Graduate Program in Sustainability Science

Graduate School of Frontier Sciences

The University of Tokyo

2019

Master's Thesis

Sustainability Impact of Autonomous Vehicles

Submitted August 23, 2019

Adviser: Professor Yoshikuni Yoshida

Co-Adviser: Assistant Professor Shogo Kudo

Maya Frederika Dhondt

SUSTAINABILITY IMPACT OF AUTONOMOUS VEHICLES

ABSTRACT

With powerful computing and AI systems becoming ubiquitous in our increasingly networked world, fully autonomous vehicles (AVs) are rapidly becoming a reality. This technology has great potential to improve road traffic safety, private mobility, and our excessive parking needs. However, claims that AVs will improve the environmental impact of transportation CO_2 emissions and the social and economic impact of traffic congestion require further examination. Whether or not AVs will also have a positive impact on traffic congestion and transport emissions depends on how the introduction of this technology is tackled by policymakers.

It has been proposed that AVs will improve congestion and consequently CO_2 emissions as safety improvements will allow for smaller, lighter, and faster vehicles, increasing vehicle throughput. It has also been suggested that the increased attractiveness and mobility associated with AVs could result in a rebound effect. This could lead to a shift from public transport and low impact modes such as walking or bicycle to private vehicle transport, increasing congestion and CO_2 emissions. To better understand these impacts of AV technology, a traffic simulation was constructed, with questionnaire survey conducted to provide simulation inputs.

309 residents of Kanagawa, Saitama, and Chiba prefectures were surveyed on their short-distance travel behaviour. More than three quarters of respondents reported possessing a driver’s license, and two thirds reported their household as owning one or more vehicles. Respondents were asked to select a transport mode for a specified trip in order to estimate demand for different modes under different trip conditions (length and purpose). To understand route choice, respondents were presented with a discrete choice experiment of routes of different lengths and journey times. Responses were used to estimate the utility

function of route choice. The demand estimation and utility function obtained were used to inform route and mode choice in traffic simulation.

Using the demand estimation acquired from simulation, trips were created on a 5 x 5 map grid with four origins and two destinations (supermarket and station). Trip time was recorded for each journey, assuming a relationship between velocity and vehicle density, vehicle length, and headway time. Route choice was made based on the direct route between the origin and destination with the highest utility.

Three scenarios of AV adoption were considered:

(1) Base scenario

In this scenario, demand shift to AVs was assumed to be 0 % (no AV technology), with a conventional vehicle headway time of 2 s.

(2) Partially automated AV 100 % adoption scenario

In this scenario, partially automated AVs were assumed, where a driver was still required, but a headway time for these partial AVs was assumed to be 0.5 s.

(3) Fully automated AV 100 % adoption scenario

In this scenario, demand shift to AVs was assumed to be 100 % (full AV technology), with an AV headway time of 0.5 s.

In addition to this, two cases of decision-making were implemented for route choice and mode choice demand shift (i.e. travel time for the utility function):

(1) "Imperfect knowledge" decision-making

For this decision-making case, imperfect knowledge about traffic conditions by travellers was assumed. Travel time for the route choice and mode choice demand shift calculations was estimated using a "typical" velocity rather than the true travel time (for cars, taxis, buses, and AVs).

(2) "Full knowledge" decision-making

For this decision-making case, full knowledge about traffic conditions by travellers was assumed. True travel time was used for the route choice and mode choice demand shift calculations.

For each scenario in each decision-making case, traffic congestion, traveller utility, and CO_2 emissions were investigated.

Compared to the base and partial scenarios, a modal shift away from lower-impact modes (such as bicycles) and public transportation (such as buses) was observed in the AV scenario. This was due to the increased "attractiveness" (shorter travel time) of AVs over conventional vehicles resulting in a rebound effect. However, this effect on did not directly translate to increased travel times and congestion as it was balanced by the effect of the increased speed of AVs due to the reduced headway time, which was also observed in the partial scenario. In the "imperfect knowledge" case, slight congestion increase was observed in the AV scenario by observing velocity change from high- to low-congestion conditions, likely due to greater demand shift and poorer route optimisation in this case.

Traveller utility was understood to decrease with congestion and trip time. Over the simulation, trip time increased due to increasing congestion on certain route segments. In both decision-making cases, trip time was reduced in the AV scenario, although since the number of AV trips was much greater in this scenario this effect was reduced over time. With a slight increase in congestion observed in the "imperfect knowledge" case, traveller utility can be interpreted to decrease in the AV scenario compared to the partial scenario over the base scenario.

In simulation, conventional vehicles (cars, partial AVs, taxis, and buses) were assumed to have internal combustion engines (ICEs), while AVs were assumed to be electric vehicles (EVs). A relationship between fuel consumption and velocity was assumed for each vehicle type, and this was used to derive the amount of CO_2 emissions produced in each scenario. Emissions were reduced by similar factors in both the AV and partial scenarios,

demonstrating that the positive effect of electric powertrain balanced with the negative rebound effect of the increased demand for AVs when compared with the positive effect of the increased speed of partial AVs. Additionally, CO_2 emissions overall decreased in the “full knowledge” case, showing that route-optimisation in this case had an important role in reducing overall emissions.

Key words: autonomous vehicle, AV, sustainability, traffic simulation

ACKNOWLEDGEMENTS

I received invaluable support from several individuals and organisations throughout my master's research and would like to take the opportunity to thank everyone who advised and encouraged me throughout this process.

In particular, I would like to express my gratitude to Professor Yoshikuni Yoshida, without whose support and advice this project would not have been possible. The many hours devoted to the review and consideration of my work are greatly appreciated. I would also like to acknowledge the input of Professor Shogo Kudo whose advice supported the realisation of this project.

Additionally, I would like to acknowledge the instrumental support of the MEXT scholarship provided by the Japanese Ministry of Education, Culture, Sports, Science, and Technology, without which I would not have been able to pursue this opportunity.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF FIGURES (continued)

LIST OF FORUMULAS AND EQUATIONS

LIST OF ABBREVIATIONS

ADAS	Advanced Driver-Assistance Systems
AV	Autonomous Vehicle
BIBD	Balanced Incomplete Block Design
EV	Electric Vehicle
ICE	Internal Combustion Engine
SAV	Shared Autonomous Vehicle
VMT	Vehicle Miles Travelled
V2I	Vehicle to Infrastructure
V2V	Vehicle to Vehicle
V2X	Vehicle to External Environment
WTP	Willingness to Pay

LIST OF UNITS OF MEASUREMENT

UK £	United Kingdom pounds
kg	kilogram
kg / L	kilograms per litre
kg / kWh	kilograms per kilowatt hour
km	kilometre
km / h	kilometres per hour
km / kWh	kilometres per kilowatt hour
km / L	kilometres per litre
kWh	kilowatt hour
L	litres
m	minutes
s	seconds
US $	United States dollars

1 INTRODUCTION

"From a technological point of view, AVs have arrived; the bigger task is for cities to integrate them. As autonomous driving matures, one thing is all but certain: the world's mobility challenges will increasingly be met with silicon rather than asphalt."

(Claudel & Ratti, 2015)

As Moore observed in his eponymous law, the components of integrated circuits have been increasing exponentially in recent decades, leading to ever more compact and complex microprocessors (Moore, 1965). More powerful and sophisticated computing systems are making possible the realisation of technology once confined to science fiction storylines – fully autonomous vehicles (AVs).

Although much research and development energy has been devoted to the creation of AVs, there is a dearth of research into the possible impacts of widespread adoption of this technology. This is significant, because the rapid development of AVs is a reality – Google's Waymo AVs have already driven more than 16 million km on public roads and Japanese Prime Minister Shinzo Abe has expressed that he expects Japan to be exhibiting AVs by the opening of the Tokyo 2020 Olympics (Abe, 2015; Krafcik, 2018). We can soon expect AVs in our cities. The policy decisions governing the introduction of this technology are being made now.

There is a requirement for detailed consideration of the impact of this technology in order to maximise the opportunities presented by it – there is the possibility that AVs could be the answer to mobility sustainability in urban transport. However, this will only be the case if the problems and opportunities are considered holistically. Policy decisions must not be reactionary, and should consider the social, ethical, economic, and environmental consequences of adopting this technology.

1.1 Private Vehicle Transport Today

There are many problems associated with today's private vehicle transport.

1.1.1 Safety

There are 1.35 million road accident fatalities and 50 million injuries annually. In 1990, road traffic injuries were ranked 9th in the WHO rank order of Disability-Adjusted Life Years (DALYs) for the leading causes of the global burden of disease. By 2030, it is predicted that road traffic injuries will have risen to 5th on this list, and this is already the leading cause of death for children and young adults aged 5-29 years, disproportionately affecting lower-income countries (WHO, 2004; WHO, 2018).

1.1.2 Parking

Parking is an immense waste of resources (both land and vehicular). Cars are estimated to be parked some 95 % of the time (Barter, 2013), with vast land area given over to parking – exceeding 10 % of the area of some cities (Chester, et al., 2015).

1.1.3 Mobility

Driving is restricted by license, with access limited by age and ability. As the population of more developed countries continues to age, the impact of age-related impairments inhibiting driving will continue to increase also (Malasek, 2016).

1.1.4 Traffic congestion

Traffic congestion is enormously costly in both productivity and in environmental impact. For example, in 2015 7 billion hours were spent in traffic across the US alone, wasting 11 billion litres of fuel (Schrank, et al., 2015). In 2018, it was estimated drivers in the UK lost an average of 178 hours a year to congestion, costing UK £ 8 billion (INRIX, 2019).

1.1.5 Emissions

Transport is a significant contributor to air pollution and global CO_2 emissions. Research shows that outdoor air pollution contributes to more than 3 million deaths worldwide each year (WHO, 2016). This is particularly relevant in urban environments. In

Tokyo, for example, it is estimated that 20.4 % of all lung cancer deaths are due to diesel particulate matter pollution, compared to a Japanese average of 11.5 % (TMG, 2002). In some cities, transport is estimated as the source of nearly 30 % of all emissions responsible for global warming. The IPCC estimate that one quarter of global CO_2 emissions are attributable to transport (IPCC, 2014).

1.2 Autonomous Vehicles (AVs)

1.2.1 What is an AV?

An AV is a vehicle which makes decisions in traffic based on the rules and constraints of its programming (Maurer, et al., 2016). This requires a high degree of automation, the technology by which a process or procedure is performed without human assistance (Groover, 2010). Six levels of automation are defined by SAE International, where Level 0 indicates no automation and Level 5 indicates full automation (Table 1) (SAE International, 2014).

Table 1 Levels of vehicle automation

Automation	Description
Level 0	No automation – human driver required for all functions
Level 1	“Hands on” – car executes steering or acceleration
Level 2	“Hands off” – more than one function automated, but driver controls others
Level 3	“Eyes off” – car drives itself, but driver must remain attentive
Level 4	“Mind off” – driver can safely perform other activities
Level 5	“Steering wheel optional” – no driver required

A vehicle can be described as autonomous from an automation level of Level 4.

1.2.2 Acceptance of AV technology

There have been a number of surveys regarding AV technologies, with many focusing on awareness and acceptance (concerns and perceived benefits), in addition to willingness to pay for the technology (WTP). These studies were summarised by Becker & Axhausen (2017) and are presented with updates in Table 2.

Table 2 Summary of questionnaire surveys related to AV technology

Author	Year	Type	Target	N	Method
J. D. Power	2012	Report	US (vehicle owners)	17,400	Not found
Cisco Systems	2013	Report	10 countries	1,514	Not found
Continental	2013	Report	4 countries (vehicle users)	4,000	Phone
Silberg et al. (KPMG)	2013	Report	US (experts)	32	Focus groups
Schoettle & Sivak	2014	Report	US, UK, Australia	1,533	Online
Payre, et al.	2014	Journal	France (drivers)	421	Online
Brown et al. (Deloitte)	2014	Report	6 countries (automotive consumers)	23,000	Not found
Ipsos MORI	2014	Report	UK	1,001	Online
Rödel et al.	2014	Proceedings	Austria (staff and students)	336	Online
Seapine Software	2014	Report	US	2,039	Online
Howard & Dai	2014	Proceedings	US (Berkeley, likely adopters)	107	Paper
Kyriakidis, et al.	2015	Journal	109 countries	5,000	Online
Schoettle & Sivak	2015	Report	US (drivers)	505	Online
Bansal, et al.	2016	Journal	US (Austin)	347	Online
Krueger, et al.	2016	Journal	Australia	435	Online
Yamamoto, et al.	2016	Journal	Japan (Nagoya)	803	Online
Zmud, et al.	2016	Report	US (Austin)	556	Online
Bansal & Kockelman	2017	Journal	US	2,167	Online
Hulse, et al.	2018	Journal	UK	925	Online

Opinions regarding AV technology skew slightly positive, with greater trust observed in emerging markets (Bansal, et al., 2016; Cisco Systems, 2013; Hulse, et al., 2018; J. D. Power and Associates, 2012; Payre, et al., 2014; Schoettel & Sivak, 2014). Positive experience with ADAS (advanced driver-assistance systems) increases acceptance of AV technology (Continental, 2013; Rödel, et al., 2014). Interest generally decreases with increasing vehicle autonomy (Brown, et al., 2014; J. D. Power and Associates, 2012;

Kyriakidis, et al., 2015; Rödel, et al.; 2014; Schoettel & Sivak, 2014; Schoettel & Sivak, 2015).

The main perceived benefits include increased safety, convenience, and multi-tasking (Bansal, et al., 2016; Howard & Dai, 2014; Ipsos MORI, 2014; J. D. Power and Associates, 2012). The highest enthusiasm for AV technology was demonstrated for the use cases of highways and traffic (Continental, 2013; J. D. Power and Associates, 2012; Payre, et al., 2014). One study indicated an interest in AV technology for impaired driving (Payre, et al., 2014).

Concerns about the technology are primarily related to cost, failure of the vehicle, and handing over control to the vehicle (Bansal, et al., 2014; Howard & Dai, 2014; Hulse, et al., 2018; J. D. Power and Associates, 2012). Lesser concerns include liability issues, hacking, and personal data protection (Kyriakidis, et al., 2015; Seapine Software, 2014). Experts have fewer concerns over reliability (Continental, 2013). Concern over cost is related to an increased interest in vehicle sharing (J. D. Power and Associates, 2012). Additionally, there is a perceived loss in status associated with AVs by auto-enthusiasts (Ipsos MORI, 2014; J. D. Power and Associates, 2012).

WTP is generally below projected prices for automated features (Bansal & Kockelman, 2017; Kyriakidis, et al., 2015). In one study, 20 % of respondents demonstrated an average WTP of US $ 3,000 (J. D. Power and Associates, 2012). In another, 25 % of respondents expressed an average WTP of US $ 2,020 (Schoettel & Sivak, 2014). In Payre, et al. (2014), 78 % of respondents expressed an average WTP of US $ 1,624.

Demographic influence over responses varies, with gender, age, income, and residence seen as significant in many studies. A higher interest in AV technology is demonstrated by young people (Brown, et al., 2014; J. D. Power and Associates, 2012; Rödel, et al., 2014). Brown, et al. (2014) indicated this preference is due to a decreased interest in driving by young people, who feel vehicle costs are high and lifestyle needs can be met by walking and public transportation. This was echoed by Silberg, et al. (2013). Other predictors of increased acceptance include urban residence, being male, and being a premium vehicle owner (Bansal, et al., 2016; Ipsos MORI, 2014; J. D. Power and Associates, 2012; Rödel, et al., 2014). Howard & Dai (2014) observed that males were more concerned with liability issues, females with handing over control to the AV. Higher income respondents were most concerned about liability, with lower income respondents more concerned about AV failure.

Ipsos MORI (2014) hypothesised that the public is yet to see the benefit of driverless cars, and that interest and WTP will increase once these benefits are more widely realised. This was echoed by Zmud, et al. (2016), who split their sample by intent to use AVs as experts expect the public will tilt towards acceptance once the technology is better known. A similar sentiment was expressed by Bansal, et al. (2016). Silberg, et al. (2013) proposed three possible adoption scenarios for AV technology. Key blockers for adoption indicated included a lack of consumer engagement and a lack of legal and regulatory framework, leading to slow technology breakthroughs on V2X (vehicle to external environment) systems. They proposed that increasing interest and focus on AV technology results from current mobility being expensive and inefficient – vehicles are typically unused 22 hours a day, with hours of lost productivity at the wheel, as well as being unsafe (15 traffic deaths per 100,000 in 2010, 93 % attributable to human error), with younger generations having increasingly less interest in driving (1978: 75 % 17-year olds licensed, 2008: 49 %).

In a choice experiment performed by Krueger, et al. (2016), participants were asked to provide details of a recent trip (purpose, means, distance, travel time, waiting time, cost). They were then given a presentation about SAVs (shared autonomous vehicles). Then, they were asked to perform choice tasks, selecting a transport mode for a specified trip. Travel time, waiting time, and fares were significant determinants of SAV use. A model using survey data created by Zmud, et al. (2016) predicted an increase in VMT and reduction in transit use with the introduction of AV technology.

In a demand forecast and simulation of SAVs, Yamamoto et al. (2016) demonstrated a rebound effect, where an increase in demand for SAVs led to congestion and increased VMT in simulation in Meito-ku, Nagoya City. They also demonstrated a potential for shift from private ownership to shared ownership models, although 50 % of respondents maintained they would prefer to own their own AV.

1.2.3 Status of AV technology

Many of the features described as “autonomous” today (such as Tesla’s autopilot) are actually only ADASs of SAE Level 2, and thus fail the definition described in Section 1.2.1 (Banks, et al., 2018). However, the first Level 4 vehicles are now already in development (Kyriakidis, et al., 2019).

Autonomy is achieved with a combination of sensor-based solutions (cameras, radar, and lidar), synthesised by AI to fully map surroundings and watch for obstacles. Communication-based solutions enhance safety and enable features such as real-time route optimisation, increased lane capacity, and reduced energy consumption. V2I (vehicle to infrastructure) communication involves data exchange with surrounding infrastructure to respond to traffic regulation (e.g. speed limits, traffic signals, etc.). V2V (vehicle to vehicle) communication involves data exchange between vehicles for the prevention of collisions. V2X communication subsumes these, in addition to other communication technologies (to pedestrians, for example) (Weiß, 2011). The achievement of V2X requires, in general, a large public investment in “intelligent” infrastructure and is still in its infancy. Silberg, et al. (2013) highlight that this convergence of sensor-based and connectivity-based solutions is required for the adoption of this technology, with implementation of legal frameworks and retrofitting additionally important.

There is much excitement and speculation about this technology, but when can we actually expect AVs? According to recent studies, the public expects full (Level 5) automation in more than 50 % of vehicles by 2030 (Kyriakidis, et al., 2015). Most experts agree that AVs will be a significant percentage of the fleet by 2040 (Mosquet, et al., 2015).

1.2.4 AVs as a solution to today's transport problems

AVs have been proposed as a solution to these problems, envisioning transport that is adequate, accessible, and has minimised negative environmental impact. For the first three problems discussed in Section 1.1, the improvements AVs offer are clear:

(1) AVs are likely to be far safer than the cars we drive today. Human error has been identified as the critical reason for some 90 % of crashes (NHTSA, 2015), with alcohol involved in more than 30 % of motorist fatalities in the US in 2011 (Anderson, et al., 2014). Although new safety challenges (such as cybersecurity and data protection) may arise as the result of automation, these significant safety issues related to human error can be eliminated.

(2) AVs could significantly reduce or shift parking demand. If a car can drive itself, it can return home once it has completed its drop-off or drive to a parking garage outside the dense urban core where parking is cheaper. Otherwise, in a car sharing model, the vehicle could move directly on to its next trip.

(3) AVs could lead to increased mobility. Those who were previously unable to drive, such as those with impairments that prevent them from driving could all become users of AVs.

However, for the final two problems discussed, the impact of AVs is less obvious:

(4) It is unclear whether AVs will lead to a decrease in congestion. The two thirds of traffic congestion estimated to be caused by traffic accidents and bottlenecks (FHWA, 2004) could be improved with increased safety and traffic programming. However, a rebound effect from the increased attractiveness of driving could

result in a higher number of vehicle users and an increase in the number of vehicle miles travelled (VMT). Preliminary studies challenge the ideas that VMT and congestion will decrease with the introduction of AVs. Estimating SAV demand by simulation in Japan, Yamamoto et al. (2016) showed an increase in VMT. Ma et al. (2017) showed a decrease in vehicle ownership but an increase in vehicle use rate, with no overall reduction in VMT when designing an optimal SAV system. In a demand estimation simulation for AVs in Switzerland, Meyer et al. (2017) predicted favouring of urban sprawl and a negative impact on public transportation. In questionnaire survey on the behavioural impacts of AVs, Zmud et al. (2016) demonstrated a preference for private vehicle ownership over shared vehicle systems, with no change in VMT.

(5) It is likewise unclear whether AVs will lead to a decrease in traffic emissions since emissions are significantly related to VMT. A scientometric and bibliometric review of AV research showed that research categories are dominated by technical disciplines, with low incidence of environmental and social research on AVs (Gandia, et al., 2019).

This technology is worth consideration in cities like Tokyo, where efforts are being made to improve environmental sustainability (Fujita & Hill, 2007) and rapid population ageing is impacting driver safety (Morita & Sekine, 2013), but further investigation is required.

1.3 Research Objectives

1.3.1 Research problem and aim

AVs could be turned into a major positive in improving transport safety, accessibility, and in reducing parking demand. They could also result in improvements in traffic congestion and emissions, but these effects remain unclear.

This research aims to offer insight into the potential impacts of AV technology.

1.3.2 Research questions and expected outcomes

This research will explore the impacts that AVs are likely to have on congestion and traffic emissions:

(1) Will traffic congestion increase or decrease?

(2) Will transport emissions increase or decrease?

(3) Will user satisfaction (utility) increase or decrease?

2 METHODS

2.1 Questionnaire Survey

Online questionnaire survey was used to gain an understanding of user transport mode choice and route choice for use in traffic simulation. Full script of the questionnaire survey administered can be reviewed in APPENDIX A: Questionnaire Survey.

2.1.1 Respondent selection

Respondents were selected as a sample of the population of Saitama, Chiba, and Kanagawa prefectures. Tokyo was excluded from the survey as a population with a higher percentage of drivers was targeted. The sample was selected to reflect the gender and age structure of the surveyed population.

2.1.2 Respondent profile

Respondents were asked to provide information about their age, gender, and location. Additionally, they were asked about their driver's license status and vehicle ownership. To understand mode choice for commuting and shopping trips, respondents were asked about their typical commuting and shopping behaviour.

2.1.3 Mode choice

Transport mode choice was investigated through the questionnaire survey. Respondents were asked to select a mode for a journey of specified distance (0-2.5 km) and purpose (commuting or shopping).

2.1.3.1 demand estimation

Responses were used to estimate demand for different transport modes under different journey conditions (distance and purpose) for traffic simulation.

2.1.4 Route choice

Route choice was investigated through questionnaire survey. Respondents were asked to select one route from three on the basis of route journey times, route distances, and whether or not the route was a "typical" route choice.

2.1.4.1 utility function

Route choice was modelled using discrete choice theory (Ben-Akiva & Lerman, 1985). Route choice was assumed to be made based on utility, with the route with the greatest utility selected. Route utility (U) was assumed to be a function of route journey time (t), distance (d), the number of turns required for the route (n), and a "typical" dummy variable (typ), where c_i are attribute coefficients.

$$U = c_1 t + c_2 d + c_3 n + c_4 typ \tag{1}$$

2.1.4.2 conjoint analysis

A multi-attributed item can be decomposed into specific contributions of each attribute and possibly their interactions. If there are a large number of possible hypothetical alternatives for a given problem, a subset of the possible alternatives can be chosen, for which a number of experimental design methods exist. A full factorial design includes all combinations of the attribute levels (profiles). To reduce the demand on the questionnaire respondent, these profiles can be reduced to an orthogonal set and combined into a balanced incomplete block design (BIBD) (Rao, 2013).

In the route choice section of the survey, three attributes (time, distance, and "typical" dummy variable) with three (10, 15, 20 minutes), three (4, 6, 8 km), and two ("no", "yes") levels respectively resulted in 3 x 3 x 2 = 18 profiles. These were reduced to an orthogonal set of 9 uncorrelated profiles and combined into a BIBD of 12 blocks combining 3 profiles, with each combination of profiles appearing once within the set of blocks.

These 12 blocks (choice sets) were administered as 12 questions, with each route choice depicted on a grid map and the time and distance attributes explained with associated average velocity.

2.1.4.3 parameter estimation

Using the number of responses preferring each of the routes presented, the attribute coefficients of the utility function were estimated using a multinomial logit model.

First, the estimated utility function for each route choice was calculated using Equation 1 with an assumed value of $c = 1$. The utility of each route was then exponentiated to obtain the estimated choice probabilities, P.

$$P = \frac{e^U}{\sum e^U} \tag{2}$$

The coefficients were then estimated with maximum likelihood estimation, i.e. by maximising the log-likelihood. The log-likelihood (L) was determined as the sum of the utility of each route (with a normalisation factor subtracted) multiplied by the number of respondents who chose that route (n).

$$\operatorname{Log} L = \sum n(U - \operatorname{Log} e^U) \tag{3}$$

The t-values of the attributes where then estimated to determine the significance of the choice experiment values.

2.2 Traffic Simulation

Traffic simulation was used to gain an understanding of traffic congestion under various scenarios of AV adoption. The use of AVs for short distance trips was explored, to the supermarket, and to the station (last mile). The application of AV technology to short distance journeys as an egress mode for multimodal trips is an interesting use case in addition to as a replacement for conventional passenger cars (Yap, et al., 2016; Moorthy, et al., 2017). In order to simulate traffic, an artificial map was devised, and trips were planned between various origins and destinations. Object-oriented programming (Ruby) was used to create the simulation.

2.2.1 Simulation map

The simulation map was conceptualised as a grid of road segments connecting nodes. Segment length was set to 500 m. Four origins were placed on a “residential” street on one side of the map, and two destinations (the station and the supermarket) were placed on the other side (Figure 1).

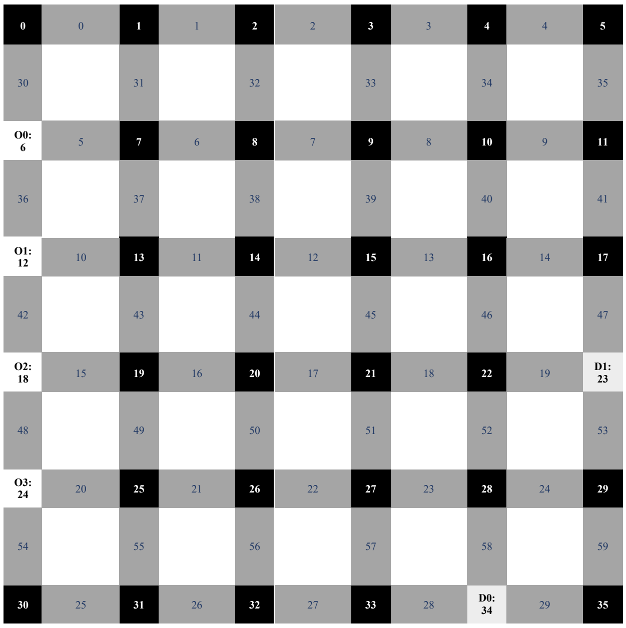


Figure 1 Simulation grid showing 60 segments (0-59) and 36 nodes (0-35) with the locations of four origins at nodes 6, 12, 18, and 24, and two destinations at nodes 34 (station) and 23 (supermarket)

2.2.2 Velocity

Velocity (v) for vehicles (cars, taxis, buses, and AVs) was assumed to be a function of the density of vehicles on the road segment (n), the length of the vehicle (plus safety distance) (L), and the safety "headway time" (spatial interval between vehicles) (T) required between the vehicle and the vehicle in front, as shown in Equation 4 (Friedrich, 2016).

$$v = \frac{1}{T}\left(\frac{1}{n} - L\right) \tag{4}$$

The length of cars was assumed to be 7.5 m, including a 3 m safety distance, and the length of buses was assumed to be 17 m, including a 3 m safety distance. The headway time of conventional vehicles on urban roads was assumed to be 2 s, and the headway time of AVs on urban roads was assumed to be 0.5 s. This relationship is illustrated in Figure 2.

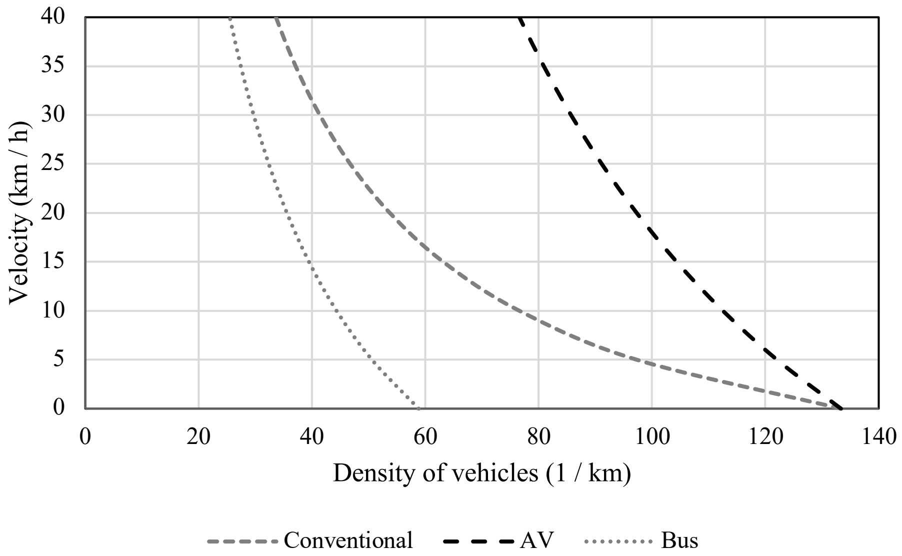


Figure 2 Relationship between vehicle density and velocity on urban roads for conventional vehicles (cars), AVs, and buses

In practice in simulation, a maximum velocity (speed limit) of 40 km / h and a minimum velocity of 5 km / h were assumed limits for the equation.

The velocity of a vehicle (car, taxi, bus, or AV) on a road segment was then determined by the density and modal mix of vehicles on that road segment.

The velocity of bicycles and motorbikes were assumed to be constant, with $v_{bicycle}$ = 14 km / h and $v_{motorbike}$ = 30 km / h.

2.2.3 Travel time

Base travel time (t) was assumed to be a simple function of route distance (d) and mode velocity (v), plus an intersection waiting time of 20 s at each node. Additionally, some modal modifiers were assumed.

Travel time by bus was assumed to include the waiting time for the next bus according to the bus timetable in addition to the base travel time. Three bus routes were devised on the map. The first bus route was implemented from origin 0 to destination 0, via origins 1, 2, and 3. The second bus route was implemented from origin 3 to destination 1, via origin 3. The third bus route was implemented from origin 0 to destination 1, via origins 1 and 2. The timetable of each bus was randomised, with buses departing approximately every 25 minutes and stopping at each origin en route.

Travel time by car was assumed to include the driver's time if the destination was the station (assuming a driver dropping a passenger off to take the train), and parking time if the destination was the supermarket.

Travel time by taxi was modified with penalty factor to reflect waiting time and additional cost associated with this mode.

2.2.4 Mode choice

Mode choice was determined using demand estimation obtained from the questionnaire survey as described in Section 2.1.3.1. In the case of demand shift due to the introduction of AV technology (AV scenario), travellers in the simulation were given the opportunity to change from their initial mode to AV if travel time by AV was shorter than the travel time by their initial mode (true travel time in the "full knowledge" case, assumed travel time in the "imperfect knowledge" case) (decision-making cases are defined in Section 2.2.7 and simulation scenarios in Section 2.2.8).

2.2.5 Route choice

Possible routes were defined as all direct routes between the origin and destination, with travel only possible towards the destination (i.e. if origin is to the east and north of the destination, travel is only possible to the west and south).

Route choice was determined using the utility function described in Equation 1. For the "imperfect knowledge" case, the utility of each possible route between the origin and destination of the trip was calculated using the "typical" travel time of that route (using a base velocity of 30 km / h), the distance of that route (with a segment length of 500 m), the number of turns required for that route, and a random "typical" route dummy variable. The chosen route was then the one with the maximum utility. For the "full knowledge" case, the true travel time of that route was used.

2.2.6 Emissions calculation

Fuel efficiency for ICE (internal combustion engine) vehicles is a curved function of velocity – at lower speeds, fuel efficiency is lower, increasing to a peak around 70 km / h, then tapering again at speeds exceeding this (Sobrino, Monzon, & Hernandez, 2016). At low speeds (under 40 km / h), however, the fuel efficiency curve can be observed as a linear function. Using Japanese government data on fuel consumption at different velocities, an approximate equation for this line can be obtained (Agency for Natural Resources and Energy, 2012) (Figure 3).

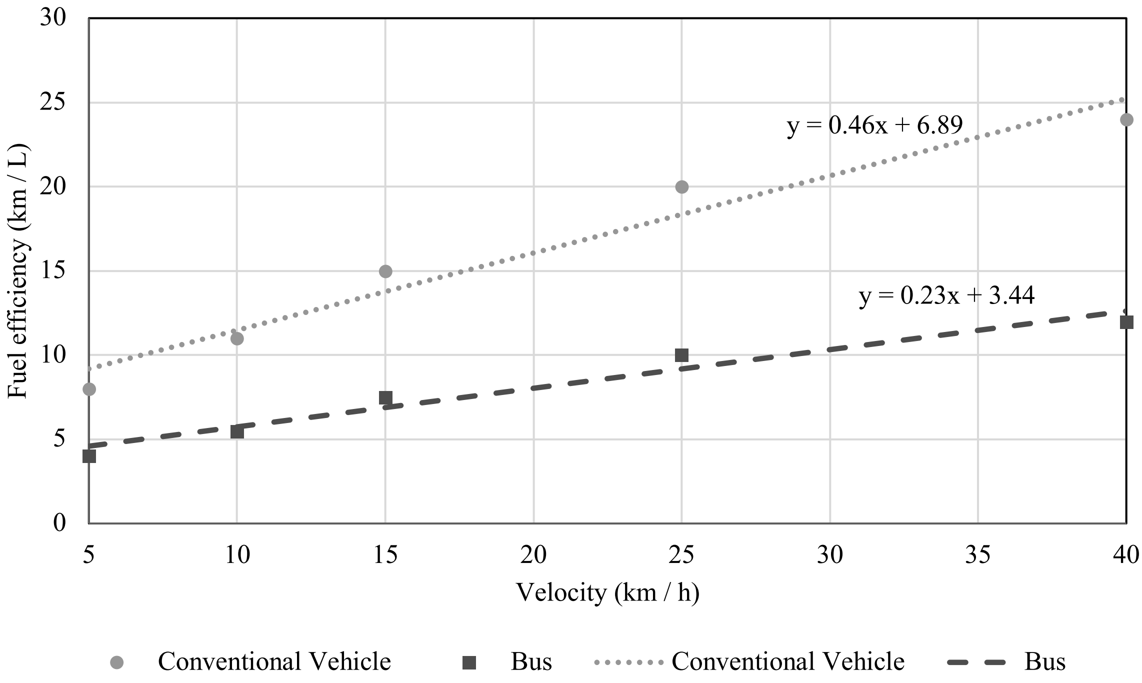


Figure 3 Fuel consumption trend at low velocities for conventional vehicles (cars) and buses

For electric vehicles (EVs), fuel consumption was assumed in simulation to be approximately 7 km / kWh (EV Database, 2019).

To determine emissions, fuel consumption in litres or kilowatt hours can be converted to kilograms of CO_2 using the conversion factors 2.61 kg / L of CO_2 for petrol, and 0.41 kg / kWh of CO_2 for grid electricity (Carbon Trust, 2016).

In simulation, cars, taxis, partially automated AVs and buses were assumed to be ICE vehicles, with fully automated AVs assumed to be EVs.

2.2.7 Decision-making cases

Two decision-making cases were implemented in the simulation for route choice and mode choice demand shift:

(3) "Imperfect knowledge" decision-making

For this decision-making case, imperfect knowledge about traffic conditions by travellers was assumed. Travel time for the route choice and mode choice demand shift calculations was estimated using a “typical” velocity of 30 km / h, rather than the true travel time (for cars, taxis, buses, and AVs). This represents a realistic picture, where route choices are made based on typical traffic conditions, without detailed knowledge about the current specific traffic conditions.

(4) “Full knowledge” decision-making

For this decision-making case, full knowledge about traffic conditions by travellers was assumed. True travel time was used for the route choice and mode choice demand shift calculations. This represents a more technologically-advanced picture where traffic conditions can be accurately known and predicted.

2.2.8 Simulation scenarios

Three simulation scenarios were considered:

(4) Base scenario

Here demand shift to AVs was assumed to be 0 % (no AV technology), and the headway time (spatial interval between vehicles) for conventional vehicles was assumed to be the standard 2 s.

(5) Partially automated AV 100 % adoption scenario

Here a partially automated AV scenario was assumed, where a driver was still required, but headway time for these partial AVs was assumed to be 0.5 s.

(6) Fully automated AV 100 % adoption scenario

Here demand shift to AVs was assumed to be 100 % (full AV technology), with 0.5 s headway time for AVs.

2.2.9 Simulation algorithm

The simulation algorithm is summarised in Figure 4.

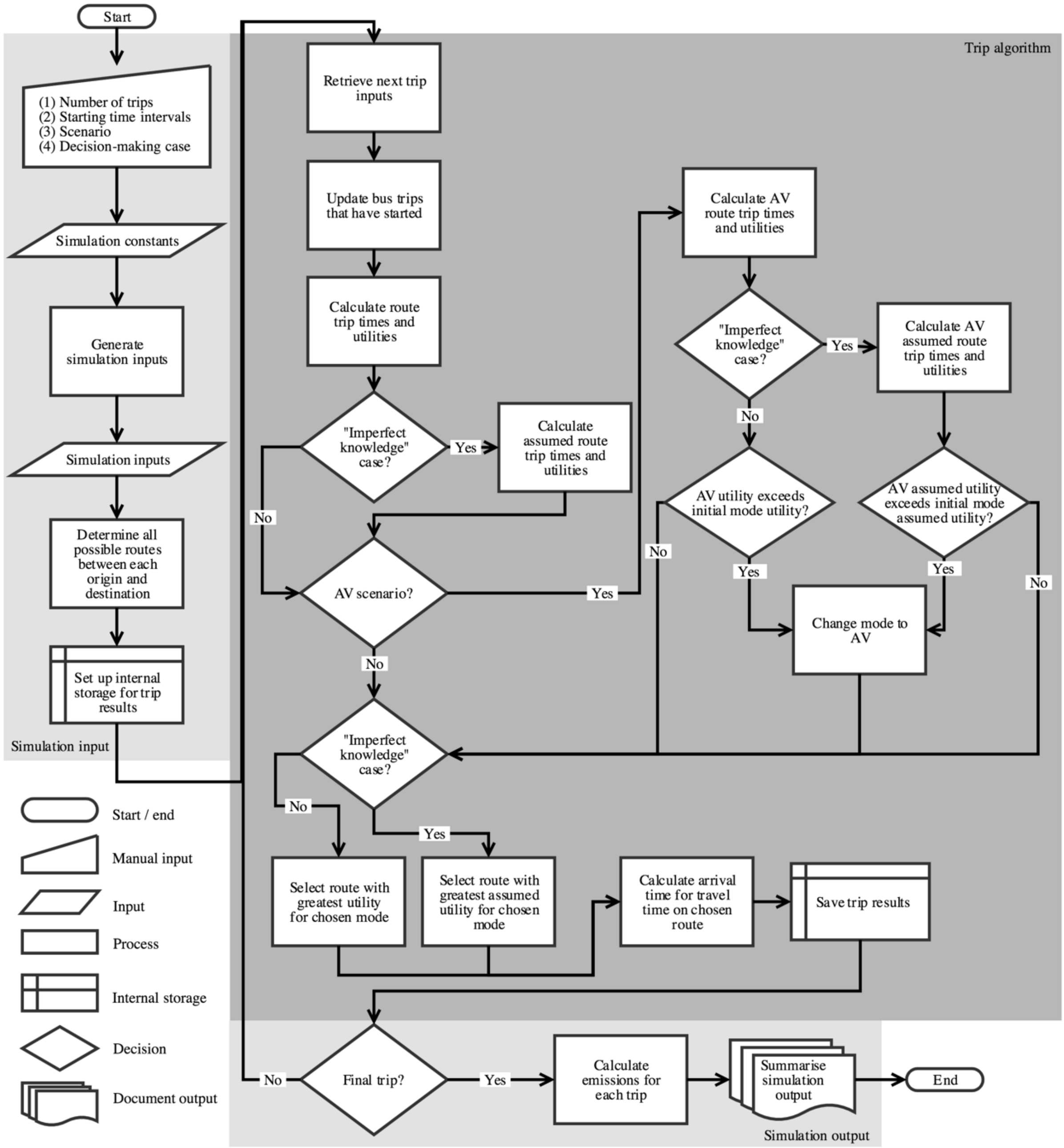


Figure 4 Summary of simulation algorithm

Simulation constants include simulation map and segment length (see Section 2.2.1), automobile length, bus length, conventional vehicle headway time, AV headway time, speed limit, minimum speed, motorbike velocity, and bicycle velocity (see Section 2.2.2), parking penalty, taxi penalty, intersection wait, bus routes, bus frequency (see Section 2.2.3), and

CO_2 conversion factors (see Section 2.2.6), demand shift (scenario-based, see Section 2.2.8) and the number of trips. For each simulation run, 1000 trips were generated.

Simulation inputs include start times, origins, destinations, mode choices, and route dummy variables are generated. Start times were generated successively with a starting time interval of 0.1 s. Origins, destinations, and route dummy variables were generated randomly. Mode choices were generated using demand estimation obtained from questionnaire survey responses, described in Section 2.2.4.

The travel time, route choice, emissions, assumed travel time calculations, and modal shift criteria are described in Section 2.2.3, Section 2.2.5, , Section 2.2.6, Section 2.2.7, and Section 2.2.8 respectively.

Full simulation code can be reviewed in APPENDIX B: Simulation Code.

3 RESULTS

3.1 Questionnaire Survey

Questionnaire survey was used to gain an understanding of user transport mode choice and route choice for use in traffic simulation.

3.1.1 Respondent profile

309 responses were received from residents of Kanagawa (48.2 %), Chiba (24.9 %), and Saitama (26.9 %) prefectures. Respondents ages aligned to an acceptable degree with the demographic structure of these prefectures (Figure 5).

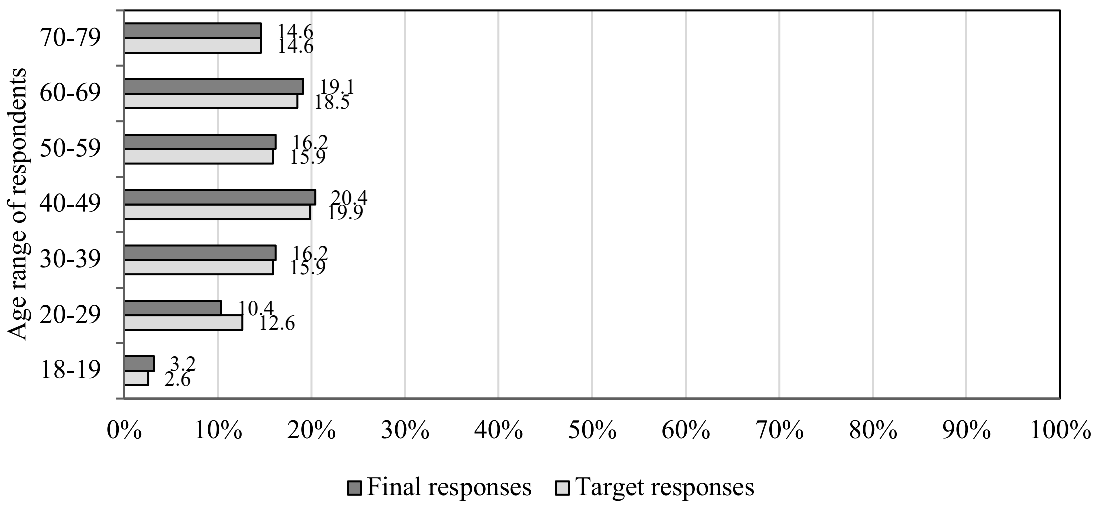


Figure 5 Age ranges for questionnaire survey respondents, including target and final number of responses

Respondents reported their employment status as company employees (32.4 %), public sector employees (1.0 %), self-employed (6.5 %), students (3.9 %), part-time employees (7.4 %), homemakers (29.4 %), and other (19.4 %). 31.1 % of respondents stated their household owns no vehicle, whereas 59.5 % stated their household owns a single vehicle, and 9.4 % stated their household owns more than one vehicle. 78.6 % of respondents reported possessing a driver's license.

3.1.2 Commuting and shopping behaviour

Asked about their commuting activity, 47.2 % of respondents stated that they typically commute to work or school. Of these respondents, 65.8 % stated their typical commute length to take an hour or less (full responses shown in Figure 6).

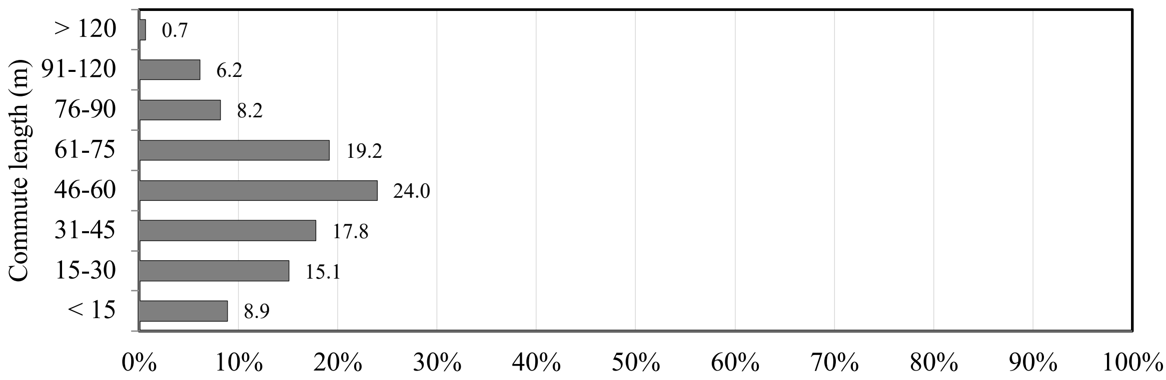


Figure 6 Commute length of questionnaire survey respondents who typically commute (146 of 309 respondents)

Of these respondents, most (55.5 %) reported train or train plus bicycle or walking as their typical commuting mode. 14.4 % of remaining respondents commuted by car, 11.6 % by bicycle or walking alone, and 4.1 % by motorbike. The remaining 14.4 % of respondents reported some combination of the above modes, bus, and taxi (full responses shown in Figure 7 and Figure 8).

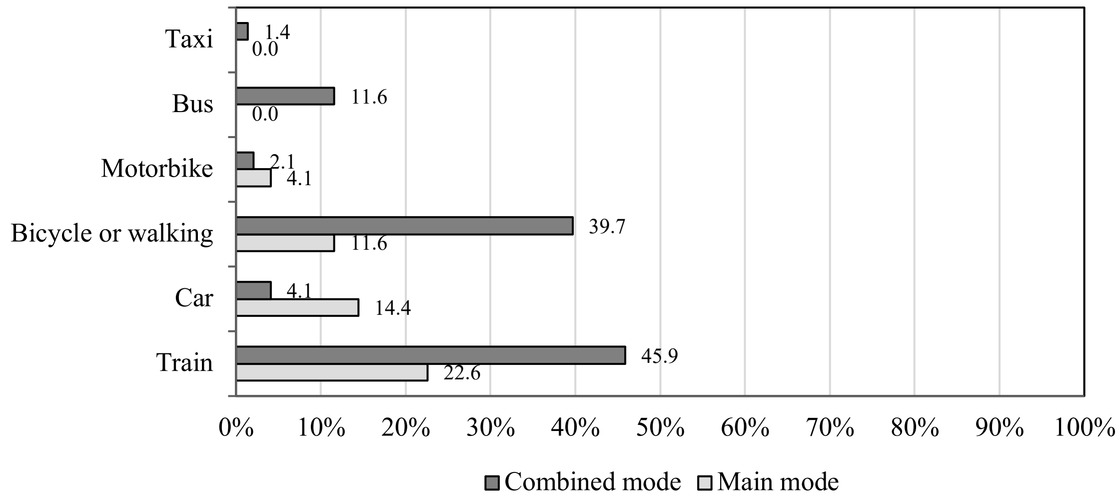


Figure 7 Commuting mode of questionnaire survey respondents who typically commute (146 of 309 respondents), including (1) the number of these respondents who indicated a mode as their typical main commuting mode (77 of 146 respondents), (2) the number of these respondents who indicated a mode as their typical commuting mode combined with others (69 of 146 respondents)

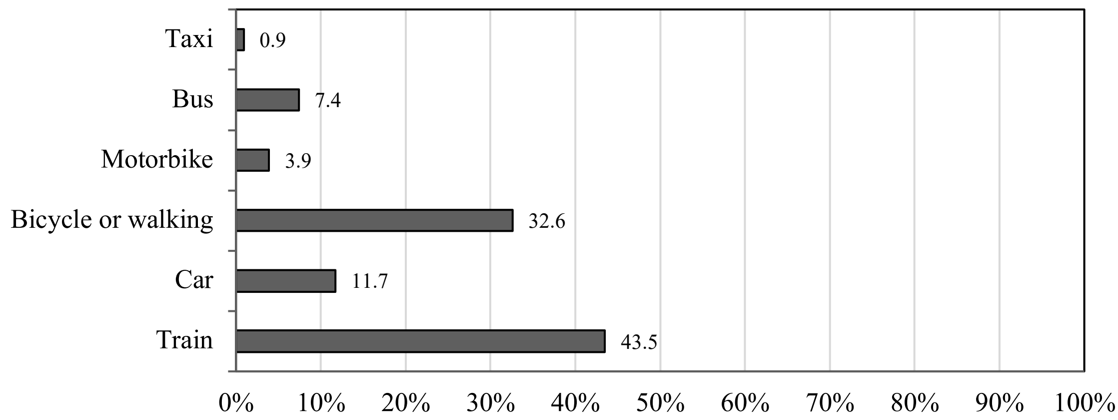


Figure 8 Commuting mode of questionnaire survey respondents who typically commute (146 of 309 respondents), including (3) the number of these respondents who indicated a mode as a mode they use for commuting (230 responses total)

Asked about their shopping activity, 80.3 % of respondents stated that they typically go to the supermarket more than once a week. Of these respondents, 60.8 % stated their typical trip to the supermarket takes 10 minutes or less (full responses shown in Figure 9).

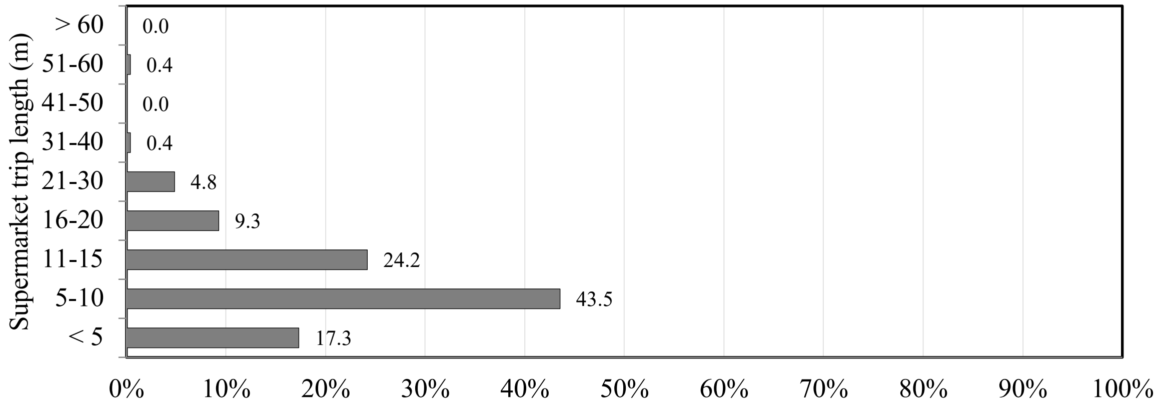


Figure 9 Supermarket trip length of questionnaire survey respondents who typically go to the supermarket more than once a week (248 of 309 respondents)

Of these respondents, most (54.8 %) reported bicycle or walking as their typical supermarket trip mode. 27.8 % of remaining respondents went to the supermarket by car, 1.2 % by motorbike, and 0.4 % by train. The remaining 15.7 % of respondents reported some combination of the above modes, train, bus, motorbike, and taxi (full responses shown in Figure 10 and Figure 11).

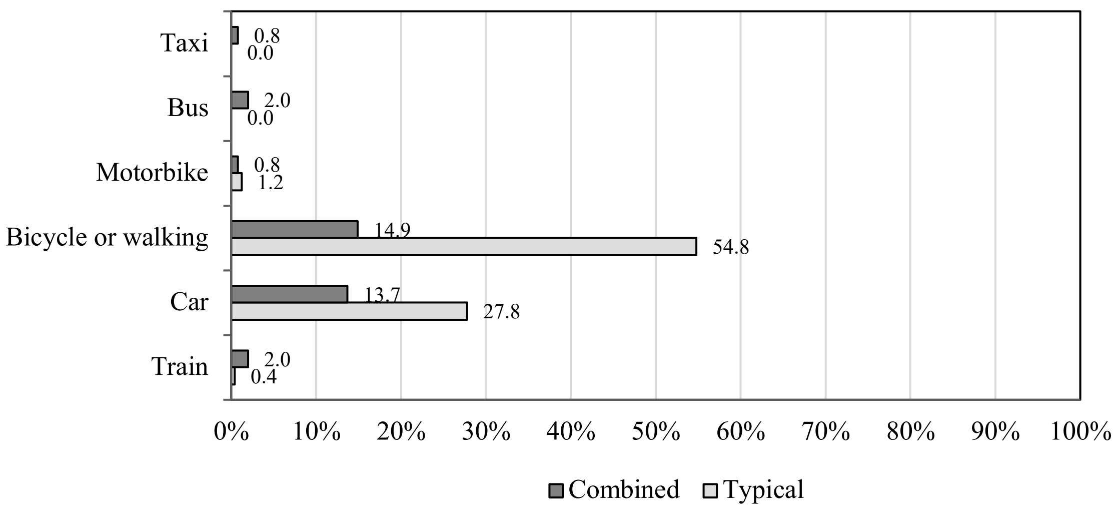


Figure 10 Supermarket trip mode of questionnaire survey respondents who typically go to the supermarket more than once a week (248 of 309 respondents), including (1) the number of these respondents who indicated a mode as their typical main supermarket trip mode (209 of 248 respondents), (2) the number of these respondents who indicated a mode as their typical supermarket trip mode combined with others (39 of 248 respondents)

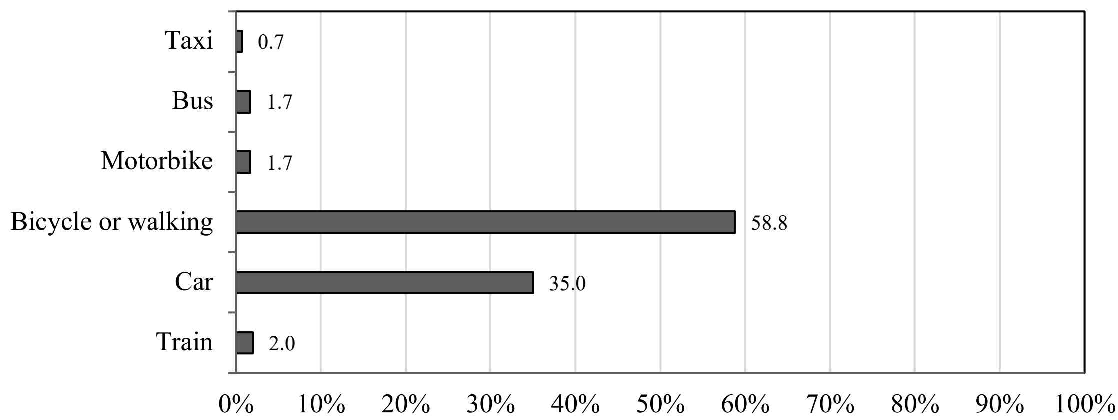


Figure 11 Supermarket trip mode of questionnaire survey respondents who typically go to the supermarket more than once a week (248 of 309 respondents), including (3) the number of these respondents who indicated a mode as a mode they use for supermarket trips (294 responses total)

3.1.3 Mode choice

Respondents were asked to select a mode for a commuting trip from home to the station of a specified distance. Over short distances bicycle or walking was the strongly preferred mode, with car and bus becoming preferred for distances over 2 km. Full responses are shown in Figure 12.

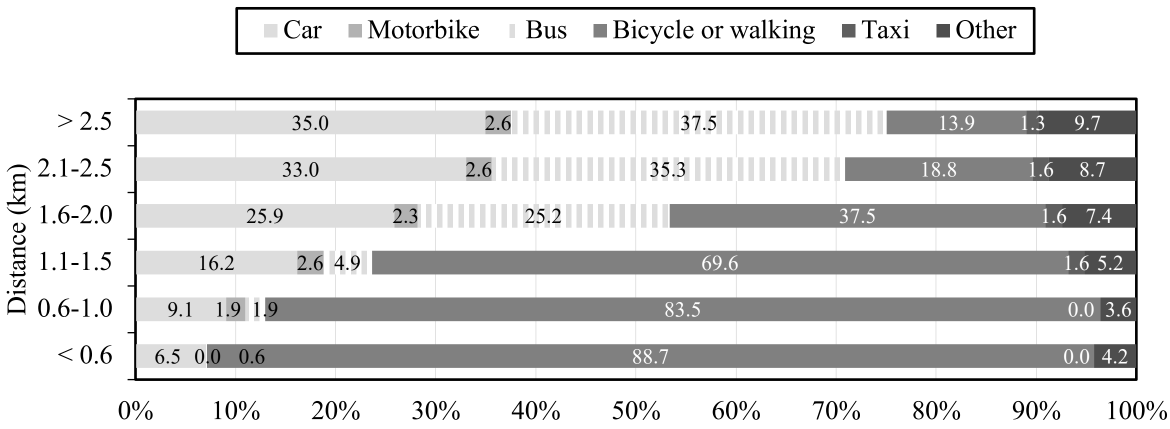


Figure 12 Mode choice for a trip from home to the station of a given distance

Then, respondents were asked to select a mode for a shopping trip from home to the supermarket of a specified distance. Over short distances bicycle or walking was the strongly preferred mode, with car becoming preferred for distances over 1.5 km. Full responses are shown in Figure 13.

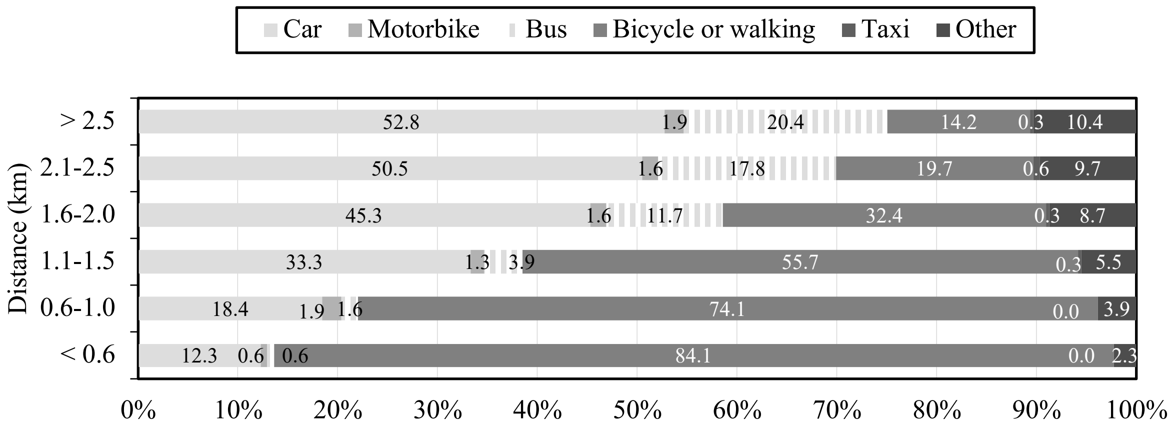


Figure 13 Mode choice for a given trip from home to the supermarket of a given distance

3.1.3.1 demand estimation

Responses described in Section 2.1.3 were directly used to estimate demand for different transport modes for short trips to the station or the supermarket. These demand estimations were then used in traffic simulation.

3.1.4 Route choice

Respondents were asked to select one route from three on the basis of route journey times, route distances, and whether or not the route was a “typical” route choice.

3.1.4.1 conjoint analysis

The full set of choices was determined from three attributes with three, three, and two variable values respectively, creating a full choice set of 18 combinations. An orthogonal set of nine profiles of these combinations was determined. Then, a BIBD of 12 questions consisting of combinations of these nine profiles was created. The number of responses for each route for each question are shown in Figure 14.

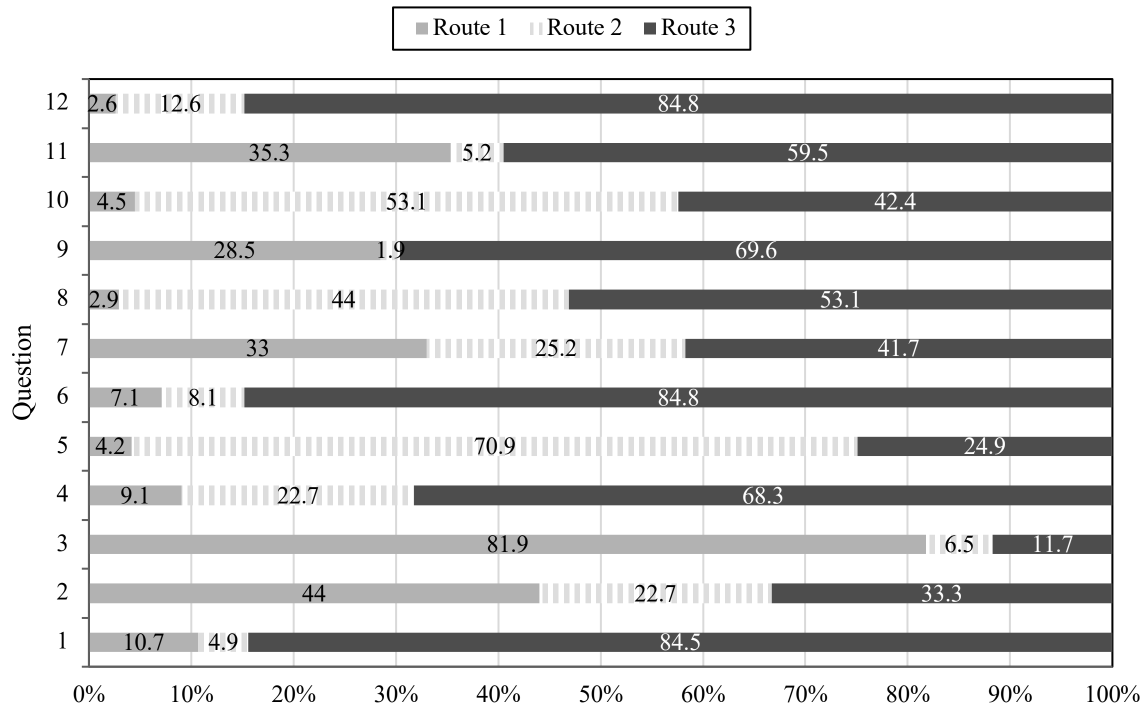


Figure 14 Route choice for a given trip from home to the station of given distances and travel times

3.1.4.2 Parameter estimation

Using the number of responses preferring each of the routes presented, the attribute coefficients of the utility function were estimated using a logit model.

Coefficients and associated t-values are shown in Table 3.

Table 3 Estimated coefficients and t-values for utility function

Attribute	Coefficient value	t-value
Time	- 0.22	- 13.2
Distance	- 0.29	- 4.2
Typical route dummy variable	0.65	7.3
Number of turns	- 0.15	- 3.0

3.2 Traffic simulation

Traffic simulation was used to understand the impact of AV technology on congestion, traveller utility, and transport emissions.

Two decision-making cases were implemented:

(1) “Imperfect knowledge” decision-making

(2) “Full knowledge” decision-making

Within both of these decision-making cases, three scenarios were examined:

(1) Base scenario

(2) Fully automated AV 100 % adoption scenario

(3) Partially automated AV 100 % adoption scenario

3.2.1 Traffic congestion

In the AV scenario, private vehicles were selected over lower-impact modes (such as bicycles) and public transportation (buses). The total number of vehicles of each mode that were selected in each decision-making case are shown in Figure 15 and Figure 16.

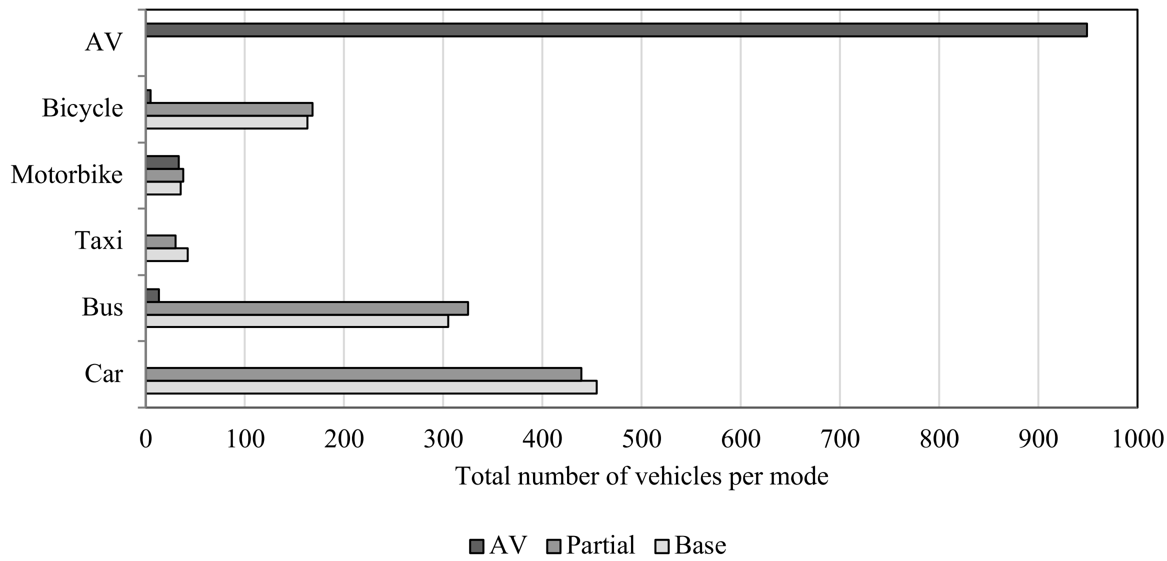


Figure 15 Total vehicles selected in 1000 trips in each scenario for the "full knowledge" decision-making case

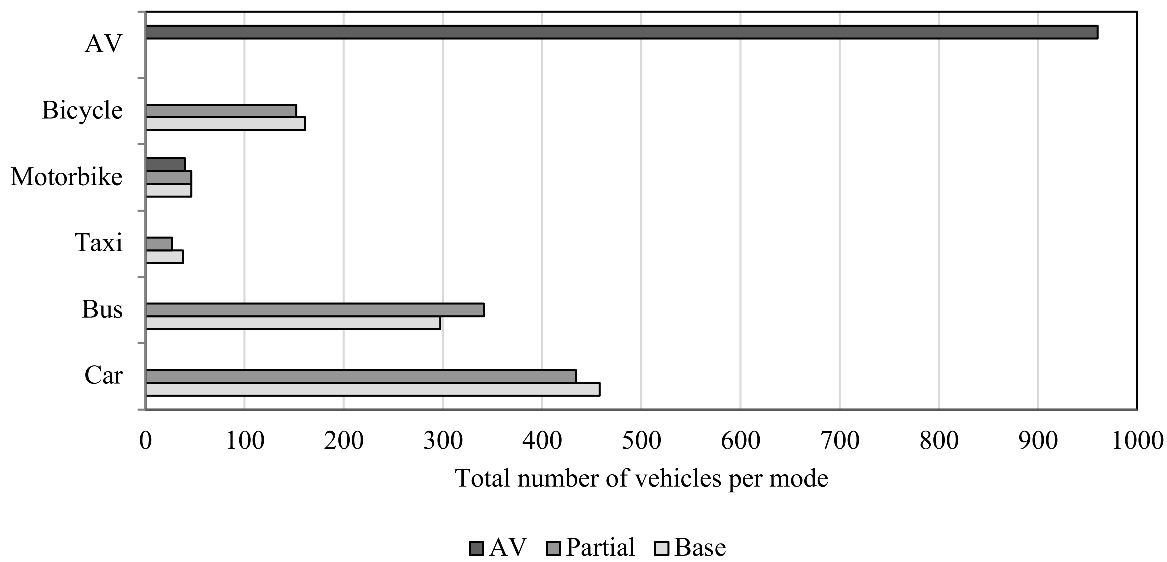


Figure 16 Total vehicles selected in 1000 trips in each scenario for the "imperfect knowledge" decision-making case

The number of higher-impact vehicles (cars, taxis, buses, AVs) on the road almost doubled in the AV scenario in both decision-making cases (Figure 17).

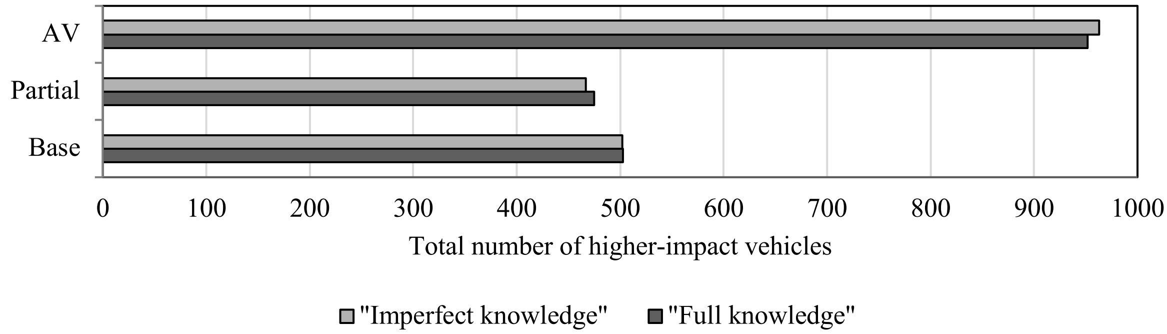


Figure 17 Total higher-impact vehicles (car, taxi, bus, AV) selected in 1000 trips in each scenario for the two decision-making cases

This was due to the increased attractiveness of AVs – no parking time, no driver requirement, and smaller headway time – also understood as a rebound effect. This was not observed in the partial scenario, as in this case parking and a driver were still required, and demand shift was not observed. A slight increased demand shift to AV can be observed in the "imperfect knowledge" case. This is due to a slightly increased shift from bicycle and bus to AV without full knowledge of traffic congestion.

This was directly reflected in the VMT of higher-impact vehicles (Figure 18).

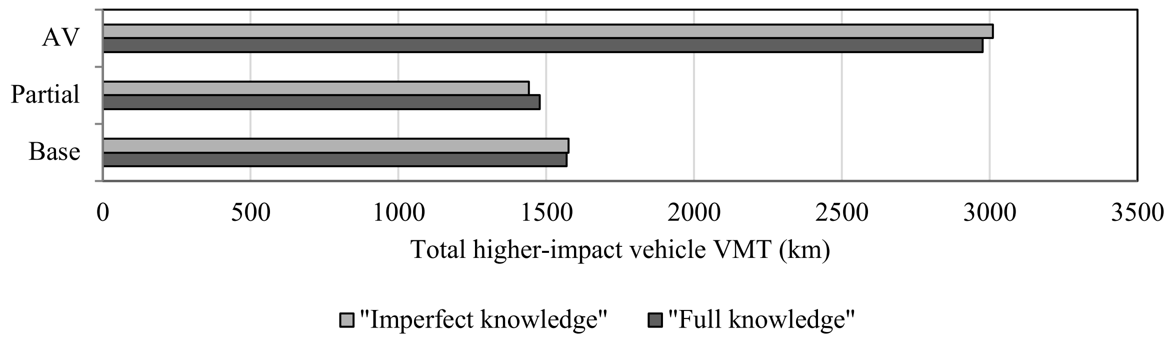


Figure 18 Total VMT for higher-impact vehicles (car, taxi, bus, AV) in 1000 trips in each scenario for the two decision-making cases

Average velocity across the AV scenario was similar to the base scenario, with a slight increase in the partial scenario (Figure 19).

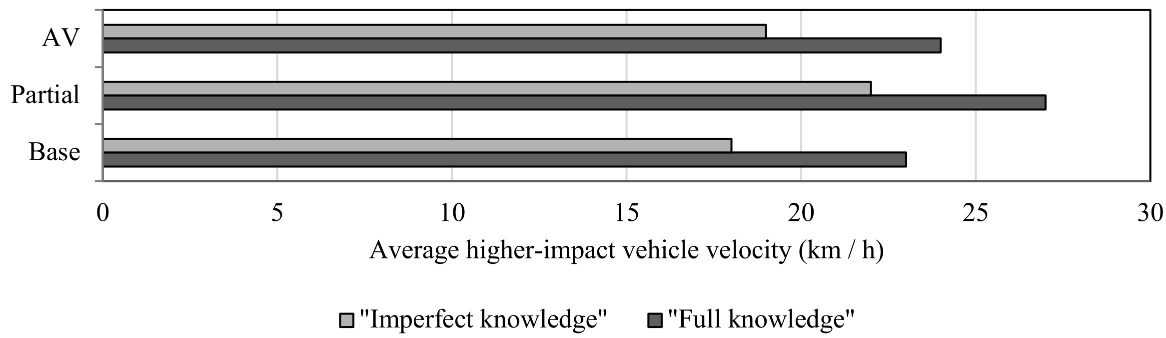


Figure 19 Average velocity for higher-impact vehicles (car, taxi, bus, AV) in 1000 trips in each scenario for the two decision-making cases

This is due to the rebound effect effectively balancing the increased speed of AVs (smaller headway time) in the AV scenario.

Average velocity across the board was lower in the "imperfect knowledge" case – this was due to route choice being less optimised with lack of knowledge about traffic congestion, leading to greater congestion.

To examine congestion due to AV introduction, the average velocity change from low-congestion to high-congestion stages of the simulation was observed (Figure 20).

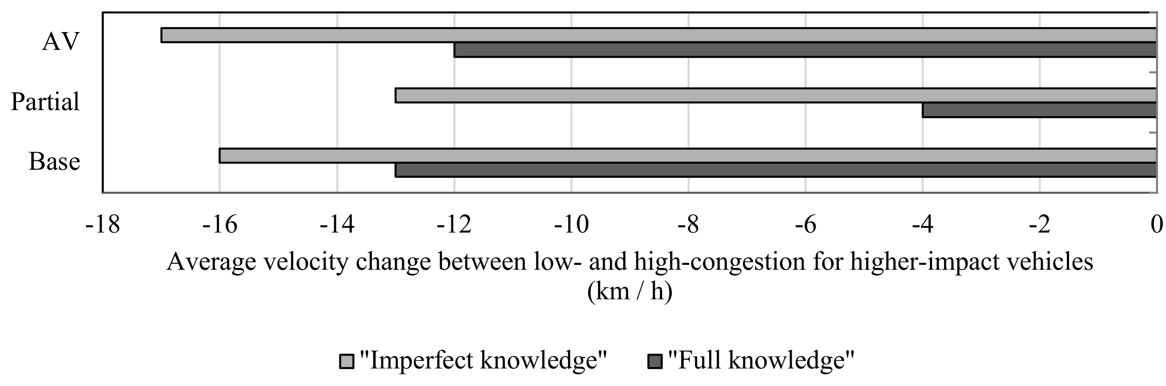


Figure 20 Average velocity change for higher-impact vehicles (car, taxi, bus, AV) in 1000 trips in each scenario for the two decision-making cases

Low-congestion was observed as velocity at the first trip of that mode in the simulation where the number of vehicles on the road was low, and high-congestion as the last trip of that mode in the simulation where the number of vehicles on the road was high.

Here, increased congestion can be observed in the AV scenario of the "imperfect knowledge" case. This can be interpreted as a greater reduction in velocity due to the demand shift from lower-impact modes to AVs. This is not observed in the "full knowledge" case, however, likely due to greater route choice optimisation and the effect of the increased speed of AVs balancing the increased demand in this case.

3.2.2 Traveller utility

Traveller utility can be interpreted to increase in the AV scenario due to decreased travel time in both decision-making cases (Figure 21 and Figure 22).

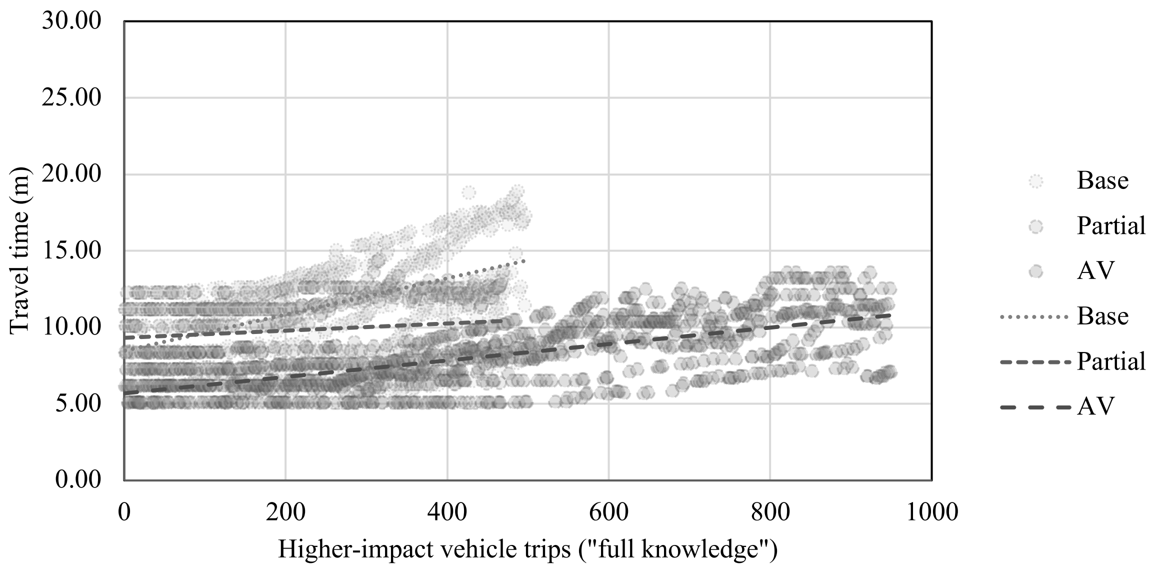


Figure 21 Travel time distribution for higher-impact vehicles (car, taxi, bus, AV) in 1000 trips in each scenario for the "imperfect knowledge" decision-making case

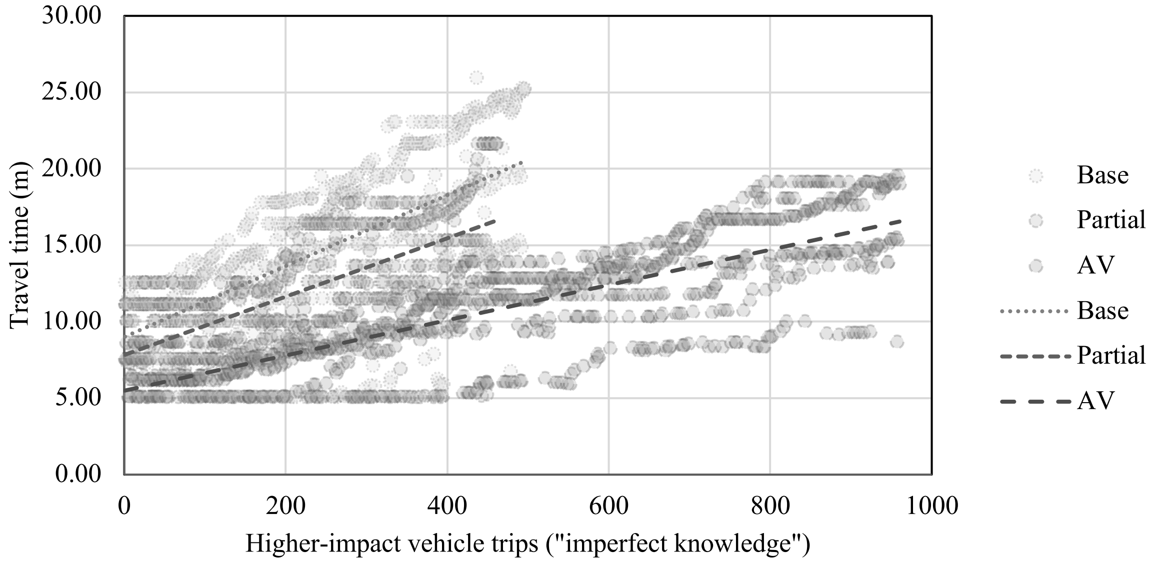


Figure 22 Travel time distribution for higher-impact vehicles (car, taxi, bus, AV) in 1000 trips in each scenario for the “full knowledge” decision-making case

However, this effect can be seen to begin to balance over time, since the number of trips by higher-impact vehicles doubles in the AV scenario. This leads to an increase in travel time over time in the AV scenario compared to the partial scenario in the “full knowledge” case.

It can also be interpreted to decrease in the “imperfect knowledge” case due to the increased congestion demonstrated in Figure 20.

3.2.3 Transport emissions

Transport CO_2 emissions can be observed to be reduced by a similar amount in both the AV and partial scenarios in both decision-making cases (Figure 23).

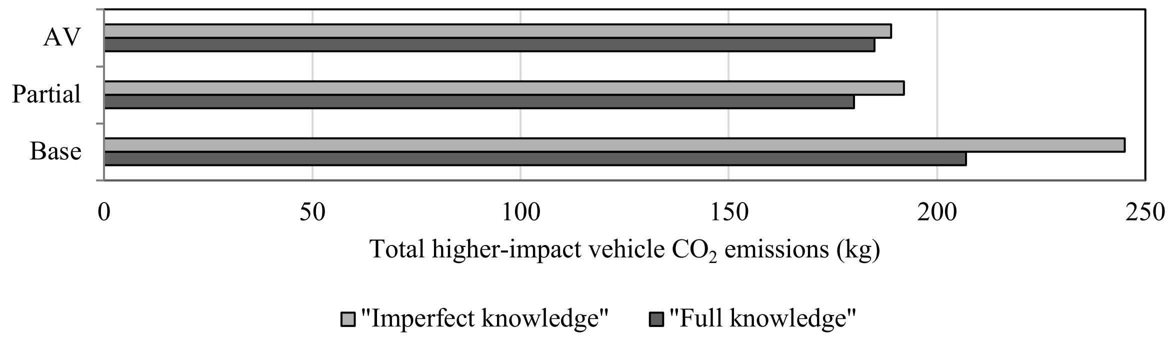


Figure 23 CO_2 emissions for higher-impact vehicles (car, taxi, bus, AV) in 1000 trips in each scenario for the two decision-making cases

In the AV scenario, emissions are reduced due to shift to electric powertrain, but this effect is being balanced by the large demand shift to AVs. In the partial scenario, emissions reduction is due to the increased velocity of vehicles due to the reduced headway time.

The increased emissions observed in the “imperfect knowledge” case are due to the increased congestion observed in this case.

4 DISCUSSION

AV technology holds much promise for improving the nature of private vehicle transport, providing a safer service than the vehicles we use today, in addition to increased mobility due to the removal of licensing barriers. Additional claims that the technology will have a positive effect on traffic congestion and transport emissions requires further examination, and consequently has been evaluated using questionnaire survey and traffic simulation in this study.

4.1 Smaller, Lighter, Faster?

The design and regulation of AVs is likely to be directly affected by the increased safety of this technology. The design of vehicles is heavily influenced by the danger of driving, with larger, heavier vehicles preferred for improved safety in the event of a collision (if only for the driver). Speed limits are similarly informed by safety concerns. With improved safety, the size, weight, and speed of vehicles can be optimised for outcomes other than safety – congestion and emissions reductions among them. However, there is the expectation that the extra sensor equipment required by AVs could lead to increased weight, drag, and power use.

In simulation, this effect was explored with the variation of minimum "headway time", i.e. the shortest time between vehicles on a road without the reduction of speed, typically a consideration of physical route capacity and vehicle safety. Assuming increased safety and no need to consider driver reaction time, this time can be reduced (Iglínski & Babiak, 2017). This effect was explored alone in the partially automated AV case, and in combination with other AV effects in the fully automated AV case.

The reduction of headway time resulted in a significant reduction in congestion and emissions.

4.2 Rebound Effect

Emissions were also reduced in the fully automated AV case. This was due to the substitution of ICE vehicles for EVs implemented in this scenario, which produce fewer emissions. While it is not a given that AVs will be electric vehicles, with many countries now legislating against the continued use of ICE vehicles in the future, it can be seen to be likely.

However, increased VMT was observed – a rebound effect due to the increased attractiveness of AVs. The enhanced route utility for this mode resulted in a demand shift away from conventional vehicles, but also from public transport buses, and lower impact modes such as bicycles. While the simple model of utility explored in this simulation doesn't account for the full spectrum of mode choice attributes, it can be interpreted that the increased attractiveness (due to effects quoted in acceptance surveys such as improved safety and multi-tasking) could likewise result in a rebound effect leading to an overall increase in VMT. Another possible source of VMT increase would be the reduced burden of driving resulting in urban sprawl, and longer commutes by private vehicle.

In simulation, the rebound effect balanced with the reduced emissions due to the electric powertrain – resulting in a similar emissions reduction to the partially automated AV case which only considered reduced headway time.

This shows that technological improvements in safety alone, without other features of automated and autonomous driving can have a significant role to play in the reduction of CO_2 emissions and should be prioritised by producers and policymakers.

Additionally, it shows that the substitution for EVs is extremely important for the reduction of CO_2 emissions. Given demand shift to AVs, emissions improvements achieved with design changes discussed in Section 4.1 could be negated without the use of electric powertrain (Reichmuth, 2018). Continued focus on shifting powertrain from combustion to electricity should be kept by policymakers, with special emphasis in the case of AVs.

4.3 Knowing Traffic

Two cases of decision-making were explored. Firstly, a case where decisions were made based on the “typical” traffic conditions of the area, but imperfect understanding of the true traffic conditions, and secondly, a case where route choice and demand modal shift were made with full knowledge of the traffic conditions of the area. The first case reflects the current situation, where trips are often started without detailed information about traffic conditions. The second case reflects a future vision, where V2X systems can allow the AV up-to-date and accurate information about traffic conditions.

Congestion, emissions, and travel times all increased in the first case as compared to the second case. This was due in the main to a lack of route-optimisation. Additionally, traffic congestion slightly increased with the introduction of AVs in the first case, whereas it slightly decreased in the second case. This was due to a greater demand shift from lower-impact modes to AVs in this case.

These simple explorations of decision-making show that better information can impact traffic and emissions, even in the absence of other innovations. This provides a business case for governments for investment in V2X systems.

4.4 Reducing Zero-Occupancy Trips

A feature of AV introduction is the advent of zero-occupancy trips where empty vehicles contribute to traffic. As AVs are more attractive as a result of not requiring a driver, a greater number of travellers will select this option. Although not fully explored in simulation, the significant demand shift away to AVs in the AV scenario would result in an increased number of vehicles on return journeys.

Disincentivising zero-occupancy trips should be a priority for policymakers, with approaches including a preference for shared vehicle implementations or multi-occupant vehicles (such as carpool lanes in use in some cities).

5 CONCLUSIONS

AV technology has the potential to greatly alter private vehicle transport today. In safety, mobility, and parking, it is uncontroversial to say that AVs are likely to have a positive impact. Whether or not AVs have a positive impact on traffic congestion and transport emissions greatly depends on how problems and opportunities are tackled by policymakers. Social, ethical, economic, and environmental consequences of adopting this technology should be considered, and action taken to shape its introduction.

To better understand the impact of AV technology on traffic congestion and transport emissions, questionnaire survey and traffic simulation were undertaken.

Questionnaire survey was used to understand the short-distance travel behaviour of residents of Kanagawa, Saitama, and Chiba prefectures. Firstly, responses were used to estimate demand for different transport modes for journey types of different distances and purposes. Then, the utility function for route choice was estimated using responses for the discrete choice experiment section of the survey. This data was used to inform route and mode choice in traffic simulation.

Traffic simulation was used to understand the effect of AV technology on traffic. Three scenarios of AV adoption were explored – the base scenario (no adoption), a partial scenario (partially automated AV adoption), and the AV scenario (full AV adoption). Additionally, two cases of decision-making for route choice and demand shift to AVs were implemented – a “full knowledge” case, where decisions were made based on true traffic conditions, and an “imperfect knowledge” case, where decisions were made based on assumed traffic conditions.

Firstly, the impact of AV technology on traffic congestion was explored. A modal shift away from lower-impact modes (such as bicycles) and public transportation (such as buses) was observed in the AV scenario. This led to a direct increase in VMT for higher-impact vehicles (cars, taxis, buses, and AVs), which wasn’t observed in the partially automated AV scenario. While average velocity slightly increased and trip time slightly decreased in the AV scenario over the base scenario, these effects weren't as large compared to the partially automated AV scenario. This was due to the increased attractiveness of AVs over conventional vehicles resulting in a rebound effect not observed in the partial scenario. The effects were slightly more exaggerated in the "imperfect knowledge" decision-making case, where a slightly greater demand shift to AVs and poorer route optimisation were observed, leading to more traffic in the AV scenario.

Traveller utility was also considered. Here, a relationship between traffic congestion, trip time, and traveller utility was assumed, with decreased congestion and trip time leading to an increase in traveller utility. Over the simulation, trip time increased due to increasing congestion on certain route segments. In both decision-making cases, trip time was reduced in the AV scenario, although since the number of AV trips was much greater in this case this effect was reduced over time. A slight increase in congestion as interpreted through velocity change from low- to high-congestion situations in the AV scenario could be interpreted as a decrease in traveller utility.

Finally, CO_2 emissions were investigated. Conventional vehicles (cars, partial AVs, taxis, and buses) were assumed to be ICE vehicles, where fuel consumption is related to velocity. AVs were assumed to be EVs, with a constant fuel consumption. CO_2 emissions were reduced by comparable amounts in both the AV and partial scenarios, demonstrating the balancing of the EV effect and rebound effect in the AV case. CO_2 emissions overall decreased in the "full knowledge" case, showing that route-optimisation in this case had a significant effect on overall traffic and emissions.

While AVs have great potential to improve private vehicle transport, this simulation shows there are considerations to be made to maximise the positive impact of this technology. Safety improvements, electric powertrain, and V2X communication for optimisation of traffic are effects that have a significant impact on congestion and CO_2

emissions and should be prioritised by producers and policymakers. Some extensions of this research are suggested in Section 6.

6 RECOMMENDATIONS

The simulation presented here begins a quantitative analysis of the impact of the introduction of AV technology. In the future, further expansions could improve the quality of simulation output.

Firstly, the route choice model could be altered from a static calculation to a dynamic calculation. Presently, route choice is calculated at the beginning of a trip and is fixed after that point. To better represent true traffic, the option to alter route choice during the trip should be added.

Currently the only choice attributes considered for mode demand estimation and demand shift are travel time, distance, and destination. A more sophisticated model of mode choice could be constructed including further choice attributes such as waiting time, cost, etc.

In simulation, only private AVs are considered. Given the conjecture that models of vehicle ownership are likely to shift with the introduction of AVs, SAVs should also be considered to paint a full picture of the possible impacts.

Finally, a real map and person trip data set could be used to present results that can be better verified against real-world data. By expanding the grid used to a real map, a better assumption of dummy variables could also be implemented. Additionally, in the "imperfect knowledge" decision-making case, decisions are made based on an assumed velocity that is constant across routes and times. To better represent true traffic, a simple mapping of peak and off-peak traffic in different routes could be used.

7 REFERENCES

Abe, S. (2015). *STS フォーラム第１２回年次総会における安倍総理スピーチ [Remarks by Prime Minister Shinzo Abe at the annual meeting of the Science and Technology in Society (STS) Forum].* Kyoto International Conference Centre STS Forum October 4 2015.

Agency for Natural Resources and Energy. (2012). *資源エネルギー庁平成２４年度省エネルギー設備導入等促進事業報告書（自動車実走行燃料消費情報等提供事業）[Report on promotion of energy saving equipment (actual fuel consumption of automobiles)].* Government of Japan, Ministry of Economy Trade, and Industry.

Anderson, J. M., Kalra, N., Stanley, K. D., Sorensen, P., Samaras, C., & Oluwatola, O. A. (2014). *Autonomous vehicle technology: A guide for policymakers.* CA, Santa Monica: Rand Corporation.

Banks, V. A., Eriksson, A., O'Donoghue, J., & Stanton, N. A. (2018). Is partially automated driving a bad idea? Observations from an on-road study. *Applied Ergonomics, 68*, 138-145.

Bansal, P., & Kockelman, K. M. (2017). Forecasting Americans' long-term adoption of connected and autonomous vehicle technologies. *Technology Research Part A, 95*, 46-63.

Bansal, P., Kockelman, K. M., & Singh, A. (2016). Assessing public opinions of and interest in new vehicle technologies: An Austin perspective. *Transportation Research Part C, 67*, 1-14.

Barter, P. (2013). *Cars are parked 95% of the time - let's check.* Retrieved from Reinventing Parking: www.reinventingparking.org

Becker, F., & Axhausen, K. W. (2017). Literature review on surveys investigating the acceptance of autonomous vehicles. *Transportation, 44*(6), 1293-1306.

Ben-Akiva, M., & Lerman, S. R. (1985). *Discrete Choice Analysis: Theory and Application to Travel Demand.* Cambridge, MA: MIT Press.

Brown, B., Drew, M., Erenguc, C., Hasegawa, M., Hill, R., Schmith, S., & Gangula, B. (2014). *The changing nature of mobility: exploring consumer preferences in key markets around the world.* Deloitte 2104 Global Automotive Consumer Study.

Carbon Trust. (2016). *Conversion factors.* Retrieved from Carbon Trust: www.carbontrust.com

Chester, M., Fraser, A., Matute, J., Flower, C., & Pendyala, R. (2015). Parking infrastructure: a constraint on or an opportunity for urban redevelopment? A study of Los Angeles County parking supply and growth. *Journal of American Planning Association, 81*(4), 268-286.

Cisco Systems. (2013). *Automotive industry global data.* Cisco Customer Experience Research.

Claudel, M., & Ratti, C. (2015). *Full speed ahead: How the driverless car could transform cities.* Retrieved from McKinsey & Company: www.mckinsey.com

Continental. (2013). *Continental mobility study 2013.*

EV Database. (2019). *Electric Vehicle Database*. Retrieved from www.ev-database.uk

FHWA. (2004). *Traffic congestion and reliability: Linking solutions to problems.* Federal Highway Administration. Cambridge Systematics, Inc. & Texas Transportation Institute.

Friedrich, B. (2016). The effect of autonomous vehicles on traffic. In Maurer, M., et al. (eds.), *Autonomous Driving: Social, Technical and Legal Aspects* (pp. 317-334). Berlin: Springer.

Fujita, K., & Hill, R. C. (2007). The zero waste city: Tokyo's quest for sustainable environment. *The Journal of Comparative Policy Analysis, 9*(4), 405-425.

Gandia, R. M., Antonialli, F., Cavazza, B. H., Neto, A. M., de Lima, S. A., Sugano, J. Y., . . . Zambalde, A. L. (2019). Autonomous vehicles: scientometric and bibliometric review. *Transport Reviews, 39*(1), 9-28.

Groover, M. P. (2010). *Fundamentals of Modern Manufacturing: Materials, Processes and Systems.* MA, Danvers: Wiley.

Howard, D., & Dai, D. (2014). Public perceptions of self-driving cars: The case of Berkeley, California. *93rd Annual Meeting of the Transportation Research Board.* Washington DC.

Hulse, L. M., Xie, H., & Galea, E. R. (2018). Perceptions of autonomous vehicles: Relationships with road users, risk, gender and age. *Safety Science, 102*, 1-13.

Iglínski, H., & Babiak, M. (2017). Analysis of the potential of autonomous vehicles in reducing the emissions of greenhouse gases in road transport. *Procedia Engineering 192*, (pp. 353-358). Novy Smokovec.

INRIX. (2019). *Congestion costs UK nearly £8 billion in 2018*. Retrieved from INRIX: www.inrix.com

IPCC. (2014). Working Group III contribution to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change. In Edenhoder, O., et al., *Climate Change 2014 Mitigation of Climate Change.* Cambridge & New York, NY: Cambridge University Press.

Ipsos MORI. (2014). *Loyalty automotive survey.*

J. D. Power and Associates. (2012). *J. D. Power and Associates 2012 U.S. automotive emerging technologies study.*

Krafcik, J. (2018). *Where the next 10 million miles will take us.* Retrieved from Waymo: www.waymo.com

Krueger, R., Rashidi, T. H., & Rose, J. M. (2016). Preferences for shared autonomous vehicles. *Transportation Research Part C, 69*, 343-355.

Kyriakidis, M., de Winter, J. C., Stanton, N., Bellet, T., van Arem, B., Brookhuis, K., . . . Happee, R. (2019). A human factors perspective on automated driving. *Theoretical Issues in Ergonomics Science, 20*(3), 223-249.

Kyriakidis, M., Happee, R., & de Winter, J. C. (2015). Public opinion on automated driving: Results of an international questionnaire among 5000 respondents. *Transportation Research Part F, 32*, 127-140.

Ma, J., Li, X., Zhou, F., & Hao, W. (2017). Designing optimal autonomous vehicle sharing and reservation systems: a linear programming approach. *Transportation Research Part C, 84*, 124-141.

Malasek, J. (2016). A set of tools for making urban transport more sustainable. *Transport Research Procedia 14*, (pp. 876-885). Warsaw.

Maurer, M., Gerdes, J. C., Lenz, B., & Hermann, W. (2016). *Autonomous driving: technical, legal and social aspects.* Berlin: Springer.

Meyer, J., Becker, H., Bösch, P. M., & Axhausen, K. W. (2017). Autonomous vehicles: the next jump in accessibilities? *Research in Transportation Economics, 62*, 80-91.

Moore, G. E. (1965). Cramming more components onto integrated circuits. *Electronics, 38*(8), 114-117.

Moorthy, A., De Kleine, R., Keoleian, G., Good, J., & Lewis, G. (2017). Shared autonomous vehicles as a sustainable solution to the last mile problem: A case study of Ann Arbor-Detroit area. *SAE International Journal of Passenger Cars - Electronic and Electrical Systems, 10*(2), 328-336.

Morita, K., & Sekine, M. (2013). Analysis of traffic accidents caused by elderly drivers based on numbers of rear-end collided. *Transactions of Scoiety of Automotive Engineers of Japan, 44*(3), 903-908.

Mosquet, X., Lang, N., Rüßmann, M., Mei-Pochtler, A., Agrawal, R., & Schmieg, F. (2015). *Revolution in the driver's seat: the road to autonomous vehicles.* The Boston Consulting Group.

New York State Energey Research and Development Authority. (2015). *New York State greenhouse gas inventory and forecast: inventory 1990-2011 and forecast 2012-2030.* New York State Energy Research and Development Authority Final Report (June 2015).

NHTSA. (2015). *Critical reasons for crashes investigated in the National Motor Crash Causation Survey.* National Highway Traffic Safety Administration.

Payre, W., Cestac, J., & Delhomme, P. (2014). Intention to use a fully automated car: attitudes and a priori acceptability. *Transportation Research Part F, 27*, 252-263.

Rödel, C., Stadler, S., Meschtscherjakov, A., & Tscheligi, M. (2014). Towards autonomous cars: the effect of autonomy levels on acceptance and user experience. *Proceedings of the 6th International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, (pp. 1-8). Seattle, WA.

Rao, V. R. (2013). *Applied Conjoint Analysis.* Berlin & Heidelberg: Springer.

Reichmuth, D. (2018). *How important is it for self-driving cars to be electric?* Retrieved from Union of Concerned Scientists: www.ucsusa.org

SAE International. (2014). *Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems.* Ground Vehicle Standard J3016_201401.

Schoettel, B., & Sivak, M. (2014). *A survey of public opinion about autonomous and self-driving vehicles in the US, the UK, and Australia.* University of Michigan, Transportation Research Institute Report.

Schoettle, B., & Sivak, M. (2015). *Motorists' preferences for different levels of vehicle automation.* University of Michigan, Transportation Research Institute Report.

Schrank, D., Eisele, B., Lomax, T., & Bak, J. (2015). *2015 urban mobility scorecard.* Texas A&M Transportation Institute.

Seapine Software. (2014).

Silberg, G., Wallace, R., Matsuzak, G., Plessers, J., Brower, C., & Subramanian, D. (2013). *Self-driving cars: the next revolution.* KPMG & Center for Automotive Research.

Sobrino, N., Monzon, A., & Hernandez, S. (2016). Reduced carbon and energy footprint in highway operations: the highway energy assessment (HERA) methodology. *Networks and Spatial Economics, 16*(1), 395-414.

TMG. (2002). *東京都環境基本計画 2002 [Tokyo Environmental Master Plan 2002].* Tokyo Metropolitan Government, Bureau of Environment.

Weiß, C. (2011). V2X communication in Europe - from research projects towards standardization and field testing of vehicle communication technology. *Computer Networks, 55*, 3103-3119.

WHO. (2004). *WHO world report on road traffic injury prevdention.* Geneva: World Health Organisation.

WHO. (2016). *WHO releases country estimates on air pollution exposure and health impact.* Retrieved from World Health Organisation: www.who.int

WHO. (2018). *WHO global status report on road safety.* Geneva: World Health Organisation.

World Health Organisation. (2011). *WHO world report on disability.* WHO.

Yamamoto, M., Kaji, D., Hattori, Y., Yamamoto, T., Tamada, M., & Fujigaki, Y. (2016). 自動運転シェアカーに関する将来需要予測とシミュレーション分析 [Demand forecast and simulation analysis of autonomous vehicle sharing system]. *Deno Technical Review, 21*, 37-41.

Yap, M. D., Correia, G. C., & van Arem, B. (2016). Preferences of travellers for using automated vehicles as last mile public transport of multimodal train trips. *Transportation Research Part A, 94*, 1-16.

Zmud, H., Sener, I. N., & Wagner, J. (2016). *Consumer acceptance and travel behaviour impacts of automated vehicles.* Texas A&M Transportation Institute.

8 APPENDIX A: Questionnaire Survey

移動経路に関するアンケート ・ Route and mode choice questionnaire survey

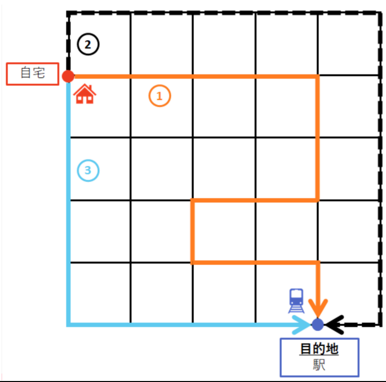
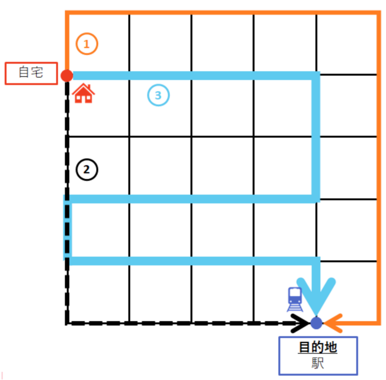
Question			Answers	
1	Gender	あなたの性別をお知らせください。 （ひとつだけ）「必要」	Male	男性
			Female	女性
2	Age	あなたの年齢をお知らせください。 （ひとつだけ）「必要」	Under 17	17 歳以下
			18-24	18-24 歳
			25-34	25-34 歳
			35-44	35-44 歳
			45-54	45-44 歳
			55-64	55-54 歳
			65-74	65-74 歳
			Over 75	75 歳以上
3	Residence	あなたのお住まいを知らせください。 （ひとつだけ）「必要」	Ibaragi	茨城県
			Tochigi	栃木県
			Gunma	群馬県
			Saitama	埼玉県
			Chiba	千葉県
			Kanagawa	神奈川県
			Yamanashi	山梨県
			Other	上記以外の地域
4	Employment	あなたの職業をお知らせください。 （ひとつだけ）「必要」	Company employee	会社員
			Public sector employee	公務員
			Self-employed	自営業
			Student	学生
			Part-time	アルバイト
			Homemaker	専業主婦（主夫）
			Other	その他
5	Vehicle ownership	現在あなたの世帯で車を所有していますか？ （ひとつだけ）「必要」	No vehicles	車を１台所有している
			One vehicle	車を複数台所有している
			More than one vehicle	車を所有していない
6	Driver's license	あなたは運転免許（普通免許）をもっていますか？ （ひとつだけ）「必要」	Yes	運転免許を持っている
			No	運転免許を持っていない
7	Commuting	ふだん通勤あるいは通学をしていますか？ （ひとつだけ）「必要」	Yes	通勤・通学をしている
			No	通勤・通学をしていない
8	Commute length	通勤・通学の片道の時間をお答えください。 （ひとつだけ）「必要」 「通勤・通学をしている」と回答した方に伺います。	Less than 15 minutes	15 分未満
			15-30 minutes	15 分～30 分
			31-45 minutes	31 分～45 分
			46-60 minutes	46 分～1 時間
			61-75 minutes	1 時間 1 分～1 時間 15 分
			76-90 minutes	1 時間 16 分～1 時間 30 分
			91-120 minutes	1 時間 31 分～2 時間
			More than 120 minutes	2 時間 1 分以上
9	Commute mode	通勤・通学の手段をお答えください。 利用する交通手段をすべて選んでください。 （ただし日によって自家用車またはバスを使い分けるときには、よく使うどちらかの手段を選んでください。電車とバスを乗り継いで行くときには両方を選んでください。） （いくつでも）「必要」	Car	自家用車
			Motorbike	バイク
			Bus	バス
			Bicycle or walking	自転車・徒歩
			Train	電車
			Taxi	タクシー
			Other	その他

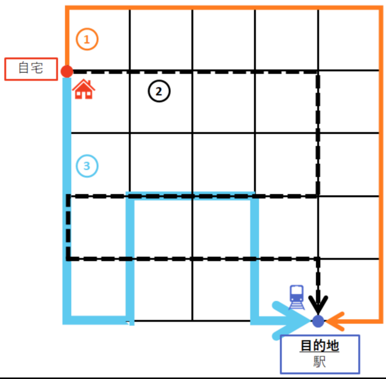
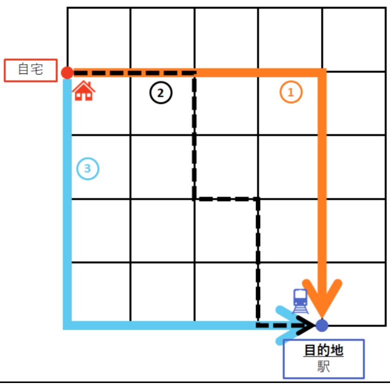
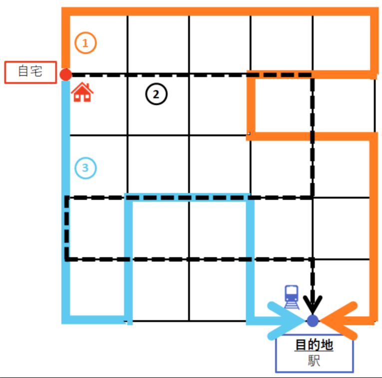
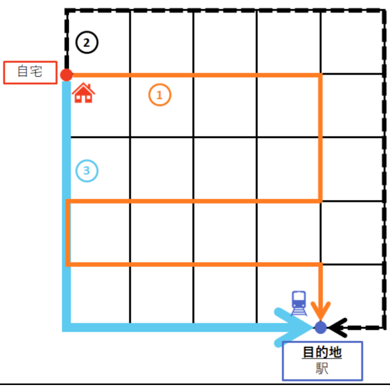
		「通勤・通学をしている」と回答した方に伺います。		
10	Shopping	あなたがスーパーへ買い物に行く頻度をお答えください。 （ひとつだけ）「必要」 「週に１回程度またはそれ以上の頻度でスーパーに買い物に行く」と回答した方に伺います。	Less than once a week	週に１回程度またはそれ以上
			Once a week or more	週に１回程度未満
11	Shopping length	最もよく使うスーパーまでの片道の所要時間をお答えください。 （ひとつだけ）「必要」 「週に１回程度またはそれ以上の頻度でスーパーに買い物に行く」と回答した方に伺います。	Less than 5 minutes	5 分未満
			5-10 minutes	5 分～10 分
			11-15 minutes	11 分～15 分
			16-20 minutes	16 分～20 分
			21-30 minutes	21 分～30 分
			31-40 minutes	31 分～40 分
			41-50 minutes	41 分～50 分
			41-60 minutes	51 分～60 分
			More than 60 minutes	1 時間 1 分以上
12	Shopping mode	最もよく使うスーパーへ行くときに、ふだん使う交通手段をお答えください。（ただし日によって自家用車またはバスを使い分けるときには、よく使うどちらかの手段を選んでください。電車とバスを乗り継いで行くときには両方を選んでください。） （いくつでも）「必要」 「週に１回程度またはそれ以上の頻度でスーパーに買い物に行く」と回答した方に伺います。（いくつでも）「必要」 「週に１回程度またはそれ以上の頻度でスーパーに買い物に行く」と回答した方に伺います。	Car	自家用車
			Motorbike	バイク
			Bus	バス
			Bicycle or walking	自転車・徒歩
			Train	電車
			Taxi	タクシー
			Other	その他

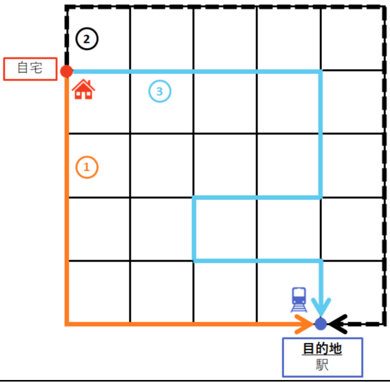
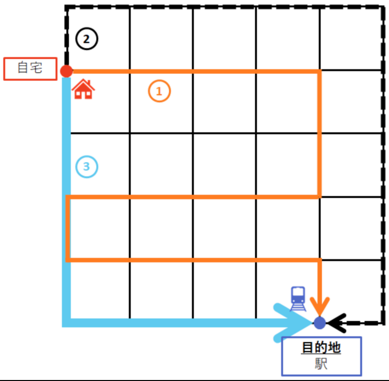
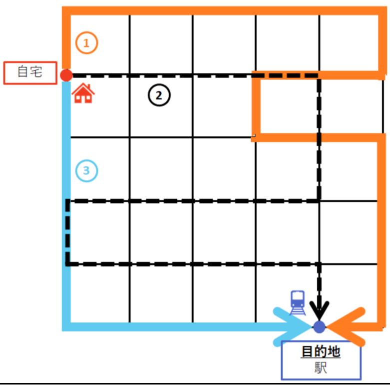
Questions			Answers						
13	Home to station mode choice	自宅から駅に行く場合を考えてください。また、あなたの自宅から駅までの距離が、以下の(1)から(6)のような場合を想像してください。(1)から(6)の距離の場合に、駅への移動で使うと思われる交通手段をお答えください。交通渋滞はないものとします。（横にそれぞれひとつずつ）「必要」							
			Car	Motorbike	Bus	Bicycle or walking	Train	Taxi	Other
			自家用車	バイク	バス	自転車・徒歩	電車	タクシー	その他
	Less than 0.5 km	(1) ～0.5 km（徒歩 6 分以下）							
	0.6-1 km	(2) 0.6 km～1 km（徒歩 6 分～12 分）							
	1.1-1.5 km	(3) 1.1 km～1.5 km（徒歩 13 分～18 分）							
	1.6-2.0 km	(4) 1.6 km～2 km（徒歩 19 分～24 分）							
	2.1-2.5 km	(5) 2.1 km～2.5 km（徒歩 25 分～30 分）							
	More than 2.5 km	(6) 2.6 km～（徒歩 30 分以上）							
14	Home to supermarket mode choice	自宅からスーパーに買い物に行く場合を考えてください。また、あなたの自宅からスーパーまでの距離が、以下の(1)から(6)のような場合を想像してください。(1)から(6)の距離の場合に、スーパーへの移動で使うと思われる交通手段をお答えください。交通渋滞はないものとします。（横にそれぞれひとつずつ）「必要」							

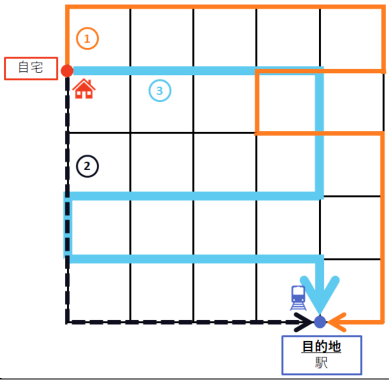
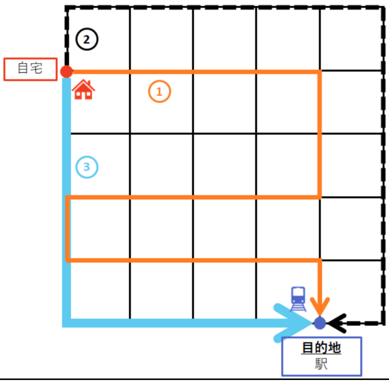
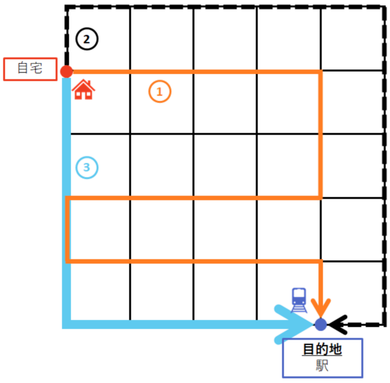
			Car	Motorbike	Bus	Bicycle or walking	Train	Taxi	Other
			自家用車	バイク	バス	自転車・徒歩	電車	タクシー	その他
	Less than 0.5 km	(1) ～0.5 km（徒歩 6 分以下）							
	0.6-1 km	(2) 0.6 km～1 km（徒歩 6 分～12 分）							
	1.1-1.5 km	(3) 1.1 km～1.5 km（徒歩 13 分～18 分）							
	1.6-2.0 km	(4) 1.6 km～2 km（徒歩 19 分～24 分）							
	2.1-2.5 km	(5) 2.1 km～2.5 km（徒歩 25 分～30 分）							
	More than 2.5 km	(6) 2.6 km～（徒歩 30 分以上）							

次の Q15 から Q26 まで、自宅から駅までの経路（3 パターン）が記された地図が提示されます。地図はそれぞれの設問ごとに異なります。
提示された地図と、各 3 つの経路の所要時間、距離、普段から通る道・通らない道の情報が記された選択肢をよく確認し、あなたが自家用車で自宅から駅に向かうことと想像して出発地から目的地までの経路（道順）のうち、あなたが最も好ましいと思うのを選んでください。
ご自身が車を運転できない、ご自宅に自家用車がない方は、どなたかに車に乗せてもらって移動することを想定してお答えください。

Questions			Answers	
15	(1) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」  	Route 1 • Time: 10 minutes • Distance: 6 km (Average velocity: 36 km/h) • Typical route: no	経路 1 ●所要時間：10 分 ●距離：6 km （平均時速 36 km/h） ●普段は通らない道
			Route 2 • Time: 15 minutes • Distance: 6 km (Average velocity: 24 km/h) • Typical route: no	経路 2 ●所要時間：15 分 ●距離：6 km （平均時速 24 km/h） ●普段は通らない道
			Route 3 • Time: 10 minutes • Distance: 4 km (Average velocity: 24 km/h) • Typical route: no	経路 3 ●所要時間：10 分 ●距離：4 km （平均時速 24 km/h） ●普段は通らない道
16	(2) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」  	Route 1 • Time: 10 minutes • Distance: 6 km (Average velocity: 36 km/h) • Typical route: no	経路 1 ●所要時間：10 分 ●距離：6 km （平均時速 36 km/h） ●普段は通らない道
			Route 2 • Time: 20 minutes • Distance: 4 km (Average velocity: 12 km/h) • Typical route: no	経路 2 ●所要時間：20 分 ●距離：4 km （平均時速 12 km/h） ●普段は通らない道
			Route 3 • Time: 10 minutes • Distance: 8 km (Average velocity: 48 km/h) • Typical route: yes	経路 3 ●所要時間：10 分 ●距離：8 km （平均時速 48 km/h） ●普段も通る道
17		どの移動経路を選びますか？ （ひとつだけ）「必要」	Route 1 • Time: 10 minutes	経路 1 ●所要時間：10 分

	(3) Route choice		• Distance: 6 km (Average velocity: 36 km/h) • Typical route: no	●距離：6 km （平均時速 36 km/h） ●普段は通らない道
			Route 2 • Time: 15 minutes • Distance: 8 km (Average velocity: 32 km/h) • Typical route: no	経路 2 ●所要時間：15 分 ●距離：8 km （平均時速 32 km/h） ●普段は通らない道
			Route 3 • Time: 20 minutes • Distance: 6 km (Average velocity: 18 km/h) • Typical route: yes	経路 3 ●所要時間：20 分 ●距離：6 km （平均時速 18 km/h） ●普段も通る道
18	(4) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 20 minutes • Distance: 4 km (Average velocity: 12 km/h) • Typical route: no	経路 1 ●所要時間：20 分 ●距離：4 km （平均時速 12 km/h） ●普段は通らない道
			Route 2 • Time: 15 minutes • Distance: 4 km (Average velocity: 16 km/h) • Typical route: yes	経路 2 ●所要時間：15 分 ●距離：4 km （平均時速 16 km/h） ●普段も通る道
			Route 3 • Time: 10 minutes • Distance: 4 km (Average velocity: 24 km/h) • Typical route: no	経路 3 ●所要時間：10 分 ●距離：4 km （平均時速 24 km/h） ●普段は通らない道
19	(5) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 20 minutes • Distance: 8 km (Average velocity: 24 km/h) • Typical route: no	経路 1 ●所要時間：20 分 ●距離：8 km （平均時速 24 km/h） ●普段は通らない道
			Route 2 • Time: 10 minutes • Distance: 8 km (Average velocity: 48 km/h) • Typical route: yes	経路 2 ●所要時間：10 分 ●距離：8 km （平均時速 48 km/h） ●普段も通る道
			Route 3 • Time: 15 minutes • Distance: 6 km (Average velocity: 24 km/h) • Typical route: no	経路 3 ●所要時間：15 分 ●距離：6 km （平均時速 24 km/h） ●普段は通らない道
20	(6) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 15 minutes • Distance: 8 km (Average velocity: 32 km/h) • Typical route: no	経路 1 ●所要時間：15 分 ●距離：8 km （平均時速 32 km/h） ●普段は通らない道
			Route 2 • Time: 15 minutes • Distance: 6 km (Average velocity: 24 km/h) • Typical route: no	経路 2 ●所要時間：15 分 ●距離：6 km （平均時速 24 km/h） ●普段は通らない道
			Route 3 • Time: 15 minutes • Distance: 4 km (Average velocity: 16 km/h) • Typical route: yes	経路 3 ●所要時間：15 分 ●距離：4 km （平均時速 16 km/h） ●普段も通る道

21	(7) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 20 minutes • Distance: 4 km (Average velocity: 12 km/h) • Typical route: no	経路 1 ●所要時間：20 分 ●距離：4 km （平均時速 12 km/h） ●普段は通らない道
			Route 2 • Time: 20 minutes • Distance: 6 km (Average velocity: 18 km/h) • Typical route: yes	経路 2 ●所要時間：20 分 ●距離：6 km （平均時速 18 km/h） ●普段も通る道
			Route 3 • Time: 15 minutes • Distance: 6 km (Average velocity: 24 km/h) • Typical route: no	経路 3 ●所要時間：15 分 ●距離：6 km （平均時速 24 km/h） ●普段は通らない道
22	(8) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 20 minutes • Distance: 8 km (Average velocity: 24 km/h) • Typical route: no	経路 1 ●所要時間：20 分 ●距離：8 km （平均時速 24 km/h） ●普段は通らない道
			Route 2 • Time: 10 minutes • Distance: 6 km (Average velocity: 36 km/h) • Typical route: no	経路 2 ●所要時間：10 分 ●距離：6 km （平均時速 36 km/h） ●普段は通らない道
			Route 3 • Time: 15 minutes • Distance: 4 km (Average velocity: 16 km/h) • Typical route: yes	経路 3 ●所要時間：15 分 ●距離：4 km （平均時速 16 km/h） ●普段も通る道
23	(9) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 10 minutes • Distance: 8 km (Average velocity: 48 km/h) • Typical route: yes	経路 1 ●所要時間：10 分 ●距離：8 km （平均時速 48 km/h） ●普段も通る道
			Route 2 • Time: 15 minutes • Distance: 8 km (Average velocity: 32 km/h) • Typical route: no	経路 2 ●所要時間：15 分 ●距離：8 km （平均時速 32 km/h） ●普段は通らない道
			Route 3 • Time: 10 minutes • Distance: 4 km (Average velocity: 24 km/h) • Typical route: no	経路 3 ●所要時間：10 分 ●距離：4 km （平均時速 24 km/h） ●普段は通らない道
24	(10) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」	Route 1 • Time: 20 minutes • Distance: 8 km (Average velocity: 24 km/h) • Typical route: no	経路 1 ●所要時間：20 分 ●距離：8 km （平均時速 24 km/h） ●普段は通らない道
			Route 2 • Time: 20 minutes • Distance: 4 km (Average velocity: 12 km/h) • Typical route: no	経路 2 ●所要時間：20 分 ●距離：4 km （平均時速 12 km/h） ●普段は通らない道
			Route 3 • Time: 15 minutes • Distance: 8 km	経路 3 ●所要時間：15 分 ●距離：8 km

			(Average velocity: 32 km/h) • Typical route: no	（平均時速 32 km/h） ●普段は通らない道
25	(11) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 10 minutes • Distance: 8 km (Average velocity: 48 km/h) • Typical route: yes	経路 1 ●所要時間：10 分 ●距離：8 km （平均時速 48 km/h） ●普段も通る道
			Route 2 • Time: 20 minutes • Distance: 6 km (Average velocity: 18 km/h) • Typical route: yes	経路 2 ●所要時間：20 分 ●距離：6 km （平均時速 18 km/h） ●普段も通る道
			Route 3 • Time: 15 minutes • Distance: 4 km (Average velocity: 16 km/h) • Typical route: yes	経路 3 ●所要時間：15 分 ●距離：4 km （平均時速 16 km/h） ●普段も通る道
26	(12) Route choice	どの移動経路を選びますか？ （ひとつだけ）「必要」 	Route 1 • Time: 20 minutes • Distance: 8 km (Average velocity: 24 km/h) • Typical route: no	経路 1 ●所要時間：20 分 ●距離：8 km （平均時速 24 km/h） ●普段は通らない道
			Route 2 • Time: 20 minutes • Distance: 6 km (Average velocity: 18 km/h) • Typical route: yes	経路 2 ●所要時間：20 分 ●距離：6 km （平均時速 18 km/h） ●普段も通る道
			Route 3 • Time: 10 minutes • Distance: 4 km (Average velocity: 24 km/h) • Typical route: no	経路 3 ●所要時間：10 分 ●距離：4 km （平均時速 24 km/h） ●普段は通らない道

9 APPENDIX B: Simulation Code

traffic-sim-start.rb

```
require_relative 'traffic-sim.rb'

runs = 100
demand_shift = 0
conv_hdwy_time = 2
assumed = 0
sim_start = Time.now

# create simulation object with simulation constants
simulation = Trips.new(runs, demand_shift, conv_hdwy_time, assumed, sim_start)
# print label indicating start of simulation
puts "Start: #{sim_start.strftime("%F %H:%M:%S")}"
# run simulation for x trips
trip_num = 0
runs.times do
  trip_result = simulation.trip(trip_num)
  trip_num += 1
end
# print label indicating end of simulation
puts "End: #{Time.now.strftime("%F %H:%M:%S")}"
```

traffic-sim.rb

```
# require 'csv' in order to output average_data to csv file
require 'csv'
# require file where route generation is stored
require_relative 'traffic-sim-paths'
# require file where additional generation methods are stored
require_relative 'traffic-sim-generate'
# require file where additional calculation methods are stored
require_relative 'traffic-sim-calculate'

class Trips
  # variables for constants
  attr_reader :ms_to_kmh, :kmh_to_ms, :intersec_wait, :assumed
  attr_reader :dist, :auto_length, :bus_length, :av_hdwy_time, :conv_hdwy_time
  attr_reader :v_max, :motbike_v, :bike_v, :parking, :taxi_pen, :demand_shift
  attr_reader :start_times, :origins, :dests, :mode_choices
  # variables for buses
  attr_reader :bus1_timetable, :bus2_timetable, :bus3_timetable
  attr_reader :bus1_route, :bus2_route, :bus3_route
  # variables for simulation map
  attr_reader :segs, :rts_0_0, :rts_1_0, :rts_2_0, :rts_3_0
  attr_reader :rts_0_1, :rts_1_1, :rts_2_1, :rts_3_1, :rts_secs
  # variables for output file names
  attr_reader :sim_out_fname, :time_ev_n_fname, :summary_fname
  # variables for storing trip outcomes
  attr_accessor :route_choices, :new_mode_choices, :arr_times
  attr_accessor :bus1_arr_t, :bus1_s_st_t, :bus1_s_arr_t, :bus1_s_v, :bus1_v
```

```
attr_accessor :bus2_arr_t, :bus2_s_st_t, :bus2_s_arr_t, :bus2_s_v, :bus2_v
attr_accessor :bus3_arr_t, :bus3_s_st_t, :bus3_s_arr_t, :bus3_s_v, :bus3_v
attr_accessor :auto_s_st_t, :auto_s_end_t, :bus_s_st_t, :bus_s_end_t
attr_accessor :bike_s_st_t, :bike_s_end_t, :mbike_s_st_t, :mbike_s_end_t
attr_accessor :nums_segs, :velocities, :vel_segs, :chosen_vels, :chosen_segs
# variable for switch counter
attr_accessor :no_switch_av

# initialise: set up class variables
def initialize(runs, demand_shift, conv_hdwy_time, assumed, sim_start)
  # set variables (fixed constants)
  @ms_to_kmh = (60 * 60).to_f / 1000 # convert m/s to km/h
  @kmh_to_ms = 1000.to_f / (60 * 60) # convert km/h to m/s
  @dist = 500 # length of one segment (distance)
  @auto_length, @bus_length = 7.5, 17 # length of automobile / bus
  @av_hdwy_time, @conv_hdwy_time = 0.5, conv_hdwy_time # headway time
  @v_max = 40 * @kmh_to_ms # speed limit
  @motbike_v = 30 * @kmh_to_ms # constant motorbike velocity
  @bike_v = 14 * @kmh_to_ms # constant bicycle velocity
  @parking = 300 # constant additive parking penalty
  @taxi_pen = 1.5 # constant multiplicative taxi penalty
  @demand_shift = demand_shift # demand shift probability (0 - 100 %)
  @intersec_wait = 20 # intersection wait (s)
  @start_int = 0.1 # interval between starting times (s)
  @assumed = assumed # whether decisions are full- or imperfect-knowledge case
  # set switch counter to track number of travellers who switch to
  # AV (i.e. AV utility exceeds prior mode)
  @no_switch_av = 0
  # set segments array (60 segments on 6 x 6 node map)
  @segs = Array.new(60) { |i| i }
  # generate object to be used to create map and paths, start times,
  # origins, dests, bus timetable
  generate = Generate.new
  # generate class variables (trips and bus)
  @start_times = generate.generate_start_times(runs, @start_int)
  @origins = generate.generate_origins(runs)
  @dests = generate.generate_destinations(runs)
  @bus1_timetable = generate.generate_bus_timetable(60)
  @bus2_timetable = generate.generate_bus_timetable(60)
  @bus3_timetable = generate.generate_bus_timetable(60)
  # set bus routes
  @bus1_route = [36, 42, 48, 54, 25, 26, 27, 28]
  @bus2_route = [48, 15, 16, 17, 18, 19]
  @bus3_route = [36, 42, 15, 16, 17, 18, 19]
  # generate maps for each origin and dest map_origin_destination
  # origin 0 - 3 = different map points
  # dest 0 = station, dest 1 = shop
  map_0_0 = generate.generate_map_0(1)
  map_1_0 = generate.generate_map_0(2)
  map_2_0 = generate.generate_map_0(3)
  map_3_0 = generate.generate_map_0(4)
  map_0_1 = generate.generate_map_1(1)
  map_1_1 = generate.generate_map_1(2)
  map_2_1 = generate.generate_map_1(3)
```

```
map_3_1 = generate.generate_map_1(4)
# generate all possible routes for each origin / dest
@rts_0_0 = generate.generate_paths_final(map_0_0)
@rts_1_0 = generate.generate_paths_final(map_1_0)
@rts_2_0 = generate.generate_paths_final(map_2_0)
@rts_3_0 = generate.generate_paths_final(map_3_0)
@rts_0_1 = generate.generate_paths_final(map_0_1)
@rts_1_1 = generate.generate_paths_final(map_1_1)
@rts_2_1 = [[15, 16, 17, 18, 19]]
@rts_3_1 = generate.generate_paths_final(map_3_1, 'r')
@rts_secs = [
  @rts_0_0[0].length, @rts_1_0[0].length, @rts_2_0[0].length,
  @rts_3_0[0].length, @rts_0_1[0].length, @rts_1_1[0].length,
  @rts_2_1[0].length, @rts_3_1[0].length
]
# set class variables (storing trip outcomes)
@route_choices = Array.new(@start_times.length, 0)
@new_mode_choices = Array.new(@start_times.length, 0)
@arr_times = Array.new(@start_times.length, 0)
@bus1_arr_t = Array.new(@bus1_timetable.length, 0)
@bus2_arr_t = Array.new(@bus2_timetable.length, 0)
@bus3_arr_t = Array.new(@bus3_timetable.length, 0)
@bus1_s_st_t = Array.new(@bus1_timetable.length) { Array.new(60, 0) }
@bus1_s_arr_t = Array.new(@bus1_timetable.length) { Array.new(60, 0) }
@bus2_s_st_t = Array.new(@bus2_timetable.length) { Array.new(60, 0) }
@bus2_s_arr_t = Array.new(@bus2_timetable.length) { Array.new(60, 0) }
@bus3_s_st_t = Array.new(@bus3_timetable.length) { Array.new(60, 0) }
@bus3_s_arr_t = Array.new(@bus3_timetable.length) { Array.new(60, 0) }
@bus1_v = Array.new(@bus1_timetable.length, 0)
@bus2_v = Array.new(@bus2_timetable.length, 0)
@bus3_v = Array.new(@bus3_timetable.length, 0)
@bus1_s_v = Array.new(@bus1_timetable.length) {
  Array.new(bus1_route.length, 0) }
@bus2_s_v = Array.new(@bus2_timetable.length) {
  Array.new(bus2_route.length, 0) }
@bus3_s_v = Array.new(@bus3_timetable.length) {
  Array.new(bus3_route.length, 0) }
@auto_s_st_t = Array.new(@start_times.length) { Array.new(60, 0) }
@auto_s_end_t = Array.new(@start_times.length) { Array.new(60, 0) }
@bus_s_st_t = Array.new(@start_times.length) { Array.new(60, 0) }
@bus_s_end_t = Array.new(@start_times.length) { Array.new(60, 0) }
@bike_s_st_t = Array.new(@start_times.length) { Array.new(60, 0) }
@bike_s_end_t = Array.new(@start_times.length) { Array.new(60, 0) }
@mbike_s_st_t = Array.new(@start_times.length) { Array.new(60, 0) }
@mbike_s_end_t = Array.new(@start_times.length) { Array.new(60, 0) }
@velocities = Array.new(@start_times.length, 0)
@nums_segs = Array.new(@start_times.length) { Array.new(60, 0) }
@vel_segs = Array.new(@start_times.length) { Array.new(60, 0) }
@chosen_vels = Array.new
@chosen_segs = Array.new
@mode_choices = generate.generate_mode_choices(@origins, @dests, @dist)
# set class variables (output average_data file names)
@sim_out_fname = "sim_out_shift#{
  demand_shift}_#{sim_start.strftime("%m%d%H%M%S")}.csv"
```

```ruby
  @time_ev_n_fname = "time_ev_n__shift#{
    demand_shift}_#{sim_start.strftime("%m%d%H%M%S")}.csv"
  @summary_fname = "summary_shift#{
    demand_shift}_#{sim_start.strftime("%m%d%H%M%S")}.csv"
  # generate output files + headers
  generate.generate_output_file_headers(@sim_out_fname, @time_ev_n_fname)
  #summary average_data file headers
  CSV.open(summary_fname, "w") do |csv|
    csv << ["Simulation Start", sim_start.strftime("%m%d%H%M%S")]
    csv << ["Conv. Headway Time", @conv_hdwy_time]
    csv << ["Demand Shift", demand_shift]
    csv << ["Start Interval", @start_int]
    csv << ["Segment Length", @dist]
  end
end

# main: run calculation for each trip in the simulation
def trip(trip_num)
  # retrieve trip variables for current trip
  start_time, mode_choice = start_times[trip_num], mode_choices[trip_num]
  origin, dest = origins[trip_num], dests[trip_num]
  # retrieve routes for current trip and determine which bus is available
  routes = Array.new
  journey = ""
  case origin
  when 0
    dest == 0 ? routes = rts_0_0 : routes = rts_0_1
    dest == 0 ? bus = 1 : bus = 3
    dest == 0 ? journey = 1 : journey = 5
  when 1
    dest == 0 ? routes = rts_1_0 : routes = rts_1_1
    dest == 0 ? bus = 1 : bus = 3
    dest == 0 ? journey = 2 : journey = 6
  when 2
    dest == 0 ? routes = rts_2_0 : routes = rts_2_1
    dest == 0 ? bus = 1 : bus = "2 or 3"
    dest == 0 ? journey = 3 : journey = 7
  when 3
    dest == 0 ? routes = rts_3_0 : routes = rts_3_1
    dest == 0 ? bus = 1 : bus = 2
    dest == 0 ? journey = 4 : journey = 8
  end
  # update bus progress throughout simulation (trip-independent)
  update_bus_progress(1, start_time)
  update_bus_progress(2, start_time)
  update_bus_progress(3, start_time)
  # generate random route dummy variables for each route
  route_dummies = generate_route_dummies(routes)
  # calculate number of turns and distance for each route
  route_turns = calculate_route_turns(routes)
  route_dist = routes[0].length * dist
  # calculate route segment trip times and start times for chosen mode
  route_seg_trip_times, route_seg_start_times =
    calculate_route_s_trip_times(
```

```
    trip_num, start_time, bus, origin, routes, mode_choice, dest)
  # if chosen mode is AV, Car, Taxi, or Bus, subtract extra intersection wait
  final_sec = routes.length * routes[0].length - routes[0].length
  if mode_choice == "AV" || mode_choice == "Car" ||
    mode_choice == "Taxi" || mode_choice == "Bus"
    if final_sec == 0 || mode_choice == "Bus"
      route_seg_trip_times[-1] -= intersec_wait
    else
      routes[0].length.times { |i|
        route_seg_trip_times[final_sec + i] -= intersec_wait }
    end
  end
  # calculate route segment trip times and start times for AV
  av_rt_s_trip_ts, av_rt_s_st_ts =
    calculate_route_s_trip_times(
      trip_num, start_time, bus, origin, routes, "AV", dest)
  if final_sec == 0
    av_rt_s_trip_ts[-1] -= intersec_wait
  else
    routes[0].length.times { |i|
      av_rt_s_trip_ts[final_sec + i] -= intersec_wait }
  end
  # calculate total trip time for each route for chosen mode
  route_trip_times = Array.new
  if mode_choice == "Bus"
    route_trip_time = 0
    routes[0].length.times { |i| route_trip_time += route_seg_trip_times[i] }
    route_trip_times << route_trip_time
  else
    route_trip_times = sum_route_s_trip_times(routes, route_seg_trip_times)
  end
  # add parking penalty if mode choice is Car and destination is Shop
  if mode_choice == "Car" && dest == 1
    route_trip_times.map! { |i| i + parking }
  end
  # calculate total trip time for each route for AV
  av_route_trip_ts = sum_route_s_trip_times(routes, av_rt_s_trip_ts)

  # calculate typical trip times for chosen mode and AV
  typical_trip_time = calculate_typical_trip_time(
    start_time, mode_choice, dest, routes[0].length, routes[0][0], bus)
  typical_av_trip_time = calculate_typical_trip_time(
    start_time, "AV", dest, routes[0].length, routes[0][0], bus)
  if mode_choice == "Bus"
    typical_trip_times = [typical_trip_time]
  else
    typical_trip_times = Array.new(routes.length, typical_trip_time)
  end
  typical_av_trip_times = Array.new(routes.length, typical_av_trip_time)
  # calculate utility for each route for chosen mode and AV
  utilities = calculate_utility(
    route_trip_times, route_dist, route_dummies, route_turns)
  av_utilities = calculate_utility(
    av_route_trip_ts, route_dist, route_dummies, route_turns)
```

```
  # calculate typical trip utilities for chosen mode and aV
  typical_utilities = calculate_utility(
    typical_trip_times, route_dist, route_dummies, route_turns)
  typical_av_utilities = calculate_utility(
    typical_av_trip_times, route_dist, route_dummies, route_turns)
  # determine maximal utility for chosen mode and AV
  trip_utility = calculate_largest(utilities)
  av_utility = calculate_largest(av_utilities)
  # determine maximal typical urility for chosen mode and AV
  typical_utility = calculate_largest(typical_utilities)
  typical_av_utility = calculate_largest(typical_av_utilities)
  # if demand shift is greater than 0 and AV utility exceeds utility of
  # chosen mode, demand shift probability of switching to AV
  # use assumed utilities if imperfect-knowledge case
  if assumed == 1
    new_mode_choice = mode_choice
    if typical_utility < typical_av_utility
      rand_n = rand(100)
      if rand_n < demand_shift
        new_mode_choice = "AV"
        @no_switch_av += 1
      end
    end
  else
    new_mode_choice = mode_choice
    if trip_utility < av_utility
      rand_n = rand(100)
      if rand_n < demand_shift
        new_mode_choice = "AV"
        @no_switch_av += 1
      end
    end
  end
  # if switched to AV, update route segment trip times and utility
  if new_mode_choice != mode_choice
    route_seg_trip_times, route_seg_start_times = av_rt_s_trip_ts, av_rt_s_st_ts
    route_trip_times = av_route_trip_ts
    utilities, trip_utility = av_utilities, av_utility
    typical_trip_time = typical_av_trip_time
    typical_utilities, typical_utility = typical_av_utilities, typical_av_utility
  end
  # determine route choice based on maximal utility
  route_choice = 0
  if assumed == 1
    route_choice = determine_route_choice(typical_utilities, typical_utility)
  else
    route_choice = determine_route_choice(utilities, trip_utility)
  end
  # save trip time, route dummy variable, and route turns to variables
  trip_time = route_trip_times[route_choice]
  route_dummy = route_dummies[route_choice]
  route_turn = route_turns[route_choice]
  # store segment start and trip times
  seg_start_times = Array.new(segs.length, 0)
```

```
seg_trip_times = Array.new(segs.length, 0)
routes.length.times do |i|
  if route_choice == i
    routes[0].length.times do |j|
      if new_mode_choice == "Bus"
        seg_start_times[routes[i][j]] = route_seg_start_times[j + i]
        seg_trip_times[routes[i][j]] = route_seg_trip_times[j + i]
      else
        seg_start_times[routes[i][j]] =
          route_seg_start_times[(j * routes.length) + i]
        seg_trip_times[routes[i][j]] =
          route_seg_trip_times[(j * routes.length) + i]
        end
    end
  end
end
# calculate arrival time based on route and mode choice
intersecs = routes[0].length - 1
arrival_time = calculate_arrival_time(
  new_mode_choice, start_time, trip_time, dest, intersecs, intersec_wait)
# store route choice and arrival time in master arrays
route_choices[trip_num] = route_choice
arr_times[trip_num] = arrival_time
new_mode_choices[trip_num] = new_mode_choice
seg_end_times = Array.new(segs.length, 0)
segs.length.times { |i|
  seg_end_times[i] = seg_start_times[i] + seg_trip_times[i] }
# store segment start and end times for chosen mode in master arrays
case new_mode_choice
when "Bus"
  bus_s_st_t[trip_num] = seg_start_times
  bus_s_end_t[trip_num] = seg_end_times
when "Car"
  auto_s_st_t[trip_num] = seg_start_times
  auto_s_end_t[trip_num] = seg_end_times
when "AV"
  auto_s_st_t[trip_num] = seg_start_times
  auto_s_end_t[trip_num] = seg_end_times
when "Bicycle"
  bike_s_st_t[trip_num] = seg_start_times
  bike_s_end_t[trip_num] = seg_end_times
when "Motorbike"
  mbike_s_st_t[trip_num] = seg_start_times
  mbike_s_end_t[trip_num] = seg_end_times
when "Taxi"
  auto_s_st_t[trip_num] = seg_start_times
  auto_s_end_t[trip_num] = seg_end_times
end
# calculate travel time and average velocity for chosen mode and route
travel_time = (arrival_time - start_time).round(2)
typical_travel_time = typical_trip_time
avg_velocity = 0
if journey < 5 && new_mode_choice == "Car"
  typical_travel_time -= intersecs * intersec_wait
```

```
    typical_travel_time /= calculate_car_trip_time_adj
    typical_travel_time += intersecs * intersec_wait
  elsif journey > 4 && new_mode_choice == "Car"
    avg_velocity =
      ((route_dist.to_f / (travel_time - parking)) * ms_to_kmh).round(2)
  else
    avg_velocity = ((route_dist.to_f / travel_time) * ms_to_kmh).round(2)
  end
  # calculate total number of vehicles on chosen route
  number = 0
  routes[route_choice].each { |i| number += nums_segs[trip_num][i] }
  # update velocities array
  velocities[trip_num] = avg_velocity
  # print trip data to csv file
  CSV.open(sim_out_fname, "a+") do |csv|
    csv << [
      trip_num, "Journey #{journey}", origin, dest, route_dist,
      route_choice, mode_choice, new_mode_choice, travel_time, avg_velocity,
      number, typical_travel_time.round(2)
    ]
  end
  # if final trip, print time evolution and summary average_data to csv file
  if trip_num == (start_times.length - 1)
    output_time_evolution
    print_summary
    calculate_environment
  end
end

# calculate segment start times for each mode
def calculate_segment_start_time(prev_seg_start_time, prev_seg_trip_time)
  segment_start_time = prev_seg_start_time + prev_seg_trip_time
end

# calculate first segment trip times for each mode, based on start time
def calculate_first_segment_trip_time(
  trip_num, start_time, segment, bus, origin, mode_choice, dest)
  case mode_choice
  when "Bus"
    first_segment_trip_time = calculate_trip_time(
      start_time, segment, trip_num) + calculate_bus_wait_time(
      start_time, segment, bus) + intersec_wait
  when "Car"
    if dest == 0
      first_segment_trip_time = (calculate_trip_time(
        start_time, segment, trip_num) * calculate_car_trip_time_adj) +
        intersec_wait
    else
      first_segment_trip_time = calculate_trip_time(
        start_time, segment, trip_num) + intersec_wait
    end
  when "AV"
    first_segment_trip_time = calculate_trip_time(
      start_time, segment, trip_num) + intersec_wait
```

```
  when "Bicycle"
    first_segment_trip_time = dist / bike_v.to_f
  when "Motorbike"
    first_segment_trip_time = dist / motbike_v.to_f
  when "Taxi"
    first_segment_trip_time = (
    calculate_trip_time(start_time, segment, trip_num) * taxi_pen) +
    intersec_wait
  end
  first_segment_trip_time
end

# calculate segment trip times for each mode, based on segment start time
def calculate_segment_trip_time(trip_num, start_time, segment, mode_choice, dest)
  case mode_choice
  when "Bus"
    segment_trip_time = calculate_trip_time(
      start_time, segment, trip_num) + intersec_wait
  when "Car"
    if dest == 0
      segment_trip_time = (calculate_trip_time(
        start_time, segment, trip_num) * calculate_car_trip_time_adj) +
        intersec_wait
    else
      segment_trip_time = calculate_trip_time(start_time, segment, trip_num) +
      intersec_wait
    end
  when "AV"
    segment_trip_time = calculate_trip_time(start_time, segment, trip_num) +
    intersec_wait
  when "Bicycle"
    segment_trip_time = dist / bike_v.to_f
  when "Motorbike"
    segment_trip_time = dist / motbike_v.to_f
  when "Taxi"
    segment_trip_time = (calculate_trip_time(
      start_time, segment, trip_num) * taxi_pen) + intersec_wait
  end
  segment_trip_time
end

# calculate velocity for trip based on segment start time and segment
def calculate_velocity(start_time, segment, trip_num = "N/A")
  num_conv = calculate_num_autos_spec(start_time, segment, "Car") +
    calculate_num_autos_spec(start_time, segment, "Taxi")
  num_bus = calculate_num_buses(start_time, segment)
  num_av = calculate_num_autos_spec(start_time, segment, "AV")
  num = num_conv + num_bus + num_av
  velocity = 0.to_f
  if num == 0
    velocity = v_max
  else
    velocity =
     (1 - ((num_av.to_f / dist) * auto_length) -
```

```
          ((num_conv.to_f / dist) * auto_length) -
          ((num_bus.to_f / dist) * bus_length)) /
        (((num_av.to_f / dist) * av_hdwy_time) +
          ((num_conv.to_f / dist) * conv_hdwy_time) +
          ((num_bus.to_f / dist) * conv_hdwy_time))
    if velocity > v_max
      velocity = v_max
    elsif velocity < (5 * kmh_to_ms)
      velocity = (5 * kmh_to_ms)
    end
  end
  if trip_num != "N/A"
    nums_segs[trip_num][segment] = num
    vel_segs[trip_num][segment] = (velocity * ms_to_kmh).round(2)
  end
  velocity
end

# calculate trip time based on segment start time and segment
def calculate_trip_time(start_time, segment, trip_num = "N/A")
  trip_time = dist / calculate_velocity(start_time, segment, trip_num)
end

# calculate bus waiting time
def calculate_bus_wait_time(start_time, segment, bus)
  bus_wait_time = 0
  if bus == "2 or 3"
    i, j = 0, 0
    i += 1 while start_time > bus2_timetable[i]
    j += 1 while start_time > bus3_timetable[j]
    update_bus_progress(2, bus2_timetable[i])
    update_bus_progress(3, bus3_timetable[j])
    bus2_wait_time = bus2_s_st_t[i][segment] - start_time
    bus3_wait_time = bus3_s_st_t[j][segment] - start_time
    bus2_wait_time < bus3_wait_time ?
      bus_wait_time = bus2_wait_time : bus_wait_time = bus3_wait_time
  elsif bus == 1
    i = 0
    i += 1 while start_time > bus1_timetable[i]
    update_bus_progress(1, bus1_timetable[i])
    bus_wait_time = bus1_s_st_t[i][segment] - start_time
  elsif bus == 2
    i = 0
    i += 1 while start_time > bus2_timetable[i]
    update_bus_progress(2, bus2_timetable[i])
    bus_wait_time = bus2_s_st_t[i][segment] - start_time
  elsif bus == 3
    i = 0
    i += 1 while start_time > bus3_timetable[i]
    update_bus_progress(3, bus3_timetable[i])
    bus_wait_time = bus3_s_st_t[i][segment] - start_time
  else
    puts "Bus error!"
  end
```

```
  bus_wait_time
end


# calculate number of vehicles on segment of road
def calculate_num_vehicles(start_time, segment)
  num_vehicles = calculate_num_autos(start_time, segment) +
    calculate_num_buses(start_time, segment)
end

# calculate number of private vehicles (car, av, taxi) on segment of road
def calculate_num_autos(start_time, segment)
  num_autos = 0
  auto_s_st_t.length.times do |i|
    if start_time >= auto_s_st_t[i][segment]
      if start_time < auto_s_end_t[i][segment]
        if mode_choices[i] == "Car" ||
           mode_choices[i] == "AV" || mode_choices[i] == "Taxi"
          num_autos += 1
        end
      end
    end
  end
  num_autos
end

# calculate number of specified vehicles (car, taxi, or AV) on segment of road
def calculate_num_autos_spec(start_time, segment, mode_choice)
  num_autos_spec = 0
  auto_s_st_t.length.times do |i|
    if start_time >= auto_s_st_t[i][segment]
      if start_time < auto_s_end_t[i][segment]
        num_autos_spec += 1 if mode_choices[i] == mode_choice
      end
    end
  end
  num_autos_spec
end

# calculate number of buses on segment of road
def calculate_num_buses(start_time, segment)
  num_buses = 0
  bus1_timetable.length.times do |i|
    if start_time > bus1_s_st_t[i][segment]
      num_buses += 1
      num_buses -= 1 if start_time > bus1_s_arr_t[i][segment]
    end
  end
  num_buses
end

def calculate_route_s_trip_times(
  trip_num, start_time, bus, origin, routes, mode_choice, dest)
  if mode_choice == "Bus"
```

```
      route_seg_start_times = Array.new(routes[0].length, 0)
      route_seg_start_times[0] = start_time
      route_seg_trip_times = Array.new(routes[0].length, 0)
      route_seg_trip_times[0] = calculate_first_segment_trip_time(
        trip_num, start_time, routes[0][0], bus, origin, mode_choice, dest)
      (routes[0].length - 1).times do |i|
        route_seg_start_times[i + 1] =
          calculate_segment_start_time(
            route_seg_start_times[i], route_seg_trip_times[i])
        route_seg_trip_times[i + 1] = calculate_segment_trip_time(
          trip_num, route_seg_start_times[i + 1],
          routes[0][i + 1], mode_choice, dest)
      end
    else
      route_seg_start_times = Array.new
      routes.length.times { route_seg_start_times.push(start_time) }
      route_seg_trip_times = Array.new
      routes.length.times do |i|
        route_seg_trip_times.push(calculate_first_segment_trip_time(
          trip_num, start_time, routes[i][0], bus, origin, mode_choice, dest))
      end
      # populate second to tenth section start and trip times for each route
      ((routes.length * routes[0].length) - routes.length).times do |i|
      route_seg_start_times.push(
        calculate_segment_start_time(
          route_seg_start_times[i], route_seg_trip_times[i]))
        if i < (routes.length * 1)
          route_seg_trip_times.push(
            calculate_segment_trip_time(
              trip_num, route_seg_start_times[i + routes.length],
              routes[i][1], mode_choice, dest))
        elsif i < (routes.length * 2)
          route_seg_trip_times.push(
            calculate_segment_trip_time(
              trip_num, route_seg_start_times[i + routes.length],
              routes[i - routes.length][2], mode_choice, dest))
        elsif i < (routes.length * 3)
          route_seg_trip_times.push(
            calculate_segment_trip_time(
              trip_num, route_seg_start_times[i + routes.length],
              routes[i - (routes.length * 2)][3], mode_choice, dest))
        elsif i < (routes.length * 4)
          route_seg_trip_times.push(
            calculate_segment_trip_time(
              trip_num, route_seg_start_times[i + routes.length],
              routes[i - (routes.length * 3)][4], mode_choice, dest))
        elsif i < (routes.length * 5)
          route_seg_trip_times.push(
            calculate_segment_trip_time(
              trip_num, route_seg_start_times[i + routes.length],
              routes[i - (routes.length * 4)][5], mode_choice, dest))
        elsif i < (routes.length * 6)
          route_seg_trip_times.push(
            calculate_segment_trip_time(
```

```
            trip_num, route_seg_start_times[i + routes.length],
            routes[i - (routes.length * 5)][6], mode_choice, dest))
      elsif i < (routes.length * 7)
        route_seg_trip_times.push(
          calculate_segment_trip_time(
            trip_num, route_seg_start_times[i + routes.length],
            routes[i - (routes.length * 6)][7], mode_choice, dest))
      elsif i < (routes.length * 8)
        route_seg_trip_times.push(
          calculate_segment_trip_time(
            trip_num, route_seg_start_times[i + routes.length],
            routes[i - (routes.length * 7)][8], mode_choice, dest))
      else
        route_seg_trip_times.push(
          calculate_segment_trip_time(
            trip_num, route_seg_start_times[i + routes.length],
            routes[i - (routes.length * 8)][9], mode_choice, dest))
      end
    end
  end
  return route_seg_trip_times, route_seg_start_times
end

# output time evolution
def output_time_evolution
  seg_cong_n = Array.new
  t = start_times[0]
  until t >= (start_times[-1] + 2000)
    snapshot_n = Array.new
    snapshot_n << t
    60.times { |i| snapshot_n << calculate_num_vehicles(t, i) }
    seg_cong_n << snapshot_n
    t += 10
  end
  routes = Array.new
  origins.length.times do |i|
    case origins[i]
    when 0
      dests[i] == 0 ? routes = rts_0_0 : routes = rts_0_1
    when 1
      dests[i] == 0 ? routes = rts_1_0 : routes = rts_1_1
    when 2
      dests[i] == 0 ? routes = rts_2_0 : routes = rts_2_1
    when 3
      dests[i] == 0 ? routes = rts_3_0 : routes = rts_3_1
    end
    chosen_segs << routes[route_choices[i]]
  end
  chosen_segs.each_with_index do |x, i|
    chosen_vels_trip = Array.new
    x.each do |j|
      chosen_vels_trip << vel_segs[i][j]
    end
    chosen_vels << chosen_vels_trip
```

```ruby
  end
  CSV.open(time_ev_n_fname, "a+") { |csv| seg_cong_n.each { |i| csv << i } }
end

def calculate_environment
  # fuel efficiency inputs
  conv_fuel_slope = 0.4584
    # conv_fuel_cons = conv_fuel_slope * v + conv_fuel_intercept
  bus_fuel_slope = 0.2292
    # bus_fuel_cons = bus_fuel_slope * v + bus_fuel_intercept
  conv_fuel_intercept = 6.8896
  bus_fuel_intercept = 3.4448
  elec_fuel_cons = 7.to_f # km / kWh
  co2_petrol = 2.61163 # kg / L
  co2_elec = 0.41205 # kg / kWh
  num_conv, num_elec, conv_co2, elec_co2 = 0, 0, 0, 0
  conv_total_v, elec_total_v = 0, 0
  new_mode_choices.length.times do |i|
    if new_mode_choices[i] == "Car" || new_mode_choices[i] == "Taxi"
      conv_co2_trip = 0
      num_conv += 1
      conv_total_v += velocities[i]
      chosen_vels[i].each do |j|
        conv_co2_trip += (co2_petrol / ((j * conv_fuel_slope) +
          conv_fuel_intercept)) * (dist.to_f / 1000)
      end
      conv_co2 += conv_co2_trip
    end
    if new_mode_choices[i] == "AV"
      num_elec += 1
      elec_total_v += velocities[i]
      elec_dist = chosen_segs[i].length * dist
      elec_co2_trip = ((co2_elec / elec_fuel_cons) * (elec_dist.to_f / 1000))
      elec_co2 += elec_co2_trip
    end
  end
  num_bus, bus_co2, bus_total_v, total_vmt_bus = 0, 0, 0, 0
  bus1_timetable.length.times do |i|
    if bus1_timetable[i] <= arr_times.max
      total_vmt_bus += bus1_route.length * dist
      update_bus_progress(1, bus1_timetable[i]) if bus1_arr_t[i] == 0
      bus_co2_trip = 0
      num_bus += 1
      bus1_route.each_with_index do |x, j|
        bus1_s_v[i][j] = (dist.to_f /
          (bus1_s_arr_t[i][x] - bus1_s_st_t[i][x])) * ms_to_kmh
      end
      bus1_v[i] = ((bus1_route.length * dist).to_f /
        (bus1_arr_t[i] - bus1_timetable[i])) * ms_to_kmh
      bus_total_v += bus1_v[i]
      bus1_s_v[i].each do |j|
        bus_co2_trip += (co2_petrol / ((j * conv_fuel_slope) +
          bus_fuel_intercept)) * (dist.to_f / 1000)
      end
```

```
      bus_co2 += bus_co2_trip
    end
  end
  bus2_timetable.length.times do |i|
    if bus2_timetable[i] <= arr_times.max
      total_vmt_bus += bus2_route.length * dist
      update_bus_progress(2, bus2_timetable[i]) if bus2_arr_t[i] == 0
      bus_co2_trip = 0
      num_bus += 1
      bus2_route.each_with_index do |x, j|
        bus2_s_v[i][j] = (dist.to_f /
          (bus2_s_arr_t[i][x] - bus2_s_st_t[i][x])) * ms_to_kmh
      end
      bus2_v[i] = ((bus2_route.length * dist).to_f /
        (bus2_arr_t[i] - bus2_timetable[i])) * ms_to_kmh
      bus_total_v += bus2_v[i]
      bus2_s_v[i].each do |j|
        bus_co2_trip += (co2_petrol / ((j * conv_fuel_slope)) +
          bus_fuel_intercept) * (dist.to_f / 1000)
      end
      bus_co2 += bus_co2_trip
    end
  end
  bus3_timetable.length.times do |i|
    if bus3_timetable[i] <= arr_times.max
      total_vmt_bus += bus3_route.length * dist
      update_bus_progress(3, bus3_timetable[i]) if bus3_arr_t[i] == 0
      bus_co2_trip = 0
      num_bus += 1
      bus3_route.each_with_index do |x, j|
        bus3_s_v[i][j] = (dist.to_f /
          (bus3_s_arr_t[i][x] - bus3_s_st_t[i][x])) * ms_to_kmh
      end
      bus3_v[i] = ((bus3_route.length * dist).to_f /
        (bus3_arr_t[i] - bus3_timetable[i])) * ms_to_kmh
      bus_total_v += bus3_v[i]
      bus3_s_v[i].each do |j|
        bus_co2_trip += (co2_petrol / ((j * conv_fuel_slope) +
          bus_fuel_intercept)) * (dist.to_f / 1000)
      end
      bus_co2 += bus_co2_trip
    end
  end
  conv_average_v = conv_total_v.to_f / num_conv
  elec_average_v = elec_total_v.to_f / num_elec
  bus_average_v = bus_total_v.to_f / num_bus
  total_average_v = (conv_total_v + elec_total_v + bus_total_v).to_f /
    (num_conv + num_elec + num_bus)
  conv_impact, bus_impact, elec_impact = conv_co2, bus_co2, elec_co2
  env_impact = conv_impact + bus_impact + elec_impact

  # print summary data to csv
  CSV.open(summary_fname, "a+") do |csv|
    csv << ["Bus VMT", total_vmt_bus.to_f / 1000]
```

```ruby
    csv << []
    csv << ["", "Petrol", "Elec."]
    csv << ["CO2 Production", "#{co2_petrol} kg / L", "#{co2_elec} kg / kWh"]
    csv << []
    csv << ["", "Total", "Conv.", "Bus", "Elec."]
    csv << ["Number",
             (num_conv + num_bus + num_elec), num_conv, num_bus, num_elec]
    csv << ["Avg. Vel.",
             total_average_v, conv_average_v, bus_average_v, elec_average_v]
    csv << ["Environmental Impact",
             "#{env_impact} kg", "#{conv_impact} kg",
             "#{bus_impact} kg", "#{elec_impact} kg"]
  end
end

# print summary of simulation data to csv
def print_summary
  trip_times = []
  start_times.length.times { |i| trip_times << (arr_times[i] - start_times[i]) }
  modes = ["Total", "Car", "Bus", "Taxi", "Motorbike", "Bicycle", "AV"]
  total_trips = [0, 0, 0, 0, 0, 0, 0]

  # determine number of trips for each mode
  trip_times.length.times do |i|
    modes.length.times do |j|
      if j == 0
        total_trips[j] += 1
      else
        total_trips[j] += 1 if new_mode_choices[i] == modes[j]
      end
    end
  end

  trip_data = Array.new
  examined_mode = ""
  if demand_shift == 100
    trip_data << [
      "AV", "Number", "First Trip", "Average Velocity", "Last Trip",
      "Average Velocity", "Velocity Change"]
    examined_mode = "AV"
  else
    trip_data << [
      "Car", "Number", "First Trip", "Average Velocity", "Last Trip",
      "Average Velocity", "Velocity Change"]
    examined_mode = "Car"
  end

  first_trip_time_total, last_trip_time_total = 0, 0
  first_trip_vel_total, last_trip_vel_total = 0, 0
  vel_diff_total = 0
  8.times do |j|
    journey_array = Array.new
    new_mode_choices.length.times do |i|
      if j < 4
```

```
        if new_mode_choices[i] == examined_mode &&
          origins[i] == j && dests[i] == 0
          journey_array << (arr_times[i] - start_times[i]).round(2)
        end
      else
        if new_mode_choices[i] == examined_mode &&
          origins[i] == (j - 4) && dests[i] == 1
          journey_array << (arr_times[i] - start_times[i]).round(2)
        end
      end
    end
    first = journey_array[0]
    last = journey_array[-1]
    length = rts_secs[j] * dist
    first_v = 0
    last_v = 0
    if j > 3 && examined_mode == "Car"
      first_v = ((length.to_f / (first - parking)) * ms_to_kmh).round(2)
      last_v = ((length.to_f / (last - parking)) * ms_to_kmh).round(2)
    else
      first_v = ((length.to_f / first) * ms_to_kmh).round(2)
      last_v = ((length.to_f / last) * ms_to_kmh).round(2)
    end
    count = journey_array.length
    first_trip_time_total += first
    first_trip_vel_total += first_v
    last_trip_time_total += last
    last_trip_vel_total += last_v
    vel_diff_total += (last_v - first_v)
    trip_data << [
      "Journey #{j + 1}", count, first, first_v, last, last_v,
      (last_v - first_v)]
  end
  trip_data << [
    "Average", "", (first_trip_time_total / 8),
    (first_trip_vel_total / 8), (last_trip_time_total / 8),
    (last_trip_vel_total / 8), (vel_diff_total / 8)]

  total_velocity, total_velocity_conv = 0, 0
  total_velocity_bus, total_velocity_av = 0, 0
  velocities.length.times do |i|
    total_velocity += velocities[i]
    total_velocity_conv += velocities[i] if new_mode_choices[i] == "Car"
    total_velocity_conv += velocities[i] if new_mode_choices[i] == "Taxi"
    total_velocity_bus += velocities[i] if new_mode_choices[i] == "Bus"
    total_velocity_av += velocities[i] if new_mode_choices[i] == "AV"
  end
  average_velocity = total_velocity.to_f / velocities.length
  average_velocity_conv = total_velocity_conv.to_f / (total_trips[1] +
    total_trips[3])
  average_velocity_bus = total_velocity_bus.to_f / total_trips[2]
  average_velocity_av = total_velocity_av.to_f / total_trips[6]

  total_vmt_conv, total_vmt_av = 0
```

```ruby
  new_mode_choices.length.times do |i|
    if new_mode_choices[i] == "Car" || new_mode_choices[i] == "Taxi"
      j = origins[i] + (dests[i] * 4)
      total_vmt_conv += rts_secs[j] * dist
    elsif new_mode_choices[i] == "AV"
      j = origins[i] + (dests[i] * 4)
      total_vmt_av += rts_secs[j] * dist
    end
  end

  # print summary data to csv
  CSV.open(summary_fname, "a+") do |csv|
    csv << []
    csv << ["", "Total", "Conventional", "Bus", "AV"]
    csv << [
      "Average Velocity", average_velocity,
      average_velocity_conv, average_velocity_bus, average_velocity_av]
    csv << []
    [modes].each { |i| csv << i }
    [total_trips].each { |i| csv << i }
    csv << []
    trip_data.each { |i| csv << i }
    csv << []
    csv << ["Conv. VMT", total_vmt_conv.to_f / 1000]
    csv << ["AV VMT", total_vmt_av.to_f / 1000]
  end
end

def calculate_typical_trip_time(
  start_time, mode_choice, dest, route_segments, start_segment, bus)
  route_distance = route_segments * dist
  intersection_waits = (route_segments - 1) * intersec_wait
  typical_velocity = (30 * kmh_to_ms)
  bus_wait = calculate_bus_wait_time(start_time, start_segment, bus)
  typical_trip_time = route_distance.to_f / typical_velocity
  case mode_choice
  when "Bus"
    typical_trip_time += bus_wait
    typical_trip_time += intersection_waits
  when "Car"
    if dest == 0
      typical_trip_time *= calculate_car_trip_time_adj
      typical_trip_time += intersection_waits
    else
      typical_trip_time += intersection_waits
      typical_trip_time += parking
    end
  when "AV"
    typical_trip_time += intersection_waits
  when "Bicycle"
    typical_trip_time = route_distance / bike_v.to_f
  when "Motorbike"
    typical_trip_time = route_distance / motbike_v.to_f
  when "Taxi"
```

```
    typical_trip_time *= taxi_pen
    typical_trip_time += intersection_waits
  end
  typical_trip_time
end

def update_bus_progress(bus_number, start_time)
  # update arrival times / segment progression for started bus trips
  if bus_number == 1
    bus1_arr_t.length.times do |i|
      if bus1_arr_t[i] == 0
        if bus1_timetable[i] <= start_time
          segment_trip_time = 0
          bus_arrival_time = bus1_timetable[i]
          bus1_route.each do |j|
            bus1_s_st_t[i][j] = bus_arrival_time
            segment_trip_time = 0
            if j != bus1_route[-1]
              segment_trip_time = calculate_trip_time(bus_arrival_time, j) +
                intersec_wait
            else
              segment_trip_time = calculate_trip_time(bus_arrival_time, j)
            end
            bus_arrival_time += segment_trip_time
            bus1_s_arr_t[i][j] = bus_arrival_time
          end
          bus1_arr_t[i] = bus_arrival_time
        end
      end
    end
  elsif bus_number == 2
    bus2_arr_t.length.times do |i|
      if bus2_arr_t[i] == 0
        if bus2_timetable[i] <= start_time
          segment_trip_time = 0
          bus_arrival_time = bus2_timetable[i]
          bus2_route.each do |j|
            bus2_s_st_t[i][j] = bus_arrival_time
            segment_trip_time = 0
            if j != bus2_route[-1]
              segment_trip_time = calculate_trip_time(bus_arrival_time, j) +
                intersec_wait
            else
              segment_trip_time = calculate_trip_time(bus_arrival_time, j)
            end
            bus_arrival_time += segment_trip_time
            bus2_s_arr_t[i][j] = bus_arrival_time
          end
          bus2_arr_t[i] = bus_arrival_time
        end
      end
    end
  else
    bus3_arr_t.length.times do |i|
```

```
        if bus3_arr_t[i] == 0
          if bus3_timetable[i] <= start_time
            segment_trip_time = 0
            bus_arrival_time = bus3_timetable[i]
            bus3_route.each do |j|
              bus3_s_st_t[i][j] = bus_arrival_time
              segment_trip_time = 0
              if j != bus3_route[-1]
                segment_trip_time = calculate_trip_time(bus_arrival_time, j) +
                  intersec_wait
              else
                segment_trip_time = calculate_trip_time(bus_arrival_time, j)
              end
              bus_arrival_time += segment_trip_time
              bus3_s_arr_t[i][j] = bus_arrival_time
            end
            bus3_arr_t[i] = bus_arrival_time
          end
        end
      end
    end
  end
end
```

traffic-sim-generate.rb

```
require_relative 'traffic-sim-paths'

class Generate

  attr_reader :full_map

  def initialize
    # set up map (6 x 6 nodes, 0-indexed labels)
    @full_map = [
      [0, 1, 2, 3, 4, 5],
      [6, 7, 8, 9, 10, 11],
      [12, 13, 14, 15, 16, 17],
      [18, 19, 20, 21, 22, 23],
      [24, 25, 26, 27, 28, 29],
      [30, 31, 32, 33, 34, 35]]
  end

  # generate 100 random starting times
  def generate_start_times(runs, start_interval)
    start_times = Array.new(runs, 0)
    start_times.length.times do |i|
      i > 0 ? start_times[i] = start_times[i - 1] + start_interval : start_times[i]
= start_interval
    end
    start_times
  end
```

```ruby
# generate random origins
def generate_origins(runs)
  origins = Array.new
  runs.times { origins << rand(4) }
  origins
end

# generate random dests
def generate_destinations(runs)
  dests = Array.new
  runs.times { dests << rand(2) }
  dests
end

# generate bus timetable (20 times per day)
def generate_bus_timetable(number)
  bus_timetable = Array.new(number, 0)
  bus_timetable.length.times do |i|
    bus_timetable[i] = rand(60) + (i * ((60 * 60 * 24) / number))
  end
  bus_timetable
end

def generate_paths_final(map, direction = 'f')
  trip_paths = Paths.new(map)
  if direction == 'r'
    trip_paths.generate_paths_reverse(1, 0, "")
    trip_paths.transform_paths_i
    trip_routes = trip_paths.transform_paths_routes_reverse
  else
    trip_paths.generate_paths(0, 0, "")
    trip_paths.transform_paths_i
    trip_routes = trip_paths.transform_paths_routes
  end
end

def generate_map_0(j)
  map_0 = Array.new
  (@full_map.length - j).times { |i| map_0 << @full_map[i + j][0..4] }
  map_0
end

def generate_map_1(j)
  if j < 4
    map_1 = @full_map[j..3]
  else
    map_1 = @full_map[3..j]
  end
end

# print headers to csv files
def generate_output_file_headers(sim_out_fname, time_ev_n_fname)
  # output data file headers
  CSV.open(sim_out_fname, "w") do |csv|
```

```
    csv << ["No.", "Journey", "O", "D", "Distance", "Route", "Mode",
      "New Mode", "Travel Time", "Average Velocity", "Number of Vehicles",
      "Typical Trip Time"]
  end
  # time evolution data file headers
  ev_labels = ["t"]
  seg_labels = Array.new
  60.times { |i|
    ev_labels << "#{i}"
    seg_labels << "#{i}" }
  CSV.open(time_ev_n_fname, "w") { |csv| [ev_labels].each { |i| csv << i } }
end

def generate_mode_choices(origins, dests, dist)
  dists = Array.new(origins.length, 0)
  origins.length.times do |i|
    case origins[i]
    when 0
      dests[i] == 0 ? dists[i] = dist * 8 : dists[i] = dist * 7
    when 1
      dests[i] == 0 ? dists[i] = dist * 7 : dists[i] = dist * 6
    when 2
      dests[i] == 0 ? dists[i] = dist * 6 : dists[i] = dist * 5
    when 3
      dests[i] == 0 ? dists[i] = dist * 5 : dists[i] = dist * 6
    end
  end
  mode_choices = Array.new
  origins.length.times do |i|
    mode_choices << determine_mode_choice_prob(dists[i], dests[i])
  end
  mode_choices
end

def determine_mode_choice_prob(route_distance, destination)
  # demand estimation AV, car, motorbike, bus, bicycle, taxi
  station_demand = [
    [0.0, 6.5, 0.0, 6.0, 88.7, 0.0, 4.2],
    [0.0, 9.1, 1.9, 1.9, 83.5, 0.0, 3.6],
    [0.0, 16.2, 2.6, 4.9, 69.6, 1.6, 5.2],
    [0.0, 25.9, 2.3, 25.2, 37.5, 1.6, 7.4],
    [0.0, 33.0, 2.6, 35.3, 18.8, 1.6, 8.7],
    [0.0, 35.0, 2.6, 37.5, 13.9, 1.3, 9.7]]
  shop_demand = [
    [0.0, 12.3, 0.6, 0.6, 84.1, 0.0, 2.3],
    [0.0, 18.4, 1.9, 1.6, 74.1, 0.0, 3.9],
    [0.0, 33.3, 1.3, 3.9, 55.7, 0.3, 5.5],
    [0.0, 45.3, 1.6, 11.7, 32.4, 0.3, 8.7],
    [0.0, 50.5, 1.6, 17.8, 19.7, 0.6, 9.7],
    [0.0, 52.8, 1.9, 20.4, 14.2, 0.3, 10.4]
  ]
  weights = Array.new
  if destination == 0
    if route_distance < 600
```

```
        weights = station_demand[0]
      elsif route_distance < 1100
        weights = station_demand[1]
      elsif route_distance < 1600
        weights = station_demand[2]
      elsif route_distance < 2100
        weights = station_demand[3]
      elsif route_distance < 2600
        weights = station_demand[4]
      else
        weights = station_demand[5]
      end
    else
      if route_distance < 600
        weights = shop_demand[0]
      elsif route_distance < 1100
        weights = shop_demand[1]
      elsif route_distance < 1600
        weights = shop_demand[2]
      elsif route_distance < 2100
        weights = shop_demand[3]
      elsif route_distance < 2600
        weights = shop_demand[4]
      else
        weights = shop_demand[5]
      end
    end
    modes = ["AV", "Car", "Motorbike", "Bus", "Bicycle", "Taxi", "Other"]
    cum_weights = weights
    (weights.length - 1).times { |i| cum_weights[i + 1] += cum_weights[i] }
    rand_n = rand(100)
    i = 0
    i += 1 while cum_weights[i] <= rand_n
    rand_mode = modes[i]
    rand_mode == "Other" ? mode = modes[rand(1...6)] : mode = rand_mode
    mode
  end
end
```

traffic-sim-paths.rb

```
class Paths

  attr_reader :map
  attr_reader :row_count
  attr_reader :col_count
  attr_accessor :paths
  attr_accessor :paths_i
  attr_accessor :routes

  def initialize(map)
    @map = map
    @row_count = @map.length # total number of rows (m)
```

```ruby
    @col_count = @map[0].length # total number of columns (n)
    @paths = [] # empty array to store all possible routes
    @paths_i = []
    @routes = []
  end

  # method to generate all possible paths between top left and
  # bottom right of input map
  def generate_paths(current_row, current_col, path)
    if current_row == row_count - 1
      i = current_col
      while i < col_count
        path += "-#{map[current_row][i]}"
        i += 1
      end
      paths << path
      return
    end

    if current_col == col_count - 1
      i = current_row
      while i < row_count
        path += "-#{map[i][current_col]}"
        i += 1
      end
      paths << path
      return
    end

    path += "-#{map[current_row][current_col]}"
    generate_paths(current_row + 1, current_col, path)
    generate_paths(current_row, current_col + 1, path)
  end

  # method to generate all possible paths between bottom left and
  # top right of input map
  def generate_paths_reverse(current_row, current_col, path)
    if current_row == 0
      i = current_col
      while i < col_count
        path += "-#{map[current_row][i]}"
        i += 1
      end
      paths << path
      return
    end

    if current_col == col_count - 1
      i = current_row
      while i >= 0
        path += "-#{map[i][current_col]}"
        i -= 1
      end
      paths << path
```

```ruby
      return
    end

    path += "-#{map[current_row][current_col]}"
    generate_paths_reverse(current_row - 1, current_col, path)
    generate_paths_reverse(current_row, current_col + 1, path)
  end

  # transform array of strings to arrays of integers
  def transform_paths_i
    paths.length.times do |i|
      path = paths[i].split("-")
      path.shift
      path.map! do |j|
        j.to_i
      end
      paths_i << path
    end
  end

  # transform path array from node routes to segment routes
  def transform_paths_routes
    paths_i.length.times do |i|
      path = Array.new
      (paths_i[0].length - 1).times do |j|
        if (paths_i[i][j] + 1) == (paths_i[i][j + 1]) # columns
          if paths_i[i][j] < 5
            path << paths_i[i][j]
          elsif paths_i[i][j] < 11
            path << paths_i[i][j] - 1
          elsif paths_i[i][j] < 17
            path << paths_i[i][j] - 2
          elsif paths_i[i][j] < 23
            path << paths_i[i][j] - 3
          elsif paths_i[i][j] < 29
            path << paths_i[i][j] - 4
          elsif paths_i[i][j] < 35
            path << paths_i[i][j] - 5
          else
            puts "Route error!"
          end
        elsif (paths_i[i][j] + 6) == (paths_i[i][j + 1]) # rows
          path << paths_i[i][j] + 30
        else
          puts "Route error!"
        end
      end
      routes << path
    end
    routes
  end

  # transform path array from node routes to segment routes
  # for reverse trip
```

```ruby
  def transform_paths_routes_reverse
    paths_i.length.times do |i|
      path = Array.new
      (paths_i[0].length - 1).times do |j|
        if (paths_i[i][j] + 1) == (paths_i[i][j + 1]) # columns
          if paths_i[i][j] < 5
            path << paths_i[i][j]
          elsif paths_i[i][j] < 11
            path << paths_i[i][j] - 1
          elsif paths_i[i][j] < 17
            path << paths_i[i][j] - 2
          elsif paths_i[i][j] < 23
            path << paths_i[i][j] - 3
          elsif paths_i[i][j] < 29
            path << paths_i[i][j] - 4
          elsif paths_i[i][j] < 35
            path << paths_i[i][j] - 5
          else
            puts "Route error!"
          end
        elsif (paths_i[i][j] - 6) == (paths_i[i][j + 1]) # rows
          path << paths_i[i][j] + 24
        else
          puts "Route error!"
        end
      end
      routes << path
    end
    routes
  end
end
```

traffic-sim-calculate.rb

```ruby
# calculate largest number of a set
def calculate_largest(inputs)
  largest = inputs[0]
  inputs.length.times { |i| largest = inputs[i] if inputs[i] > largest }
  largest
end

# determine route choice based on which route has the max utility
def determine_route_choice(trip_utilities, trip_utility)
  route_choice = 0
  until trip_utilities[route_choice] == trip_utility
    route_choice += 1
  end
  route_choice
end

# calculate arrival time based on mode choice, start time, and trip times
def calculate_arrival_time(
  mode_choice, start_time, trip_time, dest, intersections, intersection_wait)
```

```ruby
  if mode_choice == "Car" && dest == 0
    trip_time -= intersections * intersection_wait
    trip_time /= calculate_car_trip_time_adj
    trip_time += intersections * intersection_wait
  end
  arrival_time = start_time + trip_time
end

# calculate adjusted time calculation for car journeys
def calculate_car_trip_time_adj
  time_val_outbound, time_val_return = 1, 1
  car_trip_time_adjustment = (2 * time_val_outbound) + time_val_return
end

# generate random dummy variables for each route
def generate_route_dummies(routes)
  route_dummies = Array.new
  routes.length.times { |i| route_dummies << rand(2) }
  route_dummies
end

# calculate number of turns for each route
def calculate_route_turns(routes)
  route_turns = Array.new
  routes.length.times do |i|
    turns = 0
    routes[i].length.times do |j|
      if (routes[i][j] + 1 != routes[i][j + 1]) ||
         (routes[i][j] + 6 != routes[i][j + 1])
        turns += 1
      end
    end
    route_turns << turns
  end
  route_turns
end

# calculate utility for each route
def calculate_utility(
  route_times, route_distance, route_dummies, route_turns)
  time_coeff = -0.224653
  distance_coeff = -0.285347
  route_coeff = 0.6493104
  turns_coeff = -0.14844
  utilities = Array.new
  route_times.length.times do |i|
    utility = (time_coeff * route_times[i]) +
      (distance_coeff * route_distance) +
      (route_coeff * route_dummies[i]) + (turns_coeff * route_turns[i])
    utilities << utility
  end
  utilities
end
```

```ruby
def sum_route_s_trip_times(routes, route_segment_trip_times)
   # calculate total trip time for each route
  route_trip_times = Array.new(routes.length, 0)
  routes.length.times do |i|
    routes[0].length.times do |j|
      route_trip_times[i] +=
        route_segment_trip_times[(j * routes.length) + i]
    end
  end
  route_trip_times
end
```