Static and Dynamic Affordance Learning in Vision-based Direct Perception for Autonomous Driving

by

Jean Marie Uwabeza Vianney

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Mechanical and Mechatronics Engineering

Waterloo, Ontario, Canada, 2019

Examining Committee Membership

The following served on the Examining Committee for this thesis. The decision of the Examining Committee is by majority vote.

Supervisor: Dongpu Cao
Associate Professor, Dept. of Mechanical & Mechatronics Engineering,
University of Waterloo

Internal Member: Amir Khajepour
Professor, Dept. of Mechanical & Mechatronics Engineering,
University of Waterloo

Internal-External Member: Jun Liu
Associate Professor, Dept. of Applied Mathematics,
University of Waterloo

Author's Declaration

This thesis consists of material all of which I authored or co-authored: see Statement of Contributions included in the thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Statement of Contribution

Chapter 4 of this thesis consists of a paper that was co-authored by myself, my supervisor, Dr. Cao, and a PhD student, Mr. Sun. I developed the methodology, data preparation, coding and training. Mr. Sun assisted in reviewing the documentation and contributed to editing the contents.

Chapter 5 of this thesis consists of a paper that was co-authored by myself, my supervisor, Dr. Cao, and a PhD student, Mr. Sun. I formulated the methodology, designed and developed a pipeline that was used for data collection and data annotation. I also collected data and did necessary coding to produce results. Mr. Sun assisted in interpreting the results and drafting the content.

Abstract

The recent development in autonomous driving involves high-level computer vision and detailed road scene understanding. Today, most autonomous vehicles are using the mediated perception approach for path planning and control, which highly rely on high-definition 3D maps and real-time sensors. Recent research efforts aim to substitute the massive HD maps with coarse road attributes. In this thesis, We follow the direct perception-based method to train a deep neural network for affordance learning in autonomous driving. The goal and the main contributions of this thesis are in two folds.

Firstly, to develop the affordance learning model based on freely available Google Street View panoramas and Open Street Map road vector attributes. Driving scene understanding can be achieved by learning affordances from the images captured by car-mounted cameras. Such scene understanding by learning affordances may be useful for corroborating base maps such as HD maps so that the required data storage space is minimized and available for processing in real-time. We compare capability in road attribute identification between human volunteers and the trained model by experimental evaluation. The results indicate that this method could act as a cheaper way for training data collection in autonomous driving. The cross-validation results also indicate the effectiveness of the trained model.

Secondly, We propose a scalable and affordable data collection framework named I2MAP (image-to-map annotation proximity algorithm) for autonomous driving systems. We built an automated labeling pipeline with both vehicle dynamics and static road attributes. The data collected and annotated under our framework is suitable for direct perception and end-to-end imitation learning. Our benchmark consists of 40,000 images with more than 40 affordance labels under various day time and weather even with very challenging heavy snow. We train and evaluate a ConvNet based traffic flow prediction model for driver warning and suggestion under low visibility condition.

Acknowledgements

I would like to sincerely thank my supervisor Prof. Dongpu Cao for his guidance, valuable advice and financial support that allowed my research to be a success and stay within the scope. I would like to thank Chen Sun for valuable discussions in some work presented in this thesis. I sincerely thank all volunteers that manually labeled and verified thousands of sample images.

I would also like to thank my fellow graduate students and researchers from the Cognitive Autonomous Driving lab for their motivation and delicious hotpots during my study.

Dedication

This thesis is dedicated to my family, *the Nyanjwenges*. To my brother Aloys Munyeshyaka, thank you for being an inspiring figure and mentor to me. *Ndabakunda!*

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Autonomous driving involves several key aspects. For any autonomous driving task, a system must first be able to perceive and comprehend the driving environment. It must then reason and make decisions around the most optimal driving action. This thesis focuses on vision-based perception and comprehension layer in the autonomous driving framework. In recent years, there has been a great success in deep neural networks and compute power [18, 45]. Such success has led to advanced perception techniques in computer vision. Using a large scale and annotated driving dataset, a convolutional neural network (CNN) architecture can be used to learn patterns for autonomous driving. To realize a ubiquitous and robust autonomous driving solution, the vehicle must be equipped with several sensors such as LiDAR, Camera, RADAR and/or ultrasonic. These sensors are usually fused to give a comprehensive perception of the environment. Each sensor has its benefits but also some downsides. For instance, LiDAR performs well at capturing range information but with poor resolution, while cameras have high resolution but requires an extra computation step to extract range information [81]. Hence, sensor fusion has been employed to have all

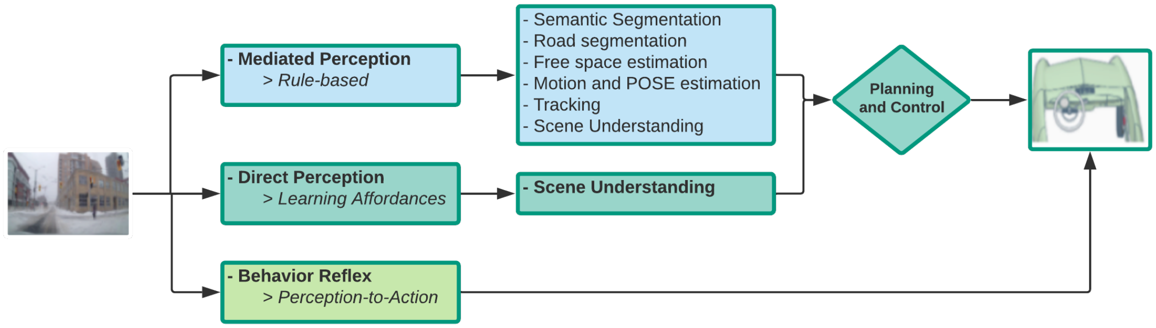


Figure 1.1: Approaches for vision-based perception in autonomous driving.

sensors complementing each other. In this thesis we only use camera-based system given that LiDAR are expensive. Publicly available LiDAR datasets such as KITTI could thus not be aligned to my design and approach.

In literature, three well-discussed paradigms have been used for vision-based perception in autonomous driving [17, 67, 85]. Namely, mediated perception for total scene input to enable rule-based drive-command inference, behavior reflex for predicting action from pixel inputs, and direct perception for making vehicle control inferences from estimated driving affordances. Figure 1.1, shows a break down of the perception problem. Given a set of driving scene image/video as input, we select which paradigm to follow. i.e. whether to use behavior reflex, mediated or direct perception. Depending on the selected approach, several sub-tasks might be completed and their outputs are used to make planning and driving control decisions. Street scene understanding is a common sub-task that must be tackled; except for behavior reflex approach in which driving actions are informed by directly learned patterns between steering angle (from human driver) and input scene. In [17], it is pointed out that the mediated perception may add unnecessary complexity to the perception layer by detecting redundant objects that may not be useful in driving control decisions. On the other hand, behavior reflex may not be robust enough to adapt to all traffic and driving scenarios. This is due to varying complex environments. Consequently, we follow the direct perception approach and focus on learning static and dynamic affordances in the driving environment. As will be discussed in chapter 5, dynamic affordances borders very closely to the behavioral reflex approach. However, my approach focuses on end-to-end learning of decomposed driving tasks (dynamic affordances) which can easily be examined and its prediction verified before being used in a reasoning and decision-making layer.

In [35], Gibson presents the theory of affordance and defines the affordances of the environment as what it offers the animal. In the context of driving, this could be interpreted to mean such cues which a driving environment occupied by a vehicle in the instance of time offers the driver, to influence or inform the driving behavior. Gibson points out that the values and meaning of things in the environment can directly be perceived. The driver perceives such cues via his/her eyes. In the case of autonomous driving, the cues which the immediate driving environment offers to the vehicle are perceived through sensors such as camera, LiDAR, and/or RADAR.

To comprehensively learn affordances in a driving environment, one must have a framework of collecting and labeling large scale driving scenes for training, validating and testing a DeepNet model. Such a framework is presented in Figure 1.2. We use location-based feature matching to do automatic labeling of driving scenes. With sensors mounted on a vehicle recording location and driving video and an interface to access vehicle input and dynamics information such as longitudinal acceleration, steering angle and wheel speed, we label driving scenes for dynamic affordances. Similarly, using vehicle location and a

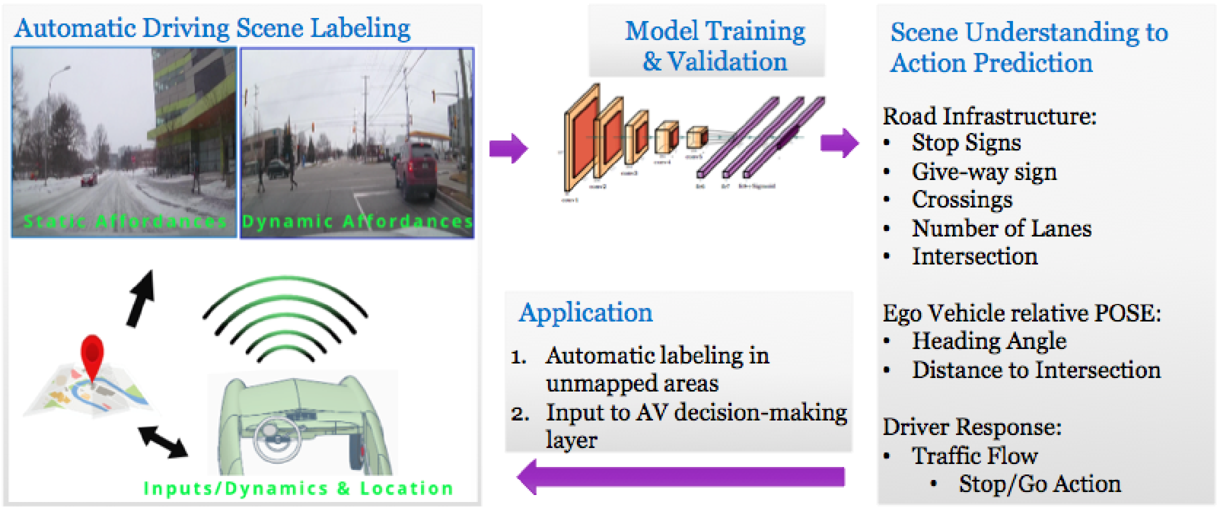

Automatic Driving Scene Labeling
Static Affordances
Dynamic Affordances
Inputs/Dynamics & Location
Model Training & Validation
Scene Understanding to Action Prediction
Road Infrastructure:
• Stop Signs
• Give-way sign
• Crossings
• Number of Lanes
• Intersection
Ego Vehicle relative POSE:
• Heading Angle
• Distance to Intersection
Driver Response:
• Traffic Flow
• Stop/Go Action
Application
1. Automatic labeling in unmapped areas
2. Input to AV decision-making layer

Figure 1.2: Overview of automatic labeling, training and prediction in autonomous driving context.

geo-referenced web map, we query road static affordances such as stop signs, traffic lights, and crossings from the map and tag found features to a synchronized image.

With a labeled large dataset, we train a CNN model for scene understanding to action prediction. Trained and validated models can be used to infer on road infrastructure. Given an image input, the models can predict the number of lanes in the image, whether it contains stop signs, crossings or intersection. The models can also predict an ego vehicle's relative position and orientation to the road. Finally, using dynamic affordance models, we can predict driver responses and make driving decisions in a traffic flow such as whether a vehicle should stop or move. Application of the approach discussed in this thesis can be used as follows:

- To automatically label driving scenes in areas scantly covered by web maps such as rural roads. Human checkers can then verify the labels instead of starting the labeling process from scratch.

- To corroborate HD (High Definition) maps. Most autonomous driving vehicles today use on-board HD 3D maps for road infrastructure representation. Such maps require large storage space and frequent updates [55].

- The predicted affordances can be used as the inputs to autonomous vehicle reasoning and decision making layer.

The rest of this thesis is divided into five chapters. Chapter 2 gives a deep dive into a literature review of the current vision-based perception methods, chapter 3 discusses data collection and annotation, chapters 4 and 5 discusses static and dynamic affordances learning, respectively. Chapter 6 outlines the conclusion of this thesis and future work.

Chapter 2

Literature Review

2.1 Autonomous Driving

2.1.1 History

Since the 1980's there have been many concerted efforts by governments, universities and private research centers to advance intelligent transportation systems. Earliest among these initiatives include Eureka PROMETHEUS [1] and NAVLAB [75]. Pomerleau conducted first autonomous driving demo in 1989. As stated in [57], he presented a 3-layer backpropagation network called ALVINN (Autonomous Land Vehicle In a Neural Network). ALVINN takes in video images and ranging information and then infers on the direction the vehicle should take [57]. Arguably, the DARPA challenge has had the most effect on igniting interest in autonomous driving by the research community. The first DARPA Grand Challenge in

[1] https://www.eurekanetwork.org/project/id/45

2004 required an autonomous vehicle to traverse a 132-mile course through the Mojave Desert in less than 10 hours. For this challenge, no vehicle was able to complete more than 5% of the course. However, just a year later, Stanley (from Stanford University team) was among the 4 vehicles to complete the same challenge within the allocated time [53]. The DARPA Urban Challenge in 2007 required autonomous vehicles to navigate and manage urban traffic scenarios [46]. The good performance by the teams in the competition such as Carnegie Mellon's Tartan Racing [2] prompted the likes of Google to start research and development of autonomous vehicles.

2.1.2 Current State

Today, autonomous driving research is clouded with many companies venturing into autonomous driving as a business. Google's Waymo, Uber and GM's Cruise Automation lead the pack and all have fleets driving autonomously in cities such as Phoenix, Las Vegas and San Francisco in the US [83]. The task of ubiquitous autonomous driving is so challenging and consequently, the existing fleets of autonomous driving vehicles only operate in geofenced areas where high detailed maps have been pre-built. No vehicle has achieved full autonomy yet (Level 5 on SAE self driving level chart. See Figure 2.1). While Waymo is arguably the best with millions of miles of autonomous driving on public roads and billions of miles in simulated driving, it has only tested up to level 4 autonomous driving [5]. This requires an experienced human driver to be in the driving seat and attentive enough to take over if autonomous driving software fails [51].

[2] https://www.cmu.edu/news/archive/2007/November/nov4_tartanracingwins.shtml

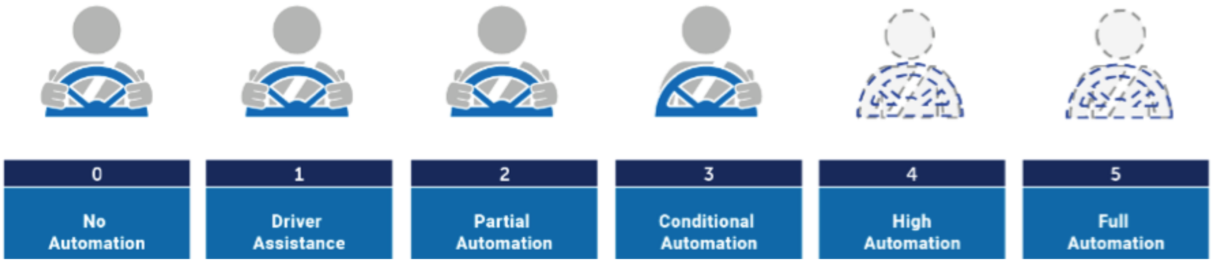


Figure 2.1: Automation levels according to Society of Automotive Engineers (SAE) [51].

In Figure 2.1, SAE lists 6 automation levels. Level 0 involves no automation at all and the driver performs all driving tasks. Level 1 has some form of driver assistance such as collision warnings. Level 2 is a Partial Automation level with the ability to automate vehicle acceleration and steering wheel for assisted lane change and other maneuvers. However, the driver must remain engaged at all times. Tesla Autopilot [3] offers level 2 of automation. In level 3 (conditional automation), while the driver must be ready to take control of the vehicle at all times, he/she is not required to monitor the environment, and the vehicle must be able to sense and understand the static and dynamic features of the environment. High Automation (level 4) is geo-fenced in that the vehicle must be able to perform all driving tasks under certain conditions mostly constrained in a specified geo-location. A human driver is still required to be present in the vehicle. Many autonomous driving vehicles offering ride-sharing services today fall either in level 3 or level 4 capable of only driving in certain geographical areas under strict conditions. The ultimate goal is to achieve Full Automation level (level 5) with the vehicle able to perform all driving functions with no restrictions and ubiquitously [51].

[3] https://www.tesla.com/en_CA/autopilot

2.1.3 Benefits of Autonomous Driving

The projected social and economic benefits of autonomous driving vehicles are enormous. The autonomous driving technology will cause total disruption to the transportation sector, as we know it today. It will impact vehicle safety, congestion and travel behavior [26]. People will be able to continue working in office-like vehicles while traveling from home to work or vise-Versa. Hence, being able to live far from cities where life is more affordable. If the current success by Uber share-rides is anything to go by, there will be fewer and fewer people owning vehicles while depending on shared autonomous vehicles to pick them up on time by subscribing to an on-demand service [26]. There will be much fewer parking spaces. The sick, disabled and elderly will benefit most from such a service, which will also reduce travel time, saving fuel and lowering emissions [26]. However, the success of autonomous driving technology will not come easy. It requires huge investments in sensor and control, perception, prediction, and planning research, along with setting new policies to guide the deployment of autonomous driving vehicles from the testing stage to the full-adoption stage. As this thesis focuses on perception, we give a review of perception approaches and required sensors in the following subsection.

2.2 Sensors, Datasets and Perception Paradigms in Autonomous Driving

2.2.1 Sensors

For the autonomous driving vehicle to safely navigate from point A to point B, it must be able to perceive, understand and localize its self in the environment. The vehicle can achieve this using several sensors (see Figure 2.2) that can be categorized into two types: exteroceptive and proprioceptive Sensors.

Exteroceptive sensors are used for environment perception and distance to object prediction. They include LiDAR (Light Detection and Ranging), RADAR, camera and ultrasonic [15]. Most autonomous driving vehicles use LiDAR sensors as primary sensors for perception since they accurately capture the environment in 3D point cloud representation [15]. However, LiDAR has low resolution due to sparse point clouds and may not be efficient for small object detection such as traffic signs. They are also quite expensive and may not work in harsh weather conditions. Consequently, a robust and optimal autonomous driving perception layer should use multiple fused sensors. In this thesis, we advance the ideal also expressed in [81] that cameras are a good alternative to LiDAR. Cameras are quite affordable and offer high resolution with color and texture [15]. Data captured with a camera can be presented in several forms including 2D image, the depth map and 3D point clouds [16, 27, 81, 88]. In this thesis, we use monocular images to learn affordances from the environment.

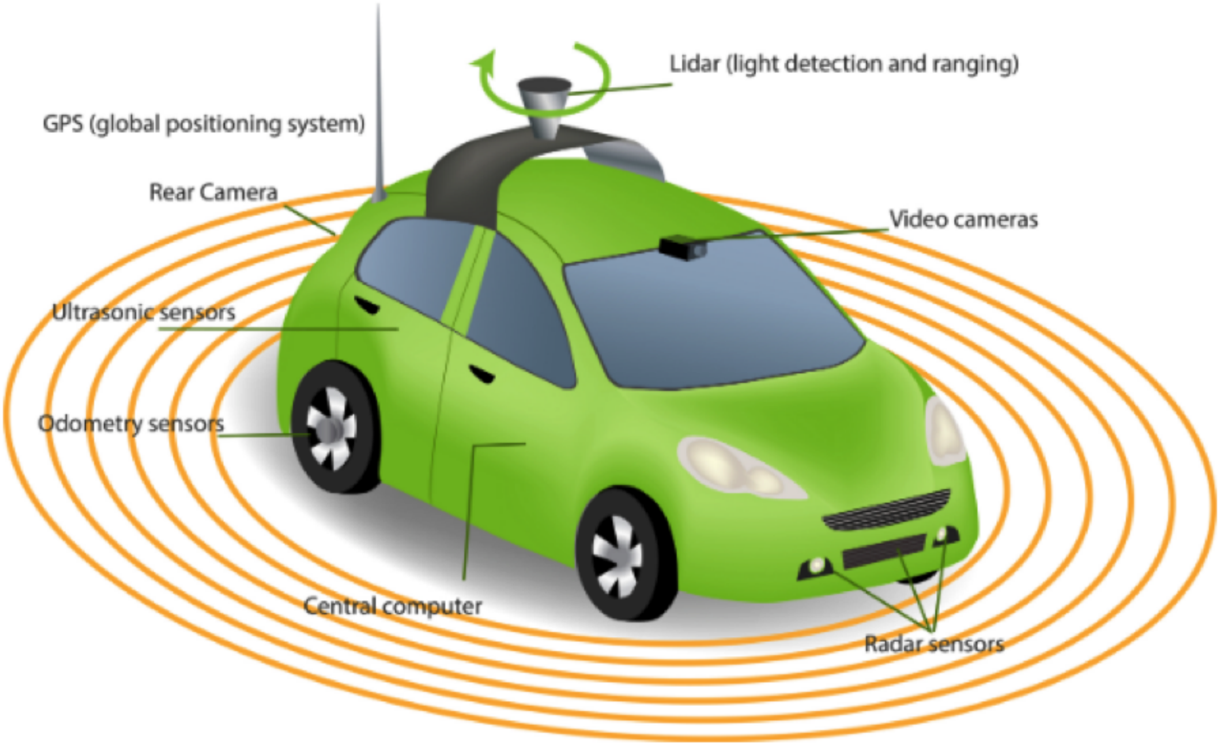


Figure 2.2: Sensors that enable an autonomous vehicle to perceive and navigate through an environment [4].

Proprioceptive sensors measure or give information about the autonomous vehicle itself (ego vehicle). They include but not limited to GNSS, IMU and encoders [15]. Such sensors are usually fused in a Kalman filter to offer a refined localization solution [63]. In this thesis, we used vehicle proprioceptive sensors to get real-time vehicle updates such as steering wheel, speed and throttle input.

2.2.2 Perception Paradigms

Mediated perception follows a computational/representational view as expressed in cognitive science [78]. With the mediated perception approach, an entire scene is parsed to make a driving decision. It involves multiple sub-tasks for recognizing objects relevant to driving such as road free space segmentation, traffic signs, and object detection [17]. Since mediated perception involves solving sub-components of the bigger perception problem, researchers mainly focus on solving various challenging sub-components of mediated perception. In [30], Geiger et al. focused on scene understanding and presented a novel model for multi-object traffic scene understanding from movable platforms. Their model does not rely on GPS, LiDAR or map inputs. Rather, they segment a video sequence to interpret driving scene layout visual cues such as free and occupied space, vanishing points and 3D scene flow. Geiger et Al. divide the visual cues into topology and geometrical models from which they can make scene layout inferences such as the number and location of streets as well as position and orientation of traffic participants. Figure 2.3 shows the topology model for road intersection classified into 7 parts [30]. We use the intersection definition in Figure 2.3, in this thesis for intersection affordance learning. 2D and 3D object detection

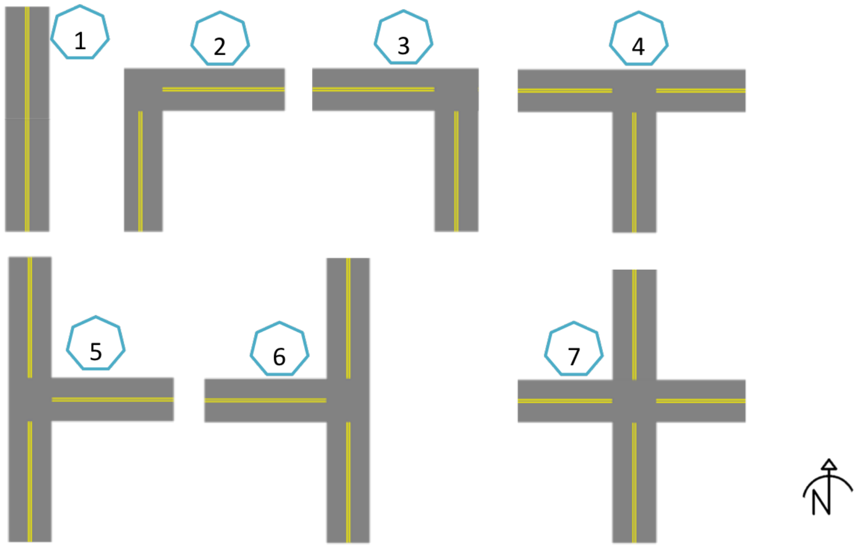


Figure 2.3: Intersection topology with north as the driving direction. Redrawn from the topology model defined by Geiger et Al [30].

and semantic segmentation approaches described in [25, 49, 58, 82, 87] are all part of the mediated perception.

Traffic scene and driving context understanding are ongoing challenges in autonomous driving. Over the past few years, the focus has been put towards scene understanding as a primary challenge in autonomous driving, especially since DARPA Urban Challenge [13]. One type of strategy for the static driving context understanding is simultaneous localization and mapping (SLAM) [20]. A virtual representation of the road, traffic, and surrounding buildings can be constructed based on the pair-matching of real-time sensor data and pre-stored HD maps. With the detailed driving context representation, the detailed path planning and driving policy can be further derived. However, the main bottlenecks for this type of approaches are the high requirements on computing power and data transmission [69].

Vision-based methods try to mimic the human driver using camera recorded images as major sensory input. German Ros et. al presented an Offline-Online perception framework in [61] where the 3D semantic maps are pre-stored offline and online semantic segmentation can be achieved by performing SVM based classification on video-sequences. While the re-localization process in this framework can be achieved real-time, the online retrieval of semantics does not necessarily adapt to environmental change. Authors in [74] proposed a unified multi-net structure that performs the joint classification, detection and semantic segmentation in real-time. Such driving context understanding methods like semantic segmentation with camera images eventually aim to assist the control design for the ego vehicle. The research group at Princeton University demonstrated the idea of directly learning the affordances from an image using the direct perception approach [17]. They train images (from a car racing game TORCS) using a ConvNet to predict affordances such as host vehicle distance to the front vehicle or left/right lanes for driving action. They tested their approach both in virtual and real environments and reported a good performance in close range to the state-of-the-art deformable parts model car detector [31]. Based on the determined affordances, they built a simple rule-based controller for vehicle control in TORCS. This idea proves that meaningful driving affordances can be incorporated into autonomous driving decision making.

Axel Sauer et. al [64] examined the idea of direct perception by extending the driving scenario in urban driving using more photo-realistic simulation platform CARLA [24]. The images with the affordance attribute attached in both works are collected easily through the provided simulation API. However, affordance annotation is a challenging task in real driving environments since it requires a certain level of understanding of the current driving environment. In 2016, Ari Seff et. al [68] presented the affordance learning methods by combining the Google Street view panoramas and OSM road attributes. They used cropped Google street view panoramas to train a CNN model for a list of selected static road attributes. However, Google street views were mostly collected in summer at day time with very clear visibility that may not capture extreme cases of driving under heavy conditions such as snowy roads. Hence, such data may not be enough for robust model training. The other issue of directly mapping affordance from OSM is that the static road attributes may be outdated and left with outdated annotations. In this thesis, we train a CNN model with images downloaded from Google Street View. However, we also collected data under various time, visibility and weather conditions using an iPhone App developed for this purpose. The OSM attributes are queried and corroborated with phone sensors and vehicle proprioceptive sensors. The vehicle dynamics and driver's control input are also collected. Consequently, data collected using our framework could also be used as a benchmark for end-to-end imitation learning and control design.

2.2.3 Dataset Benchmarks for Autonomous Driving

Recent research [14, 84, 85] demonstrated that data-driven perception models often surpass the hard-coded reasoning in context prediction leveraging the large-scale data since much more expert driving experience can be exploited. However, the process of data-set collecting and labeling often requires a huge amount of effort. The aforementioned research works [17, 24, 64] use simulation data since the ground truth information is programmed and can be exported through provided APIs. However, there is still a gap between the simulated environment and the real-world data [79]. Many open-sourced driving datasets received increasing attention in recent years. The Caltech lane dataset [8] focused on lane marking whereas KITTI [32] provides fairly well-annotated images and LiDAR dataset for 2D and 3D object detection. However, the KITTI dataset is not suited for affordance learning since, only pedestrian, cyclist and car classes are annotated. With high-end expensive sensors the vehicle dynamic state estimation could be achieved by methods mentioned in recent review paper [36]. Recently, Xu et. al published the BDD100K dataset [85] where the diverse driving data are collected in a distributed way in collaboration with Uber drivers across California and New York and annotated by human labor. These datasets were collected and labeled with a deliberately designed system but are not automated and still quite expensive. OpenStreetMap (OSM) [37] is an open-source mapping project started since 2004, where over 21 million miles of road geographical information is available for public use. In [68], the authors trained a CNN model to predict road attributes using images from Google Street View (GSV). They presented an automatic labeling method based on location matching with attributes from OSM. We follow a similar trend with [68] where we use 'cheap' data with automatic labeling to teach a model to predict important driving cues given image inputs.

In 2016, the Cityscape benchmark [22] collected various urban driving scenes across 50 cities for semantic segmentation tasks. Seokju Lee et al. open-sourced their benchmark for lane and road marking detection under various weather and day time in [50]. However, accurate annotation is time and labor-consuming. Baidu proposed their annotation pipeline along with the Apolloscape [43] benchmark in 2018. At the same time, UC Berkeley released BDDV dataset [89], which provided a semantic evaluation benchmark containing large-scale driving datasets distributed across four cities. They also provided a user-friendly labeling interface for both bounding box and region annotation. Although the aforementioned benchmarks and annotation approaches provide a promising way of scalable annotation framework for autonomous driving, the annotation process is not automated, human annotators have to go through every image and draw either bounding boxes and curved areas for segmentation tasks. In this thesis, we propose an automatic affordance labeling framework that can widely be distributed via smart-phones for crowd-sourcing data collection efforts.

Chapter 3

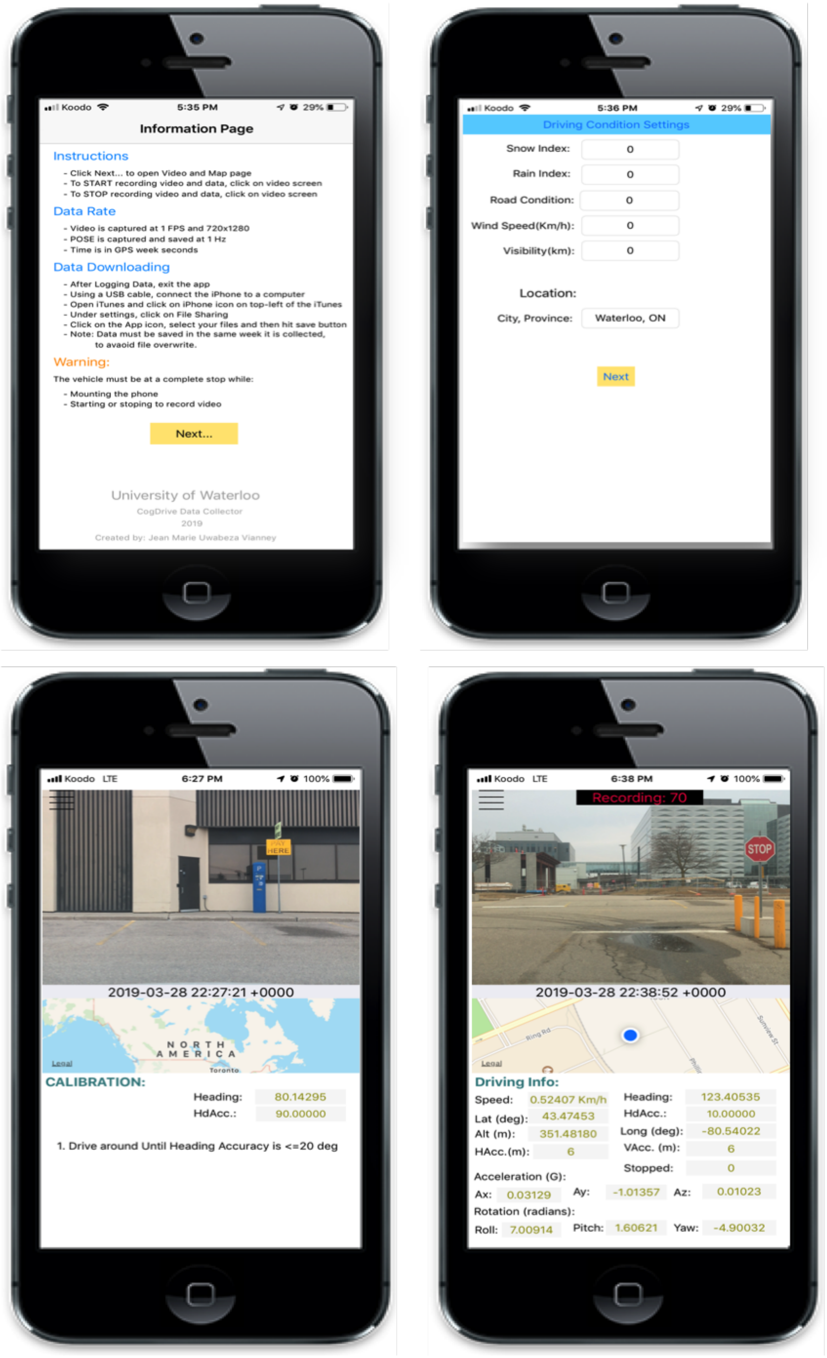
CogDrive Data Collection App

3.1 App Design

We designed an iPhone app specifically to help collect dynamic data for driving. The app can record a driving scene video while also logging the phone's location, orientation, acceleration, and speed. Using the app we were also able to log driving events such as road condition and visibility during data collection. The app was designed to be user-friendly and interactive for ease of distribution. The app has several pages including Information page, Driving Condition Settings page, Calibration page, and Main Data Logging page. Figure 3.1 shows the designed CogDrive app.

3.1.1 Information Page

Information page provides simple instructions on operation of the app as presented here:



Information Page

Instructions

- Click Next... to open Video and Map page
- To START recording video and data, click on video screen
- To STOP recording video and data, click on video screen

Data Rate

- Video is captured at 1 FPS and 720x1280
- POSE is captured and saved at 1 Hz
- Time is in GPS week seconds

Data Downloading

- After Logging Data, exit the app
- Using a USB cable, connect the iPhone to a computer
- Open iTunes and click on iPhone icon on top-left of the iTunes
- Under settings, click on File Sharing
- Click on the App icon, select your files and then hit save button
- Note: Data must be saved in the same week it is collected,
  to avaoid file overwrite.

Warning:

The vehicle must be at a complete stop while:

- Mounting the phone
- Starting or stoping to record video

Next...

University of Waterloo
CogDrive Data Collector
2019
Created by: Jean Marie Uwabeza Vianney

Driving Condition Settings

Snow Index:	0
Rain Index:	0
Road Condition:	0
Wind Speed(Km/h):	0
Visibility(km):	0

Location:

City, Province: Waterloo, ON

Next

2019-03-28 22:27:21 +0000

CALIBRATION:

Heading: 80.14295
HdAcc.: 90.00000

1. Drive around Until Heading Accuracy is <=20 deg

2019-03-28 22:38:52 +0000

Driving Info:

Speed:	0.52407 Km/h	Heading:	123.40535
Lat (deg):	43.47453	HdAcc.:	10.00000
Alt (m):	351.48180	Long (deg):	-80.54022
HAcc.(m):	6	VAcc. (m):	6
		Stopped:	0

Acceleration (G):

Ax: 0.03129 Ay: -1.01357 Az: 0.01023

Rotation (radians):

Roll: 7.00914 Pitch: 1.60621 Yaw: -4.90032

Figure 3.1: CogDrive app designed for dynamic affordance data collection. Top left image shows Information page. Top right image shows the Driving Condition Settings. Bottom left image shows a Calibration page while bottom right image shows the Main page for data recording.

1. Once the app has been opened, click on **Next** button to go to next page
2. To **START** recording video and logging data, click anywhere on the video screen
3. To **STOP** recording video and logging data, click anywhere on the video screen

It also provides information on accessing and downloading data after logging. The data can be downloaded via iTunes as follows:

1. After data has been logged, exit the app
2. Using a **USB cable**, connect the iPhone to a computer with iTunes
3. Open iTunes and click on **iPhone icon** on top-left of the iTunes
4. Under settings, click on **File Sharing**

While mounting the phone on dashboard and starting or stopping to record datasets, the vehicle must be at a complete stop for safety. The data is logged at the following rate:

1. Video is captured at 1 FPS (Frame Per Second) and 720x1280 resolution
2. Position and Orientation info is logged at 1 Hz and time is in GPS week seconds

3.1.2 Driving Condition Settings Page

The weather and road condition information matters a lot to a human driver. Consequently, such information matters a lot to an autonomous driving vehicle for it to be able to operate in all conditions without compromising safety and comfort. Having such information incorporated in the training dataset is crucial and we made an effort to capture it as accurately as possible. The Driving Condition Settings page allows associating driving condition events to the data being recorded.

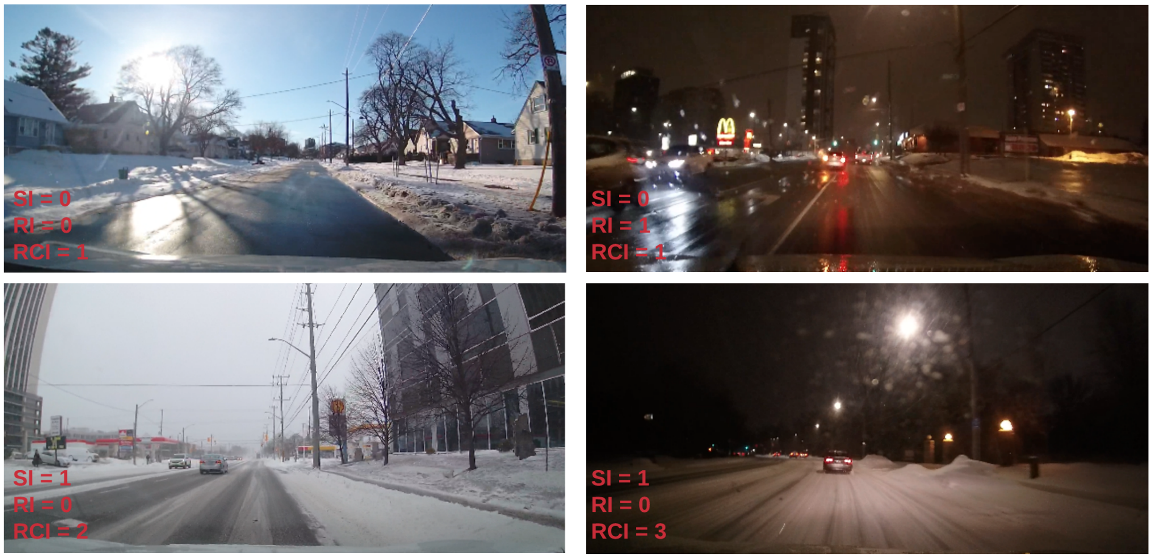


Figure 3.2: Images showing driving scenes under various weather conditions. SI = Snow Index, RI = Rain Index, and RCI = Road Condition Index.

Snow Index

A user can enter a numeric integer value between 0 and 6 to represent the snow level at the time of data collection. The snow levels (TABLE. 3.1) are classified based on snow types defined by NSIDC [1]. Figure 3.2 (bottom) illustrates the driving scene with snow flurry.

[1] https://nsidc.org/

Table 3.1: Indices ranging from 0 - 6 to indicate snow level at the time of data collection

Snow Index	**Snow Level Definition**
0	Not snowing or after snow
1	Snow Flurry
2	Freezing rain
3	Drifting/blowing snow
4	Snow burst/snow storm
5	Blizzard
6	Thunder snow

Rain Index

The rain Index value can be an integer between 0 to 3. Setting the rain Index similar to snow and road condition indices is based on a user opinion about the rain severity on the day and time of data collection (see Figure 3.2). It also depends on how they match that opinion to a rain level definition expressed in TABLE. 3.2.

Table 3.2: Indices ranging from 0 - 3 to indicate rain level at the time of data collection

Rain Index	**Rain Level Definition**
0	Not raining or after rain
1	Light rain - visibility not affected/freezing rain
2	Moderate rain - visibility affected but not normal driving behaviour
3	Heavy rain - visibility affected and driving behaviour affected

Road Condition

The road condition is an important indicator that greatly influences the driving behavior. For instance, with snow deposits on the road (see Figure 3.2 bottom right), the driver must drive slower than normal and leave a larger following gap. They must also learn to anticipate and quickly and safely react to the events near traffic lights and four or three-way intersections. Such driving behaviours must be incorporated within autonomous driving perception and comprehension layer. Consequently, we collect data attaching road condition, rain level and snow deposits level indices to help train and test the perception layer in a realistic way encountered in the environment. We also record wind speed and visibility from The Weather Channel [2] as additional information that can be used in planning and prediction layer. Visibility may not affect the autonomous vehicle due to effective sensor fusion but will surely influence the behaviour of other road participants and hence could be useful in road scene behaviour analysis. In our case, since we use a visible light camera only, these attributes are absolutely important even for the perception layer. TABLE. 3.3 shows classified road condition levels.

Table 3.3: Indices ranging from 0 - 3 to indicate road condition level at the time of data collection

Road Condition Index	**Road Condition Level Definition**
0	Clear/Dry road
1	Wet but no snow on the road
2	Light snow deposits on the road but road lanes visible
3	Snow deposits - road lanes not visible

3.1.3 Calibration, Main Logging, and OSM Limit Alert

The Calibration page as shown in Figure 3.1 displays the heading and heading accuracy measurements along with the calibration instruction for the user to follow. The calibration steps are only revealed in sequence. i.e. after the last step is completed, a new step will be displayed until calibration is completed. The Main Logging page (see Figure 3.1) displays

[2] https://weather.com/en-CA/weather/today/l/CAON4756:1:CA

the driving scene on top, the map with time in the middle and the driving info on the bottom of the page. Some of the displayed Driving info are speed, location, acceleration, and rotation of the phone while driving. Also, some sensor accuracy such as heading and location accuracy are displayed on the screen. As will be shown in chapter 4 and 6, the location-based query of the road attributes from OSM, dictates that the OSM map must be downloaded beforehand. Since the OSM map is large and requires a lot of memory and time to download, only a segment of the map covering Kitchener-Waterloo was downloaded. To alert the user when they start driving in an area not covered by the downloaded OSM map, the app will display a yellow transparent layer over the map. When this happens, the user must reroute back into the zone covered by the downloaded map in which case the yellow layer would disappear. This is a simple geo-fencing technique but saves a user a lot of time collecting data in an area where road attributes would return nil on a query.

3.2 Calibration Procedure and System setup

3.2.1 Calibration Procedure

The iPhone [3] consists of inexpensive low-grade sensors such as accelerometer, gyroscope and GNSS receiver. Consequently, any measurements by non-calibrated iPhone sensors would result in noisy and biased measurements. For our data collection using CogDrive Data Collector app, we first calibrate gyroscope and accelerometer sensors. However, we must note that this only reduces noise to some degree by first taking an average of measurements

[3] https://developer.apple.com/documentation/coremotion/cmmotionmanager?language=objc

while holding the device steady for 60 seconds. No effort was put into correcting the GNSS positioning. However, as will further be explained in chapter 5, we record both horizontal and vertical positioning accuracy. The positioning accuracy is used to sort coordinates that are used in image-to-map proximity query of road features. Below is the procedure we followed during iPhone accelerometer and gyroscope sensor calibration:

1. While the vehicle is at a complete stop, mount the iPhone on a dashboard in portrait mode

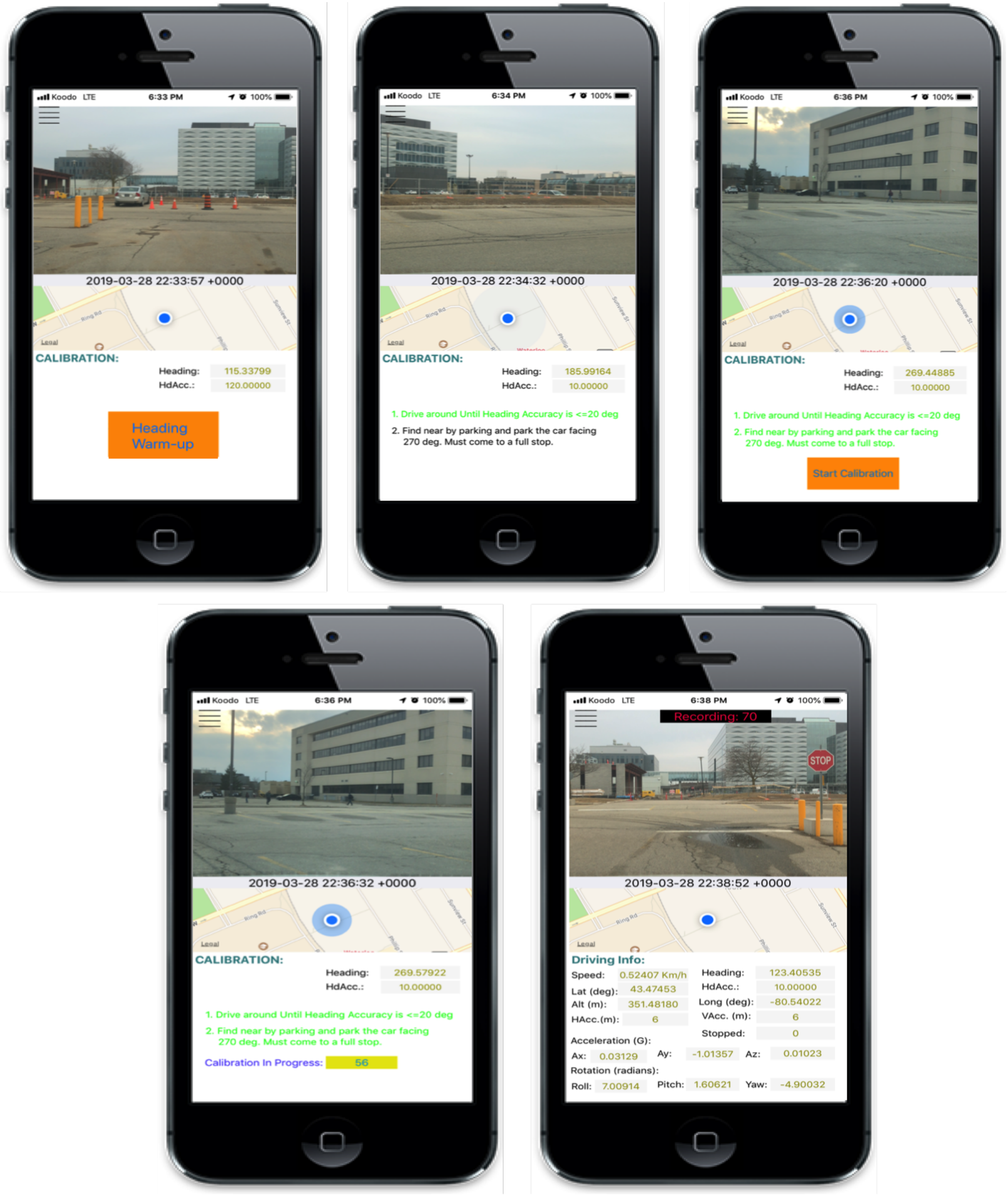
2. Open the CogDrive Data Collector app, read the instructions and click **NEXT** button on the Information page

3. Complete the Driving Condition Settings and then click on **NEXT** button to move to the calibration page

4. Now, click on **Heading Warm-up** button and drive around until heading accuracy drops below 20 *deg* (see top left, and top middle images in Figure 3.3).

   Driving around with some varying acceleration and away from metal structures helps to improve the magnetometer sensor reading and improves heading accuracy. in iPhone, the heading is measured by the magnetometer sensor. Gyroscope measurements are referenced to the north when the iPhone is lying on a flat surface with z axis facing up [1].

5. Find nearby parking and safely park the car facing 270 *deg*. Make sure the vehicle is at a full stop and then click on **Start Calibration** button (see top left and bottom left images in Figure 3.3). The message *Calibration in Progress* will be displayed with



2019-03-28 22:33:57 +0000

CALIBRATION:

Heading: 115.33799

HdAcc.: 120.00000

2019-03-28 22:34:32 +0000

CALIBRATION:

Heading: 185.99164

HdAcc.: 10.00000

1. Drive around Until Heading Accuracy is <=20 deg
2. Find near by parking and park the car facing 270 deg. Must come to a full stop.

2019-03-28 22:36:20 +0000

CALIBRATION:

Heading: 269.44885

HdAcc.: 10.00000

1. Drive around Until Heading Accuracy is <=20 deg
2. Find near by parking and park the car facing 270 deg. Must come to a full stop.

2019-03-28 22:36:32 +0000

CALIBRATION:

Heading: 269.57922

HdAcc.: 10.00000

1. Drive around Until Heading Accuracy is <=20 deg
2. Find near by parking and park the car facing 270 deg. Must come to a full stop.

Calibration In Progress: 56

Recording: 70

2019-03-28 22:38:52 +0000

Driving Info:

Speed: 0.52407 Km/h

Heading: 123.40535

Lat (deg): 43.47453

HdAcc.: 10.00000

Alt (m): 351.48180

Long (deg): -80.54022

HAcc.(m): 6

VAcc. (m): 6

Stopped: 0

Acceleration (G):

Ax: 0.03129 Ay: -1.01357 Az: 0.01023

Rotation (radians):

Roll: 7.00914 Pitch: 1.60621 Yaw: -4.90032

Figure 3.3: Images showing sensor calibration steps.

a count down from 60. After 60 seconds, the average measurements will be computed for all sensors along $X-Y-Z$ axes. Since the vehicle was at a stop and the phone was rigidly mounted to the vehicle's dashboard, any averaged measurements are taken as noise n. The new measurement is computed as shown in Eq. 3.1.

$$\hat{x} \quad = \quad x + n \tag{3.1}$$

Parking the car at 270 *deg* ensures that the iPhone is oriented to the north when in portrait mode and $+y$-axis facing up as shown in Figure 3.4.

6. Calibration is now complete. Click on the video screen to start recording data.

3.2.2 System setup

We collect driving scenes dataset using a setup that includes a smartphone and a camera mounted on the vehicle dashboard as shown in Figure 3.4. The setup also includes a CAN bus OBDII interface. Such setup is affordable, lightweight and can easily be distributed for cloud sourcing. Consequently, a temporal and large dataset can be collected from various geographical locations in a short period. Such a solution offers redundancy and increases the reliability of data by ensuring multiple human driver behaviors are represented during autonomous vehicle direct perception training. The navigation sensors (GPS/IMU) within the phone are less accurate compared to more expensive survey-grade navigation systems. However, for ubiquitous dataset collection, a much cheaper solution is needed and we present our system setup as an efficient and affordable alternative.

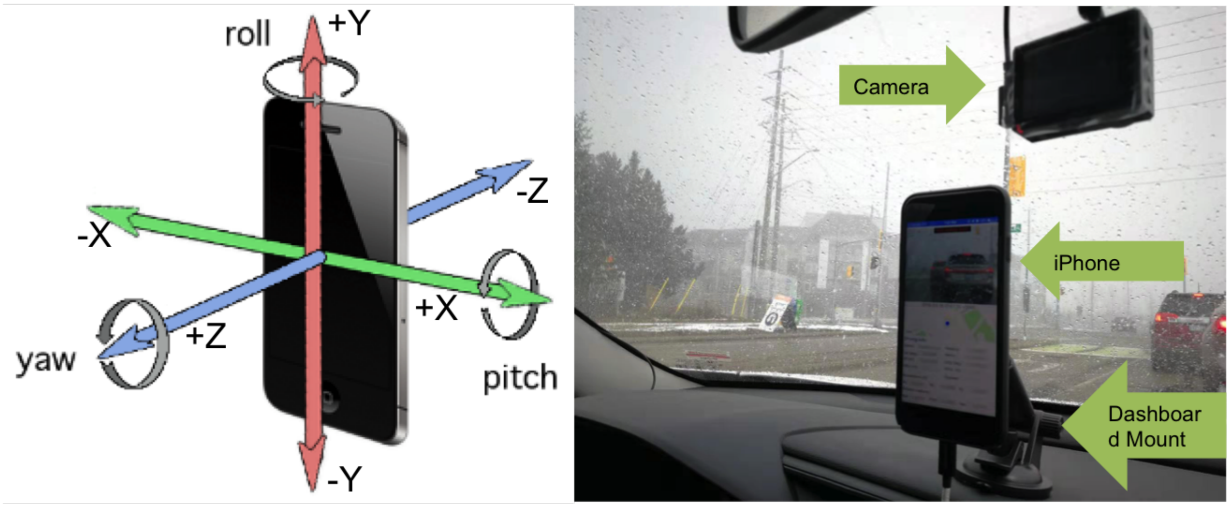


Figure 3.4: The phone is mounted with it's Z-axis parallel to the vehicle's X-axis, ego vehicle forwarding direction is the same as the $-Z$ direction in iPhone coordinate system. (a) iPhone coordinate reference system (the reference Figure courtesy of nomtek [3]). (b) The phone and dash camera set up in the ego vehicle.

Chapter 4

Static Affordance Learning

In recent years, autonomous driving technology has become closer to fully being realized. There are many driving forces to this realization, key among them is the advance in perception techniques such as CNN(Convolutional Neural Networks). The perception techniques allow an autonomous driving vehicle to understand the driving environment, which is one of the most important steps for vehicle path planning and control. In many applications [45], the autonomous vehicle must be equipped with a ubiquitous and robust state-of-the-art vision-based system for it to be able to sense and understand different driving scenes.

Static affordance learning involves learning to identify and locate most invariant features in the immediate driving environment. Some of these features include driving space, intersections, number of lanes and whether a road is a one way or both way street. Identifying such features do not necessarily require classifying the features in the environment. This is informed by the realization that most driving behaviors are influenced by simple rules concerning features in the environment. For instance, it shouldn't matter whether there is a tall wall, trees or parked vehicles on the side of the road. Since they all infer that there is an obstacle and hence a driver must avoid the obstacles by realizing and following the driving space. In case of the number of lanes, such information only informs the driver about the type of environment. Knowing the number of lanes for a particular road can inform a driver about the skill level required and the expected traffic flow. However, to make the successful maneuver through the traffic, one must be able to determine the relation about the ego vehicles to other participants in the road. Consequently, although it would be simplistic to think that learning static affordances would be enough to successfully drive, In this chapter, we show that static affordance learning is an important layer required for complete scene understanding.

There are several challenges in the direct perception approach. Since the low-level control is decided based on a given set of road attributes, the affordances to be learned must be pre-defined by humans. Selecting the suitable affordances usually requires feature engineering and driving scene-based analysis [71]. After deciding on the coarse road inference layout, we need to collect and label road data to develop a relatively good and robust model. Ari Seff et al. proposed a method that leverage the google street view images and OpenStreetMap (OSM) [37] for automatic-labeling and model training [68]. In this chapter, we follow the same line with [68] for automatic labeling procedure to collect training and testing data. Furthermore, we trained a Convolutional Neural Network (CNN) to detect static traffic scene affordances from a single street view image. We have tested the effectiveness of the method and accuracy of our model in experiments. The key contributions discussed in this chapter are: (1) *Efficient CNN Training Model*: Instead of using pre-trained AlexNet [48] CNN model on Places database [91] to get good weight as [68], we created a customized CNN architecture based on VGG11 and AlexNet. Using the non-initialized model and training on third the number of training images used, we were able to obtain results comparable to [68]. (2) *Validation on Automatic labeling*: We collect data near the Waterloo area in Canada while Ari Seff's data are collected in San Francisco, Bay area. We verified their automatic labeling methodology in a different geographical location. Further, in addition to testing our model on the San Francisco GSV images, we examined our network on KITTI [32] tracking dataset which is collected in Europe to demonstrate the generalization ability for our model. (3) *Refining the affordances by driving scene*: We refine the definition of selected road attributes from [68]. The affordances set may change according to different driving scenarios.

The rest of the chapter is arranged as follows: Section 4.1 outlines the data collection, affordance selection and auto-labeling methodology. Section 4.2, demonstrates the network design and training methodology compared with recent research works. Experimental results and discussion are shown in Section 4.3, along with the conclusion in Section 4.4.

4.1 Dataset and Labeling

As we have discussed in the previous section, a deep network was used to train data and determine affordances such as host vehicle to road relative orientation, number of lanes and driveable space. However, this requires a huge number of labeled images to be able to train a reliable model. There are several real-world street scenes labeled data sources such as KITTI [32] and synthetic data (from games and movies) such as Virtual KITTI [28] and FlyingThings3D [52]. For tasks such as determining bike lanes, wrong-way vs. right-way, the available data in [28, 32, 52] and most other open-source autonomous driving benchmark datasets [85] may not be sufficient or labeled for static affordance learning tasks. Consequently, as proposed in [68], we take advantage of huge free and open-source imagery and corresponding attributes repository available on GSV and OSM, to train a ConvNet model to predict the road attributes. Figure 4.1 left, shows a standard OSM map covering an area over the University of Waterloo's ring road. While to the right, the Figure shows the same area highlighting the high density of map layers and attributes available in the OSM map database.

4.1.1 Data Collection

OpenStreetMap [37] is a community-driven and local knowledge-based open data platform. The contributed data is tied together using location information in a World Geodetic Coordinate System (WGS84). Similarly, Google has a huge deposit of street view imagery with each panorama encoded with vehicle true heading at the time of image capture and location information in the WGS84 coordinate system. Using location neighborhood constraint, it becomes possible to associate each panorama from GSV with nearby road attributes such as the number of lanes or if an intersection is likely in view [37]. The accuracy of feature association is directly affected by the location accuracy in both GSV and OSM and whether information in both sources was updated in the same time frame. As will be highlighted later, we found some mislabeled affordances due to time latency and unresolved location differences especially at bridges or close road networks where a small location deviation

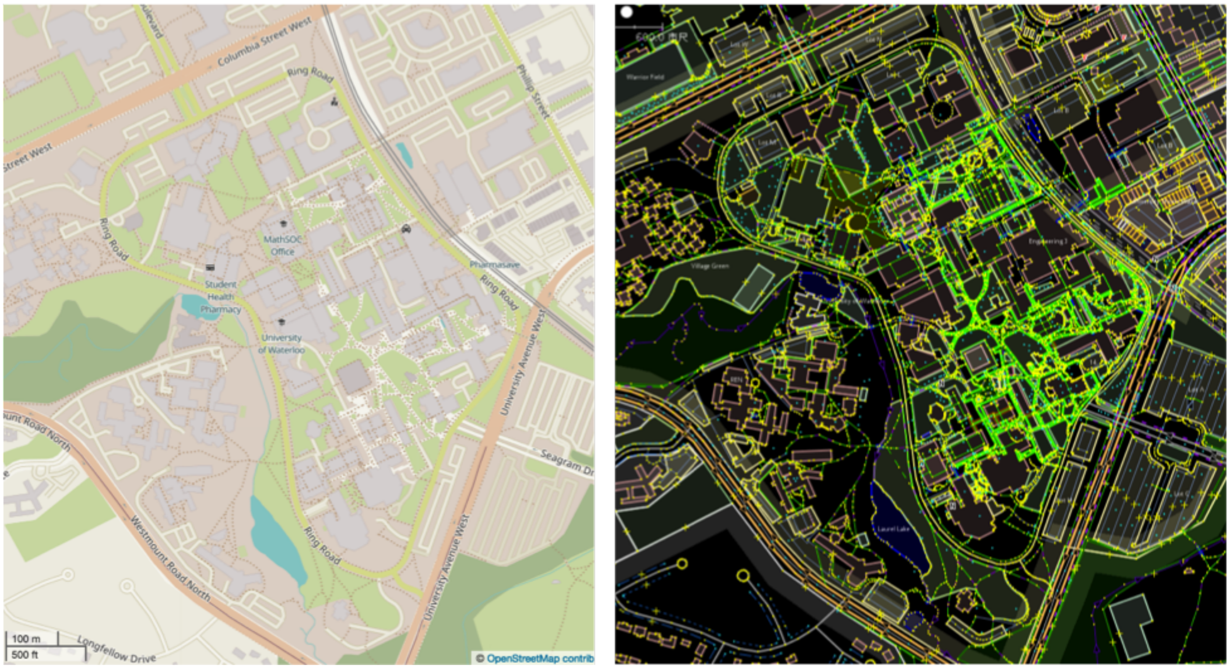


Figure 4.1: A standard OSM map covering area near the University of Waterloo in Ontario Canada is shown in (a). The same area highlighting the high density of attributes available in OSM such as road polylines, parcels near the road and building polygons are shown in (b). Each node in the map contains coordinate information that can be used to associate it with other location-based features from sources such as Google Street View panoramas.

would associate features of one road to an image showing a different road.

Google Street View Panoramas

We have downloaded over one hundred thousand panorama images starting with a seed panorama image ID at the University of Waterloo (shown in Figure 4.2). These panoramas were then cropped and warped into 227 × 227 × 3 sized images with a field of view (FOV) of 100 degrees. The image size and FOV were kept similar to what [68] used after finding them sufficient for driving scene view. Each image is encoded with coordinates in the

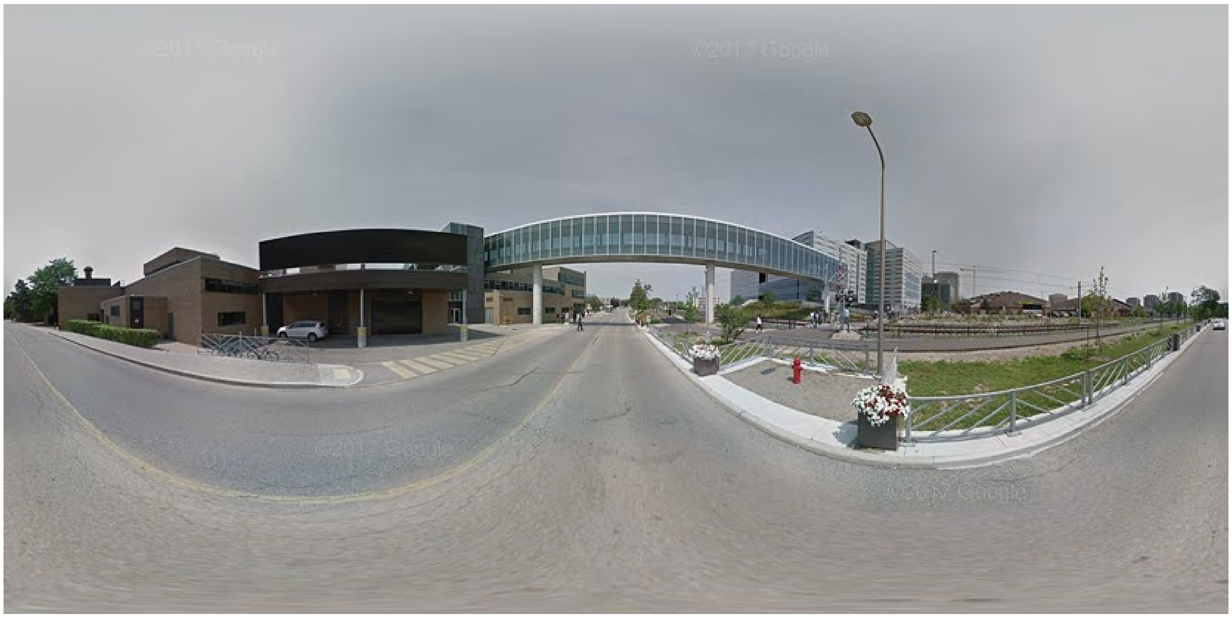


Figure 4.2: A seed panorama used as a starting point for downloading GSV panoramas. The pedestrian bridge in view connects Engineering buildings 3 and 5 (E3 to E5) at University of Waterloo.

WGS84 reference system. This is later used to query and overlay with data from OSM.

OpenStreetMaps Vectorized Data

OSM [37] is a vectorized map with attributes contributed by volunteers. Attributes include poly-lines such as those defining extents of road networks, bike lanes, and traffic markings. It also includes point features such as stop signs, traffic lights, and speed signs 4.1. Its data availability may be lacking in rural areas or small towns since volunteers tend to contribute to maps around where they reside.

4.1.2 Affordance Labeling

Here we discuss the affordance set selection. It is still an open question nowadays for the optimized road attributes selection for the driving context understanding. Chen et al. proposed 13 affordance indicators in [17] for multi-lane tracks in TORCS. It is rather simplified since there is neither intersection, pedestrian nor traffic light in TORCS. Authors of [64] advanced the affordance learning in the single lane urban scenario simulated in CARLA where 6 affordances were selected. Both works utilized the global information embedded in the simulation engine such as the global information for all the agents in the map and the distance for the vehicle between the road centerline. It is rather difficult for us to obtain this global information in real data, hence we choose the target road attributes based on the available OSM and GSV data. The OSM dataset and GSV panoramas are encoded with location coordinates in the WGS84 reference frame. Consequently, it was possible to query and overlap an image cropped from GSV panoramas with corresponding attributes in the OSM data. We conclude the list of automatically labeled affordances in the TABLE. 4.1.

Table 4.1: Road Attributes labeled for Kitchener-Waterloo region GSV images

Labeled Affordances	**Data Type**	**Range**
Heading-Angle	Continuous	$[-\pi, \pi]$
Driveable-Heading	Boolean	{True, False}
Intersection-Ahead	Boolean	{True, False}
Distance-to-Intersection	Continuous	[0, 30] m
Number-of-lanes	Discrete	{1, 2, 3}
Wrong-Way	Boolean	{True, False}
Bike-Lane	Boolean	{True, False}

Lane Following

We label the affordances *Heading-Angle* to represent the current ego vehicle heading angle corresponding to the driving lane. This is an important attribute for predicting steering wheel input during lane following. We further extend the angle prediction to a classification problem to compare human capability to identify the heading angle from a single image. The detailed comparison will be discussed in Section 4.3. Some examples of our model prediction and labels are demonstrated in Figure 4.3. The labeled vehicle heading angles are calculated based on OSM lane attributes and GSV panorama applied rotations.

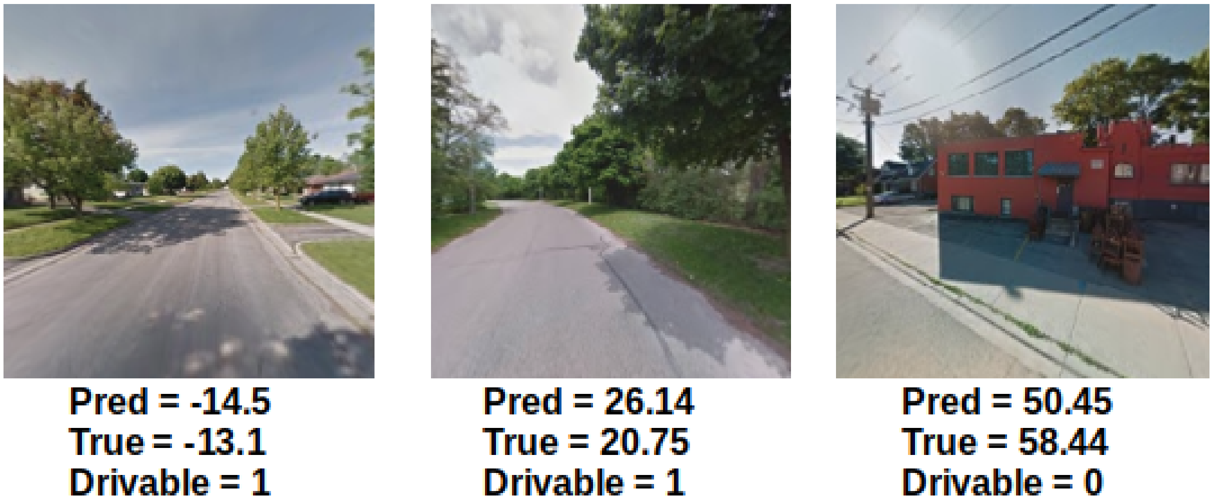


Figure 4.3: Vehicle heading angle (°) prediction result and drivable classification.

Intersection Handling

Intersections are some of the most common scenarios in the urban driving setting where a human driver needs to decide the high-level command to follow the planned driving path.

We denote the the following two road attributes for the intersection handling: *Intersection-Ahead* and *Distance-to-Intersection*. The GPS coordinates representing the camera reference point are extracted from each GSV panorama and then used to query intersections from OSM appearing within 30 meters and in the direction of travel. If an intersection is found, the image will be labeled with 1 indicating an intersection ahead. otherwise a label of 0 is assigned (Figure 4.4). For *Distance-to-Intersection*, we query intersections within 100 meters of the camera reference point and then use coordinate inverse to compute the distances. This is similar to the parameters specified in [68]. It is an easier task to identify or measure the distance between intersection when approaching one, as we can see from the top three images in Figure 4.4.

However, the estimation error of our model grows when predicting a view at the intersection (see the bottom three images in Figure 4.4). The visual inputs at intersections usually are not as structured as general road segments. The open view of an unstructured terrain confuses CNN based model since only one shot of the image is given. We believe the prediction results can be improved by using memory-based models such as LSTM [34] that are capable of capturing temporal information.

Multi-lane Handling

It is rather important for autonomous driving to first identify multi-lane driving context especially in urban or highway driving environments before performing path planning and driving maneuver control. The attribute *Number-of-Lanes* identifies the number of lanes in the current driving road scene. It is addressed in [68] where they only include one-way

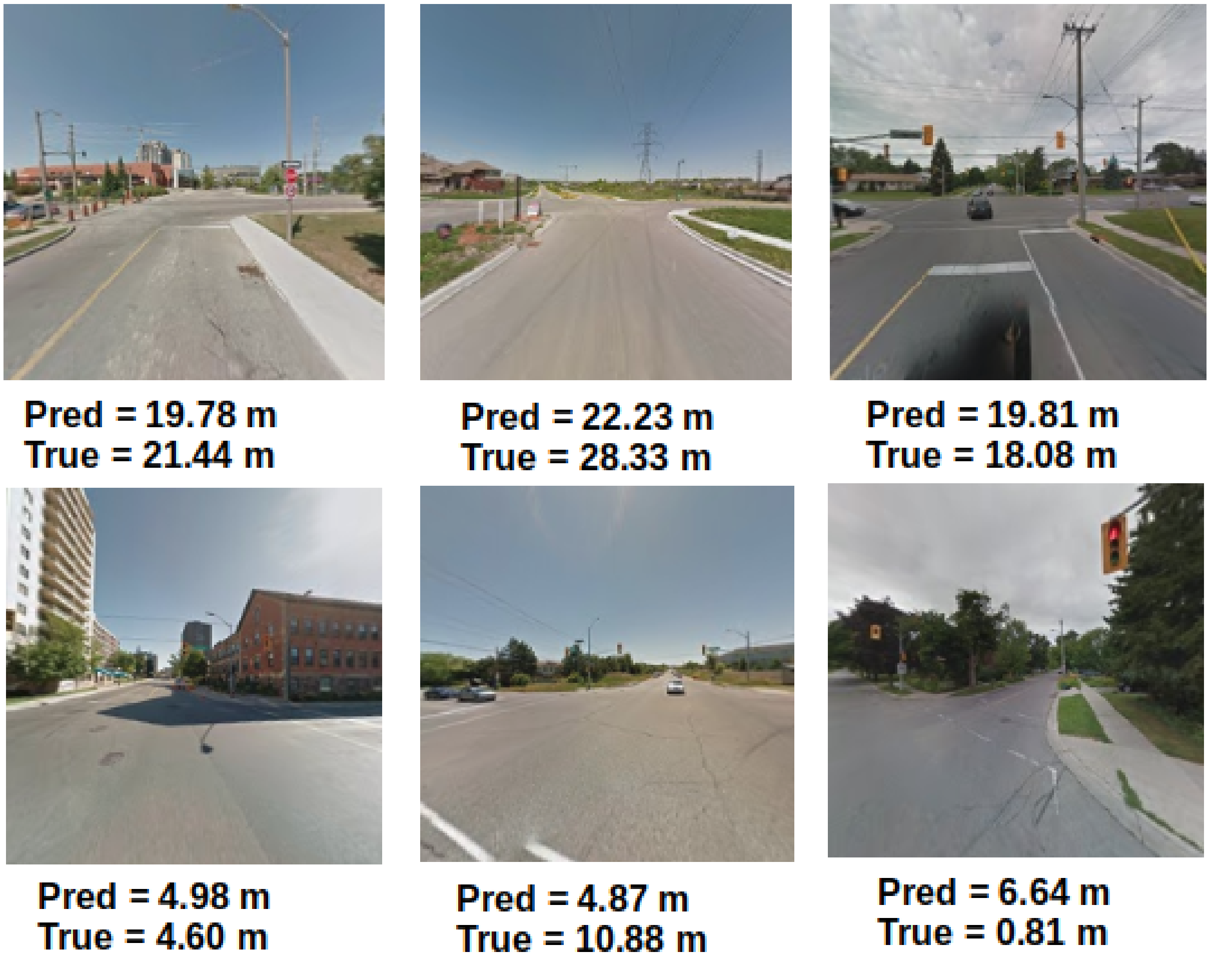
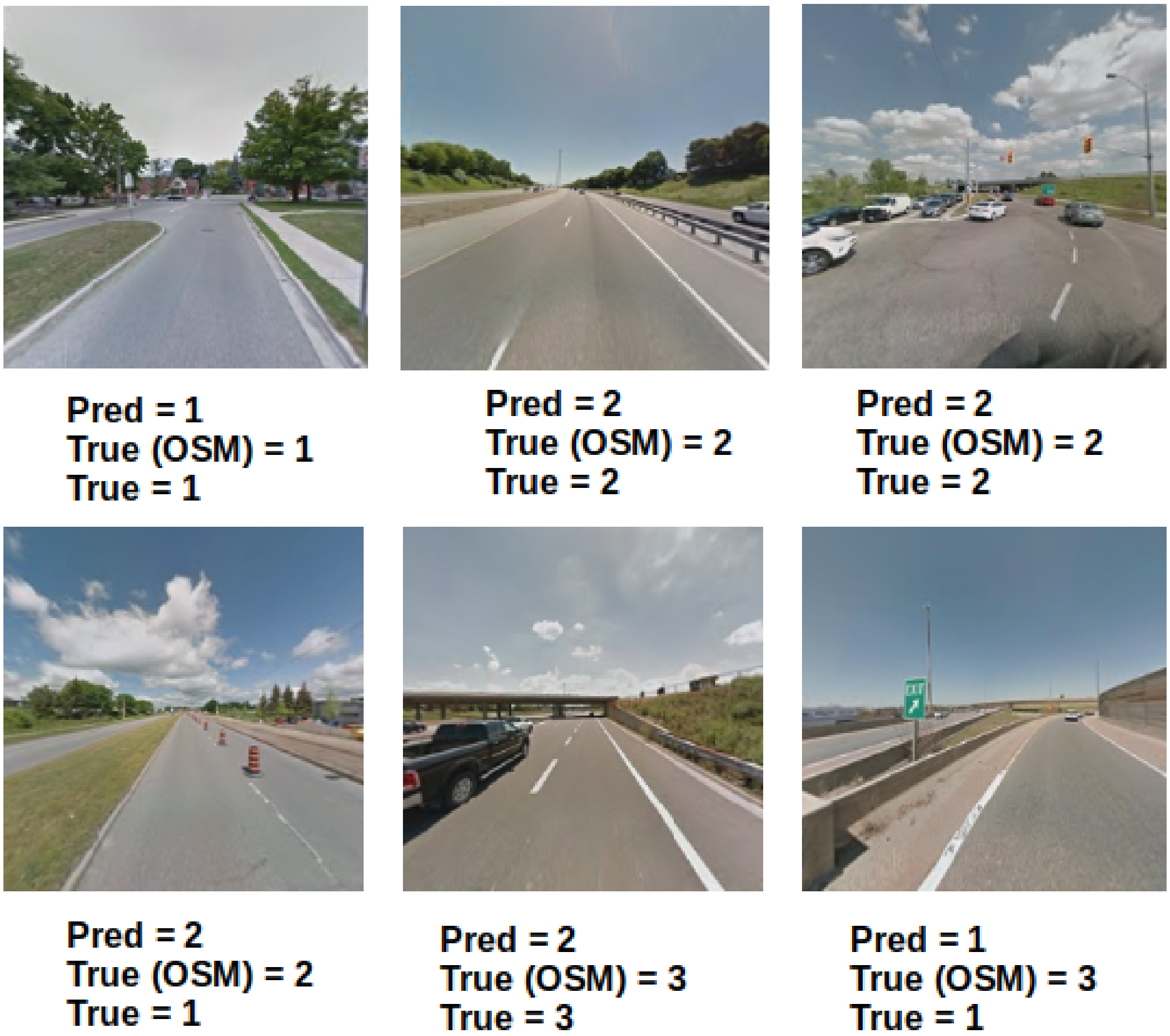


Figure 4.4: Comparison between prediction and true labels on distance to intersection. The 'true' distance label is calculated by measuring the distance between the GSV referring point and the center of the intersection in OSM.

roads in training data due to the inconsistency for two-way roads when considering the driving direction. In our work, we further included the images of two-way roads in our training set. We find that our model prediction was consistent with the labels for the most part, except when there was occlusion, lanes were not visible or the label was incorrect. In Figure 4.5, the top three images demonstrated the effectiveness of our model prediction. Despite the curved road shown in the top right image, the model was able to generalize well and made a correct prediction. Yet the task for predicting the number of lanes from a single shot of image input is still challenging due to the lack of clear lane markings in some cases or other vehicles on the road obstructing the camera view. However, this can be remedied by aggregating predictions over a certain time interval such as 10 seconds.

Unfortunately, the dynamic change of the road segments and obstruction of the view may result in false predictions. We list three typical false predictions at the bottom of Figure 4.5. The road constructions or other dynamic changes may result in an inconsistency between the expected truth of the driving context and the static road attributes labels. The static OSM data cannot adapt to recognize the traffic cones as demonstrated in the bottom left of Figure 4.5. Furthermore, the GPS location accuracy may lead to mislabels especially near intersections or highway ramps (bottom right in Figure 4.5).

The road attribute *Bike-Lane* is true if there exists a bike lane based on the given panorama. Our trained model to predict bike lanes performed 3% worse than Seff et. Al's in [68]. However, It should be pointed out that the validation accuracy was affected by the mislabeled images. As explained in the previous results, in some cases, the models made correct predictions despite incorrect labels. For bike lanes, this is still the case. As can be seen in Figure 4.6 on the left image, the model predicted the road to have no bike lanes



Pred = 1
True (OSM) = 1
True = 1

Pred = 2
True (OSM) = 2
True = 2

Pred = 2
True (OSM) = 2
True = 2

Pred = 2
True (OSM) = 2
True = 1

Pred = 2
True (OSM) = 3
True = 3

Pred = 1
True (OSM) = 3
True = 1

Figure 4.5: Multi-lane prediction using our model trained on labels provided by OSM. In some cases when there is a dynamic change (construction, road change etc), incorrect labels can occur (Bottom left & right). The narrow view and occlusions by dynamic objects (Bottom middle) may also result in false prediction.

and this is correct from visual inspection. However, the image label indicated that there is a bike lane. It is likely that the previous views of the road had bike lanes but ended before the intersection. Bike lane may be confused with the highway emergency lanes due to the CNN model only taking a single shot image as input. One false prediction example is given in Figure 4.6 on the right where the model may treat rural road with a bike lane as the highway ramp or emergency lane.

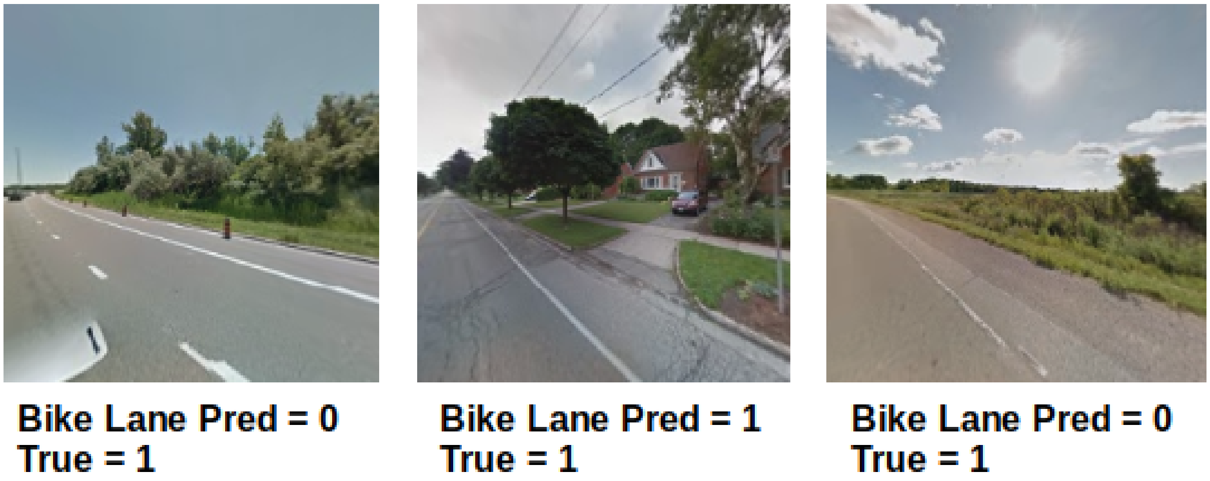


Figure 4.6: Bike lane prediction using our trained CNN model.

In driving, humans can easily tell whether they are driving on the right side of a two-way street. This is a very important rule of driving and driving in a wrong way can result in a catastrophic head-on collision. Hence, it is of the essence for an autonomous vehicle to be able to recognize the right side of driving. Consequently, *Wrong-Way* classification is based on such driving rules that you must drive on the right side of the road. By carefully examining the left and middle images in Figure 4.7, one can verify that indeed the model makes correct predictions. For the image on the left, it is correctly predicted to be the

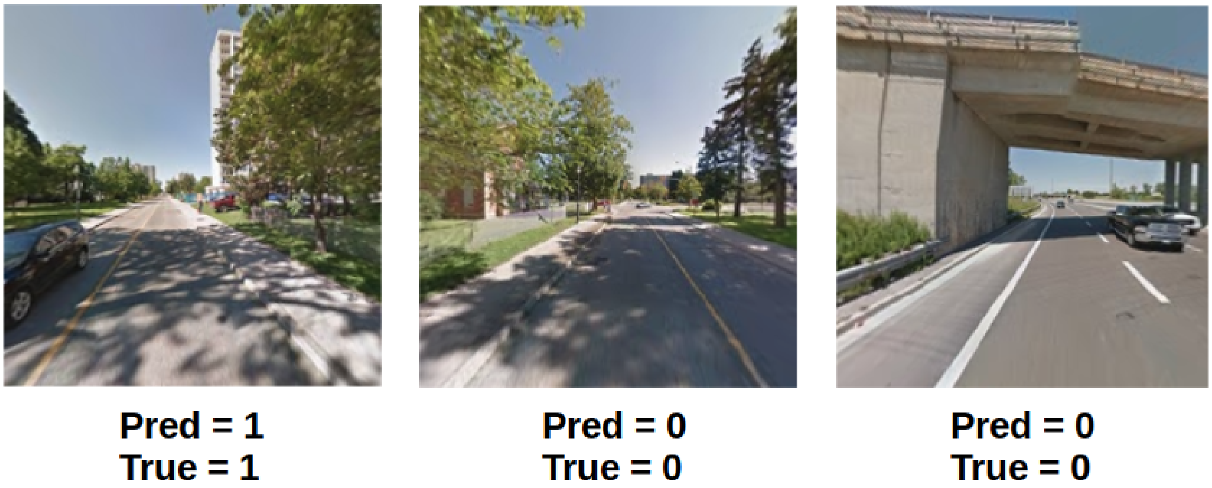


Figure 4.7: Right or wrong way classification using our trained CNN model. The right way corresponds to label 1 and wrong way is labeled as 0.

right-way. This is informed by the driver's view largely being on the right side of the road. The middle image is classified as a wrong way of driving, which is correct since the driver view mostly falls on the left side of the road. Image to the right of Figure 4.7 is less ambiguous to classify and correctly predicted as the wrong way.

4.2 Model Design and Training

Convolutional Neural Networks have been widely used in computer vision since AlexNet [48]. We employ existing methods to configure our CNN network for training and testing of our affordance learning approach. In this subsection, we describe the CNN network and hyper-parameters used to guarantee best results. As shown in Figure 4.8, our network comprised of five Convolution layers and three fully connected (FC) layers each with 4096 channels. We used a 3×3 receptive field for each convolution layer as was found effective in [72]. It produced better validation accuracy for all trained affordances. For all convolution and fully connected dense layers, we used rectified linear (ReLu) as the activation function. We also applied padding and max-pooling to preserve input size and spatial resolution, respectively, through convolution layers. We employed batch normalization [11], after the first two convolution layers and each fully connected layer. This increased the robustness of the weights and reduced overfitting while also preserving the learned features.

The output layer structure depends on whether the model is for regression or binary classification. For regression, we used an output layer with one kernel and no activation function, i.e. the outputs were not scaled into probability output. The model was compiled using RMSprop [77] optimizer with a learning rate of 0.0001 and mean squared error (MSE) as the loss function. The accuracy of our regression model was reported in mean absolute error (MAE). The output layer for a binary classification model had one kernel and used sigmoid as the activation function to scale the predictions into values between 0 and 1. Similar to regression, the compiling was done using an RMSprop [77] optimizer with a learning rate of 0.0001.

Our model was built in Keras [19] running on top of TensorFlow [1] framework. As shown in Figure 4.8, the first convolution layer input accepts an RGB image of size $227 \times 227 \times 3$ and passes it through 96 filters of size 3×3 with ReLu as activation function, strides of 1 pixel and padding set to 'same' i.e., it outputs same dimensions as input. This is followed by a scaled and centered batch normalization [11] layer. A 3×3 max pooling layer with strides of two pixels and padding set to 'valid' (no padding), comes next.

[1] https://www.tensorflow.org/

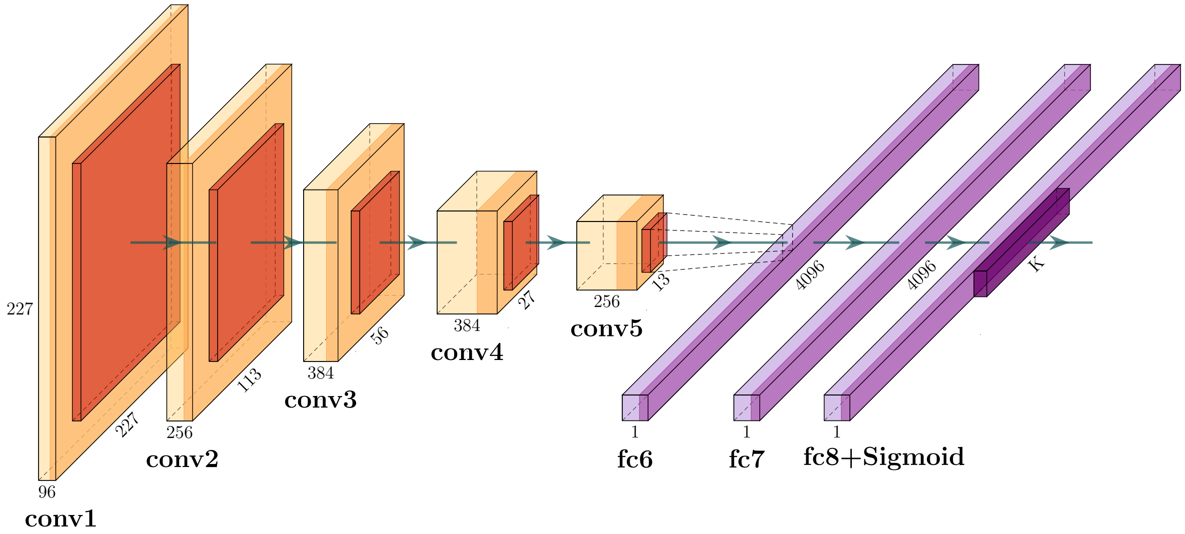


Figure 4.8: The architecture of the proposed CNN. The input is a warped and cropped GSV panorama and the output layer consists of selected features and affordance indicators. Note that we perform batch normalization after convolution layers 1, 2 and each fully connected (FC) layers to reduce over-fitting.

The second convolution layer has 256 filters of size 3 × 3. The activation function, stride, padding, and regularizer are set similar to the first convolution layer. The batch normalization and max-pooling layers follow (with similar set up as previous layers). The 3rd and 4th convolution layers have 384 filters of size 3 × 3 and are separated by a max-pooling layer. Another max-pooling layer is inserted before the 5th convolution layer with 256 filters of size 3x3 (all convolution layers maintain a similar structure to the first layer. they only differ in the number of filters). A flattening layer is implemented before the first fully connected layer. All the fully connected layers have 4096 neurons with ReLu as the activation function and L-2 norm (0.0001) regularizer. Each of them is separated by a batch normalization layer. Before training, the images were normalized by changing the pixel values to float and diving with 255. random images were augmented by applying a rotation of 22°, width and height shift of 0.2, shear of 0.2 and zoomed by a factor of 0.2. The images were trained in batches with a batch size of 32 and 50 epochs. The steps per epoch for both training and validation depended on the total number of images as in Eq. (4.1). Images and corresponding labels were also randomly shuffled during the training phase.

$$StepsPerEpoch = \frac{Number\ of\ Images}{Batch\ Size} \tag{4.1}$$

In [68], their model was pre-trained on Places Database and still it took about 50K iterations to obtain their results. In comparison, our model was trained on a random initialization without pre-trained weights. We report superior results after just 10k iterations. The detailed model performance comparison and cross-validation are given in section 4.3.

4.3 Results and Discussion

In this section, we present the quantitative evaluation result of the proposed model. We also discuss the improvement of our model and current data collection pipeline for autonomous driving.

4.3.1 Accuracy Evaluation

We validate our CNN models performance across three different geographical regions, namely data collected from Waterloo (abbreviated as W), data used in [68] collected in San Francisco, Bay Area (abbreviated as SF) and KITTI tracking data collected from Europe. We also provide a comparison between human baseline and model prediction on classification tasks.

Our CNN models vs. Human

We asked five human volunteers to label 1000 images for each affordance. We evaluate our models on the same images and compare results which are presented in Figure 4.9. We focused on classification tasks as we found it difficult for humans to meaningfully measure angles or distances from low-quality images.

Consequently, we did not consider distance to an intersection and heading angles were deduced to binary classification by asking humans to predict whether the image showed a negative rotation (left rotation with respect to the road) or positive rotation (right rotation with respect to the road). Each human volunteer was first shown ten example images and corresponding labels for each affordance under consideration. This was done to train the human volunteers by highlighting the image to affordance association in the context of driving. We then let each volunteer label provided images per affordance without access to OSM derived true labels. Consequently, for each affordance, we generated a single set of human labels by combining five individual labels using a consensus model [39]. As evident in Figure 4.9, our CNN model predictions, and human labels were within ±5.8% of each

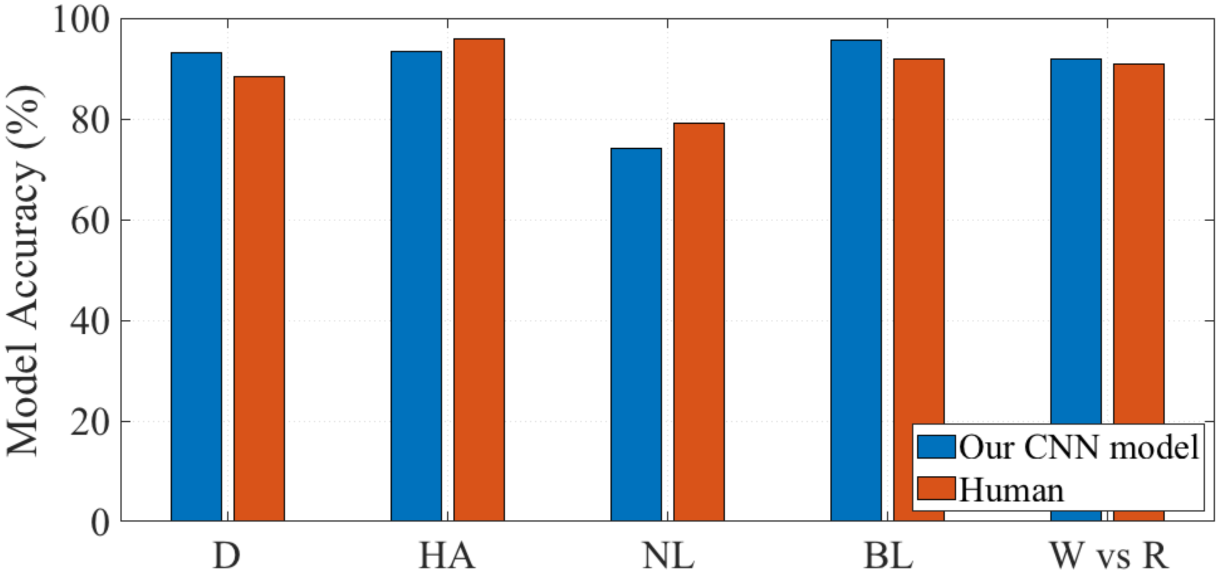


Figure 4.9: Comparison between human baseline and our trained CNN model on the classification prediction accuracy (higher is better). The tasks investigated here are driveable (D), heading angle (HA), number of lanes (NL), bicycle lanes (BL) and right way or wrong way prediction(W vs R). Predictions are made on a single image with poor resolution and not a sequence of images. Consequently, we see that our CNN model performed better or comparable to humans mainly due to poor image resolution.

other. Our model performed better than humans for driveable space (D), bike lane (BL) and wrong-way vs. right-way (W vs R) affordances. Also, it had comparable results for the number of the lane (NL) and driving heading angle (HA) affordances.

Model Generalization Test

To find out how well our model would generalize on data collected in different geographical locations, we took advantage of GSV panoramas from San Francisco Bay Area available for download on [68] data page. We did cross-validation by comparing the prediction on San Francisco Bay area GSV images by a model trained on Waterloo dataset, prediction on Waterloo GSV images by model trained on both Waterloo and San Francisco datasets and prediction on San Francisco Bay area GSV images by Model trained on Waterloo and San Francisco datasets.

We then used the CNN models for heading angle (HA), intersection distance (ID) and number of lanes (NL) affordances that were trained on Waterloo dataset (from henceforth referred to as model set 1) to predict on San Francisco Bay Area images. The driving scenes vary greatly from one geographical location to another. We recognized that data augmentation applied to a training dataset might not be robust enough for cross-geographical driving scene inference since it is relatively hard to augment buildings and other features (such as trees, grass, and curbs) proximity to the road.

Hence, we trained new models for HA, ID, and NL using a dataset with half of the images from San Francisco and another half from Waterloo (referred to as model set 2). This model was tested on a Waterloo dataset and San Francisco dataset, independently.

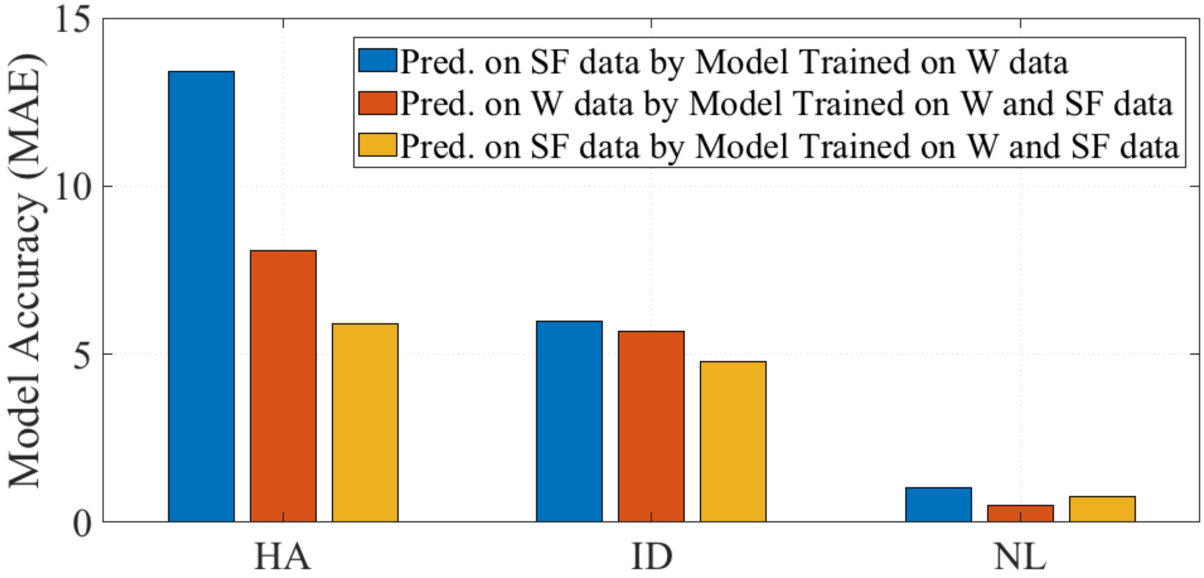


Figure 4.10: Comparison of mean absolute error between our CNN model trained and tested on datasets across different geographical regions (lower is better). The tasks investigated here are heading angle (HA), intersection distance (ID) and number of lanes (NL).

We were careful to make sure that the test images were never used during the training and validation of the models. We used same testing dataset from SF in both model set 1 and 2. This provided consistency for cross validation.

The MAE between the predictions and true labels were computed and plotted in Figure 4.10. The MAE plot shows that model set 2 is more accurate and generalizes better than model 1. The model set 2 performed best in all affordances. We should also point out that the difference in MAE for both models should be examined independently for each affordance. For instance, the number of lanes in Waterloo range from 1 to 4 lanes. Therefore an MAE of 1.04 in the number of lanes would be considered too big since it means that a model would likely be predicting the wrong number of lanes most of the time. However, an MAE of 4.8 meters for intersection distance may be tolerable given that the intersection distance in consideration, ranges from 0 to 30 meters.

In TABLE. 4.2, we compare the performance of the proposed architecture and trained models by predicting on the San Francisco testing dataset. Three sets of the model are compared. We use the model proposed in [68] trained on SF data as a baseline. The other two models are the aforementioned model 1 and 2. We compare the accuracy relative to the training data size used in HA, ID and NL affordance training.

Our models trained on both Waterloo and San Francisco datasets performed better than models in [68] for HA and NL affordances, despite only using one third of their data size. We used 4K images while [68] used over 12K images for training. Although our second model trained on just Waterloo (W) datasets while it reports the worst accuracy, it is still comparable to Seff et Al. results for all three affordances.

Moreover, the model trained using only 4K images on the combined data outperforms the other models in most of the regression tasks. Given the results in TABLE. 4.2 it is best to use data collected in various geographical locations and different conditions to train a perception model that could generalize well.

	Model in [68]	Ours on W	Ours on W & SF
Train Samples	>12K	~6K	~4K
ID (MAE)	4.3	6.01	4.77
HA (MAE)	9.2	13.4	5.89
NL (MAE)	0.9	1.04	0.76

Table 4.2: The comparison across models proposed in [68] trained based on SF data, our architecture trained on W data as well as the same architecture trained on combined SF and W data. All the models compared here are tested on same SF data. Our model trained on W and SF data reports best results.

Driving Heading Angle prediction on KITTI dataset

To demonstrate that our model had potential application in lane following and reliable heading prediction from a single image, the model was used to predict on KITTI tracking dataset. The results are plotted in Figure 4.11 with the angular rate around Z axis for each image as included in the image metadata from the KITTI website.

The plot clearly shows a similar trend between our CNN model predicted heading angle with the reported angular rate at the time of image capture. Note that the size of the KITTI [33] tracking images is 1242×375 in width and height, while, our trained model takes $227 \times 227 \times 3$ input. Hence, we had to resize the KITTI tracking images. It is impossible to crop the KITTI images to fit the input size of our model without cutting out any road features. Similarly, we could not train the model with input size of 1242×375

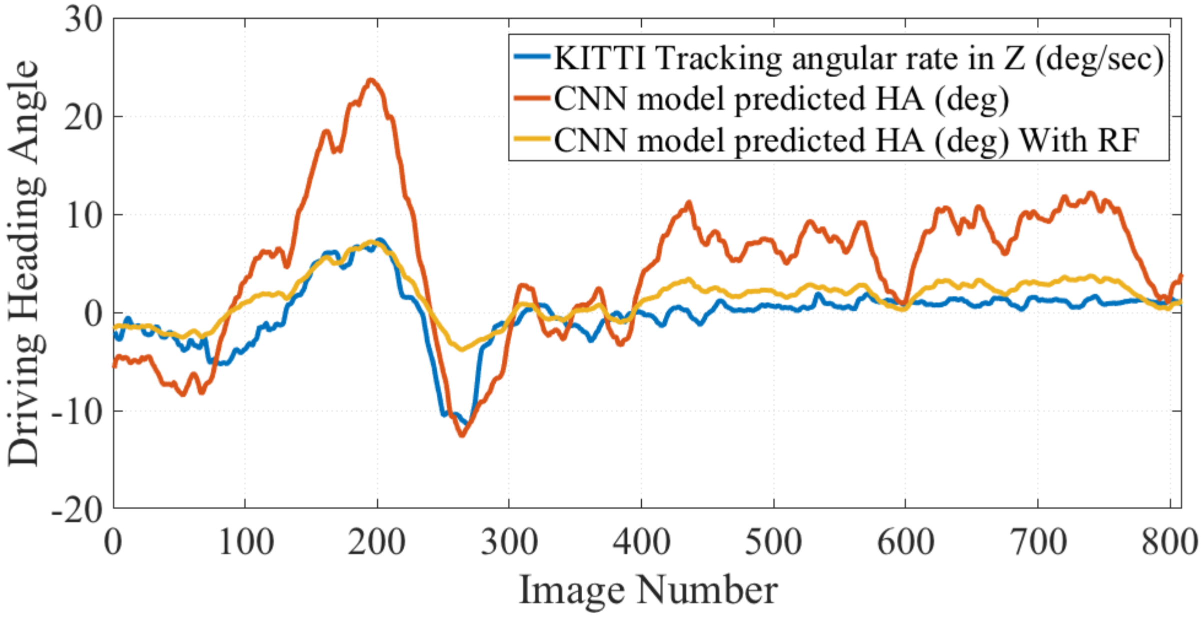


Figure 4.11: Comparison on regression task of vehicle heading angle prediction. We feed the resized KITTI images (collected in Europe) into our CNN model trained on data collected in the Waterloo area in Canada. The blue line corresponds to the ground truth measurement from KITTI, and the red line corresponds to the raw prediction result from our CNN model without considering the distortion of the input image. The CNN prediction with applying the resizing factor is plotted in yellow.

as the GSV panoramas [68] had a resolution of 832 × 416. Unfortunately, we lose spatial resolution and introduces distortion by shrinking the image horizontally. We can observe the magnitude difference between the red line (raw CNN model prediction) and the blue line (KITTI ground truth). To demonstrate this issue, we applied the resizing factor (RF) and plotted the new heading angle magnitudes. As evident in Figure 4.11, the heading angles with resizing factors applied are very close to the KITTI tracking changing in angles at the image capture. It would be good to verify observations presented in Figure 4.11 using other datasets. Unfortunately, at the time of conducting this research, there was no other public datasets (except KITTI tracking) that attached precise heading measurements on sequence of images.

4.3.2 Automatic Labeling for driving datasets

The automatic labeling for driving data by leveraging existing OSM and GSV data is a complementing way or cheap substitution of generalizing training data workflow for autonomous driving. The growing use of OSM data for training may contribute to more accurate static labeling in return. Furthermore, these road attributes can be used to corroborate and reduce over-reliance on expensive high-definition maps needed in complete driving scene understanding.

It is rather important to increase the accuracy of automatic labeling as we have demonstrated in previous sections. The correctness of automatic labeling was defined by several factors. First, positioning in both GSV and OSM data carries a degree of error. GSV panoramas are collected using Google Street cars equipped with the navigation system (GPS/INS) whose accuracy depends on the environment [41]. The positioning accuracy in these areas can range in meters. When GPS and IMU are combined to create a fused solution, a centimeter-level accuracy (after post-processing or using real-time kinematics) can be achieved. However, this is true in open sky areas as GPS signal is easily obstructed in areas with a lot of buildings or trees causing deterioration of accuracy to decimetre-level accuracy even with a high-end IMU [92]. These positioning challenges are inherited by the collected panoramas and contribute to mismatching with OSM data. Another factor is that OSM data is contributed by volunteers and hence integrity of its data varies and may not be up-to-date. This is highlighted in Figure 4.6 where the left image is labeled as having a bike lane but in reality, it is the road segment before the current location. Moreover, the OSM road attributes data and GSV images are static which means that they cannot be represented well when there is a dynamic change of the road segments such as road construction, change of weather, etc. The road construction may affect the correctness of the labeling more as shown in the bottom left image of Figure 4.5.

The GSV panoramas are collected by google streetcars mostly on sunny days with a clear view with almost no variation on the weather. The driving data from different weather such as raining and snowing are necessary to train a robust perception model that could generalize well. The prediction error may also be inherited from the downside of the CNN architecture where only a single shot of front view image is used as input.

4.4 Conclusion

In this chapter we proposed an efficient CNN model for driving affordances learning by leveraging online static databases. We annotate training data using Google Street View imagery near the University of Waterloo and queried near static road features from the Open Street Map.

We examined our trained model based on different dataset across geographical regions. The quantitative results indicated the effectiveness of our CNN model for affordance prediction across driving data collected in Waterloo area in Canada, California area in the US and Europe respectively. This chapter aimed to extend the automated pipeline approach for training static affordance learning using a robust and efficient CNN model. The trained model can infer on a driving scene image and predict static affordances such as driveable space, number of lanes, heading angle of the ego vehicle relative to the road and distance to the intersection. We also highlight realized issues in the discussed annotation pipeline. We found that some images might be mislabeled due to occlusion, error in positioning and differences in time of collection and updates for the data sourced from GSV and OSM databases.

Chapter 5

Dynamic Affordance Learning

In this chapter, we discuss efforts to design a distributed way of collecting visual driving data and under various weather conditions for dynamic affordance learning. Dynamic affordances include dynamic features and dynamic rules governing and influencing autonomous driving.

Dynamic features in the driving environment can be categorized into ego vehicle (the autonomous driving vehicle whose perception and location are being considered in the driving context) and moving obstructions to the ego vehicle. The moving obstructions may include other vehicles, cyclists, pedestrians and can include geese, antelopes, horses, and elephants depending on an environment. The random occurrence of some of the dynamic obstructions leads to a long tail problem in autonomous driving. Dynamic rules include such rules and signals that define which dynamic feature(s) has the right-of-way in a given situation during driving. The dynamic affordances learning described in this chapter can be summarized as methods of observing, establishing and incorporating the relation of the ego vehicle to other dynamic features and their response to the dynamic rules of driving.

Autonomous driving became a popular research field in recent years. The information technology and autonomous driving systems can in all be used to promote better commuting choices, provide the best route planning, improve bus scheduling and routing and finally reduce travel time and traffic congestion. A safe and robust autonomous driving system could also greatly reduce traffic accidents caused by human drivers [79]. To design a safe and robust autonomous driving system, the ability to understand the driving environment as well as the current vehicle state is essential [86]. The techniques used in environment perception varied from simple object marker detection by hand-crafted rules [44] to recent deep learning approach [62]. The final goal is essentially to have an affordable and robust system applicable under diverse environments.

One of the most frequently used autonomous driving framework among car companies is the modular pipeline approach where expensive LiDAR, high accurate GPS and 3-D high definition maps are used to reconstruct the consistent world representation of the surrounding environments [32]. The ego vehicle then takes all the information into account and make further control decisions. However, such way of perception is very expensive and raises problems in storage space and poses limits to the deployment area.

Furthermore, as mentioned in [17], the human driver only needs relatively compact driving information to make driving and control decisions. Instead of reconstructing the three-dimensional high definition map with bounding boxes of other traffic participants, a compact driving affordance set may be an efficient enabler for control decisions. Consequently, end-to-end learning [12, 59] and direct perception [17] attempt to directly map camera images to either control inputs or driving scene affordances. The end-to-end learning for autonomous driving enjoys non-expensive annotation of the training dataset, however, it is hard to interpret the control decisions. The direct perception approach proposed in [17] leverage interpret-ability by using compact annotations of driving scene affordances.

Autonomous driving systems trained in both ways are highly dependent on the distribution and label accuracy of training datasets. The data collection and annotation for neural network-based training methods resulted in several problems associated with how to collect driving data in a scalable way in a diverse environment, and how to ease the human annotation efforts. This work provides the following contributions.

1. We present an affordable, scalable driving data collection scheme with an automated labeling pipeline for the autonomous driving system as shown in Figure 5.1 to tackle the aforementioned problems. The proposed image-to-map annotation proximity algorithm (I2MAP) query Open Street Map (OSM) [37] automatically on static road labels. The customized confidence mask can be applied in the post-processing stage where the ill-labeled training data samples can be avoided. It is worth mentioning that the whole data labeling process is automated.

2. We introduce the CogData winter driving dataset where driving data under various driving scenarios and weather conditions are included. The dataset consists of about 40,000 images with more than 40 labels including driver's input, ego vehicle dynamics, and OSM road attributes. The dataset could be used in various tasks such as high-level driving scene understanding, dynamic affordance learning indirect perception

and vehicle control strategy in end-to-end learning.

3. A traffic flow prediction network is trained and evaluated. It could act as smart driver assistance and we tested it using driving scenes captured in various weather conditions including snowing and night time.

The rest of this chapter is organized as follows. In Section 5.1, we present the scalable and affordable data collection framework. The traffic flow prediction network and affordance learning based on our benchmark are introduced in Section 5.2. Finally, a conclusion is given in section 5.3.

5.1 Data Collection & Annotation Framework

In this section, we introduce a proposed cheap sensor setup and affordance annotation framework. As demonstrated in Figure 5.1, our set up include a front camera, iPhone and Panda (Gray version) OBDII Interface from comma.ai [1]. We use Honda Civic LX 2017 as our ego vehicle and static road attributes are queried from OSM and associated with images based on our proposed image-to-map proximity (I2MAP) annotation method. Figure 3.4 (right) presents a phone and camera set up in the vehicle.

The data from iPhone and vehicle sensors are time-tagged at every second (in UTC) which makes synchronization across all sensors possible. The static road attributes are queried from OSM and associated with images based on our proposed image-to-map proximity (I2MAP) method. It is possible to use raw GPS logs from Panda, to further improve

[1] https://comma.ai/shop/products/panda-obd-ii-dongle

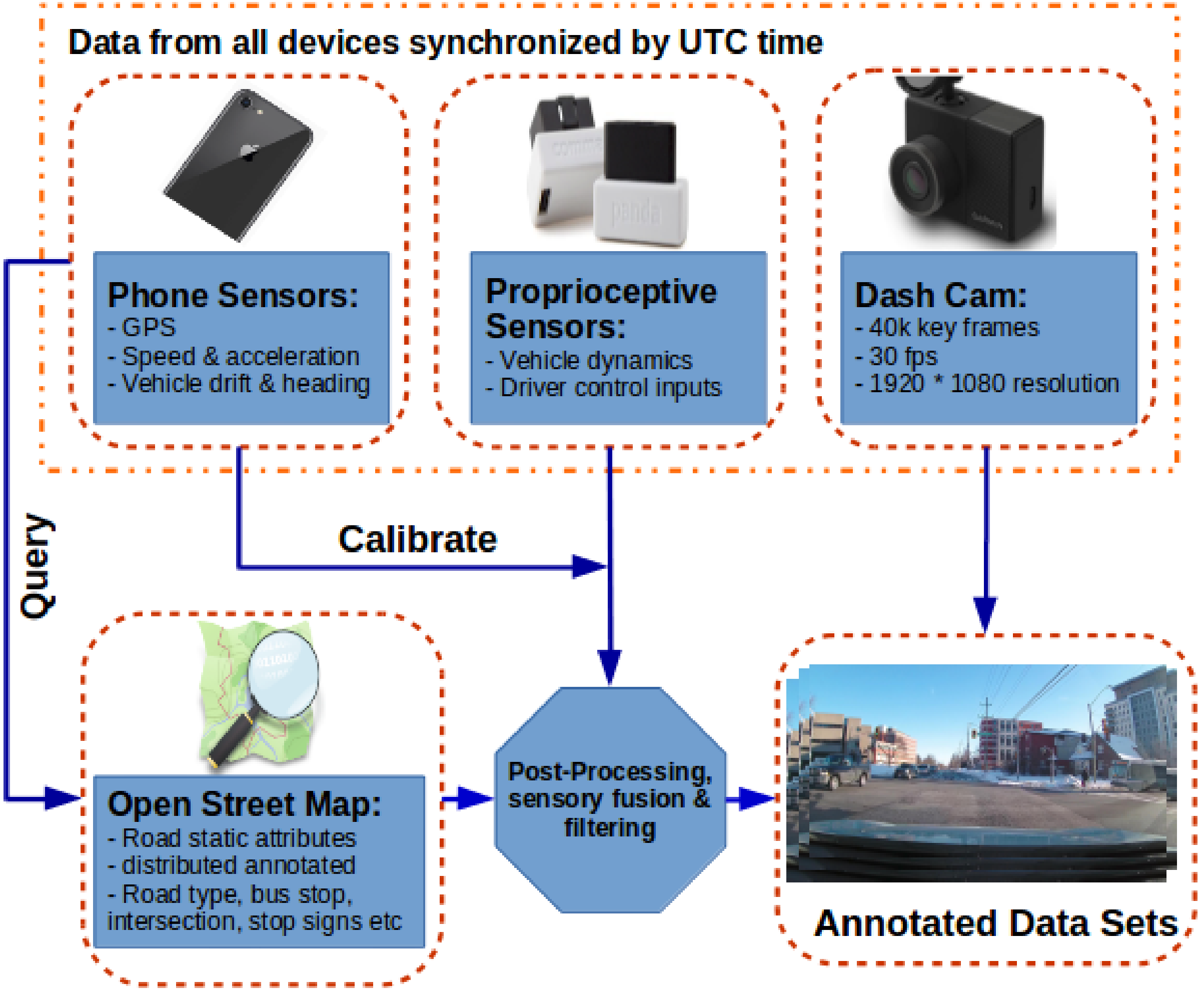


Figure 5.1: A framework demonstration of proposed driving data collection and automatic annotation pipeline.

GNSS positioning accuracy as suggested in the Laika algorithm proposed by [65]. However, we do not log raw GNSS data as the GPS data streaming format is not available at this time. To this end, we not that our setup suffers from poor positioning as the iphone GPS sensor measured within average accuracy of about 5 meters. In our automatic labeling, we filtered out any measurement that recorded positioning accuracy greater than the 5 meters.

OSM offers rich geospatial data and covers many cities and towns around the world. It is contributed to by a community of GIS (Geographic Information Systems) professionals and engineers. The database includes not only the standard 2D map but also location-based and descriptive attributes about road networks such as the location of intersection and type of intersection. We downloaded the OSM data covering Kitchener-Waterloo and its vicinity as was introduced in chapter 4. The following subsection will give more in-depth details of the sensor setup and automatic labeling and synchronization.

5.1.1 Sensor Setup

Phone Data Collector App: The camera and iPhone are mounted on Honda Civic LX 2017 dashboard (see Figure . 3.4 (right)) while the Panda OBDII interface was hooked into the vehicle CAN bus connector to read various vehicle sensors transmitted. We built an iPhone App capable of logging phone POSE (position and orientation estimates), accelerometer and gyroscope sensor readings while recording driving scenes at the same time (Details about the CogDrive Data Collector app are given in chapter 3). We set both video recording and sensor logging at 1Hz. However, the iPhone sensors such as GNSS receiver, gyroscope, accelerometer, and magnetometer have relatively low accuracy. In this regard, we implement an onboard automatic accelerometer and gyro calibration, also described in chapter 3. The calibration takes place after the phone is mounted on the vehicle dashboard and we only need to calibrate the phone in an upward portrait plane where the phone Z-axis is parallel to vehicle's X-axis (see Figure 3.4).

After the iPhone is mounted and the app launched, a user is asked to start heading warm-up (see chapter 3 for detailed calibration procedure). The phone heading is determined using magnetometer sensors which can be affected by metals. Therefore, a user first has to drive around until the heading accuracy is below 20° and then park in an area away from physical structures. The iPhone used an initial attitude reference frame [1] which assumes a device to be lying on a flat surface with a vertical Z-axis facing up while X-axis points to true north (see Figure 3.4). Therefore, the user is asked to drive slowly until the iPhone heading is matched with the reference frame. At this point, the iPhone X-axis will be pointing to true north matching the device attitude orientation. To reach better accuracy, we find the device average sensors noise by averaging the sensor reading within a one-minute time frame. The computed average noise is subtracted from corresponding measurements in real-time as the user collects data.

Garmin Dash Camera We found that the horizontal view angle for the captured iPhone videos was only about 60° Field of View (FOV) and hence not suitable for front view image collection. Instead, we use Garmin Dash Cam 45 with a 122° FOV. This camera records videos at 30FPS with a frame resolution of 1920×1080. The camera gives 3 channels (RGB) and has a night color mode setting which helps capture relatively good images at night. Each frame is tagged with time (in UTC), GPS position and movement speed. Unfortunately, this information is not logged to any file and hence can't be used as labels or for data synchronization. However, they can be used to visually check and verify synchronized labels from other sensors or sources.

Vehicle Proprioceptive Sensors Vehicles have many proprioceptive sensors such as ones capturing steering angle and throttle input accessible via vehicle CAN Bus. We can access such information using Panda OBDII Interface (Grey version). The Panda grey version has a GPS receiver and comes with a Tallysman GPS antenna. The CAN messages are decoded based on a dbc file matching our vehicle model. The vehicle CAN messages are logged and decoded in real-time. Messages of interest such as longitudinal acceleration and steering angle are captured and saved to a separate file at 1 Hz.

5.1.2 Data Synchronization

A data collection work-flow must be followed to guarantee a harmonized synchronization. The sensory data from all sources are synchronized with Coordinated Universal Time (UTC) [10]. Note that the GNSS position on iPhone is only updated once every second. This constrains us to synchronize the recorded data from various sensors at 1Hz maximum.

However, we find that while it is critical to consider higher frequencies in real-time driving scene predictions, for data collection purpose it is not necessary as there are barely any major changes in street scenes within one second. Even when driving at 100 km/h, the surrounding environment is of highway with only gradual terrain changes. Other vehicles on a highway, are also unlikely to make drastic positional changes with respect to ego vehicle in less than a second. Hence, we found that the 1Hz keyframe is sufficient to capture any driving scene happenings while minimizing the use of redundant frames in a model training.

I2MAP Algorithm: We propose an image-to-map annotation proximity algorithm (I2MAP algorithm) to overlay road feature attributes to recorded driving scenes. This is possible since the OSM also provides coordinates (in WGS84 Reference Frame) for the reported attributes. The OSM to image matching algorithm was first proposed by Seff et al [68]. However, they were using Google Street View (GSV) images which are collected using survey-grade GNSS receiver and high-end IMU. We use low-grade iPhone sensors to achieve the same task of automatic image labeling using OSM attributes.

Consequently, labels from OSM (see TABLE. 5.1) are constrained to an intersection and forward direction or straight road sections. For instance, for an image to be classified as having a bus stop, a distance and azimuth between iPhone logged coordinates corresponding to that image and the OSM coordinates for a nearby intersection are first computed. The bus stop must be within 55 meters of an intersection but 25 meters from the same intersection (towards the ego vehicle). This makes sure that bus stops are always labeled only in an image taken outside the intersection but not far from intersection. Most bus stops in the Kitchener-Waterloo area are usually found close to and either before or after an intersection. We also constrain the computed azimuth to indicate a forward driving direction. This mitigates any possibility of a bus stop found at the opposite side of the road from being considered. However, it also means that bus stops appearing after an intersection were not considered.

Even though the applied constraints resulted in fewer images being labeled (if OSM queried label was true but could not pass imposed constraints, the field is left blank), it helps to significantly reduce the number of false positives. We use the haversine formula in Eq. (5.1) to compute the distance d between two WGS84 coordinates.

$$\begin{aligned} h &= \sin^2(\frac{\phi_1 - \phi_2}{2}) + \cos\phi_1 \cos\phi_2 \sin^2(\frac{\lambda_1 - \lambda_2}{2}) \\ d &= 2R \arcsin(\sqrt{h}) \end{aligned} \tag{5.1}$$

where ϕ and λ correspond to latitude and longitude accordingly. The estimation of earth radius is denoted by R.

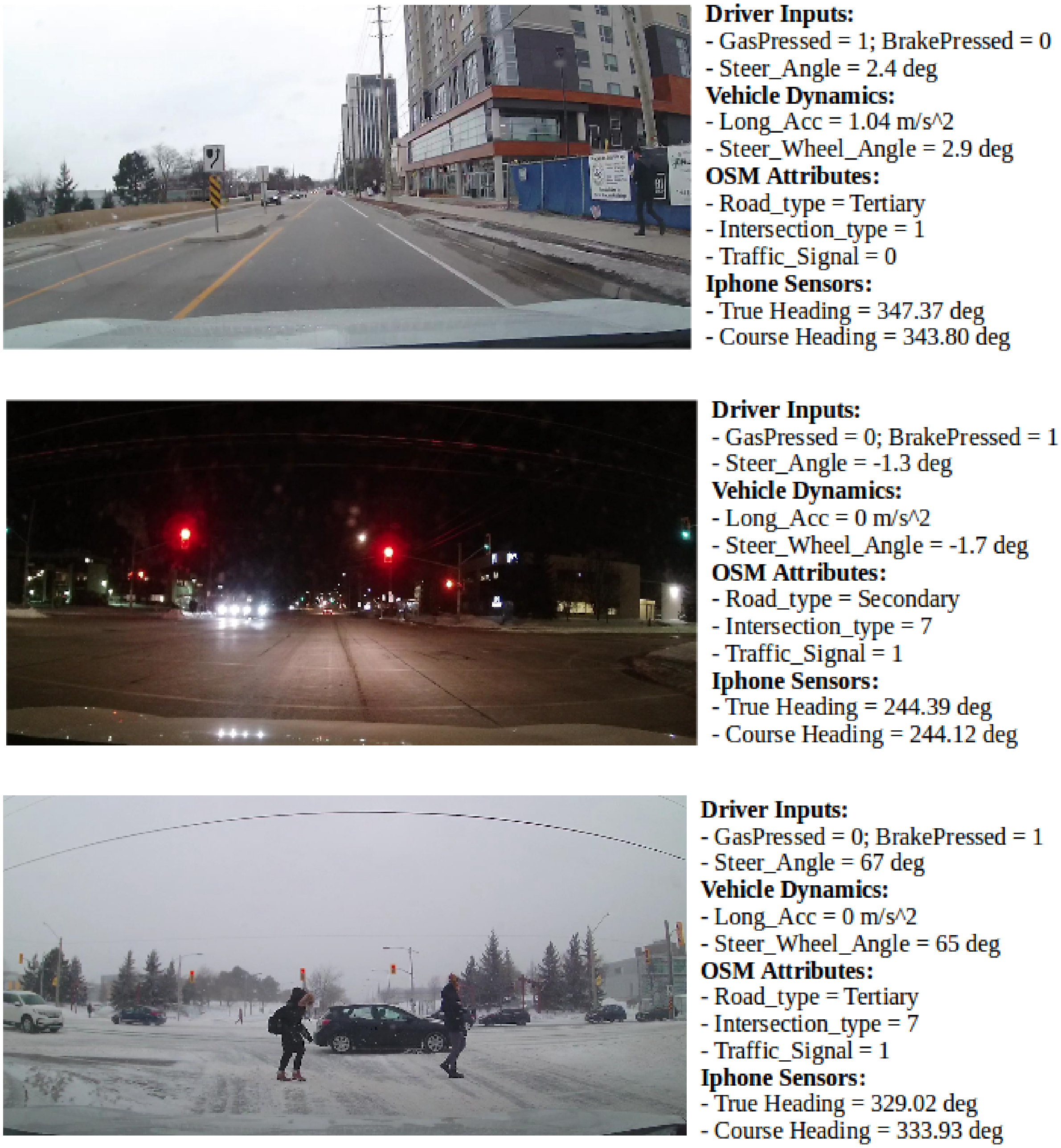
The accuracy of vehicle driving path and lane localization can further be improved by heading angle correlation, the detail result will be demonstrated in next section. Here we calculate the azimuth θ between two WGS84 coordinates by

$$\theta = \arctan \frac{\sin L}{\cos\phi_1 \tan\phi_2 - \sin\phi_1 \cos L} \tag{5.2}$$

where L denotes the positive eastward longitude.

5.1.3 Automatic Annotation

The useful labels and driving affordances for direct perception and end-to-end training are automatically calculated and attached to each front view image. Figure 5.2 provides an example of the collected front view driving images, assigned annotations with driver inputs, vehicle dynamics, and environmental affordances. The intersection types are annotated based on intersection topology in Figure 2.3.



Driver Inputs:
- GasPressed = 1; BrakePressed = 0
- Steer_Angle = 2.4 deg

Vehicle Dynamics:
- Long_Acc = 1.04 m/s^2
- Steer_Wheel_Angle = 2.9 deg

OSM Attributes:
- Road_type = Tertiary
- Intersection_type = 1
- Traffic_Signal = 0

Iphone Sensors:
- True Heading = 347.37 deg
- Course Heading = 343.80 deg

Driver Inputs:
- GasPressed = 0; BrakePressed = 1
- Steer_Angle = -1.3 deg

Vehicle Dynamics:
- Long_Acc = 0 m/s^2
- Steer_Wheel_Angle = -1.7 deg

OSM Attributes:
- Road_type = Secondary
- Intersection_type = 7
- Traffic_Signal = 1

Iphone Sensors:
- True Heading = 244.39 deg
- Course Heading = 244.12 deg

Driver Inputs:
- GasPressed = 0; BrakePressed = 1
- Steer_Angle = 67 deg

Vehicle Dynamics:
- Long_Acc = 0 m/s^2
- Steer_Wheel_Angle = 65 deg

OSM Attributes:
- Road_type = Tertiary
- Intersection_type = 7
- Traffic_Signal = 1

Iphone Sensors:
- True Heading = 329.02 deg
- Course Heading = 333.93 deg

Figure 5.2: Examples of our automatic annotation for affordance learning in (a) normal day urban driving; (b) complex intersection night scene; (c) snowy condition. The intersection type is automatically classified and annotated by the topology classification proposed in [31]

Vehicle Dynamics Schafer et Al. in [65] provide driving scene imagery with vehicle dynamics annotations such as steering angle and longitudinal acceleration focusing on driving pose estimation. However, their data is collected in summer and mostly on the highway. Also, they do not provide any road attributes. In our work, we provide vehicle dynamic information presented in Table 5.1. We find these labels to be most reliable and can be used to predict the driver's intent. The driver's input such as steering angle and longitudinal acceleration information can be used in imitation learning. The steering angle also shows a driver's intent to turn or change lane.

With careful processing of this signal, steering angles corresponding to a change of lane, merging or following a curvature road can be studied and consequently, a ConvNet model can be trained to predict lane changing, merging and lane following behavior given sequence of images as inputs. Knowing whether a vehicle's brake is pressed can associate the driving scene view with stopping action as shown in Figure 5.2 (b) and (c). Similarly, a clear to move view such as in Figure 5.2 (a) can be connected to a *gas pressed* label. Similar information can be derived from speeding information at vehicle front and rear wheels.

OSM Attributes Human drivers rely a lot on cues from the environment to be able to make critical decisions while driving. Cities and municipalities spend a lot of resources in putting up road signs and painting crossings especially in residential areas. They do this to communicate to drivers and other road participants about the driving environment and in turn, it improves safety. Consequently, autonomous vehicles must be able to understand every attribute from the environment. Luckily, using open-source maps such as OSM, it is possible to query and automatically label driving scenes captured by a camera, with static affordances from the environment. We present data that includes 12 different attributes (see Table 5.1) from the environment. Important affordances such as the location of the give way signs, road crossings, and traffic signals are attached to each image frame. These road attributes can inform a lot about the size and expected traffic on a particular road. This is true even when the lanes are not visible due to being covered with snow.

Labels From Vehicle Sensor	Labels From iPhone	Labels From OSM
Longitudinal Acceleration	GPS Coordinates	Road Type
Engine Torque & Estimate	Speed, Heading Angle & Drift	Intersection Detection*
Steer Angle & Steering Wheel Angle	Estimated Attitude: Roll, Pitch, Yaw	Intersection Type
Engine RPM, Odometer & Pedal Gas	Gyro & Accelerometer Measurements	Intersection Distance
Gas Pressed* & Brake Pressed*	Vertical, Horizontal & Heading Accuracy	Bike Lane* & One_way*
Front Left & Front Right Wheel Speed	Moving Traffic* & Snow Index	Number of Lanes
Rear Left & Rear Right Wheel Speed	Rain index & Road Condition	Bus Stop* & Stop Sign*
	Wind Speed & Visibility	Traffic Signal*
		Road Crossing* & Give Way*

Table 5.1: Each label in this table is attached to every key frame. The labels with a * mark suggest a binary type label.

5.2 CogData Analysis

5.2.1 Data Statistics

CogData is a large–scale and possibly the most challenging driving dataset publically available. It offers the most diversely labeled images for vision-based driving scene understanding. It contains more than 700 one-minute video sequences of real-world winter weather driving. It contains various challenging road conditions including snow-covered roads and freezing rain. The data was both collected during day and nighttime.

Over 42000 key-frames (sampled from original videos at 1fps) are synchronized with parallel sensor logs from iPhone and the car's information such as steering angle and throttle input. Also, static road attributes are queried from OpenStreetMap and associated with images based on image-to-map location proximity (I2MAP). Consequently, each image is tagged with static and dynamic affordances labels from over 40 classes shown in TABLE. 5.1. The data was collected in the Kitchener-Waterloo area covering over 1000 KM of the highway and residential roads. The map in Figure 5.2.1 shows the overall data coverage. Roads covered on different days are colored differently.

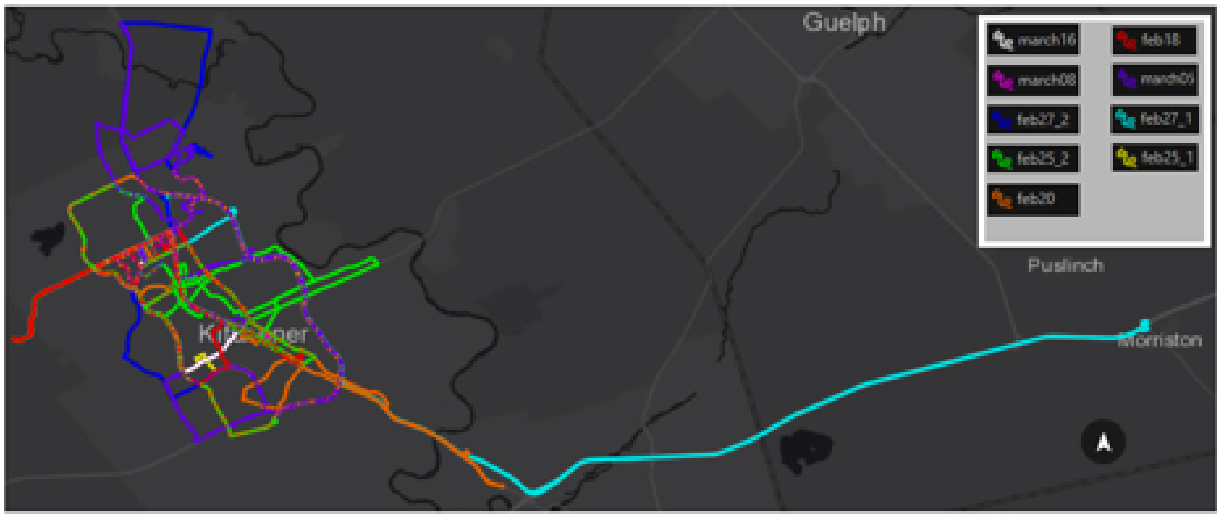


Figure 5.3: Map showing roads in Kitchener-Waterloo covered during data collection. Roads driven on different days are colored differently.

The recorded data consists of various driving scenarios across various road types, day time and night time (see Figure 5.4). We show the time distribution of the collected data in Figure 5.5 (right). We also demonstrate the road type distribution of our benchmark.

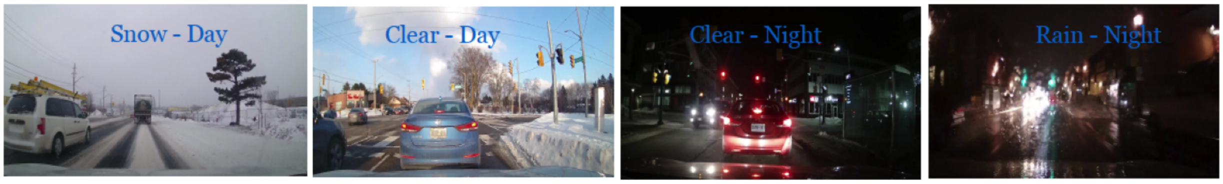


Figure 5.4: Samples of the recorded data. Our data contains images recorded during snow, clear sky, rainy and both at night and day time.

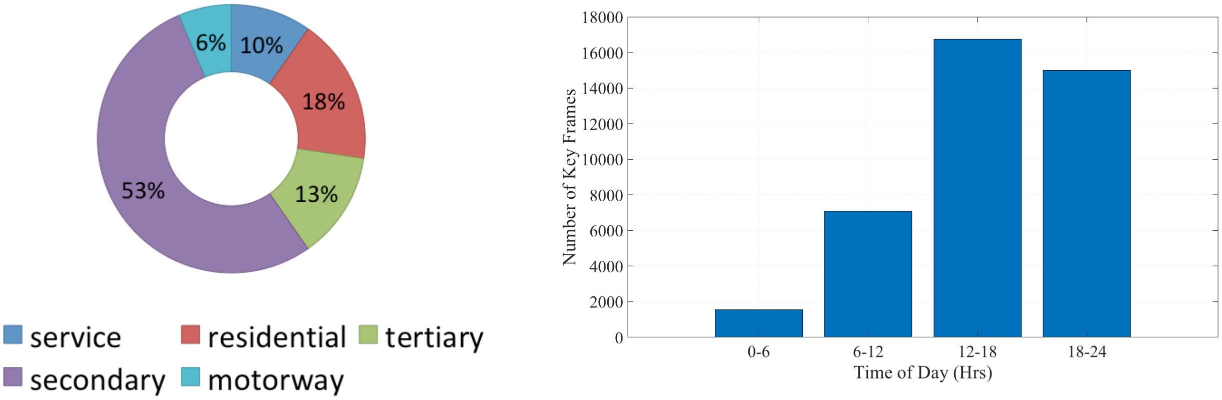


Figure 5.5: The sampled driving data distribution over (a) road types and (b) sampled time.

Our dataset contains 53% secondary roads which is the highest of all road types. The secondary roads mostly indicate a route with two lanes and traffic moving both ways. The recorded high percentage is a true reflection of most road networks that we collected data from. The residential roads follow with 18% of the total dataset and then tertiary roads with 13%. In OSM, tertiary roads are commonly used to refer to roads connecting minor to major roads. Our data has only 10% representation of the motorways as shown in Figure 5.5 (left). This is because our data collection focused on covering urban driving and only drove around major highways for about 50 km. The smallest represented road types with 6% are the service roads which are used to provide access to business areas and public gathering places such as business parks and campsites.

We use OSM as a source for automatic annotation for an end goal of affordance learning and network training. However, with road type information which we find relatively accurate when samples are visually compared to imagery, the OSM can be a direct source of road attributes for real-time autonomous driving especially in rural areas with no coverage of high definition maps [55].

5.2.2 Drift Angle & Intersection Calibration

In this section, We demonstrate how the vehicle dynamics and phone sensory output can be used in calibration with OSM attribute matching based on GPS query. The GPS from the phone sensor may suffer a loss of accuracy for localization leading to an error in pose estimation and intersection localization [65]. Consequently, we constraint the I2MAP algorithm to only consider 3-arm and 4-arm intersection types (intersection types 4, 5, 6 and 7 shown in Figure 2.3). We also only indicate that an image contains an intersection if it appears within 55 meters to the intersection but not more than 10 meters from the intersection. Also, the direction of travel must be approaching the intersection and not moving away. A similar approach was used in [68] where they specified an image with an intersection label to be within 30 meters from the intersection.

However, they only consider 4-arm intersections (intersection type 7) and use Google Street View images. The constraints we asserted improve labeling accuracy considering data collected with a low-grade localization system. However, for OSM labels, any user needs to examine a sample set to quantify the accuracy of the data before use. In any

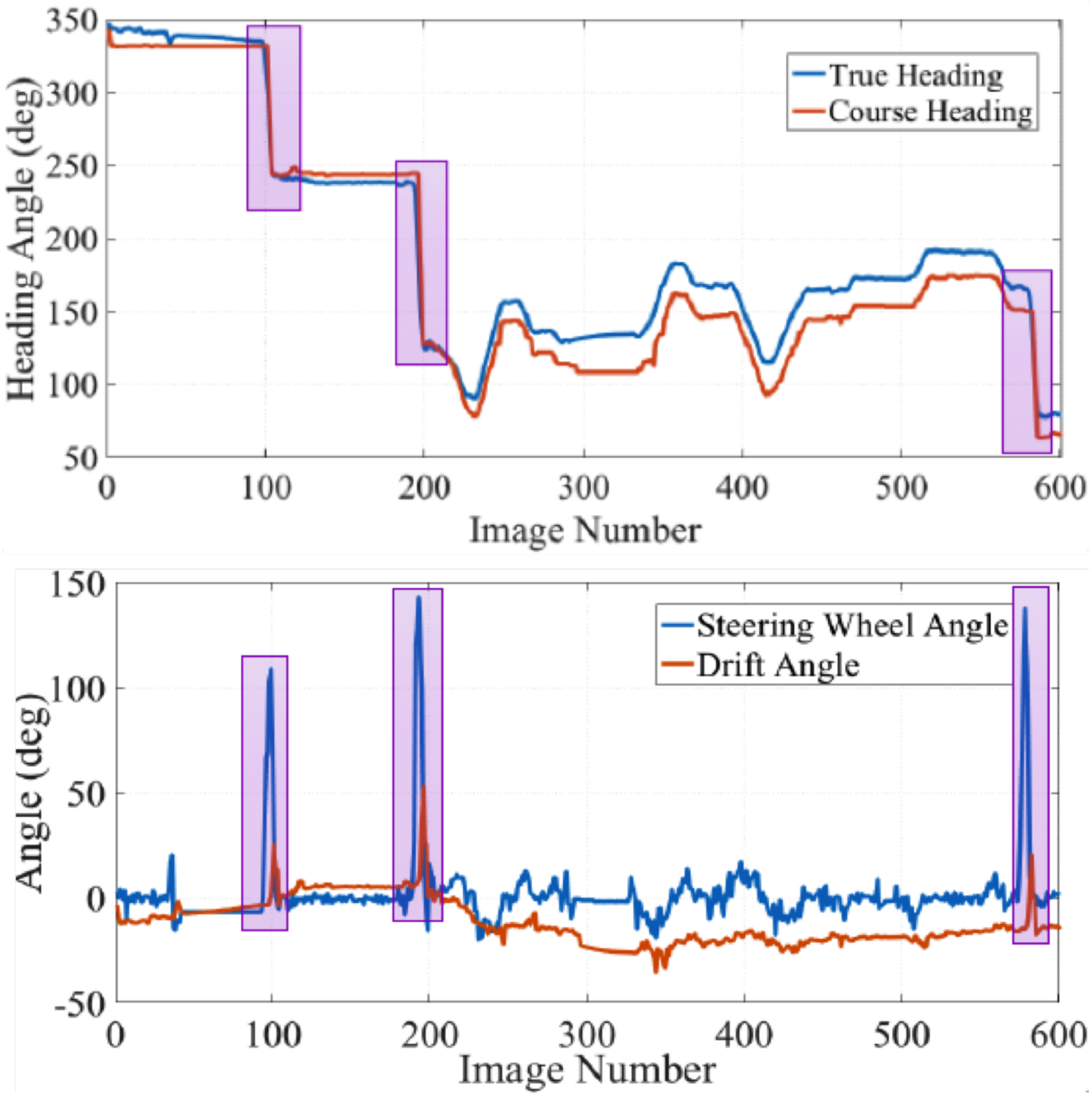


Figure 5.6: Top plot shows that at a sharp angle, the true heading significantly differs from course heading. The bottom plot compares the steering wheel angle logged from the vehicle CAN bus with drift angle (difference of the course heading from the true heading). The transparent magenta rectangle boxes indicate abrupt changes in drift angle and heading angle, which effectively coincides with ego vehicle making a turn at intersections.

case, validating and correcting automatically generated labels will be much faster than generating the labels from no label-base at all. For instance, we combine true heading, course heading and steering wheel angle to demonstrate the effectiveness of calibration with sensory outputs from different devices. The true heading α_t and course heading α_c are collected based on sensors from a phone with reference corresponding to true north. Vehicle true heading corresponds to the angle between vehicle heading and the true north whereas the course heading denotes the angle between the direction of travel (along the lane) and true north. We calculated the drift angle β_d by Eq. 5.3 since both of them are collected from the phone, the drift angle could be a good estimator for calibration with vehicle dynamics.

$$\beta_d = \alpha_c - \alpha_t \tag{5.3}$$

As we can see from Figure 5.6, the heading angle of ego vehicle will have an abrupt change when making a turn at intersections, especially those 3-arm and 4-arm intersection types with a turn larger than 45° due to the change of lane direction. The calculated drift angle β_d based on Eq. 5.3 and steering wheel angle from CAN bus are plotted in Figure 5.6 (bottom) where it is shown that the sensor output from the phone share similar trends with vehicle steering wheel angles, especially at the intersections.

Course and true heading measurements can be analysed further to learn driver behaviour in lane change. However, a more precise navigation system (with survey grade GPS and IMU) would be required to fully capture meaningful variations between driving forward, cornering, and lane change.

5.3 Dynamic Affordance: Traffic Flow Prediction

Traffic flow in major cities can range from smooth to most challenging. Many collisions happen because drivers' attention is deviated either by fatigue or carelessness. In this regard, there have been many efforts from car manufacturers into developing early warning systems such as lane departure and forward-collision warnings. However, it costs an extra dollar to add these features and hence many buyers opt-out and buy vehicles with basic features. Also, systems such as lane departure warning only work in clear weather with no snow covering the road.

Consequently, we trained a CNN model to predict when traffic is moving or when a driver should be stopping given an image input based on the proposed CogData. Such information is not only useful for driver warnings but it can be used in an autonomous driving decision-making algorithm. However, for autonomous driving, a complex model that can incorporate temporal information might be more favorable.

There are several variables that constitute when a vehicle should be stopping. For instance, a vehicle should be stopping if at or approaching red lights. Also, even if the vehicle has a right-of-way but there are pedestrians in front, then it should stop. We do not train the CNN model to recognize red light, pedestrians or other vehicles, instead, our focus is to train for the stopping or moving decisions based on observation.

5.3.1 CNN Model Training

We used a well known ResNet50 architecture [38] pre-trained on ImageNet [23]. We use Keras built on top of the TensorFlow framework [19] for model construction. We resize each image to 227 × 227 × 3 before feeding it to the model. We make sure that there was equal representation of the categories. Images representing moving traffic are 50% while the rest represented stopping traffic scenes.

For this single task, we only use about 3000 images for training (70%) and validation (30%). The testing images were kept separate and only used for model visual prediction analysis presented later in this section. From the ResNet50 convolution base layer, we add a max-pooling layer of size 2x2 and strides of 2. we then add a flattening layer before a dense layer of 4096 neurons. We constraint learned weights using L2 regularization [19] of 0.001 and compile the model using RMSprop [19] with learning rate of 1e-5. The loss function is set to binary cross-entropy and then training is carried out using batch sizes of 32 with 50 epochs. Each epoch had 63 training and validation steps. In the end, our trained model achieved a validation accuracy of 94.68%. we present a simplistic sketch of the training process in Figure 5.7.

5.3.2 CNN Model Visual Prediction Analysis

We used the trained model to suggest actions of either to stop or drive given the driving scene. Some of the driving scenes along with the predicted actions are presented in Figure 5.8.

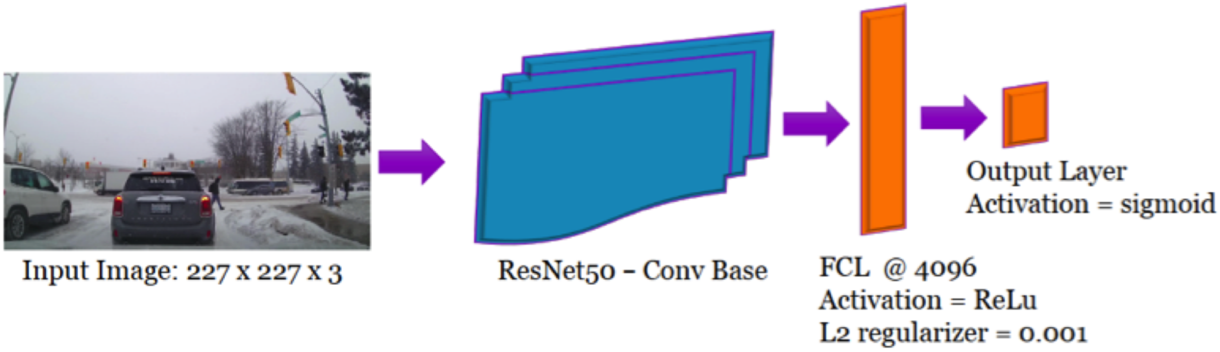


Figure 5.7: We used a ResNet50 as convolution base and added a single Fully Connected Layer (FCL) of 4096 neurons. The input is an image of size 227 x 227 x 3. The output layer uses sigmoid as an activation function.

We present scenes of day time with snowy roads and night time. The leftmost image on the top row of Figure 5.8 shows a snowy road with road totally invisible. Vehicles ahead of the ego vehicle appear far away. The image was correctly classified and the model suggested that it is safe to drive. The middle top image in the same Figure indicates a snowy road with a vehicle on the left lane but very close to ego vehicle. The model can recognize that such scene-setting suggests a safe to drive action. Closer examination of the top right image and the bottom center image in Figure 5.8 indicates that the model doesn't just associate the green traffic light with clear to move action but also considers the actions and positions of other participants. It is also able to read the intention of the ego vehicle given its orientation on the road. The top right image clearly shows that the traffic lights are green and even another vehicle (white) shown in the scene continued to move straight. However, the model predicted that the ego vehicle intended to turn right and given that there are pedestrians, the prediction is a *suggest to stop* action given the learned dynamic

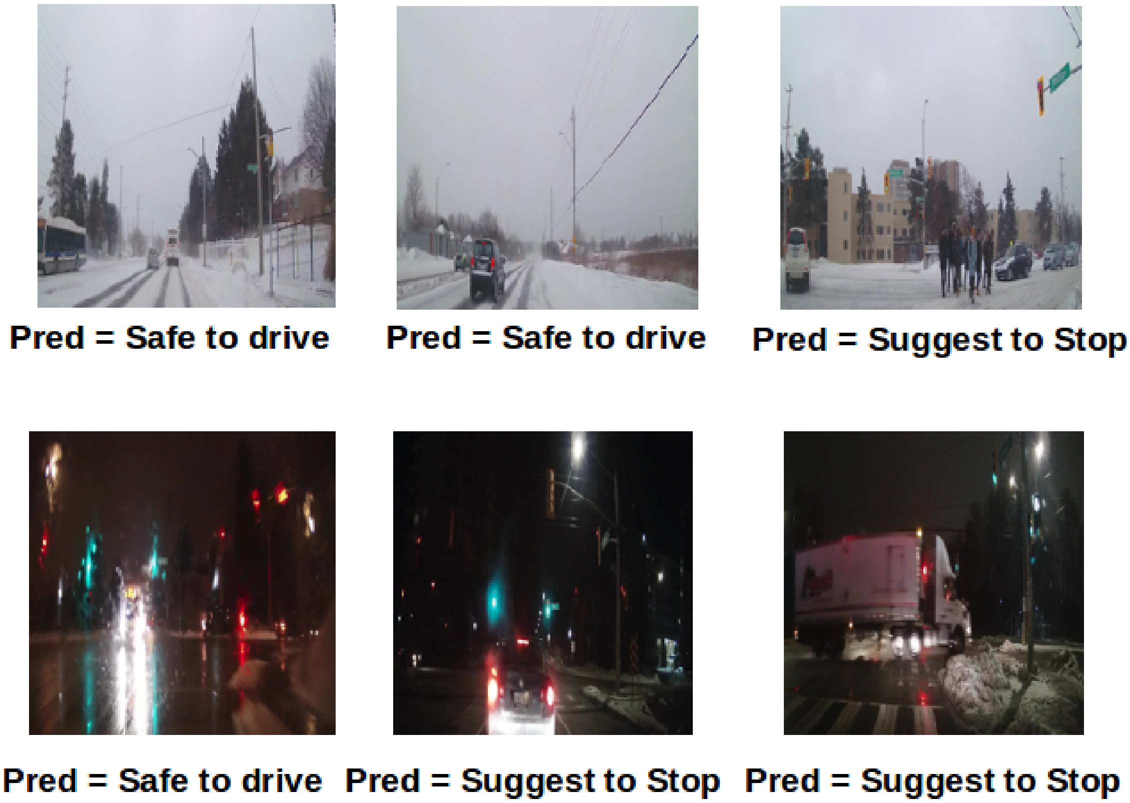


Figure 5.8: Our model prediction on traffic flow under severe weather and visibility conditions.

affordances. Similarly, the bottom center image in Figure 5.8 shows a scene with clear green lights. However, the model learned the difference in lighting between a moving and stopping vehicle (in road setting). Consequently, it was able to suggest that the correct action at that instance was to stop even though the lights were green.

The bottom right image in Figure 5.8 indicates a case where the traffic light is red but the view is obstructed by a large cargo vehicle moving in the adjacent traffic. This scenario represents an obstruction object that the model can correctly act upon without necessarily classifying it. Finally, the bottom left image in the same Figure points out the difficulty of driving at night while it is heavily raining. the traffic lights and illuminations of other participating vehicles might be exaggerated and misleading. However, the model can learn the most important traffic flow cues given the intent of the ego vehicle. Hence, in this image, the model was not misled by various red lights present and still correctly predicted that the traffic is moving. The highlighted scenarios show the robustness of the model which can make correct traffic flow suggestions based on learned dynamic affordances of the complex traffic flow scenes. Such prediction can further be improved by adding an LSTM (long short-term memory) [40] capability to model training.

5.4 Conclusion

In this work, we introduced a driving data collection and automatic annotation framework designed for direct perception and imitation learning. The collected data from distributed devices are synchronized and annotated with filtered labels. The proposed benchmark includes vehicle dynamics and road attributes under various scenarios including day time and night time under various weather conditions (no rain, rain, snow).

Furthermore, we train and evaluate the traffic flow prediction network for harsh weather driving aid system and demonstrate its effectiveness. We concluded that the proposed data collection and annotation framework can easily be deployed at a larger scale and in different geographic locations, to enhance the dynamic affordance learning trained model generalization ability.

Chapter 6

Conclusion and Future Work

In Chapter 3, we presented a Cogdrive dataset collected in various severe weather conditions including snow and rain conditions. The dataset was collected both at night and during day time. A CogDrive Dataset Collector App was developed to help collect large datasets needed for training robust static and dynamic affordance learning CNN models. The design and calibration procedure of the app was presented in chapter 3. Using the developed app and I2MAP algorithm, we demonstrate that it is easier and cheaper to collect real-world driving scene dataset required for affordance learning compared to the mediated perception pipeline. In chapter 4, we highlighted that geospatial open source databases such as GSV and OSM can be leveraged as a source of data for training affordance learning models. In chapters 4 and 5, we also mentioned issues involved in data collection. Some of the highlighted issues are inaccurate positioning and trying to match datasets from different sources with a possible offset in database updating.

In this thesis, we successfully demonstrated that static and dynamic affordance learning is an important layer of perception. This layer must be incorporated for the autonomous vehicles to have a complete and ubiquitous scene understanding regardless of the driving environment. This approach is also better equipped in dealing with the long-tail problem in autonomous driving vehicles than a mediated perception approach. As was discussed in chapter 5 of this thesis, the model trained for traffic flow dynamic affordance, was able to correctly predict the best course of action even in the most dynamically challenging driving scenes (see Figure 5.8). The argument here is that by learning driving affordances, the models can make reasonable estimates about the object in question given the ego vehicle's position and orientation.

For future work, we plan to incorporate Convolutional Neural Network with memory-based algorithms such as LSTM, and environment-to-agent feedback loop approaches such as reinforcement learning into our pipeline. Such algorithm fusion will help us study dynamic driving behavior at all levels of difficulty in real driving situations. We will also expand our abilities to automatically annotate driving datasets with static and dynamic attributes by improving the phone-based localization accuracy. With such realization, we can add features to the CogDrive app, to collect and label the datasets in realtime, making cloud sourcing data collection a possibility. In time, we will also use monocular depth estimation algorithms to mine distance-based affordance information from a single image. Contribution of this thesis can be summarised as follows:

- We designed an affordable dashboard-based system that included building the CogDrive Data Collection phone app. The system was used to collect location-based image datasets consisting of about 40,000 images with more than 40 labels including driver's input, ego vehicle dynamics, and OSM road attributes.

- We developed an affordance learning model for the Kitchener-Waterloo area, based on freely available Google Street View imagery and OpenStreetMap road information. This model was cross-validated on data collected in San Francisco Bay Area.
- We developed a scalable and affordable data collection and automatic labeling framework for dynamic affordance learning based on image-to-map proximity algorithm.
- We presented a detailed analysis for lane following and taking a turn at an intersection using the course and heading angle information.
- We trained a traffic flow prediction network. It could act as a smart driver assistance and was tested using driving scenes captured in various weather conditions including snowing and night time.

References

[1] CMAttitudeReferenceFrame - Core Motion apple developer documentation. https://developer.apple.com/documentation/coremotion/cmattitudereferenceframe?language=objc. Accessed: 2019-03-17.

[2] iOS Device Compatibility Reference camera features overview. https://developer.apple.com/library/archive/documentation/DeviceInformation/Reference/iOSDeviceCompatibility/Cameras/Cameras.html. Accessed: 2019-03-17.

[3] iphone gyroscope. https://www.nomtek.com/scanning-rooms-with-an-iphone/8rhft/. Accessed: 2019-03-17.

[4] Lidr is the latest game-changing advancement for autonomous vehicles. https://innovationatwork.ieee.org/lidr-is-the-latest-game-changing-advancement-for-autonomous-vehicles. IEEE Innovation at Work. Oct 2018. Accessed: 2019-04-10.

[5] Waymo self-driving Technology 2019. https://waymo.com/tech/. Accessed: 2019-03-17.

[6] Mohammed Al-Qizwini, Iman Barjasteh, Hothaifa Al-Qassab, and Hayder Radha. Deep learning algorithm for autonomous driving using googlenet. In *Intelligent Vehicles Symposium (IV), 2017 IEEE*, pages 89–96. IEEE, 2017.

[7] Talal Al-Shihabi and Ronald R Mourant. A framework for modeling human-like driving behaviors for autonomous vehicles in driving simulators. In *Proceedings of the fifth international conference on Autonomous agents*, pages 286–291. ACM, 2001.

[8] Mohamed Aly. Real time detection of lane markers in urban streets. In *Intelligent Vehicles Symposium, 2008 IEEE*, pages 7–12. IEEE, 2008.

[9] Kiam Heong Ang, Gregory Chong, and Yun Li. Pid control system analysis, design, and technology. *IEEE transactions on control systems technology*, 13(4):559–576, 2005.

[10] E Arias and B Guinot. " coordinated universal time utc: historical background and perspectives. In *Journees systemes de reference spatio-temporels*, 2004.

[11] Nils Bjorck, Carla P Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. In *Advances in Neural Information Processing Systems*, pages 7694–7705, 2018.

[12] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, et al. End to end learning for self-driving cars. *arXiv preprint arXiv:1604.07316*, 2016.

[13] Martin Buehler, Karl Iagnemma, and Sanjiv Singh. *The DARPA urban challenge: autonomous vehicles in city traffic*, volume 56. springer, 2009.

[14] Luca Caltagirone, Mauro Bellone, Lennart Svensson, and Mattias Wahde. Simultaneous perception and path generation using fully convolutional neural networks. *arXiv preprint arXiv:1703.08987*, 2017.

[15] Sean Campbell, Niall O'Mahony, Lenka Krpalcova, Daniel Riordan, Joseph Walsh, Aidan Murphy, and Conor Ryan. Sensor technology in autonomous vehicles: a review. In *2018 29th Irish Signals and Systems Conference (ISSC)*, pages 1–4. IEEE, 2018.

[16] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018.

[17] Chenyi Chen, Ari Seff, Alain Kornhauser, and Jianxiong Xiao. Deepdriving: Learning affordance for direct perception in autonomous driving. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2722–2730. IEEE, 2015.

[18] Xue-Wen Chen and Xiaotong Lin. Big data deep learning: challenges and perspectives. *IEEE access*, 2:514–525, 2014.

[19] François Chollet et al. Keras, 2015.

[20] Siddharth Choudhary, Alexander JB Trevor, Henrik I Christensen, and Frank Dellaert. Slam with object discovery, modeling and mapping. In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 1018–1025. IEEE, 2014.

[21] Felipe Codevilla, Matthias Müller, Antonio López, Vladlen Koltun, and Alexey Dosovitskiy. End-to-end driving via conditional imitation learning. In *International Conference on Robotics and Automation (ICRA)*, 2018.

[22] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[23] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[24] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[25] Xinxin Du, Marcelo H Ang, Sertac Karaman, and Daniela Rus. A general pipeline for 3d detection of vehicles. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3194–3200. IEEE, 2018.

[26] Daniel J Fagnant and Kara Kockelman. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice*, 77:167–181, 2015.

[27] Huan Fu, Mingming Gong, Chaohui Wang, Kayhan Batmanghelich, and Dacheng Tao. Deep ordinal regression network for monocular depth estimation. In *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition, pages 2002–2011, 2018.

[28] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4340–4349, 2016.

[29] Philippe Gaussier, Sorin Moga, Mathias Quoy, and Jean-Paul Banquet. From perception-action loops to imitation processes: A bottom-up approach of learning by imitation. *Applied Artificial Intelligence*, 12(7-8):701–727, 1998.

[30] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3d traffic scene understanding from movable platforms. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):1012–1025, 2013.

[31] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3d traffic scene understanding from movable platforms. *IEEE transactions on pattern analysis and machine intelligence*, 36(5):1012–1025, 2014.

[32] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[33] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[34] Felix A Gers, Jürgen Schmidhuber, and Fred Cummins. Learning to forget: Continual prediction with lstm. 1999.

[35] James J Gibson. *The ecological approach to visual perception: classic edition*. Psychology Press, 2014.

[36] Hongyan Guo, Dongpu Cao, Hong Chen, Chen Lv, Huaji Wang, and Siqi Yang. Vehicle dynamic state estimation: state of the art schemes and perspectives. *IEEE/CAA Journal of Automatica Sinica*, 5(2):418–431, 2018.

[37] Mordechai Haklay and Patrick Weber. Openstreetmap: User-generated street maps. *Ieee Pervas Comput*, 7(4):12–18, 2008.

[38] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[39] Enrique Herrera-Viedma, Francisco Herrera, and Francisco Chiclana. A consensus model for multiperson decision making with different preference structures. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 32(3):394–402, 2002.

[40] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[41] Sinpyo Hong, Man Hyung Lee, Ho-Hwan Chun, Sun-Hong Kwon, and Jason L Speyer. Observability of error states in gps/ins integration. *IEEE Transactions on Vehicular Technology*, 54(2):731–743, 2005.

[42] Jilin Huang, Ivan Tanev, and Katsunori Shimohara. Evolving a general electronic stability program for car simulated in torcs. In *Computational Intelligence and Games (CIG), 2015 IEEE Conference on*, pages 446–453. IEEE, 2015.

[43] Xinyu Huang, Xinjing Cheng, Qichuan Geng, Binbin Cao, Dingfu Zhou, Peng Wang, Yuanqing Lin, and Ruigang Yang. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 954–960, 2018.

[44] Bernd Jähne and Horst Haußecker. Computer vision and applications. *A Guide for Students and Practitioners*, 2000.

[45] Joel Janai, Fatma Güney, Aseem Behl, and Andreas Geiger. Computer vision for autonomous vehicles: Problems, datasets and state-of-the-art. *arXiv preprint arXiv:1704.05519*, 2017.

[46] Sören Kammel, Julius Ziegler, Benjamin Pitzer, Moritz Werling, Tobias Gindele, Daniel Jagzent, Joachim Schröder, Michael Thuy, Matthias Goebl, Felix von Hundelshausen, et al. Team annieway's autonomous system for the 2007 darpa urban challenge. *Journal of Field Robotics*, 25(9):615–639, 2008.

[47] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[48] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[49] Jason Ku, Melissa Mozifian, Jungwook Lee, Ali Harakeh, and Steven L Waslander. Joint 3d proposal generation and object detection from view aggregation. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–8. IEEE, 2018.

[50] Seokju Lee, Junsik Kim, Jae Shin Yoon, Seunghak Shin, Oleksandr Bailo, Namil Kim, Tae-Hee Lee, Hyun Seok Hong, Seung-Hoon Han, and In So Kweon. Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1947–1955, 2017.

[51] Matthew.lynberg.ctr@dot.gov. Automated vehicles for safety, Jun 2019.

[52] N. Mayer, E. Ilg, P. Häusser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. arXiv:1512.02134.

[53] Michael Montemerlo, Sebastian Thrun, Hendrik Dahlkamp, David Stavens, and Sven Strohband. Winning the darpa grand challenge with an ai robot. In *AAAI*, pages 982–987, 2006.

[54] Urs Muller, Jan Ben, Eric Cosatto, Beat Flepp, and Yann L Cun. Off-road obstacle avoidance through end-to-end learning. In *Advances in neural information processing systems*, pages 739–746, 2006.

[55] Teddy Ort, Liam Paull, and Daniela Rus. Autonomous vehicle navigation in rural environments without detailed prior maps. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2040–2047. IEEE, 2018.

[56] Yunpeng Pan, Ching-An Cheng, Kamil Saigol, Keuntaek Lee, Xinyan Yan, Evangelos Theodorou, and Byron Boots. Agile autonomous driving using end-to-end deep imitation learning. *Proceedings of Robotics: Science and Systems. Pittsburgh, Pennsylvania*, 2018.

[57] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in neural information processing systems*, pages 305–313, 1989.

[58] Charles R Qi, Wei Liu, Chenxia Wu, Hao Su, and Leonidas J Guibas. Frustum pointnets for 3d object detection from rgb-d data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 918–927, 2018.

[59] Viktor Rausch, Andreas Hansen, Eugen Solowjow, Chang Liu, Edwin Kreuzer, and J Karl Hedrick. Learning a deep neural net policy for end-to-end control of autonomous vehicles. In *American Control Conference (ACC), 2017*, pages 4914–4919. IEEE, 2017.

[60] Stephan R Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *European Conference on Computer Vision*, pages 102–118. Springer, 2016.

[61] German Ros, Sebastian Ramos, Manuel Granados, Amir Bakhtiary, David Vazquez, and Antonio M Lopez. Vision-based offline-online perception paradigm for au-

tonomous driving. In *Applications of Computer Vision (WACV), 2015 IEEE Winter Conference on*, pages 231–238. IEEE, 2015.

[62] Stuart J Russell and Peter Norvig. *Artificial intelligence: a modern approach*. Malaysia; Pearson Education Limited,, 2016.

[63] Kamal Saadeddin, Mamoun F Abdel-Hafez, and Mohammad Amin Jarrah. Estimating vehicle state by gps/imu fusion with vehicle dynamics. *Journal of Intelligent & Robotic Systems*, 74(1-2):147–172, 2014.

[64] Axel Sauer, Nikolay Savinov, and Andreas Geiger. Conditional affordance learning for driving in urban environments. *arXiv preprint arXiv:1806.06498*, 2018.

[65] Harald Schafer, Eder Santana, Andrew Haden, and Riccardo Biasini. A commute in data: The comma2k19 dataset. *arXiv preprint arXiv:1812.05752*, 2018.

[66] Klaus Schossmaier, Ulrich Schmid, Martin Horauer, and Dietmar Loy. Specification and implementation of the universal time coordinated synchronization unit (utcsu). *Real-Time Systems*, 12(3):295–327, 1997.

[67] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:187–210, 2018.

[68] Ari Seff and Jianxiong Xiao. Learning from maps: Visual common sense for autonomous driving. *arXiv preprint arXiv:1611.08583*, 2016.

[69] Heiko G Seif and Xiaolong Hu. Autonomous driving in the icity—hd maps as a key challenge of the automotive industry. *Engineering*, 2(2):159–162, 2016.

[70] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a formal model of safe and scalable self-driving cars. *arXiv preprint arXiv:1708.06374*, 2017.

[71] David Silver, J Andrew Bagnell, and Anthony Stentz. Learning from demonstration for autonomous navigation in complex unstructured terrain. *The International Journal of Robotics Research*, 29(12):1565–1592, 2010.

[72] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[73] Chen Sun, Jean M. Uwabeza Vianney, and Dongpu Cao. Affordance learning in direct perception for autonomous driving. *arXiv preprint arXiv:1903.08746*, 2019.

[74] Marvin Teichmann, Michael Weber, Marius Zoellner, Roberto Cipolla, and Raquel Urtasun. Multinet: Real-time joint semantic reasoning for autonomous driving. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1013–1020. IEEE, 2018.

[75] Charles Thorpe, Martial Hebert, Takeo Kanade, and Steven Shafer. Vision and navigation for the carnegie-mellon navlab. *Annual Review of Computer Science*, 2(1):521–556, 1987.

[76] Sebastian Thrun, Mike Montemerlo, Hendrik Dahlkamp, David Stavens, Andrei Aron, James Diebel, Philip Fong, John Gale, Morgan Halpenny, Gabriel Hoffmann, et al. Stanley: The robot that won the darpa grand challenge. *Journal of field Robotics*, 23(9):661–692, 2006.

[77] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2):26–31, 2012.

[78] Shimon Ullman. Against direct perception. *Behavioral and Brain Sciences*, 3(3):373–381, 1980.

[79] Fei-Yue Wang, Nan-Ning Zheng, Dongpu Cao, Clara Marina Martinez, Li Li, and Teng Liu. Parallel driving in cpss: A unified approach for transport automation and vehicle intelligence. *IEEE/CAA Journal of Automatica Sinica*, 4(4):577–587, 2017.

[80] Qing Wang, Long Chen, and Wei Tian. End-to-end driving simulation via angle branched network, 2018.

[81] Yan Wang, Wei-Lun Chao, Divyansh Garg, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8445–8453, 2019.

[82] Jian Wei, Jianhua He, Yi Zhou, Kai Chen, Zuoyin Tang, and Zhiliang Xiong. Enhanced object detection with deep convolutional neural networks for advanced driving assistance. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[83] Kyle Wiggers. Lyft users will be able to hail driverless waymo cars in phoenix, May 2019.

[84] Yang Xing, Chen Lv, Long Chen, Huaji Wang, Hong Wang, Dongpu Cao, Efstathios Velenis, and Fei-Yue Wang. Advances in vision-based lane detection: algorithms,

integration, assessment, and perspectives on acp-based parallel vision. *IEEE/CAA Journal of Automatica Sinica*, 5(3):645–661, 2018.

[85] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. *arXiv preprint*, 2017.

[86] Jian-Ru Xue, Jian-Wu Fang, and Pu Zhang. A survey of scene understanding by event reasoning in autonomous driving. *International Journal of Automation and Computing*, 15(3):249–266, 2018.

[87] Guorun Yang, Hengshuang Zhao, Jianping Shi, Zhidong Deng, and Jiaya Jia. Segstereo: Exploiting semantic information for disparity estimation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 636–651, 2018.

[88] Yurong You, Yan Wang, Wei-Lun Chao, Divyansh Garg, Geoff Pleiss, Bharath Hariharan, Mark Campbell, and Kilian Q Weinberger. Pseudo-lidar++: Accurate depth for 3d object detection in autonomous driving. *arXiv preprint arXiv:1906.06310*, 2019.

[89] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv preprint arXiv:1805.04687*, 2018.

[90] Hongyi Zhang, Andreas Geiger, and Raquel Urtasun. Understanding high-level semantics by modeling traffic patterns. In *Proceedings of the IEEE international conference on computer vision*, pages 3056–3063, 2013.

[91] Bolei Zhou, Aditya Khosla, Agata Lapedriza, Antonio Torralba, and Aude Oliva. Places: An image database for deep scene understanding. *arXiv preprint arXiv:1610.02055*, 2016.

[92] Junchuan Zhou, Johannes Traugott, Bruno Scherzinger, Christian Miranda, and Adrian Kipka. A new integration method for mems based gnss/ins multi-sensor systems. In *Proceedings of the International Technical Meeting of the Ion Satellite Division (ION GNSS), Tampa, FL, USA*, pages 14–18, 2015.

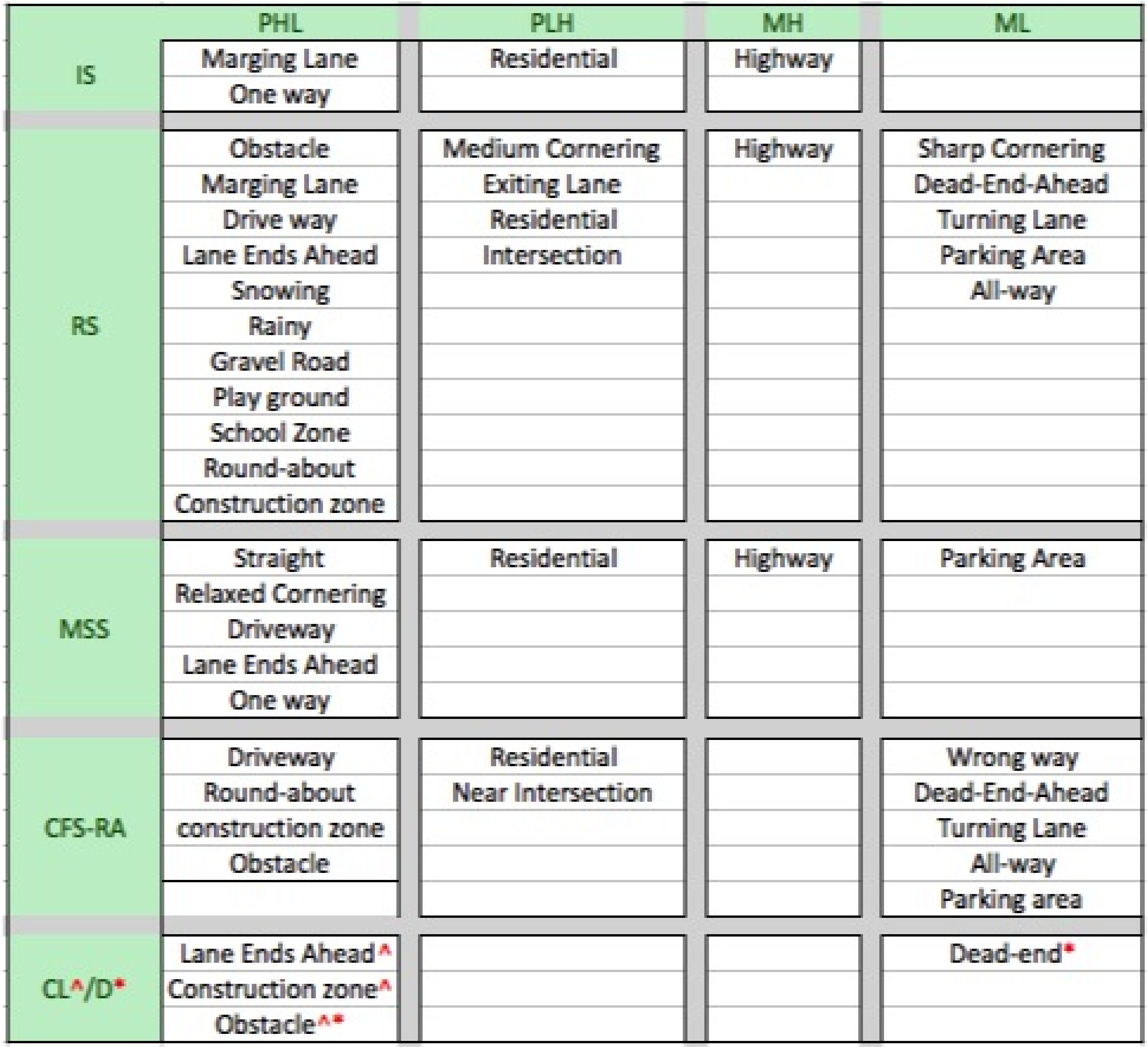
APPENDICES

Appendix A

Classifying Driveable Space Based on Driving Affordances

In an attempt to use driving affordances for autonomous vehicle decision making, we classify driveable space considering likely next dynamic driving task and likely current speed state. For instance, having predicted affordances such as the number of lanes, bike lanes, one-way and driveable heading, we can make a fuzzy-based determination of whether the road view in that instance, is of a highway or residential road. Consequently, such a revelation would inform the vehicle decision to either reduce or increase speed if the system detects a transition in the environment from the highway to a residential road or vise Versa. Another example would be to use classification of distance to an intersection to alert the vehicle to cautiously proceed, reduce speed or stop altogether.

As shown in Figure A.1, the predicted or generated affordances (generated affordances



	PHL	PLH	MH	ML
IS	Marging Lane	Residential	Highway	
	One way			
RS	Obstacle	Medium Cornering	Highway	Sharp Cornering
	Marging Lane	Exiting Lane		Dead-End-Ahead
	Drive way	Residential		Turning Lane
	Lane Ends Ahead	Intersection		Parking Area
	Snowing			All-way
	Rainy			
	Gravel Road			
	Play ground			
	School Zone			
	Round-about			
	Construction zone			
MSS	Straight	Residential	Highway	Parking Area
	Relaxed Cornering			
	Driveway			
	Lane Ends Ahead			
	One way			
CFS-RA	Driveway	Residential		Wrong way
	Round-about	Near Intersection		Dead-End-Ahead
	construction zone			Turning Lane
	Obstacle			All-way
				Parking area
CL^/D*	Lane Ends Ahead^			Dead-end*
	Construction zone^			
	Obstacle^*			

Figure A.1: Affordances classified in the union of five categories of likely next dynamic action and four categories of likely vehicle current speed state. Affordances with a red star suggest changing direction behavior, while those with a red hat, suggest changing lane instead. Affordances with both red star and a hat, suggest the possibility of either changing the lane or direction.

are determined indirectly by combining several predicted affordances) are first classified into five categories of the likely next dynamic driving states. Category 1 includes such affordances that would invoke the vehicle to Increase Speed (IS). Category 2 leads to Reduce Speed (RS) action. Category 3 leads to Maintain Similar Speed (MSS) action, while category 4 would alert the vehicle to Come to a Full Stop or Related Actions (CFS-RA). Category 5 includes realizations such as the presence of obstacle ahead zone which requires the vehicle to either Change Lane or Direction (CL/D).

Likely current speed states are inferred based on the affordances in the immediate environment. The current speed states category 1 is Possible High or Lower speed (PHL), where the environment context suggests that the likelihood of the vehicle current speed being high, is higher than the likelihood of current speed being lower. Category two (PLH) is a direct inverse of category 1. Category 3 and 4 include such driveable spaces that suggest a strong likelihood of speed state to be Most likely High speed (MH) and Most likely Low speed (ML) respectively, but there is no gray area.

In formulating the presented classification of driveable space, we recognize that there are no clear boundaries among classes. Hence, such classification may greatly depend on individuals carrying out the task of classification. Also, this is mainly a task that would require extensive validation and consultation to generate consensus classification criteria.

Appendix B

Traffic Flow Prediction Sample Images

In Figure B.1 and Figure B.2 , we show predictions for our model trained on data collected using the proposed dynamic affordance learning framework. We trained the model to predict traffic flow pertaining to *stop* = *predicted [0]* and *go* = *predicted [1]* action.

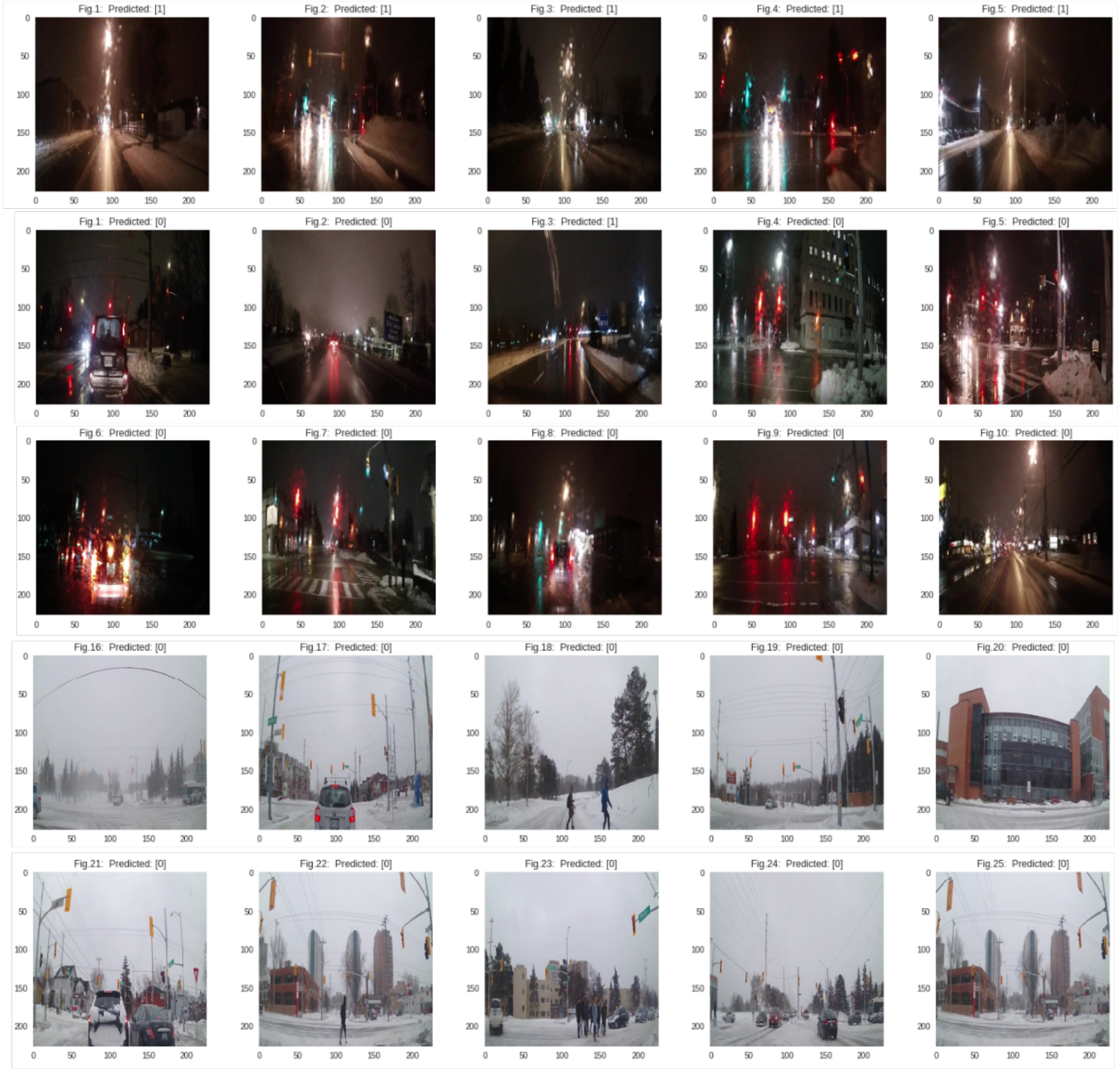


Figure B.1: Predictions sample 1. At night and during the day predictions. The model learned to make traffic flow decisions at night. It was able to differentiate through street lights and vehicle lighting and signals.Predicted: [1] means a go action is suggested while predicted: [0] indicates that a stop action is required.

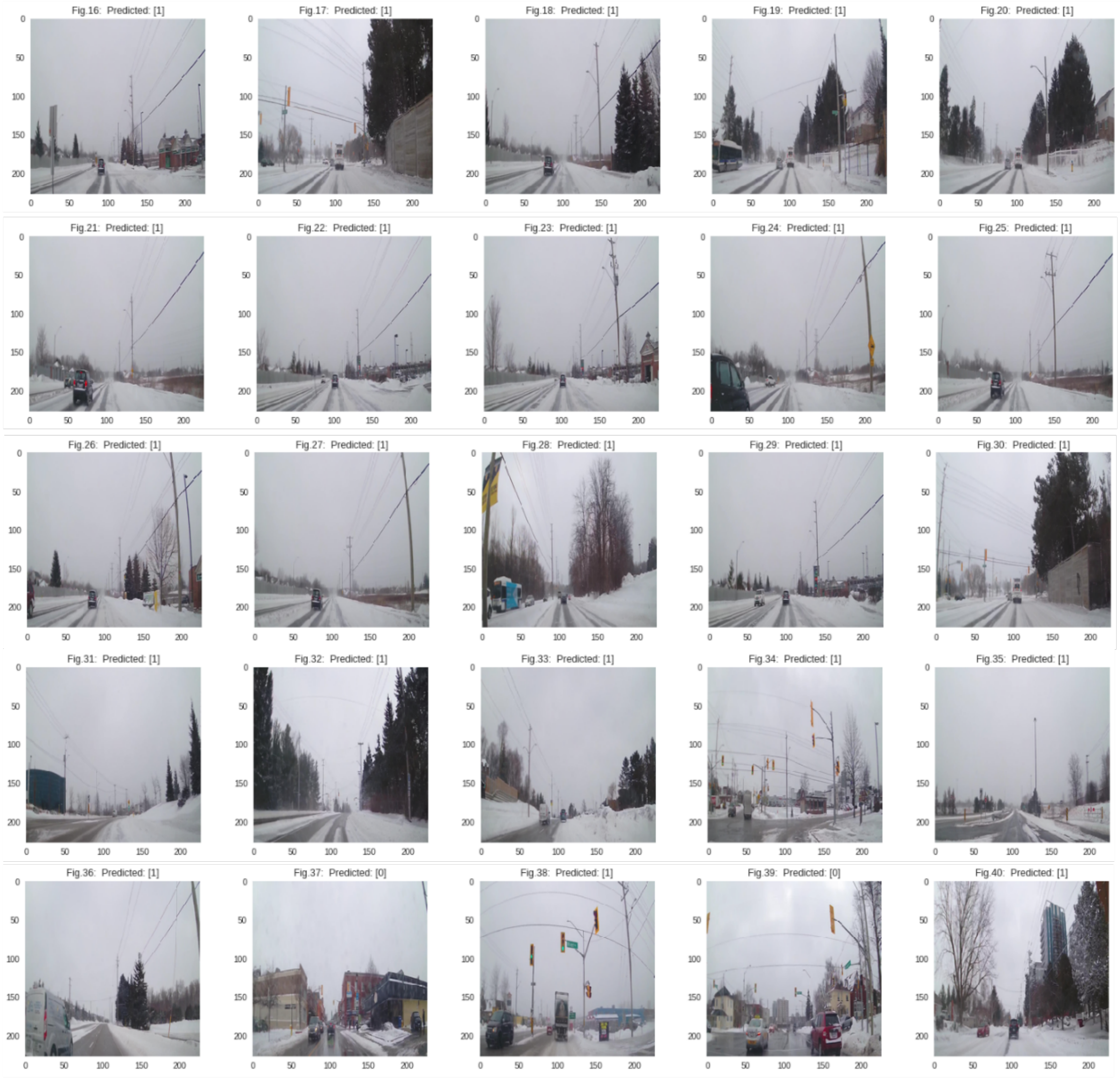


Figure B.2: Predictions sample 2. Predictions on roads covered with snow. The model was able to make correct predictions even with snow deposits.Predicted: [1] means a go action is suggested while predicted: [0] indicates that a stop action is required.