Subsystem requirements

## Subsystem name

“Crab detection with single-camera”

## Subsystem objective

The program should check every X seconds the camera feed of a single camera input.

If the frame captured is “different enough” from the previous captured frame (the 2 frames are blurred and grayscale, the output frames will have the colors), the program runs a model inference to detect if there are crabs.

If the confidence of the inference surpasses a certain threshold, the frame gets labeled with the appropriate bounding box, class and confidence value. This is done for each crab detected

All the frames labeled get saved in a leaf path and a .json file will list all the useful information about the captured frames (the inference, the name of the captured frame, the timestamp of the moment the frame was captured)

If the user presses the ‘q’ key, the program terminates.

You may easily change the following options:

* Path of the model used for the inferences
* ID of the camera (in case there’s more than one camera)
* Interval (in seconds) for capturing the camera feed
* Leaf path where to save the program (relative to the folder where the program is stored)
* Show/not show the camera feed
* Confidence threshold
* Escape key (change from ‘q’ to another key)
* Color and thickness of boxes and texts
* Font size for the labeling

## Outputs form subsystem

Leaf folder containing:

* The captured images that satisfy the previously stated conditions
* A .json file listing all the useful information about the captured frames.

## Inputs to subsystem

The subsystem needs a single-camera feed. The program works with a usb camera, it may present compatibility issues if you use the Raspberry pi Camera Module 3. I wasn’t able to test the program with that hardware since it wasn’t available and it might require some sort of setup and fixes by the ROS2 team.

## Output performance targets

The component should “be able to detect crabs”, but since the program should be able to run on an embedded system a few compromises had to be made: in order to allow the deployment of the program on a resource-constrained edge device, the nano version of YOLO11 was fine-tuned for the task of detecting crabs.

This choice is expected to allow for the best possible performance at the expense of the quality of the predictions.

The subsystem’s main goal is to facilitate spatial distribution and behavioral studies of crabs in Venice’s lagunas.

## Input performance requirements

The component needs either decent storage space in the embedded system in order to store the captured frames or a reliable connection to another device in order to send the frames and the .json file.

Software documentation

## Main program

“CrabDetection<System>.py” is the main program to run.

Use different program based on the system you’re on (see compatibility)

The program “CrabDetectionRaspberry.py” should be appropriately modified by the ROS2 team to properly function in the final embedded system.

## Compatibility

### Windows 11 (recommended pycharm)

The program functions correctly with the following conditions:

* **OS:** Windows 11
* **Python version:** 3.10
* **Libraries:**
  + **os:** create (or check if it already exists) the path directory for saving captured frames
  + **time**: accurately check the time between the previous inference and the current time
  + **datetime**: timestamps
  + **json:** write the .json file
  + **cv2:** get the camera feed and manipulate frames
  + **ultralytics (8.3.107):** run inferences using the fine-tuned model
  + **keyboard**

#### Setup

Download the folder or unzip “CrabProject.zip”

Install all the libraries mentioned above

#### Run

Open the folder containing the main program in the terminal and run CrabDetectionWindows.py

### Raspberry Pi 5 with Ubuntu Noble Numbat Ubuntu 24.04 LTS "Noble Numbat”

#### Setup

Download the folder or unzip “CrabProject.zip”

Open the terminal inside the folder “CrabProject” and follow the steps below:

1. **Install python 3.10**

sudo add-apt-repository ppa:deadsnakes/ppa

sudo apt update

sudo apt install python3.10 python3.10-venv python3.10-dev

1. **Create and activate the venv (to install libraries only for this project and avoid conflicts)**

python3.10 -m venv venv

source venv/bin/activate

1. **Install all necessary libraries (exactly these versions, otherwise you’ll run into errors)**

pip install –upgrade pip

pip install datetime

pip install opencv-python

pip install torchvision ==0.21.0

pip install torch==2.6.0

pip install ultralytics==8.3.107

1. Modify the variable “MODEL\_PATH” inside CrabDetectionRaspberry.py and insert the absolute path to the weights (the “yolo11n.pt” file inside the “Model” folder)

*Ex. "/home/gianmarco/Documents/SensingRigs/CrabProject/Model/yolo11n.pt"*

1. **Run the program (see below)**
2. **Deactivate venv**

Deactivate

#### Run

To run the program, just use the command:

python3.10 CrabDetectionRaspberry.py

You may change other variables (from CAMERA\_ID to FONT\_SIZE) to fix issues or customize some settings

#### Errors

If you run into errors, it’s likely due to incompatibility with the libraries. You may try these steps:

1. **Deactivate venv**

Do step 6. of the setup

1. **Delete previous venv**

rm -r venv

1. **Repeat step 2 of the setup**
2. **Instead of doing step 3, you may try this program:**

pip install -r requirements.txt

1. **Repeat step 5 of the setup**