Improving Video Game Balance Testing Using Autonomous Agents

Cristiano Politowski, Doctor of Philosophy (Software Engineering) Thesis Defense, Concordia University, Montreal, Canada

Dr. Antonio Bucchiarone	*External Examiner*
Dr. Wahab Hamou-Lhadj	*External to Program*
Dr. Emad Shihab	*Examiner*
Dr. Juergen Rilling	*Examiner*
Dr. Bruno Lee	*Chair*
Dr. Yann-Gaël Guéhéneuc	*Thesis Supervisor*
Dr. Fabio Petrillo	*Thesis Co-Supervisor*

UNIVERSITÉ Concordia UNIVERSITY

Outline

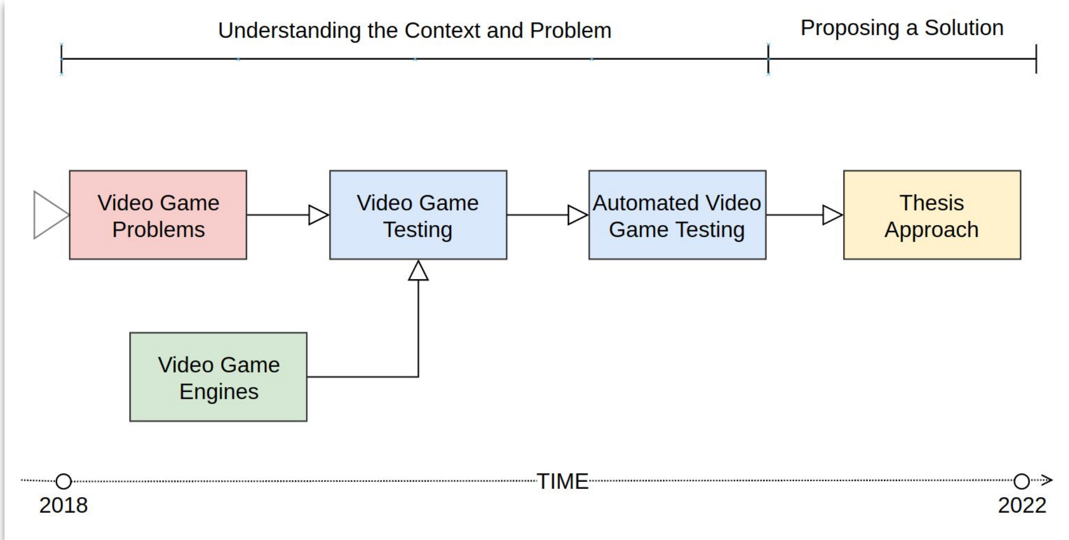
Video Game **Problems**.

Video Game **Engines**.

Video Game **Testing**.

Automated Video Game **Testing**.

Automated Video Game **Testing** for **Balancing**.



50 years of evolution

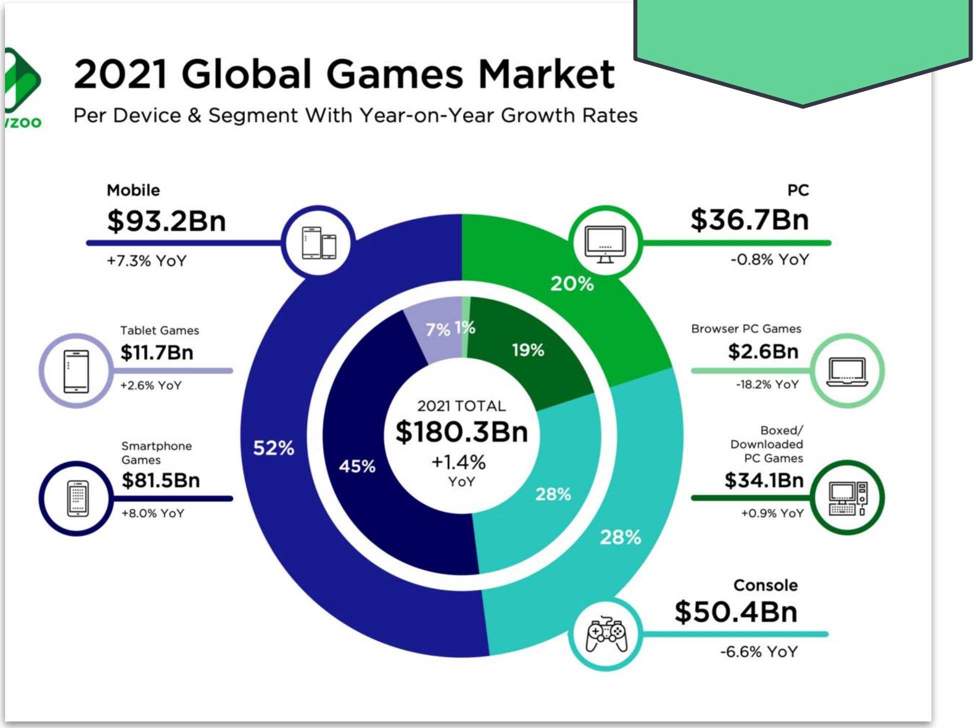
The Game Industry is Big and Billionaire

$180 Bi

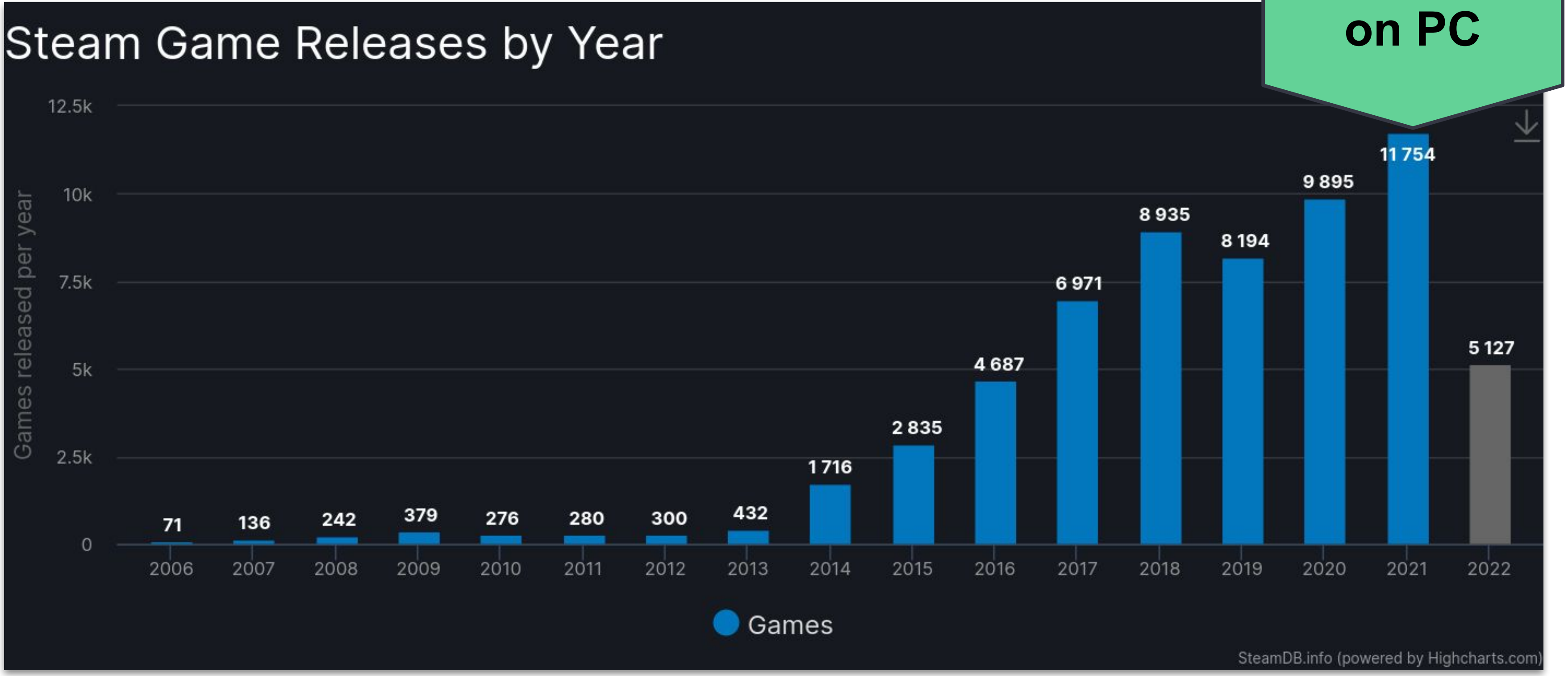
The game industry has increased its **profits** over the years.

The **cost** of big video games is doubling for each new console generation.

PS5 games will cost **$200 millions**, at least.



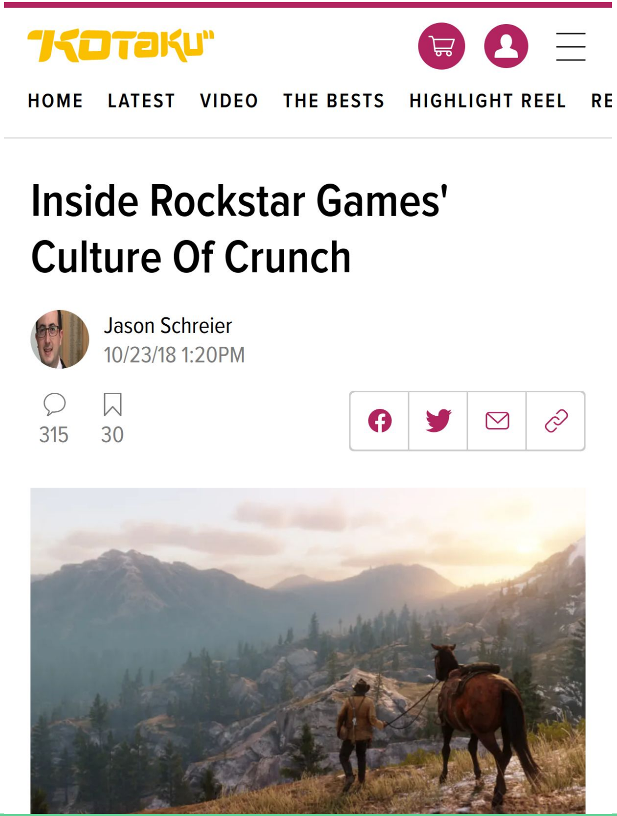
The Game Industry is Overcrowded



SteamDB.info (powered by Highcharts.com)

12K only on PC

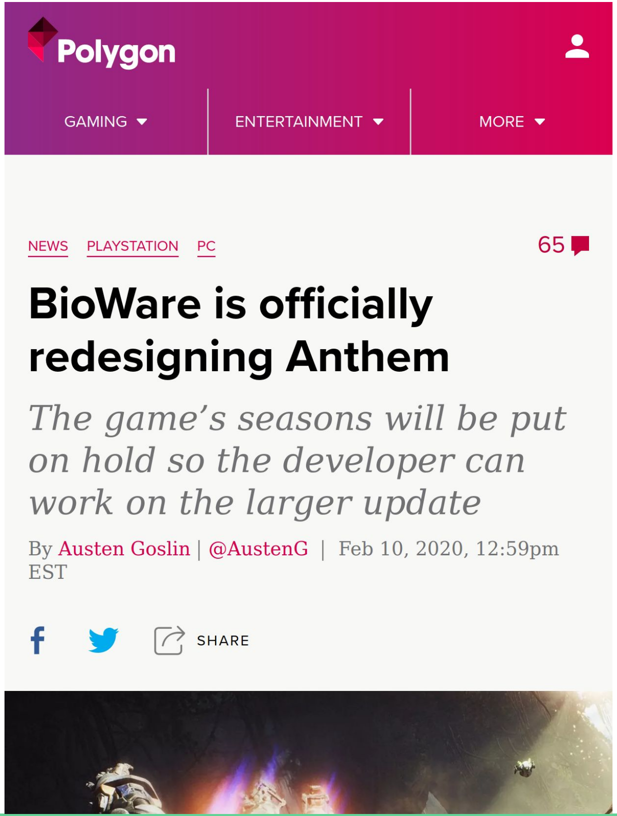
The Game Industry Has Problems



Inside Rockstar Games' Culture Of Crunch

Jason Schreier
10/23/18 1:20PM

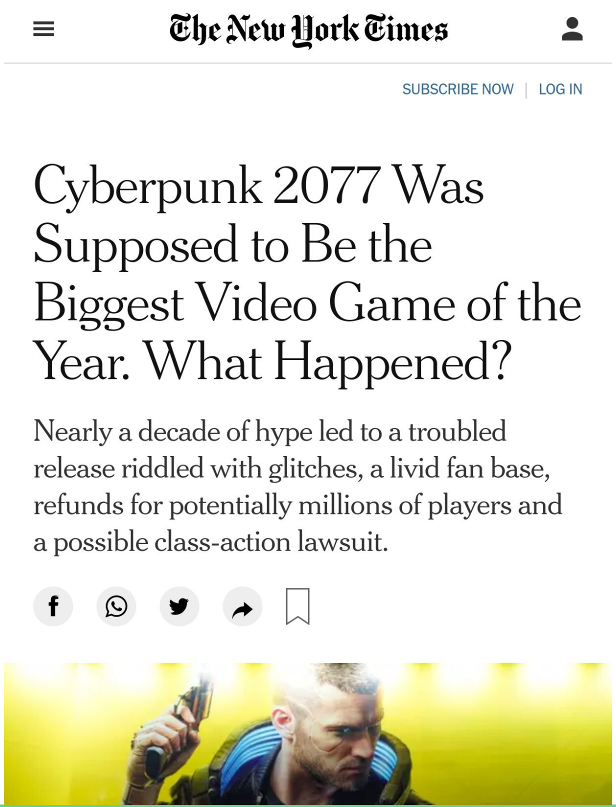
315 30



BioWare is officially redesigning Anthem

The game's seasons will be put on hold so the developer can work on the larger update

By Austen Goslin | @AustenG | Feb 10, 2020, 12:59pm EST



Cyberpunk 2077 Was Supposed to Be the Biggest Video Game of the Year. What Happened?

Nearly a decade of hype led to a troubled release riddled with glitches, a livid fan base, refunds for potentially millions of players and a possible class-action lawsuit.

Game Testing

Manual video game testing (playtesting).

Control the quality of the game.

- Bugs.
- Game performance.
- Game completion.
- Balance.

In my thesis, we research, design, and implement a game-testing approach to balance video games.

Instead of manually testing games,
we use autonomous agents
to assess the game's balance.

Outline

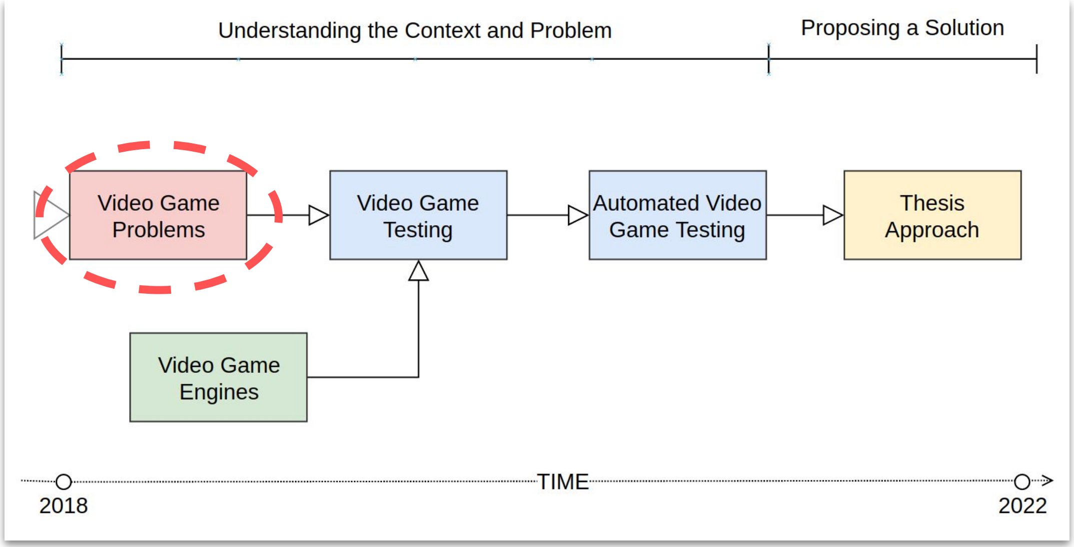
Video Game **Problems.**

Video Game **Engines.**

Video Game **Testing.**

Automated Video Game **Testing.**

Automated Video Game **Testing** for **Balancing.**



Video Game **Problems**[0,1,2]

[0] Dataset of Video Game Development Problems (2020) - *17th International Conference on Mining Software Repositories*

[1] Game Industry Problems: An Extensive Analysis of the Gray Literature (2021) - *Information and Software Technology Journal*

[2] Video Game Project Management Anti-patterns (2022) - *6th Workshop on Games and Software Engineering*

Video Game Problems: **Introduction**

Investigate problems in the video-game industry.

RQ.1: What are the **problems** in game industry?

RQ.2: How do they **evolve over time**?

Video Game Problems: **Method**

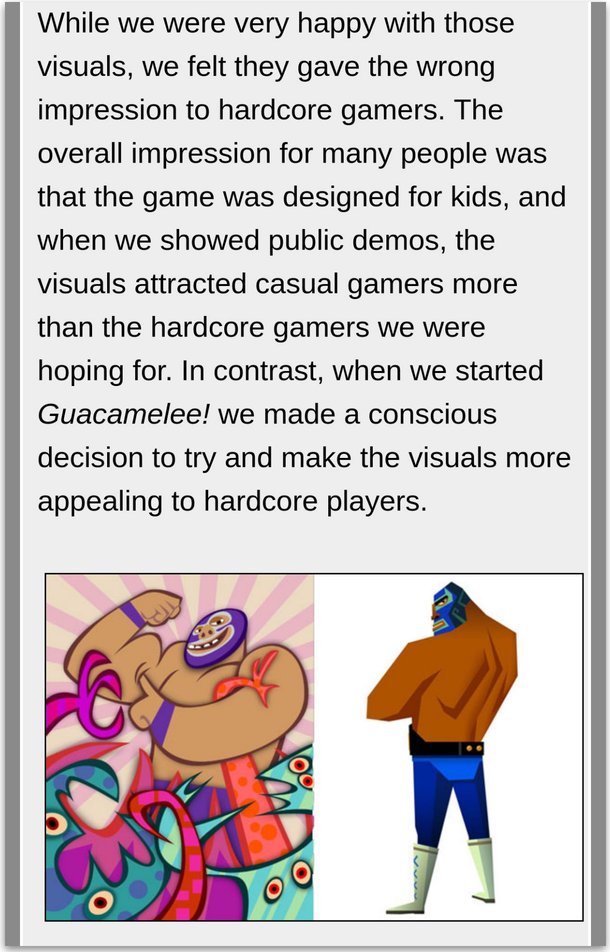
Postmortems ("war stories").

Articles written by game developers.

"What went wrong and right".

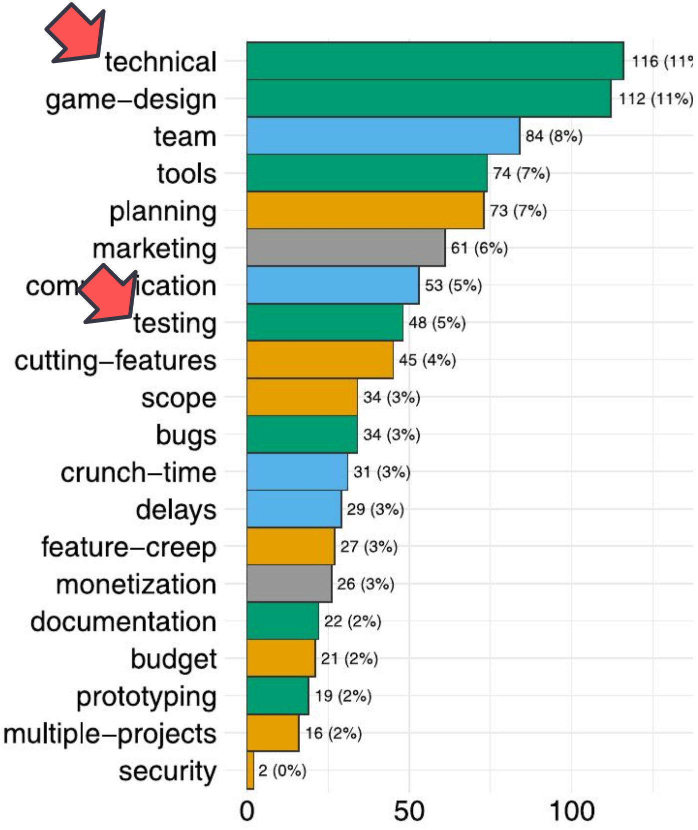
200+ postmortems from 1997-2019.

We found a total of **927** problems.

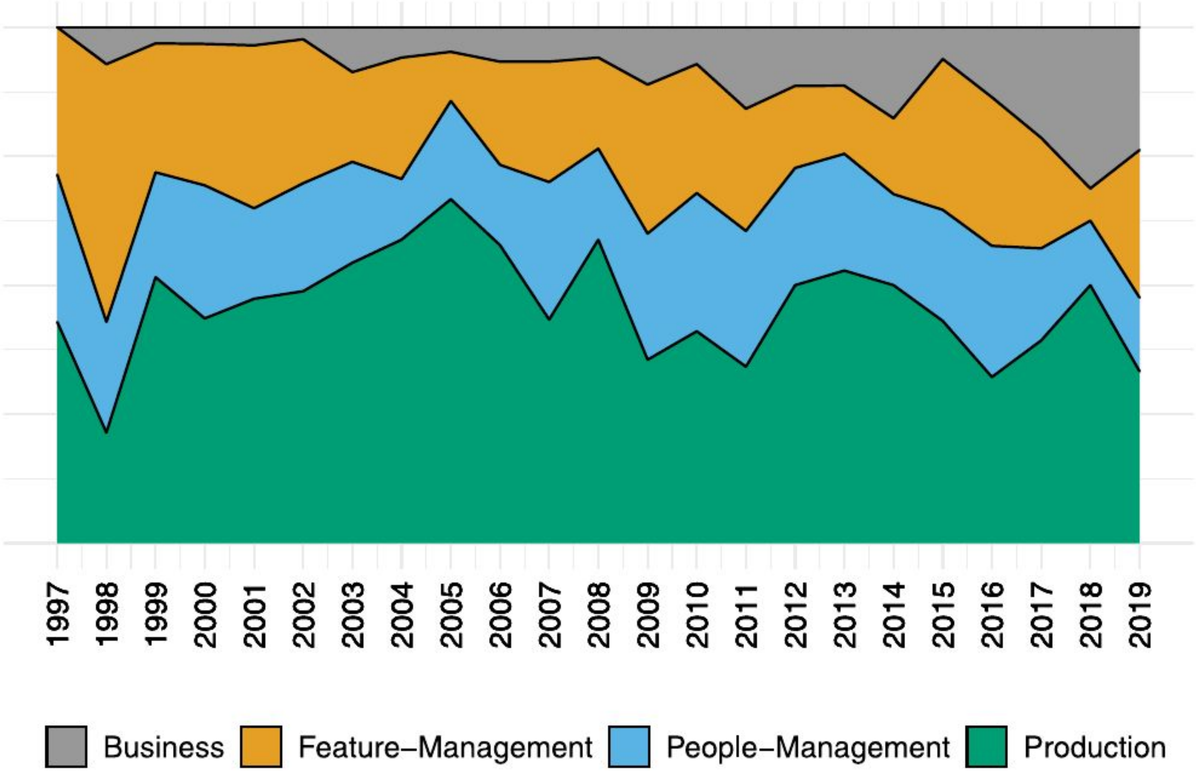


While we were very happy with those visuals, we felt they gave the wrong impression to hardcore gamers. The overall impression for many people was that the game was designed for kids, and when we showed public demos, the visuals attracted casual gamers more than the hardcore gamers we were hoping for. In contrast, when we started *Guacamelee!* we made a conscious decision to try and make the visuals more appealing to hardcore players.

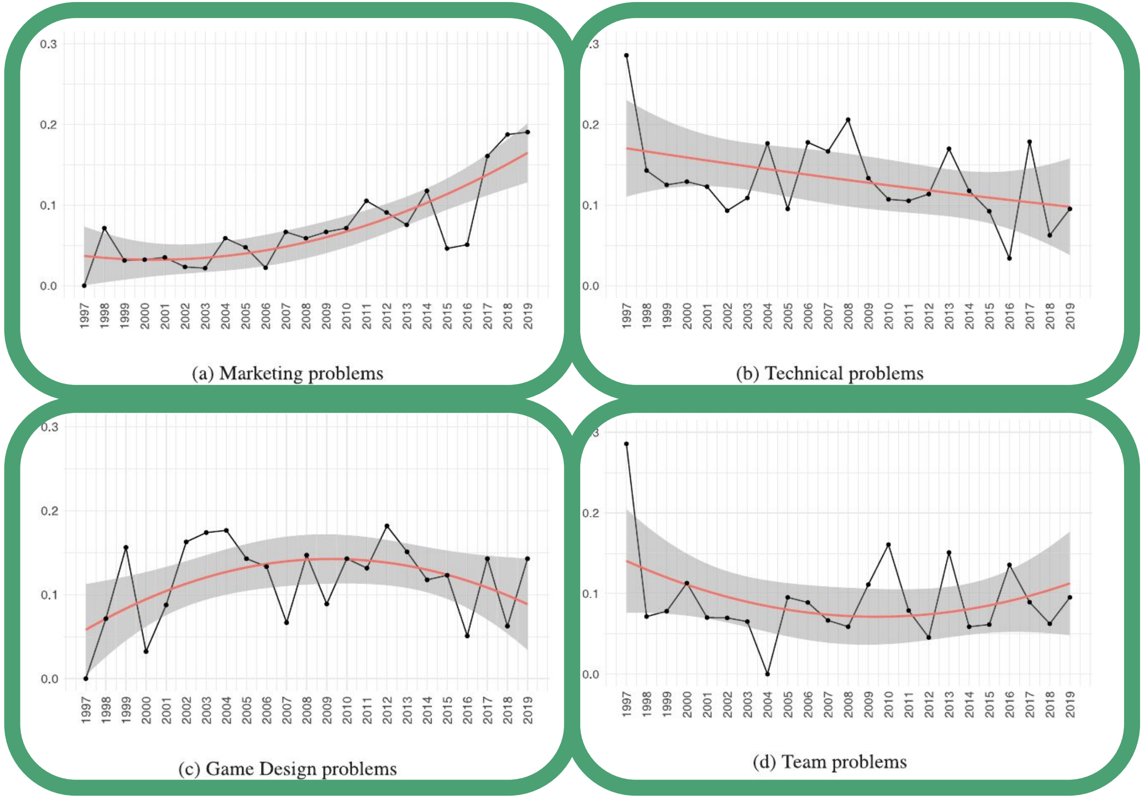
Video Game Problems: **Results**



Video Game Problems: **Results Over the Years**



Video game problems: **Trends Over the Years**



(a) Marketing problems

(b) Technical problems

(c) Game Design problems

(d) Team problems

Video Game Problems: **Threats to Validity**

Dataset based only on postmortems.

Dataset only has successful projects.

Developers might not tell the whole story.

Video Game Problems: **Conclusion**

Many problems require **project-specific** solutions that are hard to **generalize**.

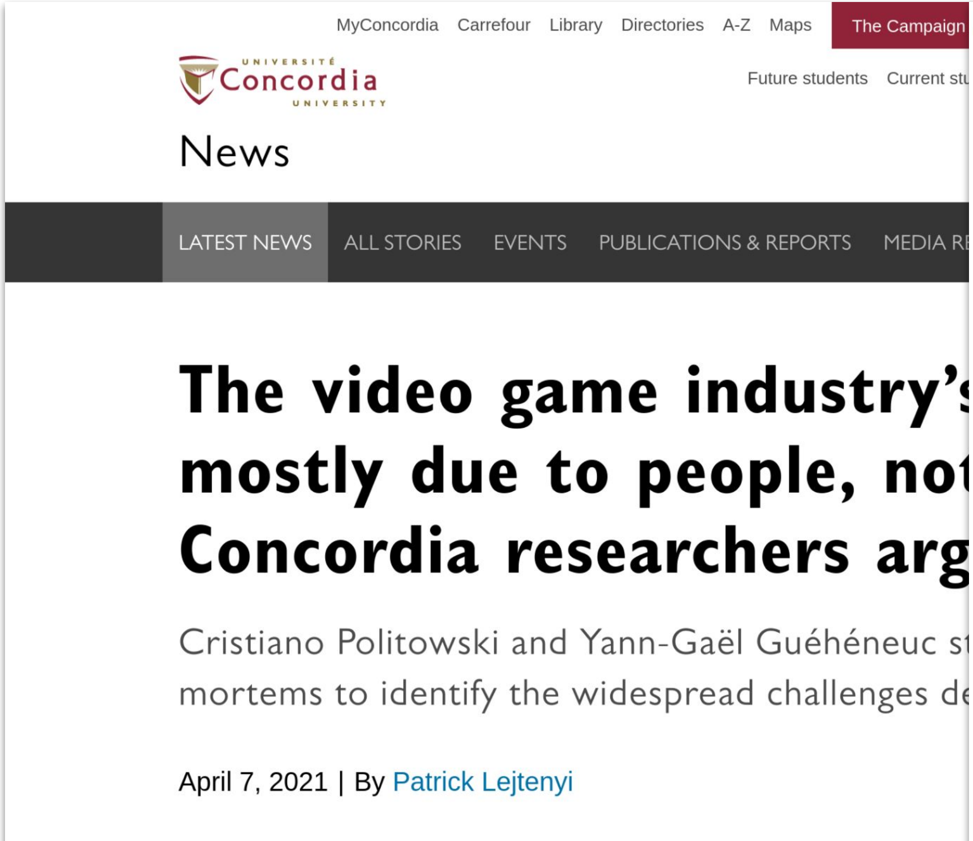
Only 5% **testing** problems (“we did not test enough”).

No traditional testing techniques like **unit testing.**

No automated testing techniques like in Software Engineering.

Testing, playtesting, and even QA used interchangeably.

Video Game Problems: **Outcome**



MyConcordia Carrefour Library Directories A-Z Maps The Campaign

Université Concordia University

Future students Current stu

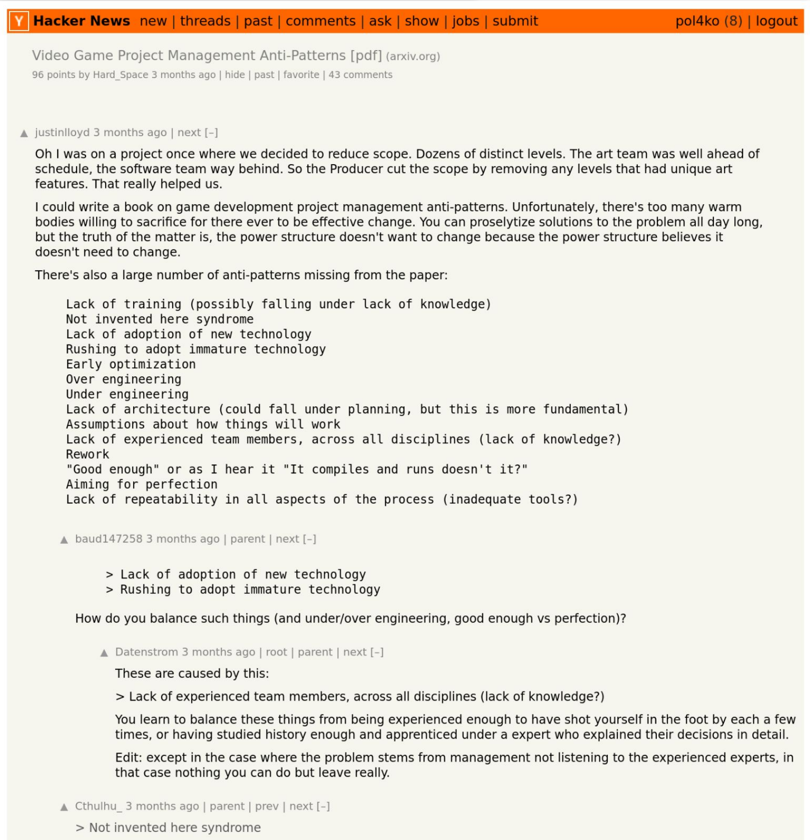
News

LATEST NEWS ALL STORIES EVENTS PUBLICATIONS & REPORTS MEDIA RE

The video game industry's mostly due to people, not Concordia researchers arg

Cristiano Politowski and Yann-Gaël Guéhéneuc st mortems to identify the widespread challenges de

April 7, 2021 | By Patrick Lejtenyi



Y **Hacker News** new | threads | past | comments | ask | show | jobs | submit pol4ko (8) | logout

Video Game Project Management Anti-Patterns [pdf] (arxiv.org)
96 points by Hard_Space 3 months ago | hide | past | favorite | 43 comments

▲ justinlloyd 3 months ago | next [–]

Oh I was on a project once where we decided to reduce scope. Dozens of distinct levels. The art team was well ahead of schedule, the software team way behind. So the Producer cut the scope by removing any levels that had unique art features. That really helped us.

I could write a book on game development project management anti-patterns. Unfortunately, there's too many warm bodies willing to sacrifice for there ever to be effective change. You can proselytize solutions to the problem all day long, but the truth of the matter is, the power structure doesn't want to change because the power structure believes it doesn't need to change.

There's also a large number of anti-patterns missing from the paper:

```
Lack of training (possibly falling under lack of knowledge)
Not invented here syndrome
Lack of adoption of new technology
Rushing to adopt immature technology
Early optimization
Over engineering
Under engineering
Lack of architecture (could fall under planning, but this is more fundamental)
Assumptions about how things will work
Lack of experienced team members, across all disciplines (lack of knowledge?)
Rework
"Good enough" or as I hear it "It compiles and runs doesn't it?"
Aiming for perfection
Lack of repeatability in all aspects of the process (inadequate tools?)
```

▲ baud147258 3 months ago | parent | next [–]

```
> Lack of adoption of new technology
> Rushing to adopt immature technology
```

How do you balance such things (and under/over engineering, good enough vs perfection)?

▲ Datenstrom 3 months ago | root | parent | next [–]

These are caused by this:

> Lack of experienced team members, across all disciplines (lack of knowledge?)

You learn to balance these things from being experienced enough to have shot yourself in the foot by each a few times, or having studied history enough and apprenticed under a expert who explained their decisions in detail.

Edit: except in the case where the problem stems from management not listening to the experienced experts, in that case nothing you can do but leave really.

▲ Cthulhu_ 3 months ago | parent | prev | next [–]

> Not invented here syndrome

Outline

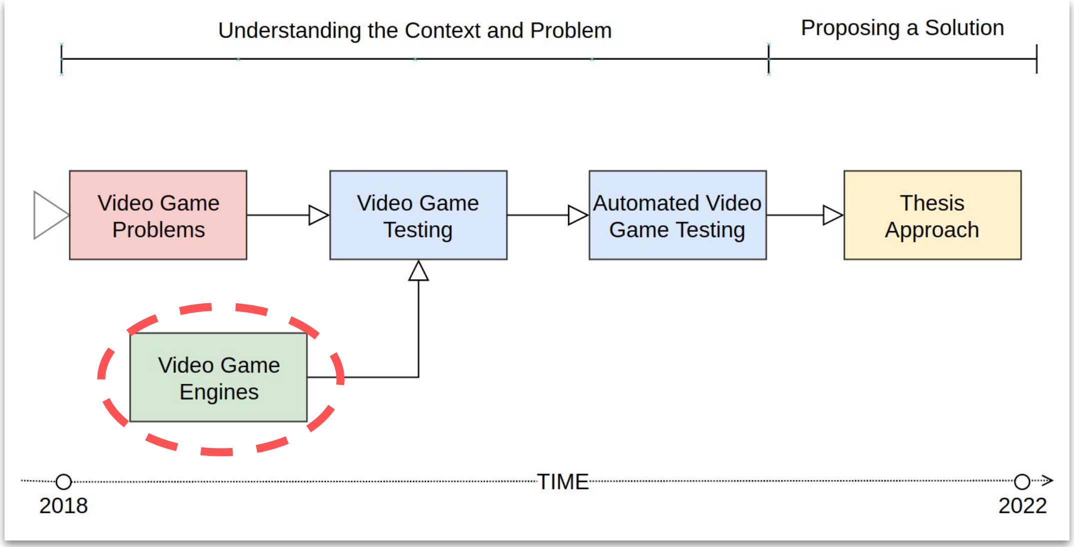
Video Game **Problems.**

Video Game **Engines.**

Video Game **Testing.**

Automated Video Game **Testing.**

Automated Video Game **Testing** for **Balancing.**



Video Game **Engines**[0]

[0] Are Game Engines Software Frameworks? A Three-Perspective Study (2021) - *Journal of Systems and Software*

Video Game Engines: **Introduction**

Compare open-source **video-game engines** with traditional open-source **software frameworks**.

RQ: Do game engines share **similar characteristics** with software frameworks?

Video Game Engines: **Method**

We study open-source game engines from three perspectives:

- **Literature.**
- **Code.**
- **Human**.

We survey **124 engine developers**.

Video Game Engines: **Results - Literature**

Framework:

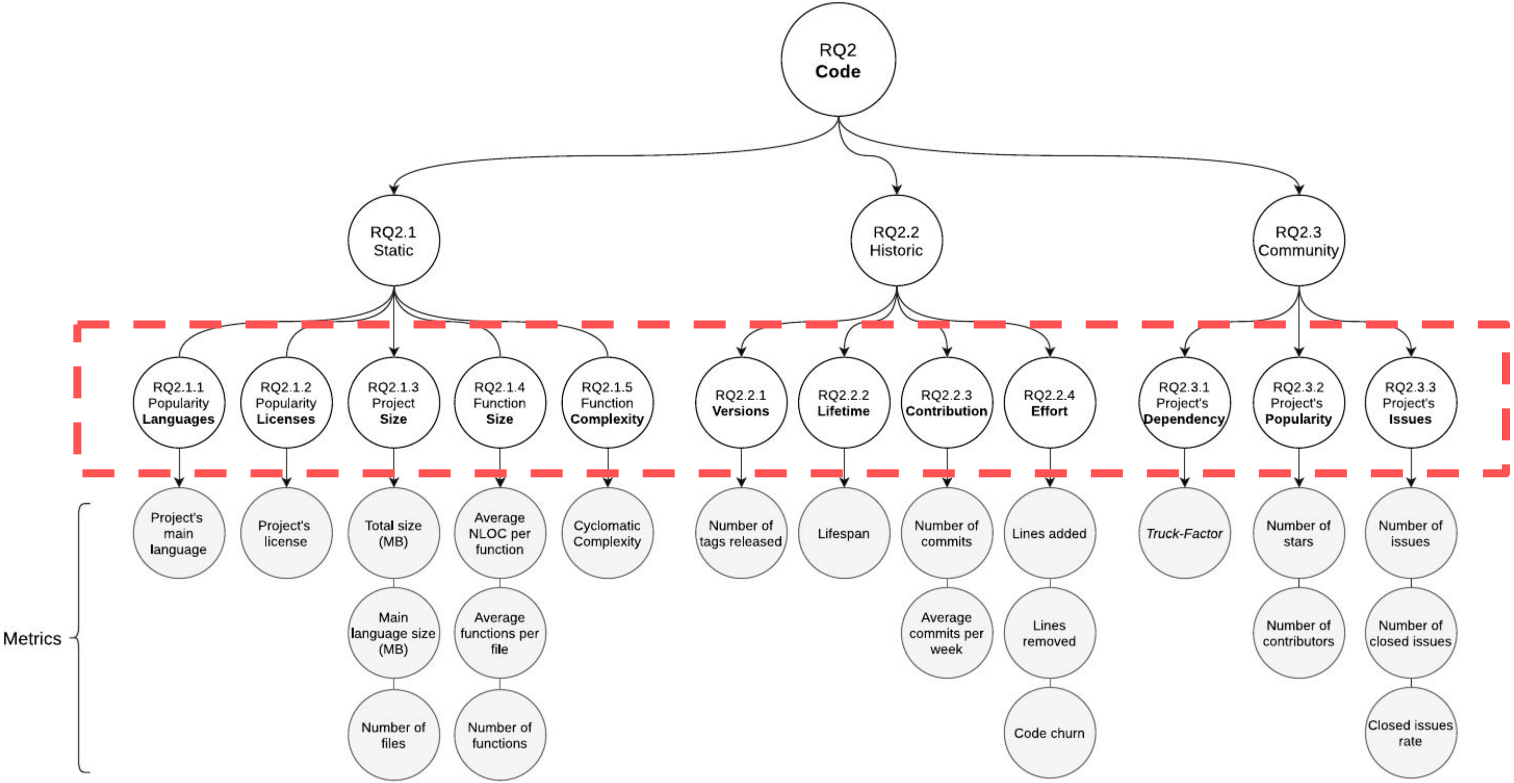
- A reusable set of libraries
- Help developers focus their work on higher level tasks
- Provide a reusable pattern to speed up development [0]

Game Engine:

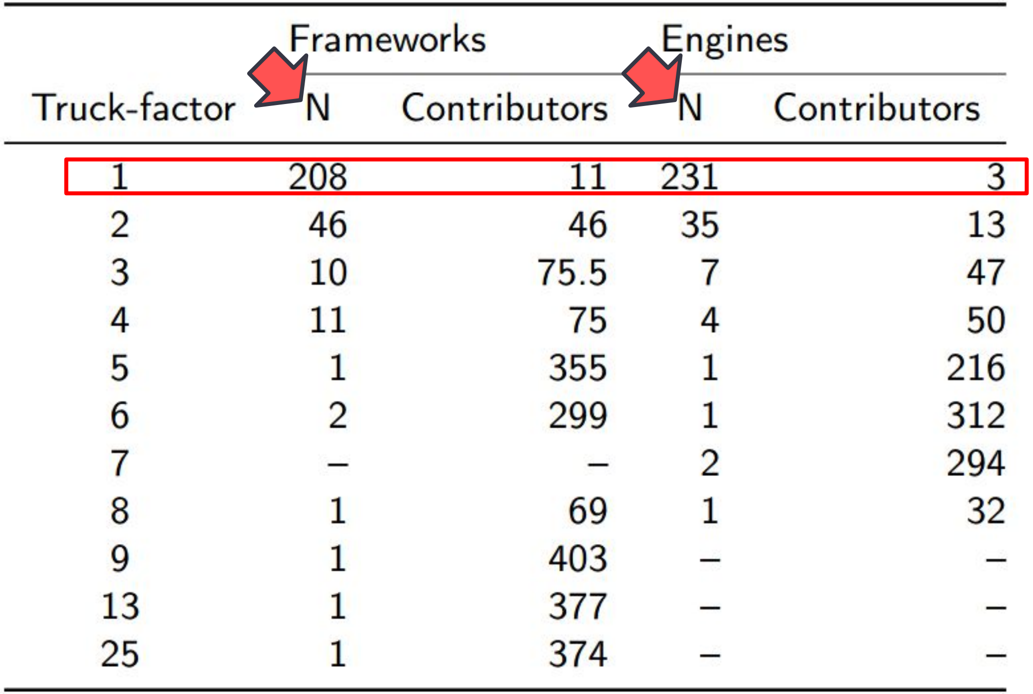
- Technology “behind the game” [1]
- Separation of game engine from “assets” [2]
- Work collaboratively on the game as a team [2]

[0] Github definition
[1] ID Software/John Carmack (DOOM game circa 1993)
[2] Henry Lowood. Game Engines and Game History, 2014

Video Game Engines: **Results - Code**



Video Game Engines: **Results - Code – Dependency**

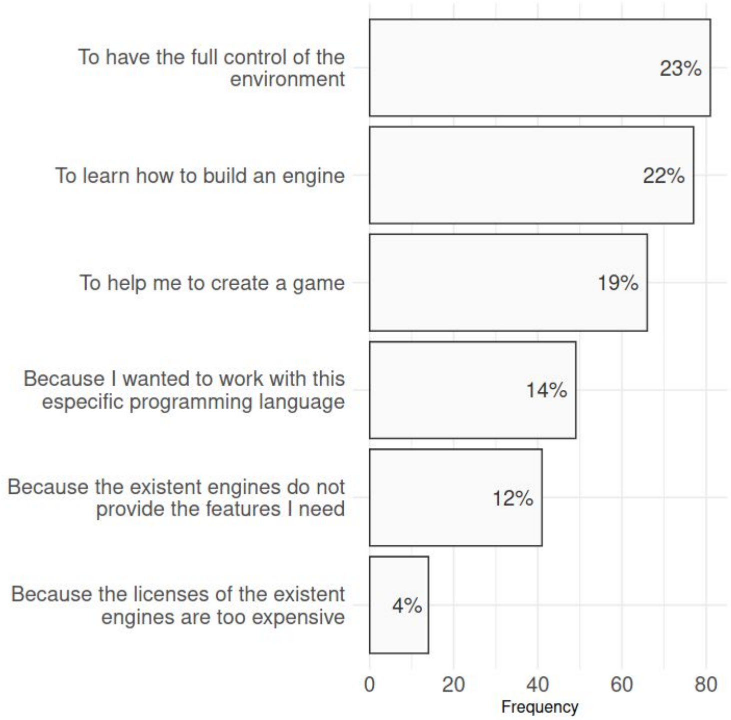


	Frameworks		Engines	
Truck-factor	N	Contributors	N	Contributors
1	208	11	231	3
2	46	46	35	13
3	10	75.5	7	47
4	11	75	4	50
5	1	355	1	216
6	2	299	1	312
7	–	–	2	294
8	1	69	1	32
9	1	403	–	–
13	1	377	–	–
25	1	374	–	–

Low
dependency
(good)

Video Game Engines: **Results - Human – Survey**

Why did you create or collaborated with a video-game engine project?



Video Game Engines: **Threats to Validity**

Popularity of the frameworks could lead to unfair comparisons.

Most of the open-source engines are personal projects.

Tag system on github is not 100% reliable.

Great part of the released games use closed-source engines (Unreal & Unity).

Video Game Engines: **Conclusion**

Open-source game engines **share similarities** with open-source **frameworks**.

Open-source game engines are mainly **personal** projects.

The communities around **framework** projects are **larger**.

More care should be given to the **documentation** (low truck-factor).

Outline

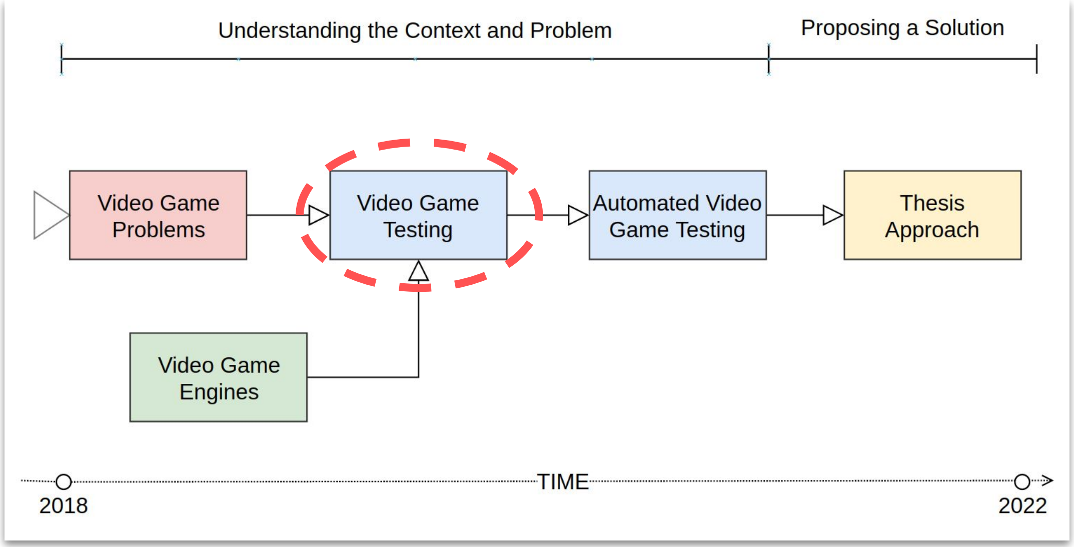
Video Game **Problems.**

Video Game **Engines.**

Video Game **Testing.**

Automated Video Game **Testing.**

Automated Video Game **Testing** for **Balancing.**



Video Game **Testing**[0]

[0] A Survey of Video Game Testing (2021) - *2021 International Conference on Automation of Software Test (AST)*

Video Game Testing: **Introduction**

Investigate: **processes**, **techniques**, **gaps**, **concerns**, and **point-of-views**.

RQ: What is the **testing process** in game development?

Video Game Testing: **Method**

We used two sources: **academic** and **gray** literature.

- 96 papers.
- 200 Postmortems (48 testing problems).
- 5 talks.
- 4 tech blogs.

Video Game Testing: **Results**

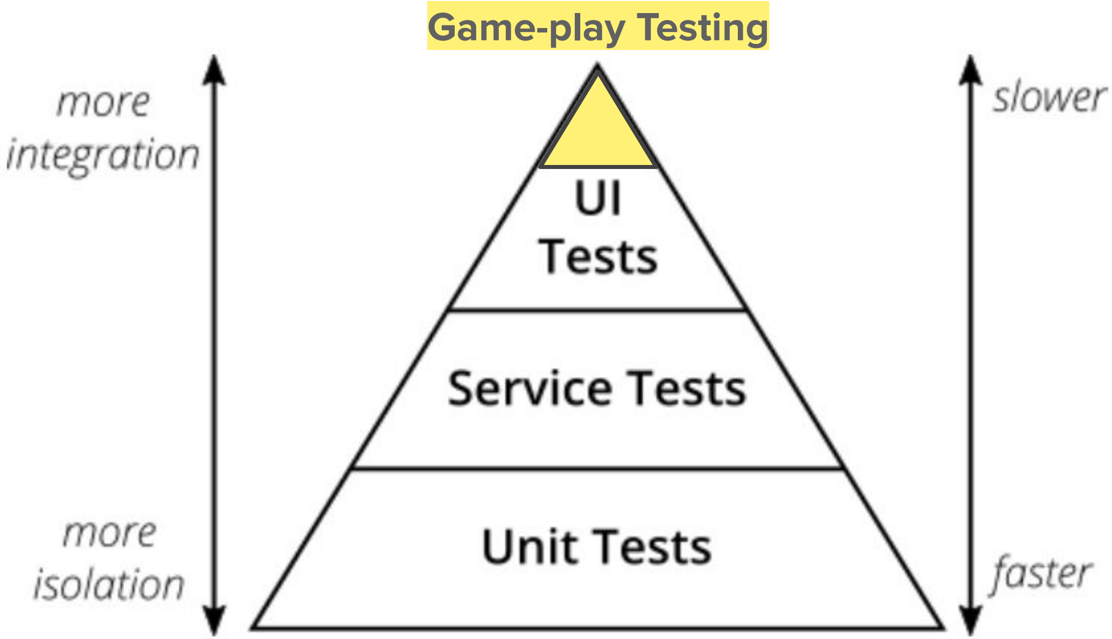
Testing a game means **playing** it.

Industry relies on **testers' cognition**.

Automation is **difficult**.

It is **hard** to **generalize** to different games.

Video Game Testing: **Results - Testing Pyramid**



https://martinfowler.com/articles/practical-test-pyramid.html

Video Game Testing: **Threats to Validity**

Hard to find trusted information.

Academic studies diverge greatly.

Absence of a common field.

Video Game Testing: **Conclusion**

The **testing strategies** should take into account the **particularities** of each game.

Game testers (QA testers) should work alongside **software testers** (engineers).

Automation is **overlooked**, as it relies on **manual** testing.

Subjective factors: **fun** and **balancing**.

Game **studios** acknowledged the **importance** of testing.

Main issue: **lack of testing**.

Outline

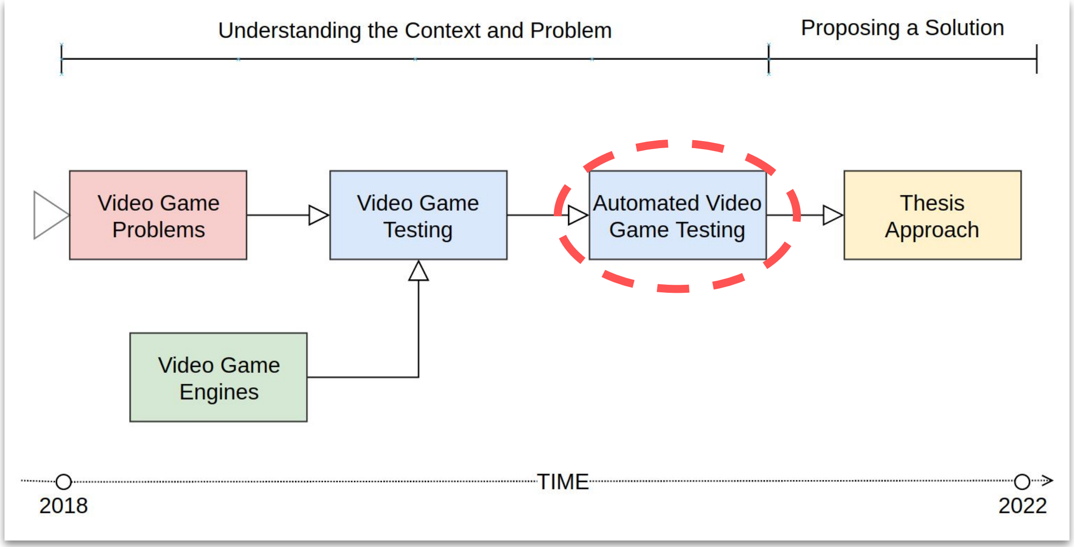
Video Game **Problems.**

Video Game **Engines.**

Video Game **Testing.**

Automated Video Game **Testing.**

Automated Video Game **Testing** for **Balancing.**



Automated Video Game **Testing**[0]

[0] Towards Automated Video Game Testing: Still a Long Way to Go (2022) - *6th Workshop on Games and Software Engineering*

Automated Video Game Testing: **Introduction**

Search, **identify**, and **catalogue** automated video game testing **techniques**.

We investigate **gaps** between academic solutions in the literature.

The **needs** of video game developers in the industry.

Automated Video Game Testing: **Method**

Literature Review and Online **Survey.**

We used a recent **study** [0] that already collected works about video game testing.

We performed **full snowballing** and add further exclusion criteria.

We applied a **survey** with video game developers.

[0] A. M. Albaghajati and M. A. K. Ahmed, “Video Game Automated Testing Approaches: An Assessment Framework”, Transactions on Games, IEEE CS Press, 2020.

Automated Video Game Testing: **Method - Lit. Review**

The **whole dataset** consists of **166 papers** from 2004 to 2021.

Classifying criterias:

- **Study type:** Theoretical or Applied.
- **Testing:** Yes or No.
- **Automated:** Yes or No.
- **Machine Learning:** Yes or No.
- **Test Objective:** The goal of the testing.

114 papers had applied approach:

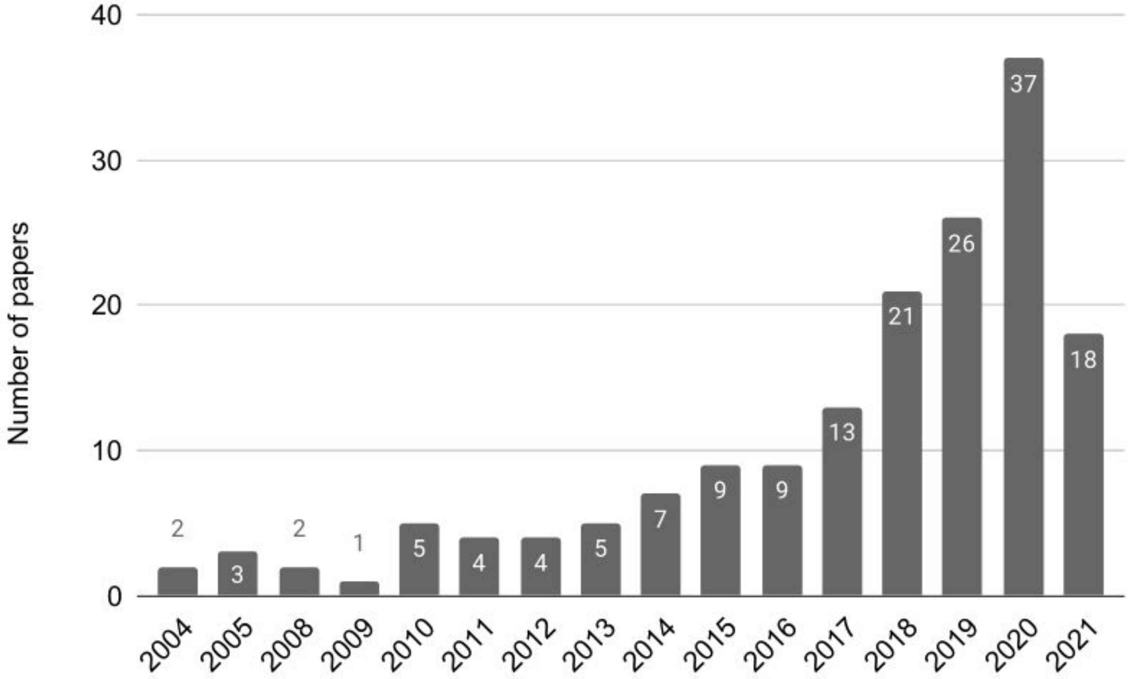
- **80** used **automation.**
- **53** used some types of **intelligent agents** or **heuristic**.

Automated Video Game Testing: **Results - Test. Objectives**

In that set of **53 papers**, the most discussed **test objectives** are:

- **Balancing** the gameplay (19 papers).
- Game **exploration** (11 papers).
- Finding **bugs** (6 papers).
- Player **modeling** (6 papers).
- Game mechanics, UI, UX, visual correctness, collision, and visualization.

Automated Video Game Testing: **Results - Histogram**



Automated Video Game Testing: **Method - Survey**

We divided the survey into four sections:

- **Background**.
- **Manual playtesting activities**.
- **Assess academic techniques/solutions** for automated playtesting.
- **Future of game testing** (optional).

We focused on the three most common testing objectives:

❑ **Balancing**
❑ **Exploration**
❑ **Finding bugs**

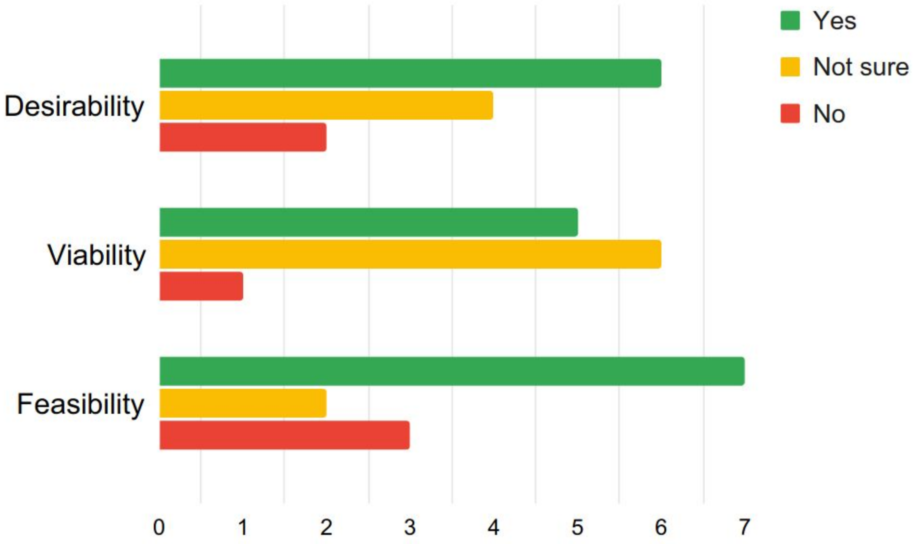
Automated Video Game Testing: **Results - Survey Example**

`Automated Game Testing with ICARUS: Intelligent Completion of Adventure Riddles via Unsupervised Solving (2017)`

Uses autonomous agents to complete the game like a “speedrun” and spot crashes/freezes and blocker (soft lock).

To automate the process, the authors used the script language Lua on top of the Visionary game engine.

This solution was deemed as the most feasible of all solutions and very desirable and viable.



Automated Video Game Testing: **Threats to Validity**

It is not trivial to **translate** complex **solutions** for the survey.

Testers do not have the **macro** vision of the process.

The **low number** of **respondents** might introduce bias to the study.

The respondents do not have **knowledge** in **AI**.

Automated Video Game Testing: **Conclusion**

We conclude that **there is still a long way to go for video game testing**. Especially on how should we test video games.

Testing goals and testing **oracles** should be clear in the papers.

Offer a **replication package** and source code.

Practitioners: new **testing approaches** that do not disrupt the **workflow**.

Researchers: implement a **feasible automated solution**.

Outline

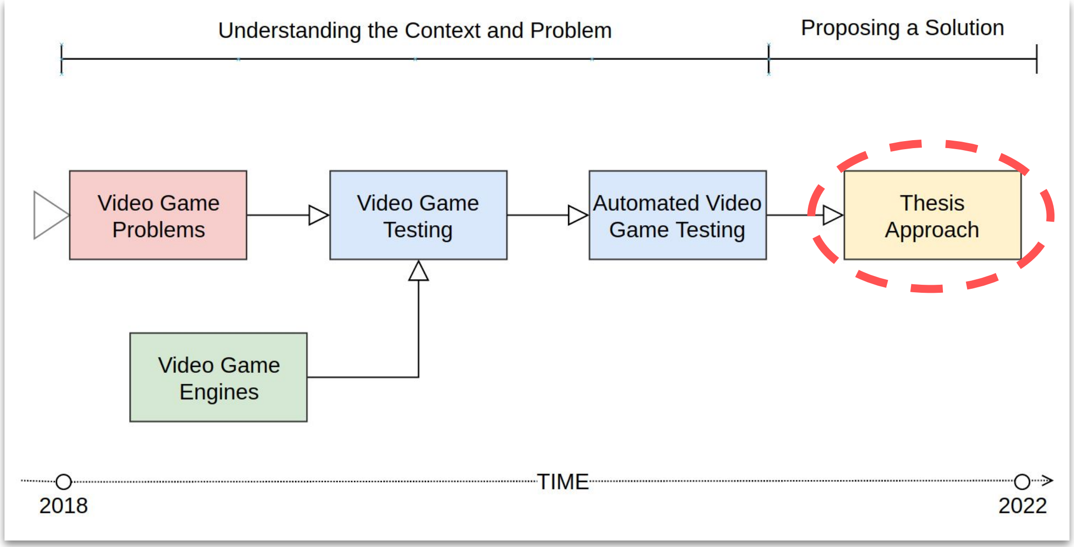
Video Game **Problems.**

Video Game **Engines.**

Video Game **Testing.**

Automated Video Game **Testing.**

Automated Video Game **Testing** for **Balancing.**



Automated Video Game **Testing** for **Balancing**

Automated Video Game Testing for Balancing: **Introduction**

For every new **feature** or **change**, the game must be **tested**.

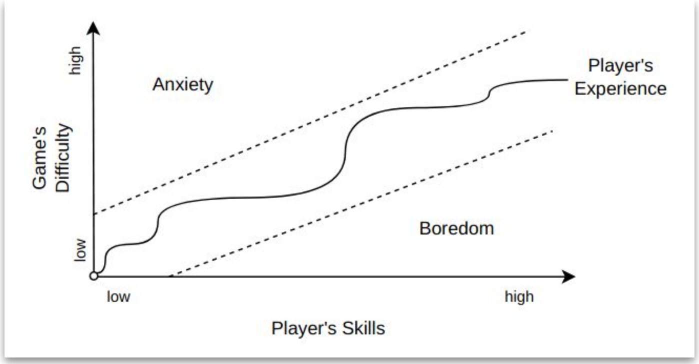
Is it now **too easy**, **too hard**, **impossible to complete**, etc.

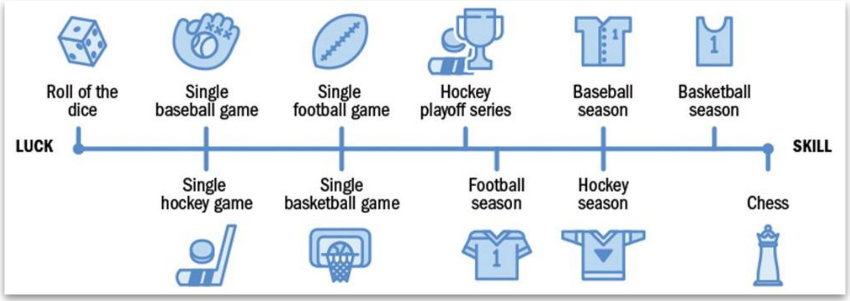
Objective: **Test** and **assess** the **balance** of the game.

Automated Video Game Testing for Balancing: **Method**

Challenge vs. Success[0]: keeping the player **engaged** considering the game **difficulty** and the player's **skills**.

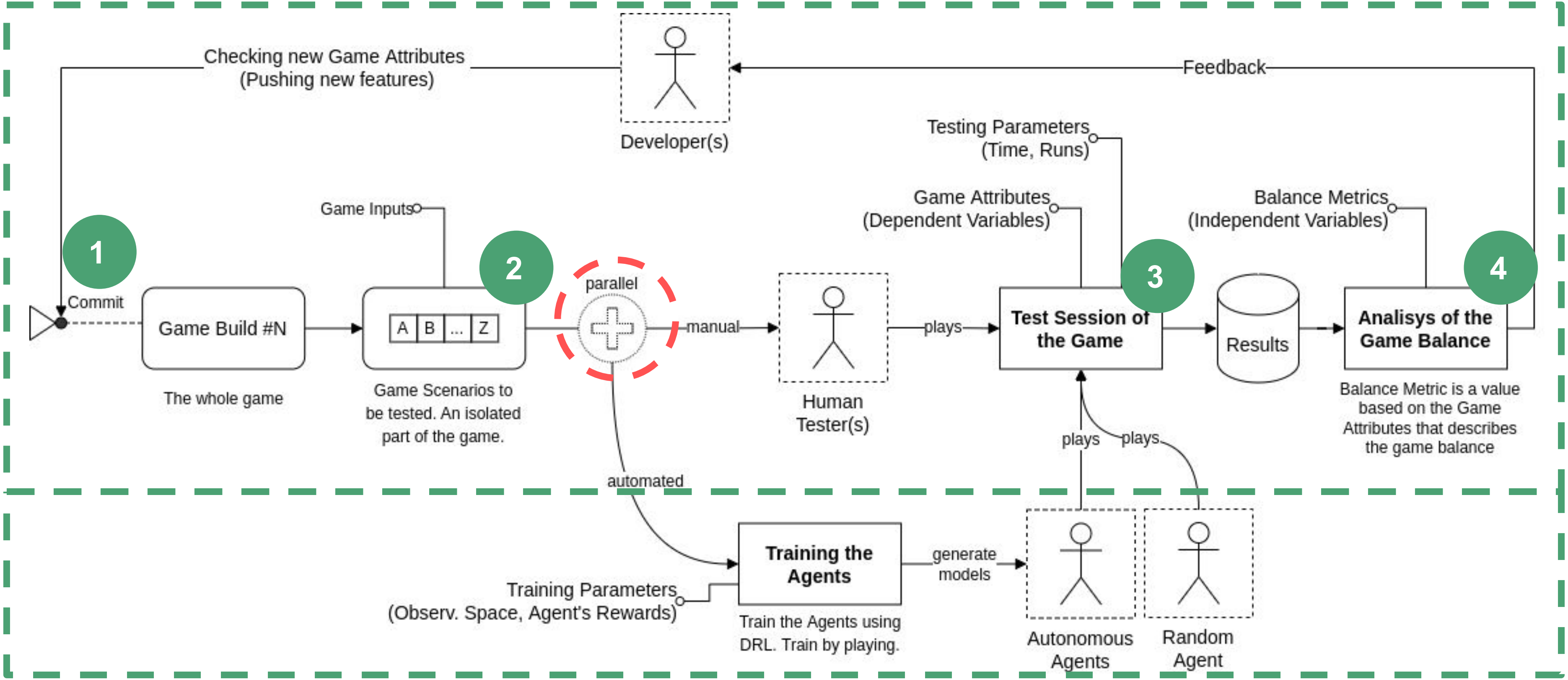
Skill vs. Chance[0]: success depends more or less on **luck** instead of the players' **skills**.





[0] J. Schell, The art of game design: a book of lenses (2008)

Automated Video Game Testing for Balancing: **Method**



Automated Video Game Testing for Balancing: **Case B**

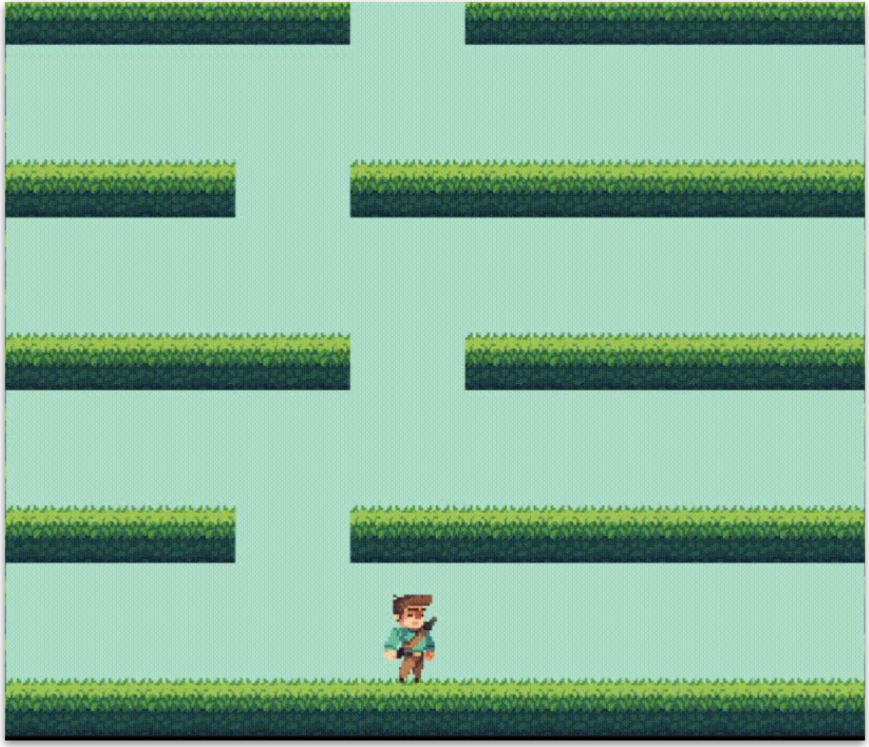
Platform has gaps randomly generated.

Keep the character below the bottom line.

If player survives above the platform: one point (`+1 point`).

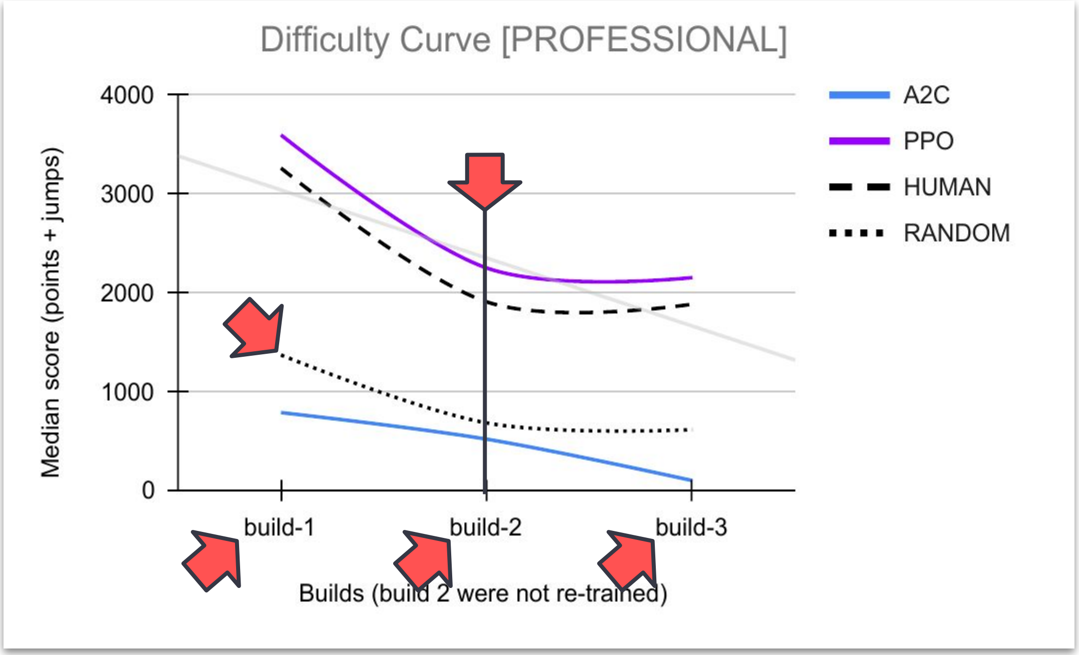
The scrolling speed increases as time passes.

Actions: `LEFT`, `RIGHT` and `JUMP`.

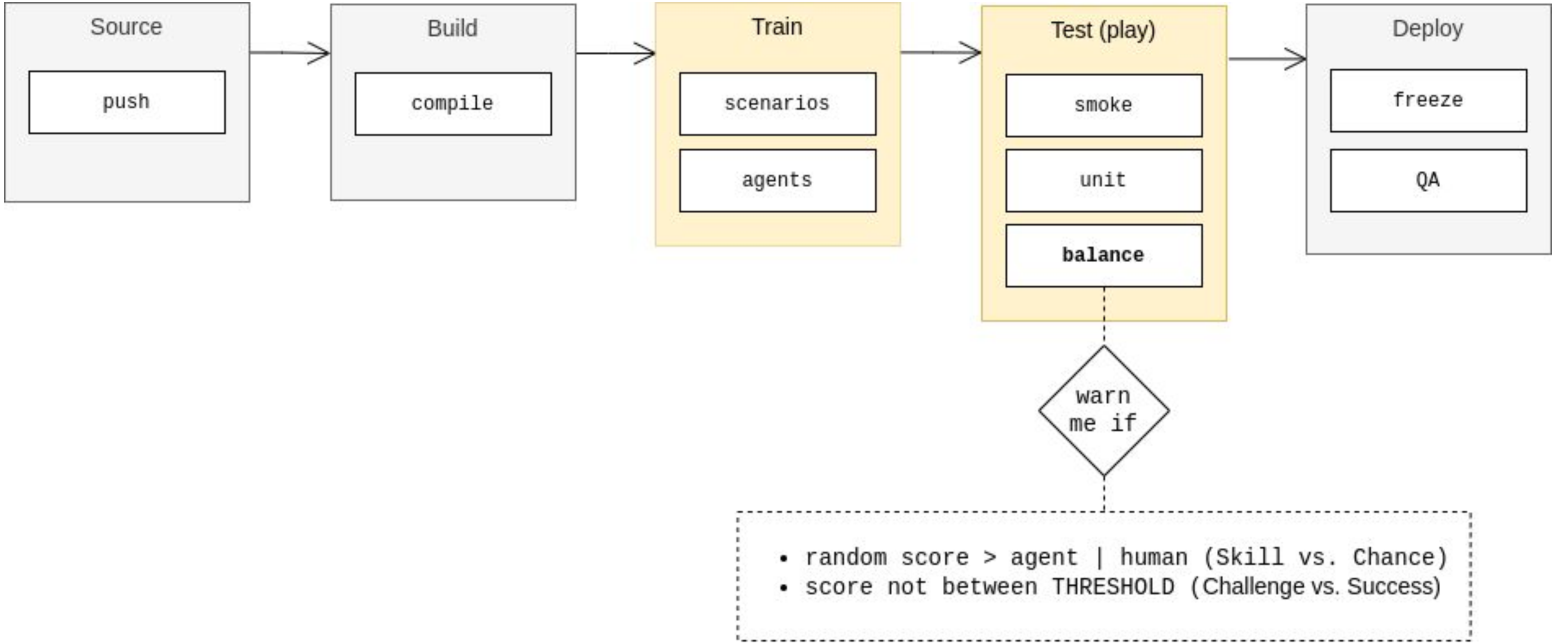


Automated Video Game Testing for Balancing: **Case B**

Build	Shift Speed	Gaps
#1	1	1
#2	2	1
#3	2	2



Automated Video Game Testing for Balancing: **C.I. Example**



- random score > agent | human (Skill vs. Chance)
- score not between THRESHOLD (Challenge vs. Success)

Automated Video Game Testing for Balancing: **Threats**

Initial effort to set the environment.

Reward functions to make agents play properly.

Observation space and relation with actions.

Definition of **done** and the **time** spent **training.**

Automated Video Game Testing for Balancing: **Conclusion**

Automate game testing to **balance** video games using **autonomous** agents.

Training the agents, **playing** the game, and **assessing** the game balance.

Validate with two platform games.

Game developers can adopt a **development pipeline** with automated testing that provides **quick feedback** about the game **balance state**.

Conclusion

Video Game Problems: **Conclusion**

Many prob

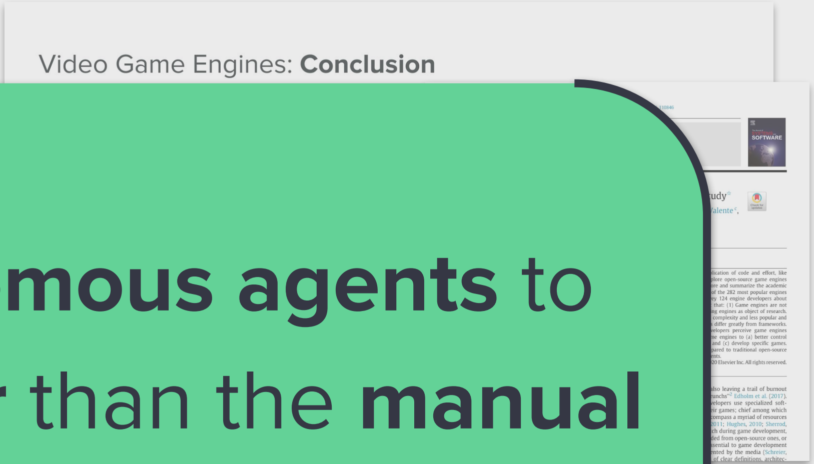
Only 5

No tr

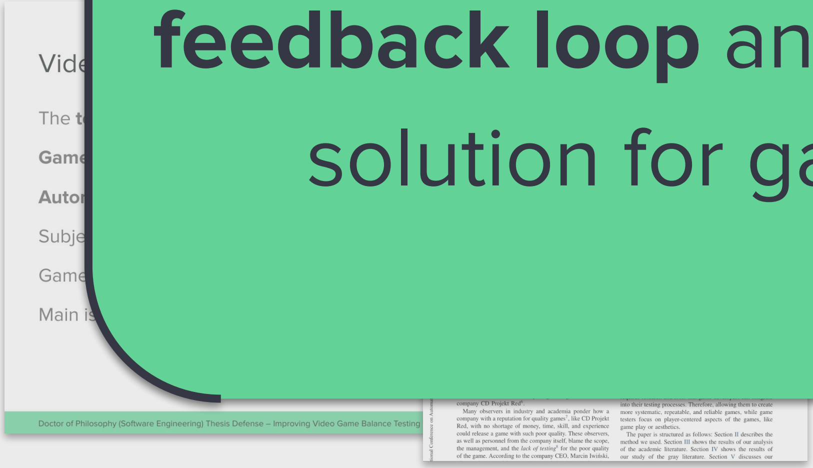
No a

Testi

Doctor of Ph



Video Game Engines: **Conclusion**



Vide

The t

Game

Auto

Subje

Game

Main i

Doctor of Philosophy (Software Engineering) Thesis Defense – Improving Video Game Balance Testing

ng Way to Go

Doctor of Philosophy (Software Engineering) Thesis Defense – Improving Video Game Balance Testing

The use of **autonomous agents** to **test** games is **faster** than the **manual feedback loop** and provides a **viable** solution for game **balancing**.

Future Work

Short Term:

- Find easier ways to train the agents.
- Add different ways for agent to play the game.
- Explore other balance problems.

Mid Term:

- Expand an existing postmortem dataset by automating the text analysis.
- Display the information on an online platform.
- Validate with a survey.

Long Term:

- Design and create a tool (plugin), so that game developers can use it from within the game engine.
- Chose one open-source game engine and study its architecture.
- Embed an AI training framework into the game engine.

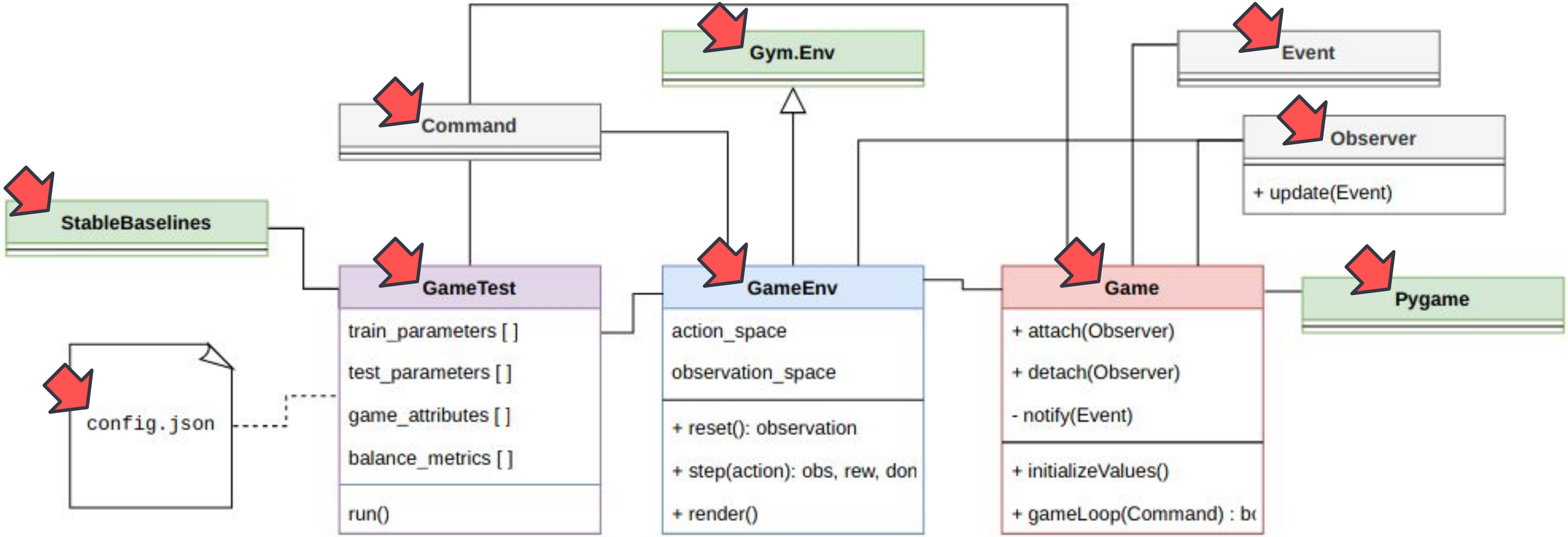
Improving
Video Game Balance Testing
Using Autonomous Agents

Cristiano Politowski, Doctor of Philosophy (Software Engineering) Thesis Defense, Concordia University, Montreal, Canada

Dr. Antonio Bucchiarone	*External Examiner*
Dr. Wahab Hamou-Lhadj	*External to Program*
Dr. Emad Shihab	*Examiner*
Dr. Juergen Rilling	*Examiner*
Dr. Bruno Lee	*Chair*
Dr. Yann-Gaël Guéhéneuc	*Thesis Supervisor*
Dr. Fabio Petrillo	*Thesis Co-Supervisor*

UNIVERSITÉ Concordia UNIVERSITY

Automated Video Game Testing for Balancing: **Architecture**



Automated Video Game Testing for Balancing: **Discussion**

Balancing the “Challenge vs. Success”

Balancing the “Skill vs. Chance”

Checking the game balancing hypothesis

Training the agents already hints for balance issues

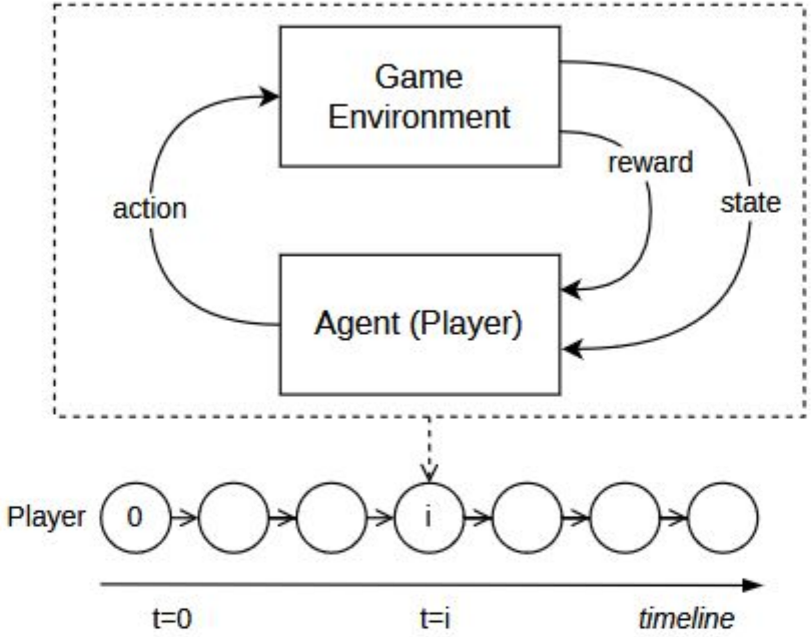
Players got tired of testing

Deterministic vs. Stochastic

Automated Video Game Testing for Balancing: **Training**

Markov Decision Process (MDP).

- The **agent**, which interacts with the **environment** using a **policy**.
- The **state**, which is the observation (representation) of the environment.
- The **action**, a set of possible decisions (move, attack, jump, etc).
- A **reward**, is the feedback we use to measure the success or failure of the agents' actions in achieving some goal (winning, surviving, etc).



Automated Video Game Testing for Balancing: **PPO & A2C**

Proximal Policy Optimization (**PPO**) is a Reinforcement Learning algorithm which aims to maximize the probability of a set of actions being taken by the agent [0]. Given these actions make the agent get rewards above average during its interaction with the environment. PPO is an on-policy algorithm, meaning it learns by comparing the current set of actions taken with the previous one, without using a replay memory.

[0] Schulman et. al., Proximal Policy Optimization Algorithms (2017)

A2C is a synchronous variant of Asynchronous Advantage Actor-Critic (A3C) [1] that uses agents running in parallel to explore different parts of the environment. The algorithm does not need to use a replay memory. Similar to PPO, after reaching a terminal state (e.g: game over) or a maximum number of actions, the algorithm updates its policy, which is the function that generates the set of actions to be taken by the agent. This update is done to make the policy more likely to generate actions that will lead to high rewards.

[1] Mnih et. al., Asynchronous Methods for Deep Reinforcement Learning (2016)

Automated Video Game Testing for Balancing: **A2C**

A2C is a synchronous variant of Asynchronous Advantage Actor-Critic (A3C) that uses agents running in parallel to explore different parts of the environment [1].

Actor: a model (neural net) which takes as input the state and outputs the action.

Critic: another model which takes as input the state and outputs the value of that state (i.e., the expected future cumulative discounted reward your agent will receive starting at that state).

The actor is trained by producing an advantage (using the critic and the "next state" as input) and using that to change the probability of producing that action given that state.

If the critic produces a high advantage, the actor will be more likely to choose that action given that state (and vice-versa).

[1] Mnih et. al., Asynchronous Methods for Deep Reinforcement Learning (2016)

Automated Video Game Testing for Balancing: **PPO**

PPO is basically a variant of A2C.

Proximal Policy Optimization (PPO) is a Reinforcement Learning algorithm which aims to maximize the probability of a set of actions being taken by the agent [0].

PPO basically clips the effect of the advantage such that an actor's action distribution for a particular state doesn't move too much during training.

With A2C, there can be issues where a particular training trajectory can significantly influence an actor's preferred action, causing it to be bad at exploration (among other things).

PPO helps fix this by preventing it from being as influenced during any particular training round as much.

[0] Schulman et. al., Proximal Policy Optimization Algorithms (2017)

Automated Video Game Testing for Balancing: **Params**

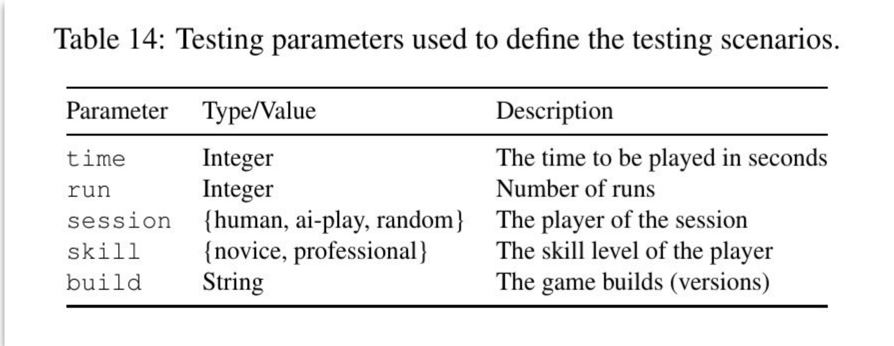


Table 14: Testing parameters used to define the testing scenarios.

Parameter	Type/Value	Description
`time`	Integer	The time to be played in seconds
`run`	Integer	Number of runs
`session`	{human, ai-play, random}	The player of the session
`skill`	{novice, professional}	The skill level of the player
`build`	String	The game builds (versions)

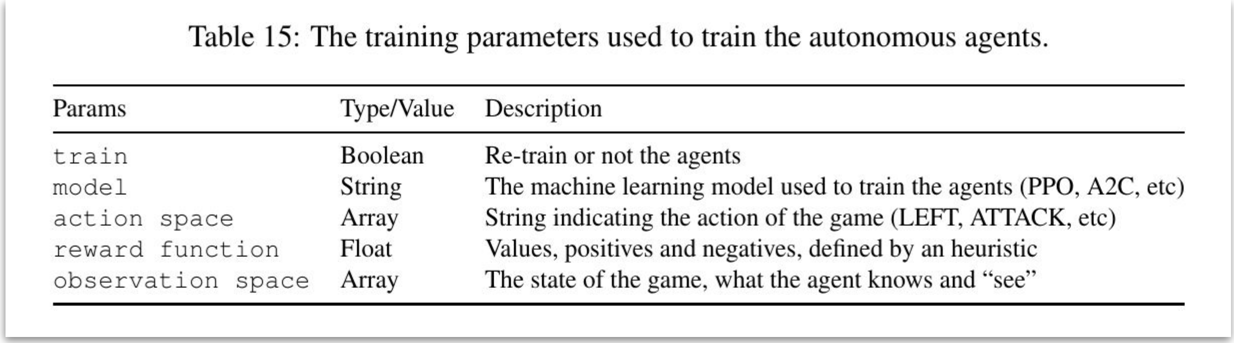


Table 15: The training parameters used to train the autonomous agents.

Params	Type/Value	Description
`train`	Boolean	Re-train or not the agents
`model`	String	The machine learning model used to train the agents (PPO, A2C, etc)
`action space`	Array	String indicating the action of the game (LEFT, ATTACK, etc)
`reward function`	Float	Values, positives and negatives, defined by an heuristic
`observation space`	Array	The state of the game, what the agent knows and “see”

Automated Video Game Testing for Balancing: **Case A**

The goal is to kill as many bats as possible without being hit.

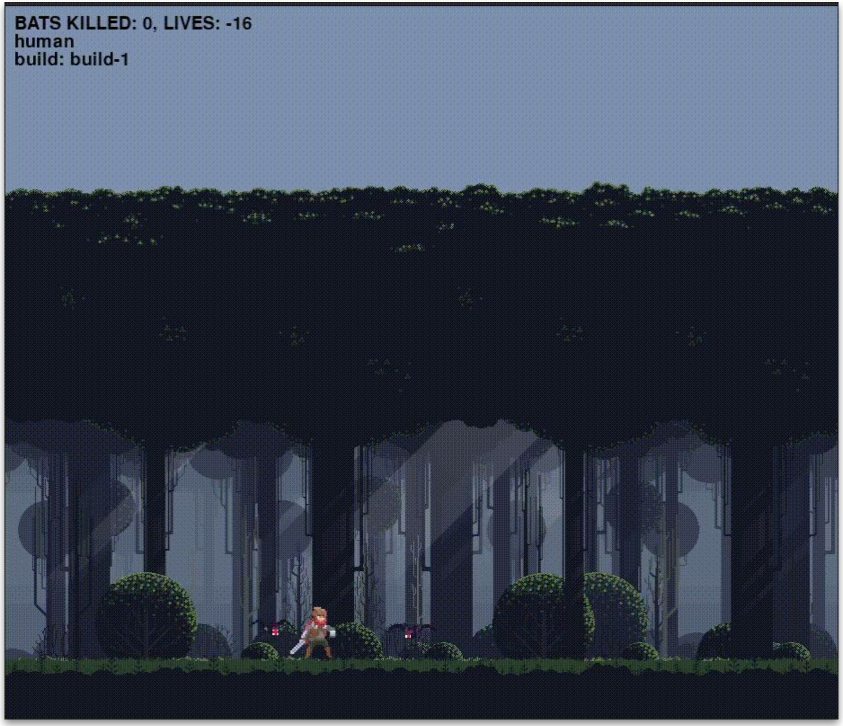
For each bat killed, the player gets one point (`+1 score`).

For each hit taken, the player loses one life (`-1 life`).

The bats spawn faster as they are killed.

The bats spawn in random locations.

The actions are: `LEFT, RIGHT, ATTACK, JUMP`



```
BATS KILLED: 0, LIVES: -16
human
build: build-1
```

Automated Video Game Testing for Balancing: **Rewards**

BAT_KILLED **+5**

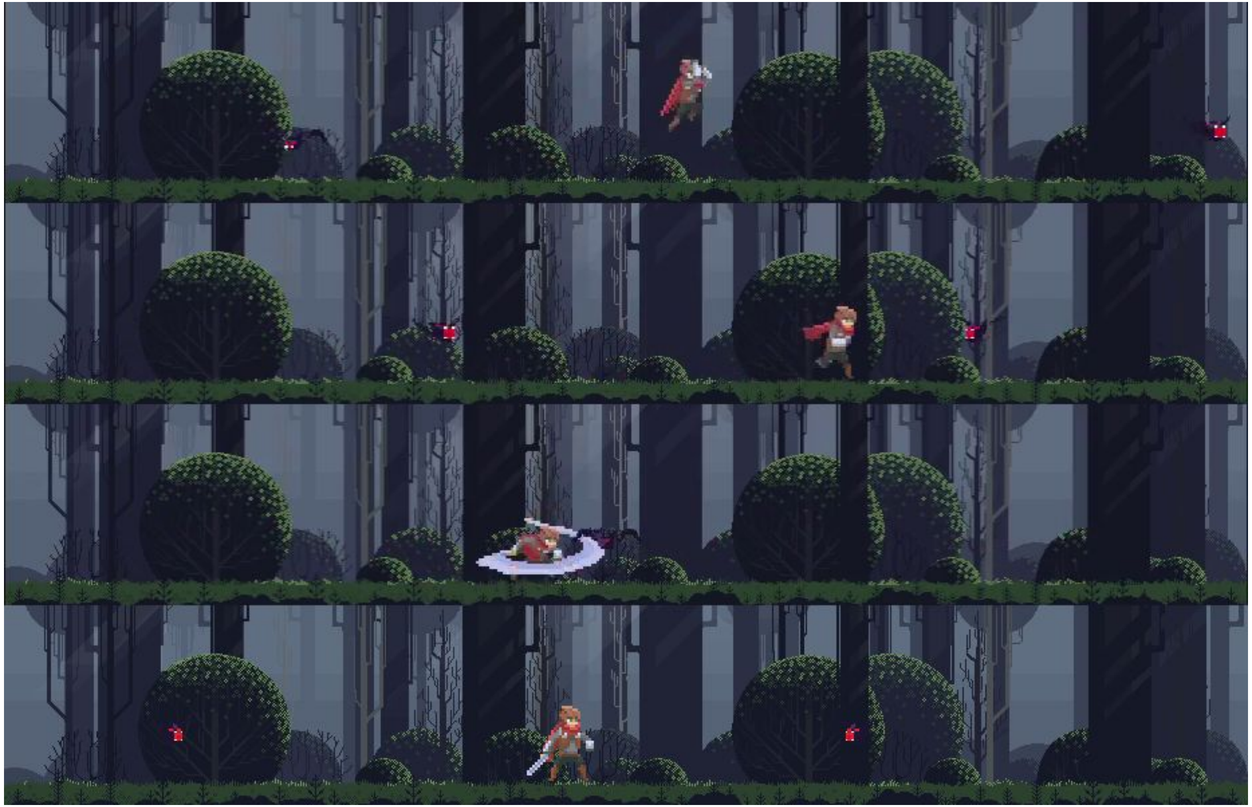
HIT_TAKEN **-5**

ATTACK **-0.1**

JUMP **-0.2**

MOVING_TOWARDS **+0.1**

FACING_NEAREST_BAT **+0.2**



Automated Video Game Testing for Balancing: **Case A**

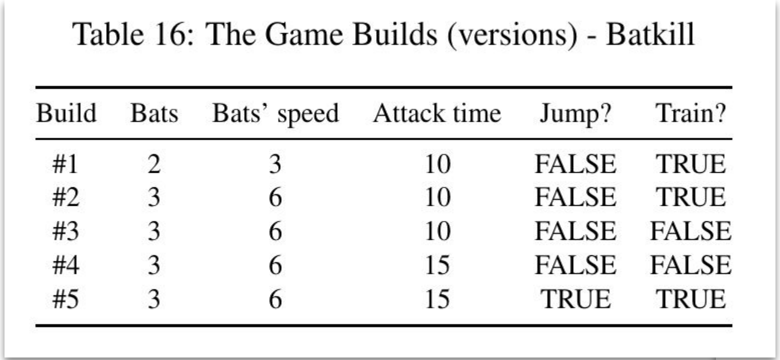
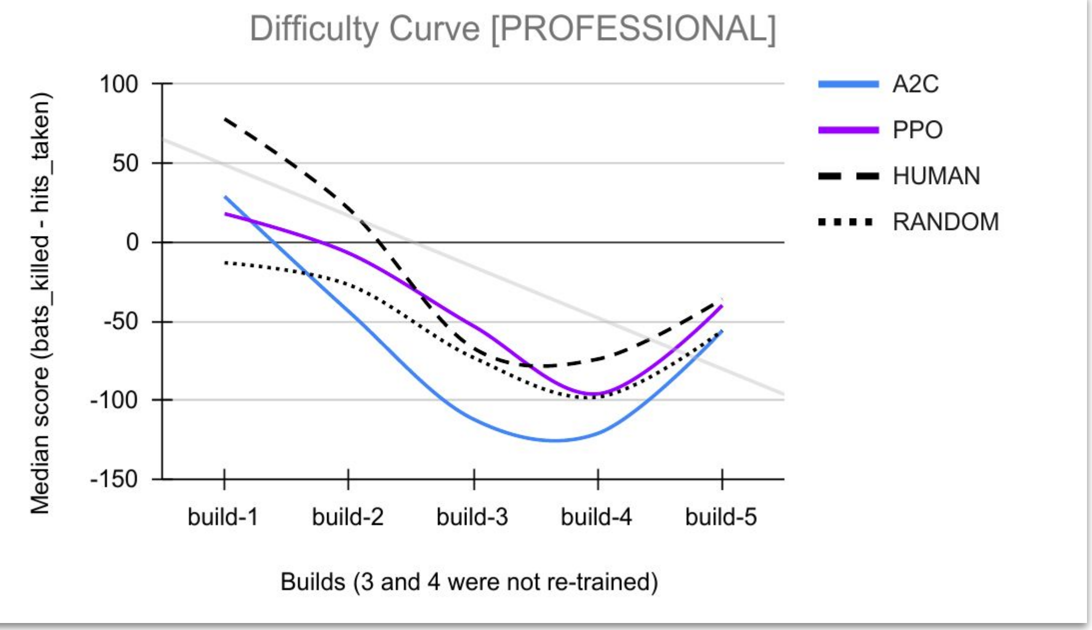


Table 16: The Game Builds (versions) - Batkill

Build	Bats	Bats' speed	Attack time	Jump?	Train?
#1	2	3	10	FALSE	TRUE
#2	3	6	10	FALSE	TRUE
#3	3	6	10	FALSE	FALSE
#4	3	6	15	FALSE	FALSE
#5	3	6	15	TRUE	TRUE



Automated Video Game Testing for Balancing: **Case A**

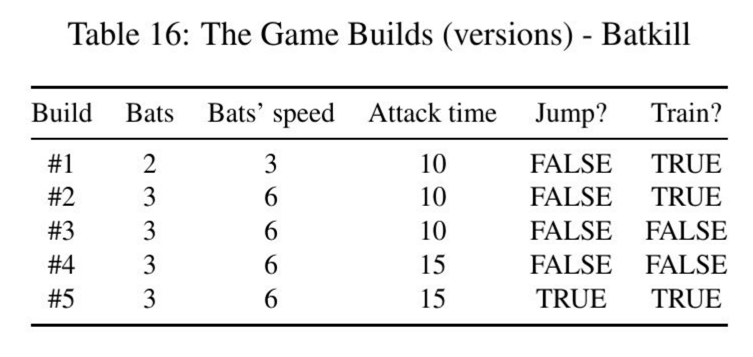
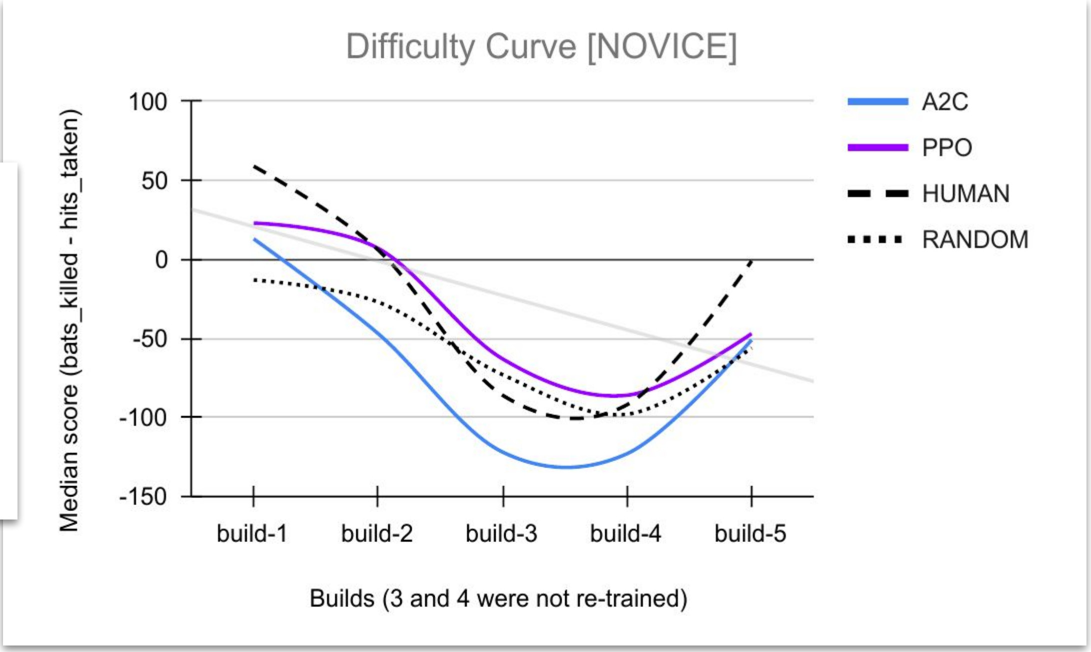
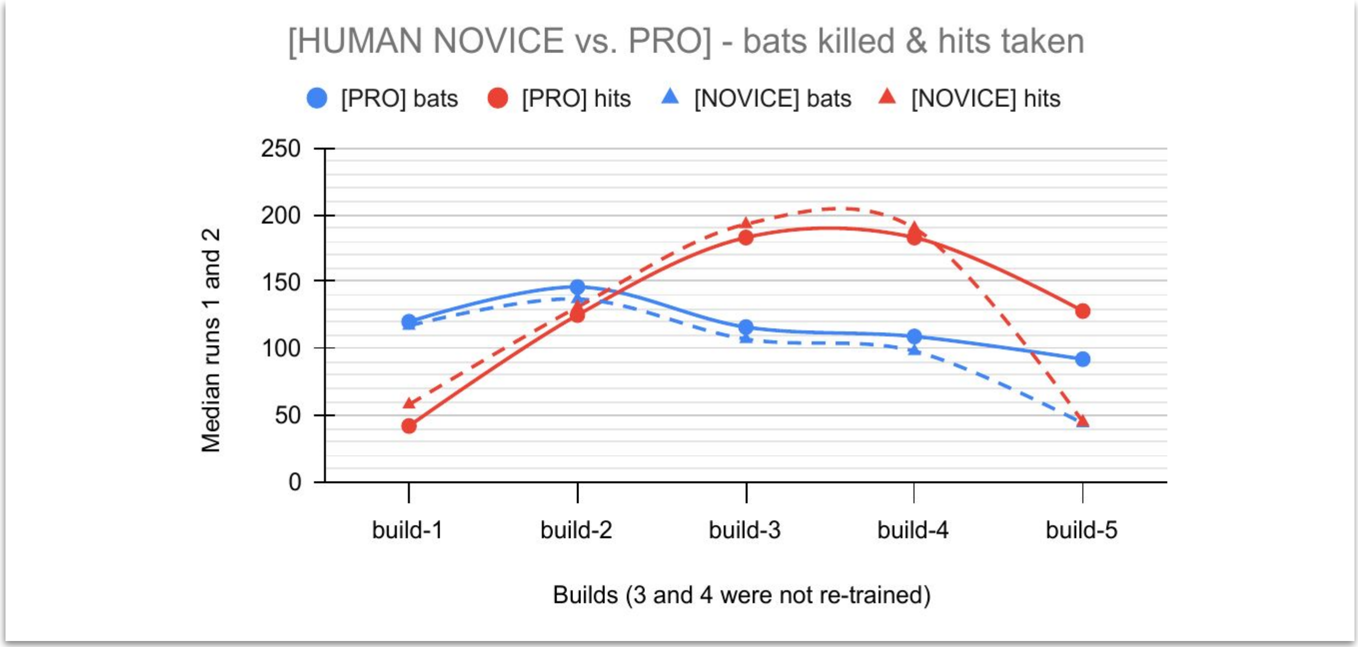


Table 16: The Game Builds (versions) - Batkill

Build	Bats	Bats' speed	Attack time	Jump?	Train?
#1	2	3	10	FALSE	TRUE
#2	3	6	10	FALSE	TRUE
#3	3	6	10	FALSE	FALSE
#4	3	6	15	FALSE	FALSE
#5	3	6	15	TRUE	TRUE



Aut. V. G. Test. for Balancing: **Case Study A - Results**



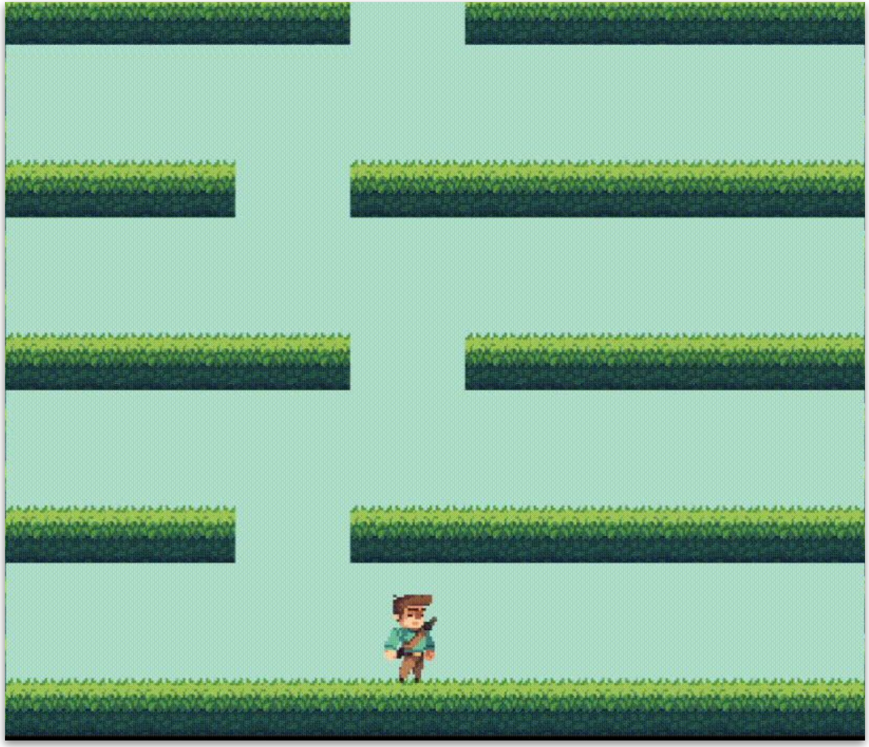
Automated Video Game Testing for Balancing: **Rewards**

We check if the character is not under the gap and, if the distance between the character and the gap is decreasing, we give +100 and -100 if increasing.

If the character is under the first gap, we check first if he is facing the second gap (in the upper row), and give +100.

If the character's position on Y-axis is decreasing, the reward is +100 + BTR, otherwise, -100 - BTR.

We give a penalty to the agent so he can get out of the first platform. Thus, if the BTR is bigger than zero and the character repeats the previous step, we give -100.



Automated Video Game Testing for Balancing: **Case B**

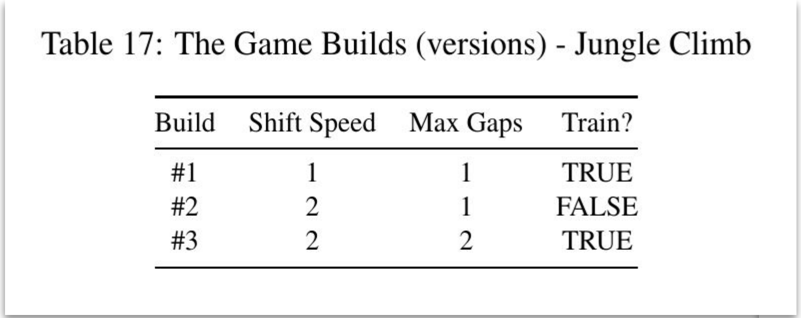
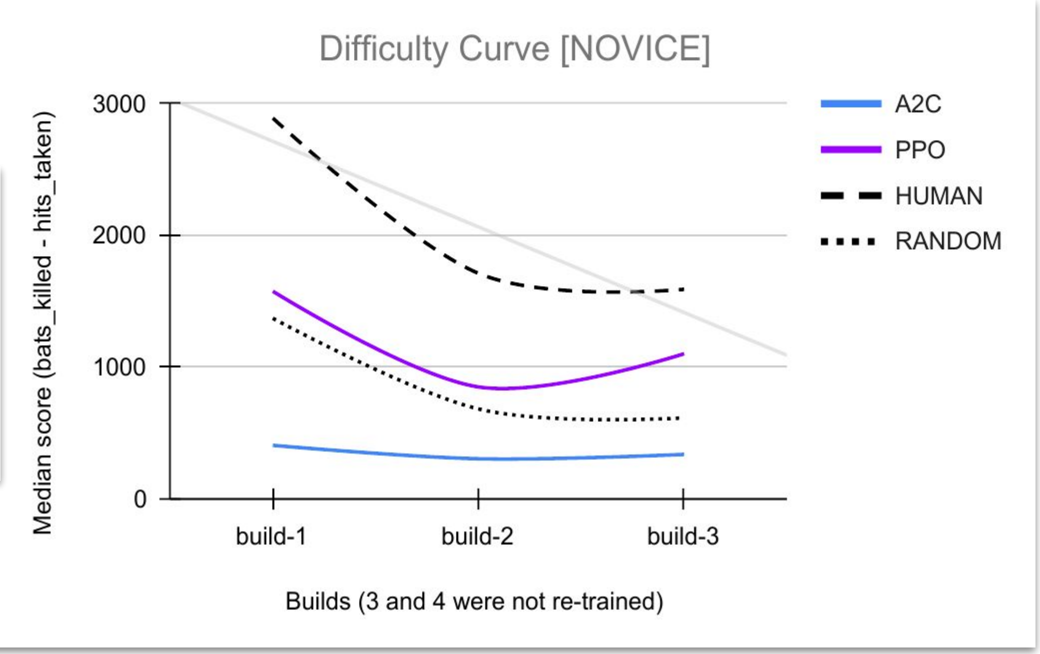


Table 17: The Game Builds (versions) - Jungle Climb

Build	Shift Speed	Max Gaps	Train?
#1	1	1	TRUE
#2	2	1	FALSE
#3	2	2	TRUE


Difficulty Curve [NOVICE]
Median score (bats_killed - hits_taken)
build-1
build-2
build-3
Builds (3 and 4 were not re-trained)
A2C
PPO
HUMAN
RANDOM