New Jersey Space Grant Consortium Final Report

Deep Dive into OpNav Through Image Segmentation and Object Tracking

Name of Researcher: Harsh Zaveri

Email and Phone: zaveriharsh4114@gmail.com, 551-275-5632

Academic Unit: Department of Computer Science

Name of University/College: New Jersey City University

Adviser: Dr. Moitrayee Chatterjee

Adviser Contact Info: mchatterjee@njcu.edu

Adviser Approval Signature: Moitrayee Chatterjee

Abstract

OpNav, the emergency return system for NASA's Orion spacecraft, uses Optical Navigation. This research is about how to enhance this system using Image Segmentation and Object Tracking. Reinforcement Learning, a Machine Learning technique that trains software to make decisions and achieve the most optimal result. Using this technique with segmented vision-based data and object tracking technology, enhanced optical navigation can be achieved on Earth and outer space.

Introduction

With the rise of AI and Elon Musk's success with Tesla in autonomous driving, different Machine Learning techniques and Neural Networks are researched with the focus of vision-based navigation, with different approaches. Convolutional Neural Network (CNN) is a very powerful tool when it comes to recognizing patterns in images. The training process for CNN takes millions of labelled test data. These labels are informative for machines, as they are used as information to determine the outcome of the image, the drivability. It's like how students learn the alphabet before learning a new language. We can understand the used concept like this – the alphabet is the labelled data. The words are the trained images, and the sentences are the use of CNN, the trained image, and Reinforcement Learning.

Related Work

To begin, autonomous navigation on Earth is considered as a basic concept. Some of the research papers we studied had a similar approach to autonomous navigation. (8) suggest autonomous driving supporting the importance of agriculture. As the development of high-level machinery is very crucial for agricultural fields to grow, semantic segmentation and autonomous driving is proposed to use as agricultural machinery as the results of these are getting better and better every day. The proposed technique to achieve this consists of Image Processing technology for autonomous navigation, with preprocessing, segmentation, and then feature extraction. (5) approach autonomous navigation with segmentation. It shows a representation of robotic outdoor navigation just based on images captured by an on-board camera. Using egocentric images for navigation directly and adapting learning problem as a navigation task is proposed with navigation-oriented pixel-wise loss weighting method to make safety-critical areas more important for the agent. The 3-level affordance method is used where green, yellow, and red colors are used to differentiate the drivability level for autonomous vehicles.

Problem Description

The current stand on this approach towards autonomous navigation relies totally on the image-based datasets being used, processed, and labelled as data for the machine learning agent. The major problem mostly every researcher has stated on this topic is either directly or indirectly related to the image-based datasets. (5) faced problems with grayscale, color-based differentiation and labelling of the images as they used CNN as well. On the other hand, (8) faced a similar problem with segmentation while their approach was using a different kind of neural network. The problem we faced while looking for the datasets and libraries to train these images was that there were companies like Audi, NVIDIA, and libraries like Cityscapes, Neptune.ai and others had datasets that can be used to approach vision-based autonomous navigation. The only problem is that these datasets are in huge numbers, sized in terabytes.

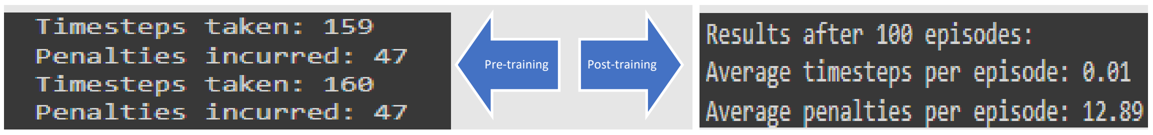
Which is why using a local device couldn't be an option. CNN also requires a high-performance device as it runs the best with a powerful external processor. So, we were left with a very restricted number of datasets and libraries and a lack of resources which fulfilled our requirements.

Experiment Setup

To learn about Reinforcement learning, there were many API's and documentations that provided with environments to train and learn about an agent. One that stood out of all these options was the Gym API, as it provided us with environments in which games could be solved using machine learning. For the datasets, after seeking help from our university, we got our hands on Coral's Edge TPU, an accelerator which processes images with the use of CNN which matched our requirement perfectly. There was also the pycoral library by Coral which we used to test image based data.

Research and Results

Reinforcement Learning has proved to be very efficient in today's world, as it is used in technologies like Siri and Google Assistant which has proved to be successful on a huge scale. The characteristics of reinforcement learning allow us to train it for predictive texts, question answering – used in the above-mentioned applications, and other real-life applications like image processing, automation, and gaming. To learn about RL in detail, being someone who is into gaming, I chose the 'Taxi' environment from Gym API. It has the objective of picking up a passenger from one of the four listed locations and dropping them to their desired destination. It seemed like a virtual, animated imitation of what we aim for in real life. In the first step, we learned that training the model with a q-function and making it train for the maximum number of rewards was necessary.



```
Timesteps taken: 159
Penalties incurred: 47
Timesteps taken: 160
Penalties incurred: 47
```

```
Results after 100 episodes:
Average timesteps per episode: 0.01
Average penalties per episode: 12.89
```

Without training the model with Q-function and sending the agent to the 'learning' period, running the script gave us about 47 penalties per 160 timesteps. Whereas after a determined 'learning' period for the agent, the penalty count was almost around 40% less than before. This pushes the idea of identification and (5) 'safety-critical' idea. If an agent is trained to achieve and differentiate correctly between the drivable and non-drivable areas, it will perform very close to the human level with the right amount of training.

After understanding RL and the ways it can be used, it came down to using the Edge TPU, the pycoral library, and its datasets, setting up the library was the first challenge. As we were using just the accelerator, there was a lack of resources as mostly everyone used other processors and devices like the Raspberry Pi. Working with image-based datasets didn't require such devices. So, using Git-bash, and the Python version that met the requirements, we set up the TPU and successfully cloned the pycoral library. There were different kinds of environments where an image can be trained to achieve different kinds of outcomes. We chose to go for Semantic Segmentation and Object Detection examples. The test-data in the repository consisted of basic images which were easy-to-recognize, but the way Segmentation and Object Detection worked on these test-data was phenomenal. However, the test-data did not have any kind of images that would help us with navigation. So, we tried to get some pictures from google and test them out. As these images were not so good in quality and lost quality even more after downloading them, we were not expecting crisp outcome from it, like the test-data. So, we went with the best ‘road-based’ images we could get. The outcome was very surprising because a few images we trained had very close to a perfect outcome, whereas other images we trained had very weak outcome. As you can see in Table 1, one case of training stood out, because when we trained the same image for segmentation and object detection, the outcomes went to an extreme. For Segmentation, the outcome was very poor, but the training went exceptionally well for Object Detection for the same image.

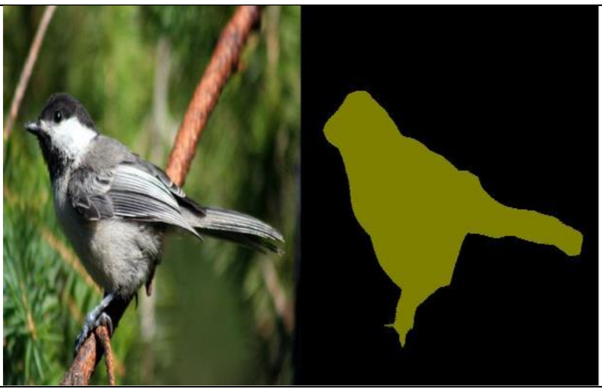
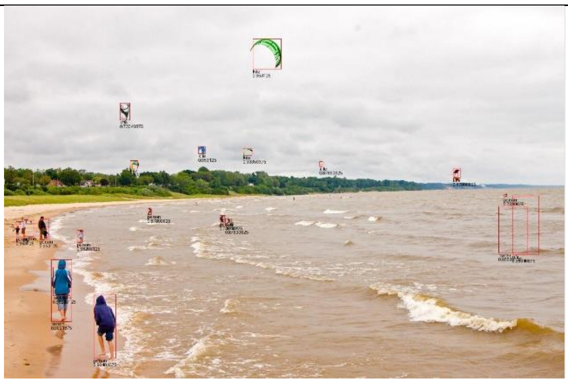
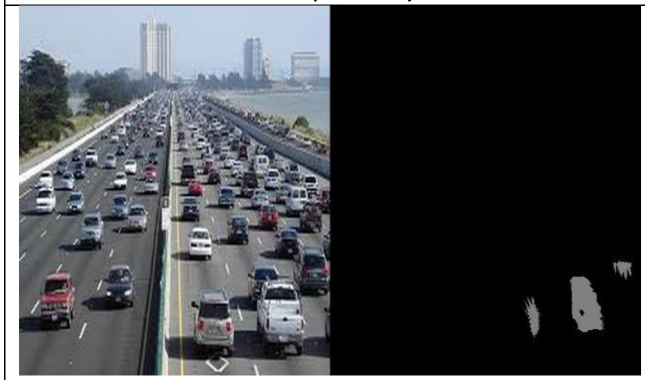
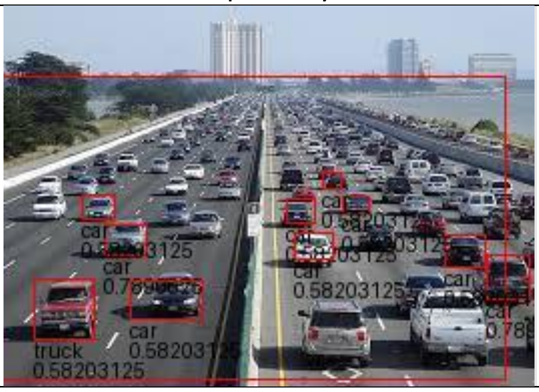
	
Semantic Segmentation with the test-data in the repository.	Object Detection with the test-data in the repository.
	 car 0.58203125 car 0.7890625 car 0.58203125 truck 0.58203125
Semantic Segmentation with the downloaded image data.	Object Detection with the downloaded image data.

Table 1

Conclusion

The images used from the repository's 'test-data' and the images imported from the internet had different outcomes. Considering the quality difference of those images, understanding the images on a pixel-level would be hard for a machine, which explains the difference we got by training the same image for different models, as segmentation is done on a pixel-level. Therefore, some images in test-data useful for navigation would be a great help to determine the accuracy of pycoral library and the Edge TPU.

Future Work

Reinforcement Learning agents prove to be very successful when it comes to learning and gaining the maximum number of rewards. If rewards are set to navigate safely, RL can be very useful to speed up the process of achieving autonomous driving. Feeding information in form of segmented images can help navigate and object detection can help get information of the surroundings at the same time, which can be a very helpful asset in space, considering data of stars, other planes and more are fed into the agent and trained respectively.

References

1. (n.d.). Apollo Scape. Retrieved August 9, 2024, from https://apolloscape.auto/
2. *Deep Learning for Siri's Voice: On-device Deep Mixture Density Networks for Hybrid Unit Selection Synthesis*. (n.d.). Apple Machine Learning Research. Retrieved August 9, 2024, from https://machinelearning.apple.com/research/siri-voices

3. *Designing AI Chips with AI: Reinforcement Learning for Arithmetic Circuit Design*. (n.d.). NVIDIA. Retrieved August 9, 2024, from https://www.nvidia.com/en-us/on-demand/session/gtcspring22-s41622/
4. *Examples*. (n.d.). Coral. Retrieved August 9, 2024, from https://coral.ai/examples/
5. G. Humblot-Renaux, L. Marchegiani, T. B. Moeslund and R. Gade, "Navigation-Oriented Scene Understanding for Robotic Autonomy: Learning to Segment Driveability in Egocentric Images," in IEEE Robotics and Automation Letters, vol. 7, no. 2, pp. 2913-2920, April 2022, doi: 10.1109/LRA.2022.3144491.
6. *Introduction to RL and Deep Q Networks*. (2023, September 26). TensorFlow. Retrieved August 9, 2024, from https://www.tensorflow.org/agents/tutorials/0_intro_rl
7. Jain, S. (2023, September 25). *A Beginner's Guide to Deep Reinforcement Learning*. GeeksforGeeks. Retrieved August 9, 2024, from https://www.geeksforgeeks.org/a-beginners-guide-to-deep-reinforcement-learning/
8. Li, J., Yin, J., & Deng, L. (2021, May 17). *A robot vision navigation method using Deep Learning in Edge Computing Environment - EURASIP Journal on advances in Signal Processing*. SpringerOpen. https://asp-eurasipjournals.springeropen.com/articles/10.1186/s13634-021-00734-6
9. NASA. (n.d.). *Orion Optical Navigation Image Processing Software (OPNAV)(MSC-26456-1)*. NASA. https://software.nasa.gov/software/MSC-26456-1
10. Nguyen, A. (2021, July 30). *15 Best Open-Source Autonomous Driving Datasets | by Alex Nguyen | Analytics Vidhya*. Medium. Retrieved August 9, 2024, from

https://medium.com/analytics-vidhya/15-best-open-source-autonomous-driving-datasets-34324676c8d7

11. *9 Real-Life Examples of Reinforcement Learning*. (2022, September 27). Online Business Courses at Leavey School | SCU Online. Retrieved August 9, 2024, from https://onlinedegrees.scu.edu/media/blog/9-examples-of-reinforcement-learning
12. *Self-Driving Cars With Convolutional Neural Networks (CNN)*. (2024, April 22). neptune.ai. Retrieved August 9, 2024, from https://neptune.ai/blog/self-driving-cars-with-convolutional-neural-networks-cnn
13. *Taxi*. (n.d.). Gymnasium Documentation. Retrieved August 9, 2024, from https://gymnasium.farama.org/environments/toy_text/taxi/#taxi
14. *What is a convolutional neural network (CNN)*. (n.d.). Arm. Retrieved August 9, 2024, from https://www.arm.com/glossary/convolutional-neural-network
15. *What is Reinforcement Learning? - Reinforcement Learning Explained*. (n.d.). AWS. Retrieved August 9, 2024, from https://aws.amazon.com/what-is/reinforcement-learning/