Object Detection-Based Behaviour using YOLO on Thymio

Francesco Saverio Zuppichini[1] and Alessia Ruggeri[2]

Abstract— **The ability to classify in booth space and domain objects is encoded in an enormous variety of species on the planet. Recently, deep learning approaches have been developed to make machines able to mimic the same task with interesting results. With this paper, we propose a minimal architecture to archive the same goal using a two wheels robots, the Thymio, with a mounted frontal camera in order to act based on the object detected in the surrounding.**

I. Introduction

Finding objects in space by identifying their identity and their coordinates is a basic task that almost each animal is able to accomplish by feeding the inputs from their eyes to the brain. The ability to solve this problem is extremely important in robotics, where a machine needs to act accordingly based on its surrounding. With our architecture, only the raw data from the camera are used as input to an Object Detection model and the decision are based directly on the prediction, without the need of any other sensor.

The paper is structured as follows: we will first describe the architecture by showing each individual part of the topology and how they interact; then, we will discuss the challenges during the development process and finally we will show the results.

II. Architecture

Our architecture is composed by:

1) A robot with a mounted camera and an on-board computer with a running ROS node
2) A web server running the Object Detection model
3) A ROS node running on a local machine

As a robot, we used the *MightyThymio* [**guzzi2018eaai**], which is a robot developed on top of the Thymio, a small mass-produced mobile robot usually used in primary and secondary education. The MightyThymio has a powerful Odroid C1 on board, a 1GB quad-core ARM Linux machine, with a HD camera mounted on top. The authors also implemented a ROS node to communicate with it and open-sourced the code.

III. YOLO: REAL-TIME OBJECT DETECTION

YOLO [**DBLP:conf/cvpr/RedmonDGF16**] is a trained model for real-time object detection. While prior work on object detection repurposes classifiers to perform detection, YOLO handle the object detection as a regression problem to spatially separated bounding boxes and associated class probabilities. YOLO avoids resorting to complex pipelines that are slow and hard to optimize; instead, it uses a single convolutional neural network that simultaneously predicts multiple bounding boxes and class probabilities for those boxes. Using YOLO, You Only Look Once at an image to predict what objects are present and where they are.

YOLO neural network uses features from the entire image to predict each bounding box simultaneously across all classes. The convolutional neural network contains 24 convolutional layers, that extract features from the image, followed by 2 fully connected layers, whose aim is to predict the output probabilities and coordinates. The system divides the input image into an $S \times S$ grid; if the center of an object falls into a grid cell, that grid cell is responsible for detecting that object. Each grid cell predicts B bounding boxes and confidence scores for those boxes. These confidence scores reflect how confident the model is that the box contains an object and also how accurate it thinks the box is that it predicts. There is also a fast version of YOLO, designed with a neural network with fewer convolutional leayers and fewer filters in those layers.

YOLO is extremely fast, it reason globally about the image when making predictions, it learns generalizable representations of objects and it is open source. YOLO still lags behind other detection systems in accuracy; while it can quickly identify object in images, it struggles to precisely localize some objects, especially small ones. However, it is still the state-of-the-art model for object detection in real-time.

For our project, we used the third version of YOLO, or YOLOv3 [**Yolo3**], which has been improved by making it a little bigger than before, but more accurate.

IV. Results

V. Conclusion