Tecnológico de Monterrey

Reporte final del reto

Jonathan Uriel Anzures Garcia-A0127723

Carlo Nicolas Matamoros Méndez-A01782733

Luis Eduardo Hernandez Tellez-A01783356

Leonardo André Flores Mendoza - A01787221

08 de Junio del 2025

Modelación computacional de

sistemas electromagnéticos

Introducción.

En esta etapa final del reto se integran los conocimientos adquiridos en prácticas previas sobre campo eléctrico, movimiento de partículas en campos electromagnéticos, campo magnético de dipolos y fuerza sobre conductores con corriente. El objetivo fue modelar y analizar el comportamiento de una bobina con corriente en presencia de un campo magnético no uniforme generado por dos imanes, y determinar la fuerza resultante que actúa sobre ella.

Desarrollo matemático.

La ley principal que rige este fenómeno es la Ley de Lorentz, que establece que una carga en movimiento en presencia de un campo magnético experimenta una fuerza proporcional al producto vectorial entre su velocidad y el campo. En este contexto, la bobina se modela como un conductor con corriente, por lo que la fuerza total sobre ella se calcula como:

$$\vec{F} = I \int_{\text{bobina}} d\vec{r}_c \times \vec{B} \quad (1)$$

Donde:

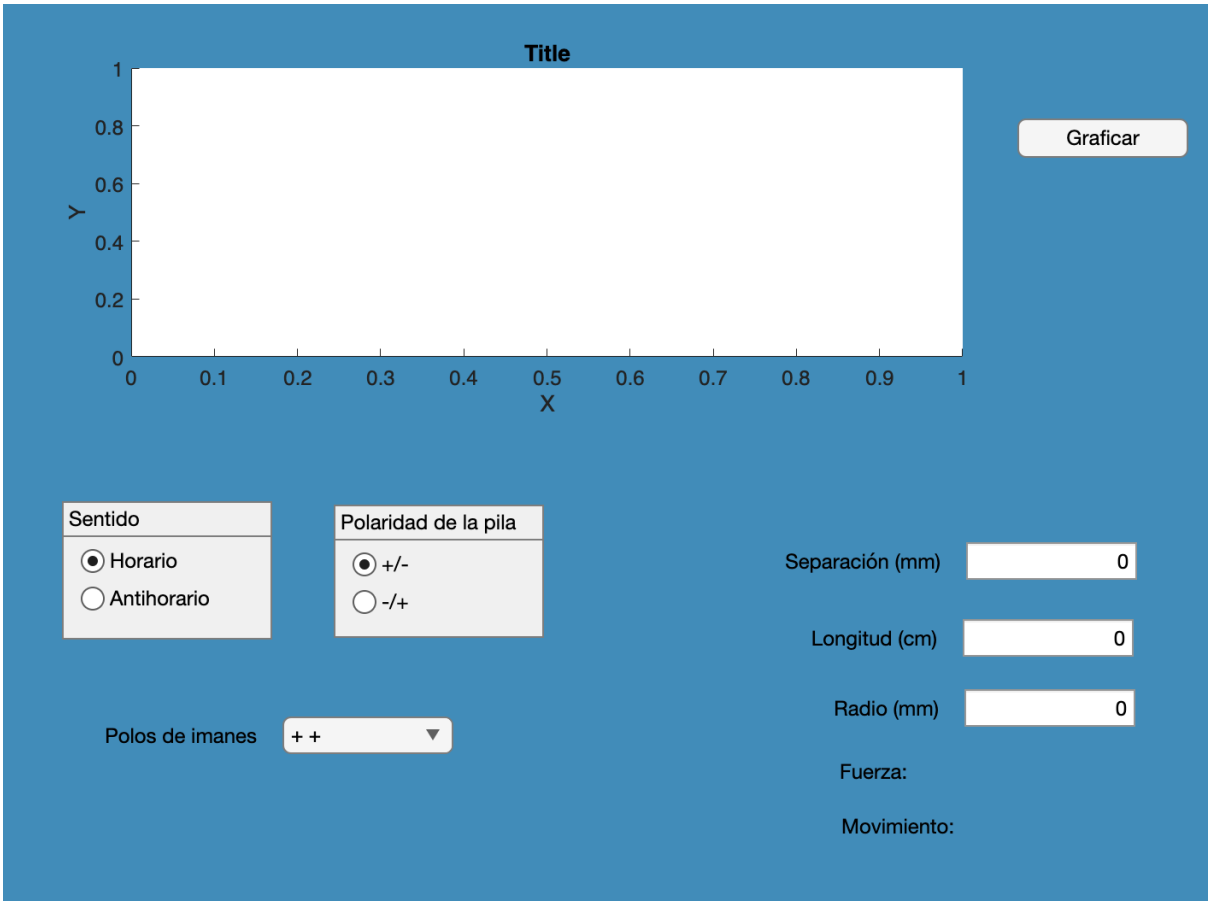
- I es la corriente que transita a lo largo de la porción de la bobina que cierra un circuito
- El campo magnético B es la suma de los campos de ambos imanes
- Y rc es la ecuación de la bobina.

El campo magnético B es obtenido mediante la siguiente ecuación:

$$\vec{B}(\vec{r})\frac{\mu_0}{4\pi r^5}[3(\vec{m}\cdot\vec{r})\vec{r}-\vec{m}r^2]$$

Donde r representa la distancia desde el imán hasta cada punto de la bobina

Mediante estos fundamentos físicos fuimos capaces de modelar una interfaz en App Designer que nos permite visualizar una simulación que refleja el objetivo de este reto:



Al iniciar la simulación podemos apreciar una interfaz sencilla de comprender y manipular, en la cual podemos escoger el sentido de nuestra bobina, así como la polaridad de la batería que estaremos utilizando en el modelo, de la misma manera podemos ingresar la separación de la bobina, la longitud, su radio y la combinación de los imanes que se utilizarán para la simulación. En la esquina inferior derecha podemos observar los Labels que reflejarán los resultados obtenidos dada la simulación.

Para que esta simulación fuera posible de ejecutarse, fue necesario llevar la siguiente estructuración del código:

```
%Constantes
mu0 = 4 * pi * 1e-7;

m = 5.31;               % Magnitud del momento dipolar
Lb = app.LongitudcmEditField.Value/100;   % Longitud de la bobina (m)
t = linspace(0,Lb,1000);
s = app.SeparacinmmEditField.Value/1000; %separacion de la bobina (m)
R = app.RadiommEditField.Value/1000; %radio de la bobina (m)
Lp = 0.042; %Largo de la pila (m)
```

Iniciamos definiendo las constantes y variables ya conocidas para la ejecución, utilizamos valores reales de nuestro modelo físico ya que queríamos que tuviera una similitud lo más cercana con la vida real

```
 %Sentido de la bobina
 horario=app.SentidoButtonGroup.SelectedObject;
if(horario==app.HorarioButton)
   k = 2 * pi ./ s;
   x_cable = t;
   y_cable = R * cos(k * t);
   z_cable= R * sin(k * t);
else
    k = -2 * pi / s;
    x_cable = t;
    y_cable = R * cos(k * t);
    z_cable = R * sin(k * t);
end

%Polaridad de Pila
pila_sent = app.PolaridaddelapilaButtonGroup.SelectedObject;
if (pila_sent == app.Button)
     I = 1;
else
     I=-1;
end

% Asignacion de signos de cada iman
 switch app.PolosdeimanesDropDown.Value
     case '+ +'
         m1 = m;
         m2 = m;
     case '- -'
         m1 = -m;
         m2 = -m;
     case '+ -'
         m1 = m;
         m2 = -m;
     case '- +'
         m1 = -m;
         m2 = m;
 end
```

Posteriormente hicimos algunas condiciones para los inputs de la interfaz y que funcionara adecuadamente escogiendo desde la app

```
%posicion de imanes
x_centro = Lb / 2;
r1 = [x_centro - Lp/2, 0, 0];
r2 = [x_centro + Lp/2, 0, 0];

%graficar la bobina y campo magnetico
xrange = linspace(0,Lb, 10);
yrange = linspace(-R, R, 10);
zrange = linspace(-R, R, 10);

[x, y, z] = meshgrid(xrange, yrange, zrange);


R1 = sqrt((x-r1(1)).^2 + (y-r1(2)).^2 + (z-r1(3)).^2);
R2 = sqrt((x-r2(1)).^2 + (y-r2(2)).^2 + (z-r2(3)).^2);

Bx1 = mu0 * (3 * m1(1) * (x - r1(1)).^2 - m1(1) * R1.^2) ./ (4 * pi * R1.^5);
By1 = mu0 * 3 * m1(1) * ((x - r1(1)).*(y - r1(2))) ./ (4 * pi * R1.^5);
Bz1 = mu0 * 3 * m1(1) * ((x - r1(1)).*(z - r1(3))) ./ (4 * pi * R1.^5);

Bx2 = mu0 * (3 * m2(1) * (x - r2(1)).^2 - m2(1) * R2.^2) ./ (4 * pi * R2.^5);
By2 = mu0 * 3 * m2(1) * ((x - r2(1)).*(y - r2(2))) ./ (4 * pi * R2.^5);
Bz2 = mu0 * 3 * m2(1) * ((x - r2(1)).*(z - r2(3))) ./ (4 * pi * R2.^5);

Bx = Bx1 + Bx2;
By = By1 + By2;
Bz = Bz1 + Bz2;

Bmag = sqrt(Bx.^2 + By.^2 + Bz.^2);

Bx = Bx./Bmag;
By = By./Bmag;
Bz = Bz./Bmag;
```

En este apartado hicimos el proceso para obtener el campo magnético mediante la fórmula del campo de un dipolo magnético en coordenadas cartesianas y poderlo graficar con la debida función, así mismo obtuvimos el posicionamiento de los imanes dentro de la simulación

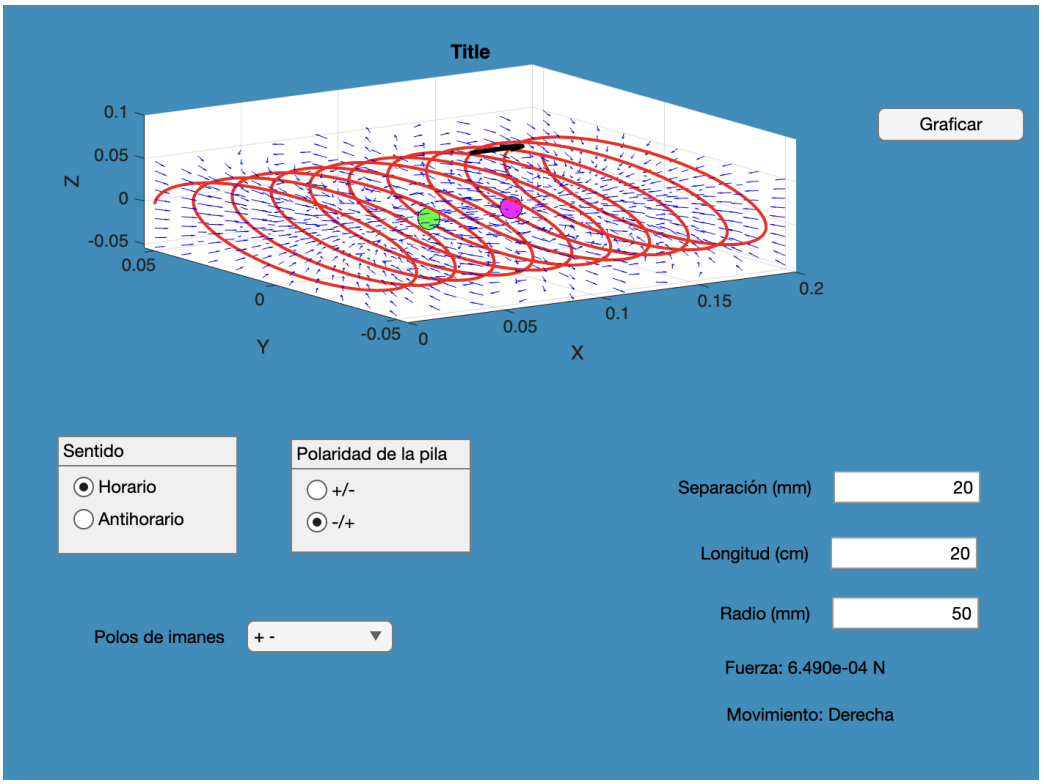
```
% Posición sobre la bobina
rc = @(t) [t; R * cos(k * t); R * sin(k * t)];
drc = @(t) [1; -R * k * sin(k * t); R * k * cos(k * t)];

B = @(r, mvec, r0) (mu0 / (4*pi)) * ((3 * dot(mvec, r - r0) * (r - r0) - mvec * norm(r - r0)^2) / norm(r - r0)^5);
Btotal = @(t) B(rc(t), [m1; 0; 0], r1') + B(rc(t), [m2; 0; 0], r2');
%Ley de Lorentz
F = I * integral(@(t) cross(drc(t), Btotal(t)), 0, Lb, 'ArrayValued', true);
```

Continuamos con la definición de la trayectoria de la bobina y su derivada, se calcula el campo magnético total generado por los dos imanes, y se aplica la Ley de Lorentz mediante una integral para obtener la fuerza neta que actúa sobre la bobina. Esta fuerza permite determinar si el tren se moverá y en qué dirección

```
app.FuerzaLabel.Text = ['Fuerza: ', num2str(F(1), '%.3e'), ' N'];
if F(1) > 1e-12
 app.MovimientoLabel.Text = 'Movimiento: Derecha';
elseif F(1) < -1e-12
 app.MovimientoLabel.Text = 'Movimiento: Izquierda';
else
 app.MovimientoLabel.Text = 'Movimiento: No se mueve';
end


 % Flecha de dirección del tren
 if abs(F(1)) > 1e-12
     x_arrow = Lb / 2;
     y_arrow = 0;
     z_arrow = R + 0.02;
     dx = sign(F(1)) * 0.03;
     dy = 0;
     dz = 0;

     hold(app.UIAxes, 'on')
     quiver3(app.UIAxes, x_arrow, y_arrow, z_arrow, dx, dy, dz,'Color', 'k', 'LineWidth', 3, 'MaxHeadSize', 2)
 end

 %Graficas

quiver3(app.UIAxes, x, y, z, Bx, By, Bz, 0.5, 'Color', 'b')
hold(app.UIAxes, 'on')
plot3(app.UIAxes, x_cable, y_cable, z_cable, 'r', 'LineWidth', 2)
plot3(app.UIAxes, r1(1), r1(2), r1(3), 'ko', 'MarkerSize', 15, 'MarkerFaceColor', 'g')
plot3(app.UIAxes, r2(1), r2(2), r2(3), 'ko', 'MarkerSize', 15, 'MarkerFaceColor', 'm')
grid(app.UIAxes, 'on')
view(app.UIAxes, 3);
```

Finalmente implementamos unas condiciones para que se vea reflejado el resultado en los labels utilizados en la interfaz, y continuamos el proceso de graficar todo lo ya descrito previamente.

Donde finalmente nuestra app se ve así con una configuración de imanes + -, con sentido horario, una polaridad de pila -/+, y una separación de 20 mm, una Longitud de 20 cm y un radio de 50 mm



Conclusión

Esta simulación nos permitió entender de manera clara y visual cómo las leyes del electromagnetismo en especial la Ley de Lorentz y el modelo del campo magnético de un dipolo, explican el movimiento de un tren impulsado por imanes. Al modelar la trayectoria de la bobina, calcular el campo magnético en cada punto y aplicar la integral de la fuerza, pudimos determinar no solo si el tren se mueve, sino también en qué dirección lo hace. Más allá de los cálculos, este reto nos acercó a la física desde una perspectiva aplicada, mostrándonos cómo conceptos abstractos toman forma en un sistema real y nos permiten explorar cómo distintas configuraciones afectan el comportamiento del sistema.