GENG5512 MPE Engineering Research Project Part 2

Final Report

Enhancing Camera Based Autonomous Navigation of Shuttle Bus via Policy Pre-Training with Self-Supervised Learning

Ivan Sossa Gongora
22964938
School of Engineering, University of Western Australia

Supervisor: Thomas Bräunl
School of Engineering, University of Western Australia

Word count: 8109

School of Engineering
University of Western Australia

Submitted: 15/10/2025

DECLARATION OF CONTRIBUTION

My contribution

This investigation was an experimental/modelling project where state-of-the-art pretraining works were modified and tested on the nUWAy2 shuttle bus in Eglinton. My contribution was not to the actual design of these algorithms and techniques, but their application to this specific use case. This involved:

- Creating custom models that were compatible with the pretrained encoders.
- Modifying and extending these models for the downstream Eglinton navigation task.
- Creating my own pretraining dataset for the pretraining scripts.
- Modifying the pretraining scripts for compatibility with my own dataset.
- Creating my own validation visualisation maps such as saliency heat maps and depth maps.
- Collecting and analysing the results.

The other component of this investigation was to utilise existing software by current researchers in the REV Project to be able to test the produced models for performance in the setup at Amberton Beach. These researchers have functional model training, simulation, visualisation, validation and shuttle running scripts that use tools, such as PyTorch, that are compatible with my models. I therefore did not create my own version of this workspace and instead made minor modifications to this software for integration and compatibility with my own models, visualisations and training procedure.

Use of AI tools

AI tools were used solely for general research purposes and to check and improve the quality of written English in this report. No AI tools were used to generate any of the report's content.

In accordance with University Policy, I certify that:

The above information is correct, and the attached work submitted for assessment is my own work and that all material drawn from other sources has been fully acknowledged and referenced.

Student signature  Date 13/10/2025

Supervisor confirmation
To the best of my knowledge, the student's contribution outlined above is correct.

Supervisor signature  Date 14/10/2025

Abstract

This project investigates self-supervised pre-training methods to improve the robustness and generalisation of camera-based navigation for the nUWAy autonomous shuttle operating in Eglinton, Western Australia. Current end-to-end navigation models rely on supervised learning and are highly sensitive to camera placement and environmental changes, requiring costly and time-consuming manual data collection for each new route. To address this, state-of-the-art pretraining methods, such as Policy Pre-Training via Geometric Modelling (PPGEO) and Monodepth2, were adapted and applied to the shuttle's grayscale camera system. A custom Australian driving dataset was created to align pre-training with local suburban environments, as existing international unlabelled datasets are less suitable for the driving policy in Australia.

The investigation involved integrating the PPGEO pipeline with the shuttle's navigation architecture, executing depth and policy pre-training, fine-tuning multiple encoder-freezing strategies, and evaluating performance through training loss curves, saliency visualisations, simulation tests, and initial real-world shuttle deployments. Results showed that policy pre-training with official depth and pose models demonstrated stable convergence and formed a strong foundation for downstream training, while depth pre-training quality was limited by dataset scale and noise.

This work demonstrates the feasibility of adapting self-supervised pre-training methods to real shuttle deployments and establishes a framework for future research into scalable, data-efficient navigation systems.

Acknowledgements

I would like to acknowledge, first and foremost, my mother, father, and sister, whose unwavering support throughout this demanding year has been invaluable. Their patience and understanding have not gone unnoticed. I would also like to express my sincere gratitude to my supervisors and colleagues from the UWA REV team for their guidance, mentorship, and constant support. In particular, I wish to thank Zhihui Lai, Kieran Quirke-Brown, Lee Le, and Thomas Bräunl for their incredible support and encouragement.

The Eglinton Shuttle has existing software for training, model running, data/results generation and the like. These are all excellent pieces of work that most of my generation relied upon. I want to especially acknowledge Kieran Quirke-Brown and Zhihui Lai as the authors of the majority of the software which allowed this investigation to take place and succeed.

Contents

List of Figures

Nomenclature

ACO	Action-Conditioned Contrastive Policy Pretraining
CAM	Class Activation Map
CBD	Central Business District
CNN	Convolutional Neural Network
DepthNet	Depth model architecture used in project
DNN	Deep Neural Network
E Stop	Emergency Stop
Epoch	Full training iteration over a dataset
LR	Learning Rate
MAE	Mean Absolute Error
nUWAy	REV Project Shuttle bus
PoseNet	Pose model architecture used in project
PPGEO	Policy Pre-Training via Geometric Modelling
PyTorch	Python tool used for DNN training
ResNet	Visual Encoder architecture used in project
RGB	Red-Green-Blue

1. Introduction, Literature Review and Project Objectives

1.1 Problem Statement

The Renewable Energy Vehicle (REV) project has developed an autonomous shuttle bus that is in its final stages of approval for public road trials. This shuttle was an expansion of the scope of this research team's campus driving and involves driving in real traffic through a pre-planned circuit route in a roundabout-style suburban residential area in Eglinton, one of Perth's northern suburbs. This route is intended as an alternative transport option for the public throughout this suburb and all the way through to its point of interest, Amberton Beach [1]. After research and experimentation on various approaches to navigation, the camera based end-to-end method has proven as one of the best performing and most practical options for the hardware set up and available resources in the project [1].

Figure 1: Eglinton route map and scene examples. [2]

This autonomous navigation method involves deep learning models being trained to directly map raw grayscale camera sensor input to driving commands using deep neural networks (DNNs). This method reduces the complexity of the system, the amount of hardware required and the computational load on the bus [2], which all contribute to its success in the current Eglinton route.

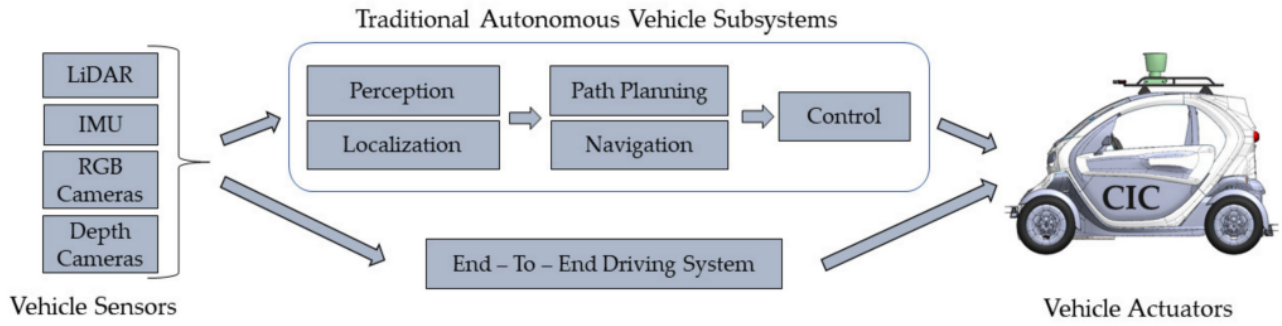


Figure 2: Two main DNN options in autonomous current autonomous navigation systems. [3]

However, the major downside of this simpler method is its reduced interpretability, causing the models to struggle to generalize beyond their training scenarios. This is also because learning a robust visuomotor policy from scratch is notoriously data-inefficient [4]. The setup in Eglinton

currently works due to its pre-defined route that it was trained on, but it proves brittle under moderate changes in lighting, camera placement, or extending it to a new route. Such changes have forced retraining of the models, which rely on fully supervised learning. This is where training maps camera images to their corresponding ground truth vehicle controls using labelled examples, and thus inherently depends on large, annotated datasets. This dataset currently can only be obtained by the manual driving of an operator in the shuttle through the environment and route that needs to be learnt, hence the time consuming nature of the process. Collecting such data for each new deployment or change is therefore costly and impractical.

These limitations have implications for multiple stakeholders. For researchers, this motivates the need for new neural network training approaches if the camera based-end-to-end models are to be scaled beyond their current context. The bodies that fund the transportation of the buses, such as the WA department of transport, face increased costs, as each new route or condition may necessitate data collection and model retraining. Ultimately, scaling autonomous driving will require data-efficient training methods so that vehicles can adapt to new environments without extensive manual labelling.

1.2 Project Background

One of the most promising and novel approaches to this problem is the use of algorithms and techniques that automate the production of the dataset labels from raw input data. The extended literature review in appendix A encapsulates theoretical background and framework of the main techniques investigated in this project. Methods such as Policy Pre-Training via Geometric Modelling (PPGEO), Action-Conditioned Contrastive Learning (ACO) [5], and Monodepth2 have demonstrated that geometric and motion cues can be learned from raw video without manual labels. PPGEO, for example, pretrains visual encoders by predicting depth and ego-motion across frames, while ACO learns policy-related features through contrastive learning with pseudo-action labels. Monodepth2 [6] focuses on self-supervised monocular depth estimation to extract 3D scene structure from video. These approaches have been shown to improve downstream driving performance in simulation, yet few have been evaluated on real shuttle deployments in changing real-world environments. The benefit with this type of data is its abundance and free availability on the internet from various sources, meaning that obtaining a large dataset becomes a simple matter of searching and browsing.

1.3 Project Objectives

This project is intended to extend the state of the art by testing the practical value of such pretraining techniques on real, physical platforms in public roads.

The primary objective of this project is to investigate self-supervised pretraining methods that can enhance the performance, robustness, and generalisation of the camera-based navigation system used on the nUWAy autonomous shuttle bus in Eglinton.

The working **hypothesis** is that these self-supervised pretraining techniques trained on large-scale unlabelled video will generalise more effectively and require fewer labelled examples during downstream training.

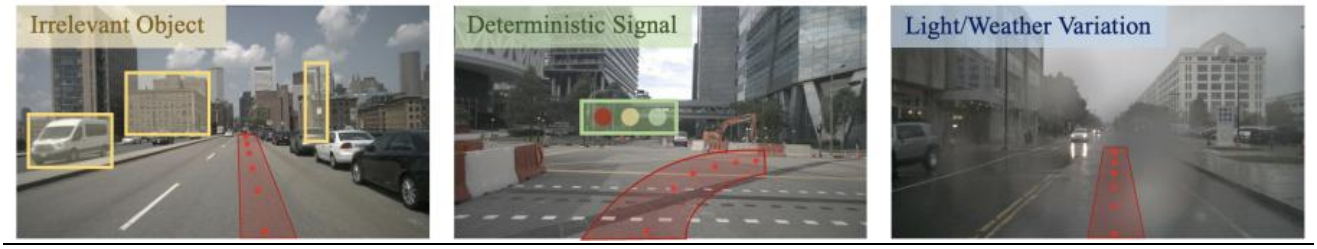


Figure 6: Examples of Visuomotor driving policy features that are learnt by self-supervised pre-training methods such as PPGEO [4]

Performance will be evaluated through a combination of offline and real-world metrics. Offline validation will compare training and validation loss between pretrained and baseline models to assess sample efficiency. Qualitative analyses will include saliency maps to visualise model attention, and depth maps to confirm that pretrained models better capture scene geometry. The most promising navigation models will also be deployed for live driving trials on the Eglinton shuttle. During these tests, key performance indicators will include the number of emergency stops (E stops), manual interventions, and the ratio of autonomous to manual driving time. Route-tracking data will also be analysed to identify where and under what conditions the pretrained models outperform or underperform the current baseline.

This research contributes to the broader work of the Renewable Energy Vehicle (REV) project at the University of Western Australia, which aims to advance camera-based end-to-end navigation for public-road autonomy. The findings will help determine whether self-supervised pretraining can reduce the need for large, labelled datasets, thereby saving significant time and resources for both the REV research team and the Western Australian Department of Transport, which supports the program. Improved robustness would reduce the frequency of retraining when expanding routes or adapting to new environments, directly lowering operational costs. In the longer term, this project lays the groundwork for future studies into scalable self-supervised learning pipelines and contributes to the developing safe, efficient, and sustainable autonomous public transport.

2. Model Formulation

The methodology of this investigation was structured as a progressive modelling and implementation process, as preexisting state-of-the-art deep neural network models were utilised, adapted and combined to create tailored models for the specific downstream task of the predefined Eglinton route navigation. It can also be considered an experimentation investigation in that several models from differing pretraining techniques were tested and compared for their performance. This work comprised of five main stages:

1. Evaluation of existing state-of-the-art pretrained models within an Eglinton simulation environment to establish a performance baseline.

2. Development of a custom unlabelled driving dataset designed to better capture the visual and contextual characteristics specific to the Eglinton route.

3. Self-supervised pre-training and simulation testing of models using the newly created dataset to assess the influence of domain-specific data on learned representations.

4. On-vehicle validation of all trained models through controlled navigation trials on the real nUWAy shuttle bus.

5. Comparative analysis of the resulting models, including cross-comparison between different pre-training strategies and against a baseline model trained without pre-training.

The following sub sections will chronologically step through these stages and describe the derivations, modifications, validations, software tools used, input options and limitations of all of the differing models that were experimented on.

2.1 Model Derivation for Eglinton Navigation

The investigation began with exploring the effectiveness of the PPGEO pretraining pipeline on the navigation downstream task, as well as its improvement in image feature attention. This paper conducted its pretraining on a specific model architecture, the ResNet-34 visual encoder. As shown in Figure 3 below, this is a convolutional DNN (CNN) that dynamically receives an input image Red-Green-Blue (RGB) colour image and outputs a feature map which indicates the features in the input image that the model focuses on. This identical encoder architecture was used to ensure compatibility and to enable fair comparison of encoder performance during fine-tuning on the Eglinton dataset.

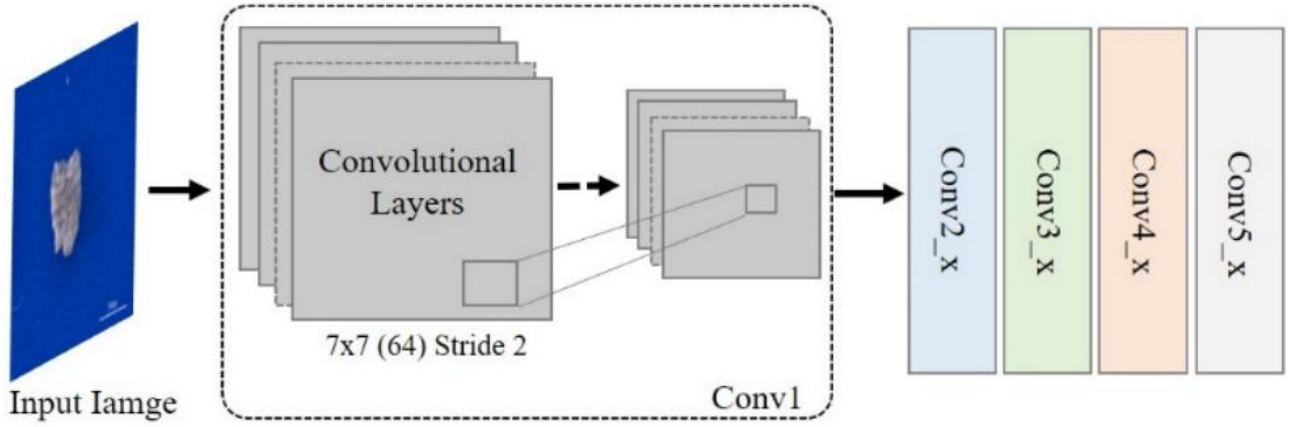


Figure 3: Illustration of ResNet model structure [7].

There was however one tweak to the first layer, as the Eglinton shuttle operates on grayscale camera input for reduced latency and processing overhead, meaning that the 3-channel image input was reduced to one channel, and any pretrained encoders loaded into the Eglinton model must have its 3 channel conv1 channel averaged for compatibility. This minor adjustment was not suspected to have any significant effect on the encoder performance, as it just means that it cannot learn image representations based on colour, and no driving policy required for Eglinton navigation requires colour identification, hence the current grayscale camera setup. It should be noted that this reduction does have a limitation on potential generalisation to all traffic conditions, as traffic lights require colour recognition, however there are no plans for the Amberton beach trials to expand its scope to routes involving traffic lights in the foreseeable future.

To build the full navigation for Eglinton, this visual encoder must be expanded to include steering outputs. One of the simplest approaches is to use a straightforward mapping from the output of the encoder to the input of a navigation decoder, with the intersecting layer being the feature map that is output by ResNet-34. Figure 4 below shows this top-level system architecture for such a model design, which was adopted in this investigation.

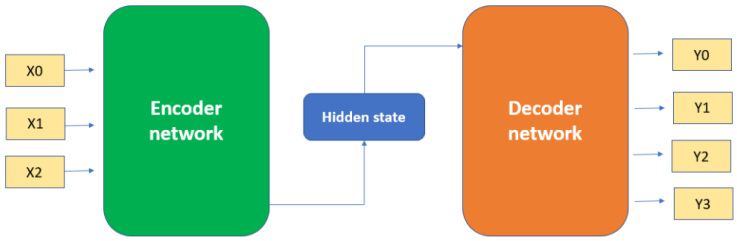


Figure 4: Basic structure of an encoder-decoder DNN architecture [8].

The decoder part of this model must therefore reduce and simplify to two outputs: speed and steering. The hidden layers chosen for this decoder are based on standard modern architectures of navigation heads that attach to CNNs. Figure 5 illustrates the functioning of this generic structure.

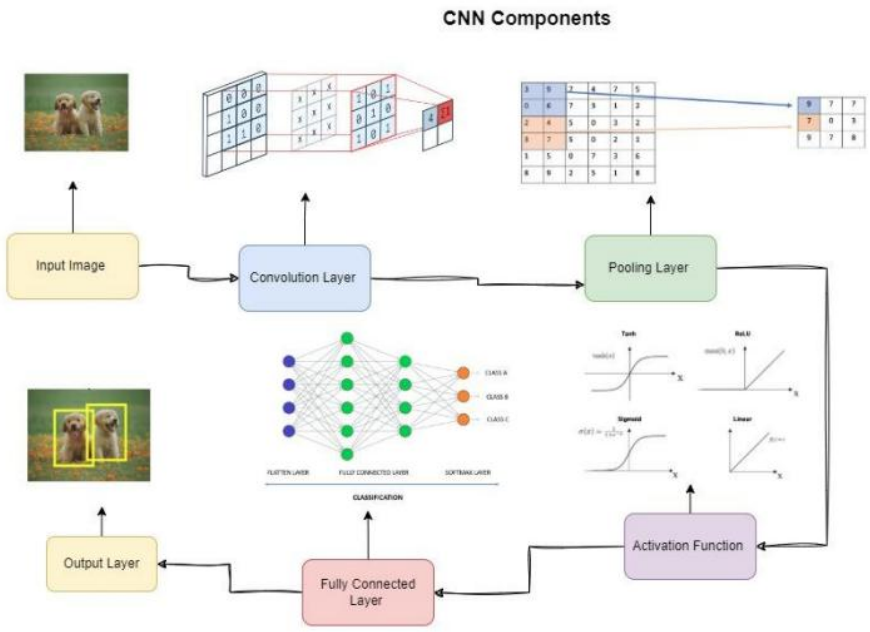


Figure 5: Basic architecture template of the decoder of a CNN [8] .

2.2 Adaptation of Eglinton Train and Simulation/Validation Scripts

To enable fine-tuning and evaluation of the customised navigation model, the existing Eglinton training scripts were adapted to ensure full compatibility with the modified encoder-decoder architecture. These training scripts, previously validated on functional end-to-end models for the shuttle's camera system, were configured to effectively train the model using the labelled Eglinton dataset. Dataset loading, preprocessing, and augmentation routines were standardised to preserve consistency across experiments and to facilitate reproducible fine-tuning runs.

A notable addition to the functionality of the training script was the customization of the model layer freezing options. This modification was explored due to the custom RGB-grayscale averaging that was applied to pretrained models before fine-tuning. There were concerns that a simple averaging would keep the early encoder layers from adapting to grayscale input, as the PPGEO conducted pretraining on an RGB dataset and allowing these layers to be trained on the grayscale dataset would potentially be beneficial to the model performance. An option was also added to unfreeze the entire visual encoder, as this is how current functional shuttle models are fine-tuned in Eglinton. Although keeping the entire encoder unfrozen ruins the point of encoder pretraining, it was thought to provide a good comparison to demonstrate the difference between the effect of supervised and self-supervised encoder training. Keeping the encoder unfrozen could also prove an optimal unison between both methods, as pretraining could mean less fine-tuning (less epochs or smaller dataset) is required [9]. The final comparison would be to fine-tune a model with ImageNet weights (default PyTorch visual encoder weights, useful for generic image detection) as a baseline to measure the effect of PPGEO pretraining in general. To summarise, the four models for comparison would thus consist of:

1. **No Pretraining** – the encoder was initialised with default PyTorch ImageNet weights and trained from scratch on the Eglinton dataset, serving as a baseline.
2. **PPGEO Frozen** – the encoder weights from the official PPGEO pretraining were loaded and fully frozen, ensuring that the geometric representations learned from large-scale driving data were preserved during navigation training.
3. **PPGEO Unfrozen** – the same pretrained weights were used, but all encoder parameters were unfrozen to allow full gradient updates and adaptation to the Eglinton environment.
4. **PPGEO Partially Frozen** – all encoder layers except the early convolutional blocks (*conv1* and *layer1*) were frozen. This configuration enabled limited adaptation to grayscale image inputs while retaining the deeper geometric priors of the pretrained model.
5.

These freezing strategies were implemented as switchable configurations within the training pipeline, allowing consistent experimental control while analysing the contribution of pretrained feature representations to navigation robustness and generalisation.

The simulation and validation scripts were likewise adapted to operate with the modified network outputs. These scripts dynamically iterated through sequential Eglinton image data to simulate real-time driving conditions, displaying the model's predicted steering commands alongside the corresponding ground-truth control signals for visual comparison as shown in Figure 6. This is done by plotting a vector to display the relative speed (-1 to 1) and steering angle ($-180° \; to \; 180°$), as demonstrated in Figure 7. This simulation stage functioned as a crucial intermediate validation step, allowing model behaviour, such as consistency, response delay and smoothness, to be observed safely in a virtual environment before deployment on the physical shuttle.

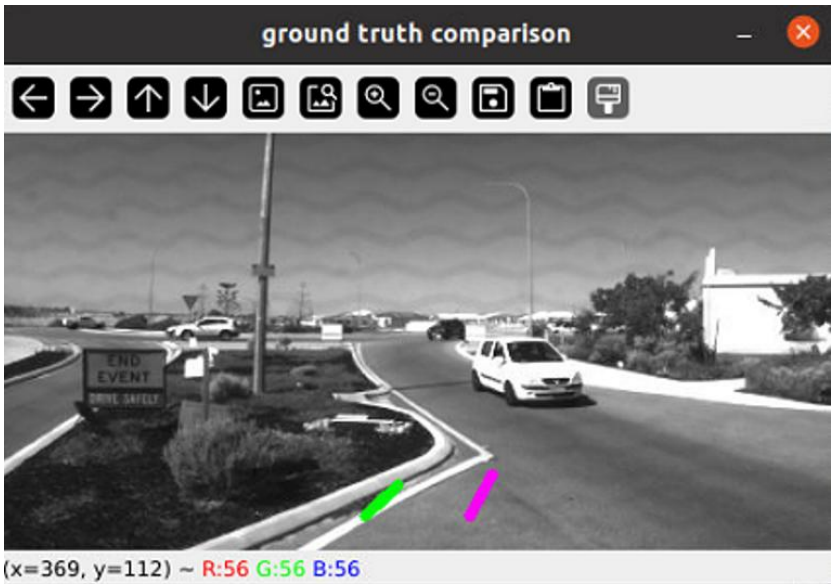


Figure 6: Example of Eglinton Navigation Command Simulation

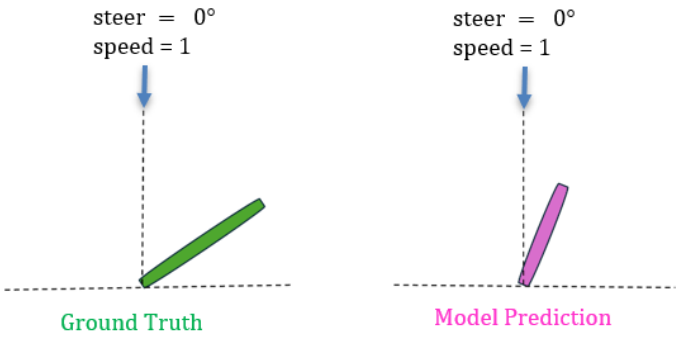


Figure 7: Custom Visualisation of Relative Driving Outputs on Shuttle Bus

Further adaptation was required to restore compatibility between the saliency-map visualisation functions and the new encoder structure. The original scripts employed gradient-based saliency methods that were incompatible with the updated model interface and did not align with the attention-map methodology used in the PPGEO paper. To address this, a custom implementation of the Eigen-CAM [9] algorithm was integrated to generate feature-attention heatmaps from the ResNet-34 encoder's final convolutional layer, as shown below in Figure 8 . These visualisations highlighted the spatial regions of each input frame most influential in the model's steering and speed decisions. This interpretability mechanism served as a key validation tool for assessing model generalisation, enabling analysis of whether the encoder focused on relevant driving cues (such as lane boundaries and obstacles) or exhibited signs of overfitting to dataset-specific artefacts.

Figure 8: Visualisation of custom Eigen-CAM heatmap created. It follows a traditional heatmap colour scheme where red indicates areas of the model's highest focus, whilst blue indicates minimal attention.

These scripts were executed on a workstation with an NVIDIA GeForce RTX 3060 on which the sorted Eglinton dataset was already set up from successful existing model implementations. PyTorch was chosen as the neural network software tool for compatibility with the PPGEO pipeline.

Functionality for plotting the training and validation loss over the course of the epochs was already existing in these training scripts, which utilised mean absolute error (MAE) loss, and are automatically logged for display and analysis.

2.3 Creation of Custom Unlabelled Dataset

While the original PPGEO model was pre-trained on diverse global driving data, its environmental features and traffic regulations differ from Australian road scenes. It utilised a dataset compiled by the ACO pretraining paper, in which raw online YouTube recordings of high quality, dash-cam-like footage throughout various cities worldwide were exploited for their abundance and relevance in driving policy. ACO acquired this dataset using a singular, YouTube channel "J Utah" as it uploads long, high quality, diverse videos from around the world. An example of these videos is shown in Figure 9 below. Although this channel has uploaded a number of videos in the major cities in Australia, none of these videos were used in the pretraining dataset. In fact, majority of the dataset was based in highly dense central business districts (CBDs) from major international cities. This type of scenery does not reflect the suburban scenery of Eglinton. More generally, this dataset would not be useful for a shuttle operating in Australia as the driving policy, such as the side of the road driven, is contradictory in most countries.

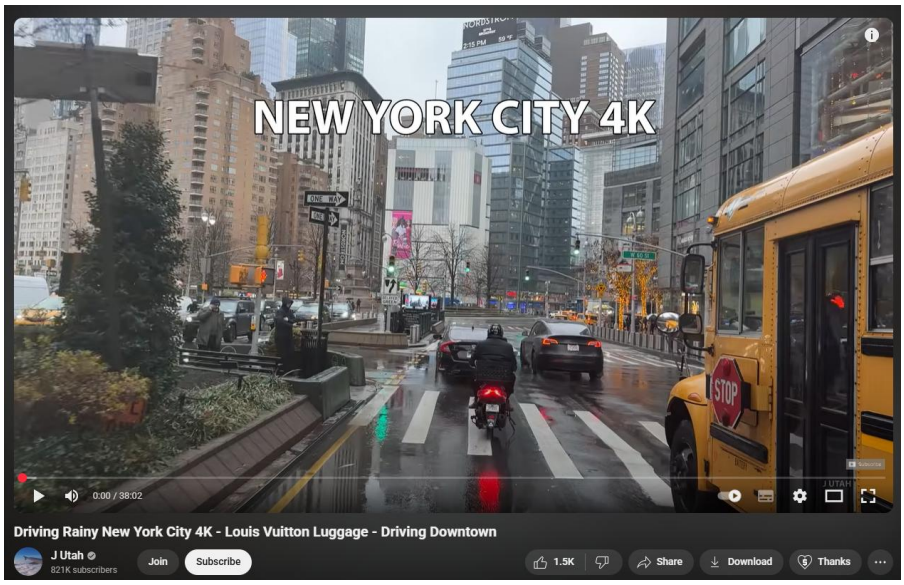


Figure 9: Example of a YouTube video in the dataset [10].

To assess the influence of geographic and visual domain alignment, a custom pre-training dataset was proposed, consisting exclusively of Australian driving footage, and more specifically, footage suburban footage in Perth that matched the type of scenery present in a Suburb such as Eglinton.

The objective was to determine whether self-supervised geometric pre-training on region-specific data could yield an encoder better suited to the nUWAy shuttle's operating environment.

A curated list of publicly available Australian driving videos was compiled from YouTube. Selection criteria included:

- Forward-facing dashboard or roof-mounted perspective
- Stable 1080p recording with minimal occlusion
- Representation of varying illumination and weather conditions
- Urban and suburban driving consistent with Perth and Eglinton environments

These videos would have their frames extracted at 1 Hz which allow consecutive frames to show sufficient relative motion whilst not being so distant that no motion between the frames could be inferred [4]. The frames were organised into the directory structure expected by the PPGEO training scripts to ensured compatibility with the existing data loader modules without altering the PPGEO source code.

The extraction originally was written to also convert the frames to grayscale images, which was later removed and instead, a colour transformation parameter RandomGrayScale(p = 0.2) had its probability increased to 0.5 so that the pretraining pipeline could operate as intended, while learning on a greater amount of grayscale images.

The frame in Figure 10 is an example product of this process. Although they appear poor quality, this is consistent with the resolution recommended in ACO. An example of a frame from ACO is also provided in Figure 11, which shows the same quality and frame driving view as the examples from the custom dataset.

Figure 10: converted dataset example

Figure 11: ACO dataset example [5].

2.4 Custom PPGEO Pretraining

The official PPGEO repository's self-supervised pre-training pipeline was executed on the custom dataset. Stage 1 trained the DepthNet and PoseNet to convergence over 20 epochs.

Stage 2 froze these networks and trained a ResNet-34 visual encoder to predict ego-motion from single frames for 20 epochs.

The outputs of this process were three model checkpoints: a pre-trained depth decoder, a pose encoder, and a PPGEO ResNet-34 encoder.

The newly trained ResNet encoder was imported into the Eglinton navigation pipeline following the same four encoder-freezing configurations defined earlier. The remaining network components and training parameters were unchanged to ensure direct comparability.

All checkpoints, logs, and dataset metadata were archived for reproducibility and future reuse in cross-domain transfer learning experiments.

2.5 Remaining Model Validation Tests

At this point, there have been various navigation models created that are ready to be compared for performance. Ultimately, the greatest possible validation is physical deployment on the shuttle bus, as simulations are limited in what they can show. It cannot be determined with certainty from the driving output vectors whether the driving commands of a model would lead to a crash or poor navigation in general, as the sequential frames from the simulation script are only taken from the ground-truth data. The model outputs give a general idea of how it would attempt to drive, but the simulation does not propagate the vehicle's next pose based on the model's outputs.

Training and validation loss curves, although more definitive for performance analysis, are also not entirely representative of how a model navigates on the actual shuttle, as a quality drive must also be smooth, consistent, and follow a natural driving style. It is therefore also insufficient to conclude model performance based on the MAE loss curves alone.

Saliency visualisation maps however are different in that they serve as a complement to the other validation methods. They are a qualitative measure of model performance as absolute error cannot be defined in what a model focuses on within an image. This quality can only be interpreted and compared to how a how a human driver focuses on visual features while navigating the road. Since this is a subjective matter, the saliency map does not determine objective performance but assists in gaining a better idea of the improvement in generalisation of the visual encoder. This is because it can be said that a more generalized model should focus on visual features that are relevant to driving decisions such as road boundaries, cars, pedestrians or lane markings. A model with low validation loss but random saliency can therefore be said to have overfitted to the training data rather than having learned an appropriate driving policy that makes it robust to changes in the input environment. Figure 12 shows a simple example of the effect of this overfitting to irrelevant parts of a trends in a dataset. The two major validation metrics used in the results are therefore real shuttle performance in conjunction with the Eigen-CAM heatmaps produced by the visual encoders of each model.

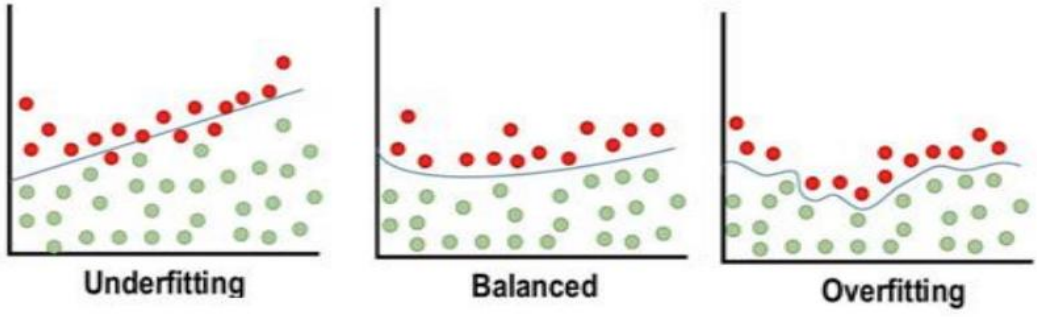


Figure 12: Visualisation of the effect of a CNN model overfitting to a specific dataset [8].

To test the models on public roads in Eglinton, the CUDA PyTorch pipeline had to be integrated into the existing ROS2 architecture on the shuttle. This is a comprehensive and well maintained system which consists of many modularised nodes that work to ensure the end-to-end DNN navigation is run at a high level of quality, safety and reliability. This system is also designed to be

able to easily log many forms of data that are needed to analyse the performance of the models' navigation at each point in time, such as:

- Manual Interventions
- E Stops
- Autonomy %
- The model being run
- The model's output speed
- The GPS location of the data point

The existing PyTorch scripts of the previously functioning camera models were therefore modified and adapted for compatibility with the newly created models and the shuttle was run in autonomous mode under supervision by trained personnel. The recorded and saved data was then converted into specialised HTML map visualisations that display the route navigated by the bus in for each model's run. This route was plotted with varying legends to display differing performance metrics. The crucial metrics that were chosen for this validation were:

- **E Stops:** Where the run caused the low-level safety Lidars to trigger an emergency stop
- **Manual Interventions:** Where the manual controller override was triggered by the shuttle operator due to concern that the model's navigation will lead to an unwanted manoeuvre or an eventual crash.
- **Autonomy %:** Indicates relatively how often a model navigated a section of the route autonomously.
- **Average shuttle speed**
- **Slow node:** Shows where slow nodes were triggered on the route. Additional to the low level E stop, the navigation has higher level ROS2 based slow node which serves as an intermediate precaution to prevent too many high-speed E stops, which simply forces a speed output down scaling. Slow node engagements therefore can reflect poor model performance, however not as poor as an E stop.

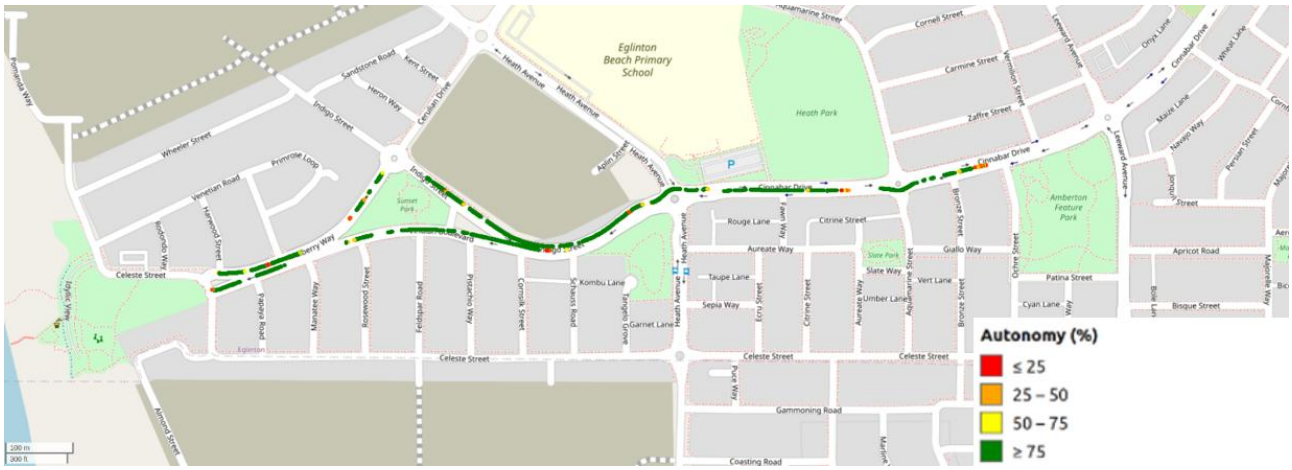


Figure 13: Example of output map used for real shuttle navigation

An example of this output map is shown in Figure 13 above. These metrics were determined to provide sufficient preliminary indications as to whether the pretrained models show potential for navigation performance. Due to time constraints, the models were not trained over the same number of epochs as existing functional shuttle models are typically trained on, however a consistent comparison over 10 epoch between the following epochs was deemed to provide fair validation:

- PPGEO pretrained (Frozen Encoder)
- PPGEO pretrained with customised dataset (frozen and unfrozen)
- ImageNet pretrained with unfrozen encoder

Here, the ImageNet pretrained model serves as the baseline model that effectively does not utilise any PPGEO pretrained but rather a standardised visual encoder pretraining. Since this encoder has learnt no driving policy, it must be trained in unfrozen mode to allow full supervised learning on the labelled Eglinton dataset. The intent is to determine how a frozen pretrained encoder (thus has not learnt on any Eglinton labelled data) performs on real navigation compares to this baseline encoder that represents current models that do not use any self-supervised pretraining.

2.6 Depth Map Production

Stage 1 of the PPGEO pretraining pipeline outputs Depth and Pose encoders and decoders which are then used for self-supervised learning in stage 2 for driving policy learning. Although PPGEO utilises these as auxiliary models in an intermediate step, they can also produce useful outputs for the nUWAy2 system. In particular, the DepthNet model, which is pretrained following the Monodepth2 framework, can produce meaningful depth estimations of an input image which could be useful in future work for camera-based obstacle avoidance. The performance of this model can be visualised using a depth map such as the one shown in Figure 14 below from in Monodpeth2 [6].

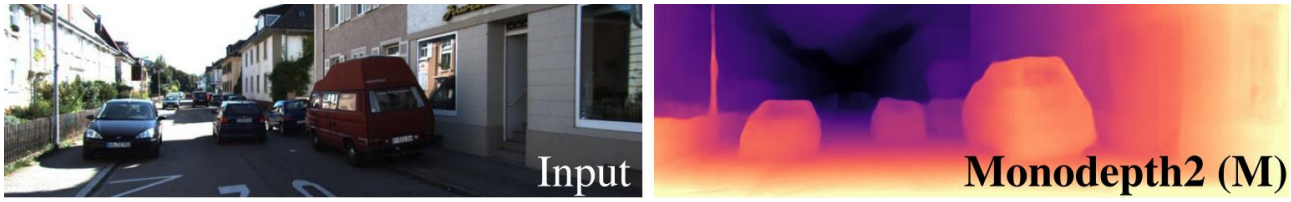


Figure 14: Example output from Monodepth2 depth map visualisation [6].

In this use case of pretraining, it cannot be hypothesised that the custom unlabelled dataset will provide a meaningful improvement to the depth predictions, as depth is universal and independent of driving location, unlike driving policy. Despite this, pretraining outputs of the following datasets were simulated:

- The custom dataset
- The original PPGEO dataset
- The custom dataset on top of the original dataset

It was hypothesised that the significantly larger original PPGEO dataset would provide better results; however, it was still seen as useful to investigate the performance of the custom dataset for further validation of its quality in varying pretraining schemes.

3. Results and Discussion

3.1 Custom Dataset

Obtaining a large unlabelled dataset for unsupervised learning via freely available online media has its benefits in convenience, however it does come with some quality and reliability costs. There are a number of YouTube channels found that appeared consistent, high quality and relevant to the models being trained, however perfection is difficult to guarantee. Upon further inspection of some of the videos used (usually around 30 minutes to 1 hour in length), there are small sections in the dataset that are irrelevant and create noise in the model. Figure 17 below is an example of this, where the driver parks his car at a POI and begins to film the view, thus not driving or providing any meaningful driving example to the model. Such noises in the dataset are difficult to filter for as the intention of the unlabelled datasets is that they are very large. It is therefore impractical and counter-productive to the benefit of the pretraining to have to manually filter and sort the data. The noisy frames were therefore kept in the dataset which could lead to sub optimal results.

Another limitation that was identified in the dataset was the presence of irrelevant features in the frame due to the camera set up in the vehicle, causing parts of the vehicle such as the dashboard, rear-view mirror or windshield wipers, to be visible. An example from the custom dataset is shown in Figure 18. This was also observed to cause issues with the resulting encoders that were trained.

Overall, the dataset that was obtained largely represents the type of input data received throughout the Eglinton route, and crucially beyond this route, as this opens the opportunity for the models to generalize if the route is expanded in the future. The side-by-side example below in Figure 18 and Figure 19 demonstrates this, despite having the vehicle noise. Figure 16 is a great example of a frame extract which is very similar to the type of roundabout inputs seen in Amberton Beach such as in Figure 15. This is the benefit of the custom dataset, as many of the videos were selected from footage driven in Perth suburbs that appeared similar to Eglinton.

The other downside of this dataset selection is that although the internet provides an increasingly overwhelming amount of data, it was still difficult to obtain a quality dataset that was comparable in size to the worldwide dataset used in the original paper. At an international level, it is trivial to find over 100 hours of near perfect footage, however in the custom dataset compilation, at approximately the 20-hour mark, the remaining videos that could be found began decreasing in quality and usability, thus were not included. This resulted in a far smaller dataset, approximately 5 times smaller. It was seen as acceptable to have a much smaller dataset since the model were known to only be intended for a specific suburb, however the pretraining pipeline may not have been given sufficient learning data to build an acceptable policy. These concerns will be confirmed in the following sections.

Custom Dataset

Figure 16: Roundabout image from custom dataset

Figure 17: Unwanted image from custom dataset

Figure 18: Lane following image from custom dataset

Eglinton Dataset

Figure 15: Raw image input from Eglinton setup at one of the routes typical roundabout scenes

Not Relevant for Eglinton

Figure 19: Raw image input from Eglinton setup at a typical lane following scene

3.2 PPGEO Depth Pretraining

The first attempt of training a DepthNet and PoseNet was using only the custom dataset. Depth estimation was therefore required to be learnt here purely from the inherent structure of this new dataset. The training and validation loss performance over the recommended 20 epochs [4], is shown in Figure 20 and Figure 21 respectively. It should be noted here the step scale in the PPGEO logging does not represent the 20 epochs that occurred, but the number of 'steps'. These meaning of these steps are not important as the total steps shown in the training curves are equivalent to the 20 total epochs, meaning that the dataset was iterated over 20 times. This high step size caused the resulting training loss curve to be very volatile as compared to the curve that would result from an epoch scale, and so smoothing operation was performed on this curve to better visualise what the per-epoch training loss would display as. Additionally, since the pretraining is self-supervised, the loss metric is not absolute but rather a normalised, unitless value that in this case, represents the per-pixel photometric reconstruction loss [6].

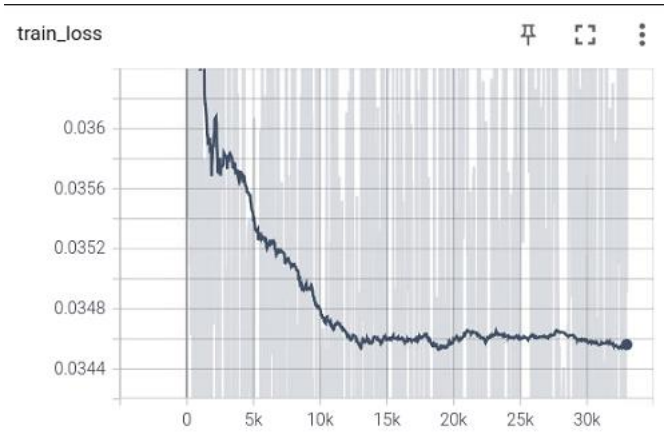


Figure 20: Smoothed photometric reconstruction training loss vs. steps during depth pretraining on only the custom dataset.

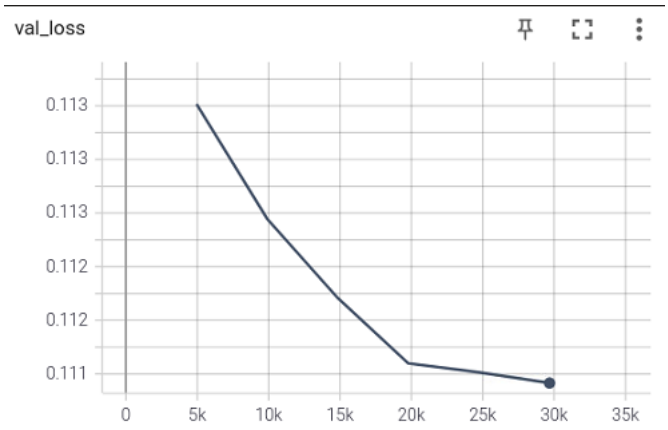


Figure 21: Photometric reconstruction validation loss vs. steps during depth pretraining on only the custom dataset

These loss curves indicate a high level of consistency and learning throughout this pretraining stage. Both exhibit exponential decaying nature, as expected in a sufficiently sized DNN training session. Although training quickly begins to plateau at around halfway through the steps, the validation appears delayed in it plateau, and so the training length appears appropriate and perhaps on the shorter side for optimal training exploitation. Nevertheless, the loss curves indicated that the encoder had progressively improved its depth estimation over time and thus could provide a useful depth map for downstream tasks.

From these promising curves, the next step was to visualise what the depth map predicts from a live input frame in Eglinton. This is a subjective, qualitative analysis as no suitable ground truth depth data was available from the shuttle bus. The depth map visualisation tool from Monodepth2 was recreated in the Eglinton simulation scripts, some of examples of which are shown in Figure 22 [6].

Figure 22: Depth Map visualisation of input Eglinton images (middle) from Monodepth2 models trained on custom dataset (left) vs. official dataset (right)

The quality of depth maps produced by the custom dataset was clearly worse than that of the official model published on the full dataset. It did however roughly verify the loss curves, in that it learnt to roughly distinguish differences in relative depth of the main features of each image, such as the cars on the shoulder of the road and the close-proximity bushes in the central road island.

As discussed in the model formulation, this was expected due to depth not having any influence from dataset context. Additionally, it is suspected that the noise from the custom dataset discussed in the previous section, plays a large role in this reduced quality. In both examples, the bottom of the frame (closest part of the road to the shuttle) appeared to have the highest depth which is clearly the opposite of what is expected. This part of the frame is exactly the area in which the custom dataset occasionally contains part of the vehicle's dash or windshield wipers. This likely confused the pretraining algorithm that these static pixels are indicative of far away objects that display little relative motion between frames, hence the high depth estimation.

Depth pretraining therefore seems to be only a function of the quality and size of the dataset acquired. The official dataset of worldwide footage is larger and of overall higher quality as there is limited unlabelled driving data from Australia as compared to the global availability.

A positive take away from this simulation is that there is potential for monocular camera models to be able to predict depth for use in various downstream navigation tasks such as obstacle avoidance or path planning. The dataset from ACO used for the depth maps on the right of Figure 22 is open source and readily available, thus there is already a great start point that can be integrated into any navigation system worldwide [5].

After this analysis, it was hypothesised that restarting the depth pretraining on the custom dataset, but this time with the official PPGEO depth encoder weights as the pretrained starting point, could only improve the quality of the depth prediction. The resulting training and validation loss curves are shown in Figure 23 and Figure 24 respectively.

Figure 23: Smoothed photometric reconstruction training loss vs. steps during depth pretraining on the custom dataset on top of the official dataset.

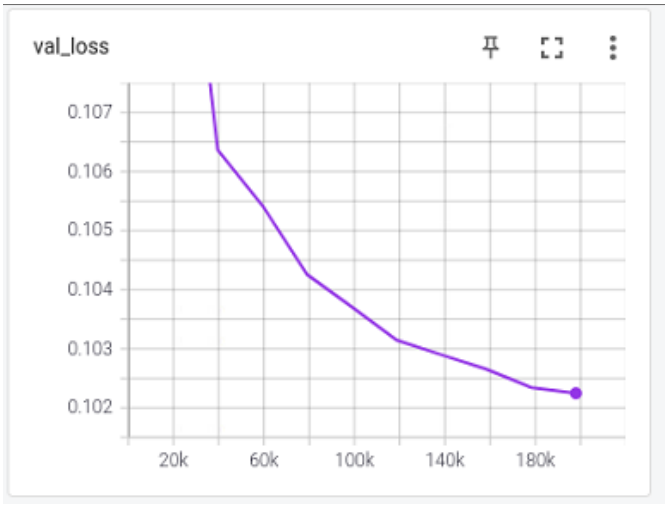


Figure 24: Photometric reconstruction validation loss vs. steps during depth pretraining on the custom dataset on top of the official dataset.

Despite both loss curves appearing similar in nature to the first attempt, the training loss this time was alarmingly approximately three times larger. Example outputs are shown in Figure 25 below.

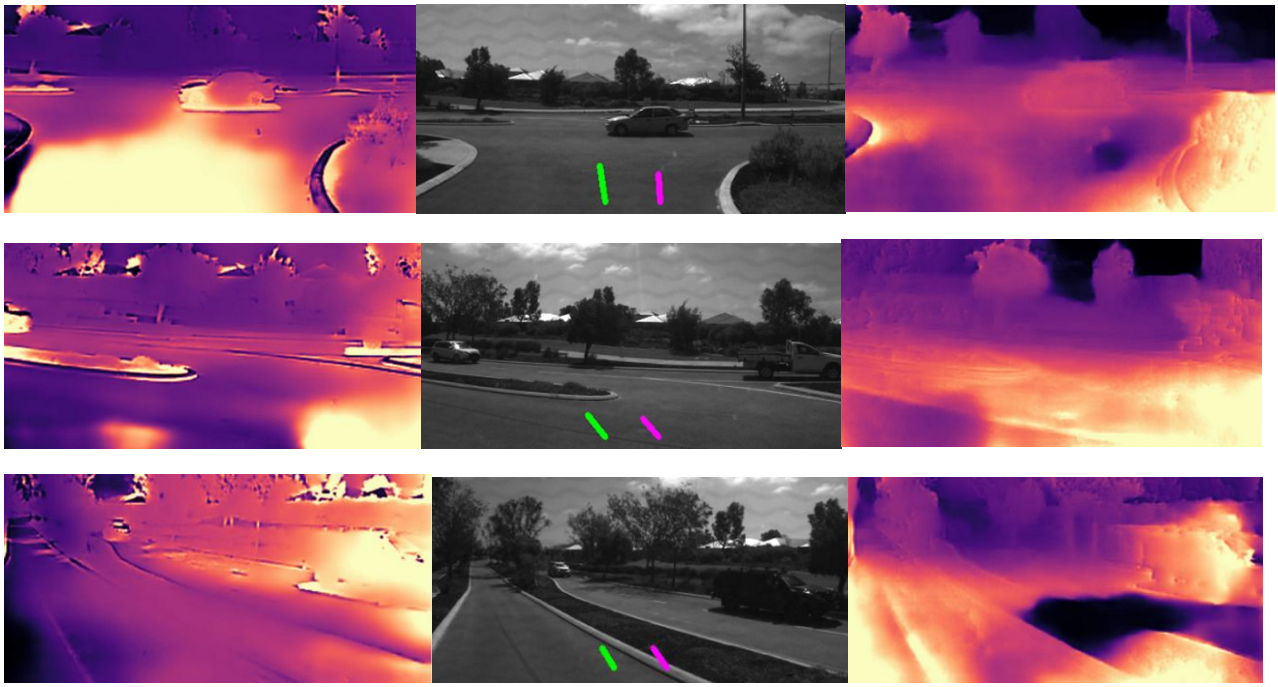


Figure 25: Depth Map visualisation of input Eglinton images (middle) from Monodepth2 models trained on custom dataset + official dataset (left) vs. official dataset only (right)

The visualisation results of this experiment were also underwhelming, indicating that the quality of the custom dataset simply was not of high enough quality for this particular pretraining algorithm, or that warmstarting the pretraining on the official model weights had a negative effect on its learning behaviour. It is suspected that this is due to slight differences in the nature of the dataset, such as its normalisation, encoding or camera intrinsics. The official dataset was compiled from a singular online source, meaning that its nature was potentially entirely consistent, whilst the custom dataset was compiled from varying sources recorded from varying cameras and in varying data formats. This may have led the custom dataset to start at a completely different local minimum as compared to the default dataset, leading to the backwards progression of the depth estimation.

3.3 PPGEO Policy Pretraining

Due to the unreliable quality of the DepthNet created in the previous section, it was decided against running the stage 2 PPGEO pretraining using these models. Stage 2 directly learns driving policy through ego motion learnt using the depth estimations from stage 1, and so it was assumed that utilising these poor-quality depth models would only cascade if used in stage 2. Therefore, the official, high quality DepthNet and PoseNet from the PPGEO pretraining that uses the ACO dataset, was used in stage 2, except the stage 2 pretraining was still executed purely using the custom dataset. This decision was taken because the depth estimation is not the core of the hypothesis and was not the focus of PPGEO, and so there was an anticipation that this custom dataset would not have such poor results in stage 2, where the core of the driving policy is learnt.

The resulting training and validation loss curves are shown below. The same logging characteristics, loss metrics and training epochs apply here as they did in stage 1.

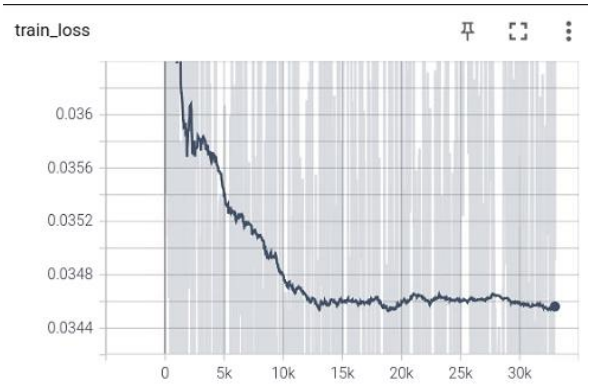


Figure 26: Smoothed photometric reconstruction training loss vs. steps during driving policy pretraining on the custom dataset.

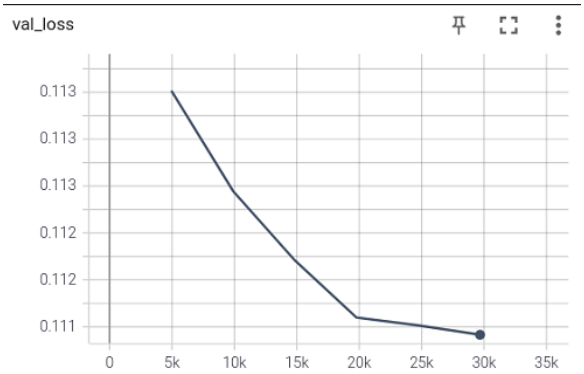


Figure 27: Photometric reconstruction validation loss vs. steps during driving policy pretraining on the custom dataset.

These loss curves appear similar in magnitude and nature to that of the initial stage 1 pretraining run which was the run that produced the far smaller training loss. This was promising as the loss continuously decreased and began plateauing, however stage1 visualisation results proved that the loss curves are not sufficient in verifying the final quality of model performance, and so qualitative visual validation using Eigen-CAM is necessary before any conclusions are drawn.

3.4 Loss Curves of Initial Eglinton Fine-Tuned Navigation Models

Before attention map saliency could be compared, the full Eglinton navigation models needed to be fine-tuned so that a comparison is made with a controlled model that represents the current working models whose visual encoders are fine-tuned from scratch. The newly produced PPGEO pretrained model was therefore fine-tuned in the varying freezing modes along with the official PPGEO encoder and the models without pretraining. This resulted in 8 different models being fine-tuned for comparison:

- PPGEO (official) unfrozen
- PPGEO partially frozen
- PPGEO frozen
- Custom PPGEO frozen
- Custom PPGEO partially frozen
- Custom PPGEO unfrozen
- ImageNet unfrozen
- Scratch (completely random weight) unfrozen

This selection of models would allow for a complete analysis of the effect of the pretrained encoders in terms of:

- Performance of frozen encoders to unfrozen encoders on the Eglinton dataset
- Performance of PPGEO pretrained encoders compared to default pretrained (ImageNet) encoders and randomly weighted encoders
- Comparison between custom pretraining and official pretraining
- Whether pretrained unfrozen encoders fine-tune faster than non-pretrained unfrozen encoders

The resulting loss curves of these models on the Eglinton lane following dataset is shown in Figure 28 below.

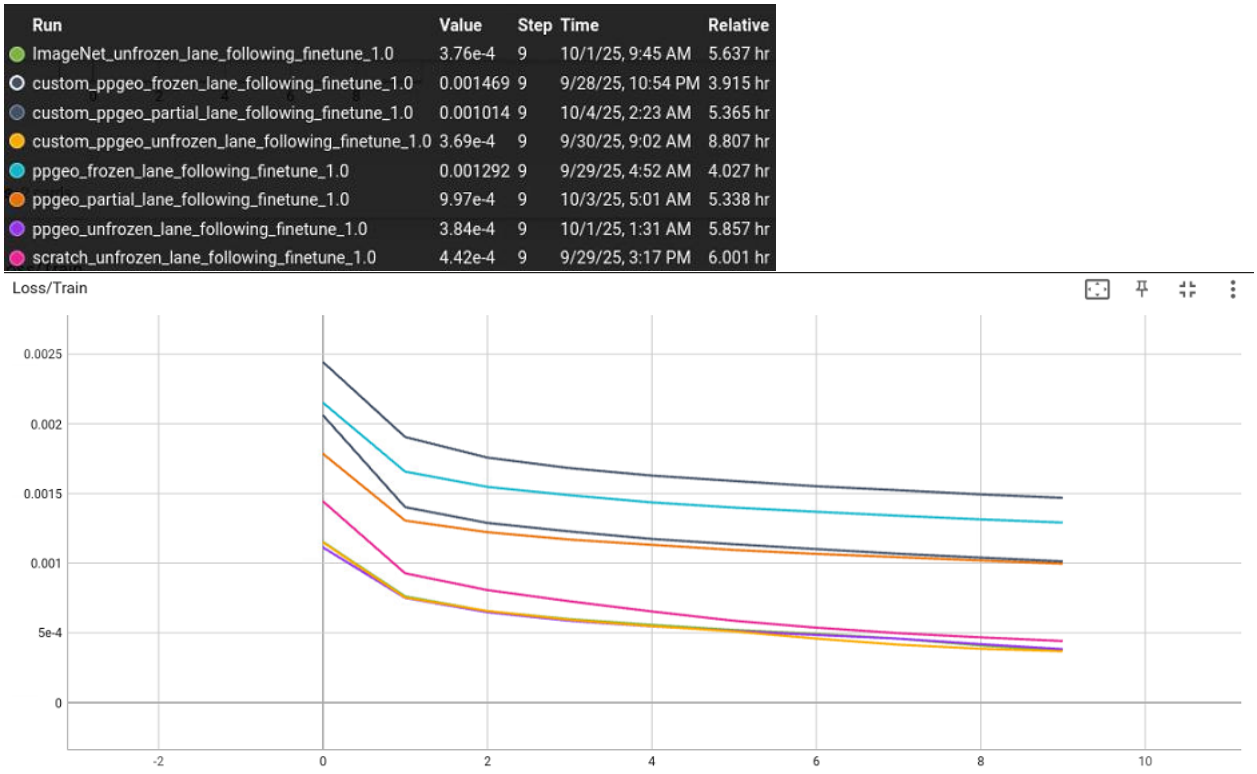


Figure 28: MAE loss of Eglinton Navigation models fine-tuned on Eglinton lane following dataset. (Note, there are two dark blue curves. As indicated by the legend values, the curve with higher loss is the custom PPGEO frozen model)

From these results, several observations can be made:

- Encoder freezing consistently produced higher loss curves than the equivalent unfrozen model (same trend with partial freezing) and begin to plateau faster.
- All fully unfrozen models have negligible difference in training loss behaviour and magnitude except for the scratch model which performs well yet slightly worse.
- The official PPGEO encoder performs slightly better here than the custom PPGEO model for both freezing modes.

These results were underwhelming as it showed no sign of benefit for freezing encoders or using PPGEO pretraining in any fashion. This form of validation however, is the least representative and most minimal as it is based on absolute loss. This type of validation tends to benefit the overfitting setups that can occur with a labelled dataset such as the Eglinton dataset. This suspicion of model overfitting must be verified by the saliency attention heatmaps as overfitting will be obvious if the unfrozen models focus on irrelevant parts of the specific dataset that aren't generalisable beyond this training route.

3.5 Model Attention Visualisations

Table 1 and Table 2 below show the results of this saliency comparison for each model for the same Eglinton camera input.

Table 1: Comparison between models of Eigen-CAM saliency for a simulation Eglinton Input Image coming out of a roundabout

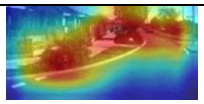
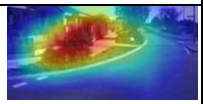
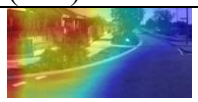
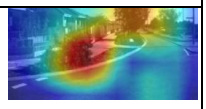
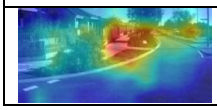
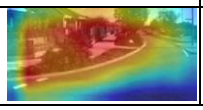
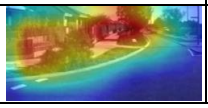
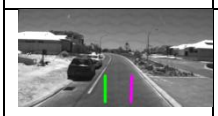
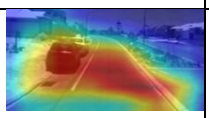
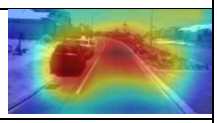
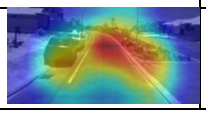
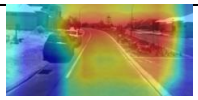
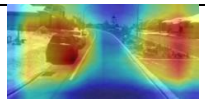
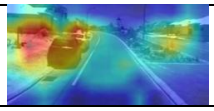
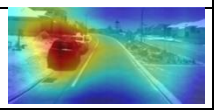
Input	Custom frozen	Custom Partial	Custom Unfrozen (Full)	Custom Unfrozen	PPGEO Frozen
					
Scratch Frozen	PPGEO Partial	PPGEO Unfrozen	Scratch Unfrozen	ImageNet Frozen	ImageNet Unfrozen
					

Table 2: Comparison between models of Eigen-CAM saliency for a simulation Eglinton Input Image with a car parked on the left shoulder, requiring attention

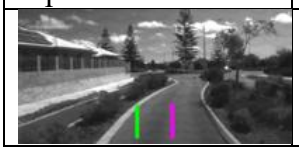
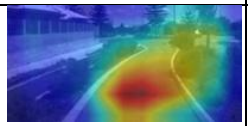
Input	Custom frozen	Custom Partial	Custom Unfrozen (Full)	Custom Unfrozen	PPGEO Frozen
					
Scratch Frozen	PPGEO Partial	PPGEO Unfrozen	Scratch Unfrozen	ImageNet Frozen	ImageNet Unfrozen
					

Here the benefit of pretraining as well as encoder freezing becomes much clearer. After inspection of many additional input examples, it was concluded that:

- Frozen pretrained encoders perform the best.
- Partially frozen pretrained encoders are close to their corresponding frozen counterparts, but overall, the partial unfreezing leads to slightly worse attention maps.
- The frozen default models, ImageNet and Scratch, have the worst saliency.
- The custom model outperforms in majority of Eglinton input frames, however the official PPGEO encoder is also superior in many examples.

This provides strong validation of the hypothesis that the pretrained encoders have potential to be more generalised models beyond the route in which models are fine-tuned on, as although they have higher overall MAE loss, they focus on more relevant driving features in the input images, thus have the potential to be more robust in a more diverse set of environments and inputs. Table 3 below reinforces this conclusion as the same pretrained model's Eigen-CAM saliency is displayed over the course of unfrozen fine-tuning.

Table 3: Comparison of the same custom PPGEO unfrozen model as it progresses over its training epochs

Input	Initial	Epoch 1	Epoch 5	Epoch 8
				

After just a singular epoch of supervised training, the attention of the model shifts from a robust, driving policy related map to a completely irrelevant one. Therefore, models trained on this labelled Eglinton dataset without freezing the encoder must overfit their models for the specifics of the route they are trained on, which make them brittle to slight changes in environment and route. It is no surprise that slight shifts in angles to the shuttle camera have forced complete recollection of the labelled dataset, as the models have adjusted to the specifics of the image input at the older camera angle.

The other validation from this visualisation is the benefit of compiling a custom dataset that is more suitable for the conditions seen in Eglinton. Overall, the custom frozen model outperformed the official PPGEO model in around 60% of examples, suggesting that the driving policy specific to a Perth suburban scene was learnt in the custom pretraining, causing this model to focus on Eglinton input specifics more often than the generalised worldwide dataset was able to.

3.6 Real Shuttle Tests in Eglinton

Due to time and resource constraints, not all of the produced Eglinton models were tested on the real shuttle, and so only four models were selected to validate the pretrained models to a basic extent. The main validation sought was whether the models with frozen pretrained PPGEO encoders show potential to navigate the real route with no supervised learning despite having significantly greater MAE training/validation loss. Therefore, only the following models were run:

- Official PPGEO frozen
- Custom PPGEO frozen
- ImageNet **unfrozen**
- **Fully fine-tuned** custom PPGEO **unfrozen**

Since it has already been demonstrated that on the Eglinton route, unfrozen models have superior performance, the 10 epoch ImageNet model was tested as a baseline reference of the way existing models are currently run there. The fully fine-tuned model also represents this, but it will more closely represent the potential performance of this model architecture, as existing architectures are fine tuned to much further than the 10 epochs (30 – 50 epochs) that was standardised in this investigation for a controlled comparison, and can even serve as an indicator for future comparisons of this model with the current best models running on the bus.

Below shows the training/validation loss behaviour of the Eglinton model architecture when fine-tuned to close to its full potential. Training these models with 10 epoch results in learning rate (LR) update, whilst this fully fine-tuned model had its LR lowered twice, as shown in Figure 29, indicating it did reach close to its full potential in MAE loss performance.

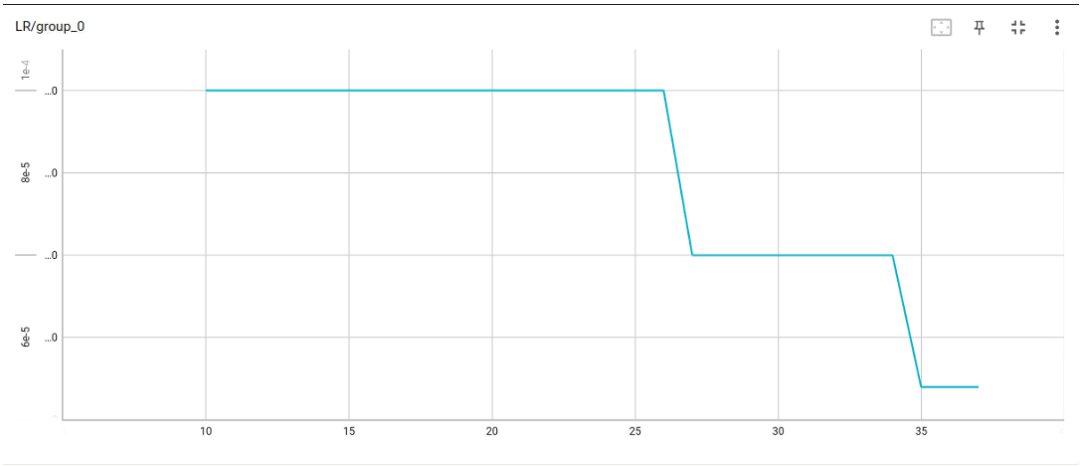


Figure 29: LR progression over the training epochs of the fully unfrozen Eglinton navigation model

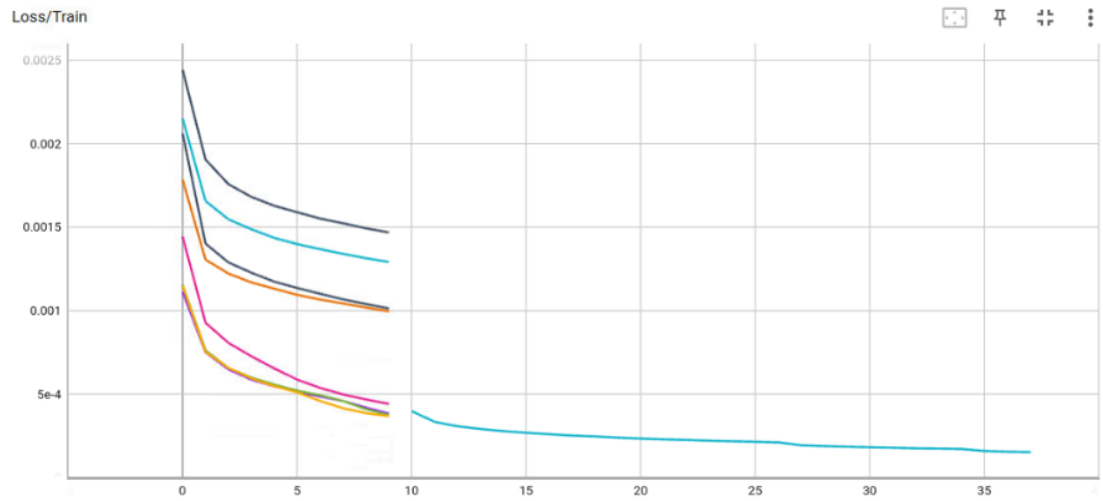


Figure 30: MAE training loss performance of the fully fine-tuned unfrozen model (lower light blue curve, not the higher shorter curve)

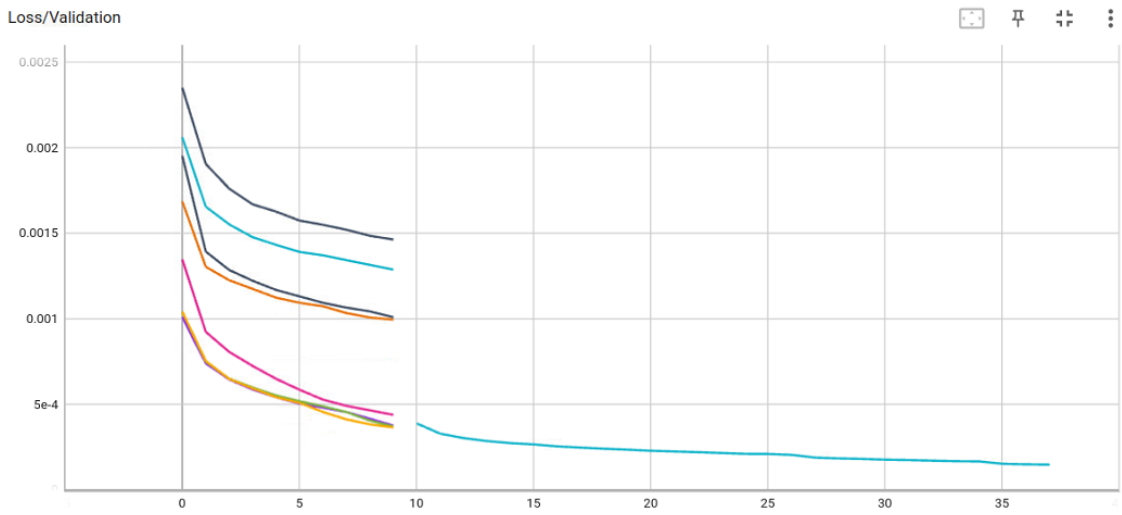


Figure 31: MAE validation loss performance of the fully fine-tuned unfrozen model (lower light blue curve, not the higher shorter curve)

The following sub sections will show the full set of map validation results for each model, after which a comparison will be performed.

3.6.1 ImageNet Unfrozen (Baseline)

Figure 32: Eglinton map visualisation of ImageNet unfrozen model navigation performance metrics

3.6.2 Custom Unfrozen

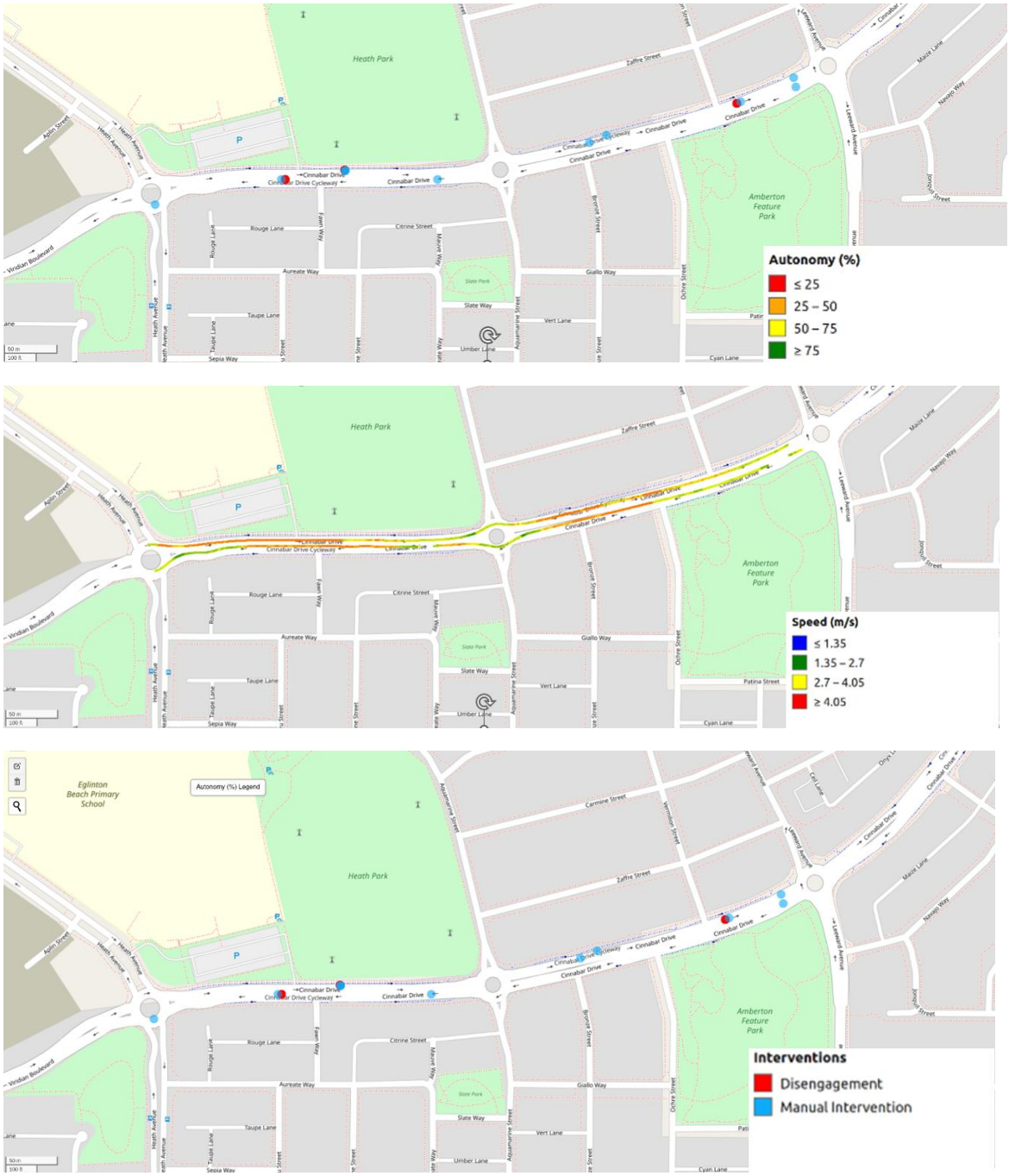


Figure 33: Eglinton map visualisation of custom unfrozen model navigation performance metrics

3.6.3 Custom Frozen

Figure 34: Eglinton map visualisation of custom frozen model navigation performance metrics

3.6.4 Default PPGEO Frozen

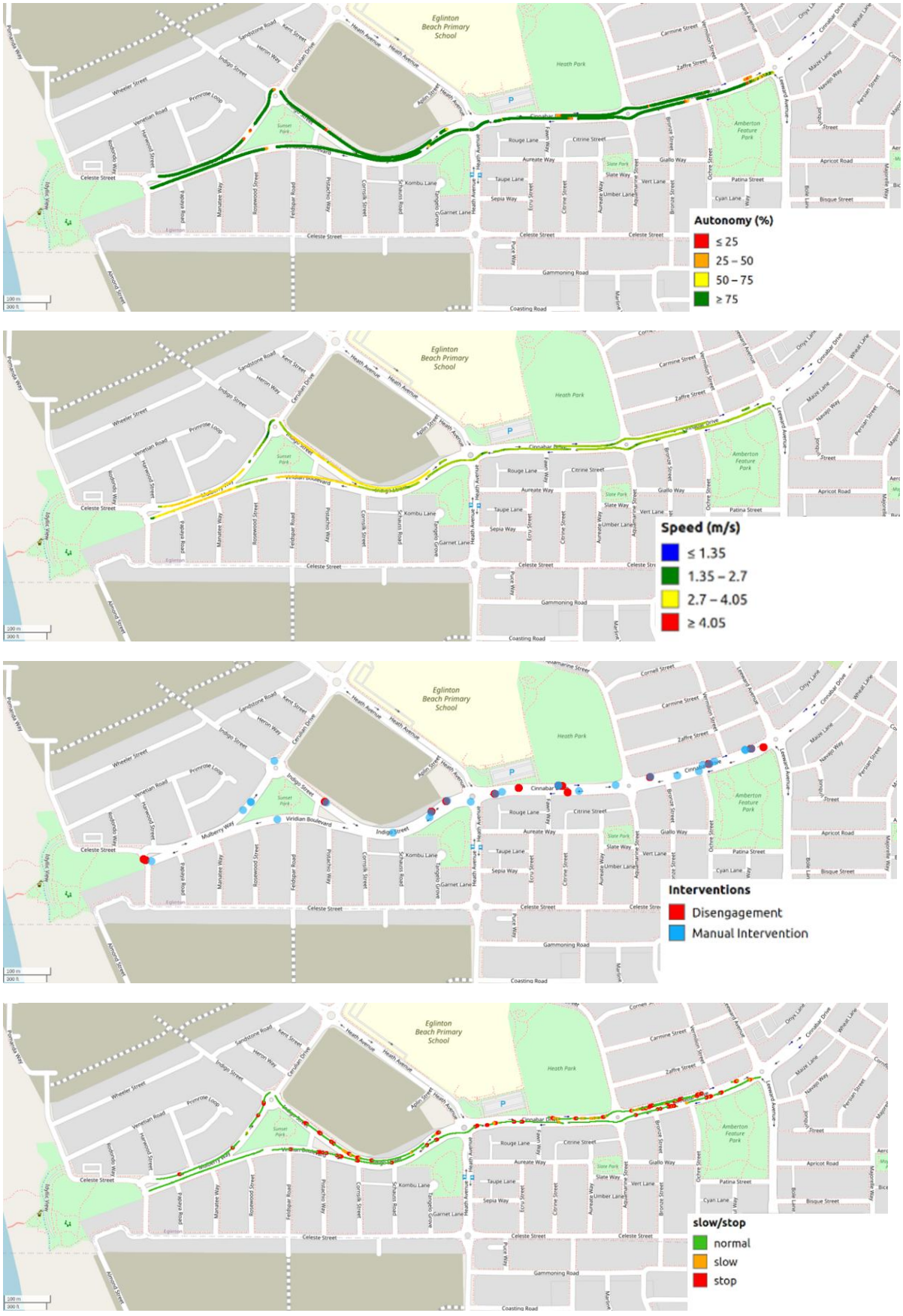


Figure 35: Eglinton map visualisation of official PPGEO frozen model navigation performance metrics

3.6.5 Limitation of Trials

It should be noted that the results above aren't entirely representative of each model's performance due to noise. In some runs, due to the volume of following traffic, manual overrides needed be performed, irrespective of the model's output, to perform a manual pull-in manoeuvre (which these lane following models were not trained do perform yet) to uncongested the traffic.

Additionally, some E stops were triggered during these forced manual interventions. The speed maps are also subjective to the speed multiplier set on the model output. The models trained on only 10 epochs were not sufficiently fine-tuned to the route and thus did not perform sufficiently well at the full speeds. This led to the decision to scale down their speed outputs. This is reflected in the speed maps however it would have shifted the results of E stops, manual interventions and slow node engagement to appear better than they would be at the full speeds that were run by the fully fine-tuned model.

Another limitation is that the runs only covered at most one full lap of the route, which is not sufficient data for reliable validation. For example, the autonomy % metric is more suited to compare how often a run is able to navigate a section of the route autonomously, not just over one run. The resulting map one run is therefore limited, only serving to show which parts of the single run it did navigate in autonomous mode. Future work should be run on the same speed multiplier, properly validate the physical runs over more laps, and filter out data that only arose out of a need to decongest traffic.

3.6.4 Comparison

The models with frozen encoders were not able to navigate at the full regular speed allowed by the shuttle at Amberton Beach. This is what caused such a large number of E stops or necessitated manual interventions. At slower speeds, the models were able to navigate through some of the more difficult manoeuvres, but not all.

The fully fine-tuned unfrozen models validated the model architecture's suitability for the Eglinton navigation task. It was able to navigate at the full high speeds during regular lane following, and at the regular low speed during turning and more complex lane following. This model seemed to have a performance that could merit comparison to the shuttle's current best working models.

The ImageNet unfrozen encoder performed in between these two extremes. It was run at the full speeds as it was able to navigate easily at the reduced speeds, however it did not perform close to the level of the unfrozen model that was trained to the point of plateau. It was expected in general that the unfrozen models would have better performance, but the question was how far off the frozen pretrained models were to the unfrozen. This model was also trained on 10 epochs and thus served as the baseline control comparison for the frozen models that were also trained on 10 epochs. The difference is that, from the validation loss curves, it is evident that the plateauing of the frozen models is far earlier than the unfrozen models, and so fair test of these models on a full fine-tuning would likely yield a greater separation in performance to the frozen encoder models.

Although the frozen encoders did not perform so well, their ability to perform some navigation through complex parts of the route, such as the roundabout sections, although at a scaled down speed, does give some indication that with further pretraining or dataset improvement, encoders have the ability to navigate on real roads with the added benefit of being more robust and generalized to changes in the route or environment.

4. Conclusions and Future Work

This project investigated the use of self-supervised pre-training methods to enhance the robustness and generalisation of camera-based navigation models for the nUWAy autonomous shuttle operating in Eglinton. Existing PPGEO pretraining pipelines were adapted and integrated with the shuttle's grayscale navigation system, and a custom driving dataset was created to better match local environmental characteristics. The adapted models were evaluated through training and validation loss analysis, saliency visualisations, simulation testing, and initial real-world trials. Results showed that while depth pre-training quality was strongly dependent on dataset size and cleanliness, policy pre-training using official depth and pose models yielded promising convergence behaviour and provided a foundation for improved downstream navigation models.

Future work should focus on expanding and cleaning the custom dataset to reduce noise and improve pre-training performance. Additional and more thorough real-world shuttle trials over extended routes and environmental conditions are needed to assess generalisation in practice.

References

[1] K. Quirke-Brown, Z. Lai, T. Tan, Y. Du and T. Bräunl, "Developing an Autonomous Shuttle Service," in *Australasian Transport Research Forum 2023 Proceedings*, Perth, 2023.

[2] Z. Lai, K. Quirke-Brown, L. Le, X. Kong and T. Braunl, "Shuttle Bus Performing Common Maneuvers in Roundabout-Style Suburban Residential Area Using Grayscale Camera-Based End-to-End Driving," in *IEEE TRANSACTIONS ON INTELLIGENT VEHICLES*, Perth, 2025.

[3] P. J. Navarro, L. Miller, F. Rosique and C. Fernández-Isla, "End-to-End Deep Neural Network Architectures for Speed and Steering Wheel Angle Prediction in Autonomous Driving," *Electronics ,* vol. 10, no. 11, 2021.

[4] P. Wu, L. Chen, H. Li, J. Yan and Y. Qiao, Policy Pre-training for Autonomous Driving via Self-supervised Geometric Modeling, Shanghai: International Conference on Learning Representations, 2023.

[5] Q. Zhang, Z. Peng and B. Zhou, "Learning to Drive by Watching YouTube Videos: Action-Conditioned Contrastive Policy Pretraining," *European Conference on Computer Vision (ECCV),* 2022.

[6] C. Godard, O. Mac Aodha, M. Firman and G. Brostow, "Digging Into Self-Supervised Monocular Depth Estimation," in *The International Conference on Computer Vision (ICCV)*, 2019.

[7] B. Wang, R. Sun, X. Yang, B. Niu, T. Zhang, Y. Zhao, Y. Zhang, Y. Zhang and J. Han, "Recognition of Rare Microfossils Using Transfer Learning and Deep Residual Networks," in *Biology*, 2023.

[8] A. Kumar, "Demystifying Encoder Decoder Architecture & Neural Network," 2024.

[9] M. M. Taye, "Theoretical Understanding of Convolutional Neural Network: Concepts, Architectures, Applications, Future Directions," in *Computation*, 2023.

[10] R. Wang, Z. Zhuang, S. Jin, N. Ingelhag, D. Kragic and F. T. Pokorny, "Feature Extractor or Decision Maker: Rethinking the Role of Visual Encoders in Visuomotor Policies," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[11] M. Yeasin and M. Muhammad Bany, "Eigen-CAM: Class Activation Map using Principal Components," in *IJCNN*, 2020.

[12] J. Utah, *Driving Rainy New York City 4K - Louis Vuitton Luggage - Driving Downtown,* 2025.

[13] T. Bräunl, Z. Lai, Y. Du and X. Kong, "The REV Project," UWA, 2023. [Online]. Available: https://therevproject.com/.

Appendices

Appendix A: Literature Review

The autonomous navigation of the nUWAy autonomous shuttle buses, some of which are shown in *Figure 36* , are developed by The University of Western Australia's REV team. The current system includes a modular ROS2-based architecture and is equipped with a sensor-rich platform including cameras, LiDAR, and GPS/IMU [10].

Figure 36: nUWAy Autonomous Shuttle Buses [10].

A camera-only, end-to-end neural network control system based on a modified PilotNet architecture has been deployed in the Eglinton Amberton Beach trial [1]. It performs reliably only on one trained route shown in *Figure 36*, however, remains brittle under environmental changes such as lighting variation or slight shifts in camera position. It would also require a complete manual training in order to extend to more routes, as the non-pre-trained neural network requires a large, labelled dataset. These challenges illustrate a need for more robust, generalisable vision-based models.

Figure 37: Current functional autonomous route in Amberton Beach [1].

This literature review situates the proposed work within ongoing research in vision-based driving, self-supervised learning, and policy pre-training, highlighting the limitations of existing solutions and the potential of PPGEO-based approaches.

1.1 Self-Supervised Depth and Ego-Motion Estimation

Monodepth2 [6] is a seminal work demonstrating that monocular depth and ego-motion can be learned from raw video using photometric reconstruction losses. This is a form of visual representation learning via self-supervised learning that is serves as a useful foundation for perception in the form of depth prediction. Key contributions include minimum reprojection loss, occlusion masking, and multi-scale depth supervision. Figure 3 shows the resulting depth maps produced by this framework on a raw driving image. It is clear from this output that Monodepth2 could be used by the encoders of the neural network in the shuttle buses as an initial as a foundational learning stage.

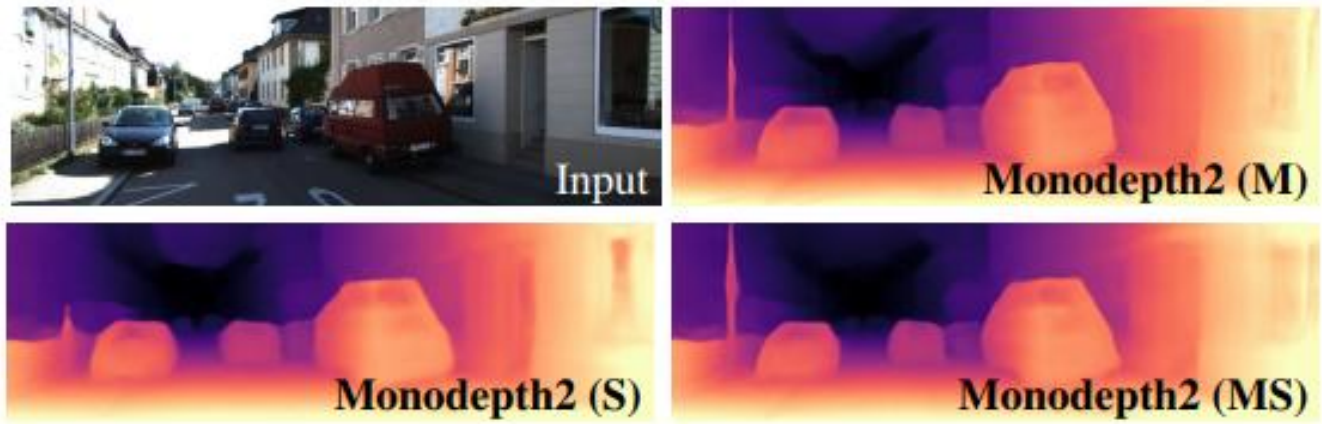


Figure 3: Sharp, High quality depth maps produced by Monodepth2 self-supervised model [6].

This method exploits geometric consistency and provides a scalable, label-free way to pretrain visual features. PPGEO inherits these components in its Stage 1, using them to train a backbone encoder on depth and pose tasks. This geometry-aware encoder can then be reused for downstream control policies with significantly fewer driving labels.

1.2 Policy Pre-Training from Unlabelled Video

PPGEO [4] further contributes by introducing a second stage of training in which the visual encoder predicts future ego-motion from a single frame. This encourages the model to encode cues that anticipate vehicle motion, such as road curvature or obstacles. This makes the learned representation more aligned with driving actions, even before any imitation or reinforcement learning occurs. Figure 4 visually breaks down this model of training into the three main stages: the pretrained depth estimation discussed in the previous section, the visual encoder incorporated by PPGEO, and lastly the downstream task fine-tuning requires depending on the application.

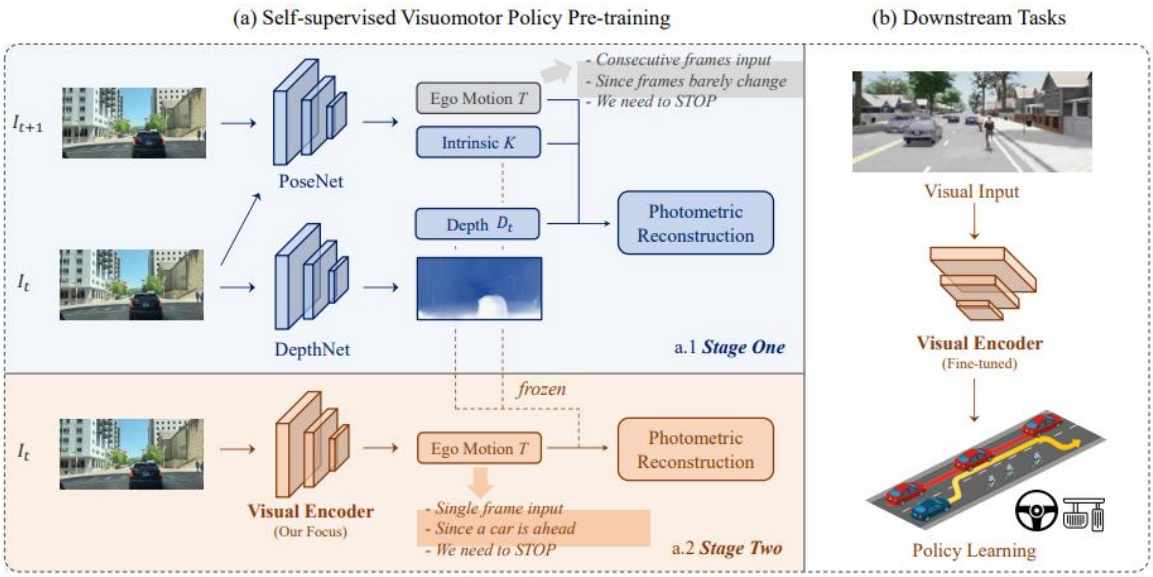


Figure 4: Overview of PPGEO [4].

In this case the downstream task would be the decoder for navigation output commands which filter out to just two values; throttle and steering angle, as discussed earlier. The fine tuning requires supervised training on the actual Eglinton labelled dataset that has already been acquired through manual driving along the specified route. The difference is that the pretraining is desired to reduce the size of this labelled dataset required for functionality, as the dataset collection is a time-consuming process, while unlabelled driving data is far easier to acquire via the internet. The important difference with PPGEO is that the fine-tuning and validation is done via CARLA simulation, while this project seeks implementation in a real shuttle bus, which could introduce additional challenges.

An alternative approach is presented, Action-Conditioned Contrastive Pre-Training (AC-CP)[5]. This method generates pseudo-action labels using a small, labelled dataset and an inverse dynamics model, then applies contrastive learning to improve action-aligned visual encoding. While effective, it requires initial supervision and is better suited for high-level decision making. In contrast, PPGEO uses dense pixel-level predictions and is more applicable to low-level visuomotor control.

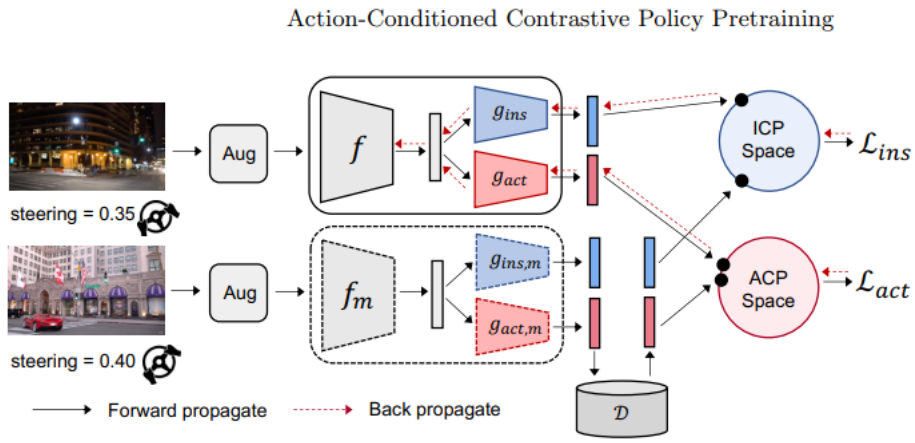


Figure 5: ACO Training Pipeline [5].

1.3 Dataset Considerations

One key limitation of existing pretraining pipelines like PPGEO is that their datasets are dominated by U.S. and European road environments. This can reduce model generalisability when applied to other geographies such as the Australian roads for this project's objectives.

Appendix B: Software Created

- All of the code used to produce this work is available on GitHub: https://github.com/SossaG/PPGEO-UWA-REV
- The original code for PPGEO that will be adapted in the next steps is available from the GitHub on their paper: https://github.com/OpenDriveLab/PPGEO
- The original code for the Monodepth2 paper used in depth map production can be found on their GitHub: https://github.com/nianticlabs/monodepth2/blob/master/test_simple.py