

# TFG – Arquitectura de computadors i sistemes operatius

**FITA#03:** Preparació d'un entorn  
virtual adequat per a la realització de la  
PoC.

## **Gestió del projecte a GitHub:**

Branca del repositori:

<https://github.com/UOC-Assessments/uoc.tfg.jbericat/tree/FITA%2303>

Dashboard de seguiment de les tasques associades a la fita:

<https://github.com/UOC-Assessments/uoc.tfg.jbericat/projects/9>

**Estudiant:** Jordi Bericat Ruz

**Professor col·laborador:** Daniel Rivas Barragan

**Semestre:** Tardor 2021/22 (Aula 1)

**Versió:** ESBORRANY\_v5

# Índex

3. Preparació d'un entorn virtual adequat per a la realització de la PoC. ....	1
3.1 - Tasques de recerca i investigació.....	1
3.1.1 - Estudi de la API de AirSim per a Python .....	1
3.1.2 - Recerca de projectes basats en AirSim .....	2
3.2 - Preparació dels elements de l'entorn <i>UE “LandscapeMountains Environment”</i> .....	2
3.2.1 - Simulació d'incendis forestals .....	2
3.2.2 - Simulació de condicions nocturnes .....	3
3.3 - Preparació dels actors: Modes de <i>AirSim</i> .....	4
3.3.1 - Mode "Computer Vision" .....	4
3.3.1.1 - Captura aleatòria d'imatges per a generar els jocs de proves .....	5
3.3.1.2 – Simulació de visió tèrmica nocturna tipus FLIR .....	6
3.3.2 - Mode "Multirotor".....	9
3.3.2.1 - Implementació del mode “patrulla” individual (script “drone_patrol.py”) .....	9
3.3.2.2 - Implementació de mode “patrulla” col·lectiu (programa “swarm_patrol.py”).....	10
3.4 – <i>Wrapping-up</i> : Generació de l'entorn final per a la realització de la PoC.....	11
4. Bibliografia .....	20

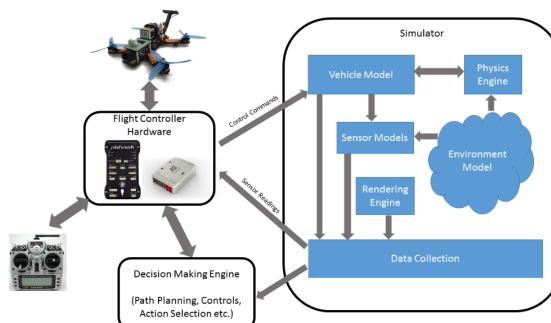
### 3. Preparació d'un entorn virtual adequat per a la realització de la PoC.

#### 3.1 - Tasques de recerca i investigació

##### **3.1.1 - Estudi de la API de AirSim per a Python**

El paper que juga la API de *AirSim* en l'arquitectura del projecte és el de actuar com a interfície entre l'algorisme de DL/VC al cloud (en el cas que ens ocupa, l'entorn Python del PC de desenvolupament fa de cloud) i els *companion computers* dels drons del simulador (Edge Computing / IoT device + sensors device). Un cop revisada la documentació, es comprova que amb les funcionalitats que proporcionen les crides de la API podrem implementar la part dels requeriments de la PoC que correspon a la interacció amb els drons, concretament:

- 1) Podrem efectuar un control programàtic dels drons
- 2) Podrem obtenir les dades necessàries (imatges, posició GPS, dades sensors, etc) de cada dron



<https://www.microsoft.com/en-us/research/wp-content/uploads/2017/02/aerial-informatics-robotics.pdf>

Els següents enllaços proporcionen tota la informació necessària per a la seva correcta utilització (API versió 1.6.0):

- **Informació general al respecte de la API de AirSim**  
<https://microsoft.github.io/AirSim/apis/#airsim-apis>
- **Documentació del paquet “airsim” per a python**  
[https://microsoft.github.io/AirSim/api\\_docs/html/](https://microsoft.github.io/AirSim/api_docs/html/)

### 3.1.2 - Recerca de projectes basats en AirSim

Els següents projectes d'exemple proporcionats juntament amb el codi font de la plataforma *AirSim* poden servir de partida per a implementar el codi de l'apartat 3.3:

- Definició d'un camí o *path* pre-establert mitjançant vectors de velocitat:  
<https://github.com/Microsoft/AirSim/wiki/moveOnPath-demo>
- Implementació de múltiples drons (eixam o *swarm*)  
[https://github.com/Microsoft/AirSim/blob/master/PythonClient/multirotor/multi\\_agent\\_drone.py](https://github.com/Microsoft/AirSim/blob/master/PythonClient/multirotor/multi_agent_drone.py)
- Visió tèrmica nocturna:  
<https://www.microsoft.com/en-us/research/publication/airsim-w-a-simulation-environment-for-wildlife-conservation-with-uavs/>  
<https://microsoft.github.io/AirSim/InfraredCamera/>

## 3.2 - Preparació dels elements de l'entorn UE “LandscapeMountains Environment”

### 3.2.1 - Simulació d'incendis forestals

Recrearem diferents escenaris d'incendi forestals simulats de manera aleatòria utilitzant els components de la llibreria M5VFXvol2 del motor gràfic *Unreal Engine* (veure secció 2.2):

1) Per a desar els canvis, crearem un nou projecte basat en l'entorn que ja hem adaptat a l'apartat 2 i que anomenarem “*TFG-UE-Environment\_v1.uproject*”

```
cd ~/Workspaces/uoc.tfg.jbericat/src/UnrealEnvironments/  
cp -r LandscapeMountains-AirSim-VFX_v2/  
src/UnrealEnvironments/TFG-UE-Environment_v1/  
cd TFG-UE-Environment_v1/  
mv landscapeMountains.uproject TFG-UE-Environment_v1.uproject
```

2) Seguidament, utilitzant l'editor del *Unreal Engine* (UE4Editor), només cal arrossegar des del content browser al viewport tots aquells efectes desitjats del paquet M5VFXVOL2 que hi ha a la carpeta **M5VFXVOL2/Particles/Reference**:

### 3.2.2 - Simulació de condicions nocturnes

Per a recrear un escenari nocturn amb molt baixa / nula visibilitat, s'han hagut de modificar alguns paràmetres de l'entorn UE, així com realitzar algunes modificacions. Al següent enllaç es poden consultar les accions que calen realitzar per a aconseguir els resultats que es mostren més a baix:

<https://www.worldofleveldesign.com/categories/ue4/lighting-night-time-part2-moon-bp-sky.php>





Cal notar que després de realitzar aquest canvi caldrà fer un “**build**” de la il·luminació del projecte abans de generar els binaris finals.

### **3.3 - Preparació dels actors: Modes de AirSim**

#### **3.3.1 - Mode "Computer Vision"**

El mode *Computer Vision* (CV) de *AirSim* ens proporciona una major flexibilitat a l'hora de fer ús de les seves característiques de captura d'imatges, tant de manera programàtica com de manera manual, i sense dependre del moviment de la càmera annexa al multirotor (dron) ni mostrar-lo en les imatges. Així doncs, utilitzarem el mode CV per a prendre les imatges necessàries per a generar els jocs de proves i entrenar la NN.

Per a configurar *AirSim* amb el mode CV, primer caldrà ajustar el seu fitxer de configuració de la següent manera:

```
{  
    "SettingsVersion": 1.2,  
    "CameraDefaults": {  
        "CaptureSettings": [  
            {  
                "ImageType": 0,  
                "Width": 640,  
                "Height": 512,  
                "FOV_Degrees": 90,  
                "AutoExposureSpeed": 100,  
                "MotionBlurAmount": 0  
            },  
            {  
                "ImageType": 7,  
                "Width": 640,  
                "Height": 512,  
                "FOV_Degrees": 90,  
                "AutoExposureSpeed": 100,  
                "MotionBlurAmount": 0  
            }  
        ]  
    },  
    "SimMode": "ComputerVision"  
}
```

Bloc X: Fitxer de configuració ~/Documents/AirSim/settings.json

### 3.3.1.1 - Captura aleatòria d'imatges per a generar els jocs de proves

Fent ús de la API que proporciona *AirSim* per al llenguatge de programació *Python*, s'ha implementat l'script següent per tal de poder realitzar captures aleatòries d'imatges (això és, que no necessàriament pertanyin a incendis forestals) de manera automatitzada:

```
/uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC/computer_vision_mode.py
```

Això ens permetrà ampliar el joc de proves corresponent que es generarà en fites posteriors del projecte.

Un cop executat l'script de captura d'imatges, aquestes restaran desades al directori

```
usr/lib/dataset/buffer/
```

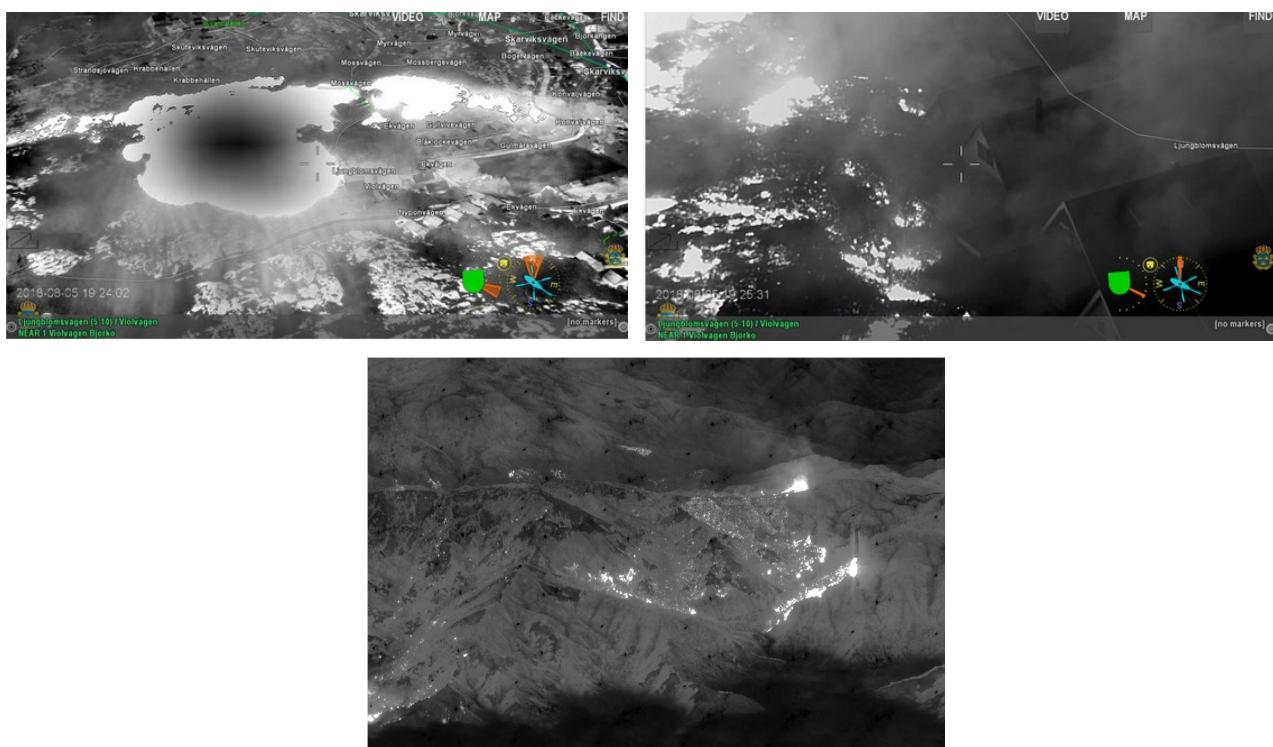
 del repositori. A continuació es mostra un exemple d'execució:

The screenshot shows a terminal window with several tabs at the top: PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL. The TERMINAL tab is active, displaying the command: '(condapy373) jbericat@TFG-UOC:~/Workspaces/uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC\$ python computer\_vision\_mode.py'. The terminal is black with white text, and the command is highlighted in blue.

### 3.3.1.2 – Simulació de visió tèrmica nocturna tipus FLIR

Per tal de recollir les imatges que ens permetran entrenar el model de xarxes neuronals sense haver de reposicionar la càmera de manera manual davant d'un incendi en totes les instàncies d'escenari que es generin durant els jocs de proves, s'utilitzarà la funcionalitat de posicionament automàtic per etiqueta (*tags*) que implementa l'script `capture_ir_segmentation.py` i al qual es fa referència més endavant en aquesta secció.

A més, s'ha decidit modificar el codi font del mode de visió tèrmica nocturna “IR” rudimentari que incorpora *AirSim*<sup>1</sup> per tal de reproduir de la manera més fidel possible la presa i anàlisi d'imatges aèries d'incendis en condicions de nocturnitat, tot tenint en compte les limitacions del simulador. Concretament, després d'efectuar algunes cerques amb els cercadors d'Internet habituals s'ha pogut comprovar que, en situacions reals<sup>2</sup>, la captura d'imatges nocturnes es realitza normalment mitjançant la utilització de càmeres *FLIR* amb tecnologia *MWIR* incorporades als vehicles aeris d'extinció d'incendis. Un exemple d'imatges capturades utilitzant aquesta tecnologia en situacions similars a la de la *PoC* d'aquest projecte són les següents:



**Imatges obtingudes de** [https://www.strategypage.com/military\\_photos/military\\_photos\\_20080805232034.aspx](https://www.strategypage.com/military_photos/military_photos_20080805232034.aspx) i  
<https://www.flir.co.uk/news-center/military/swedish-national-police-use-flir-in-key-fight-against-recent-swedish-wildfire/>

<sup>1</sup> <https://microsoft.github.io/AirSim/InfraredCamera/>

<sup>2</sup> <https://www.flir.co.uk/news-center/military/swedish-national-police-use-flir-in-key-fight-against-recent-swedish-wildfire/>

Concretament, la implementació ha consistit en utilitzar de la implementació de partida de l'script `capture_ir_segmentation.py` per a prendre dues imatges del mateix escenari; una en RGB "estàndard" (que s'ha convertit a B/N) i una altra amb la IR tèrmica *built-in* que inclou *AirSim*, on cada píxel en color blanc (valor "255,255,255" amb codificació RGB) forma part d'un objecte o element de l'entorn UE que genera calor en el rang del que és habitual en un incendi (per a simplificar descartem altres cossos que poden generar calor i que estiguin per sota de 200 graus centígrads / 400 farenheit). Després només ha calgut buscar en quina posició de la matriu de píxels s'ha detectat calor en la imatge IR tèrmica per a seguidament projectar el valor d'aquests píxels a la imatge en color RGB, de manera que se superposen ambdues imatges aconseguint un efecte semblant als de les imatges de referència anteriors:



Només resta afegir que els objectes de UE que emeten calor s'assignen de manera automàtica mitjançant la modificació de l'script `create_ir_segmentation_map.py` associat (que s'ha d'executar primer), així com amb l'assignació de les etiquetes / nom d'objecte adequats a l'entorn LME de UE.

## Consideracions finals:

- Per a simular la visió tèrmica amb IR es podria haver utilitzat directament una imatge en RGB en condicions d'il·luminació convertida a B/N, però en aquests casos no seria possible simular la detecció de calor rere columnes de fum (per posar un exemple concret).
- La mida de les imatges (640x512) simulen les de **I'streaming de vídeo** d'una càmera FLIR comercial específica per a drons<sup>3</sup> i es defineix a l'arxiu de configuració de AirSim settings.json (veure Bloc X, [apartat 3.3.1](#))

El codi relacionat amb aquesta implementació es pot trobar al següent directori del repositori:

```
uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC/create_ir_segmentation.py  
  
uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC/capture_ir_segmentation.py
```

A continuació es mostra un exemple d'execució:

```
(condapy373) jbericat@TFG-UOC:~/Workspaces/uoc.tfg.jbericat/src/AirSim$ /home/jbericat/anaconda3/envs/condapy373/bin/python /home/jbericat/Workspaces/uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC/create_ir_segmentation_map.py  
Connected!  
Client Ver:1 (Min Req: 1), Server Ver:1 (Min Req: 1)  
camera_response.npy not found. Using default response.
```

```
conda activate condapy373  
Connected!  
Client Ver:1 (Min Req: 1), Server Ver:1 (Min Req: 1)  
  
/home/jbericat/Workspaces/uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC/capture_ir_segmentation.py:128: DeprecationWarning: The binary mode of fromstring is deprecated, as it behaves surprisingly on unicode inputs. Use frombuffer instead  
    imgId = numpy.fromstring(responses[0].image_data_uint8, dtype=numpy.uint8)  
/home/jbericat/Workspaces/uoc.tfg.jbericat/src/AirSim/PythonClient/TFG-PoC/capture_ir_segmentation.py:131: DeprecationWarning: The binary mode of fromstring is deprecated, as it behaves surprisingly on unicode inputs. Use frombuffer instead  
    imgIdScene = numpy.fromstring(responses[1].image_data_uint8, dtype=numpy.uint8)  
XYZW: [[ 0.]  
[-10.]  
[ 0.]]  
XYZW derot: [[ 9.20730496]  
[-2.83662799]  
[-2.6793798]]  
Object projected to pixel  
[885.65183344 347.51245759].
```

<sup>3</sup> <https://www.flir.com/products/vue-tz20-r/>

### 3.3.2 - Mode "Multirotor"

El mode multirotor permet activar la simulació de vehicles aeris tipus dron a AirSim, així com el seu control programàtic mitjançant la API proporcionada<sup>4</sup> i es caracteritza per la seva fidel recreació de la física relacionada amb el moviment / desplaçament del dron. Un altre factor a destacar és el gran ventall de sensors i interfícies API que ofereix i que són d'especial ajuda en el desenvolupament d'aplicacions basades en visió per computador i xarxes neuronals.

L'activació del mode multirotor es realitza mitjançant l'arxiu de configuració d'AirSim

`settings.json`:

```
{  
    "SettingsVersion": 1.2,  
    "SimMode": "Multirotor"  
}
```

*Bloc X: Fitxer de configuració ~/Documents/AirSim/settings.json*

#### 3.3.2.1 - Implementació del mode “patrulla” individual (script “drone\_patrol.py”)

Aquest programa de vol pre-establert ens permetrà realitzar “vols de reconeixement” amb un sol dron sense la necessitat d'establir cap tipus de mecanisme de detecció d'obstacles, etc. i ens serà d'ajuda durant el desenvolupament dels jocs de proves que necessitarem durant fites posteriors.

La implementació d'aquesta funcionalitat s'ha obtingut íntegrament d'un altre projecte per a *AirSim* ja existent:

<https://github.com/Microsoft/AirSim/wiki/moveOnPath-demo>

El codi font adaptat amb els paràmetres adequats per al cas d'ús (això és: alçada, vectors de desplaçament i velocitat) es troba al següent path del repositori:

`src/AirSim/PythonClient/TFG-PoC/drone_patrol.py`

Cal notar que la API de *AirSim* no disposa de cap crida per a inicialitzar el dron en unes coordenades determinades. Per tant, per a què el dron segueixi la ruta pensada, s'ha configurat l'actor “*PlayerStart*” de l'entorn UE amb les següents coordenades d'inici:

<sup>4</sup> <https://astrobear.top/2020/01/15/AirSimMultirotorAPIs/>

- X = 46180
- Y = 38280
- Z = 18120

### 3.3.2.2 - Implementació de mode “patrulla” col·lectiu (programa “swarm\_patrol.py”)

TO-DO → Es posposa la implementació d'aquest mode fins al moment en què s'hagi d'utilitzar, ja que només cal modificar `drone_patrol.py` (d'aquesta manera es podran definir millor els requisits).

### 3.4 – Wrapping-up: Generació de l'entorn final per a la realització de la PoC

Ara ja es disposa de tots els elements necessaris per a generar el mapa o entorn final per a realitzar la PdC. Com que la decisió final ha sigut la de utilitzar imatges que simulen visió termal nocturna de tipus FLIR, i la simulació d'aquesta que s'ha implementat (veure [apartat 3.3.1.2](#)) té la limitació de no poder assignar un *blueprint* de calor als efectes VFX de foc de tipus emitter, aleshores s'haurà d'afegir un objecte "base" a cada flama o zona amb foc de tipus StaticMeshActor.

D'altra banda, per a generar l'entorn final de la PoC s'ha hagut de seguir exactament la seqüència que s'indica a continuació, ja que no és possible treballar amb agilitat i estabilitat amb l'editor d'entorns UE4 degut a les limitacions de la CPU de la estació de desenvolupament utilitzada (Intel i5 amb data de fabricació del 2010):

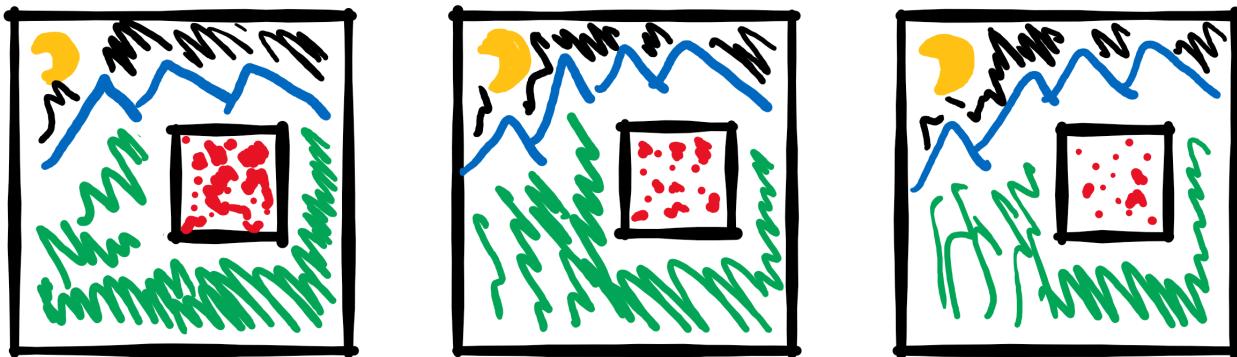
1. Definició de diferents zones clarament delimitades que simulin diferents nivells d'intensitats i/o condicions d'incendi (criteris: alçada flames, fum, extensió, visibilitat, accessibilitat del terreny, etc):

La captura següent mostra una vista aèria de totes les zones seleccionades:



El mapa està pensat, d'una banda, per a què d'un mateix escenari s'hi puguin generar diferents *training datasets*, jocs de proves, experiments, etc. sense haver de tornar a editar l'entorn LME modificat (molt costós en termes de temps) un cop generats els binaris del mateix. A partir d'aquest punt només caldrà executar aquests binaris de la mateixa manera que s'executa la "distribució" estàndard de *AirSim* (sense modificar l'entorn), així com definir mitjançant la seva API (python) i del fitxer de configuració `settings.json` en quines zones i amb quines condicions es realitzarà cada experiment.

De l'altra banda, la configuració dels elements del mapa, també està orientada a què es pugui resoldre el problema de classificar les imatges processades mitjançant tècniques de DL i NN's, en funció de les característiques dels incendis definides a la taula XX més avall. Així doncs, en fites posteriors es definiran amb més detall tant les característiques (*features*) que s'han de reconèixer per tal de realitzar la classificació (p.e. +/- intensitat del foc i +/- extensió de terreny) com les accions que caldrà emprendre en funció del resultat. A mode conceptual però, la idea inicial és la de poder classificar amb una NN diferents tipus d'imatges en funció del mode operacional en el que es trobin els drons (reconeixement, monitoreig, extinció etc). Per exemple; en mode de reconeixement, les imatges IR-termal aèries i **Ilunyanes** d'un incendi de alta, moderada o baixa intensitat i localitzats podrien tenir un aspecte semblant al següents sketches, respectivament:



La següent taula resumeix la posició, extensió i nivells d'intensitat dels incendis forestals simulats delimitats per a cada zona:

Zona	Coordenades				Intensitat	Extensió
	NW	NE	SW	SE		
#1	(-11495, -37985, 14289)	(34616, -47018, 19652)	(-8420, -14050, 17180)	(34253, -17952, 11687)	Alta	Molt estesa
#2	(20597, 30507, 18861)	(31506, 33793, 19799)	(16080, 41923, 20045)	(27889, 45620, 19000)	Moderada	Localitzada
#3	(53784, 5943, 10176)	(57370, 8070, 11310)	(50020, 13850, 10270)	(54040, 15610, 10790)	Baixa	Localitzada
#4	(45220, 29500, 16070)	(59410, 29670, 18200)	(45460, 50040, 19960)	(58970, 50770, 17870)	Baixa	Molt estesa
#5	(96894, 33663, 18410)	(110697, 48665, 17836)	(86339, 37386, 17971)	(82762, 58568, 16764)	Baixa	Moderadament estesa

## 2. Col·locar els objectes tipus `staticMeshActor` que simulen la emissió de calor a cadascuna de les àrees delimitades al punt anterior

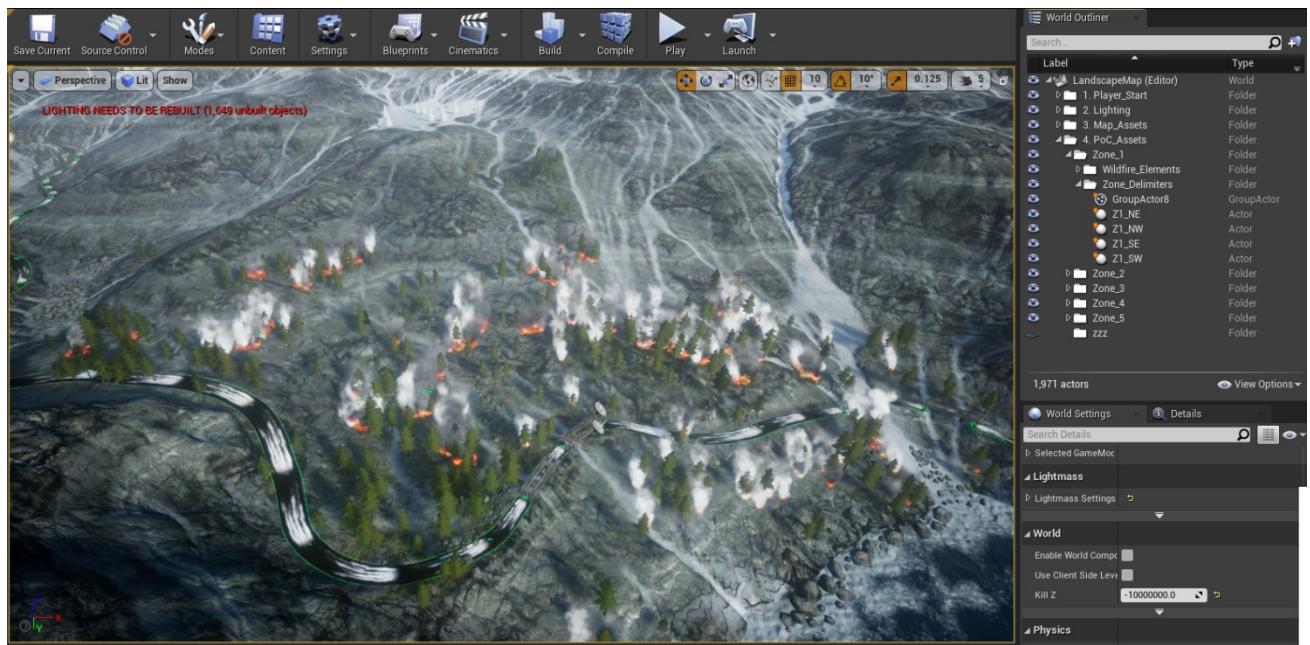
S'utilitzen objectes de la llibreria M5VFXVOL2 utilitzada per a simular els VFX de foc (veure [secció 3.2.1](#)), la qual també inclou alguns `assets` accessoris demostratius. Concretament, s'utilitzen els objectes `BF_mesh_firewood` i `grass_mesh` localitzats a la carpeta de components de l'entorn de UE -> `content/M5VFXVOL2/Props/Meshes/`

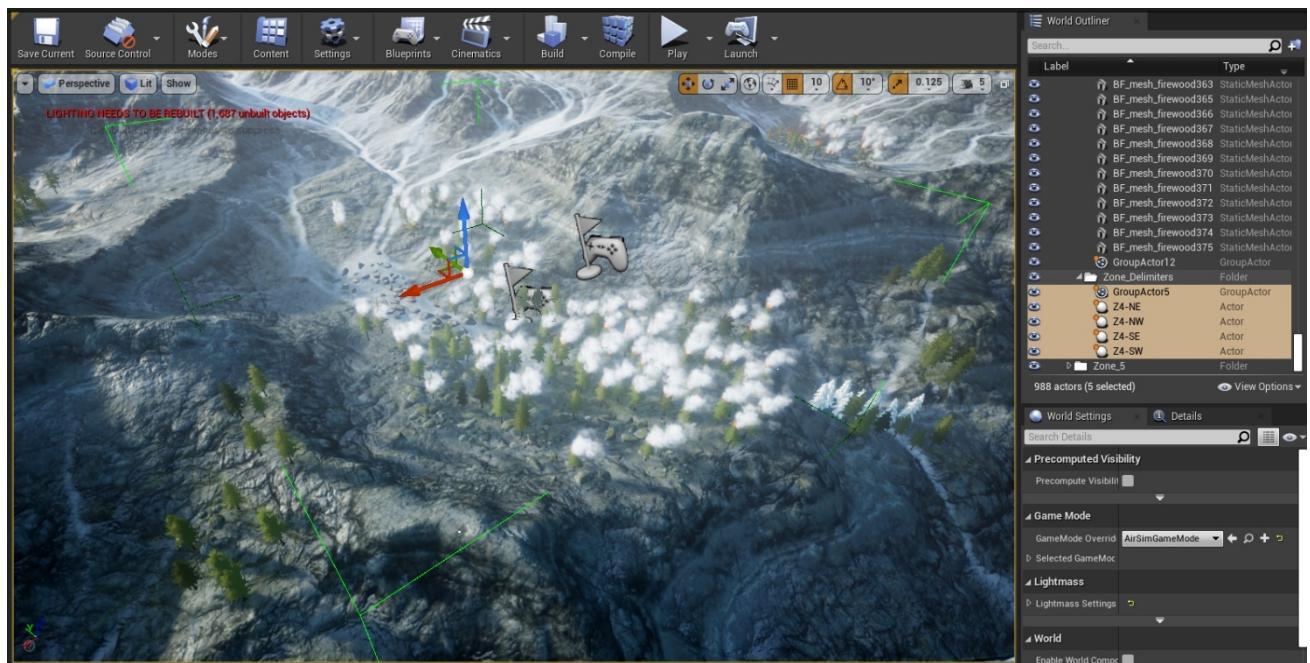
## 3. Desar i reobrir el projecte (num. versió +1)

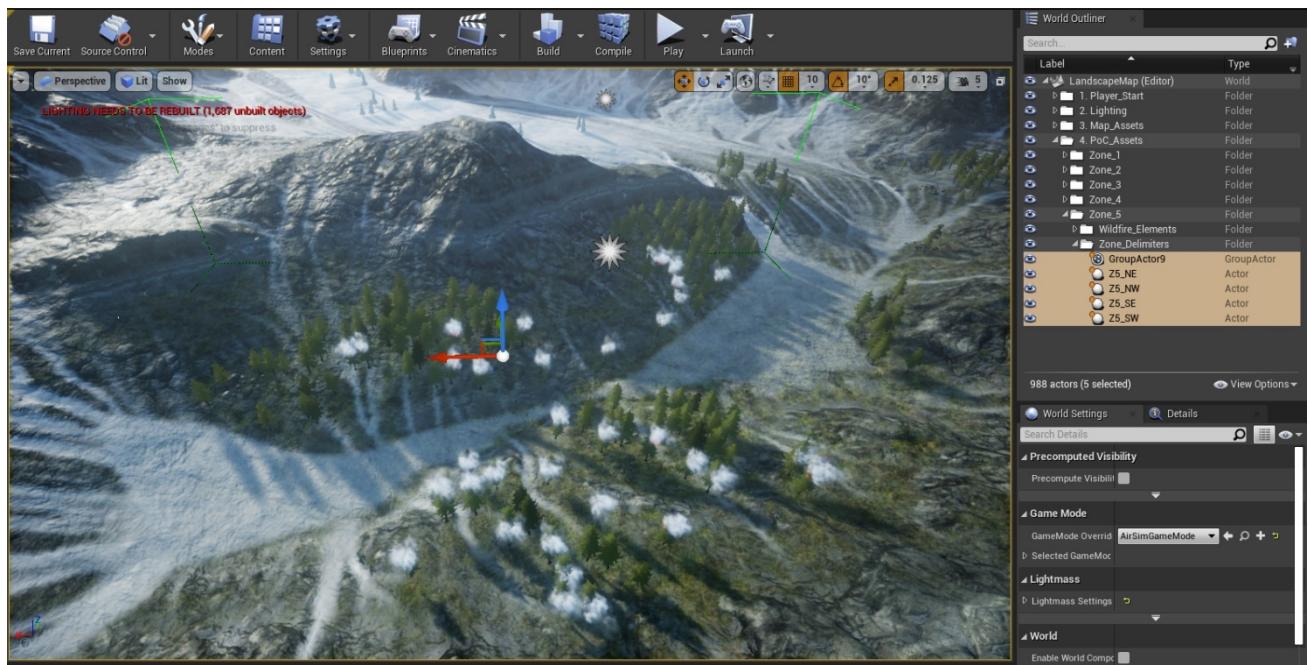
La versió del projecte que inclou totes les modificacions i ampliacions realitzades fins aquest punt es desa fora del repositori (2.4Gb) a la carpeta `/src/UnrealEnvironments/TFG-UE-Environment_v12`

## 4. Afegir els VFX de foc de tipus `emitter SUPERPOSATS` sobre cadascun dels objectes de tipus `StaticMeshActor` afegits al punt 2

Cadascuna de les 5 zones definides al punt 1 resten amb la següent configuració un cop afegits els VFX de simulació d'incendis:

Zona #1:Zona #2:

Zona #3:Zona #4:

Zona #5:

## 5. Desar i reobrir el projecte (num. versió +1)

La versió del projecte que inclou totes les modificacions i ampliacions realitzades fins aquest punt es desa fora del repositori (2.4Gb) a la carpeta /src/UnrealEnvironments/TFG-UE-Environment\_v19

## 6. Establir diferents posicionaments de la càmera per a prendre les imatges (això és alçada i angle, ja que la càmera es mou automàticament a les coordenades on hi ha els StaticMeshActors que simulen emissió de calor). -> Accions: Definir posicionaments, desar en una estructura de dades i "injectar-les" al codi python `capture_ir_segmentation.py` com a input data

Es determina que, conceptualment, aquest punt no correspon fer-lo en aquesta fita del projecte. Sobre l'issue #75 i s'assigna al [dashboard de control de la FITA#04](#)

## 7. Testejar tot el setup AMB il·luminació

La captura dinàmica d'imatges tèrmiques amb FLIR simulat s'efectua correctament en condicions de simulació diürnes:



## 8. Desactivar la il·luminació de l'entorn

Es realitzen les accions definides a la [secció 3.2.2](#) d'aquest document.

## 9. Testejar tot el setup SENSE il·luminació

Es torna a fer la mateixa captura del comentari anterior, però ara es tracta d'una imatge IR tèrmica aèria en [condicions d'il·luminació nocturnes](#):



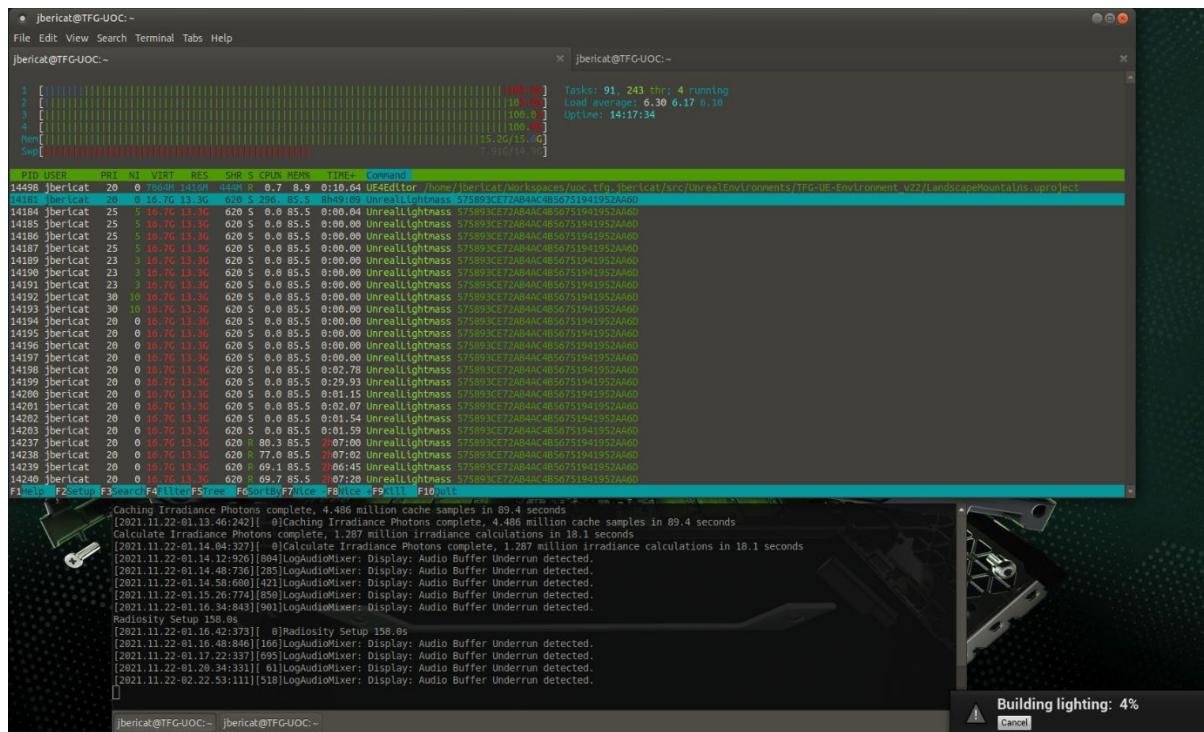
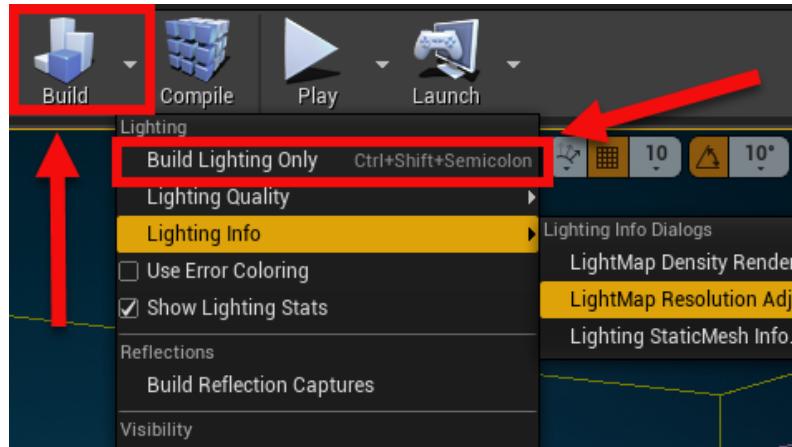
 L'efecte de simulació de IR tèrmica aconseguit s'acosta bastant al que es pretenia inicialment.

## 10. Desar i reobrir el projecte (num. versió +1)

La versió del projecte que inclou totes les modificacions i ampliacions realitzades fins aquest punt es desa fora del repositori (2.4Gb) a la carpeta /src/UnrealEnvironments/TFG-UE-Environment\_v22

## 11. Generar els binaris de l'entorn final UE4 de la PoC

Finalment es procedeix a generar els binaris de l'entorn UE4. Tanmateix, primer cal realitzar un *build* de la il·luminació:



Un cop finalitzat el procés anterior es procedeix a generar els binaris de l'entorn per a la seva execució independent de UE4Editor

**TO-DO: adjuntar captura de la correcta execució dels binaris**

## **4. Bibliografia**

<https://microsoft.github.io/AirSim/design/>

### **Paper**

You can read more about our architecture and design in [our paper \(work in progress\)](#). You may cite this as,

```
@techreport{MSR-TR-2017-9,
    title = {{A}erial {I}nformatics and {R}obotics Platform},
    author = {Shital Shah and Debadeepa Dey and Chris Lovett and Ashish Kapoor},
    year = {2017},
    institution = {Microsoft Research},
    number = {{M}{S}{R}-{T}{R}-2017-9}
}
```

R}-{T}{R}-2017-9}}

<https://microsoft.github.io/AirSim/InfraredCamera/>

<https://www.microsoft.com/en-us/research/publication/airsim-w-a-simulation-environment-for-wildlife-conservation-with-uavs/>

<https://www.flir.co.uk/news-center/military/swedish-national-police-use-flir-in-key-fight-against-recent-swedish-wildfire/>