Computing Project Technical Plan

Name: Nechaev Ilia
Course: Computing, Software developing
Supervisor/Supervisory Team: Dr Panayiotis Andreou

Title

An effective low computation cost line detection algorithm for Duckiebot

Summary

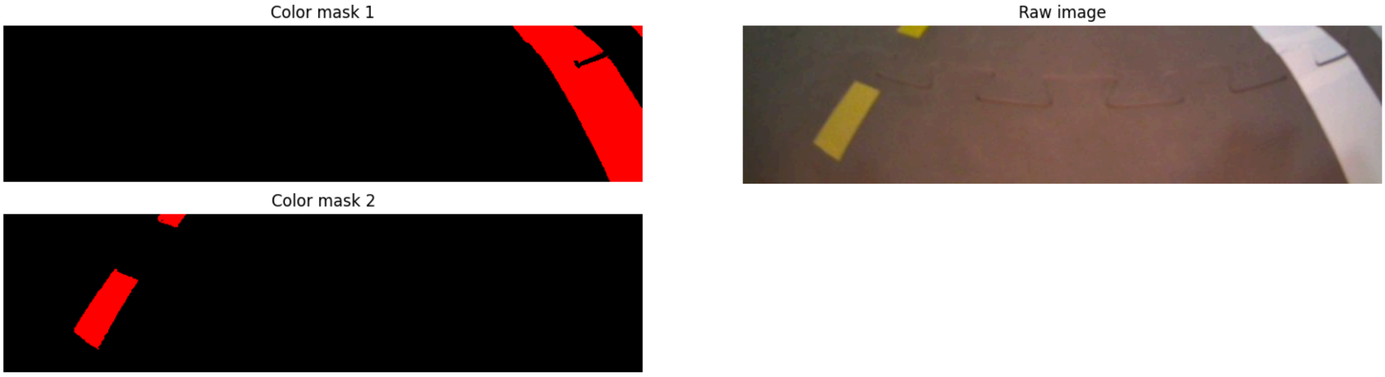
The project is to design a new line detection algorithm for Duckiebot, which will be light-insensitive and light-weighted in computation terms. The current solution used in Duckiebot is extremely unpredictable, it can change its behaviour even with little change of light: open or closed curtains with switched on in-door lights can significantly impact on current solution output.

My solution must meet two requirements: to be light-independent and to be light-weighted in terms of computation speed. To achieve this, I want to use machine learning techniques, but without deep learning models, because they are usually computational-heavy. Also I need to construct the dataset of marked up images of road marking to train my model..

So during this project I will:

1. Construct image dataset of road marking
2. Create (probably computational-heavy) reinforcement learning model for marking up the dataset
3. Create light-independent and light-weighted in terms of computation speed algorithm for detecting lines of road marking

Possible result of my algorithm:



Color mask 1

Raw image

Color mask 2

Current solution result:

Deliverables

- Construct a dataset of marked up images of road marking.
- Design and Develop the algorithm of line detection
- Package with ROS (robotic operating system) node, which contains line detection algorithm
- Evaluation of the algorithm compared to others.
- Thesis report

Constraints

- Low-computational power of Duckiebot processor prohibits the use of computationally-heavy AI algorithms

Key Problems

- Constructing the dataset with appropriate line markers that will be used as input
- Inexperience developing applications in ROS, lack of proper documentation
- Low-computational power of Duckiebot processor

System and Work Outline

Methodology: Agile Iterative and Incremental development of the components
Prioritisation:

- Must:
    - Design, develop and deploy algorithm using reinforcement learning technique for marking up dataset
    - Design and develop light insensitive and light-weight algorithm for detecting road marking
- Should:
    - Deploy light insensitive and light-weight algorithm for detecting road marking on Duckeibot
- Could:
    - Crossroad crossing algorithm based on computer vision techniques
- Won't be:
    - Light-weight general-purposes line detection algorithm, which can be used for any type of road markup

MS1. Setup Infrastructure

1. Assembly of DuckieBot [1]
2. Creating test road networks

MS2. Construct Image Segmentation Dataset

1. Setup image repository
2. Develop software for DuckieBot to send images to repository [2]
3. Collect images from DuckieBot camera
4. Download images from external repositories
5. Research on appropriate techniques for markup lines [4][5][7][8]
6. Develop and algorithm to markup lines in collected images [4][5][7][8]

MS3. Design, Develop and Deploy Image Recognition Algorithm

1. Perform scientific literature review on marking algorithms and ML models relevant to the problem, such as the best model, choosing hyperparameters.
2. Design Develop and Deploy computationally heavy algorithm that will create masks using reinforcement learning techniques
3. Design Develop and Deploy light-weight algorithm that will detect lines on road [6][9][10][11][12][13]
4. Test and Evaluate the performance and accuracy of the algorithm on DuckieBot [2][3]
5. Perform optimization/tuning of the algorithm's parameters

MS4. Thesis Writeup

Technical stack:

1. Languages and libraries that will be used:
   a. Python with NumPy, MatPlotLib, SciPy, Pandas, PyTorch, Scikit-learn
   b. May be C++ or Rust for MS2.2 step
2. Other technologies:
   a. Docker
   b. Linux OS (Ubuntu 22.04 and Ubuntu 20.04)
   c. Git

Personal development:

1. Skills that will be developed:
   a. Deploying ML models
   b. Creating ML pipeline
   c. Creating and deploying EM algorithm
   d. Deepening knowledge in classical ML (machine learning)
   e. Deepening knowledge in RL (reinforcement learning)
   f. Deepening knowledge in DL (deep learning)
2. Skills that will be acquired:
   a. Developing applications for ROS
   b. Docker (creating and adjusting docker images)

IDEs:

1. Jupyter Notebooks (in Google Colab) to develop ML models
2. PyCharm by JetBrains for developing ROS packages

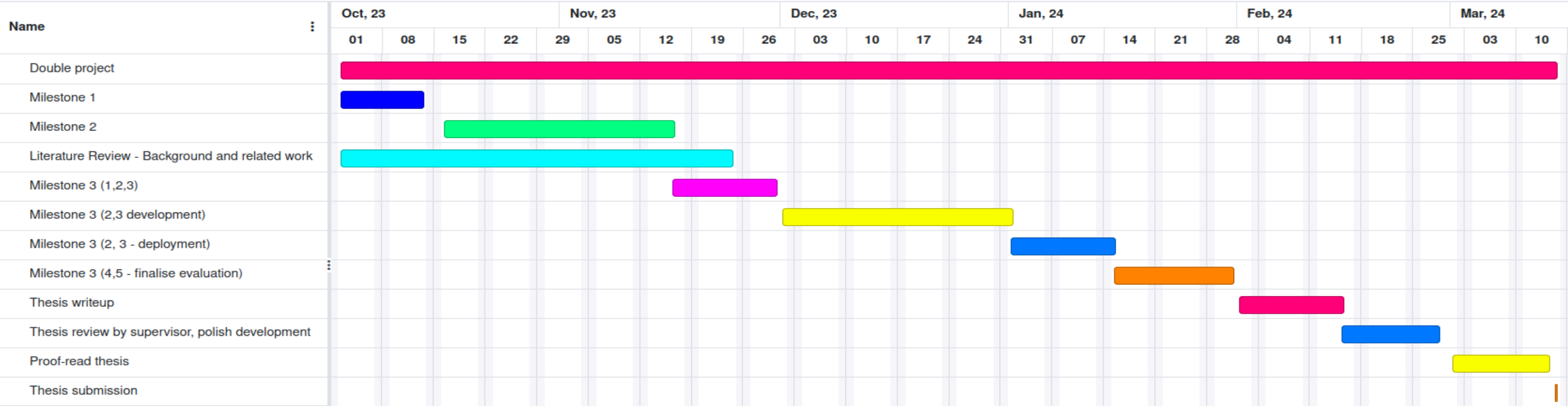
To evaluate resulting ML model I will probably use F1 score [14]:

$$F1 = \frac{2 * Recall * Precision}{Recall + Precision},$$ where

$$Precision = \frac{True\ positive}{True\ positive + False\ positive}$$ and

$$Recall = \frac{True\ positive}{True\ positive + False\ Negative}$$

Project Activities



1. 01.10-15.10 Milestone 1
2. 15.10-15.11 Milestone 2
3. 01.10-24.11 Literature Review - Background and related work
4. 16.11-30.11 Milestone 3. Steps: 1, 2, 3. Design part
5. 01.12-31.12 Milestone 3. Steps: 2, 3. Development part
6. 01.01-15.01 Milestone 3. Steps: 2, 3. Seployment part
7. 15.01-31.01 Milestone 3. Steps: 4, 5. Finalise evaluation
8. 01.02-15.02 Thesis writeup
9. 15.02-28.02 Thesis review by supervisor, polish development
10. 01.03-14.03 Proof-read thesis
11. 15.04 Thesis submission

Risk Analysis

Risk	Severity	Likelihood	Action
Lack of knowledge	Medium	High	Read more articles, books, listen to lectures
Duckiebot crash	High	Low	Fix it by myself or buy broken parts

Options

Other approaches for lane detection:

1. Extremely deep NN (not fit my bot due to lack of computational power)
2. Currently used non-NN approach. It’s based on checking if the pixel fits the colour range (not fit my idea, because this approach is light sensitive)

Potential Ethical or Legal Issues

The project does not use individuals and the test road networks will not capture images of people.
There is no danger in using the Duckiebot for others, because it’s small (12*20*10 cm) and light, so it can’t deal any damage to people or their belongings.

Commercial Analysis

It is a research project. There is no aim to gain profit with it. But algorithm that I want to develop can be used on small bots that should be automated and which can be used in area with marking similar to road one (warehouses for example)

Factor name	Description	Is this a cost or a benefit	Estimated Amount	Estimate of when paid
Duckiebot	Buying the Duckiebot	Cost	~440$ + shipping	Already paid
Road	Rubber carpets and tape to create road	Cost	30€	Already paid
Google colab subscription	Subscription for using google remote server to fit my models	Cost	10$ per month * 9 month = 90 $	This amount is paid during academic year each month

Employability Contribution

Developing a line detection algorithm for the Duckiebot will help me to increase my knowledge in computer vision. Also during this project, I will use a lot of such libraries and frameworks on Python as PyTorch, Scikit-learn, Pandas, NumPy and MatPlotLib, which will increase my knowledge in data science (data preprocessing, machine learning).

References

[1] Duckietown Inc., *Assembly - Duckiebot DB21J,* Duckietown, viewed 1 October 2023
[2] Duckietown Inc., *Duckietown Developer Manual*, Duckietown, viewed 1 October 2023
[3] ROS, *Documentation, ROS, viewed 1 October 2023*
[4] Nikolenko S, Alexandrov T, Chernyavsky I, 2014 ‘Segmentation of MALDI imaging results based on graphical models’*, SPIIRAS Proceedings, vol.* 2, DOI:10.15622/sp.21.8
[5] Nikolenko S, 2019, *‘Synthetic Data for Deep Learning’,* Springer
[6] Golovanov S, Kurbanov R, Artamonov A, Davydow A, and Nikolenko S, 2018, ‘Building Detection from Satellite Imagery Using a Composite Loss Function’, *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, Salt Lake City, UT, USA, pp. 219-2193, DOI: 10.1109/CVPRW.2018.00040.
[7] Yuzhikov V, 2012, *Restoring defocused and blurred images (translated from Russian), Habr, viewed 1 October 2023*
[8] Avidan S and Shamir A, 2007, ‘Seam carving for content-aware image resizing’, A*CM SIGGRAPH papers,* DOI:10.1145/1275808.1276390
[9] Nikolenko S, Kadurin A, Arkhangelskaya E, 2017, ‘*Deep Learning’,* Piter
[10] Goodfellow I, Bengio Y and Courville A, 2016, ‘*Deep Learning*’ MIT Press

[11] Brownlee J, 2019, ‘*Deep Learning for Computer Vision: Image Classification, Object Detection, and Face Recognition in Python*’, Independently Published
[12] Andrew G, Zhu H, Zhu M, Chen B, Kalenichenko D, Wang W, Weyand T, Andreetto M and Adam H, 2017, ‘MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications’, Computing Research Repository
[13] Zhang X, Zhou X, Lin M and Jian Sun, 2018, ‘ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices’, IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Salt Lake City, UT, USA, 2018 pp, DOI:10.1109/CVPR.2018.00716
[14] Thomas Wood, *F-Score,* DeepAI, viewed 6 October 2023