International Journal of Engineering Education Vol. 32, No. 1(B), pp. 542–552, 2016
Printed in Great Britain

0949-149X/91 $3.00+0.00

Application of Video Game Artificial Intelligence Techniques for Design of a Simulation Software System for Transportation Engineering Education*

M. T. CHAN, J. T. CHAN, C. GELOWITZ and C. W. CHAN
Faculty of Engineering and Applied Science, University of Regina, Regina SASK S4S 0A2. Canada. E-mails: {chan99ma, chan111j, craig.gelowitz, christine.chan}@uregina.ca

The objective of the paper is to present application of some artificial intelligence (AI) principles and techniques for design of a simulation system for teaching concepts in transportation engineering. The AI techniques can support development of key features of the simulation system, such as a road system, a vehicle, rectilinear and curvilinear motion, braking patterns, etc. By adopting a simulation system for learning, students, as users of the system, can actively participate in constructing graphic scenarios that illustrate different concepts in transportation engineering. The simulation system can support visualizations of various scenarios, which illustrate concepts often presented as simple equations in a traditional lecture hall setting. In addition to discussing application of the AI techniques for designing the simulation system, the paper also presents an overview of research studies related to gamification of educational software systems and its application in transportation engineering education.

Keywords: simulation software; artificial intelligence techniques; transportation engineering education; video game

1. Introduction

The need to improve pedagogy in science and engineering education is widely recognized as an important goal. A discussion that mainly refers to science and engineering education in the United States indicated that almost all students (98%) who dropped out of engineering and 86% of those who continued their studies cite "poor teaching by faculty" as a major concern [1]. The poor pedagogy to which they refer is essentially the lecture format characteristic of undergraduate science and engineering classes [2]. Canada similarly faces a shortage of highly trained scientists and engineers. A report published by Council of Canadian Academies, which is entitled "Enabling sustainability in an interconnected world", suggests that in the field of information, communications, and technologies or ICT, Canada faces projected shortages in ICT talent on both the supply and demand sides: "On the demand side, there is a growing need for ICT professionals in the ICT sector as well as in other sectors such as finance, education, and manufacturing [3]. On the supply side, universities (a key talent pool for the ICT sector) may not be producing sufficient graduates in ICT-related fields, such as computer science and engineering" [4].

Gamification of science and engineering education has the potential to alleviate this problem of poor pedagogy. Gamification refers to the application of game principles and mechanics to non-game contexts so as to make practical activities or tasks more fun and enjoyable. The intent of gamification is to change "the way in which specific activities and processes operate" by redesigning work processes with game mechanisms for a fun and enjoyable experience [5]. Hence, applying gamification to science and engineering education has the potential to not only represent and communicate difficult concepts in a more understandable and possibly, fun format, the approach can enhance effectiveness of the learning process because students are more likely to undertake the learning activities when they are enjoyable, and as a consequence, learning outcomes will be improved.

The objective of this paper is to discuss how some design principles and artificial intelligence (AI) techniques used for developing a contemporary video game in car racing can be useful for developing gamified simulation software systems for teaching transportation engineering. This paper proceeds as follows. Section 2 describes some background literature on gamified simulation systems for transportation engineering education. Section 3 presents the design principles and AI techniques for a video game, *Racer*, which can be adapted for development of gamified simulation systems for teaching concepts in transportation engineering. Section 4 presents a discussion about the benefits of using AI techniques supported in the Unity game engine for the design and development of an educational simulation system in transportation engineering. Section 5 includes the conclusion and also discusses some directions for future work.

* Accepted 11 August 2015.

2. Background literature

Computer simulation and visualization can play an important role in science and engineering education. Mayo believed gamification in science and engineering education can enhance of student motivation, and that video games have the potential to address systemic deficiencies in science and engineering education, often conducted in the traditional class lecture format. He included five reasons for this potential: (i) massive reach of the online game compared to the number of people that sit together in a classroom, (ii) video games promote experimental and inquiry-based learning, which are effective learning paradigms for teaching science and engineering subjects, (iii) playing a video game is found to trigger the release of dopamine in the brain, which better chemically "primes" the brain for learning, (iv) the engaging effect of a video game often means students spend more time on the task of learning, and (v) several studies have documented that learning outcomes of games are higher than those of the lecture format. For example it was documented that the game of Geography Explorer increased learning outcomes by 15–40% and the game of Virtual Cell increased learning outcomes by 30–63% [2].

For transportation engineering in particular, a significant amount of research work has taken place at University of Minnesota in U.S.A., where the suite of web-based simulation modules called Simulation Transportation for Realistic Engineering Education and Training (STREET), was developed. The objective of the STREET project was to enable University of Minnesota to become the epicenter for development of simulation-based teaching materials that provide undergraduate students with an interactive learning environment in transportation engineering. STREET provides modules in traffic simulation, geometric design, demand modeling, and include the modules of: ROAD (Roadway Online Application for Design), OASIS (Online Application for Signalized Intersection Simulation), ADAM (Agent-based Demand and Assignment Model), SONG, SAND (Simulation and Analysis of Network Design), SOFT (Simulation of Freeway Traffic), CLUSTER (Clustered Locations of Urban Services, Transport, and Economic Resources), ABODE (Agent Based Model of Origin Destination Estimation) and ANGIE (Agent-based Network Growth model with Incremental Evolution) [6].

The STREET project is associated with a number of evaluation studies. For example, Liao et al. [7, 8] discusses the interactive traffic simulation environment, which supports pre-generated traffic scenarios that were incorporated into an undergraduate civil engineering class. By collecting feedback from instructors and students, enhancements were made to the simulation tool so that users can make changes to the model and examine the impacts on traffic. The web-based traffic simulation framework can incorporate large road networks and supports consideration of multiple factors involved in traffic management strategies, making it a useful tool for training transportation professionals. Liao et al. [9] discussed using the traffic control simulation framework as the foundation for developing an enhanced traffic game, whose goal was to interest high school students in transportation careers. The approach involved integration of educational game modules into the curricula for teaching intelligent transportation control and management. A developed curriculum was evaluated through observation of student participation and engagement during five summer camp sessions in 2008 and 2009, and the survey results indicated student excitement toward the game and their positive receptiveness to the new traffic engineering curriculum.

Liao and Levinson [10] reported on the development of Roadway Online Application for Design (ROAD), which assists students in geometric design of roadways by providing a contour map in the background as a reference on the computer screen. ROAD enabled students to design roadway geometry efficiently and modify the design easily within some given economic and environmental parameters. When the design is completed, ROAD can generate a three-dimensional roadway geometric model, which allows students to test the design by placing themselves in the driver's seat and maneuver through the designed roadway at the maximum designed speed.

Chen and Levinson [11] investigated the efficacy of using simulation for teaching the topic of transportation network growth by conducting an experiment at the Civil Engineering Department of the University of Minnesota. A network growth simulator program (SONG 1.0) was incorporated into a senior/graduate class in transportation system analysis. Instead of aiming to generate a perfect reproduction of the real world SONG 1.0 aims to help students explore the micro-world of transportation network systems so that they are stimulated and encouraged to come up with innovative ways of conceptualizing the network growth and planning process. The experimental results demonstrate that the use of SONG 1.0 effectively enhanced students' in-depth understanding about development of network patterns, which in turn helped to develop their skills in problem-solving and decision-making.

Zhu et al. [12] reported on a transportation planning software package called Agent-Based Demand and Assignment Model (ADAM). The

agent-based modeling approach assumes that the aggregate urban travel demand arises from a multi-dimensional selection process, which includes choices related to residential and business locations, trip origins, trip destinations, routes, etc. All agents in the model are assumed to have individual characteristics, goals, and patterns of travel behaviors. The agents exchange information on their experiences and adjust their travel choices according to the available information. ADAM was incorporated as part of a teaching strategy, and evaluations showed that ADAM helped students in forming opinions, evaluating projects, and making judgments. Students who used ADAM reported an enhanced understanding of the concepts taught in the class, and students who prefer visual and active learning were found to benefit more than others.

Kasaraneni [13] presented a web-based game that demonstrates driver-behaviors at signalized intersections. It was used as a supplementary tool in an introductory course on Transportation Engineering, and helped students understand decisions of drivers at the onset of the yellow signal based on existing traffic conditions. The game was implemented on the .NET Framework using Microsoft Visual C#, ASP, and Microsoft Access (all trademarks of Microsoft, U.S.A.). Based on collected data on student engagement in the course, it was found the game helped students to better focus their attention. The creative instructional activities also provided students with hands-on experience of simulated real world scenarios.

The game discussed in [13] was further developed in [14] with the objective of investigating game-aided pedagogy for improving engagement and learning outcomes of students in transportation engineering. The enhanced game consists of three layers: (i) a web-based user interface to display simulated traffic flows, (ii) a game control layer to implement logic controls based on the game context and user commands, and (iii) a data source layer, which consists of the two databases of traffic snapshots and traffic scenarios. Assessment of student engagement and learning outcomes was conducted by means of questionnaires and statistical analysis on the collected data indicated the game enhanced learning outcomes.

Fang and Pines [15] studied how the traditional learning experience can be enhanced with use of the computer tools at the University of Hartford. A learning environment that incorporates two simulation tools supports students in learning about traffic flow theory and advanced control strategies. The two traffic simulation tools simulate the interactive dynamics among driver behaviors, vehicle characteristics and advanced traffic control management strategies in urban and freeway transportation networks. The two models support analysis of traffic operations related to a full range of functionally classified roadways such as freeways, surface streets and basic transit systems. The conclusion of the investigation is that the simulation technology-enhanced learning activities better support students in exploring complex traffic modeling processes and are effective in de-emphasizing the more passive teaching mode of instructor-led "chalk and talk".

Shi et al. [16] proposed a simulation software platform that supports easier incorporation of customized models and more user-friendly graphic interfaces for development of simulation systems in transportation engineering. The software platform was developed based on structural design patterns such as the Model-View-Controller pattern, and the advantages of the platform design are demonstrated using browser techniques to animate traffic on an online map.

Shi et al. [17] presented some models that were built on the web-based simulation platform discussed in [16]. The models tackle different tasks such as vehicular behavior modeling, vehicular dynamics implementation, testing of routing algorithms, vehicle generation modeling, traffic flow simulation, etc. The simulation platform is shown to reduce the workload of simulation-developers by providing a specialized and compact data structure, an intricate mechanism to inject customized models, a user-friendly web interface for both configuring and coding the program, and useful automation procedures. The simulation platform consists of the three modules of SimModel, SimEngine and SimFramework, which support development of modules for data modeling, simulation flow, and user interfaces, respectively. The platform's open source and open structure enable users to easily tailor a complicated simulation program by extending the default data and simulation models, and supports automatic network generation from web Geographic Information System or GIS applications. Kim [18] investigated the value of gamification on undergraduate education and found that gamification is more effective in motivating the learning desire, improving the level of communication and understanding, and reducing the stress of learning in engineering education. In this context, we believe that elements of game design and technology are useful for developing gamified simulation software programs. In the next section, some design principles and artificial intelligence techniques used in a contemporary car racing video game will be discussed in terms of their applicability for developing a simulation program for transportation engineering education.

3. Design and implementation of a simulation software system for transportation engineering

Concepts in an introductory level transportation-engineering course are often described or are expressed as equations in textbooks and lectures. Students often assume the role of the passive audience. We suggest that these concepts can be presented visually to students, who would assume the more active role of a participating user of a simulation software system. With the visualized animation that gamification enables, the concepts can become alive in the classroom and students can actively determine specific properties of the objects so that they can more easily understand the concepts in transportation engineering. This section presents the design and implementation of some common components and features of a simulation software system for teaching transportation engineering. The implementation platform for the design is the game engine of Unity [19].

3.1 General features of a simulation software in transportation engineering

A simulation system for teaching transportation engineering concepts provides a platform in which different scenarios about transportation and traffic can be visualized. Some common entities or concepts involved in any transportation system include: (a) a road system, (b) one or more vehicles, (c) the concept of driving, and (d) obstructions on the road, such as a wall, a sign post, traffic lights, etc. With these basic entities, different scenarios that illustrate various transportation engineering concepts can be configured or illustrated. In the following, the design of the gamified components of these four features will be presented.

3.1.1 A road system

The design of a road system consists of two considerations: (i) the road itself, and (ii) some way to navigate on the road. The road gives the users a 3D graphical representation of where the cars are allowed to drive; it also provides a path on which the vehicles can drive on and navigate through turns and corners. The latter consideration can be implemented using the waypoint system in the Unity game engine. The waypoints specify key points on the road. These waypoints work in conjunction with a vehicle's driving system to enable the vehicle to drive on the road.

3.1.2 Vehicles

A vehicle that travels on a road system can be of diverse types such as a car, a recreational vehicle or RV, a truck, a taxi, a semi, a smart car, and the vehicle can have varying weights, heights, velocities, and tires. Since all different types of vehicles can travel on a road system, it is important that diverse types of vehicles can be supported in the simulation software system. It is also important that the user, or the student, be given the option to define a type of vehicle with specific properties should the necessity arises.

The game engine of Unity (trademark of Unity Technologies) supports three components that enable the vehicle to be defined according to user preferences; they are (i) the rigid-body component, (ii) the wheel-collider component, and (iii) the car body mesh collider component.

The rigid-body component enables the user to modify properties of the rigid body object and how physical forces are exerted upon it. Using the rigid body interface, the user can modify the properties of a car such as its mass, drag, and angular drag, as well as its kinetic properties of velocity, angular velocity, rotation, and angular rotation. Fig.1 shows the rigid body interface, which enables users to define the values of the properties of a car.

The component of the wheel collider in Unity supports the high-level abstraction implementation of wheel-type objects. This component enables users to specify the various properties of a car that directly affects the frictions between the tires and a road, and hence, a car's ability to navigate on a road system. In a simulation software system, the user can control the values of the properties of forward and sideways frictions, which are critical for determining how a car drives on a road and in simulating its braking distance. Figure 2 shows the interface of the wheel collider component in Unity, which enables the user to provide input to the various properties of the wheel-collider component.

For example, the coefficient of sideways friction of the wheel collider component plays an important role in situations when the car may slip, overturn, or when it is travelling in super-elevation or in curvi-

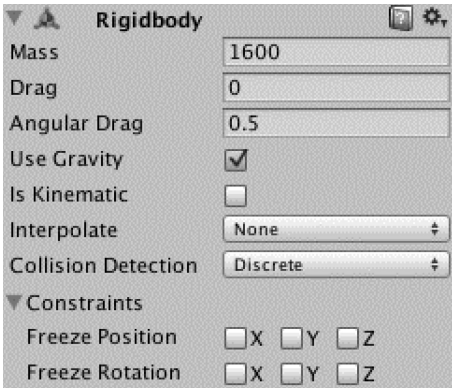

Fig. 1. A screenshot of the interface of the rigidbody component.

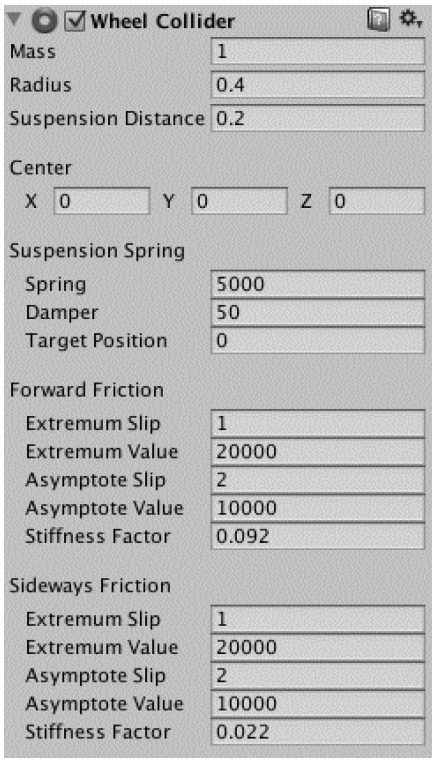

Fig. 2. A screenshot of the interface of the wheel-collider component.

linear motion. On the other hand, when simulating braking distance in rectilinear motion, the coefficient of forward friction is proportional to the forward friction stiffness value in the wheel collider component, which can be defined by the user in the interface. When simulating curvilinear motion, the coefficient of sideways friction is determined by the stiffness value in the parameter of sideways friction of the wheel collider component. Fig.3 is a screenshot of the wheel-collider component of a car object.

The car-body mesh collider component consists of a mesh collider, which detects collisions should a car collides with an object such as a wall or a lamppost. In other words, this component provides data such as the direction from which the impact came as well as the magnitude of the force exerted by the impact. Figure 4 shows the mesh collider that surrounds the car object.

3.1.3 Driving

Driving involves manipulating a car along a road system. Since the road system can be straight or curved, flat (horizontal) or elevated, clear or with obstacles, ensuring the car can travel smoothly along a road can be a complex endeavour. Two game AI techniques useful for ensuring a car can

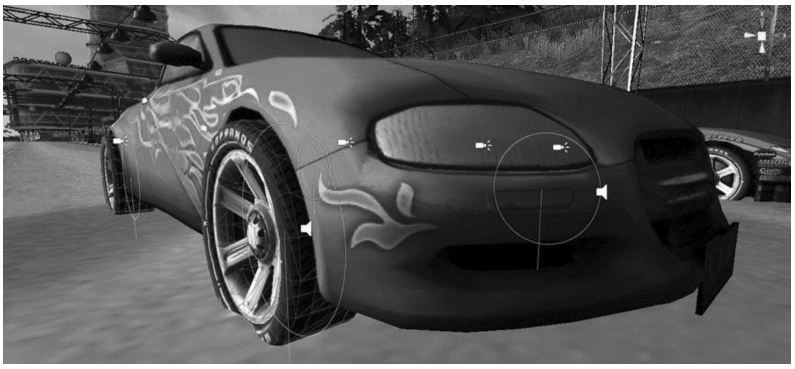

Fig. 3. A screenshot of the wheel colliders (displayed as green rings) on the wheels of a car object.

Fig. 4. A screenshot of the car body mesh collider component.

drive along a clear road, which has no obstacles, include: (i) the waypoint system with vector calculation, and (ii) a conditional monitoring system. They are discussed as follows.

Waypoint system with vector calculations
The simple waypoint system with vector calculations is a technique used for controlling the car. The waypoint system includes a set of waypoints and each waypoint is a coordinate in three-dimensional or 3D space, which represents a key position on the road. This system was used by many early car-racing games because of its effectiveness and simplicity. The waypoints can be stored as a sorted list of positions in 3D space, which can be implemented using the List generic class from the System.Collections.Generic namespace [20]. When a car travels along a road, the list of waypoints is provided to the car as input, and the car iterates through the list until all the waypoints have been passed. These waypoints are placed at key points on the road, and they enable the driving system of a car to know which direction to steer while driving on the road. To enable the car to drive along a road, the simulation system iterates to the next waypoint in the list when it detects that the car is within a specified distance from its current waypoint. Fig.5 shows a screenshot of the waypoints at key positions on a road.

In order to turn the car towards its current waypoint, the system performs a series of vector calculations, which determine the amount of steering that needs to be provided to the car. These calculations are based on an initial vector created by both the position of the current waypoint as well as the current position of the car itself. The series of calculations produces an output vector, which provides the basis for the steering and braking output levels of the car.

However, possibly due to the nonlinear relationship between the input and output vectors, using only the method of waypoints with vector calculations cannot control the car. Then, it becomes necessary to augment the waypoint system with the conditional monitoring system.

Conditional monitoring system
Due to the nonlinear relationship between the input and output vectors, the steering and braking output levels of a car that is travelling on a road system cannot be determined solely from the calculated output vector in the waypoints system. In these instances, the conditional monitoring system can be adopted because it can further refine the steering and braking output levels applied to the car. In other words, depending on the values of the initial calculated output vector produced from the waypoint system's vector calculations, the steering and braking output levels applied to the car can be further adjusted by the conditional monitoring system. When a simulation system adopts both the foundational waypoint system and the conditional monitoring system, a car can travel along a road system satisfactorily.

Instead of adding the conditional monitoring system, the alternative design of deriving and incorporating a nonlinear mathematical function can also be considered. Since this option may require modeling of the mathematical function using artificial neural networks, this alternative is considered overly complex.

3.1.4 Obstacle avoidance

Since a road system can involve obstacles, it is important to include the feature of obstacle avoidance in a simulation software system for teaching concepts about transportation engineering. When traveling on a road system, if the car should encounter some obstruction such as a wall, it needs to avoid it by braking in front of it or driving around it. Before the car can decide what the appropriate response is to the obstacle, it needs to first detect

Fig. 5. A screenshot of waypoints at key positions on a road.

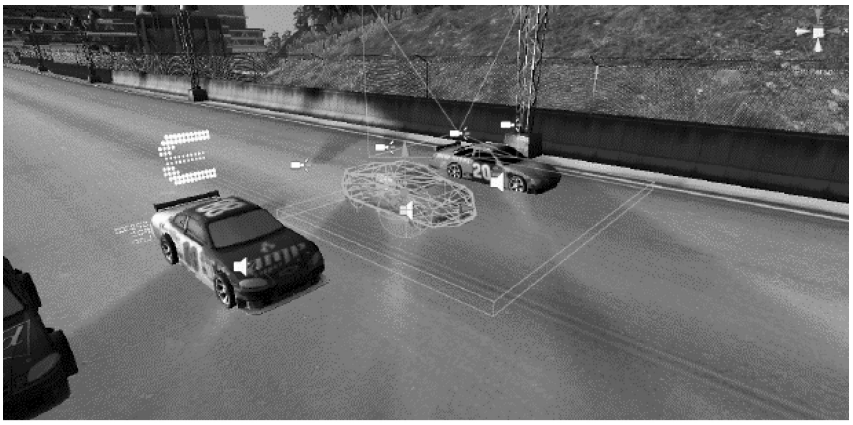

Fig. 6. A screenshot of the trigger detection area attached to a car.

the obstacle. This feature can be included in the simulation system design by making use of the game AI technique of artificial environment perception, which is discussed as follows.

Artificial environment perception by means of trigger detection is a popular mechanism used in contemporary video games, and it supports detecting a specific game object when it is within the vicinity of another game object. This method can determine if the system-controlled car may run into some object such as a wall, and the approach involves defining a trigger area attached to the car. To prevent the collision from happening, the simulation system can adjust the steering and braking output levels applied to the car. If a wall enters into the trigger area of a car, the simulation system becomes aware of where the wall is with respect to the car and it would adjust the output levels accordingly so as to steer the car away from the wall. With this addition of the trigger area, the simulation system can effectively control the car so that it can navigate along a road system.

The trigger detection mechanism is also useful for illustrating the concept of braking distance in both cases of rectilinear motion and curvilinear motion. In rectilinear motion, the brake is applied to halt the car when it is moving in a straight direction. In this case, when the trigger area of a car detects that there is an oncoming object, such as a lamppost, the system can apply the brake gradually with consistent force to bring the car to a halt. Braking in this case is used to halt a car that is about to have a head-on collision. Curvilinear motion requires the brake to be applied intermittently in order to adjust the speed of the car when it is turning on a curved part of the road, ensuring the car can negotiate the turn safely. The trigger area enables the car to do this by sensing if a wall is close to the car. If a wall enters the trigger area, the driving system responds by applying the brake to help the car increase its turning angle and avoid collision with the wall.

An alternative game AI technique considered for implementing artificial environment perception is the ray-casting technique. Ray-casting is a mechanism commonly used in artificial environment perception, which involves drawing a ray or line from an object along a specified direction. In a simulation system, the line or ray will be drawn in the direction the car is heading. This drawn line would be used to determine if the car is about to collide with another object. However, the ray-casting mechanism can only detect collisions in one direction while the trigger detection mechanism can detect collisions in all directions. Hence, the trigger detection mechanism is preferred in the design of the simulation system. Figure 6 shows a screenshot of the trigger detection area attached to a car.

3.2 Implementation of specific concepts in transportation engineering

The four general features of the simulation software system discussed in section 3.1 can be used for implementation of specific concepts in transportation engineering. Three sample concepts and their software implementations will be presented; they are (i) braking distance, (ii) super-elevation of a curved highway, and (iii) the elevation of a point.

3.2.1 Braking distance

Design of a road system involves consideration of the braking distance of a vehicle, which is travelling along a road. The concepts involved can be categorized into linear or curvilinear motion of a vehicle, along a straight or curved road. If it is a curved road, the motion can be further divided into (i) horizontal curvilinear (i.e. on a flat curved road), or (ii) vertical curvilinear (i.e. on a curved road that travels up and down hilly terrain). In a transportation engineering class, the braking distance is often presented as an

equation, which expresses the relationships among the variables of initial velocity or speed, final speed, gravity, coefficient of friction between the road and the tires, and the grade of the road. On the last variable, an uphill road has a positive grade value, and a downhill road has a negative grade value. The equation for braking distance is often used in road design, such that if a velocity of v_o is allowed on the road, then the road must be at least D_b meters long so that cars can come to a stop safely. The equation for braking distance is as follows [21],

$$D_b = \frac{v_o^2 - v^2}{2g(f \pm G)} \tag{1}$$

where D_b = braking distance,
v_o = initial velocity;
v = final velocity;
g = gravity;
f = coefficient of friction;
G = gradient (+ve = uphill, −ve = downhill).

This can be modeled in a simulation software so that the student or user will input values for the velocities, and the equation embedded in the system will determine the length of the road so as to accommodate safe deceleration of a vehicle.

3.2.2 Super-elevation of a curved highway

In designing a curved road, it is important to consider the concept of super-elevation, which refers to the difference in height between the two edges of a curved road. When negotiating a horizontal curved road, the outer edge must be at a slight incline so that a car can travel on it without flying off the road. Typically, the maximum slope or super-elevation that a road can have is approximately 0.12 feet per foot of horizontal distance. The equation that guides design of super-elevation of a curved road relates several variables including the super-elevation rate, the velocities of a vehicle, and the radius of curvature of the road; the equation for designing a curved road is as follows [21],

$$e_{des} = \frac{v^2}{gR} - f_s \tag{2}$$

where

e_{des} = rate of superelevation;
v = velocity;
g = gravity;
R = radius;
f_s = side friction.

A simulation software system can incorporate the equation so that invalid user input values for any of the parameters will result in the system's output screen showing a vehicle flying or falling off the road.

3.2.3 Elevation of a point

In designing a vertically curved road, which spans across some undulating terrain, consideration must be given to the elevation of various points on the road. This is important when the road crosses some obstruction, such as an overpass, at various points along the vertical curved road. To guide design of such a road, the equation for calculating the curved elevation of a point P is given as follows [21],

$$Elevation\ of\ P = \left[elevation\ of\ VPC + \left(\frac{G_1}{100}\right)x\right] + y \tag{3}$$

where

elevation of VPC = elevation of start of a curve;
G_1 = initial gradient;
x = horizontal distance;
Y = any vertical offsets.

The relationship represented by the equation can be illustrated visually in a simulation software system, which can use the equation to check if vehicles can functionally travel along the vertical curved road when an overpass crosses the road. The user of the simulation system can assign some values to the variables in the equation, and if they are not valid, the car can be shown to collide with the obstruction.

The system design involves defining the entities of: (i) the road and its properties, e.g. whether it is straight or curved, horizontal and curved or elevated and curved, (ii) the car and its properties, e.g. its weight, height, centre of gravity, forward friction, side friction, etc., (iii) weather conditions, e.g. whether it is snowy, rainy, icy etc. The properties of the entities involved and the weather conditions become variables, whose values are defined by the user, who is a student, through a system interface. The values entered by the student will be fed to the equations that define the different scenarios of motion, which can then calculate whether the scenario defined by the student is valid. If it is, the system can show on the output interface that the traveling car successfully braked or continue its path on the road according to the equations. For example, the student can assign values to the parameters of: (i) the mass of the car in the rigid body component's interface, and (ii) the side friction stiffness of a car's tire in the interface of the wheel collider component. By varying the input values, the system can simulate the different scenarios of a car driving on a road. By using Unity's built-in high-level abstraction components, users can adjust the

attributes of the car and observe the results as the driving system tries to compensate for the new variable values in order to enable the car to drive satisfactorily along the road system. Alternatively, the output interface of the simulation software system can show the car rolling or sliding off the road if the values entered by the user do not describe a valid scenario. Inclusion of parameters related to the weather conditions can be left for future work.

Another concept in transportation engineering is braking patterns, which include (a) normal braking, (b) accelerated braking, and (c) stonewall braking. Normal braking is the act of slowing down gradually when there is no immediate danger of collision while accelerated braking is the act of decelerating quickly in anticipation of an impending collision. Stonewall braking is the actual occurrence of a collision which forces the car to stop abruptly. The simulation software can illustrate visually these concepts by defining one or more than one car with its or their properties. In scenario (a), one car has to be defined, and its braking pattern can be implemented by defining the side friction parameter. Since in scenario (b), two cars are involved and the braking pattern of the first car affects that of the second one, two cars need to be defined. In scenario (c), a stationery object, such as a wall, and a car are involved.

All three of these braking patterns can be simulated using the implementation mechanisms of the car's artificial environment perception system through the use of a trigger area. The trigger area detects how far away an obstacle is and hence, the distance of a potential collision. Based on the distance, the system can determine whether normal or accelerated braking is required. In addition, the trigger area also detects the direction of the potential collision, so the car can adjust its turn to steer away from the object. If the potential colliding object is a safe distance away, the driving system will apply normal braking to gradually adjust the car's speed. However, if the colliding object is close to the car, accelerated braking will be applied to slow the car down more suddenly. The value of the side friction parameter can be adjusted to support braking while turning. If the car does come into collision with an object, the mesh collider component detects the magnitude and direction of the force of the impact. The information can be displayed to the user so that it can be used for further analysis.

4. Discussion

Application of game AI design principles and techniques can support design and development of a simulation system for teaching concepts in transportation engineering. The Unity game engine supports effective development of the simulation system with its high-level abstraction programming tools and intuitive user interface. These features support developers in design and implementation of the concepts in transportation engineering so that developers can focus on the software logic and ignore lower level development details such as graphics rendering and physics calculations.

The AI techniques useful for developing an educational simulation system for transportation engineering were implemented in a video game called *Racer* [22]. The combined tools of MonoDevelop and Unity enabled an efficient development process. MonoDevelop consists of auto-correction features for many libraries and software development kits (SDK) used in Unity [19, 23]. Using the combined tools, the waypoint system can be easily implemented because the Unity engine supports positioning waypoints in 3D space. Hence, the developer can use this Unity built-in technique of a waypoint system instead of the more traditional AI techniques such as the A* search algorithm in designing the road system. Unity's built-in component modules such as the components of rigid body, wheel collider, and mesh collider also support efficient development because these components consist of high-level abstractions of physics implementations, which would otherwise take hours to program manually. Furthermore, these components include interfaces that enable users to change properties of the objects, such as the car's mass, and the wheels' side friction. These properties are key parameters that determine friction and hence, the braking distance of cars. In terms of obstacle avoidance, the trigger detection technique is preferred over the ray-casting technique for artificial environment perception because the ray-casting technique only supports artificial environment perception in one single direction per casted ray. By comparison, the trigger detection technique supports simultaneous perception in all directions surrounding the car. The Unity engine has built-in high-level abstractions for trigger detection, which supports efficient implementation.

More wide-spread adoption of simulation software systems has been suggested as a possible solution for systemic deficiencies in the undergraduate science and engineering education. Mayo champions this cause with the words: "video games can teach science and engineering better than lectures" [2]. Similarly, Felder suggested that the "learning styles of most engineering students and teaching styles of most engineering professors are incompatible in several dimensions. Many or most engineering students are visual, sensing, inductive, and active, and some of the most creative students are global; most engineering education is

auditory, abstract (intuitive), deductive, passive, and sequential. These mismatches lead to poor student performance, professorial frustration, and a loss to society of many potentially excellent engineers" [24].

If simulation software systems can indeed enhance the learning experience of undergraduate engineering students, and as a consequence, fewer students will decide to abandon their training in science and engineering, then building more effective simulation systems for engineering education is a good approach for alleviating the problem of student attrition in undergraduate engineering education. The design principles and AI techniques presented can contribute to more efficient development of this type of effective simulation software systems in transportation engineering.

Future work can proceed in several directions. First, the design principles and AI techniques can be adopted for development of a simulation software system for teaching transportation engineering concepts. When the software system is developed, then user responses to the system can be collected and the simulation system's effect on learning outcomes can be evaluated. Secondly, other concepts in transportation engineering can be incorporated in the software design and implemented in the simulation system. For example, the concept of "shockwave effect" refers to the speed in which vehicles begin to slow down and stop at an intersection due to stopping of other preceding cars, which eventually result in a line-up of more vehicles stopping at the same intersection [21]. The shockwave effect can be described with an equation, and a simulation program can illustrate the effect visually in its output interface. Considerations of weather conditions can be incorporated in the system design. Thirdly, capabilities of the implementation platform can be expanded by including other modules from Unity's Asset Store. For example, additional modules that describe physics with higher accuracy, such as *Car Physics* and *Edy's Vehicle Physics* can be obtained from the Unity Asset Store and incorporated into the development platform of the simulation system [25] [26]. These modules would enable the developer to more realistically simulate the driving of a vehicle on a road system.

5. Conclusion

Some design principles and AI techniques supported in the combined tools of MonoDevelop and Unity have been presented. These principles and techniques enable development of useful features in a simulation software system for teaching transportation engineering concepts. By adopting the combined tools, development of a simulation software system can be more efficiently and effectively completed. The functionalities of the combined tools can also be expanded by adding other modules.

Acknowledgements—The first and second authors wish to acknowledge the generous support of the Graduate Studies Scholarship and Dr. Derril McLeod Environmental Systems Scholarship in Engineering and Applied Science from the University of Regina in Regina, Saskatchewan of Canada.

References

1. E. Seymour, *Talking about Leaving: Why Undergraduates Leave the Sciences*, Westview Press, Boulder, CO, 2000.
2. M. J. Mayo, Games for Science and Engineering Education, *Communications of the ACM—Creating a science of games,* **50**(7), 2007, pp. 31–35.
3. D. Ticoll, Labour Supply/Demand Dynamics of Canada's Information and Communications Technology (ICT) Sector, *Final Report, Nordicity*, 2012, Ottawa, ON, Canada. http://nordicity.ca/media/20121112pnzutcbz.pdf Accessed 15 July 2015.
4. Council of Canadian Academies, *Enabling Sustainability in an Interconnected World: The Expert Panel on the Potential for New and Innovative Uses of Information and Communications Technologies (ICT) for Greening Canada*, Council of Canadian Academies, 2014, Ottawa, ON, Canada.
5. L. C. Wood and T. Reiners, Gamification in Logistics and Supply Chain Education: Extending Active Learning, *Kommers, P., Issa, T., Isaias, P. (eds.) IADIS International Conference on Internet Technologies & Society* Perth, Australia, 2012, pp. 101–108.
6. C. Liao, H. Liu and D. M. Levinson, Simulating Transportation for Realistic Engineering Education and Training, *Journal of the Transportation Research Board*, (2109), 2009, pp. 12 –21.
7. C. Liao, T. Morris and M. Donath, Web-Based Traffic Simulation Framework for ITS Education and Training, *12th World Congress on Intelligent Transport Systems (ITS)*, San Francisco, CA, 6–10 November 2005.
8. C. Liao, T. Morris and M. Donath: Development of an Internet-Based Traffic Simulation Framework for Transportation Education and Training, *Transportation Research Record: Journal of the Transportation Research Board,* (1956), 2006, pp. 184–192.
9. C. Liao, D. B. Glick, S. Haag and B. Baas, Development and Deployment of Traffic Control Game: Integration with Traffic Engineering Curriculum for Teaching High School Students, *Transportation Research Record*, (2199), 2010, pp. 28–36.
10. C. Liao and D. Levinson, ROAD: Interactive Geometric Design Tool for Transportation Education and Training, *Journal of Professional Issues in Engineering Education and Practice*, **139**(2), 2013, pp. 116–122.
11. W. Chen and D. Levinson, Effectiveness of Learning Transportation Network Growth through Simulation, *Journal of Professional Issues in Engineering Education and Practice*, **132**(1), 2006, pp. 29–41.
12. S. Zhu, F. Xie and D. Levinson, Enhancing Transportation Education through Online Simulation Using an Agent-Based Demand and Assignment Model, *Journal of Professional Issues in Engineering Education Practice*, **137**(1), 2011, pp. 38–45.
13. Y. Kasaraneni, *Improving Dilemma Zone Protection Control Issue at Signalized Intersection Using a Web-Game*, http://scholar.lib.vt.edu/theses/available/etd-09182009-143008/unrestricted/Kasaraneni_Y_T_2009.pdf Accessed 15 July 2015.
14. M. Abbas and L. D. McNair, Game-Aided Pedagogy to Improve Students' Learning Outcomes and Engagement in Transportation Engineering, *121st ASEE Annual Conference*

& Exposition, Paper ID #10091. American Society for Engineering Education, Indianapolis, IN, 2014.
15. F. Fang and D. Pines, Integrating Simulation into Transportation Engineering Education, *AC 2007-261. American Society for Engineering Education*, 2007.
16. X. Shi, J. Jin, Y. Cheng, S. T. Parker, J. Zhang and B. Ran, *Conceptual Design for a Research Oriented Web-based Traffic Simulation Platform,* http://www.topslab.wisc.edu/publications/2013/Conceptual%20Design%20for%20a%20Research%20Oriented%20Web-based%20Traffic%20Simulation%20Platform%20%2813-3055%29.pdf Accessed 15 July 2015.
17. X. Shi, J. Jin, Y. Cheng, S. T. Parker, J. Zhang and B. Ran, *Prototype of a Web-based Research-and-Education-Oriented Traffic Simulation Platform*, http://www.topslab.wisc.edu/publications/2014/Prototype%20of%20a%20Web-based%20Research-and-Education-Oriented%20Traffic%20Simulation%20Platform%20(14-2766).pdf Accessed 15 July 2015.
18. S. Kim, Effects of the Gamified Class in Engineering Education Environments, *Journal of Convergence Information Technology (JCIT),* **8**(13), 2013, pp. 253–260.
19. Unity—Game Engine, http://unity3d.com Accessed 15 July 2015.
20. Microsoft Developer Network, http://msdn.microsoft.com/en-US/ Accessed 15 July 2015.
21. C. S. Papacostas and P. D. Prevedouros, *Transportation Engineering and Planning (3rd Edition),* Prentice Hall, Upper Saddle River, NJ, 2001.
22. M. T. Chan, Development of a Car Racing Simulator Game Using Artificial Intelligence Technique, Unpublished report submitted to the course "Applied Artificial Intelligence", Faculty of Engineering and Applied Science, University of Regina, Winter 2013.
23. MonoDevelop, http://monodevelop.com Accessed 15 July 2015.
24. R. M. Felder and L. K. Silverman, Learning and Teaching Styles in Engineering Education, *Engineering Education*, **78**(7), 1988, pp. 674–681.
25. Unity Asset Store—Car Physics (Multipurpose), https://www.assetstore.unity3d.com/en/#!/content/15573 Accessed 15 July 2015.
26. Unity Asset Store—Edy's Vehicle Physics, http://u3d.as/content/edy/edy-s-vehicle-physics/1Ba Accessed 15 July 2015.

Marvin T. Chan is a Masters student in Software Systems Engineering at the Faculty of Engineering and Applied Science of the University of Regina, Saskatchewan, Canada. He completed a Bachelor's degree in the Software Systems Engineering Program at University of Regina. His research interests include video game technology and creative technologies.

Jonathan T. Chan is a third year undergraduate student of the Environmental Systems Engineering Program at the Faculty of Engineering and Applied Science of the University of Regina, Saskatchewan, Canada. He studied transportation engineering and has research interests in transportation technologies and green energy projects.

Craig M. Gelowitz is an Associate Professor and Program Chair for the Software Systems Engineering Program at the University of Regina in the Faculty of Engineering and Applied Science. His research interests include machine learning, contextually aware software, software engineering methodologies and ubiquitous computing. He has held previous positions in industry and as a research manager of Telecommunications Research Laboratory (TRLabs), which was the largest industry invested R&D consortium in Information and Communications Technologies (ICT) in Canada before accepting a faculty position at the University of Regina in 2011. He is also the Section Chair for the IEEE South Saskatchewan Section.

Christine Chan is Canada Research Chair Tier 1 in Energy and Environmental Informatics and Professor of Engineering in Software Systems Engineering at Faculty of Engineering and Applied Science of University of Regina in Saskatchewan, Canada. Dr. Chan received M. Sc. degrees in Computer Science and Management Information Systems from the University of British Columbia, and Ph.D. degree in Applied Sciences from Simon Fraser University. She is Editor of Engineering Applications of Artificial Intelligence and Area Editor of International Journal of Information Technology and Social Change. She also serves as Editorial Board Member of another three international journals, and one of her papers won the Top Ten Most Cited Article 2005-2010 Award of the journal of Engineering Applications of Artificial Intelligence (Elsevier). She has published over 250 technical publications, of which over 100 are in international journals.