Universidade Vigo

Estudio y prueba de algoritmos PAT para un sistema de enlace óptico en espacio libre

Alejandro Comesaña Almuiña

Trabajo de Fin de Grado
Escola de Enxeñaría de Telecomunicación
Grado en Ingeniería de Tecnologías de Telecomunicación

Tutores
Fernando Antonio Aguado Agelet
Alejandro Camanzo Mariño

Curso 2023/2024

Índice

1. Introducción **3**

2. Problemas en las comunicaciones por satélite **3**
2.1. Cuestión económica 4

3. Objetivos **4**

4. Algoritmos PAT **5**
4.1. PAT grueso 5
4.2. PAT fino 8
4.3. Comunicación de datos 9

5. Estado del arte **10**

6. Metodología **10**
6.1. Desarrollo de la interfaz 10
6.2. AD56X4 12
6.3. Conexión externa con la lente 15

7. Pruebas y resultados **15**

8. Conclusiones y líneas futuras **16**

Referencias **17**

A.Interfaz **19**

1. Introducción

Los satélites artificiales y las sondas de exploración surgieron de la necesidad de establecer enlaces fiables de amplia cobertura, con los cuales poder recibir y enviar señales en cualquier punto de una amplia superficie geográfica . El poder emitir una señal y recibirla con una intensidad similar simultáneamente en cualquier punto de una gran superficie geográfica es su característica más importante.

Este tipo de comunicaciones recibe el nombre de comunicaciones ópticas en espacio libre (FSO) , como su propio nombre indica es una tecnología de comunicación óptica. Esta técnica utiliza la luz, bien sea visible o infrarroja,para transmitir la información.

La comunicación óptica en espacio libre es capaz de proporcionar datos de alta velocidad a larga distancia de forma muy eficiente, sin agotar las bandas de radio frecuencia[1]. Esto es gracias a que la luz infrarroja utilizada tiene una frecuencia mucho mayor que las ondas de radio , lo que permite enviar más datos en cada transmisión. Con lo cual, los FSO es una tecnología más eficiente en cuanto tamaño y potencia que los enlaces de radiofrecuencia.

2. Problemas en las comunicaciones por satélite

Los plataformas satelitales pueden tener diversas alturas e inclinaciones sobre la superficie terrestre, pero suelen ser diseñadas para operar , como mínimo, a 160 km de altura.

Sin embargo, surge un problema a estas alturas, los algoritmos que conforman el sistema de apuntamiento, adquisición y seguimiento (PAT), son difíciles de realizar debido a los movimientos y vibraciones de las plataformas[3] , a la estrechez del haz de transmisión y a los errores de apuntamiento que proceden , principalmente,de las turbulencias atmosféricas[4] lo que provoca un giro de la polarización de las ondas originado por la deformación de las gotas de lluvia. En la Figura 1 puede verse una gráfica de como se atenúa una señal en función de su frecuencia y de la intensidad de la lluvia .Otros fenómenos climáticos como las tormentas de polvo también son fuentes de errores[5].

Además de esto hay otros problemas relacionados con la atmósfera, que se presentan incluso en condiciones de cielo descubierto (sin precipitaciones o polvo)que provocan divergencias en el haz, que afecta especialmente a a sistemas que trabajan por encima de los 10GHz [7]. La parte baja de la atmósfera produce absorción de las ondas de radio debidos a los gases atmosféricos [8], este fenómeno no tiene un valor constante, pero está siempre presente y se produce por la presencia de oxígeno, vapor de agua y otras moléculas. Otro fenómeno atmosférico es el de la refracción, que se produce por la variación de la densidad de la atmósfera con su altura y por las discontinuidades en ella, esto tiene como resultado pequeños desvíos en la trayectoria de la onda que pueden provocar multitrayecto.

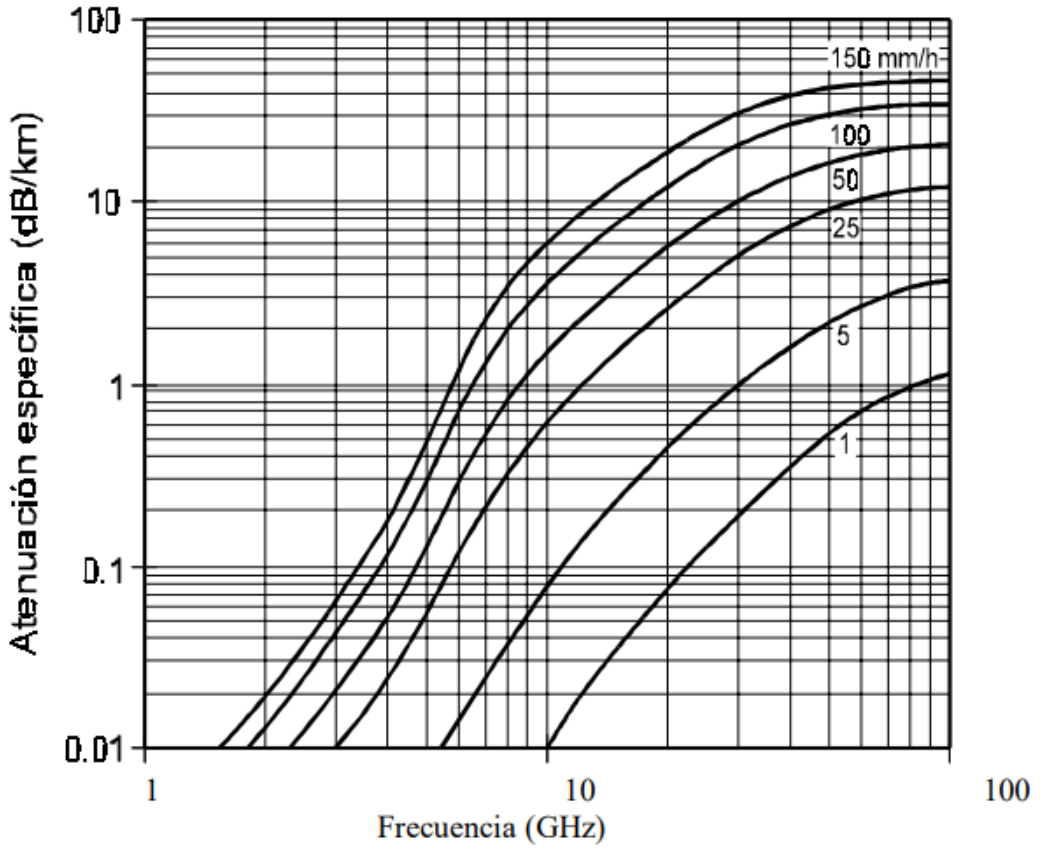

Figura 1: Atenuación en función de la frecuencia e intensidad de lluvia.Fuente [6]

El entorno espacial también influye negativamente, la radiación , la contaminación y los cambios de temperatura causan cambios en las propiedades ópticas de los dispositivos, como variaciones en su índice de reflexión.

2.1. Cuestión económica

A mayores de los errores de apuntamiento,las misiones espaciales tienen que lidiar con problemas relacionados con cuestiones económicas. Estas misiones suelen ser muy costosas[9] y los componentes y tecnologías empleadas tienen pocas posibilidades de reparación una vez salen de la estación terrestre, por este motivo se necesita que los componentes sean confiables y robustos. Por estos motivos, es necesario reducir al mínimo las posibilidades de error en tipo de misiones.

3. Objetivos

Como hemos visto, es de vital importancia mejorar el rendimiento de los sistemas aéreos bajo los efectos de los errores de apuntamiento, este es el propósito principal de este

proyecto, así, como el de entender el funcionamiento y los posible fallos de los algoritmos PAT, para así, poder corregir sus errores y mejorar su rendimiento.

En primer lugar, vamos a realizar un estudio de los algoritmos PAT más comunes hoy en día , para posteriormente crear un programa que simule los múltiples errores de puntería que pueden sufrir los distintos algoritmos que se usan tanto en el proceso de PAT grueso como fino, es decir, vamos a crear programa que moviendo un láser simule los errores mas típicos que sufren los sistemas aéreos. De esta forma , asegurarnos que estas perturbaciones pueden ser corregidas de forma satisfactoria por los componentes del sistema que conforma el algoritmo PAT.

4. Algoritmos PAT

En los puntos anteriores ya se ha definido lo que son los algoritmo PAT, en este punto vamos a hacer hincapié en su funcionamiento y diseño al ser este un punto fundamental para este proyecto.

Los algoritmos PAT suelen consistir de dos partes:

1. PAT grueso: para lograr un alineación aproximada entre el transmisor y el receptor.
2. PAT fino: el transmisor usa un rayo láser con un ángulo de divergencia estrecho y mejora la alineación antes de empezar a enviar los datos del receptor.

A continuación se va a detallar los puntos importantes de este proceso PAT.

4.1. PAT grueso

Iniciamos con el proceso de PAT grueso,para tener un enlace preciso entre la estación terrestre y el satélite hay que proporcionar alguna referencia de apuntamiento . Una vez tenemos una referencia acerca de la posición aproximada del receptor, el primer paso es escanear el área en la que se espera encontrar el receptor, usando una señal láser de baliza con una divergencia de haz que permita que la potencia transmitida busque dentro del área mencionada. A la hora de elegir la longitud de onda del láser hay que tener en cuenta la ganancia de la antena frente a este ancho de haz. La ganancia de una antena óptica viene dada por :

$$G = \frac{\pi D^2}{\lambda} \tag{1}$$

por lo tanto, la ganancia es inversamente proporcional a la longitud de onda, entonces, lo ideal sería usar longitudes de onda bajas para así tener una alta ganancia, pero surge un problema, el ancho de haz del sistema es proporcional a λ, es decir, si la longitud de onda es muy baja también lo será el ancho de haz, lo que incrementará los errores de apuntamiento. Entonces, se necesita una buena relación entre la ganancia y el ancho de haz. Un valor típico para esta longitud de onda es el de 1550 nm ya que, para la elección

de la longitud de onda hay que tener en cuenta también el fenómeno de la absorción y la dispersión atmosféricas que depende de la longitud de onda. El haz que atraviesa la atmósfera puede sufrir de estas degradaciones si la longitud de onda está dentro de la banda de absorción principal. La longitud de 1550 nm no sufre grandes degradaciones debido a estos fenómenos ni tampoco debido a la dispersión Rayleigh que ocurre cuando las partículas de las ondas se dispersan, al interactuar con partículas con una longitud de onda mucho menor, en la Figura 2 se puede ver una gráfica que muestra como afecta la reflectancia causada por Rayleigh en función de distintas longitudes de onda, donde se aprecia que para el valor mencionado la reflectancia es muy baja , por lo tanto las pérdidas de la señal se verán reducidas. Esta longitud de onda también proporciona una alta potencia de transmisión, lo que permite transportar mas información, y es una longitud de onda segura para los ojos humanos lo que la hace una elección bastante adecuada.

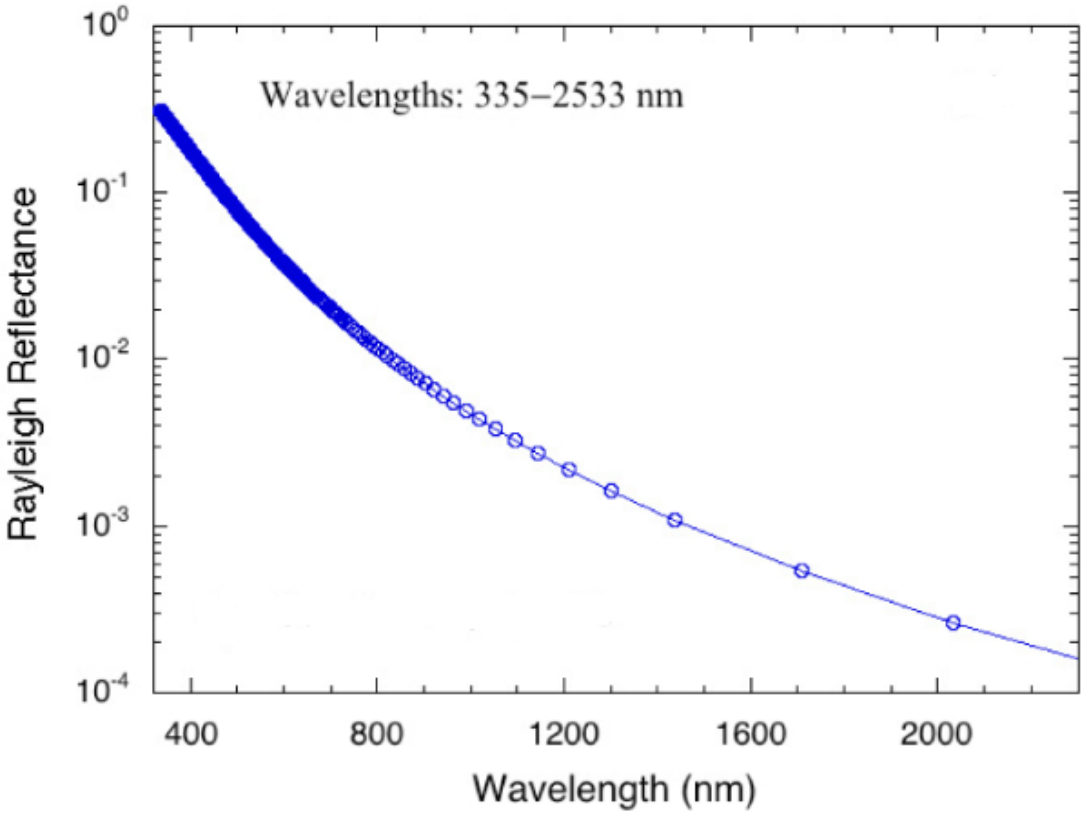

Figura 2: Gráfica de la reflectancia causada por la dispersión Rayleigh. Fuente [10]

La búsqueda dentro de la región a escanear se realiza con el espejo de dirección rápida (FSM) del transmisor. Para buscar dentro de este área hay que hacer un barrido de la región, existen varias formas, pero vamos a destacar las dos que son más utilizadas:

1. Escaneo continuo en espiral: se realiza un barrido en espiral del área, que en coor-

denadas polares se puede describir como:

$$r_{\mathrm{s}} = \frac{L_\theta}{2\pi}\theta_{\mathrm{s}} \tag{2}$$

Donde L_θ representa la longitud del paso del ángulo de divergencia del haz de la baliza.

2. Escaneo en espiral escalonada: es una variante del anterior método, en el que el haz gira en curvas cerradas a medida que explora la región de incertidumbre.

En la Figura 3 podemos ver un esquema típico en el que un transmisor hace un barrido en el área de incertidumbre del receptor,donde se ven las dos técnicas de escaneo mencionadas. En ambos métodos, la longitud del paso depende del ángulo de divergencia del haz y, al aumentarlo, reduciría el tiempo de adquisición, pero, también necesitaríamos una mayor potencia de transmisión, lo que incrementa los costos.

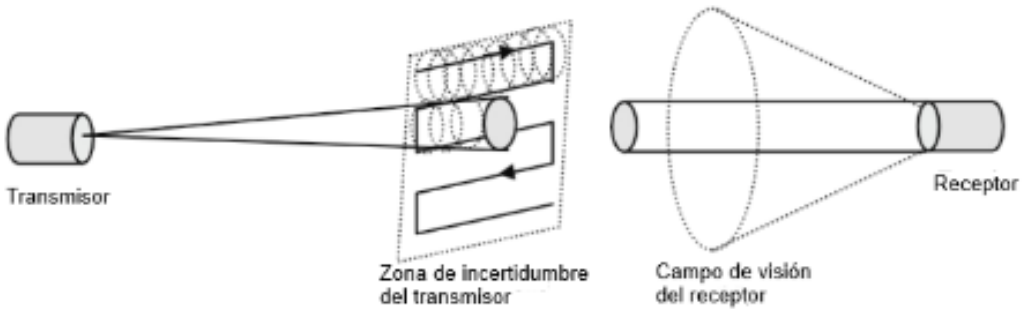

Figura 3: Ejemplo de técnica de adquisición entren un transmisor y un receptor

El receptor, por su parte, escanea su campo de visión para encontrar la señal de baliza del transmisor. Este campo de visión debe ser elegido con cuidado , un campo demasiado grande provocará un ruido de fondo muy elevado , sin embargo, un campo muy pequeño hará que los patrones de búsqueda sean muy lentos. El receptor , consta de un telescopio receptor, un filtro (para reducir el ruido de fondo) y de un fotodetector entre otros componentes. El telescopio consiste en una serie de lentes que enfocan la señal óptica hacia el fotodetector que convierte la señal óptica en una señal eléctrica que pasa a la unidad de procesamiento y luego al demodulador. A mayores en el receptor hay un detector de cuadrantes, que se encarga de medir la suma de la potencia óptica en cada cuadrante del fotodetector, así como de la potencia total detectada por el sensor de la posición del haz, estos datos son enviados de vuelta al transmisor por el canal de realimentación. La potencia óptica de cada cuadrante es enviada al controlador de alineación del ángulo de llegada, este controlador ajusta los ejes del FSM, de forma que intenta maximizar la potencia que es recibida por los fotodetectores.

Toda esta información se envía de vuelta al transmisor por el canal de realimentación. El

controlador PAT registra estos datos de realimentación durante el proceso de exploración. Así, el controlador tiene una serie de datos , compuesto por las intensidades de la señal en el receptor, a partir de esto, se puede determinar la posición aproximada del receptor, buscando la zona de intensidad máxima.

La intensidad depende de la sensibilidad de la lente del receptor, que depende de la técnica usada por el receptor óptico para detectar los fotones y del ruido. Para detectar los fotones se puede hacer una detección coherente, donde se mezcla señal entrante con una señal más fuerte generada en un oscilador local, esta mezcla de señales da como resultado una amplificación lineal y la conversión de señal óptica en eléctrica. En cuanto al ruido, hay distintas fuentes de ruido que afectan: el ruido de fondo, que es debido al entorno que rodea al fotodetector, el ruido de corriente oscura, que es una pequeña corriente que circula por el dispositivo cuando no recibe ninguna potencia y el ruido térmico que depende la interacción térmica de los electrones.

La sensibilidad se mide en términos de la media de fotones que recibe por bit, lo cual puede verse como:

$$n = \frac{P_{\text{av}}}{hvR} \tag{3}$$

Donde hv es la energía fotónica de la longitud de onda y R la velocidad binaria de los datos.

4.2. PAT fino

Este proceso es similar al PAT grueso, su diferencia principal radica en el FSM. En este punto la carga útil comenzará a tomar importancia (la carga útil son los componentes que cumplen el propósito principal del satélite,por ejemplo, en un satélite de comunicaciones serían los transpondedores para emitir señales de TV), este componente se divide en dos subsistemas; la óptica del receptor de baliza y la óptica del transmisor.

La carga útil comenzará a transmitir y usará el FSM para alinear la señal del enlace descendente con la baliza del enlace ascendente. El FSM se orienta a partir de las imágenes de una cámara de seguimiento de la baliza, que se procesan para determinar el ajuste de corrección a partir de la diferencia entre un punto del conjunto del plano focal de la cámara generado por el láser de la baliza y un punto del láser de transmisión de la carga útil. Por tanto, dirigiendo el FSM, se pueden alinear los puntos del láser de calibración y de la señal de la baliza, alineando así la señal de transmisión de enlace descendente con la baliza.

El algoritmo PAT no termina una vez hemos establecido el enlace con el sistema, después de esto debemos realizar constantemente el proceso de PAT fino para mantener la alineación entre el emisor y el receptor, ya que, una vez conocemos la posición final hay que conservar su trayectoria dentro de unos estrechos límites, normalmente, las posiciones angulares de los satélites no deben sufrir variaciones mayores a los rangos de $\pm$ 0.5 a $\pm$ 1 grados [11].Estos límites podrían ser sobrepasados debido a movimientos

aleatorios de la huella del haz en el plano de apertura del receptor, lo que disminuye el rendimiento de los sistemas, al introducir atenuaciones en la señal detectada, así como a las irregularidades de la esfera terrestre,además de los distintos tipos de errores ya mencionados.En la Figura 4 podemos ver un esquema típico de estos algoritmos, donde el controlador PAT es el encargado de controlar el FSM, dependiendo de los valores de potencia detectados en el sensor de la posición del haz.

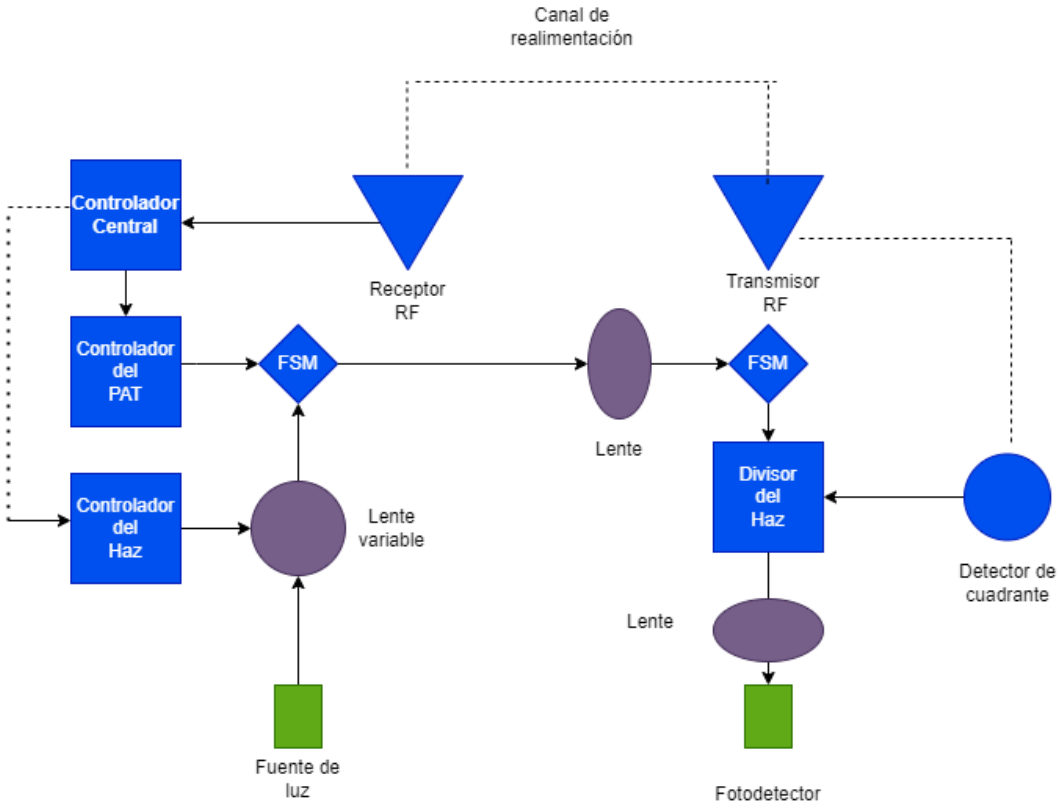

Figura 4: Diagrama de bloques de un algoritmo PAT

4.3. Comunicación de datos

Una vez que se ha establecido el enlace entre las dos estaciones puede iniciarse la comunicación de datos. Para esta transferencia de datos hay varios esquemas de modulación válidos, pero los más populares son los basados en métodos binarios, como la modulación de amplitud en cuadratura (QAM), en los que la información se transmite en cada período de símbolo mediante la variación de dos niveles de intensidad. A mayores, hacen falta códigos de corrección de errores muy potentes, ya que debido a todos los problemas ya expuestos la atenuación de las señales es muy elevada y por ello necesitamos ganancias de codificación muy elevadas.

5. Estado del arte

Las comunicaciones por satélite son una tecnología en continuo desarrollo gracias a los distintos avances tecnológicos, motivados por la urgente necesidad de extender las comunicaciones a zonas remotas o aisladas, como las propuestas llevadas por empresas como OneWeb [12], con su constelación de satélites, y Space X[13], con el Jupiter-3 (entre más misiones que lideran), que han creado distintas misiones que intentan mejorar la red de comunicaciones global. Estas misiones son muy necesarias para acabar con la desconexión digital, ya que incluso en España sigue habiendo muchos pueblos aislados en cuanto a su forma de comunicarse con el exterior. Además, el ejército estadounidense necesita una red de comunicaciones espaciales de alta velocidad para su red multicapa de estaciones[14], esta comunicación debe ser capaz de transferir grandes cantidades de información de una forma fiable y veloz.

En el contexto del estudio de nuestro proyecto, podríamos destacar por su gran importancia y la empresa que la respalda a la misión *CubeSat Laser Infrared CrosslinK*[15] [16] llevada a cabo por la NASA , este satelite puede apreciarse en la Figura 5.En esta misión, en el contexto de los algoritmos PAT, se realizan una serie de pruebas para la caracterización del ruido en el fotodiodo del proceso del PAT fino. Otras misión interesant de mencionar es *Deep Space Optical Communications (DSOC)* de la NASA, que trata de mejorar los enlaces descentes de datos y los sistemas de calibración de los satelites[17].

La existencia de este tipo de misiones es un buen indicativo de la necesidad de investigación y de proporcionar soluciones a los problemas mencionados que sufren las comunicaciones por satélites.

6. Metodología

En este apartado vamos a detallar los pasos a seguir para realizar los programas necesarios para poder generar y mostrar unas turbulencias similares a las que sufriría una plataforma satelital. Para ello se va a dividir el trabajo en tres parte , la primera será crear un programa que simule los errores ya mencionados. La segunda parte será otro programa que sea capaz de convertir dichos errores en coordenadas válidas para mover un láser que simule la plataforma satelital y así poder visualizar de forma más sencilla estos errores . Por último, la tercera parte, estará compuesto por el hardware necesario para conectar dichos programas con la láser y poder visualizar sus movimientos.

6.1. Desarrollo de la interfaz

Para el desarrollo de la interfaz vamos a usar el entorno de desarrollo de Arduino IDE para el desarrollo del código, este entorno es adecuado para la realización de este proyecto, puesto que , proporciona una versión simplificada de C++, lo que ayuda con las labores

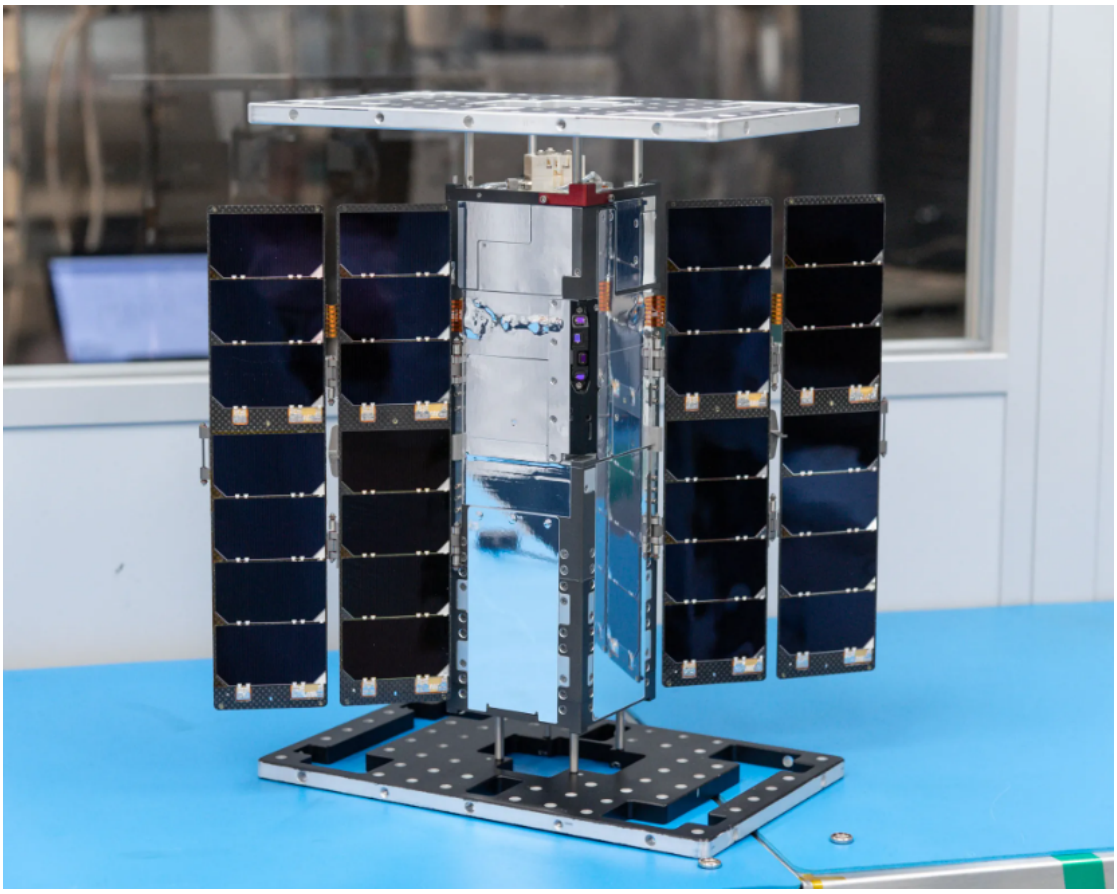

Figura 5: Fotografía del satélite *CubeSat Laser Infrared CrosslinK*.Fuente[18]

de programación y ayuda con la integración del código con otros códigos necesarios para hacer el proyecto completo, así como un entorno de simulación intuitivo y sencillo. Para la realización de las pruebas vamos a usar el microcontrolador Raspberry Pi 4 Model B.

En este paso se va a hacer un programa que simules los errores que sufriría el espejo de dirección rápida de un algoritmo PAT. Para ello, primero, creamos una conexión entre mi ordenador personal, que ha sido utilizado para en este proyecto y el microcontrolador Raspberry. Esto lo podemos hacer con un código muy sencillo que encienda y apague alternativamente unos sensores LED presentes en el microcontrolador, con esto nos aseguramos que la conexión establecida es fiable y no hay errores aparentes en el microcontrolador. Después de esto, se puede iniciar el programa que genere los errores en si, para esto se genera unos puntos aleatorias (uno para el eje X y otro para el Y) dentro de un rango definido , siguiendo una distribución gaussiana.

Inicialmente, se pensó que estos puntos se generarán realizando un barrido en espiral o escalonado, similar al que se realiza en el proceso de PAT grueso, pero,esta opción fue descartada , ya que , de esta forma no estamos moviendo el espejo de forma aleato-

ria y no estaríamos viendo realmente como afecta el ruido. Posteriormente, realizamos un programa cuyos puntos fueran obtenidos de forma puramente aleatoria, pero esta opción también se descartó al poder darse saltos muy grandes y muy repetidos en las coordenadas de nuestro espejo.

Finalmente se sigue una distribución gaussiana, ya que, proporciona variaciones con propiedades estadísticas similares a las del proceso real[20], ya que, el error de apuntamiento causado por la distribución de los satélites puede describirse mediante esta distribución, por ejemplo, en el caso de la incertidumbre en la elevación su función de probabilidad del ángulo del error de apuntamiento de elevación suele modelarse con la siguiente fórmula:[21]

$$f(\theta_{\mathrm{v}}) = \frac{1}{\sqrt{(2\pi)}\sigma_{\mathrm{v}}} exp[-\frac{(\theta_{\mathrm{v}} - \mu_{\mathrm{v}})^2}{\sigma_{\mathrm{v}}^2}] \tag{4}$$

donde θ_v es el ángulo de error de elevación, μ_v es el valor medio y σ_v la desviación estándar. Otro motivo para elegir esta distribución es que podemos generar diferentes niveles de variación.

En la Figura 6 puede verse el fragmento del código que crea un valor siguiendo una distribución gaussiana. En la primera iteracción del código generamos dos valores siguiendo una distribución gaussiana, después , en cada iteracción se dará un paso en cada eje, generado también con una distribución gaussiana ,y que no tiene porque ser igual en cada eje.

El rango anteriormente mencionado esta diseñado para asegurarnos que las variaciones no sean demasiado grandes, ya que, de lo contrario, estaremos trabajando con voltajes demasiado altos (el microcontrolador que estamos empleando tiene un voltaje de entrada a nivel alto de 3.3V[22]) que podrían afectar negativamente al circuito, además, así nos aseguramos que los saltos no sean excesivamente grandes.

Estos puntos obtenidos los trasladamos al código que vamos a explicar en el próximo apartado , que será el encargado de transformar dichos puntos, que son generados en cada ejecución del programa en unas coordenadas válidas.

6.2. AD56X4

El AD56X4[19] es un convertidor digital-analógico (DAC) el cuál usaremos para generar una señal analógica con nuestro microcontrolador. Este DAC usa una interfaz SPI[23], la cual es muy interesante de cara a este proyecto por dos factores fundamentales, su alta tasa de transferencia, que permite operar con altas velocidades de reloj así como una tecnología full-duplex, la cual nos da la posibilidad de enviar y recibir datos simultáneamente.

Para realiza el código necesario para este punto usaremos el editor de código fuente Visual Studio Core, como lenguaje de programación usaremos C++, esto es debido a la compatibilidad con el ecosistema de Arduino que usamos ,ya que, C++ no requiere

```
float generarGaussiano(float media, float desviacion){

float epsilon = 0.0000001;
static float z2;
  static bool z2existe = false;
  float u1, u2, z1;

  if (z2existe) {
    z2existe = false;
    return media + z2 * desviacion;
  }

  do {
    u1 = random(0, RAND_MAX) * (1.0 / RAND_MAX);
    u2 = random(0, RAND_MAX) * (1.0 / RAND_MAX);
  } while (u1 <= epsilon);

  z1 = sqrt(-2.0 * log(u1)) * cos(dosPI * u2);
  z2 = sqrt(-2.0 * log(u1)) * sin(dosPI * u2);
  z2existe = true;

  return media + z1 * desviacion;
}
```

Figura 6: Código para generar una valores según una distribución gaussiana en Arduino.

ninguna capa adicional para compatibilizar las bibliotecas usadas.

El AD56X4 recibirá los valores generados por la interfaz, desde una llamada desde dentro de la propia interfaz.Estos valores los convertirá en dos ángulos dentro del rango de movimiento permitido por el láser, un ángulo para cada eje. Después, se convertirán en dos impulsos eléctricos que se transferirán a la interfaz SPI que será la encargada de mover el láser a partir del circuito que se verá a continuación.

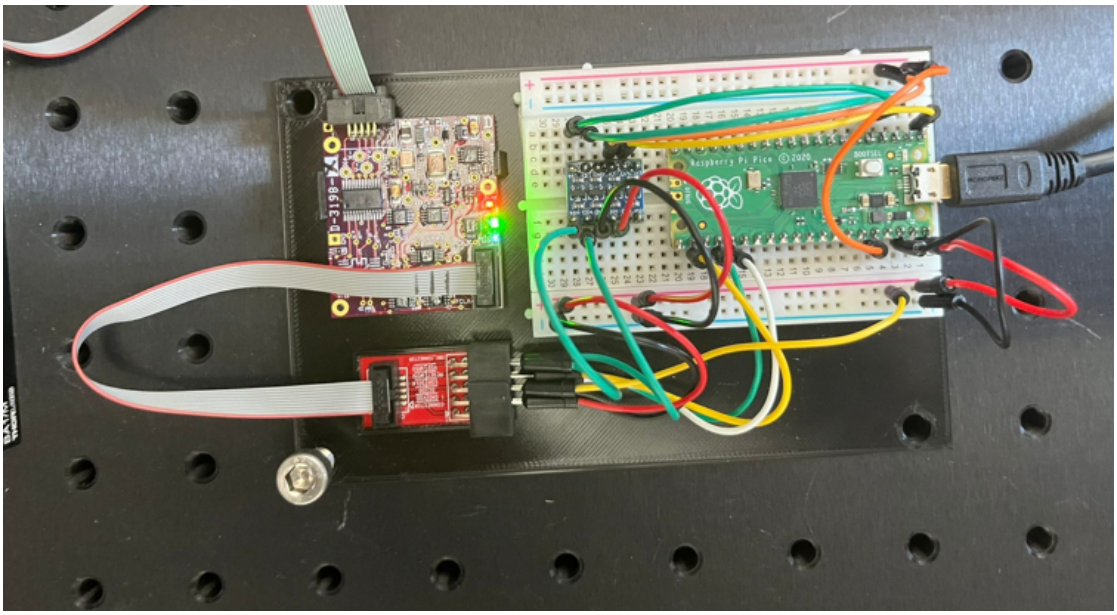

Figura 7: Microcontrolador Raspberry Pi

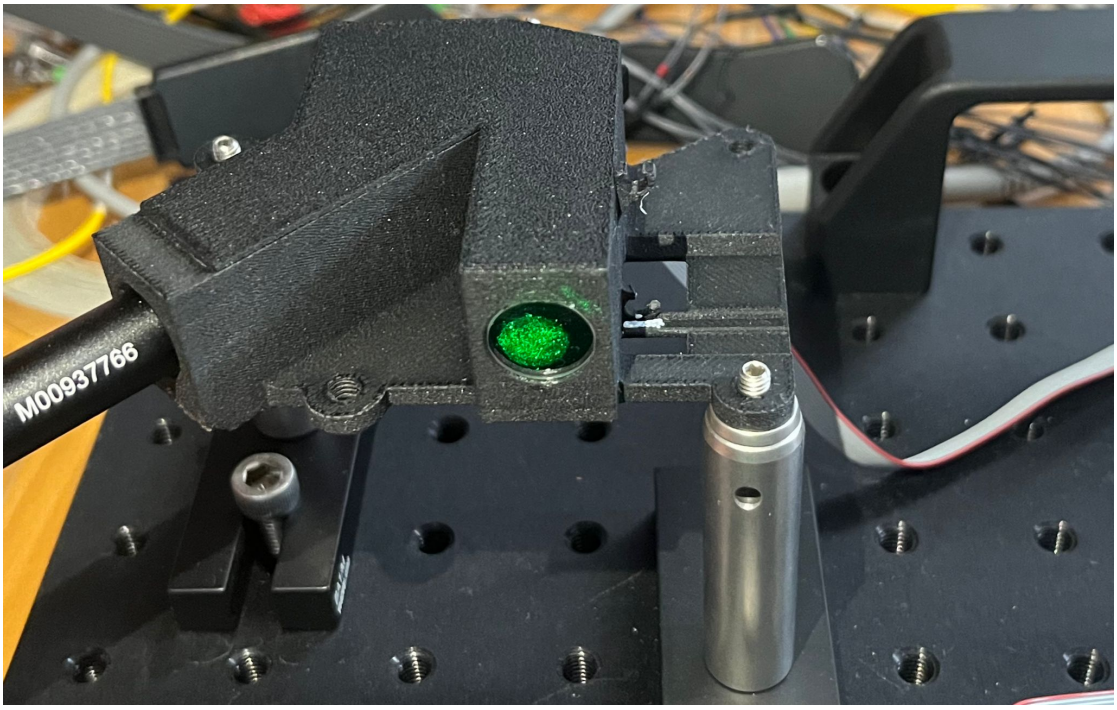

Figura 8: Lente que usamos para transmitir el láser

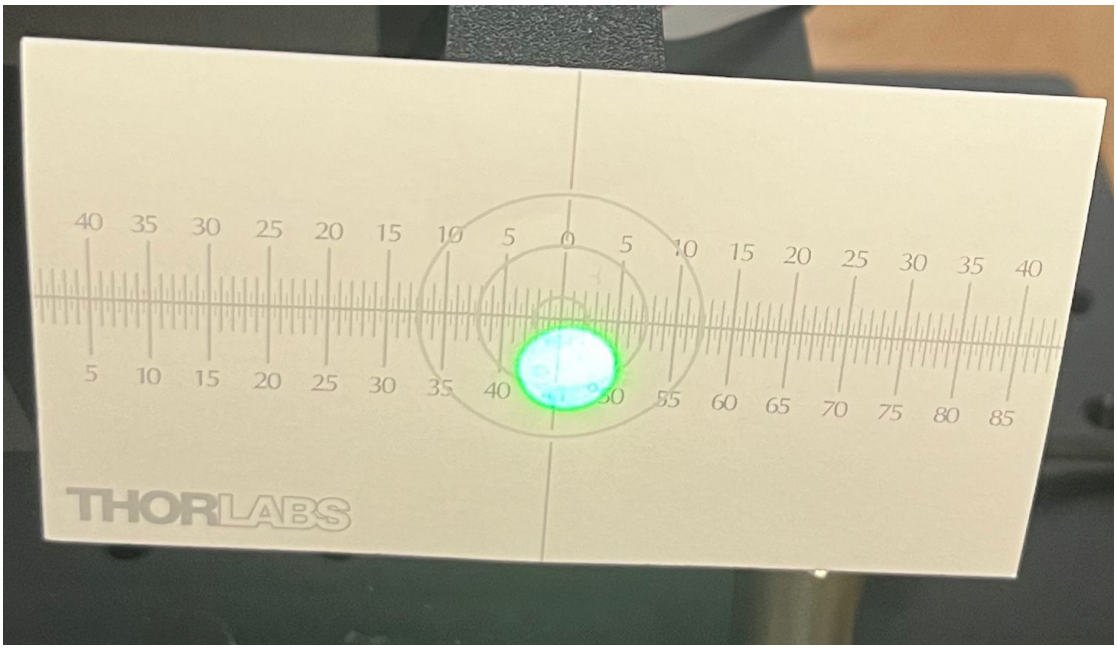

Figura 9: Visualización del láser.

6.3. Conexión externa con la lente

En la Figura 7 se puede ver una fotografía del microcontrolador que utilizamos, así como el cableado necesario para que la Raspberry envié los impulsos eléctricos hacia la lente. La Raspberry Pi está conectada al ordenador donde se ejecutan los programas anteriores, desde ahí recibe los impulsos eléctricos. El microcontrolador se conecta con una memoria RAM (dispositivo rojo en la Figura 7) ,la cual sirve como almacén temporal para los valores generados por el programa. La conexión se realiza mediante el bus de datos del microcontrolador, que permite la transferencia de datos, y el chip selector, que habilita o deshabilita a la RAM para recibir datos. Mediante el bus de direcciones indica en que posiciones de la memoria RAM quiere almacenar estos valores Esta memoria se conecta directamente con el circuito electrónico que es el componente que se va ligado al láser y que será el encargado de transportarle los impulsos eléctricos.

En la Figura 8 puede observarse la lente cuando está encendida. Para facilitar la visión del láser y su movimiento durante las pruebas se ideo el sistema que se ve en la Figura 9, que ayuda al ojo humano a seguir el movimiento del láser sin afectar a su visión.

7. Pruebas y resultados

En cuanto a los resultados obtenidos son bastante satisfactorios, ya que el circuito de conexión con el microcontrolador trabaja con tiempos de procesado adecuados y los movimientos creados en el se ven reflejados correctamente con el láser usado.

8. Conclusiones y líneas futuras

Para acabar, en este trabajo se han los errores más comunes que sufren los algoritmos PAT, dentro de este apartado podríamos destacar los siguientes puntos la necesidad de una continua mejora, a pesar de que los algoritmos PAT funcionen correctamente , esta claro que sufren una gran cantidad de errores en su funcionamiento, mejorarlo ayudaría enormemente en las comunicaciones globales. En relación a los objetivos marcados en este TFG los programas realizados cumplen las expectativas de simular los errores de los algoritmos PAT lo que los convierte en una buena herramienta útil de cara a la prueba de los algoritmos PAT.

En cuanto a las ,líneas futuras este proyecto esta pensado para englobarse dentro de un proyecto mayor, para mejorar sus resultados y corregir posibles errores. Quedando así la integración dentro un proyecto de mayor envergadura, idealmente, debería usarse en un algoritmo PAT que realice pruebas en tierra antes de su final lanzamiento, la cual no es posible de realizar en estos momentos por motivos de tiempo en la realización de este trabajo. Además, dada a la naturaleza del código, sería muy sencillo cambiar el ruido gaussiano con el que se realizan los movimientos del láser por uno de distintas características, que puedan amoldarse mejor a otro tipo de proyecto o descubrimientos futuros en este ámbito e incluso, añadir varios errores a la vez, en caso de que fuera necesario.

Referencias

[1] Public: *Mejoras en las comunicaciones láser*
https://www.nasa.gov/technology/6-things-you-need-to-know-about-nasas-laser-communications-

[2] Public: *Libro Comunicaciones por satélite*
https://ingetelecom.wordpress.com/wp-content/uploads/2007/08/comunicaciones-por-satelite.pdf

[3] Public: *PAT sin baliza y control adaptativo del haz mediante lentes de enfoque variable*
https://pubs.aip.org/aip/app/article/6/2/020801/123535

[4] Public: *Errores de apuntamiento debido a turbulencias atmosféricas*
https://ieeexplore.ieee.org/abstract/document/4914860?casa_token=P1D1seSs7CEAAAAA:O3ns6GcWDe-XB6dMMIhhWhgpwHPk_D6YYK6MemmJMH3XV9s4BgKI82gmfilSkX_NeDV77q0

[5] Public: *Rendimiento de los enlace de comunicación en un entorno con polvo.*
https://www.researchgate.net/profile/Mazin-A-Ali/publication/340174148_On_the_performance_of_free_space_optical_communication_link_over_dust_environment/links/5e7c5094a6fdcc139c046591/On-the-performance-of-free-space-optical-communication-link-over-dust-environment.pdf

[6] Public: *Libro Comunicaciones por satélite por Carlos Rosado.*
https://uahc.wordpress.com/wp-content/uploads/2014/01/comunicaciones-por-satelite.pdf

[7] Public: *RECOMENDACIÓN UIT-R P.1814.*
https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.1814-0-200708-I!!PDF-S.pdff

[8] Public: *RECOMENDACIÓN UIT-R P.618-8.*
https://www.itu.int/dms_pubrec/itu-r/rec/p/R-REC-P.618-8-200304-S!!PDF-S.pdf

[9] Public: *Coste de los pequeños satélites*
https://nanoavionics.com/blog/how-much-do-cubesats-and-smallsats-cost/

[10] Public: *Efectos de la radiancia Rayleigh*
https://opg.optica.org/oe/fulltext.cfm?uri=oe-24-11-12414&id=341081

[11] Public: *Libro donde se tratan temas como el mantenimiento de las órbitas*
https://ingetelecom.wordpress.com/wp-content/uploads/2007/08/comunicaciones-por-satelite.pdf

[12] Public: *Constelación de satélites de One Web*
https://www.sciencedirect.com/science/article/pii/S009457651630515X?casa_token=raTK1gek8gcAAAAA:fQE7WGbG5P4e7DeXWAu4mQL_TqxfHNHsXTEj9IUPpwyXlWDMUPBgWZZJmP58WGL3yB6Bum7

[13] Public: *Noticia sobre el lanzamiento del satélite de Space X*
https://www.xataka.com/espacio/spacex-ha-lanzado-satelite-telecomunicaciones-grande-mundo-asi-enor

[14] Public: *Comunicaciones láser en pequeños satélites*
https://digitalcommons.usu.edu/smallsat/2018/all2018/418/

[15] Public: *Misión CubeSat Laser Infrared Crosslink*
https://ieeexplore.ieee.org/document/9749715

[16] Public: *Testing Misión CubeSat Laser Infrared Crosslink*
https://digitalcommons.usu.edu/cgi/viewcontent.cgi?article=4617&context=smallsat

[17] Public: *Misión DSOC de la NASA*
https://www.nasa.gov/wp-content/uploads/2023/07/dsoc-fact-sheet-06152023.pdf?emrc=b01937

[18] Public: *Articulo de la Nasa de la misión CubeSat Laser Infrared CrosslinK*
https://www.nasa.gov/smallspacecraft/what-is-click/

[19] Public: *Datasheet AD5624R*
https://www.analog.com/en/products/ad5624r.html

[20] Public: *Influencia del ruido en Comunicaciones por Satélite*
https://ieeexplore.ieee.org/abstract/document/1095034?casa_token=wXe0K409RMoAAAAA:5z8Vh_Tx56ke9QhrFNN72cYvHx3Oks4TNJ9_BuWm2Cpb0qcQI2ZAxeAjFc4sIsHnFClh9IU

[21] Private: *"Free Space Optical Communication"por Hemani Kaushal y V.K. Jain Subrat Kar.*

[22] Public: *Datasheet Raspberry Pi 4*
https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf

[23] Public: *Interfaz SPI de Arduino*
https://docs.arduino.cc/learn/communication/spi/

[24] Public: *Proyecto Wiring*
https://www.wiring.org.co/

Anexo

A.Interfaz

En este apartado se va a profundizar en el código creado para la interfaz de este proyecto, que es el encargado de generar las coordenadas para mover el espejo. Este código ha sido generado en su totalidad en el entorno de desarrollo integrado multiplataforma Arduino IDE bajo las normas del lenguaje de programación C++. Este entorno proporciona una biblioteca software del proyecto Wiring[24].

Para empezar con el código, se definen los includes, bibliotecas , constantes y demás variables que son necesarias, esto puede verse en la Figura 10.

```
5  #include <Arduino.h> // include necesarios para el proyecto
6  #include "ad56X4.h"
7
8  AD56X4 controlador;  // instanciar el objeto de la clase AD56X4 de la biblioteca AD56X4 creada en C++
9
10 const float dosPI = 6.28318530717958647692528866; //constantes necesarias
11
12 int i;
13
14 //Creamos una instancia de la struct
15 struct dimensiones{
16
17   //Definimos una estructura para definir un punto en dos dimensiones
18   float x;
19   float y;
20 };
21
22 dimensiones inicial;
23 dimensiones actual;
```

Figura 10: Variables iniciales necesarias

Además de estas variables, son necesarias tres funciones, una para generar un valor inicial según una distribución gaussiana, que puede verse en la Figura 6 y otras dos para dar los pasos necesarios en cada eje, realmente estas dos funciones son iguales, pero, se crean dos, ya que de lo contrario los pasos serían iguales en cada eje. Esta función para el eje X puede verse en la Figura 11.

```
27 //Funciones para dar un paso en cada eje
28 float aleatoriox(dimensiones actual, float paso){
29
30     actual.x =actual.x + paso;
31     return actual.x;
32
33
34 }
```

Figura 11: Función para dar un paso en el eje X

Después de esto, esta una de las funciones básicas de Arduino IDE , setup() ,el código dentro de esta función solo se ejecutará una vez cada vez que se ejecute el código creado. Este código se ve en la Figura 12.

```
68  void setup() {
69
70
71   controlador.begin();  // inicializamos las funciones de AD56X4
72   controlador.initialize();
73
74    Serial.begin(96000);
75    randomSeed(analogRead(0)); //función para generar números aleatorios
76    i=1;
77
78  }
```

Figura 12: Función setup de Arduino

Por último, esta la segunda función de Arduino IDE, loop(), el código encerrado en esta función se ejecutará en bucle hasta que paremos la ejecución del código. Aquí, llamaremos a la función que genera un valor gaussiano en la primera ejecución, Figura 13.

```
void loop() {
  // put your main code here, to run repeatedly:
  if(i==1){


  //generamos los valores gaussianos con media 0 y desviacion 1
  float valorGaussianox= generarGaussiano(0,1);
  float valorGaussianoy=generarGaussiano(0,1);

  //ajustamos los valores guassianos para que estén dentro del intervalo [-1.25 ; 1.25]
  float valorxescalado = valorGaussianox * 1.25;
  float valoryescalado = valorGaussianoy * 1.25;

  actual.x = valorxescalado;
  actual.y= valoryescalado;
  i = 2;
```

Figura 13: Llamada a la función que genera los valores según una distribución gaussiana.

En las demás ejecucionesFigura 14, se dará un paso en cada eje, también generado según una distribución gaussiana, estos valores se enviarán a AD56X4, en cada ejecución, que será la encargada de convertirlos en las coordenadas capaz de mover la lente como ya se explicó.

```
}if(i!=1){


 //generamos el pasode forma gaussiana tambien
 float pasox = generarGaussiano(0,1);
 float pasounox = pasox/10; //ajustamos el paso para que no sea demasiado grande
 float pasoy = generarGaussiano(0,1);
 float pasounoy = pasoy/10;
 actual.x =aleatoriox(actual, pasounox);
 actual.y= aleatorioy(actual,pasounoy);
 float limite =1.25;//valor limite , el punto actual no puede ser mayor a +-1.25 en ningun eje
 float nlimite = -1.25;
 //si alguno de los valores supera el limite, lo reiniciamos a un nuevo valor aleatorio Gaussiano , aquí se puede dar un paso muy grande, sería lo unico a corregir
 if(actual.x >limite || actual.x<nlimite){
  actual.x = generarGaussiano(0,1) * 1.25;

 }
 if(actual.y >limite || actual.y<nlimite){
  actual.y = generarGaussiano(0,1) * 1.25;
 }
  Serial.print("x:");
  Serial.print(actual.x);
  Serial.print(",y:");
  Serial.println(actual.y);


  //Le pasamos los distintos angulos a la funcion que mueve el espejo
  controlador.deflect_to(actual.x, actual.y);



}
```

Figura 14: Fin del loop en Arduino IDE.