
    # Creamos una lista vaca
    lista = []

    # Pedimos al usuario que ingrese 10 datos y los agregamos a la
        lista
    for i in range(num):
        while True:
            dato = ser.readline().decode().strip()
            try:
                valor = float(dato)
                lista.append(valor)
                break
            except:
                print("Ingrese un valor numrico")

    # Calculamos el promedio de los valores en la lista
```

```
        if len(lista) > 0:
            promedio = sum(lista) / len(lista)
            print("La lista es:", lista)
            print("El promedio de los valores en la lista es:", promedio)
        else:
            print("La lista est vaca")
        return promedio

    #Funcin que se encarga de graficar los datos obtenidos del fotodiodo
    def Graficar(name):

        # Cargar datos CSV sin etiquetas
        df = pd.read_csv('80.txt', names=['columna1', 'columna2'])

        # Eliminar duplicados de x
        x_unique, y_unique = np.unique(df['columna1'],
            return_index=False), df['columna2'][np.unique(df['columna1'],
            return_index=True)[1]]

        # Interpolar los datos
        x_smooth = np.linspace(x_unique[0], x_unique[-1], 200)
        y_smooth = make_interp_spline(x_unique, y_unique, k=3)(x_smooth)

        # Crear grfico con la curva suavizada
        plt.plot(x_smooth, y_smooth, color='red', label='Curva suavizada')

        #Crear grfico con la curva no suavizada
        plt.plot(x_unique, y_unique, color='blue', label='Curva no
            suavizada')

        # Mostrar los puntos originales
        plt.scatter(x_unique, y_unique, color='black', label='Datos
            obtenidos')

        #Barras de error
        plt.errorbar(x_unique, y_unique, xerr=2.51E-03, yerr=0.05,
            fmt='o', color='black',
                   ecolor='black', elinewidth=0.5, capsize=2)

        # Aadir leyenda
        plt.legend()

        # Agregar etiquetas y ttulo al grfico
        plt.xlabel('ngulo de incidencia  (grados)')
        plt.ylabel('Intensidad')
        plt.title('Curva de intensidad en funcin del ngulo de incidencia')


        # Mostrar el grfico
        return plt.show()

#Deteccin, apertura y comprobacin de los puertos:
```

```
mot = Deteccin_de_puertos("Motor :")
ardu = Deteccin_de_puertos("Arduino :")

#inicializacin

com2 = "0ho0"
init2 = com2.encode('utf_8','strict')
mot.write(init2)

#Leer una lnea de respuesta
response = mot.readline()

# Crear un archivo para guardar los datos

file_name = input("Ingrese el nombre del archivo con el que desea que se
   guarde :")
file = open(file_name + ".txt","w")

while True:
   try:
      #Ingreso del desplazamiento angular
      des = float(input("Ingrese el desplazamiento (entre 0 y 359.99).
         \nO ingrese 999 para finalizar"))
      if des == 999:
         print("Programa detenido")
         break
      if 0 <= des < 360:

         #Conversin a hexadecimal y codificacin
         dec_num = (float(des)*np.pi)/(4.38282*pow(10,-5)*180)
         hexa = decimal_a_hexadecimal(int(dec_num))
         hexa = hexa.zfill(8)

         #tipo de movimiento
         tipo = Tipo_de_movimiento(1,2)
         hexa = str(tipo + hexa)
         #print(hexa)
         str_encoded= hexa.encode('utf_8','strict')

         # Enviar una cadena de bytes
         mot.write(str_encoded)

         #Verificacin del hexadecimal unido
         #str_encoded

         # Leer una lnea de respuesta
         response = mot.readline()
         print("En hexadecimal : ", response.decode())

         # Respuesta y conversin a decimal

         prueba1 = response.decode()
         res_des = int(prueba1[3:11],16)
```

```
            #Conversin a decimal a grados de la posin del motor

            save_des = res_des*(4.38282*pow(10,-5)*180)/np.pi

            #Impresin de la posin del motor en grados
            print ("En grados : ","{0:.2f}".format(save_des))


            # Leer el segundo dato de la comunicacin serial
            dato2 = Data_prom(ardu, 10)

            #Impresion datos promedio de fotodiodo
            print ("Promedio intesidad : ", dato2)

            # Escribir los datos en el archivo
            file.write("{0},{1}\n".format(save_des, dato2))

         else:
            print("Por favor ingresa un nmero vlido")
      except ValueError:
         print("Por favor ingresa un nmero vlido")

#Cerrar Archivo

file.close()

#Graficar

Graficar(file_name)

# Mostrar el grfico
plt.show()

#Cerrar puerto
mot.close()
mot.is_open

ardu.close()
ardu.is_open

# Cargar datos CSV sin etiquetas
df = pd.read_csv('80.txt', names=['columna1', 'columna2'])

# Eliminar duplicados de x
x_unique, y_unique = np.unique(df['columna1'], return_index=False),
   df['columna2'][np.unique(df['columna1'], return_index=True)[1]]

# Interpolar los datos
x_smooth = np.linspace(x_unique[0], x_unique[-1], 200)
y_smooth = make_interp_spline(x_unique, y_unique, k=3)(x_smooth)

# Crear grfico con la curva suavizada
```

```
plt.plot(x_smooth, y_smooth, color='red', label='Curva suavizada')

#Crear grfico con la curva no suavizada
plt.plot(x_unique, y_unique, color='blue', label='Curva no suavizada')

# Mostrar los puntos originales
plt.scatter(x_unique, y_unique, color='black', label='Datos obtenidos')

#Barras de error
plt.errorbar(x_unique, y_unique, xerr=2.51E-03, yerr=0.05, fmt='o',
   color='black',
          ecolor='black', elinewidth=0.5, capsize=2)

# Aadir leyenda
plt.legend()

# Agregar etiquetas y ttulo al grfico
plt.xlabel('ngulo de incidencia  (grados)')
plt.ylabel('Intensidad')
plt.title('Curva de intensidad en funcin del ngulo de incidencia')

# Mostrar el grfico
plt.show()
```

Referencias

[1] Mochán WL. Plasmons. Reference Module in Materials Science and Materials Engineering. 2016;(July 2013):1-13.

[2] Li H, Zhang L. Photocatalytic performance of different exposed crystal facets of BiOCl. Current Opinion in Green and Sustainable Chemistry. 2017;6:48-56. Available from: http://dx.doi.org/10.1016/j.cogsc.2017.05.005.

[3] Wang D, Loo JFC, Chen J, Yam Y, Chen SC, He H, et al. Recent advances in surface plasmon resonance imaging sensors. Sensors (Switzerland). 2019;19(6).

[4] DIY LED-photometer With Arduino for Physics or Chemistry Lessons : 5 Steps (with Pictures) - Instructables;. Available from: https://www.instructables.com/DIY-LED-photometer-With-Arduino-for-Physics-or-Che/.

[5] Vázquez PME, Veiras FE, Ciocci Brazzano L, Sorichetti PA. Design method of optical detection systems based on transimpedance amplifiers. Review of Scientific Instruments. 2021;92(11). Available from: https://doi.org/10.1063/5.0054869.