

**ADAPTACIÓN E IMPLEMENTACIÓN DEL SISTEMA TCAS II EN AERONAVES
NO TRIPULADAS**

Memoria del Trabajo Final de Grado
Gestión Aeronáutica
realizado por
Cristian Sánchez Arias
y dirigido por
Romualdo Moreno Ortiz
Sabadell, 5 de Julio de 2016.

El sotasignat, Romualdo Moreno Ortiz.

Professor/a de l'Escola d'Enginyeria de la UAB,

CERTIFICA:

Que el treball a què correspon aquesta memòria ha estat realitzat sota la seva direcció per en Cristian Sánchez Arias.

I per tal que consti firma la present.

Signat:

Sabadell, 05 de Juliol de 2016.

HOJA DE RESUMEN – TRABAJO FINAL DE GRADO DE LA ESCUELA DE INGENIERIA

Título del Trabajo Final de Grado (obligatorio en tres idiomas: Català, Castellà, Anglès)	
Autor[a]: Cristian Sánchez Arias.	Fecha: <i>Juliol, 2016</i>
Tutor[a]/s[es]: Romualdo Moreno Ortiz, Ernesto Santana.	
Titulación: Gestió Aeronàutica	
Palabras clave (mínimo 3) <ul style="list-style-type: none">• Català: simulació, sistema de seguretat aèria, drons (UAV).• Castellà: simulación, sistema de seguridad aérea, drones (UAV).• Anglès: simulation, aviation security system, drones (UAV).	
Resumen del Trabajo Final de Grado (extensión máxima 100 palabras) <ul style="list-style-type: none">• Català: En el següent Treball Fi de Grau s'exposarà l'adaptació i implementació del actual sistema de seguretat aèria TCAS II en els UAVs. Per portar-ho a terme, es farà un estudi sobre com funciona l'algoritme del TCAS II en les aeronaus i es modificarà per a que tingui la mateixa utilitat i efectivitat en els UAVs. Un cop estudiat i modificat l'algoritme, aquest s'adaptarà al codi del programa de simulació aèria Mission Planner per a mostrar les seves funcionalitats en els UAVs. Finalment, es mostrarà mitjançant simulacions com amb aquest sistema els UAVs podrien compartir l'espai aeri de forma segura amb altres aeronaus.• Castellà: En el siguiente Trabajo Final de Grado se expondrá la adaptación e implementación del actual sistema de seguridad aérea TCAS II en los UAVs. Para llevarlo a cabo, se hará un estudio sobre cómo funciona el algoritmo del TCAS II en las aeronaves y se modificará para que tenga la misma utilidad y efectividad de los UAVs. Una vez estudiado y modificado el algoritmo, éste se adaptará al código del programa de simulación aérea Mission Planner para mostrar sus funcionalidades en los UAVs. Finalmente, se mostrará mediante simulaciones como con este sistema los UAVs podrían compartir el espacio aéreo de forma segura con otras aeronaves.• Anglès: On the following Final Grade Work the adaptation and the implementation of the current flight security system TCAS II on the UAVs is going to be exposed. We are going to do a study about how works the TCAS II algorithm on the airplanes and it is going to be modified in order to get the same utility and effectiveness in the UAVs. Once the study and the modifications of the algorithm are done, this one is going to be adapted to the code of the simulation system Mission Planner in order to show its features on the UAVs. Finally, we will show through simulations how this system would allow the UAVs share the flight space with other airplanes in a safe way.	

Capítulo 1: Introducción.	5
1.1. Motivación y presentación general del trabajo.	5
1.2. Objetivos.	6
1.3. Planificación del trabajo.	7
1.4. Organización de la memoria.	8
Capítulo 2: Estudio de viabilidad.	11
2.1. Establecimiento del alcance del sistema.	11
2.2. Estudio de la situación actual.	12
2.3. Definición de Requisitos del Sistema.	14
2.4. Estudio Alternativas de Solución	15
Capítulo 3: Fundamentos.	17
3.1. Estado del arte.	17
3.2. Glosario de términos.	19
3.3. Estudio y análisis del sistema de seguridad TCAS II.	19
3.4. Diagrama de flujo del algoritmo.	29
3.5. Introducción al software Mission Planner.	30
Capítulo 4: Implementación.	33
4.1. Adaptación del sistema TCAS II a los UAVs.	33
4.2. Implementación de la adaptación en los UAVs.	35
4.3. Resultados de simulación.	44
Capítulo 5: Conclusiones.	53
5.1. Conclusiones.	53
5.2. Objetivos alcanzados.	53
5.3. Futuras implementaciones.	54
Capítulo 6: Bibliografía.	55

Capítulo 1: Introducción

1.1. Motivación y presentación general del trabajo.

Actualmente, los UAVs son aeronaves no tripuladas que sólo tienen conexión directa con su estación base. Esto hace que los UAVs estén reconocidos como un peligro por las instituciones aéreas como EASA y FAA, ya que éstas no son capaces de obtener información en vuelo de los UAVs como pueden hacerlo con las aeronaves. A parte de ello, los propios UAVs no contienen un dispositivo capaz de mantener un vuelo autónomo de manera segura como lo tienen las aeronaves con el sistema de seguridad aérea TCAS II.

Es por ello que este proyecto está centrado en el estudio del sistema de seguridad aérea TCAS II de las aeronaves para adaptarlo e implementarlo en las controladoras de los UAVs. En él podremos ver el análisis que se ha hecho al sistema TCAS II, de qué manera se ha adaptado en los UAVs, sus justificaciones y, por último, la implementación que se ha llevado a cabo.

La implementación del sistema de seguridad aérea TCAS II está realizada sobre el programa de vuelo Mission Planner, un programa de código abierto que es utilizado actualmente por una amplia comunidad de usuarios como software de planificación de misiones y monitorización y control de vuelos en estaciones de control en tierra para UAVs.

Una vez modificado el código, podremos simular en el programa Mission Planner, junto a una máquina virtual, diferentes situaciones entre UAVs para comprobar cuál es su respuesta a la implementación que hemos llevado a cabo hasta llegar al punto que ellos respondan y actúen de forma autónoma de la manera más eficaz y óptima.

La implementación final del proyecto consistiría en comprobar de manera real la reacción de los UAVs tras implantarle el código generado en este proyecto a través del mismo Mission Planner enlazado a su controladora de vuelo.

Las motivaciones que nos llevan a desarrollar este proyecto son las de mejorar el sector de los UAV en el mundo aeronáutico. Los UAVs no están concebidos al mismo nivel de importancia que otras aeronaves ya que, por las causas nombradas anteriormente, estas aeronaves todavía no se han adaptado bien al espacio aéreo y no es viable ni seguro dejar a los UAVs compartir el espacio aéreo con otras aeronaves.

Desde los inicios de los UAVs hemos podido ver numerosas noticias de accidentes o incidentes causados por UAVs multirrotores a aeronaves comerciales debido a que han obstruido o colisionado con ellas. Estos problemas son debidos a que las torres de control y las aeronaves no son capaces de localizar los UAVs ya que éstos no están capacitados para mostrar su localización a cualquier estación que no sea la propia. La solución a todos estos problemas, que son los que están impidiendo a los UAVs compartir el espacio aéreo con otras aeronaves, es dotar a los UAVs de un sistema de seguridad aérea como el TCAS II.

Esta solución es una gran motivación debido a que, una vez adaptado e implantado el sistema en los UAVs, éste no comporta un gran coste de producción y sería un producto que tendría una entrada muy rápida al mercado de los UAVs multirrotores y se podría estabilizar en él con gran facilidad debido a que podría convertirse en un dispositivo aceptado por las instituciones haciéndolo obligatorio en los UAVs para que estos puedan evolucionar y definitivamente establecerse en el espacio aéreo sin comportar ningún problema de seguridad aérea.

1.2. Objetivos.

Para llevar a cabo el desarrollo del proyecto presentado de manera resumida en el apartado anterior, nos planteamos una serie de objetivos:

En primer lugar, un objetivo fundamental consiste en realizar un estudio completo del algoritmo utilizado en el TCAS II, entender detalladamente su procedimiento y su funcionamiento en todas las situaciones que se llevan a cabo en un vuelo.

Seguidamente, otro objetivo será realizar la mejor adaptación del propio sistema a los UAVs, ya que las aeronaves actuales y los UAVs multirrotores tienen un comportamiento muy diferente. Por lo tanto, muchas de las reacciones que genera el sistema sobre la aeronave tienen que ser adaptadas a las prestaciones de los UAVs.

Una vez tengamos el código lo mejor adaptado posible para los UAVs, nuestro próximo objetivo será su propia implementación. Para ello implementaremos el código en el programa Mission Planner para que éste simule las acciones del sistema TCAS II de la manera más eficaz. Es muy importante implementarlo de la mejor manera para que no genere problemas en el sistema, de manera que el vuelo de los UAVs resulte seguro. Utilizaremos una máquina virtual junto al Mission Planner para que simule el comportamiento de los UAVs y con el programa Mission Planner probaremos la adaptación del sistema.

Por último, una vez el código funcione correctamente, nuestro último objetivo será llevar a cabo pruebas en UAVs reales y que éstos reaccionen correctamente, de la misma manera que habremos comprobado en el simulador virtual.

En definitiva, el objetivo genérico del proyecto es dotar a los UAVs de un sistema de seguridad aérea que les permita compartir el espacio aéreo con otras aeronaves de una forma segura y eficaz. Para ello tenemos el objetivo de conseguir que el UAV esté siempre presente en los radares de las demás aeronaves y de las torres de control y que sean los propios UAVs los que se adapten en el espacio aéreo evitando posibles conflictos. Para ello, llevaremos a cabo todos los puntos nombrados anteriormente para que los UAVs sean los que reaccionen en los conflictos modificando sus rutas y permitiendo a las demás aeronaves

continuar con sus rutas sin tener que modificarla, pero siendo conscientes de que un UAV está evitando el conflicto.

1.3. Planificación del trabajo.

Febrero: Estudio del entorno software de simulación y análisis bibliográfico.

Semana 08-14: Estudio del código del software de código abierto Mission Planner.

Semana 15-21: Estudio del código del software de código abierto Mission Planner.

Semana 22-28: Búsqueda de trabajos previos sobre el sistema TCAS aplicado a los UAVs.

Semana 29-06: Selección del documento del algoritmo TCAS a implementar.

Marzo: Análisis exhaustivo del algoritmo seleccionado para establecer las modificaciones necesarias para su implementación en UAVs.

Semana 07-13: Estudio del algoritmo del sistema TCAS implementado en aeronaves.

Semana 14-20: Estudio del marco teórico del sistema TCAS.

Semana 21-27: Redacción del informe donde se explica detalladamente cómo funciona el algoritmo del sistema TCAS II y su desarrollo en las aeronaves cuando este funciona.

Semana 28-03: Redacción de un Estudio de Viabilidad sobre la implementación de este proyecto en el mercado real de los UAV.

Abril: Programación del código preliminar necesario para la implementación del sistema TCAS en el Mission Planner.

Semana 04-10: Creación de código en el Mission Planner para generar un punto aleatorio que simulará ser otro UAV en la simulación.

Semana 11-17: Redacción del Informe Previo a entregar sobre el TFG y acabar el código a implementar en el Mission Planner sobre la generación de un punto aleatorio en el espacio aéreo que simulará ser otro UAV.

Semana 18-24: Creación del código en el que se establezcan rangos de seguridad en los UAV para empezar a trabajar en los avisos cuando los rangos se encuentren.

Semana 25-01: Creación del algoritmo para que los UAV se muevan. Hacer que los UAV se muevan a través de la Máquina Virtual y salte el aviso de TA cuando el rango salte.

Mayo: Implementación de los mecanismos de alerta del TCAS en el Mission Planner.

Semana 02-08: Implementar el algoritmo que genere el aviso del rango TA.

Semana 09-15: Implementar el código que simule el rango RA.

Semana 16-22: Finalizar implementación del código que simule el rango RA y empezar el código que calcule la corrección en vuelo de los UAV.

Semana 23-29: Finalizar el código de cálculo de la maniobra a ejecutar para evitar la colisión.

Semana 30-05: Redacción de la memoria del trabajo hasta la fecha.

Junio: El objetivo es acabar el código que simule el sistema TCAS II en los UAV.

Semana 06-12: Implementar el código que ejecute la maniobra anticolidión.

Semana 13-19: Generar el código que ejecute la reacción de los UAV para evitar la colisión.

Semana 20-26: Finalizar memoria.

Semana 27-03: Finalizar memoria.

Julio: El objetivo es tener la memoria acabada y hacer la presentación del trabajo.

Semana 04-10: Finalizar y entregar el TFG, y empezar a hacer la presentación de soporte.

Semana 11-17: Finalizar la presentación de soporte.

Semana 18-24: Presentar el trabajo de TFG.

1.4. Organización de la memoria.

En el "*capítulo 1: Introducción*" daremos a conocer qué motivaciones nos han llevado a desarrollar este proyecto y qué objetivos nos hemos establecido a superar con el fin de cubrir unas necesidades presentes en el sector de los UAVs en el mundo aeronáutico.

En el "*capítulo 2: Estudio de viabilidad*" analizaremos la situación actual y el motivo por el cual nuestro proyecto aportaría una mejora a ella. En él, también, analizaremos el impacto de nuestro proyecto y qué otras soluciones podrían haber para solventar la situación actual.

En el "*capítulo 3: Fundamentos*" entraremos más concretamente en el sistema de seguridad aérea TCAS II y su algoritmo. Primero de todo presentaremos otros proyectos en los que se han desarrollado proyectos parecidos al nuestro o en el que se analiza la posibilidad de adaptar el sistema en los UAVs. Seguidamente analizaremos cómo funciona el algoritmo del sistema TCAS II. Por último, se hará una introducción al software Mission Planner con el cual trabajaremos en las pruebas de simulación posteriormente.

En el "*capítulo 4: Implementación*", primeramente, adaptaremos el algoritmo del sistema TCAS II para los UAVs en función del estudio hecho previamente. Una vez adaptado, implementaremos el algoritmo en el código del software Mission Planner para que éste ejecute las funciones del sistema de seguridad aérea TCAS II en las posteriores simulaciones. Por último, mostraremos los resultados de las simulaciones llevadas a cabo desde el software Mission Planner para mostrar el funcionamiento de la adaptación e implementación desarrollada.

En el "*capítulo 5: Conclusiones*" mostraremos las conclusiones a las cuales hemos llegado una vez se ha concluido el proyecto. En él también mostraremos qué objetivos se han conseguido y cuáles no se han completado parcialmente o completamente. Por último, presentaremos una serie de mejoras y avances que se podrían implementar en el futuro con la finalidad de mejorar y ampliar el proyecto desarrollado.

En el "*capítulo 6: Bibliografía*" daremos a conocer cuál ha sido la bibliografía que se ha utilizado y nombrado durante el curso del proyecto.

Capítulo 2. Estudio de viabilidad.

2.1. Establecimiento del alcance del sistema

- **Descripción general del sistema:**

El trabajo planteado se basa en la implementación del sistema de seguridad aérea TCAS II en el software de los UAV actuales.

El objetivo principal es poder dotar a los UAV de los recursos para poder adaptarse al espacio aéreo de una forma autónoma gracias a las prestaciones con las que le dotaría el sistema TCAS II. Este sistema actuaría de una forma autónoma y permitiría a los UAV la capacidad de obtener una ruta predeterminada, y así, poder volar y evitar posibles conflictos aéreos sin necesidad de perder en un futuro su ruta inicial, solo modificándola si se genera algún conflicto aéreo.

El método de trabajo propuesto para este proyecto es trabajar a través del programa Mission Planner, el cual nos da la libertad de poder modificar su código con la finalidad de adaptarlo a nuestras necesidades. De esta manera podremos desarrollar los algoritmos necesarios que simulen el trabajo que efectúa el TCAS II.

El procedimiento a seguir será investigar cómo funciona el sistema TCAS II y cuáles son los algoritmos que éste utiliza con las aeronaves ya que, a través de su estudio, tendremos una base de la cual partir y en ella trabajar y modificarla en función de los requerimientos que nos presenten los UAV debido a su complejidad única en comparación con las aeronaves convencionales.

- **Objetivos EVS:**

Nuestro estudio de viabilidad se basará en unos puntos concretos para poder determinar si la implementación del sistema creado es viable en el mercado actual y en su utilización en los UAV actuales.

Para ellos seguiremos unos determinados pasos:

1) Estudio del mundo aeronáutico de UAV, en concreto las prestaciones de estas aeronaves en el mercado y sus posibilidades en función de las capacidades de los paquetes software existentes.

2) Importancia del sistema TCAS II en el mundo aeronáutico y sus funciones actuales en el mercado.

3) Análisis del impacto que tendría la implementación del sistema TCAS II en los UAVs actuales.

4) Análisis de la utilidad real que tendría hoy en día el sistema TCAS II en función de los trabajos que se realizan y de la normativa aérea establecida por las instituciones reguladoras de la operación de UAVs.

5) Análisis de las posibilidades que podría llegar a tener el sistema TCAS II sin tener en cuenta posibles restricciones legales.

2.2. Estudio de la situación actual.

En este punto del estudio de viabilidad nos basaremos en analizar en qué situación se encuentra el mercado de UAV y su implementación laboral en función de las tendencias empresariales y de la adaptación a la legalidad establecida.

Primero de todo, los UAV tienen una tendencia empresarial muy alejada del objetivo de nuestro proyecto, y es que, hoy en día, un gran porcentaje de los trabajos laborales que se llevan a cabo con UAVs se basan en el mundo filmográfico y en el de instrucción para el título de piloto. Esta realidad podría ser un pequeño escalón a superar en el principio de la entrada en el mercado, pero no acabaría siendo un impedimento ya que no llegaríamos a competir con este mercado, sino que estaríamos abriendo nuevas vías de trabajo con UAV y con posibilidad de acabar apoyando ese estilo de trabajo con UAV gracias a nuestras nuevas prestaciones.

Es por ello que la causa de que actualmente la mayoría de trabajos sean puramente filmográficos es a causa de que no existe ningún producto parecido al que se va a desarrollar en este proyecto. La elaboración de este dispositivo de seguridad aérea para los UAVs sería imprescindible para generar otra línea de trabajo mucho más amplia en los UAVs.

Seguidamente, podemos encontrar numerosas trabas legales en la implementación de nuestro sistema en el mercado de UAV. Los problemas que nos podríamos encontrar serían:

1) Límites de vuelo por espacio aéreo, donde un UAV no puede superar una distancia de separación, de lo que podríamos considerar en nuestro caso, su estación base. Y es que, actualmente, un UAV no puede volar a más de 150 metros de altura ni alejarse a más de 500 metros de distancia de su estación base. Esta restricción nos perjudicaría ya que estas distancias están justificadas por la visión del piloto en el momento de pilotar el UAV, donde no se permite que el mismo piloto pierda la visión del UAV por si tiene que actuar debido a un conflicto aéreo.

Eso justifica la poca distancia que permite separarse, es decir, no contempla la posibilidad de que el UAV se pilote de forma automática con una ruta predeterminada y el sistema TCAS II establecido en el UAV hiciese la función de piloto para prevenir los conflictos que pudiesen producirse con su tráfico circundante.

Pero si se analiza la situación, esta barrera legal es una consecuencia de que en el mercado no exista ningún dispositivo capaz de cumplir todos estos requisitos. En nuestro caso, la implementación del sistema TCAS II daría la posibilidad a los drones de poder cubrir esas necesidades de seguridad aérea en vuelo autónomo, convirtiendo esta implementación en la solución al problema.

2) También podemos encontrar otro problema de espacio aéreo que dificultaría la función de nuestro sistema. Y es que, actualmente, no está permitido para los UAV volar en ciertos espacios aéreos, los cuáles engloban todo ese espacio donde haya un tráfico aéreo de aeronaves, ya sean comerciales o de aviación ultraligera, ni sobrevolar espacios urbanos por el peligro de accidentes que puedan causar daños a seres humanos o a edificios, por fallo del UAV o del mismo piloto.

Con este límite legal, los UAVs están limitados en su vuelo ya que no pueden ser controlados por nadie más a parte del piloto.

La solución a ello sería la implementación del sistema TCAS II en los UAV ya que, como actualmente sólo puede ver la información de la posición del UAV y su control el propio piloto, con el nuevo sistema podríamos generar información abierta para que las torres de control puedan obtener su información e integrar a los UAVs como una aeronave más del espacio aéreo.

En definitiva, el nuevo sistema le proporcionaría a los UAVs las prestaciones necesarias para poder asegurar a las demás aeronaves una seguridad en vuelo y la posibilidad a ATC de conocer la posición de UAVs en su espacio de influencia y así poder tenerlos en cuenta en sus maniobras.

Por otro lado, la prohibición de poder sobrevolar espacios urbanos es un tema ajeno a las prestaciones de nuestro sistema, pero es un problema que, de la misma forma que en el caso anterior, con un sistema capaz de mostrar información a ATC y poder adaptarse al espacio aéreo en función a las otras aeronaves, se puede solucionar.

3) Por último, como problema legal, nos encontramos una restricción aérea para los UAVs que afectaría a la autonomía de vuelo. El problema que nos encontramos es que ningún UAV puede superar el peso de 25Kg en vuelo si este no es uno militar o bien es un UAV matriculado y el piloto tiene la habilitación, como mínimo, de piloto de

aviación deportiva. Por lo tanto, este límite de peso nos afectaría en la autonomía de vuelo de los UAV, los cuales difícilmente llegarían a sobrepasar la hora de vuelo. Por lo tanto, los UAV no podrían hacer unas rutas de vuelo de largo alcance, quitando peso entonces a las prestaciones que te pueda ofrecer nuestro sistema de seguridad aérea en los UAV en según qué aplicaciones.

2.3. Definición de Requisitos del Sistema

Los requisitos del sistema se basarán en unos requerimientos muy básicos ya que la implementación del sistema propuesto no requiere un gran cambio en la configuración de los UAVs.

1) Documentación del sistema TCAS II para poder analizar su funcionamiento. El análisis hecho en su documentación servirá para poder sentar las bases de los algoritmos a utilizar en el código del UAV.

2) Será necesaria la utilización del programa Visual Studio para la creación de los algoritmos que simularán las funciones del sistema TCAS II. Una vez creadas no será necesaria la utilización de dicho programa para poder volar los UAV con estas prestaciones.

3) Será necesario el programa de simulación de vuelo Mission Planner, ya que éste es el mismo que está vinculado al programa Visual Studio y nos permite la modificación del código para poder introducir la función del Sistema TCAS II. Este programa será siempre necesario para establecer la ruta del UAV y poder llevar el control de su vuelo y enviar la información adecuada para que actúe el sistema TCAS II en caso que fuese necesario por un conflicto aéreo. El programa Mission Planner será el que hará la función de estación base de los UAVs.

4) Referente al UAV, éste no necesitará ningún añadido diferente a los que llevan actualmente. Sólo será importante establecer un dispositivo que asegure la conexión completa entre la controladora del UAV y la estación base, ya que todas las acciones llevadas a cabo por el sistema TCAS II implementarán en la estación base y se enviarán al UAV calculadas para que éste sólo reciba el movimiento que tiene que hacer para evitar el conflicto.

2.4. Estudio de Alternativas de Solución

Gracias a que el mercado de los UAV es relativamente nuevo, hoy en día no existen numerosos dispositivos que puedan eclipsar la función que se quiere desarrollar en este proyecto.

Hoy en día solo podemos encontrar algunos de ellos que tienen la opción de establecer al UAV una ruta predeterminada, pero estos no son capaces de modificar la ruta una vez esta está en marcha.

Si nos centramos en el mercado actual, podemos ver que DJI es la marca reina en el mundo UAV, donde un gran porcentaje de consumidores de materiales de UAV, ya sea por hobby o por profesión, acuden a él. Es por eso, que si nos fijamos en los dispositivos creados por DJI que se asemejen a nuestro proyecto, sólo podemos encontrar uno denominado DATA-Link. Es un dispositivo muy simple, el cual está muy limitado en lo que se refiere a prestaciones.

Este dispositivo funciona de manera que el UAV está conectado por telemetría a un dispositivo IOS, concretamente IPAD. El IPAD está conectado por Bluetooth al dispositivo DATA-link, donde en el mismo se dibuja una ruta aérea con una simulación de Google maps. Esta aplicación sólo ofrece la posibilidad de establecer un número de waypoints concretos y con unas velocidades y alturas concretas.

El problema lo podemos encontrar en que la información de su posición en el espacio aéreo sólo lo recibe su estación base por telemetría y ninguna aeronave puede llegar a notar su presencia a través de otros dispositivos. Una vez se genera un conflicto aéreo, la única manera de que el UAV reaccione es mediante una intervención manual del piloto. Una vez éste ha intervenido y parado el vuelo autónomo del UAV, la ruta establecida se pierde.

En conclusión, nuestro sistema permitiría al UAV un nivel de autonomía y evasión de colisiones en vuelo que no se puede encontrar en ningún producto actual del mercado.

Capítulo 3: Fundamentos.

3.1. Estado del arte.

En la bibliografía especializada podemos encontrar referencias de proyectos en los que se afirma haber aplicado, o intentado aplicar, el sistema TCAS II en UAVs. A continuación se comentan brevemente algunas de estas referencias:

- UAS Safety: Unmanned Aerial Collision Avoidance System (UCAS) [1]

En este documento podemos ver de la mano de la FAA (Federal Aviation Administration) cómo se explican los UAS (Unmanned Aerial System) y la implementación de ellos en los UAVs. En este documento se afirma, según AOPA (American Owners and Pilots Association) que los UAV no pueden tener UAS ya que los UAV no pueden detectar y evitar otra aeronave.

- Implementación TCAS con sensores [2]

En este documento podemos ver cómo han utilizado el sistema TCAS II para poder crear un sistema de seguridad aérea. La diferencia es que, en este proyecto, el TCAS II es sólo una parte del sistema. Únicamente detectan los peligros basándose en el sistema TCAS II, pero lo hacen a través de sensores en vez de por GPS. Una vez tienen el peligro, transmiten la corrección para que el piloto la haga.

La diferencia con nuestro proyecto sería que el nuestro se basaría al completo en el sistema TCAS II y el UAV detectaría el peligro y lo evitaría de manera autónoma, sin la interacción del piloto.

- Can the Traffic Collision Avoidance System (TCAS) be used to prevent collisions with drones? [3]

Este artículo está publicado en la página web quora. Propone la opción de realizar el proyecto que estamos desarrollando y exponiendo.

En él podemos ver dos opiniones que ejemplifican muchas de las deducciones que hemos expuesto en el trabajo.

El primero de ellos, Paul Tomblin, es un piloto privado (PP-ASEL-IA). En su opinión da a conocer que, el problema más grande de introducir un sistema de seguridad como el TCAS II en los UAV, sería que el sistema consumiría bastante energía, por lo tanto, los UAV necesitarían más alimentación.

El segundo es Glen Stansfield, ingeniero de aviación civil durante 42 años. En su opinión muestra su optimismo con introducir el sistema en los UAV, concretando exactamente qué modelo del sistema es el mejor para los UAV. De la misma manera

que hemos expuesto nosotros, Glen explica como el modelo TCAS sólo alerta del rango TA, el TCAS II implementa la función RA para evitar colisiones de forma vertical y el TCAS III hace la misma función que el TCAS II pero con correcciones en el RA verticales y horizontales. Finalmente da su opinión sobre que el TCAS II sería el modelo más idóneo para los UAV, tal y como nosotros hemos escogido.

- CATUAV is developing a TCAS for miniUAV [4]

La empresa aeronáutica CATUAV está desarrollando un sistema TCAS para un miniUAV basado en el mismo protocolo y comunicaciones que se usan en la aviación general, el cual no requiere una instalación de sistemas a bordo de los UAV. Este proyecto está aún en desarrollo y no hay una muestra de su proyecto.

- ACAS Manual [5]

El siguiente manual sobre ACAS contiene un apartado que nos habla de la posible implementación de un sistema ACAS en los UAV. Nos expone los requerimientos que habría en diferentes casos sobre cómo implementarlo, si es sólo con TA, con TA y RA, con correcciones, etc.

Por último, expone como conclusión que su implementación sería posible, pero se tendría que adaptar por el hecho de que en los UAV no hay un piloto en cabina.

- UAV might inspire new TCAS rules [6]

En este artículo se expone la necesidad de adaptar el sistema TCAS a los UAV para que éstos puedan compartir el espacio aéreo de una forma segura. En él se pone en duda qué modelo sería el más adecuado, si el TCAS II o TCAS III, sin dar una respuesta a ello.

- NASA, FAA, Industry Conduct Initial Sense-and-Avoid Test [7]

En este artículo podemos encontrar las pruebas que ha llevado a cabo la NASA con un sistema denominado ACAS Xu, un UAS adaptado a los UAV. El objetivo de este proyecto es analizar la respuesta del algoritmo del sistema sobre los UAV.

- UAS sobre integración en el espacio aéreo no segregado [8]

En este artículo podemos encontrar una descripción de los sistemas UAS y ACAS, el cual tiene un apartado en el que especifica la adaptación de los propios en los UAV.

3.2. Glosario de términos.

UAV	Unmanned Aerial Vehicle.
EASA	European Aviation Safety Agency.
FAA	Federal Aviation Administration.
TCAS	Traffic alert and Collision Avoidance System.
TA	Traffic Advisory.
RA	Resolution Advisory.
TAU	Variable τ que define el tiempo hasta la colisión de dos aeronaves.
CPA	<i>Closest Point Approach</i> . Punto de aproximación más cercano.
DMOD	Variable de distancia de límite horizontal.
ZTHR	Variable de distancia de límite vertical.
ALIM	Variable de distancia de límite vertical.
RA2D	Resolution Advisory 2 Dimensions
RA3D	Resolution Advisory 3 Dimensions

Tabla 3.2.1. Tabla de términos técnicos utilizados

3.3. Estudio y análisis del sistema de seguridad TCAS II.

El sistema de seguridad aérea TCAS II se basa en un sistema de avisos a las aeronaves en función de sus posiciones en el espacio aéreo. Estos avisos tienen como objetivo evitar posibles colisiones entre dos aeronaves en vuelo.

Podemos destacar dos zonas aéreas donde el TCAS II actúa. La primera de ellas es la zona TA (Traffic Advisory), el cual tiene un rango mayor al RA (Resolution Advisory). Los rangos del TA y del RA se basan en función a la propia aeronave la cual es el centro de dichos rangos. El primer rango es el del TA, con mayor espacio aéreo, y después encontramos el RA, el cual tiene un rango menor.

Las funcionalidades de ellos son diferentes. El TA simplemente se basa en reconocer los vectores de posición de las dos aeronaves para tenerlas controladas y avisar a las aeronaves de que hay un tráfico cercano.

En cambio, el RA actúa sobre las aeronaves. Presenta dos pasos, y es que, cuando dos aeronaves entran en el rango RA, en primer lugar, avisan del peligro y transmiten al piloto un movimiento para poder evitar la posible colisión. Todo y así, si las dos aeronaves siguen dentro del rango del RA y estos no han reaccionado al movimiento sugerido por el sistema TCAS II, el sistema actúa en el piloto automático y efectúa un cambio de rumbo y dirección.

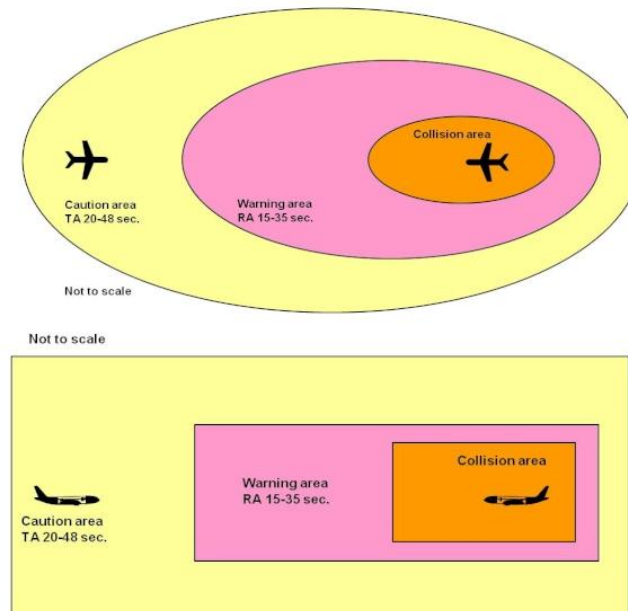


Figura 3.3.1 Rangos espacio aéreo TCAS II.

-Movimientos sugeridos por el sistema en el primer aviso del RA

Upward sense			Downward sense		
RA	Required vertical rate (ft/min.)	Aural	RA	Required vertical rate (ft/min.)	Aural
Climb	1500	Climb, climb	Descend	- 1500	Descend, descend
Crossing Climb	1500	Climb, crossing climb; climb, crossing climb	Crossing Descent	- 1500	Descend, crossing descend; descend, crossing descend
Maintain Climb ³²	1500 to 4400	Maintain vertical speed, maintain	Maintain Descent ³²	- 1500 to - 4400	Maintain vertical speed, maintain
Maintain Crossing Climb ³²	1500 to 4400	Maintain vertical speed, crossing maintain	Maintain Crossing Descent ³²	- 1500 to - 4400	Maintain vertical speed, crossing maintain
Level Off ³³	0	Level off, level off	Level Off ³³	0	Level off, level off
Reversal Climb ³⁴	1500	Climb, climb NOW; climb, climb NOW	Reversal Descent ³⁴	- 1500	Descend, descend NOW; descend, descend NOW
Increase Climb ³⁴	2500	Increase climb, increase climb	Increase Descent ³⁴	- 2500	Increase descent, increase descent
Preventive RA	No change	Monitor vertical speed	Preventive RA	No change	Monitor vertical speed
RA Removed	n/a	Clear of conflict	RA Removed	n/a	Clear of conflict

Figura 3.3.2. Movimientos evasión TCAS II.


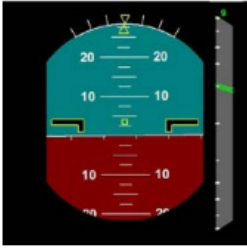

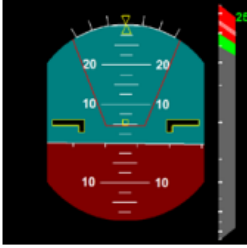
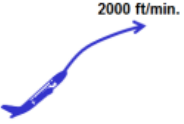

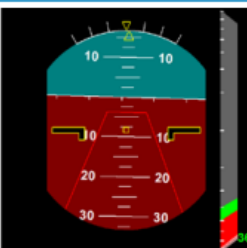
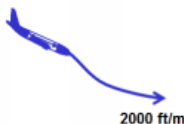
Advisory	Aural annunciation	IVSI	EFIS	Manoeuvre
Traffic advisory	Traffic, traffic			No manoeuvring
Adjust Vertical Speed (downward sense reduction to 2000 ft/min.) (version 7.0 only)	Adjust vertical speed, adjust			
Adjust Vertical Speed (upward sense reduction to 2000 ft/min.) (version 7.0 only)	Adjust vertical speed, adjust			

Figura 3.3.3. Ejemplos de avisos y maniobras aeronaves con sistema TCAS II.

Si nos centramos en su algoritmo para decidir las acciones correctoras en función de los rangos, podemos ver cómo ello funciona entorno al tiempo como variable. El sistema toma vectores precisos de información sobre las dos aeronaves y asume las trayectorias lineales. A partir de ello, el algoritmo categoriza, de una manera completa y correcta, todos los encuentros geométricos que pueden causar un RA con un intervalo de tiempo establecido

Lógica de trabajo

La forma de trabajar del TCAS II es siempre en función al tiempo. Tanto los rangos TA y RA, como los límites utilizados para las alarmas están basados en una variable de tiempo llamada tau.

La variable τ (tau) tiene la función de estimar el tiempo hasta el CPA (Closest Point Approach), según su rumbo y dirección, entre la propia aeronave y otra en peligro. Cuando el valor de tau coincide con el valor de CPA es cuando existe un peligro de colisión entre las dos aeronaves y se activan los avisos y cambios de dirección del sistema. En cambio, si estos valores no coinciden, podemos ver como τ es sólo una aproximación del valor del CPA pero estos no coinciden.

Para poder calcular el valor de tau podemos encontrar una fórmula que se define de la siguiente manera: Tau es el valor definido por *el range (r) dividido por el closure rate (ṙ)*.

$$\tau \equiv -\frac{r}{\dot{r}}$$

Los valores y métodos presentados sobre el cálculo de tau son utilizados para los parámetros horizontales del sistema TCAS II, pero para la dimensión vertical del sistema podemos encontrar unas pequeñas variables en el cálculo del llamado “*Vertical τ (tau)*”. Esta variable está definida por el cálculo de la separación vertical dividido por la “*vertical closure rate*”.

Una vez que se conocen las fórmulas para poder calcular los límites horizontales y verticales del RA, podemos ver cómo estos son empleados para la toma de decisiones del sistema sobre la aeronave.

Para saber cuándo un RA emite, necesitamos basarnos en los límites establecidos horizontal y verticalmente, y en el valor de tau en función de posición de la aeronave. Primero, horizontalmente, tenemos el valor DMOD, el cual establece el límite horizontal, y ZTHR que establece el límite vertical. Después tenemos el valor τ (tau) establecido como límite de tiempo de colisión.

Cuando el valor del *range* horizontal y la separación vertical de la aeronave es menor que los límites, y el valor tau de la aeronave es menor que el límite τ (tau), tenemos una emisión del RA.

Todos estos valores pueden variar mucho en función de la altura, velocidad y separaciones de las aeronaves. Por ello, tenemos una variable como referencia llamada *Sensitivity Level*, la cual nos da unas referencias para poder ser más precisos.

A causa de la sensibilidad del valor en función de las variables comentadas anteriormente, la fórmula presentada de tau anterior es muy poco precisa. La fórmula anterior puede conllevar 2 problemas:

- Peligros de encuentros con bajos “*range closure rates*”.
- Altas “*closure rates*” con largas distancias perdidas.

Para solucionar los problemas con la fórmula de τ (tau), podemos encontrar una fórmula más desarrollada y precisa:

$$\tau_{mod} \equiv -\frac{r^2 - DMOD^2}{r\dot{r}}$$

Podemos encontrar una tabla donde los valores de *Sensitivity Level* nos marcan unos valores concretos para los límites anotados anteriormente:

Table 1. TCAS Sensitivity Level Definition and Alarm Thresholds for RAs

Ownship Altitude (feet)	SL	TAU (sec)	DMOD (nmi)	ZTHR (feet)	ALIM (feet)	HMD (feet)
1000 - 2350	3	15	0.20	600	300	1215
2350 - 5000	4	20	0.35	600	300	2126
5000 - 10000	5	25	0.55	600	350	3342
10000 - 20000	6	30	0.80	600	400	4861
20000 - 42000	7	35	1.10	700	600	6683
> 42000	7	35	1.10	800	700	6683

Tabla 3.3.4. Tabla de límites del rango RA del sistema TCAS II en función del nivel de vuelo.

En esta tabla podemos encontrar una variable no nombrada hasta el momento, la variable ALIM. Esta variable se usa para poder determinar si un RA es correctivo o preventivo.

Modelos vectorizados del RA

El sistema TCAS II nos plantea 4 modelos diferentes, donde los algoritmos utilizados tienen una utilidad única en función de los requerimientos de la aeronave.

Modelo 1:

El primer modelo se basa en el RA2D (Resolution Advisory 2 Dimensions), donde el algoritmo funciona de la siguiente manera:

El algoritmo nos devuelve true si en los cálculos se encuentra que τ , es decir, tau, coincide con el CPA. El resultado de este algoritmo es el momento t (tau), donde habría la colisión.

Proposition 1. For all vectors $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i$, $\mathbf{v} = \mathbf{v}_o - \mathbf{v}_i$, distance $D > 0$, and time $B \geq 0$, $CD2D_\infty(\mathbf{s}, \mathbf{v}, D, B)$ returns true if and only if there exists a time $t \geq B$ where $\|\mathbf{s} + t\mathbf{v}\| < D$.

Con la siguiente fórmula se pueden calcular los espacios horizontales y verticales teniendo en cuenta la posición y la velocidad de la aeronave, y el *Sensitivity Level*. El problema es que con esta fórmula no podríamos obtener el tiempo τ (tau).

```
TCASII_RA( $\mathbf{s}_o, s_{oz}, \mathbf{v}_o, v_{oz}, \mathbf{s}_i, s_{iz}, \mathbf{v}_i, v_{iz}$ )  $\equiv$ 
  let  $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i, \mathbf{v} = \mathbf{v}_o - \mathbf{v}_i, s_z = s_{oz} - s_{iz}, v_z = v_{oz} - v_{iz}$  in
    Horizontal_RA( $\mathbf{s}, \mathbf{v}$ ) and
    Vertical_RA( $s_z, v_z$ ) and
    CD2D $_\infty(\mathbf{s}, \mathbf{v}, \text{HMD}_\ell, 0)$ ,
```

A partir de la fórmula anterior, y con un añadido, podemos encontrar el tiempo tau:

```
TCASII_RA_at( $s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, t$ )  $\equiv$ 
  let  $s = s_o - s_i, v = v_o - v_i, s_z = s_{oz} - s_{iz}, v_z = v_{oz} - v_{iz}$  in
    Horizontal_RA( $s + tv, v$ ) and
    Vertical_RA( $s_z + tv_z, v_z$ ) and
    CD2D $_{\infty}(s, v, HMD_{\ell}, t)$ .
```

Modelo 2:

El modelo 2 nos plantea una función muy parecida a la del modelo 1, también basada en el RA2D, pero en vez de devolvernos el tiempo τ (tau), si éste existe, nos devuelve un intervalo de tiempo. En este intervalo de tiempo es donde se encuentra τ (tau), pero no nos especifica el valor de τ (tau).

Proposition 2. *For all vectors $s = s_o - s_i$, $v = v_o - v_i$, and lookahead time interval $[B, T]$, $RA2D_{\ell}(s, v, B, T)$ returns true if and only if there exists a time $t \in [B, T]$ where $Horizontal_RA_{\ell}(s + tv, v)$ holds.*

Se utilizan las mismas variables que en el modelo 1, pero se añade una función para poder calcular el intervalo de tiempo que se devuelve como resultado:

```
RA2DTimeInterval( $s, v, B, T$ )  $\equiv$ 
  let  $a = \|v\|^2$ ,
     $b = 2(s \cdot v) + TAU_{\ell}\|v\|^2$ ,
     $c = \|s\|^2 + TAU_{\ell}(s \cdot v) - DMOD_{\ell}^2$  in
  if  $a = 0$  and  $\|s\| \leq DMOD_{\ell}$  then
     $[B, T]$ 
  else
    let  $\theta = \Theta(s, v, DMOD_{\ell}, 1)$  in
    if  $\|s\| \leq DMOD_{\ell}$  then
       $[B, \theta]$ 
    elseif  $(s \cdot v \geq 0$  or  $b^2 - 4ac < 0$  then
       $[T + 1, 0]$ 
    elseif  $\Delta(s, v, DMOD_{\ell}) \geq 0$  then
       $[\text{root}(a, b, c, -1), \theta]$ 
    else
       $[\text{root}(a, b, c, -1), \text{root}(a, b, c, 1)]$ 
  endif.
```


Modelo 3:

En el modelo 3 podemos encontrar una gran diferencia con los dos modelos anteriores, y es que este modelo se basa en el RA3D (Resolution Advisory 3 Dimensions), en vez de 2 Dimensiones.

Este modelo presenta similitudes con el modelo anterior, ya que sólo nos devuelve el intervalo de tiempo donde se encuentra τ (tau), pero no nos da el valor de la propia variable.

Proposition 3. *For all vectors $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i$, $\mathbf{v} = \mathbf{v}_o - \mathbf{v}_i$, and lookahead time interval $[B, T]$, if $\text{RA2DTimeInterval}_\ell(\mathbf{s}, \mathbf{v}, B, T)$ returns the time interval $[t_{in}, t_{out}]$, then for all times $t \in [B, T]$, $\text{Horizontal_RA}_\ell(\mathbf{s} + t\mathbf{v}, \mathbf{v})$ holds if and only if $t \in [t_{in}, t_{out}]$.*

La función para poder encontrar el intervalo es diferente a la del anterior modelo ya que este tiene en cuenta 3 dimensiones:

```
RA3DTimeInterval $_\ell$ ( $\mathbf{s}_o, \mathbf{s}_{oz}, \mathbf{v}_o, \mathbf{v}_{oz}, \mathbf{s}_i, \mathbf{s}_{iz}, \mathbf{v}_i, \mathbf{v}_{iz}, B, T, \text{hmdf?}$ )  $\equiv$ 
  let  $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i, \mathbf{v} = \mathbf{v}_o - \mathbf{v}_i, s_z = s_{oz} - s_{iz}, v_z = v_{oz} - v_{iz}$  in
  if hmdf? and not  $\text{CD2D}_{\infty}(\mathbf{s}, \mathbf{v}, \text{HMD}_\ell, B)$  then
     $[T, B]$ 
  elseif  $v_z = 0$  and  $|s_z| > \text{ZTHR}_\ell$  then
     $[T, B]$ 
  else
    let  $[t_{inz}, t_{outz}] = \text{RAZTimeInterval}_\ell(s_z, v_z, B, T)$  in
    if  $t_{outz} < B$  or  $T < t_{inz}$  then
       $[T, B]$ 
    else
      let  $[t_{in}, t_{out}] = [\max(B, t_{inz}), \min(T, t_{outz})]$ ,
           $[t_{inxy}, t_{outxy}] = \text{RA2DTimeInterval}_\ell(\mathbf{s}, \mathbf{v}, B, T)$ ,
           $[t_1, t_2] = [\max(t_{in}, \min(t_{out}, t_{inxy})), \max(t_{in}, \min(t_{out}, t_{outxy}))]$  in
      if  $t_{inxy} > t_{outxy}$  or  $t_{outxy} < t_{in}$  or  $t_{inxy} > t_{out}$  then
         $[T, B]$ 
      if hmdf? and  $\text{HMD}_\ell < \text{DMOD}_\ell$  and
         $(\mathbf{s} + t_{inz}\mathbf{v}) \cdot \mathbf{v} \geq 0$  and  $\|\mathbf{s} + t_{inz}\mathbf{v}\| > \text{HMD}_\ell$  then
           $[T, B]$ 
      elseif hmdf? and  $\text{HMD}_\ell < \text{DMOD}_\ell$  then
        let  $\theta = \Theta(\mathbf{s}, \mathbf{v}, \text{HMD}_\ell, 1)$ ,
             $t_3 = \text{if } \|\mathbf{v}\| = 0 \text{ then } T \text{ else } \max(B, \min(\theta, T))$  endif,
             $t_4 = \min(t_2, t_3)$  in
           $[t_1, t_4]$ 
      else
         $[t_1, t_2]$ 
    endif.
```

Modelo 4:

Por último, el modelo 4 es el más completo y complejo de todos ya que, este modelo, nos proporciona como resultado el tiempo τ (tau) en un intervalo y en RA3D.

Proposition 4. *For all ownship states s_o, s_{oz}, v_o, v_{oz} , intruder states s_i, s_{iz}, v_i, v_{iz} , and lookahead time interval $[B, T]$, $RA3D(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, B, T)$ returns *true* if and only if there exists a $t \in [B, T]$ where $TCASII_RA_at(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, t)$ holds.*

El cual se basa en la suma de funciones presentadas en los modelos anteriores:

$$\begin{aligned} RA3D(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, B, T) &\equiv \\ \text{let } [t_{in}, t_{out}] &= RA3DTimeInterval_{\ell}(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, B, T, \text{true}) \text{ in} \\ t_{in} &\leq t_{out}. \\ TCASII_RA_at(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, t) &\equiv \\ \text{let } s &= s_o - s_i, v = v_o - v_i, s_z = s_{oz} - s_{iz}, v_z = v_{oz} - v_{iz} \text{ in} \\ &\text{Horizontal_RA}_{\ell}(s + tv, v) \text{ and} \\ &\text{Vertical_RA}_{\ell}(s_z + tv_z, v_z) \text{ and} \\ &CD2D_{\infty}(s, v, HMD_{\ell}, t). \end{aligned}$$

Corrective Resolution Advisory

Los 4 modelos planteados anteriormente nos presentan las maneras de detectar una posible colisión en el espacio aéreo para que el sistema pueda avisar al piloto del peligro. Pero como se ha explicado, en el espacio RA hay 2 acciones, la primera de aviso al piloto con una sugerencia de movimiento para evitar la colisión, y más adelante, una corrección de rumbo y dirección directa del sistema TCAS II sobre la aeronave si el piloto no ha corregido el movimiento con el primer aviso del RA.

Para poder desarrollar el algoritmo que pueda calcular el movimiento requerido de corrección, antes debemos plantear otras funciones internas de la misma:

Primero la función que calcula la distancia vertical entre la aeronave y la posición de colisión:

$$\begin{aligned} \text{sep_at}(s_{oz}, v_{oz}, s_{iz}, v_{iz}, v, a, \epsilon, t) &\equiv \\ \text{let } o &= \text{own_alt_at}(s_{oz}, v_{oz}, |v|, a, \epsilon \text{ sign}(v), t), \\ i &= s_{iz} + tv_{iz} \text{ in} \\ \epsilon(o - i). \end{aligned}$$

Después tenemos la función que nos encuentra la altura que tendríamos en el punto de colisión:

```
own_alt_at( $s_{oz}, v_{oz}, v, a, \epsilon, t$ )  $\equiv$ 
  let  $s = \text{stop\_accel}(v_{oz}, v, a, \epsilon, t)$ ,
       $q = \min(t, s)$ ,
       $l = \max(0, t - s)$  in
     $\epsilon q^2 \frac{a}{2} + qv_{oz} + s_{oz} + \epsilon l v$ .
```

A continuación, la función que calcula el tiempo en el que la aeronave alcanzaría la velocidad vertical del objetivo:

```
stop_accel( $v_{oz}, v, a, \epsilon, t$ )  $\equiv$ 
  if  $t \leq 0$  or  $\epsilon v_{oz} \geq v$  then 0
  else  $\frac{\epsilon v - v_{oz}}{\epsilon a}$ 
endif.
```

Seguidamente la función que nos calcula la dirección correcta teniendo en cuenta la separación vertical con ALIM como valor límite:

```
RA_sense( $s_{oz}, v_{oz}, s_{iz}, v_{iz}, v, a, t$ )  $\equiv$ 
  let  $o_{\uparrow} = \text{own\_alt\_at}(s_{oz}, v_{oz}, v, a, 1, t)$ ,
       $o_{\downarrow} = \text{own\_alt\_at}(s_{oz}, v_{oz}, v, a, -1, t)$ ,
       $i = s_{iz} + tv_{iz}$ ,
       $u = o_{\uparrow} - i$ ,
       $d = i - o_{\downarrow}$  in
  if  $\text{sign}(s_{oz} - s_{iz}) = 1$  and  $u \geq \text{ALIM}_{\ell}$  then 1
  elsif  $\text{sign}(s_{oz} - s_{iz}) = -1$  and  $d \geq \text{ALIM}_{\ell}$  then -1
  elsif  $u \geq d$  then 1
  else -1
endif.
```

Por último, si el RA se vuelve correctivo por la falta de reacción del piloto con el primer aviso del RA, tenemos la fórmula que genera true si ese cálculo está basado en un valor de corrección y no de aviso:

```
corrective( $s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, v, a$ )  $\equiv$ 
  let  $s = s_o - s_i, v = v_o - v_i, s_z = s_{oz} - s_{iz}, v_z = v_{oz} - v_{iz}$ ,
       $t = \tau_{\text{mod}_{\ell}}(s, v)$ ,
       $\epsilon = \text{RA\_sense}(s_{oz}, v_{oz}, s_{iz}, v_{iz}, v, a, t)$  in
   $\|s\| < \text{DMOD}_{\ell}$  or
   $(s \cdot v < 0 \text{ and } \epsilon(s_z + tv_z) < \text{ALIM}_{\ell})$ ,
```

Cálculo del espacio TA (Traffic Advisory)

De la misma manera que anteriormente hemos podido ver qué valores y qué funciones calculan los valores para el espacio aéreo RA, también podemos encontrar una función que nos calcula cuándo una aeronave está dentro del primer espacio aéreo TA y una tabla de referencias en función de su *Sensitivity Level*:

$$\text{TA3D}(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, B, T) \equiv$$

$$\text{let } [t_{in}, t_{out}] = \text{RA3DTimeInterval}_t(s_o, s_{oz}, v_o, v_{oz}, s_i, s_{iz}, v_i, v_{iz}, B, T, \text{false}) \text{ in}$$

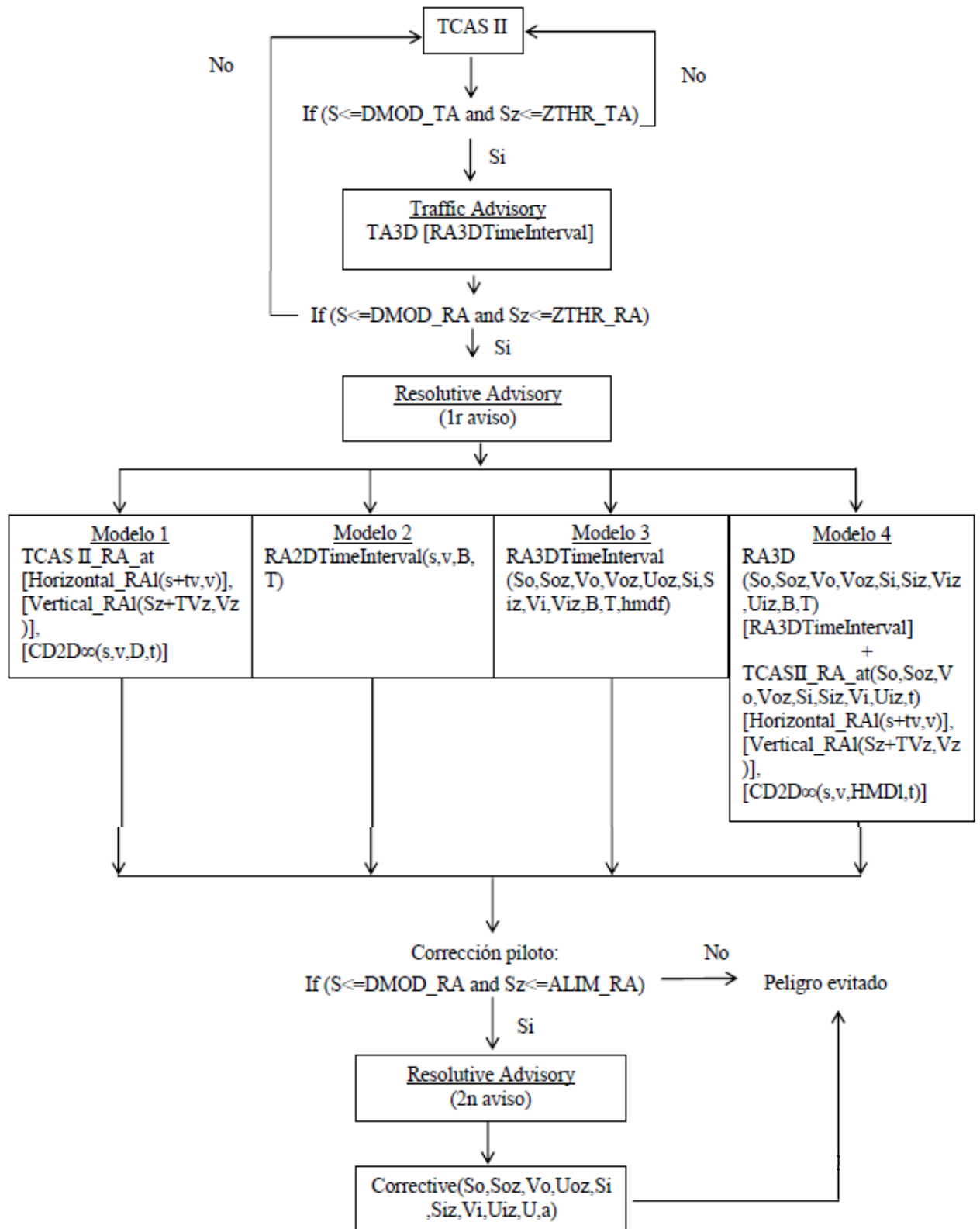
$$t_{in} \leq t_{out}.$$

Table 2. TCAS Sensitivity Level Definition and Alarm Thresholds for TAs

Ownship Altitude (feet)	SL	TAU (sec)	DMOD (nmi)	ZTHR (feet)
1000 - 2350	2	20	0.30	850
1000 - 2350	3	25	0.33	850
2350 - 5000	4	30	0.48	850
5000 - 10000	5	40	0.75	850
10000 - 20000	6	45	1.0	850
20000 - 42000	7	48	1.3	850
> 42000	7	48	1.3	1200

Tabla 3.3.5. Tabla de límites del rango TA del sistema TCAS II en función del nivel de vuelo.

3.4. Diagrama de flujo del algoritmo



3.5. Introducción al software Mission Planner.

Para contextualizar la explicación sobre las funcionalidades del software Mission Planner, en primer lugar, introduciremos algunos conceptos acerca de las controladoras de vuelo de los UAVs.

La controladora de vuelo es el dispositivo electrónico encargado de hacer funcionar al UAV de manera que éste siga el plan de vuelo deseado por el piloto. La controladora constituye el nexo de unión de los principales componentes del UAV. Componentes como motores, receptor de la emisora, GPS, IMU, etc están todos relacionados entre sí a través de la controladora de vuelo, que coordina la acción individual de cada uno de ellos para que, en conjunto, cumplan con el objetivo mencionado de ejecutar el plan de vuelo deseado por el piloto.

Las controladoras pueden ofrecer distintos modos de vuelo. Cualquier controladora dispondrá de modo de vuelo manual, en el que el piloto, humano en este caso, mediante una emisora de radio control (u otros dispositivos equivalentes, como por ejemplo un *joystick*), ejerce el control directo sobre el movimiento de la aeronave en los tres ejes, controlando también la velocidad de vuelo. Adicionalmente, algunas controladoras disponen de modo de vuelo estabilizado (conocido como Atti, en el que el piloto, también humano, controla la aeronave de forma manual pero dispone de ayudas automáticas, típicamente la estabilización de la aeronave cuando el piloto deja de ejercer acción de control. Más aún, también hay un último modo de pilotaje, el cuál es el denominado modo de vuelo GPS. En este vuelo, el control de la aeronave es del piloto pero tiene una ayuda constante de un GPS el cual le permite mantener la posición deseada.

Finalmente, algunas controladoras disponen de modo de vuelo automático. En este caso, la controladora suele denominarse autopiloto, y es capaz de ejecutar de forma automática un plan de vuelo constituido por una secuencia de órdenes almacenadas en la memoria de la controladora. Estos planes de vuelo se elaboran mediante herramientas software (como Mission Planner) que también permiten enviar el plan de vuelo elaborado a la memoria de la controladora.

En este proyecto nos hemos basado en la controladora de UAVs de Arduino denominada Ardupilot. Esta controladora nos permite gran libertad ya que tiene su software de código abierto Mission Planner, sobre el cual se ha trabajado en el proyecto. Ardupilot es una controladora configurada para poder llevar a cabo vuelos con diferentes aeronaves, ya sean multirrotores, ala fija, monomotores o vehículos terrestres. Ardupilot destaca por ser un *open source*, que como en nuestro caso, nos permite tener una sólida base de programación para nuestra aeronave y nos da la libertad para establecerle las modificaciones que nosotros deseemos en su código.



Figura 3.5.1. Controladora de vuelo Ardupilot.

Mission Planner es un software gratuito que nos permite establecer conexión con la controladora Ardupilot del UAV y utilizarla como estación base del UAV. En él podemos encontrar la posibilidad de establecerle unas rutas predeterminadas y generar acciones al UAV para que este pueda volar autónomamente. Otras muchas acciones que nos permite hacer es la opción de configurar las prestaciones de las aeronaves, configurándola de la manera que nos resulte más útil en función de la finalidad que se le quiera dar.

Podemos establecer numerosas rutas pre programadas a través de la descarga de ellas, las cuales se adaptan directamente a nuestro software. Por último, nos permite tener información constantemente de la aeronave. Podemos conocer el estado de la batería, su modo de vuelo, velocidades y alturas, configuraciones de la aeronave, recorrido de la aeronave a través de mapas. Nos transmite toda esta información a través de telemetría directamente a nuestra estación base para poder analizarla y reconocerla fácilmente.

Como hemos comentado, nos permite establecerle una ruta y llevar a cabo un vuelo autónomo, pero también nos da la posibilidad de pilotar la aeronave en FPV (First Person View. Para una explicación más detallada sobre las funciones del software Mission Planner puede consultarse la documentación publicada por Ardupilot sobre su funcionamiento en su web oficial.

Para cumplir con los objetivos de este proyecto, ha sido necesario modificar (ampliar) el código fuente de Mission Planner para dotarlo de nuevas funciones que simulan el sistema TCAS II, pudiendo establecer una relación directa con la controladora de vuelo del UAV y así llevar a cabo la función del TCAS II.

En las siguientes imágenes mostraremos unos cambios visibles en el programa Mission Planner que mostrarán los accesos al nuevo código para simular el TCAS II. Las nuevas funciones están configuradas de color azul para hacer destacarla como cambio.

Capítulo 4: Implementación.

4.1. Adaptación del sistema TCAS II a los UAVs.

Como hemos podido ver al principio de la explicación del sistema TCAS II, éste está compuesto por dos rangos de espacio. En las aeronaves, el primer rango avisa al piloto de la aparición de otra aeronave en su primer rango de peligro, el llamado Traffic Advisory. Este primer rango solo consiste únicamente en avisar al piloto de ello, y no es hasta el segundo rango (Resolution Advisory), el cual tiene menor radio, que le muestra una acción de evasión para que ejecute el piloto. En el caso de que éste no varíe su ruta, si la distancia entre aeronaves entra en el menor rango del Resolution Advisory, el sistema actúa automáticamente sobre la aeronave ejecutando un movimiento de evasión.

Ahora bien, los UAVs multirrotores son aeronaves con un comportamiento muy diferente a las aeronaves convencionales, y es por ello que estos procedimientos necesitan una modificación para ser óptimos.

En primer lugar, se han modificado las actuaciones de los rangos. El primer rango establecido es el de Traffic Advisory, avisando a la estación de tierra del UAV de la presencia de otro UAV. Después de ello, si la distancia se reduce hasta entrar en el rango del Resolution Advisory, éste no envía un movimiento de evasión al piloto para que lo ejecute, sino que actúa directamente en el UAV generando un movimiento de evasión. El porqué de este cambio en los procedimientos viene justificado por el hecho de que los UAV son aeronaves que se pilotan de manera autónoma y con ello se quiere mantener el comportamiento de los UAV evadiendo el peligro de manera autónoma sin obligar al piloto de la estación base a intervenir.

El cálculo de los propios rangos también ha sido modificado. Para hacer el cálculo, como se explicó anteriormente, existen unas variables que constituyen límites verticales y horizontales.

El límite horizontal DMOD y el límite vertical ZTHR tienen dos formas de interpretarse, y es que si la velocidad de las aeronaves es baja, éstos están definidos por distancia entre aeronaves, pero si la velocidad de las aeronaves es de crucero-alta, estos límites están definidos por una variable de tiempo llamada τ (TAU) que representa el tiempo que falta para que se genere una colisión.

Para la adaptación de las definiciones de los límites, podemos interpretar que la velocidad de los UAVs multirrotores siempre es baja, ya que difícilmente un UAV superará los 10m/s de velocidad. Por ello, definiremos en todo momento las variables límite con valores de distancias (en metros) y no con una variable de tiempo τ (TAU) como se hace con las aeronaves en el TCAS II debido a su gran velocidad de vuelo.

En consideración a las dos aeronaves que generen el conflicto aéreo, el límite que se ha establecido es una separación vertical mínima de 5 metros el rango RA. Los UAVs multirrotores, a pesar de generar viento turbulento de forma vertical hacia abajo, su potencia no llega a generar viento turbulento más de 3 metros. Es así que, en el rango TA, no se tendrá en cuenta el límite vertical ya que su único objetivo es avisar sobre un tráfico de otra aeronave en su espacio aéreo.

En los rangos horizontales será donde se establecerá la diferencia entre los dos rangos TA y RA. El primer rango, el Traffic Advisory, tendrá un límite de 150 metros. Está basado en la velocidad media sobre la tierra de los UAVs multirrotores que se ha establecido en 3 m/s, lo que ofrece a la estación tierra un margen de aviso de 50 segundos. El segundo rango horizontal tiene un rango de 70 metros, lo cual permite al UAV un margen de 23 segundos para ejecutar la maniobra de evasión si es necesaria.

Si nos centramos en qué modelo vectorial se utiliza para los multirrotores, nos hemos basado en el modelo 4 para adaptarlo a los UAVs multirrotores. Como está explicado anteriormente, el modelo 4 está basado en un cálculo de 3 dimensiones y dándonos a conocer el momento concreto de impacto con la variable τ (TAU).

En nuestro caso, como queremos saber la distancia concreta en que se generaría el impacto, nos hemos basado en el modelo 4 para conocer ese valor sobre las 3 dimensiones. Cuando hablamos de las 3 dimensiones que utilizamos, nos referimos a los 3 ejes del espacio X, Y y Z, a partir de los cuales calculamos las distancias verticales y horizontales de los UAVs, para así obtener los valores de colisión en distancia que necesitamos para su funcionamiento.

Por último, los movimientos establecidos para los UAVs multirrotores en su evasión dependerá de la situación en que se encuentren. El sistema TCAS II estará reconociendo la situación continuamente, por ello, si el aviso Resolutory Advisory salta, el UAV reconocerá cuál es la distancia de separación, tanto vertical como horizontal. El movimiento de evasión más importante en los UAVs multirrotores, al igual que para otras aeronaves, es un movimiento vertical. En los UAVs multirrotores es debido a que los motores tienen mayor facilidad de impacto en el eje vertical que en el horizontal a causa de su diseño y enfoque de los motores. Los posibles movimientos de evasión son los siguientes:

- Si un UAV tiene una altura igual o mayor al otro UAV en conflicto, el primero ascenderá 10 metros y tendrá un desplazamiento horizontal de 60 metros con la intención de salir del rango RA.
- Si un UAV tiene una altura menor al otro UAV en conflicto, el primero descenderá 10 metros y tendrá un desplazamiento horizontal de 60 metros con la intención de salir del rango RA.

- En el caso de que nuestro UAV cumpla la norma de tener una altura menor, pero su altura de vuelo sea menor de 15 metros, para evitar que su altura sea demasiado baja respecto al terreno, le hará ascender 10 metros y tendrá un desplazamiento horizontal de 60 metros con la intención de salir del rango RA.

4.2. Implementación de la adaptación en los UAVs.

- Implementación en el código del software Mission Planner:

Para poder llevar a cabo todas las acciones explicadas anteriormente, hemos tenido que modificar el código del programa para así poder generar esos nuevos botones en el programa y que éstos funcionen correctamente.

Para hacer la función de activar el sistema TCAS II, tenemos el nuevo botón añadido "Activar TCAS II".

```
private void Activar_TCAS(object sender, EventArgs e)
{
    CustomMessageBox.Show("TCAS Activado, uso único en UAV_1");
    timer2.Enabled = true;
}
```

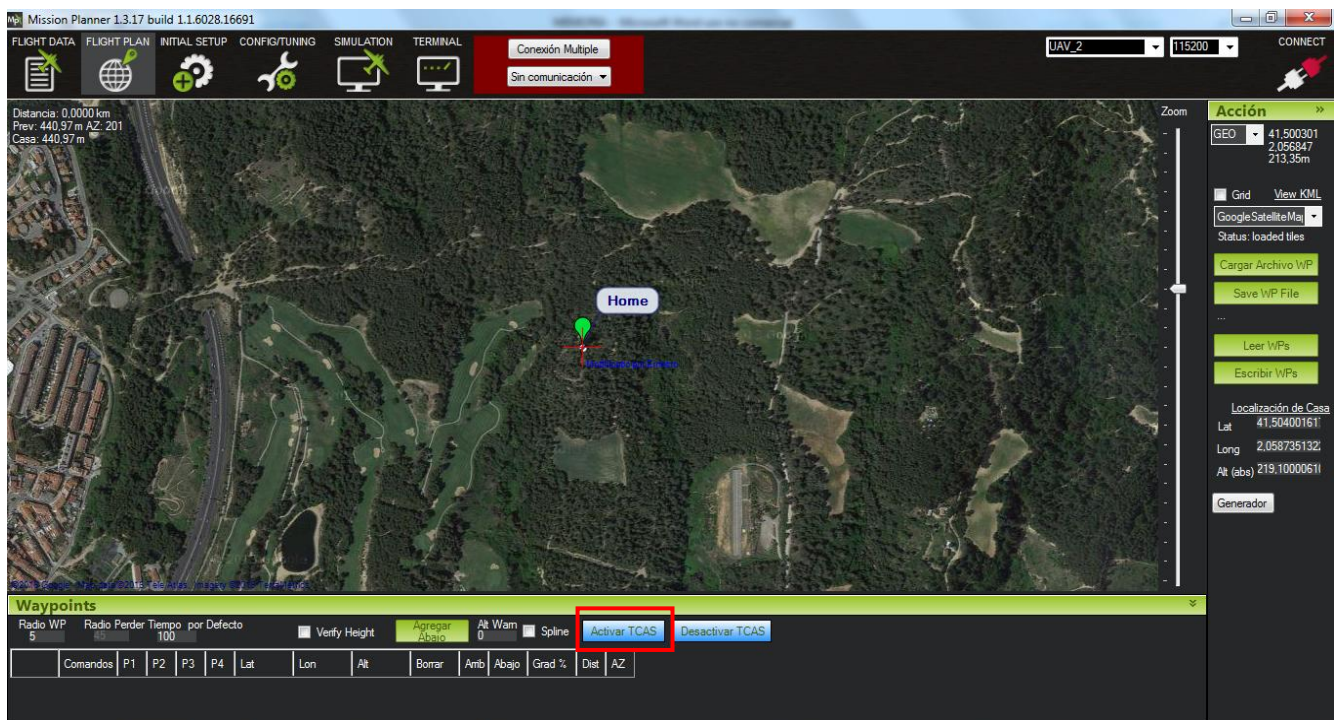


Fig. 4.2.1. Creación del nuevo botón para activar el sistema TCAS II para UAVs.

Nos muestra un mensaje conforme se ha activado el sistema y activa un timer que estará constantemente calculando las distancias entre los dos UAVs. Este timer se encargará de calcular las distancias y de hacer saltar el primer aviso, el Traffic Advisory. El código del timer que controla el primer rango es el siguiente:

```
private void timer2_Tick(object sender, EventArgs e)
{
    PointLatLngAlt position1 = new PointLatLngAlt();
    PointLatLngAlt position2 = new PointLatLngAlt();

    if (timer2.Enabled == true)
    {
        foreach (var port in MainV2.Comports)
        {
            if (port.ToString() == ("UAV_2"))
            {
                MainV2.comPort = port;
            }
        }
        position1.Lat = MainV2.comPort.MAV.cs.lat;
        position1.Lng = MainV2.comPort.MAV.cs.lng;

        foreach (var port in MainV2.Comports)
        {
            if (port.ToString() == ("UAV_1"))
            {
                MainV2.comPort = port;
            }
        }

        position2.Lat = MainV2.comPort.MAV.cs.lat;
        position2.Lng = MainV2.comPort.MAV.cs.lng;
        double x = DistanceTo(position1.Lat, position1.Lng,
            position2.Lat, position2.Lng);
        if (x < 150)
        {
            if (x > 145)
            {
                MessageBoxTemporal.Show("Tu distancia es menor a
                    150 metros. Distancia entre los UAVs :" + x,
                    "TRAFFIC ADVISORY", 2, true);
            }
            timer3.Enabled = true;
        }
    }

    if (MainV2.comPort.MAV.cs.wp_dist == 0 &&
        MainV2.comPort.MAV.cs.alt != 0)
    {
        MainV2.comPort.setMode("Auto");
        timer3.Enabled = false;
    }
}
```

Como podemos apreciar, dentro del timer podemos encontrar que llama a la función "DistanceTo", la cual es una función nueva añadida para hacer el cálculo de las distancias entre UAVs. El código de este cálculo es el siguiente:

```
public static double DistanceTo(double lat1, double lon1, double lat2, double lon2)
{
    double rlat1 = Math.PI * lat1 / 180;
    double rlat2 = Math.PI * lat2 / 180;
    double theta = lon1 - lon2;
    double rtheta = Math.PI * theta / 180;
    double dist =
        Math.Sin(rlat1) * Math.Sin(rlat2) + Math.Cos(rlat1) *
        Math.Cos(rlat2) * Math.Cos(rtheta);
    dist = Math.Acos(dist);
    dist = dist * 180 / Math.PI;
    dist = dist * 60 * 1.1515;
    dist = dist * 1.609344;
    return dist * 1000;
}
```

Una vez explicado el primer timer, podemos pasar al segundo timer, el cual se activa cuando entramos dentro del primer rango del Traffic Advisory. Podemos encontrar este segundo timer con el nombre de timer3 en el anterior código y ver cómo éste se activa.

El código del segundo timer, el cual calcula las distancias en el segundo rango Resolution Advisory y actúa efectuando la maniobra de evasión si es necesario, es el siguiente:

```
private void timer3_Tick(object sender, EventArgs e)
{
    PointLatLngAlt position1 = new PointLatLngAlt();
    PointLatLngAlt position2 = new PointLatLngAlt();
    double altitud1;
    bool pr = true;

    foreach (var port in MainV2.Comports)
    {
        if (port.ToString() == ("UAV_2"))
        {
            MainV2.comPort = port;
        }
    }
    position1.Lat = MainV2.comPort.MAV.cs.lat;
    position1.Lng = MainV2.comPort.MAV.cs.lng;
    position1.Alt = MainV2.comPort.MAV.cs.alt;

    foreach (var port in MainV2.Comports)
    {
        if (port.ToString() == ("UAV_1"))
        {

```

```

        MainV2.comPort = port;
    }
}

position2.Lat = MainV2.comPort.MAV.cs.lat;
position2.Lng = MainV2.comPort.MAV.cs.lng;
position2.Alt = MainV2.comPort.MAV.cs.alt;

altitud1 = position1.Alt - position2.Alt;
if (altitud1 < 0) {
    altitud1 = altitud1 * -1;
}

double x = DistanceTo(position1.Lat, position1.Lng,
position2.Lat, position2.Lng);
if (x < 70)
{
    bool a = false;
    if (altitud1 < 5)
    {
        if (x > 60)
        {
            a = true;
        }

        timer2.Enabled = false;

        Locationwp evasion = new Locationwp();

foreach (var port in MainV2.Comports)
{
    if (port.ToString() == ("UAV_1"))
    {
        MainV2.comPort = port;
    }
}

if (a == true)
{
    MainV2.comPort.setMode("GUIDED");
    if (position1.Alt <= position2.Alt)
    {
        evasion.alt = MainV2.comPort.MAV.cs.alt + 10;
        evasion.lat = MainV2.comPort.MAV.cs.lat + 0.0005;
        evasion.lng = MainV2.comPort.MAV.cs.lng + 0.0001;
        MainV2.comPort.setGuidedModeWP(evasion);
        a = false;
    }
    if (position1.Alt > position2.Alt)
    {
        if (position2.Alt <= 15)
        {
            evasion.alt = MainV2.comPort.MAV.cs.alt + 10;
            evasion.lat = MainV2.comPort.MAV.cs.lat + 0.0005;
            evasion.lng = MainV2.comPort.MAV.cs.lng + 0.0001;
            MainV2.comPort.setGuidedModeWP(evasion);
            a = false;
        }
        if (position2.Alt > 15)

```



```

    {
        evasion.alt = MainV2.comPort.MAV.cs.alt - 10;
        evasion.lat = MainV2.comPort.MAV.cs.lat + 0.0005;
        evasion.lng = MainV2.comPort.MAV.cs.lng + 0.0001;
        MainV2.comPort.setGuidedModeWP(evasion);
        a = false;
    }
}
}

if (MainV2.comPort.MAV.cs.wp_dist == 0 &&
    MainV2.comPort.MAV.cs.alt != 0)
{
    MainV2.comPort.setMode("Auto");
    timer2.Enabled = true;
    timer3.Enabled = false;
}
}
}

```

Seguidamente, cuando todo este proceso está finalizado, encontramos el nuevo botón añadido de "Desactivar el TCAS II", el código del cual es el siguiente:

```

private void Desactivar_TCAS(object sender, EventArgs e)
{
    CustomMessageBox.Show("TCAS Desactivado");
    timer2.Enabled = false;
    timer3.Enabled = false;
}

```

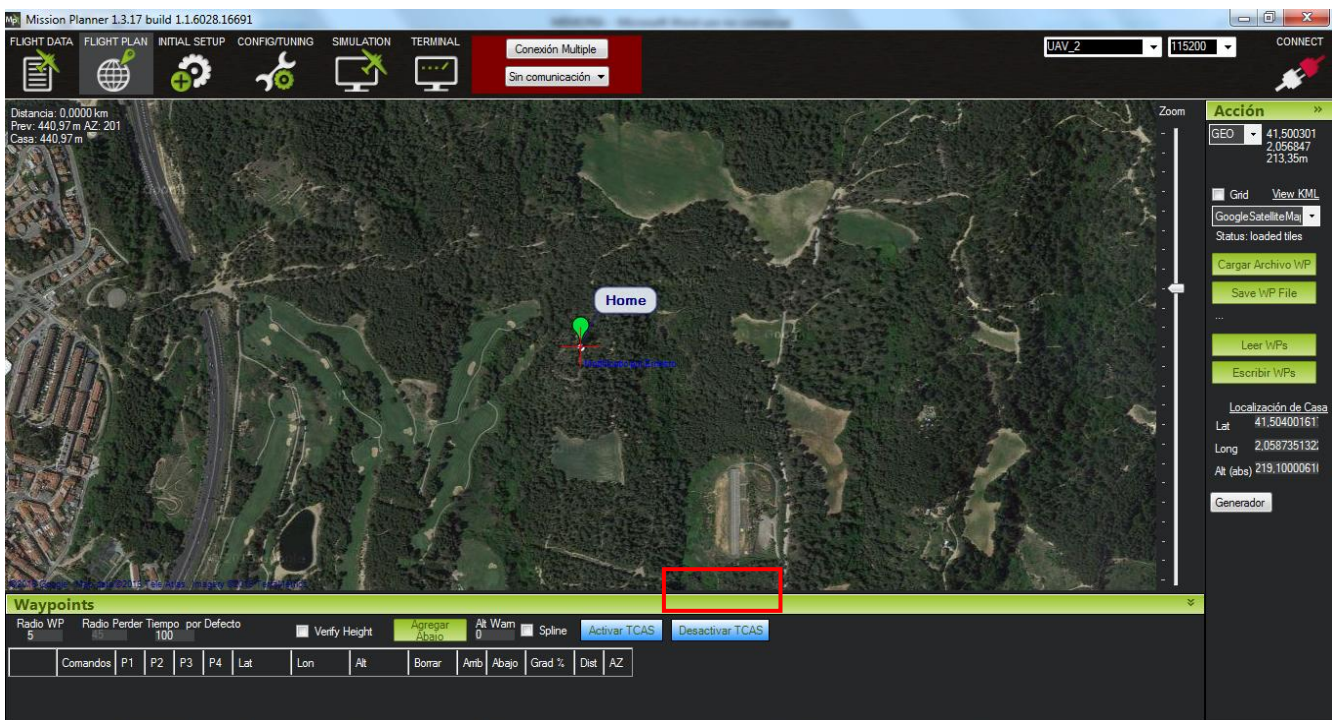


Fig. 4.2.2. Creación del nuevo botón para desactivar el sistema TCAS II para UAVs.

En el podemos ver cómo nos muestra un mensaje de sistema desactivado y cómo desactiva los dos timers para que no continúen calculando las distancias entre aeronaves ni salte ningún aviso.

Para una mayor comodidad a la hora de armar y desarmar los UAVs, como ellos son independientes entre ellos, se ha creado dos funciones que nos permiten armar y empezar la ruta o desarmar y aterrizar automáticamente para cada UAV.

```
private void myButton4_Click(object sender, EventArgs e)
{
    foreach (var port in MainV2.Comports)
    {
        if (port.ToString() == ("UAV_1"))
        {
            MainV2.comPort = port;
        }
        MainV2.comPort.setMode("Stabilize");
        MainV2.comPort.doARM(!MainV2.comPort.MAV.cs.armed);
        MainV2.comPort.setMode("Auto");
    }
}

private void myButton5_Click(object sender, EventArgs e)
{
    foreach (var port in MainV2.Comports)
    {
        if (port.ToString() == ("UAV_2"))
        {
            MainV2.comPort = port;
        }
        MainV2.comPort.setMode("Stabilize");
        MainV2.comPort.doARM(!MainV2.comPort.MAV.cs.armed);
        MainV2.comPort.setMode("Auto");
    }
}
```




Fig.4.2.3. Creación de botones para el despegue y aterrizaje de los UAVs.

Por último, añadir el efecto del mensaje del rango Traffic Advisory, el cual es un estilo de mensaje temporal también añadido al código para que sea un mensaje que aparezca y desaparezca de forma automática a diferencia de los que vienen predeterminados en el programa. El código de este estilo de mensaje es el siguiente:

*Este código ha sido extraído de una página oficial del código de programación C#.

```
public class MessageBoxTemporal
{
    System.Threading.Timer IntervaloTiempo;
    string TituloMessageBox;
    string TextoMessageBox;
    double TiempoMaximo;
    IntPtr hndLabel = IntPtr.Zero;
    bool MostrarContador;

    MessageBoxTemporal(string texto, string titulo, double tiempo,
        bool contador)
    {
        TituloMessageBox = titulo;
        TiempoMaximo = tiempo;
        TextoMessageBox = texto;
        MostrarContador = contador;

        if (TiempoMaximo > 99) return; //Máximo 99 segundos
        IntervaloTiempo = new
            System.Threading.Timer(EjecutaCada1Segundo,
                null, 1000, 1000);
        if (contador)
        {
            DialogResult ResultadoMensaje = MessageBox.Show(texto +
                TiempoMaximo.ToString("00"), titulo);
            if (ResultadoMensaje == DialogResult.OK)
                IntervaloTiempo.Dispose();
        }
    }
}
```

```

    }
    else
    {
        DialogResult ResultadoMensaje = MessageBox.Show(texto +
        "...", titulo);
        if (ResultadoMensaje == DialogResult.OK)
            IntervaloTiempo.Dispose();
    }
}

public static void Show(string texto, string titulo, double
tiempo, bool contador)
{
    new MessageBoxTemporal(texto, titulo, tiempo, contador);
}

void EjecutaCada1Segundo(object state)
{
    TiempoMaximo--;
    if (TiempoMaximo <= 0)
    {
        IntPtr hndMBox = FindWindow(null, TituloMessageBox);
        if (hndMBox != IntPtr.Zero)
        {
            SendMessage(hndMBox, WM_CLOSE, IntPtr.Zero, IntPtr.Zero);
            IntervaloTiempo.Dispose();
        }
    }
    else if (MostrarContador)
    {
        // Ha pasado un intervalo de 1 seg:
        if (hndLabel != IntPtr.Zero)
        {
            SetWindowText(hndLabel, TextoMessageBox +
            "\r\nEste mensaje se cerrará dentro de " +
            TiempoMaximo.ToString("00") + " segundos");
        }
        else
        {
            IntPtr hndMBox = FindWindow(null, TituloMessageBox);
            if (hndMBox != IntPtr.Zero)
            {
                // Ha encontrado el MessageBox, busca ahora el
                texto
                hndLabel = FindWindowEx(hndMBox, IntPtr.Zero,
                "Static", null);
                if (hndLabel != IntPtr.Zero)
                {
                    // Ha encontrado el texto porque el
                    MessageBox
                    // solo tiene un control "Static".
                    SetWindowText(hndLabel, TextoMessageBox +
                    "\r\nEste mensaje se cerrará dentro de " +
                    TiempoMaximo.ToString("00")+"segundos");
                }
            }
        }
    }
}

const int WM_CLOSE = 0x0010;

```

```

[System.Runtime.InteropServices.DllImport("user32.dll",
SetLastError = true)]
static extern IntPtr FindWindow(string lpClassName, string
lpWindowName);
[System.Runtime.InteropServices.DllImport("user32.dll",
CharSet = System.Runtime.InteropServices.CharSet.Auto)]
static extern IntPtr SendMessage(IntPtr hWnd, UInt32 Msg, IntPtr
wParam, IntPtr lParam);
[System.Runtime.InteropServices.DllImport("user32.dll",
SetLastError = true,
CharSet = System.Runtime.InteropServices.CharSet.Auto)]
static extern IntPtr FindWindowEx(IntPtr hwndParent, IntPtr
hwndChildAfter,
string lpszClass, string lpszWindow);
[System.Runtime.InteropServices.DllImport("user32.dll",
SetLastError = true,
CharSet = System.Runtime.InteropServices.CharSet.Auto)]
static extern bool SetWindowText(IntPtr hWnd, string lpString);

```

4.3. Resultados de simulación.

Para iniciar la implementación, en primer lugar, debemos establecer la conexión con los dos UAVs multirrotores de la máquina virtual. Este proceso es gracias a una modificación del código que permite al Mission Planner tener el control de más de 1 UAV. Esta parte del código, que no formaba parte de los objetivos del proyecto, ha sido desarrollada previamente en el Laboratorio de UAVs de la UAB¹ por Ernesto Santana, tutor del proyecto.

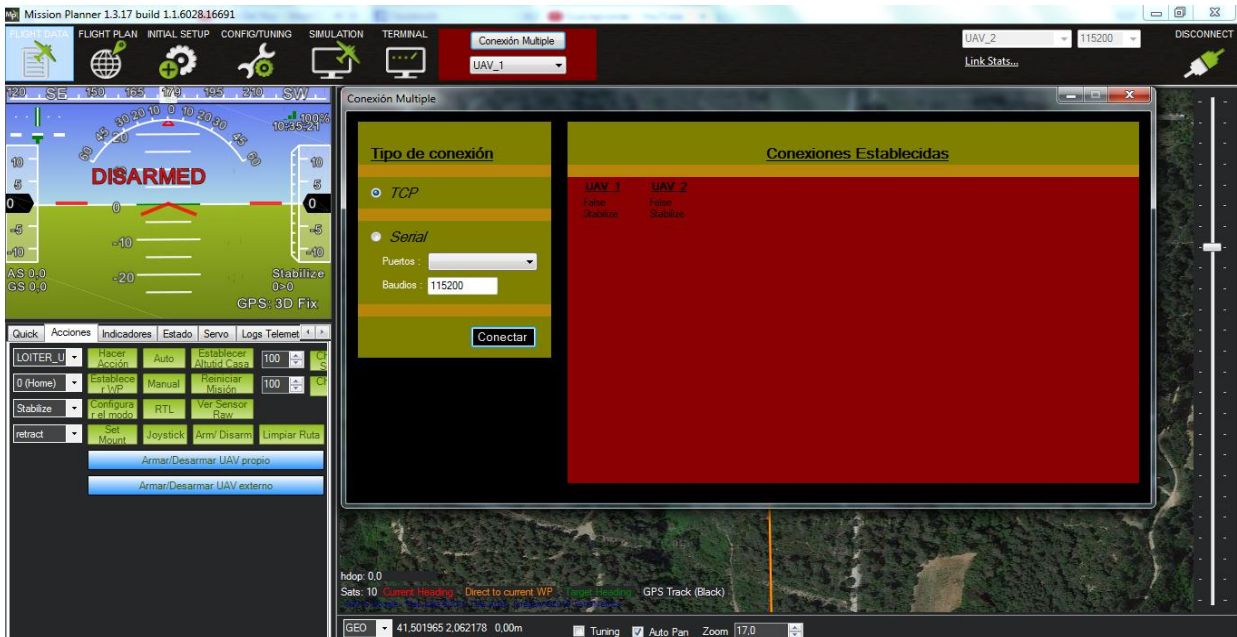


Fig. 4.3.1. Ejecución de conexión entre diferentes UAVs en el software Mission Planner.

Una vez realizado el paso anterior, para efectuar una simulación de funcionamiento del sistema TCAS II desarrollado, lo primero que debemos hacer es establecer una ruta para cada UAV multirrotor; una ruta establecida de tal forma que sepamos a priori que generará una situación de conflicto entre ambos UAVs que hará que el TCAS II se active.

En esta simulación hemos establecido unas rutas que nos permitirán observar la respuesta del sistema TCAS II frente a los siguientes 3 casos:

- El UAV, con una altura de crucero de 22 metros, sobrevuela 2 metros por encima al UAV en conflicto, el cuál vuela a 20 metros. Ello generará un desplazamiento vertical de ascenso de 10 metros y un movimiento horizontal de 60 metros para alejarse del rango RA.
- El UAV, con una altura de crucero de 18 metros, vuela 2 metros por debajo al UAV en conflicto, el cuál vuela a 20 metros. Ello generará un desplazamiento vertical de descenso de 10 metros y un movimiento horizontal de 60 metros para alejarse del rango RA.

¹ Laboratorio de UAVs de la UAB: Unitat de Logística i Aeronàutica. Departament de Telecomunicació i d'Enginyeria de Sistemes

- El UAV, con una altura de crucero de 10 metros, vuela 2 metros por debajo al UAV en conflicto, el cuál vuela a 12 metros. Ello generará un desplazamiento vertical de ascenso de 10 metros en vez de descenso, debido a que vuela a menos de 15 metros de altura, y un movimiento horizontal de 60 metros para alejarse del rango RA.

*Ruta UAV_1 piloto.

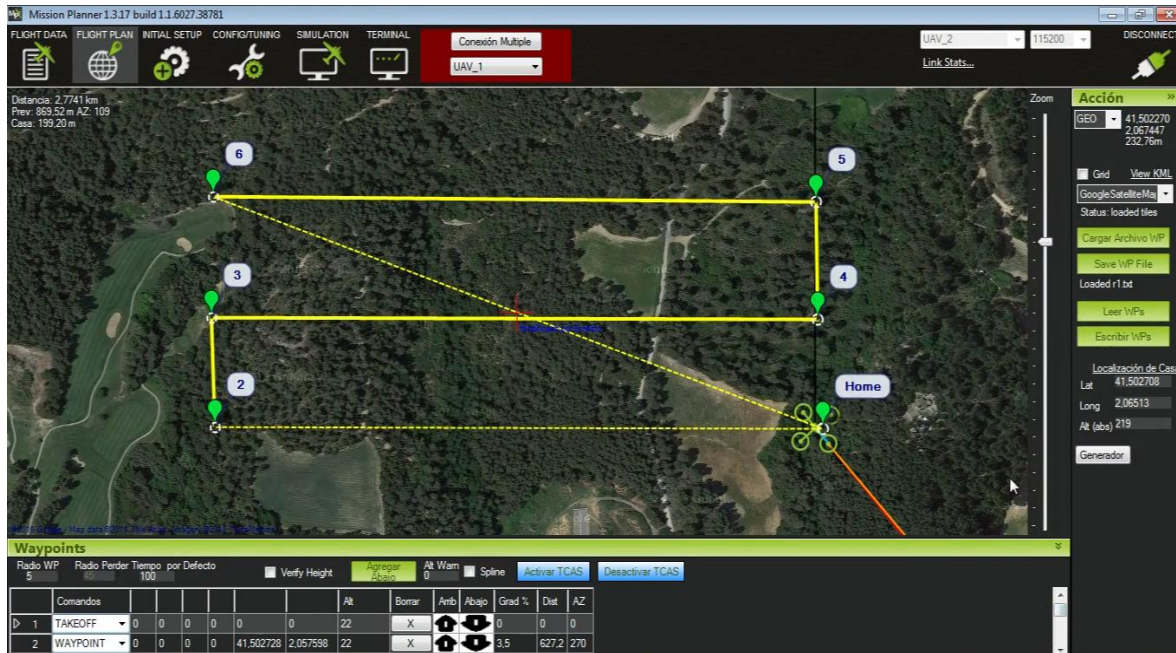


Figura 4.3.2. Configuración de la planificación de vuelo para el UAV_1.

Posición (Latitud, Longitud)	Altura (metros)
Home (41,5027095 , 2,0650649)	22
2 (41,5027280 , 2,0575976)	22
3 (41,5037083 , 2.0575547)	18
4 (41,5036922 , 2,0650649)	18
5 (41,5047448 , 2,0650434)	10
6 (41,5047930 , 2,0575762)	10 + Aterrizaje

Tabla 4.3.3. Tabla con las coordenadas geográficas de la ruta planificada para el UAV_1.

En esta imagen y esta tabla podemos ver la ruta establecida al UAV_1 para que genere una serie de conflictos con diferentes situaciones y así mostrar su funcionalidad.

*Ruta UAV_2 conflicto.

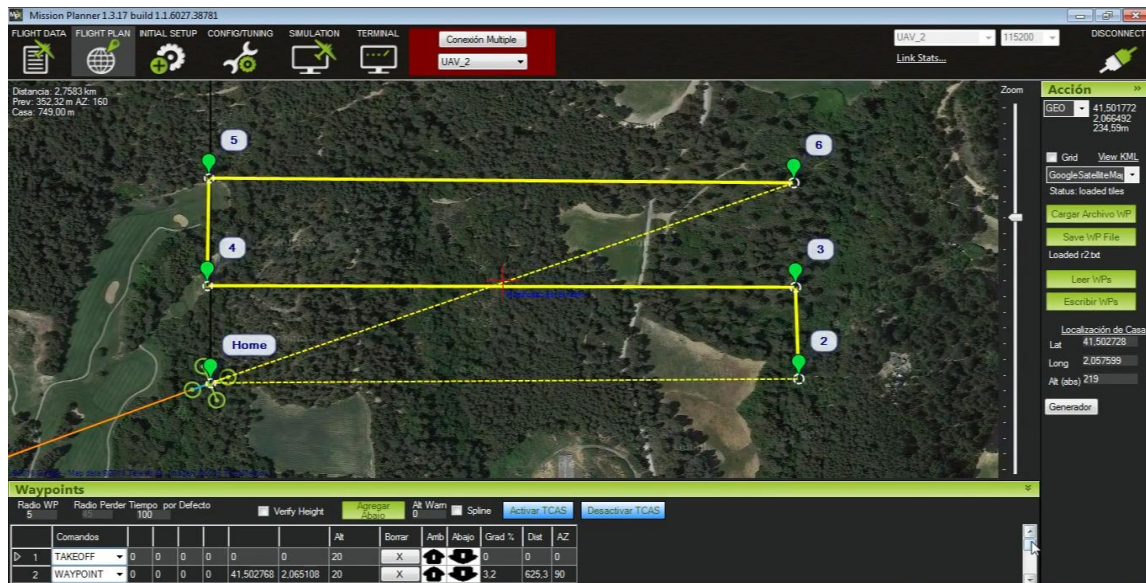


Figura 4.3.4. Configuración de la planificación de vuelo para el UAV_2.

Posición (Latitud, Longitud)	Altura (metros)
Home (41,5027280 , 2,0575976)	20
2 (41,5027095 , 2,0650649)	20
3 (41,5036922 , 2,0650649)	20
4 (41,5037083 , 2.0575547)	20
5 (41,5047930 , 2,0575762)	12
6 (41,5047448 , 2,0650434)	12 + Aterrizaje

Tabla 4.3.5. Tabla con las coordenadas geográficas de la ruta planificada para el UAV_2.

En esta imagen y esta tabla podemos ver la ruta establecida al UAV_2 para que genere una serie de conflictos con diferentes situaciones y así mostrar su funcionalidad. Seguidamente armamos los UAVs para que estos empiecen sus correspondientes rutas.

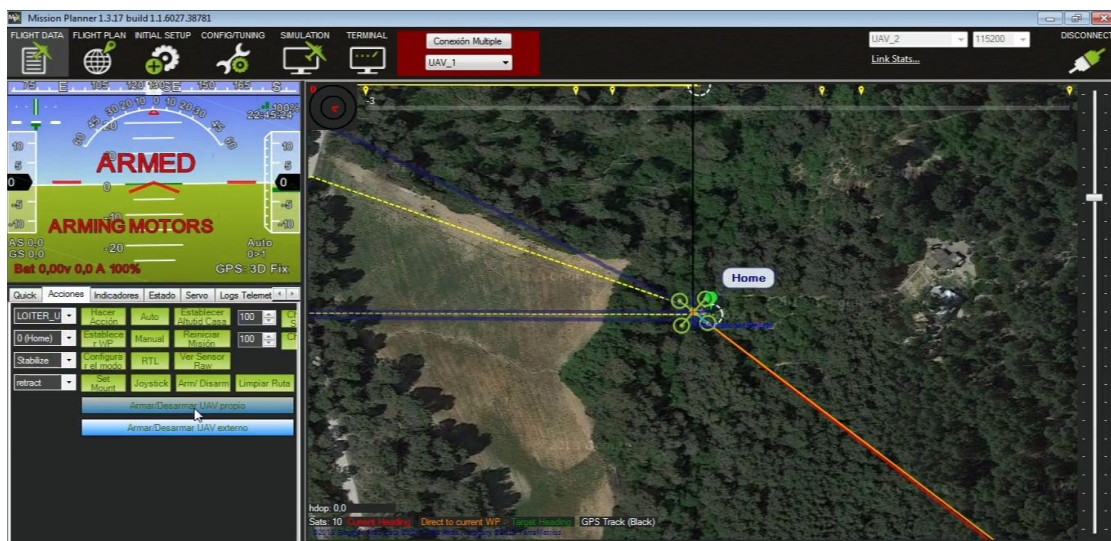


Figura 4.3.6. Despegue de los UAVs.

Mission Planner 1.3.17 build 11.60738781

FLIGHT DATA | FLIGHT PLAN | INITIAL SETUP | CONFIGURATION | SIMULATION | TERMINAL

Conexión: Multiple
UAV_1

UAV_2 | 115200 | DISCONNECT

Link Status...

Distancia: 2.7583 km
Prev: 448.84 m AZ: 222
Casa: 334.79 m

Zoom

Acción

- Grid View KML
- Google Satellite Map
- Status: loaded tiles
- Cargar Archivo WP
- Save WP File
- Loaded r2bt
- Leer WP
- Escribir WP
- Localización de Casa
- Lat: 41.50272389
- Long: 2.065022706
- Alt (abs): 219.1000061
- Generador

TCAS Activado, uso único en UAV_1

Waypoints

Radio WP: 5 | Radio Pender Tiempo: 100 | por Defecto

☐ Verify Height ☐ Agregar Abajo ☐ Alt Warn 0 ☐ Spline ☐ Activar TCAS ☐ Desactivar TCAS

	Comandos						Alt	Borneo	Arriba	Abajo	Grad %	Dist	AZ
D: 1	TAKEOFF	0	0	0	0	0	20	X	0	0	0	0	
2	WAYPOINT	0	0	0	0	41.502768	2.065108	20	X	3.2	625.3	90	

Simulación primer escenario

Mission Planner 1.3.17 build 1.1.6027.38781

FLIGHT DATA FLIGHT PLAN INITIAL SETUP CONFIG/TUNING SIMULATION TERMINAL

UAV_2 115200 DISCONNECT

Link Stats...

Conexión Múltiple
UAV_1

Altitude (m) 21,98
GroundSpeed (m/s) 2,93
Dist to WP (m) 402,00
Yaw (deg) 270,07
Vertical Speed (m/s) 0,00
DistToMAV 217,24

Quick Acciones Indicadores Estado Servo Logs Telem...

hdop: 0.0
Stats: 10 Current Viewport Direct to current WP Target Location GPS Track (Black)

GEO 41.502727 2.062417 21.58m

TRAFFIC ADVISORY

Tu distancia es menor a 150 metros. Distancia entre los UAVs :148,28313816320102

Aceptar

47

Seguidamente podemos ver la reacción del UAV cuando, después del primer aviso, no se ha modificado la ruta y los dos UAVs continúan en dirección uno al otro. Aquí podemos apreciar como el rango Resolution Advisory ha saltado y ha actuado directamente en nuestro UAV multirrotor haciéndole subir 10 metros sobre su altura original de vuelo (a 32 metros ya que volaba a 22 m) y con un desplazamiento horizontal de 60 metros.

*Momento en el que se activa la evasión del conflicto.



Figura 4.3.9. Activación de la evasión.

*Momento en el que se acaba la maniobra de evasión y retoma su ruta.

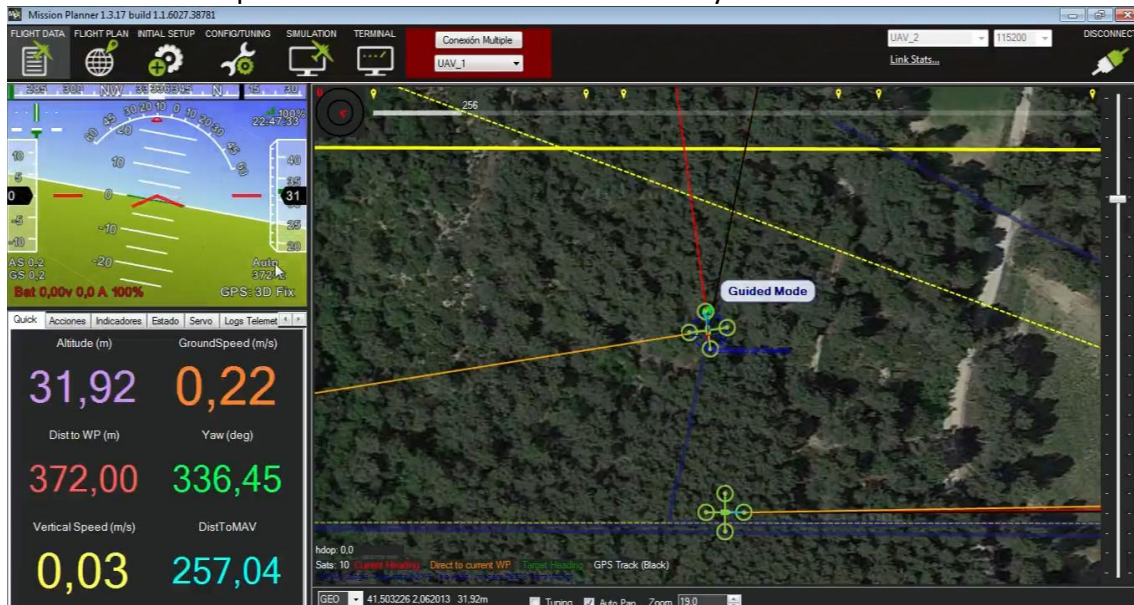


Figura 4.3.10. Finalización de la maniobra de evasión.

Simulación segundo escenario

En el siguiente caso, la altura en crucero del UAV externo es de 20 metros, 2 más que nuestro UAV, el cuál vuela a 18 metros. Con ello se demuestra como nuestro UAV descenderá 10 metros y ejecutara un movimiento horizontal de 60 metros debido a que el UAV externo sobre vuela el nuestro.

*Momento en el que se activa la evasión del conflicto.

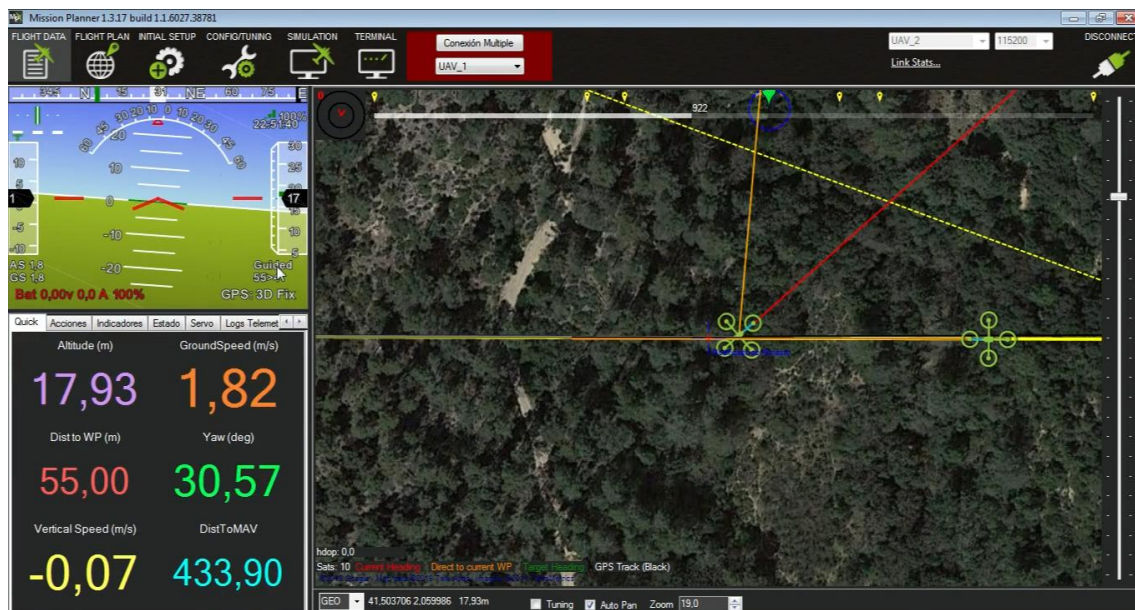


Figura 4.3.11. Activación de la evasión.

*Momento en el que se acaba la maniobra de evasión y retoma su ruta.

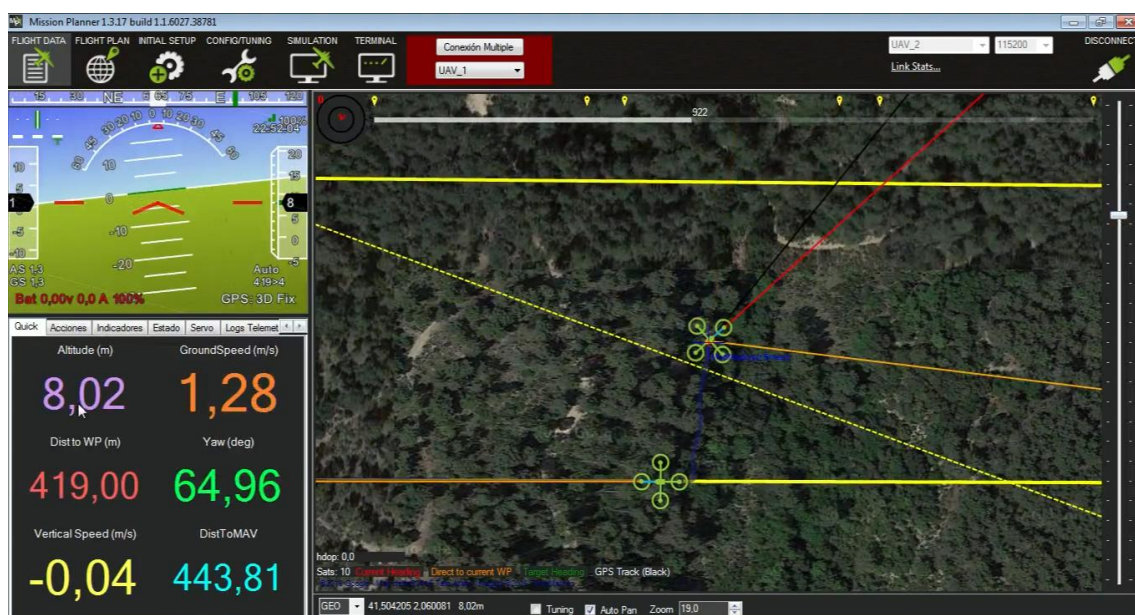


Figura 4.3.12. Finalización de la maniobra de evasión.

Simulación tercer escenario

Por último, podremos ver como nuestro UAV, que vuela a 10 metros, ejecutará una maniobra de evasión debido a que el UAV en conflicto se dirige hacia él a 12 metros. En este caso, todo y volar por debajo, la maniobra de evasión será en ascenso a 10 metros y un desplazamiento horizontal de 60 metros debido a que su actual altura en crucero es menor a 15 metros y descender podría ser peligroso por la baja altura de vuelo.

*Momento en el que se activa la evasión del conflicto.



Figura 4.3.13. Activación de la evasión.

*Momento en el que se acaba la maniobra de evasión y retoma su ruta.

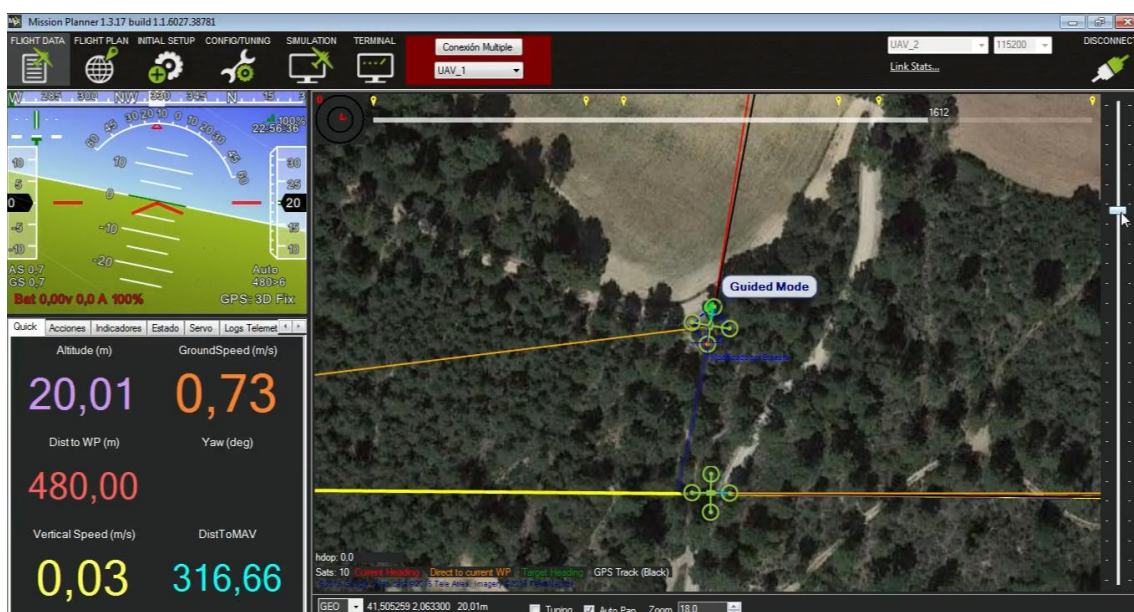


Figura 4.3.14. Finalización de la maniobra de evasión.

Por último, una vez hemos finalizado nuestro vuelo seguro gracias al sistema de seguridad aérea TCAS II para UAVs multirrotores, procedemos a desactivar el sistema TCAS II.

*Momento en el que los dos UAVs han aterrizado en sus puntos correspondientes.

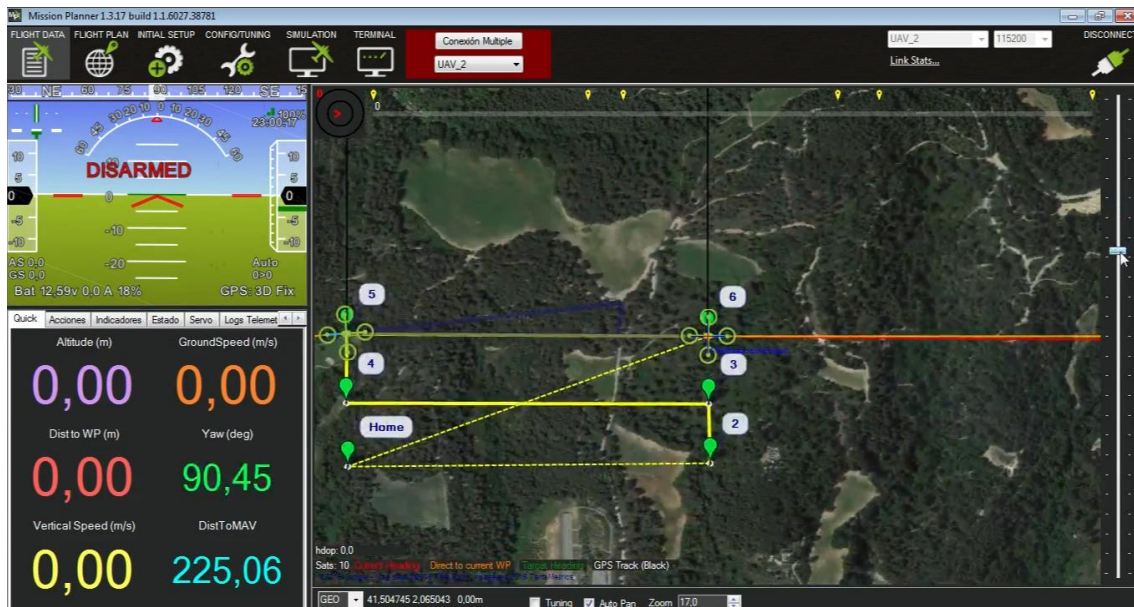


Figura 4.3.15. Finalización de las rutas programadas en los UAVs.

*Desactivación sistema de seguridad aérea TCAS II.

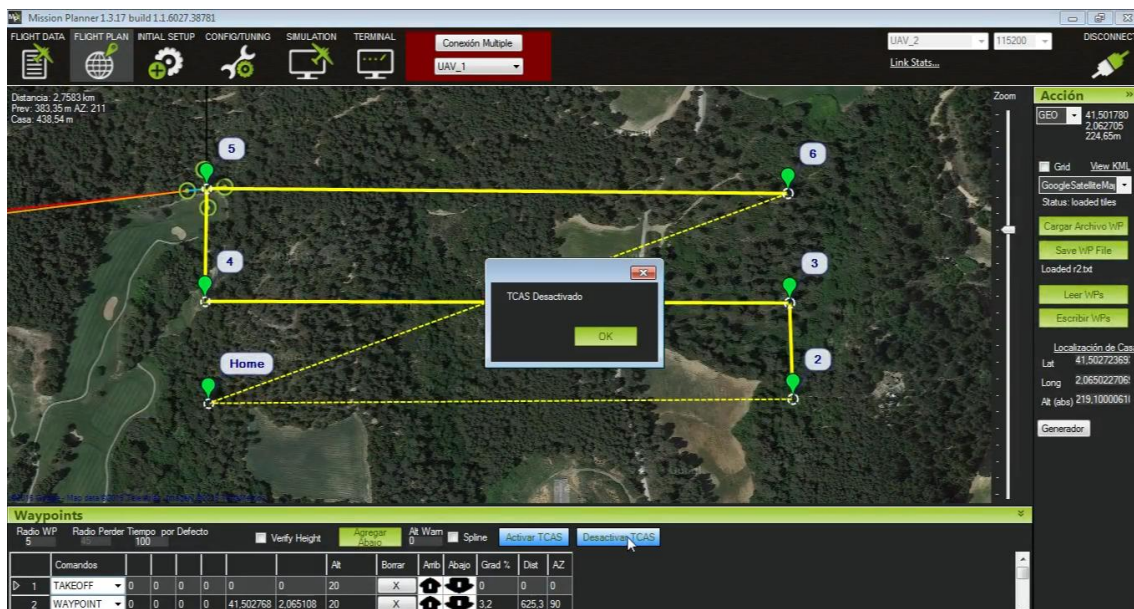


Figura 4.3.16. Desactivación del sistema de seguridad aérea para UAVs TCAS II.

Capítulo 5: Conclusiones

5.1. Objetivos alcanzados.

En el desarrollo del proyecto, hemos podido apreciar la complejidad del sistema de seguridad aérea TCAS II y la gran implicación que este necesita. Es por ello, que algunos de los objetivos planteados en el inicio del proyecto, no han sido alcanzados por falta de tiempo.

Primero de todo, los objetivos del estudio del algoritmo del sistema de seguridad y la adaptación del mismo a los UAVs han sido completados, ya que hemos conseguido comprender todos los procedimientos que éste sigue. Una vez hemos comprendido su funcionamiento, hemos podido hacer una adaptación a los UAVs con éxito, donde todos los parámetros y algoritmos adaptados han funcionado en las pruebas de simulación.

Referente al objetivo de la implementación, todo y que se ha completado la mayor parte, ha faltado mejorar un aspecto en el movimiento horizontal para cumplir el objetivo al completo. El movimiento horizontal actual en el algoritmo es un movimiento básico que ejecuta el UAV, pero en un futuro, se debería mejorar y ejecutar un movimiento horizontal en función de las direcciones de los dos UAVs.

Por último, el objetivo de hacer pruebas con UAV reales no se ha podido llevar a cabo por falta de tiempo, ya que sólo hemos sido capaces de ejecutar el sistema de seguridad aérea TCAS II con éxito a través de un simulador.

5.2. Conclusiones.

A partir del trabajo llevado a cabo en este proyecto sobre la adaptación e implementación del sistema de seguridad aérea TCAS II en UAVs, hemos podido constatar la gran necesidad que tiene el sector de los UAVs de mejorar para poder compartir el espacio aéreo con otras aeronaves. Hemos podido apreciar cómo, desde el aspecto legal, estos están muy limitados todo y tener un gran potencial en el sector aeronáutico.

La importancia del proyecto que se ha desarrollado, una vez estudiada la situación actual de los UAVs, tiene mucha más importancia de la que le reconocimos en el inicio. Este proyecto puede ser el inicio de un gran paso para los UAVs en el sector aeronáutico. A partir de él se consigue la posibilidad de que se reconozca a un UAV como una aeronave segura junto a las demás aeronaves.

Como hemos dicho, los UAVs tienen un gran potencial, y es que, después de desarrollar este sistema en ellos, hemos apreciado el gran abanico de oportunidades de mejora que tienen los UAVs frente a cualquier otra aeronave debido a que es electrónico al completo, incluido la controladora aérea la cual es el corazón del UAV. A partir de ello, el mundo de los UAVs tiene un potencial de crecimiento a un ritmo muy alto, pudiendo así mejorarlos en gran

manera y consiguiendo que éstos estén a un nivel de prestaciones mayores a otras aeronaves.

El mundo de los UAVs multirrotores es relativamente nuevo, eso significa que aún quedan muchas cosas por desarrollar. Con este proyecto tenemos la sensación de que es el inicio de su maduración en el sector y de la gran mejora que estos tendrán en el ámbito de la seguridad de vuelo.

5.3. Futuras implementaciones.

a) Mejora respecto aeronaves. La adaptación e implementación que se ha llevado a cabo en este proyecto está basado en un UAV de referencia y siempre respecto a otro único UAV. Pero uno de los objetivos de este proyecto es que el sistema de seguridad sea válido contra cualquier tipo de aeronave, no solo UAV. Para ello, como futura mejora, sería importante detallar más el código para que nuestro UAV tenga una variación de límites y movimientos de evasión en función de la aeronave que se encontrase, ya sea otro UAV, un avión privado o un avión comercial. Como todos ellos tienen características muy diferentes respecto a velocidades y movimientos, el sistema de seguridad necesita una amplitud de casos en los cuales sea capaz de adaptar su evasión en función de la aeronave externa.

b) Como hemos basado el sistema de seguridad en función de otros UAVs, hemos establecido una evasión principal en el eje vertical ya que es el más importante para evitar la colisión. En el proyecto hay establecido un movimiento horizontal para complementar el movimiento de evasión y mejorarlo, pero sería una futura mejora un estudio de un sistema de evasión horizontal más completo donde se tuviera en cuenta las posiciones exactas y direcciones de las dos aeronaves para optimizar el movimiento horizontal. El porqué de ello es debido a que si nos encontramos dos aeronaves en la misma posición, podemos tener diferentes casos en función de la dirección que tenga. Por ello, una mejora sería un estudio de un algoritmo más completo al actual en el eje horizontal.

c) Una vez hemos conseguido que nuestro nuevo sistema de seguridad aérea funcione en nuestro UAV, continuamos teniendo un impedimento en los UAVs. Este impedimento es que su información de posición sigue siendo única y sólo puede verla la estación base del UAV. Para ello, una mejora sería añadirle un dispositivo que haga la función de transpondedor como por ejemplo en ADS-D modo S para que el UAV pueda transmitir su posición a otras aeronaves y a posibles torres de control.

6. Bibliografía

1. Asmat, J., Rhodes, B., Umansky, J., Villavicencio, C., Yunas, A., Donohue, G., & Lacher, A. (2006, April). UAS safety: Unmanned aerial collision avoidance system (UCAS). In "Systems and Information Engineering Design Symposium, 2006 IEEE" (pp. 43-49). IEEE.
2. Temizer, S., Kochenderfer, M. J., Kaelbling, L. P., Lozano-Pérez, T., & Kuchar, J. K. (2010, August). „Collision avoidance for unmanned aircraft using Markov decision processes.” In AIAA Guidance, Navigation, and Control Conference, Toronto, Canada.
3. Tomblin, P. (2016). "TCAS prevent collision with drones." [online] Quora.com. Available at: <http://www.quora.com/Can-the-Traffic-Collision-Avoidance-System-TCAS-be-used-to-prevent-collisions-with-drones> [Accessed 21 Apr. 2016].
4. User, S. (2016). "CATUAV | UAV Services - 07/09/12 CATUAV develops a TCAS system." [online] Catuav.com. Available at: <http://www.catuav.com/home-en?id=10> [Accessed 21 Apr. 2016].
5. Haffner, P., Sen, S., Spatscheck, O., & Wang, D. (2005, August). ACAS: automated construction of application signatures. In "Proceedings of the 2005 ACM SIGCOMM workshop on Mining network data" (pp. 197-202). ACM.
6. Aviation International News. (2016). "UAVs might inspire new TCAS rules." [online] Available at: <http://www.ainonline.com/aviation-news/aviation-international-news/2006-12-06/uavs-might-inspire-new-tcas-rules> [Accessed 21 Apr. 2016].
7. NASA. (2016). "NASA, FAA, Industry Conduct Initial Sense-and-Avoid Test." [online] Available at: http://www.nasa.gov/centers/armstrong/Features/acas_xu_paves_the_way.html [Accessed 21 Apr. 2016].
8. Martín, L. F. R. *Introducción a los UAV y una visita al Salón de Seguridad y Defensa "hOMSEC13"*.

Firmado: Cristian Sánchez Arias.
Fecha: 05/07/2016.