University of Nevada, Reno

Intelligent Transportation Systems and Virtual Reality as Technology Accelerators for Smart Infrastructure Monitoring

A thesis submitted in partial fulfillment of the
requirements for the degree of Master of Science
in Computer Science and Engineering

by

Niloofar Malekghaini

Dr. Frederick C. Harris Jr., Advisor

August, 2024

THE GRADUATE SCHOOL

We recommend that the thesis
prepared under our supervision by

Niloofar Malekghaini

Entitled

Intelligent Transportation Systems and Virtual Reality as Technology Accelerators for Smart Infrastructure Monitoring

be accepted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE

Dr. Frederick C. Harris, Jr.,
Advisor,

Dr. Hamed Ebrahimian
Graduate School Representative

Dr. Sergiu M. Dascalu
Committee member

Markus Kemmelmeier, Ph.D., Dean,
Graduate School

August, 2024

Abstract

This work extends the application of intelligent transportation systems (ITS) to the field of structural health monitoring (SHM) for bridge infrastructures and develops a virtual reality (VR) platform as a technology accelerator. In the proposed SHM method, video recordings of traffic are captured using traffic cameras and processed using a combination of YOLO and Deep SORT models, along with projective mapping, to extract the time history of vehicle tire locations. This step, referred to as traffic tracking, predicts the points at which traffic loads excite the bridge. This information is integrated with the bridge's vibration response and the finite element (FE) model to develop a digital twin of the bridge. The necessary models for the traffic tracking component are trained using Google Open Images and tested in a VR environment that simulates traffic on a roadway with installed cameras. Utilizing VR instead of real-world tests significantly accelerates technology development. After achieving promising results in the VR environment, the prototype of the proposed technology has been developed and deployed on a real-world bridge.

Dedication

I dedicate this thesis to my family.

Acknowledgments

I would like to thank my committee Dr. Frederick C. Harris, Jr., Dr. Hamed Ebrahimian, and Dr. Sergiu M. Dascalu for their invaluable guidance throughout my graduate studies.

This material is based in part upon work supported by the National Science Foundation under grant number OIA-2019609 and United States Department of Transportation Small Business Innovative Research (SBIR) program Phase II under Contract number 6913G619C100048.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

Contents

List of Tables

List of Figures

List of Acronyms

Abbreviation	Full Name
ASCE	American Society of Civil Engineering
AP	Average Precision
CNN	Convolutional Neural Network
COCO	Common Objects in Context
Deep SORT	Deep Simple Online and Real-Time Tracking
EY	East Yutan
FE	Finite Element
FN	False Negative
FP	False Positive
GPS	Global Positioning System
IoU	Intersection Over Union
ITS	Intelligent Transportation Systems
mAP	Mean Average Precision
NDE	Non-Destructive Evaluation
NMS	Non-Maximum Suppression
OMA	Operational Modal Analysis
PDF	Probability Distribution Function
SHM	Structural Health Monitoring
SORT	Simple Online and Real-Time Tracking
TP	True Positive
VR	Virtual Reality
YOLO	You Only Look Once

Chapter 1

Introduction

America's bridge infrastructure, which is a vital component of its transportation network, is aging and deteriorating. With over 600,000 bridges across the U.S., nearly 40% are 50 years or older. In 2019, one in 13 bridges were structurally deficient and at the current rate of improvement, it is estimated to take more than 50 years to carry out all the repairs that are currently necessary. Replacing or rehabilitating such a vast number of bridges cannot be done quickly. As a result, structurally deficient and outdated bridges will continue to be in use for an extended period. To ensure public safety, it is essential to develop techniques and solutions for operational bridge monitoring, providing actionable information to guide maintenance and rehabilitation decisions.

Traditionally, various methods have been employed for monitoring bridge structures. Among these, non-destructive evaluation method [58, 78] is the most prevalent due to its high resolution for damage localization [11]. However, widespread adoption of NDE faces significant challenges due to its high cost, labor-intensive nature, and inevitable traffic disruption [66]. With the increasing number of aging bridges [95] and according to American Society of Civil Engineering (ASCE) guideline, it is a need for introducing new methods for operational health monitoring of bridges. These new methods should be technology-based, cost-effective, and involve minimal installation and traffic disruption.

Over recent decades, digital twinning using output-only time-domain finite element (FE) model updating has become a valuable tool for damage localization of

civil structures as a cost-effective with minimal installation load work approach [23, 51, 84]. This method employs physics-based models and structural measurements, combined with stochastic filtering approaches like Bayesian inference, to estimate unmeasured input excitation and model parameters reflecting the health status of structure. Bayesian output-only time-domain FE model updating has been successfully applied to structures subjected to unknown temporal input histories at known invariable locations, such as earthquake and wind loads [20, 30, 65, 88]. However, its application for operational health monitoring of bridges is challenging due to the spatial and temporal variability of traffic excitation. Temporal variability arises from interactions between vehicle suspension systems and bridge dynamic systems, road roughness, tire assemblies, and driveline excitation [32]. Spatial variability results from vehicle movements along the bridge.

Recent advancements in artificial intelligence and computer vision techniques present promising solutions to the challenges of digital twinning for operational health monitoring of bridges [13, 38, 56]. It is now feasible to capture video recordings of traffic and use them to classify vehicles and track the footprints of their tires on bridge decks. This study proposes utilizing detection, classification and tracking algorithms to estimate the static weight and spatial variability of traffic load on bridge. This data serves as input for the bridge digital twinning process. During this process, the tire history of the dynamic load of traffic as well as the damage status of bridge are determined. The identified damage status of bridge is used for asset management and maintenance of structure, while the traffic tracking data and dynamic load estimates can be used to improve intelligent transportation systems (ITS) through traffic management and weight in motion.

In the proposed method of this study, networks of traffic camera and accelerometers are installed on the bridge to collect video recordings of traffic traversing on the bridge and the resulting vibration response of the bridge. By implementing the proposed traffic tracking algorithm, a sequence of You Look Once (YOLO) [77] and Deep Simple Online and Real-Time Tracking (Deep SORT) [96] models are used to

estimate the class of vehicles and time history of the location of their tires. Using perspective transformation, the tracked tire locations are converted from pixel coordinates to a world or FE model coordinate system. This information is then fused with vibration data and used as input for the digital twinning process.

This study focuses on the development and evaluation of the computer vision component for tracking the time history of the location of tires. This step is referred to as the traffic tracking technology component. To achieve the goal of this study, a virtual reality (VR) [41, 59] testing platform is developed using Unity [91] which significantly accelerates technology development by accurately simulating roadway test scenarios for system components under evaluation. In this platform, the bridge roadway is simulated and equipped with a network of cameras. As the traffic traverses the simulated bridge road, the VR platform records the view of cameras as well as the ground truth time history of tire locations as in a world coordinate system. The recorded videos are then processed using the proposed traffic tracking algorithm to track tire locations in the world coordinate system. By comparing the estimated locations with the ground truth data, the performance of the proposed traffic tracking algorithm is evaluated. Additionally, this method is assessed in a real-world setting by instrumenting the East Yutan (EY) [68] bridge in Nebraska with networks of traffic cameras. Unlike the VR simulation environment, the real-world setting does not provide ground truth data, so the evaluation focuses on the feasibility of performing detection and tracking, with manual review ensuring reasonable results.

The rest of this thesis is structured as follows: Chapter 2 provides background knowledge integral to understanding the material of this work. It includes the evolution of bridge structural health monitoring methods, challenges for application of digital twinning for operational health monitoring of bridges, potential application of ITS for bridge digital twinning as well as application of VR technologies to simulate test experiments that are financially and practically not feasible to conduct entirely in a real-world setting. Chapter 3 details the proposed method for operational health monitoring of bridges. This chapter also elaborates the proposed traffic tracking

component and its main components, as well as the role of VR in development of the technology components. Chapter 4 outlines the methodology including the development of perspective transformation, training and evaluation of deep learning models, and the VR platform. Chapter 5 shows the experiments and results in the VR and real-world setting. Finally conclusions as well as future work are contained in Chapter 6.

Chapter 2

Background and Related Work

2.1 Bridge Structural Health Monitoring and Traditional Methods

Transportation systems plays an important role in economic growth and social development of any nation. One of the essential components of transportation system is its network of bridges, which enhances mobility and facilitates logistics by connecting countries, regions, and communities. Therefore, appropriate maintenance and life cycle management of bridges are necessary to ensure transportation safety. Structural health monitoring (SHM) of bridges are a family of methods to monitor the performance behavior of bridge structures and detect, localize, and quantify damage within the structure. These methods are utilized to identify safety threats at an early stage, allowing for the prioritization of repair, retrofit, or replacement efforts.

Traditionally, various methods have been employed for SHM of bridge, including visual inspections [78], vibration-based health monitoring [28, 63], and non-destructive evaluation (NDE) [11] methods. Visual inspection is one of the most common monitoring approaches which involves a detailed examination of the visible components of a structure to identify any signs of damage or deterioration. During a visual inspection, inspectors look for indicators of various damage mechanisms such as concrete deterioration (including concrete delamination and concrete degradation) and steel corrosion [9]. Examples of these damage mechanisms are shown in Figure 2.1. Visual inspection is costly due to the need for specialized equipment and

(a) (b) (c)

Figure 2.1: Different damage mechanisms: (a) bridge deck delamination in a drilled hole, and (b) bridge deck degradation due to alkali-silica reaction, and (c) Steel rebar corrosion.

trained personnel. The method is also labor-intensive due to the requirement of significant time and effort to access and examine all parts of the structure. Moreover, the results are inherently subjective, as the method relies on the inspector's experience and judgment. Most importantly, in this approach, damage mechanisms might be undiscovered until catastrophic failure in cases where damage is hidden and not visible during inspections [7, 34, 62]. Figure 2.2 shows an example of bridge visual inspection.

Vibration-based methodologies encompass both data-driven [28] and model-based approaches [63], aiming to detect structural damage by analyzing the vibration responses of bridge structures during regular operation, such as passage of traffic. The vibration responses of bridge is measured using traditional sensors, e.g. accelerometers and are used to extract modal parameters from the vibration responses [17]. Modal parameters, which typically include natural frequencies and mode shapes of structure, serve as fundamental indicators reflecting the characteristics and condition of the bridge. Figure 2.3 shows an example of derived modal properties for a bridge. Operational modal analysis (OMA) [10, 14, 83] refers to techniques determining the modal properties of structures using vibration measurements collected under operating conditions. However, the precision of these techniques are compromised by various interference [54, 73], thereby diminishing the accuracy of vibration-based SHM [16]. Additionally, the modal properties that can be reliably identified often do not effectively capture aging- and degradation-related damage mechanisms in bridge

Figure 2.2: Visual inspection of bridge.

structures [26]. Such damage typically identifiable in higher structural modes, the identification of which poses challenges and uncertainties.

NDE [78] techniques are the most accurate methods that are used in practice for SHM of bridges. These techniques have the capability to identify different damage mechanisms and localize them with a high resolution [11] through the utilization of advanced sensors, imaging techniques, and signal processing algorithms. However, despite their accuracy, the widespread adoption of NDE techniques poses significant challenges. Firstly, the implementation of NDE methods incurs substantial costs, primarily due to the procurement and maintenance of the equipment [58, 66]. Secondly, conducting NDE often necessitates the temporal closure of traffic flow on bridges to ensure the safety of personnel and equipment during inspection activities [58]. The traffic interruption can result in logistical complexities, inconvenience to commuters, and potential economic impacts on transportation networks. Operating NDE equipment and interpreting the results of inspection require specialized expertise and

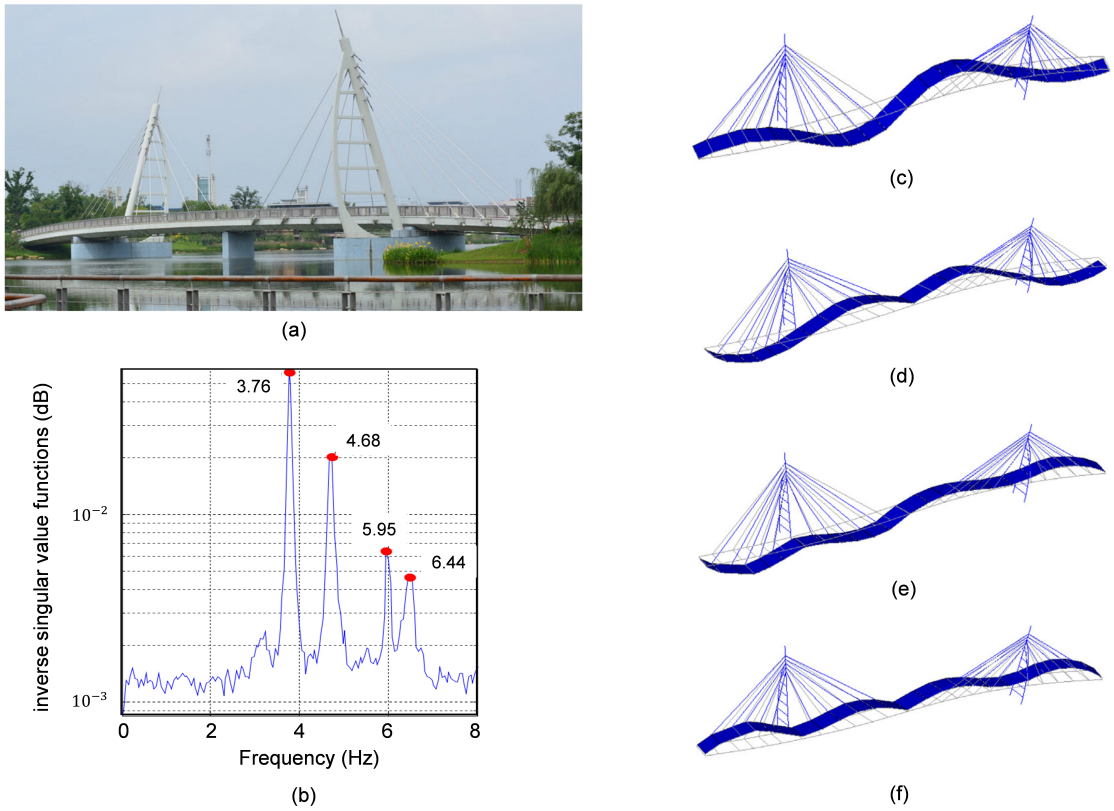


Figure 2.3: Modal properties: (a) Real-world bridge, (b) Modal frequencies, (c) First mode shape, (d) Second mode shape, (e) Third mode shape, and (f) Forth mode shape..

training to ensure safety, accuracy and reliability [19] which is another challenge for organizations seeking to deploy NDE techniques for bridge SHM. Given the increasing number of aging bridges across the United States, solely depending on NDE methods is cumbersome and inefficient [95]. Hence, there is a need to introduce complimentary screening methods to detect, localize, and quantify damage in bridge members in order to guide the targeted application of NDE techniques.

2.2 Digital Twinning and Its Challenges for Operational Health Monitoring of Bridges

In recent decades, digital twinning has become a valuable tool for damage localization of structures. This approach leverages a physics-based model of the structure [102] alongside stochastic filtering techniques, such as Bayesian inference [21, 30, 84], to update the initial FE model. The model updating process ensures that the updated model accurately represents the actual structural system. For this purpose, statistical frameworks are used to minimize the discrepancies between the measured vibration responses and those predicted by the FE model [37, 45, 81]. The updated model is then maintained as a digital twin of the structure, which can be used for virtual sensing and interrogated for health monitoring and damage diagnosis of the structure [2, 30, 47, 55, 57, 101].

Bayesian output-only time-domain FE model updating is a powerful method for digital twinning, which estimates not only the uncertain mechanics-based model parameters but also the unmeasured input excitation [23, 24, 51, 84]. The estimates of uncertain mechanics-based FE model parameters, facilitate damage localization [56], and the estimates of load of vehicles are used for various purposes, depending on the specific problem under study [30]. The Bayesian inference is schematically shown in Figure 2.4. As can be seen, the measured responses of the structure (shown as Y) are the input to the Bayesian inference to update the mechanics-based model of the structure. In this process, the unknown model parameters as well as the unmeasured inputs (here load of vehicles), presented as $\boldsymbol{\theta}$, are treated as random variables. The uncertainties of these parameters are characterized by a joint Probability Density Function (PDF) – shown as $p(\boldsymbol{\theta})$. The prior uncertainties in unknown parameters, including the mean vector ($\hat{\boldsymbol{\theta}}^{-}$) and covariance matrix ($\hat{P}_{\boldsymbol{\theta}}^{-}$), are propagated through the FE model of the bridge and stochastic FE-predicted responses, $\hat{Y}(\boldsymbol{\theta})$, are estimated. Minimizing the discrepancies between Y and $\hat{Y}_{\boldsymbol{\theta}}^{-}$ using Bayes' theorem results in the posterior estimates of mean vector and covariance matrix, known as $\hat{\boldsymbol{\theta}}^{+}$ and $\hat{P}_{\boldsymbol{\theta}}^{+}$.

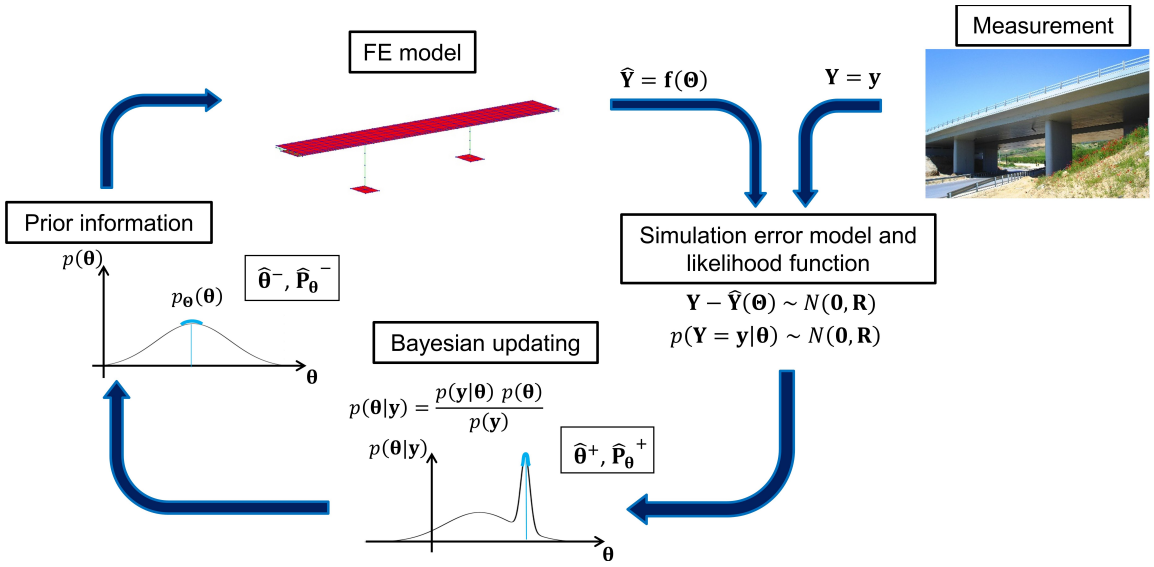


Figure 2.4: Schematic representation of the sequential Bayesian inference method for model updating.

The Bayesian output-only time-domain FE model updating method has demonstrated successful implementation in structures subjected to unknown input excitation time histories (temporally variable) that are applied at known non-variable locations (spatially non-variable) [20, 21, 22, 30, 65, 86, 88]. However, the application of this method for operational health monitoring of bridges presents challenges due to the variability of traffic excitation both temporally and spatially. Temporal variability of traffic arises from the interaction between the suspension systems of vehicles and the dynamic systems of bridges [100], road roughness, tire assembly, and driveline excitation [32]. Spatial variability, on the other hand, results from the movement of vehicles along the bridge. Recent studies focus on leveraging advancements in intelligent transportation systems (ITS) to estimate the spatial variability of traffic. Understanding this spatial variability paves the way for the development of bridge digital twins and SHM of bridges [56].

2.3 Intelligent Transportation Systems

ITS includes a range of technologies and applications designed to enhance the efficiency, safety, and effectiveness of transportation networks [15, 25]. The core part of ITS involves integrating information and communication technologies with transportation infrastructure and vehicles to improve traffic management [99], reduce congestion [89] and enhance safety [48]. On the other hand, ITS has been used to increase the transportation safety in areas such as traffic conflict analysis [1, 79], advanced driver assistance systems [3], speed safety cameras [29], and traffic incident management [99].

Recent advancements in ITS have emphasized the significance of vision-based vehicle detection and tracking algorithms for traffic flow predictions [4, 5, 12, 39, 60, 87, 97, 98]. In the majority of ITS studies, vehicle detection is carried out using You Only Look Once (YOLO) model [77], which is capable to detect, classify and determines the location of detection using bounding boxes. In 2016, YOLO [77] model was published as a state-of-the-art single-stage object detector at that time. YOLO feeds the image once through a single convolutional neural network and predicts the bounding boxes and class probabilities directly from one evaluation. The initial convolutional layers of the network extract features from the image while the fully connected layers predict the output probabilities and coordinates. For this purpose, YOLO divides an input image into a grid of cells and each cell predicts several bounding boxes, confidence scores for those boxes and class probabilities. The confidence scores reflect how confident the model is that the box contains an object, and the probabilities are conditioned on the grid cell containing an object. Then, YOLO calculates class-specific confidence scores for each box by multiplying the conditional class probabilities and the individual box confidence predictions. The YOLO model is schematically shown in Figure 2.5.

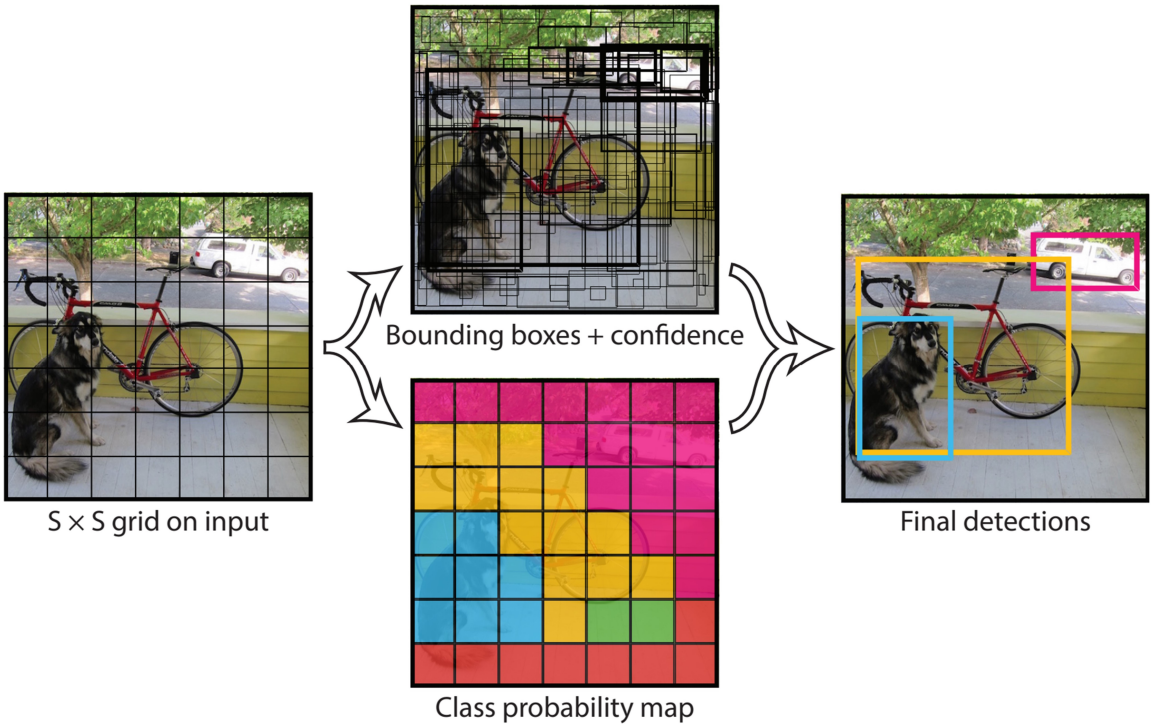


Figure 2.5: YOLO models detection as a regression problem. It divides the image into a grid and for each grid cell predicts bounding boxes, confidence for those boxes, and C class probabilities.

YOLO model has evolved over years to improve the performance [42]. YOLOv8 is one of the latest versions of YOLO, which in addition to detection and classification, it goes a step further and supports segmentation [93]. Instance segmentation involves identifying individual objects in an image and segmenting them from the rest of the image. The output of an instance segmentation model is a set of masks or contours that outline each object in the image, along with class labels. Instance segmentation is useful when both the location and the exact shape of objects in an image are required. Figure 2.6 shows a comparison between application of YOLO for object detection and instant segmentation.

(a)

(b)

Figure 2.6: Application of YOLO for: (a) Object detection, and (b) Instant segmentation.

In 2016, Simple Online and Real-Time Tracking (SORT) algorithm was introduced [44] to achieve tracking of detected objects to maintain their identity across frames as they move or change in appearance. For this purpose, SORT uses Kalman filter [46] and Hungarian matching algorithm [8]. Kalman filter predicts the next state of each tracked object based on its previous state and motion dynamics. After obtaining predictions, Hungarian algorithm associates the predictions with the detection. SORT implements a multi-stage matching process, referred to as matching cascade, where tracks are prioritized based on their age and stability, with more established tracks being matched first. SORT model computes a motion-based cost matrix that used the predicted positions of tracks and the current positions of detection. How-

ever, SORT fails to maintain accuracy due to occlusions and viewpoint changes. An enhanced version of SORT, referred to as Deep Simple Online and Real-time Tracking (Deep SORT) [96], is introduced in [44] incorporating a deep learning-based feature embedding model. Compared to SORT, Depp SORT learns a more discriminative feature representation for each detected object using a convolutional neural network to compare the visual similarity between tracks and detection [92]. In Deep SORT, matching cascade is implemented using a cost calculation that combines both motion and appearance costs. The motion-based cost is derived from the Kalman filter prediction, while the appearance-based cost is based on the similarity of the appearance descriptors. Figure 2.7 depicts the integration of YOLO and Deep SORT for object tracking on a video recording.

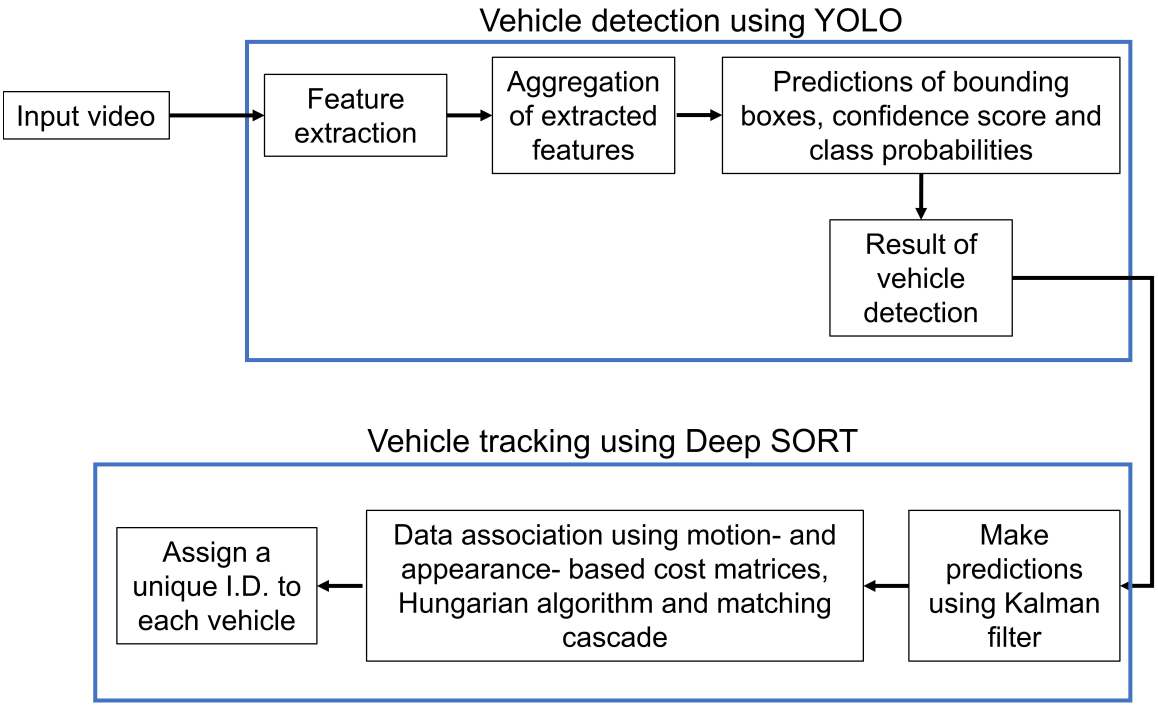


Figure 2.7: Detecting and tracking vehicles using YOLO and the Deep SORT algorithm.

2.4 Synthetic Data and Virtual Reality Technologies in ITS

A difficult task in development of deep learning and computer vision models is collecting large sets of real data for training and validation. This challenge arises due to several factors, including the manual annotation effort required, the difficulty in collecting representative examples of the target scenes or patterns of interest, and, more recently, privacy-related restrictions [67]. To address this issue, a new trend has emerged in research studies utilizing synthetic images as training and/or validation data, either alone or in combination with real images [43, 50, 85, 80].

While synthetic images are two-dimensional (2D) static visuals, virtual reality (VR) is defined as a computer-generated simulation of a three-dimensional (3D) immersive environment that can be interacted with as if it were real [41, 59]. VR has the ability to simulate time-variant 3D environments which makes it suitable to simulate test experiments that are financially and practically not feasible to conduct entirely in a real world setting [6, 36, 64, 74, 75, 76]. For example, [76] designed and implemented a VR-based simulation testbed, linking a computer-generated environment to the system running the autonomous vehicle to visually evaluate the performance of an advanced driver assistance controller in a real traffic. Moreover, [90] used an immersive VR-based driving simulator to study driver behavior, techniques and adaptability.

In recent VR applications, Unity [91] has become a widely used game engine for implementing visualization modules [69, 70, 71]. Unity is a commercial game engine that offers extendable scripting capabilities in the C# language. It employs the object-oriented composition paradigm, in which component classes implement various types of encapsulated behavior that can be attached to container objects, allowing for a modular and flexible development process. Unity's extensible model is integrated with a visual environment editor, enabling developers to change properties visually and create references to other objects within the environment. Additionally,

Unity's extensive asset store provides a vast library of pre-made assets and tools that can accelerate development and improve the quality of VR applications. Figure 2.8 shows an example of developed environment in Unity.

Figure 2.8: Platoon of cars within the virtual Clemson University environment developed using Unity [76].

Chapter 3

A New Paradigm: Application of ITS in Bridge Health Monitoring

This study introduces ITS as a key technology component in an innovative technology-based approach for SHM of bridges. This technology component combines computer vision techniques and deep learning methods within a digital twinning framework. Moreover, a VR simulation is introduced as a platform for accelerating technology development by simulating roadway test scenarios for system components under evaluation. Section 3.1 outlines the new health monitoring technology and its pipeline covering the deployment of monitoring sensors and data collection, data pre-processing, digital twinning and interpretation of results. ITS is incorporated in this pipeline as a part of data pre-processing and is referred to as traffic tracking component. Afterwards, a detailed discussion on the traffic tracking technology component is provided in Section 3.2. This section discusses the primary phases of traffic tracking component and the use of deep learning and computer vision models. Finally, Section 3.3 explains the application of VR in acceleration of technology development.

3.1 Big Picture: ITS and Digital Twinning

The digital twin of a bridge refers to a high-fidelity FE model that can be used for damage identification and localization of the bridge structure. Development of bridge digital twins requires understanding regarding the following items:

- Input excitation to the bridge structure, which is determined through:

  - Load of vehicles traversing on the bridge.
  - Time history of the locations in which the load of vehicles are applied to the bridge structure. These are the time history of the footprint of vehicles' tires.

- Output response of the bridge structure, which is determined through:
  - Vibration response of the bridge structure to the excitation induced by traffic, e.g., acceleration or displacement response.

In the proposed technology-based method, the output response is collected using sensors, e.g. accelerometers, installed on bridge. On the other hand, the input excitation is inferred by application of computer vision and deep learning methods on video recordings of traffic captured by traffic cameras installed on the bridge. In this method, video recordings are processed to detect and classify vehicles, and track the location of their tires. The classification results provide initial estimates regarding the weight of vehicles, and the tracking results provide the time history of the location traffic input load. Then, the extracted information on input excitation and output responses are fused together and used as the input to the digital twinning process. In this process the initial FE model of bridge is updated to truly replicate the real-world structure. Using Bayesian output-only time-domain FE model updating, the damage status of bridge as well as the dynamic load of vehicles are estimated. Later, using estimates of damage status of various bridges, the prioritization plan for rehabilitation of bridges are developed. Moreover, using the estimates of vehicles dynamic load, remote load rating and traffic management can be possible. Figure 3.1 presents the schematic view of the proposed technology-based method for digital twinning of bridges and highlights its four main steps: Instrumentation and data collection, Data pre-processing, Digital twinning, and Interpretation of estimates. Figure 3.2 shows the pipeline of proposed method and discusses its implementation as outlined below:

- **Step 1: Instrumentation and data collection**

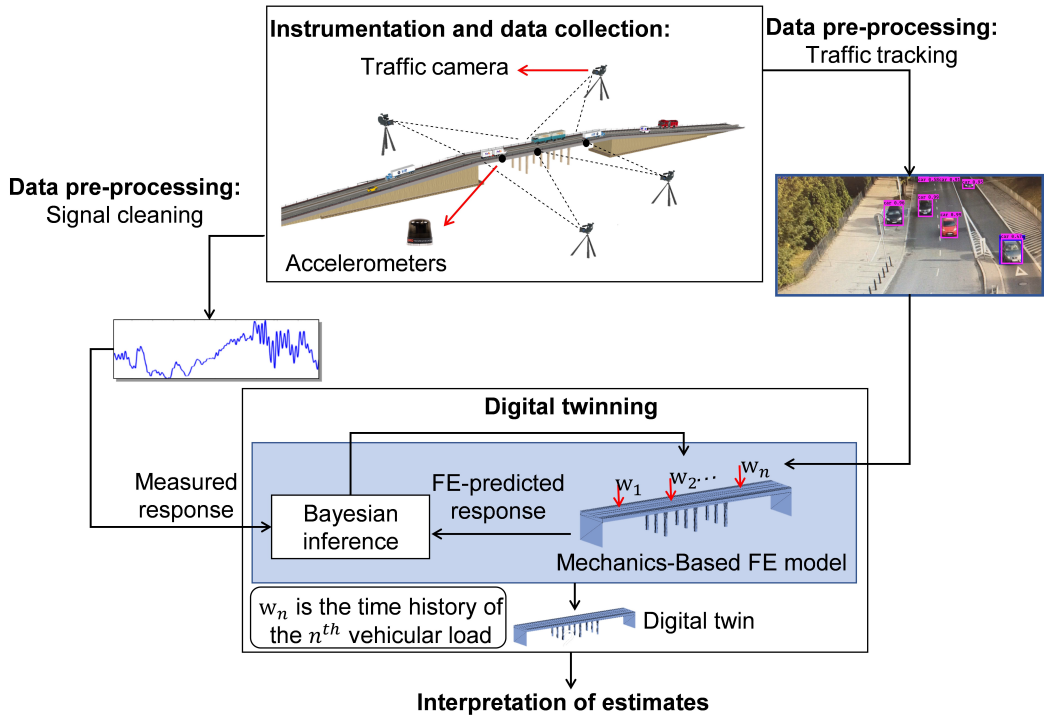


Figure 3.1: Schematic representation of proposed method for operational health monitoring of bridges.

The bridge is instrumented with traffic cameras and accelerometers to collect the video recordings of traffic and dynamic response of bridge to the given traffic.

- The accelerometers are installed at locations in which the response of structure has the least noise-to-signal ratio and the highest sensitivity to the deterioration of structure.
- The traffic cameras are installed along the length of bridge in a way that their vantage point provide a continues coverage over the length of bridge.

- **Step 2: Data pre-processing**

  Prior to being used in the digital twinning process, the collected data may require processing.

  - The video recordings of traffic are processed using deep learning and computer vision techniques to detect the class of vehicles and track the location

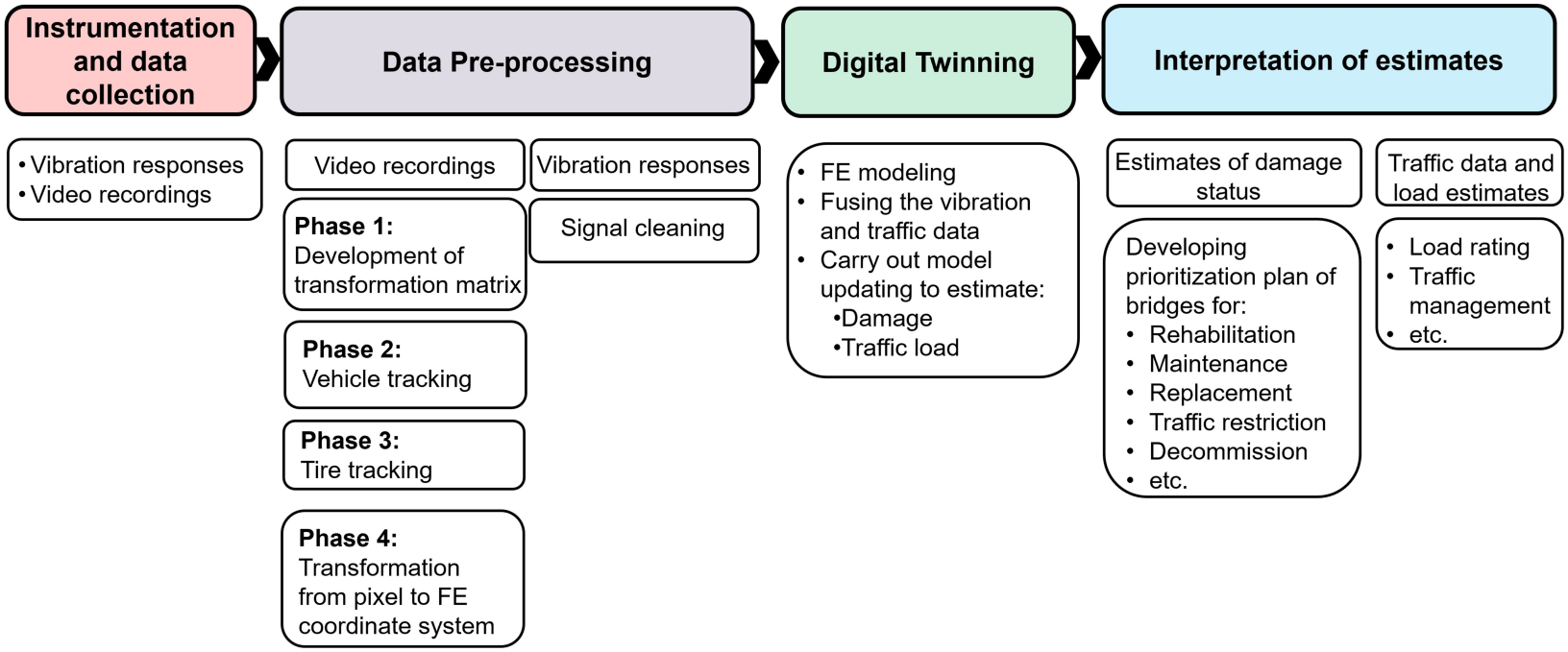


Figure 3.2: Pipe line for the proposed method.

of their tires on the bridge deck. This task is referred to as *traffic tracking component* and includes four main phases: Development of transformation matrix, vehicle tracking, tire tracking and transformation to FE coordinate system. These phases will be discussed in detail in Section 3.2.

  - The utilization of acceleration signals do not require any specific processing other than usual cleaning process.

- **Step 3: Digital twinning**

  Using the pre-processed collected measurements, the digital twin of bridge is developed. For this purpose, the following steps are carried out:

  - Setting up the Bayesian inference and selecting the FE model parameters to be estimated. This part is not the focus of this study.
  - Preparing the input to the Bayesian inference:
    * The initial FE model of the bridge structure is developed. This model is the high-fidelity FE model developed using the available information about the bridge structure, e.g., as-built drawings and material nominal values.
    * The collected acceleration signals and tracked location of tires are synchronized, and fused with the identified class of vehicles.
  - Running the model updating process and estimate:
    * The damage-related FE model parameters.
    * Time history of the dynamic load of traffic traversing on the bridge.

- **Step 4: Interpretation of estimates**

  At this stage, the estimates of damage-related FE model parameters, the dynamic load of vehicles as well as the traffic tracking data are interpreted as follows

  - The estimates of damage-related model parameters are used to infer the deterioration level of bridge and the required maintenance action.

- The estimates of traffic tracking and load are used to improve ITS through load rating and traffic management.

3.2 Key Technology Component: Traffic Tracking

The pre-processing of video recordings, outlined as a sub task in step 2 of the proposed method in Section 3.1, is the main component of the proposed technology. This technology component is proposed to be implemented in four main phases (see Figure 3.2. For each recorded frame:

- In the **first phase**, a perspective transformation matrix is established from the pixel to FE model coordinate systems. This matrix is developed using the intrinsic and extrinsic parameters of the cameras as well as the known locations of fixed landmarks in the pixel and FE model coordinate systems [18, 31, 82].

- In the **second phase**, vehicles are detected and classified using YOLOv8 algorithm resulting in the classification of vehicles and generation of corresponding bounding boxes around the detected vehicles. Subsequently, the detection results are integrated with Deep SORT algorithm, which extracts patches corresponding to the bounding boxes to track the trajectory of each vehicle.

- Moving to the **third phase**, YOLOv8 instance segmentation is performed within each bounding box to detect and predict the segmentation mask of vehicle's tires. The predicted masks of tires are then tracked individually and the coordinates of their contact points with the bridge's deck are determined in the pixel coordinate system.

- In the **fourth phase**, the locations of tires' contact points are computed in the FE model coordinate system. By applying the derived perspective transformation matrix to the pixel coordinates of the tires' contact points, the location of these points can be computed in the FE model coordinate system. By integrating the tracked location of tires from different camera views, a complete

understanding of each vehicle's tires' locations on the bridge is derived.

3.3 Technology Development Accelerator: VR

Development and test of the introduced traffic tracking technology component requires significant efforts on instrumentation of bridge roadways with traffic cameras, collection of video recordings, as well as measurement of the ground truths of tire locations. To speed up the technology development and reduce the test cost, this study proposes development of a VR platform which is capable of simulating a bridge's roadway with cameras installed on the bridge shoulders. Using VR, different test scenarios are simulated while the traffic is being recorded by cameras and the ground truth data are measured. The video recordings are used to implement the developed traffic tracking component and the ground truth data are used to evaluate the performance of the technology component. Further description of the VR platform and its main components are the subject of following sections.

Chapter 4

Methodology

This chapter explains the development of traffic tracking component and the VR simulation platform. Section 4.1 describes the architecture of traffic tracking technology component, as well as the deep learning and computer vision models used in its development. Section 4.2 details on the development of VR platform in Unity, as well as its functional and non-functional requirements.

4.1 Traffic Tracking Component

The architecture of proposed traffic tracking component is shown in Figure 4.1 which shows that the traffic tracking technology component is performed on each frame in four main phases. These phases are explained in details throughout the following subsections. Section 4.1.1 elaborates on development of transformation matrix between the pixel and FE coordinate systems. Section 4.1.2 and Section 4.1.3 describe the training process of detection and tracking models on vehicle and tire objects, respectively. Finally, Section 4.1.4 details the application of developed transformation matrix on the tracked location of tires.

4.1.1 Phase 1: Development of transformation matrix

In each frame of recorded videos, a transformation matrix is developed between the pixel and the FE model coordinate systems. Figure 4.2 shows the commonly used coordinate systems in computer vision and the transformation between them. These

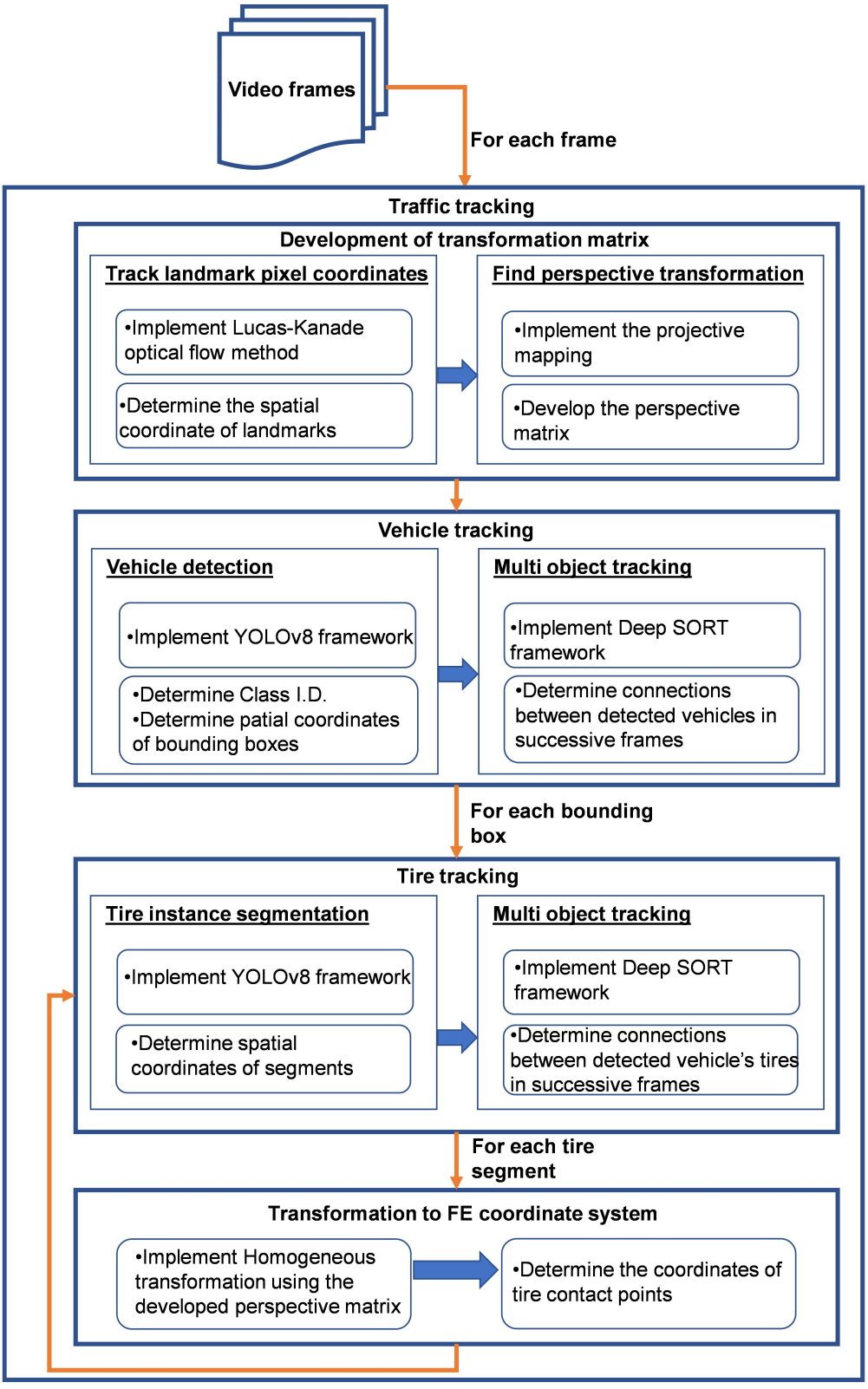


Figure 4.1: Diagram of the proposed method.

systems are explained as follow:

- World coordinate system (or FE model coordinate system here): It is a 3D basic Cartesian coordinate system with an arbitrary origin. A point in this coordinate system can be denoted as $\mathbf{P}_{FE} = (X_{FE}, Y_{FE}, Z_{FE})$.

- Camera coordinate system: It is a 3D coordinate system that measures relative to the camera's origin. The z-axis of the camera coordinate system usually faces outward or inward to the camera lens. The transformation between the world/FE coordinate system to camera coordinate system can be carried out by rotation and translation operations.

- Image coordinate system: It is a 2D coordinate system that has the 3D points in the camera coordinate system projected onto a 2D plane of a camera with a Pinhole Model [40].

- Pixel coordinate system: This 2D coordinate system represents the integer values by discretizing the points in the image coordinate system. Pixel coordinates of an image are discrete values within a range that can be achieved by dividing the image coordinates by pixel width and height.

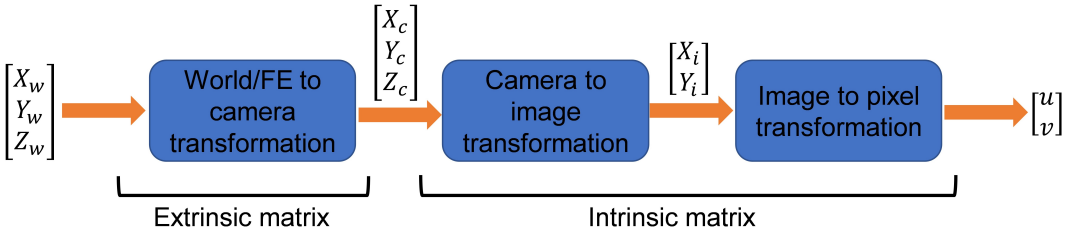


Figure 4.2: Commonly used coordinate systems in computer vision.

Equation 4.1, shows that how the FE model and pixel coordinate systems are related. In this equation, points in pixel coordinate system are shown as $\begin{pmatrix} u \\ v \\ w \end{pmatrix}$. The left hand side 3×4 transformation matrix is the camera intrinsic matrix, and the right hand side 4×4 transformation matrix is the camera extrinsic matrix. Camera intrinsic

matrix, which depends on camera properties (such as focal length, pixel dimensions, resolution, etc.), converts points from the camera coordinate system to the pixel coordinate system. In this matrix, the term f is camera focal length, ρ_u and ρ_v are dimensions of each pixel (pixel width and height), and c_x and c_y are transnational distances between origin of pixel and image coordinate systems. Camera extrinsic matrix, which depends on the position and orientation of the camera, converts points from FE model coordinate system to camera coordinate system. In this matrix, the term $\mathbf{R}_{3\times3}$ is the rotational matrix and $\mathbf{t}_{3\times1}$ is the transnational vector.

$$\begin{pmatrix} v \\ u \\ w \end{pmatrix} = \begin{pmatrix} \frac{f}{\rho_u} & 0 & c_x & 0 \\ 0 & \frac{f}{\rho_v} & c_y & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} \mathbf{R}_{3\times3} & \mathbf{t}_{3\times1} \\ \mathbf{0}_{1\times3} & \mathbf{1}_{1\times1} \end{pmatrix} \begin{pmatrix} X_{FE} \\ Y_{FE} \\ Z_{FE} \\ 1 \end{pmatrix} \tag{4.1}$$

While the camera intrinsic matrix is only dependent on camera properties and not changing from one frame to another, the camera extrinsic matrix might vary between successive frames. This variation is due to the movement of camera and change in the position and orientation of camera caused by different conditions, e.g., wind. Hence, the camera extrinsic matrix needs to be calculated in each frame. For this purpose, few number of fixed landmarks are selected and tracked from one frame to another frame. Fixed landmarks are referred to the non-moving objects that are in the view of the given camera (e.g., light poles or traffic lights). Due to the movement of camera, the pixel coordinates of fixed landmarks may change in subsequent video frames. Using the tracked coordinates in pixel coordinate system and the known coordinates in FE coordinate system, the extrinsic matrix can be developed. Tracking the fixed landmarks is carried out using optical flow. Optical flow is defined as the pattern of apparent motion of image objects between two consecutive frames. It is a 2D vector field where each vector is a displacement vector showing the movement of points from first frame to second. Optical flow works on the following assumptions:

- The pixel intensities of an object do not change between consecutive frames.
- Neighbouring pixels have similar motion.

In order to track fixed landmarks, the fixed landmarks are selected manually by user on the first frame of each video recording. Then, the salient features such as corners or edges are extracted from the fixed landmarks. Utilizing the extracted features along with optical flow algorithms, the movement of fixed landmarks are monitored and tracked across consecutive frames. After finding the pixel coordinates of fixed landmarks, the camera extrinsic matrix is developed utilizing the known coordinates in FE model, as well as rotation and translation operations.

In this study, the iterative Lucas-Kanade method is used to estimate the optical flow equations utilizing the least squares [52, 53]. For this purpose, two pyramid levels are used and size of the search window at each pyramid level is set to 5×5. The iterative process is terminated when one of the following criteria are met:

- Search window moves by less that 0.01.
- 50 number of iterations are done at each pyramid layer.

This approach is implied using the function *calcOpticalFlowPyrLKimplements* from the *OpenCV* library of *Python* programmin language.

It is noteworthy that transformation from pixel to FE coordinate system can be streamlined using projective mapping, also known as the perspective or homogeneous transformation [35]. For this purpose, the four fixed landmarks are selected on a single plane and the combination of intrinsic and extrinsic matrices are calculated using an affine mapping. The details of this process are explained in Section 4.1.4.

4.1.2 Phase 2: Vehicle Tracking

In this study, the YOLOv8s-seg model is utilized on each recorded frame for detection and classification of vehicles. This model is pre-trained on COCO (Common Objects in Context) dataset [49], which contains over 80 object classes, including different types of vehicles such as cars, trucks, buses, and motorcycles. This pre-trained model is used for the purpose of vehicle detection and classification of this study.

The YOLOv8s-seg model is utilized for vehicle detectinon and classification with the following criteria:

- The confidence threshold for non-maximum suppression (NMS) is set to 0.5. So, detections with a confidence score below 0.5 are filtered out. The primary goal of NMS is to reduce multiple detections of the same object instance to a single detection.

- The intersection over union (IoU) threshold is set to 0.7. So, if the IoU between two boxes is greater than 0.7, the one with lower confidence is suppressed. Calculation of IoU is further explained in the next section.

- Multi-labeling is disabled. So, NMS assigns each detection box to at most one class label based on the highest confidence.

- The optional list that filters detection by specific class indices is set in a way to filter out any detection other than vehicles.

Once vehicles are identified in the video frames, they are subsequently tracked using Deep SORT model. Deep SORT integrates the detection results from YOLO with the tracking algorithm. Deep SORT extracts a deep appearance feature vector for each detected object using a convolutional neural network (CNN-based) feature extractor. Then, associate the detected objects across frames using a matching algorithm (such as the Hungarian algorithm) that takes into account both the location and appearance of the objects. Finally, tracks that have not been associated with any objects for a certain number of frames or that have low confidence scores are removed. The Deep SORT algorithm is utilized with the following criteria:

- Nearest neighbor distance metric is specified as a cosine distance. The cosine distance measures the angle between feature vectors and is often used to measure the similarity between feature vectors of detected vehicles.

- The maximum cosine distance threshold is set to 2, which indicates that objects

whose feature vectors have a cosine distance greater than 2 will be considered dissimilar.

- No limit is implied on the maximum number of nearest neighbors that are considered for association in the nearest neighbor matching process.
- Minimum number of detection before confirming a track is set to 3. It means that newly allocated tracks are activated after 3 frames .
- Maximum age of tracks is set to 10. This criterion means that any track not associated with a detection for 10 consecutive frames will be deleted.

4.1.3 Phase 3: Tire tracking

Similar to the vehicle detection and classification, tire detection and segmentation is performed using YOLOv8m-seg model. However, this model is not pre-trained on tire objects and needs to be trained for the purpose of this study. The training process and evaluation of the trained model are discussed in the following subsections.

4.1.3.1 Training of model

The training dataset for instance segmentation of tires is prepared using OpenImagesV7 [33]. OpenImages is a dataset provided by Google, designed for machine learning and computer vision tasks. It contains a large collection of annotated images (over 1.9 million), making it valuable for training and evaluating algorithms in object detection, image classification, and segmentation. To prepare the training dataset, 35000 raw images and their masks for instance segmentation of tires are downloaded from OpenImages. These images are classified as wheel dataset in the OpenImages dataset. The dataset is divided to three subsets as follow:

- Training dataset, which contains 80% of the dataset.
- Validation dataset, which contains 20% of the dataset.

The dataset is organized based on the YOLO format including separate folders for training and validation data with separate sub-folders for images and masks. The model is trained using the deep learning framework of *PyTorch 2.1.1*, complemented by *CUDA Toolkit 11.2* and *cuDNN 8.1* for optimized GPU acceleration. The following parameters are set for the training process:

- Images are resized to 640×640 pixels.
- Batch size is set to 16, which means that 16 images are processed simultaneously during training process.
- Epochs are set to 100, which means that the training process iterates 100 times over the entire dataset.
- Initial weight are set to the weights of pre-trained YOLOv8s-seg model.

Moreover, the properties of the high-performance computing system is as follows:

- Intel Core i7-12700K 12-Core Processor, which provides significant computational power for handling large datasets and complex deep learning tasks
- NVIDIA GeForce GTX 1080 Ti GPU, known for its high performance in parallel processing, essential for deep learning tasks that require substantial computational resources for training models efficiently.

4.1.3.2 Evaluation of trained model

After training the YOLO model on tire data set, the model is evaluated by being executed on validation dataset. As an example, Figure 4.3 shows an example of performance of the trained model on the validation dataset.

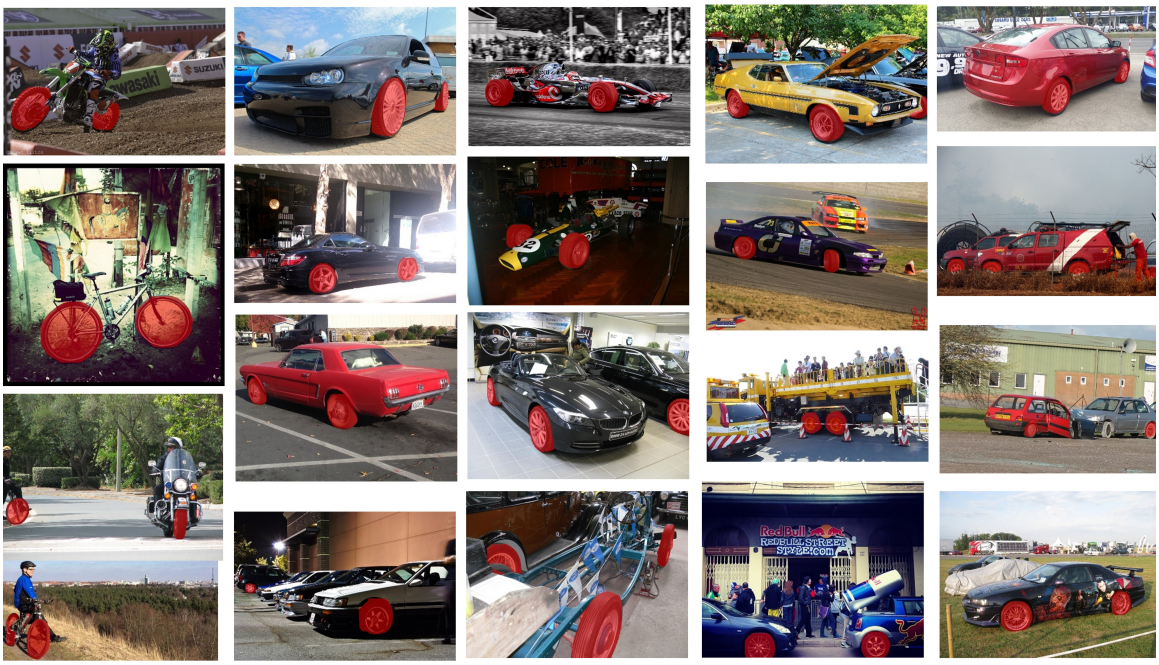


Figure 4.3: Examples of tire segmentation on validation dataset.

In this study, to evaluate the performance of the trained model, the metrics of *precision*, *recall*, and *mean average precision (mAP)* are utilized. In order to calculate these metrics, it is necessary to calculate the following parameters:

- True positive (TP): TP indicates the total number of pixels correctly identified within the segmented instance where the algorithm accurately detects a tire.
- False positive (FP): FP indicates the total number of pixels incorrectly identified as part of a tire segment when no tire is present there.
- False negative (FN): FN indicates the total number of pixels that the segmentation algorithm has incorrectly not classified as not part of the tire. In other words, FN indicates instances where the algorithm fails to detect an existing tire in the image.

The above parameters can be calculated using IoU. The term IoU measures the overlap between the ground truth segments and the predicted segments, as outlined in Equation 4.2. In this equation, the term A refers to the area of ground truth

segment and B refers to the area of predicted segment. When the IoU value is greater than a predefined threshold (here 0.7, as explained in Section 4.1.2), it qualifies the predicted segment as a TP; otherwise, it qualifies the predicted segment as a FP. It is noteworthy that the TP, FP and FN of Equations 4.3 and 4.4 are calculated using IoU, which facilitates the calculation of precision and recall.

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{4.2}$$

Using the calculated IoU values, TP, FP and FN are calculated as follow:

- TP counts the predictions where IoU is greater than the predefined threshold for at least one ground truth mask.
- FP counts the predictions where IoU is smaller than the predefined threshold for all ground truth masks.
- FN counts the ground truths where there is no prediction with IoU greater than the predefined threshold.

After calculating TP, FP and FN, the precision and recall metrics can be calculated. Precision indicates how good the model is at identifying true positives (avoiding mistakes), and is calculates as bellow:

$$Precision(P) = \frac{TP}{TP + FP} \tag{4.3}$$

Recall indicates that how well the model finds all the relevant things it should (not missing anything important). The following equation shows how this metric is calculated.

$$Recall(R) = \frac{TP}{TP + FN} \tag{4.4}$$

Using the precision and recall metrics, average precision (AP) is calculated. AP measures how well the model predicts the objects and their locations in an image and summarizes the precision-recall tradeoff of a model over different thresholds.

For this purpose, the precision-recall curve is developed for each class by varying the confidence threshold from 0 to 1, and compute the precision and recall for each threshold. AP is calculated as the area under the precision-recall curve, as defined in Equation 4.5.

$$\mathrm{AP}_c = \sum_{k=0}^{n-1} [(\mathrm{Recall}_c(k) - \mathrm{Recall}_c(k+1)) \times \mathrm{Precision}_c(k)] \tag{4.5}$$

In the above equation, the subscript c refers to the object class (here, tire) and the term k counts the confidence scores (n in total) used to calculate the precision-recall curves. The confidence score is a measure of the model's certainty that a predicted object is present in the image and belongs to the specific class (here, tire). The confidence score is typically calculated as the product of the two following probabilities:

- The probability that the predicted mask contains an object ($P(\mathrm{Object})$).
- The probability that the mask corresponds to a specific class ($P(\mathrm{Class} \mid \mathrm{Object})$).

The mean average precision (mAP) is calculated by averaging the AP values across all classes, as illustrated in Equation 4.6. This metric provides an overall assessment of the model's performance across different classes. In this equation, the term C is the total number of classes.

$$\mathrm{mAP} = \frac{1}{C} \sum_{c=1}^{C} [AP_c] \tag{4.6}$$

Segmentation models are usually evaluated using mAP50 and mAP50-95 metrics tailored to assess their performance in delineating object boundaries accurately across images. These two metrics are defined as follow:

- mAP50: Represents the mAP calculated using confidence scores ranging from 0 to 0.50. This metric evaluates how well the segmentation model delineates object boundaries with lower confidence, indicating its performance at less certain thresholds.

- mAP50-95: Represents the mAP computed with confidence scores from 0.50 to 0.95. This metric provides a comprehensive evaluation across instances where the model accurately segments objects with higher confidence. This metric is particularly useful in scenarios where precise localization of objects is critical.

Table 4.1 presents the performance evaluation of the trained model on the validation dataset, using precision, recall, mAP50, and mAP50-95 metrics. The 'Value' columns in this table show the absolute values of these metrics, while the 'Change' columns indicate the improvement of the trained model compared to its initial epoch using pre-trained weights. Throughout the training process, all evaluation metrics show improvement. Notably, the mAP50-95 metric demonstrates the greatest enhancement, highlighting the model's robustness in accurately localizing objects. Conversely, the recall metric exhibits the smallest improvement, suggesting that the model may occasionally fail to detect positive instances present in the dataset.

Table 4.1: Evaluation of trained YOLO model.

Model	**Precision^{val}**		**Recall^{val}**		**mAP^{val} 50**		**mAP^{val} 50-95**	
	Value	Change	Value	Change	Value	Change	Value	Change
YOLOv8s-seg	65.73%	+37.43%	61.3%	+29.1%	58.2%	+37.2%	88.2%	+40.3%

4.1.4 Phase 4: Transformation to FE coordinate system

After the subsequent application of YOLO and Deep SORT on each bounding box (detected vehicles), the segmentation mask of each tire of vehicle is predicted. At this stage, the pixel coordinates of the segmented area for tires are extracted. These coordinates represent a closed curve with a contact point with the surface of bridge. The pixel coordinate of the contact point is extracted and referred to as $\begin{pmatrix} u \\ v \\ w \end{pmatrix}$. Using the tracked location of fixed landmarks (explained in Section 4.1.1) and projective mapping, the coordinate of contact point is derived in the FE coordinate system. As already mentioned in Section 4.1.1, to streamline the transformation of coordinates

from pixel to FE systems, homogeneous transformation is used.

By selecting the fixed landmark locations on a single plane, combination of intrinsic and extrinsic camera transformations of Equation 4.1 can be calculated using projective mapping, also known as the perspective or homogeneous transformation [35]. Homogeneous transformations are used extensively for perspective camera transformations [27]. In homogeneous notation, 2D points are represented by 3D vectors, and for affine and projective mappings, points in source space (here pixel coordinate system) are denoted by $\begin{pmatrix} u \\ v \\ w \end{pmatrix}$ and points in the destination space (here FE coordinate system) are denoted by $\begin{pmatrix} X_{FE} \\ Y_{FE} \\ Z_{FE} \end{pmatrix}$. The general form of a projective mapping in the homogeneous matrix notation is as follows:

$$\begin{pmatrix} X'_{FE} \\ Y'_{FE} \\ Z_{FE} \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} u' \\ v' \\ w \end{pmatrix} \tag{4.7}$$

where $(X_{FE}, Y_{FE})^T = \left(\frac{X'_{FE}}{Z_{FE}}, \frac{Y'_{FE}}{Z_{FE}}\right)^T$ for $Z_{FE} \neq 0$, and $(u, v)^T = \left(\frac{u'}{w}, \frac{v'}{w}\right)^T$ for $w \neq 0$. Although there are 9 coefficients in the matrix above, these mappings are homogeneous, so any nonzero scalar multiple of these matrices gives an equivalent mapping. Hence there are only 8 degrees of freedom in a 2D projective mapping. Without loss of generality, it can be assumed that $i = 1$. This projective mapping can be determined from the source and destination coordinates of the four corners of a quadrilateral. Let the correspondence map $\begin{pmatrix} u_k \\ v_k \end{pmatrix}$ to $\begin{pmatrix} X_{FE_k} \\ Y_{FE_k} \end{pmatrix}$ for vertices numbered $k = 0; 1; 2; 3$. To compute the forward mapping matrix M_{sd}, assuming that $i = 1$, eight equations and eight unknowns a-h are set, which can be rewritten as an 8×8 system

$$\begin{bmatrix} u_0 & v_0 & 1 & 0 & 0 & 0 & u_0x_0 & v_0x_0 \\ u_1 & v_1 & 1 & 0 & 0 & 0 & u_1x_1 & v_1x_1 \\ u_2 & v_2 & 1 & 0 & 0 & 0 & u_2x_2 & v_2x_2 \\ u_3 & v_3 & 1 & 0 & 0 & 0 & u_3x_3 & v_3x_3 \\ 0 & 0 & 0 & u_0 & v_0 & 1 & u_0y_0 & v_0y_0 \\ 0 & 0 & 0 & u_1 & v_1 & 1 & u_1y_1 & v_1y_1 \\ 0 & 0 & 0 & u_2 & v_2 & 1 & u_2y_2 & v_2y_2 \\ 0 & 0 & 0 & u_3 & v_3 & 1 & u_3y_3 & v_3y_3 \end{bmatrix} \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} X_{\mathrm{FE}_0} \\ X_{\mathrm{FE}_1} \\ X_{\mathrm{FE}_2} \\ X_{\mathrm{FE}_3} \\ Y_{\mathrm{FE}_0} \\ Y_{\mathrm{FE}_1} \\ Y_{\mathrm{FE}_2} \\ Y_{\mathrm{FE}_3} \end{bmatrix} \tag{4.8}$$

This linear system can be solved using Gaussian elimination or other methods for the coefficients *a*-*h*, and can be performed by the *getPerspectiveTransform* function available in the *OpenCV* library of Python programming language.

4.2 VR Platform

After development of the pipeline for the traffic tracking technology component and training its required deep learning models, the pipeline is tested on a simulated testbed environment using VR and the Unity game engine. The VR is required to simulate a roadway that is instrumented with traffic cameras on the shoulder of road. Moreover, defined by the user, a traffic of vehicles with various speeds and directions can traverse on the road while the video of traffic are recorded by the cameras. The developed VR needs to be capable of recording synchronized footage of traffic while fixed landmarks are in the view of cameras. For each traffic scenario, the synchronized recordings of cameras are saved as *mp4* file. At a same time, the location of the outer edge of tires (ground truth) in contact with the surface of road are recorded and reported as *CSV* files. Sections 4.2.1 to 4.2.5 explain the various assets and features used in development of the VR, and Section 4.2.6 discuses `C#` code implementation for animating traffic and explains the process of saving the ground truth data for tire locations as well as saving the camera recordings.

4.2.1 Bridge Road

The bridge road is simulated based on the properties of East Yutan Bridge [68], NE. EY is a four-span bridge with a total length of 122 meters in the east-west direction and a total width of 9.8 meters. This width is divided into two lanes: one lane designated for eastbound traffic, allowing vehicles to traverse in the eastward direction, and the other lane for westbound traffic, allowing vehicles to traverse in the westward direction. Figure 4.4 shows the top view of the bridge along with the defined world/FE coordinate system. As can be seen in this figure, the origin of the world coordinate system is selected at the mid-width of the roadway on the west

entrance, and the X and Y axis are selected in the longitudinal and traverse directions of the roadway. Modeling of the roadway, its line dividers and curbs are discussed in this section. The roadway is also marked with fixed landmarks that will be discussed later.

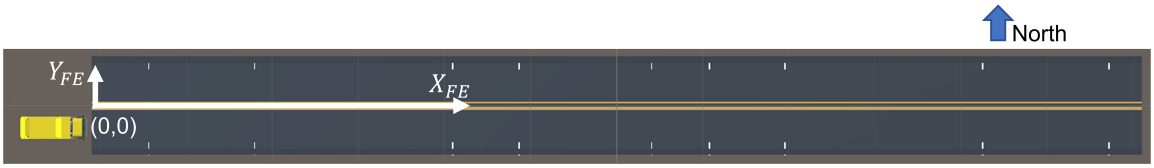


Figure 4.4: Top view of the simulated roadway and the global X and Y axis.

The road is modeled using a 3D *Plane* object with the given dimensions. The graphics of the road, including its shape and appearance, are set using the *Mesh Renderer*, and the physics of the road is set through the *Mesh Collider*. To assign the graphics, the road object is assigned with a *Standard Shader* which is a built-in shader with a comprehensive set of features to render real-world hard surface objects. The *Standard Shader* also incorporates an advanced lighting model called physically based shading (PBS) that simulates the interactions between materials and light in a way that mimics reality by following principles of physics. The color of road is selected in *Opaque* rendering mode as a shade of gray with values of $\begin{pmatrix}56 & 56 & 56\end{pmatrix}$ for RGB channels. The reflectivity and light response of the surface are modified by setting the *Metallic* and *Smoothness* levels. The prior is set to 0.5 presenting medium reflection of the environment on the bridge road, and the latter is set to 0.5 representing a medium micro-surface detail. The micro-surface detail is a concept used in the lighting calculations and the amount of light that is diffused as it bounces off the object. Taken to its extreme, a perfectly smooth surface reflects light like a mirror (smoothness level equal to 1), and less smooth surfaces reflect light over a wider range of angles (as the light hits the bumps in the micro-surface), and therefore the reflections are spread across the surface in a more diffuse way. Moreover, the *Lighting* component is set to cast a shadow when a shadow-casting light shines on it. The *Mesh Collider* is set using *plane mesh* which refers to a flat, 2D mesh shape

that is typically used for objects that represent flat surfaces. Moreover, the *Layer Override* feature which specifies the collision between different objects is set to zero reflecting no collision between road and other objects.

The road divider lines are modeled using 3D *Plane* objects with the width of 25 cm. They are positioned at the middle and along the length of road with the clear distance of 30 cm. The *Standard Shader* with *Opaque* color mode and a shade of yellow with RGB values of $\begin{pmatrix}211 & 156 & 69\end{pmatrix}$ is selected. The *Metallic* and *Smoothness* levels as well as *Collider Mesh* are defined similar to those of the road object.

Curbs are modeled at the two sides of the road using *Cube* objects with the dimension of $0.7\,\mathrm{m} \times 0.7\,\mathrm{m} \times 122\,\mathrm{m}$. The color is selected as dark gray with RGB values of $\begin{pmatrix}51 & 51 & 51\end{pmatrix}$. The *Box Colliders* are defined for the curbs, while the rest of the settings are similar to those of the road object.

4.2.2 Fixed Landmarks

In this preliminary study, fixed landmarks are modeled as white markings at known locations on the two sides of the roadway (as symmetric pairs with respect to the X axis). Figure 4.5 shows the layout of landmarks and a close look to four of the markings. These markings have a dimension of $75\mathrm{cm} \times 7.5\mathrm{cm}$ and are modeled using similar properties as road divider lines, explained in Section 4.2.1.

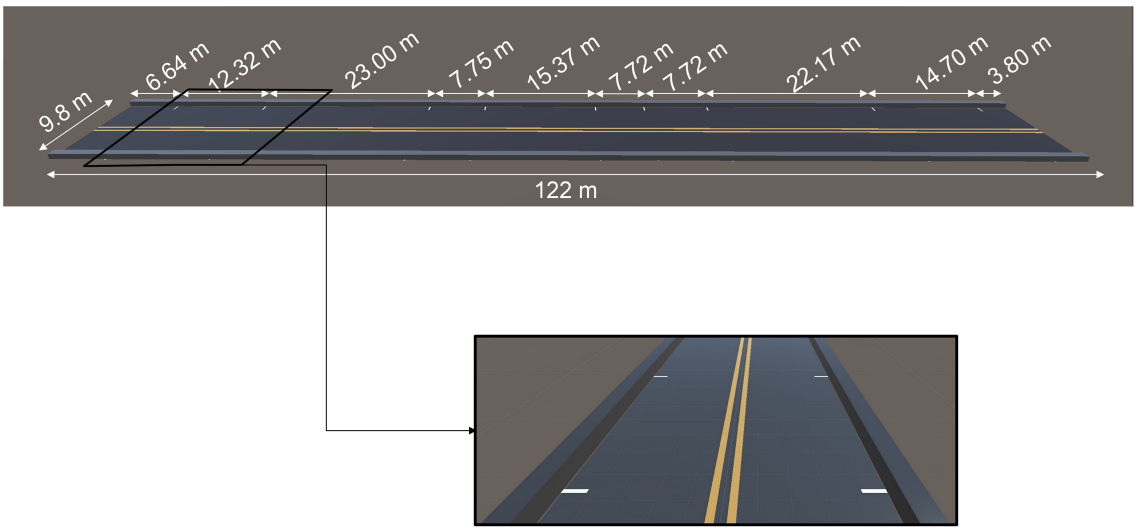


Figure 4.5: Layout of fixed landmarks and a close view over four of them.

4.2.3 Cameras

The cameras are modeled using the *Camera Object*. This object is a device through which the player views the simulated environment. The *Clear Flag* of cameras are set to *Skybox* with *Background* of blue to create the illusion of sky around the scene. *Perspective Projection* is assigned to the camera objects simulating how the human eye sees the world: Objects appear smaller as they get further away from the camera, creating a sense of depth and realism. The cameras' *Field of View* are set to 50° along the vertical axis which is a common value to provide a natural viewing experience. The cameras are located five meters above the surface of road at the two sides of the road in locations shown in Figure 4.6. Moreover, the cameras are installed with ∓30° rotation in the global XY plane and 30° in the global YZ plane as can be seen in Figure 4.6. The I.Ds. of cameras as well as few examples of their view when traffic is on the bridge are shown later in Figure 4.2.5.

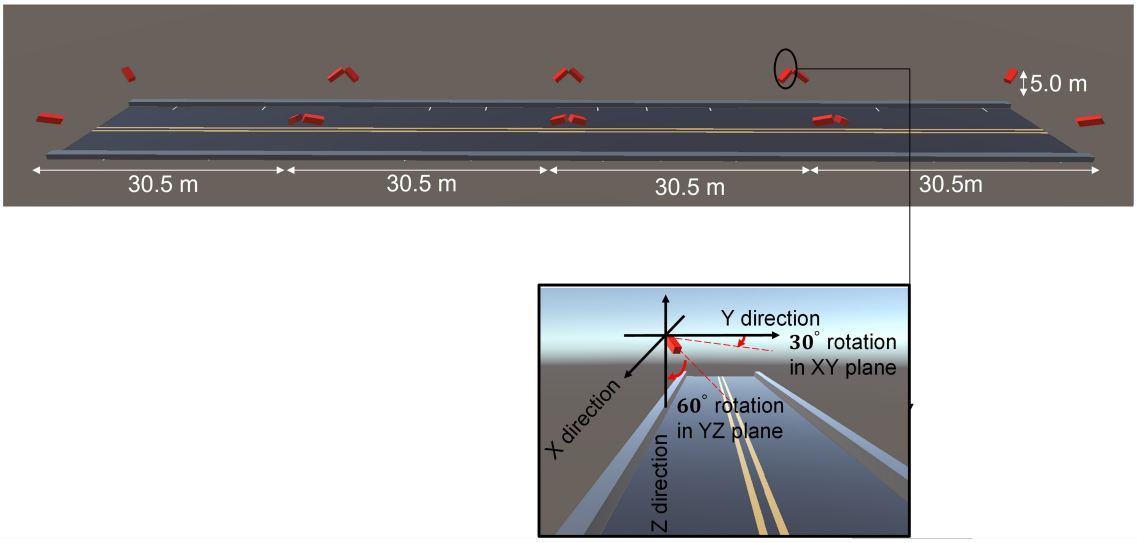


Figure 4.6: Layout of cameras and their angles.

4.2.4 Light

In this study, light is simulated using *Directional Light* which is useful for creating effects such as sunlight in the scene. Behaving in many ways like the sun, directional lights can be thought of as distant light sources which exist infinitely far away. The direction of light is set to have a 50° angle withing the global XY plane and an angle of 50° within the global YZ plane. The color is set as a warm light using RGB of $(255 \quad 244 \quad 214)$. The light component is defined using the *Mixed Mode* which combines elements of both real-time and baked lighting. Real-time lighting updates dynamically in real-time and is useful for objects that move or change frequently but can be expensive in terms of performance. On the other hand, baked lighting is pre-computed and stored in lightmaps and is less demanding at runtime compared to the real-time lighting. Mixed mode strikes a balance between real-time and baked lighting, optimizing performance while still allowing for dynamic lighting changes. Figure 4.7 represents the modeled directional light.

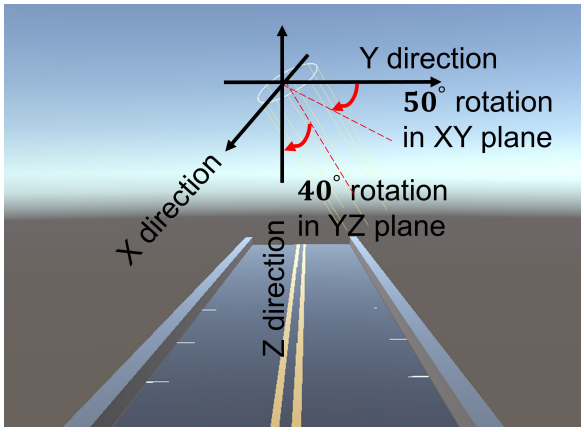


Figure 4.7: Directional light.

4.2.5 Vehicle

For the purpose of this study, vehicles are modeled as tri-axial dump truck (see Figure 4.8(a)) which is 7.63 m long, 2.92 m wide and 2.76 m high. The truck is a *Parent Rigid Body* object asset imported from [94], including child objects of body, container, cover, spare wheels, piston and tires. The child objects are shown in Figure 4.8. Features regarding the graphics and physics of the objects are defined using *Cube* objects with *Mesh Renderer* and combination of *Box* and *Triangle Colliders* - For more details, see [94]. The side view of the truck with its assigned colliders is shown in Figure 4.9. Moreover, Figure 4.10 shows example of camera views when the truck is on the bridge. It is noteworthy that the physics of animation of traffic, including the movement of truck rigid body as well as the rotation of tires are defined in the `C#` script which is further explained in Section 4.2.6.

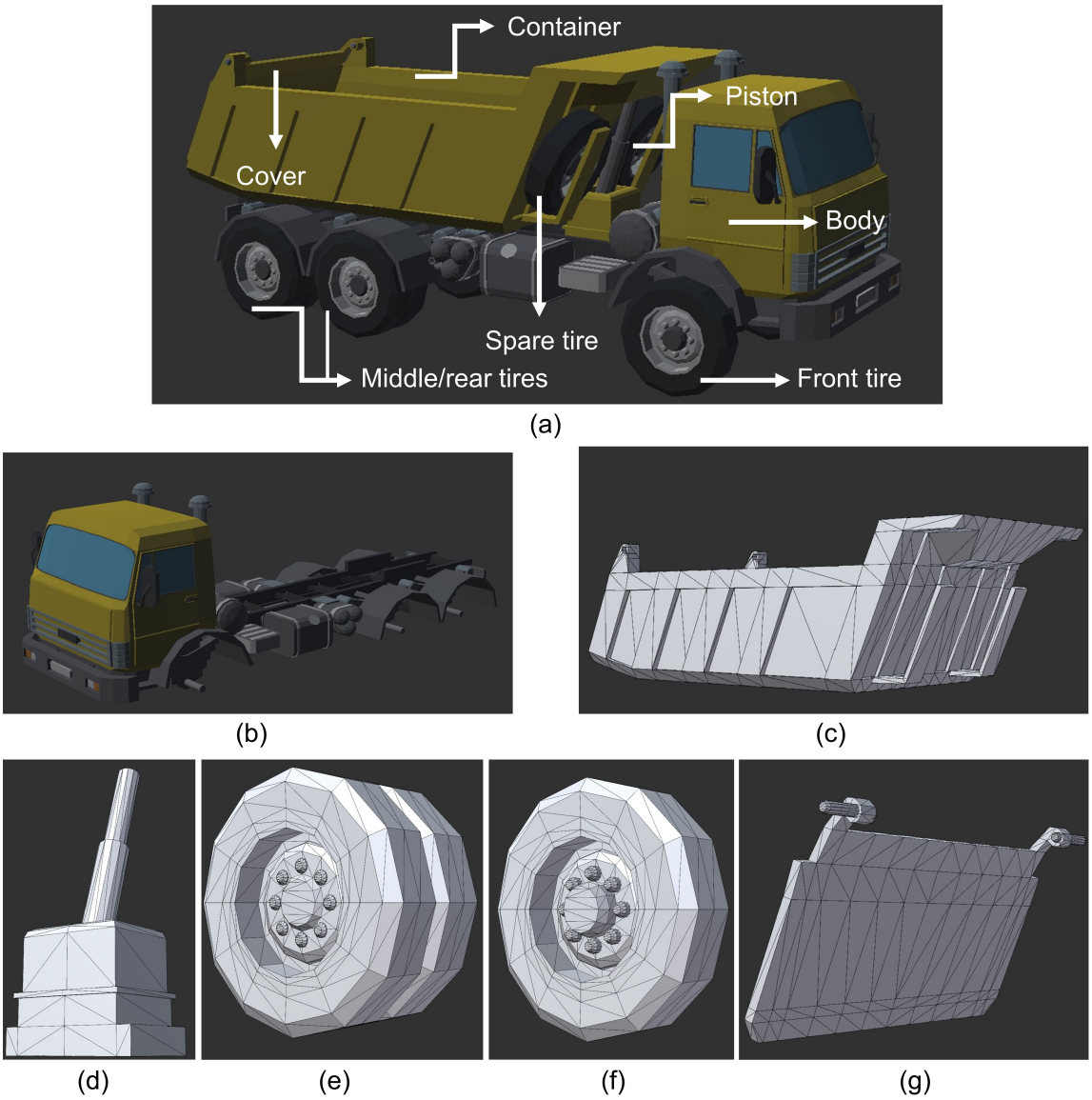


Figure 4.8: Truck asset and its child objects: (a) Annotated parent object, (b) body, (c) container, (d) piston, (e) middle/rear tire, (f) front/spare tire, (g) cover.

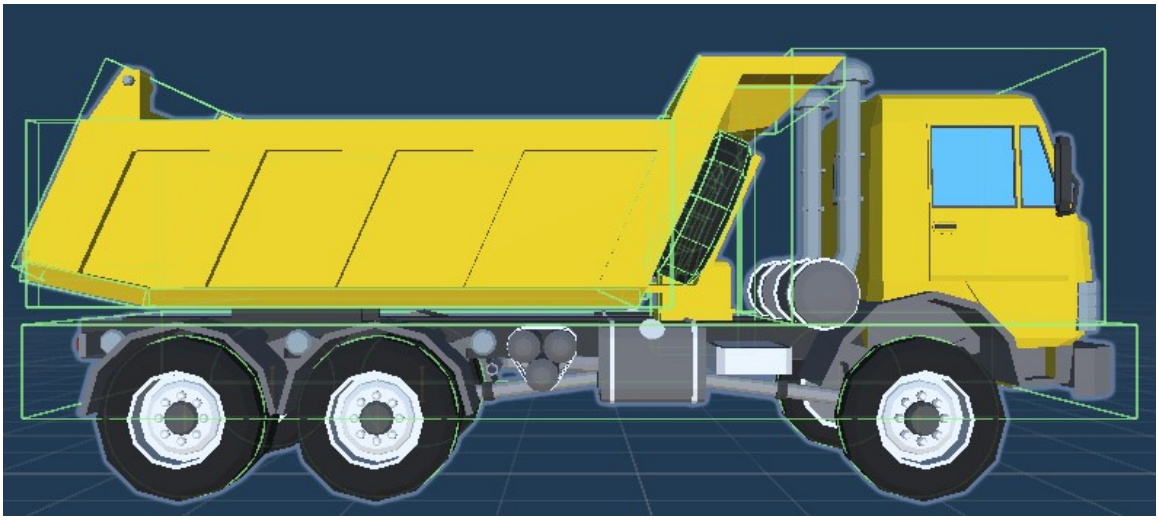


Figure 4.9: Side view of truck rigid body and the colliders.

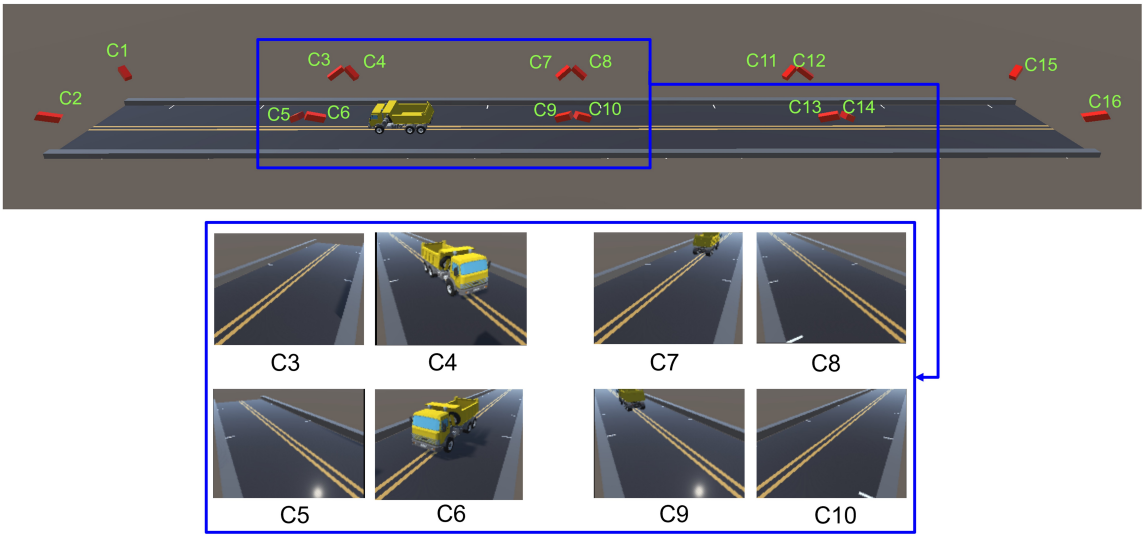


Figure 4.10: Camera I.Ds. and an example of their recording when truck is on the bridge.

4.2.6 Traffic Animation and Saving Outputs

The `C#` script is attached to the truck game object and is responsible for animating the traffic on the bridge, recording and saving the footage of camera, as well as saving the ground truth location of tires. The code includes five main functions - Start, RepeatingFunction, Update, FixedUpdate and CameraSwitch - listed below:

- **Start**: This function initializes the script when the traffic simulation starts and retrieves the *Parent Rigid Body Component* (truck) using the *GetComponent* method. Moreover, this function retrieves the dimension of the tires by calling the *GetComponent* method on a reference to the tire *GameObject* to obtain its *Renderer* component and calculates the required dimensions, e.g., half of the tires' width using the *bounds.size.y* property of the *Renderer* component.

  The Start function also sets the active camera. Active camera is the camera which its recording is being shown to the user using a user-defined *CameraSwitch* function. Then, a CSV file that stores the ground truth location of tires is initialized. This file includes 12 columns reporting the X and Y coordinates of front, middle and rear tires on North and South bounds. Moreover, the Start function schedules to call the *RepeatingFunction* function every $\frac{1}{30}$ second (30 frame per second).

- **RepeatingFunction**: This function reads the positions of the tire centers using the *transform.position.x* and *transform.position.y* properties. Using the calculated tire width from the Start function, it then calculates the positions of the outer edges of the tires. Then, the *RepeatingFunction* function writes this information to the CSV file.

  Moreover, by comparing the y coordinate of tires with the length of bridge, this function checks if the truck exit the bridge. If so, the game would quit.

- **Update**: This function checks for user input using *Input.GetKey*, and switch camera to the target camera using the user-defined *CameraSwitch* function. The user input serves as the camera identifier, specifying which camera's footage to display to the user.

- **FixedUpdate**: This function moves the truck forward and rotates its tires. The translation of truck is carried out using *transform.Translate(Vector2.forward * speed)*. Transform is a property of the *GameObject* that provides access to

its *Transform* component. The *Transform* component handles the position, rotation, and scale of the *GameObject*. Here, *Translate* is a method that moves the truck in space based on the specified vextor. *Vector1.forward* represents a direction along the positive X.

In addition, using the tire dimensions and the traveled distance in each frame, the rotation of tires are calculated and applied to their components using *transform.Rotate* method.

- **CameraSwitch**: This function switches the active camera based on the user-input camera I.D. It iterates through the list of cameras and enables the selected camera by setting its *enabled* property to *True*, while disabling the other cameras by setting their *enabled* properties to *False*.

Chapter 5

Experiments and Results

5.1 Application of proposed traffic tracking method using VR

Various scenarios of traffic are modeled using Unity. The scenarios are conducted with one or two trucks traveling in one or two directions. The traffic scenarios include:

- Travel of a single truck in the middle of the bridge, straddling the line between the two lanes.
- Travel of two trucks in opposite directions and different lanes.
- Travel of a single truck with changing lane at the middle of bridge length.

In the developed experiments, trucks travels the bridge with a fixed speed between 8 and 800 kilometer per hour (kph). The description of scenarios are explained in Table 5.1 using the below legends:

- In the Direction columns:
  - "X" represents that the truck goes towards direction X;
  - "XY" represents that trucks go in opposite directions at a same time (truck 1 towards direction X and truck towards direction Y);
- In the Speed columns:
  - "X" represent that the truck goes with speed X kph;

  - "X-Y" represent that truck 1 goes with speed X kph and truck 2 goes with speed Y kph;

- In the Lane columns:
  - "C" stands for center line of bridge, "N" stands for North lane and "S" stands for South lane;
  - "X-Y" represent that truck1 traverses in lane "X" and truck 2 traverses in lane "Y";
  - "X—Y" represent that the truck strats the travel in lane "X" and and then changes to lane "Y" in the middle of bridge;

Table 5.1: Test logs

No.	**Direction**	**Speed (kph)**	**Lane**
1	W	8	C
2	W	16	C
3	W	32	C
4	W	48	C
5	W	64	C
6	W	80	C
7	E	80	C
8	EW	8-8	S-N
9	EW	16-16	S-N
10	EW	32-32	S-N
11	EW	64-64	S-N
12	W	16	N—S
13	E	32	N—S

Figures 5.1, 5.2 and 5.3 show synchronized snapshots of traffic cameras from test scenarios No.6, 9 and 13, respectively. These figures show the footage of cameras at the $\frac{1}{3}$ of the total time of scenarios in tests No.6 and No.9, and the $\frac{1}{2}$ of the total time of test scenario No.13. In test No.6, shown in Figure 5.1, a single truck traverses on the bridge on its center line from East to West direction with a speed of 80 kph. In test No.9, shown in Figure 5.2, two trucks traverse on the bridge with speeds of 16

kph and in different directions. The truck on the North lane traverses towards the West direction, and the truck on the South lane traverses towards the East direction. In test No.13, shown in Figure 5.3, a single truck traverses on the bridge from West to East direction starting on the North lane and changes the lane to the South one in the middle of the bridge.

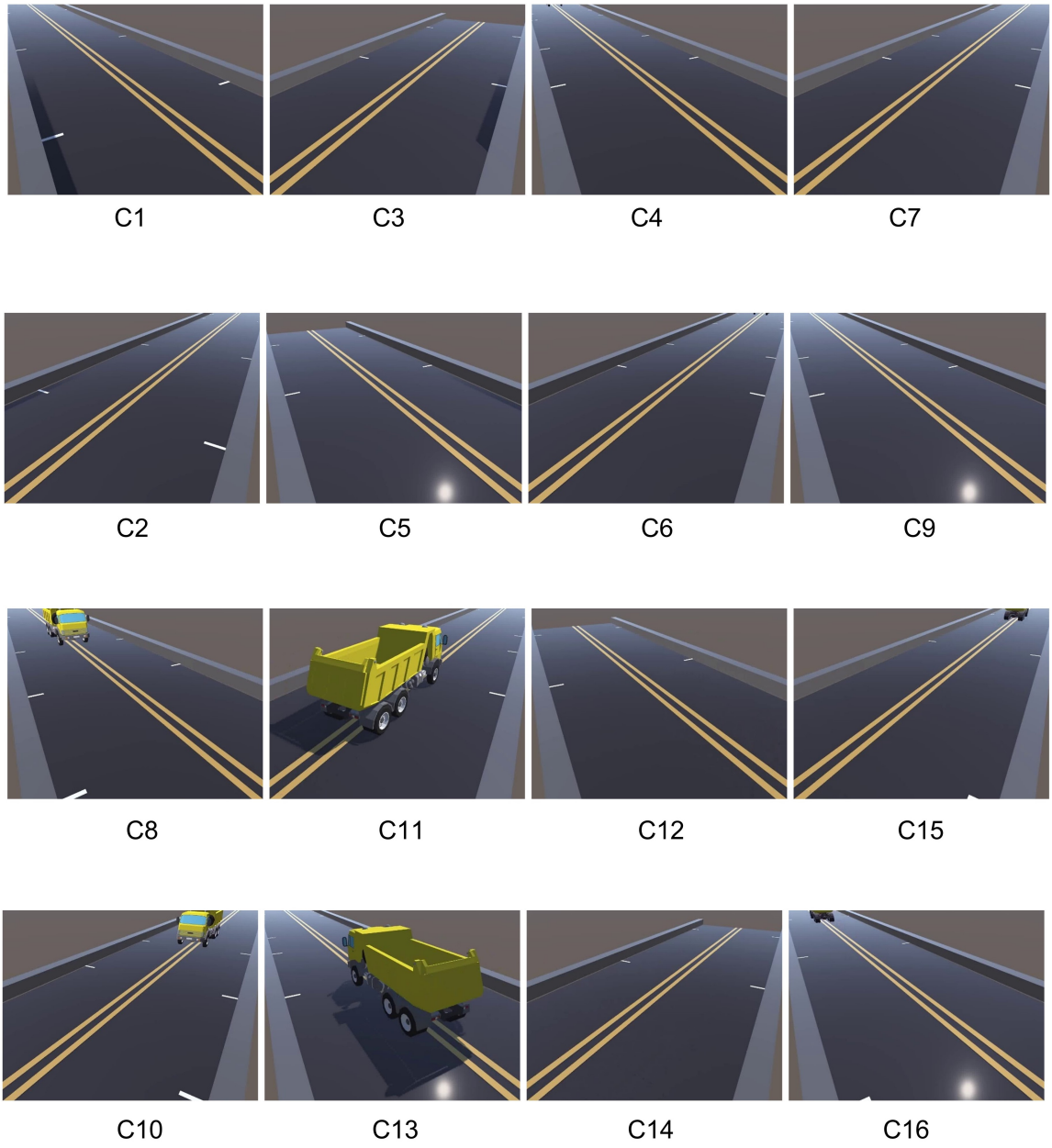


Figure 5.1: Synchronized view of traffic cameras for test No.6 in which a single truck traverses on the bridge on its center line from East to West direction with a speed of 80 kph.

The proposed traffic tracking method is applied on the test scenarios of Table 5.1. For the sake of presentation, the results are shown in details for the test scenario No.7 and cameras 4 and 6. These two cameras are installed at a same longitudinal coordinate of bridge on the North and South shoulders of bridge (see Figure 4.10).

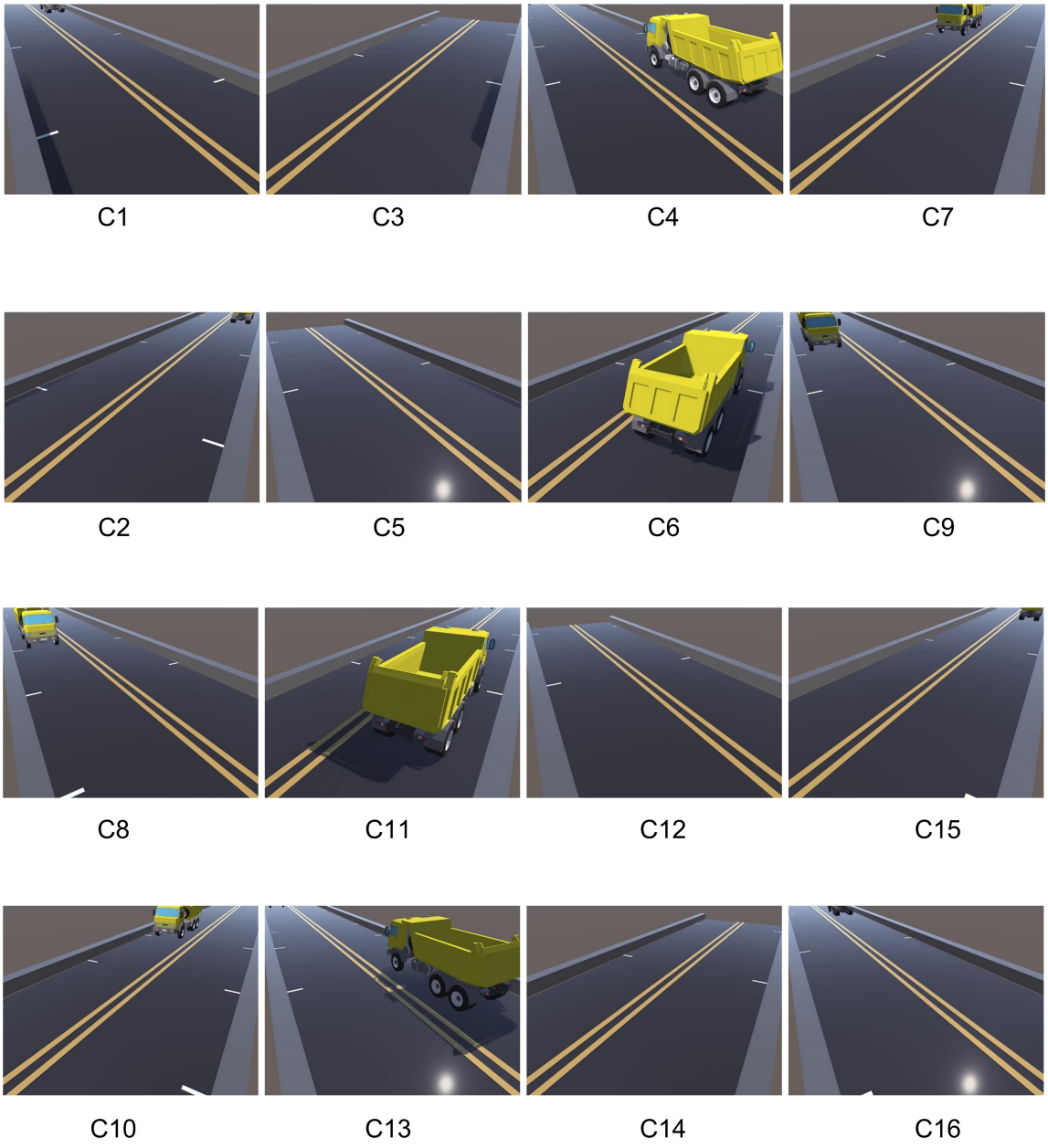


Figure 5.2: Synchronized view of traffic cameras for test No.9 in which two trucks traverse on the bridge with speeds of 16 kph and in different directions. The truck on the North lane traverses towards the West direction, and the truck on the South lane traverses towards the East direction.

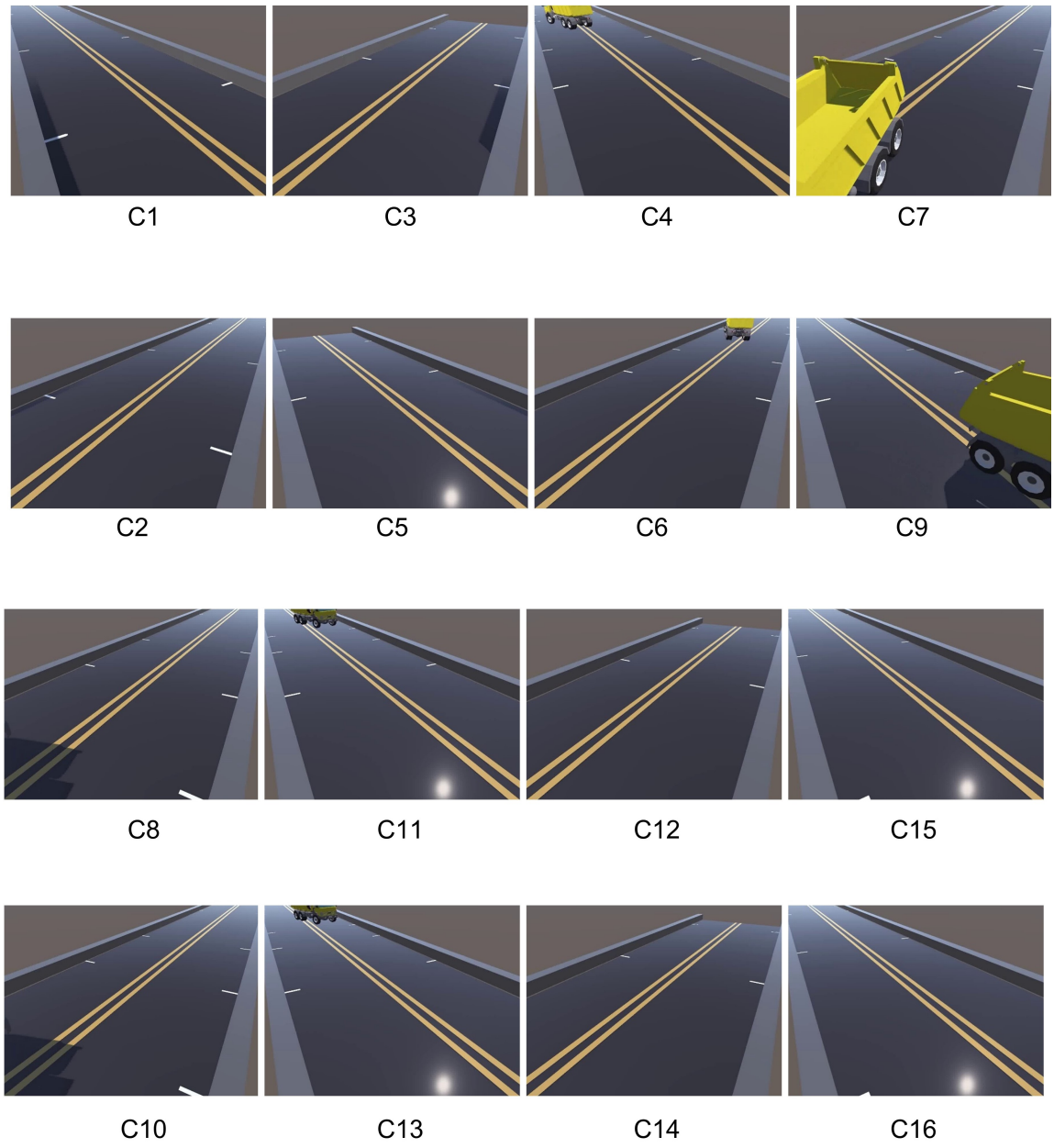
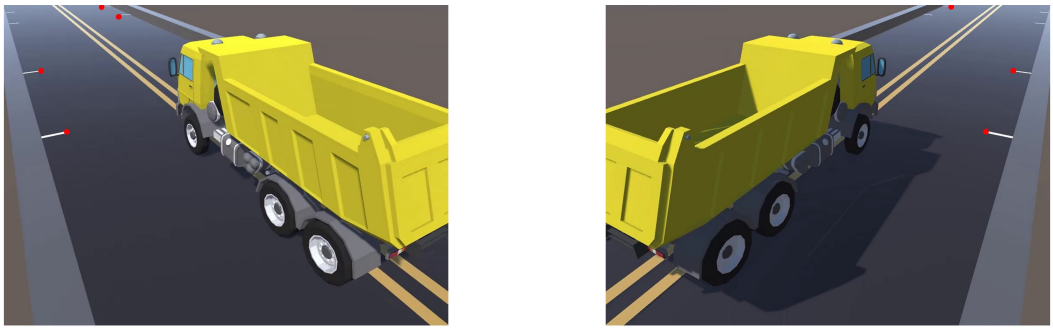


Figure 5.3: Synchronized view of traffic cameras for test No.13 in which a single truck traverses on the bridge from West to East direction starting on the North lane and changes the lane to the South one in the middle of the bridge.

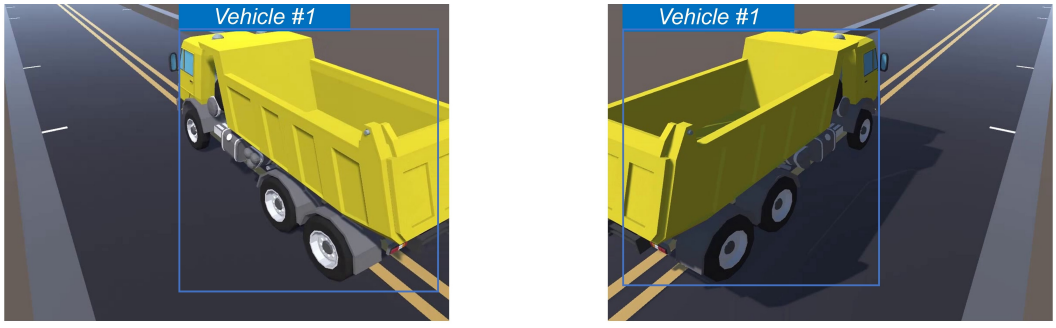
Figure 5.4 shows the synchronized recordings of cameras 4 and 6 at a given time instant. As discussed in Section 4.1.1, each frame is processed to track the pixel coordinates of fixed landmarks. For this purpose, the first frame of video recording of each camera are shown to the user to manually select the landmarks. Then, landmarks are being tracked in each camera recording using optical flow. The tracked location of landmarks are shown by red circles in Figure 5.4.



(a) (b)

Figure 5.4: Captured frames and tracked landmarks for: (a) C4, and (b) C6.

Vehicle tracking algorithm is applied on the recordings of Figure 5.4. The results are shown in Figure 5.5. As can be seen in this figure, the truck is tracked in both camera footage and its location is determined by a bounding box.



(a) (b)

Figure 5.5: Vehicle detection and tracking in frame captured by: (a) C4, and (b) C6.

After tracking the vehicle, tire tracking algorithm is implemented inside each detected bounding box to detect and extract the instance segmentation of vehicle's tires. The segmented tires are assigned a unique I.D. In Figure 5.6, the I.D. format is

denoted as "Tire X-Y," where X is categorized as F (front), M (middle), or R (rear), and Y indicates the direction of the tire, distinguishing between N (North) and S (South) lanes. Having the instance segmentation of tires, the coordinates of the outer edge of tires' contact points with the roadway are determined in the pixel coordinate system. Figure 5.7 shows a closed look over the tracked location of tires' contact points. In this figure, the tracked location of the contact point of one tire from each recording is shown.

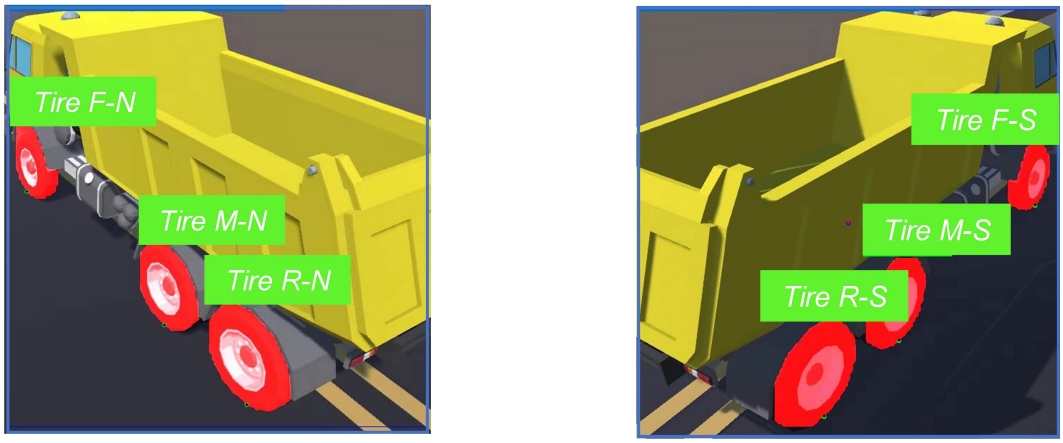


Figure 5.6: Vehicle detection and tracking in frame captured by: (a) C4, and (b) C6.

Figure 5.7: Tracked contact points: (a) Tire R-N in frame captured by C4, and (b) Tire R-S in frame captured by C6.

To transfer the location of the contact points of tires from pixel to the FE model

coordinate system, the transformation matrix is established and applied to the pixel coordinates of the tires' contact points. The tracked location of tires in the FE coordinate system are compared with the collected ground truths. For the given time instant of Figure 5.4, the comparison is shown in Figure 5.8. Calculating the euclidean distance between the ground truth and tracked location of tires reveals the maximum variation of 26cm at the given time instant.

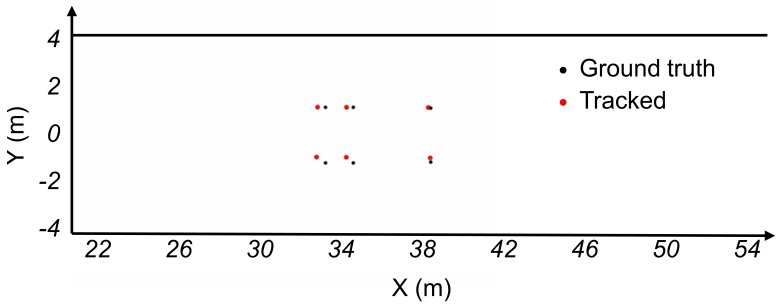


Figure 5.8: Comparison between the tracked ground truth location of tires at a given time instant.

For a better understanding regarding the performance of the proposed traffic tracking method, the prediction error is calculated as a function of truck location. For this purpose, at each recorded frame of test No.7, the euclidean distance between the ground truth and tracked location of all six tires are calculated and the maximum value is reported. Figure 5.9 shows the maximum discrepancies between the prediction and ground truth as a function of the location of vehicle. The maximum error is 33cm.

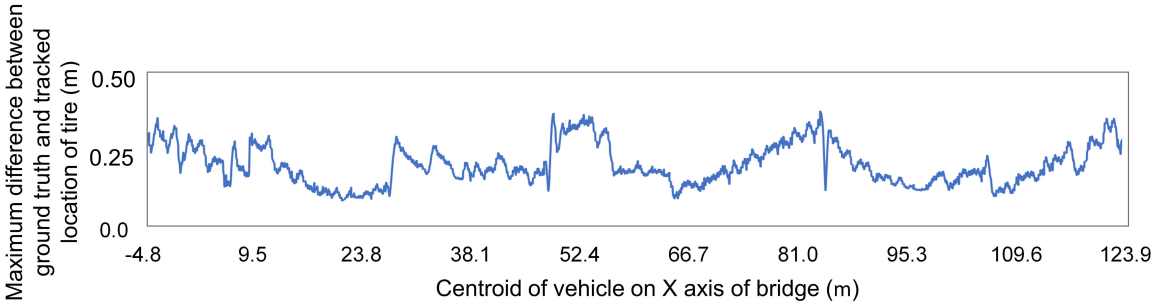


Figure 5.9: Variability of the maximum discrepancy between the tracked and ground truth location of tires as a function of the centroid of vehicle on the length of bridge.

5.2 Application of proposed method using real-world data

After development of the proposed traffic tracking method and its validation using simulation data in a VR environment, the performance of the method is evaluated using real-world data. The real-world data consists of video recordings collected from EY bridge. Unlike the VR simulation environment, the real-world setting does not provide ground truth data, so the evaluation focuses on the feasibility of performing detection and tracking, and manual review of the outputs ensuring the reasonable results. Figure 5.10 shows a view of EY bridge, and Figure 5.11 shows the tri-axle dump trucks. Sections 5.2.1 and 5.2.2 show a brief review of the first two steps of the proposed method in Figures 3.1 and 3.2.

Figure 5.10: East Yutan (EY) bridge.

(a) (b)

Figure 5.11: Trucks on EY bridge: (a) Truck 1, and (b) Truck 2.

5.2.1 Instrumentation and data collection

5.2.1.1 Camera network

The network of traffic cameras was deployed using six Raspberry Pi cameras (also referred to as traffic camera here) which were charged and run for six hours using a 10,000 mAh battery pack. The cameras were installed on tripod supports placed on the roadway shoulder at approximate height of six meters over the bridge deck. The tripod legs were locked using sandbags. The traffic camera network recorded the synchronized passage of traffic across multiple camera views with frame rate of 30 Hz. A GPS (global positioning system) clock was added to the camera network serving as a time server allowing for synchronized recordings from the cameras. Figure 5.12 shows the traffic cameras, battery packs and installation of traffic cameras on tripods positions on the bridge shoulder. Figure 5.13 shows the layout of cameras and landmarks, and Figure 5.14 shows a snapshot of camera recordings during passage of a truck.

5.2.1.2 Accelerometer network

The acceleration response of the bridge was measured using two networks of wired and wireless accelerometers. The wireless sensing network included battery-powered triaxial MEMS wireless accelerometers and a V-Link-200 node [61]. The V-Link-

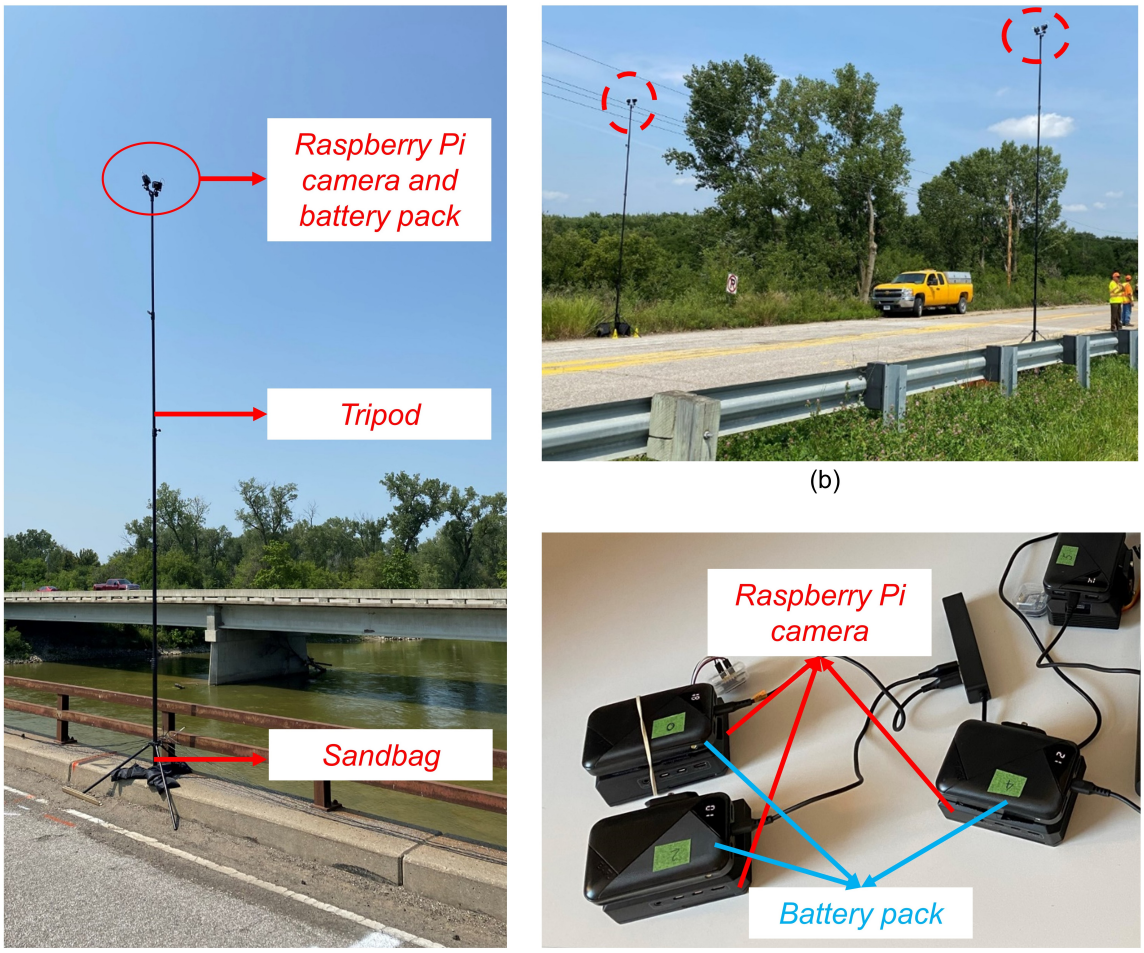


Figure 5.12: Set up of traffic cameras (Raspberry Pi): (a) view of Raspberry Pi cameras, tripod and sandbags, (b) close look to Raspberry Pi cameras mounted on tripods, (c) Raspberry Pi cameras and battery packs being charged.

200, which was shared between the networks of cameras and wireless accelerometers, was used for synchronization between the recordings of wireless accelerometers and traffic cameras. Data collection and coordination between the wireless nodes were carried out through the wireless USB data acquisition gateway. To install the accelerometers, zinc-plated steel washers were glued to the bridge curbs, providing the foundation for securely attaching the accelerometers using magnetic bases. The wired accelerometer network included uniaxial piezoelectric accelerometers [72]. To install the accelerometers, steel boxes were glued to the bridge curbs, providing the foundation for securely attaching the accelerometers using bolts. Figure 5.15 shows the

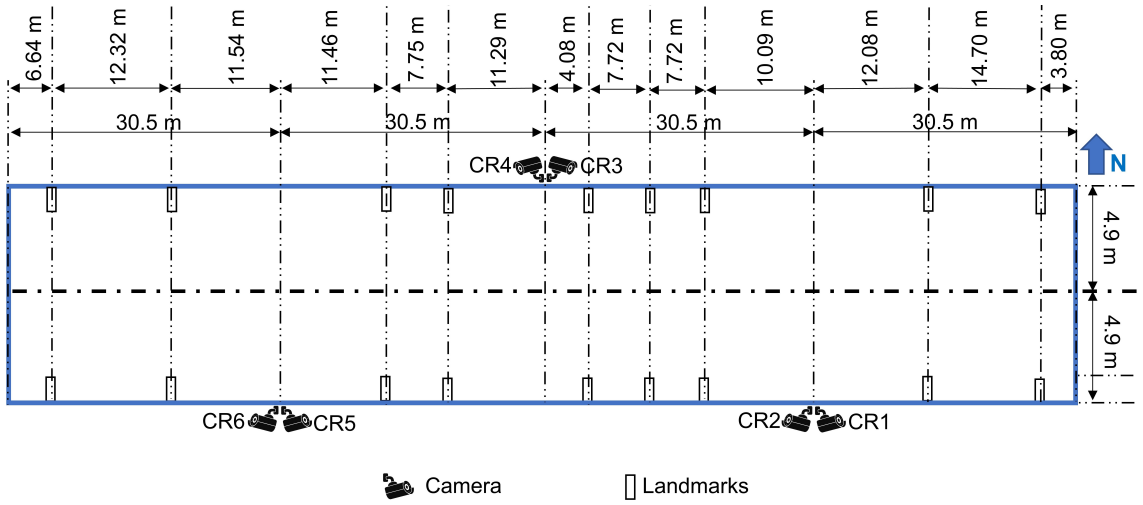


Figure 5.13: EY Bridge geometry and layout of traffic cameras and landmarks in the real-world setting.

Figure 5.14: A snapshot of camera recordings on EY bridge.

installed accelerometers.

Figure 5.15: Wired and wireless accelerometers at the shared spot: (a) side view, (b) top view.

5.2.2 Pre-processing of video recordings

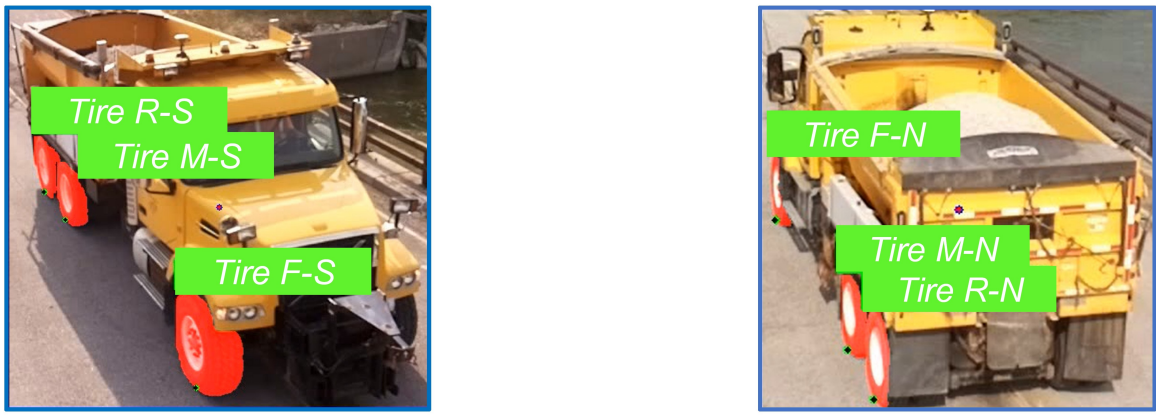
As mentioned in Chapter 3, prior to being used for the purpose of digital twinning, the collected measurements require pre-processing. Pre-processing step includes signal cleaning for collected acceleration responses and the four-phase traffic tracking process on collected video recordings. This section presents the results of applying traffic tracking process on the video recordings in which truck 1 traverses towards east direction on the center line of bridge with the speed of 8 kph. This process is similar to one explained in details in Section 5.1, but using real-world data. Figure 5.16 shows the synchronized frames captured by CR2 and CR3. Figure 5.16 shows the synchronized frames captured by CR2 and CR3. Figure 5.17 and Figure 5.18 present the vehicle tracking and tire tracking results implemented on the video recordings.

(a) (b)

Figure 5.16: Frames captured by camera: (a) CR2, and (b) CR3.

(a) (b)

Figure 5.17: Vehicle detection and tracking in frame captured by: (a) CR2, and (b) CR3.



(a) (b)

Figure 5.18: Tire detection and tracking in frame captured by: (a) CR2, and (b) CR3.

Using the marked fixed landmarks on the bridge deck, and the same process as explained in Section 5.1, the tracked location of tires are determined in the FE coordinate system. Figure 5.19 shows a summary of data pre-processing process.Although cleansing the acceleration measurements is not the focus of this study, the traffic tracking results demonstrate reasonable outputs. This provides a promising foundation for future advancements in structural health monitoring and intelligent transportation systems.

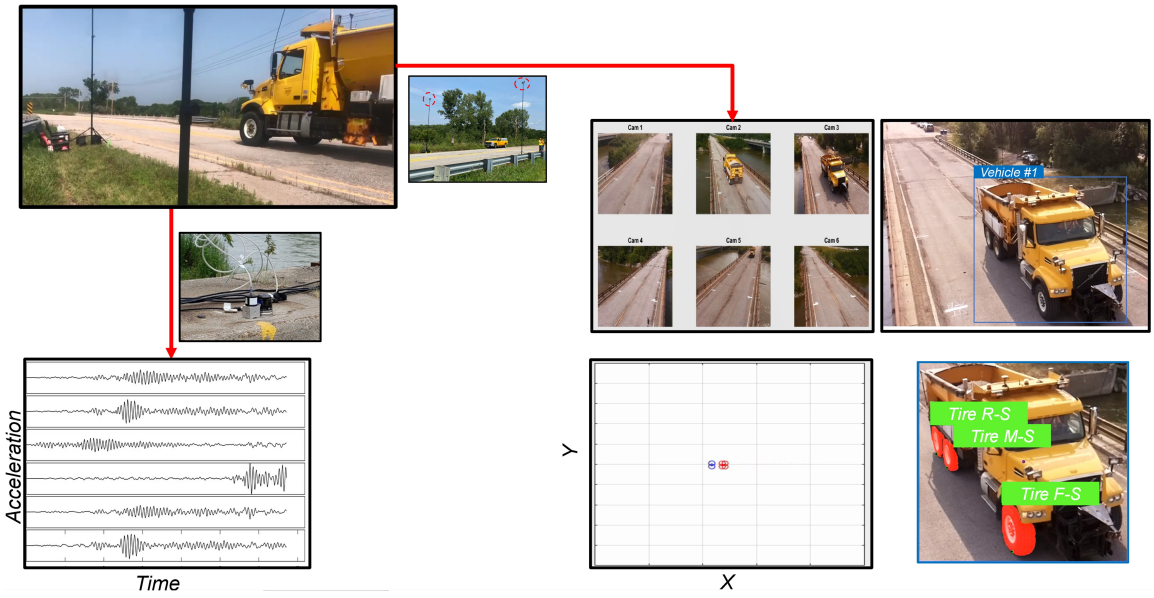


Figure 5.19: A schematic summary over data pre-processing step on EY bridge.

Chapter 6

Conclusion and Future Work

6.1 Summary of research work performed and major findings

The research work presented in this thesis was focused on development and evaluation of a computer vision and deep learning components within an innovative technology-based solution for operational health monitoring of bridge structures. For this purpose, first, a new method for operational health monitoring of bridges was introduced based on a multi-modal data fusion and physics-based Bayesian model updating technique to create the digital twin of the bridge structure. In this method, the bridge structure is instrumented with networks of traffic camera and accelerometers to collect video recordings of traffic and the resulting vibration response of bridge structure. Computer vision and deep learning techniques are used to extract the time history of the location of the tire of vehicles on the bridge. This information, is fused with the collected vibration data and are used as input to a Bayesian inference. Using Bayesian inference, finite element model of bridge is updated and time history of the dynamic load of vehicles on the bridge are estimated. The updated model is interrogated for health estimation of bridge, and the traffic data can be used to improve ITS.

The computer vision and deep learning components of the proposed health monitoring approach were developed and evaluated using a virtual reality (VR) testing platform. For this purpose, a two-lane 122m-long bridge road was simulated in Unity and instrumented with 16 cameras. Traffics of one or two trucks ran on the bridge

and the video recordings of cameras as well as the ground truth time history of the location of tires were collected from VR. The video recordings were processed using the proposed traffic tracking algorithm in which You Only Look Once (YOLO), Deep Simple Online and Real-time Tracking (Deep SORT) algorithms and perspective mapping were implemented. Using the pre-trained YOLOv8, vehicles were classified and detected through bounding boxes, and using Deep SORT, vehicles were tracked and assigned with a unique identified (I.D.). Then, YOLOv8 and Deep SORT were implemented on each bounding box to track the instance segmentation of tires. Having the instance segmentation of tires, the contact point of tires with road were extracted in pixel coordinate system. The perspective transformation between pixel and world (or FE) coordinate systems were developed and used to transform the tire contact points from pixel to world coordinate system. Vehicle detection carried out using pre-trained (using COCO dataset) YOLOv8. However, for the purpose of tire instance segmentation, this model was trained using the Google OpenImage dataset. Moreover, the perspective transformation matrix was developed using the known location of fixed landmarks in world and pixel coordinate system. To track the location of fixed landmarks in video recordings, optical flow algorithm were implemented. Comparing the tracked location of tires with the collected ground truth revealed maximum of 33 cm error, which was an acceptable level of error for this preliminary study.

A preliminary assessment of proposed traffic tracking algorithm was carried out in a real-world setting by instrumenting the East Yutan bridge in Nebraska with networks of traffic cameras. Unlike the VR simulation environment, the real-world setting did not provide ground truth data, so the evaluation carried out with manual review ensuring reasonable results.

6.2 Recommendations for future research work

The work presented in this thesis represents the development of a crucial component of a technology-based framework for operational health monitoring of bridges. This component can be enhanced and improved in several aspects to accelerate a practical

technology solution. Potential areas for improvement and future research are listed below.

- Scaling the developed VR for larger number and variety of class of vehicles traversing on the bridge, and studying its effect on the performance of traffic tracking algorithm.

- Studying the environmental effects,e.g., light and precipitation, on the performance of traffic tracking algorithm. For this purpose, it is required to simulate VR scenarios covering different environmental conditions.

- Reducing the number of cameras deployed on the bridge and studying its effects on the performance of traffic tracking algorithm. For this purpose, it is required to simulate VR scenarios with various number of cameras deployed at different locations.

- Studying the effect of tire tracking error on the digital twinning and health monitoring results and suggest a maximum bound on error.

- Studying further real-world validation scenarios on different bridges and a larger number of vehicles. The objective of these validation studies is to ensure the stability of the proposed algorithm.

While the implementation of the first three items is relatively straightforward, the last two present more significant challenges. These challenges stem from their complexity and the advanced technical requirements needed for successful execution.

References

[1] Mohamed Abdel-Aty, Zijin Wang, Ou Zheng, and Amr Abdelraouf. Advances and applications of computer vision techniques in vehicle trajectory generation and surrogate traffic safety indicators. *Accident Analysis & Prevention*, 191:107191, 2023. ISSN: 0001-4575. DOI: 10.1016/j.aap.2023.107191.

[2] Mohammad Abedin, Francisco J De Caso y Basalo, Nafiseh Kiani, Armin B Mehrabi, and Antonio Nanni. Bridge load testing and damage evaluation using model updating method. *Engineering structures*, 252:113648, 2022. DOI: 10.1016/j.engstruct.2021.113648.

[3] Pedro Azevedo and Vítor Santos. Comparative analysis of multiple yolo-based target detectors and trackers for adas in edge devices. *Robotics and Autonomous Systems*, 171:104558, 2024. ISSN: 0921-8890. DOI: 10.1016/j.robot.2023.104558.

[4] Jahongir Azimjonov and Ahmet Özmen. A real-time vehicle detection and a novel vehicle tracking systems for estimating and monitoring traffic flow on highways. *Advanced Engineering Informatics*, 50:101393, 2021. ISSN: 1474-0346. DOI: 10.1016/j.aei.2021.101393.

[5] Jahongir Azimjonov and Ahmet Özmen. Vision-based vehicle tracking on highway traffic using bounding-box features to extract statistical information. *Computers & Electrical Engineering*, 97:107560, 2022. ISSN: 0045-7906. DOI: 10.1016/j.compeleceng.2021.107560.

[6] Jeremy Bailenson. *Experience on demand: What virtual reality is, how it works, and what it can do*. WW Norton & Company, 2018. ISBN: 978-0-393-35685-4. URL: https://wwnorton.com/books/Experience-on-Demand/ (visited on 07/26/2024).

[7] John Bennetts, Paul J Vardanega, Graham T Webb, Steve R Denton, and Neil Loudon. Analysis of visual inspection data for a sample of highway bridges in the uk. *Proceedings of the Institution of Civil Engineers - Forensic Engineering*, 176(3):79–88, 2023. ISSN: 2043-9903. DOI: 10.1680/jfoen.21.00005.

[8] Alex Bewley, Zongyuan Ge, Lionel Ott, Fabio Ramos, and Ben Upcroft. Simple online and realtime tracking. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 3464–3468, 2016. DOI: 10.1109/ICIP.2016.7533003.

[9] Eric Bianchi and Matthew Hebdon. Visual structural inspection datasets. *Automation in Construction*, 139:104299, 2022. ISSN: 0926-5805. DOI: 10.1016/j.autcon.2022.104299.

[10] Rune Brincker and Carlos E. Ventura. *Frequency-domain identification*. In *Introduction to Operational Modal Analysis*. John Wiley & Sons, Ltd, 2015. Chapter 10, pages 261–280. ISBN: 9781118535141. DOI: 10.1002/9781118535141.ch10.

[11] Giovanni Bruno. Micro non-destructive testing and evaluation. *Materials*, 15(17), 2022. ISSN: 1996-1944. DOI: 10.3390/ma15175923.

[12] Jingwei Cao, Hongyu Zhang, Lisheng Jin, Jiawang Lv, Guoyang Hou, and Chengtao Zhang. A review of object tracking methods: from general field to autonomous vehicles. *Neurocomputing*, 585:127635, 2024. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2024.127635.

[13] Rajeev Kumar Chauhan and Kalpana Chauhan. Intelligent toll collection system for moving vehicles in india. *Intelligent Systems with Applications*, 15:200099, 2022. ISSN: 2667-3053. DOI: 10.1016/j.iswa.2022.200099.

[14] Ge-Wei Chen, Piotr Omenzetter, and Sherif Beskhyroun. Modal systems identification of an eleven-span concrete motorway off-ramp bridge using various excitations. *Engineering Structures*, 229:111604, 2021. ISSN: 0141-0296. DOI: 10.1016/j.engstruct.2020.111604.

[15] Gen Chen and Jia wan Zhang. Intelligent transportation systems: machine learning approaches for urban mobility in smart cities. *Sustainable Cities and Society*, 107:105369, 2024. ISSN: 2210-6707. DOI: 10.1016/j.scs.2024.105369.

[16] Etienne Cheynet, Jonas Snæbjörnsson, and Jasna Bogunović Jakobsen. Temperature effects on the modal properties of a suspension bridge. In Shamim Pakzad and Juan Caicedo, editors, *Dynamics of Civil Structures, Volume 2 - Proceedings of the 35th IMAC, A Conference and Exposition on Structural Dynamics 2017*, Conference Proceedings of the Society for Experimental Mechanics Series, pages 87–93. Springer New York LLC, 2017. DOI: 10.1007/978-3-319-54777-0_12. Publisher Copyright: © The Society for Experimental Mechanics, Inc. 2017.; 35th IMAC Conference and Exposition on Structural Dynamics, 2017 ; Conference date: 30-01-2016 Through 02-02-2016.

[17] A.K. Chopra. *Dynamics of Structures*. Prentice-Hall international series in civil engineering and engineering mechanics. Pearson Education, 2007. ISBN: 9788131713297. URL: https://books.google.com/books?id=0dU1bDaRyP4C.

[18] Sergio Correia and Stephan Luck. Digitizing historical balance sheet data: a practitioner's guide. *Explorations in Economic History*, 87:101475, 2023. ISSN: 0014-4983. DOI: 10.1016/j.eeh.2022.101475. Methodological Advances in the Extraction and Analysis of Historical Data.

[19] Sattar Dorafshan and Hoda Azari. Deep learning models for bridge deck evaluation using impact echo. *Construction and Building Materials*, 263:120109, 2020. ISSN: 0950-0618. DOI: 10.1016/j.conbuildmat.2020.120109.

[20] Hamed Ebrahimian, Rodrigo Astroza, Joel P Conte, and Costas Papadimitriou. A nonlinear model inversion method for joint system parameter, noise, and input identification of civil structures. *Procedia engineering*, 199:924–929, 2017. ISSN: 1877-7058. DOI: 10.1016/j.proeng.2017.09.240. X International Conference on Structural Dynamics, EURODYN 2017.

[21] Hamed Ebrahimian, Rodrigo Astroza, Joel P Conte, and Costas Papadimitriou. Bayesian optimal estimation for output-only nonlinear system and damage identification of civil structures. *Structural Control and Health Monitoring*, 25(4):e2128, 2018. DOI: 10.1002/stc.2128.

[22] Hamed Ebrahimian, Monica Kohler, Anthony Massari, and Domniki Asimaki. Parametric estimation of dispersive viscoelastic layered media with application to structural health monitoring. *Soil Dynamics and Earthquake Engineering*, 105:204–223, 2018. DOI: 10.1016/j.soildyn.2017.10.017.

[23] Saeed Eftekhar Azam, Eleni Chatzi, and Costas Papadimitriou. A dual kalman filter approach for state estimation via output-only acceleration measurements. *Mechanical Systems and Signal Processing*, 60-61:866–886, 2015. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2015.02.001.

[24] Saeed Eftekhar Azam, Costas Papadimitriou, and Eleni Chatzi. Recursive bayesian filtering for displacement estimation via output-only vibration measurements. In *Proceedings of the 2014 World Congress on Advances in Civil, Environmental, and Materials Research*, volume 1, 2014. URL: http://www.i-asem.org/publication_conf/acem14/7.SMM/W4H.4.SM460_573F.pdf (visited on 07/26/2024).

[25] Mohamed Elassy, Mohammed Al-Hattab, Maen Takruri, and Sufian Badawi. Intelligent transportation systems for sustainable smart cities. *Transportation Engineering*, 16:100252, 2024. ISSN: 2666-691X. DOI: 10.1016/j.treng.2024.100252.

[26] Charles R Farrar, Phillip J Cornwell, Scott W Doebling, and Michael B Prime. Structural Health Monitoring Studies of the Alamosa Canyon and I-40 Bridges. Technical report LA-13635-MS, US Department of Energy (US), July 2000. DOI: 10.2172/766805.

[27] James D Foley and Andries Van Dam. *Fundamentals of interactive computer graphics*. Addison-Wesley Longman Publishing Co., Inc., 1982. ISBN: 978-0-201-14468-0.

[28] Michael L Fugate, Hoon Sohn, and Charles R Farrar. Vibration-based damage detection using statistical process control. *Mechanical systems and signal processing*, 15(4):707–721, 2001. DOI: 10.1006/mssp.2000.1323.

[29] Iván García-Aguilar, Jorge García-González, Daniel Medina, Rafael Marcos Luque-Baena, Enrique Domínguez, and Ezequiel López-Rubio. Detection of dangerously approaching vehicles over onboard cameras by speed estimation from apparent size. *Neurocomputing*, 567:127057, 2024. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2023.127057.

[30] Farid Ghahari, Niloofar Malekghaini, Hamed Ebrahimian, and Ertugrul Taciroglu. Bridge digital twinning using an output-only bayesian model updating method and recorded seismic measurements. *Sensors*, 22(3), 2022. ISSN: 1424-8220. DOI: 10.3390/s22031278.

[31] Mohammad Ghanatian, Mohammad Reza Hairi Yazdi, and Mehdi Tale Masouleh. Experimental study on controlling suspended cable-driven parallel robots for autonomous video-capturing of football games and obtaining the statistics of the games. *Mechatronics*, 95:103058, 2023. ISSN: 0957-4158. DOI: 10.1016/j.mechatronics.2023.103058.

[32] Thomas D. Gillespie. Heavy truck ride. In *SAE International Congress and Exposition*, 1985. DOI: 10.4271/850001.

[33] Google. Openimages dataset, 2022. URL: https://storage.googleapis.com/openimages/web/index.html. Accessed: 6 29, 2024.

[34] Benjamin A. Graybeal, Brent M. Phares, Dennis D. Rolander, Mark Moore, and Glenn Washer. Visual inspection of highway bridges. *Journal of Nondestructive Evaluation*, 21:67–83, 2002. DOI: 10.1023/A:1022508121821.

[35] Paul Heckbert. Projective mappings for image warping, 1999. URL: http://graphics.cs.cmu.edu/courses/15-463/2006_fall/www/Papers/proj.pdf (visited on 07/26/2024). Excerpted from pages 17-21 of Fundamentals of Texture Mappingand ImageWarping, Paul Heckbert, Master's thesis, UCB/CSD 89/516, CS Division, U.C. Berkeley, June 1989,

[36] Mohsen Hejrati and Deva Ramanan. Analysis by synthesis: 3d object recognition by object reconstruction. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2449–2456, 2014. DOI: 10.1109/CVPR.2014.314.

[37] Masaru Hoshiya and Etsuro Saito. Structural identification by extended kalman filter. *Journal of engineering mechanics*, 110(12):1757–1770, 1984. DOI: 10.1061/(ASCE)0733-9399(1984)110:12(1757).

[38] Zijian Hu and Wei Ma. Demonstration-guided deep reinforcement learning for coordinated ramp metering and perimeter control in large scale networks. *Transportation Research Part C: Emerging Technologies*, 159:104461, 2024. ISSN: 0968-090X. DOI: 10.1016/j.trc.2023.104461.

[39] M Ilamathi, Sabitha Ramakrishnan, and Rakhul Kumar Babusankar. Proactive hybrid learning framework for real-time multi-vehicle detection in unregulated traffic environments. *Image and Vision Computing*:105081, 2024. DOI: 10.1016/j.imavis.2024.105081.

[40] Bernd Jahne and Bernd Jaehne. *Practical handbook on image processing for scientific applications*. CRC Press, Inc., 2nd edition, 2004. ISBN: 0-8493-1900-5. URL: https://www.routledge.com/Practical-Handbook-on-Image-Processing-for-Scientific-and-Technical-Applications/Jahne/p/book/9780849319006 (visited on 07/26/2024).

[41] Jason Jerald. *The VR book: Human-centered design for virtual reality*. Morgan & Claypool, 2015. ISBN: 978-1-970001-12-9. DOI: doi.org/10.1145/2792790.

[42] Peiyuan Jiang, Daji Ergu, Fangyao Liu, Ying Cai, and Bo Ma. A review of yolo algorithm developments. *Procedia Computer Science*, 199:1066–1073, 2022. ISSN: 1877-0509. DOI: 10.1016/j.procs.2022.01.135. The 8th International Conference on Information Technology and Quantitative Management (ITQM 2020 & 2021): Developing Global Digital Economy after COVID-19.

[43] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. Driving in the matrix: can virtual worlds replace human-generated annotations for real world tasks? In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 746–753, Singapore, Singapore. IEEE Press, 2017. DOI: 10.1109/ICRA.2017.7989092.

[44] R. E. Kalman. A New Approach to Linear Filtering and Prediction Problems. *Journal of Basic Engineering*, 82(1):35–45, March 1960. ISSN: 0021-9223. DOI: 10.1115/1.3662552.

[45] Chan Ghee Koh, Lin Ming See, and Thambirajah Balendra. Estimation of structural parameters in time domain: a substructure approach. *Earthquake Engineering & Structural Dynamics*, 20(8):787–801, 1991. DOI: 10.1002/eqe.4290200806.

[46] H. W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955. DOI: 10.1002/nav.3800020109.

[47] Yong Li, Rodrigo Astroza, Joel P. Conte, and Pedro Soto. Nonlinear fe model updating of seismic isolated bridge instrumented during the 2010 mw 8.8 maule-chile earthquake. *Procedia Engineering*, 199:3003–3008, 2017. ISSN: 1877-7058. DOI: 10.1016/j.proeng.2017.09.397. X International Conference on Structural Dynamics, EURODYN 2017.

[48] Yanqi Lian, Guoqing Zhang, Jaeyoung Lee, and Helai Huang. Review on big data applications in safety research of intelligent transportation systems and connected/automated vehicles. *Accident Analysis & Prevention*, 146:105711, 2020. ISSN: 0001-4575. DOI: 10.1016/j.aap.2020.105711.

[49] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–

755. Springer, Springer International Publishing, 2014. DOI: 10.1007/978-3-319-10602-1_48.

[50] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: a survey. *International journal of computer vision*, 128:261–318, 2020. DOI: 10.1007/s11263-019-01247-4.

[51] E. Lourens, E. Reynders, G. De Roeck, G. Degrande, and G. Lombaert. An augmented kalman filter for force identification in structural dynamics. *Mechanical Systems and Signal Processing*, 27:446–460, 2012. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2011.09.025.

[52] Bruce D Lucas and Takeo Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI'81: Proceedings of the 7th international joint conference on Artificial intelligence*, volume 2, pages 674–679, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc., 1981. DOI: 10.5555/1623264.1623280.

[53] Bruce David Lucas. *Generalized image matching by the method of differences*. Master's thesis, Carnegie Mellon University, 1985. URL: https://www.ri.cmu.edu/pub_files/pub4/lucas_bruce_d_1984_1/lucas_bruce_d_1984_1.pdf (visited on 07/26/2024).

[54] Filipe Magalhães, Álvaro Cunha, and Elsa Caetano. Online automatic identification of the modal parameters of a long span arch bridge. *Mechanical Systems and Signal Processing*, 23(2):316–329, 2009. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2008.05.003.

[55] Niloofar Malekghaini, Farid Ghahari, Hamed Ebrahimian, Matthew Bowers, Eric Ahlberg, and Ertugrul Taciroglu. A two-step fe model updating approach for system and damage identification of prestressed bridge girders. *Buildings*, 13(2), 2023. DOI: 10.3390/buildings13020420.

[56] Niloofar Malekghaini, Farid Ghahari, Hamed Ebrahimian, Matthew Bowers, Hoda Azari, and Ertugrul Taciroglu. Time-domain finite element model updating for operational monitoring and damage identification of bridges. *Structural Control and Health Monitoring*, 2023(1):4170149, 2023. DOI: 10.1155/2023/4170149.

[57] Niloofar Malekghaini, Farid Ghahari, Hamed Ebrahimian, and Ertugrul Taciroglu. Output-only finite element model updating for post-earthquake monitoring. In *14th International Conference on Applications of Statistics and Probability in Civil Engineering, ICASP14*, Dublin, Ireland, 2023. URL: http://hdl.handle.net/2262/103624.

[58] D.M McCann and M.C Forde. Review of ndt methods in the assessment of concrete and masonry structures. *NDT & E International*, 34(2):71–84, 2001. ISSN: 0963-8695. DOI: 10.1016/S0963-8695(00)00032-3.

[59] Kiki McMillan, Kathie Flood, and Russ Glaeser. Virtual reality, augmented reality, mixed reality, and the marine conservation movement. *Aquatic Conservation: Marine and Freshwater Ecosystems*, 27:162–168, S1, September 2017. DOI: 10.1002/aqc.2820. Special Issue:From Science to Solutions: The MPA Legacy from the 2016 Hawai'i World Conservation Congress.

[60] Riddhi Mehta and Dr. Ankit Shah. An insight into real time vehicle detection and classification methods using ml/dl based approach. *Procedia Computer Science*, 235:598–605, 2024. ISSN: 1877-0509. DOI: 10.1016/j.procs.2024.04.059. International Conference on Machine Learning and Data Engineering (ICMLDE 2023).

[61] Microstrain. Wireless accelrometers, 2024. URL: https://www.microstrain.com/. Accessed: 2024-07-05.

[62] M. Moore, Brent M. Phares, B. Graybeal, D. Rolander, and G. Washer. *Reliability of Visual Inspection for Highway Bridges, Volume I: Final Report*. 2001. URL: https://rosap.ntl.bts.gov/view/dot/33883.

[63] John E Mottershead and MI Friswell. Model updating in structural dynamics: a survey. *Journal of sound and vibration*, 167(2):347–375, 1993. DOI: 10.1006/jsvi.1993.1340.

[64] Yair Movshovitz-Attias, Vishnu Boddeti, Zijun Wei, and Yaser Sheikh. 3d pose-by-detection of vehicles via discriminatively reduced ensembles of correlation filters. In *BMVC 2014 - Proceedings of the British Machine Vision Conference 2014*, January 2014. DOI: 10.5244/C.28.53.

[65] Mansureh-Sadat Nabiyan, Faramarz Khoshnoudian, Babak Moaveni, and Hamed Ebrahimian. Mechanics-based model updating for identification and virtual sensing of an offshore wind turbine using sparse measurements. *Structural Control and Health Monitoring*, 28:e2647, 2021. DOI: 10.1002/stc.2647.

[66] Archana Nair and CS Cai. Acoustic emission monitoring of bridges: review and case studies. *Engineering structures*, 32(6):1704–1714, 2010. DOI: 10.1016/j.engstruct.2010.02.020.

[67] Sergey I Nikolenko. *Synthetic data for deep learning*, volume 174 of *Springer Optimization and Its Applications*. Springer, 2021. ISBN: 978-3-030-75177-7. DOI: 10.1007/978-3-030-75178-4.

[68] University of Nebraska-Lincoln. NOBL: nebraska outdoor bridge lab overview page, 2024. URL: https://engineering.unl.edu/outdoorbridgelab/. Accessed: 2024-07-05.

[69] Muhammad Omer, Sam Hewitt, Mojgan Hadi Mosleh, Lee Margetts, and Muhammad Parwaiz. Performance evaluation of bridges using virtual reality. In *Proceedings of the 6th European Conference on Computational Mechanics (ECCM 6) and 7th European Conference on Computational Fluid Dynamics (ECFD 7) Glasgow, UK*, 2018. URL: https://congress.cimne.com/eccm_ecfd2018/admin/files/filePaper/p1846.pdf (visited on 07/26/2024).

[70] Muhammad Omer, Lee Margetts, Mojgan Hadi Mosleh, and Lee S Cunningham. Inspection of concrete bridge structures: case study comparing conventional techniques with a virtual reality approach. *Journal of Bridge Engineering*, 26(10):05021010, 2021. DOI: 10.1061/(ASCE)BE.1943-5592.000175.

[71] Muhammad Omer, Lee Margetts, Mojgan Hadi Mosleh, Sam Hewitt, and Muhammad Parwaiz. Use of gaming technology to bring bridge inspection to the office. *Structure and Infrastructure Engineering*, 15(10):1292–1307, 2019. DOI: 10.1080/15732479.2019.1615962.

[72] PCB Piezotronics, Inc. Model 393b04 — pcb piezotronics, 2024. URL: https://www.pcb.com/products?m=393b04. Accessed: 2024-07-05.

[73] Bart Peeters and Guido De Roeck. Reference based stochastic subspace identification in civil engineering. *Inverse problems in Engineering*, 8(1):47–74, 2000. DOI: 10.1080/174159700088027718.

[74] Bojan Pepik, Michael Stark, Peter Gehler, and Bernt Schiele. Teaching 3d geometry to deformable part models. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3362–3369. IEEE, 2012. DOI: 10.1109/CVPR.2012.6248075.

[75] Veljko Potkonjak, Michael Gardner, Victor Callaghan, Pasi Mattila, Christian Guetl, Vladimir M Petrović, and Kosta Jovanović. Virtual laboratories for education in science, technology, and engineering: a review. *Computers & Education*, 95:309–327, 2016. DOI: 10.1016/j.compedu.2016.02.002.

[76] Vibhor Rastogi. *Virtual reality based simulation testbed for evaluation of autonomous vehicle behavior algorithms*. Master's thesis, Clemson University, 2017. URL: https://tigerprints.clemson.edu/all_theses/2773/ (visited on 07/26/2024). School of Computing.

[77] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: unified, real-time object detection. In *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. DOI: 10.1109/CVPR.2016.91.

[78] Sardar Kashif Ur Rehman, Zainah Ibrahim, Shazim Ali Memon, and Mohammed Jameel. Nondestructive test methods for concrete bridges: a review. *Construction and Building Materials*, 107:58–86, 2016. DOI: 10.1016/j.conbuildmat.2015.12.011.

[79] Qiaoqiao Ren, Jie He, Ziyang Liu, and Min Xu. Traffic flow characteristics and traffic conflict analysis in the downstream area of expressway toll station based on vehicle trajectory data. *Asian Transport Studies*, 10:100138, 2024. ISSN: 2185-5560. DOI: 10.1016/j.eastsj.2024.100138.

[80] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: ground truth from computer games. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 102–118. Springer International Publishing, 2016. DOI: 10.1007/978-3-319-46475-6_7.

[81] AJ Roffel and S Narasimhan. Extended kalman filter for modal identification of structures equipped with a pendulum tuned mass damper. *Journal of Sound and Vibration*, 333(23):6038–6056, 2014. DOI: 10.1016/j.jsv.2014.06.030.

[82] Andrejs Rudzitis and Margarita A. Zaeva. Alternative inverse perspective mapping homography matrix computation for adas systems using camera intrinsic and extrinsic calibration parameters. *Procedia Computer Science*, 190:695–700, 2021. ISSN: 1877-0509. DOI: 10.1016/j.procs.2021.06.081. 2020 Annual International Conference on Brain-Inspired Cognitive Architectures for Artificial Intelligence: Eleventh Annual Meeting of the BICA Society.

[83] Siti Shahirah Saidin, Sakhiah Abdul Kudus, Adiza Jamadin, Muhamad Azhan Anuar, Norliyati Mohd Amin, Zainah Ibrahim, Atikah Bt Zakaria, and Kunitomo Sugiura. Operational modal analysis and finite element model updating of ultra-high-performance concrete bridge based on ambient vibration test. *Case Studies in Construction Materials*, 16:e01117, 2022. ISSN: 2214-5095. DOI: 10.1016/j.cscm.2022.e01117.

[84] Omid Sedehi, Costas Papadimitriou, Daniz Teymouri, and Lambros S Katafygiotis. Sequential bayesian estimation of state and input in dynamical systems using output-only measurements. *Mechanical Systems and Signal Processing*, 131:659–688, 2019. DOI: 10.1016/j.ymssp.2019.06.007.

[85] Chong Shang, Haizhou Ai, Zijie Zhuang, and Long Chen Rui Chen. Improving pedestrian detection in crowds with synthetic occlusion images. In *2018 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*, pages 1–4. IEEE, 2018. DOI: 10.1109/ICMEW.2018.8551575.

[86] Nima Shirzad-Ghaleroudkhani, Mojtaba Mahsuli, S Farid Ghahari, and Ertugrul Taciroglu. Bayesian identification of soil-foundation stiffness of building structures. *Structural Control and Health Monitoring*, 25(3):e2090, 2018. DOI: 10.1002/stc.2090.

[87] Huansheng Song, Haoxiang Liang, Huaiyu Li, Zhe Dai, and Xu Yun. Vision-based vehicle detection and counting system using deep learning in highway scenes. *European Transport Research Review*, 11(51):16, 2019. DOI: 10.1186/s12544-019-0390-4.

[88] Mingming Song, Rodrigo Astroza, Hamed Ebrahimian, Babak Moaveni, and Costas Papadimitriou. Adaptive kalman filters for nonlinear finite element model updating. *Mechanical Systems and Signal Processing*, 143:106837, 2020. DOI: 10.1016/j.ymssp.2020.106837.

[89] Peixiang Sun, Qifeng Yu, and Kesi You. Intelligent traffic management strategy for traffic congestion in underground loop. *Tunnelling and Underground Space Technology*, 143:105509, 2024. ISSN: 0886-7798. DOI: 10.1016/j.tust.2023.105509.

[90] Seyyed Meisam Taheri, Kojiro Matsushita, and Minoru Sasaki. Virtual reality driving simulation for measuring driver behavior and characteristics. *Journal of transportation technologies*, 7(2):123, April 2017. DOI: 10.4236/jtts.2017.72009.

[91] Unity Technologies. Unity real-time development platform — 3d, 2d vr &; ar engine. URL: https://unity.com/ (visited on 06/21/2024).

[92] Soo Siang Teoh and Thomas Bräunl. Symmetry-based monocular vehicle detection system. *Machine Vision and Applications*, 23:831–842, 2012. DOI: 10.1007/s00138-011-0355-7.

[93] Ultralytics. Yolov8. 2023. URL: https://github.com/ultralytics/ultralytics (visited on 06/18/2024).

[94] Unity. Unity asset store, 2024. URL: https://assetstore.unity.com/. Accessed: 2024-07-02.

[95] Federal Highway Administration U.S. Department of Transportation. *Bridge Condition by Functional Classification Area 2019*. 2019. URL: https://www.fhwa.dot.gov/bridge/nbi/no10/fcarea19.cfm (visited on 07/26/2024).

[96] Nicolai Wojke, Alex Bewley, and Dietrich Paulus. Simple online and realtime tracking with a deep association metric. In *2017 IEEE international conference on image processing (ICIP)*, pages 3645–3649. IEEE, 2017. DOI: 10.1109/ICIP.2017.8296962.

[97] Hao (Frank) Yang, Jiarui Cai, Chenxi Liu, Ruimin Ke, and Yinhai Wang. Cooperative multi-camera vehicle tracking and traffic surveillance with edge artificial intelligence and representation learning. *Transportation Research Part C: Emerging Technologies*, 148:103982, 2023. ISSN: 0968-090X. DOI: 10.1016/j.trc.2022.103982.

[98] Zi Yang and Lilian S.C. Pun-Cheng. Vehicle detection in intelligent transportation systems and its applications under varying environments: a review. *Image and Vision Computing*, 69:143–154, 2018. ISSN: 0262-8856. DOI: 10.1016/j.imavis.2017.09.008.

[99] Huang Yijing, Wanyue Wei, Yang He, Wu Qihong, and Xu Kaiming. Intelligent algorithms for incident detection and management in smart transportation systems. *Computers and Electrical Engineering*, 110:108839, 2023. ISSN: 0045-7906. DOI: 10.1016/j.compeleceng.2023.108839.

[100] Ka-Veng Yuen, Houzuo Guo, and He-Qing Mu. Bayesian vehicle load estimation, vehicle position tracking, and structural identification for bridges with strain measurement. *Structural Control and Health Monitoring*, 2023:33, 752776, 2023. DOI: 10.1155/2023/4752776.

[101] Yue Zheng, Honglei Wu, Xin You, and Hao Xie. Model updating-based dynamic collapse analysis of a rc cable-stayed bridge under earthquakes. In *Structures*, volume 43, pages 1100–1113. Elsevier, 2022. DOI: 10.1016/j.istruc.2022.07.023.

[102] Chenyu Zhou, Mark D. Butala, Yongjia Xu, Cristoforo Demartino, and Billie F. Spencer. Fe-based bridge weigh-in-motion based on an adaptive augmented kalman filter. *Mechanical Systems and Signal Processing*, 218:111530, 2024. ISSN: 0888-3270. DOI: 10.1016/j.ymssp.2024.111530.