Cicero
A mixed-initiative AI-assisted game design tool

Tiago Machado
New York University
tiago.machado@nyu.edu

Andy Nealen
New York University
nealen@nyu.edu

Julian Togelius
New York University
julian.togelius@nyu.edu

ABSTRACT

This paper describes the design of Cicero, a mixed-initiative tool which allows human users and intelligent agents to work together in game prototyping tasks. Cicero is built-in on top of the General Video Game AI Framework (GVGAI) and the Video Game Description Language (VGDL). Among its features, Cicero has a stats tool, a visualization system, a mechanics recommender, a retrospective analysis application and a query system for in-game data. Cicero is constantly evaluated and its features are the results of users' tests and feedback.

KEYWORDS

Gameplay Sessions, Game Analysis and Visualizations, Game Prototyping, Retrospective Analysis

ACM Reference format:
Tiago Machado, Andy Nealen, and Julian Togelius. 2017. Cicero
A mixed-initiative AI-assisted game design tool. In *Proceedings of FDG'17, Hyannis, MA, USA, August 14-17, 2017,* 3 pages.
DOI: 10.1145/3102071.3102090

1 INTRODUCTION

Game design tools is a common place nowadays. There is a myriad of them available, from open-source to commercial ones. They bring a set of features that virtually can attend the desires of independent developers, game companies and players around the world. In general, these tools offer assistance in tasks like editing game graphics, physics, UI, sound and etc. However, when dealing with Artificial Intelligence, there are not so many examples and even a study tried to identify what would be the requirements for AI Game Design Tools [10]. In the recent years, we have been seeing many efforts in areas like game telemetry and game analytics. Companies like Unity [1] and Bioware [20] are investing in tools that empower developers with data collection and analysis about their players. What are they doing? How long are they playing a certain title? How to keep them playing? These are just a few questions that tools based on telemetry and analytics can answer about player habits. Certainly, there are many other computational methods that make these tools able to assist developers when they need to understand their users (which is all the time). However, when talking about game development, properly, the level of AI-assistance is not so expressive. Fortunately, some works have been presenting new game engine tools based on AI. They offer help on tasks like level creation [14, 17], music and sound design and even can generate games entirely from scratch. The problem is that many of these tools are strictly attached to only one single game. If someone wants to apply their ideas on a new project, it will be necessary to rewritten everything again. Cicero[1] is a tool that allows the use of AI for game development tasks in different game genres inside the spectrum of 2D tile-based games. Its main influences and its features highlights will be showed in the following sections.

2 BACKGROUND

In this section, we present some works that are an influence for the development of Cicero. We divided them into two categories: AI Game design assisted tools and Game telemetry and visualization analysis

2.1 AI Game design assisted tools and Game telemetry and visualization analysis

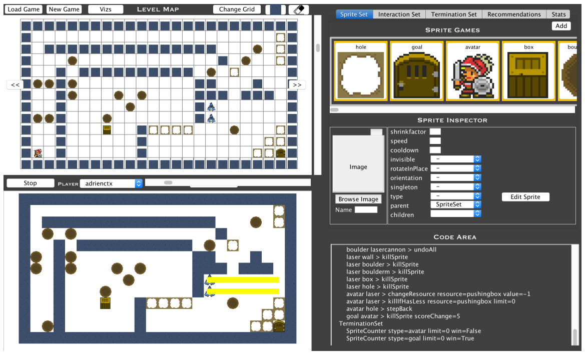
Tanagra [17] is a tool that creates and analysis levels for 2D-platform games. Ropossom [14] does a similar service offering level and analysis assistance. It is focused on the mobile game *Cut The Rope.* A bit more generic is Sentient Sketchboook [6] whose main functions assists users by recommending levels in real-time for strategy and rogue-like games. As stated before, despite the impressive results, these systems are very straight related to one single game or a game genre in the best scenario. There are works with presents a more generic use, but they are related to specific contents like music [13] or are used as examples for Domain Specific Languages (DSL) [5]. When trying to achieve a high level of generality, with core components like rules and mechanics, the examples are not so many.

2.2 Game telemetry and visualization analysis

Despite in its infancy [2], designers and developers already are benefiting their works with the current game telemetry and analytics tools [4, 19]. For example, they are aware of all the events happening in a game (deaths, enemies attacked, items used, etc.) and they can even predict when players are becoming uninterested in a particular title [8]. Exactly like the examples in the subsection above, many of the telemetry and visualization tools presented are also straight-attached to one single project. For example, *Data Cracker* is focused on a game of the popular franchise: *Dead Space* [9]. It collects and provides analysis about users' performances in the game. In [19], the authors uses the 2D runner game, *Cure Runners,*

FDG'17, Hyannis, MA, USA
© 2017 ACM. 978-1-4503-5319-9/17/08…$15.00
DOI: 10.1145/3102071.3102090

[1] link for the video: https://tinyurl.com/lh5wfar



shrinkfactor
speed
cooldown
invisible –
rotateInPlace –
orientation –
singleton –
type –
parent SpriteSet
children

```
    boulder lasercannon > undoAll
    laser wall > killSprite
    laser boulder > killSprite
    laser boulderm > killSprite
    laser box > killSprite
    laser hole > killSprite
    avatar laser > changeResource resource=pushingbox value=-1
    avatar laser > killIfHasLess resource=pushingbox limit=0
    avatar hole > stepBack
    goal avatar > killSprite scoreChange=5
  TerminationSet
    SpriteCounter stype=avatar limit=0 win=False
    SpriteCounter stype=goal limit=0 win=True
```

Figure 1: Cicero's main UI. On the top left, the level editing panel. On the bottom left, a game running. On the top right, the panel to choose and edit game contents. On the middle right, panel to adjust contents' parameters. On the bottom right, the VGDL code of the game being edited.

as an investigation about how to integrate game analytics tools into a game development cycle. Finally, the game *Super Mario Bros 3* is the test bed for a telemetry system that gathers every player input to understand better gameplay sessions [16]. Commercial tools like Unity Analytics [1] offer a package of analytical services for a broad set of game genres. Both Unity Analytics and Ubisoft's DNA are compared against G-Player [3]. It is a visualization system whose interface allows users to navigate through data collected from a mod of the game *Fallout:New Vegas*. Its spatial and temporal analysis were well received by domain experts that in many aspects consider it superior to the results of the commercial packages available.

3 CICERO SYSTEM

Cicero is an acronym for Computationally Intelligent Collaborative EnviROnment for game and level design. It is built on top of the GVGAI Framework and the VGDL language. It contains features for creating and editing games as well as features for game telemetry and analytics. See figure 1.

3.1 VGDL and GVGAI

The Video Game Description Language allows developers to prototype games with a few lines of code [12]. The games that can be created are similar to classical ones developed for the Atari 2600 system. The GVGAI is a framework that uses games as a test bed for general artificial intelligence [11]. The framework allows the development of intelligent agents who learns how to play the games without previous acknowledgment about them. Therefore, when a user is interacting with Cicero UI, underneath the system is creating VGDL lines of code. When the user wants to simulate a game, she can pickup one of the available agents and the system will use it to run a simulation.

3.2 Cicero Features

Besides the tasks of creating, editing and simulate VGDL games, Cicero offers the following features: a stats tool, a mechanics recommender, a visualization control, a retrospective analysis tool and a query system. Each one of them will be detailed in the following subsections.

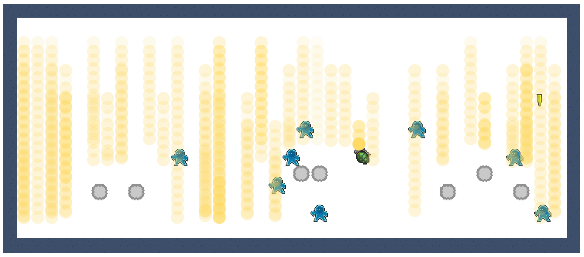


Figure 2: The visualization control showing the trajectories of all the bombs dropped by the enemies in a VGDL Space Invaders clone.

3.2.1 Stats Tool. The stats tool keeps a record of every interaction which takes place in a game. An interaction in VGDL is part of the core mechanics of each game and specify what happens when two sprites overlap. A real-time report shows a diagnosis with all the interactions in use. It creates a rank with the most to the least accessed (which includes interactions not accessed at all). This way, based on the diagnostic, the user can design test cases to identify if the not accessed rules are due to a design flaw.

3.2.2 Mechanics Recommender. The inspiration of the Mechanics Recommender comes from the situationalist school of creativity, defined by Ben Scheneiderman [15]. In this school, the idea of creativity is based on unpredictable events that trigger creative thinking. This feature works whenever a user presses a button to get recommendations from the system. The system compares the game in development with the ones available in the VGDL library. Then it defines a rank of similarities and suggests sprites and interactions (the core elements which represents game mechanics in VGDL) [7]. It is worth to say that sprites in VGDL are not just related to images. Actually, they embed behaviors like random moves, chasing and shooting.

3.2.3 Visualization control. The visualization control shows information about every game object and every game event. The control adapt itself to any game and allow the users to select what they want to see. See figure 2

3.2.4 SeekWhence. SeekWhence is a retrospective analysis tool [18]. It allows users to navigate in stored gameplay sessions (See figure 3). The users can play these sessions frame-by-frame and inspect carefully the many events that take place during a particular game tick. SeekWhence is integrated with the visualization controls to enhance the user analytical skills when inspecting her game. It also allows that any frame can be exported to the Cicero main UI, which allows user's editing and change the gameplay story from that point to the end.

3.2.5 Query System. Even with a retrospective analysis tool, it is difficult to know specific events about a gameplay session. In a single frame, many events happen with many game elements. In order to help users to find the information they are looking for we developed a query system. It works by storing every interaction

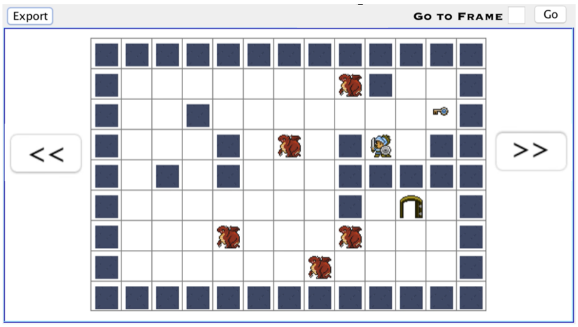


Figure 3: SeekWhence main UI

in a database. It stores what event happened, who were involved, where it happened and when. The query system is integrated with SeekWhence and its results are showing to the users as a list of panels. By clicking on any one of them, it jumps to the exact frame when the event happened and highlights the elements and place.

4 NEXT STEPS

Cicero is periodically tested. Users perform tasks in order to explore the tool and provide us with their feedback. For our next steps, we will build an agent evaluation tool. It will print a tree showing all the possible choices and the decisions taken by an agent at every state of a gameplay session. We believe that this feature integrated with what we already have will improve the users when developing and comparing new and already implemented agents.

ACKNOWLEDGMENTS

Tiago Machado is supported by the Conselho Nacional de Desenvolvimento Cientfico e Tecnolgico (CNPQ), under the Science without Borders scholarship 202859/2015-0

REFERENCES

[1] 2017. Unity Technologies. Unity Game Engine. https://unity3d.com. (2017). Accessed: 2017-03-01.

[2] Brian Bowman, Niklas Elmqvist, and TJ Jankun-Kelly. 2012. Toward visualization for games: Theory, design space, and patterns. *IEEE transactions on visualization and computer graphics* 18, 11 (2012), 1956–1968.

[3] Alessandro Canossa, Truong-Huy D Nguyen, and Magy Seif El-Nasr. 2016. G-Player: Exploratory Visual Analytics for Accessible Knowledge Discovery. (2016).

[4] Magy Seif El-Nasr, Anders Drachen, and Alessandro Canossa. 2013. *Game Analytics: Maximizing the Value of Player Data.* Springer Publishing Company, Incorporated.

[5] Andre W.B. Furtado, Andre L.M. Santos, and Geber L. Ramalho. 2011. SharpLudus Revisited: From Ad Hoc and Monolithic Digital Game DSLs to Effectively Customized DSM Approaches. In *Proceedings of the Compilation of the Co-located Workshops on DSM'11, TMC'11, AGERE! 2011, AOOPES'11, NEAT'11, & VMIL'11 (SPLASH '11 Workshops).* ACM, New York, NY, USA, 57–62. DOI: http://dx.doi.org/10.1145/2095050.2095061

[6] Antonios Liapis, Georgios N Yannakakis, and Julian Togelius. 2013. Sentient Sketchbook: Computer-aided game level authoring.. In *FDG.* 213–220.

[7] Tiago Machado, Ivan Bravi, Zhu Wang, Andy Nealen, and Julian Togelius. 2016. Shopping for Game Mechanics. (2016).

[8] Tobias Mahlmann, Anders Drachen, Julian Togelius, Alessandro Canossa, and Georgios N Yannakakis. 2010. Predicting player behavior in tomb raider: Underworld. In *Computational Intelligence and Games (CIG), 2010 IEEE Symposium on.* IEEE, 178–185.

[9] Ben Medler and others. 2009. Generations of game analytics, achievements and high scores. *Eludamos. Journal for Computer Game Culture* 3, 2 (2009), 177–194.

[10] Mark J. Nelson and Michael Mateas. 2009. A Requirements Analysis for Videogame Design Support Tools. In *Proceedings of the 4th International Conference on Foundations of Digital Games (FDG '09).* ACM, New York, NY, USA, 137–144. DOI: http://dx.doi.org/10.1145/1536513.1536543

[11] Diego Perez, Spyridon Samothrakis, Julian Togelius, Tom Schaul, Simon Lucas, Adrien Couëtoux, Jeyull Lee, Chong-U Lim, and Tommy Thompson. 2015. The 2014 general video game playing competition. (2015).

[12] Tom Schaul. 2013. A video game description language for model-based or interactive learning. In *Computational Intelligence in Games (CIG), 2013 IEEE Conference on.* IEEE, 1–8.

[13] Marco Scirea, Yun-Gyung Cheong, Mark J. Nelson, and Byung-Chull Bae. 2014. Evaluating Musical Foreshadowing of Videogame Narrative Experiences. In *Proceedings of the 9th Audio Mostly: A Conference on Interaction With Sound (AM '14).* ACM, New York, NY, USA, Article 8, 7 pages. DOI: http://dx.doi.org/10.1145/2636879.2636889

[14] Noor Shaker, Mohammad Shaker, and Julian Togelius. 2013. Ropossum: An Authoring Tool for Designing, Optimizing and Solving Cut the Rope Levels.. In *AIIDE.*

[15] Ben Shneiderman. 2007. Creativity support tools: Accelerating discovery and innovation. *Commun. ACM* 50, 12 (2007), 20–32.

[16] Brian A. Smith and Shree K. Nayar. 2016. Mining Controller Inputs to Understand Gameplay. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16).* ACM, New York, NY, USA, 157–168. DOI: http://dx.doi.org/10.1145/2984511.2984543

[17] Gillian Smith, Jim Whitehead, and Michael Mateas. 2010. Tanagra: A mixed-initiative level design tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games.* ACM, 209–216.

[18] Guenter Wallner and Simone Kriglstein. 2016. Visualizations for Retrospective Analysis of Battles in Team-based Combat Games: A User Study. In *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play (CHI PLAY '16).* ACM, New York, NY, USA, 22–32. DOI: http://dx.doi.org/10.1145/2967934.2968093

[19] Günter Wallner, Simone Kriglstein, Florian Gnadlinger, Michael Heiml, and Jochen Kranzer. 2014. Game User Telemetry in Practice: A Case Study. In *Proceedings of the 11th Conference on Advances in Computer Entertainment Technology (ACE '14).* ACM, New York, NY, USA, Article 45, 4 pages. DOI: http://dx.doi.org/10.1145/2663806.2663859

[20] Georg Zoeller. 2010. Development telemetry in video games projects. In *Game developers conference.*