UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Escuela Técnica Superior de Ingeniería Informática

Mejoras a la herramienta ArduSim mediante la creación de una interfaz para sensores virtuales

Trabajo Fin de Grado

Grado en Ingeniería Informática

AUTOR/A: Faubel Marco, Pablo

Tutor/a: Tavares de Araujo Cesariny Calafate, Carlos Miguel

Director/a Experimental: WUBBEN, JAMIE

CURSO ACADÉMICO: 2021/2022

RESUMEN

Con el avance de la tecnología y el abaratamiento de los drones, se ha comenzado ha hacer uso de ellos para numerosas tareas en la ultima década. El uso de estos para tomar medidas de contaminación del aire empieza a ser una realidad, pero, aunque los costes hayan disminuido, un error durante el vuelo de uno de ellos puede resultar catastrófico.

Para poder avanzar en este campo del uso de drones para la acción por el clima de manera más sostenible, este proyecto presenta la posibilidad de avanzar mediante el uso de simulaciones. Para ello, se propone la implementación de un protocolo de sensores de contaminación en la aplicación de ArduSim, aplicación de código abierto que permite controlar y realizar simulaciones en tiempo real del vuelo de drones.

Palabras clave: ArduSim, kriging en contaminación, sensores de contaminación en drones, simulación de contaminación.

ABSTRACT

With the advancement of technology and the cheapening of drones, they have begun to be used for numerous tasks in the last decade. The use of these to take measures of air pollution is beginning to be a reality, but, although costs have decreased, an error during the flight of one of them can be catastrophic.

In order to advance in this field of the use of drones for climate action in a more sustainable way, this project presents the possibility of advancing through the use of simulations. For this is proposed the implementation of a pollution sensor protocol in the ArduSim application, an open source application that allows real-time control and simulation of drone flight.

Keywords: ArduSim, pollution kriging, drone pollution sensors, pollution simulation.

RESUM

Amb el desenvolupament de la tecnologia i l'abaratiment dels drons, s'ha començat a fer ús d´aquests per a nombroses tasques en l´ultima dècada. L'ús d'aquests per prendre mesures de contaminació de l'aire comença a ser una realitat, però encara que els costos s´hagen disminuit, un error durant el vol d'un pot resultar catastròfic.

Per poder avançar en aquest camp de l'ús de drons per a l'acció pel clima de manera més sostenible, aquest projecte presenta la possibilitat d'avançar mitjançant simulacions. Per això, es proposa la implementació d'un protocol de sensors de contaminació a l'aplicació d'ArduSim, aplicació de codi obert que permet controlar i realitzar simulacions en temps real del vol de drones.

Paraules clau: ArduSim, kriging en contaminació, sensors de contaminació en drons, simulació de contaminació.

ÍNDICE

1. Introducción……6
1.1. Motivación……6
1.2. Objetivos……...6
1.3. Estructura del documento……..8
2. Estado del arte en uso de drones para medir contaminación……9
3. El simulador de ArduSim……10
4. Propuesta de entorno de sensorización virtual……12
4.1. Arquitectura general……..12
4.2. Comunicaciones en red y formato de mensajes……..13
4.3. Procesamiento de datos de entrada……..14
4.4. Modelado de ruido gaussiano……15
4.5. Estructura global del código……16
5. Validación……..19
5.1. Análisis del tiempo de generación del mapa de calor……19
5.2. Análisis del consumo de recursos de cada sensor virtual……21
5.3. Simulación en punto fijo: análisis de la variabilidad de los datos generados ……..23
5.4. Simulación en movimiento: análisis de la tendencia de los resultados ……..24
6. Conclusiones……26
Bibliografía……27
Anexo I: ODS……30

ÍNDICE DE FIGURAS

Figura 1: Interfaz de Ardusim……..11
Figura 2: Arquitectura del proyecto……12
Figura 3: Mensajes del servidor……..13
Figura 4: Relación entre la varianza y el rango……14
Figura 5: Formula del modelo circular……15
Figura 6: Grafica de una distribución gaussiana……..16
Figura 7: Diagramas de flujo, tratamiento de datos y servidor……17
Figura 8: Diagrama de flujo de la ejecución de la aplicación……..18
Figura 9: Tiempo que tarda en calcular el valor de contaminación con 6 estaciones……..20
Figura 10: Tiempo que tarda en calcular el valor con 48 estaciones…….20
Figura 11: Media y mediana de tiempo según el número de estaciones medidas……..20
Figura 12: Tendencia de crecimiento de media y mediana……21
Figura 13: Captura 1 del comando top……..22
Figura 14: Captura 2 del comando top……..22
Figura 15: Histograma de valores devueltos. Valor real 37.982……23
Figura 16: Ejecución con movimiento de los drones……25

1. Introducción

Este trabajo fin de grado (TFG) trata el tema de la contaminación ambiental, y su monitorización usando drones. A continuación, se motiva debidamente por qué se ha elegido esta temática en el contexto de los actuales retos de nuestra sociedad. Más adelante, se definen los objetivos del TFM, y por último se detalla la estructura de este documento.

1.1. Motivación

El proyecto presentado se realizó pensando en una mejora en el ámbito de la monitorización ambiental. Dada la presencia que tiene a día de hoy el campo de la contaminación global, se optó por aportar nuestro granito de arena para ayudar en la reducción de emisiones de gases contaminantes.

Desde un principio nos una buena oportunidad el tratar de mejorar el campo de la monitorización medioambiental mediante el uso de nuevas tecnologías, concretamente, haciendo uso de vehículos no tripulados o drones. Teniendo esto en mente se optó por implementar una mejora a una aplicación denominada ArduSim que permite realizar simulaciones de vuelo de este tipo de vehículos. Dicha mejora, dotaría al sistema de la capacidad de simular unos sensores de contaminación en los drones que permitan la monitorización de gases contaminantes en tiempo real.

Otra de las posibilidades observadas en este proyecto sería la reducción de costos monetarios y materiales que se generarían al realizar diferentes estudios del comportamiento de los drones de manera simulada. Además, se obtendría otra una importante reducción en comparación al modelo de monitorización mediante el uso de estaciones meteorológicas.

Para acabar, también resulta muy interesante la movilidad que aportan los drones a este modelo, pues estos pueden desplazarse casi por cualquier lugar e iniciar el vuelo desde donde se requiera para realizar el monitoreo, eliminando así un importante inconveniente de las estaciones.

1.2. Objetivos

- Desarrollar un nuevo protocolo para la aplicación de ArduSim que implemente la simulación de sensores de contaminación en los drones. Este formaría el objetivo principal del proyecto.

  -Indagar en el funcionamiento y la estructura que toma la aplicación de ArduSim para comprender el funcionamiento y como se debe implementar el nuevo protocolo

-Estudiar el manejo de los drones en la aplicación y la manera de programar su comportamiento utilizando los métodos ya implementados en ArduSim.

- Desarrollar un módulo software externo capaz de modelar un sensor virtual de cualquier tipo, ofreciendo lecturas a los drones de la herramienta ArduSim mediante comunicaciones TCP/IP.

- Investigar en el funcionamiento y uso del método kriging.

  -Conocer el funcionamiento del método de interpolación geoestadística y la manera de configurarlo para generar una matriz con datos de contaminación para la simulación.

  -Estudiar la manera de configurar el método para generar un conjunto de datos de contaminación los más similares posibles a los reales.

- Generar un servidor que estime los valores de contaminación en todo el plano haciendo uso de kriging y se comunique con la aplicación de ArduSim para emular el funcionamiento de sensores a bordo de los drones.

  -Desarrollar un servidor UDP/IP que permita comunicar los drones de la aplicación como clientes de este para el envío de los datos de contaminación.

- Estudiar la contaminación medioambiental, específicamente sobre los gases de contaminación del aire.

  -investigar sobre la contaminación del aire para plasmarlo de manera correcta en la simulación que se realicen.

- Demostrar los conocimientos obtenidos a lo largo del grado.

  -Confirmar el conocimiento adquirido sobre lenguajes de programación.

  -Diseñar algoritmos para el funcionamiento del proyecto.

- Aprender nuevos conceptos que se prevén utilizar para el desarrollo del proyecto.

  -Realizar un proyecto de esta magnitud.

  -Conocer el lenguaje de programación Python.

  -Trabajar con una aplicación de grandes dimensiones y adaptarse a su funcionamiento para programar de acuerdo con los estándares ya establecidos por los otros programadores.

1.3. Estructura del documento

Esta memoria de TFG se estructura de la siguiente manera:

- En el presente capítulo se detalla la motivación y principales objetivos de este trabajo.

- En el capítulo 2 se hace un breve estudio del estado del arte en uso de drones para medir contaminación, destacando las aportaciones más relevantes realizadas por la comunidad científica hasta el momento.

- En el capítulo 3 se ofrece una visión general del simulador de drones ArduSim, el cual servirá de base para la realización de este TFG.

- En el capítulo 4 se detalla la propuesta de un entorno de sensorización virtual. En concreto se presenta la arquitectura del sistema, detalles de implementación, aspectos relacionados con la red y las comunicaciones, y el modelado de ruido en los sensores.

- En el capítulo 5 se hace una validación de la solución propuesta, lo cual incluye análisis del tiempo de generación de los mapas de calor, el consumo de recursos asociado a cada máquina virtual, así como experimentos para validar los datos generados tanto en un punto fijo como en movimiento.

- Finalmente, en el capítulo 6, se presentan las conclusiones del trabajo, destacando también la relación del mismo con las asignaturas de la carrera, y posibles trabajos futuros.

2. Estado del arte en uso de drones para medir la contaminación

Las primeras ideas de este planteamiento no son ninguna novedad, ya a finales de la década pasada aparecían artículos y planteamientos sobre el uso de drones para medir la contaminación con el fin de reducirla, ya que esta afecta negativamente no solo a la salud de los seres vivos, sino que llega a afectar hasta las estructuras de construcciones.

Ya en 2015, según el artículo [7], el gobierno chino estableció un programa para llevar a cabo este planteamiento ya que en algunas de sus ciudades sobrepasaban por mucho los niveles límite recomendados para la salud llegando a ser áreas críticas. El uso de drones frente al de estaciones base genera las ventajas de ser capaces de recoger datos de distintas zonas, midiendo los niveles en varios puntos de manera más rápida y del ahorro en los costes que se generan al no tener que crear las instalaciones necesarias para las bases de medición.

También en Estados Unidos se ha planteado esta tecnología con el propósito de generar ahorros en las empresas industriales evitando costes por sanciones, aparte de mejorar las condiciones de seguridad para los trabajadores. Para ello se implementó un prototipo utilizando una mezcla de tecnologías utilizadas en distintos dispositivos, entre ellas se encuentran las tecnologías GPS y de cámaras de alta resolución para trazar mapas de polución. También se da uso a sensores en miniatura integrados en los drones del menor peso posible para afectar lo mínimo posible al vehículo.

En los laboratorios de Intel también se ha desarrollado un proyecto similar, en el cual se fusiona un medidor de calidad del aire con un smartphone, donde este último se encarga de mandar los datos en tiempo real a un servidor que opera en la nube.

Otro artículo publicado en 2017, [8] sobre una tesis realizada en la Pontificia Universidad Católica del Perú destaca el ahorro de costes que se logra con estos drones en comparación a la creación de estaciones o instalaciones específicas para este fin. En este informa que el presupuesto de construcción de una caseta es casi 5 veces mayor al necesario por los drones, además del aumento que supondría anualmente para el mantenimiento de la instalación. Por otro lado, la capacidad de los drones se ve reducida en cuanto a la altura a la que alcanzan, la cual llega a ser mucho mayor para las estaciones o globos de medición.

Para finalizar, a inicios del 2019, se realizó en la universidad Técnica de Creta (Grecia) una tesis relacionada con este campo, [9], en trataba el uso de vehículos aéreos no tripulados (UAV) para visualizar en 3D los contaminantes del aire en áreas urbanas, cuyo objetivo era capturar las emisiones contaminantes del aire y crear una herramienta de visualización 3D.

3. El simulador de ArduSim

ArduSim es una aplicación diseñada tanto para realizar simulaciones de vuelo de drones en tiempo real como para controlar drones reales, creado principalmente para el estudio de la coordinación de los patrones de vuelo o la implementación de protocolos que se ajusten a estos como el que se desarrolla en este proyecto. Para esto ArduSim permite añadir cuantos drones necesitemos para trabajar con ellos de forma simultánea siempre y cuando las prestaciones de nuestra computadora los soporte sin problemas.

La aplicación trata de simular las condiciones reales que tendría el vuelo de un o conjunto de drones, por lo que podemos encontrar características destacables como la posibilidad de configurar las condiciones meteorológicas adversas para estos, como podría ser el viento. También podemos configurar ciertas características que tendría un dron como la capacidad en miliamperios de la batería, lo que restringiría el tiempo de vuelo. Todo ello permite realizar estudios y pruebas sobre algoritmos de detección de colisiones entre drones y trabajar en protocolos internos dentro de ArduSim para programar el modo de evitarlas.

El funcionamiento interno de la aplicación divide el control de los drones o vehículos no tripulados de manera que cada dron sea autónomo del resto, para poder desarrollar el detector de colisiones. Dentro de cada dron virtual se encuentran dos bloques reconocibles que se encargan de darle a cada dron su comportamiento, el primero consta del controlador que manejará el desplazamiento del dron, además de la simulación de la física que afecte a este y la potencia suministrada a los motores en cada momento. El segundo bloque consiste en un conjunto protocolos lógicos que manejan el detector de colisiones, para ellos dispone como mínimo de dos hilos de ejecución, los cuales realizan las tareas de emitir trasmisiones de difusión y escuchar las que realizan otros drones. Esta segunda forma el protocolo de deleción de colisión que implementa ArduSim.

Para poder obtener el código se requiere de un programa competente para desarrollo en Java, se recomienda el uso de Eclipse o IntelliJ, y el código del programa se puede obtener de github, plataforma donde se comparte código libre por la comunidad de programadores de todo el mundo, siguiendo el enlace de [6]. A parte del código necesario se encuentra información de utilidad, bien sobre el funcionamiento de la aplicación como de los métodos que forman la aplicación para familiarizarse rápidamente con estos y poder entender y adaptarse al manejo de ArduSim.

Internamente los archivos que componen la aplicación también los podemos encontrar en el enlace al github, pero cabe destacar la localización y la estructura que toman los protocolos implementados ya que sobre esto es donde de desarrolla este proyecto. Los protocolos se ubican en el directorio ArduSim.scr.main.java.com y cada uno de ellos dispone como mínimo de dos capetas internas, gui y logic, encargadas de almacenar el código que afecte a la interfaz gráfica y a las funciones lógicas del programa respectivamente. Y estos protocolos se ejecutan por separado en función del que se escoja al iniciar la ejecución del método principal.

Para finalizar, la interfaz utilizada durante la ejecución puede observase en la Figura 1.

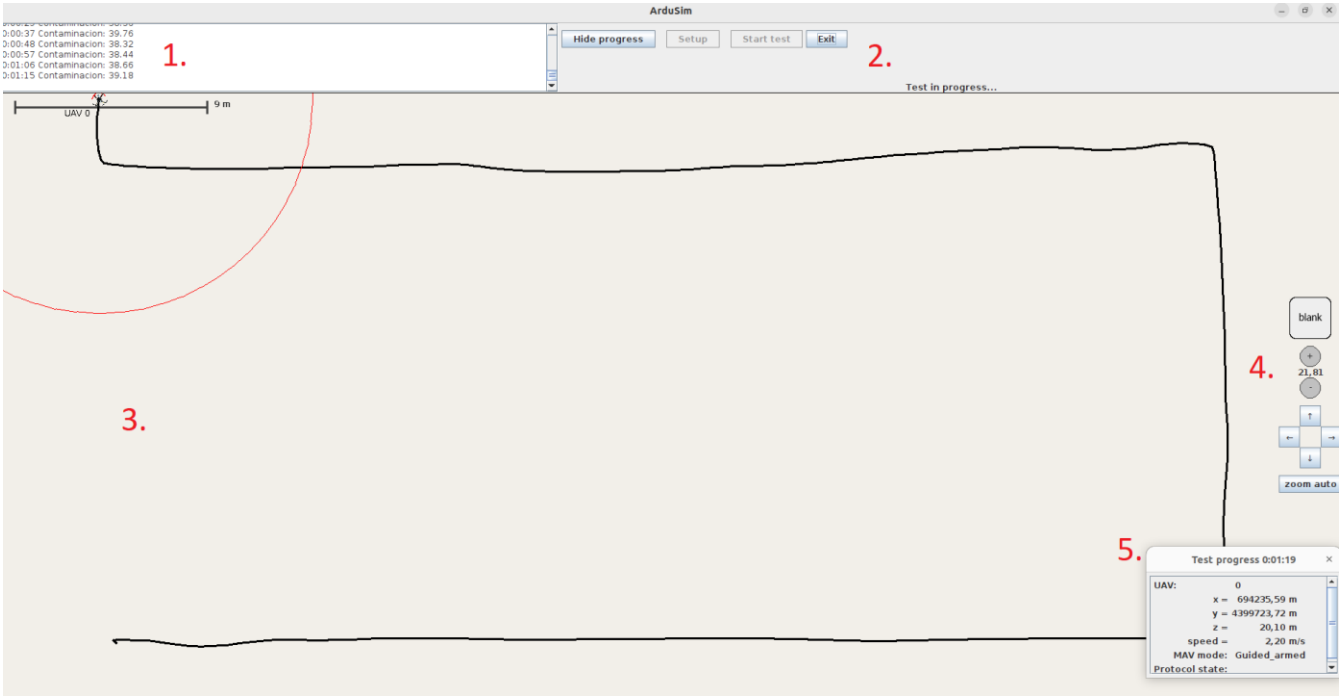

Figura 1: Interfaz de Ardusim

En la figura se puede observar las diferentes partes de la interfaz numeradas, la primera parte es la terminal donde se muestra la información conforme de ejecuta la aplicación. La segunda parte está formada por un conjunto de botones que habilitan algunas acciones para interactuar con la ejecución, estas se tratan del inicio de la ejecución una vez todos los procesos estén listos y de la interrupción de esta en caso de necesitar cancelarla. En la tercera parte, la cual ocupa el grueso de la pantalla se muestra el desarrollo de la trayectoria de los drones, así como la ubicación del plano en la que se encuentran ya que ArduSim tiene la implementación de funcionar con Google maps, permitiendo acceder a los planos de las zonas sobrevoladas. En la cuarta zona se disponen de un conjunto de botones que permiten desplazar el plano para poder visualizar la zona deseada. Y para acabar la última zona muestra información sobre las características de los drones en tiempo real de ejecución, así como sus coordenadas y la altura y velocidad en la que se encuentran.

En caso de estar interesado en conocer mejor la aplicación de ArduSim es muy recomendable revisar su documentación en la web [6].

4. Propuestas de entorno de sensorización virtual

En este apartado se comenta el planteamiento o estructura que se la ha dado al proyecto para poder alcanzar los objetivos planteados inicialmente. Veremos la arquitectura general que toma el proyecto, la manera en la que se realizan la comunicación en red, como se gestionan los datos de entrada, el ruido que se ha programado para darle mayor verosimilitud al sistema y para acabar, la estructura que ha tomado el código.

4.1. Arquitectura general

La arquitectura que toma el proyecto si divide en dos bloques importantes, el primero de ellos es el desarrollo del servidor, y el segundo consiste en la implementación de un protocolo dentro de la aplicación de ArduSim.

El primer punto consiste en un conjunto de datos de entrada al sistema compuestos por valores dispersos en un plano de dos dimensiones. Estos puntos son considerados a manera de estaciones de medición, en las que se supone un valor de contaminación ambiental real, los cuales serán utilizados para la estimación del resto de valores del plano.

Con estos se aplicará un modelo de interpolación geoestadística o kringing, el cual, haciendo uso de fórmulas matemáticas en las que se tiene en cuenta la autocorrelación entre los valores de las estaciones, permite realizar una estimación de los valores del resto del plano.

Hasta este punto todo se encuentra programado en Python, sin embargo, la aplicación ArduSim se encuentra en java, por lo que el último apartado se realiza en este lenguaje de programación.

Por último, para simular los sensores en la aplicación de ArduSim, se conectarán, mediante un servidor UDP, los drones de la aplicación con un servidor externo a esta que se mantenga a la escucha de las peticiones realizadas por cada uno de los drones. Además, para lanzar la simulación se hace uso de una clase Helper que se hereda de la propia aplicación en la que nos apoyamos para controlar los procesos de ArduSim y manejar el desplazamiento de los UAV durante el vuelo.

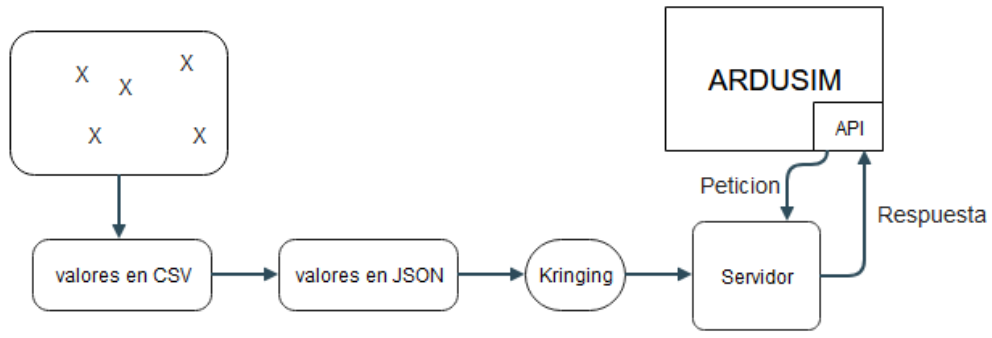

Figura 2: Arquitectura del proyecto

4.2. Comunicaciones en red y formato de mensajes

Las comunicaciones que se realizan en red en el proyecto se tratan del envió de peticiones desde los drones y la respuesta del servidor externo a la aplicación de ArduSim. Para ello, el contenido de los mensajes por parte de los drones está formado por las coordenadas en las que se encuentra en ese momento cada dron sin tener en cuenta la altura, y el servidor responderá con la estimación realizada con el método de kringing para la ubicación exacta de cada uno de ellos.

Como se comentaba anteriormente ambas partes del sistema se programan en distintos lenguajes de programación, sin embargo, para el manejo de los datos se utiliza el formato JSON.

Las peticiones realizadas por los drones tienen el siguiente formato:

{“x”: altitud, “y”: longitud}

Y las respuestas a estos que realiza el servidor se componen de la siguiente forma:

{“contamination”: valor_estimado}

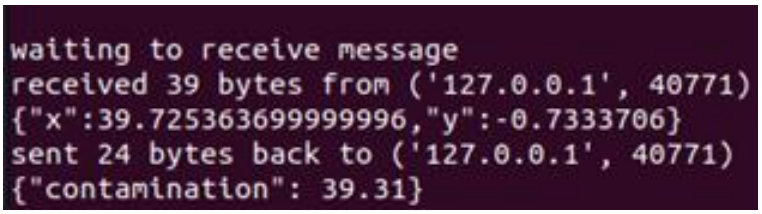

Figura 3: Mensajes del servidor

Por otro lado, es necesario remarcar que las comunicaciones del servidor son del tipo UDP, ya que al tratarse de envíos de datos pequeños agiliza el proceso frente al TCP, que invierte tiempo adicional en crear un canal de comunicación seguro. Este protocolo es menos fiable pero no supondría un error fatal en la ejecución del simulador la pérdida de un mensaje por lo que para ponerle solución se ha optado por realizar un reenvío de la petición si al cabo de cierto tiempo el dron no recibe respuesta del servidor, ya sea porque ha habido un error en la comunicación o una colisión con otro dron a la hora de comunicarse con el servidor. El estado de reenvío de la petición se mantiene un numero cierto número de veces limitado para, en caso de no obtener ninguna respuesta, continuar con la trayectoria de vuelo y que no se quede bloqueada la ejecución.

4.3. Procesamiento de datos de entrada

Los datos de entrada al sistema consisten en un conjunto de valores dispersos en un plano bidimensional. Estos se componen de tres valores, dos de ellos para indicar el posicionamiento en el plano, latitud y longitud, y el ultimo a modo de valor de contaminación en el punto asignado. Estos valores vienen dados por las estaciones de medición en formato CSV los cuales, para un mejor manejo de los datos, se pasan a formato JSON.

El último apartado del procesado de datos consiste en un método de interpolación geoestadística o kriging, el cual consiste en, mediante el uso de fórmulas matemáticas que se desarrollan más adelante, estimar el valor en el plano completo, sacado a partir de los valores de entrada que obtenemos de las estaciones. Con este podemos simular un valor de contaminación en cualquier punto para devolverlo a los drones. Este método es usualmente utilizado en el campo de la geología.

Para configurar este método se ha hecho uso de la librería pykrige, la cual da soporte a estas técnicas de interpolación para el lenguaje de Python, y permite realizar estimaciones tanto para dos como para tres dimensiones. En nuestro caso se ha trabajado de forma bidimensional.

El método funciona calculando la autocorrelación en función de la distancia en la que se encuentren los valores ya conocidos del punto a calcular y se disponen de numerosas variables para configurarlo. Las más significativas serían las siguientes:

En primer lugar, la estimación de los pesos se puede ajustar a diferentes modelos o curvas de peso en función de la distancia, se pueden seleccionar entre el gaussiano, el lineal, el circular o el exponencial, y un par más. En nuestro caso se ha decidido hacer uso del modelo circular, debido a que permite la eliminación del efecto de un valor que se encuentre a un más alejado de cierto rango ajustable, además, que este es uno de los más utilizados. Este modelo permite al sistema reducir los costes de cómputo a la hora de estimar el valor de contaminación en el caso de que configuremos un alto número de estaciones como partida, eliminando las que se encuentren demasiado lejos de la formula.

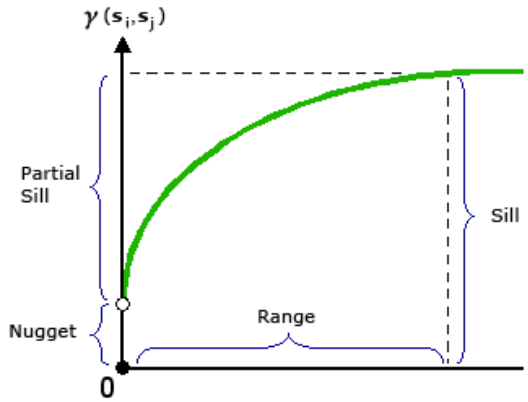

Figura 4: Relación entre la varianza y el rango

Además de la selección del modelo, existen otros parámetros configurables que sirven para describirlo, haremos uso de la Figura 4 para explicarlos. En la

gráfica mostrada se representa la varianza que puede llegar a tener un valor en función de la distancia a la que se encuentre de los puntos con valor conocido

Estos se configuran dentro del método kriging con el parámetro “variogram_model” y se compone de tres variables. El primero de ellos sería el rango, se le llama así a la distancia a la que el modelo comienza a aplanarse, esto significa que las ubicaciones que están más alejadas no se encuentran autocorrelacionadas espacialmente con el punto en el que se opera. Por otro lado, tenemos el valor sill o meseta, este consiste en el valor del eje Y en el punto donde se alcanza el rango. Por último, encontramos el valor Nugget o umbral, el cual se aplica cuando la distancia de separación es igual a 0. El valor que tomaría en un modelo medido sería el valor obtenido en la intersección con el eje Y. Este se atribuye a errores de medición, al tomar medidas sobre el mismo punto pueden aparecer diferencias en el valor medido debido al error propio que poseen todos los dispositivos de medición.

Las fórmulas que toma el funcionamiento del modelo circular se representan en la Figura 5. En el sistema se representan las siguientes variables:

-d: distancia a los puntos en los que se calcula el semivariograma

-r: rango ajustado

-n: umbral

-p: meseta parcial = meseta - umbral

En las fórmulas de todos los modelos se diferencian dos partes, una parte a la que le afecta la distancia, ajustado por el rango y la meseta del modelo y una segunda invariable a la posición, dada por el umbral. La segunda ecuación del sistema sería el caso en el que la posición del punto a correlacionar se encuentra más allá del rango, por lo que se suman las dos partes y el peso que se le otorga al valor sería nulo.

$$\begin{cases} p \cdot \left(\frac{3d}{2r} - \frac{d^3}{2r^3}\right) + n & d \leq r \\ p + n & d > r \end{cases}$$

Figura 5: Formula del modelo circular

4.4. Modelado de ruido gaussiano

En la respuesta por parte del servidor se ha implementado una función que genera un ligero error sobre los valores estimados en las coordenadas que mandan los drones. Con ellos se pretende eliminar la posibilidad de recibir

exactamente la misma información al tomar medidas en el mismo punto, con el fin de conseguir resultados más realistas en la simulación.

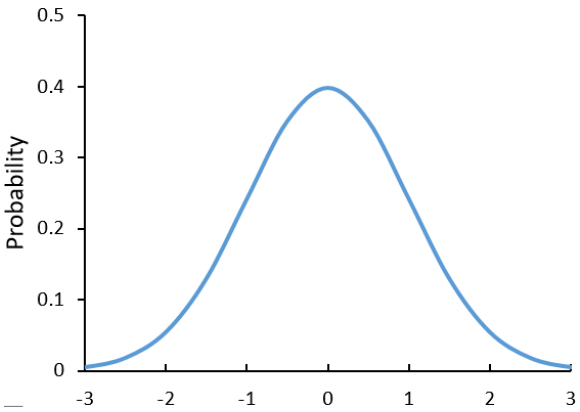

Figura 6: Grafica de una distribución gaussiana

Para ello se ha hecho uso de la función random, con una distribución normal o gaussiana. Con ella se genera un valor aleatorio con una distribución simétrica en torno a la media, la cual se ajustan en el valor estimado por el método kriging.

La desviación típica representa el rango en el que se encuentran la gran mayoría de los valores, para ser exactos, un 95% de los valores se encuentra comprendido en el rango de dos desviaciones típicas por arriba y otras dos por abajo.

Para no generar un ruido excesivo en el sistema se ha aplicado una desviación típica del 1% del valor de la media. Por lo que, el grueso de valores resultantes que el servidor devuelva, se encontrarán comprendidos entre un ±2% del valor original.

4.5. Estructura global del código

En el diseño final del proyecto se pueden destacar dos bloques en la estructura del código final. Estos se dividen en la parte del servidor y la parte interna a la aplicación de ArduSim.

En el primero de ellos se compone de un par de scripts externos a la aplicación mencionada programado en Pyhton. En este se realiza la gestión de los datos de las estaciones, los cuales tienen que ser generados previamente a la ejecución del código y se deben encontrar en el directorio correspondiente. Durante la ejecución del servidor se realiza la lectura de los datos y su paso a formato Json, posteriormente se implementa un bucle infinito donde el servidor se mantendrá a la escucha de las peticiones realizadas por los drones. En el interior de dicho bucle se realiza la recepción de la petición de un dron, mediante la cual se reciben las coordenadas en las que este se encuentra. Con las coordenadas, se realiza la estimación de la contaminación en este punto, con el

método kriging y se aplica sobre el resultado el generador de error. Una vez calculado el valor a devolver se encapsula en un mensaje Json, se devuelve al dron inicial y se reinicia el bucle. En el siguiente esquema se puede visualizar mejor.

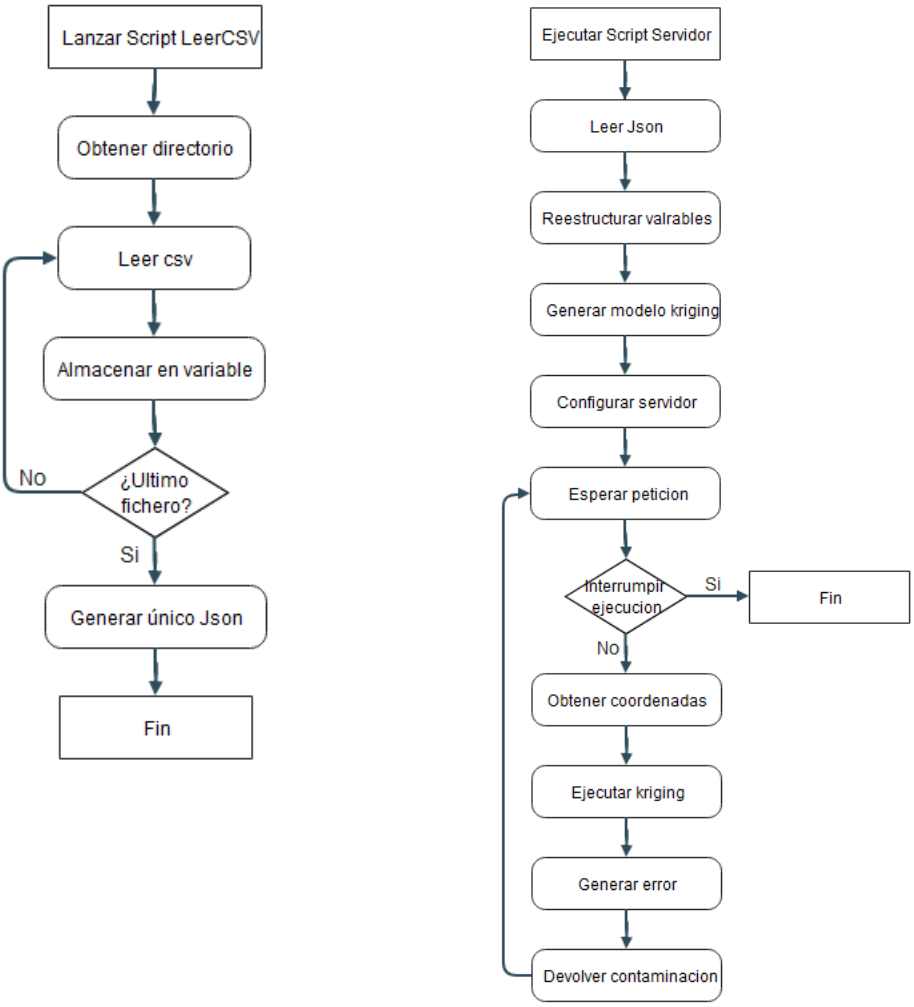

Figura 7: Diagramas de flujo, tratamiento de datos y servidor

En el apartado de la Aplicación ArduSIm se han programado dos clases. Una en la que se configura el nuevo protocolo creado, el cual sirve para comunicarse con el servidor y obtener los datos de contaminación como si fuera un sensor, y el segundo consiste en la implementación de una clase heredada en la que se han completado ciertos métodos. Esta es implementada por la aplicación de ArduSim para el control de la ejecución de las misiones de vuelo, así como algunos procesos internos que ya se encuentran en esta. En la figura 8 podemos ver el diagrama de flujo que tiene esta parte del código.

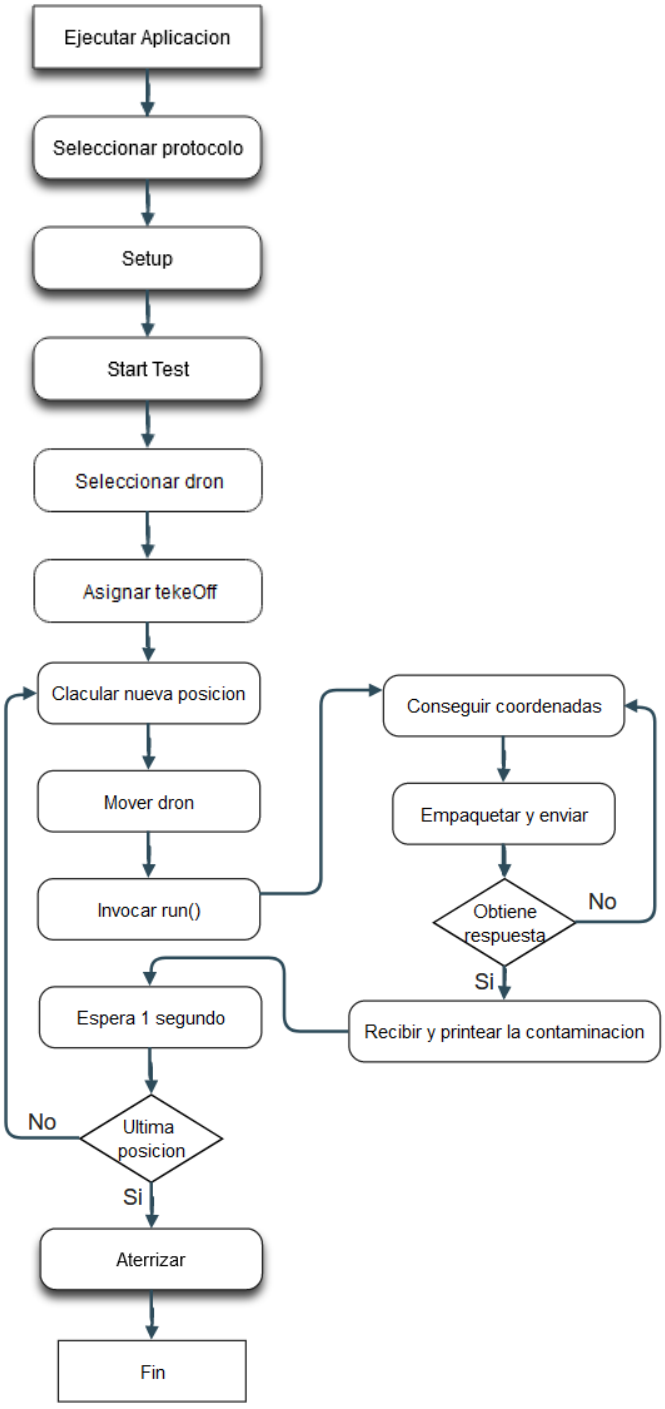

Figura 8: Diagrama de flujo de la ejecución de la aplicación

5. Validación

En este apartado veremos los resultados que se han obtenido en tras el desarrollo del proyecto, dividiendo el análisis en los diferentes apartados cuantitativos de mayor importancia en los que este se enfoca.

Para empezar, debemos comentar las características que tuvo la máquina en la que se trabajó, ya que al cambiar de computadora el comportamiento puede variar. Para la realización del trabajo y del testing se ha hecho uso de una máquina virtual con las siguientes características:

- Procesador de 1 nucleo Intel i5

- 3GB de memoria Ram

- 30 GB de almacenamiento

- 24 MB de memora de video

Partiendo de este punto podemos evaluar el comportamiento que presenta el código con estas condiciones.

5.1. Análisis del tiempo de generación del mapa de calor

El procesamiento de los datos para generar el mapa de calor es sin duda un apartado de necesario estudio, para ello se han tomado diferentes muestras de tiempo que tarda el programa en calcular los datos a devolver al dron. Para ser más precisos, el tiempo medido se comprende entre la recepción de la petición por parte del servidor y la obtención del resultado del método de interpolación.

Los resultados que se han obtenido vienen dados en microsegundos (10^-6), y han sido tomados en muestras de 100. Dado que el peso computacional del método kriging es directamente proporcional al número de puntos conocidos o estaciones establecidas en la simulación, se han realizado mediciones con una cantidad variable de estas.

Haciendo uso de las Figuras 9 y 10, por un lado, observamos variaciones en el tiempo dentro de cada una de las medidas de la con la misma cantidad de estaciones. Las que son de una mayor magnitud, como las que aparecen generalmente en la primera medida podemos atribuirlas a las interrupciones generadas por otros procesos durante la ejecución del vuelo, mientras que las que son de una menor magnitud se tratan de la gestión de la computación parralera ejecutada por el método. Como podemos observar en la figura 10 existe mucha más variación, ya que al tener un mayor número de estaciones se generan con mayor frecuencia problemas de este tipo.

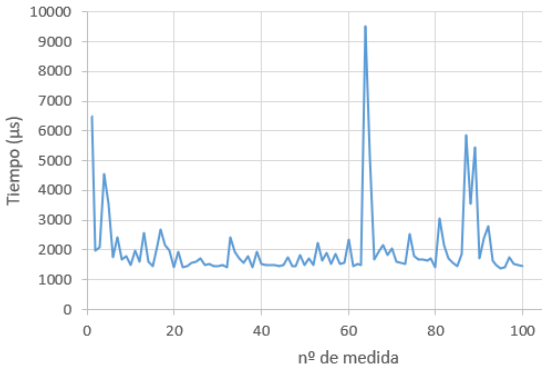

Figura 9: Tiempo que tarda en calcular el valor de contaminación con 6 estaciones

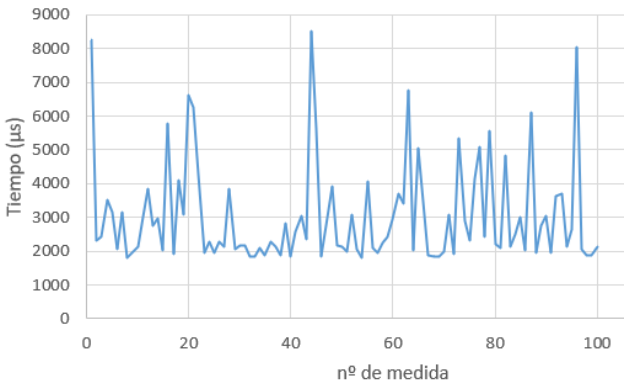

Figura 10: Tiempo que tarda en calcular el valor con 48 estaciones

Además de esto, también podemos ver un aumento en el coste temporal medio. Para poder ver mejor esto se han realizado medidas en con distintas estaciones y podemos observar el crecimiento que se produce en función del número de estaciones en las figuras 11 y 12.

Para evaluar los resultados se han graficado la media (valor ponderado medio de todos los valores) y la mediana (valor central de los valores ordenados). En la primera figura podemos ver el crecimiento progresivo del coste temporal, además se observa la variación que existe entre la media y la mediana. Esto se debe a lo que se comentaba anteriormente, los grandes errores puntuales que se generan debido a la paralelización de los procesos del kriging o de las interrupciones de otros procesos externos.

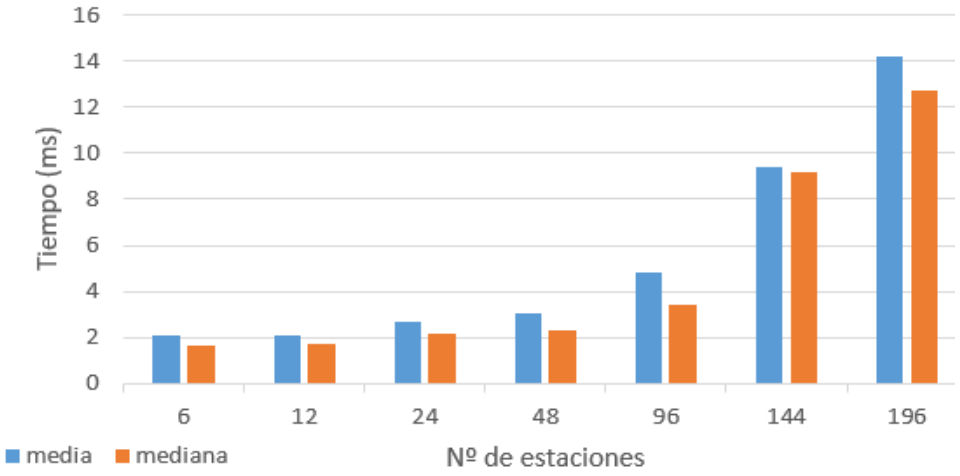

Figura 11: Media y mediana de tiempo según el número de estaciones medidas

Por otro lado, no debemos dejarnos engañar por la figura 11, el crecimiento del coste temporal no es exponencial, este es un efecto visual que se da por la separación exponencial que se le ha dado al incremento de las estaciones medidas. En la figura 12 se puede observar de forma más clara el crecimiento casi lineal que presenta el coste temporal con un pequeño incremento cuando se configuran un elevado número de estaciones.

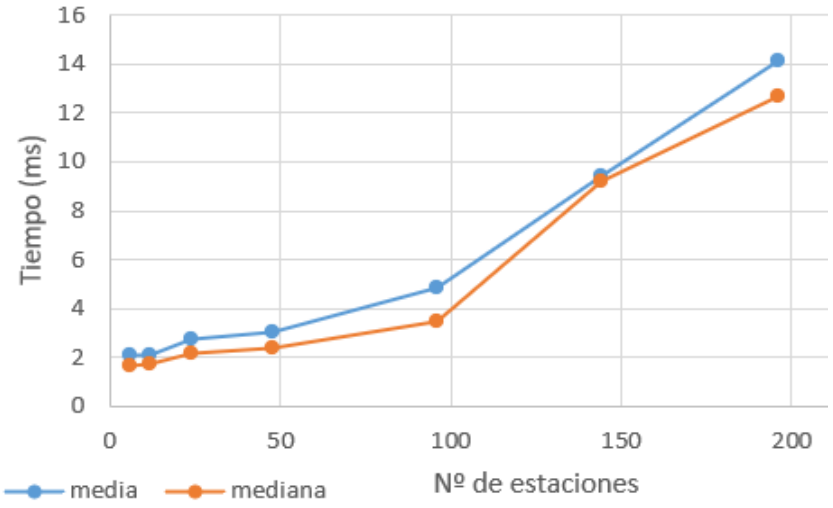

Figura 12: Tendencia de crecimiento de media y mediana

5.2. Análisis del consumo de recursos de cada sensor virtual

Para comprobar la cantidad de recursos consumidos por los sensores virtuales se ha optado por hacer uso de diferentes funciones que incluye Linux para observar estas características, en concreto se ha hecho uso de la orden top.

Este comando de Linux nos ayuda a conocer los procesos de ejecución del sistema en tiempo real entre otras cosas. Entre los datos que nos ofrece caben desatacar los siguientes, los cuales clasificaremos por filas. En las figuras 13 y 14 se ve más clara la estructura que toma la información.

Esta es la información que encontramos en cada fila:

1ª: Hora y usuarios

2º: Tareas activas y el estado en que se encuentran

3º: Consumo de la CPU. Porcentaje del uso que se le dé:

-us: Usuario

-sy: Kernel

-id: Inactivo

-wa: Espera

-hi: Interrupciones hardware

-si: interrupciones software

4º: Memoria física

5º: Memoria virtual

En las columnas posteriores podemos observar los diferentes procesos en ejecución y los porcentajes de CPU y memoria que están consumiendo cada uno de ellos.

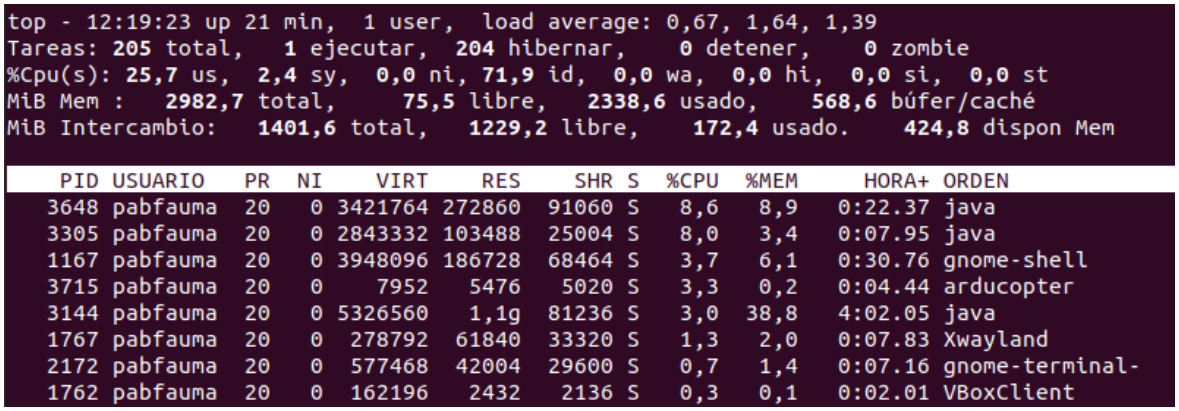

Figura 13: Captura 1 del comando top

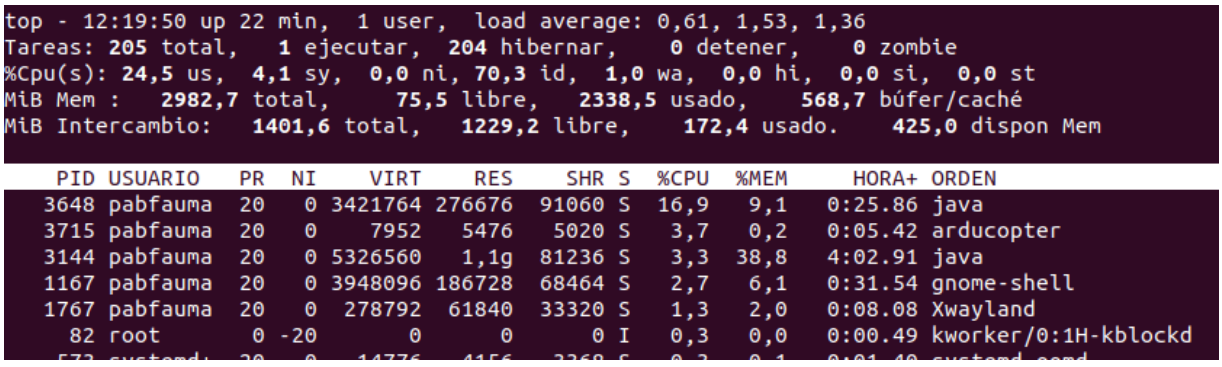

Figura 14: Captura 2 del comando top

En las figuras 13 y 14 podemos ver dos capturas realizadas de la orden top durante la ejecución del vuelo de los drones. En estas se puede observar un par de procesos denominados java que forman parte de ardusim, y estos forman el grueso del consumo de recursos, tanto en CPU (19%) y de RAM (39%). También podemos ver el consumo que genera el servidor de contaminación, denominado gnome-shell, el cual en ambas rondas en un 3-4% de CPU y un 6% de RAM.

El consumo de los drones se observa en el proceso llamado arducopter, y vemos que el consumo de memoria es casi nulo, únicamente hay que tener en cuenta el 4% de CPU que consume su ejecución.

Para finalizar, suponiendo que el aumento de drones multiplique en consumo de los procesos arducopter (dron) y shell (servidor), se produciría un aumento del consumo alrededor de un 8% de CPU por cada dron añadido. Si contamos con un 20% de consumo fijo por parte de la aplicación, se llega a la conclusión de que nuestra máquina puede manejar sin problemas 8 drones simultáneamente.

20 + 8*8 = 84% consumo de CPU

A partir de este número se podría perder agilidad y prestaciones durante la ejecución del vuelo.

5.3. Simulación en un punto fijo: análisis de la variabilidad de los datos generados

En el caso de la variabilidad de los datos resultantes se ha desarrollado en apartado 4.4 la forma en la que se ha realizado, vamos a analizar aquí los resultados de dicho método a la hora de devolver los valores de contaminación a los drones.

Para realizar este estudio se ha optado por tomar las medidas en la simulación de manera previa al despegue del dron. Puesto que la altura no se contempla en las coordenadas enviadas por los drones al servidor, se ha realizado de esta manera la medición para que el movimiento del vehículo por el viento o para estabilizarse no falseen las medidas resultantes.

Para esta, se han tomado 300 medidas sobre el punto de despegue y se han representado en la figura 15.

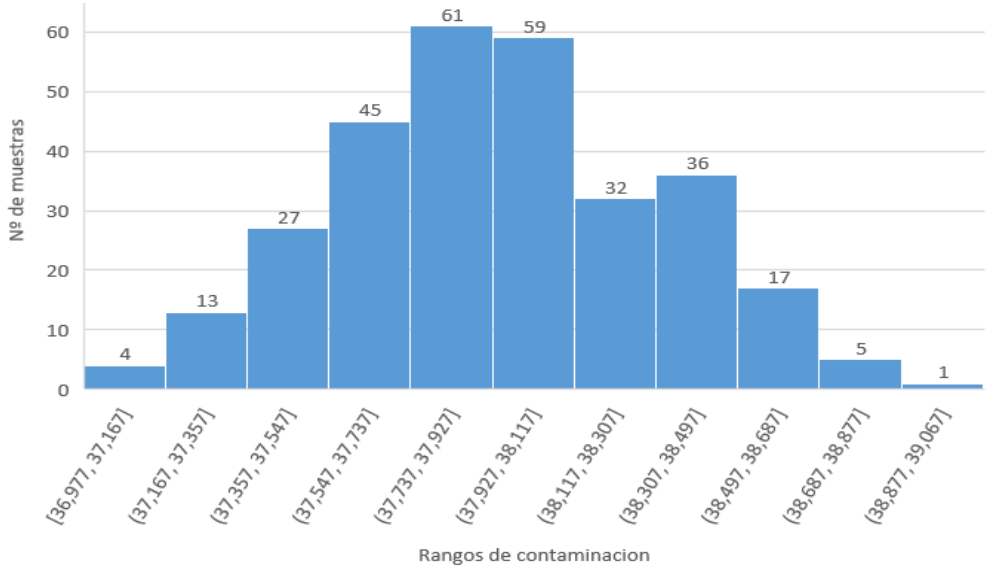

Figura 15: Histograma de valores devueltos. Valor real 37.982

En la figura se han graficado los valores obtenidos sobre el punto mencionado, en el que el valor exacto que devuelve el método kriging es de 37.982. En esta se puede observar la tendencia de la distribución gaussiana que se le ha dado

al generador de error. No es del todo precisa dado que 300 muestras con los valores aleatorios no son suficientes para conseguir un resultado 99% exacto.

Como se observa, se puede ver por el lado derecho un repunte que no debería aparecer en una distribución normal, y, además, una de las características de esta distribución es que la media coincide con el punto central, y en nuestro caso la media resultante de estas medidas es de 37.947.

5.4. Simulación en movimiento: análisis de la tendencia de los resultados

Para finalizar con el apartado de validación tenemos el análisis de la simulación con movimiento de los drones, lo que sería el comportamiento que usualmente tendría la aplicación.

Para probar su funcionamiento se ha programado una trayectoria en la clase Helper de ArduSim que permitirá a un dron desplazarse en zigzag por el plano, tomando muestras de la contaminación cada 15 metros aproximadamente. El comportamiento del dron esta predefinido a la ejecución y este no se verá modificado por los valores que pueda obtener, únicamente se verá afectado en caso de que se produzca algún error en el servidor y este no devuelva la respuesta correspondiente. Solo en este caso el recorrido de la trayectoria se detendrá un tiempo extra, ya que si se da este error el dron debe mantener su posición mientras realiza un total de 3 intentos para obtener respuesta. Si no la obtiene este continuará realizando su siguiente instrucción.

Para obtener el plano de valores en se han configurado 6 estaciones a modo de datos de entrada. Con estas se introducen en el modelo de interpolación, dejando listo al servidor para responder a cada una de las peticiones.

Tampoco olvidemos el generador de ruido, ya que afecta a los valores devueltos. Este no ha de deshabilitarse para dar un mayor realismo y sensación de movimiento al plano de valores.

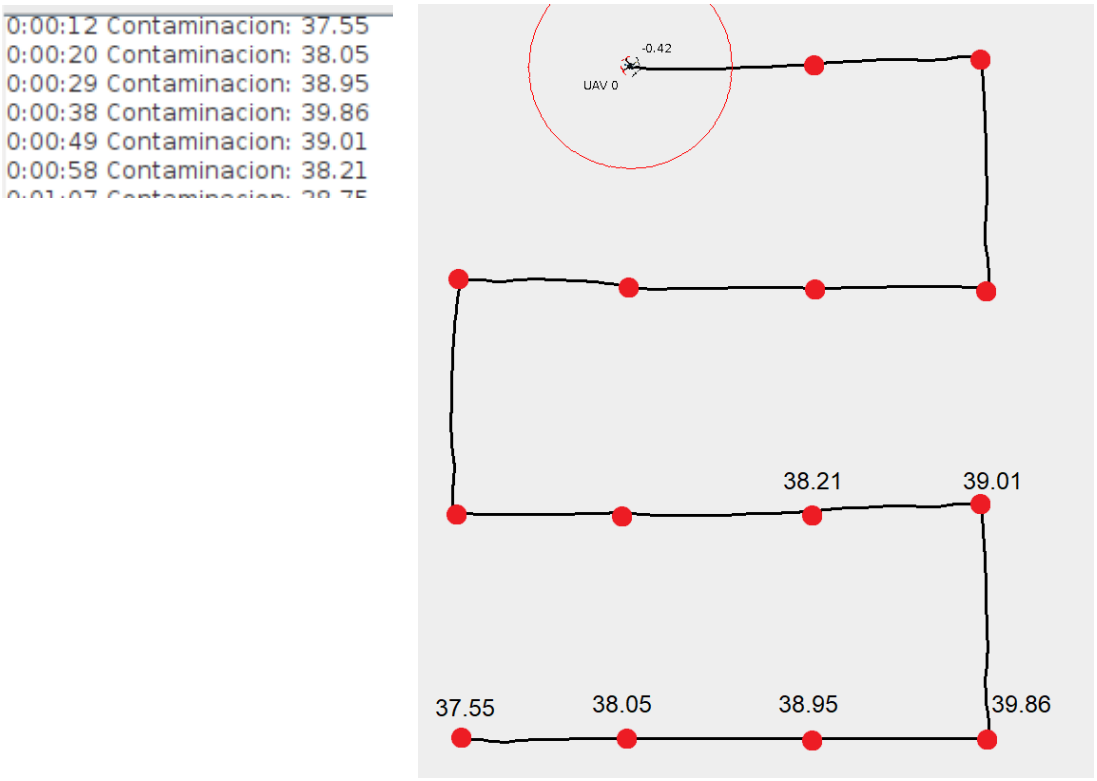

Figura 16: Ejecución con movimiento de los drones

En la figura 16 se puede observar el comportamiento que toma del dron durante un vuelo con el protocolo programado en este proyecto, haciendo uso del sensor de contaminación a medida que se desplaza por el espacio. Con los puntos medidos se puede ver como los valores de contaminación en el plano son menores en la zona Este.

6. Conclusiones

A modo de cierre del trabajo presentamos las conclusiones del proyecto realizado. Puesto que hoy en día la contaminación medioambiental es un tema de gran importancia que nos afecta a todos los seres vivos, debemos trabajar para ponerle solución lo antes posible, antes de generar daños irreversibles al planeta. Dada la tecnología actual, el uso de drones para tomar medidas de contaminación empieza a ser una realidad, pero con este proyecto damos un paso más allá para poder trabajar con ellos de forma simulada.

El resultado más relevante de este proyecto es la creación del nuevo protocolo para la aplicación de ArduSim. Esta aplicación permite realizar la simulación de vuelo de drones, y con el nuevo protocolo se le ha dotado a esta de la posibilidad de simular sensores de contaminación en los propios drones. Esto se ha realizado con un servidor externo a la aplicación que se comunica con cada uno de los drones para darle un valor estimado según fuentes con valores reales y la posición en ese momento de los drones.

Por otro lado, con el fin de alcanzar una simulación verosímil del comportamiento de los sensores, se ha dotado al sistema de métodos generadores de errores, entre los que se encuentran, una leve distorsión en el valor estimado real para generar variación, y un porcentaje de error en la comunicación entre los drones y el servidor. A parte, en este último caso, se ha implementado la forma de darle solución.

A pesar de todo lo comentado, se debe seguir investigando para obtener el máximo provecho a este proyecto, ya que por ahora solo se ha alcanzado la posibilidad de realizar las simulaciones de los sensores. En un futuro, se puede hacer uso de esto para estudiar algoritmos de decisión que permitan a los drones tomar su propio rumbo, con objetivos tales como la búsqueda de focos contaminantes en los que, tras ser detectados, se puedan tomar medidas para reducir su impacto. A largo plazo, de puede llegar a disponer de una flota de drones que permitan sobrevolar ciudades, o zonas industrializadas con gran contaminación, de manera que se encuentren los mayores focos de emisión y se tomen medidas para reducirlos.

Para finalizar, un resultado inesperado que se podría llegar a obtener es el uso de los drones con sensores con el fin de tomar mediciones en volcanes en activos. Esto es un tema donde se encuentran drones de manera más común, sin embargo, los algoritmos implementados con nuestro simulador también podrían llegar a ser de utilidad para este fin.

Bibliografía

[1] Modelos de krigging
https://pro.arcgis.com/es/pro-app/2.8/tool-reference/3d-analyst/how-kriging-works.htm

Accedido: 14 de mayo de 2022

[2] Funcionamiento de kriging
https://acolita.com/geoestadistica-interpolacion-con-kriging/

Accedido: 14 de mayo de 2022

[3] Estructura servidor UDP en Python
https://rico-schmidt.name/pymotw-3/socket/udp.html

Accedido: 28 de abril de 2022

[4] Estructura servidor UDP en Java
http://www.it.uc3m.es/celeste/docencia/cr/2003/PracticaSocketsUDP/

Accedido: 30 de abril de 2022

[5] Información de la contaminación ambiental
https://aqicn.org/map/spain/es/

Accedido: 2 de julio de 2022

[6] Github de la aplicación ArduSim
https://github.com/GRCDEV/ArduSim

Accedido: 25 de julio de 2022

[7] Articulo de Sincratech sobre el uso de drones para el control de la calidad del aire

https://sincratech.com/tecnologia/el-uso-de-drones-para-el-control-de-la-calidad-del-aire/

Accedido: 18 de julio de 2022

[8] Artículo publicado en Mongabay sobre como los drones pueden ayudar a medir la contaminación ambiental

https://es.mongabay.com/2017/01/drones-qaira-contaminacion/

Accedido: 18 de julio de 2022

[9] Articulo de Rpasdrones sobre una tesis realizada en Grecia sobre el uso de drones para medir la contaminación

https://www.rpas-drones.com/drones-3d-contaminantes-aire/

Accedido: 18 de julio de 2022

[10] Funcionamiento del comando top

https://geekytheory.com/funcionamiento-del-comando-top-en-linux/

Accedido: 26 de julio de 2022

Documentación de librerías utilizadas:

-En Python:

[11] json

https://docs.python.org/es/3/library/json.html

Accedido: 13 de abril de 2022

[12] csv

https://docs.python.org/es/3/library/csv.html

Accedido: 14 de abril de 2022

[13] pykrige

https://geostat-framework.readthedocs.io/projects/pykrige/en/stable/#

Accedido: 22 de julio de 2022

[14] socket

https://docs.python.org/es/3.10/library/socket.html

Accedido: 14 de mayo de 2022

-En Java:

[15] java.net

https://docs.oracle.com/javase/7/docs/api/java/net/package-summary.html

Accedido: 14 de mayo de 2022

[16] socket

https://docs.oracle.com/javase/7/docs/api/java/net/Socket.html

Accedido: 15 de mayo de 2022

Anexo I: Grado de relación del trabajo con los Objetivos de Desarrollo Sostenible (ODS).

Objetivos de Desarrollo Sostenibles	Alto	Medio	Bajo	No Procede
ODS 1. **Fin de la pobreza.**				X
ODS 2. **Hambre cero.**				X
ODS 3. **Salud y bienestar.**		X		
ODS 4. **Educación de calidad.**				X
ODS 5. **Igualdad de género.**				X
ODS 6. **Agua limpia y saneamiento.**				X
ODS 7. **Energía asequible y no contaminante.**				X
ODS 8. **Trabajo decente y crecimiento económico.**			X	
ODS 9. **Industria, innovación e infraestructuras.**			X	
ODS 10. **Reducción de las desigualdades**.				X
ODS 11. **Ciudades y comunidades sostenibles**.		X		
ODS 12. **Producción y consumo responsables.**	X			
ODS 13. **Acción por el clima.**		X		
ODS 14. **Vida submarina.**				X
ODS 15. **Vida de ecosistemas terrestres.**			X	
ODS 16. **Paz, justicia e instituciones sólidas.**				X
ODS 17. **Alianzas para lograr objetivos.**				X

En este apartado se trata de vincular el desarrollo de nuestro proyecto con los 17 objetivos de desarrollo sostenible declarados por la ONU. Estos consisten en 17 objetivos los cuales se pretenden alcanzar en 15 años con el impulso coordinado de todos los gobiernos. Fueron establecidos en el año 2015 y se enmarcan en la llamada agenda de 2030.

De entre todos los ODS, aquellos en los que se centraría este proyecto serán en “Adoptar medidas para combatir el camio climático y sus efectos” y en “Garantizar modalidades de consumo y producción sostenibles”.

Puesto que este proyecto trata del diseño e implementación de un protocolo para una aplicación de simulación de vuelo de vehículos no tripulados que consiste en dotar de sensores de contaminación a los drones, se puede vincular al primero mencionado ya que si se desarrolla más a futuro creando una flota de drones que busquen focos de contaminación se podrán usar para tomar medidas contra estos y así reducir la contaminación emitida afectando favorablemente al cambio climático. Y respecto al segundo se puede enlazar ya que al tratarse de una simulación se obtiene un ahorro de recursos materiales y energéticos que si se tratasen de implementar de manera física para estudiar el comportamiento se malgastarían en algo innecesario.