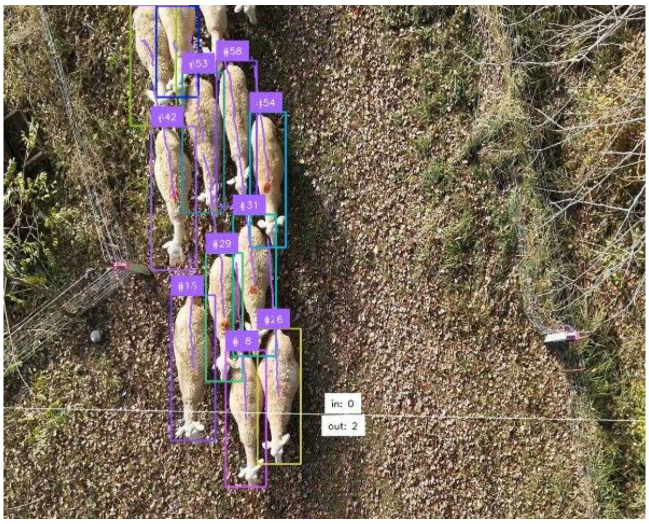
Sheep Counting from Drone Video

Counting sheep is a tedious and time-consuming task. Currently, it is done manually or with electronic loops, which remains slow and impractical when dealing with hundreds of animals. As part of the European ICAERUS project, we are exploring how computer vision and drone technology can automate this process. Learn more about the project here: https://icaerus.eu/hover-view-how-to-count-sheep-without-falling-asleep-using-drones-and-ai-to-monitor-flocks/

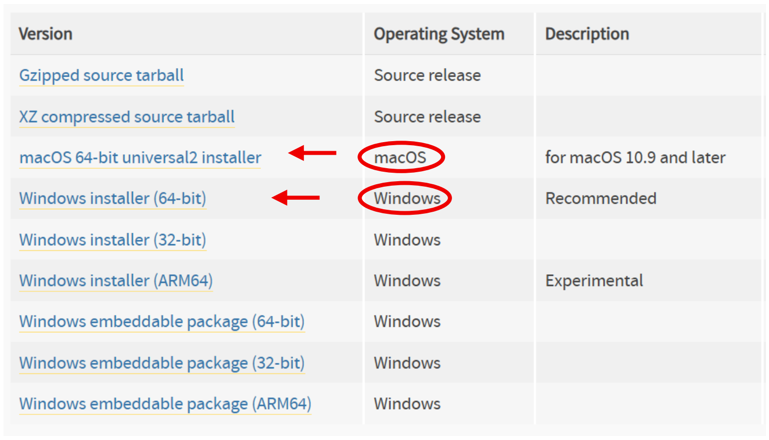
This protocol describes how to install and use a software demonstrator dedicated to counting sheep filmed by a drone. The user imports a video of their flock, taken by a drone flying about 10 m above ground while the animals pass through a corridor. The demonstrator optimizes the video for counting, then lets the user draw a virtual line adapted to the scene. An algorithm based on the YOLO AI model—specifically trained for aerial detection of sheep and goats—then identifies the animals crossing the line and automatically counts them.



1. Preparing the Host Machine

The demonstrator was developed in Python. To use it, you need to install Python on your computer and set up a development environment. In our case we use Visual Studio Code, which provides all the tools required to work with Python.

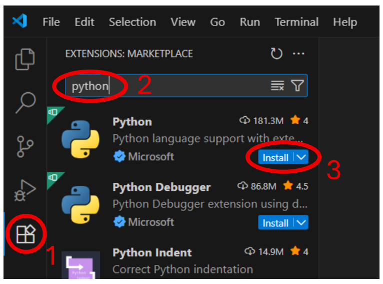
- Install Python 3.11.0:
  https://www.python.org/downloads/release/python-3110/



Version	Operating System	Description
Gzipped source tarball	Source release	
XZ compressed source tarball	Source release	
macOS 64-bit universal2 installer	[✓ macOS]	for macOS 10.9 and later
Windows installer (64-bit)	[✓ Windows]	Recommended
Windows installer (32-bit)	Windows	
Windows installer (ARM64)	Windows	Experimental
Windows embeddable package (64-bit)	Windows	
Windows embeddable package (32-bit)	Windows	
Windows embeddable package (ARM64)	Windows	

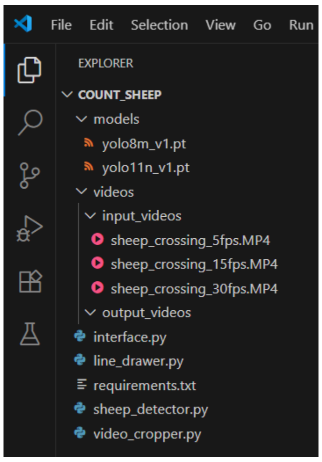
- Install Visual Studio Code (VS Code):
  https://code.visualstudio.com/download

- Install the Python extension for VS Code:
  - Open VS Code
  - Go to the Extensions tab (icon on the left shown below)
  - Search for “Python” and install Microsoft’s official extension


File Edit Selection View Go Run Terminal Help
EXTENSIONS: MARKETPLACE
python
2
Python
181.3M 4
Python language support with exte...
Microsoft
Install
3
Python Debugger
86.8M 4.5
Python Debugger extension using d...
Microsoft
Install
1
Python Indent
14.9M 4
Correct Python indentation

2. Getting the GitHub Project

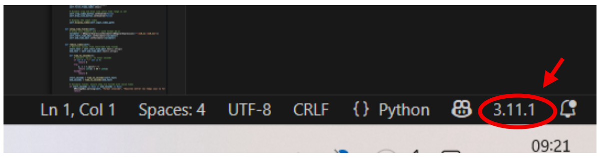
- Download the count_sheep directory to your machine.
- The directory contains:
  - models/ : AI detection models
  - videos/ : input and output videos
  - Scripts needed for the user interface
  - requirements.txt : list of Python libraries and versions
- Add your drone videos of the flock to videos/input_videos
- Open the project in VS Code:
  - Click “File > Open Folder”
  - Select the count_sheep folder



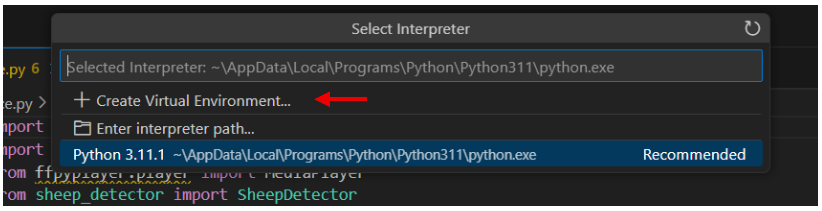
- Open the file interface.py

3. Creating the Virtual Environment

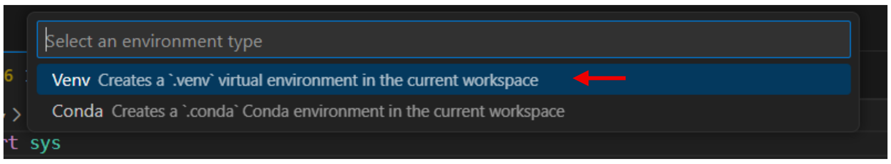
- A virtual environment must be created to install all required library versions for the interface to run.
- In VS Code, click the Python version shown in the bottom right status bar (if the bar isn't visible: "View > Appearance > Status Bar")



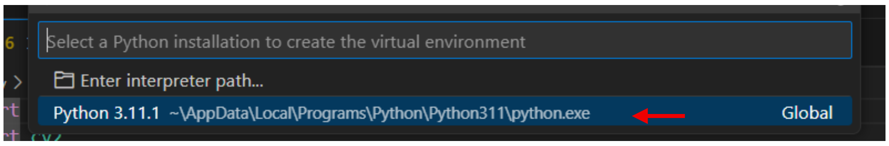
- A window appears at the top of the page: click "Create Virtual Environment"


Select Interpreter
Selected Interpreter: ~\AppData\Local\Programs\Python\Python311\python.exe
+ Create Virtual Environment...
Enter interpreter path...
Python 3.11.1 ~\AppData\Local\Programs\Python\Python311\python.exe
Recommended

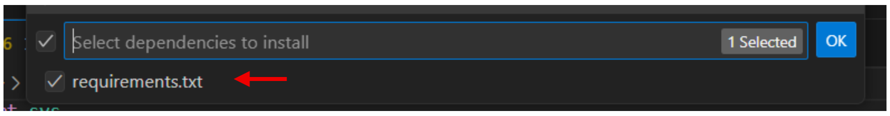
- Click "venv" (this creates a Python virtual environment)


Select an environment type
Venv Creates a `.venv` virtual environment in the current workspace
Conda Creates a `.conda` Conda environment in the current workspace

- Select the Python interpreter (3.11.x)


Select a Python installation to create the virtual environment
Enter interpreter path...
Python 3.11.1 ~\AppData\Local\Programs\Python\Python311\python.exe
Global

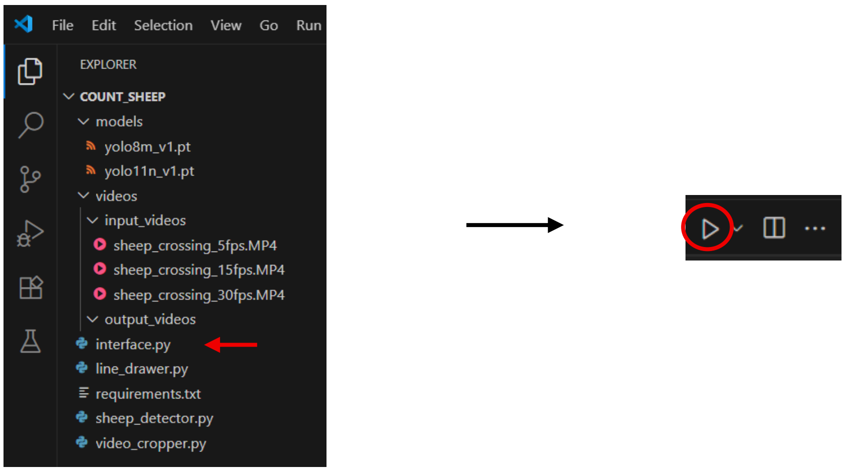
- Select the requirements.txt file and click "OK"



- Dependencies will install automatically. This can take 15–40 minutes depending on your computer.
- Check that a .venv folder has been created in the project directory.

4. Functional Testing

- Open interface.py
- Run the file by clicking the Run arrow (top right). Red lines may appear but the interface will show up after a few seconds.
- A window opens: the user interface is now operational.



<u>Selecting a Video</u>

- Choose a video for sheep/goat counting.
- Make sure it:
    - Is taken from a drone at about 15 m altitude
    - Has no visual obstacles blocking the herd
    - Shows animals moving through a defined corridor
    - Is not overly zoomed in on the passage area
- Select an appropriate FPS (frames per second): this affects the output video’s smoothness and processing time (lower FPS = faster processing).
- Choose one of the available detection models.
- Click “Start”.

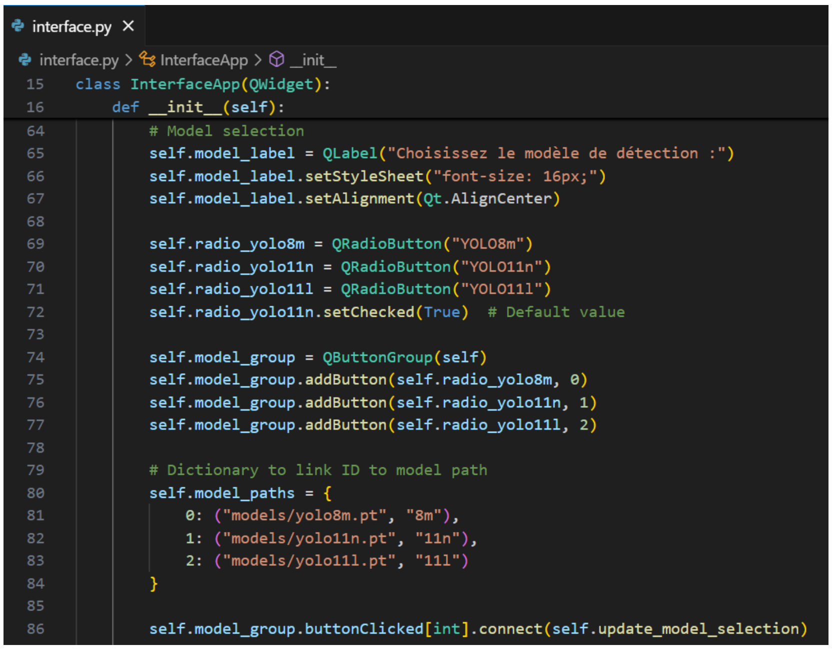
<u>Processing the Video</u>

- A preprocessing step optimizes the video for detection:
    - Trim the video to keep only the useful sequence (“minutes:seconds”)

- Crop the video to focus on the area of interest. It will be converted to 640×640 px for optimal YOLO detection. The closer your selection is to 640×640 px, the less the quality degrades.
  - Draw a counting line parallel to the direction of animal movement and long enough for all animals to cross.
  - Reset the video if the cropping or line is wrong.
  - If the crop does not show the whole herd, use another video.
- Click “Start Counting” to export the edited video to videos/crop_videos and launch the counting algorithm.
- The annotated video (detections, tracking, and counter) will display and is also exported to videos/output_videos.

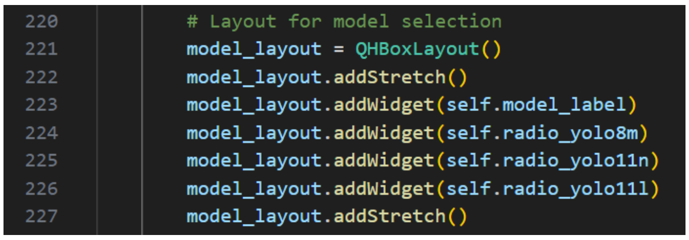
5. Modifications

- AI models evolve quickly. You can add new models to the interface by editing interface.py in the “#Model selection” section:
  - Add a new QRadioButton with the model name:
    ```
    self.radio_myvariable = QRadioButton("myyolo")
    ```
  - Add this button to the QButtonGroup with a unique value:
    ```
    self.model_group.addButton(self.radio_myvariable, 3)
    ```
  - Add the model path to the dedicated dictionary along with its suffix (version number and size):
    ```
    2: ("models/yoloxx.pt", "xx")
    ```



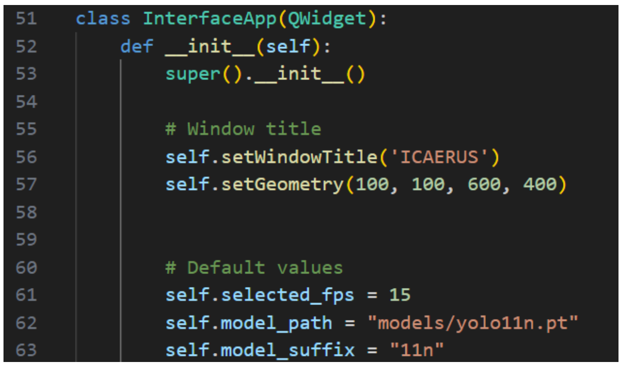
```python
class InterfaceApp(QWidget):
    def __init__(self):
        # Model selection
        self.model_label = QLabel("Choisissez le modèle de détection :")
        self.model_label.setStyleSheet("font-size: 16px;")
        self.model_label.setAlignment(Qt.AlignCenter)

        self.radio_yolo8m = QRadioButton("YOLO8m")
        self.radio_yolo11n = QRadioButton("YOLO11n")
        self.radio_yolo11l = QRadioButton("YOLO11l")
        self.radio_yolo11n.setChecked(True)  # Default value

        self.model_group = QButtonGroup(self)
        self.model_group.addButton(self.radio_yolo8m, 0)
        self.model_group.addButton(self.radio_yolo11n, 1)
        self.model_group.addButton(self.radio_yolo11l, 2)

        # Dictionary to link ID to model path
        self.model_paths = {
            0: ("models/yolo8m.pt", "8m"),
            1: ("models/yolo11n.pt", "11n"),
            2: ("models/yolo11l.pt", "11l")
        }

        self.model_group.buttonClicked[int].connect(self.update_model_selection)
```

- Finally, update the “#Layout for model selection” section to add the model choice to the interface:
  model_layout.addWidget(self.myyolo)



```python
        # Layout for model selection
        model_layout = QHBoxLayout()
        model_layout.addStretch()
        model_layout.addWidget(self.model_label)
        model_layout.addWidget(self.radio_yolo8m)
        model_layout.addWidget(self.radio_yolo11n)
        model_layout.addWidget(self.radio_yolo11l)
        model_layout.addStretch()
```

- To change default values—such as the output video FPS, the model path, or its suffix—edit the corresponding section shown below in the code.



```python
class InterfaceApp(QWidget):
    def __init__(self):
        super().__init__()

        # Window title
        self.setWindowTitle('ICAERUS')
        self.setGeometry(100, 100, 600, 400)


        # Default values
        self.selected_fps = 15
        self.model_path = "models/yolo11n.pt"
        self.model_suffix = "11n"
```