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-Assignments/uoc.tfg.jbericat/tree/FITA%2303

Dashboard de seguiment de les tasques associades a la fita:
https://github.com/UOC-Assignments/uoc.tfg.jbericat/projects/9

Estudiant: *Jordi Bericat Ruz*

Professor col·laborador: *Daniel Rivas Barragan*

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

Versió: *ESBORRANY_v4*

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 de l'entorn: *UE "LandscapeMountains Environment"* 2
3.2.1 - Simulació d'incendis forestals 2
3.2.2 - Simulació de condicions nocturnes 3
3.3 - Preparació dels actors: Modes de AirSim 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
4. Bibliografia 11

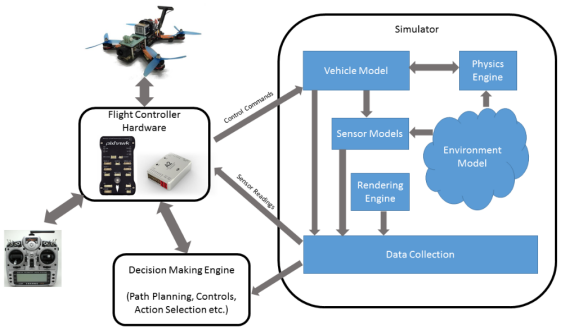
3. Preparació d'un entorn virtual adequat per a la realització de la PoC.

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/

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 multiples drons (eixam o *swarm*) 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 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 Unreal Engine (veure apartat 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 / nul·la 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 fitxer de configuració de *AirSim* 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 a 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
```

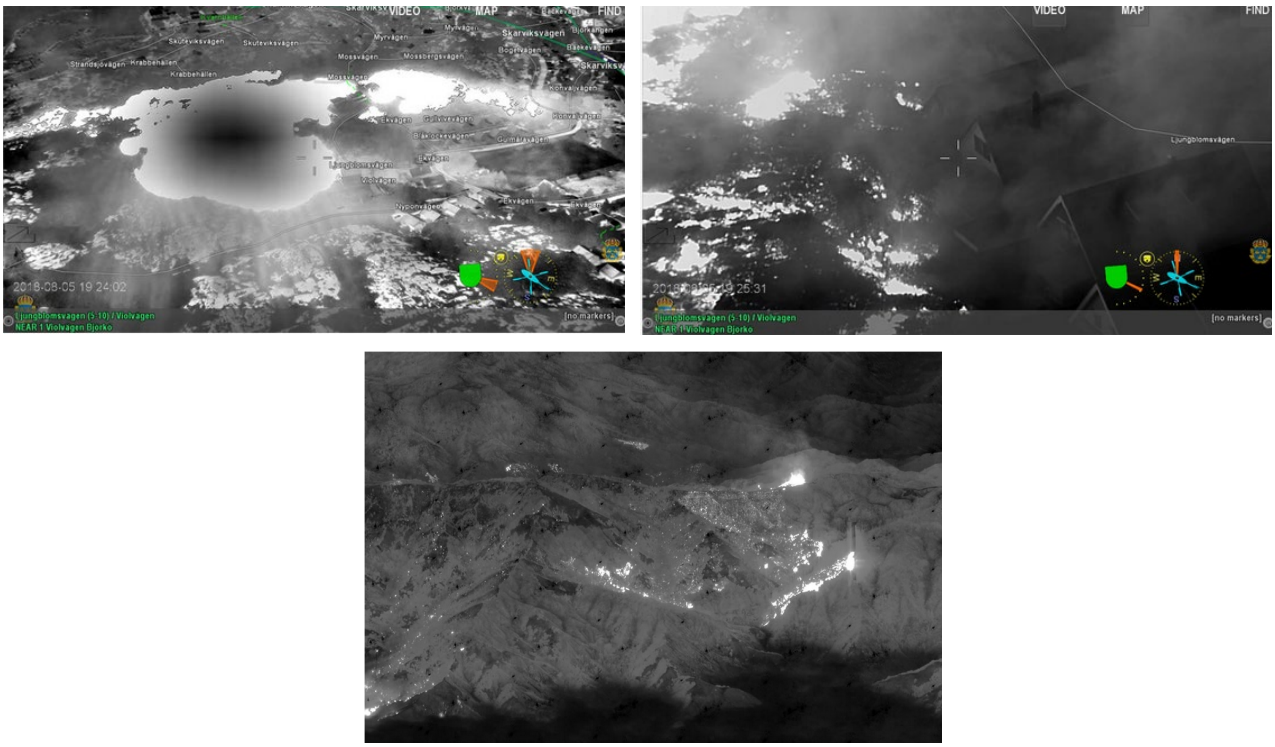
Això ens permetrà ampliar el joc de proves corresponent que es generarà en fites posteriors del projecte.

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:

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*[1] 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 comptes les limitacions del simulador. Concretament, després d'efectuar algunes cerques amb els cercadors d'Internet habituals s'ha pogut comprovar que, en situacions reals[2], 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 **i** https://www.flir.co.uk/news-center/military/swedish-national-police-use-flir-in-key-fight-against-recent-swedish-wildfire/*

[1] https://microsoft.github.io/AirSim/InfraredCamera/

[2] 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 "estandàrd" (que s'ha convertit a B/N) i una altra amb la IR tèrmica "built-in" de 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 referencia anteriors:

(S'HA DE CANVIAR AQUESTA CAPTURA PER UNA ALTRA AÈRIA EN CONDICIONS DE NOCTURNITAT I MÉS MASSA CREMANT)

Nomes 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)
- La mida de les imatges (640x512) simulen les d'una càmera FLIR comercial específica per a drons[3] 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/s
rc/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 deprecat
ed, as it behaves surprisingly on unicode inputs. Use frombuffer instead
  img1d = 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 deprecat
ed, as it behaves surprisingly on unicode inputs. Use frombuffer instead
  img1dscene = numpy.fromstring(responses[1].image_data_uint8, dtype=numpy.uint8)
XYZW: [[  0.]
 [-10.]
 [  0.]]
XYZW derot: [[ 9.20730496]
 [-2.83662799]
 [-2.67937998]]
Object projected to pixel
[885.65183344 347.51245759].
```

[3] 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[4] 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 coordinades 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:

[4] 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).

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 Debadeepta 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}}hello world
```

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/