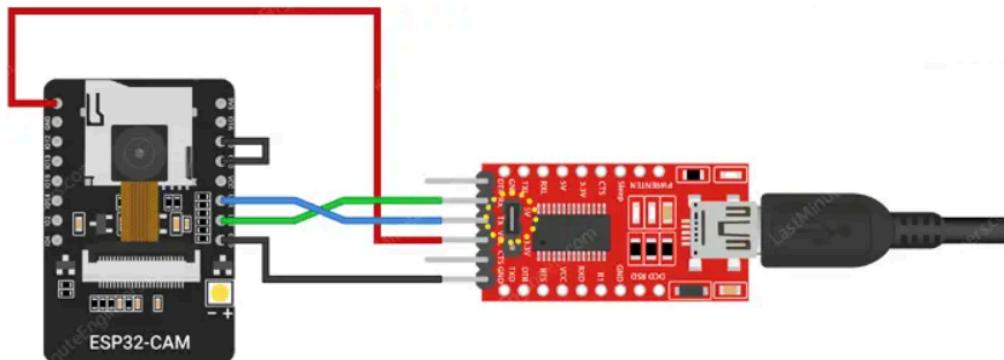


Projet Multivisio

I - Acquisition du flux vidéo avec l'ESP32

Pour commencer, réalisez le montage suivant :



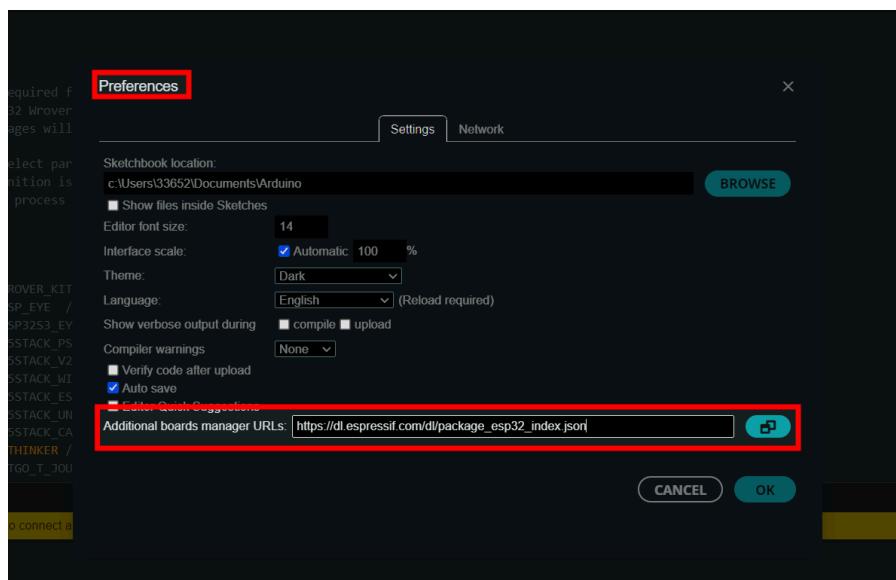
Le convertisseur USB-Série possède un cavalier permettant de sélectionner entre 3,3V et 5V. Puisque nous alimentons l'ESP32-CAM par USB, le cavalier doit être positionné sur 5V.

Pour programmer des cartes ESP32 avec l'IDE Arduino, il est nécessaire d'ajouter une configuration spécifique, car ces microcontrôleurs ne sont pas pris en charge nativement par l'IDE Arduino®. Il faut intégrer une source JSON fournie par le fabricant Espressif Systems.

Pour ce faire, rendez-vous dans : *Fichier -> Preferences*

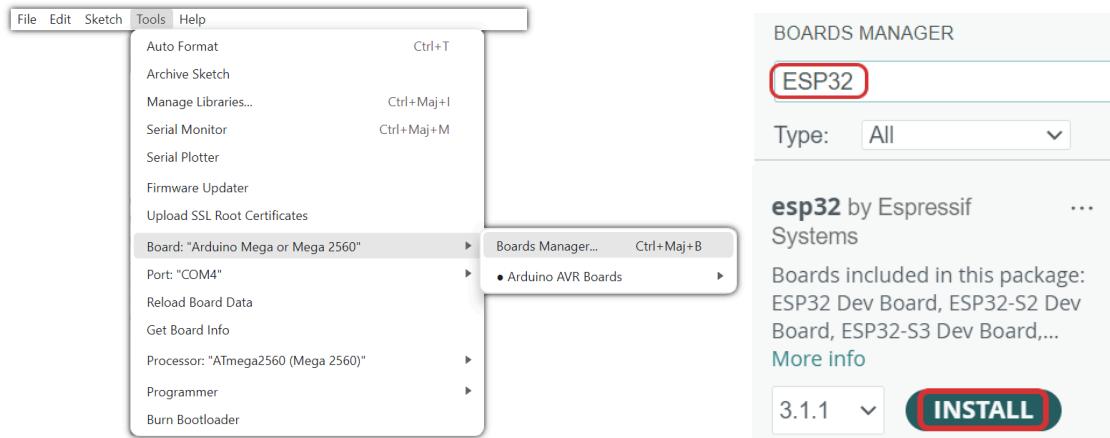
Et coller le lien suivant dans le champ de “Additional boards manager URLs” :

https://dl.espressif.com/dl/package_esp32_index.json

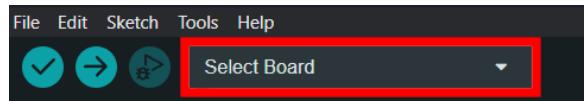


Si le champ "Additional boards manager URLs" n'est pas visible, passez la fenêtre en plein écran et il devrait apparaître.

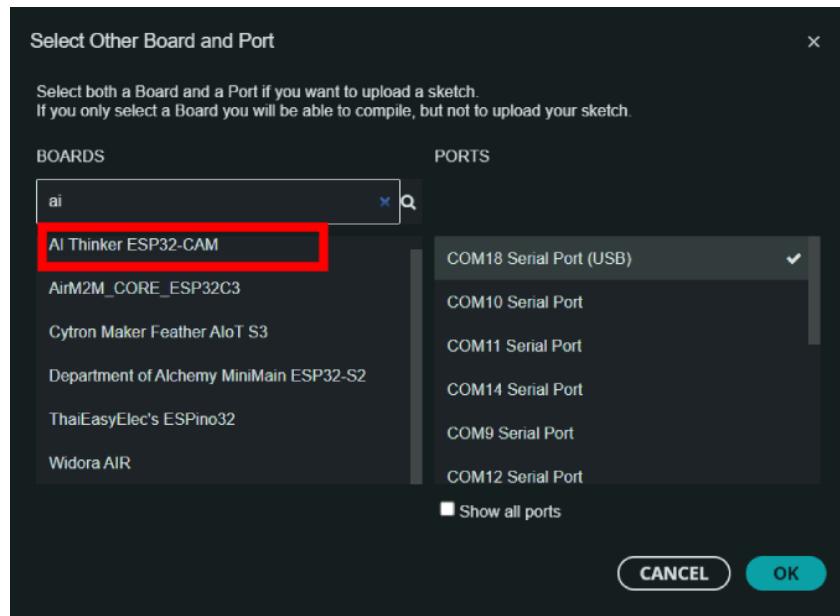
Ensuite, accédez au "Boards Manager", et installez "esp32 by Espressif Systems" :



On peut ensuite connecter la carte au PC à l'aide du câble USB.



Dans le menu "Select Board", choisissez "Select Other Board and Port", puis dans la section BOARDS, sélectionnez "AI Thinker ESP32-CAM".



Il est maintenant temps de charger le programme en C permettant de configurer la carte. Pour cela, allez dans Fichier -> Examples -> ESP32 -> Camera -> CameraWebServer.

Une fois le code chargé, vous devez maintenant spécifier le modèle de caméra utilisé en commentant la ligne `#define CAMERA_MODEL_ESP_EYE` et en décommentant `#define CAMERA_MODEL_AI_THINKER`.

Vous devriez obtenir le code suivant :

```

// Select camera model
// =====
#ifndef CAMERA_MODEL_WROVER_KIT // Has PSRAM
#define CAMERA_MODEL_ESP_EYE // Has PSRAM
#define CAMERA_MODEL_ESP32S3_EYE // Has PSRAM
#define CAMERA_MODEL_M5STACK_PSRAM // Has PSRAM
#define CAMERA_MODEL_M5STACK_V2_PSRAM // M5Camera version B Has PSRAM
#define CAMERA_MODEL_M5STACK_WIDE // Has PSRAM
#define CAMERA_MODEL_M5STACK_ESP32CAM // No PSRAM
#define CAMERA_MODEL_M5STACK_UNITCAM // No PSRAM
#define CAMERA_MODEL_M5STACK_CAMS3_UNIT // Has PSRAM
#define CAMERA_MODEL_AI_THINKER // Has PSRAM
#define CAMERA_MODEL_TTGO_T_JOURNAL // No PSRAM
#define CAMERA_MODEL_XIAO_ESP32S3 // Has PSRAM
/** Espressif Internal Boards */
#define CAMERA_MODEL_ESP32_CAM_BOARD
#define CAMERA_MODEL_ESP32S2_CAM_BOARD
#define CAMERA_MODEL_ESP32S3_CAM_LCD
#define CAMERA_MODEL_DFRobot_FireBeetle2_ESP32S3 // Has PSRAM
#define CAMERA_MODEL_DFRobot_Romeo_ESP32S3 // Has PSRAM
#include "camera_pins.h"

```

Il faut aussi renseigner les deux lignes suivantes :

```

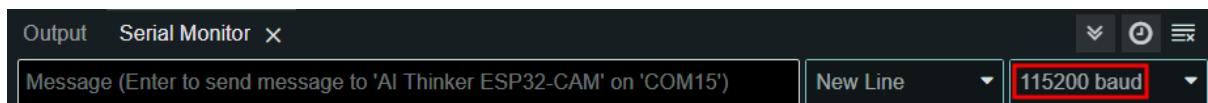
// Enter your WiFi credentials
// =====
const char *ssid = "*****";
const char *password = "*****";

```

Remplacez le SSID par le nom de votre réseau WiFi et saisissez le mot de passe correspondant. Cela permettra à la carte de se connecter à votre réseau Wifi.

Pour téléverser le programme sur la carte, appuyez d'abord sur le bouton reset (RST) de l'ESP32, puis lancez le téléchargement en cliquant sur le bouton . Une fois le téléchargement achevé, retirez le fil reliant GND à IO0, puis appuyez à nouveau sur le bouton reset de la carte.

Ouvrez le “Serial Monitor” en cliquant sur l'icône (en haut à droite)  et sélectionner 115200 bauds.



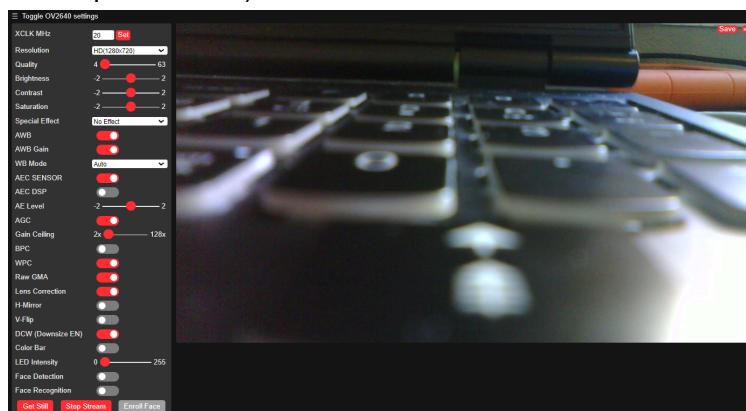
Vous devriez obtenir le message suivant dans le Serial Monitor :

```

16:44:36.440 -> WiFi connected
16:44:36.440 -> Camera Ready! Use 'http://172.20.10.12' to connect

```

En copiant l'URL obtenue, vous pouvez accéder au flux vidéo (votre PC doit être connecté au même réseau Wi-Fi que l'ESP32) :



Pour accéder au flux dans notre programme, nous utilisons ‘cv2’ (voir Figure).
L’URL à spécifier est au format "IP:PORT/stream".

Dans notre exemple, cela nous donne l’URL suivante : `url = 'http://172.20.10.12:81/stream'`

II - Import du code depuis GitHub

Le dépôt Github de notre code est accessible au lien suivant : https://github.com/titoulef/Multivisio_V2. Pour le cloner, ouvrez un terminal ou une invite de commandes et naviguez jusqu’au répertoire où vous souhaitez cloner le dépôt.

Par exemple :

```
cd chemin/vers/votre/repertoire
```

Ensuite, utilisez la commande git clone suivie de l’URL du dépôt :

```
git clone https://github.com/NomUtilisateur/NomDuDepot.git
```

III - Paramétrage pour run le programme sur une caméra

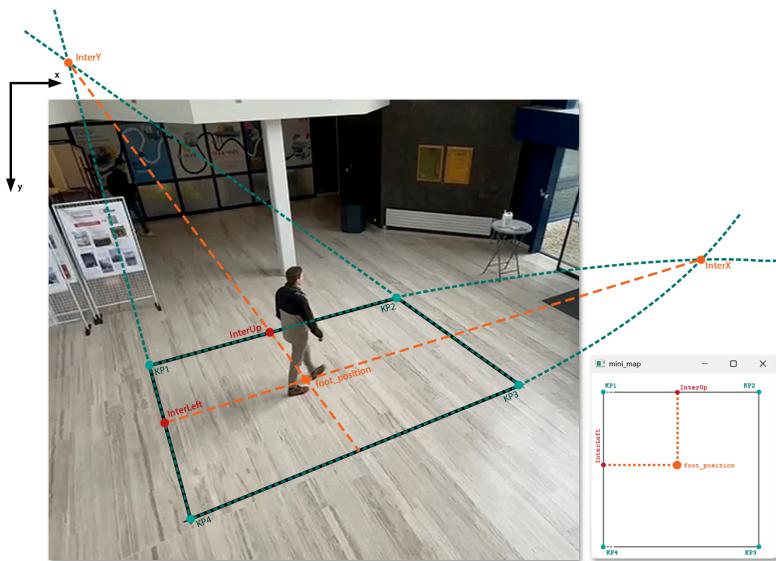
Pour une seul caméra, on utilise la méthode `loop` de `multivisio.py`.

```
multivisio.loop(streamPath, keypoints, fpsDivider=5,  
videoScale=videoScale)
```

Ensuite dans le main, remplacer le `input_stream_path` par l'url de votre caméra IP.

```
def main(): 1 usage  ▾ Titouan Lefevre *  
    # Set the video scale  
    videoScale = 0.5  
  
    #différents types de inputs  
    #caméra IP  
    input_stream_path = ('http://172.20.10.12/stream')
```

Nous devons maintenant positionner nos keypoints sur notre vidéo. Pour rappel, ces keypoints permettent ensuite de positionner les personnes/valises sur une mini carte 2D.



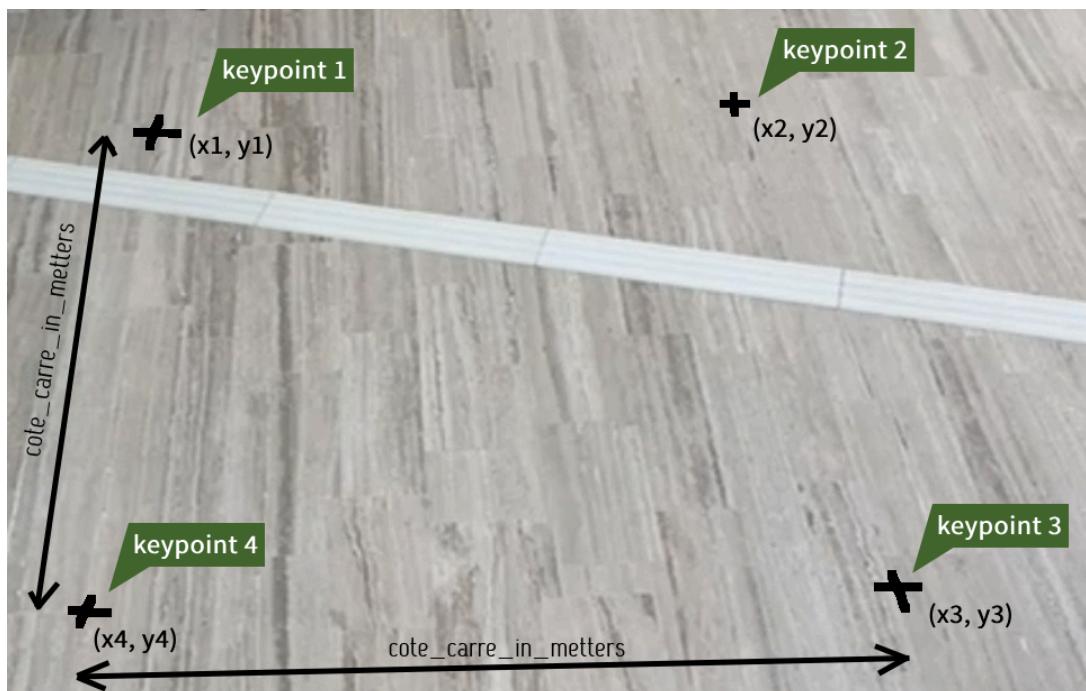
Pour ce faire, tracez quatre repères correspondant aux angles d'un carré sur le sol et notez la longueur du côté de ce carré. Ensuite, allez dans le fichier 'multivisio.py', dans la méthode `loop(input_video_path, keypoints, fpsDivider, videoScale)`, et remplacez la variable `cote_carre_in_metters` par la longueur du côté de votre carré tracé au sol, en mètres (ligne 52). La variable `radius_in_pixels` définit la distance seuil pour lequel un valise sera considéré comme perdu.

```

S0      # Configuration du rayon d'association
S1      radius_in_metter = 1.0
S2      cote_carre_in_metters = 3.0
S3      radius_in_pixel = convert_meters_to_pixel_distance(radius_in_metter, cote_carre_in_metters, 300 - 30)

```

Exécutez le "main.py". Vous devriez voir apparaître le flux vidéo ainsi qu'une mini carte. Positionnez vos caméras de manière à visualiser les quatre keypoints sur la vidéo. Il faut maintenant entrer les coordonnées de ces keypoints en pixels. Pour ce faire, cliquez avec la souris sur les keypoints visibles sur la vidéo (croix en scotch noir sur l'illustration ci-dessous). Les coordonnées des clics en pixels seront affichées dans la console.



Les coordonnées des keypoints doivent être renseigné dans le main.py de la façon suivante :

```
keypoints = [x1, y1, x2, y2, x3, y3, x4, y4]
```

Attention, l'ordre est important. Si cet ordre n'est pas respecté, vous risquez de rencontrer des problèmes avec la mini-carte.

Le programme devrait maintenant fonctionner correctement. Si vous souhaitez utiliser plusieurs caméras, vous pouvez consulter la méthode `loop2`, qui gère deux caméras.

Pour toutes questions, contacter : titouan.lefevre@ensta.fr