Report of the 1st Project on 3D Perception

Authors: Francesco De Patre, Davide Faroldi Lo Presti

Object Detection and Distance Calculation with Kinect Sensor and YOLO Algorithm.

Introduction

The project focuses on integrating a Kinect sensor with a computer vision system that uses the YOLOv3 detection algorithm to detect faces in real-time and estimate their distance from the sensor.

This system utilizes the depth map provided by the sensor to gather information that complements the algorithm's capabilities.

The main goal is to develop a real-time face detection and distance measurement system, with potential applications in fields such as robotics, autonomous driving, and surveillance.

Libraries, Algorithms, and Frameworks Used

- **YOLOv3**: A deep learning model for fast object detection, trained on a custom dataset.
- **MEGAFACE**: Dataset containing images of faces/people for training.
- **Darknet**: Framework for training YOLO.
- **freenect**: Library for interfacing with Kinect and accessing RGB and depth data.
- **OpenCV**: Used for image preprocessing, the DNN framework that handles YOLO, and output representation.
- **NumPy**: Used for matrix management and numerical operations.

System Structure

The system consists of several stages, each performing a specific function in perception.

Calibration

The Kinect is initially calibrated to adjust the tilt angle, allowing the sensor to capture images within an optimal field of view. This process is done using the freenect module, which provides functions to control the sensor's angle.

In the code, the calibration is performed at 27° and -27°, and finally reset to 0° to position the sensor horizontally.

Image and Depth Map Acquisition

At this stage, the sensor starts capturing two types of data:

- RGB Video Stream: The color image of the surrounding environment.
- Depth Map: The distance between the sensor and the detected objects, expressed in depth values.

Both types of data are acquired in parallel; specifically, the RGB image is converted from BGR (Kinect output) to RGB to be used by OpenCV, while the depth map is stored to estimate distances.

Object Detection with YOLO

Once the image is acquired, it is prepared for processing by the YOLO model. The YOLOv3 model is loaded using OpenCV's DNN framework, which allows for detection across a wide range of objects. In the code:

- The YOLO configuration and pre-trained weights are loaded using “yolov3.cfg” and “yolov3.weights”.
- Object classes are read from the “coco.names” file.

The converted RGB image is transformed into a blob, which is a representation that normalizes the image to be compatible with the YOLO model.

The algorithm then processes the image and returns the bounding boxes of the detected objects, along with their class and confidence level.

If the confidence level is sufficiently high (>= 0.5) and the object's class is of interest, the bounding box position is saved.

Object Distance Estimation

In parallel with object detection, the Kinect's depth map is used to estimate the distance between the sensor and the objects detected by YOLO.

Each detected object of interest has an associated region in the depth map, from which an average distance value can be extracted.

This value represents the approximate distance between the object and the sensor, allowing the system to provide not only the position of objects in the 2D plane but also their distance from the sensor.

Visualization and Feedback

The detected objects are displayed on the RGB image with a bounding box, and their class (e.g., Human Face). The calculated distance is overlaid on the visualization, allowing the user to observe in real-time both the classification and the distance from the sensor.

Conclusions

This project successfully integrates the use of a Kinect sensor and the YOLO algorithm to create a real-time object detection and distance estimation system. The combination of RGB and depth vision makes the system applicable in a variety of contexts. The code can be easily extended to include additional functionalities, such as detecting other objects or integrating additional sensors.