Entertainment Computing 5 (2014) 251–258

Contents lists available at ScienceDirect

Entertainment Computing

journal homepage: ees.elsevier.com/entcom

Geometrical vs topological measures for the evolution of aesthetic maps in a RTS game ☆

R. Lara-Cabrera *, C. Cotta, A.J. Fernández-Leiva

Department of "Lenguajes y Ciencias de la Computación" ETSI Informática, University of Málaga, Campus de Teatinos, 29071 Málaga, Spain

ARTICLE INFO

Article history:
Received 30 January 2014
Revised 3 July 2014
Accepted 1 August 2014
Available online 13 August 2014

Keywords:
Procedural content generation
Game aesthetics
Computational intelligence
Real-time strategy games

ABSTRACT

This paper presents a procedural content generation (PCG) method that is able to generate aesthetic maps for a real-time strategy game. The maps were characterized based on either their geometrical properties or their topological measures (obtained in this latter case from the sphere-of-influence graph induced by each map). Using these features, a distance function between maps can be defined. This function is used in turn to determine how close/far each map generated by the PCG method (a self-adaptive evolutionary algorithm) is to a collection of maps which were taken initially to be aesthetic or non-aesthetic. This correspondence guided a multi-objective evolutionary approach whereby maps close to aesthetic maps and far to non-aesthetic maps are sought. Self-organizing maps are used to ascertain whether the so-generated maps naturally cluster together with aesthetic maps, as well as to provide a qualitative assessment of the ability of each set of features to characterize the latter.

1. Introduction

The videogame industry has taken the lead role from the entertainment business, with a total consumer spent of $24.75 billion in 2011 [1] and estimated game revenues of $70.4 billion worldwide in 2013 (which represents a 6% year-on-year increase), according to Newzoo's 2013 Global Games Market Report [2]. Moreover, the number of gamers was expected to surpass 1.2 billion by the end of that year. This situation has motivated the research applied to videogames, which has been acquiring notoriety during the last years, involving many areas such as psychology and player satisfaction, marketing and gamification, computational intelligence, computer graphics, and even education and health (serious games). The quality and appealing of video-games used to rely on their graphical quality until the last decade, but now, their attractiveness falls on additional features such as the music, the player immersion into the game and interesting storylines. It is hard to evaluate how amusing a game is because this evaluation depends on each player, nevertheless there is a relationship between player satisfaction and fun [3]. Nowadays, interesting new challenges and goals are emerging within the area of video games, especially in the field of artificial and computational intelligence in games [4].

Procedural Content Generation (PCG) [5,6] refers to the algorithmic creation of game content, either with human intervention or without it, such as maps, levels, textures, characters, rules and quests, but excluding the behavior of non-playable characters and the game engine itself. The use of PCG has several advantages, including saving memory and disk space, improving human creativity and providing adaptivity to games. These benefits are well known by the industry as demonstrated by the use of PCG techniques during the development of commercial games such as *Borderlands saga* with procedurally generated weapons and items, *Skyrim* (terrains and forests), and *Minecraft* or *Terraria* with procedurally generated worlds. At the moment, there are three main goals [5] of PCG research that are currently not obtainable and it would require significant further research effort: multi-level multi-content PCG (i.e. systems that are able to generate multiple types of quality content at multiple levels of granularity in a coherent fashion while taking game design constraints into consideration), PCG-based game design (i.e. creating games where a PCG algorithm is an essential part of the game instead of being a design tool) and PCG systems that could create complete games including the rules and game engine. There is a subfield in PCG (i.e. Search-based PCG [7,8]) whose techniques apply a generate and test scheme so the content is firstly generated based on previous evaluations and then evaluated according to some criteria (this paper presents a method that follows this scheme).

☆ This paper has been recommended for acceptance by Pedro Antonio Gonzalez-Calero.

* Corresponding author.
E-mail addresses: raul@lcc.uma.es (R. Lara-Cabrera), ccottap@lcc.uma.es (C. Cotta), afdez@lcc.uma.es (A.J. Fernández-Leiva).

http://dx.doi.org/10.1016/j.entcom.2014.08.003
1875-9521/

The application of the aforementioned PCG techniques for *Planet Wars* involves, in this case, generating the maps on which the game takes place. The particular structure of these maps can lead to games exhibiting specific features. In previous work [9,10] we focused on making the game more fun to play, achieving games that are balanced (i.e., games in which none of the players strongly dominates her opponent) and dynamic (i.e., games with a high number of battles and changes in the owners of the planets). Despite we were able to accomplish our requirements of balance and dynamism, the generated maps lacked aesthetics (for example, maps with their planets clustered in a small region), which is an interesting property apart from the fun that may lead to increase the player satisfaction; in fact, fun and aesthetics are two complementary means of achieving the same goal [3]. In addition, non-aesthetic maps may confuse the player, reducing her satisfaction or even leading her to stop playing the game. This situation led us to tackle this problem [11]; therein we focused on a way to characterize maps so as to attain a method capable of producing scenarios with good aesthetics. In this paper, we have expanded the aforementioned characterization method including new topological measures that are not affected by rotation, scaling and translation, which is important to prevent two maps being considered different when they are conceptually the same (they are equal according to the gameplay). In order to obtain these measures, we characterized every map as a sphere of influence graph [12], which establishes a relationship between some set of points based on their spatial arrangement. This characterization provides a higher-level of abstraction and paves the way to measure topological properties of maps as well as providing a different perspective to analyze morphological properties of the latter. In the following, we will describe these different characterizations, and study the results obtained when an evolutionary PCG method is deployed on them.

2. Background

Real-time strategy (RTS) games offer a large variety of fundamental AI research problems [13] –such as adversarial real-time planning and decision making under uncertainty among others– have been widely used as a test-bed for AI techniques [14–18]. PCG techniques are usually employed to generate maps, as exhibited by the large number of papers on this topic [6]. For example, Mahlmann et al. [19] presented a search-based map generator for a simplified version of the RTS game *Dune 2*, which is based on the transformation of low resolution matrices into higher resolution maps by means of cellular automata. Frade et al. introduced the use of genetic programming to evolve maps for videogames (namely terrain programming), using either subjective human-based feedback [20,21] or automated quality measures such as accessibility [22] or edge-length [23]. There is another paper [24], by Liapis et al., that relies on the human evaluation of the generated content. Togelius et al. [25] designed a PCG system capable of generating tracks for a simple racing game from a parameter vector using a deterministic genotype-to-phenotype mapping. Dormans [26] presented strategies to generate levels for action adventure games using an approach that distinguishes between missions and spaces as two separate structures that need to be generated in two individual steps. There is a study by Ashlock and McGuinness [27] that introduces the so-called landscape automata for searching the space of height maps using an evolutionary algorithm. The designer is able to specify checkpoints that must be mutually accessible in order to control the generated height maps. Another level generator based on an evolutionary algorithm is presented in [28], but in this case, the fitness function depends on the difference between the difficulty curves defined by the designer and calculated for the candidate content, respectively. Some authors [10,29] presented algorithms capable of generating balanced maps for the RTS games *Starcraft* and *Planet Wars*.

As we stated before, there is a large number of papers devoted to the generation of maps and levels. They all have one thing in common, their algorithms look for carrying out various restrictions related to the gameplay, such as ensuring the accessibility of certain zones, adjusting the difficulty or balancing the gaming level of the players. This increases the fun and thus the player satisfaction. This paper deals with another way of improving this player satisfaction, that is generating maps with good aesthetics, a topic we were not able to find in the state of the art, besides our previous work.

In addition to maps and levels, there are other content that is generated with these methods. For example, Font et al. [30] presented an initial research regarding a system capable of generating novel card games. Collins [31] made an introduction to procedural music in video games, examining different approaches to procedural composition and the control logics that have been used in the past. The authors of [32] have created a prototype of a tool that automatically produce design pattern specifications for missions and quest for the game Neverwinter Nights.

In this work we focus on *Planet Wars*, a RTS game based on *Galcon* and used in the *Google AI Challenge 2010* [33], a competition about creating a computer program that plays the game of Planet Wars as intelligently as possible. Despite players make their orders on a turn-by-turn basis, they issue these orders at the same time, so we can treat this game as a real-time game. The game's objective is to conquer all the planets on the map or destroy all of your opponents ships. A game of Planet Wars takes place in outer space, precisely on a map that contains several planets with some number of ships on it. Each planet may have a different number of ships. The planets may belong to some player or may be neutral. The game has a time limit and it may end earlier if one of the players loses all his ships, implying that the player who has ships remaining becomes the winner. It is considered a draw if both players have the same number of ships when the game ends. On each turn, the player is able to send fleets of ships from any planet she owns to any other planet on the map. She may send as many fleets as she wishes on a single turn as long as she has enough ships to supply them. After sending fleets, each planet owned by a player (i.e. not neutral) will increase the number of defending ships there, according to that planet's growth rate (i.e. size). Different planets have different growth rates. The fleets will then take some number of turns to reach their destination planets, where they will then fight those opposing forces there and, if they win, take ownership of the planet. There is an important restriction: fleets cannot be redirected during travel. Players may continue to send more fleets on later turns even while older fleets are in transit. Next section will detail the structure of maps in this game, and how their properties can be characterized.

3. Methodological issues

As stated before, let us firstly detail how the maps have been represented and characterized in order to get better aesthetics, both from the geometrical and topological point of view; next, we describe the evolutionary algorithm used to generate the maps.

3.1. Representation and characterization

Game maps are sets with a certain number of planets n_p located on a 2D plane. These planets are defined by their position on the plane (coordinates (x_i, y_i)), their size s_i and a number of ships w_i. The size s_i defines the rate at which a planet will produce new