Trabajo de Fin de Máster
Máster en Ingeniería Industrial

Digitalización del proceso de localización de fugas de gas en estaciones depuradoras mediante drones

Autor: Rafael Luque Berraquero
Tutor: Jesús Capitán Fernández

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla

Sevilla, 2021

Trabajo de Fin de Máster
Máster en Ingeniería Industrial

Digitalización del proceso de localización de fugas de gas en estaciones depuradoras mediante drones

Autor:
Rafael Luque Berraquero

Tutor:
Jesús Capitán Fernández
Profesor Contratado Doctor

Dpto. de Ingeniería de Sistemas y Automática
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Trabajo de Fin de Máster: Digitalización del proceso de localización de fugas de gas en estaciones depuradoras mediante drones

Autor: Rafael Luque Berraquero

Tutor: Jesús Capitán Fernández

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

El Secretario del Tribunal

A mi familia.

A mis tutores.

A mis compañeros.

Agradecimientos

Primero de todo, quiero agradecer a mi familia por su apoyo incondicional a lo largo de los años y su ánimo en los buenos momentos, pero sobre todo en los difíciles.

Dar las gracias a mi tutor, Jesús Capitán, por su disposición a ayudarme y aconsejarme en todo lo que he necesitado. Y al resto de mis profesores porque de una manera u otra todos tienen una parte en este documento.

Con todo ello mencionar y dar las gracias a mis compañeros y amigos por formar parte de este camino en el que hemos demostrado que con esfuerzo y dedicación nos podemos superar.

Rafael Luque Berraquero

Sevilla, 2021

.

Resumen

Las emisiones fugitivas procedentes de plantas industriales de múltiples sectores representan un problema tanto para la salud y la seguridad de las personas como para una parte importante de las emisiones mundiales de gases de efecto invernadero. Concretamente, las emisiones de olores procedentes de las Estaciones Depuradoras se consideran la principal causa de las molestias percibidas por los ciudadanos que viven cerca de las instalaciones. Estas emisiones son difíciles de detectar y requieren de procesos que necesitan mucho tiempo y altos costes de ejecución.

En este trabajo, se propone el uso de robótica móvil olfativa para abordar este problema. La simulación de las fugas de gases y el movimiento autónomo de robots aéreos en entornos realistas, junto con el desarrollo de algoritmos de alto nivel para la localización de estas fuentes de emisiones y una Plataforma IoT (Internet de las Cosas) para la monitorización en tiempo real, son los tres pilares sobre los que se fundamenta la solución desarrollada. Como resultado, la integración de estas tres tecnologías representará un caso de uso de éxito en el proceso de transformación digital hacia una Industria 4.0.

Abstract

Fugitive emissions from industrial plants in numerous sectors represent a problem both for human health and safety as a significant part of global greenhouse gas emissions. In particular, odour emissions from Wastewater Treatment Plants are considered to be the main cause of annoyance perceived by citizens living close to the facilities. These emissions are difficult to detect and require time-consuming processes and high implementation costs.

In this project, the use of Mobile Robot Olfaction is proposed to address this problem. The simulation of gas leaks and movement of Unmanned Aerial Vehicles in realistic environments, together with the development of high-level algorithms for the localisation of these emission sources and an IoT Platform (Internet of Things) for real-time monitoring, are the three pillars on which the developed solution is based. As a result, the integration of these three technologies will represent a successful use case in the process of digital transformation towards Industry 4.0.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN

En este primer apartado, por un lado, se presenta de manera general el problema que se pretende abordar a través de la ejecución de este proyecto. Así como también, se plantea el enfoque adoptado para desarrollar el trabajo y los objetivos que se pretenden alcanzar al final del mismo. Para lograr este propósito, se exponen finalmente las diferentes tecnologías o herramientas que se van a utilizar para el despliegue de la solución.

1.1 Motivación

Emisiones fugitivas y gases de efecto invernadero

El IPCC (Intergovernmental Panel on Climate Change) define las emisiones fugitivas [1] como "las emisiones que no se producen intencionadamente a través de una chimenea o un respiradero" y estipula que pueden "incluir las fugas de las plantas industriales y las tuberías".

Esta definición puede variar de un sector a otro. Mientras que, en el sector de los combustibles fósiles, las emisiones fugitivas se definen de forma amplia como cualquier emisión no relacionada con el uso final del combustible, en la contaminación atmosférica, una emisión fugitiva puede definirse como la liberación de contaminantes a la atmósfera libre después de que hayan escapado a un intento de capturarlos con una campana, un sello o cualquier otro medio dedicado a garantizar la captura y retención de estos contaminantes. Es decir, no existe una definición estable y universal de las emisiones fugitivas. En la práctica, se suelen incluir: (i) las emisiones accidentales debido a falta de mantenimiento, fin de vida útil o deterioro de elementos críticos (en la Figura 1-1 se puede observar un ejemplo emisiones debido a rotura de tuberías), (ii) las fugas y los escapes difusos (válvulas, bridas o juntas de conexión defectuosas, migración de gas a la superficie cerca de pozos o minas, emisiones de pozos abandonados) y (iii) los vertidos involuntarios, pero no productivos (ventilación, quema en antorcha, desgasificación).

Figura 1-1. Localización manual de fuga peligrosa de gas en planta.

La propia naturaleza de estas emisiones hace que sean difíciles de cuantificar, y aunque no existen datos globales completos, las emisiones fugitivas representan una parte importante de las emisiones mundiales de gases de efecto invernadero [2].

El control medioambiental, la vigilancia de las emisiones y otras tareas relacionadas con la detección de gases han adquirido hoy en día una mayor importancia debido a la creciente preocupación por la salud y el medio ambiente. Por ejemplo, muchas ciudades desarrolladas de todo el mundo están impulsando una mejor infraestructura de control de la contaminación, mientras que los productores de biogás, como los vertederos,

están adoptando sistemas de vigilancia de las emisiones de metano (CH4) más estrictos. Estas medidas se derivan del deterioro palpable de la calidad del aire interior y urbano como consecuencia del creciente número de vehículos de combustión, y del hecho de que nuevos estudios médicos revelan continuamente los efectos negativos de estos contaminantes en la salud humana. El elevado impacto del CH4 en el calentamiento global, y las considerables cantidades de emisiones fugitivas que pasan desapercibidas debido a las escasas rutinas de medición han impulsado a diferentes instituciones gubernamentales, como la EPA (Agencia de Protección Medioambiental de Estados Unidos), a buscar y desarrollar mecanismos eficaces de control de las emisiones.

Estaciones Depuradoras de Aguas Residuales

Como se ha comentado, las emisiones fugitivas se producen principalmente durante los procesos de extracción, transporte, almacenamiento y transformación presentes en las actividades industriales de multitud de sectores: plantas de gas natural, plantas petroleras, plantas de carbón, refrigeración doméstica (HFC, CFC), electricidad (SF6), salud (N2O), tratamiento de aguas residuales, en las que se centra el presente trabajo, entre otras, etc.

Estas plantas de tratamiento de aguas residuales, denominadas EDAR (Estación Depuradora de Aguas Residuales), como la que se muestra en la Figura 1-2, son plantas dedicadas a la depuración y cuya función básica es recoger las aguas de una población o industria, y reducir la contaminación mediante ciertos tratamientos y procesos, para devolverlas a un cauce receptor como un río, embalse, mar, etc.

A lo largo de las diferentes etapas de tratamiento en las EDARs como son los tanques de aireación, tanques de digestión de lodos, planta de desodorización, estaciones de bombeo, tratamiento de fangos, digestores anaerobios, entre otras, se generan una serie de gases contaminantes que corren el riesgo de acabar siendo expulsados a la atmósfera. Entre los diferentes gases que se generan se pueden encontrar:

- El cloro que se usa a menudo en la purificación de agua.
- El monóxido de carbono (CO) y el dióxido de carbono (CO2) que se liberan de la descomposición de material orgánico.
- El metano, que se emite durante el manejo y tratamiento de las aguas residuales municipales a través de la descomposición anaeróbica de material orgánico.
- Y el sulfuro de hidrógeno (H2S), un gas incoloro, conocido por su olor a huevo podrido que se produce por la reducción biológica de sulfatos y la descomposición de materia orgánica.

De hecho, las emisiones de olores de las estaciones depuradoras de aguas residuales, ocasionadas principalmente por este último gas, H2S, se consideran una de las principales causas de las molestias percibidas por los ciudadanos que viven cerca de las instalaciones [3], afectando a la calidad de vida.

Efectos adversos en la salud

Y aunque casi nunca se asocia un verdadero riesgo toxicológico-sanitario al impacto de las emisiones de olores, lo cierto es el gas es una amenaza silenciosa, a menudo invisible para los sentidos del cuerpo, siendo la inhalación la principal vía de exposición al sulfuro de hidrógeno. Aunque algunas personas pueden olerlo fácilmente en pequeñas concentraciones, la exposición continuada a niveles incluso bajos de H2S amortigua rápidamente el sentido del olfato. Tras la exposición, el H2S irrita, entre otras cosas, las mucosas del cuerpo y las vías respiratorias, y entre los síntomas a corto plazo se pueden incluir dolor de cabeza, náuseas, convulsiones e irritación de los ojos y la piel [4]. En concentraciones elevadas, bastan unas pocas respiraciones para inducir la inconsciencia, el coma, la parálisis respiratoria, las convulsiones e incluso la muerte.

Por otro lado, el H2S, al ser más pesado que el aire, se acumula en zonas bajas de espacios mal ventilados. Este gas en presencia de aire y humedad puede formar ácido sulfúrico, capaz de corroer los metales, de tal manera que los equipos de las instalaciones, incluidas las superficies internas de varios componentes, se enfrentan a una reducción de la durabilidad y la resistencia al impacto, lo que puede provocar un fallo prematuro.

Mantenimiento de las instalaciones

Las actividades de mantenimiento se han convertido en una función crítica y esencial para las actividades industriales en general, que va más allá de la mera conservación y reparación al permitir la optimización de la gestión de recursos, incluyendo actividades relativas a seguridad, medio ambiente, calidad y productividad,

mejorando la gestión de los activos y reduciendo riesgos en las plantas.

Las emisiones fugitivas son difíciles de detectar y requieren de procesos que necesitan mucho tiempo y altos costes de ejecución. La naturaleza particular y complejidad de la mezcla de las sustancias volátiles, su variabilidad en el tiempo, la fuerte influencia de las condiciones atmosféricas y la subjetividad de la percepción del olor son los elementos que redundan en menores frecuencias de ejecución de los programas de inspección y por tanto actuaciones enfocadas a la corrección y no a la prevención.

Las tecnologías actuales empleadas para la detección de fugas se basan en el empleo de equipos de cuantificación de las emisiones con detectores manuales de contacto, como los denominados "sniffers" como el FID (del inglés, Flame Ionization Detector) o el PID (del inglés, Photoionization Detector). Estas técnicas requieren de una supervisión detallada con una alta dedicación de personal debido a las extensas superficies a inspeccionar y los riesgos para los operarios por la cercanía a las potenciales fuentes de emisión.

Otras técnicas empleadas son la visualización óptica del biogás utilizando un analizador infrarrojo de gases. Los gases como el metano, componente mayoritario del biogás, absorben radiación infrarroja en distintos rangos, lo que excita las moléculas permitiendo su visualización y fácil detección mediante cámaras de infrarrojos.

A la vista de todos los problemas existentes, en este sentido, la robótica móvil puede contribuir con sistemas adaptados para la automatización de los procedimientos de monitorización de emisiones. La robótica móvil olfativa (MRO, del inglés, Mobile Robot Olfaction) [5] es la rama de la robótica que se ocupa de la integración de sensores de gases y químicos a bordo de plataformas móviles comprometiendo a sistemas que pueden adquirir mediciones con una alta resolución espaciotemporal sin exponer a los operadores a entornos peligrosos, donde, como se ha dicho, pueden estar presentes altas concentraciones de contaminantes químicos. Además, las capacidades computacionales de los sistemas MRO pueden utilizarse, por ejemplo, para crear mapas de las distribuciones de los gases, planificar recorridos de vigilancia de cobertura total, discriminar entre diferentes volátiles o inferir la localización de fugas de gas.

Figura 1-2. Vista aérea de la EDAR Copero, perteneciente a la empresa sevillana EMASESA.

1.2 Objetivos

El enfoque adoptado en este proyecto para solucionar la problemática descrita en el apartado anterior se basa en el concepto de Gemelo Digital como tecnología habilitadora hacia la transformación digital vinculada a la Industria 4.0 o Industria Conectada.

Como es bien sabido, el término de Industria 4.0 se refiere a la llamada cuarta revolución industrial identificada principalmente por la transformación digital de la industria mediante la integración y digitalización de todos los procesos industriales que componen la cadena de valor, y caracterizada por su adaptabilidad, flexibilidad y eficiencia, permitiendo cubrir las necesidades de los clientes en el mercado actual.

Partiendo de estos conceptos, el objetivo que se plantea para este trabajo consiste en desarrollar el gemelo digital de un proceso, en este caso concretamente, el de la localización autónoma de fugas de gas mediante un UAV (Unmanned Aerial Vehicle) en una planta de tratamiento de aguas residuales, por medio de la integración de diferentes tecnologías. Como resultado, se obtiene una herramienta que facilita el desempeño de esta tarea en la planta real, garantizando la optimización, así como el seguimiento en tiempo real del proceso y, más importante aún, la eliminación de los riesgos asociados. Finalmente, los resultados obtenidos servirán como caso de uso de éxito en el proceso de transformación digital hacia una Industria 4.0.

Para terminar, al mismo tiempo que se puede comprobar en la Figura 1-3, se exponen los tres pilares técnicos sobre los que se apoya la solución desarrollada:

1. Simulación: se incluye tanto el simulador de fugas de gas en entornos realistas como el robot de localización autónoma, en este caso un UAV.
2. Algoritmos de localización de fugas de gas.
3. Plataforma IoT para el seguimiento en tiempo real.

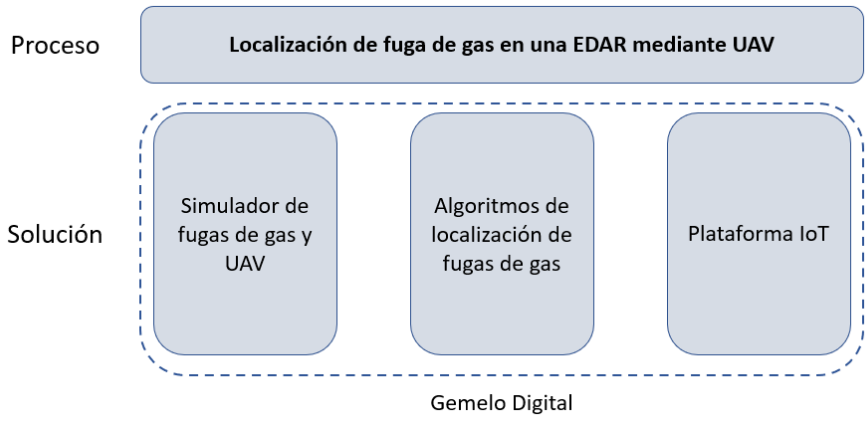


Figura 1-3. Esquema conceptual de los pilares tecnológicos sobre los que se construye la solución al problema.

2 ESTADO DEL ARTE

En esta sección se realiza un estudio del estado del arte sobre las áreas fundamentales de este proyecto mencionadas anteriormente. Este estado del arte aporta diversos conocimientos sobre el tema de investigación al leer la literatura relacionada, aprendiendo de otros investigadores, demostrando el grado de relevancia del proyecto y mostrando los diferentes enfoques que se aplican a esta misma solución.

2.1 Simuladores de gas

El desarrollo de sistemas MRO no es un problema trivial y, a pesar de los recientes logros, el potencial de los robots móviles sensibles a los gases aún no se ha aprovechado plenamente. Un obstáculo clave es la falta de conjuntos de datos adecuados que puedan utilizarse para realizar evaluaciones del proceso real. Este inconveniente tiene su origen en la gran dificultad asociada a la realización de experimentos en entornos reales, y en la actual limitación tecnológica para obtener información fiel de los gases liberados y su distribución. La recogida de datos es un proceso complejo y que requiere mucho tiempo, incluso dentro de arenas robóticas dedicadas, donde el mayor reto proviene de la gran complejidad del fenómeno de la dispersión de gases, en el que entran en juego muchas variables ambientales y topológicas. Además, este fenómeno, es muy susceptible a pequeños cambios en estas variables, lo que a menudo provoca variaciones sustanciales en los conjuntos de datos recogidos provocando que la repetibilidad entre experimentos sea difícil de conseguir.

Por estos motivos, es muy importante disponer de herramientas de evaluación adecuadas para facilitar el desarrollo de nuevos algoritmos, sensores y plataformas que conforman los sistemas MRO. En este contexto, las herramientas de simulación con capacidad para manejar adecuadamente el fenómeno de la dispersión de gases pueden utilizarse para realizar evaluaciones exhaustivas antes de pasar a las pruebas experimentales en el mundo real. En la literatura se han desarrollado y presentado diferentes enfoques, que van desde modelos simplistas basados en penachos de forma gaussiana hasta sofisticados modelos de dinámica de fluidos que consideran variables topológicas y ambientales en el proceso de cálculo. Al mismo tiempo, mientras que existen algunas implementaciones de frameworks de simulación de dispersión de gases orientados a la robótica, otros sólo consideran entornos simplificados, están desarrollados en plataformas de software robótico obsoletas o dependen de paquetes de software externos y costosos.

La simulación de la dispersión de gases (GDS, por sus siglas en inglés) es una tarea que ha sido abordada por diferentes disciplinas como la meteorología y la informática. Se han descrito varias taxonomías para clasificar los diferentes enfoques, por ejemplo, según la escala espacial (es decir, modelos macro y micro), o según la naturaleza del modelo de dispersión.

Según esta última taxonomía, los modelos simplistas de GDS pueden clasificarse como modelos de caja, gaussianos y analíticos [6]. Estos modelos hacen fuertes suposiciones sobre las propiedades espaciotemporales subyacentes de la distribución del gas. Por ejemplo, los modelos de caja asumen que la dispersión del gas puede discretizarse como una serie de cajas de concentración homogénea de gas. Los modelos gaussianos suponen que la dispersión del gas sigue una distribución de probabilidad normal. Y los enfoques analíticos utilizan modelos empíricos regulados por una serie de coeficientes, que se fijan según los datos de los experimentos. Debido a su simplicidad, estos modelos son computacionalmente baratos, pero no son muy precisos en sus estimaciones.

Los enfoques más sofisticados del GDS incluyen los modelos lagrangianos, eulerianos y basados en la dinámica de fluidos computacional (CFD, del inglés Computational Fluid Dynamics). Los modelos lagrangianos y eulerianos simulan la dispersión de gases como el movimiento aleatorio de un gran número de partículas, afectadas por el flujo del viento y los campos de turbulencia. La diferencia más importante entre estos dos modelos es que los modelos eulerianos emplean una malla cartesiana tridimensional fija como marco

de referencia en lugar de un marco móvil, como ocurre en los modelos lagrangianos. Su principal desventaja es el alto coste computacional cuando el número de partículas aumenta.

Los modelos basados en CFD, por otro lado, resuelven ecuaciones tridimensionales para el viento, la temperatura, la humedad y las concentraciones de gas. Estos modelos se utilizan para simulaciones que requieren una granularidad espaciotemporal elevada en la que los obstáculos físicos, como los objetos y las paredes, se representan explícitamente.

Existen varias implementaciones de los enfoques GDS mencionados en paquetes comerciales y de código abierto, por ejemplo, ANSYS y OpenFoam. Aunque estos paquetes de software pueden producir sofisticados modelos de dispersión, no permiten la posibilidad de incluir simultáneamente plataformas robóticas y mecanismos de detección de gases. En esta sección, se describen diferentes frameworks que sí permiten combinar simuladores de robótica móvil y GDS.

Plumesim es un ejemplo de framework de simulación orientado a aplicaciones de robótica (ver Figura 2-1). Plumesim está basado en Player/Stage [7], y permite la simulación de la dispersión de gases empleando modelos teóricos simplistas como los penachos gaussianos, mediciones adquiridas en experimentos del mundo real, o utilizando datos externos generados con herramientas CFD. Aunque Plumesim es una herramienta de simulación flexible, está limitada por los modelos muy simplistas utilizados para la dispersión de gases, la falta de apoyo a los obstáculos y el hecho de que, cuando se utilizan datos CFD, la dispersión de los gases volátiles químicos también debe ser calculada por la herramienta CFD externa, lo que aumenta notablemente la complejidad de la simulación, requiriendo algunos conocimientos más avanzados de CFD.

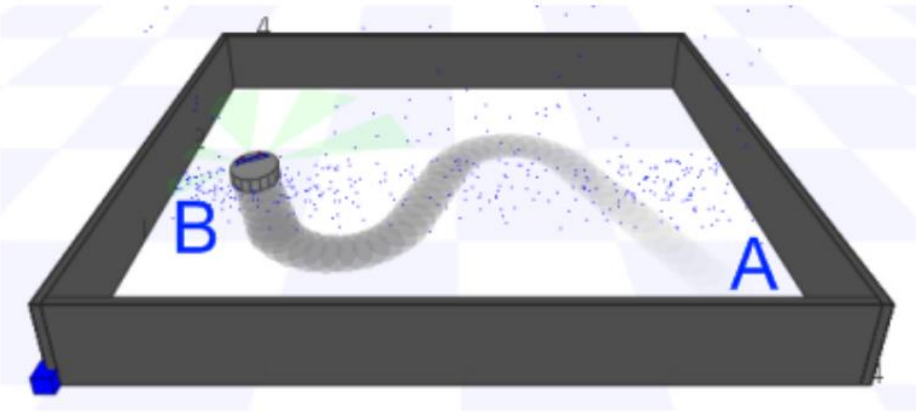


Figura 2-1. Visualización del frameowork PlumeSime en el que se observa la simulación de la dispersión de gases junto con un robot. [8].

Más recientemente, Monroy et al. [9] propusieron un marco GDS para aplicaciones robóticas en el sistema operativo robótico OpenMORA. Esta herramienta consiste en una aplicación C++ basada en una serie de módulos construidos utilizando el Mobile Robot Programming Toolkit (MRPT). Como entrada, el marco emplea instantáneas de dispersión de gas calculadas con una herramienta externa, siendo capaz de simular la dispersión dinámica de múltiples fuentes de gas. Los principales inconvenientes de este simulador son que OpenMORA carece del soporte proporcionado por marcos más extendidos como ROS (Robot Operating System), los datos del viento no están soportados, y la dispersión de gases debe ser calculada por herramientas CFD externas, algo que dificulta la flexibilidad del simulador (por ejemplo, cambiar la ubicación de una fuente de gas requiere volver a calcular todos los datos CFD, lo que es computacionalmente caro). En la Figura 2-2 se muestra la interfaz gráfica disponible para este framework.

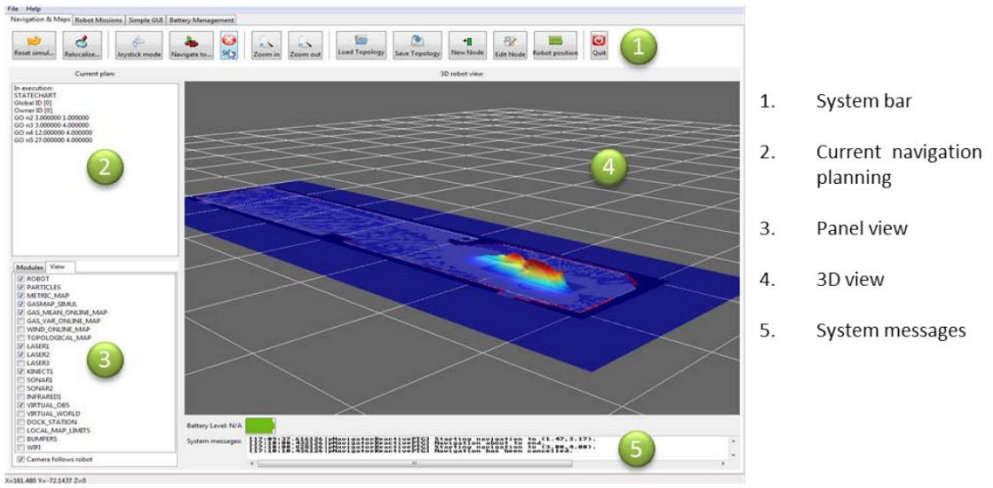


Figura 2-2. Visualización de la GUI procedente del framework propuesto por Monroy et al. [10].

Posteriormente, se presentó también un framework de simulación de dispersión de gas [11], desarrollado en ROS, el cual tiene una estructura modular que permite la simulación de mecanismos de detección de gas, obstáculos físicos y diferentes especies químicas. La dispersión de gas se lleva a cabo utilizando un modelo basado en filamentos y los patrones de flujo de viento se generan externamente utilizando herramientas de dinámica de fluidos como OpenFoam. En este caso, el simulador solo soporta una fuente emisora, la definición de la topología del entorno no es sencilla (no admite modelos CAD) y el rendimiento de la simulación se reduce considerablemente a medida que transcurre el tiempo de simulación.

De manera similar, Webots [12] también es un framework de simulación robótica de alta fidelidad de código abierto y con soporte en ROS que dispone de un plugin de simulación de olores que utiliza un modelo de dispersión atmosférica basado en filamentos propuesto por Farrell et al. [13]. Es decir, el olor se simula como filamentos en el aire, que representan paquetes de olor de cierto tamaño y concentración (la Figura 2-3 muestra la simulación de estos paquetes de olor con forma hexagonal). El plugin de simulación consta de cinco componentes principales: un modelo de viento, un modelo de propagación de filamentos de olor, un modelo de fuente de olor, así como un sensor de viento y un modelo de sensor de olor. El flujo de viento simulado es cuasi laminar y estacionario en su intensidad y dirección, es decir, no hay meandros ni ráfagas de viento.

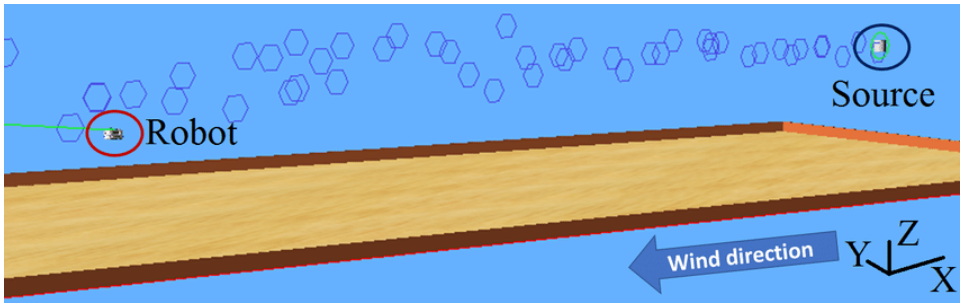


Figura 2-3. Entorno de simulación en Webots, con la fuente a favor del viento, el robot a favor del viento y las zonas de gas representadas con hexágonos azules [14].

Para terminar, GADEN es un framework de simulación diseñado para sistemas robóticos móviles y algoritmos de detección de gases. El framework se basa en los principios de la dinámica de fluidos computacional y la teoría de dispersión de filamentos, modelando el flujo de viento y la dispersión de gases en escenarios 3D y teniendo en cuenta las paredes, los muebles y otros objetos que pueden tener un impacto significativo en la dispersión de gases. Además, integra la simulación de diferentes sensores ambientales, como los sensores de gas de óxido metálico, los detectores de fotoionización o los anemómetros, y está totalmente integrado con ROS y la pila de navegación, lo que facilita enormemente las pruebas y la validación

Hay que señalar que ninguno de los trabajos mencionados anteriormente, tienen en cuenta las perturbaciones causadas por el robot móvil al atravesar el entorno de simulación. La solución directa, ejecutar la simulación

CFD en tiempo real, está limitada por la potencia de cálculo disponible, no siendo una solución factible en la mayoría de los escenarios prácticos. No obstante, trabajos recientes se han centrado en proporcionar una aproximación en tiempo real al problema modelando las interrupciones del flujo de aire causadas, por ejemplo, por los cuadricópteros, y luego integrando estas interrupciones en los datos de dispersión de gases previamente simulados. Aunque este enfoque parece atractivo, la integración de estos modelos de disrupción en los motores de dispersión de gases y de simulación robótica sigue siendo una cuestión de investigación abierta.

2.2 Algoritmos de localización de fugas de gas

El proceso de localización de fuentes de olor consiste en encontrar la posición de una fuente química de interés en el ambiente a través de una pista, es decir, un penacho creado por la fuente, influenciado por el viento. Para la mayoría de los organismos vivos, la localización de una fuente de olor es una capacidad innata, por ejemplo, ayuda a los animales a encontrar a sus parejas, a localizar su comida o sus presas, a evitar sus sustancias químicas desfavorables, siendo de diferente nivel entre las distintas especies (los perros tienen un olfato 108 veces más sensible que el del ser humano a algunos compuestos). Sin embargo, la capacidad de los animales no es suficiente para algunas situaciones, como la localización de fugas en tuberías submarinas o cuando se trata de fuentes de olor que son demasiado peligrosas para que los seres vivos se acerquen a ellas (como el gas venenoso, el fuego y las minas sin explotar). A raíz de estas situaciones, donde la búsqueda del objetivo es una tarea arriesgada tanto para los animales como para los seres humanos, es donde surge la idea de utilizar una solución robótica.

Desde principios de los años 90, muchos investigadores se han dedicado a construir un robot (o un sistema robótico que incluya varios robots) que sea capaz de localizar una fuente de liberación química. En muchos de los primeros trabajos, la localización robótica de fuentes de olor no funcionaba tan bien como sus equivalentes biológicos y es que existe un factor que no se puede olvidar: la pluma de olor se propaga a través del viento, que es turbulento. Después de salir de la fuente, el penacho se ensancha rápidamente y luego se expande a un ritmo suave y constante para formar una nube discreta y difusa sin una forma fija [15]. Desde un punto de vista más científico, esta estructura se debe a la dispersión atmosférica. Cuando las moléculas de olor son alejadas de su fuente por el viento, se generan penachos en los que suceden dos procesos: uno es la difusión molecular (las moléculas se mueven al azar y se separan gradualmente) y el otro es la difusión turbulenta en la que la turbulencia del viento despedaza los penachos. La primera es lenta y a pequeña escala, mientras que la segunda ocurre a una escala temporal y espacial mucho mayor, lo que juega un papel más importante en la formación de los penachos. En la figura 2-4 se muestra la forma de un penacho de gas, más o menos precisa, según diferentes aproximaciones.

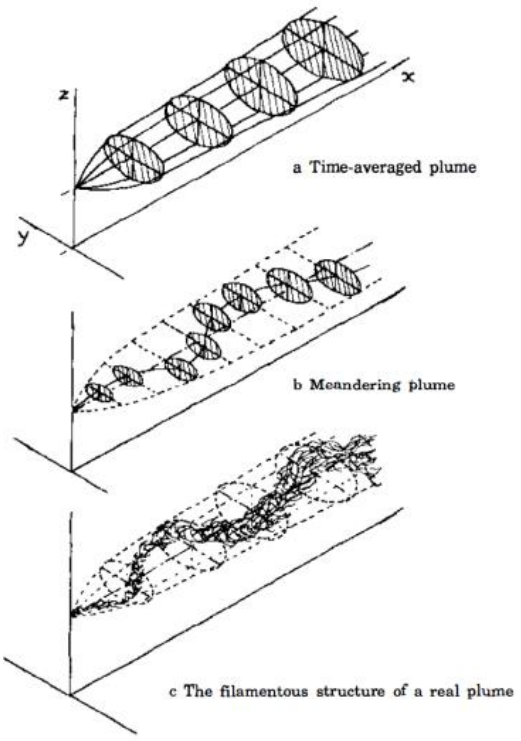


Figura 2-4. Aspecto del penacho según diferentes aproximaciones en orden creciente de complejidad. [16]

Estos fenómenos hacen que la distribución de la concentración de olores no sea un gradiente suave a lo largo de la dirección del viento, sino intermitente y con grandes fluctuaciones de concentración. Por lo tanto, es difícil recoger información precisa sobre la concentración de olores con una tecnología de detección limitada. El problema se complica aún más si, además, el sensor de olores es de respuesta lenta.

Para resolver estos problemas, en las últimas décadas han surgido trabajos de investigación sobre sensores más precisos o matrices de sensores y algoritmos de localización de fuentes de olor. En la actualidad, existen dos enfoques principales para localizar una fuente de olor: uno es el uso de redes de sensores estacionarias distribuidas espacialmente, y el otro es el uso de robots móviles con sensores. El primer enfoque es capaz de

localizar la fuente de olor de manera instantánea cuando se dan unas condiciones del entorno previstas, es decir, el viento se mueve en una dirección específica, la red de sensores estáticos mide un patrón de valores de concentración de gas que es conocido de manera que se puede "triangular" la posición de la fuente, etc. Sin embargo, el mayor inconveniente que presenta este enfoque es que se encuentra muy influenciado por las turbulencias del viento.

Bajo el segundo enfoque, el proceso de ejecución de los algoritmos de localización de fuentes de olor robóticas, para la mayoría de las investigaciones, puede dividirse en tres subtareas [17]:

- Búsqueda de la pluma de olor, que significa buscar en el entorno hasta detectar la presencia de olores;
- Seguimiento del penacho, que consiste en seguir el penacho para encontrar la fuente de olor;
- Localización de la fuente, que significa llegar a la proximidad de la fuente de olor.

Aunque el proceso puede expresarse por separado y parece ser secuencial, los límites de estas tres subtareas son imprecisos. Al igual que los perros a veces pierden la dirección mientras rastrean su objetivo, un sistema robótico de localización de fuentes de olor también puede salirse del penacho que está rastreando y tiene que reiniciar la etapa de búsqueda del penacho. Durante la etapa de rastreo de la pluma, en cuanto el sistema robótico se encuentre con la fuente de olor, dejará de rastrear la pluma. La mayoría de los algoritmos sólo abordan las dos primeras fases, mientras que, normalmente, la localización de la fuente suele hacerse utilizando otras modalidades de detección, como la visión, pero también podría hacerse basándose en la concentración medida muestreada en múltiples puntos, lo que en realidad es una tarea más subjetiva para el supervisor humano.

Cuando los robots móviles se aplican a la búsqueda de la fuente de olor, la guía más directa para ellos es la concentración química en varias posiciones y el flujo de viento. Después de tomar medidas en el entorno, los robots pueden decidir los siguientes pasos a través de un determinado algoritmo establecido. Desde el punto de vista de los métodos aplicados, se pueden dividir los algoritmos de localización de fuentes de olor existentes en cuatro categorías [18]: algoritmos basados en gradientes, algoritmos bioinspirados, algoritmos multirobot, algoritmos probabilísticos y algoritmos basados en mapas.

i) Los algoritmos basados en el gradiente son los primeros trabajos en este campo. Intentan guiar al robot hacia la fuente de olor a través del gradiente de concentración, lo que resulta en una tarea difícil de realizar en un escenario real.

ii) Los algoritmos bioinspirados están motivados por la capacidad de los seres vivos para encontrar la fuente de olor y planificar la trayectoria.

iii) Los algoritmos multirobot se diseñan de forma que la concentración de olores pueda ser muestreada en diferentes posiciones al mismo tiempo y compartida entre los robots para guiar su movimiento.

iv) Los algoritmos probabilísticos y basados en mapas modelan la localización de la fuente de olor como una distribución de probabilidad. A través de observaciones continuas, la distribución de probabilidad se convertirá en una función de Dirac y ahí es donde se encontrará la fuente.

Aunque se establece la clasificación anterior, los algoritmos de las diferentes categorías comparten características que se aprovechan de las otras estrategias, se puede decir que presentan "zonas de solapamiento" (ver Figura 2-5). Por ejemplo, los algoritmos multirobot basados en enjambres (por ejemplo, PSO y GSO) heredan realmente de comportamientos propios de seres vivos.

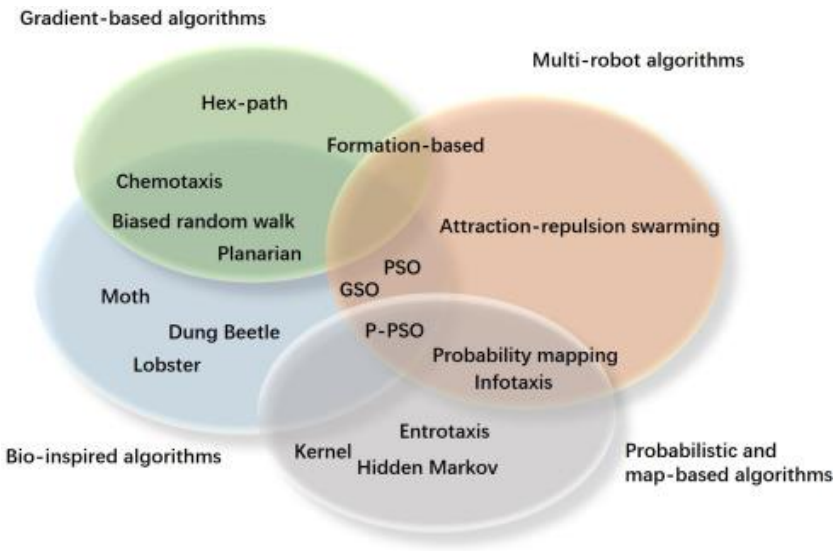


Figura 2-5. Diagrama de Venn de las 4 categorías de algoritmos de localización [18].

Algoritmos basados en el gradiente

Desde principios de los 90, los trabajos sobre algoritmos basados en gradientes son intentos preliminares en el campo de la investigación de la localización robótica de fuentes de olor. Debido a la falta de consideración sobre la estructura realista de la pluma, los investigadores asumieron un "gradiente de concentración" suave a lo largo de la dirección del viento. Diseñaron algoritmos para guiar a los robots hacia la fuente de olor explotando el gradiente de concentración y la dirección del viento, donde el gradiente se calculó tomando medidas de la concentración de olores en diferentes posiciones (mediante más de un sensor de olores simultáneamente, o un sensor utilizado varias veces).

La quimiotaxis [19] es un ejemplo de algoritmo típico basado en el gradiente de concentración química pura. La estrategia se basa en la detección de una diferencia de concentración entre dos sensores químicos o dos posiciones diferentes y un mecanismo de dirección hacia la dirección de mayor intensidad mientras se mueve hacia adelante a una velocidad constante. Bajo esta estrategia se repiten cuatro pasos: (1) se espera un breve período para permitir el tiempo de asentamiento de la medición sensorial, (2) se compara el valor de la medición en los sensores izquierdo y derecho, (3) el robot gira hacia la lectura más alta con un ángulo constante, (4) y se avanza con una longitud de paso constante.

Aunque los algoritmos basados en el gradiente son simples y fáciles de implementar, la estructura de los penachos en el aire es intermitente tanto en el tiempo como en el espacio por lo que no funcionan bien en escenarios reales. Por supuesto, existen algunas soluciones sencillas en este sentido: esperar después de cada paso para recoger un valor de concentración suficiente y tomar su media para obtener una información más precisa del gradiente, comportamiento que se asemeja al que muestran algunos animales como las estrellas de mar. Sin embargo, en la mayoría de las aplicaciones reales, se compite con el tiempo y es necesario localizar la fuente de olor rápidamente de modo la espera no puede ser una solución ideal para este propósito.

Algoritmos bioinspirados

Gracias a la evolución, todos los seres vivos han desarrollado un conjunto de principios propios para encontrar el camino hacia su comida o pareja utilizando la información olfativa y la dirección del viento, resultando en estrategias de localización de fuentes de olor muy optimizadas. Por este motivo, muchos algoritmos se inspiran en organismos vivos como polillas, bacterias, escarabajos peloteros, etc. Además, los algoritmos bioinspirados son relativamente poco costosos desde el punto de vista computacional, lo que significa que son una buena opción cuando la plataforma de hardware no es lo suficientemente potente.

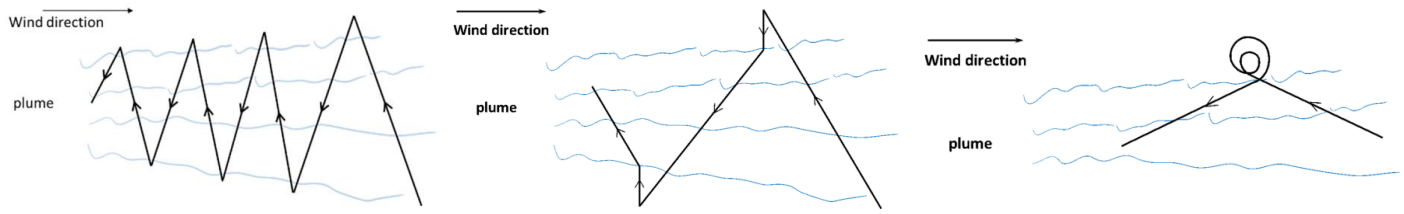
Aunque muchos algoritmos basados en el gradiente están inspirados en seres vivos, en su mayoría están relacionados con el comportamiento del movimiento de organismos como las bacterias, y cuando se aplican en flujos de fluidos dominados por la turbulencia (aire), los algoritmos inspirados en seres vivos voladores son los más fiables.

Algunos algoritmos típicos de inspiración biológica a los que los investigadores han prestado mucha atención son:

- Algoritmos inspirados en la polilla del gusano de seda. El macho de la polilla del gusano de seda localiza a las hembras mediante el seguimiento de un penacho de feromonas que éstas liberan. Según el algoritmo, si el robot detecta la pluma, entonces repite las siguientes acciones: movimiento directamente en contra del viento y oscilación de lado a lado con amplitud creciente para encontrar de nuevo el penacho hasta llegar a la fuente de olor.
- Algoritmos inspirados en la langosta [20]. Las langostas tienen un par de narices, una situada en cada una de sus antenas laterales. En presencia de una pluma de olor de comida, una langosta utiliza sus antenas para guiar su progreso. El algoritmo tiene dos reglas: (1) el robot gira hacia el lado de mayor concentración o avanza directamente si las dos antenas detectan casi la misma concentración, (2) el robot retrocede si ninguna de las dos antenas detecta la concentración. El uso de dos sensores espacialmente separados confiere una ventaja significativa en comparación con la quimiotaxis simple con un solo sensor.
- Algoritmos inspirados en el escarabajo pelotero. Al detectar el olor, el robot hace un zigzag en diagonal a través de la pluma en dirección contraria al viento y se da la vuelta cada vez que abandona el borde de la pluma.

Independientemente de la especie en la que se inspiran los algoritmos, las estrategias de movimiento de los algoritmos bioinspirados pueden dividirse de manera general en las siguientes categorías (que se describen gráficamente en la Figura 2-6) o una combinación de ellas:

- Zigzag. Se realiza un movimiento en zigzag a lo largo de la pluma en la dirección contraria al viento.
- Casting: Un robot en el penacho se desplaza a favor del viento con un ángulo β hasta que pierde el penacho durante una cierta distancia. A continuación, el robot gira y se desplaza en dirección contraria y perpendicular al viento hasta que se encuentra de nuevo con el penacho, y entonces se desplaza de nuevo en la dirección al viento con el mismo ángulo β.
- Oleada. El robot se mueve en línea recta en contra del viento.
- Espiral. El robot se mueve a lo largo de una espiral después de perder el rastro del penacho.



(a) (b) (c)

Figura 2-6. Movimientos típicos de algoritmos bioinspirados: (a) zigzag, (b) casting y (c) espiral. [18]

Dado que casi todos los algoritmos bioinspirados dependen de un solo robot con algunas reglas de movimiento preestablecidas, requieren mucha menos CPU y memoria que los algoritmos multirobot y los algoritmos probabilísticos y basados en mapas. Sin embargo, también es una carencia para esta categoría el hecho de que sea complejo desarrollar los algoritmos bioinspirados de un solo robot en algoritmos multirobot, ya que es difícil encontrar una manera de que los robots se coordinen mientras llevan a cabo su estrategia de movimiento de forma independiente.

Algoritmos multirobots

Los algoritmos multirobot existentes incluyen desde enfoques de enjambre de expansión sesgada, recorrido aleatorio sesgado, optimización de enjambre de partículas, técnicas de escalada de gradiente, infotaxis, búsqueda mediante exploración, enjambre basado en la física, enjambre de atracción-repulsión y algoritmos basados en la formación. A continuación, se describen algunos de ellos:

- Algoritmos basados en la formación: los algoritmos de localización de fuentes de olor basados en la formación se refieren a los algoritmos que llevan a un grupo de robots con una determinada distribución espacial a encontrar una fuente de olor compartiendo cierta información entre ellos sin romper su alineación. Ejemplo de una formación tridimensional puede ser una formación triangular con los robots de tierra en una línea de viento cruzado y un robot aéreo que se mueve detrás del escuadrón de tierra.

  Los algoritmos basados en la formación requieren muy poca capacidad de procesamiento, son fáciles de aplicar y no requieren más información de los sensores que el viento, el olor y las posiciones relativas de los robots. Sin embargo, una limitación de este tipo de algoritmos es que dependen del conocimiento de la dirección del viento. Esto los hace inaplicables a escenarios con poco viento o flujos muy turbulentos.

- Algoritmos basados en PSO: PSO [21] es un método computacional que mejora una solución candidata a un problema actualizándola de forma iterativa y que converge muy rápidamente. Se inspira en el comportamiento de las bandadas de pájaros que buscan comida al azar. Si sólo hay un trozo de comida en la zona que se busca, y no todos los pájaros saben dónde está la comida, la estrategia más eficaz para encontrarla es seguir al pájaro más cercano a ella.

  Cuando se aplica a la localización de fuentes de olor, el PSO se ejecuta de la siguiente manera: se inicializa un grupo de robots con ciertas posiciones y velocidades y se evalúa la distribución de la pluma, utilizando las coordenadas posicionales del robot como valores de entrada. Las posiciones y velocidades se ajustan y la función se evalúa con las nuevas coordenadas en cada paso de tiempo. Cuando un robot descubre una fuente potencial de olor que es mejor que cualquiera que haya encontrado anteriormente, la diferencia entre el mejor punto encontrado hasta ahora y la posición actual del robot se utiliza para ajustar el movimiento del robot.

  Las carencias de PSO son que los robots tendrán tendencia a moverse hacia el mismo óptimo local (en algún lugar donde no haya una fuente de olor) y que los robots no tienen en cuenta directamente las mediciones de concentración de otros robots, lo que puede considerarse como una falta de colaboración.

La ventaja más obvia de los algoritmos multirobot es su alta eficiencia. Al colaborar, los robots pueden compartir su información sobre toda el área de búsqueda, lo que puede reducir en gran medida el tiempo de búsqueda y ayudar a mantener la alineación de los robots en su posición prevista. Por contrapartida, estos algoritmos necesitan de varios robots por lo que el coste puede ser relativamente alto. Además, existe la posibilidad de que los robots se interfieran entre sí y reduzcan la eficacia.

Algoritmos probabilísticos y basados en mapas

Los algoritmos de las tres primeras categorías se basan más o menos en la información del gradiente de concentración. Si el viento es demasiado débil o demasiado turbulento, es más probable que estos algoritmos fallen.

En 2007, se propuso por primera vez la idea de Infotaxis [22], un algoritmo de localización de fuentes de olor basado en la probabilidad. Esta investigación inspiró a los investigadores a pensar en soluciones sin el cálculo del gradiente como son:

- Métodos de inferencia bayesiana: Este tipo de métodos se basan en la suposición de que la pluma se dispersa y se transmite a lo largo de una advección de flujo descendente. Si se conoce el campo de viento, es posible predecir la trayectoria del olor. El robot utiliza la velocidad del viento y la concentración de olor que ha medido y almacenado en su memoria para estimar la posible trayectoria del producto químico a lo largo del camino recorrido. Tras acumular la trayectoria probable, se construye un mapa de probabilidad de localización de la fuente [23], denominado "mapa de probabilidad de la fuente" (SLIM, del inglés Source Likelihood Map). El SLIM se actualiza a medida que el robot se mueve hasta que se acerca a la fuente y la probabilidad de que un determinado punto sea la fuente de olor aumenta hasta 1.

- Métodos Kernel: Estos algoritmos tratan el modelado de la distribución de la pluma como un problema de estimación de la densidad. No hacen fuertes suposiciones sobre la forma particular del modelo de distribución de la pluma, ni se basan en gran medida en los cálculos de la dinámica de fluidos, en su lugar, utilizan una función de ponderación gaussiana para representar la importancia de una medición obtenida en un lugar para modelar la distribución de gas en una celda de la cuadrícula.

En comparación con otras categorías de algoritmos, los algoritmos probabilísticos y los basados en mapas son

más flexibles en sus formas: se puede elegir cualquier modelo de distribución del penacho que se ajuste mejor al escenario de aplicación o incluso no hacer una suposición del modelo del penacho. Sin embargo, estos algoritmos requieren un alto coste computacional y una localización precisa de los puntos de muestreo, lo que no es fácil para algunas plataformas de hardware antiguas.

Para terminar, se incluyen tendencias actuales y retos futuros en el campo de la localización robótica de fuentes de olor que se han podido extraer a lo largo de este estudio sobre la literatura:

1. Casi no se presta atención a los algoritmos basados en el gradiente en la actualidad;
2. Los algoritmos multirobot y los algoritmos probabilísticos y basados en mapas han atraído más el interés de los investigadores (gracias a la actualización de las tecnologías de hardware);
3. Los investigadores están tratando de desarrollar algoritmos de localización de fuentes de olor en escenarios 3D, ya sea en el entorno subacuático o en el aire;
4. Los investigadores están tratando de abordar los problemas de múltiples fuentes de olor.

Estas tendencias se deben en gran medida al rápido desarrollo de la capacidad computacional del hardware y a las tecnologías maduras de los robots aéreos y marinos. Aunque se vislumbra un futuro floreciente en esta área de investigación, hay que tener en cuenta que todavía existen algunos retos y que queda un largo camino por recorrer.

2.3 Industria 4.0

El concepto de Industria 4.0 describe la organización de procesos productivos basados en tecnologías y dispositivos que se comunican de manera autónoma entre ellos a lo largo de la cadena de valor, sobre un modelo de fábrica inteligente (“Smart Factory”). En este modelo, los activos y procesos son monitorizados por sistemas digitales, que crean una copia virtual del mundo físico y toman decisiones en tiempo real basadas en órdenes programadas.

Este concepto está vinculado al incremento de la digitalización en la industria [24], donde los objetos físicos son integrados con la información digital o disponible en red, y que está cambiando la forma en que se diseñan, producen, comercializan, y se genera valor de los productos y servicios. Esta digitalización comprende desde la integración de innovaciones digitales en los productos, la transformación digital de los procesos de prestación de servicios, hasta la creación de nuevos modelos de negocio.

Bajo estos fundamentos, la Industria 4.0 representa un salto cualitativo en la organización y control de toda la cadena de valor a lo largo del ciclo de vida de la fabricación y entrega del producto. Esto produce un cambio de paradigma para las industrias que se basa en los siguientes principios:

- Interoperabilidad: capacidad de comunicación de todos los elementos de la fábrica, sistemas ciberfísicos (CPS, del inglés Cyber-physical System), robots, sistemas de información corporativos, productos inteligentes y las personas, así como sistemas de terceros.
- Descentralización: capacidad para el diseño de subprocesos autónomos dentro de la fábrica con elementos ciberfísicos con capacidad para tomar decisiones de forma autónoma.
- Análisis en tiempo real: capacidad de recopilar y analizar grandes cantidades de datos (Big Data) que permiten el seguimiento, control y optimización de procesos, facilitando cualquier resultado y decisión derivada del proceso de forma inmediata y en todo momento.
- Virtualización: capacidad de generar una copia virtual del tejido mediante la recolección de datos y el modelado de procesos industriales (físicos), obteniendo modelos de plantas virtuales y modelos de simulación.
- Modularidad y escalabilidad: la flexibilidad y versatilidad para adaptarse a las necesidades de la industria y el negocio en cada momento, con la capacidad de escalar la capacidad técnica del sistema de acuerdo con los requisitos técnicos que requiera la evolución de la demanda empresarial en cada caso.

Esta revolución de la Industria 4.0 está marcada por la utilización de muchas tecnologías innovadoras diferentes que proporcionan a la empresa una importante ventaja competitiva, un nivel cualitativamente

superior de comprensión y comunicación con los clientes y, en general, facilitan la vida en diversos ámbitos. A continuación, se comentan algunas de estas tecnologías habilitadoras [25] (ver Figura 2-7):

- Simulación: la principal tarea de esta tecnología es simular el control de cualquier proceso. Los simuladores crean una sensación de realidad para la persona que los utiliza y ofrecen la oportunidad de percibir la experiencia de la vida real en un entorno inexistente, por alguna razón inaccesible o que pone en peligro la vida. Esta tecnología también se utiliza en la investigación científica para simular entornos naturales y humanos.
- Big Data: cantidad enorme de datos estructurados o no estructurados. El Big Data tiene tres características principales: volumen, velocidad y variedad. El procesamiento y análisis de Big Data se utiliza para optimizar diversos procesos de producción y otros procesos empresariales.
- Integración de sistemas: combinación de diferentes subsistemas en un gran sistema que funciona como uno solo. Al mismo tiempo, la funcionalidad de todos los subsistemas se mantiene al mismo nivel, y la eficiencia global del sistema combinado aumenta. Estos sistemas integrados pueden ayudar, por ejemplo, a proporcionar al cliente un servicio fundamentalmente nuevo o a automatizar actividades rutinarias, reduciendo los errores humanos.
- Gemelos Digitales: son una tecnología que permite crear una copia virtual de cualquier objeto o proceso del mundo real. Su objetivo es probar, encontrar y eliminar deficiencias para mejorar la calidad del servicio o producto. El gemelo digital proporciona apoyo a las empresas al conectar los mundos virtual y físico.

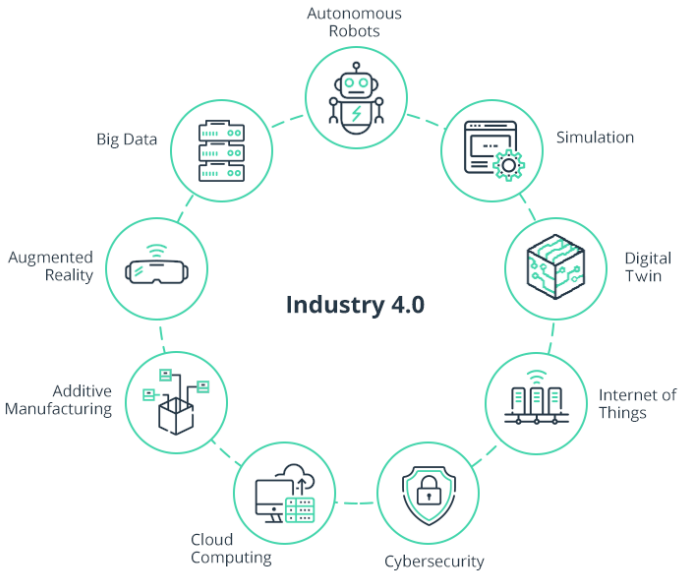


Figura 2-7. Mapa de tecnologías habilitadoras empleadas en la Industria 4.0 y la transformación digital.

De estas tecnologías, los gemelos digitales han despertado un gran interés en los últimos años a medida que el IoT se ha vuelto cada vez más omnipresente. Un gemelo digital es un modelo virtual que refleja un objeto o proceso físico a lo largo de su ciclo de vida [26]. Esta tecnología, que proporciona un puente casi en tiempo real entre el mundo físico y el digital, permite supervisar y controlar equipos y sistemas de forma remota. En última instancia, puede ejecutar modelos de simulación para probar y predecir cambios de activos y procesos en diferentes escenarios hipotéticos. Al aprovechar los gemelos digitales, las empresas pueden obtener beneficios sustanciales, como operaciones mejoradas, innovación de productos y servicios y un tiempo de comercialización más rápido. La creación de un gemelo digital requiere diferentes elementos, que incluyen: (i) sensores que capturan comportamientos operativos de activos y procesos; (ii) redes de comunicaciones, que brindan una transferencia de datos segura y confiable desde dispositivos físicos al mundo digital; y (iii) una plataforma digital, que sirve como un repositorio de datos que agrupa y almacena datos de sensores de planta con datos comerciales de alto nivel. Al combinar estas fuentes de datos, se pueden derivar conocimientos prácticos para la toma de decisiones basada en datos, utilizando algoritmos avanzados.

Como se comprueba, la tecnología de gemelos digitales brinda una visibilidad sin precedentes de los activos y la producción para detectar cuellos de botella, optimizar las operaciones e innovar el desarrollo de productos. Ejemplos de aplicaciones de los gemelos digitales para la Industria 4.0 son (i) el mantenimiento predictivo, al obtener una visión integral del estado y el rendimiento de los equipos, las empresas pueden detectar de inmediato anomalías y desviaciones en sus operaciones; (ii) planificación y optimización de procesos, una huella digital que toma datos de sensores y el ERP (del inglés, Enterprise Resource Planning) de una línea de fabricación puede analizar de manera integral los KPI (del inglés, Key Performance Indicator) importantes, como las tasas de producción y el recuento de desechos, ayudando a diagnosticar la causa raíz de las ineficiencias y pérdidas de rendimiento; (iii) diseño de productos y creación de prototipos virtuales, los modelos virtuales de productos en uso brindan información completa sobre los patrones de uso, el punto de degradación, la capacidad de carga de trabajo, los defectos incurridos, etc. Al comprender mejor las características de un producto y los modos de falla, los diseñadores y desarrolladores pueden evaluar correctamente la usabilidad y mejorar el diseño futuro de los componentes., entre otras.

2.4 Plataformas IoT

El IoT, es un concepto que fue desarrollado por primera vez por Kevin Ashton en 1999 en el contexto de la gestión de la cadena de suministro, donde describió un sistema en el que el mundo físico está conectado a Internet a través de sensores ubicuos. Desde entonces, el IoT se ha convertido en un área de investigación activa en varios ámbitos donde empresas e instituciones han creado diferentes encarnaciones de sistemas de IoT para distintos casos de uso. Los hogares inteligentes son un ejemplo de estos sistemas, pero en otros ámbitos, también se están llevando a cabo desarrollos similares, como los coches conectados, las ciudades inteligentes, la gestión de la demanda, las redes inteligentes (Smart Grids) o los sistemas de fábricas inteligentes.

El estado actual de desarrollo del IoT se caracteriza por un conjunto diverso de iniciativas, normas e implementaciones, donde la estandarización y la interoperabilidad siguen siendo un reto importante [27]. Los esfuerzos recientes en este sentido están convergiendo hacia especificaciones estándar como el protocolo Lightweight Machine to Machine (LWM2M) de la Open Mobile Alliance (OMA) y arquitecturas de referencia como la European Internet of Things Architecture (IoT-A). Estos esfuerzos reúnen a actores clave de la industria, la administración y las organizaciones de normalización, lo que permite implementaciones reutilizables en distintos ámbitos de aplicación.

Convertir la visión de IoT en una realidad implica el despliegue de dispositivos, como sensores y actuadores, que recogen y posiblemente actúan sobre los datos de su entorno:

- Sensor: Un sensor es un componente de hardware que captura información sobre el entorno físico "respondiendo a un estímulo físico (como el calor, la luz, el sonido, la presión, el magnetismo o un movimiento particular)". Los sensores transmiten la información captada mediante señales eléctricas a los dispositivos a los que están conectados. Esta conexión puede establecerse (i) por cable o (ii) de forma inalámbrica. La conexión por cable incluye la integración de los sensores en un dispositivo.
- Actuador: Un Actuador es un componente hardware que manipula el entorno físico. Los Actuadores reciben órdenes y traducen estas señales eléctricas en algún tipo de acción física. Por ejemplo, un actuador que enciende o apaga una ventilación dentro de una habitación actúa sobre el entorno físico influyendo en la humedad de la habitación. Al igual que los sensores, la conexión con el dispositivo puede establecerse (i) por cable o (ii) de forma inalámbrica, por lo que una conexión por cable incluye la integración en un dispositivo.

Estos dispositivos suelen estar restringidos debido a su limitada capacidad de batería, computación y almacenamiento. Por lo tanto, el almacenamiento y el procesamiento de los datos de estos dispositivos, así como la gestión de estos, deben realizarse en un entorno informático adecuado que proporcione un conjunto de características deseables como escalabilidad, fiabilidad, sostenibilidad y seguridad. Las características de la computación en la nube la convierten en una opción adecuada para esta función, con muchas funcionalidades y aplicaciones relacionadas con el IoT ya desplegadas en sistemas en la nube. Las plataformas de IoT en estos backends en la nube proporcionan una interfaz que interactúa con los dispositivos utilizados para la detección, y una interfaz para la interacción con los servicios de nivel superior, las aplicaciones Machine-to-Machine (M2M) o los usuarios finales.

Al mismo tiempo, el amplio espectro de posibilidades de tecnologías de comunicación y protocolos, como, por ejemplo, Bluetooth, MQTT, Wifi, Lora, telefonía, entre otras muchas, hacen que la interoperabilidad de los dispositivos no siempre sea posible, convirtiéndose en una barrera que las plataformas en la nube deben superar para garantizar el despliegue de dispositivos con diferentes propósitos.

Finalmente, a medida que las plataformas de IoT comienzan a desplegarse en el mundo real, los componentes de estas plataformas tendrán que ser capaces de escalar bien para manejar cientos y miles de dispositivos que se espera se conecten a estas plataformas.

Figura 2-8. Ilustración de ecosistema IoT conectado [28].

Como resultado, en los últimos años se han creado multitud de plataformas de este tipo con diversas arquitecturas y tamaños [29], y aunque los esfuerzos de estandarización están en curso, no hay estándares generalmente acordados para IoT en este momento, sino más bien, el desarrollo de estas plataformas ha tenido lugar a menudo de forma aislada e independiente. Estos diferentes entornos han influido no sólo en la elección de los conceptos y la tecnología, sino también en la elección de la terminología, dando como resultado, un panorama muy heterogéneo de plataformas. Al mismo tiempo, sin embargo, todas estas soluciones hacen más o menos lo mismo: permiten conectar diferentes dispositivos, acceder a sus datos y procesarlos, y utilizar los conocimientos adquiridos mediante esta actividad para crear un control automatizado.

En esta sección, se describen cuatro plataformas IoT de código abierto, FIWARE, OpenMTC, SiteWhere, Webinos y dos soluciones propietarias, AWS IoT y Azure IoT Hub de Microsoft

FIWARE

FIWARE es una plataforma de código abierto que permite acceder y gestionar información de contexto heterogénea a través de una API (del inglés, Application Programming Interface) abierta. FIWARE implementa completamente un estándar para el intercambio de información de contexto: NGSI (del inglés, Next Generation Service Interface). Además, FIWARE ofrece un conjunto de habilitadores genéricos y soluciones para proporcionar servicios inteligentes con el Context Broker como componente principal.

FIWARE ayuda a las corporaciones a posicionarse como socios estratégicos en la transformación digital de sus clientes porque es particularmente adecuado para desarrollar un sistema global de gestión inteligente que procesa/analiza la información recopilada de diferentes soluciones verticales, ayudando al cliente a supervisar el rendimiento general de su negocio y tomar decisiones inteligentes.

Esta plataforma es el resultado de un esfuerzo conjunto de diferentes organizaciones para encontrar mecanismos comunes que ayuden a las ciudades, la industria y las comunidades a ser interoperables y ha ganado relevancia debido a su enfoque neutral con respecto a los proveedores, definiendo estándares que hacen posible que todo tipo de empresas e industrias construyan y compartan soluciones inteligentes.

OpenMTC

La plataforma OpenMTC es una implementación prototipo de un middleware IoT/M2M que tiene como objetivo proporcionar una plataforma compatible con los estándares para los servicios de IoT. OpenMTC interconecta varios sensores y actuadores de diferentes dominios, agrega los datos recopilados, los envía a las aplicaciones y transmite instrucciones a los dispositivos finales para un control basado en eventos.

El middleware de integración agrega datos de múltiples pasarelas que podrían asignarse a varios dominios de aplicaciones. A través de las API REST (del inglés, Representational State Transfer), los desarrolladores de aplicaciones pueden acceder a los datos de los dispositivos, sin tener que luchar con las especificaciones de la tecnología subyacente de los dispositivos conectados a los puertos de enlace.

Estos puertos albergan adaptadores de protocolo específicos de la tecnología para recuperar los datos de los dispositivos y almacenarlos en un modelo de datos compatible con oneM2M. La conectividad entre los puertos y el backend pueden ser: redes locales, a través de Internet o redes administradas por el operador, ya que oneM2M especifica interfaces independientes de la red. El software OpenMTC está escrito en Python y puede implementarse en diferentes plataformas de hardware.

SiteWhere

SiteWhere es una plataforma abierta para monitorear y controlar dispositivos de IoT y almacenar datos de dispositivos. Tiene licencia de CPAL 1.0 y se puede implementar como una solución local con total libertad y escalabilidad, lo que elimina los precios por dispositivo o por datos de los proveedores de SaaS. También se puede implementar en un entorno de nube completa para acceso público. Viene con una arquitectura avanzada de múltiples inquilinos para ejecutar varias aplicaciones de IoT en una sola instancia de servidor. Se integra con software y servicios probados para las necesidades empresariales mejoradas de los dispositivos conectados.

La plataforma central del servidor se basa en Spring Boot y viene con un servidor Tomcat incorporado con soporte multi-tenancy. Cada motor de inquilino está aislado y tiene sus propios canales de procesamiento y almacenamiento de datos. Utiliza MongoDB para el almacén de datos predeterminado, con SiteWhere como base de datos principal y bases de datos aisladas para cada uno de los inquilinos. Los dispositivos pueden comunicarse utilizando el formato JSON o los búferes de protocolo de Google (Protobuf) a través del transporte MQTT, y se ofrecen API REST para aprovechar los servicios de la plataforma.

Webinos

webinos es un proyecto financiado con fondos europeos que tiene como objetivo ofrecer una plataforma de aplicaciones de código abierto que permite que diferentes dispositivos funcionen de forma conjunta. Estos dispositivos van desde televisores, teléfonos, tablets, sistemas de entretenimiento en el vehículo, computadores y unidades de almacenamiento conectadas a la red hasta sistemas de automatización del hogar, entre otros. Cualquier dispositivo conectado a la red puede integrarse a través de un modelo de abstracción y convertirse en parte de la plataforma webinos.

Las funciones del dispositivo pueden ser utilizadas como servicios por otros dispositivos conectados. Esto significa que las aplicaciones pueden aprovechar los servicios remotos que de otro modo no estarían disponibles en el dispositivo en el que se ejecuta la aplicación. El canal de comunicación subyacente entre dispositivos utiliza cifrado de extremo a extremo para que la comunicación sea segura en todo momento. Al utilizar el sistema de políticas integrado, los usuarios tienen el control total y pueden revocar u otorgar permiso para utilizar los servicios de webinos.

webinos proporciona un reconocimiento simple y el uso seguro de servicios en dispositivos remotos. Todos los dispositivos están conectados a una instancia central que actúa como un hub, denominado Personal Zone Hub (PZH). Cada dispositivo conectado al concentrador se denomina Proxy de zona personal (PZP) (ver Figura 2-9). Las zonas personales se pueden conectar para que los usuarios compartan el acceso a sus servicios.

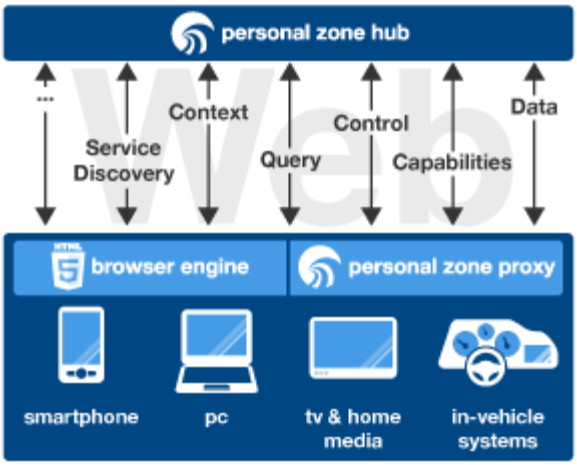


Figura 2-9. Arquitectura y servicios de la plataforma Webinos [30].

AWS IoT

AWS IoT es un conjunto de componentes cada uno de los cuales proporciona una serie de servicio facilitando el proceso de desplegar una plataforma IoT. Entre estos componentes se puede encontrar el AWS IoT Core, que forma la columna vertebral de las implementaciones de IoT. Este componente ayuda a conectar de forma segura todos los dispositivos a la nube de AWS y entre sí, y manejar los datos a gran escala. Permite además enrutar, procesar y actuar sobre los datos y mensajes que provienen de esos dispositivos y también permite desarrollar aplicaciones que pueden interactuar con sus dispositivos incluso cuando están fuera de línea.

Por otra parte, Message Broker realiza la transmisión segura de mensajes con baja latencia hacia y desde dispositivos inteligentes conectados, admitiendo una amplia gama de patrones de mensajes. AWS IoT Device Defender audita continuamente las políticas asociadas con los dispositivos. También monitorea la flota de dispositivos en busca de comportamientos anormales que puedan indicar un posible problema de seguridad y alerta si algo no se ve bien, como por ejemplo un tráfico de datos desde el dispositivo a una dirección IP no autorizada, o picos en el tráfico saliente que podrían indicar que el dispositivo está participar en un ataque DDoS. AWS IoT Analytics es un servicio de análisis totalmente administrado para analizar fácilmente los datos de IoT. Filtra, transforma y enriquece los datos de IoT antes de almacenarlos en un almacén de datos optimizado para IoT. Estos son solo algunos de los múltiples componentes que ofrece esta plataforma.

El AWS IoT Device SDK (Kit de Desarrollo de Software) se utiliza para acelerar la conexión de dispositivos inteligentes con AWS IoT Core y las aplicaciones móviles de los usuarios, autenticar e intercambiar mensajes con los protocolos MQTT, HTTP o WebSockets.

Microsoft Azure IoT Hub

Como se puede ver en la Figura 2-10, IoT Hub es un servicio administrado, hospedado en la nube, proporcionado por Microsoft Azure, que actúa como centro de mensajes para la comunicación entre una aplicación de IoT y los dispositivos conectados. Puede conectar millones de dispositivos y soluciones de backend de forma segura. Como su nombre indica, Azure IoT Hub es el núcleo de todas las soluciones cuya cloud sea Azure.

El IoT Hub permite la comunicación de forma bidireccional con todos los dispositivos IoT que estén conectados y registrados, pudiendo a la vez conectar otros elementos de Azure como por ejemplo una herramienta de análisis de flujos de datos (Stream Analytics) para que lean de esta los mensajes y así poder realizar un procesado inicial o un guardado en una base de datos.

Se admiten varios patrones de mensajería, como telemetría del dispositivo a la nube, carga de archivos desde dispositivos y métodos de solicitud-respuesta para controlarlos desde la nube. IoT Hub también admite la supervisión para ayudar a realizar un seguimiento del reconocimiento de dispositivos, la conexión y los errores.

IoT Hub está capacitado para gestionar millones de dispositivos conectados de manera simultánea y millones

de eventos por segundo para admitir las cargas de trabajo de IoT.

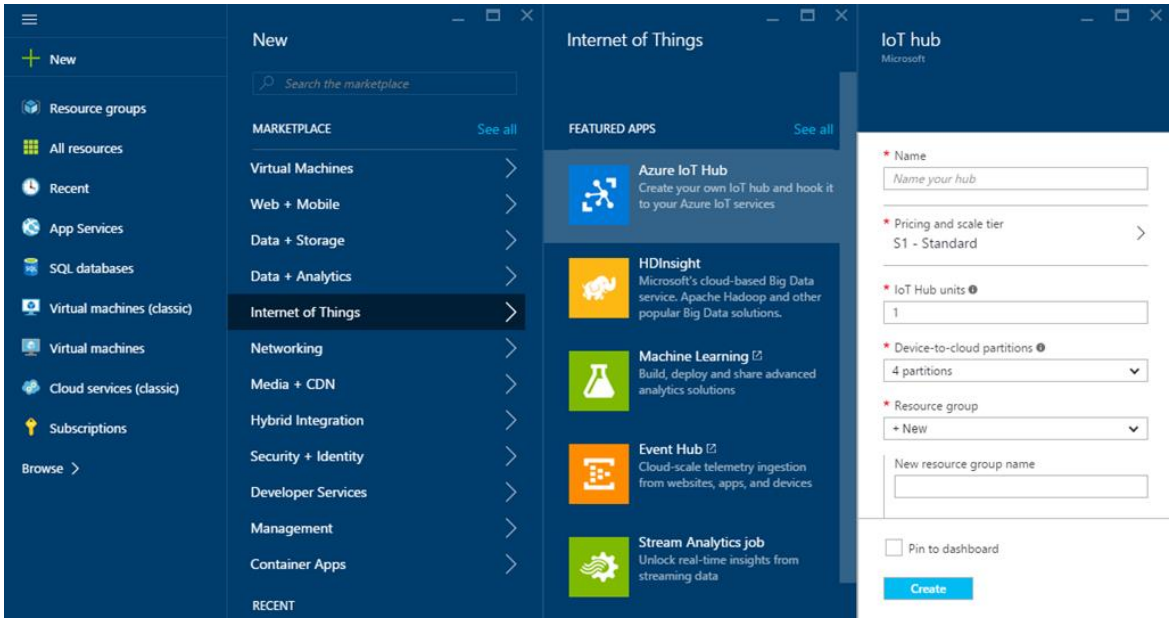


Figura 2-10. Visualización de la interfaz del servicio IoT Hub de Microsfot Azure [31].

3 SIMULACIÓN

La dispersión de gases es un fenómeno de gran complejidad en el que entran en juego muchas variables ambientales y topológicas y en el que pequeños cambios de esas variables provocan variaciones sustanciales en los conjuntos de datos recogidos, motivando que la repetibilidad entre experimentos sea difícil de conseguir.

En este contexto, las herramientas de simulación con capacidad para manejar adecuadamente el fenómeno real de la dispersión de gases pueden utilizarse para realizar evaluaciones exhaustivas antes de pasar a las pruebas experimentales en el mundo real facilitando así el desarrollo de nuevos algoritmos, sensores y plataformas que conforman los sistemas MRO.

3.1 GADEN

En este trabajo, se hace uso de la herramienta GADEN, un framework de simulación de dispersión de gases en entornos 3D de código abierto, dirigido a aplicaciones relacionadas con MRO y desarrollado por el Grupo de Investigación MAchine Perception and Intelligent Robotics (MAPIR[1]) de la Universidad de Málaga (UMA). La modularidad de GADEN permite al usuario implementar o importar diferentes modelos de sensores, integrar algoritmos robóticos ampliamente extendidos para tareas como la localización, la planificación de trayectorias o el mapeo, y evaluar los algoritmos MRO implementados, todo ello bajo un marco de simulación unificado. En concreto, las contribuciones que aporta GADEN en el contexto de la simulación de dispersión de gases para aplicaciones relacionadas con la robótica pueden resumirse como sigue:

- Está totalmente desarrollado con el sistema operativo para robots ROS. Podría decirse que ROS es el sistema operativo de robótica más utilizado en el mundo académico y en la industria.
- Soporte completo de modelos 3D-CAD (modelos de diseño asistido por ordenador) para definir fácilmente entornos complejos (múltiples habitaciones, obstáculos, entradas/salidas, etc.).
- Se apoya en OpenFoam, una librería de herramientas de dinámica de fluidos computacional de código abierto, para obtener campos de viento reales dentro del entorno 3D para una amplia variedad de condiciones de viento (números de Reynolds, velocidades del viento, etc.).
- Implementa la teoría de dispersión de gases basada en filamentos sobre un campo vectorial de flujo de viento que evoluciona en el tiempo para obtener distribuciones de gas precisas en 3D.
- Admite múltiples fuentes de gas y diferentes sustancias químicas simultáneamente.
- Tiene en cuenta las fuerzas de gravedad y de flotación considerando el peso molecular de los gases liberados.
- Simula diferentes tecnologías de detección de gases, incluyendo detectores de gas de óxido metálico (MOX) y de fotoionización. En cuanto a las mediciones del flujo del viento, se simula un anemómetro ultrasónico 2D de uso común.

GADEN[2] está implementado como un paquete ROS, que contiene los diferentes componentes del simulador, manuales y datos de prueba.

[1] http://mapir.isa.uma.es/mapirwebsite/
[2] https://github.com/MAPIRlab/gaden

3.1.1 Estructura del simulador

El simulador presentado sigue una estructura de tres etapas [32]. En la primera etapa, se define el entorno especificando las dimensiones, los obstáculos y las entradas/salidas físicas del viento, entre otros parámetros. A continuación, mediante herramientas CFD, se simulan los patrones de flujo de viento dentro del entorno, y, finalmente, en la última etapa, se calcula la estimación de la dispersión de gases aplicando el modelo de dispersión de filamentos. En los siguientes apartados se describen las tres etapas para obtener una visión general del proceso completo de simulación.

Etapa 1: Definición del entorno

A lo largo de este trabajo, el término "entorno" se refiere al conjunto de paredes, puertas, ventanas, obstáculos y, en general, cualquier elemento que pueda influir o interrumpir los flujos de viento y, en consecuencia, la dispersión de gases. La definición precisa del entorno es una de las claves para el éxito de la simulación de los flujos de viento mediante herramientas CFD. Estas herramientas numéricas suelen requerir que el entorno se proporcione en un formato muy específico y complejo que satisfaga ciertos criterios para garantizar una solución numérica válida y, por tanto, precisa. Por ejemplo, OpenFOAM, define el entorno como un "polyMesh" (malla de poliedros arbitrarios), delimitada por caras poligonales arbitrarias. Esto ofrece una gran libertad para la generación y manipulación de la malla, algo crucial cuando la geometría del dominio es compleja o cambia con el tiempo.

Otros frameworks anteriores definían el entorno de forma manual, es decir, modelando el “polyMesh” a mano, especificando los vértices y las aristas de los diferentes elementos de la malla, y, en consecuencia, los entornos presentados eran muy simplistas (normalmente un rectángulo con un par de obstáculos en él). Para superar esta limitación y permitir la simulación de la dispersión de gases en entornos realistas, el framework utilizado en este trabajo soporta el uso de herramientas automatizadas para la generación del “polyMesh” a partir de modelos CAD.

Como ilustración, la Figura 3-1 muestra tres entornos diferentes diseñados con complejidad creciente y sus respectivas mallas listas para iniciar la simulación de viento CFD. Los tres escenarios representan una habitación vacía con un gran número de puertas (entradas) y ventanas (salidas), un entorno de varias habitaciones con mobiliario básico y una habitación tipo oficina con objetos detallados como mesas y sillas.

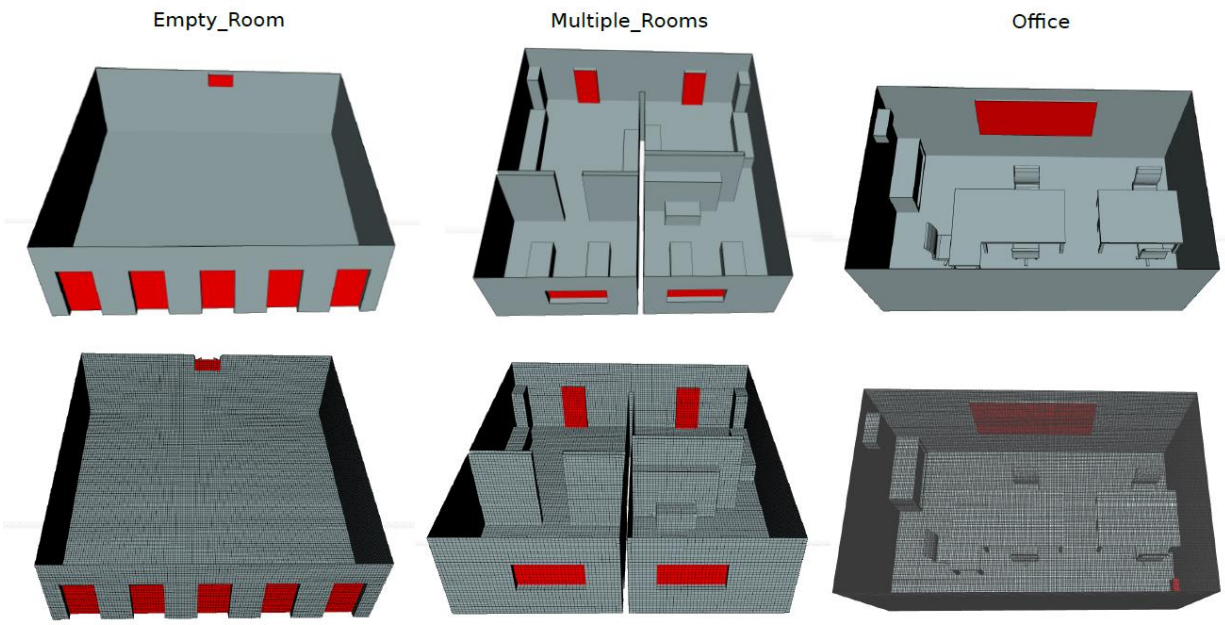


Figura 3-1. Ejemplos de entornos con obstáculos en orden creciente de complejidad.

Existe una amplia gama de herramientas CAD disponibles tanto online como offline y para diferentes niveles de experiencia. Independientemente de la elección, existen algunas consideraciones a tener en cuenta a la hora

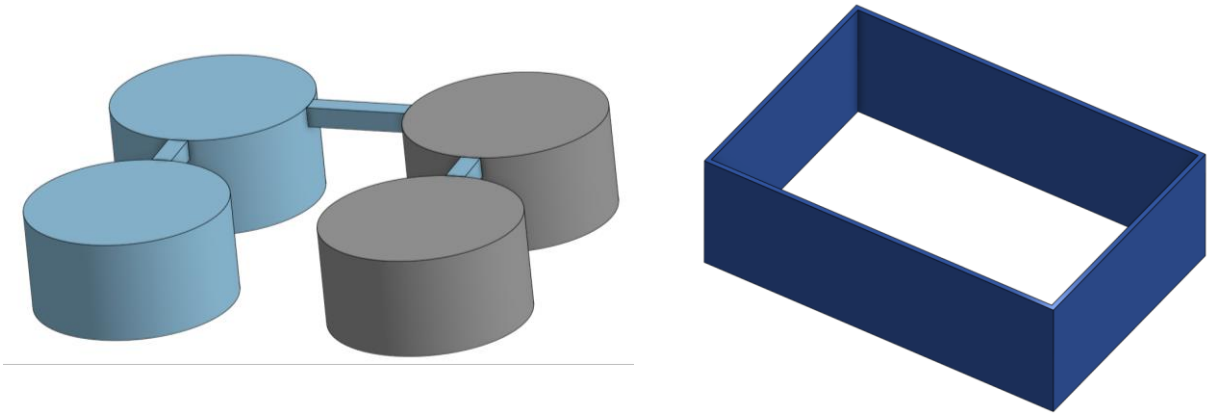
de generar el modelo CAD de un nuevo entorno:

a) El modelo CAD que se va a emplear se compone de dos “piezas” [33]: la “pieza_1” representa las paredes, el mobiliario y los objetos en general que dan forma al ambiente, mientras que la “pieza_2” representa el volumen "interior" del ambiente donde se van a liberar los gases. En general, esto se consigue creando varias partes en el diseño CAD, una que represente las paredes y los obstáculos, y luego se obtiene el espacio interior como una parte CAD diferente.

b) La “parte_1” (objetos) es utilizada por GADEN para dispersar los gases, evitando que los filamentos atraviesen paredes o muebles, mientras que la “pieza_2” es utilizada como entrada al CFD para obtener las condiciones de flujo de viento que regirán el experimento.

c) Se pueden definir múltiples partes CAD para la “pieza_1”, como por ejemplo paredes, mesas, puertas, etc., pero se recomienda exportar sólo una parte CAD para la “pieza_2” (el volumen interior) ya que la mayoría de las herramientas CFD sólo aceptan una pieza.

Para para las simulaciones de este trabajo se selecciona una región de la EDAR Copero en la que se encuentran 4 torres con pasarelas intermedias. El modelo CAD de esta región de la planta se realiza empleando una aplicación online gratuita llamada OnShape[3]. En primer lugar, se obtienen los modelos CAD de la “pieza_1” (ver Figura 3-2) que representan las paredes y los objetos del entorno (4 torres y pasarelas) que utilizará el simulador de filamentos GADEN.

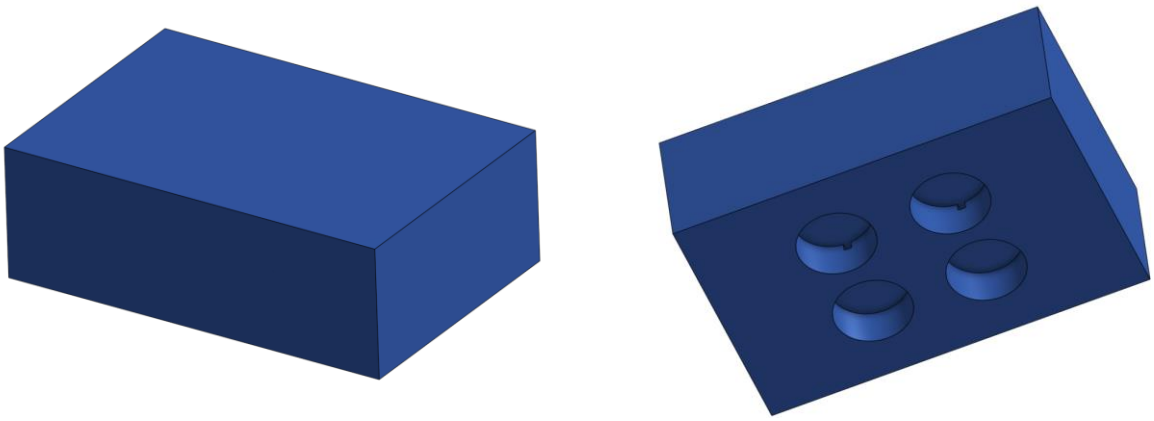


(a) (b)

Figura 3-2. Modelo CAD de la “parte_1” seleccionada para este trabajo. A la izquierda se pueden observar las 4 torres con sus pasarelas y a la derecha las “parades” que representan los límites del entorno de simulación.

A continuación, se obtiene el modelo CAD del volumen interior (“parte_2”) que se utilizará para la simulación del viento mediante CFD (ver Figura 3-3).

[3] https://cad.onshape.com



(a) (b)

Figura 3-3. Vistas del modelo CAD del volumen interior ("parte_2") utilizado para simular la evolución del viento mediante CFD.

Etapa 2: Simulación CFD del viento

La simulación de los flujos de viento dentro del entorno recién generado es un proceso muy complejo y depende de múltiples parámetros. El enfoque más común es obtener una aproximación más o menos realista mediante CFD. Este simulador se basa en el software gratuito y de libre acceso denominado OpenFOAM, posiblemente uno de los proyectos de CFD de código abierto más extendidos y activos. OpenFOAM toma como entrada el polyMesh del volumen interior del entorno, junto con una extensa lista de parámetros que controlan el proceso de simulación, como las condiciones de viento en el límite, la simulación transitoria o estable, el nivel de turbulencia, es decir, el número de Reynolds, las propiedades de los materiales, el solver, etc.

En general, las personas no expertas en dinámica de fluidos pueden encontrar este software demasiado complejo y con demasiados parámetros, lo que conduce a simulaciones infructuosas en muchas ocasiones. Para paliar este inconveniente, en los últimos años han aparecido diferentes herramientas gráficas para controlar OpenFOAM desde una interfaz más amigable. Este es el caso, por ejemplo, de SimScale[4], una herramienta online que permite importar el volumen interior del entorno como un modelo CAD, generar la malla requerida por OpenFOAM, y configurar los parámetros más importantes de la simulación CFD a través de una interfaz fácil e intuitiva.

Los pasos más importantes a seguir para procesar el modelo CAD del volumen interior y obtener la simulación de flujo de viento deseada son los siguientes:

a) Importar el modelo CAD a SimScale. Esto simplifica enormemente la configuración del entorno en OpenFoam. Existen diferentes herramientas que permiten crear el polyMesh que OpenFOAM requiere a partir de un modelo CAD, aquí se emplea la integrada en la plataforma SimScale.

b) Generar el PolyMesh a partir del modelo CAD. Se recomienda no utilizar la herramienta de malla automática, sino intentar obtener una malla más rígida basada en cubos utilizando la opción "Hex-dominant parametric". El motivo de esto es que GADEN trabajará con una malla 3D perfecta de celdas cúbicas, por lo que cuanto más se acerquen las simulaciones de OpenFOAM a esta malla perfecta, mejores resultados se obtendrán.

   Para la opción "Hex-dominant parametric" se necesitan definir las dimensiones del entorno exterior. Dado que sólo interesa la simulación del viento dentro del entorno, se puede establecer con las dimensiones reales del entorno. Del mismo modo, se debe establecer el número de celdas en cada dimensión, asegurándose de que las dimensiones del entorno son múltiples de la dimensión de las celdas utilizadas, ya que en otro caso surgirán problemas de celdas que caen en mitad de un obstáculo o en mitad del espacio libre.

c) Ejecutar la simulación. Una vez generada la malla, se deben establecer los parámetros de simulación:

[4] https://www.simscale.com

Dinámica de fluidos - Flujos incompresibles - Modelo de turbulencia k-epsilon - Simulación transitoria con el solver PIMPLE. Se selecciona la malla para simular sobre ella y se establece como medio el aire.

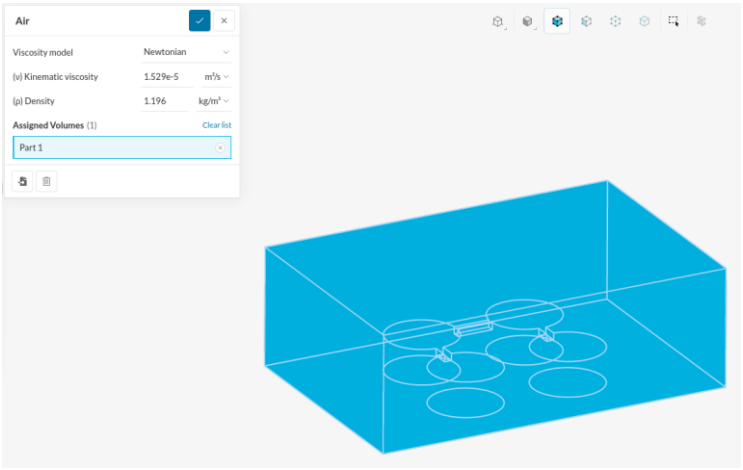
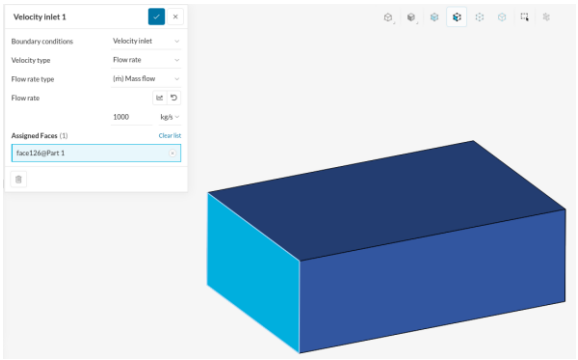
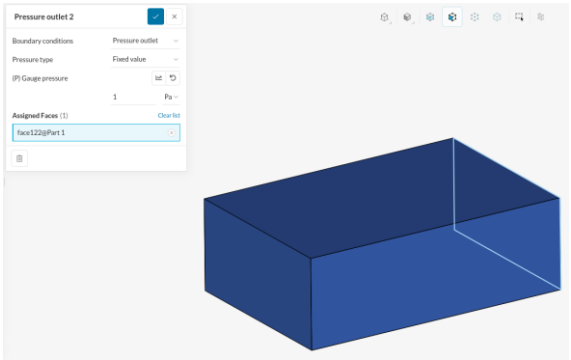


Figura 3-4. Visualización de la aplicación SimScale al seleccionar como medio el aire en el volumen “interior”.

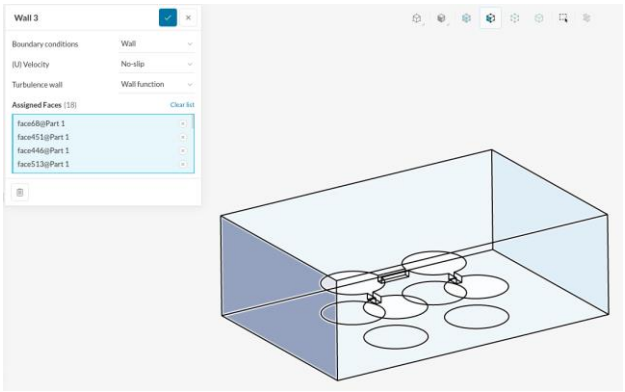
A continuación, se establecen las condiciones de contorno, fijando una velocidad de viento de entrada para las entradas del entorno (en el caso de este trabajo se trata de uno de los laterales de la planta), y una presión constante en las salidas (el extremo contrario de la planta). Las paredes y los obstáculos también necesitan ser definidos como condiciones de contorno "deslizante/no deslizante" (contornos laterales de la planta que no corresponden a la entada y la salida). La selección de las condiciones de contorno se muestra en la Figura 3-5.



(a)



(b)



(c)

Figura 3-5. Visualización de la aplicación SimScale para las condiciones de contorno de la simulación: (a) velocidad del aire de entrada a través de una cara lateral del modelo CAD, (b) presión de salida en la cara opuesta, y (c) fronteras.

Se recomienda establecer un paso de simulación de intervalo corto para permitir una buena convergencia de los algoritmos, pero que permita un "paso de tiempo ajustable", estableciendo el número máximo de courant a 0,7. Como esta opción hará que los pasos de tiempo varíen en longitud, es necesario establecer los parámetros de escritura en "Adjustable at runtime" y seleccionar el intervalo de tiempo para guardar los datos.

Finalmente, para las condiciones de contorno elegidas (velocidad de entrada 0.1 m/s y presión atmosférica a la salida) se presentan en la Figura 3-6, diferentes imágenes del campo vectorial generado de velocidades del viento.

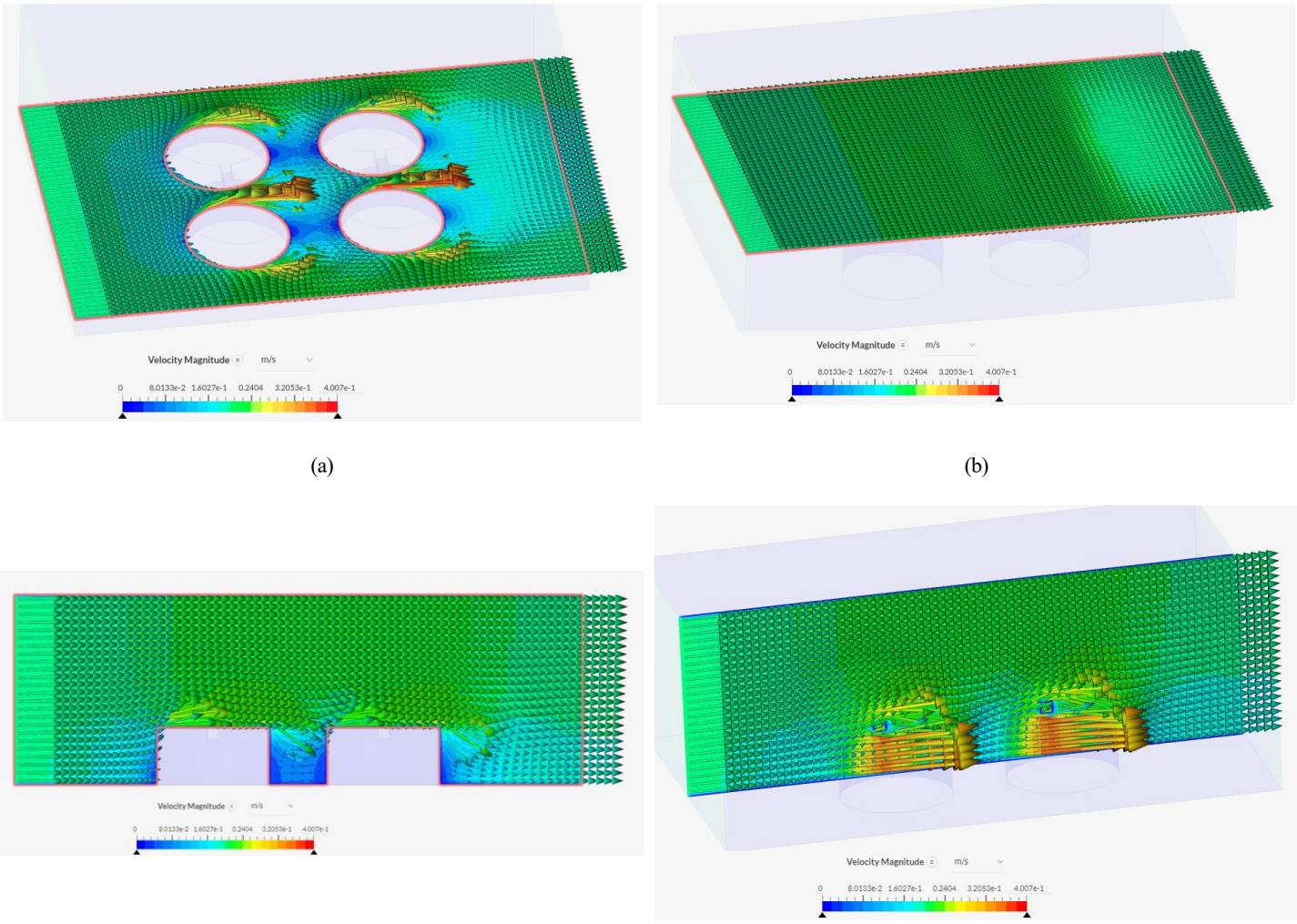


Figura 3-6. Resultados del campo vectorial del viento tras la simulación. Se visualiza el campo vectorial para distintos cortes en el plano z (a) a una altura intermedia, (b) a una altura por encima de las torres; y en el plano y, (c) a una distancia en la que el plano interseca con las torres, y (d) a lo largo del centro del volumen.

Hay que tener en cuenta que para la mayoría de las simulaciones ni todas las celdas son cubos perfectos, ni tienen el mismo tamaño, por lo que para solucionar este problema, ya que GADEN necesita una malla 3D perfecta con tamaños de celda uniformes, se utiliza el visor de OpenFOAM, ParaView[5](Figura 3-7), mediante la cual se puede aplicar el filtro "centerCells" (seleccionado la opción "Vertex Cells" para poder mostrar el centro de las celdas como puntos en el marco de visualización de Paraview), y luego exportar los datos a un archivo CSV. Este archivo será una lista de puntos (x,y,z) con sus correspondientes condiciones de viento (u,v,w), simplificando mediante este método el problema de las dimensiones no uniformes de las celdas.

[5] https://www.paraview.org/

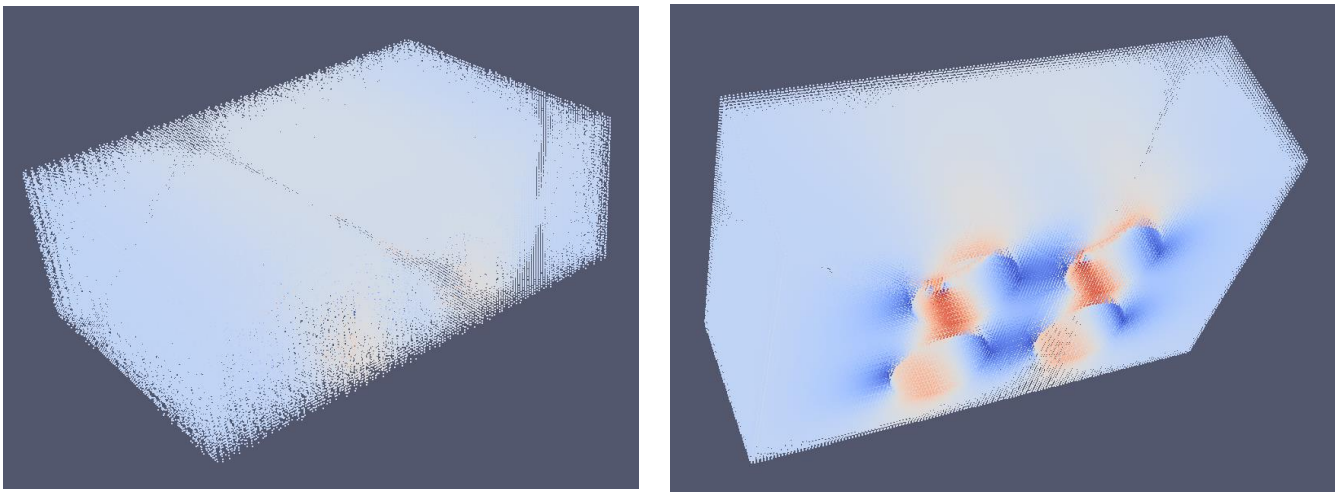


Figura 3-7. Visualización del resultado en la aplicación Paraview tras aplicar el filtro “centerCells”.

Etapa 3: Simulador de dispersión de gases

Una vez definido el entorno y generada la simulación de los patrones de flujo de viento, se procede a la simulación de dispersión de gases en 3D. Esta etapa se realiza mediante una implementación en C++ del modelo de dispersión atmosférica basado en filamentos. Las principales razones detrás de la elección de este modelo de dispersión específico son que representa un buen compromiso entre la complejidad y la potencia de cálculo, está diseñado para replicar las estadísticas de exposición a corto y largo plazo de un producto químico que evoluciona en un flujo turbulento, y no se basa en un modelo matemático de penacho, sino que se basa en el modelado de diferentes fenómenos físicos que ocurren durante la dispersión de gases. Todas estas características hacen que este modelo de dispersión sea ideal para los entornos complejos que suelen encontrarse en las aplicaciones robóticas, permitiendo una dispersión química variable en el tiempo y teniendo en cuenta la presencia de obstáculos. Sin embargo, hay que señalar que no tiene en cuenta los gradientes impulsados por la temperatura ni las fuerzas de gravedad, entre otros parámetros avanzados que pueden influir en la dispersión del gas.

El modelo de dispersión de filamentos establece que una liberación química puede representarse como una secuencia de ráfagas, cada una de ellas compuesta por n filamentos, y cada filamento se modela como una distribución normal 3D de moléculas. Las ráfagas son manipuladas por la difusión turbulenta y molecular, mientras que, al mismo tiempo, son transportadas por el fenómeno de advección por el viento. Se modelan tres fenómenos de dispersión según el tamaño relativo de la ráfaga con respecto a los remolinos de viento, como sigue:

- Los remolinos más grandes que la ráfaga la transportan en su totalidad. Este efecto se modela como un flujo de advección sobre los filamentos, actualizando su ubicación según las condiciones locales del viento (obtenidas de la simulación de viento CFD).
- Los remolinos del orden del tamaño de la ráfaga provocan un crecimiento/distorsión importante de la misma. Esto puede verse como un efecto difusivo sobre los filamentos de una ráfaga, y se modela como un ruido blanco aleatorio que afecta a su ubicación.
- Los remolinos más pequeños que la ráfaga mezclan sus componentes, causando poco movimiento o crecimiento de la misma. Esto representa el componente de difusión de las moléculas dentro de un filamento, siendo modelado como un crecimiento continuo pero lento del tamaño del filamento con el tiempo.

En cada iteración, el simulador actualiza la ubicación de los filamentos liberados de acuerdo con los fenómenos anteriores para estimar la nueva dispersión del gas en el entorno. Se considera un cuarto efecto de dispersión que tiene en cuenta la gravedad y las fuerzas de flotación que, según la sustancia química liberada, hacen que los filamentos decaigan o se eleven con respecto al campo de viento.

3.1.2 Implementación del simulador

El simulador de dispersión de gases GADEN ha sido implementado como un meta-paquete ROS y codificado en C++. El motivo es facilitar su integración con muchos otros módulos implementados en ROS, por ejemplo, RVIZ para visualizar los datos, NDT-MCL o RF2O para la localización del robot, o paquetes de olfato artificial como los paquetes KernelDM+V o GMRF para el mapeo de la distribución de gases. Concretamente, el software se ha organizado en cuatro subpaquetes que tienen diferentes propósitos y que se describen en detalle a continuación.

Filament Simulator Package (GADEN_preprocessing y GADEN)

Estos paquetes son el núcleo del simulador de dispersión de gases. Se encargan del movimiento de las ráfagas y de los filamentos en cada paso de tiempo, así como de la estimación de la concentración de gas en función de la localización y forma de los filamentos activos. Entre sus principales tareas destaca:

- Crear nuevos filamentos en función de la ubicación 3D de la fuente de gas, el tipo de gas y la tasa de liberación.
- Actualizar la ubicación de los filamentos en función de las condiciones locales del viento, la naturaleza del producto químico y la estructura del entorno, es decir, los límites y los obstáculos.
- Eliminar los filamentos que alcanzan las salidas definidas en el entorno simulado, evitando la acumulación indeseada de la concentración de gas en esos lugares y, en consecuencia, proporcionando una simulación más realista.
- Estimar la concentración de gas en cada punto del entorno (mapa cuadriculado 3D) a partir de los filamentos presentes y su forma y tamaño actuales.

La ejecución de estas tareas puede ser computacionalmente costosa; por lo tanto, este paquete está diseñado para ejecutarse previamente a la simulación, registrando en cada paso de tiempo el mapa 3D actual de las concentraciones de gas en un archivo.

Environment Pkg

El objetivo de este paquete es visualizar el entorno, es decir, para trazar la concentración de gas estimada en el contexto apropiado, este paquete carga el modelo CAD del entorno y lo traduce a la herramienta RViz para su visualización.

Player Pkg (GADEN_player)

Es el encargado de "reproducir" la simulación simplemente leyendo los resultados de la fase anterior y también proporcionando diferentes sensores simulados de gas y viento. Su principal utilidad es visualizar en RViz la dispersión del gas como una nube de puntos y hacer que los datos de gas y viento estén disponibles en la arquitectura ROS para otros paquetes, como la concentración de gas o el vector viento en una ubicación del espacio determinada.

Este paquete no tiene básicamente ninguna carga computacional, estando sólo limitado por el tamaño del entorno y la tasa de refresco deseada. Además, facilita la carga de múltiples mapas de cuadrícula de concentración de gas simultáneamente, lo que permite considerar múltiples fuentes de gas y diferentes especies químicas.

Simulated Sensor Pkg

Finalmente, para explotar el simulador de dispersión de gases, la herramienta proporciona diferentes emuladores de sensores que requiere el "Player pkg", como son el valor de la concentración de gas o el vector viento, simulando la salida que proporcionarán dichos sensores. Actualmente, se han implementado tres tecnologías de sensores:

- Los sensores de gas de óxido metálico son una de las tecnologías más comunes y ampliamente difundidas para la olfacción robótica. Estos sensores son muy sensibles, pero carecen de selectividad, por lo que el comportamiento simulado debe considerar la presencia de múltiples gases. Estos sensores se modelan siguiendo la ficha técnica del fabricante para estimar la relación de resistencia RS/R0 a partir de la concentración proporcionada por el simulador. A continuación, se aplica un filtro de paso bajo para simular los tiempos de respuesta de subida y bajada, y finalmente se estima la resistencia del sensor RS en

el tiempo t dada la resistencia de referencia R0.

- PID. Los detectores de fotoionización son otro tipo de sensor de gas que es muy fiable, proporciona la concentración de gas en unidades ppm y muestra una respuesta casi inmediata. Sin embargo, este tipo de detector no puede utilizarse para clasificar o reconocer el gas detectado, sino sólo para estimar la concentración, siendo la medida de concentración que proporcionan estos dispositivos un valor proporcional a la concentración real del gas por un factor que depende del tipo de gas.
- Anemómetro. GADEN también incluye un emulador de anemómetro ultrasónico para analizar la dirección y la fuerza del viento en cualquier lugar. Concretamente, modela un anemómetro 2D, que emite la velocidad y la dirección del viento. En la implementación actual no se modelan el ruido de las mediciones ni los intervalos de confianza del anemómetro.

Olfaction Msgs Pkg (olfaction_msgs[6])

Aunque GADEN es un paquete autocontenido, también depende de un paquete externo que define algunos mensajes relacionados con los datos que proporcionan los sensores emulados.

La Figura 3-8 muestra el resultado final del simulador en la que se puede ver una fuga de gas propagándose en el entorno diseñado y bajo las condiciones de contorno previamente seleccionadas.

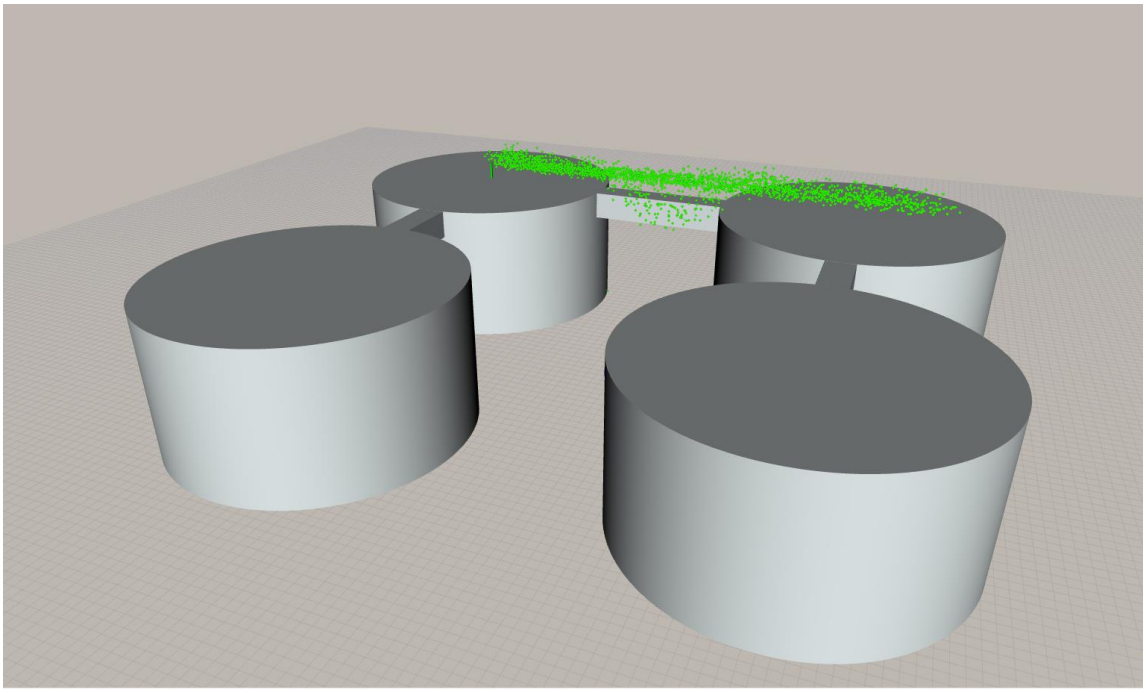


Figura 3-8. Visualización en RViz de la fuga de gas simulada, en un instante concreto, para las condiciones de contorno seleccionadas y en el entorno modelado anteriormente.

En resumen, GADEN es una herramienta modular desarrollada en ROS que permite que algoritmos de detección de gases y de robótica, como la navegación y la percepción 3D, pueden integrarse y validarse simultáneamente. GADEN, además, ofrece la posibilidad de importar sofisticados modelos CAD del entorno, a diferencia de otras herramientas de simulación existentes que solo permiten definir modelos geométricos simplistas del entorno. GADEN permite simular entornos en los que pueden estar presentes simultáneamente múltiples fuentes de gas de diferentes tipos. Esto es especialmente útil para validar los algoritmos de discriminación de gases y el mapeo de diferentes compuestos químicos.

La arquitectura dual de GADEN, es decir, los paquetes que se ejecutan de forma previa (offline) y los que corren en tiempo de ejecución (online), permiten aliviar las exigencias computacionales de la simulación de la dispersión de gases. En cambio, los motores de simulación existentes calculan la dispersión de gases mientras se ejecuta el simulador. El inconveniente de este enfoque es que, a medida que avanza el tiempo de

[6] https://github.com/MAPIRlab/olfaction_msgs

simulación, las demandas computacionales aumentan sustancialmente, limitando, de este modo, la usabilidad del simulador.

Según artículos de los autores, investigaciones futuras incluirán el modelado de las perturbaciones causadas por el movimiento del robot móvil en la dispersión de gases, así como la generación automática de los modelos CAD que definen el entorno de simulación mediante la generación de nubes de puntos densas a partir de cámaras RGB-D o lidar 3D (del inglés LiDAR, Light Detection and Ranging o Laser Imaging Detection and Ranging).

3.2 UAV

En los últimos años se ha producido un notable incremento en el número de aplicaciones de los vehículos aéreos no tripulados (UAV) debido a sus actuales niveles de autonomía y cognición, haciéndolos aptos para realizar muchas tareas diferentes que suelen abordarse mediante algoritmos de alto nivel.

Como consecuencia, la tecnología de los UAV avanza rápidamente lo que se ha traducido en un amplio espectro de plataformas y pilotos automáticos (software de a bordo para su funcionamiento básico) que se encuentran a disposición de la comunidad. Esta variabilidad también se debe a las limitaciones específicas asociadas a cada aplicación. Según el contexto, pueden ser necesarias plataformas con diferente capacidad de carga útil, maniobrabilidad o funcionalidades de piloto automático. No obstante, los algoritmos de alto nivel deben ser capaces de operar los UAV de forma transparente, independientemente del piloto automático o de la plataforma que haya por debajo.

Recientemente se han desarrollado varios frameworks para el funcionamiento de los UAV, tanto de propiedad como de código abierto. Además, algunas organizaciones de código abierto han propuesto protocolos de comunicación estándar para los UAV, como por ejemplo MAVLink, que a pesar del creciente uso de este protocolo, todavía hay muchos pilotos automáticos que no lo soportan. También hay frameworks más elaborados y complejos para tratar con equipos de múltiples robots aéreos o heterogéneos, o para proponer soluciones completas que incluyan la plataforma de hardware.

Estos sistemas de piloto automático permiten que un UAV, como un dron, realice misiones completas de forma autónoma sin la necesidad de un control remoto manual, tratando de abstraer al usuario del piloto automático de la plataforma. Estas misiones pueden incluir entrega de carga, mapeo, vigilancia y muchas otras aplicaciones. Los operadores utilizan estaciones de control en tierra para establecer los parámetros de la misión y el piloto automático dirige al dron u otra nave no tripulada para completar la tarea.

Los UAV completamente autónomos pueden llevar a cabo planes de vuelo completos, incluidos VTOL o despegues de pista, maniobras en vuelo y aterrizaje. Las opciones para pilotar los UAV manualmente o con una función manual asistida también están disponibles en algunos sistemas de piloto automático UAV además de la configuración totalmente automatizada. La mayoría de los pilotos automáticos de UAV tienen sistemas de redundancia únicos o múltiples para mantener el vehículo en vuelo y operativo en caso de fallo del sistema. Estos sistemas están disponibles para todo tipo de UAV, incluidos los de alas fijas, multirrotores, helicópteros drones, dirigibles, etc. Algunos fabricantes utilizan un conjunto común de hardware y software para operar todos estos vehículos mediante el uso de fases de vuelo personalizadas y canales de control. Otros pueden usar el mismo hardware con software diferente.

Al mismo tiempo, es esencial integrar los marcos de software de los UAV con los entornos de simulación para poder pasar de las plataformas simuladas a las reales con poco esfuerzo. Dado que los vehículos aéreos no tripulados son más sensibles y frágiles que los vehículos terrestres comunes, las simulaciones preliminares para verificar el correcto funcionamiento de todo el sistema son aún más críticas. Para ello, existen simuladores muy extendidos, como Gazebo, V-REP o AirSim, que permiten a los investigadores probar sistemas UAV y multi-UAV, para la asignación de tareas o algoritmos de planificación de trayectorias, entre otros.

En este trabajo se utiliza la UAL[7] (del inglés, Unmanned aerial vehicle Abstraction Layer) desarrollada por el Grupo de Investigación de Robótica, Visión y Control (GRVC[8]) de la Universidad de Sevilla (US), un

[7] https://github.com/grvcTeam/grvc-ual
[8] https://grvc.us.es/

framework para el funcionamiento de UAV que proporciona a los usuarios una interfaz abstracta independiente del piloto automático.

Esta UAL proporciona una forma sencilla de simular múltiples UAV y su interfaz hace que el uso de robots reales o simulados sea transparente para el usuario [34]. Además, está basada en el framework de ROS, lo que facilita a los desarrolladores la tarea de crear aplicaciones robóticas más sofisticadas.

La interfaz de la UAL incluye un conjunto relevante de funcionalidades que todo UAV debería implementar [35].

- Realizar una maniobra de despegue a una altura determinada.
- Ir desde la posición actual a un waypoint especificado en coordenadas geográficas (o cualquier otra global) con una guiñada especificada.
- Establecer las velocidades lineales y la tasa de guiñada.
- Aterrizaje en la posición actual.
- Recuperación del modo de vuelo manual.
- Establecer la posición de inicio en la posición actual.
- Obtención de la última estimación de pose del UAV.
- Obtención de la última estimación de velocidad del UAV.

Además de la ventaja principal de abstraer a los usuarios del piloto automático, la UAL también proporciona herramientas que ayudan a los usuarios a probar fácilmente sus algoritmos en simulación. En concreto, la UAL está totalmente integrada con el conocido simulador de robots de código abierto Gazebo[9]. Este simulador permite la creación rápida de prototipos de robots y la creación de nuevos escenarios, y ya está integrado en ROS.

UAL viene con dos posibilidades de simulación en Gazebo: una simulación ligera y la simulación de software en bucle (SITL, del inglés Software in the Loop) de PX4. La primera utiliza un backend Light que proporciona un modelo simple del UAV (ver modelo utilizado en la Figura 3-9), evitando la dinámica, y muestra el modelo en Gazebo. Esto es particularmente útil para realizar simulaciones con un gran número de UAVs, donde la atención se centra en el comportamiento de alto nivel y no en tener una dinámica realista para los UAVs, lo que puede implicar problemas computacionales.

La segunda opción de simulación se basa en el firmware PX4, que es un software de piloto automático de código abierto. Además de las funciones habituales del piloto automático, el firmware PX4 incluye un entorno de simulación SITL basado en Gazebo y RotorS. Este SITL cuenta con varios plugins de Gazebo que simulan los sensores (por ejemplo, IMU, GPS, etc.) y la dinámica (por ejemplo, las velocidades y fuerzas del rotor) del UAV. La UAL viene con la posibilidad de ejecutar simulaciones SITL con el PX4 SITL, utilizando el backend MAVROS. Esta característica permite a los usuarios ejecutar en la simulación el mismo software que en la plataforma real, replicando los comportamientos de bajo nivel del piloto automático (como las transformaciones de waypoints y el cambio de modo) de una manera más realista.

[9] http://gazebosim.org

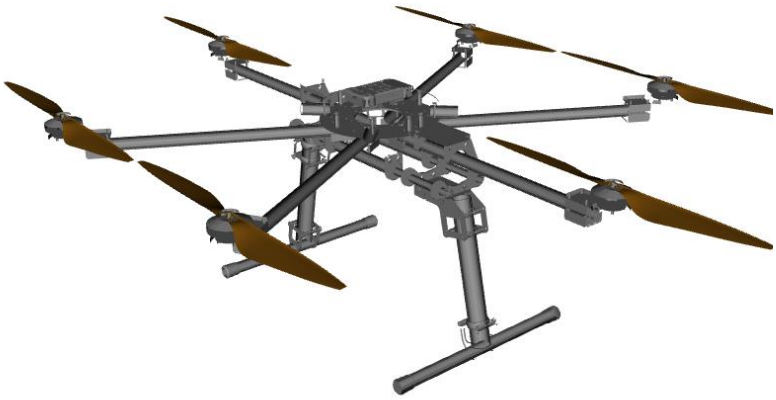


Figura 3-9. Modelo MBZIRC del framework grvc-ual

Con estas dos herramientas, se despliega el simulador final compuesto, por un lado, por la fuga de gas, y, por otro, el UAV, como se puede ver en la Figura 3-10. En este punto, los sensores emulados por GADEN, el sensor de concentración de gas y el anemómetro, se colocan respectivamente uno sobre el UAV acompañándolo en su movimiento, y el otro, en un punto más o menos realista de la planta de manera que facilita la información sobre el viento en la instalación.

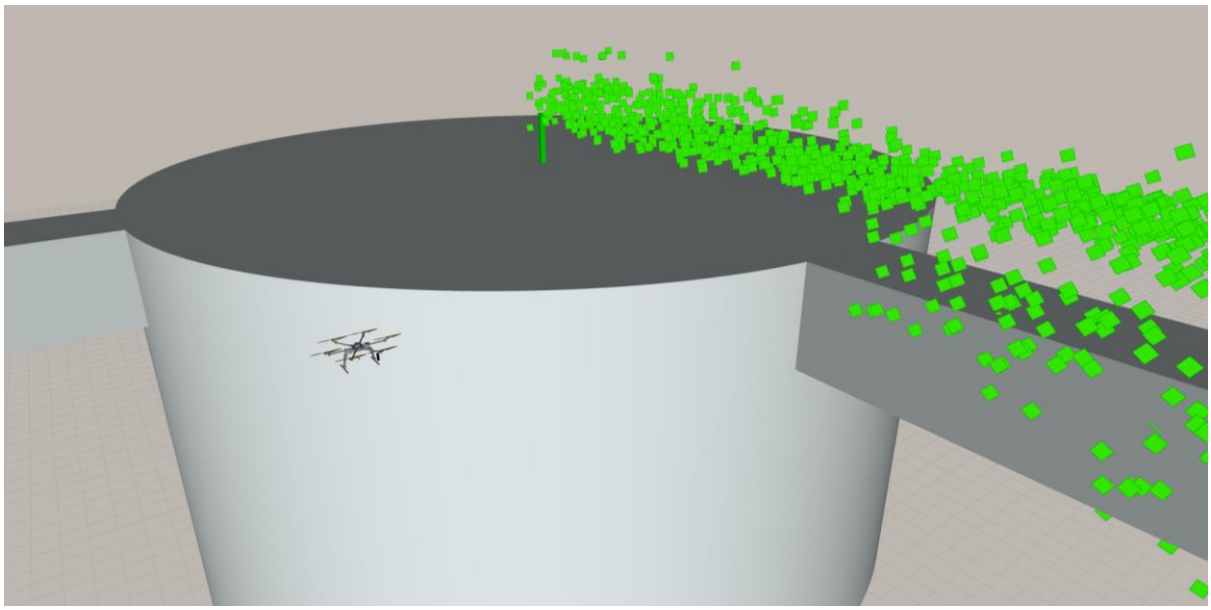


Figura 3-10. Visualización en RViz donde se pueden observar los componentes del simulador: la fuga de gas y el UAV.

Una vez desplegado el simulador, se procede ahora a desarrollar los algoritmos de alto nivel que van a gobernar el movimiento autónomo del UAV en su tarea de exploración del entorno en busca de la fuente de emisión de gas.

4 ALGORITMOS DE LOCALIZACIÓN

Como ya se ha comentado en secciones anteriores, el proceso de localización de fuentes de gas se divide en tres etapas: en primer lugar, la exploración del entorno en busca de la presencia de una emisión química, a continuación, la búsqueda de la fuente guiada por estímulos químicos y otros estímulos sensoriales, y, finalmente, la propia localización de la fuente, es decir, la verificación de la ubicación de la fuente identificada.

La primera etapa puede considerarse un evento desencadenante, que no suele llevarse a cabo como una etapa activa, sino que se ejecuta en segundo plano mientras el robot realiza otras operaciones no relacionadas, como patrullar, explorar, entregar, etc. Si mientras se realizan estas tareas, el sensor a bordo del robot detecta un nivel de concentración anormal, entonces se activa la tarea de localización de la fuente de gas y comienza el proceso de búsqueda.

4.1 Estrategias de búsqueda de la fuente

Esta etapa implica la búsqueda del punto de liberación de gas, basándose principalmente en los datos químicos detectados, pero también en el flujo de viento o en los objetos y la estructura del escenario. El éxito de esta etapa depende en gran medida de lo bien que se ajuste el algoritmo a las condiciones ambientales, que determinan la forma en que se dispersa el olor. En este sentido, las estrategias de localización de fuentes de gas pueden clasificarse en dos tipos [36]: aquellas diseñadas para funcionar bajo la presencia de un penacho, también conocidas como estrategias de seguimiento de penachos, y aquellas que no dependen de la existencia de un penacho bien formado y a favor del viento, es decir, están dominadas por la turbulencia.

- Seguimiento del penacho

A valores medios de Reynolds, la dispersión de los olores se produce principalmente a través del transporte debido a las corrientes (advección), lo que hace que se forme un penacho de olor a favor del flujo de la fuente. En la mayoría de los escenarios reales, este penacho no es recto y continuo, sino que, dada la naturaleza variable en el tiempo de los campos de flujo y el predominio de la turbulencia sobre la difusión, los penachos tienden a serpentear, se vuelven irregulares y, en menor medida, a dispersarse. Esto da lugar a valores máximos de concentración mucho más elevados que la media, y a fluctuaciones en los gradientes instantáneos en magnitud y dirección. Además, la estructura del penacho puede incluso cambiar si la dirección del flujo de aire se desplaza considerablemente. Por lo tanto, los fallos ocasionales son casi inevitables en el rastreo de penacho, siendo clave para el éxito no sólo el rastreo del penacho sino también los mecanismos de recuperación para reubicar el penacho perdido en caso de fallo.

La investigación sobre estrategias robóticas de seguimiento de penachos comienza con enfoques puramente quimiotácticos. Estos algoritmos implican la toma de medidas de la concentración química en más de una posición espacialmente separada, y la determinación del gradiente químico, que se utiliza para desplazarse hacia la fuente. Las estrategias incluyen enfoques en los que se utiliza un par de sensores químicos bilaterales, cada uno de los cuales controla directamente la velocidad de una rueda, ya sea la rueda opuesta en acoplamiento cruzado, o la rueda del mismo lado, para llegar a la fuente; algoritmos basados en la bacteria E. coli; y otros métodos que explotan las obstrucciones del flujo de aire generadas por la forma del robot, y la consiguiente disparidad entre las respuestas de los sensores situados a barlovento y a sotavento, para determinar la dirección de desplazamiento.

Los resultados experimentales han demostrado que los enfoques quimiotácticos puros son relativamente ineficaces en condiciones reales, atribuyendo su bajo índice de éxito a la susceptibilidad de los algoritmos a las rápidas fluctuaciones de la concentración química y al hecho de que el gradiente de concentración a lo largo de

la línea central del penacho es extremadamente pequeño, excepto en las proximidades de la fuente.

En este punto, se mejora el rendimiento de las estrategias combinando la quimiotaxis con la anemotaxis, que consiste en aprovechar la información que aporta la dirección del flujo de viento.

Se han propuesto varios métodos para explotar la información procedente de las mediciones de la concentración química datos anemométricos, entre los que se encuentra el método del del zigzag [36], que consiste en desplazarse en sentido ascendente dentro de la pluma de olor de forma zigzagueante. Cada vez que se encuentra el límite de la pluma, el robot vuelve a entrar en ella. La estrategia se describe gráficamente en la Figura 4-1.

Para tener en cuenta la irregularidad de la señal, un robot considera que ha abandonado el penacho sólo después de no encontrar ningún rastro durante un tiempo determinado; de este modo, se ignoran las disminuciones transitorias de la concentración medida.

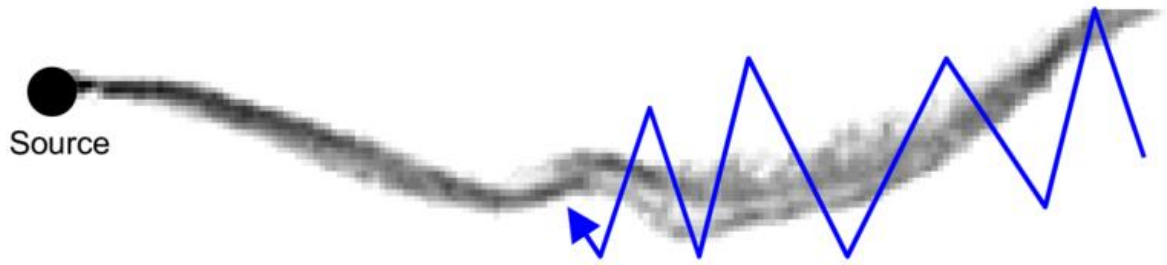


Figura 4-1. Algoritmo de localización basado en el movimiento de zigzag (seguimiento del penacho) [37].

Como se informa, el principal inconveniente de estas estrategias anemotácticas es el hecho de que, en entornos realistas y turbulentos, existe una gran dificultad para determinar con precisión la dirección del flujo del viento. Esto último conlleva, en la mayoría de las ocasiones, un gran impacto en la tasa de éxito cuando se compara con los resultados de laboratorio. Además, los obstáculos no sólo influyen negativamente en el rendimiento, sino que obligan, en muchos casos, a modificar sustancialmente el algoritmo.

- Búsqueda dominada por la turbulencia

Para escenarios de interior, la baja fuerza de los flujos de viento no garantiza la formación de penachos químicos, siendo el movimiento de otras entidades (por ejemplo, personas, o el propio robot) el responsable de la dispersión caótica de los gases. Esto implica que la mayoría de los algoritmos de seguimiento de penachos son incapaces de localizar la fuente, ya que no existe un penacho químico que seguir.

Los investigadores han intentado superar esta limitación desarrollando sistemas que van más allá del control puramente reactivo, por ejemplo, explotando otras fuentes de información disponibles en el robot móvil. Una de las primeras alternativas que aparecieron fue la combinación de estrategias tradicionales con sistemas basados en la visión. Este enfoque permite a los robots identificar a los candidatos a distancia, con lo que se reduce drásticamente el espacio de búsqueda efectivo y se mejora en gran medida la capacidad de localizar una fuente de olor cuando una pluma a favor del viento no está bien formada o incluso es inexistente.

Se han presentado también algunas estrategias para la localización de plumas, como la fluxotaxis, un enfoque basado en múltiples robots en el que se aplican técnicas de dinámica de fluidos computacional, la infotaxis, un método sin gradientes que explota la entropía esperada de las muestras futuras para guiar la búsqueda del robot, o el algoritmo SPIRAL (del inglés, Searching Pollutant Iterative Rounding ALgorithm) [38], una estrategia de localización de fuentes para entornos interiores sin fuertes corrientes de aire.

La idea de SPIRAL consiste en realizar movimientos en espiral. Durante los movimientos, el robot se detiene para tomar muestras de gas y con los datos adquiridos, calcula su proximidad a la fuente: si descubre que está más cerca de la fuente en comparación con las mediciones anteriores, el robot inicia un movimiento en espiral más, de lo contrario, continúa en el actual. De este modo, las espirales se propagan para acercarse a la fuente (ver Figura 4-2).

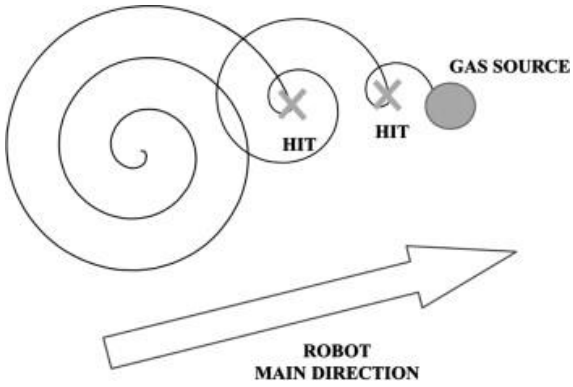


Figura 4-2. Algoritmo de localización basado en el movimiento de SPIRAL (búsqueda dominada por la turbulencia) [38].

4.2 Declaración de la fuente

La declaración de la fuente química corresponde a la última fase de la tarea de localización de la fuente de gas. Su objetivo es inspeccionar las localizaciones candidatas proporcionadas por el algoritmo de búsqueda, y verificar la presencia de la fuente de gas. En muchos casos, esta tarea se relega a un operador humano, que analizando los datos ofrecidos por el robot (por ejemplo, la trayectoria del robot seguida durante la fase de búsqueda, el mapa de distribución de gas creado, o simplemente las mediciones actuales de gas y viento) se encarga de declarar la localización de la fuente de gas, o un fallo en la fase de búsqueda.

Sin embargo, existen algunos enfoques para declarar la ubicación de la fuente de gas de forma autónoma. Los primeros trabajos basaban la declaración de la fuente en el hecho intuitivo de que la concentración de gas en las proximidades de la fuente de gas debería ser mayor. Algunos enfoques proponen estrategias heurísticas basadas en la saturación de los sensores de gas por encima de los umbrales establecidos empíricamente, mientras que otros afinan la declaración comparando la concentración química a diferentes alturas, o también, un método que identifica la fuente mediante la detección de una serie de respuestas de olor contiguas, siempre que sea probable que esa frecuencia de puntos de olor se produzca sólo cerca de la fuente.

Se propusieron métodos más sofisticados basados en el aprendizaje automático para mejorar la declaración de la fuente de gas en entornos más realistas. La declaración se abordó como un problema de clasificación, utilizando redes neuronales y máquinas de vectores de soporte. El problema con estas y otras estrategias similares es la necesidad de una fase de entrenamiento, y la suposición de que las características de la fuente durante el entrenamiento serían similares a las fuentes de olor encontradas durante la operación normal. Para terminar, existen otros que proponen un enfoque de filtro de partículas para declarar la fuente de gas.

4.3 Análisis comparativo

Una vez descritos los diferentes tipos de estrategias que se pueden adoptar para abordar el problema de localización de fuentes de gas, el objetivo en este punto del trabajo es realizar un análisis comparativo entre los dos tipos de estrategias. Para ello se ha elegido y realizado la implementación de un algoritmo característico de cada categoría y poder así evaluar el rendimiento de cada una. Los algoritmos elegidos son: zigzag (estrategia de seguimiento del penacho) y SPIRAL (estrategia de búsqueda dominada por la turbulencia).

4.3.1 Banco de pruebas

Para alcanzar este objetivo, se ha implementado en primer lugar un “banco de pruebas” en MATLAB, que sirva para evaluar el desempeño de los diferentes algoritmos. Esta herramienta parte de los datos generados por el simulador GADEN, de manera que los archivos que contienen la información del entorno para cada paso de tiempo (concentración y vector de viento) son importados y utilizados ahora en la herramienta para

simular las mismas condiciones realistas de la simulación.

En la Figura 4-3 se puede comprobar la distribución de la concentración de gas a lo largo del penacho generado por el flujo de aire, en un instante concreto de la simulación (iteración 315).

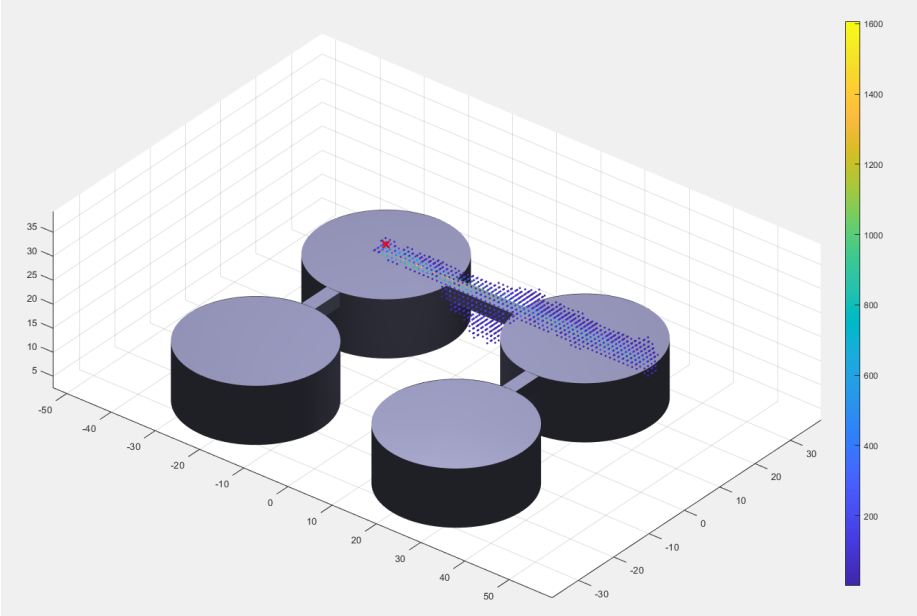
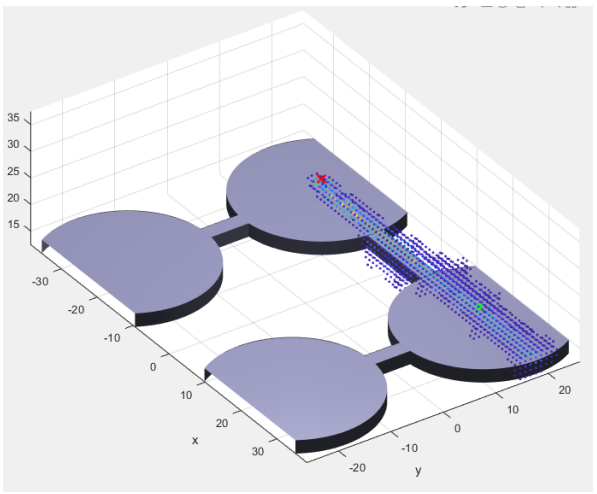
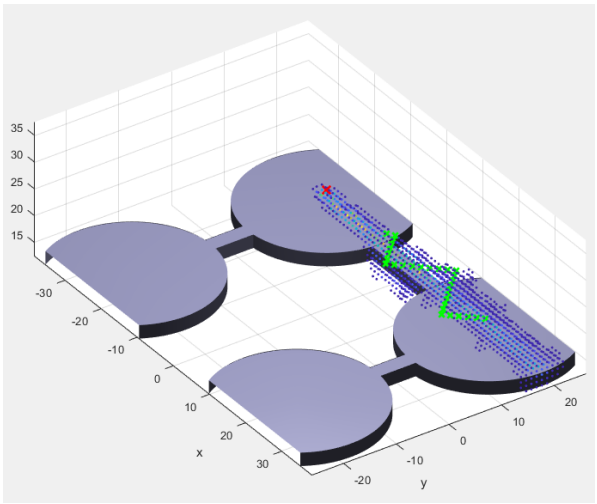


Figura 4-3. Representación de la distribución de concentración correspondiente a la iteración 315 de la simulación en la herramienta de pruebas.

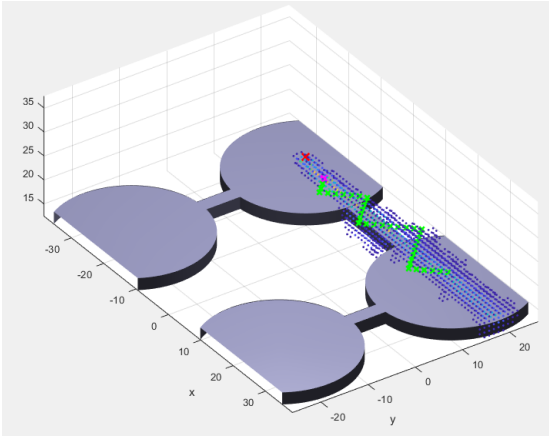
Una vez desarrollado el banco de ensayos, se lleva a cabo la implementación de los algoritmos antes mencionados, zigzag y SPIRAL, y posteriormente, se ponen a prueba bajo las condiciones realistas del entorno simuladas en la herramienta. En las siguientes Figuras 4-4 y 4-5 se puede observar la evolución del movimiento para cada uno de los algoritmos.



(a)

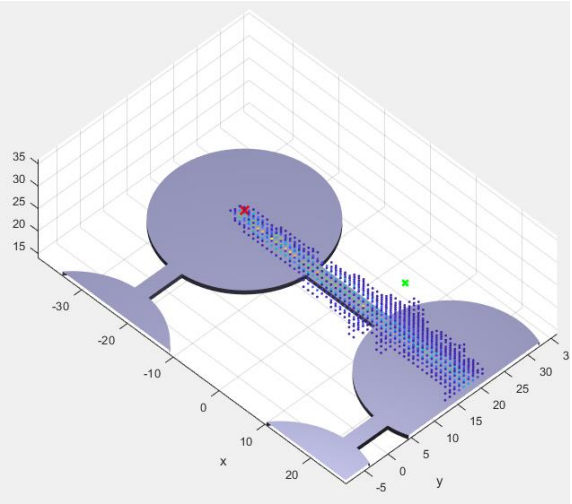


(b)

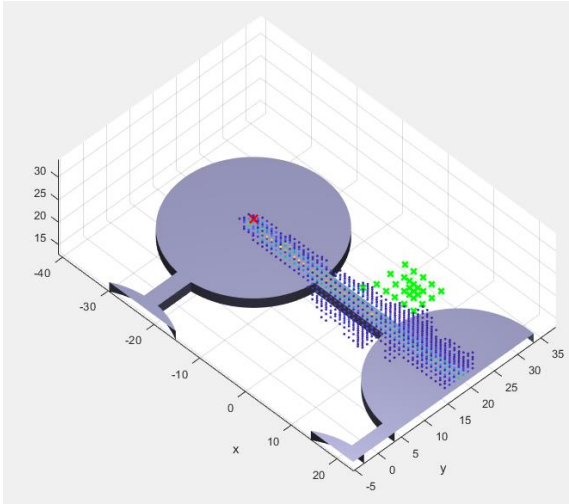


(c)

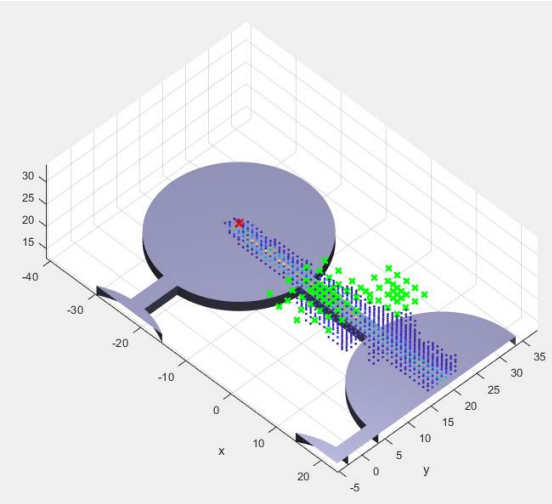
Figura 4-4. Evolución del algoritmo de seguimiento del penacho, zigzag, a lo largo del tiempo, para un estado estacionario de la fuga de gas.



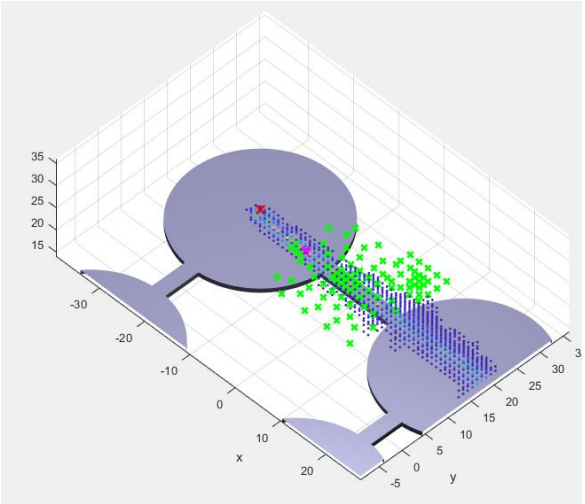
(a)



(b)



(c)



(d)

Figura 4-5. Evolución del algoritmo de búsqueda dominada por la turbulencia, SPIRAL, a lo largo del tiempo, para un estado estacionario de la fuga de gas.

En este punto, el rendimiento de los algoritmos se compara en el banco de ensayos en base a diferentes métricas como son: el número de puntos total explorados, que hace referencia al coste tanto en distancia recorrida como el tiempo requerido el algoritmo para encontrar la fuente, y la precisión a la hora de declarar la fuente de gas respecto al punto real de emisión.

Tabla 4–1. Comparación de los algoritmos de localización en el banco de ensayos.

	Zigzag	SPIRAL
Número de iteraciones (coste, distancia recorrida)	37	105
Distancia a la fuente de emisiones (precisión) Fuente (x,y,z) [metros] = (-22.5, 18.5, 17)	Posición estimada = (-16, 17.97, 16)	Posición estimada = (-12, 17.97, 16)

A la vista de los resultados, el algoritmo que presenta un mejor rendimiento es el que se basa en el seguimiento del penacho, tanto en la distancia recorrida como en la precisión a la hora de declarar la fuente. Estos resultados son esperables debido a las condiciones ambientales y el entorno bajo los cuales se está simulando la fuga de gas: un entorno al aire libre sin obstáculos que interrumpen el penacho y donde existe una corriente de aire en una dirección determinada. Otros experimentos que se pueden realizar son aquellos en los que se incluyan obstáculos intermedios que desvíen la dirección de la pluma de gas o incluso en entornos cerrados con restricciones de movimiento (habitaciones, pasillos, etc.). Esto último hace que el penacho se vuelva más puntiagudo y, por lo tanto, es muy común que se pierda el penacho, siendo necesario contar con mecanismos adecuados de recuperación del mismo.

Finalmente, tras las conclusiones obtenidas, se implementa el algoritmo de zigzag, ahora, en un nuevo paquete de ROS independiente, “odor_source_localization”. La función de este paquete es gobernar el movimiento autónomo del UAV según la heurística elegida, dirigiendo la búsqueda hacia la fuente de emisiones de olor, todo ello integrado en el mismo entorno de simulación de la fuga de gas.

Por último, señalar que la lógica correspondiente a la estrategia implementada en el entorno de ROS es la misma que en el banco de ensayos, la diferencia ahora es que mientras que los valores de concentración y viento para cada punto en la herramienta de pruebas se obtenían directamente a partir de los archivos de simulación, la información se recibe ahora de los sensores emulados que están desplegados en el simulador (MOX y anemómetro) a través de los datos que publican en los topics de ROS.

5 PLATAFORMA IOT

En este apartado se describe el tercer pilar tecnológico sobre el que se basa la solución final, una Plataforma IoT, que culmina el desarrollo del Gemelo Digital del proceso. El objetivo es conectar todos los dispositivos a esta plataforma de manera que se pueda llevar un seguimiento en tiempo real de las variables clave del sistema, así como también ejecutar órdenes de control sobre los equipos, acciones que, gracias a la definición conceptual de IoT, se pueden llevar a cabo desde cualquier punto que se encuentre conectado a la misma plataforma.

Tras la investigación del estado del arte realizada sobre la literatura actual en apartados anteriores, se opta en este trabajo por utilizar la tecnología FIWARE para el despliegue de dicha plataforma. En los siguientes apartados se detallan los principios fundamentales sobre los que se construye esta tecnología y finalmente se explica la arquitectura final de la plataforma desplegada.

5.1 FIWARE

Durante la última década, numerosos objetos físicos inteligentes, también llamados “cosas”, o en inglés, "Things", se han conectado y comunicado a través de Internet para formar la red global de dispositivos conectados denominada Internet de las Cosas. Estos dispositivos físicos inteligentes suelen contener sensores que pueden detectar, recoger y comunicar datos, y/o actuadores que pueden reaccionar a señales de control internas y externas. Las características de los dispositivos conectados permiten que las aplicaciones de IoT supervisen, agreguen, analicen, ofrezcan información empresarial, aumenten la eficiencia y proporcionen decisiones más informadas. De hecho, estas tecnologías del IoT están penetrando de forma ubicua en una amplia gama de aplicaciones en múltiples dominios y están revolucionando casi todos los aspectos de la sociedad, como la sanidad, la agricultura, el transporte, la previsión meteorológica, el hogar y las ciudades inteligentes, etc. Para la industria, el IoT ofrece el potencial de combinar la interacción máquina a máquina, la recogida de datos en tiempo real y el análisis de Big Data, permitiendo mejorar aún más la optimización dinámica, el control y la toma de decisiones basada en datos.

En este contexto, un sistema ciberfísico es un sistema mecánico en el que los componentes físicos (sistemas mecánicos, sistemas eléctricos, operadores humanos, etc.) y los componentes cibernéticos (comunicación, computación, control, etc.) están profundamente entrelazados, y los diferentes componentes interactúan entre sí para intercambiar información. Existe una gran adopción de los CPS en la industria manufacturera, que en ocasiones se denominan sistemas de producción ciberfísicos (CPPS, del inglés Cyber-physical Production Systems). Un CPPS suele consistir en una planta física acoplada a un gemelo digital, que utiliza los datos de supervisión recogidos en la planta para construir una réplica digital como "gemelo" de la planta real. De esta manera, el CPPS permite tomar decisiones más contrastadas al combinar el conocimiento de las condiciones disponibles en la planta física y el análisis de Big Data.

Aunque los conceptos de IoT y CPS muestren orígenes distintos, ya que el IoT surge principalmente desde la perspectiva de las tecnologías de la información y la comunicación (TIC), mientras que CPS ha surgido principalmente de la ingeniería y el control de sistemas, estos conceptos se solapan, ya que ambos se ocupan de conectar e interactuar con los objetos físicos y con las entidades digitales para realizar la funcionalidad y mejorar el rendimiento. El Internet de las Cosas Industrial (IIOT) y los CPS son, conjuntamente, elementos clave para acelerar la actual transformación digital de las empresas en la llamada cuarta revolución industrial (Industria 4.0).

El problema es que hay gran número de dispositivos IoT que se están conectando a Internet a pesar de que no se ha realizado una estandarización mundialmente aceptada de los protocolos ni de los ecosistemas. Además de conectar los dispositivos a través de una infraestructura de Internet, también es necesario comprender y utilizar el gran volumen de datos procedentes de distintas fuentes. Esta heterogeneidad de IoT tiene varias

dimensiones, incluyendo la heterogeneidad de los dispositivos, las tecnologías de comunicación, los protocolos, el formato de los datos, etc.

Para superar esta heterogeneidad, el mundo académico y la industria están haciendo hincapié en la importancia de la interoperabilidad, un concepto que puede generalizarse como el intercambio de información sin fisuras, la comprensión de la información compartida y la utilización del conocimiento compartido para habilitar servicios y mejorar el rendimiento. La falta de interoperabilidad es perjudicial tanto para los usuarios como para los proveedores:

- Desde el punto de vista de los usuarios, la falta de interoperabilidad significa que los servicios están ligados a los dispositivos y al software proporcionado por un único proveedor, lo que conlleva el riesgo de verse obligado a quedarse con el mismo proveedor y unos costes más elevados posteriormente.
- Desde el punto de vista de los proveedores, la incompatibilidad entre diferentes plataformas puede proteger temporalmente el entorno interno, pero es perjudicial para el mercado global del IoT, ya que la tendencia es que cada vez más aplicaciones operen en múltiples plataformas y utilicen servicios entre dominios.

Diversas organizaciones y grupos de normalización tratan de enriquecer la interoperabilidad promoviendo marcos de referencia comunes. Sin embargo, como todavía no se ha acordado una norma común, han surgido diversos marcos de referencia, cada uno de ellos definido por expertos de diferentes países y ámbitos.

Un ejemplo es FIWARE[10], un framework de código abierto que propone un conjunto de estándares para acceder y gestionar información de contexto heterogénea a través de una API abierta, apoyando el desarrollo de soluciones inteligentes más allá de los casos de uso de IoT. De hecho, FIWARE es una iniciativa financiada por la Unión Europea [39] resultado de un esfuerzo conjunto de diferentes organizaciones por encontrar mecanismos comunes que ayuden a las ciudades, la industria y las comunidades a ser interoperables. Esta plataforma ha ganado relevancia debido a su enfoque neutral con respecto a los proveedores, definiendo estándares que hacen posible que todo tipo de empresas e industrias construyan y compartan soluciones inteligentes portátiles e interoperables en todo el mundo.

Figura 5-1. Logo de FIWARE.

En plena consonancia con los objetivos de la estrategia del Mercado Único Digital (MUD), FIWARE garantiza la portabilidad, interoperabilidad y apertura de los servicios en toda Europa, contribuyendo especialmente a la digitalización de la industria europea, y, además, cuenta con un ecosistema maduro y colaborativo de desarrolladores, centros de innovación, aceleradores, ciudades y más de 1.000 PYMEs y startups.

Con presencia en todo el mundo gracias a su programa Mundus, FIWARE mejora la cohesión social fomentando la creación de empleo y el crecimiento, y ahora se traslada al mercado con el apoyo de la Fundación FIWARE. La Fundación se está convirtiendo en el actor central que impulsa la tecnología y el ecosistema de la plataforma y hace que las empresas, las instituciones y los contribuyentes sean muy visibles en una iniciativa clave de la DG CONNECT y la Comisión Europea [40].

FIWARE ofrece una serie de ventajas frente a otras plataformas que son importantes a la hora de desarrollar aplicaciones, las cuales se enumeran a continuación:

- Fácil de implementar.

[10] https://www.fiware.org/

- Garantiza la interoperabilidad.
- Fácilmente escalable.
- Interfaces de Programación de Aplicaciones abiertas.
- Permite el desarrollo de aplicaciones muy potentes y alimentadas en tiempo real.
- Puesta en marcha rápida y eficiente.
- Fácilmente replicable.
- Facilita la información de contexto de manera masiva.

A grandes rasgos, FIWARE se compone de un conjunto de componentes de propósito general denominados habilitadores genéricos (GEs, del inglés Generic Enablers), que se utilizan conjuntamente, permitiendo el desarrollo de redes inteligentes. Estos GEs cubren un gran número de funcionalidades, como el procesamiento y almacenamiento de eventos de datos, la conexión fácil para los dispositivos utilizando una amplia variedad de protocolos (UL, JSON, LWM2M), la seguridad (autenticación y autorización), la publicación y monetización de los recursos de datos contextuales, y el procesamiento y visualización de datos. En cualquier red IoT es necesario recoger, gestionar, procesar y mostrar la información del entorno, y en el caso de FIWARE, ese papel lo desempeña el Orion Context Broker, convirtiéndolo en el núcleo de cualquier plataforma construida con este software. Alrededor de este, hay un conjunto de componentes adicionales, como se puede ver en la Figura 5-2, que ofrecen una gran variedad de nuevas funcionalidades al sistema, como soporte de procesos, soporte de visualización, soporte de control de acceso, etc. La Interfaz de Servicios de Próxima Generación (NGSI) es la API RESTful estandarizada utilizada para las interacciones entre el Context Broker y otros componentes. La API NGSI define un modelo de datos, una interfaz de datos de contexto y una interfaz de disponibilidad de contexto. Las especificaciones actuales de los modelos de datos de referencia de FIWARE son NGSI-v2 y NGSI-LD.

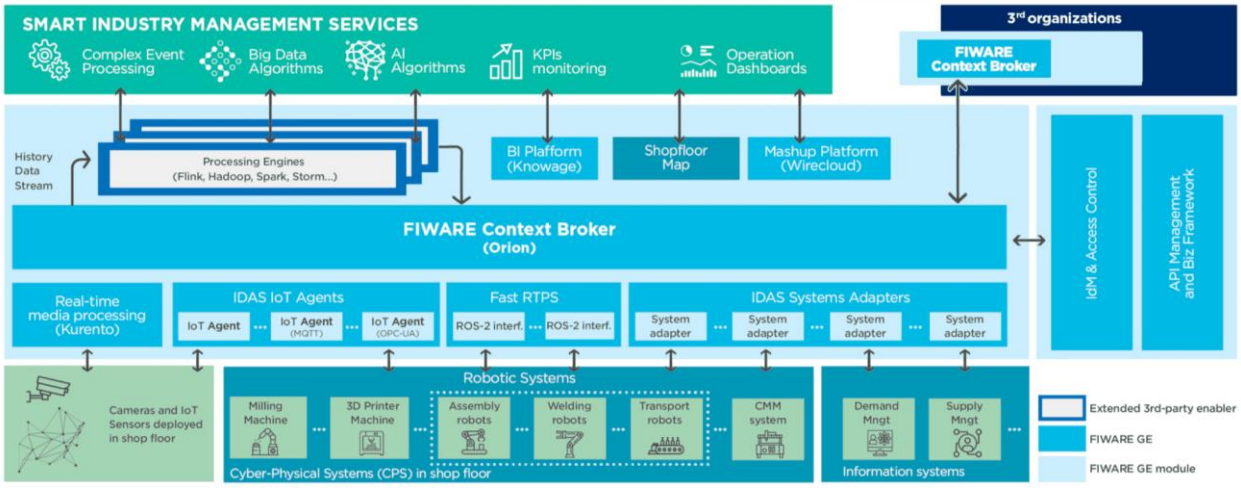


Figura 5-2. Arquitectura de la plataforma FIWARE.

5.1.1 Datos de contexto

El modelo de datos de contexto NGSI de FIWARE está definido en las especificaciones de la API abierta NGSIv2. Este modelo de datos consta de tres elementos principales (ver Figura 5-3): entidad, atributo y metadatos, que se representan mediante objetos JSON.

La entidad es el elemento central del modelo de datos de contexto y representa un objeto físico o lógico, por ejemplo, un sensor, una máquina, una sala, una alarma en un sistema, etc. Cada entidad se identifica de forma única mediante una combinación de id de entidad y tipo de entidad, donde el tipo describe el tipo de cosa representada. Por ejemplo, una entidad con id "KistaObserved_001" podría ser del tipo "EnviromentObserved".

Los atributos de contexto describen las propiedades de una entidad de contexto. Una entidad puede tener múltiples atributos, por ejemplo, "Temperatura" y "Humedad" pueden ser los atributos de la entidad "KistaObserved_001". Un atributo se caracteriza por su nombre, tipo y valor, donde el tipo es el tipo de valor NGSI del valor del atributo y el valor de los atributos contiene los datos reales de los mismos. NGSI tiene su propio sistema de tipos de valores para los valores de los atributos, de modo que los tipos de NGSI no son los mismos que los tipos de JSON.

Por último, también se pueden añadir metadatos opcionales a un valor de atributo cuando sea necesario. Los metadatos de contexto describen las propiedades del atributo., que se caracterizan por su nombre, tipo y valor, de forma similar a los atributos de contexto. Por ejemplo, el atributo "Temperatura" puede tener metadatos de "precisión" y "marca de tiempo" dentro de su valor. El tipo de metadatos es un valor NGSI del tipo del valor de los metadatos.

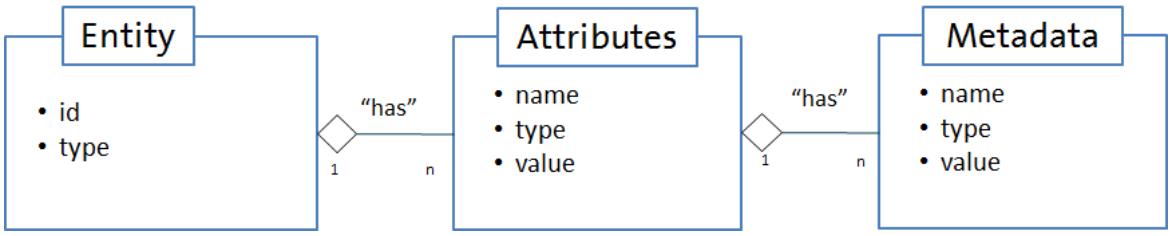


Figura 5-3. Esquema del modelo de datos de contexto NGSIv2.

5.1.2 Generic Enablers

La plataforma FIWARE está formada por un conjunto de elementos llamados “Generic Enablers”, o habilitadores genéricos, los cuales son considerados de propósito general e independientes. Estos elementos, ofrecen interfaces abiertas, que pueden ser modificadas por cualquier persona para su propio uso, tanto para el desarrollo de aplicaciones mediante APIs, como para la interoperabilidad con otros GEs.

Para el despliegue del ecosistema FIWARE, lo que se hace es ensamblar una serie de habilitadores, facilitando así la creación de aplicaciones web inteligentes. Estos componentes ofrecen funciones comunes y reusables usadas en múltiples casos de uso en varios sectores.

Los GEs se pueden dividir en cinco categorías [41]:

1. Core Context Management

   1.1. Core Context Broker components

   El Core Context Broker es el componente central y obligatorio de cualquier plataforma o solución "Powered by FIWARE". Permite gestionar la información de contexto de forma altamente descentralizada y a gran escala. El Orion Context Broker (OCB) Generic Enabler proporciona actualmente la API FIWARE NGSI v2.

   Otras implementaciones del Context Broker que soportan la especificación ETSI NGSI-LD 1.3.1. o superior están en fase de desarrollo:

   - El Orion-LD Context Broker es un Broker NGSI-LD, que soporta tanto NGSI-LD como las APIs NGSI-v2.
   - El Scorpio Broker es un Broker NGSI-LD alternativo que también puede utilizarse en entornos federados.
   - Stellio Context Broker es otra alternativa de NGSI-LD Broker.

   1.2. Core Data Connectors

   Acompañando a un componente Context Broker como parte del Core Context Management, hay una serie de Conectores de Datos disponibles:

   - STH Comet aporta los medios para almacenar un historial de datos de contexto a corto plazo en

MongoDB.

- Cygnus permite un medio para gestionar el historial de contexto que se crea como un flujo de datos y puede ser inyectado en múltiples sumideros de datos, incluyendo muchas bases de datos populares como PostgreSQL, MySQL, MongoDB o AWS DynamoDB o plataformas de Big Data como Hadoop, Storm o Spark. El componente se basa en Apache Flume.
- Draco es un mecanismo alternativo de persistencia de datos para gestionar el historial del contexto. Se basa en Apache NiFi y es un sistema de flujo de datos basado en los conceptos de la programación basada en flujos. Soporta gráficos de enrutamiento de datos potentes y escalables, transformación y lógica de mediación del sistema y también ofrece una interfaz gráfica intuitiva.
- Cosmos permite un análisis de Big Data más sencillo sobre el contexto integrado con las plataformas populares de Big Data (Spark y Flink).
- QuantumLeap admite el almacenamiento de datos de contexto en una base de datos de series temporales (CrateDB y Timescale)

2. Interface with IoT, Robots and Third-Party Systems

Hay una serie de habilitadores genéricos que facilitan la interfaz con el IoT, los robots y los sistemas de terceros, con el fin de recopilar valiosa información de contexto o desencadenar actuaciones en respuesta a las actualizaciones de contexto:

- IDAS ofrece una amplia gama de Agentes IoT que facilitan la interfaz con dispositivos que utilizan los protocolos IoT más comunes (LWM2M sobre CoaP, JSON o UltraLight sobre HTTP/MQTT, OPC-UA, Sigfox o LoRaWAN)
  - Agente IoT para JSON - un puente entre la mensajería HTTP/MQTT (con una carga útil JSON) y NGSI.
  - Agente IoT para LWM2M - un puente entre el protocolo Lightweight M2M y NGSI.
  - Agente IoT para Ultralight - un puente entre la mensajería HTTP/MQTT (con una carga útil UltraLight2.0) y NGSI.
  - Agente IoT para LoRaWAN - un puente entre el protocolo LoRaWAN y NGSI.
  - Agente IoT para OPC-UA - un puente entre el protocolo OPC Unified Architecture y NGSI.
  - Agente IoT para Sigfox - un puente entre el protocolo Sigfox y NGSI.
  - Agente IoT para ISOXML - un puente entre el protocolo ISOXML/ADAPT para maquinaria agrícola y NGSI.
- Kurento permite el procesamiento en tiempo real de datos multimedia adquiridos a partir de cámaras de vídeo u otros dispositivos de adquisición de imágenes, así como la incorporación de funciones de aplicaciones avanzadas (comunicaciones audiovisuales integradas, realidad aumentada, reproducción y grabación flexible de medios, etc.).

Otro Agente IoT en fase de desarrollo dentro del área de IoT, Robótica y sistemas de terceros es:

- FIROS, funciona como un traductor entre el dominio de la robótica y el cloud, transformando los mensajes de ROS en NGSI v2 y viceversa.

3. Context Processing, Analysis and Visualization

Existen varios GEs que facilitan el procesamiento, el análisis o la visualización de la información de contexto con el fin de implementar el comportamiento inteligente esperado en cualquier aplicación:

- Wirecloud aporta una potente plataforma web mashup que facilita el desarrollo de paneles de control operativos altamente personalizables por los usuarios finales.
- FogFlow es un marco de ejecución distribuido para soportar flujos de procesamiento dinámicos en la nube y en el propio hardware.
- Perseo introduce el Procesamiento de Eventos Complejos (CEP) definido mediante un sistema basado

en reglas, que permite disparar eventos que envían solicitudes HTTP, correos electrónicos, tweets, mensajes SMS, etc.

4. Context Data/API Management, Publication and Monetization

Se puede implementar el acceso seguro a los componentes en la arquitectura de cualquier solución "Powered by FIWARE" utilizando los habilitadores de esta categoría:

- Keyrock Identity Management aporta soporte para la autenticación segura y privada basada en OAuth2 de usuarios y dispositivos, gestión de perfiles de usuario, disposición de datos personales preservando la privacidad, Single Sign-On (SSO) y Federación de Identidades a través de múltiples dominios de administración.
- Wilma PEP Proxy aporta soporte a las funciones de proxy dentro de los esquemas de autenticación basados en OAuth2. También implementa las funciones PEP dentro de un esquema de control de acceso basado en XACML.
- AuthZForce PDP/PAP aporta soporte a las funciones PDP/PAP dentro de un esquema de control de acceso basado en el estándar XACML.

Esta área también contiene un capítulo para la publicación y monetización de los recursos de datos de contexto, disponibles a través del componente central OCB:

- CKAN Extensions proporciona una serie de complementos que permiten ampliar las capacidades actuales de la plataforma de publicación de datos abiertos CKAN, para facilitar la publicación de conjuntos de datos que coincidan con los datos contextuales en tiempo real, la asignación de condiciones y políticas de acceso a esos conjuntos de datos y la asignación de esquemas de precios y pago por uso a los conjuntos de datos.
- Biz Framework aporta soporte de backend a la monetización de la API de contexto/datos basada en las API comerciales abiertas de TM Forum.
- Idra es capaz de federar sistemas de gestión de datos abiertos existentes basados en tecnologías heterogéneas (por ejemplo, CKAN, SOCRATA, DKAN, etc.) proporcionando una única API y un formato de metadatos estándar (DCAT-AP) para descubrir conjuntos de datos abiertos.

5. Deployment Tools

Los componentes de FIWARE se pueden implementar utilizando técnicas de “contenedorización” estándar. La mayoría de los componentes están disponibles directamente como imágenes de Docker[11], además, se puede usar docker-compose para el despliegue de múltiples contenedores. Estos archivos se usan en entornos de desarrollo, mientras que para aplicaciones orientadas a producción totalmente escalables es necesario seguir las indicaciones de FIWARE Helm Chart.

5.1.2.1 Orion Context Broker

Como se ha comentado. el Orion Context Broker Generic Enabler es el componente central y el único obligatorio para desarrollar cualquier solución "Powered by FIWARE". Este permite gestionar la información de contexto de forma altamente descentralizada y a gran escala. Proporciona la API FIWARE NGSIv2, que es una sencilla pero potente API RESTful que ayuda a realizar actualizaciones, consultas o suscribirse a cambios en la información de contexto [42]. La Figura 5-4 muestra los diferentes tipos de operaciones de gestión de la información de contexto que puede realizar el OCB.

El OCB contiene información sobre el contexto actual, pero solo mantiene el último valor de las entidades y atributos. Para almacenar y obtener información de contexto histórica, es necesario el uso de otros GEs.

[11] https://www.docker.com/

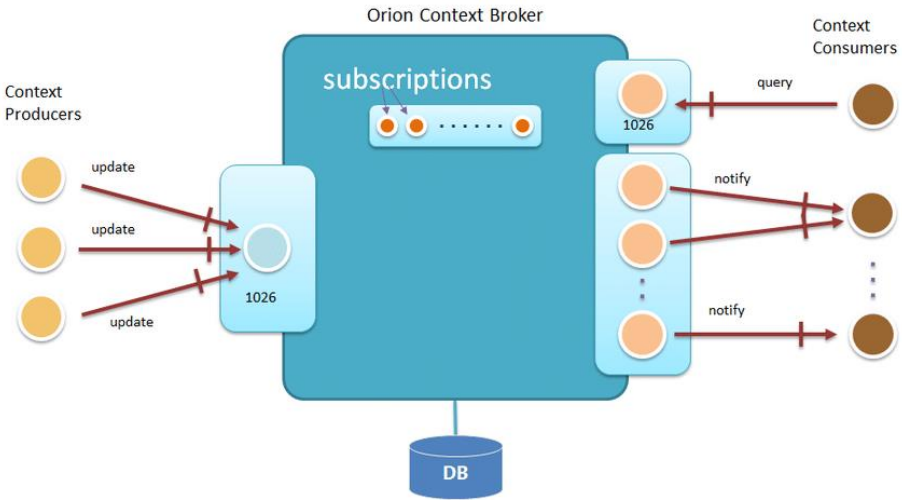


Figura 5-4. Operaciones de gestión de la información de contexto que realiza el Orion Context Broker.[43]

5.1.2.2 FIROS

Como se ha mencionado en esta sección, uno de los grandes retos del IoT es superar la heterogeneidad de los dispositivos que hablan diferentes protocolos. En lugar de intentar resolver esta batalla de protocolos a nivel de dispositivo, FIWARE ofrece soluciones que permiten la coexistencia de estándares.

Un Agente IoT es un componente que permite a un grupo de dispositivos enviar sus datos y ser administrados desde un Context Broker utilizando sus propios protocolos nativos, es decir, un Agente IoT traduce un protocolo específico de IoT al modelo de información de contexto NGSI o viceversa. Como el OCB utiliza exclusivamente solicitudes NGSI-v2 para todas sus interacciones, cada Agente IoT proporciona una interfaz North Port NGSI-v2 que se utiliza de intermediario de contexto y todas las interacciones debajo de este puerto se producen mediante el protocolo nativo de los dispositivos conectados.

En efecto, esto brinda una interfaz estándar para todas las interacciones de IoT en el nivel de administración de información de contexto. Cada grupo de dispositivos puede usar sus propios protocolos patentados y mecanismos de transporte, mientras que el Agente de IoT asociado ofrece un mecanismo para manejar esta complejidad.

FIROS[12] es un ejemplo de Agente IoT que representa el enlace entre ROS y FIWARE. Concretamente, se trata de un nodo ROS que se comunica con el Context Broker para publicar y escuchar los datos del robot. En otras palabras y como se representa en la Figura 5-5, FIROS funciona como un traductor entre el campo de la robótica y el cloud, transformando los mensajes provenientes de los topics del entorno de ROS en el formato NGSI para publicarlos en la nube, y viceversa [44].

[12] https://firos.readthedocs.io/en/latest/index.html

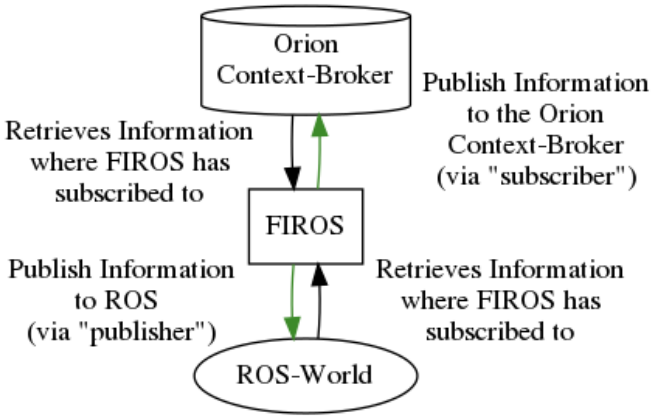


Figura 5-5. Representación del flujo de datos en el Agente IoT FIROS.

Para arrancar FIROS es necesario completar los siguientes archivos config.json y topics.json de configuración:

- config.json

Este archivo contiene la configuración relacionada con el entorno de lanzamiento de FIROS. A continuación, se listan los parámetros que pueden ser manipulados por el usuario:

- "environment": a través de este atributo se puede seleccionar una configuración de entorno específica.
- "server": contiene información relacionada con el servidor de FIROS e incluye el atributo "port", puerto en el que FIROS está escuchando, por defecto es 10100. Se puede cambiar el puerto si se experimentan errores. Esto suele ocurrir cuando este puerto ya está ocupado por otra aplicación.
- "contextbroker": comprende información relacionada con la configuración del OCB y contiene los atributos "address" y "port", dirección IP y puerto del Context Broker donde FIROS debe publicar los datos, y el atributo "subscriptions" que a su vez puede contener lo siguiente:
    - "throttling": frecuencia de actualización a la que el Context Broker envía las actualizaciones al robot. El valor predeterminado es 0.
    - "subscription_length": tiempo de expiración de la suscripción.
    - "subscription_refresh_delay": frecuencia de actualización de la suscripción para evitar su caducidad.
- "log_level": las opciones disponibles son "INFO" (por defecto), "DEBUG", "WARNING", "ERROR" o "CRITICAL".
- "node_name": nombre del nodo de ROS de la instancia de FIROS. El valor por defecto es "firos".
- "ros_subscriber_queue": tamaño de la cola del publicador. Por defecto es 10.
- "rosbridge_port": puerto ROS donde escuchar. Por defecto es 9090.
- "pub_frequency": frecuencia de publicación en milisegundos.

- topics.json

```
{
    "/turtle1/cmd_vel": ["geometry_msgs/Twist", "publisher"],
    "/turtle1/pose": ["turtlesim/Pose", "subscriber"]
}
```

Este archivo de configuración describe la información que la instancia de FIROS debe publicar en el entorno FIWARE y en el entorno de ROS.

Este JSON en particular escucha el rostopic /turtle1/pose con el tipo de mensaje "turtlesim/Pose" y envía todos

los datos recuperados al Context Broker. Por otro lado, publica los datos en /turtle1/cmd_vel después de recibir una notificación del Context Broker del tipo geometry_msgs/Twist.

5.1.2.3 QuantumLeap

QuantumLeap es un habilitador alternativo creado específicamente que ofrece una API REST para almacenar, consultar y recuperar datos espaciotemporales de NGSI v2 y NGSI-LD (soporte experimental) sobre bases de datos de series de tiempo (CrateDB y TimescaleDB). QuantumLeap ofrece una alternativa al STH-Comet con las siguientes diferencias:

Tabla 5–1. Comparación de herramientas para la persistencia de datos en FIWARE.

QuantumLeap	STH-Comet
Interfaz NGSI v2 para notificaciones	Interfaz NGSI v1 para notificaciones
Conserva los datos en la base de datos CrateDB o TimescaleDB	Persiste los datos en la base de datos MongoDB
Ofrece su propio punto final HTTP para consultas (actualmente para CrateDB), pero también puede consultar CrateDB y TimescaleDB	Ofrece su propio punto final HTTP para consultas, también puede consultar MongoDB directamente
Admite consultas de datos complejas	Conjunto limitado de consultas
Aprovecha dos DBMS SQL nativos distribuidos y escalables	MongoDB es una base de datos NoSQL basada en documentos

Aunque QuantumLeap y STH-Comet comparten objetivos similares, Comet no soporta múltiples bases de datos (sólo MongoDB) y tampoco soporta NGSI v2. Aunque Comet es un buen software en sí mismo, algunas de las necesidades y suposiciones que impulsaron su desarrollo ya no son actuales. QuantumLeap comenzó como una exploración de una forma alternativa de hacer que los datos históricos estén disponibles para el ecosistema FIWARE sin comprometerse con una base de datos específica.

La especificación de la API REST, denominada NGSI-TSDB, proporciona un mecanismo uniforme y familiar (para los desarrolladores de FIWARE) para acceder a los datos de las series temporales que permite implementar servicios como QuantumLeap para soportar de forma transparente múltiples backends de bases de datos.

QuantumLeap convierte los datos semiestructurados de NGSI en formato tabular y los almacena en una base de datos de series temporales, asociando cada registro de la base de datos con un índice temporal y, si está presente en los datos de NGSI, con una ubicación geográfica. Los clientes REST pueden entonces recuperar las entidades del NGSI filtrando los conjuntos de entidades mediante rangos de tiempo y operadores espaciales. Desde el punto de vista del cliente, estas consultas se definen sobre las entidades del NGSI y no sobre las tablas de la base de datos. Sin embargo, la funcionalidad de consulta disponible a través de la interfaz REST es bastante básica y las consultas más complejas suelen requerir que los clientes utilicen la base de datos directamente.

Normalmente QuantumLeap adquiere los datos IoT en forma de entidades NGSI, a través de las notificaciones NGSI establecidas por adelantado con el Context Broker. Por lo tanto, para que este GE reciba datos, el cliente debe crear una suscripción en el OCB especificando qué entidades deben ser notificadas cuando se produce un cambio. Como se ha mencionado anteriormente, las entidades NGSI entrantes se convierten en registros de base de datos y se almacenan en uno de los backends de la base de datos de series temporales configurados. A partir de este momento, cuando los agentes de la capa IoT envían datos al Broker, si los datos pertenecen a las entidades señaladas por la suscripción del cliente, Orion reenvía los datos a QuantumLeap mediante el envío de entidades NGSI al punto final de notificación.

Un principio rector en el diseño de QuantumLeap ha sido la capacidad de utilizar múltiples bases de datos de

series temporales. Esta elección de diseño se justifica por el hecho de que un producto de base de datos puede ser más adecuado que otro dependiendo de las circunstancias. Actualmente, QuantumLeap puede utilizarse tanto con CrateDB como con Timescale.

- CrateDB es el backend por defecto. Es fácil de escalar gracias a su arquitectura "shared-nothing" que se presta bien a la “contenedorización”, por lo que es relativamente fácil gestionar un clúster de base de datos CrateDB en contenedor, por ejemplo, utilizando Kubernetes. Es capaz de ingerir un gran número de datos por segundo y pueden consultarse en tiempo real mediante SQL, con extensiones incorporadas para consultas temporales y geográficas. A continuación, se muestran algunos ejemplos de sentencias para consultar la base de datos, y la visualización en la propia interfaz de CrateDB (Figura 5-6).

```
//Listar todas las tablas
SHOW TABLES

//Consultar una tabla
SHOW CREATE TABLE "doc"."etgeometry_msgs%2fposewithcovariancestamped";

//Mostrar el atributo x de las 100 primeras filas de la tabla
SELECT
  pose['pose']['value']['position']['value']['x']['value']
FROM "doc"."etgeometry_msgs%2fposewithcovariancestamped" limit 100;
```

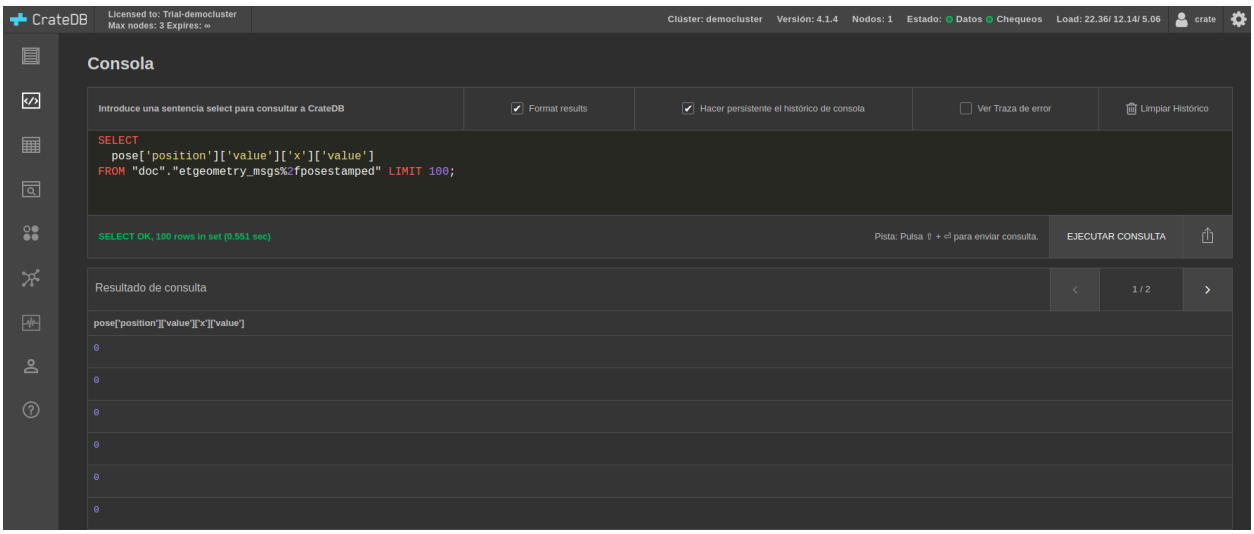


Figura 5-6. Ejemplo de histórico de datos mostrado a través de peticiones a la base de datos CrateDB.

CrateDB ofrece también una API de Postgres, que simplifica su integración. Por ejemplo, se puede aprovechar el plugin PostgreSQL de Grafana para visualizar las series temporales almacenadas, donde se realizan consultas a la base de datos como las que se han mostrado anteriormente.

- TimescaleDB escala PostgreSQL para datos de series temporales a través de la partición automática en el tiempo y el espacio, pero conserva la interfaz estándar de PostgreSQL. En otras palabras, TimescaleDB expone lo que parecen tablas regulares, pero en realidad son sólo una abstracción (o una vista virtual) de muchas tablas individuales que comprenden los datos reales. En combinación con la extensión PostGIS puede soportar geo-series de tiempo.

5.1.2.4 Grafana

El habilitador genérico Grafana es una potente herramienta de visualización que se puede utilizar para mostrar gráficos y paneles de control y monitorizar los datos persistentes [45]:

- Visualización: Gráficos rápidos y flexibles del lado del cliente con multitud de opciones. Los plugins ofrecen muchas formas diferentes de visualizar las métricas y los registros.
- Paneles dinámicos: paneles de control dinámicos y reutilizables con variables de plantilla que aparecen

como desplegables en la parte superior del panel.

- Métricas: explorar datos a través de consultas ad-hoc y desglose dinámico. Se puede dividir la vista y comparar diferentes rangos de tiempo, consultas y fuentes de datos.
- Registros: registros con filtros de etiquetas.
- Alertas: reglas de alerta para las métricas más importantes. Grafana evaluará continuamente y enviará notificaciones a sistemas como Slack, PagerDuty, VictorOps, OpsGenie.
- Mezcla de fuentes de datos: mezclar diferentes fuentes de datos en el mismo gráfico. Se puede especificar una fuente de datos por consulta. Esto funciona incluso con fuentes de datos personalizadas.

Se puede conectar fácilmente al backend de QuantumLeap, CrateDB o Timescale. En ambos casos, la fuente de datos a utilizar es Postgres Datasource que normalmente viene con las versiones recientes de Grafana.

Para añadir esta fuente de datos hay que tener en cuenta los siguientes campos requeridos:

- Nombre: es el nombre que se otorga al Datasource, CrateDB.
- Host: la url completa donde se desplegó CrateDB, pero en el puerto 5432.
- Base de datos: este es el nombre del esquema de la base de datos. Por defecto, crate utiliza el esquema "doc" de la base de datos, pero si se está utilizando cabeceras multi-tenancy, el esquema será definido por el tenant del tipo de entidad.
- Usuario: se utiliza el usuario de crate.
- Modo SSL: deshabilitar.

La siguiente Figura 5-7 muestra un ejemplo de cómo debería ser la configuración de la fuente de datos.

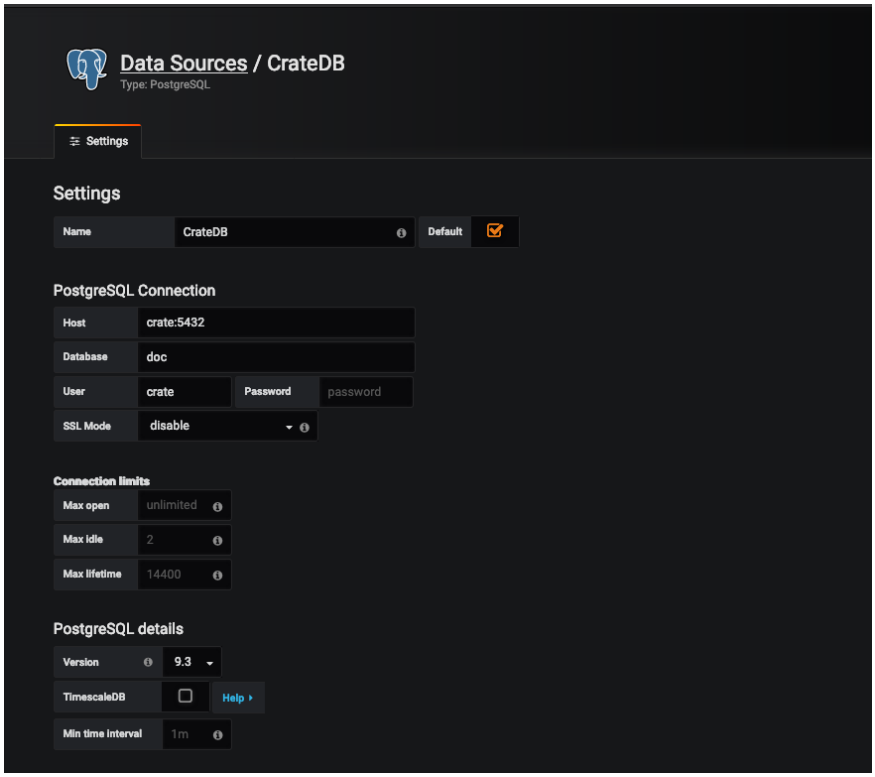


Figura 5-7. Ejemplo de formulario para añadir un nuevo DataSource conectado a la base de datos CrateDB.

Una vez configurada la fuente de datos, ya se pueden empezar a generar los paneles de control a través de los diferentes widgets. Cada una de las visualizaciones, de forma interna, realizará las consultas a la base de datos seleccionada. En las siguientes Figuras 5-8, 5-9 y 5-10, se muestra la configuración de los paneles de control utilizados en este trabajo para la visualización y seguimiento de las variables clave del proceso. Para los tres

casos, se define, por un lado, las consultas a la fuente de datos conectada a CrateDB, y, por otro, se define el tipo de gráfico y las opciones de visualización.

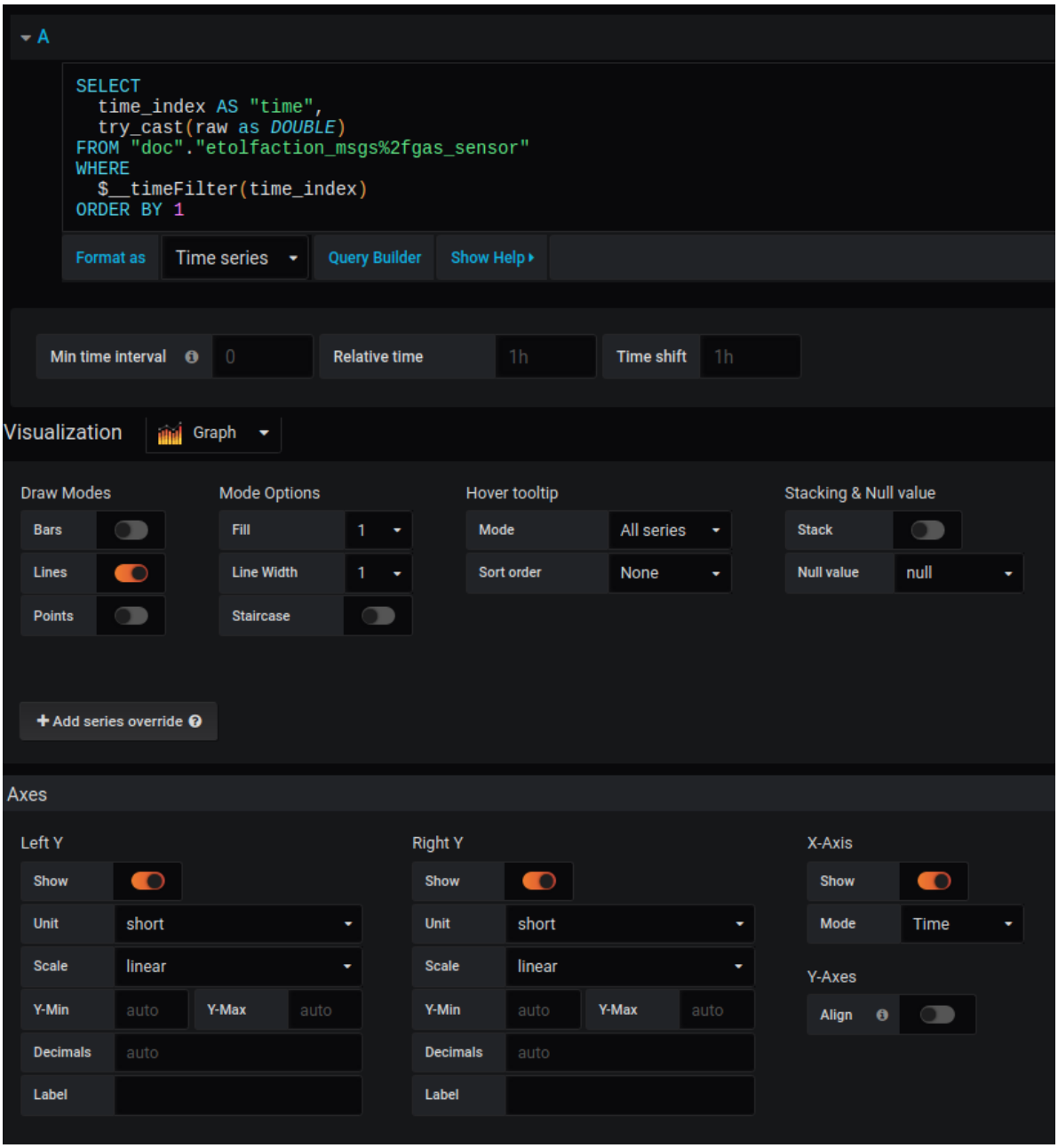


Figura 5-8. Configuración de un Gráfico 2D para las medidas del sensor de concentración de gas a lo largo del tiempo.

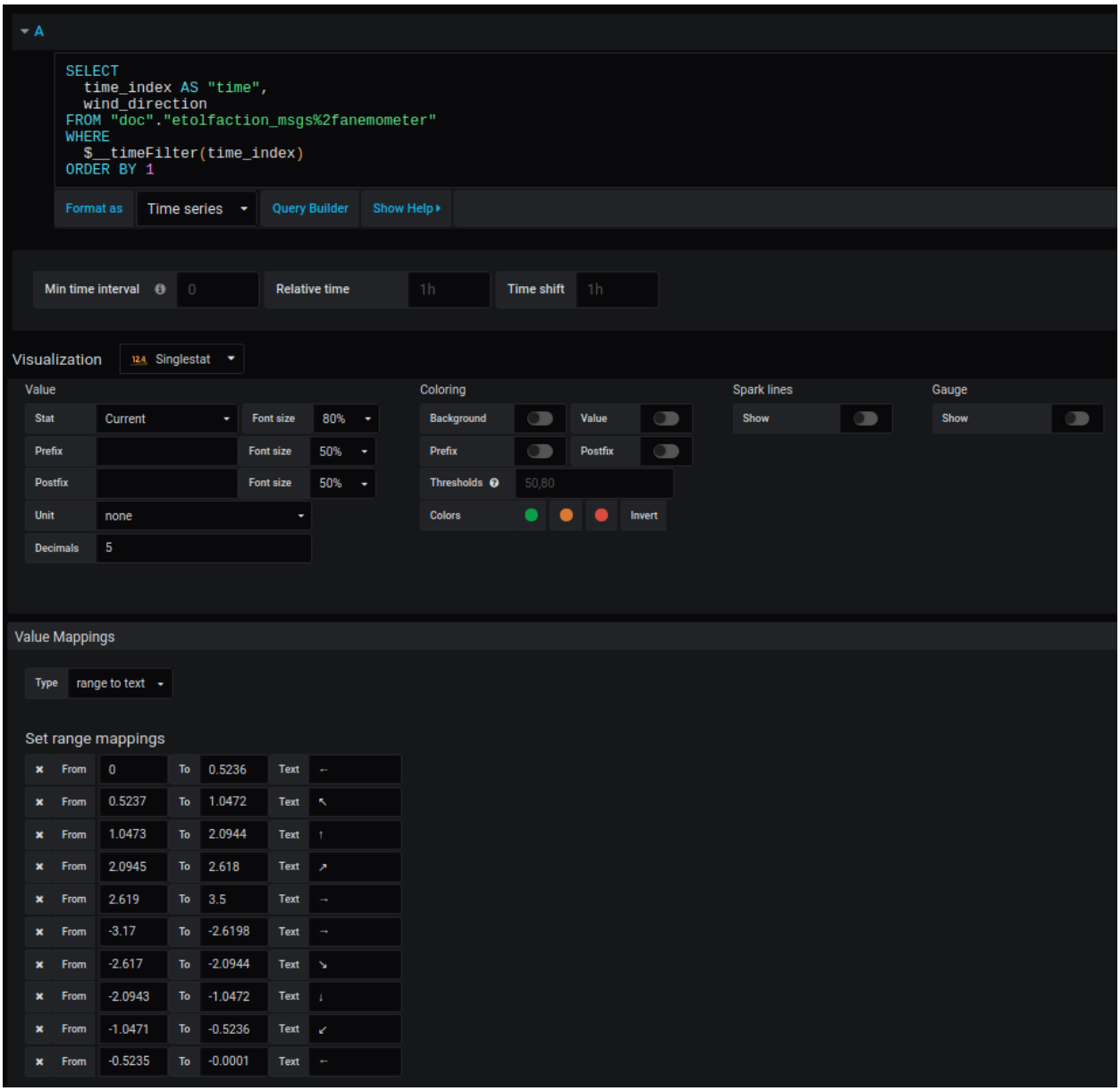


Figura 5-9. Configuración del panel de control para las medidas del anemómetro de velocidad y dirección del viento en la planta.

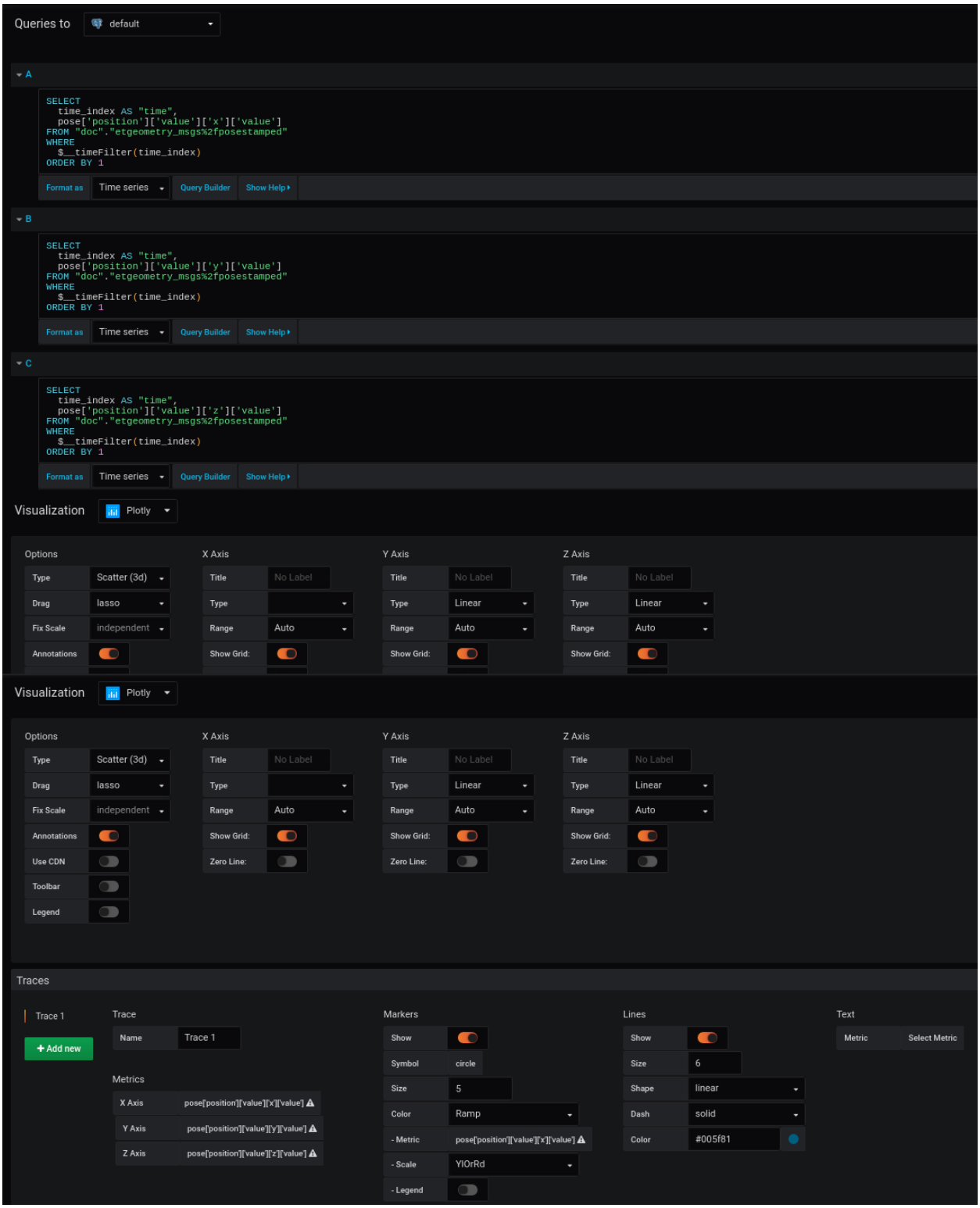


Figura 5-10. Configuración del gráfico 3D para el seguimiento de la posición del UAV.

5.2 Implementación de la plataforma

Partiendo de los GEs detallados en los apartados anteriores, se realiza en este trabajo el despliegue de una plataforma basada en FIWARE que permite el seguimiento en tiempo real y la monitorización de la información de los sistemas desarrollados en las secciones previas: se tiene, por un lado, el sensor de concentración de gas (acoplado al UAV), el anemómetro, situado en una posición concreta de la planta, y el robot UAV, cuya posición es conocida, simulando el comportamiento de un sensor GPS. Bajo estas consideraciones, la arquitectura general constará de los siguientes elementos:

- Los habilitadores genéricos de FIWARE:
    - El Orion Context Broker que recibirá solicitudes utilizando NGSI-v2.
    - El IoT Agent FIROS, que recibirá la información de los dispositivos (sensor de gas, anemómetro y GPS) y las convertirá en solicitudes NGSI-v2 para que el intermediario de contexto altere el estado de las entidades.
    - QuantumLeap, suscrito a los cambios de contexto, mantiene un histórico de los datos en una base de datos CrateDB. El puerto 8668 es donde el servicio estará escuchando las notificaciones del Context Broker y desde donde los usuarios pueden consultar los datos.
    - Grafana, para consultar y recuperar los datos de la base de datos CrateDB para proveer visualizaciones y proporcionar métricas. El contenedor se conecta al puerto 3000 internamente y al puerto 3003 externamente.
- Una base de datos de MongoDB:
    - Utilizado por Orion Context Broker para contener información de datos de contexto, como entidades de datos, suscripciones y registros.
- Una base de datos CrateDB:
    - Se utiliza como un sumidero de datos para almacenar datos de contexto históricos basados en el tiempo.
    - Ofrece un punto final HTTP para interpretar consultas de datos basadas en el tiempo.
    - El contenedor CrateDB escucha en dos puertos: la interfaz de administración está disponible en el puerto 4200 y el protocolo de transporte está disponible en el puerto 4300.

A continuación, en la Figura 5-11, se puede observar el esquema de la arquitectura descrita previamente, así como los puertos utilizados para la comunicación entre los diferentes componentes.

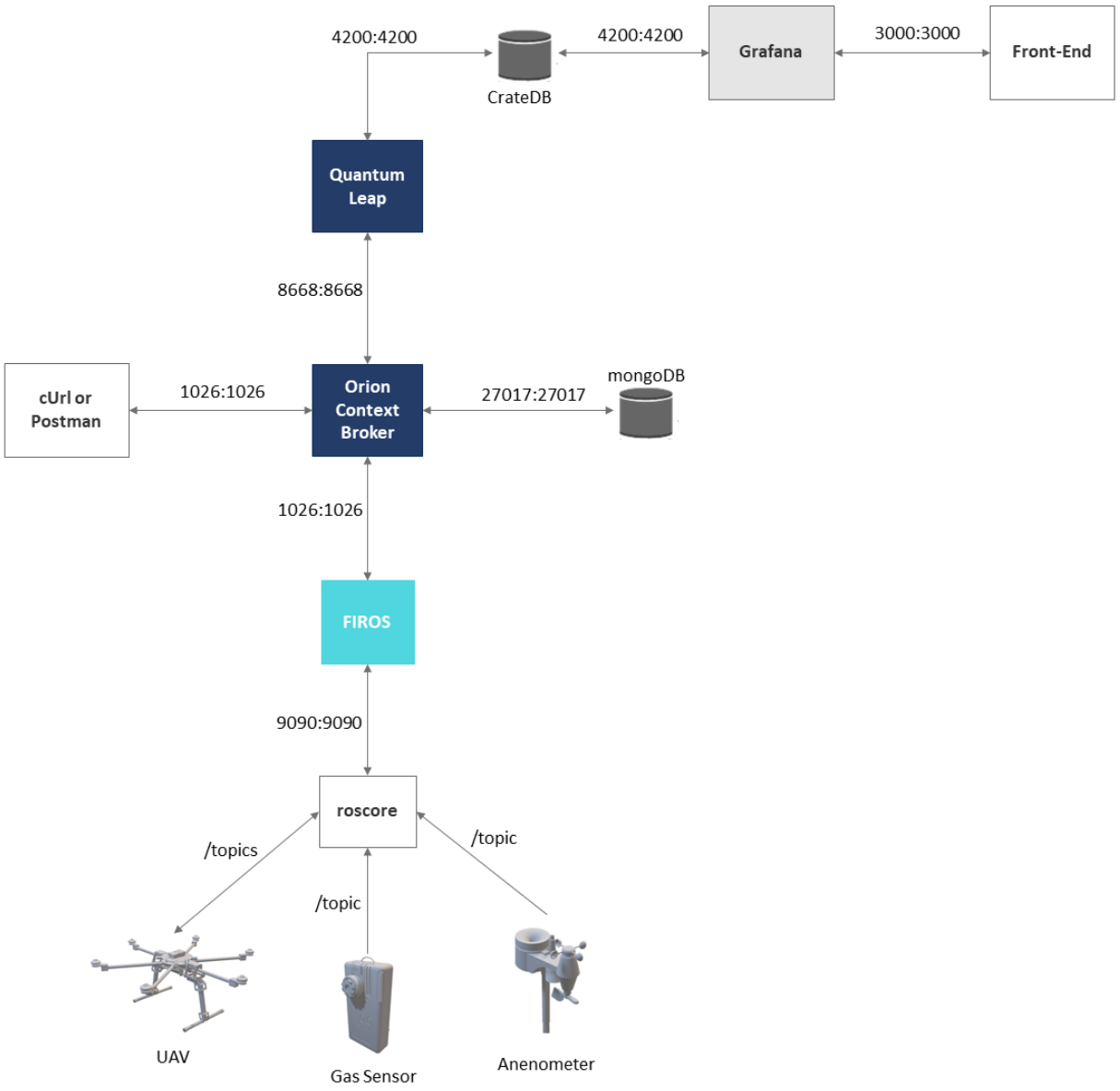
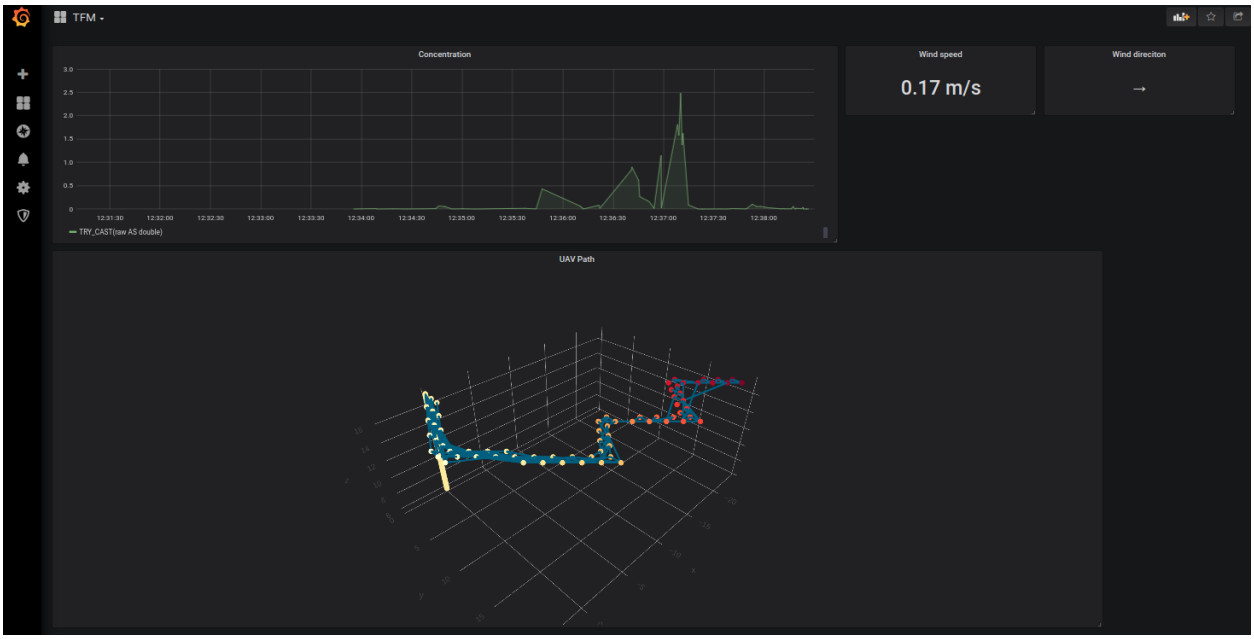
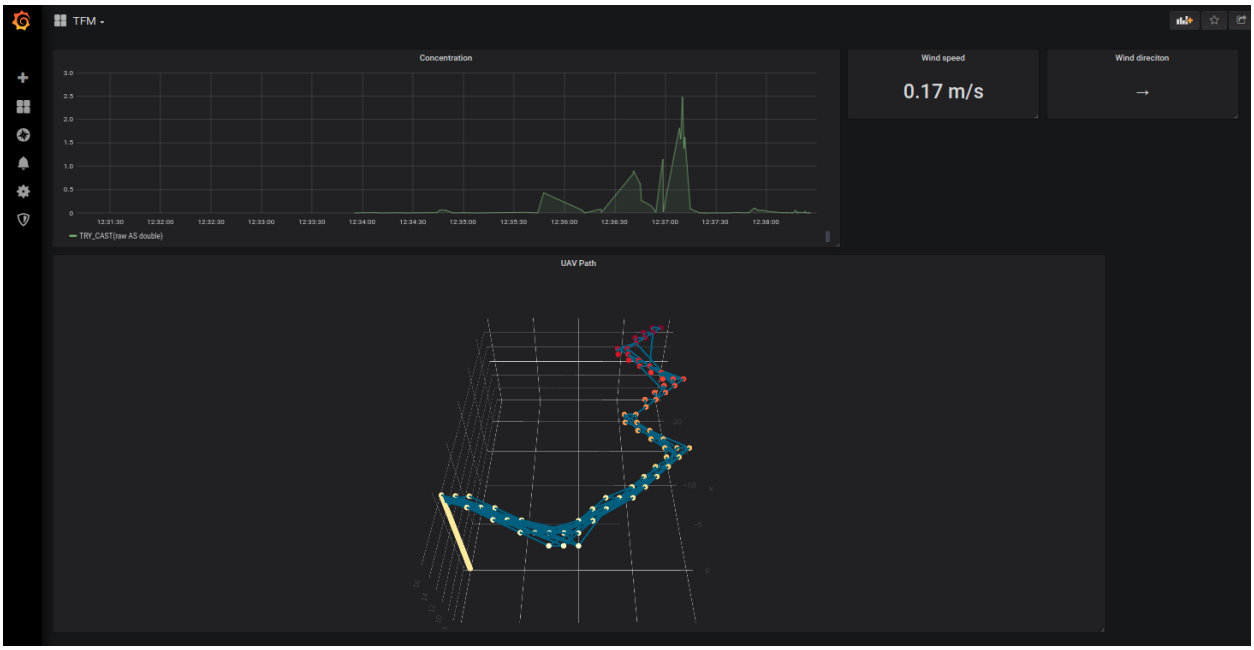


Figura 5-11. Arquitectura de la plataforma FIWARE desplegada donde se puede comprobar la conexión entre los diferentes componentes.

Finalmente se muestra la interfaz gráfica de Grafana con todos los paneles de control (Figura 5-12).



(a)



(b)

Figura 5-12. Visualización de los paneles de control desplegados en la plataforma final.

El objetivo de estas herramientas de visualización es obtener datos del proceso para poder tomar decisiones a partir de información más detallada y con un alto valor añadido. En el ejemplo mostrado (Figura 5-12), se puede observar que el UAV ha realizado un movimiento basado en el algoritmo de localización zigzag, y, además, en la gráfica de medida de concentración de gas, se aprecia el cambio en la pendiente de este valor para las diferentes oleadas, así como también, que el valor máximo para cada una de ellas va aumentando a medida que el robot se aproxima a la fuente de emisión de olor. El detalle de estas deducciones se puede ver en la Figura 5-13.

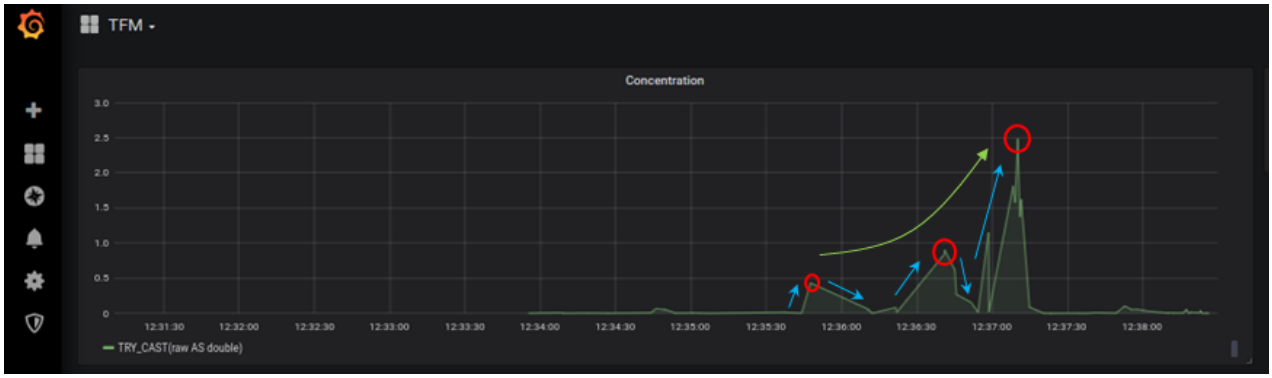


Figura 5-13. Conclusiones obtenidas a partir de la visualización de los datos.

6 CONCLUSIONES. TRABAJOS FUTUROS

El uso de robótica móvil en el proceso de detección y localización de fugas de gas presenta evidentes ventajas:

- Se elimina el riesgo toxicológico asociado a las personas encargadas de realizar la inspección.
- Se reduce la emisión de gases de efecto invernadero generados en diferentes procesos de la planta.
- Se optimizan los recursos al reducir los costes asociados a la ejecución del mantenimiento correctivo.

Además, las capacidades computacionales de los sistemas MRO pueden utilizarse para crear mapas de distribuciones de los gases, planificar recorridos de vigilancia de cobertura total, discriminar entre diferentes sustancias volátiles o inferir la localización de fugas de gas.

Para este proceso concretamente, el gemelo digital ha desempeñado un papel fundamental pues, desde el punto de vista de la simulación, ha permitido el diseño de diferentes estrategias de búsqueda y así como evaluar su desempeño para resolver el problema.

Desde el punto de vista del ecosistema IoT, la plataforma de código abierto FIWARE facilita enormemente la tarea de despliegue de los diferentes componentes que conforman la plataforma, así como la versatilidad que ofrece a la hora de poder utilizar dispositivos de diferentes proveedores y que en principio presentarían problemas de incompatibilidad (diferentes protocolos, etc.). Al mismo tiempo, permite tener la información recogida por todos los sensores centralizada y accesible desde cualquier punto, posibilitando el seguimiento y representación de los datos en tiempo real que aportan un valor añadido en la toma de decisiones estratégicas de la empresa. Además de poder representar los datos, también se pueden enviar órdenes a los dispositivos/actuadores. En este trabajo, por ejemplo, es posible ordenar al UAV que se mueva a una posición concreta, y todo ello a través de la API que proporciona FIWARE.

En cuanto a los desarrollos posteriores al resultado de este trabajo se contempla la ejecución de pruebas adicionales con otros tipos de algoritmos de localización de fugas de gas, con el fin de obtener una comparación más extensa y precisa de las ventajas de usar los diferentes tipos de algoritmos.

Para este mismo propósito resulta útil introducir mejoras en el banco de ensayos diseñado, de tal forma que se pueda utilizar con la simulación dinámica de la fuga de gas y no solo en un estado estacionario. De esta manera se puede observar cuál sería el comportamiento de los algoritmos frente a un entorno más realista.

En un nivel mayor de complejidad, y para aumentar la verosimilitud de los resultados teóricos con respecto a los resultados experimentales, el simulador de fugas de gas debería tener en cuenta las corrientes de viento generadas por el movimiento del propio robot y su influencia sobre el penacho. Estas consideraciones, aunque suponen un aumento exponencial de la capacidad de computación requerida, pues se deben ajustar los patrones de flujo de viento en tiempo real teniendo en cuenta la posición del robot, mostrarían la capacidad de las heurísticas subyacentes a los algoritmos para hacer frente a situaciones todavía más dinámicas, y si realmente son capaces de alcanzar el objetivo bajo esta coyuntura.

Finalmente, también resulta conveniente la realización de pruebas experimentales con sistemas reales (sensor de concentración de gas, anemómetro, UAV), con el objetivo de verificar la exactitud de los resultados obtenidos a partir de la simulación. Además, se puede estudiar la influencia que tendrían las corrientes generadas por los motores del UAV sobre el sensor de medida de concentración de gas acoplado a bordo del robot (ver Figura 6-1).

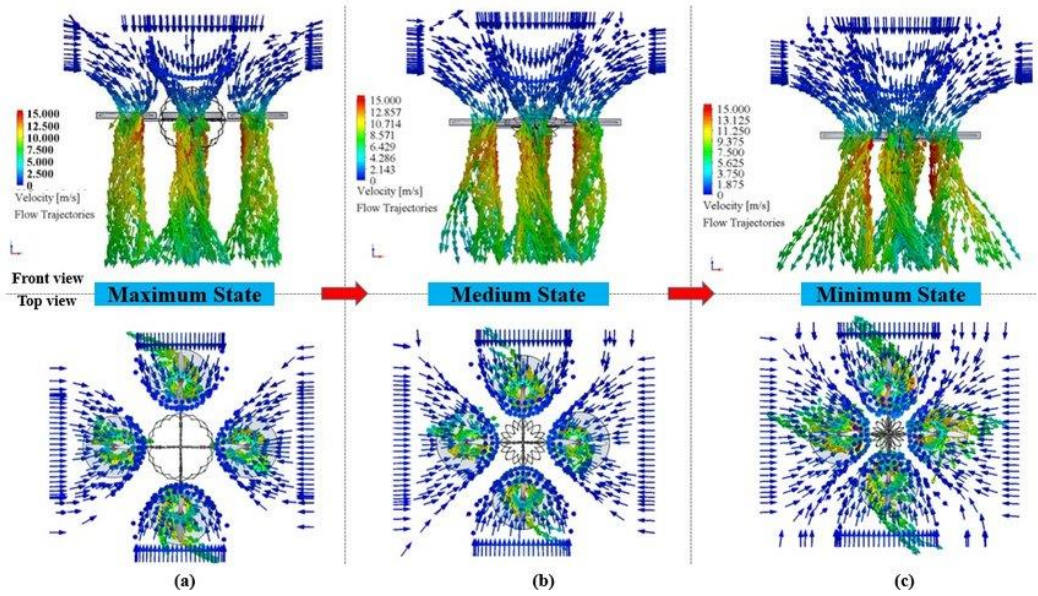


Figura 6-1. Simulación CFD de las corrientes generadas por un quadrotor en diferentes estados [46].

REFERENCIAS

[1] IPCC (1996). Revised 1996 IPCC Guidelines for National Greenhouse Gas Inventories.

[2] Laconde T. (2018). Fugitive emissions: a blind spot in the fight against climate change. Global Observatory of Non-State Climate Action.

[3] Zarra, T.; Giuliani, S.; Naddeo, V.; Belgiorno, V. (2012). Control of odour emission in wastewater treatment plants by direct and undirected measurement of odour emission capacity. Water Science & Technology, 66(8), 1627.

[4] Legator, Marvin S.; Singleton, Chantele R.; Morris, Debra L.; Philips, Donna L. (2001). Health Effects from Chronic Low-Level Exposure to Hydrogen Sulfide. Archives of Environmental Health: An International Journal, 56(2), 123–131.

[5] Loutfi, Amy & Coradeschi, Silvia & Karlsson, Lars & Broxvall, M. (2004). Putting olfaction into action: using an electronic nose on a multi-sensing mobile robot. 1. 337 - 342 vol.1.

[6] Monroy, Javier & González-Jiménez, Javier. (2018). Towards Odor-Sensitive Mobile Robots. Electronic Nose Technologies and Advances in Machine Olfaction. IGI Global. pp. 244—263.

[7] G. Cabrita, P. Sousa, and L. Marques. (2010). PlumeSim - Player/Stage Plume Simulator. ICRA Workshop on Networked and Mobile Robot Olfaction in Natural, Dynamic Environments.

[8] Cabrita, Gonçalo & Sousa, Pedro & Marques, Lino. (2010). Player/Stage simulation of olfactory experiments. IEEE/RSJ 2010 International Conference on Intelligent Robots and Systems, IROS 2010 - Conference Proceedings. 1120 - 1125.

[9] Monroy, Javier & Blanco, Jose Luis & González-Jiménez, Javier. (2013). An Open Source Framework for Simulating Mobile Robotics Olfaction.

[10] OpenMora Repository. Recuperado 29 de noviembre de 2021, de http://openmora.github.io

[11] Khaliq, A.A., Pashami, S., Schaffernicht, E.J., Lilienthal, A.J., & Bennetts, V.M. (2015). Bringing Artificial Olfaction and Mobile Robotics Closer Together: An Integrated 3D Gas Dispersion Simulator in ROS.

[12] Heudin, Jean-Claude (1998). Webots: Symbiosis Between Virtual and Real Mobile Robots. Lecture Notes in Computer Science. Virtual Worlds Volume 1434. Chapter 24, 254–263.

[13] Jay A. Farrell; John Murlis; Xuezhu Long; Wei Li; Ring T. Cardé (2002). Filament-Based Atmospheric Dispersion Model to Achieve Short Time-Scale Structure of Odor Plumes. 2(1-2), 143–169.

[14] Wikibooks contributors. Webots Odor Simulation — Wikibooks, The Free Textbook Project, 2010. Recuperado 29 de noviembre de 2021, de https://en.wikibooks.org/w/index.php?title=Webots_Odor_Simulation&stableid= 3470318

[15] Invernizzi, Marzio & Capra, Federica & Sozzi, Roberto & Capelli, Laura & Sironi, Selena. (2021). Development and Evaluation of a Fluctuating Plume Model for Odor Impact Assessment. Applied Sciences. 11. 3310.

[16] Murlis, J., Elkinton, J.S., Cardé, R.T., 1992. Odor Plumes and How Insects Use Them. Annu. Rev. Entomol. 37, 505–532.

[17] F. Rahbar, A. Marjovi and A. Martinoli. (2019). An Algorithm for Odor Source Localization based on Source Term Estimation. International Conference on Robotics and Automation (ICRA), pp. 973-979.

[18] Chen, Xin-xing; Huang, Jian (2019). Odor source localization algorithms on mobile robots: A review and future outlook. Robotics and Autonomous Systems, 112, 123–136.

[19] R.A. Russell, D. Thiel, R. Deveza, A. Mackay-Sim. (1995). A robotic system to locate hazardous chemical leaks, in: Robotics and Automation. Proceedings., IEEE International Conference on, Vol. 1, IEEE, 1995, pp. 556–561.

[20] Reeder, P., & Ache, B.W. (1980). Chemotaxis in the Florida spiny lobster, Panulirus argus. Animal Behaviour, 28, 831-839.

[21] Jatmiko, Wisnu & Sekiyama, Kosuke & Fukuda, Toshio. (2007). A PSO-based mobile robot for odor source localization in dynamic advection-diffusion with obstacles environment: Theory, simulation and measurement. Computational Intelligence Magazine, IEEE. 2. 37 - 51.

[22] Vergassola, Massimo & Villermaux, Emmanuel & Shraiman, Boris. (2007). 'Infotaxis' as a strategy for searching without gradients. Nature. 445. 406-9.

[23] Blanco, Jose Luis & Monroy, Javier & González-Jiménez, Javier & Lilienthal, Achim. (2013). A Kalman Filter Based Approach to Probabilistic Gas Distribution Mapping. Proceedings of the ACM Symposium on Applied Computing.

[24] Basco, A. I., Beliz, G., Coatz, D., & Garnero, P. (2018). Industria 4.0: fabricando el futuro (Vol. 647). Inter-American Development Bank.

[25] Havle, Celal & Ucler, Caglar. (2018). Enablers for Industry 4.0. 2018 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies (ISMSIT). pp. 1-6.

[26] Tao, Fei & Zhang, He & Liu, Ang & Nee, Andrew. (2019). Digital Twin in Industry: State-of-the-Art. IEEE Transactions on Industrial Informatics. 15. 2405-2415.

[27] Wang, Y. (2020). Enhancing interoperability for IoT based smart manufacturing: An analytical study of interoperability issues and case study (Dissertation).

[28] Vanessa, E. Boom in Digitalization & Automation in the Post COVID era. Digital Producer Magazine. Recuperado 29 de noviembre de 2021, de https://www.digitalproducer.com/boom-in-digitalization-automation-in-the-post-covid-era/

[29] Guth, J., Breitenbücher, U., Falkenthal, M., Fremantle, P., Kopp, O., Leymann, F., & Reinfurt, L. (2018). A Detailed Analysis of IoT Platform Architectures: Concepts, Similarities, and Differences. Internet of Everything.

[30] Technical University of Munich. Webinos. Recuperado 29 de noviembre de 2021, de https://www.in.tum.de/en/os/projects/webinos/

[31] Azure IoT Device connection. (s. f.-b). Thingspro Programming Guide. Recuperado 29 de noviembre de 2021, de https://thingspro-programming-guide.netlify.app/application-note/how_to_get_azure_iot_device_connection_string/

[32] Monroy, Javier & Hernandez-Bennets, Victor & Fan, Han & Lilienthal, Achim & González-Jiménez, Javier. (2017). GADEN: A 3D Gas Dispersion Simulator for Mobile Robot Olfaction in Realistic Environments. MDPI Sensors. Vol. 17, no. 7: 1479, pp. 1-16.

[33] Monroy, Javier & Hernandez-Bennets, Victor & Fan, Han & Lilienthal, Achim & González-Jiménez, Javier. (s.f.) GADEN Tutorial: How to set up your own simulations. From CAD model to gas dispersion. GitHub Repository. Recuperado 29 de noviembre de 2021, de https://github.com/MAPIRlab/gaden/blob/master/GADEN_tutorial.pdf

[34] Real, Fran & Torres-González, A. & Ramon, Pablo & Capitán, Jesús & Ollero, Anibal. (2020). Unmanned aerial vehicle abstraction layer: An abstraction layer to operate unmanned aerial vehicles. International Journal of Advanced Robotic Systems. 17.

[35] Castillejo Calle, A. (2019). Diseño e implementación de un sistema para la gestión de una flota de drones para la inspección de plantas fotovoltaicas. Trabajo Fin de Máster. Universidad de Sevilla.

[36] Kowadlo, Gideon & Russell, R. (2008). Robot Odor Localization: A Taxonomy and Survey. I. J. Robotic

Res. 27. 869-894.

[37] Lochmatter, Thomas & Raemy, Xavier & Martinoli, A. (2007). Odor Source Localization with Mobile Robots. 46.

[38] Ferri, Gabriele & Caselli, Emanuele & Mattoli, Virgilio & Mondini, Alessio & Mazzolai, Barbara & Dario, Paolo. (2009). SPIRAL: A novel biologically-inspired algorithm for gas/odor source localization in an indoor environment with no strong airflow. Robotics and Autonomous Systems. 57. 393-402.

[39] (2017). FIWARE – a European success story (Projects Story) Digital Strategy - European Comission. Recuperado 30 de noviembre de 2021, de https://digital-strategy.ec.europa.eu/en/news/fiware-european-success-story

[40] Tejado A. (2018). Deliverable 5.2: FIWARE Go-to-market strategy. Project: Bringing FIWARE to the NEXT step (FI-NEXT).

[41] Araujo, Victor & Mitra, Karan & Saguna, Saguna & Åhlund, Christer. (2019). Performance evaluation of FIWARE: A cloud-based IoT platform for smart cities. Journal of Parallel and Distributed Computing. 132.

[42] FIWARE Catalogue. (s. f.). FIWARE. Recuperado 30 de noviembre de 2021, de https://www.fiware.org/developers/catalogue/

[43] Celesti, Antonio & Fazio, Maria & Galán, Fermín & Glikson, Alex & Mauwa, Hope & Bagula, Antoine & Celesti, Fabrizio & Villari, Massimo. (2019). How to Develop IoT Cloud e-Health Systems Based on FIWARE: A Lesson Learnt. Journal of Sensor and Actuator Networks. 8. 7.

[44] Limosani, Raffaele & Manzi, Alessandro & Fiorini, Laura & Dario, Paolo & Cavallo, Filippo. (2019). Connecting ROS and FIWARE: Concepts and Tutorial.

[45] Velasquez, Washington & Tobar-Andrade, Luis & Cedeño Campoverde, Ivan. (2021). Monitoring and Data Processing Architecture using the FIWARE Platform for a Renewable Energy Systems.

[46] Zhao, Na & Luo, Yudong & Deng, Hongbin & Shen, Yantao. (2017). The deformable quad-rotor: Design, kinematics and dynamics characterization, and flight performance validation. 2391-2396.

GLOSARIO

2D: 2-Dimensiones

3D: 3-Dimensiones

API: Application Programming Interface

CAD: Computer-Aided Design

CFC: Clorofluorocarbonos

CFD: Computational Fluid Dynamics

CH4: Metano

CO: Monóxido de carbono

CO2: Dióxido de carbono

CPU: Central Processing Unit

CPAL: Common Public Attribution License

CPS: Cyber-physical System

CPPS: Cyber-physical Production System

CSV: Comma-separated Values

EDAR: Estación Depuradora de Aguas Residuales

ERP: Enterprise Resource Planning

EPA: United States Environmental Protection Agency

FID: Flame Ionization Detector

GDS: Gas Dispersion Simulation

GE: Generic Enabler

GPS: Global Positioning System

GRVC: Grupo de Investigación de Robótica, Visión y Control

GSO: Glowworm Swarm Optimization

H2S: Sulfuro de hidrógeno

HFC: Hidrofluorocarbonos

IMU: Inertial Measurement Unit

IIOT: Internet de las Cosas Industrial

IoT: Internet of Things

IoT-A: Internet of Things Architecture

IP: Internet Protocol

IPCC: Intergovernmental Panel on Climate Change

JSON: JavaScript Object Notation

KPI: Key Performance Indicator

LWM2M: Lightweight Machine-to-Machine

LiDAR: Light Detection and Ranging

M2M: Machine-to-Machine

MAPIR: MAchine Perception and Intelligent Robotics

MOX: Metal-oxide

MRO: Mobile Robot Olfaction

MRPT: Mobile Robot Programming Toolkit

MUD: Mercado Único Digital

N2O: Óxido Nitroso

NGSI: Next Generation Service Interface

OCB: Orion Context Broker

OMA: Open Mobile Alliance

PID: Photoionization Detector

PSO: Particle Swarm Optimization

PYME: Pequeña y Mediana Empresa

PZP: Personal Zone Proxy

PZH: Personal Zone Hub

REST: Representational State Transfer

ROS: Robot Operating System

SaaS: Software as a Service

SDK: Software Development Kit

SF6: Hexafluoruro de azufre

SITL: Software in the Loop

SLIM: Source Likelihood Map

SPIRAL: Searching Pollutant Iterative Rounding Algorithm

SSO: Single Sign-On

TIC: Tecnologías de la Información y la Comunicación

UAV: Unmanned Aerial Vehicle

UAL: Unmanned aerial vehicle Abstraction Layer

UL: UltraLight

UMA: Universidad de Málaga

US: Universidad de Sevilla