ESCUELA POLITÉCNICA NACIONAL

MÉTODOS NUMÉRICOS

CONTRERAS ANTHONY
ENRIQUEZ MICHAEL
JIMENEZ YASID

TABLA DE CONTENIDOS

ESCUELA POLITÉCNICA NACIONAL
FACULTAD DE INGENIERÍA DE SISTEMAS
INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN

REPORTE:

PROYECTO BIMESTRAL

1. Objetivos

1.1. Objetivo general

Diseñar e implementar un modelo matemático y un programa computacional para un prototipo del telescopio de James Webb.

1.2. Objetivo especifico

- Representar matemáticamente las limitaciones de los pistones del telescopio, garantizando que su longitud esté dentro del rango permitido.
- Crear un algoritmo de optimización que ajuste la posición del espejo para enfocarse correctamente en un punto objetivo.
- Estudiar cómo las variables principales afectan el desempeño del telescopio, evaluando su precisión y estabilidad.

2. Introducción

El telescopio es una herramienta esencial para la observación astronómica, y su precisión depende de la correcta alineación de sus espejos. Para este prototipo del telescopio de James Webb, la alineación se logra mediante un sistema de pistones que ajustan la posición del espejo de tal manera que la estrella a enfocar, el espejo secundario y primario se encuentren alineados. Sin embargo, este proceso está limitado por ciertas restricciones físicas, como las longitudes de los pistones y los ángulos de ajuste.

Este proyecto tiene como objetivo el diseño e implementación de un modelo matemático y computacional que permita controlar de manera eficiente y dinámica la posición del espejo en un prototipo de telescopio. A través del uso de un algoritmo de optimización, se busca garantizar que el espejo se alinee correctamente con un punto objetivo específico, respetando las limitaciones físicas de los pistones. Además, se buscará maximizar la perpendicularidad entre el espejo y el objetivo, lo que es crucial para una observación precisa.

3. Metodología

Para el desarrollo del proyecto, nuestro equipo se basó en la metodología Top-Down con la finalidad de comenzar desde lo mas sencillo hasta lo más complicado al momento de desarrollarlo, para ello se sigue lo siguiente:

3.1. Descripción del problema

Al desarrollar el prototipo del telescopio inspirado en el James Webb, el equipo de trabajo identificó varios aspectos esenciales para alcanzar los objetivos del proyecto. Primero, se comprendió la importancia de conocer el funcionamiento del telescopio, diferenciando las partes móviles de las fijas, y estableciendo las condiciones ideales para su correcto desempeño.

Se determinó que la estrella debe estar en el centro del espejo secundario, el cual es móvil, mientras que el espejo primario permanece estático. Esto llevó a la conclusión de que era necesario desarrollar un planteamiento matemático que permitiera alinear ambos elementos correctamente.

Además, se observó que, debido a la movilidad del espejo secundario, era imprescindible implementar una función que garantizara el enfoque eficiente de la estrella, cumpliendo con las restricciones físicas del telescopio. Entre estas restricciones destacan los límites del movimiento de los pistones, que poseen una longitud máxima de extensión y una longitud mínima de contracción. Estos pistones son responsables de mover el espejo secundario de manera que los tres elementos (la estrella, el espejo secundario y el espejo primario) estén alineados.

El principal desafío fue desarrollar un modelo capaz de controlar dinámicamente la posición del espejo secundario, respetando las limitaciones físicas y asegurando que la alineación sea precisa. Este modelo requiere el uso de técnicas de optimización para ajustar la posición del espejo con precisión y eficiencia, maximizando la perpendicularidad entre el objetivo y el espejo, un aspecto crítico para garantizar la calidad de las observaciones.

3.2.Desarrollo matemático

Retomando la metodología utilizada para el desarrollo del proyecto, se adoptó el enfoque Top-Down. Este método permitió abordar el problema desde lo más simple hasta lo más complejo, comenzando por establecer la condición de perpendicularidad como base fundamental y avanzando hacia la implementación del movimiento dinámico del espejo secundario para garantizar un enfoque preciso del telescopio. El proceso se desarrolló de la siguiente manera:

CONDICIÓN DE PERPENDICULARIDAD

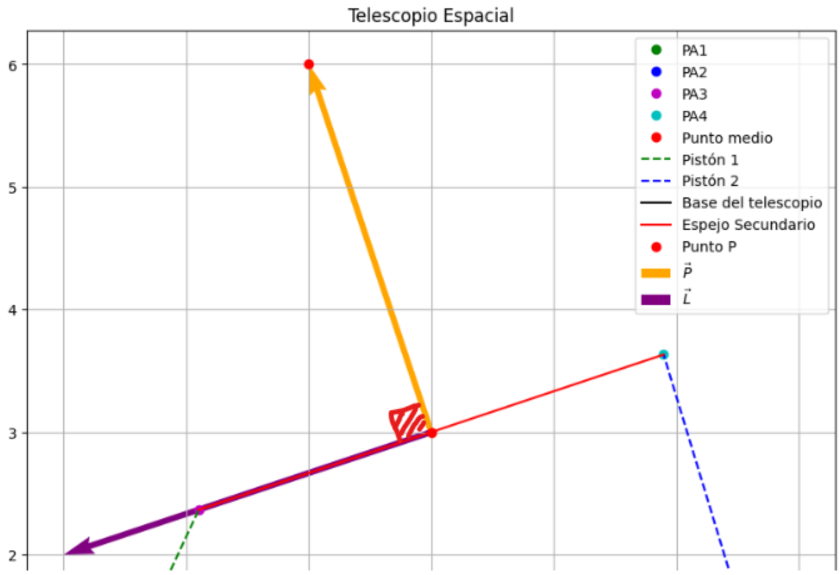


Ilustración 1 Perpendicularidad entre los vectores

Tal como se muestra en la *Ilustración 1*, una forma de garantizar que el espejo secundario se mantenga siempre perpendicular a la estrella es mediante el uso de vectores y sus propiedades. Para ello, se toman en cuenta los siguientes aspectos:

- **Estrella, P:** Se definirá a la estrella como 'P' es cual se encuentra en el plano cartesiano como $P(x_1, y_1)$
- **Punto medio del espejo secundario, L:** Llamaremos $L(x_2, y_2)$al punto medio del espejo, ubicado en una posición específica en el plano cartesiano.
- **Vector hacia el punto objetivo, $\vec{P}$:** Definimos el vector que va desde el punto medio del espejo $L(x_2, y_2)$ hacia el punto $P(x_1, y_1)$ como:

$$P = (x_1 - x_2, y_1 - y_2)$$

- **Vector normal del espejo, $\vec{L}$:** Para cumplir con la condición de perpendicularidad, necesitamos un vector normal al espejo secundario. Definimos este vector normal L como:

$$P = (-(y_1 - y_2), x_1 - x_2)$$

Cabe recalcar que utilizamos este vector para que el espejo secundario tenga la misma dirección que el vector L, para lograrlo, simplemente se debe normalizar el vector de la siguiente manera:

$$\vec{L} = \frac{\vec{L}}{\|\vec{L}\|}$$

UBICACIÓN Y FUNCIONALIDAD DE LOS PISTONES

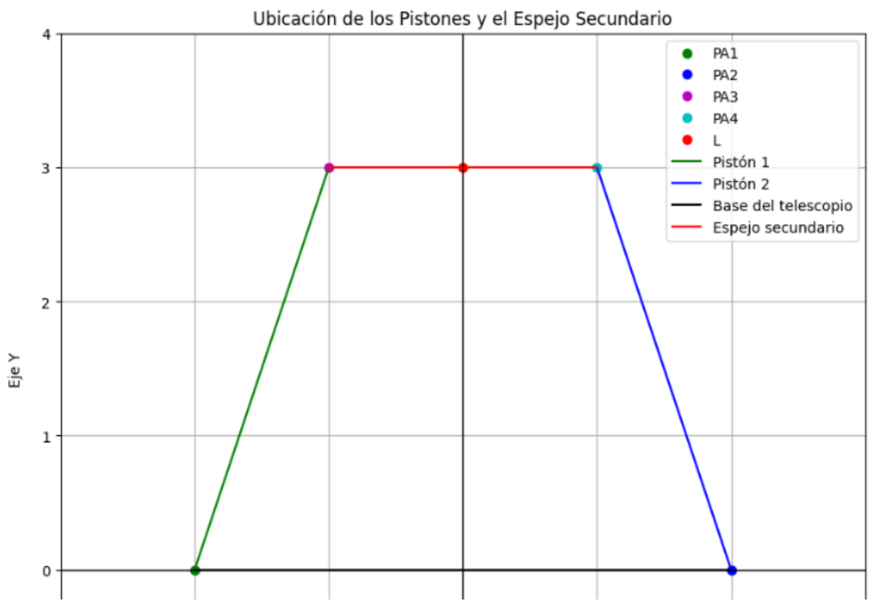


Ilustración 2 Ubicación de los pistones

Una vez creada la condición de perpendicularidad, ahora nos encargaremos de todo lo relacionado al funcionamiento y ubicación de los pistones. Por lo que se ha considerado lo siguiente:

- **Base del telescopio, B:**
  - Este tiene una longitud A [m].

- El punto medio de la base B está dado por $P(0,0)$
- El punto de anclaje izquierdo para el pistón 1 en la base está dado por

$$PA1 = (-(A/2),0)$$

- El punto de anclaje derecho para el pistón 2 en la base está dado por

$$PA2 = ((A/2),0)$$

- **Pistones, PS:**
    - Cada pistón tiene una misma longitud máxima de B_max [m].
    - Cada pistón tiene la posibilidad de contraerse B_min [m].
- **Espejo secundario, L:**
    - Este tiene una longitud C [m].
    - El punto medio del espejo secundario está dado por $L(x_2, y_2)$
    - El punto de anclaje izquierdo para el pistón 1 en el espejo secundario está dado por PA3:

$$PA3 = (x_2 - \frac{C}{2} \cdot \vec{L}_x, y_2 - \frac{C}{2} \cdot \vec{L}_y)$$

- El punto de anclaje derecho para el pistón 2 en el espejo secundario está dado por:

$$PA4 = (x_2 + \frac{C}{2} \cdot \vec{L}_x, y_2 + \frac{C}{2} \cdot \vec{L}_y)$$

Tomando en cuenta estas consideraciones, procederemos a crear una función para calcular las longitudes en los movimientos de los pistones.

RESTRICCIONES FÍSICAS

```
La longitud del Pistón_1 = 5.10 [m] y la longitud del Pistón_2 = 1.41 [m] están en el rango permitido [5.0, 10]
Y el angulo 108.43° esta dentro del rango permitido [0° a 180°].
```

Ilustración 3 Resultados de la ejecución

Como se observa en la *Ilustración 3*, el programa proporciona las longitudes de los pistones y el ángulo que forma el vector P con respecto al eje x. Esto se realiza con el objetivo de garantizar el cumplimiento de las restricciones físicas de los pistones, así como de asegurar la funcionalidad del programa. Para ello se sigue:

Los pistones se extienden entre los puntos de anclaje (PA1, PA2) y el espejo (PA3, PA4). La longitud de un pistón es simplemente la distancia euclidiana entre estos dos puntos.

Longitud de un pistón:

La restricción para la longitud de los pistones consiste, esencialmente, en que su longitud se mantenga dentro de un rango definido

$$(B_{min} \leq L_i \leq B_{max})$$

Este rango garantiza que los pistones operen de manera segura y eficiente, respetando los límites físicos establecidos.

Para el Pistón 1, que va desde el PA1 a PA3:

$$PistonUno = L_1 = \sqrt{(x_{PA3} - x_{PA1})^2 + (y_{PA3} - y_{PA1})^2}$$

Para el Pistón 2, que va desde el PA2 a PA4:

$$PistonDos = L_2 = \sqrt{(x_{PA4} - x_{PA2})^2 + (y_{PA4} - y_{PA2})^2}$$

Cálculo del ángulo de P con respecto al eje x

El ángulo de este vector con respecto al eje x se puede calcular usando la función arctangente de los componentes ‘y’ y ‘x’ del vector P.

$$\theta_2 = arctan2(y_1 - y_2, x_1 - x_2)$$

OPTIMIZACIÓN DINÁMICA

El algoritmo de optimización ajusta las coordenadas $L(x_2, y_2)$ del espejo para minimizar una función de costo que combina tres aspectos clave:

- *Perpendicularidad:* Se maximiza la perpendicularidad del espejo al vector $\vec{\boldsymbol{P}}$ minimizando el producto punto $\vec{\boldsymbol{P}} \cdot \vec{\boldsymbol{L}}$
- *Restricciones de longitud:* Penaliza las configuraciones en las que los pistones exceden su rango permitido.
- *Restricciones angulares:* Asegura que los pistones formen ángulos válidos con respecto a la base.

Explicación de la función costo

En el código, la función 'costo' implementa esta función matemática y se usa la librería 'scipy.optimize.minimize' para encontrar la mejor posición $L(x_2, y_2)$ que cumple con las restricciones. Los pasos son:

a) *Se inicializa $L(x_2, y_2)$ en una posición aproximada:*
   - Se define una posición inicial para el punto medio del espejo (x_2, y_2), ubicada razonablemente cerca del punto objetivo para facilitar la optimización.
   - Se establecen límites de valores para (x_2, y_2) que corresponden al espacio físico permitido por la estructura del telescopio. Estos límites aseguran que el espejo no salga de su área de operación:

     Para x_2: límites horizontales que evitan que el espejo exceda el ancho del sistema.

     Para y_2: límites verticales que aseguran que el espejo permanezca dentro del rango de los pistones.

   - Ejemplo de límites usados en el código:

$$x_2 \in [-15,15] \qquad\qquad y_2 \in [0,10]$$

b) Se calcula el costo para cada posición evaluada, considerando la perpendicularidad, las longitudes de los pistones y las restricciones angulares.

c) La optimización ajusta iterativamente (x_2, y_2) hasta encontrar una configuración que minimice el costo.

3.3.Diagrama de flujo

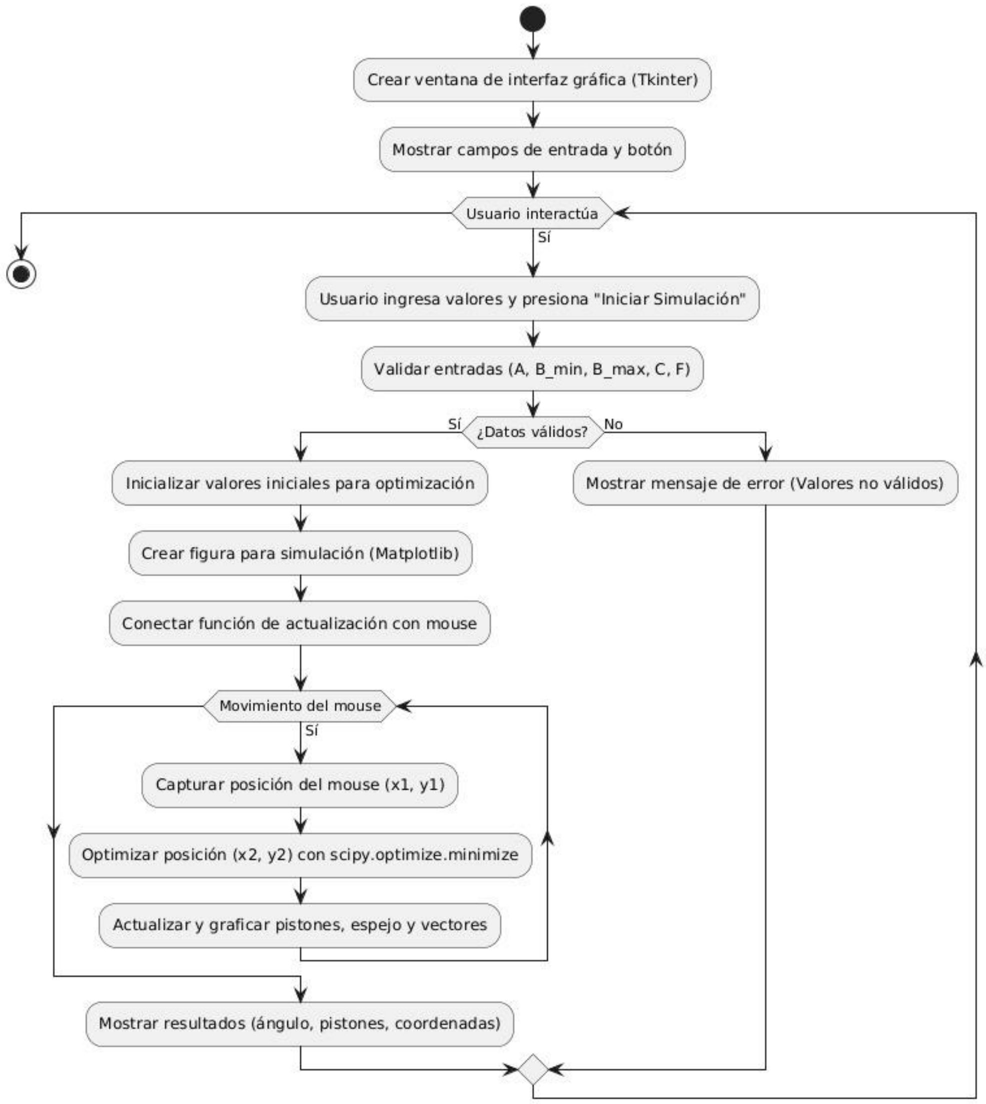


Ilustración 4 Diagrama de flujo del programa

3.4. Detalles de la implementación

Dado que los detalles de implementación constituyen una descripción técnica del programa desarrollado, se abordarán desde el lenguaje y las herramientas utilizadas hasta las limitaciones que presenta el prototipo del telescopio espacial.

LENGUAJES Y HERRAMIENTAS USADAS

Para el desarrollo del proyecto se utilizó el lenguaje de programación Python, seleccionado por su simplicidad en la programación y su amplia disponibilidad de librerías de fácil comprensión. Además, se empleó Visual Studio Code como entorno de desarrollo integrado debido a su interfaz amigable, especialmente adecuada para estudiantes.

Las principales librerías utilizadas fueron:

- **Tkinter**: Para construir la interfaz gráfica y permitir al usuario ingresar parámetros personalizados.
- **Scipy**: Para implementar un algoritmo de optimización basado en el método minimize.
- **Numpy**: Para realizar cálculos vectoriales y matemáticos fundamentales.
- **Matplotlib**: Para generar representaciones gráficas dinámicas del sistema del telescopio.

ESTRUCTURA DEL PROGRAMA

Para facilitar la comprensión de nuestro programa, se decidió crear una tabla que enumera todas las funciones utilizadas, junto con una breve descripción de su propósito y funcionamiento. Esto permite una referencia rápida y clara de las funcionalidades implementadas en el proyecto.

Función	**Descripción**
`costo(pos, A, B_min, B_max, C, x1, y1, F)`	Esta función calcula el costo de una configuración de pistones para ajustar la posición del espejo secundario. El costo tiene en cuenta varios factores: la longitud de los pistones, el ángulo del espejo, y la alineación con el foco del espejo primario.
`telescopio(A, B_min, B_max, C, F)`	Esta función genera la visualización interactiva del telescopio y actualiza la posición del espejo secundario en tiempo real en función de la posición de la estrella, que el usuario mueve con el mouse. Utiliza la función minimize para la optimización.

`update_plot(event)`	Se ejecuta cada vez que el usuario mueve el mouse. Calcula la nueva posición de la estrella a partir de las coordenadas del mouse y optimiza la configuración de los pistones para ajustar el espejo. Actualiza la visualización en tiempo real.
`crear_interfaz()`	Crea la interfaz gráfica de usuario utilizando la librería Tkinter. Permite al usuario ingresar valores para la base del telescopio, longitudes de los pistones, el espejo, y el foco. Al hacer clic en "Iniciar Simulación", la simulación se ejecuta.
`on_submit()`	Función que se ejecuta cuando el usuario presiona el botón "Iniciar Simulación". Recoge los valores ingresados por el usuario en la interfaz y valida que sean correctos antes de llamar a la función telescopio para ejecutar la simulación.

ASPECTOS TECNICOS RELEVANTES

En el diseño del sistema de optimización y simulación del telescopio, se implementaron varias restricciones y validaciones clave para asegurar que las configuraciones generadas fueran físicamente viables y que el programa operara dentro de parámetros razonables. A continuación se describen las principales validaciones:

- **Longitudes fuera del rango:** Si las longitudes de los pistones no están dentro de [bmin,bmax], el programa no podrá continuar.
- **Posición fuera de los límites físicos:** Las coordenadas (x2,y2 o las de los puntos de anclaje deben estar dentro de los rangos físicos de la estructura del telescopio.
- **Falta de perpendicularidad:** Si el producto punto entre P y L no es cercano a cero, los vectores no son perpendiculares.

- **Ángulos fuera de rango:** Los ángulos de los pistones deben estar entre 0° y 180°.
- **Condiciones de entrada no válidas:** Valores de entrada fuera de los rangos esperados para las coordenadas y longitudes.

LIMITACIONES

Una de las principales limitaciones del sistema es cuando los valores de las coordenadas o las posiciones de los componentes del telescopio están demasiado cerca unos de otros o muy cerca de la estructura del telescopio. Esto puede causar problemas como:

1. **Pérdida de precisión**: Los cálculos pueden no ser precisos cuando las posiciones son muy cercanas, lo que genera errores pequeños pero importantes.

2. **Errores en ángulos y distancias**: Si las distancias entre los puntos se hacen muy pequeñas, los cálculos de ángulos o alineación pueden fallar.

3. **Inestabilidad en la optimización**: La optimización puede volverse inestable, y el sistema podría no encontrar una solución correcta si los puntos están demasiado cerca.

Para evitar estos problemas, se deben establecer límites que aseguren que los valores no sean demasiado pequeños o cercanos a la estructura del telescopio.

4. Resultados

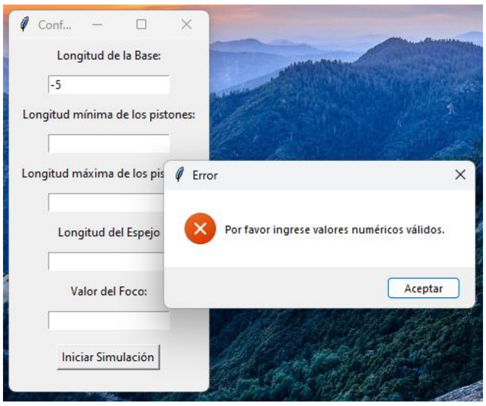
En esta sección, se presenta la ejecución del programa, especificando los valores que pueden tomar los parámetros de entrada y cómo estos afectan la representación gráfica del sistema. Se explica cómo el usuario puede interactuar con el programa, ingresar diferentes valores y observar los resultados visualmente, destacando cómo cada parámetro influye en la simulación del telescopio.

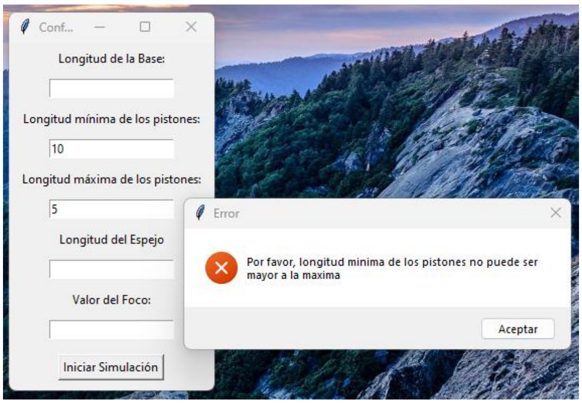
CONTROL DE ENTRADA

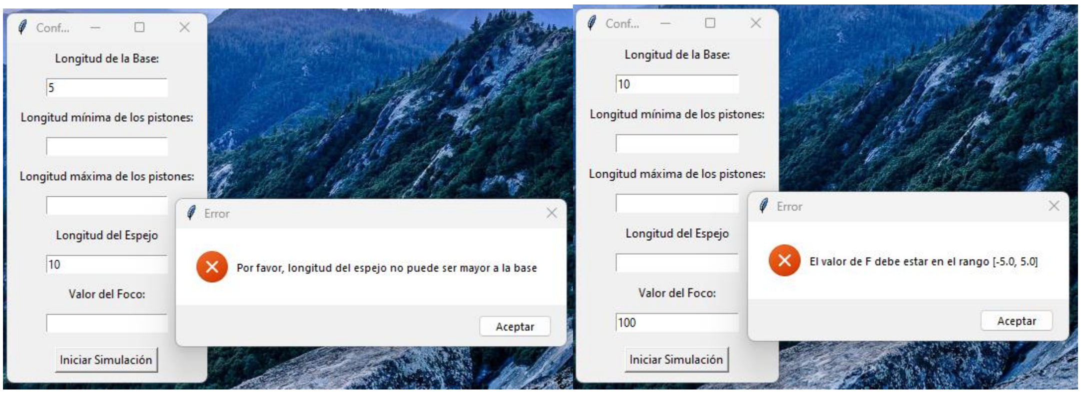
El control de los valores de entrada es esencial para asegurar un funcionamiento correcto del programa. Se implementan verificaciones para evitar errores, tales como:

- **Numeración negativa en las longitudes**: Se valida que las longitudes no sean negativas, ya que esto no tiene sentido en el contexto físico del telescopio.
- **Entrada de caracteres no numéricos**: Se asegura que los parámetros ingresados sean números, evitando el uso de letras u otros caracteres no válidos.
- **Validación de las longitudes de los pistones**: Se verifica que la longitud mínima del pistón no sea mayor que la longitud máxima, garantizando que las restricciones físicas se respeten.
- **Longitud del espejo secundario**: Se comprueba que la longitud del espejo secundario sea mayor que la base para asegurar que pueda moverse adecuadamente.
- **Posición del espejo primario**: Se valida que la coordenada del espejo primario se encuentre fuera de la base, lo que es necesario para mantener la alineación adecuada.

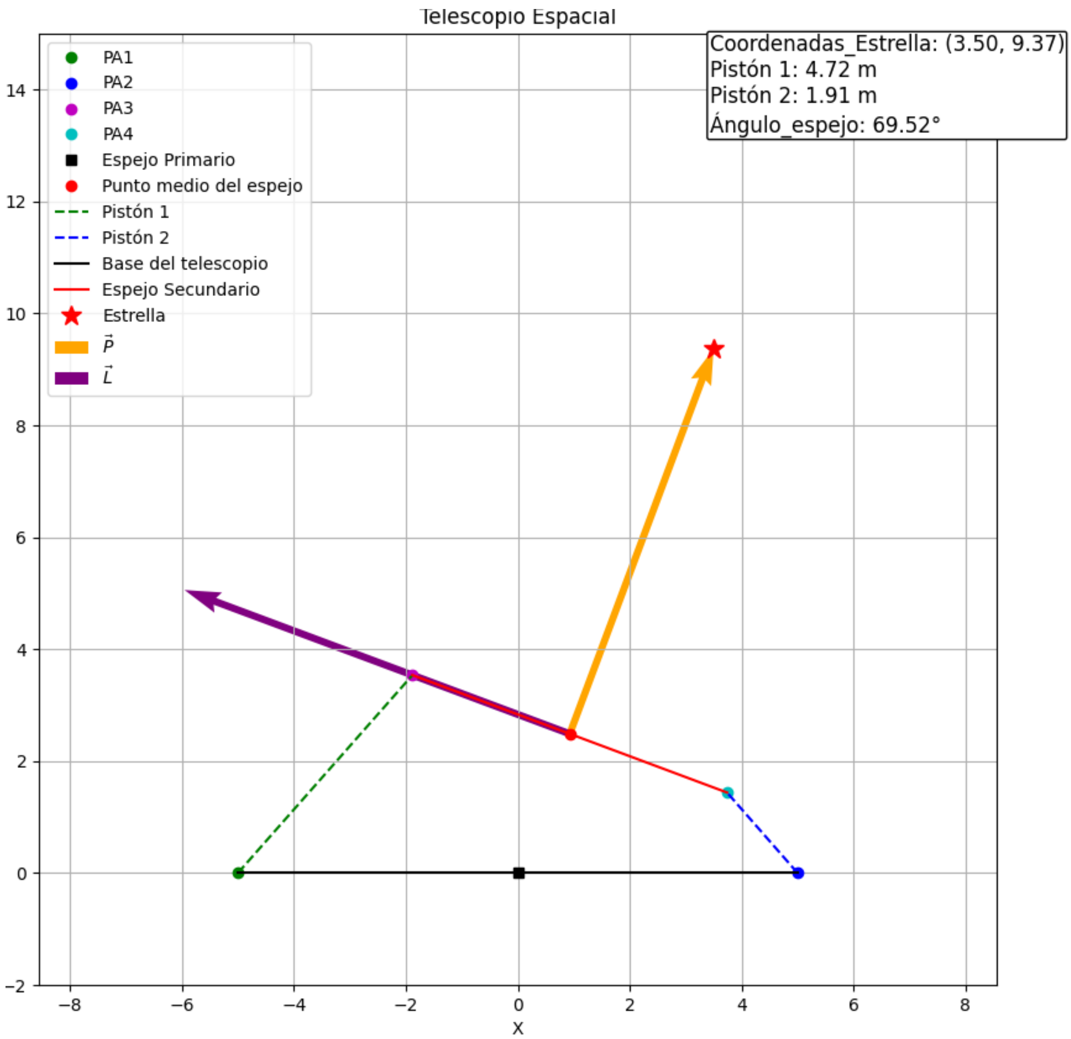
Este control garantiza que todos los parámetros de entrada sean coherentes y que el telescopio funcione de manera eficiente y segura.







CORRECTA EJECUCIÓN



5. Conclusión

La optimización del punto medio del espejo secundario resultó ser un desafío complejo, ya que requiere ajustar con precisión los componentes del telescopio para mejorar la calidad de la imagen. La inclusión de un nuevo parámetro en el espejo primario complicó aún más el proyecto, aumentando los cálculos y ajustes necesarios. Sin embargo, este desafío me permitió aprender nuevas herramientas como Tkinter para crear interfaces y mejorar habilidades en programación matemática, lo que fue fundamental para realizar las simulaciones y cálculos del diseño. En general, el proyecto nos enseñó la importancia de la matemática en la resolución de problemas complejos y cómo la programación y las librerías adecuadas son esenciales para llevar a cabo un proyecto técnico de este tipo.