J. Appl. Environ. Biol. Sci., 4(9S)363-370, 2014

ISSN: 2090-4274
Journal of Applied Environmental and Biological Sciences
www.textroad.com

Adaptive Track Generation for Car Racing Game Using Evolutionary Algorithm

Naba Shaukat, Javaria Arshad, Sadaf Jabeen, Rabia Liaquat, and Kashif Zafar

Department of Computer Science, National University of Computer and Emerging Sciences, Lahore

Received: September 12, 2014
Accepted: November 23, 2014

ABSTRACT

This paper presents an adaptive track generation technique based on evolutionary algorithm. The proposed car racing game will have unpredictable and challenging tracks for the player based on player profiling. The game will have unlimited number of new tracks and the element of surprise and attraction for the player. For providing the surprise element which all other racing games lack and everlasting attraction of challenges for players, this paper describes the use of Evolutionary Algorithm for the track generation and uses player modeling techniques for allocating suitable tracks to the player based on his performance. The proposed technique performs well in car racing game and can be further extended to strategy based games. The interest of the player remains consistent with new stages with new unknown tracks and hurdles.

KEYWORDS*:* Genetic algorithms, human computer interactions, unpredictable tracks, logic and analytical module

1 INTRODUCTION

This research explores the possibility of an interactive racing game based on the Evolutionary Algorithm to retain the interest of players by presenting them with unpredictable and unknown new environments. It will provide everlasting element of surprise and joy to the players. The gaming world specially the racing game genre has already been evolved very much in past few decades from the origin of this genre. Highly interactive and realistic racing games have been developed to attract the user and provide user with the profound entertainment and fun while exploring the race, but all these highly interactive games lack the absolute element of surprise and does not provide with unlimited number of tracks and challenges. This paper describes the complete game design along with different methodologies and techniques used in the development of such games. The detailed description of the evolutionary technique for track generation is described in subsequent sections with the implementation details.

2. BACK GROUND & LITERATURE WORK

Game industry is being much flourished over the past few years. Lots of games have been developed specially of racing game's genre. But due to the competition for the game developers, many games are not able to retain the gained user interest and fail to flourish in the game industry in long term. The main reason for failure of these games is that after they capture player's interest, aren't able to provide challenging environment which is every player's need. The main purpose of any game is to let the players enjoy and have fun no matter how many times they have played the game. Racing Games genre is the traditional all time favorite game genre that every youngster looks for and loves to play. The main aim of any game is to provide its players everlasting enjoyable environment and challenges. The study shows that all the racing games that have been developed so far have predefined tracks and levels, which can make the players bored of playing so many times on same tracks and levels, if they play game very often [1].To provide everlasting pursuits and challenging environment (close to infinite tracks) to the players, so that the game will make the players want to play game ceaselessly, player should be provided unpredictable challenges to keep his\her interest focused. For giving everlasting challenges and fun, our game will create and provide evolutionary tracks every time whenever a player gets through any track.

Once a track is successfully finished, new unpredictable track with new challenges (generated by evolutionary algorithm) will be generated on the basis of player's expertise. In this way every time player

* **Corresponding Author:** Kashif Zafar, Department of Computer Science, National University of Computer and Emerging Sciences, kashif.zafar@nu.edu.pk

doesn't need to run on the same track which he/she has already mastered and got bored of. The game will captivate player's interest towards new challenges and fun every time he/she will enter the game [1], [2].

To solve the problem of providing new and exciting challenges the Evolutionary algorithms gave a full support. The use of logic-based AI in games has a history as long as artificial intelligence and computational intelligence in games is now an established field. But the use of the Artificial Intelligence techniques for providing the player satisfaction is relatively a new idea. In the Togelius et al [1], the authors presents the various approaches to solve the problem at hand i.e. to provide the satisfaction of player through evolutionary challenges and everlasting goals. The approaches include sensors, neural networks and evolutionary algorithms. In the sensors approach game attributes take the form of sensors. The control methods are then based on these special sensors. For example in our game problem the sensors would be speed of the car, average velocity, and average distance. In the second approach the control methods are based on neural networks. These networks are all standard fully connected feed forward nets (MLPs) with the transfer function. Only the weights of the networks are changed by evolution or back propagation, but the topology remains fixed. The third approach the paper represents is evolutionary algorithm. Our research project is based on this particular approach. We choose the Genetic evolutionary to provide the player satisfaction through provision of new and evolutionary tracks. Details of algorithm are discussed further.

3. **GOALS AND OBJECTIVES**

The main goal for this research is to be able to make an Evolutionary Racing Game which will never let the player lose his/her interest and will present the user with new and unpredictable challenges and fun. The goals and objectives of this game are described below:

- Providing highly interactive graphical user interface
- Calculating player expertise and on the basis of player expertise defining player profile
- Creating Unlimited and Unseen tracks by Evolutionary Algorithm
- Generating genetic evolutionary and unpredictable tracks based upon user expertise
- Defining the car controls and physics that have been used in the game
- Providing interactive platform for players to play the game e.g. Mobile based (Android) or PC games

4. **SCOPE**

The scope of this game is divided in two parts the design scope and the application scope.

4.1. **Design Scope.** The design scope of this research is based upon the literature review and brainstorming, and evaluating the best design for game engine. It is visionary scope that tells us how we imagine and visualize this game. The design scope of this game includes:

- Simple yet pioneering game interface
- User profiling featuring customized user personal information
- Innovative themes for scenes in the game
- Graphical interface for tracks and scenes
- Physics and mathematical modeling for design
- Graphical modes of racing including both the cars and the racing bikes
- Complete design with HCI (human computer interaction) principles.

4.2. **Application Scope.** The application scope of this game includes the scope of implementation and execution of the project. Specifying the scope of how we will implement different modules, what will be the level of implementation of the game?

The main application scope is described as:

- Game will be implemented in modern platform i.e. Unity 3D
- User profiling based on user expertise during game play
- Implementation of evolutionary algorithm and related artificial intelligence techniques
- Development of fitness levels for mapping evolutionary generated tracks with user profile
- Single player mode implementation
- Up to date user profiling as expertise improvement

5.**GAME ARCHITECTURE**

5.1.1.**Game System.** In this Game which is "Racing Game based upon Evolutionary Algorithm" our main aim is to develop a Game Engine that will be generating evolutionary tracks, each time a player will complete a race track and demand for a new challenge the game engine should be able to evolve the current track and generate evolved tracks and present the user with most suitable in regards to player's performance. The evolvement of tracks that our game engine will in cooperate will be using Genetic Algorithm (GA).

Therefore, the major part of this Game's Engine will be Track Evolution Using Genetic Algorithm. So, the system's focus for this major part will be on:

- Defining Genetic Algorithm's Components
- Defining the Track Chromosome that will evolve the tracks just like Biological Gene Evolution and generated chromosome
- Defining the best suitable Fitness Function and Algorithms to testify the track, i.e. whether the generated track is appropriate for race or not, discarding the inappropriate and impossible to race tracks
- The choice of track that will be presented to player from the evolved track set, based upon the Player's performance either player wants to play the track of same level or is he/she is striving for more challenging hard tracks by showing his/her best level expertise
- The Game engine will also be maintaining the Player's profile and will be calculating it and updating it based upon algorithms defined for profiling in the system each time player will race
- Complete integration of all the above modules and optimization is also a very important part of the Game engine

5.1.2 **Software System.** The Software used for the development of the overall game development is Unity 3D. Unity 3D is a game engine for developing highly integrated and interactive 3D games. System's Modules and then integration of all these modules with game representation is done with unity 3D game engine.

Game Engine is a Game Development Ecosystem; it provides powerful rendering engines and API's for good and interactive Game interface. It provides great performance and also work-flow. It gives fully integrated set of tools and workflow for 3D or 2D game development, easy multiplatform publishing.

5.2. **Game System Architecture.** The architecture of this Game follows a path and modular structure as shown in Figure 1. It defines the internal structure of how the game's program work and revolutionize the track generation process. The main functionalities of the game or program's architecture is creating and managing user profiles and generating the evolutionary tracks to achieve the goal of our Game project. The game follows the 3-tier architecture of Software development.

As mentioned in the above shown player work model, the new track will be accessible when the player has already completed the current track. The detailed description of these architectural components of our project is given as:

The Game follows the standard, 3 – tier Architectural design [3].

- The interface layer provides the player with Game play and highly interactive graphical game view. It also contains the event listeners that helps the Player interact with the Game and enables the player controls during race
- Logic layer of Game System includes all the major logical functionalities and modules that are part of the game system
  - EA (evolutionary algorithm) Module that is very fundamental part of this game project and is used in the Track evolution and new track generation, this module is also responsible for evaluating the player profile and handling all other game objects that are needed to be handled intelligently during race
  - This layer also includes Mathematical and Analytical Module that includes mathematical functionalities and all other physics fundamental that are mapped in race for player car control and game track are integrated with other modules for setting the complete race environment
  - All other modules and Game objects including Sounds or Audio Controller and all other utilities used by system are handled in this module
- The data layer records the Player Profiling and tracks data of track components and from track generation

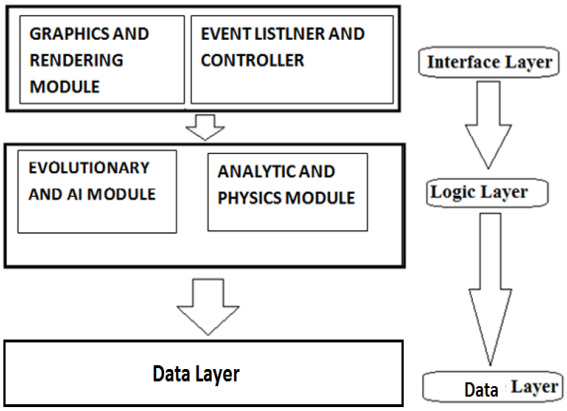


Figure1. 3-tier Game System Architecture Figure

6. GAME DESIGN

The Game Design of this game is very simple and consists of few modules which are based on overall game's structure and architecture. The basic Game design consists of three major parts which have been mentioned in the Game's Architecture shown above i.e.

- Graphical Interface
- Logic and Analytical Module
- Data Module.

Graphical module covers all the graphical work that is part of the game's system from Display Scenes for different options till the rendering of the Race Track and its environment.

The detailed Game System design can be these game modules, which are given as:

A. Track Generation Module Using Evolutionary Algorithms
B. Player Work – flow Model
C. Analytical Logic Module
D. Graphics Module

These modules are described below:

6.1. **Track Generation Module Using Evolutionary Algorithm.** The Track Generation module is very important module of our project, as the main goal of this project is to provide the player the unseen and unpredictable, unlimited number of tracks using Evolutionary Algorithm [2].

Evolutionary Algorithms (EA) are a subset of evolutionary computation, a generic population – based meta-heuristic optimization algorithms. EA uses techniques which are inspired by natural biological evolution techniques i.e. reproduction, mutation, recombination and selection. Genetic algorithms (GA) are very famous technique of EA, which provides very simple yet optimized solution for particular problem. Therefore, we have chosen the Genetic Algorithms for Track generation in this game as Genetic Algorithms provides the solution from the basic and atomic units which evolve by the GA process and provide the optimized solution. As our game requires the evolution of Tracks so that no two tracks should be same as no two DNA are same in the natural process of genetic evolution. A typical genetic algorithm requires [2]:

1. Genetic Representation (genes and chromosome) of the solution domain
2. Fitness Function to evaluate the solution domain.

The basic structure of track generation which follows the structure of genetic algorithm is given as:

- Defining the Track components as Genetic representation of Track (Genes and Chromosomes)
- Generating Track Sets and Evolving Tracks through genetic operators (Population, Mutation and Crossovers)
- Applying Fitness and selecting the best fittest track from the candidate tracks (Fitness Evaluation)

6.1.1.**Defining Genetic Representation of Track Components**. A *Gene is a single component* which controls the property of an individual. In our game a gene represents a single track component i.e. Right Turn, left turn, straight line or a sharp edge etc. Given below is a single component or a gene as:

Right Curve

In our game we supported six components which are

i.Horizontal Straight line
ii.Vertical Straight line
iii.Left Curve
iv.Right Curve
v.Left Sharp Edge
vi.Right Sharp Edge

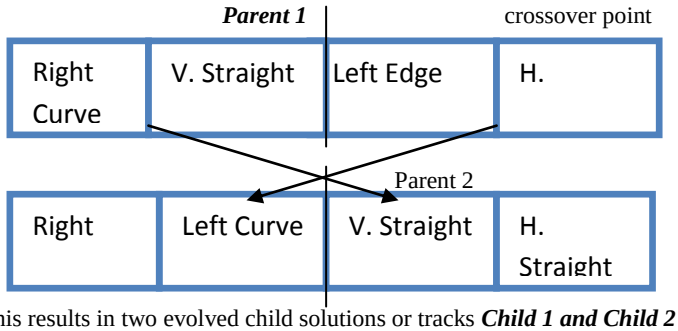
So, a track consists of combination of these track components representing a chromosome. A ***Chromosome*** is a set of genes which represents a possible solution of any problem; therefore, in our game a single chromosome is a combination of track components/genes. Theoretic representation of a chromosome for this game i.e. track can be described as:

Right Curve	H. Straight	V. Straight	Left Curve	Right Edge

So, this representation of track has been mapped in the code scripts of GA in terms of number through 1-6 each number representing a single component and at the time on run time rendering in the game scene the track prefabs and components that has been specially created using Meshes are rendering accordingly.

6.1.2.**Generating Track Population.** In a genetic algorithm, a ***population*** is a set of candidate solutions (tracks in our game) called individuals that can provide the optimum solution to an optimization problem, which are evolved toward better solutions. Each candidate solution has a set of properties which can be mutated and altered. For our game we have decided to create a population i.e. set of tracks (chromosomes) of size 50 which means a population will be having 50 tracks.

6.1.3.**Evolving Tracks.** The generated population is evolved for generating the optimized best solution for the given problem. The tracks are evolved by genetic operators which are mutation and cross over.
In genetic algorithms, *crossover* is a genetic operator used to vary the programming of a chromosome which is analogous to the natural reproduction and biological crossover. For evolving the tracks using crossover operator we have defined a crossover point in the chromosome (track) which is chosen to be the middle component, and then by that point crossover is done. By theoretic diagrams explanations it can be understood as follows:



This results in two evolved child solutions or tracks ***Child 1 and Child 2 respectively*** given as;

Child 1

Right Curve	V. Straight	V. Straight	H. Straight

Child 2

Left Edge	H. Straight	Right	Left Curve

After the crossover the tracks are being mutated so that the result could be more modified and evolved. ***Mutation*** takes a single chromosome and randomly changes one of a single gene value. In terms of our game randomly changing a track component is called mutation.

6.1.4. **Fitness Function and Selection of fittest Track.** Fitness Function defines the criteria for the fittest and possible candidate solution for the given problem. For track generation when the track has been evolved it is possible that those tracks might not be even playable, such track cannot become the candidate solutions for next track that would be presented to the player for race. Therefore, to measure this validity of tracks should be checked. In this game project it has been checked carefully that track components which can make a non-playable track.
Track fitness evaluates the best fittest track from the candidate population set that should be selected for representing to the player.

Fitness Function

The fitness function has been defined in accordance with the player rank (which is updated after every race completion). Our fitness function replicates the following criteria for selecting the best track for respective player rank.

If the current rank is *beginner* then fittest of the tracks is the one which does not provide too much difficulty level for the player and represents a track which contains less than 30% of curves or edges.

If the current rank is *intermediate* then fittest of the tracks is the one which does not provide too much difficulty level or too much easiness for the player and represents a track which contains 50% of curves or edges and 50% of the straight lines.

If the current rank is *expert* then fittest of the tracks is the one which does not provide too much easy level for the player and represents a track which contains less than 30% of straight lines.

Therefore, after the fitness evaluation, the track with best fitness result is being selected and is represented to the player for race.

The step by step approach of all this process is given in Figure 2.

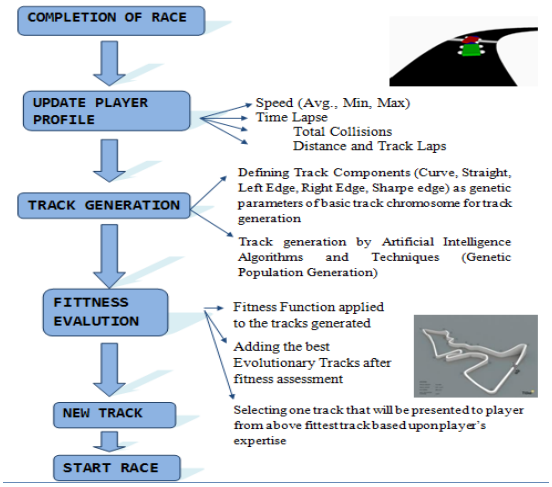


Figure2. Step by Step Process of Track generation

6.2. **Player's Work Flow Model.** The Player Work Flow model describes the player's navigation that he/she can do in the game as well as the options or tasks that can be done by the player. The player's work flow can be visualized by the given model below in Figure3:

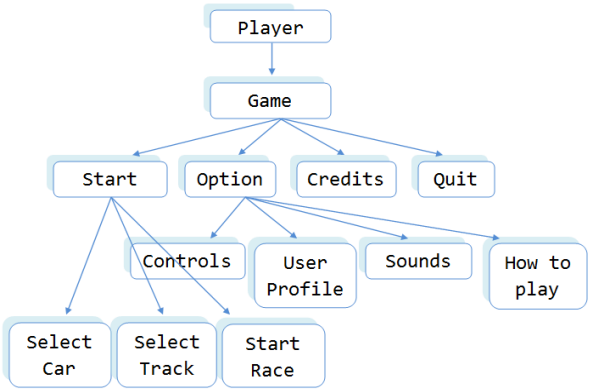


Figure3. Player's Work Flow Model

6.3.**Analytical Module.** The Analytical Logic Module in the game will perform the following functionalities:

- Mapping of Mathematical and Physics Formulae in evaluating player profile while the player will be racing; collision detection, speed and moves etc.
- Physics Model to optimize the tracks, and defining the fitness function for track evaluation
- Car Modeling Using Mathematical Models and Physics techniques
- Graphical Interface optimization of scenes, views and axis orientations by geometrical Models
- A simplified physics model allows the car to move in a realistic way. At each level, a bunch of forces are combined to result in acceleration
- The car physics react to some external inputs which will allow for the computation of the forces applied to the vehicle. These inputs are translated from an input device if the car is controlled by a human

6.4. **Graphic Module.** Graphical Module is a module which will map all the Game objects; Cars, Tracks and Race etc. in a graphical user interface screen so that user can interact with them and have a race. Major parts of Graphic module will include:

- Game Objects and Game Assets
- Game Scenes
- Rendering Models

These are the fundamental and most important parts of graphical module, apart from these there are some other modules and sub – modules of the Graphical System prior to development.

7. IMPLEMENTATION

.The Implementation of the Game which is based upon the above mentioned Game design and policies. For the development of this game we have chosen Unity 3D as the software for development, and implementing all the modules described included the car control and graphics which will provide a graphical interface to the tracks that has been generated.

The Major Focus in the implementation is upon the implementation of the Evolutionary Genetic Module that will be responsible for the new challenging Track Generation. Second major task after the implementation of Evolutionary Module is the integration of that module in the basic Game architecture that has been specified by the shown with the help of prototype.

7.1. **Unity 3D.** Unity 3D is a Cross platform game engine with its well-defined IDE. It is used for creating games for platforms like PC games or Web or Mobile games.

The reason for us to choose this platform is to be able to make a good 3D game and also due to its cross platform feature we will be able to provide our game for different platform in near future without making the game all over again [5].

7.2. **Rendering.** One of the main features that this game engine provides is the graphics engine and its ability to render the 3d scenes. We are using its powerful graphics engine to generate the tracks which will be decided on run time based on the genetic algorithm. We are using meshes to render and create the 3d track on run times which is our most important graphics module i.e. to be able to create and render the track

on screen which is to be created on run time. Apart from the track, creating the scenes and other graphical works are to be done also with the help of unity's graphics engine.

7.3. **Scripting.** Unity's scripting is done with the built on Mono-Develop, which is an open source implementation of the .NET framework. We have used this feature to write the code for different modules in our game. Mainly we have used C# language and also used JavaScript as well. The major parts that include to be implemented via scripts are:

- Complete Genetic Algorithm's implementation
- Physics and Car controls
- Scene management
- Different controlling scripts for objects
- Shaders and rendering for graphics

Apart from these many other controlling factors including the filling which is to be for saving the Game's data is also done via scripts.

7.4. **Assets.** Unity 3d provides the support of using the built in assets or creating own materials and game objects. Assets are a very important part of this game engine, even all the scripting is also being saved in the assets folder of the project. All the game's objects and related controls are in assets and one can also re-use once created game object or component easily [6].

7.5**. Physics Engine Support.** Unity provides us a powerful Physics engine that is mainly used for the game's physics. Player car control also uses this engine for running the car, and all the controlling of car movement as well as the colliders which are used for collision detection and other physics related work are used with the help of this engine.

7.6. Cross Platform. As mentioned Unity 3d provides us with its powerful feature of multi-platform ability, therefore the current game is a pc standalone project so that we could test and manage the resources and computational performance of the game due to the use of genetic algorithm. And for the future work, with the help of unity we can provide the game for other platforms without creating the game all over again.

8. REFERENCES

1. Togelius, J., Nardi, R. D., and Lucas, S. M. *Making Racing Fun Through Player Modeling and Track Evolution.* Department of Computer Science University of Essex Colchester CO43SQ, United Kingdom.
2. Kendal G., Lucas, S. M., eds. *Proceedings of the First IEEE Symposium on Computational Intelligence and Games*. IEEE Press (2005).
3. Russel, S., Norvig, P. (2009). *Artificial Intelligence A Modern Approach*. Pearson Publishing.
4. Mohammad Atwi. (2011, January 22).Concepts of Three-Tier Architecture. Retrieved from http://alitarhini.wordpress.com/.
5. Cardamone, L., Loiacono, D.L., Lanzi, P. L. (2013). *Interactive Evolution for the Procedural Generation of Tracks in a High-End Racing Game*. Milano Italy.
6. Unity (Game Engine), (n.d). Retrieved from http://en.wikipedia.org/wiki/Unity_(game_engine).
7. Christopher Pope. (2013, June 5). Unity Products and Services. Retrieved from http://blogs.unity3d.com/.
8. M. Shinwari, S. A. Khan, Towards the Railway Traffic Management Using Mobile Agents, VAWKUM Transactions on Computer Sciences, Vol. 1, No. 1, 2013