Proposing Formal Game Spaces as Foundation for Gameplay-Based Anti-Cheating Measures

Caroline Rendenbach
Technical University of Munich
Munich, Germany
caroline.rendenbach@tum.de

Daniel Dyrda
Technical University of Munich
Munich, Germany
daniel.dyrda@tum.de

2024 IEEE Conference on Games (CoG) | 979-8-3503-5067-8/24/$31.00 ©2024 IEEE | DOI: 10.1109/CoG60054.2024.10645637

***Abstract*—As the popularity of video games has increased, so has the incidence of cheating. Although researchers and game companies have developed various anti-cheating strategies, a common solution involves kernel-level implementations. However, kernel-level anti-cheating engines enjoy high privileges on a user's computer and thus raise significant privacy concerns. This paper advocates for a privacy-preserving alternative: a server-side anti-cheating engine based on formal game spaces. Our proposed solution leverages the formalization of the game space, analyzing the spatial and logical models of the video game to verify player behavior without compromising user privacy.**

***Index Terms*—Game Engineering, Space, Game Space, Anti-cheating, Server-side, Online Games, Cheat Detection**

I. Introduction

The video game industry is continuously growing, reaching $178.73 billion in revenue worldwide in 2021 [1] and is projected to reach $282.30 billion in 2024 [2]. With the revenue growing, video games and their technology have evolved substantially in the last twenty years. At the same time, cheating has become more ubiquitous since video games have shifted many functionalities online, and games often have a multiplayer aspect. This can directly impact revenue, as cheating makes players more likely to stop playing a game and less likely to make in-game purchases [3]. Additionally, as soon as cheaters use an exploit that is fixed, they often find a new way to exploit the system. Thus, there is an ongoing cat-and-mouse game between developers and cheaters, with the former trying their best to secure their game and the latter trying their hardest to break it.

II. Problem Statement

Cheating in video games is a controversial topic, as many aspects are disputed. There are many different views on what constitutes cheating, as it often depends on social norms and cultures that shape an individual's upbringing. Looking at Consalvo's three perspectives on cheating [4], we can get a clearer picture of what can be considered acceptable: for the "Purist," any outside help is cheating. For the "Code is Law" believer, anything that does not modify the game is allowed. Lastly, at the extreme, there is the "You can only cheat another player" believer, where only the intent is essential as cheating only takes place in relation to other players. Although this definition has some limitations, it is a good starting point. For this paper, we define cheating as willingly breaking the game rules to gain an advantage over other players. This definition implicitly excludes single-player games, as most of the time, the only experience an individual is ruining is their own. The definition includes well-known hacks such as wall-hacking, which is the act of using the game's memory to the advantage that walls in-game disappear for the malicious hacker, teleporting, flying, and all types of non-allowed movement, or using bots to gain an advantage, e.g., through aiming better.

Cheating, as we defined it, is often viewed to be rampant in today's multiplayer online games. However, while a significant number of people admit to cheating, even more state that they feel that cheating ruins the experience of the game for them [3]. Respondents to the study of Irdeto [3] state that they would stop playing a game if it were overrun by cheaters. This not only harms the user's experience of the game but can also impact the game's sales. Due to all these factors, game developers employ anti-cheating mechanisms. The current state-of-the-art anti-cheating measures often utilize highly intrusive methods to detect malicious behavior. For example, many anti-cheating engines employed by popular video games, such as Valorant [5] or Genshin Impact, are kernel-based [6]. As a result, the game engine processes enjoy high privileges in a user's operating system. The setup of these anti-cheat engines has implications for the user's privacy, as their data might not be protected from being read by the company. Additionally, through the engine, the user's device could be exploited by either the game company or malicious actors. A recent case of this lack of security is demonstrated by hackers who used the anti-cheat engine of Genshin Impact to gain remote access to the devices of unsuspecting players [6].

III. Goal

In the former section, we argued why a non-intrusive anti-cheating solution should be realized. By formalizing the game space, we obtain clear parameters of the game and the players' actions. With this, we can validate these to the game's valid parameters and analyze whether the player's behavior is permitted. This paper proposes a way of designing an anti-cheating engine to maintain the integrity of the game, which

will improve the players' experience in the game and establish a solution where the players do not have to reveal their personal data to enjoy their hobby. The goal of our approach is not to detect all forms of cheating–no solution will ever be able to–but the focus is on eliminating the type of cheating that deprecates the user's experience. This means that the more a user realizes that another player is cheating, the more we want to address that.

IV. Related Work

In the field of video game formalization, various methodologies have been investigated [7]–[9]. One significant approach is mathematical formalization, as explored in multiple studies [10]–[12]. Another perspective involves formalizing games through semantic analysis [13]. Additionally, the architectural perspective on game design presents a distinct approach to formalization [14], [15]. Game analytics represent another crucial methodology, particularly through the use of telemetry for spatial and spatiotemporal analysis, often formalized in map representations [16]–[19]. Furthermore, the domain of anti-cheating measures has been extensively researched, with some studies focusing on less intrusive solutions [20], [21].

V. Requirements

In order to achieve the above-mentioned goal, several requirements must be met. A proposed solution should be entirely server-side. As the focus is on online games, all information about the game between the players has to be sent through a server. This server can be used to analyze player behavior. This is done for two reasons: as it is always possible for a client to manipulate their own device, any analysis done on the client's device is not to be trusted. This is also one of the reasons why anti-cheat solutions at the kernel level are only limited since they, too, can be tampered with as they are on the client side. The other reason for this requirement is privacy: the player's information that can be accessed server-side is mostly game-related. Other metadata transmitted with the game data, such as network telemetry, i.e., IP address, time, and others, and player's account data can be collected. While much information can be gleaned from analyzing metadata, it is significantly less than what a game engine can access with kernel-level privileges on a player's device.

The former point about metadata analysis leads to the following requirement. The only data collected and logged should be game-state related. This is done to avoid misuse of the data. As our solution is a server-side solution, data collection is already limited. In order to create the greatest possible trust among users and protect their privacy, only data about the game should be collected. Another essential requirement is usability across genres, different implementations of video games, and ease of understanding. This means that any applied system should work for a range of different video games, regardless of what language or engine they were developed in, and should be at a level of complexity that is understandable and easy to use for most developers.

VI. Approach

To realize the formalization, the game space has to be defined: the game space is a set of parameters that fit together by game rules and mechanics [22]. The definitions for both are unclear and often used interchangeably; due to that, we focus on the definition established by Sicart [23], stating that "game mechanics are methods invoked by agents, designed for interaction with the game state" [23]. The methods can be described by verbs like *move, stab, jump, shoot*, and the agents can be players, meaning that they can be controlled by a human but also by a computer. The game rules often limit the methods an agent can invoke. Järvinen [24] defines game rules as "a particular set of rules available to the player in the form of prescribed causal relations between game elements and their consequence to particular game state(s)" [24, p.254]. This can be a simple rule like *there is gravity* or something more complex like *if a player shoots at another player, hits nothing in between, is not too far away, and hits the other player, the other player will take a certain amount of damage depending on where they were hit and what they were hit with*.

Since there are so many ways of cheating in video games, an anti-cheat engine must check against all the possibilities. The vast majority of cheating involves breaking the rules of one of the two models: the logical model and the spatial model. The logical model is the player's behavior, described through the methods mentioned earlier. The spatial model is the space in which the player moves. Both models can influence each other as specific methods can only be invoked in certain areas, or particular areas are only available after a specific method is performed. [22]

Violations of the spatial model through cheating can be seen as levitating, teleporting, moving through objects that limit the space, and more. At first, we focus on verifying the spatial model regarding cheating. At every moment during a video game, the player has spatial information attached to them. At the very least, the coordinates of the player's location are available. Information can also be obtained about other objects or agents in the game space and their speed of movement [16]. This information is also known as the telemetry of the player and can also include what methods agents take and the temporal information associated with them. The latter can also lead to spatio-temporal analysis. A spatial analysis does not aim to be an all-encompassing solution to verify all player behavior; it aims to be one part of a bigger toolbox. With this in mind, we use the paper "*Space Foundation System: An Approach to Spatial Problems in Games*" [25] as our foundation to formalize the game space. Through this, we partition the game space into subspaces using delimiters and anchors and realize a state graph. Players are mapped in their subspace in relation to the anchor. When crossing delimiters, real-time checks are made if that movement is allowed based on the information attached to the delimiter. Nevertheless, not only the player's movement can be validated, but all spatial movement that is related to the player, such as the movement of a weapon to shoot and the movement of the projectile towards the target.

Here, we can validate if a projectile permeates a delimiter that it should not be able to and if the projectile should be able to move that way. This type of banned movement can be more specific from game to game. Therefore, it will not be the main focus of the approach presented.

After we have verified that the player is not participating in a clearly forbidden movement, we can extend the spatial model so that it is combined with the logical aspect of the player's behavior. This is still a spatial analysis at first. However, with the inclusion of the logical model, we are looking for unusual behavior that is not as clear-cut as illegal behavior at first. To do this, we go beyond just using basic spatial information by analyzing the spatio-temporal trajectories of the players. Here, we can validate the use of game bots: since human gameplay is often erratic and irregular, bot-led movement may be easier to detect, as bots often strive for efficiency by taking the shortest path from A to B, whereas a human player would regularly take cover and look around for other players [26]. In general, bot-driven movement can also be described in terms of straight and long paths [26], although this cannot be said for all bots. Game bots have started to mimic human behavior to avoid suspicion. However, due to their unpredictable nature, this is still challenging to incorporate, as players take time to think about their next move, change goals, or even engage in behavior that might negatively affect them solely to annoy other players. In contrast, bots usually have clear goals and must use specific methods to achieve them [26]. Although what these behaviors look like can vary from game to game, it can generally be said that bots, in stark contrast to human players, have self-similarity, meaning that they regularly follow the same paths, using the same methods over and over again [27]. Here, we can employ our formalization that the player is seen in relation to the anchor, the object of influence in that subspace. Estimating cheating can be made through trajectory analysis towards and around the anchor.

As stated earlier, the focus is on multiplayer games. To solve the problem of multiple players, we utilize another tool from game analysis, which originated with Geographical Information Systems: *overlapping*. Every player gets their formalization of the game space. Overlapping is when these maps are then superimposed over each other. Through this, we can achieve composite maps by combining large information sets [28]. As Drachen [16] suggests, we can synthesize or analyze the maps in relation to each other and thus find spatial information of the players towards or around each other. The same trajectory analysis that was proposed before can be done on the information gained from overlapping as the movement in ratio to other players is different than if a player is alone in the space, e.g., hiding behind game objects or changing the course of trajectory to avoid confrontation.

Movement does not have to be the only indicator of cheating: other unusual or outlying behavior can indicate cheating. Thus, verification of the logical model could detect bots or other cheating methods through general behavioral analysis. This would work on top of the existing solution by mining the player's behavior for unusual actions. For example, if a player uses an aimbot, the player might aim quickly and clearly, whereas an average player might take a long time to aim and readjust frequently. Although this unusual behavior will be more specific to every game, some generalized statements can be made for games from a similar genre, for example, first-person shooter games, as they utilize game mechanics that are alike. To distinguish between average user behavior and cheating, we would have to mine enough data, and as is established in other fields, we could employ the use of Machine Learning (ML) to discern outlier behavior. Not only do other disciplines employ ML, but the use of it is also currently being proposed in general for video games and as an anti-cheating method. Pfau [29] compares the use of *Hidden Markov Models* (HMMs), *decision trees* (DTs), and *deep learning* (DL) to extract behavior models from gameplay. Though they primarily aim to analyze decision strategies, they can also be used to find outlier behavior, as they argue that one of their potential applications could be online crime detection, such as cheating and bots. The research found that for behavior prediction of decisions and with our use case in mind, HMMs and DTs were more applicable. Other research into the use of ML for anti-cheat detection also suggests the use of decision trees but also compares it to *Support Vector Machines* and *Naïve Bayes classifiers* [30].

VII. Discussion

Though our proposal is a good starting point, how all parts of it are implemented, especially working together, has to be explored. Moreover, our approach is limited in that it mostly relies on spatial information and is only one tool in a toolbox that an anti-cheat engine should provide. However, combining our approach with other already established server-side solutions in the anti-cheating field is possible. With this in mind, our approach shows great promise in solving the problems of anti-cheating measures that are not privacy and trust-centered. It also has the advantage that developers must clearly understand their game mechanics and rules to assess when they are broken. This could lead to an improved development process. We envision that the proposed system can also be used in the testing phase to find and eliminate bugs and unintended behavior due to its nature, which is also heavily based on game analysis tools. For example, a developer might discover that many players die at a certain point in the game space, possibly due to a non-optimal design and leading to dissatisfaction.

Although this solution tries to cover significant amounts of possible cheating, it is still essential to also implement industry-standard IT-security solutions such as secure encryption methods for all communication between client and server to secure the system from other attack vectors [31]. This is not just important with cheating in mind but also for protecting users and the game from leaking sensitive data or other malicious attacks. When exploring the option of using ML, we also have to keep in mind that it does not interfere with our requirement of being privacy-conscious. An excellent

place to start when assessing an implemented system could be the EU AI Act [32]. Moreover, the use of ML can be computationally heavy. Thus, its use has to be assessed with its pros and cons. Another critical aspect to consider is that even though most systems strive to be the most accurate in their predictions and rulings, no system is entirely perfect, and actions such as banning a player from the game should not be taken in an instant but have to be seen in the context of all their player data. This means that before a ruling takes place, some factors can be taken into consideration: if it was the first offense, how likely it is cheating or just good gameplay, how severe the offense is, and how the gameplay compares to other players' data. Based on that, the severity of the action taken as a consequence can also be argued. Certain games, such as Counter-Strike: Global Offensive, use humans to assess if cheating occurred. Senior players take a look at the reported players' gameplay and make their decisions based on that [31].

VIII. Conclusion

In this paper, we propose a server-side, non-intrusive approach for anti-cheating engines based on the formalization of the game space by constructing a graph by defining delimiters and anchors, and then verifying a player's movement through our spatial model. On top of this, we propose tools from the game analytic space to analyze players' behavior in both the spatial and logical models. Further, we envision using Machine Learning to accurately predict and assess users' gameplay. This is done to also reflect state-of-the-art research within the anti-cheating space. The next steps are to implement the proposal, test out what works best with our goals in mind, and evaluate the accuracy of our system.

References

[1] A. Kirkcaldy, "Video game industry statistics, trends and data in 2023," 2023. [Online]. Available: https://www.wepc.com/news/video-game-statistics/

[2] "Video games - worldwide," 2023. [Online]. Available: https://www.statista.com/outlook/dmo/digital-media/video-games/worldwide

[3] "Global gaming survey," 2018. [Online]. Available: https://resources.irdeto.com/irdeto-global-gaming-survey/irdeto-global-gaming-survey-report-2

[4] M. Consalvo, *Cheating: Gaining advantage in videogames*. MIT Press, 2009.

[5] T. Wilde, "The controversy over riot's vanguard anti-cheat software, explained," 2020. [Online]. Available: https://www.pcgamer.com/the-controversy-over-riots-vanguard-anti-cheat-software-explained/

[6] R. Soliven, "Ransomware actor abuses genshin impact anti-cheat driver to kill antivirus," 2022. [Online]. Available: https://www.trendmicro.com/en_us/research/22/h/ransomware-actor-abuses-genshin-impact-anti-cheat-driver-to-kill-antivirus.html

[7] R. O. D. Kelder, "Towards a framework for analytical game space design," Master's thesis, Universiteit Utrecht, Faculty of Computer Science, n.d.

[8] K. Salen and E. Zimmerman, *Rules of play: Game design fundamentals*. MIT Press, 2004.

[9] R. van Rozen, "Languages of games and play: A systematic mapping study," *ACM Comput. Surv.*, vol. 53, no. 6, 2021.

[10] S. M. Grünvogel, "Formal models and game design," *Game Studies*, vol. 5, no. 1, pp. 1–9, 2005.

[11] M. Nitsche, "Mapping time in video games." in *DiGRA Conference*, 2007.

[12] M. Cook and A. Raad, "Hyperstate space graphs for automated game analysis," in *2019 IEEE Conference on Games (CoG)*, 2019, pp. 1–8.

[13] T. Tutenel, R. Bidarra, R. M. Smelik, and K. J. D. Kraker, "The role of semantics in games and simulations," in *Computers in Entertainment (CIE)*, vol. 6, no. 4. ACM New York, 2008, pp. 1–35.

[14] D. Arribas-Bel and M. Fleischmann, "Spatial signatures - understanding (urban) spaces through form and function," *Habitat International*, vol. 128, 102641, 2022.

[15] S. Haq, "Where we walk is what we see: Foundational concepts and analytical techniques of space syntax," in *HERD: Health Environments Research & Design Journal*, vol. 12, no. 1, 2019, pp. 11–25.

[16] A. Drachen and M. Schubert, "Spatial game analytics and visualization," in *2013 IEEE Conference on Computational Inteligence in Games (CIG)*. IEEE, 2013, pp. 1–8.

[17] A. Drachen and A. Canossa, "Analyzing spatial user behavior in computer games using geographic information systems," in *Proceedings of the 13th international MindTrek conference: Everyday life in the ubiquitous era*, 2009, pp. 182–189.

[18] S. J. Kang and S. K. Kim, "Automated spatio-temporal analysis techniques for game environment," in *Multimedia Tools and Applications*, vol. 74. Springer, 2015, pp. 6323–6329.

[19] S. S. Maram, J. Pfau, J. Villareale, Z. Teng, J. Zhu, and M. S. El-Nasr, "Mining player behavior patterns from domain-based spatial abstraction in games," in *2023 IEEE Conference on Games (CoG)*. IEEE, 2023, pp. 1–8.

[20] A. Maario, V. K. Shukla, A. Ambikapathy, and P. Sharma, "Redefining the risks of kernel-level anti-cheat in online gaming," in *2021 8th International Conference on Signal Processing and Integrated Networks (SPIN)*. IEEE, 2021, pp. 676–680.

[21] J. N. Silva, "Towards automated server-side video game cheat detection," 2022.

[22] A. Isaksen, D. Gopstein, and A. Nealen, "Exploring game space using survival analysis." in *FDG*, 2015.

[23] M. Sicart, "Defining game mechanics," in *Game studies*, vol. 8, no. 2, 2008, pp. 1–14.

[24] A. Järvinen, *Games without frontiers: Theories and methods for game studies and design*. Tampere University Press, 2008.

[25] D. Dyrda and C. Belloni, "Space foundation system: An approach to spatial problems in games," in *2024 IEEE Conference on Games (CoG)*. IEEE, 2024.

[26] H.-K. Pao, K.-T. Chen, and H.-C. Chang, "Game bot detection via avatar trajectory analysis," in *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 3. IEEE, 2010, pp. 162–175.

[27] A. R. Kang, S. H. Jeong, A. Mohaisen, and H. K. Kim, "Multimodal game bot detection using user behavioral characteristics," in *SpringerPlus*, vol. 5. Springer, 2016, pp. 1–19.

[28] M. N. DeMers, *Fundamentals of Geographic Information Systems*. John Wiley & Sons, 2008.

[29] J. Pfau, J. D. Smeddinck, and R. Malaka, "Towards deep player behavior models in mmorpgs," in *Proceedings of the 2018 Annual Symposium on Computer-Human Interaction in Play*. Association for Computing Machinery, 2018, pp. 381–392.

[30] R. Spijkerman and E. Marie Ehlers, "Cheat detection in a multiplayer first-person shooter using artificial intelligence tools," in *Proceedings of the 2020 3rd International Conference on Computational Intelligence and Intelligent Systems*. Association for Computing Machinery, 2021, pp. 87–92.

[31] Q. Zhang, "Improvement of online game anti-cheat system based on deep learning," in *2021 2nd International Conference on Information Science and Education (ICISE-IE)*, 2021, pp. 652–655.

[32] "Eu artificial intelligence act," 2024. [Online]. Available: https://artificialintelligenceact.eu/de/das-gesetz/