**John Barker**

**MSDS 462 Final Project**

# VGG16 Turtlebot3 Classification to Initiate Turtlebot3 Action

## Introduction

Fine-tuning a pre-trained computer vision model can be useful when model training data is limited. VGG16 a pre-trained model was developed by K. Simonyan and A. Zisserman at the University of Oxford. Around 2007 a small company known as Willow Garage developed a robotic framework called Robotic Operating System (ROS). It is really a framework and not an OS which provides basic features for robot operation. Python and C++ can be used to program Robots to navigate, localize, map and manipulate objects in the robot's environment. The robot also has the ability to sense the surroundings using cameras, microphones and lidars. ROS provides three design features: Publish/Subscribe, Service and Actions to create software to operate a robot. In this project I use both ROS and VGG16 to implement my system.

## Data Collection

I collected my data by using an image Bing search for six types of images. I collected burger turtlebot3, waffle turtlebot3, cars, humans, roads, and people on roads. I divided the images into train, test and validate folders which contain folders named after the label of the images. This way ImageDataGenerator will label the images as it loads the images for training.

## Models

I trained two fine-tuned VGG16 models. The one I used in the project classified burger turtlebot3 and waffle turtlebot3. This is used to classify images coming in through a camera sensor so the fetching turtlebot3 would know which turtlebot3 to fetch. I also fine-tuned the VGG16 model to classify cars, humans, roads and people on roads to use for when I develop the software to have the turtlebot3 drive around and act properly when seeing cars, humans, roads, and people on roads.

**Feasibility**

I researched the mechanisms of ROS by developing feasibility code. First, I wrote a simple publisher and subscriber node using ROS framework. I created a talker which published a count. Then I created a listener to subscribe to talker and print out the count from the incoming msg.

I then built a simulated camera that loaded images and converted the image into a format that could be published in ROS. I then wrote a retrieve image script that subscribed to the simulated camera converted the image back to a form for displaying and displayed the image to prove the concept.

Last I developed a wandering robot to learn a little about how turtlebot3 navigates the environment. I used the /odom, /scan two publishers to calculate location and the next movement. I used /cmd_vel to request the next movement for turtlebot3 to perform. This was all done in turtlebot3_world a simulated environment.

**VGG16 and Turtlebot3 Simulation**

From all of this I built a simulation using the burger or waffle VGG16 model to classify images coming from the simulated camera. I published the image and my retrieve node retrieved the image and attempted to classify the image. This is where a failure occurred. Apparently, ROS is written in Python 2.7 which is unable to properly handle the floating-point data from the image in a Keras trained model. The error is the graph does not handle the data type. I faked this part to keep the simulation alive. From here I published the type of turtlebot3 in the image. A fetch node subscribed to the retrieve node and began a simulated process of fetching the proper tb3. This was accomplished in an empty simulated turtlebot3 world.

**Future Considerations**

Fix the classification problem by switching to ROS2 which uses Python 3.x. Rework the fetcher script to use the composite design pattern which is great for building step by step automated processes. Add an arm manipulator to my tb3. Build the driving part of tb3 and improve wander bot. Finally build my own tb3 worlds of road scenes and warehouses complete with inventory to use in further ROS or ROS2 development.

# Reference


Castillo, G. (2018) ECE 5463 Introduction to robotics. Retrieved from http://www2.ece.ohio-state.edu/~zhang/RoboticsClass/docs/ECE5463_ROSTutorialLecture3.pdf

Chollet . F. (2018). Deep Learning with Python. Shelter Island, NY: Manning Publications, Co.

Hassan, M. (2018) VGG16 – convolutional network for classification and detection. Retrieved from https://neurohive.io/en/popular-networks/vgg16/

Jafari, B. (2018) ROS programming: trying to make my turtlebot wander without running into obstacles. Retrieved from https://stackoverflow.com/questions/53487008/ros-programming-trying-to-make-my-turtlebot-wander-without-running-into-obstacl

Quigley, M. (2015) Programming Robots with ROS. Sebastopol, CA: O'Reilly Media, Inc

Rahman, A. (2018) How to rotate a robot to a desired heading using feedback from odometry? Retrieved from https://www.theconstructsim.com/ros-qa-135-how-to-rotate-a-robot-to-a-desired-heading-using-feedback-from-odometry/

ROS, (2020) Converting between ROS images and OpenCV images(Python). Retrieved from http://wiki.ros.org/cv_bridge/Tutorials/ConvertingBetweenROSImagesAndOpenCVImagesPython

## Appendix

Important project links:

GitHubProjectLink

Final_Project_Code

ProjectSimulationVideo

ROS_Setup_Notebook

Final_Project_Notebook

Feasibility_Image

Feasibility_Pub_Sub

Feasibility_Wander