An inclusive framework for developing a car racing simulator game using artificial intelligence techniques and usability principles

Laiali Almazaydeh*, Mohammed Alajlounei and Waleed Ababneh

Faculty of Information Technology, Al-Hussein Bin Talal University, Ma’an, Jordan
Email: laiali.almazaydeh@ahu.edu.jo
Email: Mohammed.alajlounei07@gmail.com
Email: Waleed.3babneh@gmail.com
*Corresponding author

Abstract: Computer games have become one of the most significant digital applications due to the rapid improvement of software industry. Developers create computer games based on a combination of interleaving elements that contain technical and artistic aspects. Therefore, games development needs a comprehensive ideal path including different design considerations and usability issues than other typical software. In this research, we illustrate the general framework that helps developers to develop effective interactive games with an example of our developed contemporary computer game called Race for Freedom which was inspired by the Death Race film’s story. The most significance game development challenges addressed in this research is adapting artificial intelligence techniques in the implementation phase and validating the usability of developed game in the evaluation phase.

Keywords: artificial intelligence; AI; computer games; heuristics; HCI; racing games; shooting games; usability.


Biographical notes: Laiali Almazaydeh received her PhD in Computer Science and Engineering in 2013 from the University of Bridgeport, USA. She is an Assistant Professor at the Faculty of Information Technology, Al-Hussein Bin Talal University, Jordan. Her research interests include human computer interaction, wireless sensors networks and biomedical signals processing. She has published around 38 papers in various international journals and conferences proceedings. Recently, she has been awarded two postdoc scholarships from the European Union Commission and Jordanian-American Fulbright Commission.

Mohammed Alajlounei graduated from the Al-Hussein Bin Talal University. His current research focuses on game development and virtual reality.
1 Introduction

Computer games are interactive software that are played by manipulating a user interface to generate visual feedback on a computer screen. Nowadays, computer games have become an integral part of daily life, where 97% of people of all ages playing for at least one hour per day. Current generations of computer games offer the rich virtual world through stories, multimedia, and an entire worldview (Tamborini and Skalski, 2006). There are different positive effects for human player who actively engages with the game’s storyline. First, cognitive benefits such as improving the brain function and accurate attention allocation (Green and Bavelier, 2012). Second, motivational benefits such as working toward meaningful goals and challenging tasks in the game context (Ferguson and Olson, 2013). Third, emotional benefits such as improving the players’ moods and relieve stress (Granic et al., 2014). Last, social benefits since most games can be cooperatively played. Moreover, the medical field has used these positive effects and become interested in ‘gamifying’ medical treatment, such as in reference (Kazmi et al., 2014) the use of serious games is studied in order to improve the health outcomes of dementia patients.

In the last decade, computer games development comes from the advantage of increased advancements in modern personal computers either in computing or graphics power, to develop a complete games system with vastly different themes and goals. Therefore, the fast-moving nature of computer games industry does motivate games developers and software engineers to quickly multiplying their efforts around games logic and learning techniques. In this regard, a new comprehensive methodology that consists of tools and techniques to be used to assist the development of computer games is needed. The objective of the proposed methodology is to conform the important elements that need to be covered through game development cycle.

2 Race for freedom design model

The main focus of this research is to develop a new game development life cycle (GDLC) framework. The proposed GDLC is a development framework that encompasses the process, tools, and techniques which are used to properly construct the game and successfully pass usability criteria.

While software development life cycle (SDLC) is a series of typical engineering phases to develop a software product, game development is not a product of pure engineering. A game is an interactive software created from the combination of interleaving elements including art, sounds, graphics, music, programming, acting, and the integration of those elements (Haddad and Kanode, 2009; Ramadan and Widyani, 2013).
An inclusive framework for developing a car racing simulator game

The most significance game development challenges should be addressed in this research is adapting artificial intelligence (AI) techniques in the implementation phase and validating the developed game in the evaluation phase.

The proposed GDLC illustrated in Figure 1 is composed of five development phases as follows: inspiration, prototype design, components implementation and integration, testing and evaluation. This GDLC is an iterative approach to develop a game. In the inspiration phase, we have to identify the game story by looking for new sources which would be interesting as a game environment, in addition to identifying that source settings and mechanics. The game idea could be hunted from books, plays, photography, movies etc. In the prototype design phase, a mock-up of core gameplay pattern such as characters, fun factors, mechanics and technical aspects is made, in addition to designing audio and graphics elements. The components implementation phase, is related to the creation of both source code and artistic assets. Integrating the source code and artistic assets in one element is addressed in integration phase. The next steps are testing and evaluation before releasing the game. In the following, the whole proposed game development cycle will be explained in more details with an example of our developed contemporary computer game called Race for Freedom to illustrate each step of the proposed methodology.

Figure 1  Overall model of GDLC

2.1 Inspiration

The video game race for freedom was inspired by the film Death Race 2008. The film’s story framed for a murder that did not commit but finds himself at Terminal Island, the country’s toughest prison. However, during that time the low economy and a sharp increase of convicted criminals lead to privatised prisons for profit.

From then, the warden of Terminal Island Penitentiary offers the prisoner unexpected chance to get their freedom by competing in the game called Death Race. The racers, along with their navigators, riding in cars equipped with flamethrowers and grenades launchers on a closed track at Terminal Island. The racers should activate their offensive weapons while driving and try to get score points for running others down. At the end of the race, the racer must survive criminals to win his freedom or he will die.

The Death Race film designed to deliver more than enough action scenes as it indicates from its title. Therefore, inspired by that landmark film, our developed game depicts a car race where the human player earns points by driving and shooting down
against a three game-controlled cars in any way possible. In other words, the objective of the developed game is to simulate a pattern of Death Race movie for a car racing game.

2.2 Prototype design

The core theme is detailed using a well-defined dramatic story line. This story line is then developed further to create a playable prototype, which demonstrates how a game will work during play. During this phase, the characters, situations, events are created and designed for interactive player experience (Babu and Maruthi, 2013).

In our developed game Race for Freedom, we planned the structural elements almost as the same way that a screenwriter produced the script for a film. The main designing elements here are the cars design, weapons design, layout design, background design and sounds design.

The cars in the film are vehicles that have been heavily modified with armour plating, machine guns and defensive weapons, and as the layout the passage ways with hurdles and the escape routes are taken into consideration for design purposes.

2.3 Components implementation and integration

This process is related to source code and assets creation to achieve specified structural elements. In this regard, unity game engine is used to build the game’s environment.

Unity is considered as one of 3D game development engines, which is combined with an integrated development environment (IDE), called MonoDevelop. With MonoDevelop, developers are able to create scripts in Javascripts or C# programming language in order to control the logic and behaviours for a list of defined objects within the game environment.

The detailed implementation of Race for Freedom structural elements are as follows:

1. Player-controlled car

The body of this car is the 3D rigged car model modified with 3D rigid weapon. Car rigger is used to prepare animated parts of car, which enable the developer to control the behaviours of different components of the car such as wheels.

Many scripts are attached to the player-controlled car, which involves physics concepts, as follows:

- Box colliders for collision detection, when the car collides with other objects in the game environment.
- Wheels colliders to keep contact with the ground and move the car forward.
- The current speed, wheels rotation, top speed and the player input from the keyboard is controlled by the CarController.js.
- WheelsController.js, handles the fraction of wheels.
- Enemy.js, handles soul level of player-controlled car, the player gain souls whenever he/she defeats enemy and drop from his/her souls whenever enemies hit the player-controlled car.
An inclusive framework for developing a car racing simulator game

- Speedometer, which is an animated real-time speedometer with digital and analogue-style controls that actually rolls the numbers based on the current speed taken from CarController.js.
- Map, to pinpoint where the player-controlled car is heading, where camerascript.js controls that map according to camera placed on top of the game environment.

2 Game-controlled car

There are three AI-controlled cars. The body of these cars is the 3D rigged car model modified with 3D rigid weapon.

The following are the scripts attached to the game-controlled car, which involves AI techniques:

- AICarSensors.js, handles the object detection and avoidance, where the magnetic sensors placed on car detect the tags embedded in objects in order to avoid these objects such as a wall. However, the sensors placed on the sides of the car prevents collision between cars themselves while the sensors placed on front of the car re-steer the car if an object founded in the car way. The drawing ray-casting from these sensors in all directions is adequate in that it can detect collision.

- AICarController.js controls the driving of the enemies’ cars against the player-controlled car. One of the implemented AI techniques added to this script is the waypoints system which is used as reference points in road space for purposes of automated movement. In Race for Freedom, waypoints are stored as key positions in 3D space using dynamic array. The game software system reads this array, and the car steers itself towards the first input point, and then iterates towards the next point until the car passed all the waypoints. With this technique, the enemies’ cars can efficiently simulate the human driving as it eliminates all potential path finding problems. The waypoints on the racetrack can be seen in Figure 2.

- Braking system, to make a moving car through sharp turns slow down by using a brake to prevent the coup on the road, where AIBraking.js script controls that system according to 3D boxes placed on the sharp turns on the road as shown in Figure 3.

Figure 2 Waypoints on the racetrack (see online version for colours)
3 Weapons

To have a much more realistic shooting, machine gun with many shots are added as 3D object on all cars. The weapon shoots the other cars, when the imaginary ray-casting line from the centre of weapon collides with the cars which have tags, according to predetermined distance. Moreover, to indicate the overall health of the attacked car, a health bar, which shows the depleted health amount, is used and controlled by Health.cs script.

2.4 Testing

This process is related to testing that the game is functioning as intended. The whole instructions and software functionality coding are fully tested by considering the complete flow, and then convert it to executive formula. The testing aims to ensuring that individual components such as cars and weapon are working correctly, and the whole system performs well when they are integrated entirely in it.

The developer tested the system thoroughly to ensure that the design meets the specified requirements. More testing would make a worthwhile contribution to implementation refinement, therefore, design, implementation and testing considers as iterative process, which is deeply important to create a playable prototype.

2.5 Evaluation

With interactive software development there are no general rules to design software but there are general guidelines that help delivering higher quality software. Heuristics provide a clear understanding of these guidelines with which a design is built, also heuristics help identify the important problems that are not found among testing process (Pinelle et al., 2008). Thus, to ensure constant interaction from the human player toward our developed game Race for Freedom, we used a preliminary evaluation of the set of heuristics for usability, based on the literature that were aimed for use in game development.

Usability heuristics proposed by (Nielsen and Mack, 1994) are a list of heuristics that are aimed for desktop applications. The most common of these heuristics are, system behaviour is predictable, consistent and feedback is provided. However, few of these heuristics are not applicable in the game context, such as, error prevention and recovery, which is made through syntax in desktop application, while it is happened through physical actions of human player in the game context.
Thus far, several researchers in human computer interaction (HCI) community have developed a set of game heuristics for game design and evaluation. Despite progress in game heuristics development, the main focus of those developed heuristics in reviewed literature (Desurvire et al., 2004; Clanton, 1998; Federoff, 2002) is playability evaluation, with limited coverage of usability evaluation.

A new way in usability evaluation is proposed by Pinelle et al. (2008). The authors developed ten usability heuristics by analysing 108 PC game reviews in order to identify and categorise the usability related problems. As elaborated in reference Pinelle et al. (2008), game usability can be measured as ‘the degree to which a player is able to learn, control and understand a game’. Therefore, a set of mentioned usability heuristics consider as a systematic guidelines which could be used both in early designs and in a playable prototype to develop a usable game system.

It seems reasonable that the evaluation should be done for our developed game Race for Freedom based on heuristics from Pinelle et al. (2008) in order to get a complete picture of the usability in our game development. The heuristics are listed in Table 1 as well as the detailed corresponding race for freedom usability related to each issue.

3 Results

Among all reviewed usability heuristics from Table 1, the results show that the heuristics were a good match for our game interface. However, the game provides the player with a lifelike environment interface which needs to fully focus on racing and shooting elements. During the navigation of the game, the player will be able to get a meaningful feedback to identify his/her status in the game. In addition, the player will have a temporary storage profile to store the game interaction history. Figure 4 shows the main interface with the logo of the developed game to give a specific identity, the logo consists of three elements of the game title’s meaning: the wings represent freedom, the tires represents racing, and the skull represents death. Figure 5 shows the screenshot of the game during gameplay, which includes the indicators of the four cars, speedometer, and remaining bullets amount.

Figure 4  Race for freedom logo (see online version for colours)
Table 1  Usability heuristics measures

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Provide consistent responses to the user’s action.</td>
</tr>
<tr>
<td></td>
<td>• The player experiences consistent response to his/her input (in car movement, weapon control, game physics, appropriate sounds with the driving and shooting).</td>
</tr>
<tr>
<td></td>
<td>• The effects of the AI-controlled car are consistent with the player’s expectations.</td>
</tr>
<tr>
<td>2</td>
<td>Allow users to customise video and audio settings, difficulty and game speed.</td>
</tr>
<tr>
<td></td>
<td>• The player is allowed to adjust the sound settings before starting the game and during the gameplay.</td>
</tr>
<tr>
<td></td>
<td>• Only one level of difficulty of Race for Freedom is developed.</td>
</tr>
<tr>
<td>3</td>
<td>Provide predictable and reasonable behaviour for computer controlled units.</td>
</tr>
<tr>
<td></td>
<td>• The behaviours of path finding, map exploring, and AI-controlled car are reasonable for in-game situations.</td>
</tr>
<tr>
<td>4</td>
<td>Provide unobstructed views that are appropriate for the user’s current actions.</td>
</tr>
<tr>
<td></td>
<td>• The Player has a clear and unobstructed view of the game world from a proper camera angles. Views are designed to adjust to player’s action quickly. Views also are built with two modes: first-person camera to have view while shooting and third-person camera to have view while driving.</td>
</tr>
<tr>
<td>5</td>
<td>Allow users to skip non-playable and frequently repeated content.</td>
</tr>
<tr>
<td></td>
<td>• The game not includes any type of non-interactive content.</td>
</tr>
<tr>
<td>6</td>
<td>Provide intuitive and customisable input mappings.</td>
</tr>
<tr>
<td></td>
<td>• The player can issue commands easily and quickly. Input mapping is designed by following the input conventions set by the gaming community (keyboard buttons, mouse) to shorten the learning time.</td>
</tr>
<tr>
<td>7</td>
<td>Provide controls that are easy to manage, and that have an appropriate level of sensitivity and responsiveness.</td>
</tr>
<tr>
<td></td>
<td>• The game responds to the player’s action in a responsive manner in a way that matches the real world such as timeframe of speeding up the car and the depleted health amount.</td>
</tr>
<tr>
<td>8</td>
<td>Provide users with information on game status.</td>
</tr>
<tr>
<td></td>
<td>• The player will be able to track the current status in the game (health, location in the game world, remaining bullets amount).</td>
</tr>
<tr>
<td>9</td>
<td>Provide instructions, training, and help.</td>
</tr>
<tr>
<td></td>
<td>• The game provides adequate documentation including how to interact with the game elements.</td>
</tr>
<tr>
<td>10</td>
<td>Provide visual representations that are easy to interpret and that minimise the need for micromanagement.</td>
</tr>
<tr>
<td></td>
<td>• The visual representations are designed in a way that minimises occlusion (map view, bullets amount).</td>
</tr>
</tbody>
</table>
4 Conclusions

In this research, we illustrated the workflow in developing contemporary computer game called Race for Freedom which was inspired by the Death Race film’s story. The game design concept was built using the Unity Game Engine. There are significant key phases of game development addressed in this research including: adapting AI techniques in the implementation phase and validating the usability of developed game in the evaluation phase. Thus, the demonstrated results prove that the general framework provides a thorough guidance to help developers in game development.

References


