Report of 3rd project of 3D Perception, Learning Based Data Fusion course

Authors: Francesco De Patre, Davide Faroldi Lo Presti

Detection and Localization of human faces using Learning-based Data Fusion techniques

Introduction

[Description of the project objective and brief explanation of its functioning]

This project focused on improving and integrating previous work, expanding the Kinect's sensory system, already composed of RGB, depth and GPS sensors, with a compass to calculate the global position of detections more accurately.
The data collected by these sensors is combined and processed via a neural network that incorporates an LSTM (Long Short-Term Memory) module, allowing temporal data processing and improved position estimation.
Unlike previous projects, the main objective was to develop a proof-of-concept for a real-time face detection and distance measurement system, with data fusion to determine the global position of detected faces.
This setup reflects a notable challenge: while the designed system is functional, optimal model training would require a larger dataset to improve performance and prediction accuracy.
Currently, in fact, the limited dataset at our disposal, although allowing training over 10 epochs, requires approximately 5 hours to complete each training cycle.
Our project's code is divided into three main versions, each with a specific purpose.
The first two versions are dedicated to real-time processing: one uses data from sensors and the other operates without them.
The third version, however, was developed for post-processing, allowing for in-depth analysis of the collected data.
Each version uses dedicated modules for managing the sensors and the multimodal model, optimizing the integration and management of the different components of the system.

Architecture

[Project architecture and main components]

The main components that were used in the project are different:

- **YOLOV11:** We used the latest version of Ultralytics' YOLO model for fast and accurate object detection, trained on a custom dataset for face and person detection. YOLO was chosen for its high speed and accuracy, particularly suitable for real-time sensing scenarios.
- **Free Viewpoint RGB-D Video Dataset**: Dataset of synchronized rgb and depth data for face recognition and localization, unfortunately the size of the dataset is only sufficient for a demonstration.
- **Pytorch**: Pytorch was chosen as the main framework for training the YOLO model and our LSTM-based custom neural network for data fusion. Its flexibility and support for GPU

processing made it easy to deploy and train the deep learning models needed for the project.

- **Freenect:** For interaction with sensors.
- **Folium:** Library for creating interactive maps.
- **OpenCV:** Image processing library.
- **NumPy:** Library used primarily for managing matrix operations.

The code is organized as a data fusion system that integrates real-time face and distance detection using Kinect, GPS and compass sensors.

The architecture is structured into various modules that collaborate to collect, process and visualize data.

After importing key libraries, such as YOLO and pyTorch, for inference with the multimodal model, the system opens the recorded RGB and depth video streams, along with the latitude and longitude data provided by GPS.

The depth map is converted to meters to calculate the actual distance to the detected points and the images are pre-processed for YOLO inference, which localizes the detected faces. For each face, the real distance and orientation angle with respect to the camera center is calculated.

Subsequently, a multimodal LSTM model receives as input the cropped face image and numerical data (real distance, latitude, longitude and angle) to predict the distance to the camera.

Real and predicted distances are stored in a CSV file for future analysis, and an interactive map shows the global location of each face detected using Folium, positioning faces based on latitude, longitude, and angle.

Finally, the result is recorded in an output video with graphical overlays, while a real-time control allows you to stop the program with 'q'.

This architecture allows you to combine different data sources to accurately estimate distances and global positions of faces, although it is limited by the quantity and quality of available training data.

Multimodal Network

[Description on the multimodal network that we have implemented in the project]

The MultiModalLSTMModel multimodal neural network is designed to combine visual and numerical information to estimate distances in dynamic contexts.

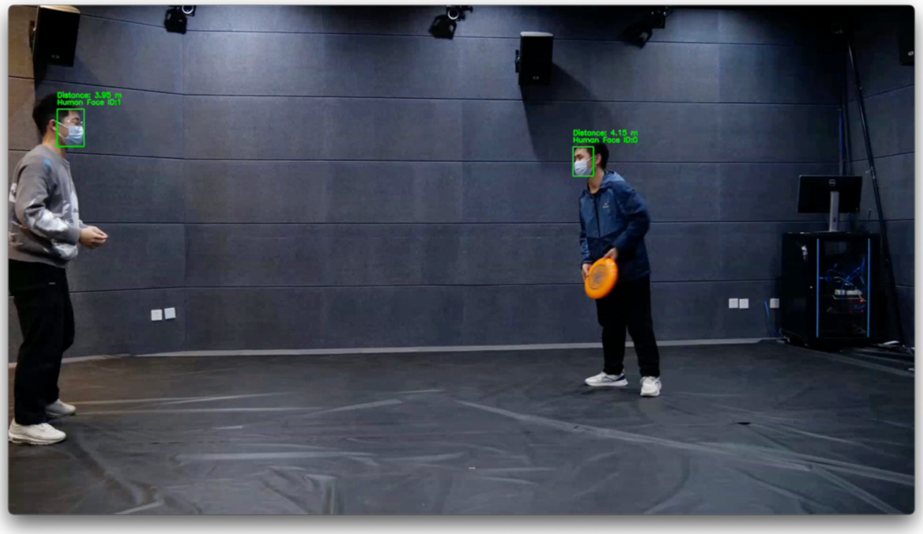
It uses a pre-trained ResNet18 model to extract visual features from sequences of RGB frames, producing a vector of size 512 for each frame. At the same time, numerical data, such as depth, latitude, longitude and orientation, are processed by a fully connected network that reduces the size of the information to 32 features.

The visual and numerical features are then combined and passed to an LSTM network to capture the temporal context, with the output of the LSTM representing a temporal sequence of the information. Finally, a layerfullyconnected is used to generate the final distance prediction.

This approach allows the network to effectively integrate information from different sources, improving the estimation capacity in complex scenarios.

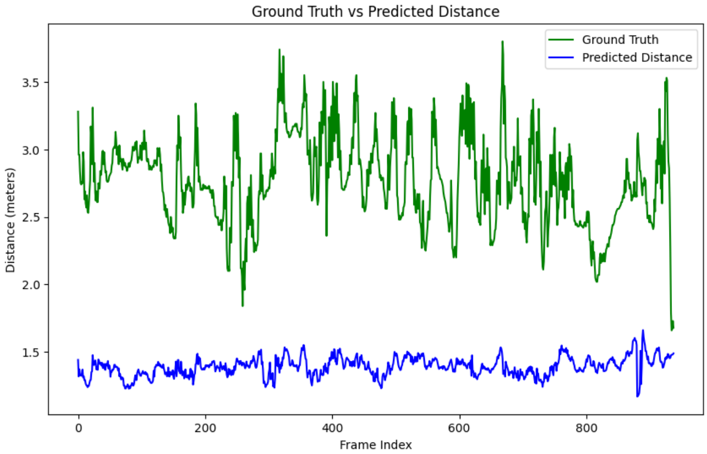
Results

[Results obtained from the project]

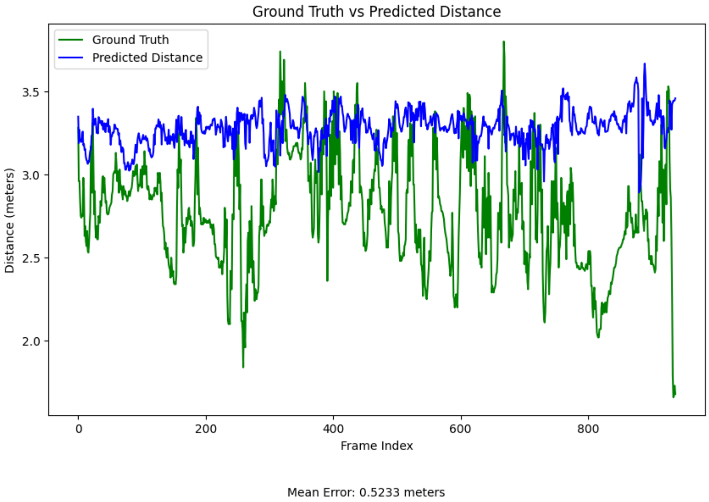


The image shows the output obtained from our project using post-processing for speed reasons.
The bounding boxes surround the detected faces indicating the face ID for distinction and the distance at which the targets are located.

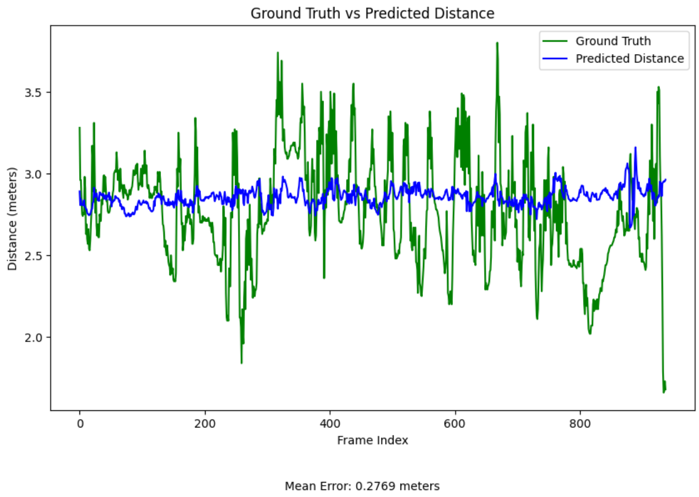
As regards the analysis of the results, however, we considered different orientations of the compass [10.2°,189°,358°].
Below are the analyzes on the average error.



The first graph represents the analysis of the system at an orientation of 10.5°, in this case the predictions follow a trend similar to the ground truth.



The second graph instead represents the analysis of the system at a greater orientation, of 189°. The predictions partially follow the ground truth but still achieve moderate accuracy.



Finally, the last graph represents the analysis of the system at an orientation of 358°.
The predictions are significantly distant from the ground truth and therefore there is a poor prediction.

The graphs show that the system relies on secondary information (for our stationary system) to make predictions about the various positions.

Problems encountered

[Problems encountered during the development of the project and the solutions adopted]

Although this project shows promising data, it is still far from perfect.
The limitations of the dataset used for training clearly emerge during predictions.
Variables that we would have liked the network to consider secondary, such as camera movement, deliberately absent here since the camera is fixed, continue to significantly influence the results.
Changes in camera orientation or position within the same evaluation file produce different results; however, the values remain within an acceptable range.
The “distance_plot_[0,1,2]” images illustrate these variations, highlighting how the results change as the orientation angle increases.
One possible solution we have considered is to treat the sensor as "fixed" for our purpose, allowing the different angles to be considered as distinct contexts.
In this way, we could train the model to interpret each angle as a predefined variation within a known scenario, improving the consistency of the results without having to excessively increase the dataset.

Conclusions

[Conclusions and possible development of the project]

This project represents an extension of previous work, enriched by the integration of data from a compass and a data fusion mechanism implemented via a neural network.
Although the evaluation results highlight the need for a larger dataset and a more intensive training cycle, the results obtained are promising and offer a preview of the potential of the system, once the current limitations are overcome.