**Traffic Congestion Management Using autonomous car**

**A BTP Report**

By

**Vikas Chelluru,** S20200020252

**Kavya Sai Isheka Yakkala,** S20200020314



**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRI CITY**

**Date 4/12/2023**

**Final Report**

**INDIAN INSTITUTE OF INFORMATION TECHNOLOGY SRI CITY**

**CANDIDATE’S DECLARATION**

I hereby certify that the work which is being presented in the BTP entitled “**Traffic Congestion Management Using autonomous car**” in the partial fulfillment of the requirements for the award of the degree of B. Tech and submitted in the Indian Institute of Information Technology Sri City, is an authentic record of my own work carried out during the time period from January 2023 to December 2023 under the supervision of Prof. Dr. E. Paul Braineard, Indian Institute of Information Technology Sri City, India.

The matter presented in this report has not been submitted by me for the award of any other degree of this or any other institute.

Signature of the student with date

**Kavya Sai Isheka Yakkala**

**Vikas Chelluru**

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Signature of BTP Supervisor with date

Dr. E. Paul Braineard

**ABSTRACT**

Traffic congestion in urban areas presents persistent challenges impacting safety and efficiency. To address these issues, this project focuses on crucial components of an autonomous traffic management system.

The project takes significant steps towards system development by creating foundational elements. It begins with the construction of a hardware-controlled car using Simulink, Raspberry Pi, Arduino, and sensors. Additionally, reinforcement learning techniques are implemented within Simulink, aiming for lane following behavior. Furthermore, a U-Net based segmentation model is crafted specifically for precise vehicle identification.

This project focuses on building separate parts without combining them yet. We're creating a hardware prototype, using Simulink for lane following experiments, and crafting a special deep learning segmentation model to identify vehicles. These steps prepare us for combining everything later.

This project lays the groundwork for smarter traffic systems by first building separate parts—a car prototype, lane-following experiment using Simulink, and a special model to spot vehicles. These steps set the stage for combining everything later on, aiming to make traffic management more efficient and advanced.

**CONTENTS**

**INTRODUCTION 5**

**LITERATURE SURVEY 6**

**METHODOLOGY 7**

**RESULTS 16**

**CONCLUSION 26**

**LIST OF FIGURES 27**

**LIST OF TABLES 28**

**LIST OF ABBREVIATIONS 25**

**ACKNOWLEDGMENTS 30**

**REFERENCES 30**

**INTRODUCTION**

Traffic congestion within urban landscapes presents a formidable challenge, impacting safety, efficiency, and environmental sustainability. Conventional traffic control measures struggle to dynamically manage traffic flow, resulting in inefficiencies and safety hazards. To address these issues, this project focuses on pioneering an autonomous traffic management system, laying foundational groundwork for smarter and adaptable traffic control mechanisms.

The project adopts a deliberate step-by-step approach, commencing with the development of a hardware-controlled car. Utilizing Simulink, Raspberry Pi and L298N motor driver, this foundational hardware prototype serves as a precursor to subsequent advancements. Emphasizing the project's pioneering methodology, reinforcement learning technique take center stage for lane following strategies. Employing sophisticated Deep Q-Network (DQN) the project aims to refine the vehicle's navigation skills within simulated environments, fostering safer driving behaviors.

Furthermore, the project undertakes the creation of a specialized U-Net based segmentation model. Designed for precise vehicle identification, this segmentation model enhances the system's ability to discern and analyze vehicular surroundings.

This approach underscores the project's emphasis on discrete advancements of individual components, devoid of immediate integration. These initial strides—ranging from the hardware prototype's construction to the utilization of advanced reinforcement learning techniques and the creation of the U-Net model—lay the groundwork for future integration. Envisioning a future of smarter traffic control systems, this exploratory journey aims to contribute to improved traffic flow, heightened safety measures, and adaptable systems meticulously tailored for the complexities of urban environments.

**LITERATURE SURVEY**

The advancement of autonomous vehicle technology and traffic management has been propelled by pivotal studies such as "U-Net: Convolutional Networks for Biomedical Image Segmentation" by Ronneberger et al. [1], and "Lane Keeping Assist for an Autonomous Vehicle Based on Deep Reinforcement Learning" by Wang et al. [2].

In [1], Ronneberger, Fischer, and Brox present the U-Net architecture, originally designed for biomedical image segmentation. Its innovative design, integrating contracting and expansive pathways, demonstrated remarkable accuracy in segmenting biomedical images. Although developed for medical imaging, the U-Net architecture's potential applicability extends to other domains, including traffic management. The paper underscores the efficacy of convolutional neural networks in segmentation tasks, inspiring similar architectures' implementation for vehicle identification and segmentation in traffic scenarios.

Expanding the scope to autonomous vehicle navigation, [2] authored by Wang, Zhuang, Wang, and Ju, delves into lane-keeping assist systems utilizing deep reinforcement learning. This study explores reinforcement learning techniques to enable vehicles to navigate within lane boundaries autonomously. By employing deep reinforcement learning methodologies, the research focuses on enhancing an autonomous vehicle's ability to maintain precise lane positioning. This work significantly contributes to advancing reinforcement learning applications in autonomous vehicle technology, aligning closely with our project's objectives to leverage similar methodologies for adaptive vehicle navigation strategies.

These seminal works [1] and [2] significantly inform our project by providing crucial insights into deep learning, reinforcement learning, and segmentation techniques applicable to autonomous vehicle technology. The successful implementation of U-Net in segmentation tasks and the application of reinforcement learning for lane-keeping systems offer foundational frameworks guiding our exploration of similar methodologies to enhance vehicle navigation and segmentation within the context of traffic management scenarios.

**METHODOLOGY**

1. **Hardware Development**

**Objective:**

The primary aim was to design a hardware-controlled vehicle utilizing Simulink, Raspberry Pi and L298N motor driver as fundamental components to lay the groundwork for subsequent functionalities.

**Description:**

The hardware setup involved assembling a miniature vehicle equipped with four motors and wheels, L298n motor driver, batteries, a Raspberry Pi microcontroller, and a camera for real-time video streaming and object detection. The integration also encompassed configuring the communication between these components.

**Components Used:**

* Four motors and wheels assembly
* Raspberry Pi microcontroller
* Batteries (2)
* L298N motor driver
* Camera module for video streaming and object detection
* Mobile phone for remote control via Wi-Fi using the Simulink app

**Hardware Connections:**

* Motors and wheels interconnected for controlled movement.
* Raspberry Pi interfaced with the motors for transmitting control signals.
* Camera module integrated with the Raspberry Pi for video feed and object detection.

**Procedure:**

**Simulink Model Creation:** Designing and configuring a Simulink model tailored for controlling the hardware-controlled vehicle. This model included modules for motor control and communication protocols.

|  |  |
| --- | --- |
|  |  |

**Fig.1 Simulink Model and Robotic car**

**Model Deployment:** Deploying the configured Simulink model onto the Raspberry Pi microcontroller, establishing a functional connection between the model and the hardware setup.

**Mobile Control Setup:** Utilizing the Simulink app on a mobile device to establish a Wi-Fi connection with the Raspberry Pi. This facilitated remote control and interaction with the deployed Simulink model to steer and maneuver the hardware-controlled vehicle.

|  |  |
| --- | --- |
|  |  |

**Fig.2 Live stream data from raspberry pi to mobile phone using UDP communication over WIFI**

**2**. **Reinforcement Learning for Lane Following**

**Objective:**

The primary aim was to implement reinforcement learning techniques to facilitate lane keeping assistance in an autonomous vehicle system.

**Description:**

The approach adopted for reinforcement learning centered on the utilization of specific agents within Simulink, notably the Deep Q-Network (DQN). This agent were integrated into the Simulink model to facilitate Lane following based on received sensor inputs and environmental observations.

**Algorithms Used:**

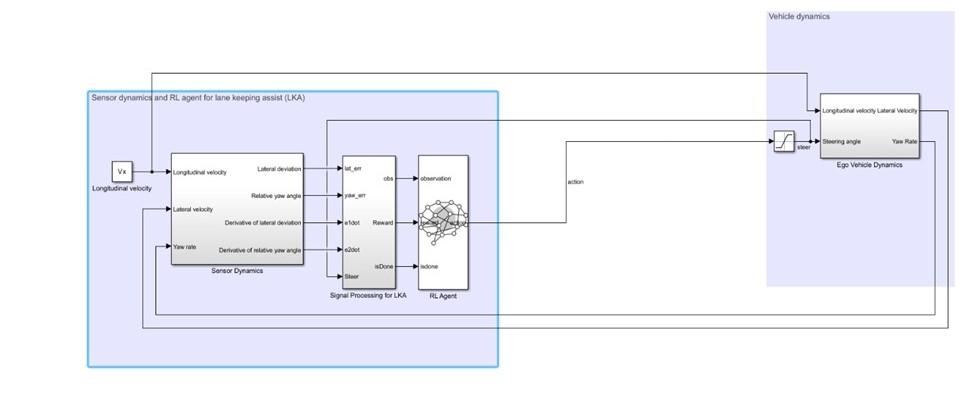
**1.Deep Q-Network (DQN):** Uses a Q-function to learn the optimal action to take in a given state by maximizing the expected future reward.

**1. State Representation:**

Utilize sensor data to construct the current state representation, encompassing the vehicle’s positional coordinates, velocity, and orientation in the environment.

**2. Action Space:**

Define the action space employing discrete actions to govern the vehicle’s steering angle and velocity adjustments.



**Fig.3 DQN Agent Model**

For this model:

* The steering-angle signal sent from the agent to the environment ranges between -15 degrees and 15 degrees.
* The environmental observations include lateral deviation (e1), relative yaw angle (e2), their derivatives (de1 and de2), as well as their integrals (∫e1 and ∫e2).
* Simulation halts when the absolute lateral deviation |e1| exceeds 1.
* The reward (ri) at each time step (t) is calculated as follows: 

Where u is the control input from the previous time step t -1.

**3.Training Process:**

* The training objective for the lane-keeping application is to ensure the autonomous vehicle maintains its position within the lane's centerline by dynamically adjusting the steering angle.
* Use experience replay and target network to stabilize the training process and improve performance.

|  |  |
| --- | --- |
|  |  |

**Fig.4 Training and Policy**

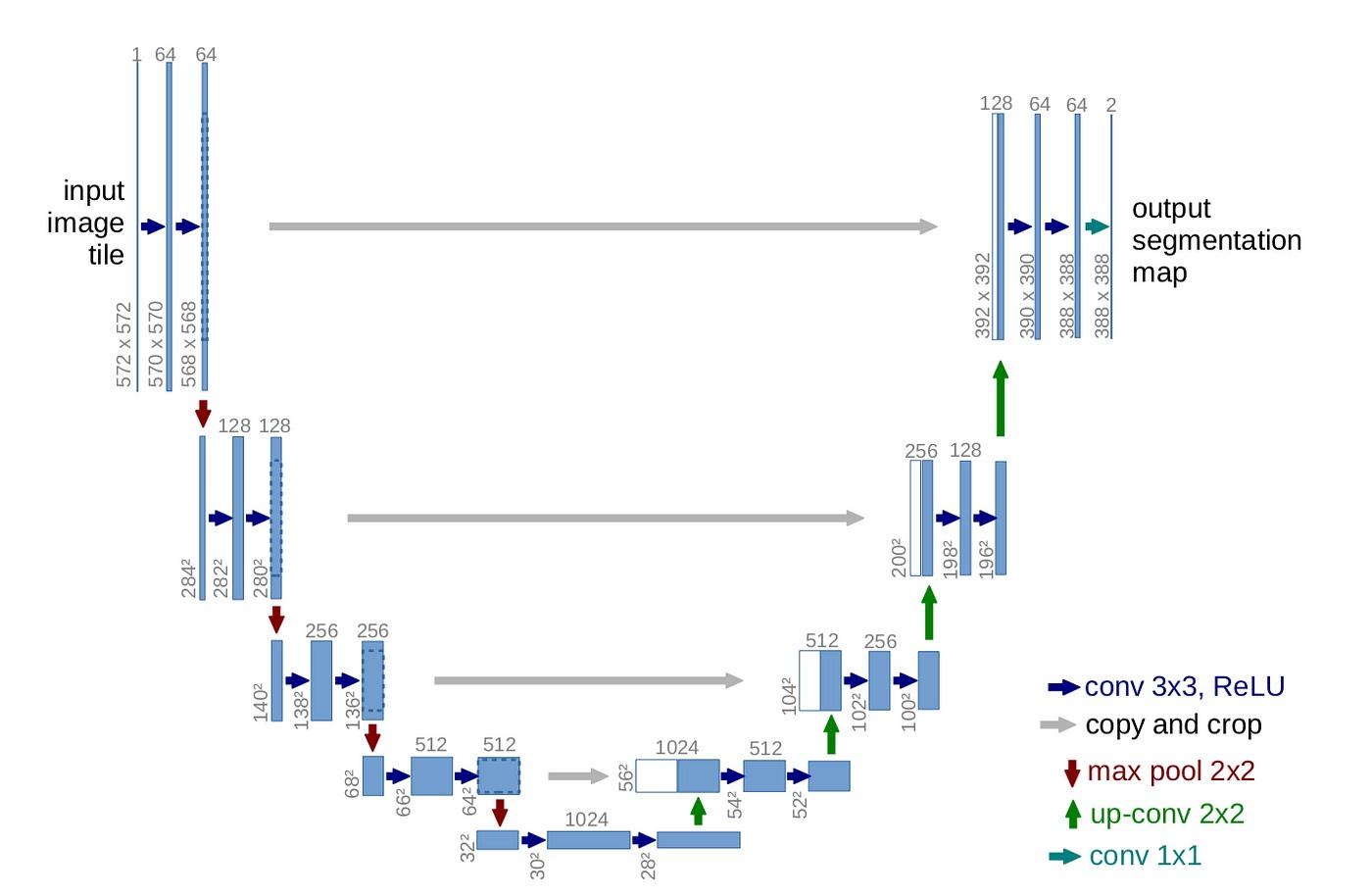
|  |
| --- |
| **DQN** |
| Lower Sample Efficiency |
| Can Handle Discrete Actions |
| More Stable Training |

**Table – 1 Advantages of DQN**

**3.U-Net based Segmentation Model:**

**1.Objective:** Modify and train a U-Net based segmentation model for precise vehicle identification.

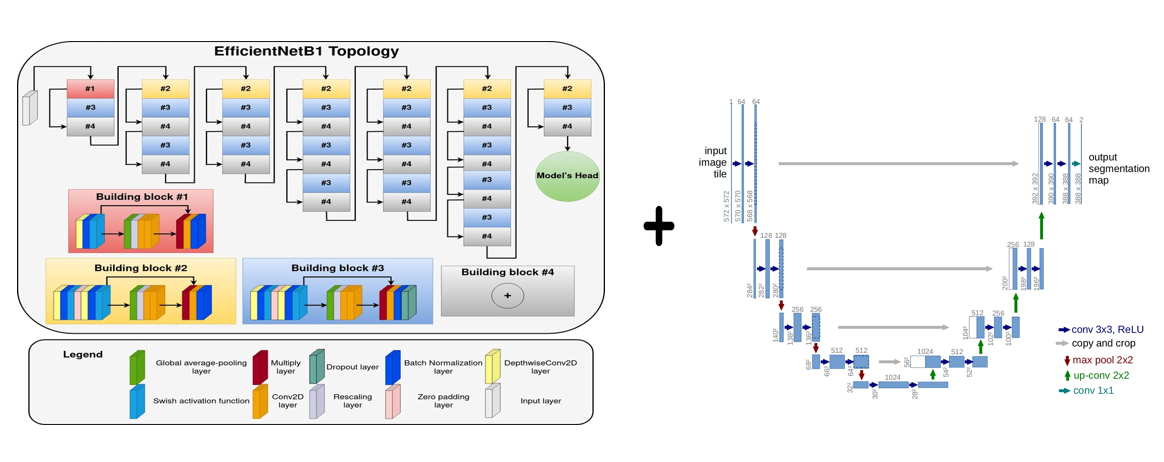
**Standard Model**: Enhanced the U-Net architecture by adding an extra level of depth and made corresponding adjustments in the feature extractor, while maintaining an input size of 416x416x3, resulting in improved performance for our specific task.



**Fig.5 Semantic Segmentation Using U-NET**

**Data Collection and Training:** To evaluate the performance of our modified U-Net model, we conducted an experiment on a small open-source dataset named 'licence\_plate\_segmentation\_india,' which consists of more than 500 images. This dataset was used to assess the model's segmentation capabilities on license plate data. give another paragraph with same meaning

**2. EfficientNetB1 + UNET(UNET-E)**

****

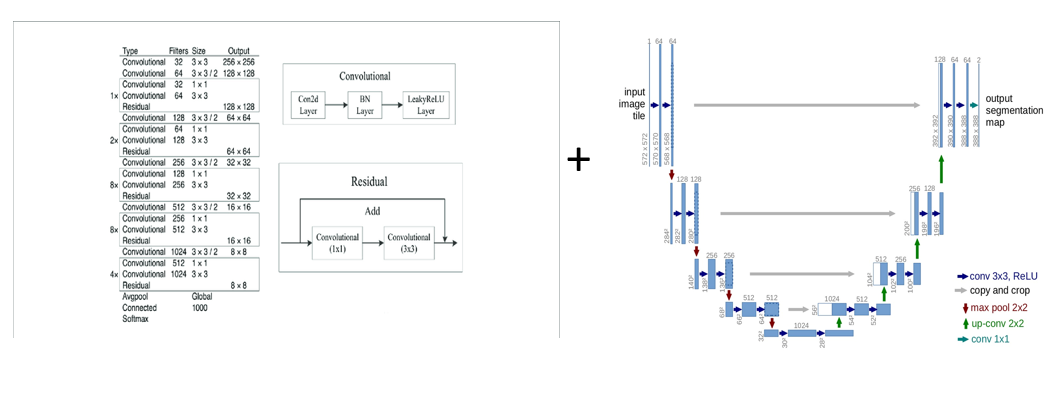
|  |  |
| --- | --- |
| **Fig.6 EfficientNetB1 + UNET (UNET-E)** |  |

**Model Modification**: Combining the last 50 layers of EfficientNet B1 with a U-Net architecture can indeed be a powerful approach for image segmentation tasks.

By combining the strength of EfficientNet B1, which has been learned from millions of images in ImageNet, with our customized U-Net model, we achieve remarkably precise image segmentation. This approach allows us to capture even the finest details and object boundaries with exceptional accuracy. compared to modified Unet.

**Data Collection and Training:** To evaluate the performance of our UNet-E model ,we conducted an experiment on a small open-source dataset named 'licence\_plate\_segmentation\_india,' which consists of more than 500 images. This dataset was used to assess the model's segmentation capabilities on license plate data.

**3. CSPDarknet53 + UNET(CSPUNET)**

****

**Fig.7 CSPDarknet53 + UNET(CSPU-Net)**

**Model Modification**: Combining the first 25 layers of CSPDarknet53 with a U-Net architecture.

Through the meticulous development of CSPDarknet53 from scratch, paired with our customized U-Net architecture, we have achieved respectable results. However, given the constraints of a modest training dataset and employing just 25 layers of CSPDarknet53, we do encounter some overfitting challenges.

**Data Collection and Training:** To evaluate the performance of our modified CSPDarknet53 + UNET(CSPU-Net) model, we conducted experiments on a small open-source dataset named 'licence\_plate\_segmentation\_india,' which consists of more than 500 images. This dataset was used to assess the model's segmentation capabilities on license plate data.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Training data** | **Training data** | **No of Epochs** | **Batch size** | **Total params** | **Trainable params** | **Non-trainable params** |
| 90% (450 images) | 10% (50 images) | 50 | 8 | 1941122(7.40 MB) | 1941122(7.40 MB) | 0(0.00 Byte) |
| 90% (450 images) | 10% (50 images) | 50 | 8 | 2082180(7.94 MB) | 2080941(7.94 MB) | 1239 (4.84 KB) |
| 90% (450 images) | 10% (50 images) | 50 | 8 | 2636069 (10.06 MB) | 2632997 (10.04 MB) | 3072 (12.00 KB) |

**Table 2: Evaluation Metrics of all three models**

**4.U-Net based Segmentation Model Modification:**

**Objective:** Train a modified UNet-based segmentation model for precise vehicle identification using real-world data and deploy it on a Flask web app.

**Data Collection:**

**Data Source:** Recorded video via a camera on Tada Road.

**Annotation:** Annotated images using Roboflow with 7 classes: auto, bike, bus, car, road lane, truck, each having different numbers of images.

**Augmentation Techniques:** Applied augmentation techniques such as brightness adjustment, shearing, noise addition, etc.

**Dataset Size:**

* Training Dataset: 1500 images
* Validation Dataset: 84 images
* Test Dataset: 25 images

**Training:**

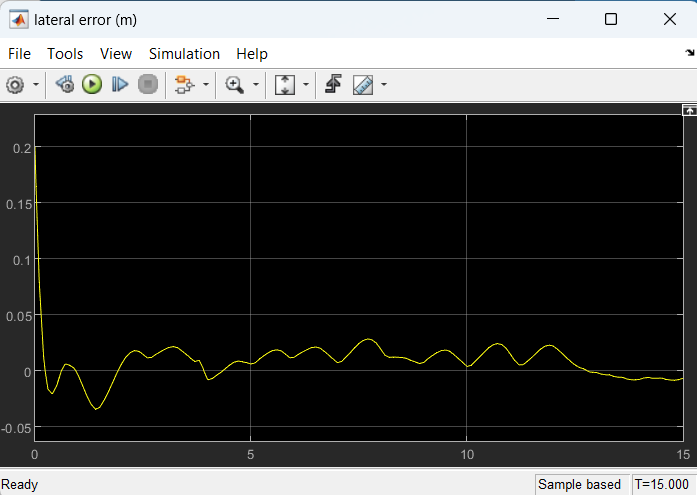
**Models Used:** CSPDarknet53 + UNET (CSPUNET), EfficientNetB1 + UNET (UNET-E), Standard UNET

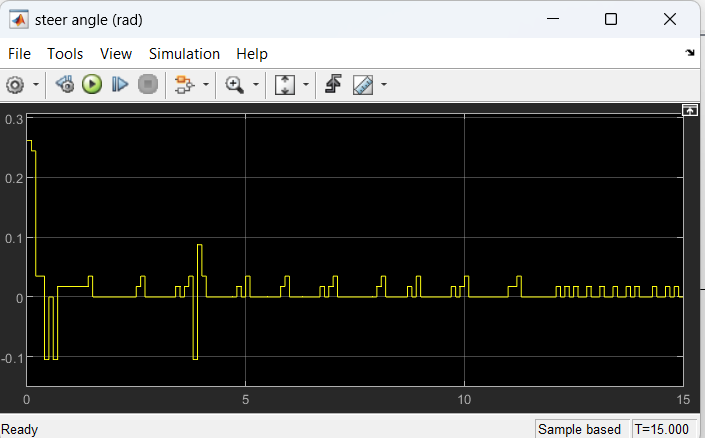
**Training Data Composition:**

Utilized the annotated dataset for training, validation, and testing of the segmentation models.

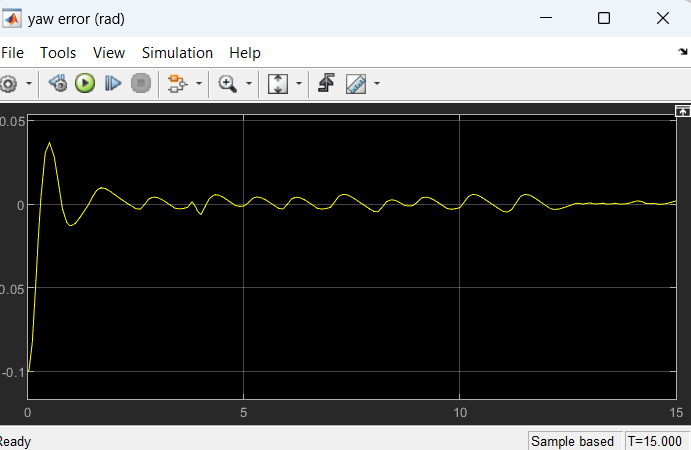
**RESULTS**

**DQN:**

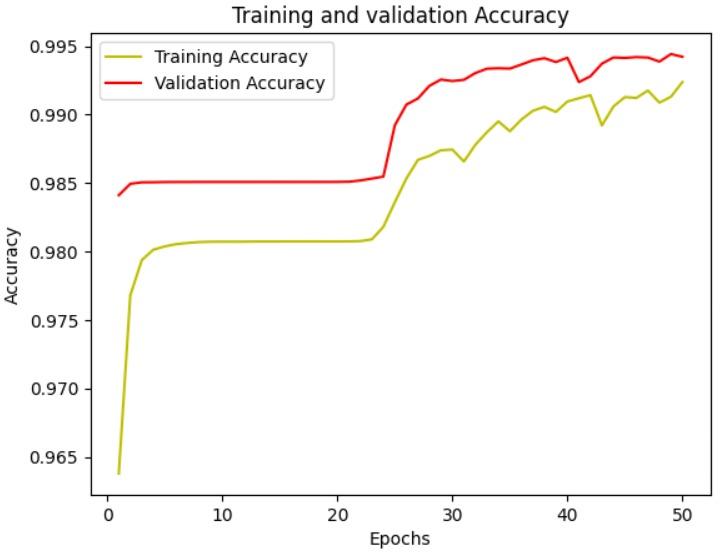
****

**Fig.8 Lateral error of DQN Agent**

**Fig.9 Steering angle of DQN Agent**

****

**Fig.10 Yaw error of DQN Agent**

****

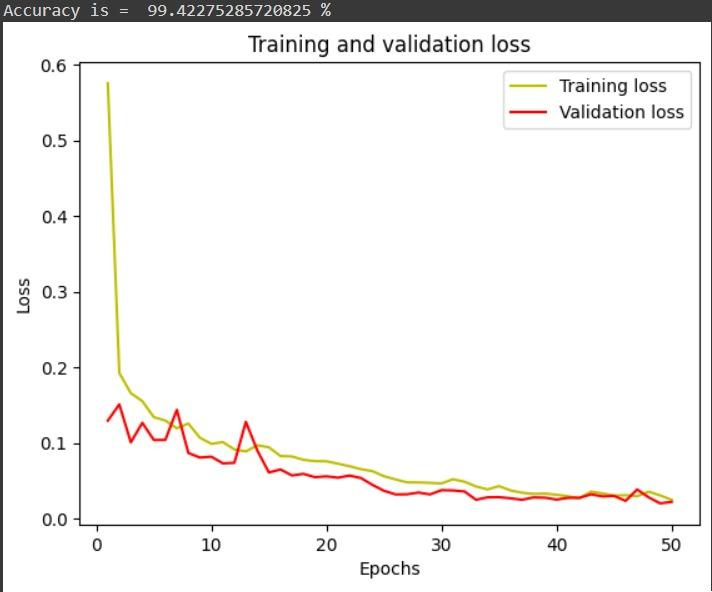
**Fig.11 Comparison of Training and validation accuracy of modified U-Net**

2/2 [==============================] - 1s 251ms/step

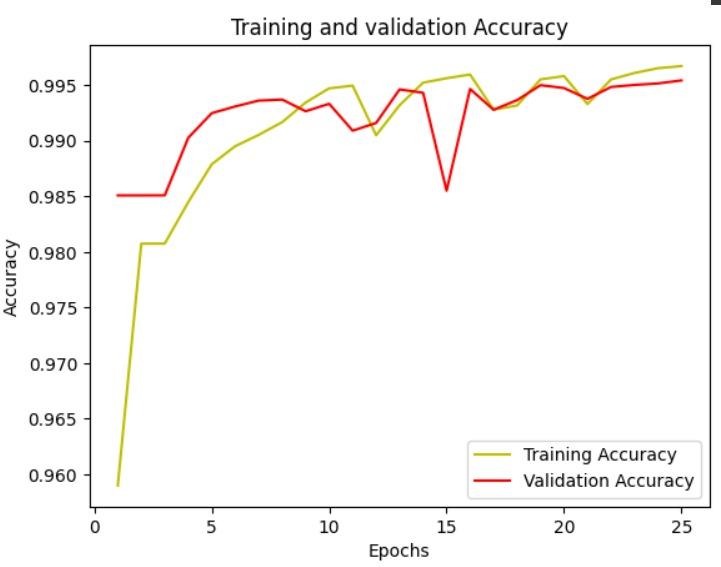
Mean IoU = 0.82348037

IoU for class1 is: 0.99416417

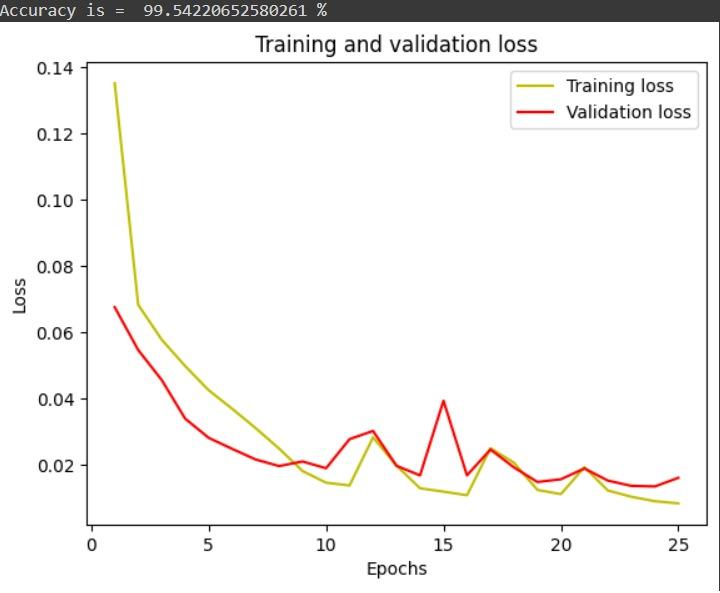
IoU for class2 is: 0.6527965

****

**Fig.12 Comparison of Training and validation loss of modified U-Net**

****

**Fig.13 Comparison of Training and validation accuracy of EfficientNetB1 + UNET (UNET-E)**

****

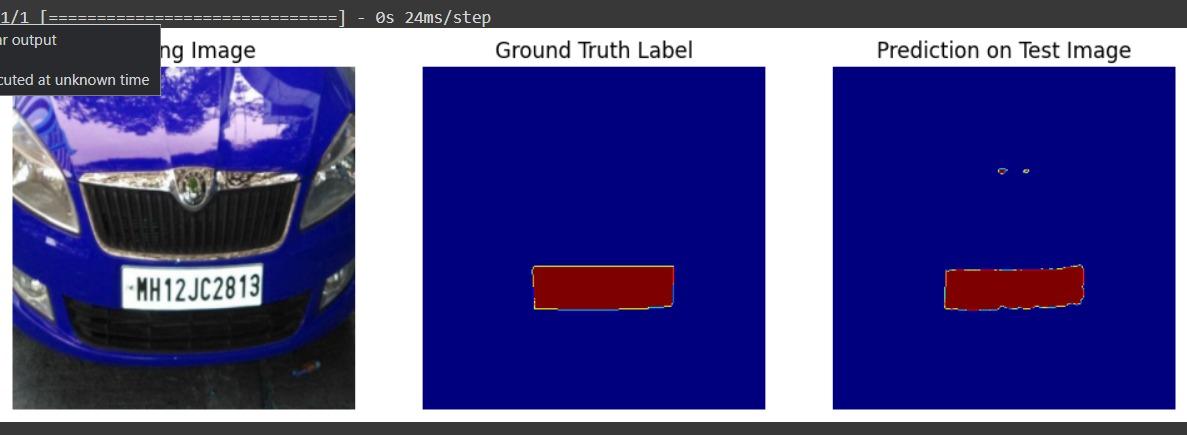
**Fig.14 Comparison of Training and validation loss of EfficientNetB1 + UNET (UNET-E)**

2/2 [==============================] - 3s 449ms/step

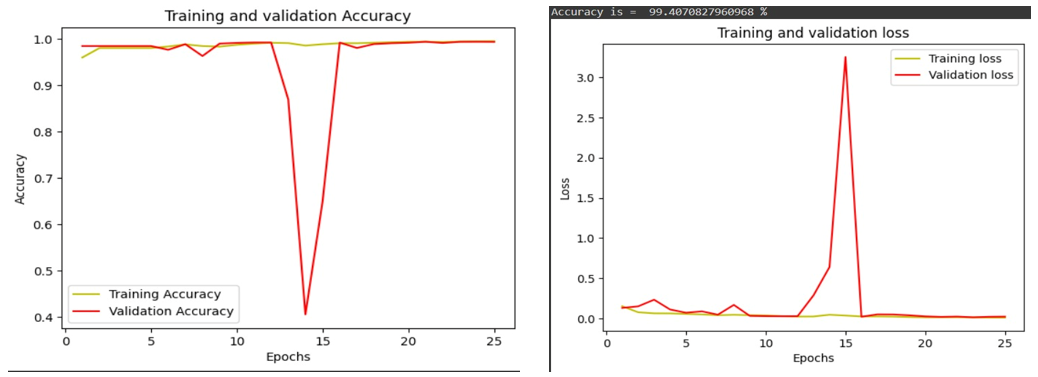
Mean IoU = 0.8579066

IoU for class1 is: 0.9953674

IoU for class2 is: 0.72044575



**Fig.15 Prediction on Test Image of UNET-E**



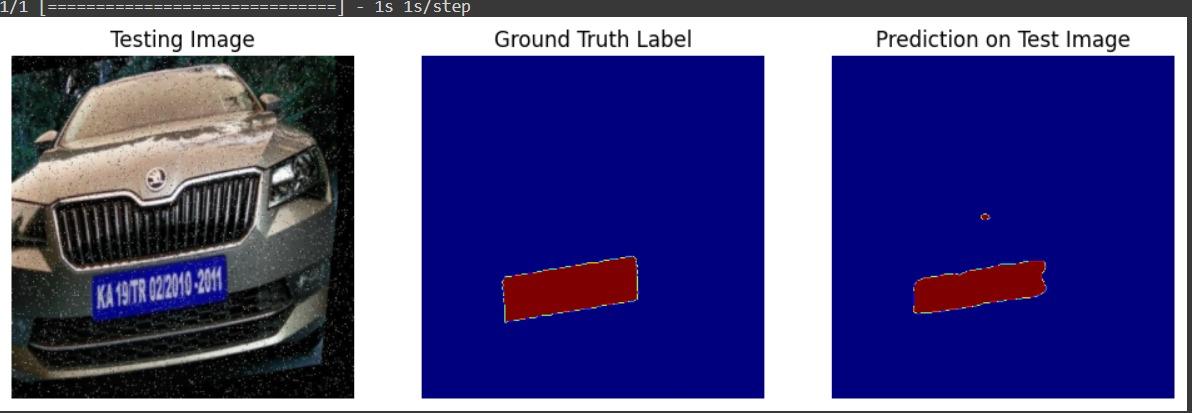
**Fig.16 CSPDarknet53 + UNET(CSPUNET)**

2/2 [==============================] - 1s 470ms/step

Mean IoU = 0.80996144

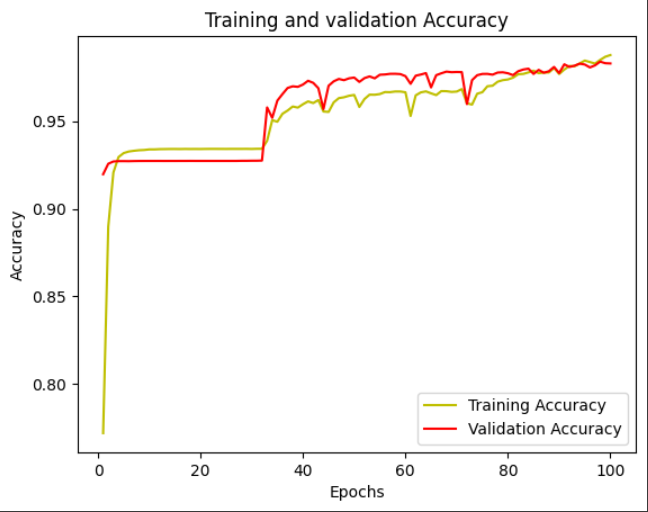
IoU for class1 is: 0.9940114

IoU for class2 is: 0.6259115

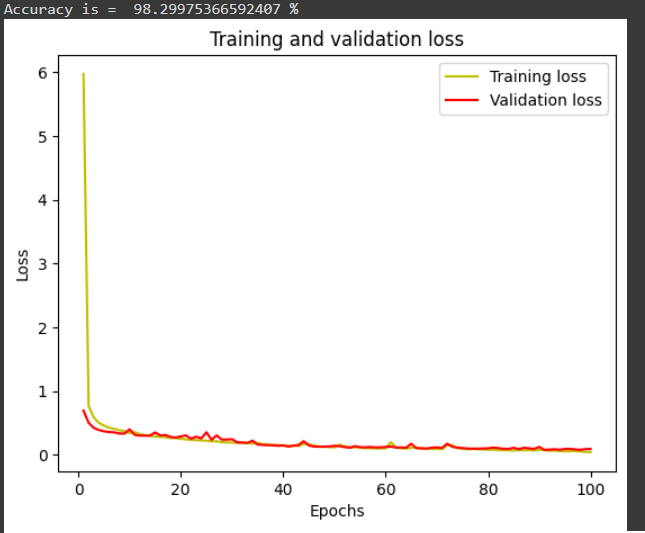


**Fig.17 Prediction on Test Image of CSPUNET**

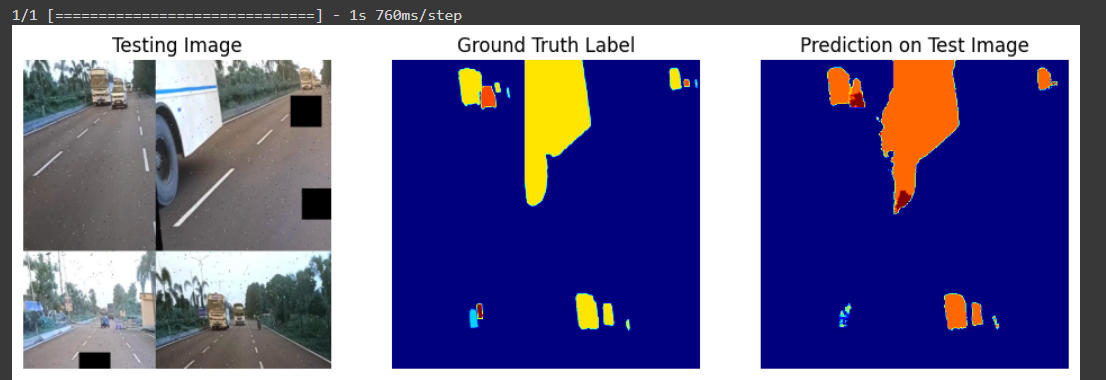
**Results of real-world data:**

****

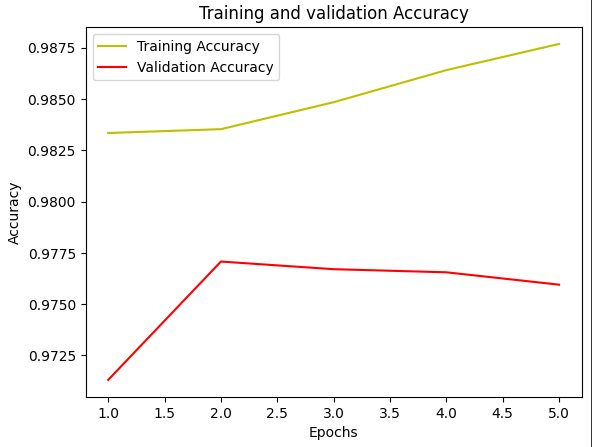
**Fig.18 Comparison of Training and validation accuracy of modified U-Net**

****

**Fig.19 Comparison of Training and validation loss of modified U-Net**

****

**Fig.20 Prediction on Test Image of modified U-Net**

****

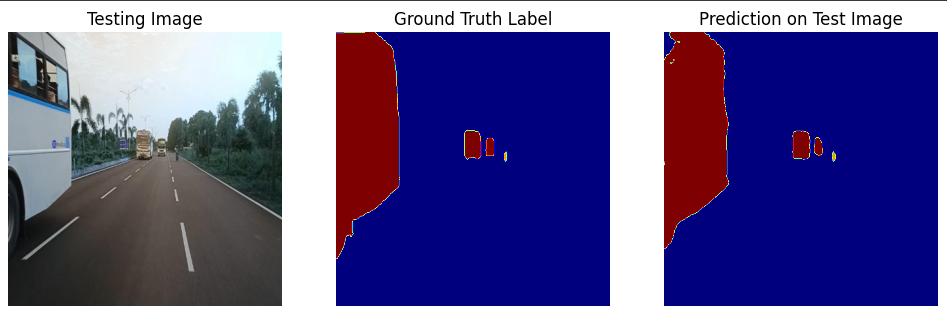
**Fig.21 Comparison of Training and validation accuracy of CSP U-Net**

****

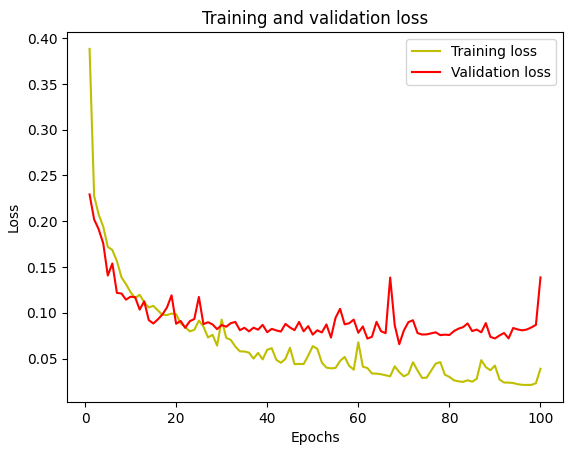
**Fig.22 Comparison of Training and validation accuracy of CSP U-Net**

2/2 [==============================] - 1s 522ms/step

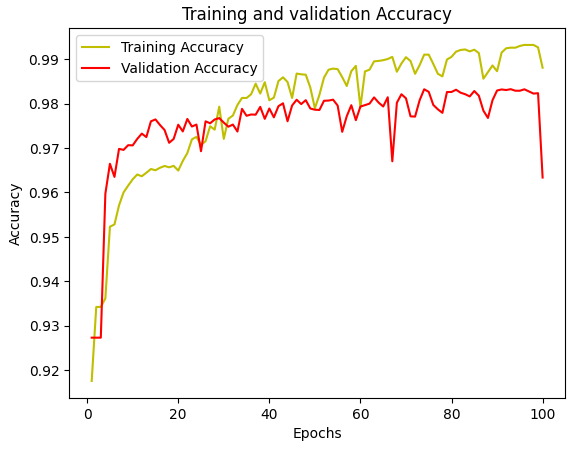
Mean IoU = 0.45063058

****

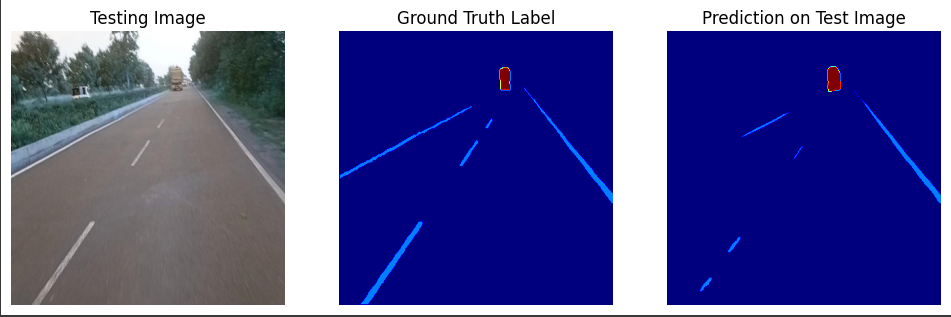
**Fig.23 Prediction on Test Image of CSPU-Net**

****

**Fig.24 Comparison of Training and validation loss of Unet-E**

****

**Fig.25 Comparison of Training and validation accuracy of Unet-E**



**Fig.26 Prediction on Test Image of Unet-E**

**CONCLUSION**

Throughout this endeavor, the project delved into the foundational elements of an autonomous vehicle system. Commencing with the hardware development phase, the construction of a hardware-controlled vehicle using Simulink, Raspberry Pi, and sensor integration laid the groundwork for subsequent functionalities. Leveraging this hardware platform, the project ventured into reinforcement learning technique, employing DQN agent within the Simulink environment for lane keeping assist. Additionally, modifications to the U-Net based segmentation model aimed at enhancing precise vehicle identification.

The reinforcement learning phase showcased the implementation of DQN agent, contributing to improved navigation within varying traffic scenarios. The DQN agent provided insights into diverse approach for steering control, addressing discrete action spaces, respectively. Concurrently, the enhancements made to the U-Net model revealed promising advancements in vehicle identification through segmentation, notably the Unet-E and CSPUnet models, exhibiting varying degrees of precision.

However, these achievements mark only the initial strides toward the comprehensive development of an autonomous traffic management system. The hardware, reinforcement learning, and segmentation model, though individually promising, are yet to be harmoniously integrated and rigorously tested in real-world scenarios. Further integration, refining the models, and comprehensive testing remain imperative for the holistic realization of an effective autonomous traffic management solution.

**List of Figures**

Fig.1 Simulink Model and Robotic car

Fig.2 Live stream data from raspberry pi to mobile phone using UDP communication over WIFI

Fig.3 DQN Agent Model

Fig.4 Training and Policy

Fig.5 Semantic Segmentation Using U-NET

Fig.6 EfficientNetB1 + UNET (UNET-E)

Fig.7 CSPDarknet53 + UNET(CSPUnet)

Fig.8 Lateral error of DQN Agent

Fig.9 Steering angle of DQN Agent

Fig.10 Yaw error of DQN Agent

Fig.11 Comparison of Training and validation accuracy of modified U-Net

Fig.12 Comparison of Training and validation loss of modified U-Net

Fig.13 Comparison of Training and validation accuracy of EfficientNetB1 + UNET (UNET-E)

Fig.14 Comparison of Training and validation loss of EfficientNetB1 + UNET (UNET-E)

Fig.15 Prediction on Test Image of UNET-E

Fig.16 CSPDarknet53 + UNET(CSPUNET)

Fig.17 Prediction on Test Image of CSPUNET

Fig.18 Comparison of Training and validation accuracy of modified U-Net

Fig.19 Comparison of Training and validation loss of modified U-Net

Fig.20 Prediction on Test Image of modified U-NET

Fig.21 Comparison of Training and validation accuracy of CSPU-Net

Fig.22 Comparison of Training and validation loss of CSPU-Net

Fig.23 Prediction on Test Image of modified CSPU-Net

Fig.24 Comparison of Training and validation loss of Unet-E

Fig.25 Comparison of Training and validation loss of Unet-E

Fig.26 Prediction on Test Image of modified Unet-E

**List of Tables**

**Table 1:** Comparison of DQN and DDPG Agents

**Table 2:** Evaluation metrics of M-UNET, UNET-E and CSPUNET

**List of Abbreviations**

**1.Simulink -** A visual programming environment employed for modeling, simulating, and analyzing dynamic systems across multiple domains.

2.**DQN** - Deep Q-Network, a reinforcement learning algorithm that employs a Q-function to learn the optimal action in a given state by maximizing future rewards.

3.**DDPG** - Deep Deterministic Policy Gradient, an actor-critic reinforcement learning algorithm used to learn a deterministic policy maximizing expected cumulative rewards.

4.**UNet** - A convolutional neural network architecture designed for biomedical image segmentation, developed by Ronneberger et al.

5.**CSPDarknet53** - A neural network architecture known for its efficiency in object detection tasks, often used as a feature extractor in conjunction with other models.

6.**EfficientNetB1** - A convolutional neural network architecture known for its efficiency in balancing accuracy and computational resources.

7.**RGB** - Red, Green, Blue, the primary colors used to represent color images.

8.**Epoc**hs - One complete pass through the entire dataset during the training process of a neural network.

**9.** **Batch size -** The quantity of training examples employed within a single iteration of the training process.

**10.Params -** Parameters, the variables or coefficients within a model that are adjusted during the training phase**.**

**11. MB -** Megabyte, a unit of digital information storage.

**12. KB** - Kilobyte, a smaller unit of digital information storage.

**ACKNOWLEDGMENTS**

We would like to thank our mentor, Prof. Paul Braineard for guiding us in this project and providing inputs and suggestions during evaluations and working. All team members have contributed equally, sharing the workload since the beginning.

**REFERENCES**

[1] Wang, Q., Zhuang, W., Wang, L., and Ju, F., “Lane Keeping Assist for an Autonomous Vehicle Based on Deep Reinforcement Learning,” SAE Technical Paper 2020-01-0728, 2020, doi:10.4271/2020-01-0728.

[2] U-Net: Convolutional Networks for Biomedical Image Segmentation Olaf Ronneberger, Philipp Fischer, and Thomas Brox.