Report of the 3rd 3D Perception Project

Authors: Francesco De Patre, Davide Faroldi Lo Presti

Detection and Localization of People Using Kinect, GPS, YOLO, and Compass, with Data Fusion Through a Neural Networkento

1. Introduction

The project focused on improving and integrating the previous project by adding a compass to the existing sensors (Kinect RGB, depth sensor, and GPS) to calculate the global position of the detections. All data are then combined and processed through a neural network with an LSTM implementation.

The main objective, different from previous projects, was to develop a proof-of-concept system for real-time face detection and distance measurement using data fusion to determine the potential global position of detected faces. This approach was chosen because training the designed model would require a substantial dataset, while ours was rather small, and still took approximately 5 hours per training cycle (10 epochs).

2. Librerie, algoritmi e framework utilizzati

- **YOLO11**: The latest deep learning model from Ultralytics for fast object detection, trained on a custom dataset
- **MEGAFACE**: Dataset containing face/person images for training
- **Free Viewpoint RGB-D Video Dataset:** Dataset containing rgb and depth data for training
- **PyTorch**: Framework for training YOLO and the custom neural network
- **Freenect**: Library to interface with the Kinect
- **OpenCV**: Used for image preprocessing and output
- **NumPy**: Used for matrix operations and numerical management
- **Folium**: Library for creating interactive maps

3. System structure

The system is composed of several phases and supports both real-time processing and post-processing. **Real-time** processing consists of the following phases:

- **Calibration**: The calibration process of the Kinect sensor cameras is based on acquiring images of a chessboard placed in various positions in front of the device. The chessboard is used by Kinect as a fixed reference since the corner points are arranged in a regular pattern. During calibration, the system detects the corners of the chessboard in RGB and depth images, associating each detected point with the corresponding coordinates in space. Once the images are collected, an algorithm calculates the intrinsic matrices of the cameras and their distortion coefficients, which are needed to correct the image and improve the accuracy of spatial perception. The obtained parameters are saved and can be used later to improve alignment between images and depth data.

- **Acquisition of RGB Images, Depth Map, GPS Signal, and Compass Data**: In this phase, the Kinect sensors capture RGB and depth data, while the GPS provides global position data and the compass provides direction data
- **Object Detection with YOLO**: Once the image is acquired, it is prepared for processing by the YOLO model. The "last.pt" model and "obj.names" classes are loaded in the code.
- **Data Input to the Network and Processing**: The data are converted into tensors, put in a frame-by-frame list (up to 10 frames), and passed to the model, which extracts features frame by frame, then concatenates them and returns its estimate.
- **Visualization and Feedback**: The detected objects are displayed on the RGB image with bounding boxes, their ID, and the distance estimated by the network. Additionally, for each frame, an HTML file containing a map centered on the GPS signal is generated and overwritten, with as many points of interest as the number of detected objects, indicating their potential position in space.

Post-processing instead includes the following phases:

- **Acquisition of RGB Images, Depth Map, GPS Signal, and Compass Data**: In this phase, the program loads the RGB video, depth video, GPS signal (contained in a file), and compass data (also contained in a file).

Object Detection with YOLO, **Data Input to the Network and Processing**, and **Visualization and Feedback** are the same as in real-time processing, and the results are also saved in a CSV file.

4. Issues Encountered

Although this project shows fairly promising data, it is far from perfect. The insufficient training due to the dataset size is evident during prediction. Data we tried to make secondary in this static project (the camera is fixed and does not move) still have a significant impact on the network's predictions. Changes in orientation or position (even in the same evaluation file) lead to different results, albeit all within an acceptable range. The "distance_plot_0-1-2" images show this variation as the orientation angle increases [10.2°, 189°, 358°].

5. Conclusions

This project builds on the previous one by adding compass data and a data fusion mechanism via a neural network. Although the evaluation data indicate the need for a larger dataset and significantly more training, it yields promising results regarding the potential capabilities it could have if it did not suffer from its current shortcomings.