1. Introduction

1.1. Autonomous Vehicles

Transportation has played an important role in the human history, evolving from the invention of the rotary wheel to the modern road cars. The invention of fuel engines in the last century marked a significant revolution, impacting faster goods transportation and extending to civilian life with the introduction of road cars. This period also witnessed the development of proper road infrastructures to accommodate the advancements in vehicle performance.

The fourth industrial revolution mark a start of new era where intelligent systems-controlled machines are developed which reduces the manual errors and enhancing performance. The automotive industry is also undergone this transformation with various intelligent driver assistance systems like Advanced Driver Assistance Systems (ADAS) which not only reducing human effort in driving but also ensuring safety. Engineers and scientists globally are working to enhance the capabilities of intelligent systems, aiming for complete autonomous driving, eliminating the need for human guidance.

The concept of autonomous driving dates back to 1926, with the development of a radio-controlled car in New York City. In 1960s, the Transport and Road Research Laboratory in the United Kingdom laid magnetic cables beneath the road, serving as a path detection tool for self-driving cars and tested its performance. Later, in 1995 Mercedes Benz achieved a significant milestone with the development of first self-sufficient autonomous vehicle, a retrofitted Mercedes S class equipped with efficient cameras and exclusive processors for parallel computing. This vehicle reached a maximum speed of 175 kmph, covering 1500 kms from Munich, Germany, to Copenhagen, Denmark, performing various manoeuvres in traffic without human assistance. In the late 2000s, carmakers like Toyota and Volvo, along with tech companies such as Google and Waymo, developed their prototypes of Autonomous Vehicles (AVs).

The parallel advancements in the field of Artificial Intelligence and efficient hardware accelerated research in self-driving technology. Notably, improvements in cameras and GPUs facilitated fast and efficient processing, while intelligent algorithms like neural networks laid the foundation for Autonomous Vehicles (AVs). In 2014, Tesla Motors launched Model S, a semi-autonomous driving car. This vehicle featured a various assistance feature, including lane detection, autonomous braking and parking, and speed limit recognition using computer vision. Tesla's entry into functional semi-autonomous driving represented a notable advancement in the field. During the same year, the Society of Automotive Engineers (SAE) had drafted a 6-level taxonomy for autonomous driving. This framework provided a standardized and structured classification system for assessing the level of autonomy in vehicles. Recognizing the growing impact of autonomous driving, several countries, including the United States of America, the United Kingdom, and Japan, made decisions to formulate laws addressing autonomous driving in the next few years. Various leading car manufacturers, including Mercedes Benz, BMW, and Volvo, have introduced their commercial fully or semi-autonomous production vehicles. They are continually enhancing their automation levels, as classified by the Society of Automotive Engineers (SAE). For instance, in 2022, Mercedes Benz achieved SAE Level 3 automation with its S-Class and EQS models, equipped with advanced driving technologies like an automated lane-keeping system. However, Tesla Motors has been a leader in this field, showcasing superior performance.

The SAE J3016 classifies autonomous vehicles into six levels, ranging from fully manual driving to fully automated systems [1].

|  |  |  |
| --- | --- | --- |
| Level 0 | No driving automation | This level involves no driving assistance system, and the entire driving process is manually controlled. |
| Level 1 | Driver assistance | Vehicles at this level feature a single automated system under the driver's supervision. |
| Level 2 | Partial driving automation | Multiple automated systems, such as automated steering and braking, work simultaneously under the driver's attention. The driver must be ready to take control at any moment. |
| Level 3 | Conditional driving automation | This level involves a system with an intelligent algorithm capable of making decisions like overtaking navigation. However, human attention is still required. |
| Level 4 | High driving automation | At this level, the vehicle can make optimistic decisions in the event of a failure, reducing the need for human interaction. This level is often suitable for confined areas or regions |
| Level 5 | Full driving automation | Here, the vehicle can perform all driving tasks without human interaction, comparable to an experienced human driver. |

Table1: Levels of driving automation defined in SAE J3016

Driverless vehicles find applications not only in passenger cars but also across various areas like shuttling people on campuses, moving goods in warehouses, and in military operations. Especially in industries, these automated machines could transport goods across the premises which save time and minimize human error. However, ensuring their safety of operation is crucial especially when it comes to road cars. Governments worldwide are creating regulations for autonomous mobility. Moreover, vehicle manufacturers are enhancing hardware and software while conducting extensive testing to ensure safety and prevent failures.

1.2. Simulators

Simulators imitate real-world systems, replicating their dynamics and features. They come in different types: software, hardware, or a mix of both. These systems replicate various functions, interacting with users, taking inputs, and providing feedback to the users. For example, driving simulators create visual environments and hardware setup for driver training and testing, Which takes control commands from users and update their states. Simulators can be classified based on various features Some of the examples are

Based on Time

Discrete Event Simulation: Each step happens at distinct intervals, which are mapped to specific durations. Driving simulators fall into this category, where observation states change at each step. The time duration of a single time step is defined by required precision.

Continuous Simulation: In this type of simulation, time moves continuously without distinct steps, backed by differential equations. These are useful for modelling continuous events like microbe growth in biology.

Based on Stochasticity:

Stochastic Simulation: This type of simulation introduces a controlled amount of randomness within sensible limits in the parameters. It's helpful for studying systems with random noise, like analysing daily usage patterns on a social media site.

Deterministic Simulation: These systems operate using deterministic algorithms without any randomness. They're commonly applied in engineering. The outcomes of these simulations remain consistent and reproducible for a specific set of parameters.

Users can tweak these parameters to test and tune the results of the system. Usually, simulations help to find the optimal parameters and it have diverse applications, for events spanning millions of years to nanoseconds. Simulations play a vital role in result analysis, safety engineering, and design processes. They save significant time during testing and can enhance the testing outcomes. For instance, testing the driving of an autonomous car for 10,000 km in the real world demands a lot of time and resources, whereas using a simulator significantly reduces this timeframe. However, there's often a discrepancy between the real world and simulations, which might lead to errors in analysis. In fields like machine learning and robotics, it contributes a crucial role. One of the three paradigm of Machine learning is reinforcement learning where a virtual agent within a simulator takes diverse actions under various conditions and learns from the feedback provided. In robotics, algorithm which drives the robots are trained using simulation for tasks like localization, pick and place, faulty detection…

1.3 Problem Statement

1.3.1 Lack of system of comparison

In the growing domain of autonomous vehicle research, simulator plays a significant role in the process of development such as algorithm training, performance evaluation across diverse environments and comparison of method. There are numerous simulators, varying in types and functionalities, available in the community for direct or indirect use in research works. In addition, the domain of self-driving vehicle research itself is broad, includes distinct applications such as indoor robots, industrial autonomous vehicles (AVs), on-road vehicles, and semi-autonomous vehicles. Each application requires a unique simulator tailored to its specific requirements.

The challenge arises due to the diversity among these simulators and their requirements, each comes with its own set of advantages, features, and limitations. Consequently, comparing and selecting the most suitable simulator becomes a complex task. For instance, a simulator might perform well in numerous aspects, yet fail to simulate a crucial sensor required by a user, making it unsuitable for that specific application. Conversely, another simulator might lack certain features but could be better suited for the user.

The aim of this thesis is to establish a concrete set of metrics for comparison purposes. The proposed metric will be generic and adaptable to individual user preferences. Users will have the flexibility to adjust the importance of criteria based on their requirements and they are universally applicable across all types of autonomous vehicle simulators. Through this systematic approach, users could able to assign personalized ratings to simulators, facilitating informed decision-making in selecting the most suitable simulator according to their preferences and needs.

1.3.2 Simulators Based on Generative AI

Generative AI had gained momentum in recent years, continuously improving its performance. This field primarily uses deep neural networks, trained on any provided data, learning its distribution. These trained models can generate novel, meaningful data samples which are not present in the training set but similar to it.

In the domain of autonomous vehicle simulators, rendering new scenarios and environments is a one of its objectives. This feature enhances its suitability for algorithm testing across diverse environments, potentially improving the model’s performance. It’s also important that these new scenarios closely resemble real-world features while being meaningful and plausible.

Sensors in Autonomous cars play a important role in understanding the real world where the vehicle navigates. These vehicles utilize data from various of sensors such as cameras, LiDAR, and radar to perceive their surroundings. The second part of this thesis aims to design and develop a simulator driven by a Generative AI model trained on real-world sensory data. The expected outcome is the generation of new, meaningful sensory data can effectively represent diverse and realistic environments. The simulator will operate on discrete time steps, ensuring that the generated data remains relative to preceding time steps.

2. Literature Review

This chapter explores different works related to the topic. The initial section examines the existing state-of-the-art autonomous vehicle simulators. The second part provides a summary of previous methodologies used to define a comparative metric for evaluating autonomous vehicle simulators. The last part of this chapter discusses about the different types of Generative Adversarial networks (GANs), a type of Generative AI algorithm utilized in this thesis.

2.1 State-of-the-Art Simulators

Numerous simulators are available in the market, but this section analyse some relevant and popular simulators and highlighting their advantages, limitations and applications.

2.1.1 CARLA

CARLA, (Car Learning to Act) is an open-source software developed collaboratively by the Computer Vision Centre (CVC) and the Barcelona Supercomputing Centre (BSC) in partnership with the Toyota Research Institute. It is primarily designed for autonomous driving research and development, which features diverse and realistic environments, various climates and wide range of sensors. CARLA operates on a server-client architecture, built on Unreal Engine 4 and utilizing the OpenDRIVE standard 1.4 to define roads and urban settings. This unique structure allows the server to manage physics of simulator and computation while enabling user to communicate the server through C++ and Python APIs, providing scalability.

A notable feature of CARLA is its seamless support for developing, training, and validating machine learning algorithms. Researchers had experimented various algorithms like modular pipelines, imitation learning, and reinforcement learning within this simulator [2]. Leveraging Unreal Engine 4, CARLA offers high-quality, realistic rendering of environments. Figure 1 showcases scenes from the simulator in different weather conditions. Additionally, it supports an array of built in sensors such as camera, LiDAR and provides the user with various metadata, and ground truth. Moreover, CARLA offers access to diverse digital assets (actors, buildings) within the environment, meticulously designed to maintain a high level of realism [2]. However, it currently offers support for only two pre-defined urban maps covering 2.9 km and 1.4 km, which limits its diversity and generalization capabilities.

Figure1: Scenes from the CARLA simulator in different weather conditions.

2.1.2 LGVSL

LG Silicon Valley Lab (LGSVL) is an open-source simulation engine developed by LG Electronics. It utilizes the Unity gaming engine to render photorealistic environments and taking advantage of technologies like the High-Definition Render Pipeline (HDRP) from Unity.

This simulator is developed in two parts: the Simulation Engine and the User Autonomous Driving (AD) Stack [3]. The Simulation Engine, a customizable open-source platform, receiving its inputs from AD stack and simulate the environment, sensors, and vehicle dynamics. The AD Stack comprises three major layers: Perception, Planning, and Control, offering various user-configurable functionalities. The AD Stack and the Simulation Engine is connected through communication bridge interface, such as Cyber RT, ensuring seamless integration [3]. While the simulator comes with various default sensors including cameras, LiDAR, and Radar, its unique feature lies in its adaptability. Users can build and configure their own sensors and importing models of real-world sensors as plugins. For instance, the plugin for Velodyne VLP-16 LiDAR replicates point cloud generation similar to its actual counterpart [3]. These sensors' data and its mounting positions are defined through JSON-formatted text, simplifying their utilization. Figure 2 showcases the array of default sensors available within this simulator.

Figure 2: Different types of sensors. Left (top to bottom): Fish-eye camera,LiDAR, Radar; Right (top to bottom): Segmentation, Depth, 3D Bounding Box.

A distinguishing aspect of this simulator is its capability to incorporate real-world maps to construct virtual environments within the simulator. Map formats like Lanelet2, OpenDRIVE, and Apollo 2 HD map can be imported and used as the virtual environment. This features of LGSVL appeals to engineers of automakers and making it a highly suitable tool for their research.

2.1.3 SUMMIT

SUMMIT (The Simulator for Urban Driving in Massive Mixed Traffic) is an open-source simulator developed as an extension of the CARLA simulator [2], inheriting its physics and visual realism. Unlike many other simulators that predominantly which simulate rule-based traffic with minimal randomness, SUMMIT stands out for its ability to replicate the aggressive and chaotic nature of real-world traffic. This distinctive feature attracts users interested in training and testing algorithms for vehicles navigating unregulated traffic scenarios. SUMMIT employs the 'Context-GAMMA', a velocity-space optimization crowd behaviour algorithm [4] to simulate traffic behaviour geometrically and topologically. Additionally, it utilizes real-world maps from OpenStreetMap, extracting features such as roads, sidewalks, and roundabouts which are incorporated into the simulator, enabling the replication of real-world maps. An illustrative example can be seen in Figure 3, which showcases the comparison between a real map and its counterpart with unregulated traffic behaviour in SUMMIT at the Magic Roundabout in England.

Figure3: Scenes in the real world and corresponding scenes in SUMMIT

SUMMIT's utilization of real-world maps, combined with CARLA's visual realism and the simulation of chaotic traffic behaviour, significantly enhances its capabilities and making is a best option for inroad commercial autonomous vehicles research.

2.2 Comparative Study on Simulators

This section examines notable studies that compare existing autonomous vehicle simulators and summarizes their approach and results.

In Guan Yang et al. work (2021), "Survey on Autonomous Vehicle Simulation Platforms," [5] the team extensively researched different autonomous vehicle simulation platforms. The objective of a simulator is broken down into five parts: Static environment simulation, Dynamic environment and behaviour simulation, Traffic flow simulation, Sensor simulation, and Vehicle dynamics simulation. They also established a taxonomy for existing simulators, categorizing them into Point Cloud-based and 3D Engine-based platforms. Point-based simulators, such as CarCraft from Waymo and Apollo from Baidu, reconstruct the environment based on simulated sensor data. Figure 4 displays the map from Apollo, a point-based simulation platform. On other hand, 3D engine-based platforms, like PanoSim, utilize gaming engines like Unity and Unreal to render environments following laws of physics (Figure 5).

Figure 4: Scene from Apollo simulator

Figure 5: Scene from PanoSim simulator

They further created a table comparing some of popular and relevant simulation platforms and their features [5] (Figure 6).

Figure 6: A Comparison table of various simulator

Although this table aids in comparing simulators roughly, it doesn’t compare sufficient features for making a concrete decision, and lacks a single metric defining the level of usability of the simulator for a user. While the categorization of simulators is provided, a clear comparative method among simulators is not clearly defined.

In Md Salman Ahmed et al.'s work (2016), an extensive study on connected vehicle simulators was presented [6]. The focus was on the domain of connected vehicles, including vehicle-to-vehicle and vehicle-to-server communication. The paper assessed several simulators based on their memory consumption, computing environment (Sequential or Parallel), and the number of vehicles they could handle. However, these results are specific to the connected vehicle domain and may not be applicable to other types of autonomous vehicle simulators effectively.

2.3 Generative Adversarial Networks

Generative adversarial networks (GAN) mark a significant advancement in the domain of Generative AI, first introduced in 2014 by Goodfellow et al. in the paper "Generative Adversarial Networks" [7]. Since then, GANs have gained substantial momentum in the field of Generative AI especially in image generation. This section discusses some noteworthy works within the domain of GANs.

2.3.1 VGAN (Video Generative Adversarial Network)

VGAN, developed by Carl Vondrick et al. [8], specializes in generating videos with its scene dynamics. The model is capable of generating videos up to a second at full frame rate. Its training involves over 2 million pre-processed videos sourced from the internet, categorized into four distinct groups: golf courses, hospital rooms, beaches, and train stations.

The architecture of VGAN employs a standard Generator-Discriminator structure. The generated video is segmented into two features: foreground and background, assuming a fixed camera resulting in a static background. The generator comprises two network streams dedicated to foreground and background, respectively. The foreground stream consists five layers of 3D spatiotemporal convolution layers (time x width x height), which upsamples the information from a low-dimensional latent code z, sampled from a standard normal distribution. A masking layer 'm' is introduced before the final layer, outlining the pixels of objects in the foreground. Meanwhile, the background stream utilizes five layers of 2D convolution layers (width x height), responsible for generating a background 'b'. The background stream uses 2D convolution layers as the background is assumed to be same for all the generated frames. The synthesis of foreground and background follows the equation

m@f+(1-m)@b

The resultant video, comprising 32 frames with dimensions of 64x64, is generated from a 100-dimensional latent code sampled from a normal distribution. The Discriminator is designed for two primary objectives: classifying realistic scenes and recognizing plausible and smooth motion between frames. It mirrors the architecture of the foreground stream of generator with a five-layered spatiotemporal convolutional setup, employing downsampling instead of upsampling. The final layer outputs a binary classification (real or not). Batch normalization and ReLU activation functions are used after every layer in both the generator and discriminator.

The VGAN is trained using Adam optimizer with a batch size of 64. Results demonstrate the model's ability to generate videos with a sharp background and a slightly blurry foreground. While the resolution of the foreground might be blurred, the motion of the generated foreground are convincing. However, the user had no control over the content of generation. Figure 7 illustrated the frames of various generated videos.

Figure 7: Videos generated using VGAN

2.3.2 ImaGINator

ImaGINator is a conditional Generative Adversarial Network (GAN) developed by Yaohui Wang et al [9]. Its primary aim is to produce human videos depicting various expressions. Unlike VGAN [8], this model generates videos conditioned on specific class labels for expressions for example smiling, jogging... The model is designed to segment the generated video into spatial and temporal parts.

ImaGINator comprises a generator and two discriminators. The generator adopts an encoder-decoder architecture with skip connections. It takes a static image featuring a person's face as input and encodes it into a latent vector 'p'. In addition, this vector is concatenated with one-hot encoded class label c, representing the expression, and random noise sampled from a standard normal distribution. This fusion embeds spatial (p) and temporal information (c) into the latent vector. The decoder, structured as a (1+2)D convolution layer, explicitly separates temporal and spatial information. It mirrors the dimensions of the encoder's architecture and had skip connections from encoder, ensuring that each decoder layer retains embedded spatial details from the corresponding encoder layer. Moreover, the embedded class label vector is integrated into every decoder layer, ensuring the preservation of temporal information throughout the model. The generated video consists of a fixed number of frames and the last layer of the generator outputs an image with all the frames.

Two discriminators serve distinct purposes: 'Di' evaluates individual frames of the generated video to classify real from fake based on appearance, while 'Dv' examines the sequence of frames alongside the class label to classify the dynamics within the frames as real or fake. The Generator is optimized on the combined loss function from both discriminators and a reconstruction loss for corresponding frames. Each discriminator's loss function optimizes the weights of corresponding discriminator. The ADAM optimizer with the same learning rate is used across all generator and discriminator components.

Evaluation metrics such as Frechet Inception Distance (FID), Peak Signal-to-Noise Ratio (PSNR), and Structural Similarity Index Measure are used to evaluate the performance of the generated videos and to monitor the training progress. Figure 8 provides a visual representation of the frames within the generated video.

Figure 8: Frames of the video generated using ImaGINator

The model was trained on various widely used datasets like MUG Facial Expression Dataset [13], NATOPS Aircraft Handling Signal Dataset [14], Weizmann Action Dataset [15], and UvA-NEMO Smile Dataset [16]. Comparative analysis shows the superior image quality compared to VGAN. Furthermore, the model's ability to control the content of generated videos using class labels holds significant potential across various applications.

2.3.3 MoCoGAN

Motion and content decomposed GAN (MoCOGAN), a Video Generation GAN introduced by Sergey Tulyakov et al. in 2016 [10], operates on a unique architecture that segments videos into Content and Motion. This segmentation allows the model to sample inputs separately from the Content and Motion subspaces (unsupervised). Notably, this architecture enables the model to generate videos depicting the same content with different actions or the same action with different content. While designed for shorter video durations, MoCOGAN doesn't require a fixed length for the generated videos. Given the shorter duration, the video's subject is assumed to remain constant.

The model comprises four networks: a Recurrent Neural Network (Rm), Generator (G1), Image Discriminator (Di), and Video Discriminator (Dv). The Generator sequentially produces frames by taking a latent image Z as input, containing zc and zm. The random vector zc represents the video's content and is sampled from a content subspace, remaining constant throughout the video as the subject remains unchanged. On the other hand, the latent vector Zm, which determines the motion trajectory of the subject, and it is recursively outputted by Rm by sampling from a motion subspace at each timestep.

The Rm's parameters are learned during training, as not all motion trajectories are physically possible. The generator G takes Z (combination of Zc and Zm) and generates video frames sequentially. The Image Discriminator assesses frame quality, while the Video Discriminator evaluates subject motion in the video. The RNN is trained exclusively on the Video Discriminator loss, while the entire Generator network is trained on the combined loss of the discriminators.

The model's training utilizes the Weizmann Dataset [15] and Tai-Chi Dataset [17], and its performance was assessed using the Average Content Distance (ACD). Visual representations in Figure 9 depict video clip frames generated by the MoCoGAN model.

Figure 9: Frames of video generated by MoCoGAN

Notably, MoCoGAN had outperformed VGAN [8] and TGAN in image quality within the videos and offers the flexibility of generating videos of varying lengths. However, it lacks direct control over content creation like ImaGINator as content is randomly sampled from the content subspace

2.3.4 Pix2pix GAN

Pix2Pix, introduced in 2016 by Philip Isola et al [11], is an image translation technique using Conditional GANs. Image translation involves transforming images from one domain to another, such as converting grayscale images to colour images. In [12], the method was applied to translate satellite terrain images into Google Maps style images.

The structure of the conditional GAN resembles that of a Normal GAN, comprising a Generator and a Discriminator. In [12], the generator adopts an encoder-decoder architecture, where the encoder consisting of blocks that include a convolution layer, Batch Normalization, and ReLU activation function. The decoder mirrors the encoder, employing skip connections (known as U-Nets) between corresponding layers to retain information and prevent loss during transmission through bottle necks [11]. The Generator takes an image from the input domain and produces an image in the target domain. On the other hand, the Discriminator intakes the paired images from both domains, uses standard convolutional layers, converging in a final layer that determines the given image (real or fake). Using PatchGAN, the Discriminator segments images into 70x70 patches and performs binary classification for each patch, this makes the discriminator suitable for any given image size.

Training involves computing the reconstruction loss between the generated image and the target image. The generator's gradients are optimized to minimize this loss. The composed model training combines the Discriminator loss and the reconstruction loss using a weighted sum for gradient update. The dataset used for training, consisting of 1097 paired images of satellite and Google Maps images. Each image is pre-processed and rescaled to 256X256 pixels before training. Figure 10 illustrates the satellite images and its corresponding google map image and generated image.

Figure 10: The satellite images and its corresponding google map image and generated image after 10 epochs

While image translation using this method has diverse applications, this method often requires paired datasets, which is difficult to obtain in most of the cases.

List of figures

Figure 1: Scenes from the CARLA simulator in different weather conditions.

Figure 2: Different types of sensors. Left (top to bottom): Fish-eye camera,LiDAR, Radar; Right (top to bottom): Segmentation, Depth, 3D Bounding Box

Figure3: Scenes in the real world and corresponding scenes in SUMMIT

Figure 4: Scene from Apollo Simulator

Figure 5: Scene from PanoSIM simulator

Figure 6: A Comparison table of various simulator

Figure 7: Frames of the video generated using VGAN

Figure 8: Frames of the video generated using ImaGINator

Figure 9: Frames of the video generated using MoCoGAN

Figure 10: The satellite images and its corresponding google map image and generated image after 10 epochs

List of Tables

Table1: Levels of driving automation defined in SAE J3016

References

[1] https://www.sae.org/news/2019/01/sae-updates-j3016-automated-driving-graphic

[2]: Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A. &amp; Koltun, V.. (2017). CARLA: An Open Urban Driving Simulator. <i>Proceedings of the 1st Annual Conference on Robot Learning</i>, in <i>Proceedings of Machine Learning Research</i> 78:1-16 Available from https://proceedings.mlr.press/v78/dosovitskiy17a.html.

[3]: G. Rong et al., "LGSVL Simulator: A High Fidelity Simulator for Autonomous Driving," 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC), Rhodes, Greece, 2020, pp. 1-6, doi: 10.1109/ITSC45102.2020.9294422.

[4]: Cai, Panpan, et al. "Summit: A simulator for urban driving in massive mixed traffic." 2020 IEEE International Conference on Robotics and Automation (ICRA). IEEE, 2020.

[5]: G. Yang et al., "Survey on Autonomous Vehicle Simulation Platforms," 2021 8th International Conference on Dependable Systems and Their Applications (DSA), Yinchuan, China, 2021, pp. 692-699, doi: 10.1109/DSA52907.2021.00100.

[6]: M. S. Ahmed, M. A. Hoque and P. Pfeiffer, "Comparative study of connected vehicle simulators," SoutheastCon 2016, Norfolk, VA, USA, 2016, pp. 1-7, doi: 10.1109/SECON.2016.7506701.

[7]: Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2020). Generative adversarial networks. Communications of the ACM, 63(11), 139-144.

[8]: Vondrick, Carl, Hamed Pirsiavash, and Antonio Torralba. "Generating videos with scene dynamics." Advances in neural information processing systems 29 (2016).

[9]: Wang, Yaohui, et al. "Imaginator: Conditional spatio-temporal gan for video generation." Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. 2020.

[10]: Tulyakov, Sergey, et al. "Mocogan: Decomposing motion and content for video generation." Proceedings of the IEEE conference on computer vision and pattern recognition. 2018.

[11]: Isola, Phillip, et al. "Image-to-image translation with conditional adversarial networks." Proceedings of the IEEE conference on computer vision and pattern recognition. 2017.

[12]: Henry, Joyce, Terry Natalie, and Den Madsen. "Pix2Pix GAN for Image-to-Image Translation." Research Gate Publication (2021): 1-5.

[13]: N. Aifanti, C. Papachristou and A. Delopoulos, "The MUG facial expression database," 11th International Workshop on Image Analysis for Multimedia Interactive Services WIAMIS 10, Desenzano del Garda, Italy, 2010, pp. 1-4.

[14]: Y. Song, D. Demirdjian and R. Davis, "Tracking body and hands for gesture recognition: NATOPS aircraft handling signals database," 2011 IEEE International Conference on Automatic Face & Gesture Recognition (FG), Santa Barbara, CA, USA, 2011, pp. 500-506, doi: 10.1109/FG.2011.5771448.

[15]: Mauthner, Thomas, Peter M. Roth, and Horst Bischof. Action recognition from a small number of frames. na, 2009.

[16]: M. Taskirar, M. Killioglu, N. Kahraman and C. E. Erdem, "Face Recognition Using Dynamic Features Extracted from Smile Videos," 2019 IEEE International Symposium on Innovations in Intelligent SysTems and Applications (INISTA), Sofia, Bulgaria, 2019, pp. 1-6, doi: 10.1109/INISTA.2019.8778400.

[17]: Sun, Shan, et al. "Taichi: A fine-grained action recognition dataset." Proceedings of the 2017 ACM on International Conference on Multimedia Retrieval. 2017.