Etude des courbes de difficulté dans les jeux vidéo avec forte variabilité du niveau des joueurs et une faible quantité de donnée

William Rao Fernandes

HESAM UNIVERSITÉ

le cnam

ÉCOLE DOCTORALE EDITE

Le CNAM - CEDRIC

THÈSE

présentée par : **William RAO FERNANDES**

soutenue le : **05 Juillet 2022**

pour obtenir le grade de : **Docteur d'HESAM Université**

préparée au : **Conservatoire national des arts et métiers**

Discipline : **Section 27**

Spécialité : **Informatique**

Équilibrage et étude des courbes de difficulté dans les jeux vidéo avec forte variabilité du niveau des joueurs et faible quantité de données

THÈSE dirigée par :

Mr CUBAUD Pierre-Henri PR1, Cnam

et co-encadrée par :

Mr LEVIEUX Guillaume MCF, Cnam

Jury		
Mme COUTURE Nadine	Professeure, ESTIA, ESTIA-Recherche	Présidente du Jury
M. CUBAUD Pierre-Henri	Professeur, CNAM, CeDRIC	Directeur de Thèse
M. SEHABA Karim	MCF, Université Lumière, SICAL	Rapporteur
M. CHAMPAGNAT Ronan	MCF, La Rochelle Université, L3i	Rapporteur

THÈSE

Affidavit

Je soussigné William RAO FERNANDES, déclare par la présente que le travail présenté dans ce manuscrit est mon propre travail, réalisé sous la direction scientifique de Pierre-Henri CUBAUD et de Guillaume LEVIEUX, dans le respect des principes d'honnêteté, d'intégrité et de responsabilité inhérents à la mission de recherche. Les travaux de recherche et la rédaction de ce manuscrit ont été réalisés dans le respect de la charte nationale de déontologie des métiers de la recherche. Ce travail n'a pas été précédemment soumis en France ou à l'étranger dans une version identique ou similaire à un organisme examinateur.

Fait à Paris, le 23/09/2022
Signature

Affidavit

I, undersigned, William RAO DERNANDES, hereby declare that the work presented in this manuscript is my own work, carried out under the scientific direction of Pierre-Henri CUBAUD and of Guillaume LEVIEUX, in accordance with the principles of honesty, integrity and responsibility inherent to the research mission. The research work and the writing of this manuscript have been carried out in compliance with the French charter for Research Integrity. This work has not been submitted previously either in France or abroad in the same or in a similar version to any other examination body.

Place Paris, date 23/09/2022
Signature

Remerciements

Je tiens dans un premier temps à remercier tout ceux qui m'ont soutenu et épaulé tout au long de ce travail de thèse. Pour commencer je remercie l'ensemble des membres du CEDRIC avec qui j'ai eu la chance de collaborer pendant ces trois années. Tout particulièrement mon encandrant Guillaume Levieux pour sa bienveillance, ses conseils tout au long de mes travaux et sa confiance. Il m'a poussé a perdre mes habitudes scolaires afin de développer mon sens critique et scientifique. J'ai eu l'occasion grâce à lui de présenter mes travaux lors d'une conférence internationale au Pérou, ce qui a été une aventure extraordinaire que je suis chanceux d'avoir pu vivre.

Je remercie également mon directeur de thèse, Pierre Cubaud. Bien que moins présent que Guillaume, il a toujours su être là lorsque j'en avais besoin.

Merci également à mes compagnons thésards, Thomas et Raphael, avec qui nous avons partagé tout autant de discussions sérieuses autour d'un café que de parties endiablées de Smash Bros.

Je remercie également ma famille, qui a toujours été là pour moi et qui m'a constamment poussé à donner le meilleur de moi-même. Je remercie bien évidemment ma copine, Claire, pour m'avoir soutenu pendant toute la durée de ma thèse.

Résumé

Ce travail de recherche porte sur l'étude des courbes de difficulté dans les jeux vidéo ainsi que sur l'adaptation dynamique de la difficulté des jeux vidéo avec forte variabilité du niveau des joueurs et une faible quantité de données. Nous nous intéressons à l'adaptation de la difficulté, car la difficulté des jeux vidéo est un facteur de plaisir et de motivation.

Après une première phase consacrée à la recherche bibliographique, nous proposons un modèle d'adaptation de la difficulté. Ce modèle, développé dans le cadre du projet DysApp, répond à trois critères indispensables pour le projet. Nous avons développé un modèle générique, utilisable dans n'importe quel jeu. De plus, notre modèle permet de cibler n'importe quel niveau de difficulté. Enfin, notre modèle fonctionne sans données au préalable.

Ce modèle a été testé lors d'une expérimentation scientifique. Nous avons utilisé un jeu de type shooter en vue du dessus développé par Unity. Nous avons modifié ce jeu afin de faire jouer un joueur contre des ennemis contrôlé par une intelligence artificielle. Le jeu se joue par round et à chaque round le joueur doit venir à bout d'un ou deux tanks ennemis, le nombre de tanks à affronter dépend de la difficulté. À chaque round, le joueur gagne s'il parvient à détruire les tanks ennemis et perd s'il se fait détruire. Les résultats concluant de cette expérimentation nous ont conforté dans l'intégration de notre modèle à l'application DysApp.

Toutes les activités de l'application DysApp utilisent notre modèle afin d'ajuster la difficulté en temps réel. Nous avons ajouté à notre modèle un choix aléatoire parmi quatre courbes de difficulté que nous avons désignées. L'application DysApp a été déployée en école afin de récolter un grand nombre de données sur son utilisation, et donc sur l'utilisation du modèle. Cependant, les versions du logiciel déployées en école avaient du retard par rapport aux versions

sur lesquelles nous travaillions, ce qui à fortement réduit le nombre de données utilisable. Tralalère nous a également invités au salon de jeux vidéo Paris Games Week 2018 afin de présenter l'application sur un stand dédié. Là encore une grande quantité de données a pu être récoltée, mais les données étant beaucoup trop disparate nous n'avons pas pu les exploiter. Nous visions en effet pour cette expérience des enfants du troisième cycle étant donné que les mini jeux de DysApp s'adresse à cette tranche d'âge. Les participants plus âgés se sont souvent ennuyé sur le jeu, n'étaient pas forcément concentrés, et cela se ressent grandement sur les données de jeu.

Nous avons ainsi décidé de mener notre propre expérimentation, sans passer par DysApp, afin de continuer nos travaux de recherche sur les courbes de difficulté. Dans un premier temps ce choix nous permet de cibler un public précis, et donc de choisir un jeu adapté à ce public. De plus, nous estimons que tester des courbes de difficulté sur un jeu plus complexe que les minis-jeux DysApp apportera des résultats plus concluants. Pour ce faire, nous avons utilisé un jeu développé par Unity, dans lequel le joueur doit se battre contre des ennemis contrôlés par une IA. Nous avons modifié la carte du jeu afin de créer une arène fermée. Nous avons également modifié l'IA des ennemis pour qu'ils se déplacent sur la carte à la recherche du joueur et l'attaquent à vue. Cette expérimentation a eu lieu à l'école nationale du jeu et des médias interactifs numériques du Cnam. Cette expérimentation montre l'importance des pics de difficulté dans les jeux vidéo pour le public ciblé, et ce, peu importe la difficulté moyenne de la courbe.

Tous ces travaux ont fait l'objet de deux articles scientifique, un publié et présenté lors de la conférence *International Conference on Entertainment Computing* à Arequipa en 2019, l'autre en cours d'écriture. Ces travaux peuvent être poursuivis de plusieurs façons possibles. Il est possible par exemple de répliquer notre deuxième expérience sur une population différente, afin d'avoir une meilleure vision des besoins de chaque catégorie de joueur. Une autre direction à prendre serait de tester notre modèle de manière plus avancée en utilisant d'autre type de jeux vidéo. Ces travaux peuvent aboutir à une meilleure connaissance des besoins de chaque catégorie de joueurs grâce à un modèle simple et utilisable par une grande variété de jeux vidéo différents.

RESUME

Mots-clés : Jeu Vidéo, Difficulté, Courbe de Difficulté, Modèle de Difficulté, Adaptation de la Difficulté, Motivation, Modélisation de Joueurs.

Abstract

This research work focuses on the study of difficulty curves in video games as well as on dynamic adaptation of video games' difficulty with high variability in the players' level and a small amount of data. We are interested in the difficulty's adaptation because video games' difficulty is a factor of pleasure and motivation.

After a first phase devoted to bibliographic research, we developed a model for difficulty adaptation. This model, developed as part of the DysApp project, meets three essential criteria for the project. We have developed a generic model that can be used in any game. In addition, our model can target any level of difficulty. Finally, our model works without prior data.

This model has been tested in a scientific experiment. We used a top view shooter game developed by Unity. We have modified this game to have a player play against enemies controlled by artificial intelligence. The the game's goal is to play against waves of enemies. Enemies spawn either alone or in pairs. In each wave, the player wins if he manages to destroy the enemies and loses if he gets destroyed.

All activities in the DysApp application use our model in order to adjust the difficulty in real time. We have added to our model a random choice among four difficulty curves that we have designed. The DysApp application was deployed in schools in order to collect a large amount of data on its use, and therefore on the use of the model. However, due to problems encountered, this data could not be used. Tralalère also invited us to the Paris Games Week 2018 video game show to present the application on a dedicated stand. Here again a large amount of data could be collected but the data recovery system did not function as expected, and a large amount of the data collected was unusable.

We therefore decided to carry out our own experimentation, without going through DysApp, in order to continue our research on the difficulty curves. To do this, we have picked up a game developed by Unity, in which the player has to fight against enemies controlled by an AI. We have changed the game map to create a closed arena. We've also changed the AI of enemies to move around the map looking for the player and attack them on sight. Each round is the same. The player and the enemy appear in the arena. If the enemy is destroyed, a new enemy appears and the player recovers their health and ammo. In the event that the player is defeated, they reappear in the arena, and the enemy is destroyed to make way for a new enemy.

All this work was the subject of two scientific articles, one published and presented at the *International Conference on Entertainment Computing* conference in Arequipa in 2019, the other is in writing. This work can be continued in several possible ways. It is possible for example to replicate our second experience on a different population, in order to have a better vision of the needs of each category of player. Another direction to take would be to test our model in a more advanced way using other types of video games. This work can lead to a better understanding of the needs of each category of players thanks to a simple model that can be used by a wide variety of different video games.

Keywords : Video Game, Difficulty, Difficulty Curve, Difficulty Model, Difficulty Adaptation, Motivation, Player Modeling.

Table des matières

Liste des tableaux

Table des figures

Introduction

Cette thèse porte sur la difficulté dans les jeux vidéo. Plus précisément, l'adaptation de la difficulté ainsi que la perception de celle-ci par les joueurs. Ce sujet est depuis quelques années très discuté par les acteurs du jeu vidéo, que ce soit dans le domaine de la recherche ou du point de vu commercial. Par ailleurs, de nombreux chercheurs ont développé des modèles d'adaptation dynamique de la difficulté afin de proposer aux joueurs une expérience immersive, en accord avec leurs capacités. Contrairement aux anciens jeux vidéo proposant des choix de difficultés fixées en amont par le développeur du jeu, la technologie actuelle permet au développeur d'adapter la difficulté de leur jeu en fonction des compétences du joueur, ce qui permet d'ouvrir le Jeu Vidéo à une population de plus en plus grande. Entre 2009 et 2017, plus de 80 recherches ont été conduites concernant l'adaptation dynamique de la difficulté [1]. La majorité de ces recherches utilisent des technologies nécessitant un grand nombre de données, telles que les réseaux de neurones[2, 3, 4, 5, 6] ou l'apprentissage automatique [7, 8, 9, 10] pour en citer quelques-unes. Des méthodes plus simples également été utilisées telles que l'algorithme que nous appellerons ici le +/-δ [11, 12] qui augmente la difficulté quand le joueur gagne et la baisse quand il perd. Cet algorithme amène à un état plus ou moins *équilibré* où le joueur a une probabilité de gagner de 50%. Comme nous le verrons, l'état de l'art manque de méthodes de niveau intermédiaire et cette thèse propose de contribuer à combler cette faille.

Cette thèse est financée par le projet DysApp qui a pour but le développement d'une application tablette afin d'aider à détecter la dyspraxie chez les enfants [13]. La dyspraxie est un trouble entraînant des difficultés à effectuer certains gestes, à suivre un rythme, à planifier dans l'espace. Ces troubles entraînent très régulièrement des difficultés scolaires. L'intérêt de l'application est de permettre aux enfants de jouer a des mini-jeux nécessitant de faire des tâches qu'un enfant dyspraxique aura du mal à faire, dans le but de détecter le plus tôt possible la dyspraxie. Dans un souci de motivation à jouer et d'inclusion, nous avons besoin d'un modèle d'adaptation dynamique de la difficulté afin que n'importe quel enfant, dyspraxique ou non, puisse jouer et prendre du plaisir à jouer.

De cette problématique ressort particulièrement le manque de méthodes intermédiaires d'adaptation de la difficulté. Nous avons développé notre propre modèle d'adaptation dyna-

mique de la difficulté afin de combler ce manque. Pour les besoins du projet, nous devons utiliser un modèle simple, qui ne dépend pas d'un nombre important de données pour fonctionner. Ce modèle doit être générique afin d'adapter la difficulté de tous les mini-jeux du projet. Enfin, nous avons besoin de pouvoir cibler n'importe quelle difficulté afin de pouvoir proposer des challenges plus ou moins compliqués.

Pour développer notre modèle, nous avons dans un premier temps étudié les modèles existant. L'un des algorithmes qui répondait au mieux à deux de nos trois critères était le +/-δ. Le principe est d'augmenter la difficulté quand le joueur gagne, et de la baisser quand il perd. Cet algorithme fonctionne avec les données du dernier essai du joueur seulement, il ne nécessite donc aucune donnée au préalable. De plus, il est facilement intégrable à n'importe quel jeu sans le moindre souci. Cependant, cet algorithme nous amène à une difficulté que l'on peut appeler *équilibrée*, c'est-à-dire où le joueur a autant de chance de gagner que de perdre. Il nous a donc fallu tester le troisième point afin de voir si en modifiant légèrement l'algorithme, nous pouvions parvenir à proposer une difficulté de notre choix. Des tests ont été réalisés sous R afin de tester la version modifiée du +/-δ, mais les résultats n'ont pas été concluant, ce qui nous a poussé à réfléchir à une autre solution.

Allart et Constant ont utilisé une régression logistique afin de traiter des données de jeu dans le but d'évaluer la difficulté des jeux[14, 11]. De ces travaux nous est venu l'idée d'utiliser une régression logistique afin d'adapter en temps réel la difficulté d'un jeu. Mais une régression logistique nécessite une certaine quantité de données avant de pouvoir prédire correctement ce que nous voulons, à savoir la difficulté du jeu. Dans cette optique, nous avons développé le δ-logit, un modèle d'adaptation dynamique de la difficulté s'appuyant dans un premier temps sur un algorithme +/-δ afin de récolter suffisamment de données pour qu'une régression logistique puisse prendre le relais et proposer un niveau de difficulté de notre choix.

Nous avons testé la précision et la vitesse de convergence du modèle lors d'une expérimentation. Nous avons modifié un jeu de tir vu du dessus afin de faire jouer un joueur contre un ou deux ennemis contrôlés par une intelligence artificielle. Le but du jeu est de survivre et de se débarrasser des ennemis. À chaque fois que le joueur se débarrasse des ennemis, il gagne. S'il

se fait détruire, il perd. Chaque tour de jeu s'enchaîne avec juste quelques secondes de pause pour que le joueur ne se sente pas submergé.

Les résultats de cette étude ont été concluant, ce qui nous a permis d'intégrer notre modèle aux mini-jeux de l'application DysApp. Nous avons également travaillé sur le modèle de récupération des données de DysApp afin de pouvoir traiter de manière efficace les données récoltées. Ces données servent en partie à créer un profil de joueur personnalisé utilisable par l'enseignant pour avoir un aperçu des performances de ses élèves. Mais ces données sont également utilisées à des fins scientifiques par le Centre de Recherches sur la Cognition et l'Apprentissage et nous même. Nous avons également développé des prototypes de mini-jeux pour l'application DysApp, afin d'aider Tralalère. En outre, nous avons testé l'application à chaque nouvelle version afin de remonter les bugs présents. Cependant, la récupération des données de DysApp s'est montrée compliqué, en particulier dû au fait que le déploiement des nouvelles versions du logiciel en école n'était pas systématique, ce qui a entraîné des problèmes pour l'utilisation des données.

Nous avons donc décidé de monter une expérimentation en dehors du cadre du projet DysApp afin d'avoir le contrôle sur tout le déroulement de celle-ci. Le but de cette expérimentation est d'étudier les préférences des joueurs concernant la difficulté dans les jeux vidéo. Pour ce faire, nous avons établi quatre courbes de difficulté que nous estimons être les plus proches possible de la difficulté des différents jeux vidéo utilisé soit en recherche, soit dans le commerce. Parmi ces courbes, deux d'entre elles ont une difficulté constante et deux ont des pics de difficulté. L'idée est de vérifier plusieurs hypothèses sur le choix des courbes de difficulté. Nous avons décidé d'utiliser un jeu de tir à la première personne pour cette expérimentation. Nous souhaitions un jeu plus complexe que les mini-jeux du projet DysApp afin d'être plus souple sur la population cible. En effet, les mini-jeux DysApp ciblant les enfants du second cycle scolaire, il aurait été compliqué de se limiter à cette tranche d'âge. Le jeu que nous avons utilisé est un jeu en libre accès développé par Unity. Nous avons modifié certains aspects du jeu afin d'avoir une arène où le joueur se bat contre un ennemi contrôlé par une IA. La difficulté de l'ennemi est directement gérée par notre modèle. L'expérimentation commence par un questionnaire sur les habitudes de jeu des participants, puis une session de 10 à 20 minutes de jeu, et enfin un

questionnaire sur l'expérience de jeu. Les résultats de cette expérimentation ont été concluant et un article scientifique est en cours de rédaction pour publier ces résultats.

La partie 1 de cette thèse présente la revue de littérature concernant à la fois la difficulté d'un point de vue psychologique, le lien entre la difficulté d'une tâche et la réalisation de cette dernière, mais également la perception de la difficulté, la difficulté dans les jeux vidéo, les façons d'adapter la difficulté dans les jeux vidéo et l'utilisation de courbes de difficulté pour gérer la difficulté au cours du temps. La partie 2 décrit la problématique de recherche du projet de recherche finançant cette thèse ainsi que les enjeux qu'apporte ce projet pour ma recherche. Les parties 3 et 4 détaillent respectivement les étapes de développement de notre modèle d'adaptation dynamique de la difficulté et la mise en place expérimentation qui a suivi afin de valider scientifiquement ce modèle. La partie 5 montre notre apport au projet DysApp, les contraintes liées au projet ainsi les problèmes rencontrés. La partie 6 s'intéresse à notre dernière expérimentation sur les courbes de difficultés, nos hypothèses ainsi que les résultats obtenus. Enfin, une brève conclusion sur mes quatre années de thèse, ce que cela m'a apporté ainsi les questions scientifiques que soulève ma thèse et que j'aimerais traiter par la suite.

Chapitre 1

État de l'art

Contenu

1.1 La difficulté source de motivation

On parle très souvent de difficulté. On peut par exemple parler de la difficulté à lire un livre, la difficulté d'apprendre une langue étrangère, et bien d'autres choses. Le plus souvent, la difficulté est reliée à une tâche à accomplir. On va donc avoir des tâches simples à accomplir et des tâches complexes, difficiles à accomplir. La difficulté peut donc être considérée comme l'effort nécessaire à une personne pour accomplir la tâche qui lui a été confiée. La théorie du "Goal Setting", ou de l'établissement d'objectif en français, a généré un grand nombre d'études scientifiques sur le sujet[15, 16, 17]. Les deux hypothèses de cette théorie sont d'une part qu'un objectif difficile à atteindre amène à une meilleure performance qu'un objectif simple, à condition que l'objectif ait été accepté par la personne effectuant la tâche. D'autre part, un objectif spécifique amène également à une meilleure performance de la personne effectuant la tâche, en comparaison à un objectif général, une instruction de "faire de son mieux" ou aucun objectif. Une tâche spécifique, généralement quantifiable, permet à l'effectuant de savoir ce qu'il vise ainsi que de mesurer ses propres progrès. Cette idée d'avoir un objectif difficile, mais atteignable, a déjà été pensée par Bandura et sa théorie de "Self-Efficacy" ou auto-efficacité, théorie qui a également fait l'objet de beaucoup d'études[18, 19, 20, 21]. Le facteur majeur de cette théorie est l'atteignabilité de l'objectif. Un objectif trop difficile sera rejeté par la personne concernée, car elle le sait irraisonnable et inatteignable. L'auto-efficacité est propre à chacun. Une personne avec une auto-efficacité élevée aura des objectifs personnels plus grands et acceptera des objectifs professionnels complexes plus facilement. Le sentiment d'auto-efficacité est également important pour l'acceptation des objectifs donnés. Plus un individu aura un sentiment d'auto-efficacité important, plus il se verra accomplir des tâches imposées plus complexes.

Lannie et Martens ont étudié l'impact de la difficulté sur l'efficacité des programmes éducatifs[22]. Ils ont montré qu'il y avait un réel impact aussi bien au niveau de la motivation qu'à celui du développement des connaissances des étudiants. Piaget et Garcia ou encore Vygotsky ont également démontré l'importance d'une difficulté adaptée au niveau de l'étudiant[23, 24]. En effet, leurs études montrent que les étudiants gagnent plus de compétences s'ils effectuent des exer-

cices ni trop simples, ni trop complexes pour eux. Gickling et al. ont également démontré qu'un niveau de difficulté des exercices approprié au niveau de l'élève permettait à ce dernier d'être plus concentré sur sa tâche, contrairement à des exercices plus simples ou plus complexes[25].

Orvis et al. défini la difficulté d'une tâche comme étant le degré auquel l'activité représente une situation personnellement exigeante nécessitant un effort cognitif ou physique considérable afin de développer les connaissances et le niveau de compétence de l'apprenant[26]. Une tâche simple est donc une tâche ne nécessitant aucun effort de la part de la personne effectuant la tâche, alors qu'une tâche complexe va forcer cette dernière à employer des ressources physiques ou cognitives à hauteur de la complexité de la tâche. Dans certains cas, l'effort nécessaire va au-delà de la capacité actuelle de la personne, ce qui la pousse à apprendre de nouvelles compétences afin d'arriver à accomplir la tâche en question. Le lien entre la difficulté de la tâche et la performance est beaucoup étudié.

Britt a montré que la difficulté de la tâche influence la motivation, les attentes quant au succès de la tâche, le stress ou l'anxiété, sont directement liés à la performance. La motivation est l'un des principaux facteurs de la performance[27]. Hughes et al. ont démontré sur une population de 250 enfants de CM2 l'importance de la difficulté pour la motivation et la performance en séparant en deux groupes les enfants, le groupe facile et le groupe difficile[28]. Chaque groupe devait effectuer une tâche, puis chaque élève avait la possibilité de refaire la tâche deux semaines plus tard. Le groupe facile a eu un taux plus élevé d'élèves choisissant de refaire la tâche, ce qui permet d'interpréter la motivation des élèves en fonction de la difficulté de la tâche demandée.

Cette idée de la difficulté comme source de motivation est au cœur de la théorie du Flow de Csíkszentmihályi[29]. La théorie du Flow consiste à proposer une tâche de difficulté adaptée aux compétences de la personne effectuant la tâche afin d'atteindre l'état de "flow", un état mental où la personne est dans un état de concentration et de satisfaction maximum. Selon Csíkszentmihályi, le "flow" est un état centré sur la motivation. Cependant, il est également à noter qu'une discordance entre la difficulté de la tâche et les compétences peut amener à des états émotionnels tels que la relaxation, le contrôle ou encore l'éveil, mais également à de

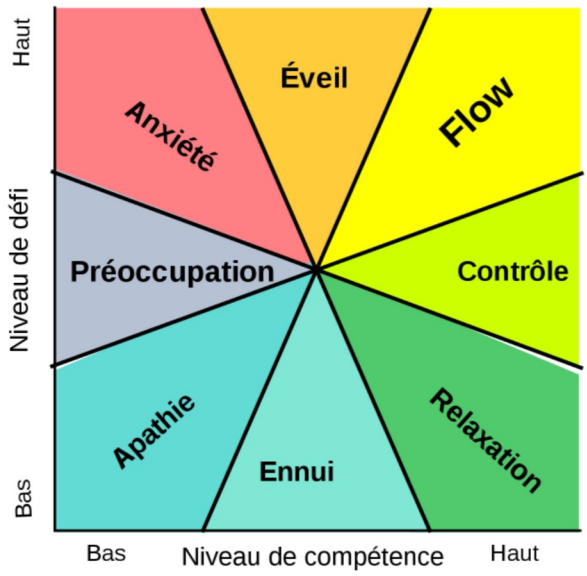


FIGURE 1.1 – États émotionnels en fonction du niveau de compétence et du niveau du défi

la préoccupation, de l'ennui ou même de l'anxiété, comme le montre la figure 1.1. Weiner a décrit que les gens réagissent avec des émotions positives s'ils sentent que leurs comportements sont la cause du succès[30]. Cependant, Weiner a montré que les individus ressentaient moins d'émotions positives si la tâche était trop facile. Ce qui confirme que la difficulté de la tâche est une variable à prendre en compte pour la motivation.

Cette section nous confirme que la motivation et la difficulté de la tâche sont liés. On apprend également que selon comment la personne effectuant la tâche perçoit la difficulté de cette dernière, la motivation à effectuer la tâche peut varier. De là se pose la question suivante : Comment un individu perçoit la difficulté d'une tâche ?

1.2 Perception de la difficulté

La difficulté d'une tâche n'est pas perçue de la même façon selon si l'individu est habitué à effectuer la tâche, ou s'il aborde le problème pour la première fois. On peut parler ici d'apprentissage. Chaque individu possède sa propre capacité d'apprentissage ainsi que ses propres compétences pour surmonter la tâche à accomplir. De ce fait, il n'est pas possible de quantifier

la difficulté d'une tâche seulement en fonction des besoins nécessaires à sa résolution. Borg et al. nous montre qu'il existe bien une relation entre la performance du sujet et la difficulté perçue par le sujet[31]. Cependant, les résultats inconsistants de ses études suggèrent que la perception de la difficulté ne se base ni sur les besoins objectifs de la tâche, ni sur une évaluation de la performance, mais sur une variable intermédiaire qui reste à déterminer. Delignières et Famose expliquent que ces résultats inconsistants sont dus à *"un manque de rigueur dans l'utilisation des méthodes psychophysiques et de leurs propriétés mathématiques"*[32]. Ils ont alors tenté de déterminer cette variable. Leurs résultats ne leur permettent cependant pas de conclure sur la nature de cette variable, bien qu'ils aient mis en évidence le fait que la perception de la difficulté ne se base ni sur une analyse des caractéristiques objectives de la tâche, ni sur une évaluation des performances réalisées, comme suggéré par les résultats de Borg[31]. Dans le même registre, Constant a montré que la perception de la difficulté dans les jeux vidéo est complexe et influencée par de nombreux paramètres[11]. Dans ses travaux, Constant définit la perception de la difficulté comme une estimation des chances d'échec du joueur. Les joueurs sont généralement trop confiants sur leurs chances de victoire à des niveaux de difficulté élevés.

La perception de la difficulté est un sujet complexe et il est compliqué d'énumérer une liste des paramètres à prendre en compte. Cependant, dans le cas des jeux vidéo, il a été découvert qu'en fonction de la difficulté, le joueur peut avoir une estimation biaiser de ses chances de succès.

1.3 La difficulté dans les jeux vidéo

De nombreux chercheurs considèrent la difficulté comme un aspect fondamental du plaisir de jeu et de la motivation à jouer[33, 34, 35, 36, 37, 38, 39]. Le challenge est au centre du gameplay. D'après Adams, *"Le gameplay est l'ensemble des challenges et des actions qui divertissent. Les gens apprécient un challenge, du moment qu'ils peuvent de façon raisonnable s'attendre à l'accomplir"*[40]. Malone différencie trois aspects fondamentaux du jeu, la difficulté, la curiosité et la fantaisie[36]. Pour lui, le challenge correspond à la probabilité qu'a le joueur de réussir le challenge proposé par le jeu. Parmi ces trois aspects, nous nous intéressons principalement à la

difficulté. Ryan et Deci nous expliquent, avec leur Self Determination Theory qu'ils appliquent au jeu vidéo, que le plaisir de jeu est lié au sentiment de réussite et de compétence, sentiment qui est augmenté si le joueur est confronté à une difficulté optimale[38]. Cette idée de difficulté optimale, adaptée aux compétences du joueur, vient de la théorie du Flow de Csikszentmihalyi[29]. Mais cette théorie explique également que différents stades émotionnels peuvent être atteint en modifiant la difficulté afin de proposer un challenge plus ou moins complexe. Juul propose une définition du jeu vidéo qui déclare qu'un jeu vidéo à des résultats quantifiables, influencé par les efforts du joueur[41]. Cette définition place le challenge comme un élément propre au jeu. Il explique également que l'échec est une étape importante dans le jeu vidéo, et donc qu'une difficulté élevée n'est pas à exclure lors d'une session de jeu[42, 43]. Sweetser et al place le challenge comme une des parties fondamentales de leur Game Flow framework, travaux qui découlent du travail de Csikszentmihalyi[37].

Loftus a étudié pourquoi les joueurs sont attirés par les jeux vidéo[44]. À cette époque, les joueurs jouaient principalement à des jeux d'arcade payants. Les joueurs étaient récompensés par un score élevé lorsqu'ils jouaient bien et devaient payer lorsqu'ils perdaient afin de conserver leurs progressions. L'objectif principal des concepteurs de jeux était d'avoir une difficulté assez dure pour que les joueurs perdent beaucoup, mais qu'ils aient l'impression de pouvoir battre le jeu pour continuer à jouer. Le chemin du succès est accessible, mais proche de l'échec et les joueurs pourraient regretter leurs décisions par la suite. Le regret est un moyen de faire réessayer les joueurs là où ils ont échoué, car ils étaient proches du succès qu'ils attendaient.

Pour comprendre l'attrait des joueurs pour la difficulté, Klimmt a décidé de faire savoir aux joueurs dans quelle difficulté ils se trouvaient lors de son expérimentation[39]. Il a découvert que les joueurs appréciaient davantage le jeu avec une difficulté inférieure, où ils avaient très peu d'échecs. Il est à noter que les joueurs ont eu 10 minutes de jeu, ce qui nous amène à suggérer que le premier contact avec le jeu devrait se faire avec une difficulté faible. Linehan et al. se sont concentrés sur quatre jeux de puzzle grand public, Portal, Portal 2, Braid et Lemmings[45]. Ils ont analysé les données de ces jeux afin de comprendre pourquoi les gens étaient attirés par ceux-ci. Pour tous les jeux, le même modèle est utilisé. Le début du jeu est vraiment simple,

car les joueurs doivent apprendre les mécanismes de base. Ensuite, chaque fois que le jeu inclut une nouvelle mécanique, les joueurs ont du temps pour se familiariser avec un puzzle de faible difficulté avant d'arriver à un niveau plus difficile. Cette étude appuie l'idée d'utiliser une difficulté simple pour la prise en main d'un nouveau jeu ou d'une nouvelle mécanique dans le jeu. De plus, Pruett explique que les premières minutes de jeu sont généralement celle qui définissent l'avis qu'on se fait sur du jeu[46]. Si le joueur échoue dès le début du jeu, il y a des risques qu'il se fasse une mauvaise opinion du jeu. Il peut donc être prudent du point de vue des designers de jeux vidéo de proposer une difficulté basse au début des jeux afin d'augmenter les chances que le joueur continue de jouer.

Alexander et al. ont testé différents niveaux de difficulté afin de trouver le niveau de difficulté le plus agréable pour tous les types de joueurs[47]. Ils montrent que les joueurs occasionnels choisissent des niveaux de difficulté qui ne correspondent pas à leurs compétences réelles. Les joueurs occasionnels semblent préférer des niveaux de difficulté inférieurs, même si leurs compétences sont supérieures à celles requises par le défi. D'autre part, les joueurs expérimentés préfèrent avoir des paramètres de difficulté en fonction de leurs compétences.

1.4 Modèles d'adaptation de la difficulté

L'adaptation de la difficulté est un sujet largement répandu et un domaine important en recherche. Ang et Mitchell définissent deux types de systèmes d'adaptation dynamique de la difficulté : l'un basé sur les choix des joueurs (pDDA), et l'autre basé sur leurs performances (rDDA)[48]. En utilisant le système pDDA, les joueurs peuvent manipuler la difficulté qui va leur être proposée tandis que le système rDDA modifie la difficulté en fonction des succès et échecs du joueur. Ils testent l'impact de ces deux systèmes sur les neuf dimensions de l'expérience du Flow. Les résultats montrent que les deux systèmes d'adaptation dynamique de la difficulté ont un impact positif sur l'expérience du joueur, car ils permettent aux joueurs d'entrer dans un état de Flow beaucoup plus rapidement et pendant une période plus longue. Ils trouvent que pDDA a un impact plus positif sur le sentiment de contrôle, car ce système permet aux joueurs de contrôler la difficulté du jeu. Leur étude établit un lien clair entre les systèmes d'adaptation dynamique de

la difficulté et une amélioration importante de l'expérience de Flow. Par extension, l'immersion et la motivation des joueurs sont également impactées par l'utilisation de tels systèmes.

Des systèmes d'adaptation dynamique de la difficulté tentent d'adapter la difficulté du jeu pendant la session de jeu, en se basant sur la performance du joueur. Contant et al. ont utilisé un algorithme qui réduit la difficulté quand le joueur perd ou augmente la difficulté quand le joueur gagne, que nous appelons l'algorithme +/-δ[11]. Cet algorithme peut être utilisé directement car il se base seulement sur le dernier résultat du joueur. Il permet de converger vers un état de difficulté que l'on peut appeler équilibré, où le joueur a une probabilité d'échec de 50%. Hocine et al. ont adapté un jeu thérapeutique à l'aide d'un système d'adaptation dynamique de la difficulté générique, qui peut être appliqué à n'importe quel jeu vidéo tant qu'une probabilité de succès peut être estimée[12]. Ils basent leur évaluation de la difficulté sur la probabilité d'échec du joueur, mais suivent une stratégie similaire à l'algorithme +/- δ, car ils réduisent la difficulté lorsque le joueur perd et l'augmentent lorsqu'il réussit et ne peuvent donc cibler qu'un état "équilibré". Les algorithmes d'adaptation dynamique de la difficulté peuvent être spécifiques à un genre de jeu, comme le "RubberBand AI", utilisé pour les jeux de sport ou de course, qui ajusteront la difficulté en comparant la position des ennemis et du joueur[49]. Cet algorithme permet d'avoir une partie serrée du début à la fin, ce qui peut être à la fois réjouissant pour le joueur qui s'empare de la victoire après une partie compliquée, mais également frustrant si ce dernier a passé la partie en tête et se fait dépasser sur les dernières secondes.

Colwell et al. ont évalué un mécanisme de jeu adaptatif en augmentant ou en réduisant le nombre d'ennemis attaquant les joueurs pendant un niveau d'un jeu de style arcade[50]. Ils évaluent le plaisir à travers un questionnaire post-match, croisé avec les données des joueurs collectées en jouant à deux versions du même jeu : une avec adaptation dynamique de la difficulté, une sans. Les résultats montrent que les joueurs faibles et forts peuvent mieux profiter de leur expérience de jeu lorsque la difficulté du jeu est adaptée à leur performance actuelle. Les joueurs ont également trouvé leur expérience de jeu plus équilibrée lors de l'utilisation des systèmes d'adaptation dynamique de la difficulté, selon le questionnaire post-match où les joueurs doivent évaluer la difficulté sur une échelle de 1 à 5.

Des méthodes d'adaptation de la difficulté plus performantes ont été développées en utilisant des algorithmes d'apprentissage. Olesen et al. ont utilisé un algorithme génétique pour faire évoluer en temps réel l'intelligence artificielle(IA) de leurs adversaires, afin d'ajuster de manière dynamique la difficulté d'un jeu de stratégie[2]. Andrade et al. propose quant à lui un algorithme de Q-Learning qui adapte dynamiquement la politique de l'adversaire, géré par une IA[9]. Spronck et al. proposent une technique similaire appelée script dynamique, qui peut être considérée comme proche du Q-learning, sauf que les actions sont remplacées par des scripts d'action créés manuellement[7]. Ces méthodes requièrent d'avoir une IA pouvant jouer contre elle-même afin de développer une politique de jeu et dépendent d'une heuristique d'évaluation de la performance du joueur. Dans l'ensemble, ces types d'algorithmes nécessitent de large quantité de données. Cependant, ces algorithmes nécessitant énormément de données, ils ne sont souvent pas accessible pour des jeux à but thérapeutique ou pour des expérimentations scientifiques ayant un budget limité[51]. Il est également à noter que si ces approches sont très intéressantes si le but est de générer des adversaires IA pour un jeu vidéo, elles peuvent présenter une autre lacune. En effet, ces approches prennent le contrôle total des comportements de l'adversaire, et si leur objectif est uniquement de créer des jeux équilibrés, alors ces comportements d'IA peuvent avoir de nombreux autres défauts, comme le manque de crédibilité que Demediuk décrit et que Ishihara essaie de corriger[52, 53].

Vincenco Moreira et al ont exploré plusieurs façons d'adapter un jeu de tir à la première personne, en utilisant des techniques d'aide à la visée[54]. Cependant, ces algorithmes sont spécifiques aux jeux de tir, et l'évaluation des performances des joueurs est basée sur une heuristique et ne donne donc pas une évaluation précise et hors contexte de la difficulté actuelle.

Xue et al. ont étudié l'adaptation de la difficulté comme un problème d'optimisation afin de maximiser l'engagement du joueur[55]. Ils ont modélisé la progression du joueur comme un graphe probabiliste. Cette approche peut être facile à mettre en place pour des jeux linéaires, mais un graphe plus complexe doit être pensé pour des jeux non linéaires.

Khajah et al. ont utilisé des techniques d'optimisation bayésienne afin de designer des jeux qui maximisent la motivation des utilisateurs[56]. Les participants étaient payés pour jouer

quelques minutes à un jeu, puis pouvait continuer de jouer ou non une fois le temps requis passé. Ils ont utilisé le temps de jeu comme une valeur proche de la motivation à jouer. Ils ont montré que des manipulations de difficulté non détectées par le joueur offraient de meilleur résultat concernant la motivation de celui-ci.

Zook et al. ont utilisé la réduction des tenseurs pour adapter la difficulté d'un jeu RPG personnalisé[57]. Leur approche permet de prédire l'efficacité des sorts pour un joueur spécifique à un moment précis, mais ne fournit pas une mesure plus générique de la difficulté comme la probabilité d'échec.

Allart et Constant ont utilisé une régression logistique à effets mixtes pour évaluer la difficulté des jeux commerciaux et expérimentaux[14, 11]. La régression logistique semble en effet bien adaptée pour prédire une probabilité d'échec à partir de quelques échantillons et de résultats binaires. Dans ces travaux, la régression logistique à effets mixtes a été utilisée parce que ces études avaient accès aux données de nombreux joueurs avec des essais répétés du même défi. De plus, ces études n'ont pas utilisé la régression logistique pour équilibrer dynamiquement le jeu, mais pour évaluer la difficulté à des fins d'analyse post-expérience. Ainsi, ils ne calculent la régression qu'à la fin, lorsque toutes les données de l'expérience sont disponibles.

La difficulté est donc un aspect fondamental des jeux vidéo, et la progression de la difficulté semble être l'une des nombreuses façons pour nous garder motivés et concentrés pendant de longues sessions de jeu. Nous considérons qu'une estimation de la difficulté d'un jeu est proche de l'estimation de la performance du joueur, que nous définissons comme la probabilité d'échec de ce dernier[58]. Une façon d'améliorer la motivation en utilisant la difficulté consiste à concevoir des courbes de difficulté qui représentent la façon dont la difficulté évoluera avec le temps. En concevant une courbe de difficulté, les concepteurs peuvent décider quand proposer du challenge aux joueurs, quand leur donner le temps de se reposer, augmenter ou diminuer progressivement la difficulté, créer des pics de difficulté. Il peut également être utilisé par un modèle d'adaptation dynamique de la difficulté comme présenté précédemment. Cependant, il existe peu ou pas de documentation sur la manière de concevoir une bonne courbe de difficulté.

1.5 Courbes de difficulté

Comme nous l'avons dit, la difficulté est pensée au fil du temps. Les jeux ont rarement le même niveau de difficulté du début à la fin, comme le montre Allart et al. en prenant deux jeux phare d'Ubisoft[14]. Byrne explique qu'une façon de planifier le rythme de difficulté est de concevoir une courbe de difficulté qui pilotera la difficulté du jeu pendant la session de jeu[59]. Selon certains concepteurs de jeux, une bonne courbe de difficulté commencera avec une difficulté basse, augmentera progressivement la difficulté jusqu'à ce qu'un événement spécifique se produise, puis elle réduira la difficulté afin de laisser le joueur profiter de son succès. Ce schéma se répète jusqu'à la fin du jeu[60, 61, 62, 46]. Souvent, la courbe de difficulté correspondra à de nouveaux mécanismes de jeu que le joueur devra maîtriser pour avancer, c'est le cas de The Legend of Zelda, un jeu emblématique de Nintendo[63]. Les joueurs recevront un nouvel objet au début d'un donjon, ce qui signifie qu'ils devront acquérir de nouvelles compétences, ce qui conduit à un pic de difficulté. Ensuite, au fur et à mesure que les joueurs explorent le donjon, ils amélioreront leurs compétences, ce qui se traduira par une diminution de la difficulté. Enfin, les joueurs affronteront le maître des donjons, un nouveau pic de difficulté. Suivre une courbe de difficulté revient donc à proposer aux joueurs des périodes de calme suivies par des moments plus agités dans le but de toujours stimuler le joueur.

Cette progression de la difficulté a un impact important sur la motivation à jouer, comme le montre Allart et al. avec leur étude de l'impact de deux courbes de difficulté de jeux sur la rétention des joueurs[14]. Les données proviennent de deux jeux industriels, Rayman Legends et Tom Clancy's : The Division. Ils ont montré que la difficulté est en soi une variable explicative de la rétention des joueurs et que les joueurs ont tendance à préférer des niveaux de difficulté plus élevés. Il est également à noter que les deux jeux ont presque toujours une probabilité d'échec sous la difficulté équilibrée de 50%. De plus, les auteurs ont montré que les gens préfèrent une difficulté moindre au début, ce qui confirme la théorie "self-efficacy" de Bandura[18]. Cet article montre également que des pics de difficulté sont présents dans les jeux Ubisoft. Il est démontré que les joueurs apprécient ces pics de difficulté lorsque le jeu n'est pas punitif, ce qui est le cas

de Rayman Legends.

Bien que la progression de la difficulté soit un sujet important pour la motivation à jouer, peu d'études semblent s'intéresser à ce sujet. Un des buts de cette thèse va donc être de proposer une évaluation de différents profils de courbe de difficulté afin d'évaluer pour chaque profil l'intérêt de son utilisation du point de vue de la motivation à jouer.

Chapitre 2

Problématique

2.1 Le projet DysApp

Cette thèse est réalisée dans le cadre du projet DysApp. L'objectif du projet DysApp est de développer un jeu vidéo sur tablette tactile permettant la pratique de la motricité fine et de la planification visio-motrice, tout en maintenant la motivation des élèves par son aspect ludique. Une telle application à pour but d'aider à la détection de la dyspraxie chez l'enfant. La dyspraxie ou troubles de la coordination motrice, concerne 5 à 6% de la population. Les enfants dyspraxiques sont très souvent également touché de dyslexie et ces deux troubles pourraient avoir en partie une cause commune. L'idée de DysApp est donc de proposer aux enseignants, aux parents d'élève, voire aux professionnels de santé un outil ludique permettant de fournir des points de repère sur le niveau de coordinations des enfants. Le but est de mettre en lumière la difficulté de certains enfants pour les tâches motrices proposées pas le jeu et de les accompagner dans leurs apprentissages par une pédagogie différenciée. La pratique du jeu a également pour but d'entraîner l'habileté motrice des enfants, en lien avec l'acquisition du langage écrit.

La validité de cet outil doit être testée scientifiquement dans le cadre scolaire, auprès d'enfant du cycle 3 (CM1 et CM2), une partie étant dyspraxique et l'autre non. Le jeu a pour but d'améliorer les processus qui font défaut chez les élèves présentant un trouble du langage écrit à savoir : la répétition de gestes en respectant un ordre précis, la planification visuo-spatiale, la planification temporelle et le rythme. Ce jeu permet par ailleurs désensibiliser les enseignants

aux manifestations précoces des troubles du langage écrit à travers un outil numérique innovant et utilisable dans le contexte de la classe. Ce projet vise une inclusion facilitée en classe par une pédagogie différenciée, et une évolution des pratiques enseignantes face aux élèves en situation de dyslexie ou de dyspraxie.

Une caractéristique importante du jeu est sa capacité d'adaptation aux compétences du joueur par la prise en compte d'indicateurs en temps réel permettant une évaluation et une adaptation du niveau de difficulté du jeu à chaque élève. C'est principalement sur ce point que nous intervenons.

Le développement d'un tel projet s'est fait en collaboration entre deux laboratoires scientifiques, le Centre de Recherches sur la Cognition et l'Apprentissage (CeRCA), spécialiste des questions d'acquisition du langage écrit et des troubles d'apprentissage. Le Cédric, reconnu dans la recherche sur les sciences numériques, et ici spécifiquement sur la difficulté dans les jeux vidéos. Enfin, Tralalère, entreprise de développement de jeux sérieux a été choisi pour s'occuper du développement de l'application.

2.2 Les enjeux pour la thèse

Ce projet nous offre, dans le cadre de cette thèse, un angle d'attaque pertinent sur la question d'adaptation de la difficulté. Tout d'abord, la tranche d'âge visée est précise, les élèves de 3ème cycle et le challenge comme facteur de motivation dépend du type d'âge visé. Cependant, peu d'études ciblent cette tranche d'âge. Ensuite, ce projet amène le besoin fondamental d'une grande plasticité du gameplay, adapté à tous les profils d'élèves, y compris dyspraxiques, et donc non spécifique aux dyspraxiques uniquement. Pour finir, la diversité du gameplay grâce à plusieurs types d'exercices amène le besoin d'une adaptation de la difficulté générique à tout type de jeu. De plus, l'application développée dans le cadre du projet DysApp va être déployée dans des écoles, où le Wi-Fi est souvent interdit et donc l'accès a une large base de données comprenant les logs des actions de tous les joueurs est impossible. De plus, une gestion des profils de joueurs par compte est nécessaire étant donné que la tablette utilisée par un élève est

susceptible de changer d'une session de jeu à l'autre.

Premièrement, nous voulons que notre système soit aussi générique que possible : nous voulons pouvoir l'utiliser avec autant de jeux différents que possible et nous appuyer sur une mesure de difficulté qui permet la comparaison entre les jeux. Suite à des travaux précédents, nous modélisons un jeu comme une suite de challenges qui peuvent être gagnés ou perdus et dont la difficulté peut être manipulée à l'aide d'un ensemble de variables. En effet, la notion de succès et d'échec est au cœur des jeux vidéo : dans beaucoup d'entre eux, les joueurs ont des objectifs clairs et leurs performances sont constamment évaluées. Chacun des succès ou échecs des joueurs a un impact sur la progression du jeu et leur est transmis à l'aide de retours audio, visuels ou haptiques. Nous proposons donc de partir de ces événements pour définir un ensemble de challenges, puis de suivre les échecs et les réussites des joueurs pour estimer leur probabilité d'échec pour ces challenges. Un tel challenge peut être, par exemple, sauter sur une plate-forme, tirer sur un autre joueur ou gagner une bataille contre des tanks ennemis. Nous considérons que la probabilité d'échec à ces challenges est proche de la difficulté d'un jeu vidéo.

Deuxièmement, nous voulons pouvoir choisir n'importe quel niveau de difficulté. Comme nous le verrons, certains algorithmes simples n'équilibrent que la difficulté vers une probabilité d'échec de 50%. Cependant, nous voulons pouvoir sélectionner instantanément n'importe quel niveau de difficulté, car de nombreux jeux ne ciblent pas une difficulté dite "équilibrée", proche des 50% de probabilité d'échec. En effet, un certain déséquilibre entre les compétences et les défis peut conduire à des états émotionnels souhaitables, par exemple l'excitation, le contrôle ou encore la relaxation, comme vu dans la section 1.1.

Enfin, nous voulons proposer un modèle qui utilise aussi peu de points de données que possible, recueillis à partir d'un seul joueur. Nous enregistrons un point de données à chaque fois que le joueur tente un challenge et voulons prédire la difficulté en utilisant moins de 20 points. Les deux objectifs précédents peuvent être atteints en utilisant diverses techniques, mais beaucoup d'entre eux nécessitent beaucoup de données générées soit par de nombreux joueurs, soit par des logiciels. Nous voulons que notre système gère un démarrage à froid et atteigne une précision suffisante avec le temps de jeu le plus court possible. De cette façon, notre modèle

peut être utilisé dans des jeux hors ligne, où les seules données disponibles peuvent être les données d'un seul joueur local commençant le jeu sans données au préalable. Par exemple, les jeux thérapeutiques nécessitent souvent une adaptation dynamique de la difficulté tout en ayant un processus de développement très complexe et peuvent ne pas générer suffisamment de revenus pour maintenir des serveurs capables de stocker les données des joueurs.

Toutes ces contraintes font que les modèles d'adaptation dynamique de la difficulté présenté dans la section 1.4 ne sont pas totalement adaptés aux besoins d'un tel projet, ce qui nous a poussé à développer notre propre modèle. Une fois ce modèle établi, nous serons en mesure de piloter la difficulté du jeu pour chaque élève. Comme décrit dans l'état de l'art (1.5), la progression de la difficulté est un enjeu majeur et nous ne pouvons pas nous permettre de viser une seule valeur fixe de difficulté. Nous devons proposer aux enfants une courbe de difficulté. Comme nous l'avons montré, l'état de l'art dans ce domaine, c'est-à-dire précisément l'impact d'une forme de courbe de difficulté d'un jeu vidéo sur la motivation joueur, est encore limité. Un enjeu de la thèse sera donc d'utiliser notre modèle pour proposer une évaluation de différents profils de courbe de difficulté.

Chapitre 3

Développement du modèle d'adaptation de la difficulté en temps réel

Contenu

3.1 Introduction

Notre objectif, comme expliqué dans le chapitre 2.2, est de développer un modèle permettant de cibler instantanément une probabilité d'échec spécifique tout en utilisant le moins de points de données possible, c'est-à-dire en n'ayant observé que quelques tentatives du joueur pour gagner le challenge. Nous souhaitons un modèle assez générique afin d'être déployable sur tout type de jeu. Afin d'adapter la difficulté le plus rapidement possible, notre modèle s'appuie sur l'utilisation d'une métavariable. Il s'agit d'un curseur permettant de modifier toutes les variables de difficulté en même temps. Étant donné que notre algorithme commence à adapter la difficulté dès la première session de jeu, il nous faut une méthode simple pour adapter la difficulté quand aucune donnée n'est disponible et une seconde méthode, plus précise, une fois qu'assez de données sont récolté. La première méthode utilisée est l'algorithme +/- δ qui nous permet de facilement récupérer les premières données nécessaires. La seconde est une régression logistique qui s'appuie sur les données récupérées par l'algorithme +/- δ pour prédire la valeur de la métavariable en fonction de la difficulté voulue.

3.2 Méta Variable θ

Nous proposons d'abord d'adapter la difficulté du jeu en utilisant une seule métavariable qui pilote toute la difficulté du jeu. L'utilisation de ce paramètre unique limite notre capacité à affiner la difficulté du jeu, mais limite également considérablement l'espace de recherche de notre algorithme. Nous avons nommé cette métavariable θ. De manière à utiliser une mesure de la difficulté qui a à la fois un sens et une valeur pour n'importe quel jeu, nous définissons un challenge comme un objectif que les joueurs tentent d'atteindre, et pour lequel ils peuvent réussir ou échouer. Par exemple, tirer sur une cible, sauter sur une plate-forme, terminer une mission. Nous trouvons alors un sous-ensemble de variables que nous pouvons modifier pour changer la difficulté de ce challenge, d'une tentative d'un joueur à l'autre. De plus, cette méthode est plus facile à mettre en place qu'une mesure particulière de performance pour chaque mini jeu, sachant que si le joueur utilise une autre stratégie que celle prévue pour atteindre l'objectif, la

mesure de performance peut être fausse, car elle se base sur un comportement supposé du joueur. Dans un jeu de stratégie par exemple, la performance d'un joueur n'est pas exclusivement due à la taille de son armée. Il faut prendre en compte la stratégie choisie par le joueur.

Une fois le challenge défini, nous définissons deux configurations de ce challenge, c'est-à-dire deux ensembles de valeurs pour les variables de difficulté qui modifie la difficulté du challenge. Ces deux configurations doivent être définies manuellement par le designer. La première configuration est le challenge le plus simple que l'algorithme est autorisé à créer, tandis que la seconde est le challenge le plus difficile. En nous limitant à l'intervalle compris entre ses deux configurations extrêmes, nous empêchons le modèle de proposer des challenges que nous considérons comme indésirables pour tout joueur. Ensuite, θ est utilisé pour interpoler linéairement chaque paramètre entre les configurations de challenge très faciles et très difficiles. Notre modèle ne conduit donc que θ pour ajuster la difficulté du jeu pour chaque joueur spécifique. La variable θ varie entre 0 et 1, et si un des paramètres de difficulté n'est pas continu, on l'interpole puis on l'arrondit à l'entier le plus proche.

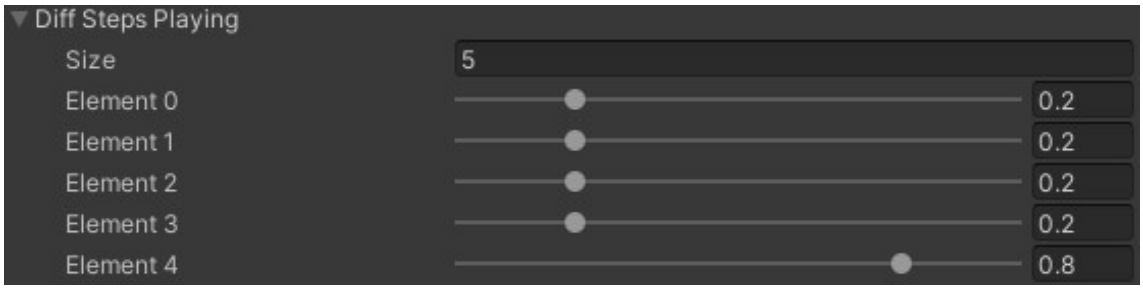


FIGURE 3.1 – Exemple de représentation d'une courbe de difficulté sous Unity
Cette courbe de difficulté se répète tous les cinq essais. Elle commence avec une difficulté basse de 20%, soit 20% de chance d'échouer. Au 5ème essai, le joueur à 80% de chance d'échouer. Cette courbe est donc une courbe de difficulté *basse* avec des pics de difficulté.

La figure 3.1 est un exemple de courbe de difficulté modélisée sous Unity. Chaque essai correspond à une tentative du joueur de surmonter le challenge auquel il fait face, essai qui peut se solder par une réussite ou un échec. Puis on enchaîne sur un nouvel essai. Cette courbe se répète tous les cinq essais, avec quatre essais avec une difficulté de 20%, et un essai à 80%. Les curseurs correspondent donc à notre métavariable θ. Toujours en prenant le même exemple,

ces curseurs modifient tous les paramètres de difficulté du jeu, présentés par la figure 3.2

```
public struct paramsDiff
{
    public float speedMove;
    public float speedTurn;
    public float timeBetweenShot;
    public float agroRange;
    public float shootRange;
    public float acceleration;

};

public paramsDiff easyDiff;
public paramsDiff hardDiff;
```

FIGURE 3.2 – Exemple de paramètres de difficulté d'un jeu Unity
Cet exemple montre les six paramètres de difficulté du jeu ainsi que la création des configurations facile et difficile.

3.3 Algorithme +/- δ

Pour adapter la difficulté en n'ayant au départ aucune information sur le joueur, nous avons étudié dans un premier temps l'algorithme +/- δ. Cet algorithme récupère le dernier essai du joueur afin d'adapter la difficulté : si le joueur gagne la difficulté augmente d'un δ (δWin), si le joueur perd la difficulté diminue d'un autre δ (δLose). En augmentant et en diminuant la difficulté en utilisant le même δ, la difficulté converge vers une probabilité d'échec de 0,5. Cependant, nous voulions voir s'il était possible d'obtenir d'autre niveau de difficulté en utilisant cet algorithme et en faisant varier les niveaux de δWin et δLose. Pour cela, nous avons simulé, en utilisant le logiciel R, 1000 joueurs ayant un niveau de compétence choisi dans une distribution normale standard avec une moyenne de 0 et un écart type de 0,1, comme le montre le listage 3.1.

Listing 3.1 – Generation des joueurs

```
sd_succes = 0.1
nb_players = 1000
sd_niveau_players = 0.1
nb_try = 60
diff_start = 0.1

players = rnorm(nb_players,0.0,sd_niveau_players)
```

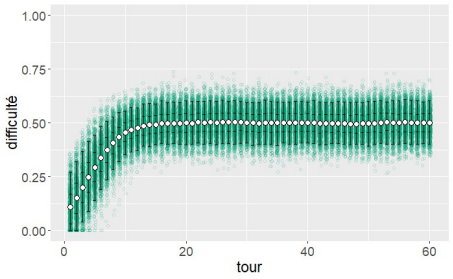
La compétence est directement ajoutée aux chances de succès du joueur, signifiant qu'un joueur avec un meilleur niveau de compétence à moins de chance d'échouer. La simulation est assez simple, nous testons pour chaque joueur généré s'il réussit ou non chacun des 60 challenges, comme montré par l'extrait de code 3.2.

Listing 3.2 – Code de la simulation

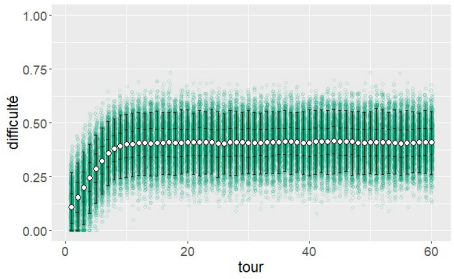
```
simu = function(diff_delta_succes,diff_delta_failure){
  MRes = matrix(nrow=nb_players,ncol=nb_try)
  xs = 1:nb_try
  for(p in 1:nb_players){
    paramDiff = diff_start
    for(i in xs){
      p_success = proba_succes(paramDiff,p)
      MRes[p,i] = 1-p_success
      res = jouer(p_success,sd_succes)

      if(res < 0.5) #si il a perdu
        paramDiff = paramDiff - diff_delta_failure
      else
        paramDiff = paramDiff + diff_delta_succes
    }
  }
}
```

En cas de succès, on augmente la difficulté de δWin, et en cas d'échec, nous la diminuons de δLose. En augmentant et diminuant la difficulté par le même δ, l'algorithme converge à une probabilité d'échec de 0,5, comme le montre la figure 3.3(a).

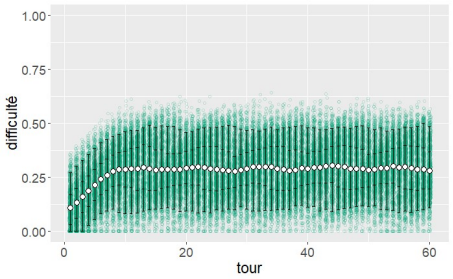
Pour cette simulation, le δ utilisé est donc le même, à savoir 0,05. Lors de cette simulation, nous avons une difficulté moyenne de 0,49 avec un écart type de 0,06, ce qui signifie qu'on estime que tous nos joueurs simulés ont été confronté à une difficulté moyenne se situant entre 0,43 et 0,55, c'est-à-dire une fréquence d'échec constaté comprise dans cet intervalle. Cependant,



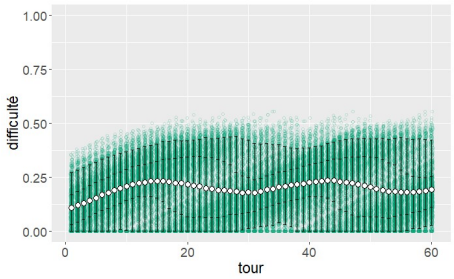
(a) Simulation avec une difficulté de 0,5



(b) Simulation avec une difficulté de 0,4



(c) Simulation avec une difficulté de 0,3



(d) Simulation avec une difficulté de 0,2

FIGURE 3.3 – Résultats de la simulation R pour tester l'algorithme +/-δ

lorsque nous tentons de converger vers d'autres difficultés, nous nous rendons compte des limites de l'algorithme. Pour avoir une difficulté de 0,4, nous avons utilisé un δWin de 0,05 et un δLose de 0,15. Bien que la moyenne soit à 0,41, l'écart type est à 0,09, soit une difficulté moyenne comprise entre 0,32 et 0,5. On se rend vite compte que l'écart entre les difficultés rencontrées par deux joueurs différents peut être trop important (fig.3.3(b)). En poussant plus loin, lorsque nous visons une probabilité d'échec de 0,3 par exemple en utilisant un δWin de 0,03 et un δLose de 0,3, la difficulté moyenne est à 0,29, mais avec un écart type de 0,11 (fig.3.3(c)). Et pour atteindre une difficulté encore plus basse de 0,2 en utilisant un δWin de 0,015 et un δLose de 0,4, l'écart type est à 0,12 (fig.3.3(d)). Bien qu'à chaque fois nous ayons réussi à atteindre la difficulté voulue en moyenne, les oscillations sont beaucoup trop importantes pour que notre modèle soit aussi précis que nous le voulons. La table 3.1 récapitule nos simulations pour chaque difficulté visées avec les δWin et δLose utilisé, la difficulté moyenne obtenue ainsi que l'écart type. De ce fait, nous avons décidé d'ajouter un autre algorithme au modèle afin de permettre au designer un choix de difficulté précis.

Difficulté visée	δWin	δLose	Difficulté moyenne	Écart type
0,5	0,05	0,05	0.49	0.06
0,4	0,05	0,15	0.41	0.09
0,3	0,03	0,3	0.29	0.11
0,2	0,015	0,4	0.21	0.12

TABLE 3.1 – Valeur de δWin et δLose pour chaque valeur de difficulté visée

3.4 Régression Logistique

L'impossibilité d'utiliser seulement l'algorithme +/- δ nous a donc poussé à ajouter une régression logistique à notre modèle, comme le montre la figure 3.4. Avec la régression logistique, nous pouvons choisir la difficulté voulue en utilisant un modèle continu convergeant avec peu de données en estimant une probabilité à partir de résultats binaires, succès ou échec. Pour récupérer les données nécessaires, nous utilisons une courte phase d'exploration au début de la session de jeu du joueur. Pour cette exploration, nous utilisons l'algorithme +/- δ. Nous commençons par une difficulté très basse, qui augmente petit à petit. À chaque tour, nous enregistrons le paramètre de difficulté du niveau ainsi que le résultat obtenu par le joueur, échec ou succès.

Afin d'explorer une plus grande portion de l'espace de jeu, nous multiplions le δ par une valeur aléatoire tirée de la distribution uniforme $\mathcal{U}(0.05, 0.1)$. Nous nommons cette valeur d'exploration *expl*δ. Cette exploration à pour but de voir comment le modèle réagit face à des difficultés dont il n'a aucune donnée. Cependant, afin de ne pas gêner le déroulement du jeu, le bruit que l'on apporte est faible. Nous pensons que ce bruit ne sera pas perçu. Nous avons mis en place une expérience afin de vérifier cette hypothèse, nous en discutons dans le chapitre de conclusion de cette thèse.

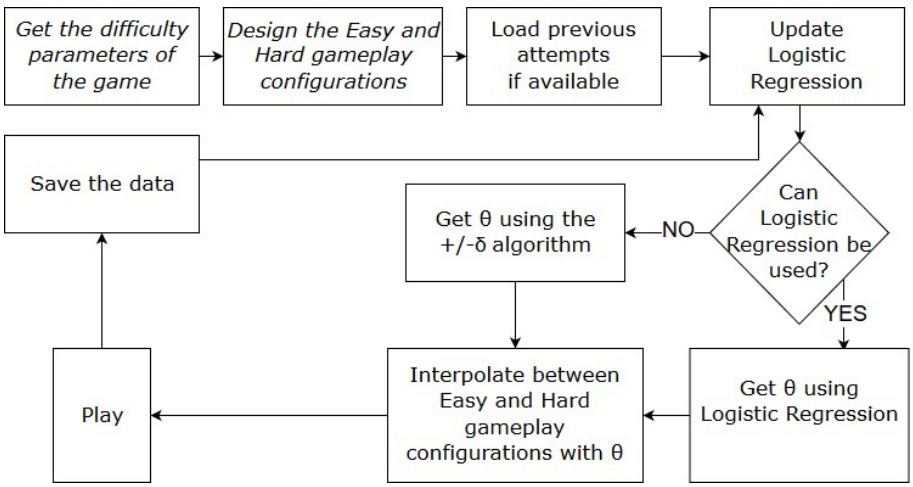


FIGURE 3.4 – Organigramme de l'algorithme δ-logit
Les étapes en italiques sont des étapes de design

Notre modèle commence à s'appuyer sur la modélisation logistique de la difficulté dès que l'algorithme +/- δ a fourni suffisamment de points de données. La régression logistique nous permet d'estimer une probabilité d'échec à partir de résultats binaires, de manière continue De plus, elle peut commencer à fournir une estimation avec 10 points de données[64]. Une fois la régression effectuée, le modèle est capable d'estimer la valeur de θ qui correspond à la probabilité d'échec souhaitée.

Pour mettre à jour la régression logistique, nous ajustons itérativement une fonction logistique aux points de données disponibles en utilisant la méthode de Newton-Raphson. Nous avons adapté le code C# fourni par McCaffrey pour l'utiliser dans Unity Game Engine pour effectuer notre expérience[65].

Pour estimer si nous pouvons passer de l'algorithme +/- δ à la régression logistique, nous effectuons plusieurs tests, résumés dans la figure 3.5. Premièrement, nous ne calculons pas la régression si nous avons rassemblé moins de 10 points de données[64]. Cependant, nous commençons avec l'algorithme +/- δ à une difficulté de 0, soit la configuration Facile du jeu, et avec $\delta = 0.05$. De ce fait, les 10 premiers points recueillis sont compris entre 0 et 0.5. Comme ces premiers points de données sont principalement échantillonnés sur les niveaux de difficulté les plus bas, nous ne commençons à effectuer la régression que si nous avons recueilli au moins

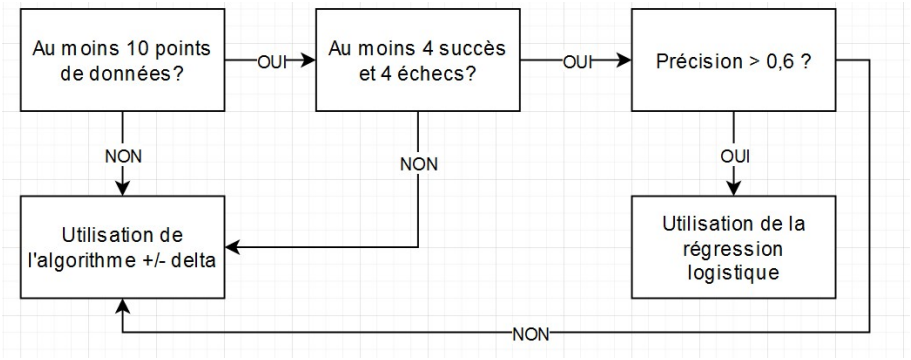


FIGURE 3.5 – Tests effectués avant de passer de l'algorithme +/- δ à la régression logistique

quatre succès et quatre échecs, car sans échec nous ne pouvons pas évaluer correctement le niveau du joueur. Si ces conditions de base sont remplies, nous effectuons la régression logistique et vérifions sa précision en utilisant une validation croisée à dix plis. La validation croisée est calculée en utilisant notre régression logistique comme prédicteur binaire [1] du succès et de l'échec et en comparant les prédictions aux résultats réels. Dans notre expérience, nous n'utilisons la régression logistique que si sa précision estimée est supérieure à 0,6. Nous avons choisi empiriquement d'utiliser 4 succès / échecs et un score de validation croisée 10 fois supérieur à 0,6. Ces valeurs se sont bien comportées dans notre expérience, présenté au chapitre 4, mais il peut être intéressant d'exécuter d'autres expériences pour étudier différentes valeurs.

Lorsque notre modèle δ-logit dispose d'une régression logistique suffisamment précise, δ-logit est capable d'estimer la valeur de θ qui correspond à une probabilité d'échec spécifique. Cette valeur peut être déterminée par n'importe quel processus : par exemple, un concepteur peut vouloir cibler des valeurs de difficulté suivant une courbe qui oscille autour d'une valeur de 0,5, permettant au joueur de vivre un gameplay globalement équilibré, tout en ayant des périodes d'excitation lorsque la difficulté est plus élevée. et un sentiment de contrôle quand il est plus bas, comme le suggère la théorie du Flow étudiée dans la section 1.1.

Nous voulons que notre algorithme continue d'explorer différentes valeurs de θ. Pour ce faire, nous proposons d'ajouter du bruit à la probabilité d'échec demandée par le jeu, de la même manière que lorsque le modèle utilise l'algorithme +/- δ. Nous appelons ce bruit *explDiff* afin de

1. Si p *(échec)* $> 0,5$, prédire l'échec et prédire le succès sinon

le différencier de *explδ* utilisé lors de la phase de récupération de données par l'algorithme +/- δ. Empiriquement, nous ajoutons une valeur tirée d'une distribution uniforme $\mathcal{U}(-0,05, 0,05)$ à la probabilité d'échec demandée lors de l'utilisation de la régression logistique pour estimer θ. De cette façon, si un concepteur demande une difficulté de 0,2, le modèle estimera la valeur de θ pour une difficulté choisie au hasard entre 0,15 et 0,25. Cela nous permet d'avoir des valeurs de θ qui varient toujours et nous considérons que la perception de la difficulté du joueur n'est pas assez précise pour percevoir une telle différence subtile, comme le montre Constant[11]. En effet, les personnes qui jouent à un jeu vidéo font souvent une évaluation irréaliste des chances réelles de leur succès. Plus précisément, les joueurs sont souvent trop confiants, sauf à de faibles niveaux de difficulté. Cependant, la perception de la difficulté est une question complexe et nous devrions étudier l'impact de ce paramètre à la fois sur la perception de la difficulté et sur la précision de l'estimation de la difficulté.

3.5 Conclusion

Ce chapitre nous présente le modèle d'adaptation dynamique de la difficulté δ-logit. Il peut être utilisé sur de nombreux types de jeux et permet à un développeur de régler la difficulté du jeu à n'importe quel niveau. Afin d'estimer approximativement la difficulté le plus rapidement possible, δ-logit pilote une seule métavariable pour ajuster la difficulté du jeu. Le modèle utilise dans un premier temps un simple algorithme +/-δ pour collecter quelques points de données, puis utilise la régression logistique pour estimer la probabilité d'échec des joueurs lorsque la plus petite quantité de données requise a été collectée. Dans le prochain chapitre, nous évaluons la précision de notre modèle lors d'une expérimentation.

Chapitre 4

Expérimentation du modèle de difficulté

Contenu

4.1 Introduction

Dans le chapitre précédent, nous avons décrit notre modèle de difficulté, nommé le δ-logit. Ce modèle est purement théorique et il est nécessaire de valider scientifiquement son utilisation en cas réel. Pour cela nous devons implémenter notre modèle dans un jeu afin de vérifier que les difficultés constatées au sein du jeu correspondent aux probabilités souhaitées. Ce chapitre présente un prototype de jeu qui utilise notre modèle de difficulté, de manière à étudier, en situation de jeu, le comportement du modèle.

4.2 Adaptation d'un jeu de tir

Dans un premier temps, il nous a fallu définir quel type de jeu utiliser pour notre expérimentation. Le but étant de tester notre modèle, et donc de faire jouer à nos participants des challenges en boucle, notre choix, c'est porté sur un jeu de tir. L'intérêt d'un tel jeu est de pouvoir enchaîner les challenges de manière soutenue pendant vingt minutes. Nous pensons que ce type de jeu est le plus apte à pousser les joueurs à se concentrer pendant le temps de l'expérimentation. Nous avons utilisé le tutoriel Unity *Tank Shooter Game*, comme le montre la figure 4.1[66]. Il s'agit d'un tutoriel portant sur la manière de réaliser un jeu de tir avec multijoueur local. Le terrain de jeu est donc une arène dans laquelle s'affronte deux joueurs. Chaque joueur contrôle un tank, peut avancer ou reculer, tourner à droite ou à gauche, et tirer des obus. Les obus explosent dès qu'ils touchent le sol, provoquant une petite explosion. Plus un tank est proche de l'explosion et plus il subira de dégâts. Les tanks proches de l'explosion se verront également projetés hors du champ d'explosion. Le but du jeu est donc de détruire son adversaire avant d'être soit même détruit. L'intérêt de s'appuyer sir ce tutoriel est que tout les assets sont libres de droit, ce qui nous a permis de les modifier à loisir afin d'avoir un jeu qui réponde au mieux à nos attentes. De plus, notre modèle ayant été développé sur Unity, ce choix de jeu nous a permis d'éviter une phase d'intégration complexe.

FIGURE 4.1 – Le tutoriel Unity
Dans le jeu développé par Unity, les deux tanks sont contrôlés par deux joueurs différents.

Nous définissons un challenge comme étant un but que le joueur cherche à atteindre, et pour lequel il peut réussir ou échouer [58]. Tant qu'un jeu peut être découpé en challenges que le joueur tente de réaliser, notre modèle peut adapter la difficulté. En partant de ce constat, nous avions pour objectif d'avoir un rythme de jeu soutenu afin d'impliquer le joueur. De plus, nous voulions que le jeu soit agréable à jouer afin que le joueur n'abandonne pas l'expérience en cours de route. Pour cela, nous avons grandement réduit la taille du terrain de jeu. Le jeu étant à la base multijoueur, nous en avons créé une version pour un seul joueur. Nous avons en effet besoin de contrôler la difficulté du jeu, et pour cela nous avons eu besoin de développer une IA contrôlant le tank ennemi.

Le fonctionnement de cette IA est simple. L'ennemi va toujours essayer d'avoir le joueur face à elle afin de pouvoir lui tirer dessus. Cette IA prédit la position du joueur en fonction de sa vitesse et de sa trajectoire afin d'être la plus précise possible. Étant donné que chaque projectile met du temps à arriver à destination et qu'ils ont une trajectoire balistique, le calcul de prédiction de la position du joueur doit être précis afin que l'ennemi ait des chances de toucher le joueur. De plus, que ce soit le joueur ou le tank ennemi, aucun des deux ne peut tirer à répétition. Chaque tank est soumis à un temps de recharge entre deux tirs. Afin de proposer une difficulté adaptée, nous avons dans un premier temps défini tous les paramètres de difficulté

TABLE 4.1 – Configuration Facile et Difficile

paramètres de jeu	Configuration Facile	Configuration Difficile
Nombre d'ennemis	1	2
Vitesse de déplacement	1.2	9.6
Vitesse de rotation	18	900
Temps entre deux tirs	3s	0.5s
Précision	0	15

du jeu.

Nous avons plusieurs paramètres qui définissent la difficulté du jeu. Ces paramètres sont la vitesse de déplacement de l'ennemi donnée en unité [1] par seconde, sa vitesse de rotation en degrés par seconde, le temps entre deux tirs en seconde, le temps d'arrivée d'un projectile en seconde, la taille de l'explosion en unité, la trajectoire du projectile, la précision qui correspond à un vecteur 2D aléatoire de taille 0 à 15 unités que l'on ajoute à la position prédite de la cible, et enfin le nombre d'ennemis. Afin de modifier la difficulté du jeu, nous avons sélectionné un certain nombre de ces variables, comme le montre la table 4.1. Nous avons, pour chaque paramètre sélectionné, défini une configuration Facile et une configuration Difficile. Ces deux configurations représentent respectivement le challenge le plus simple et le challenge le plus compliqué du jeu. Le modèle ne pourra donc pas proposer des difficultés aberrantes comme un challenge où il n'y a aucun tank ennemi, ou au contraire un challenge avec des dizaines de tanks qui peuvent tirer sans interruption.

À la suite de ces modifications, nous avons implémenté notre modèle δ-logit dans le jeu. Comme dit plus haut, le jeu étant codé sous Unity, tout comme notre modèle, l'intégration s'est faite très simplement. Le modèle a seulement besoin d'avoir accès au contrôleur des tanks ennemi ainsi qu'aux configurations Facile et Difficile définies au préalable. Le modèle δ-logit fonctionne en deux étapes précises. La première consiste à récupérer des données sur le joueur grâce à un simple algorithme +/- δ. Cette première étape est cruciale étant donné que l'on n'a aucune donnée au préalable sur le joueur. Lors de cette étape de récupération de données, nous devons récupérer au minimum dix points de données, avec au minimum quatre succès et

1. Les unités indiquées sont des unités Unity.

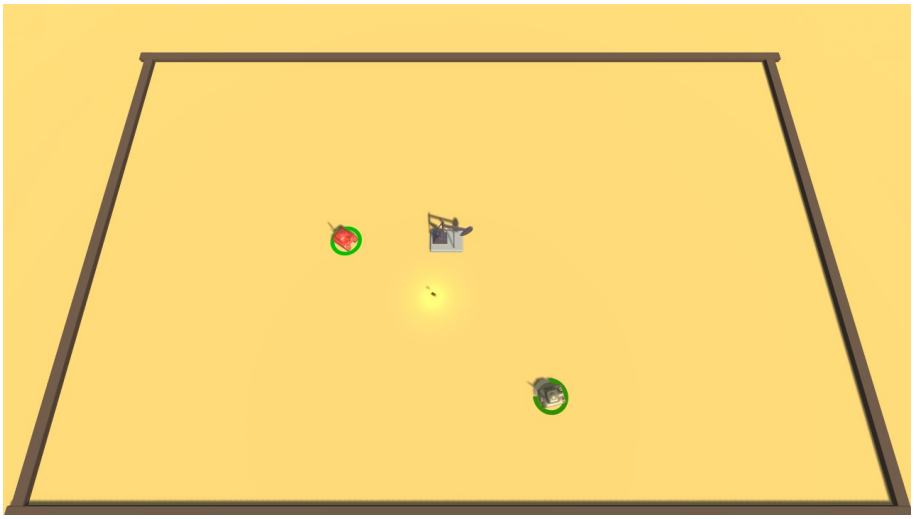


FIGURE 4.2 – Le jeu de tank
Le joueur, en bas de l'écran, se fait tirer dessus par l'ennemi.

quatre échecs. De plus, un calcul de la précision du modèle est effectué après chaque nouvelle tentative du joueur, afin de vérifier si la régression logistique est assez précise pour être utilisée. Une fois cette étape passée, le modèle utilise une régression logistique afin de prédire la valeur des paramètres pour une valeur de difficulté donnée. Pour modifier les paramètres de difficultés, nous utilisons la métavariable θ qui modifie tous les paramètres de difficulté de manière linéaire, comme montré dans le chapitre 3.2. Cette métavariable θ peut prendre des valeurs entre 0 et 1, correspondant respectivement à la configuration Facile et Difficile. Il est important de rappeler que chaque fois qu'un tank ennemi est détruit, ou bien que le joueur est détruit, le challenge est respectivement validé ou échoué. Une fois que le challenge est terminé, le modèle en propose un nouveau en suivant la courbe de difficulté. Afin que la probabilité d'échec du prochain essai corresponde à la courbe de difficulté, le modèle va calculer le θ nécessaire en se basant sur les données récoltées précédemment, puis interpoler les variables correspondantes et enfin créer un nouveau niveau avec les valeurs de contrôleur correspondantes.

Une fois tous ces changements réalisés, nous avons mis en place notre protocole d'expérimentation.

4.3 Méthodologie

Durant l'expérimentation, les participants ont joué 60 tours, correspondant en moyenne à moins de dix minutes de jeu. Les sessions de jeu étaient courtes, car nous voulions que les joueurs restent concentrés, et parce que l'objectif principal de l'expérience est d'évaluer la précision de notre modèle lorsque peu de points de données sont disponibles. Au début de l'expérimentation, nous n'avons aucune donnée sur le nouveau joueur et notre modèle débute donc avec l'algorithme +/- δ. Pour rappel, l'algorithme δ-logit fonctionne à froid en récoltant des données grâce à l'algorithme +/- δ. Lorsque suffisamment de données sont récoltées et que la régression logistique est suffisamment précise, δ-logit est capable d'estimer la valeur de þé*ta* correspondant à la difficulté visée.

Lorsque δ-logit a assez de données pour utiliser la régression logistique, nous ciblons des niveaux de difficulté spécifiques. Nous avons choisi d'évaluer notre modèle pour des probabilités d'échec de 0,2, 0,5 et 0,7. La difficulté de 0,2 est loin du réglage équilibré de 0,5 qui peut être atteint avec l'algorithme simple +/- δ, tout en restant un challenge qui peut être échoué. Cette difficulté de 0,2 est également proche de la difficulté moyenne de certains jeux AAA, comme le montre Allart [14]. Nous ciblons la difficulté 0,2 jusqu'au tour 44. Ensuite, nous testons si, après avoir échantillonné de nombreux points de données en jouant à un niveau de difficulté faible, nous sommes capables de créer des pics de difficulté précis. Ainsi, à partir du tour 45, nous entamons un cycle de trois tours avec des difficultés différentes, commençant à 0,2, montant à 0,5, se terminant à 0,7. Nous répétons ce cycle cinq fois, jusqu'au tour 60. Chaque tour prend en moyenne moins de huit secondes à compléter si le joueur comprend le but du jeu. Nous suivons une telle courbe de difficulté car elle suit le rythme de difficulté de nombreux jeux : augmenter lentement la difficulté lorsque le joueur découvre les règles du jeu, puis proposez un certain niveau de challenge, jusqu'à atteindre un pic de difficulté, comme les *boss* de beaucoup jeux vidéo.

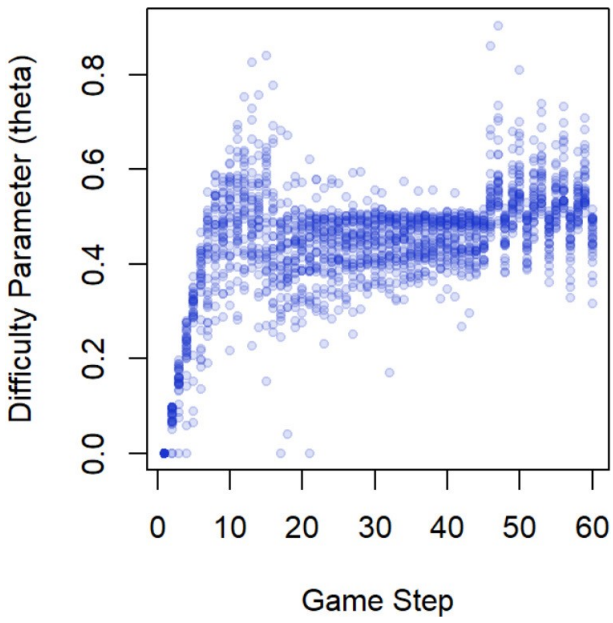


FIGURE 4.3 – Évolution du paramètre de difficulté θ

On peut apercevoir qu'en moyenne les 15 premiers tours se font avec l'algorithme +/- δ, suivi par une phase de difficulté à 0,2 jusqu'au tour 44, et vienne ensuite les pics de difficulté.

4.4 Résultats

Trente-sept participants ont joué à notre jeu de tir, avec un âge moyen de 30 ans. Nous avons dû retirer six participants, trois car le modèle δ-logit n'a pas réussi à converger [2], deux car le modèle a convergé dans les derniers tours et un qui avait un niveau très éloigné du reste des participants. Lors de l'expérience, nous avons ciblé trois probabilités d'échec, 0.2, 0.5 et 0.7. Notre modèle a réussi à atteindre des probabilités d'échec réelles de 0.14, 0.55 et 0.76 respectivement, simplement en utilisant notre métavariable θ. La difficulté réelle est estimée pour chaque participant en prenant la moyenne de leurs succès et échecs quand le modèle visait une difficulté de 0.2, 0.5 ou 0.7.

Nous avons ensuite étudié la vitesse de convergence du modèle. Nous avons d'abord calculé, pour chaque participant, le nombre de tours en +/- δ avant que le modèle ne passe pour la première fois à la régression logistique. Le modèle a mis en moyenne 15 tours pour converger ($\sigma = 1,82$ tours), ce qui correspond à 105 secondes de jeu ($\sigma = 24,52$ secondes).

2. Nous considérons que notre modèle a *convergé* lorsqu'il est capable d'utiliser la régression logistique

TABLE 4.2 – Fréquence d'échec réelle pour chaque probabilité d'échec visée

Difficulté réelle	Fréquence d'échec
0.2	0.14 [0.12, 0.16]
0.5	0.55 [0.47, 0.62]
0.7	0.74 [0.67, 0.80]

Pour chaque difficulté visée, nous renseignons les fréquences d'échec observées. Les valeurs entre crochet correspondent aux valeurs de l'intervalle de confiance données par un test binomial exact.

Nous avons également calculé la variabilité du modèle pour chaque tour. Nous avons utilisé le modèle au tour t pour prédire la probabilité d'échec pour 21 valeurs de θ, de 0 à 1 par pas de 0,05. Nous avons ensuite calculé l'erreur quadratique moyenne (RMSE) entre les prédictions au tour t et celles aux tours $t-1$, $t-2$ et $t-3$ comme indiqué par l'équation. Nous avons calculé la distance avec les dernières étapes pour avoir des valeurs plus grandes pour les modèles variant dans la même direction que pour ceux oscillant autour d'une valeur.

$$RMSE = \sqrt{\sum_{i=1}^{3} \frac{\sum_{j=0}^{20}(p_t(\theta = j/20) - p_{t-i}(\theta = j/20))^2}{3 * 21}} \tag{4.1}$$

La variabilité du modèle peut être examinée dans le temps en examinant la variation de notre métavariable θ tout au long des sessions de jeu, comme le montre la figure 4.3. En moyenne, la régression logistique a été utilisée après 15 tours, et nous pouvons voir qu'à partir du tour 20, la prédiction a tendance à être beaucoup plus stable. On peut également remarquer un pic de variation après le tour 45, correspondant aux pics de difficulté que nous avons inclus dans le jeu. Ces pics ont forcé le modèle à explorer des valeurs plus élevées de θ, et donc à se réajuster en conséquence.

Une autre façon d'examiner la précision du modèle et la vitesse de convergence consiste à examiner le résultat de la validation croisée au fil du temps. Lorsque la régression logistique est utilisée, la précision obtenue a une moyenne de 0,82 (σ = 0,06). Fait intéressant, il est à noter que pour 4 joueurs sur 31, nous sommes revenus à l'algorithme +/- δ même longtemps après les 15 premiers tours. Ces joueurs sont restés en +/- δ pour une moyenne de 2,75 tours (σ = 2,22), ce qui signifie que le modèle peut parfois perdre en précision.

Nous avons ciblé trois niveaux de difficulté (p *(échec)* = 0,2, 0,5 et 0,7) et le modèle a pu

atteindre les probabilités d'échec de 0.14, 0.55 et 0.74, comme présentées dans le tableau 4.2. Les probabilités d'échec réelles sont estimées pour tous les participants en prenant la moyenne de leur succès (1) et de leurs échecs réels (0) lorsque le modèle ciblait p *(échec)* = 0,2, 0,5 ou 0,7. Il est à noter que le modèle n'a jamais ciblé exactement ces valeurs, car un bruit uniforme $\mathcal{U}(-0,05, 0,05)$ leur a été appliqué. Les probabilités d'échec ciblées sont centrées sur 0,2, 0,5 et 0,7, mais ont un écart type de $0,03$

4.5 Discussion

Notre algorithme a réussi à adapter la difficulté du jeu à la courbe de difficulté voulue par un concepteur, voir Table 4.2. On peut cependant noter que pour p *(échec)* = 0,2 nous sommes légèrement inférieurs (0,14), alors que pour p *(échec)* = 0,5 et 0,7 nous sommes légèrement au-dessus (0,53 et 0,76). Cela pourrait s'expliquer par la nature de la progression du gameplay.

Comme expliqué précédemment, nous modifions les variables de difficulté toutes ensemble en suivant θ : les tanks deviennent plus précis, plus rapides et arrivent en plus grand nombre en même temps. Cette approche nous permet d'avoir une difficulté monotone : si nous avions choisi de changer d'abord la vitesse puis la précision, ces deux variables peuvent ne pas avoir le même impact sur la difficulté objective et créer un changement de progression lors du passage d'une variable à l'autre. Cependant, cette approche pourrait avoir l'inconvénient de compresser la difficulté objective dans une courte plage de θ. En effet, la difficulté objective peut croître de façon exponentielle avec θ car tous les paramètres augmentent au même moment. Cela se voit clairement dans notre cartographie de la courbe de difficulté θ en difficulté objective (Figure 4.3) : le jeu est très facile lorsque $\theta <= 0.4$ et très difficile lorsque $\theta >= 0,6$.

De plus, nous avons choisi d'avoir une variable de progression de gameplay qui n'a que deux valeurs possibles : le nombre de tanks ennemis. Au début, ce jeu a d'abord été conçu comme un jeu 1 contre 1, et il n'était donc pas évident de le mettre à l'échelle pour ajouter de nombreux tanks. De plus, nous nous sommes concentrés sur le réglage des comportements des tanks plutôt que de simplement changer leur nombre. Le jeu était satisfaisant et difficile à

jouer contre deux tanks. Lorsque nous avons remarqué que cela pouvait avoir un impact sur l'adaptation à la difficulté, nous avons choisi de conserver cette conception et de voir comment notre algorithme gérerait cette situation. Nous pensons que cela pourrait expliquer pourquoi la difficulté objective est inférieure à la difficulté ciblée dans le cadre facile et plus élevée dans le cadre difficile : lorsque $\theta > 0,5$, ce qui signifie qu'il y a deux tanks à battre, la difficulté augmente beaucoup plus vite que lorsque $\theta < 0,5$. Un tank à $\theta = 0{,}49$ est presque aussi fort qu'un tank à $\theta = 0{,}5$, mais le nombre de tanks crée un pic de difficulté à $\theta = 0{,}5$ et modifie la pente de l'impact de θ en difficulté objective lorsque θ franchit 0,5.

En ce qui concerne la précision de notre modèle, on peut s'interroger sur l'utilité d'être plus précis que ce que nous sommes actuellement. En effet, les jeux sont principalement créés non pas pour leurs aspects objectifs, mais pour la perception qu'en a le joueur. En tant que telle, la difficulté perçue parait donc plus importante que la difficulté précise et objective, et cette perception est complexe et influencée par de nombreux paramètres, comme le montre Constant[11]. À ce titre, il pourrait être intéressant d'étudier de quelle manière une légère variation de difficulté objective est perçue par les joueurs.

Notre modèle prend en moyenne 15 tours (une moyenne de 105 secondes de jeu) pour converger, ce qui est assez rapide pour notre jeu, permettant aux joueurs de découvrir le gameplay pendant quelques minutes à partir de la condition facile. Il est à noter que chaque tour est relativement rapide, prenant moins de 10 secondes. Lorsque nous estimons une probabilité, nous devons être en mesure de rassembler les échecs/succès des joueurs. Pour faire converger le modèle le plus rapidement possible, il est donc important de pouvoir diviser le gameplay en plusieurs défis courts, comme expliqué par Aponte[58].

Lors des premières étapes de δ-logit, nous nous appuyons sur l'algorithme +/- δ. Bien sûr, il nous est impossible de prédire la difficulté du jeu sans avoir un peu exploré les capacités du joueur. Nous avons réglé le +/- δ pour qu'il commence avec un niveau de difficulté faible et augmente lentement la difficulté (ou l'abaisse lorsque le joueur échoue). Ceci est cohérent avec les théories d'auto-efficacité de Bandura selon lesquelles l'échec, lorsqu'un sujet découvre une nouvelle tâche, peut atténuer sa motivation. Cependant, +/- δ peut être configuré pour

démarrer à partir de n'importe quel niveau de difficulté, par exemple à partir d'un niveau de difficulté choisi par le joueur. Comme nous avons conçu un réglage de difficulté très facile et très difficile, nous pourrions en calculer une interpolation pour fournir au joueur un réglage de difficulté de départ facile, moyen et difficile. Cependant, en partant d'un réglage facile, nous explorons rapidement les niveaux de difficulté faciles, obtenant rapidement plus d'informations sur les capacités du joueur avec des valeurs θ faibles que si nous partions d'un niveau moyen et nous adaptions la difficulté vers p *(échec)* = 0,5.

Il est à noter que nous utilisons toujours un paramètre d'exploration, nommé $explDiff$, pour aider notre modèle à explorer rapidement l'espace de jeu. Nous le faisons lors de l'utilisation de l'algorithme +/- δ, ainsi que lors de l'utilisation de la régression logistique. Comme nous l'avons noté dans les résultats, lors de l'utilisation de la régression logistique, dans notre exemple, cela conduit à un écart type de $0,03$ de la difficulté ciblée. Mais de temps en temps, comme $expl$ a une valeur aléatoire dans $[-0.05, 0.05]$, le concepteur ciblera une difficulté d, mais le modèle peut essayer d'atteindre $d + 0,05$ ou $d - 0,05$. Nous pensons que c'est un inconvénient mineur, mais il pourrait être utile de lancer d'autres expériences pour trouver les meilleures valeurs de $explDiff$.

On peut soutenir que notre but est de développer un modèle pour n'importe quel gameplay, mais que nous finissons par le tester sur un jeu de tir avec des ennemis pilotés par l'IA. En effet, nous avons développé notre modèle pour être utilisé sur une collection de mini-jeux pour un projet thérapeutique. Cependant, le fait de l'évaluer sur un gameplay de tir plus standard permet de montrer que le modèle ne se limite pas au contexte de mini-jeux spécifiques et peut être utilisé pour adapter un genre de jeu complexe et largement étudié.

Tant qu'un jeu peut être exprimé comme des défis que le joueur tente à plusieurs reprises de relever et que ces défis sont motivés par un ensemble de variables qui ont un impact monotone sur cette probabilité d'échec, notre modèle pourrait être utilisé pour conduire la difficulté de ces défis. Bien sûr, de tels défis peuvent être plus difficiles à identifier dans des jeux plus complexes. Dans un jeu AAA en monde ouvert comme *The Legend of Zelda : Breath of the Wild*, lorsque le joueur trouve un nouvel objet, cela ne change que la difficulté du défi actuel (par exemple

pour une arme légèrement plus puissante), ou cela change-t-il tellement le gameplay qu'un nouveau défi doit être défini (par exemple lors de l'utilisation d'un arc au lieu d'une épée) ? Et si le joueur monte un cheval en utilisant son arc, est-ce un nouveau défi ou une extension de l'arc ? Cependant, comme notre modèle utilise très peu d'échantillons, nous pensons que δ-logit pourrait être utilisé pour résoudre ces problèmes, lorsqu'il est correctement identifié. On peut en effet postuler que peu de jeux proposent des défis originaux que le joueur ne peut pas essayer plus de 15 fois.

4.6 Conclusion

Nous proposons un algorithme d'adaptation dynamique de la difficulté qui commence sans données et utilise ensuite le moins de points de données possible collectés à partir d'un seul joueur. De nombreux modèles ont été proposés pour ajuster dynamiquement la difficulté d'un jeu, mais aucun d'entre eux ne résout réellement le problème de l'utilisation de très peu de données d'un joueur tout en ciblant n'importe quelle probabilité d'échec.

Notre modèle commence par un algorithme +/- δ qui adapte la difficulté vers une probabilité d'échec de 0,5, puis utilise la régression logistique dès que possible pour suivre une courbe de difficulté spécifique, décrite en termes de probabilités d'échec, avec une précision correcte. Dans notre exemple, un jeu de tir de tanks, le modèle met en moyenne 105 secondes pour passer de +/- δ à la régression logistique et cible des difficultés de 0,2, 0,5 et 0,7, avec des difficultés réelles de 0,14, 0,55 et 0,74. La précision de la prédiction des échecs du modèle était de 0,82 ($\sigma = 0{,}06$). Nous évaluons notre modèle dans un cadre réaliste et stimulant : le tir est un mécanisme de jeu largement utilisé, nous adaptons le nombre et le comportement de l'IA de l'ennemi, et suivons une courbe de difficulté offrant une phase d'apprentissage, un plateau de difficulté faible et des pics de difficulté à des niveaux qui n'étaient presque jamais échantillonné avant.

Bien sûr, nous discutons du fait que notre approche présente quelques inconvénients. Avoir une seule métavariable doit avoir un impact sur la précision. De plus, notre modèle est continu et les discontinuités dans les variables de jeu peuvent ne pas être correctement modélisées.

Nous pensons que notre modèle pourrait être très utile pour la conception de nombreux jeux. De plus, nous pensons qu'en considérant n'importe quel jeu comme une collection de divers défis, nous pouvons utiliser plusieurs instances de notre modèle pour adapter des gameplay plus complexes. Il est à noter que notre modèle a été complètement intégré au jeu sérieux "Les Six Saisons de Brûme", jeu développé par Tralalère dans le cadre du projet DysApp.

Chapitre 5

Cas d'étude : DysApp

Contenu

5.1 Le projet DysApp

Cette thèse est financée par le projet DysApp. Le projet DysApp à pour but de développer un jeu vidéo sérieux sur tablette tactile permettant la pratique de la motricité fine et de la planification visio-motrice dans le but d'une part d'aider à la détection de la dyspraxie chez l'enfant, mais également d'entraîner l'habileté motrice des enfants. De façon générale, les difficultés des enfants dyspraxiques concernent la planification et l'exécution motrice. L'enfant rencontre des difficultés pour planifier, organiser et coordonner des actions nouvelles en séquences. Ces difficultés s'observent dans le traitement et la gestion de l'espace et dans le traitement du temps, en particulier du rythme. Le jeu fera donc intervenir trois types d'entraînements : entraînement moteur et visio-moteur, entraînement à la planification spatiale et entraînement à la planification temporelle. Pour chacun d'entre eux, différents niveaux de difficulté sont établis et l'élève est évalué en temps réel afin d'adapter le niveau de difficulté. Notre apport majeur va donc être d'intégrer notre modèle au jeu afin de pouvoir proposer des difficultés adaptées à chaque enfant. Dans un premier temps, nous pensons que cela aura pour effet de maintenir la motivation de l'enfant à jouer aux différents mini-jeux proposés. En effet, comme montré dans la section 1.1, une tâche ayant une difficulté adaptée est très souvent source de motivation. De plus, notre modèle permettra à des enfants en difficulté de pouvoir jouer au même jeu que des enfants sans difficultés, tout en ayant le même taux de succès, afin de les intégrer au mieux lors de ces activités de classe.

Ce projet est porté par le Centre de Recherches sur la Cognition et l'Apprentissage (CeRCA), et est en collaboration avec Tralalère, entreprise de création de ressources numériques éducatives et le laboratoire Cedric du CNAM.

5.1.1 Les contraintes liées à la dyspraxie

Chaque exercice doit respecter un certain nombre de règles, définies en amont par le Centre de Recherches sur la Cognition et l'Apprentissage (CeRCA) en suivant les études sur les besoins des enfants dyspraxiques.

1. Entraînement moteur et visio-moteur : il faut répéter des séquences de gestes sur tablette tactiles, gestes qui constituent des « caractères ». :
   (a) Importance de la succession des traits dans la réussite – ex. : les trois traits d'une figure doivent être dessinés dans un ordre précis ;
   (b) Importance du sens dans le dessin des traits (ex. un trait de gauche à droite) ;
   (c) La dynamique est plus importante que la qualité du produit final (si ordre et sens sont respectés cela suffit).
2. Entraînement à la planification visio-motrice : Il faut que les caractères soient inscrits dans un ordre logique par rapport aux objectifs ludiques jeu.
   (a) À certains niveaux, il faut que les caractères soient placés à des endroits particuliers (haut gauche, puis bas gauche...)
   (b) Pour les niveaux plus difficiles, il s'agit ici de présenter le modèle final à produire et le joueur devra planifier la succession des caractères par rapport à un objectif final.
3. Entraînement à la planification temporelle : La troisième caractéristique du jeu entraîne la capacité à réaliser une planification temporelle. Les élèves doivent progresser dans cette partie grâce à leur capacité à respecter un rythme. Les exigences sont donc la succession des traits tout en maintenant un rythme dans leur production.

5.1.2 Les activités retenues

5.1.2.1 Entraînement moteur et visio-moteur

Trois activités ont été retenues pour l'entraînement moteur et visio-moteur. La première est l'activité de Tracé(5.1). Le but de cette activité est de suivre un tracé généré aléatoirement. Le joueur voit apparaître sur la tablette un point ainsi que le début du tracé à suivre. Une fois qu'il appuie sur le point, il doit suivre le tracé. Le tracé n'apparaît pas entièrement, mais petit à petit au fur et à mesure que le joueur réussi à suivre correctement le tracé.

La seconde est l'activité Twister(5.2). Le but est de bloquer son pouce sur un bouton qui apparaît sur l'écran de tablette, puis de réussir à appuyer sur de nouveaux boutons qui

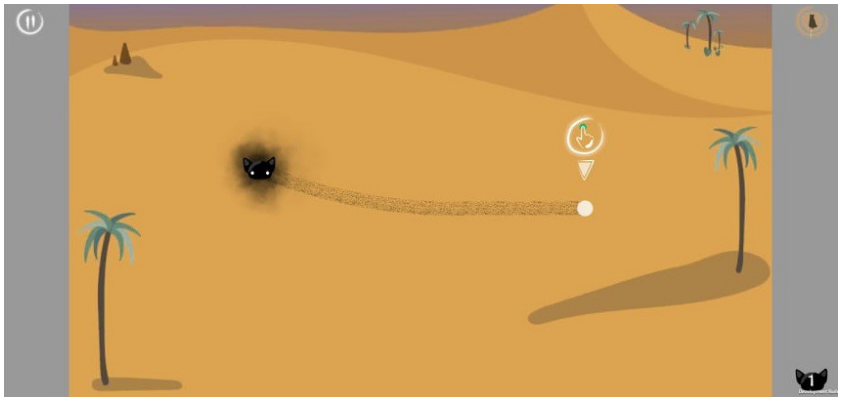


FIGURE 5.1 – Le mini-jeu Tracé

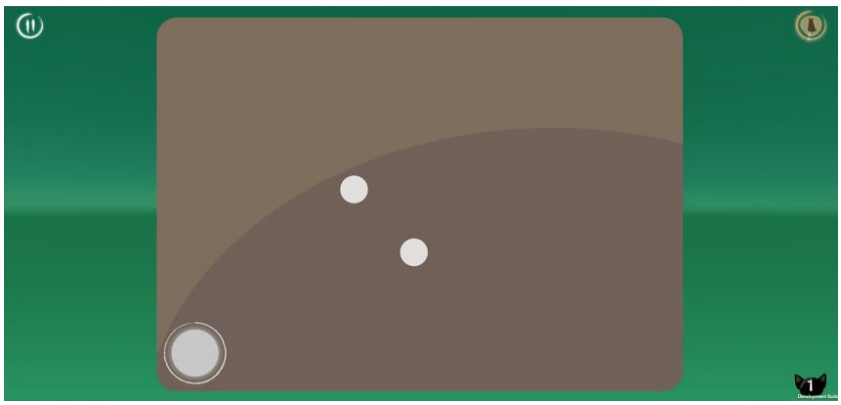


FIGURE 5.2 – Le mini-jeu Twister

apparaissent au fur et à mesure du temps passé à l'aide de ses quartes autres doigts. Une dimension temporelle est à prendre en compte, car les boutons disparaissent au bout d'un certain temps. On se rapproche de l'idée d'un jeu de tape-taupes, où le but est de taper sur des taupes qui sortent de terre le plus rapidement possible avant qu'elles ne repartent. Une calibration lors de la première session de jeu permet de connaître la taille de la main du joueur afin de placer les boutons à des endroits accessible pour le joueur.

La dernière est l'activité River(5.3). Le joueur doit placer son pouce et son index sur deux boutons affiché à l'écran, ce qui va créer une corde virtuelle entre ses deux doigts. Puis le décor commence à défiler et des obstacles apparaissent. Le joueur doit donc éviter que les obstacles ne se prennent dans la corde, sinon il perd une vie. Au bout de trois vies, le joueur perd et doit recommencer le niveau depuis le début. Afin que le joueur ne triche pas en utilisant ses deux

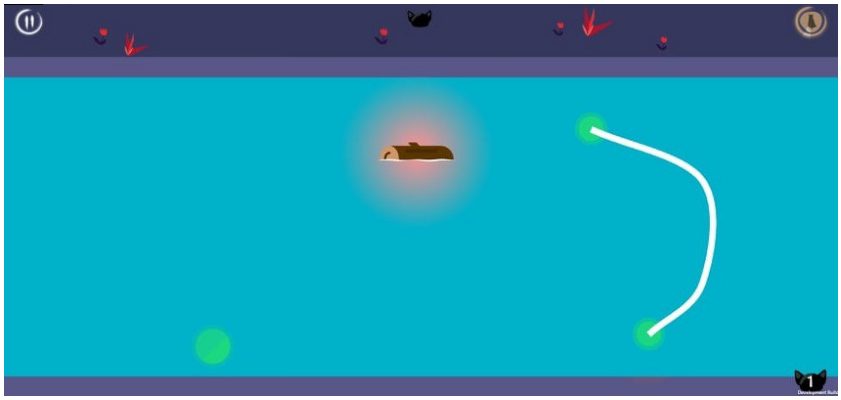


FIGURE 5.3 – Le mini-jeu River

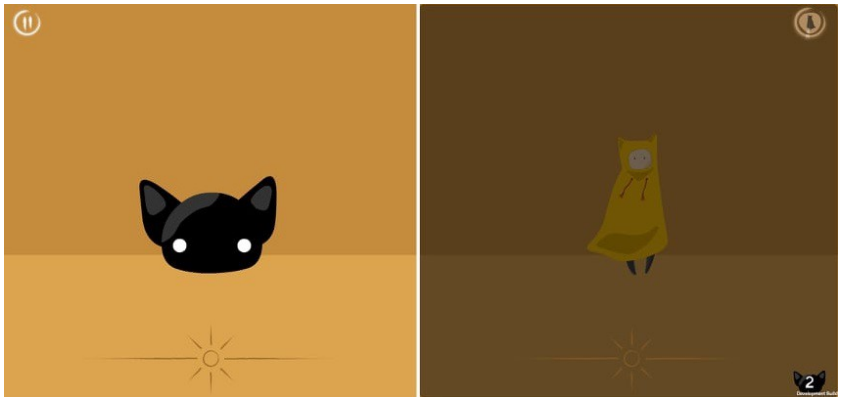


FIGURE 5.4 – Le mini-jeu Drums

mains, un bouton est à maintenir avec la main qui ne travaille pas.

5.1.2.2 Entraînement à la planification visio-motrice

Deux activités ont été retenues pour l'entraînement à la planification visio-motrice. Ce sont les activités de rythme. La première activité de rythme s'appelle Drums(5.4). Lorsque le joueur lance le jeu, une séquence sonore va être jouée. Le joueur va devoir se le rappeler et la rejouer. L'écran est donc séparé avec du côté gauche une IA qui génère le rythme à rejouer, et du côté gauche le joueur. Le côté du joueur est grisé lorsqu'il doit écouter le rythme et redevient normal quand le joueur doit rejouer la séquence. Le but ici est de rejouer la séquence dans le bon rythme. Le joueur peut donc se permettre de prendre sont temps avant de jouer le rythme.

La seconde activité, Pearl(5.5), joue une séquence sonore en boucle. Le joueur peut démarrer

FIGURE 5.5 – Le mini-jeu Pearl

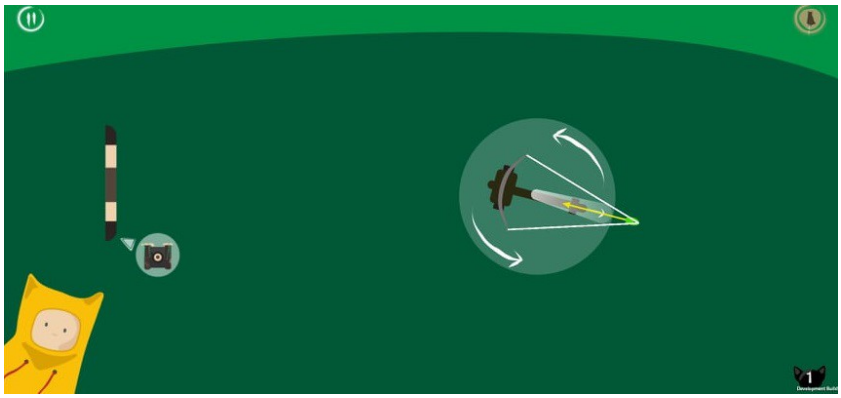


FIGURE 5.6 – Le mini-jeu Arrow

quand il le souhaite et doit se caler avec la séquence. Tout comme l'activité Drums, le joueur peut commencer à jouer le rythme dès qu'il se sent prêt. Cependant, il doit garder le rythme pendant suffisamment longtemps pour que le mini-jeu soit gagné.

Dans les deux activités, le joueur peut suivre le rythme en s'aidant soit du son, soit de l'animation du personnage.

5.1.2.3 Entraînement à la planification temporelle

Deux activités ont également été retenues pour l'entraînement à la planification temporelle. La première est l'activité Arrow(5.6). Le joueur contrôle un arc et doit tirer ses trois flèches dans une cible. L'environnement propose des obstacles fixes afin d'ajouter de la difficulté à l'activité.

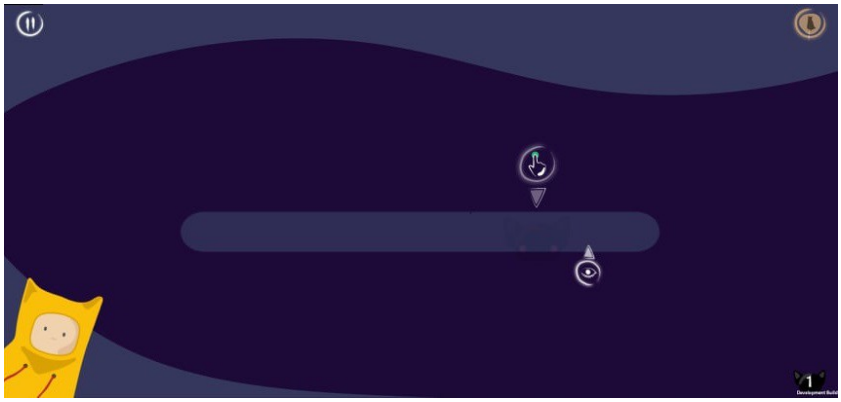


FIGURE 5.7 – Le mini-jeu Hide and Seek

La deuxième activité est Hide and Seek(5.7). Le joueur doit suivre un personnage qui se déplace le long d'un chemin a une vitesse constante. Au bout d'un moment le personnage disparaît tout en continuant à se déplacer pendant quelques secondes. Puis le jeu demande au joueur de deviner jusqu'où s'est déplacée le personnage.

5.2 Modélisation des profils

5.2.1 Architecture générale

En partant de cette base, notre travail a dans un premier temps consisté à réfléchir à l'architecture générale des traces d'utilisation ainsi qu'à la construction des profils joueurs. Nous avons besoin des traces d'utilisation de l'application pour deux tâches, l'analyse des actions du joueur et la création d'un profil personnalisé de ce joueur. Une partie des données, nécessaires à la création du profil du joueur en temps réel et l'adaptation de la difficulté des exercices peut être dans un premier temps stockée localement sous un format simplifié. Les données permettant une analyse plus globale du comportement des joueurs doivent, quant à elles, être stockées dans une base de données en ligne. Cette base permettra par la suite à l'ensemble des partenaires de pouvoir poursuivre des objectifs variés, aussi bien vis-à-vis de l'évolution des jeux que de la poursuite des études scientifiques associées au projet. Chaque entrée dans la base de données doit être horodatée, indiquer l'identifiant unique du joueur, et à minima

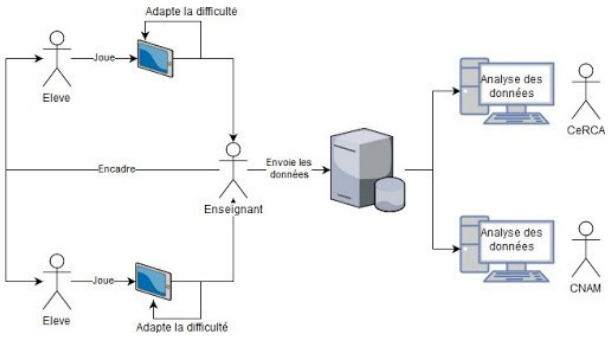


FIGURE 5.8 – Architecture du projet DysApp

l'identifiant du challenge tenté par le joueur ainsi qu'une indication de l'échec ou du succès de cette tentative. Un challenge correspond à un objectif particulier du jeu qui doit être défini pour chaque activité proposée par le jeu et dont le résultat, un succès ou un échec, doit être enregistré.

Pour récupérer les données des tablettes, il faut les envoyer au serveur. Cependant, il est possible que le Wi-Fi ne soit pas accessible en classe. Pour cela, l'application sauvegarde dans un premier temps en local sur la tablette toutes les données récupérées, et c'est à l'enseignant de connecter la tablette à internet et d'utiliser, après la session de jeu, la fonction d'envoi de données de l'application pour que tout soit enregistré sur le serveur. Ce fonctionnement permet de se passer du Wi-Fi en classe. Le jeu fonctionne donc grâce aux données stockées en local. De notre côté, nous récupérons les données envoyées sur le serveur afin de les analyser. Toutes les données récupérées sont sous forme de JSON. Nous les traitons une première fois grâce à un script afin de passer les données sous format CSV. Ensuite, nous traitons les données sous R.

5.2.2 Spécification générale des traces

Chaque trace contient des informations en commun, peu importe l'activité choisie. On retrouve dans chaque trace l'identifiant du joueur, l'identifiant du challenge ainsi que le résultat obtenu. À cela vient s'ajouter des données propres aux différentes activités. Ces données donnent des informations précises sur les actions du joueur, permettent de calculer un score en fonction du challenge proposé et de la réussite ou non à ce challenge. Tout cela permettant d'étudier le

comportement du joueur. Il faut cependant prendre en compte que les écoles n'auront pas les mêmes tablettes. La taille de l'écran ou le temps de réponse peut varier d'une tablette à l'autre. Il faut mettre en place une échelle constante afin de pouvoir analyser les données provenant de différentes tablettes sans avoir à les différencier. Ce point est particulièrement important pour l'activité "Tracé" car, sans cela, la taille de la tablette modifie la taille du tracé à suivre, et de ce fait influe sur la difficulté. Étant donné le coup de déploiement d'une expérimentation, on se limite le moins possible sur les données récoltées. Il faut donc prévoir une architecture supportant une charge importante de données. Le but est donc de décrire au mieux le comportement du joueur.

5.2.3 Construction du profil des joueurs

La création du profil des joueurs se fait avec notre modèle de difficulté, utilisant une régression logistique afin de calculer la probabilité d'échec au challenge proposé. Ce modèle nous permet, en fonction des réussites et échecs du joueur, de proposer un challenge approprié. Cependant, lors de la première session de jeu, nous n'avons pas d'information sur le niveau du joueur. Comme décrit dans le chapitre 3.5, notre système d'adaptation dynamique de la difficulté commence par une session d'apprentissage, proposant des challenges simples, permettant au joueur de prendre en main le jeu, tout en récupérant ses données de jeu pour entraîner le modèle. On se limite à une variable heuristique de difficulté, qui est calculée par le modèle pour fournir la difficulté objective du challenge, adaptée à chaque joueur.

5.3 Prototypes réalisés

Le CNAM a développé quelques prototypes de jeu afin de mieux comprendre et d'illustrer leur fonctionnement, dans le but de pouvoir plus rapidement évaluer le comportement du modèle de difficulté et donc d'ajuster au mieux la difficulté des jeux. Le but dans un premier temps fut d'aider Tralalère au niveau du game design. Nous avons développé deux prototypes afin de soumettre nos idées aux membres du projet, dans l'optique de proposer des mini-jeux qui ciblent

au mieux les besoins des enfants dyspraxiques. Nous avons également proposé des heuristiques permettant de définir la difficulté de chaque niveau généré. De plus, nous avons eu un rôle de testeur lorsque Tralalère proposait une nouvelle version du logiciel afin de trouver de potentiels bugs. Nous nous sommes principalement intéressés aux mini-jeux Tracé, Drums et River.

5.3.1 Tracé

Nous avons dans un premier temps développé un prototype du jeu Tracé, montré par la figure 5.9. Comme décrit dans la partie 5.1.2, l'objectif du jeu Tracé est de proposer au joueur de suivre un tracé avec son doigt. Il ne doit pas sortir du tracé, et il doit aller assez vite sinon le jeu considère qu'il a perdu. Un tracé est composé de points reliés par des segments de droite. La position successive de chaque point est générée en fonction des variables de complexité du tracé. Ces variables sont le nombre de points du tracé (sa longueur), sa courbure globale, la variation de cette courbure le long du tracé (changement de sens, apparition d'un angle aigu, petit plat, etc.). Elles permettent d'obtenir des tracés de différentes formes : d'une ligne droite à des tracés présentant des angles faibles et des angles forts, voire cassés. La position des deux premiers points est générée de façon aléatoire dans la zone de jeu. Les autres points à générer sont ensuite placés en fonction des deux premiers points, suivant les variables propres au challenge proposé. Plusieurs tracés sont générés et subissent diverses rotations et opération de symétrie jusqu'à obtenir un tracé dans le sens inverse de lecture qui tienne à l'écran et respecte les valeurs des variables décrites précédemment. En plus de cela, une dimension temporelle rentre en jeu. L'intérêt de l'activité étant de suivre un tracé sans s'arrêter, nous avons ajouté un compteur qui se déclenche dès que le joueur touche le premier point. Si le compteur dépasse la valeur seuil, décidée en fonction du niveau de difficulté, alors le niveau est perdu. Dans le cas ou le joueur atteint le point suivant avant la fin du compteur, le compteur se relance à chaque point, jusqu'à la fin du tracé.

Ce prototype a été validé et a été intégré à l'application DysApp.

FIGURE 5.9 – Prototype de Tracé

FIGURE 5.10 – Prototype de Drums

5.3.2 Drums

Le deuxième prototype développé par le CNAM est un prototype du jeu Drums. Le but de ce jeu est d'écouter un rythme puis de le reproduire. Nous avons développé ce prototype afin de mieux comprendre la génération de rythme, et de mieux définir la difficulté associée aux différents rythmes générés. Pour la difficulté des rythmes, nous nous basons sur le tableau de mesure de l'accent métrique défini par Toussaint [67]. Notre prototype montre deux personnages. Le premier est contrôlé par le jeu et bouge en même temps que le rythme à reproduire. Le deuxième est contrôlé par le joueur. Le joueur doit dans un premier temps écouter le rythme et le mémoriser. Puis dans un second temps, il doit le reproduire. Le score obtenu par le joueur correspond à la justesse du rythme reproduit. Pour comprendre la difficulté de ce jeu, il faut comprendre la génération de séquences rythmiques et leur complexité.

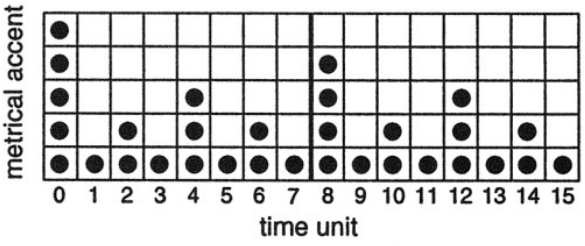


FIGURE 5.11 – Tableau de mesure de l'accent métrique

5.3.2.1 Calcul de la complexité d'une séquence

Le temps entre chaque pulsation est divisé entre 16 unités de temps de même durée (de 0 à 15). Pour chaque unité de temps, on attribue un booléen correspondant au fait qu'un son soit joué ou pas au début de cette unité de temps. Par exemple, pour 5 sons joués, on pourrait obtenir le tableau suivant :

$$[1, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0] \tag{5.1}$$

Ensuite, il faut calculer la valeur de l'accentuation de la séquence rythmique. Pour cela, faire la somme des valeurs d'accentuation correspondant à chaque son joué le long de la séquence. La valeur d'accentuation correspond au nombre de points indiqués dans chaque colonne du tableau de mesure de l'accent métrique(figure X). Sous forme synthétique, les valeurs d'accentuation des 16 unités de temps sont les suivantes :

$$[5, 1, 2, 1, 3, 1, 2, 1, 4, 1, 2, 1, 3, 1, 2, 1] \tag{5.2}$$

L'accentuation exprime à quel point les sons tombent sur une subdivision facile à appréhender pour le joueur. Par exemple, pour la séquence 5.1, on obtient l'accentuation suivante :

$$AccumulationSequence = 5 + 1 + 3 + 4 + 2 = 15 \tag{5.3}$$

On cherche à connaître l'accentuation maximale que l'on pourrait avoir pour une séquence de N sons, avec N le nombre de sons générés dans notre séquence. Par exemple, pour une séquence d'un son, l'accentuation maximale est obtenue est 5, lorsque ce son est dans la première case.

Pour 5 sons, comme dans notre exemple, l'accentuation maximale est de 17 (5+3+4+3+2 = 17). Toujours dans notre exemple 5.1, la complexité de la séquence rythmique est donc :

$$ComplexitéSequence = AccumulationMax - AccumulationSequence = 17 - 15 = 2 \quad (5.4)$$

On peut normaliser cette mesure de complexité en la divisant par la complexité maximale, qui est a priori celle obtenue pour une séquence de huit sons joués sur les intervalles de temps les moins accentués, avec

$$AccumulationSequence = 1 + 1 + 1 + 1 + 1 + 1 + 1 + 1 = 8 \quad (5.5)$$

$$AccumulationMax = 5 + 2 + 3 + 2 + 4 + 2 + 3 + 2 = 23 \quad (5.6)$$

et donc

$$ComplexitéSequence = 15 \quad (5.7)$$

Plus la complexité de la séquence est élevée, plus la séquence rythmique est jugée difficile à jouer. Plus la complexité de la séquence est faible, plus elle est complexe à jouer. Cette étude nous permet de comprendre comment faire varier la difficulté du challenge. Ce mécanisme de variations, qui répond à certaines variables, ici la complexité de la séquence, est manipulé au travers de θ par le modèle δ-logit afin de générer des séquences rythmiques jusqu'à ce que l'une des séquences générées ait la complexité correspondant à la difficulté voulue. Le modèle δ-logit nous permet donc de transformer cette heuristique en une valeur de difficulté, c'est-à-dire une probabilité d'échec pour un joueur.

5.3.3 River

Enfin, nous avons travaillé sur un prototype de River dans le même but que pour Drums, afin de comprendre comment générer des niveaux correctement avec la difficulté voulue, comme le montre la figure 5.12.

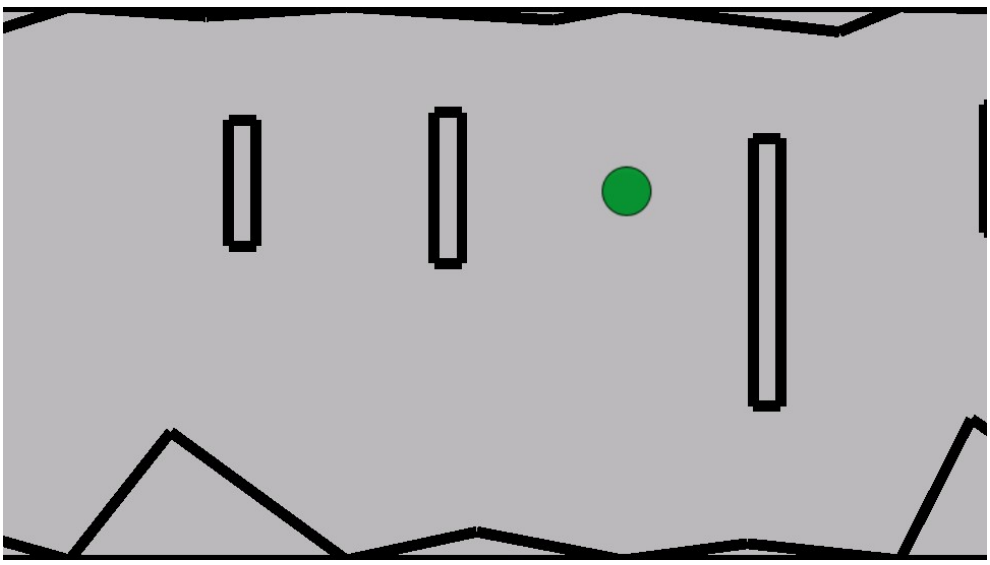


FIGURE 5.12 – Prototype de River

Bien que le jeu paraisse continu dans son déroulement, nous pouvons découper un niveau en une suite de challenge indépendant les uns des autres. La difficulté du jeu est donc calculée pour chaque challenge indépendamment. Un challenge correspond à un ensemble de chemins possibles dessinés par différents obstacles : un triangle de surface variable ancré en haut et en bas de l'écran et un obstacle rectangulaire ancré dans l'espace libre restant. Pour chaque challenge, le joueur peut emprunter deux chemins. Plus on ajoute de challenge, plus le nombre de chemins augmente. Pour un niveau ayant x obstacles centraux, soit x challenges, nous aurons 2^x chemins possibles. Pour calculer les chemins possibles, il faut projeter un segment partant de chaque sommet d'un obstacle sur la surface adjacente verticalement. Dans les faits, nous traçons donc deux segments *obstacles*. Un qui part du point le plus haut de l'obstacle centrale et qui rejoint l'obstacle en haut de l'écran et l'autre qui part du point le plus bas de l'obstacle et qui rejoint l'obstacle en bas de l'écran. Il faut ensuite obtenir le point de rencontre optimal entre la position du joueur et les deux segments *obstacles*. Le point de rencontre optimal correspond au croisement entre un segment *obstacle* et le joueur avec le moins de déplacement de la part du joueur. On construit ensuite une heuristique permettant de définir quel va être le chemin le plus probable que le joueur va suivre. On estime pour cela qu'un joueur lambda essaiera de passer par le centre des segments *obstacle* afin de limiter les risques d'échec. Parmi tous les chemins

possibles, l'heuristique choisis donc le chemin le plus simple. Nous définissons la difficulté d'un challenge en fonction du chemin le plus simple parmi les deux possibles. Il est donc à noter que même si nous considérons un challenge comme *facile*, le joueur peut essayer d'emprunter le chemin le plus compliqué des deux et ressentir le challenge comme étant plus compliqué que prévu.

Plus ce chemin est long et présente un angle élevé, plus il est considéré comme difficile. Chaque challenge correspond donc à un chemin entre deux segments. Nous pouvons donc définir la difficulté du challenge comme étant l'angle du chemin multiplié par la taille du chemin :

$$DiffChall = Angle(Chemin) * Taille(Chemin) \tag{5.8}$$

La difficulté générale du niveau est ainsi égale à la somme de la difficulté des challenges :

$$(DiffChall1 + DiffChall2 + \ldots + DiffChallX) \tag{5.9}$$

La figure 5.13 montre un exemple du jeu avec les différents chemins que le joueur peut emprunter. Dans cet exemple, nous avons deux obstacles centraux, ce qui signifie que nous avons deux challenges distincts. Chaque segment bleu correspond à un segment *d'entrée* d'un obstacle et chaque segment vert correspond à un segment de *sortie* d'un obstacle. Nous avons deux obstacles centraux, soit quatre chemins possibles. Les chemins sont représentés par une suite de segments de couleur rouge, orange, violet et jaune. Chaque chemin part donc de la position du joueur, et croise chaque segment *d'entrée* et de *sortie* d'obstacle par le point de rencontre optimal. Ici, le chemin orange est le chemin choisi par notre heuristique comme étant le plus logique à emprunter. Au vu de l'angle et de la longueur du chemin orange, la difficulté de ce niveau est relativement simple. Cependant, un joueur choisissant d'emprunter le chemin violet va certainement trouver le niveau bien plus compliqué.

Cette partie de nos travaux n'a pas encore été intégrée au mini-jeu correspondant.

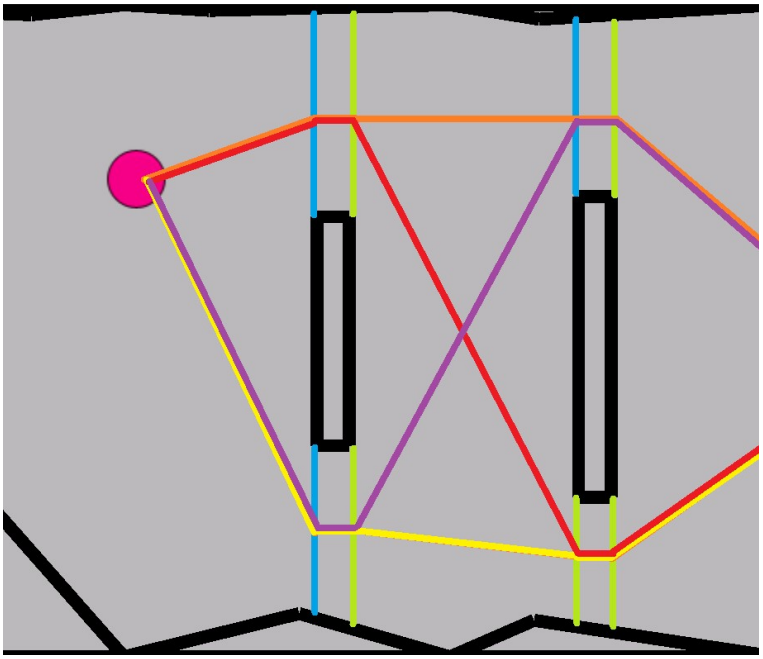


FIGURE 5.13 – Génération des chemins possibles
Nous avons sur l'image deux obstacles, soit deux challenges distincts.

5.4 Problèmes rencontrés

5.4.1 Récupération des données de jeu générées lors de la Paris Games Week 2018

Le studio Tralalère nous à proposer de participer à la Paris Games Week 2018. Il s'agit d'un salon annuel consacré aux jeux vidéo. Nous avons donc été présent au stand du jeu afin de le faire essayer aux participants. J'ai été présent les trois jours du salon afin de profiter de ce moment pour récolter un maximum de données, dans le but d'avancer sur mes projets de recherche. Cependant, la version du logiciel déployée par Tralalère lors de cet évènement avait un souci de stockage de données majeur, ce qui nous a forcé à faire des manipulations sur les tablettes après chaque nouveau joueur. Si un nouveau joueur commençait une session de jeu sans que l'on ait eu le temps de faire la manipulation, les données du joueur précédant étaient écrasée. De plus, l'afflux de personne au stand nous a empêché de faire cette manipulation correctement afin de limiter le temps d'attente des participants. Dans un sens, cela confirme

notre idée selon laquelle dans le cadre de ce type de projet, mettre en place le stockage de données est complexe et que le jeu doit pouvoir fonctionner même sans accès aux données. D'autre part, certains participants échangeaient de place avec d'autres participants en plein milieu de la session de jeu, ce qui a faussé une certaine partie des données récoltées. Cette observation correspond également au type de comportement que nous craignons de voir se produire en classe, soit parce que les enfants échangent leur tablette, soit à cause d'une erreur d'attribution des tablettes ou des comptes élèves. Nous avons donc sur les trois jours perdu énormément de données. De plus, les participants n'étaient pas forcément de l'âge cible, ce qui nous a donné des données beaucoup trop disparates. Nous avons tenté de séparer les données en groupe similaire afin d'analyser groupe par groupe les résultats. Mais les données groupe par groupe n'étaient pas suffisantes pour avoir des résultats significatifs.

5.4.2 Récupération des données de jeu générées lors du déploiement en école

Le jeu a été déployé dans les écoles partenaires tout au long du projet. Cela aurait dû nous permettre de récupérer énormément de données du jeu, d'une part pour valider le modèle d'adaptation de la difficulté que nous avons développé, mais aussi dans le but d'expérimenter sur les différentes courbes de difficultés adaptées à la tranche d'âge cible, c'est-à-dire les 9-12 ans. Cependant, à l'inverse de l'application qui recevait des mises à jour régulières et dont les versions étaient partagées régulièrement, les écoles ne recevaient pas les dernières versions de l'application. Nous nous sommes donc retrouvés avec des données de jeu en grande partie inutilisable. Certaines versions ne corrigeaient que quelques bugs mineurs d'un point de vue expérimental, mais d'autres versions du logiciel nous étaient essentielles afin de poursuivre nos recherches. Nous avions mis en place un algorithme permettant de choisir une courbe de difficulté parmi plusieurs courbes désignées par nous même, afin de pouvoir avancer sur mon sujet de thèse en étudiant les préférences des joueurs concernant la difficulté proposée par le jeu. Cependant, les versions du logiciel correspondantes à l'intégration de cette phase d'expérimentation n'ont jamais été transmise aux écoles et nous avons dû trouver une autre

solution afin de mettre en place cette expérimentation.

5.4.3 Conclusion

Le projet DysApp propose des problématiques de recherche très intéressantes pour cette thèse. L'adaptation de la difficulté de tous les mini-jeux de l'application DysApp permet d'une part de motiver les enfants à jouer, ce qui peut aider les enfants dyspraxiques à s'entraîner à effectuer des gestes qui leur sont difficiles à réaliser afin d'aider à réduire leur handicap. Mais cette application est aussi bénéfique pour les enfants non dyspraxique, car elle peut aider à améliorer le langage écrit. D'autre part, l'adaptation de la difficulté doit permettre d'intégrer plus facilement les enfants dyspraxique au sein d'une activité de classe commune à tous. Le CNAM a participé activement lors du développement des mini-jeux par Tralalère, en particulier pour le mini-jeu Tracé que nous avons développé. Nous avons été les principaux testeurs de l'application et nous avons travaillé efficacement avec Tralalère afin d'avoir une application prête à être déployée en école. Pour utiliser l'application DysApp pour une expérimentation, il nous fallait un système de récupération de données fiable, des mini-jeux fonctionnels et que notre modèle de difficulté soit intégré correctement afin d'adapter la difficulté de chaque mini-jeu. Cependant, des problèmes lors du déploiement en école ont fait que les données de jeu récoltées n'ont pas pu être exploitée correctement. Les versions du logiciel déployées en école n'étaient jamais les versions du logiciel les plus récentes. Ce problème s'explique par le nombre d'acteurs concernés par le projet : le CNAM défini un besoin technique, besoin intégré par Tralalère puis transmis au CeRCA qui supervise les expérimentations en école, qui eux-mêmes transmette une version particulière aux écoles. De ce fait, nous avons dû trouver un autre moyen pour récolter des données. Nous avons donc décidé de monter une expérimentation sans utiliser l'application DysApp.

Chapitre 6

Expérimentation sur les courbes de difficulté

Contenu

6.1 Introduction

Le but de cette expérimentation est d'étudier les préférences des joueurs concernant la difficulté des jeux vidéo. Nous savons que la difficulté d'un jeu se pense au fil du temps, Comme présenté lors de l'état de l'art, à la section 1.5, le peu d'articles qui traitent de l'intérêt des courbes de difficulté et de la façon de les modéliser sont majoritairement écrits par des concepteurs de jeux vidéo à partir de leurs propres expériences, de leur raisonnement et intuitions. Ces articles présentent différentes courbes de difficulté de jeux vidéo à succès, comme le fait Vazquez ainsi que des courbes de difficulté comme la courbe *dent de scie*, représentée par la figure 6.1 [60, 61].

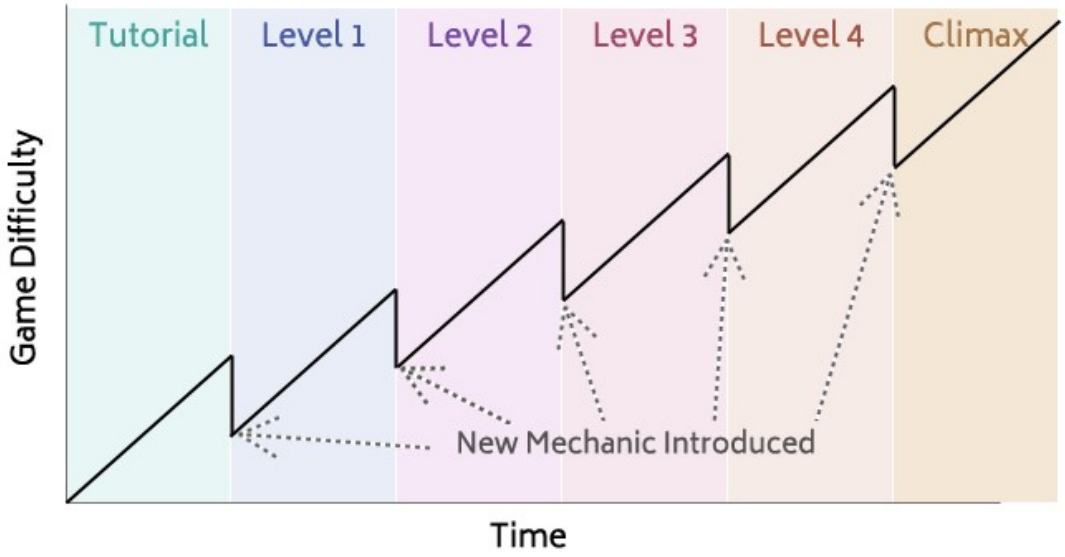


FIGURE 6.1 – La courbe de difficulté *Dent de scie* présentée par Strachan

Ces articles se basent sur les courbes de difficulté supposées de jeux vidéo à succès. Néanmoins, le succès des jeux vidéo est généralement calculé en termes de vente, et non selon le temps de jeu global passé sur ceux-ci. Il ne s'agit que très rarement d'une évaluation précise de la courbe de difficulté du jeu en question. Une approche intéressante est celle de Yee et de son questionnaire afin de modéliser ce qu'il appelle le *Gamer Motivation Profile*, ou bien Profil de Motivation du Joueur, de plus de 250.000 joueurs[68]. Yee montre que chez les jeunes joueurs, la

compétition est ce qui les motive le plus à jouer. Cependant, Yee ne s'intéresse pas à la difficulté à proprement parler, mais a des composantes telles que la compétition, la fantaisie, la stratégie ou encore la découverte, composantes qui peuvent être présente ou non dans un jeu vidéo et qui explique en partie le succès du-dit jeu vidéo pour un profil de joueur ciblé. Notre approche semble similaire à celle de Yee, mais consiste à tester non pas des composantes présentent dans un jeu vidéo, mais plusieurs courbes de difficulté sur un public cible afin de pouvoir déterminer les préférences du public ciblé concernant la difficulté des jeux vidéo. Nous cherchons donc à proposer des profils similaires du point de vue de la difficulté des jeux.

6.2 Design des courbes testées

Afin d'étudier les préférences des joueurs concernant la difficulté des jeux vidéo, nous avons choisi de proposer a des joueurs une courbe de difficulté aléatoire parmi quatre courbes de difficulté possible. Nous parlons ici de courbes de difficulté dans le temps, c'est-à-dire que la difficulté du jeu évolue au fur et à mesure du temps en suivant une courbe de difficulté. Nous retrouvons très régulièrement des courbes de difficulté qui tentent de suivre l'idée du Flow, à savoir de proposer une difficulté en accord avec les compétences du joueur [69, 70, 71]. L'idée pour modéliser une courbe de difficulté suivant le Flow est de proposer une difficulté où le joueur a autant de chance de gagner que de perdre, afin de le garder dans un état de concentration constant, proche de l'état de Flow. Notre première courbe de difficulté sera donc une courbe *plate* avec une difficulté constante et dont la probabilité d'échec sera de 0,5. Notre autre point d'accroche pour la modélisation de nos courbes de difficulté se fait du côté des jeux vidéo commerciaux. Nous savons par exemple qu'Ubisoft propose une difficulté relativement basse dans certains de ses jeux vidéo [14]. Une difficulté basse permet au joueur de se familiariser avec le jeu. De plus, la théorie du Flow nous explique qu'utiliser une difficulté plus faible que les compétences du joueur peut amener à des états émotionnels désirables tels que la relaxation par exemple[29]. Cependant, bien que les développeurs d'Ubisoft utilisent une difficulté basse dans certains jeux, ils intègrent des pics de difficulté dans leur courbe comme le montre la figure 6.2.

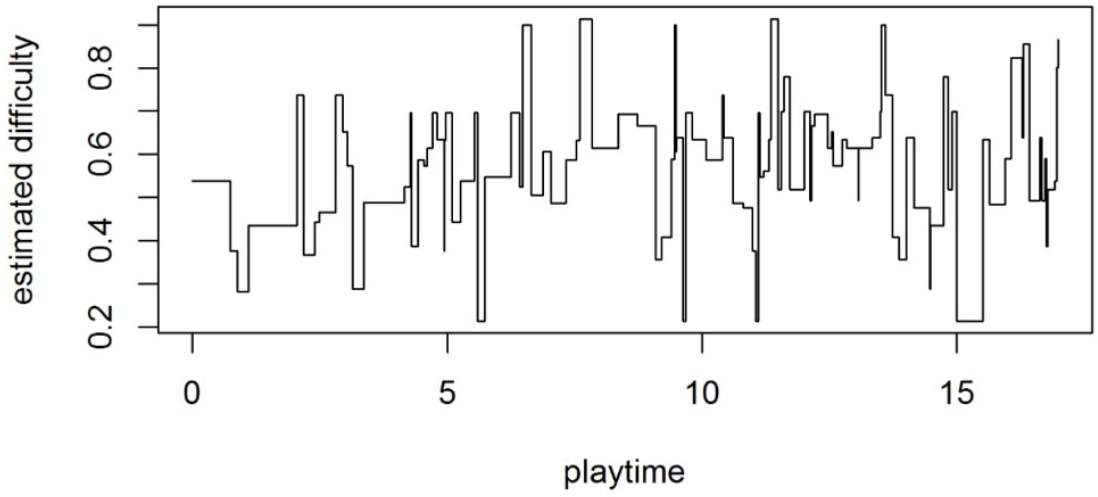
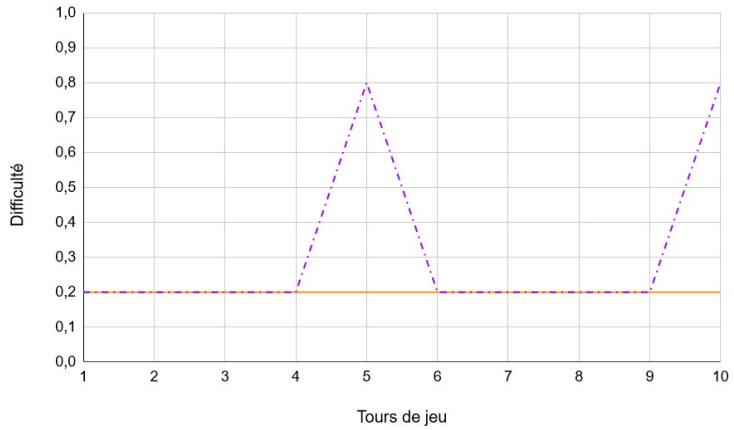
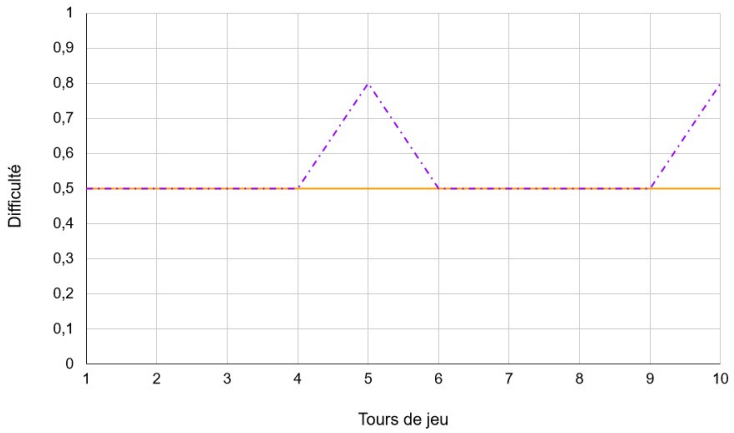


FIGURE 6.2 – Difficulté estimée d'un joueur de *Rayman Legends*
Cette figure provient de l'article *Difficulty Influence on Motivation over Time in Video Games using Survival Analysis* de Allart.

De plus, de nombreux designers de jeux vidéo indiquent utiliser des pics de difficulté pour motiver le joueur [60, 61, 62]. Notre deuxième courbe est donc une courbe avec une probabilité d'échec de 0,2 à laquelle nous avons ajouté des pics de difficulté de 0,8. Ces deux courbes représentent donc les courbes de difficulté les plus fréquentes. Nous cherchons donc à tester d'un côté la contradiction entre les courbes qui suivent le Flow, soit une courbe avec une difficulté constante de 0,5, et celle que l'on peut retrouver dans les jeux vidéo commerciaux avec une difficulté moyenne plus basse. D'un autre côté, les designers de jeux vidéo utilisent des courbes de difficulté avec des pics de difficulté tout au long de leurs jeux, ce qui est également en contradiction avec l'idée d'une courbe qui suit le Flow. De ces constatations, nous avons donc deux variables à explorer, avec chacune deux valeurs possibles : le niveau de difficulté de base, moyen ou bas, et la présence ou non de pics de difficulté. Nous avons donc quatre courbes de difficulté à tester, correspondant à la valeur basse sans pics de difficulté, basse avec pics de difficulté, moyenne sans pics de difficulté et moyenne avec pics de difficulté. La figure 6.3 représente les quatre courbes de difficulté retenues.



(a) Courbes de difficulté *basses*.



(b) Courbes de difficulté *moyennes*.

FIGURE 6.3 – Nos courbes de difficulté

Les courbes en oranges sont les courbes constantes, tandis que les courbes en violet sont les courbes avec des pics de difficulté. Les pics de difficulté surviennent tous les cinq tours de jeu et proposent une probabilité d'échec de 0,8.

6.3 Hypothèses et Méthodologie

6.3.1 Hypothèses

Avec la revue de la littérature, nous savons que connaître la difficulté du jeu a un effet sur la motivation, tout comme le fait que l'échec soit punitif ou non. La difficulté du jeu peut conduire les joueurs à un manque de motivation s'ils savent qu'ils jouent en faible difficulté. Nous avons

alors choisi de cacher aux joueurs la difficulté du jeu. Dans cette configuration, il est connu que les joueurs devraient profiter d'une difficulté correspondant à leurs compétences. Notre première hypothèse, qui concerne les courbes de difficulté constante, est donc que la courbe *moyenne* constante, qui propose une difficulté correspondant à la compétence du joueur, sera plus appréciée que la courbe Basse. De plus, nous utilisons un jeu qui n'est pas punitif, que nous présenterons dans le prochain paragraphe. Dans ce jeu, perdre conduit les joueurs à un autre essai pour une tâche similaire, la seule différence entre la tâche échouée et la nouvelle sera la difficulté de la tâche. Les joueurs affronteront le même ennemi, dans la même arène, avec les mêmes objets. Dans ce cadre-là, nous pensons que les courbes de difficulté avec des pics seront plus appréciées, car elles apporteront plus de satisfaction si le joueur réussi l'essai où la difficulté est élevée, mais il ne ressentira que très peu de frustration s'il échoue, car il pourra directement recommencer à jouer sans perte de progression. Enfin, notre dernière hypothèse est le regroupement des deux premières, à savoir que la courbe moyenne avec des pics sera la plus appréciée des quatre courbes.

Pour résumer, nos hypothèses sont les suivantes :

1. La courbe *moyenne* constante est plus appréciée que la courbe *basse* constante ;
2. Les courbes avec des pics de difficulté sont plus appréciées que les courbes constantes ;
3. La courbe *moyenne* auquel on ajoute des pics de difficulté est la plus appréciée des quatre.

6.3.2 Méthodologie

En suivant les travaux d'Aponte et al., nous considérons qu'une estimation de la difficulté du jeu est très proche de l'estimation de la performance des joueurs, que nous définissons comme leur probabilité d'échec[58]. Comme le montre la figure 6.3, les courbes représentent la probabilité d'échec à chaque essai.

Nous avons modifié le mini-jeu FPS Microgame développé par Unity pour en faire un jeu d'arène un contre un[72]. La figure 6.4 est une capture d'écran lors d'une session de jeu.

Dans ce jeu, le joueur peut avancer, reculer, aller à gauche et droite et peut déplacer la

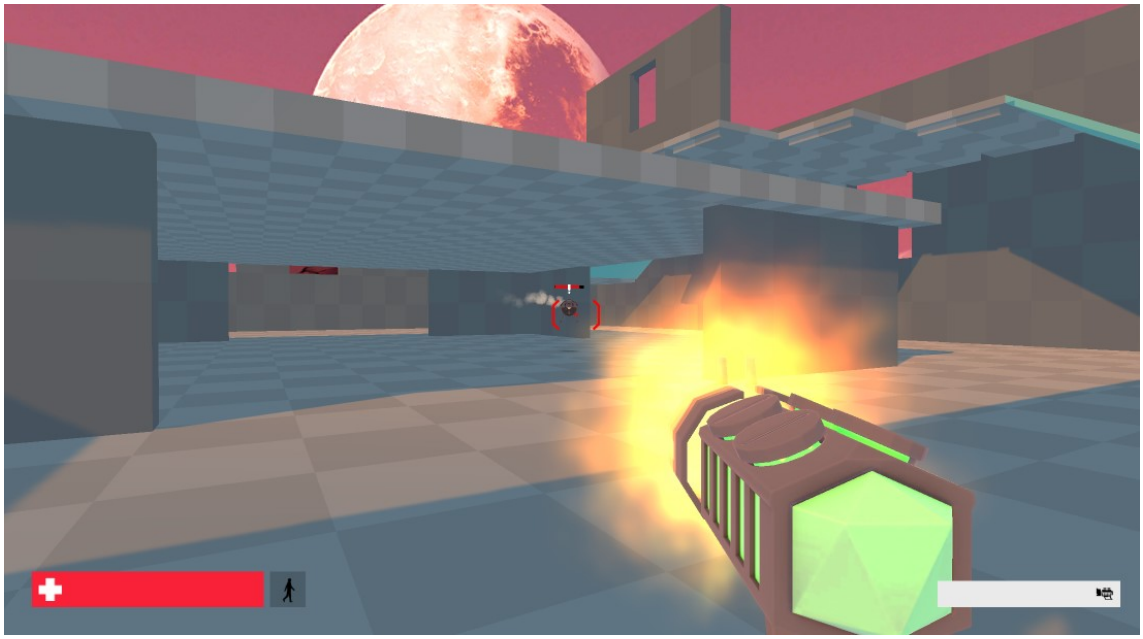


FIGURE 6.4 – Le joueur vise le robot ennemi, contrôlé par une IA
Le point d'exclamation au-dessus de l'ennemi indique qu'il a repéré le joueur

caméra à l'aide de la souris. Afin de réduire les paramètres du jeu, le joueur n'a accès qu'à une seule arme. Nous avons modifié le jeu pour en faire un jeu de tir en arène, c'est-à-dire que le joueur se retrouve dans une arène contre un adversaire. Le but du jeu est donc d'éliminer son adversaire.

L'adversaire en question est contrôlé par une IA que nous avons développée. Le comportement de base de l'IA consiste à patrouiller sur la carte et, lorsqu'elle voit le joueur, elle commence à le suivre et à lui tirer dessus. Chaque fois que le joueur bat l'ennemi, un nouvel ennemi réapparaît et nous remplissons la jauge de santé du joueur. Si le joueur meurt, nous détruisons l'ennemi, puis faisons réapparaître le joueur et un nouvel ennemi.

L'IA a différents paramètres qui changent en fonction de la difficulté. Nous pouvons changer la vitesse de déplacement, la vitesse de tir, la portée de tir et la portée de détection du joueur.

Nous avons ajouté notre modèle d'adaptation de la difficulté au jeu afin de suivre nos courbes de difficulté. Comme expliqué dans le chapitre 3, ce modèle nous permet de concevoir et de suivre toutes les courbes de difficulté et nécessite peu de points de données pour converger

vers une estimation optimale de la difficulté du jeu. Il peut être utilisé sur de nombreux types de jeux, permet à un développeur de régler la difficulté du jeu à n'importe quel niveau en deux minutes de temps de jeu. Afin d'estimer grossièrement la difficulté le plus rapidement possible, le modèle pilote une seule métavariable pour ajuster la difficulté du jeu. La difficulté du jeu dépend des variables du jeu qui sont la vitesse de déplacement de l'ennemi, la vitesse de tir, la portée de tir et la portée de détection. Il commence par un simple algorithme +/- δ pour rassembler quelques points de données, puis utilise la régression logistique pour estimer la probabilité d'échec des joueurs lorsque la plus petite quantité de données requise a été collectée.

L'expérience comporte trois phases. Nos participants devront dans un premier temps remplir un questionnaire sur leurs habitudes de jeu. Ce questionnaire est issu d'une expérimentation sur la confiance des joueurs faite par Constant[69]. Le questionnaire original contient des questions sur les habitudes de jeu, le profil d'auto-efficacité basé sur les échelles générales d'auto-efficacité et le profil d'aversion au risque de Holt [73, 74, 75]. Nous avons coupé la partie sur le profil d'aversion au risque, car dans l'étude de Constant, le joueur devait parier une monnaie du jeu à chaque nouveau challenge, ce qui n'est pas le cas de notre étude.

La deuxième partie est la session de jeu. Les participants devront jouer pendant au moins 10 minutes et jusqu'à 20 minutes. Au début de la session, nous choisissons une courbe de difficulté aléatoire entre nos quatre courbes. Pendant la session de jeu, le joueur affrontera un ennemi contrôlé par l'IA dans un combat d'arène. Au bout de cinq minutes, un bouton apparaît en haut de l'écran, demandant si le joueur s'ennuie. Si le joueur clique dessus, le jeu sélectionnera une autre courbe de difficulté. Après cinq minutes supplémentaires après le clic, le bouton réapparaît. Cliquer à nouveau sur le bouton mettra fin à la session de jeu. Nous informons les joueurs avant la session de jeu que le bouton est présent afin qu'ils ne puissent pas le manquer en jouant. En faisant cela, nous considérons que le temps de jeu est le reflet de la motivation des joueurs.

Quelle que soit la fin de la session de jeu, en cliquant sur les boutons ou en jouant 20 minutes, le joueur passera à la troisième phase. Il s'agit du questionnaire d'expérience de jeu ou GEQ, pensé par Ijsselsteijn, qui a été traduit en français [76]. Nous avons ajouté des questions

sur l'utilisation du bouton « s'ennuyer », afin de pouvoir faire un lien direct entre la courbe de difficulté donnée au joueur et la motivation à jouer.

Les séances d'expérimentation ont eu lieu au LeCNAM ENJMIN, une école de jeux vidéo. La plupart des étudiants jouent à des jeux vidéo, nous limitons donc la possibilité que les participants aient à apprendre à jouer à un FPS ou à jouer sur un ordinateur. Sur une session de jeu de 20 minutes, la découverte d'un jeu peut amener à une augmentation rapide du niveau de compétence du joueur, ce qui ne reflète pas l'expérience à proprement parler du jeu. On vise donc des joueurs pour qui cette phase de découverte des contrôles de manipulation en 3D à la première personne est déjà faite depuis longtemps.

6.4 Résultats

6.4.1 Données de jeu

67 participants ont joué à notre jeu de tir (55 hommes et 12 femmes) avec un âge moyen de 23 ans (σ = 3,24). Tous les participants ont joué pendant au moins 10 minutes et jusqu'à 20 minutes, avec une durée de jeu moyenne de 17 minutes et 40 secondes.

Nous avons dans un premier temps vérifié pour chaque participant si le modèle a fonctionné correctement. Pour cela, nous avons calculé la qualité de notre modèle pour chaque participant. Pour ce faire, nous calculons le ratio d'erreurs faites par le modèle pour chaque participant. Les erreurs du modèle correspondent aux tours de jeu où le modèle n'a pas pu utiliser la régression logistique. Il s'agit donc du nombre de tours avec erreur sur le nombre de tours total. Le taux d'erreur moyen est compris entre 0, où le modèle n'a aucune erreur, et 1 où le modèle ne fonctionne pas du tout. En utilisant une dispersion statistique pour analyser la qualité du modèle, nous avons décidé de supprimer les valeurs extrêmes, connaissant l'intervalle interquartile (IQR) de la dispersion statistique. Nous avons donc supprimé huit participants en raison de résultats de faible qualité du modèle pour ces participants. Nous avons également vérifié les performances des joueurs en calculant la difficulté moyenne de leurs essais, afin de détecter si certains joueurs n'avaient pas compris les règles du jeu, ou au contraire, s'ils avaient

trouvé un bug qui les mènerait à la victoire à chaque essai. Mais nous n'avons pas trouvé de données anormales.

Il est à noter que nous nous basons sur une évaluation indirecte de la motivation du joueur pour évaluer l'attrait des courbes. Nous calculons le temps de jeu des participants, une donnée que nous estimons proche de la motivation à jouer, comme montré par les travaux précédents de l'équipe[58].

Nous avons trois hypothèses à tester, à partir des données que nous avons collectées. Notre hypothèse 1 concerne la difficulté de base des courbes. Nous pensons que les participants apprécieront davantage le jeu sur une difficulté plus élevée, ce qui les mettra dans l'état de flow ou dans l'état d'éveil. Ces deux états devraient motiver les joueurs, c'est-à-dire, dans notre cas, les faire jouer plus longtemps. Nous nous attendons donc à ce que les participants jouent plus longtemps sur les deux courbes *moyennes*, qu'elles aient ou non des pics de difficulté.

Nous avons vérifié pour chaque participant le temps de jeu correspondant à la première courbe. Étant donné que le joueur peut cliquer sur un bouton pour nous dire quand il commence à s'ennuyer, nous utilisons le temps de jeu comme une estimation de la motivation à jouer. Comme le montre la figure 6.5, la plupart des participants ayant eu la courbe constante *moyenne* ont joué moins de temps, ce qui signifie qu'ils ont appuyé sur le bouton plus tôt, tandis que les participants avec les courbes *basses* avec ou sans pics et la courbe *moyennes* avec pics ont joué plus longtemps avant d'appuyer sur le bouton.

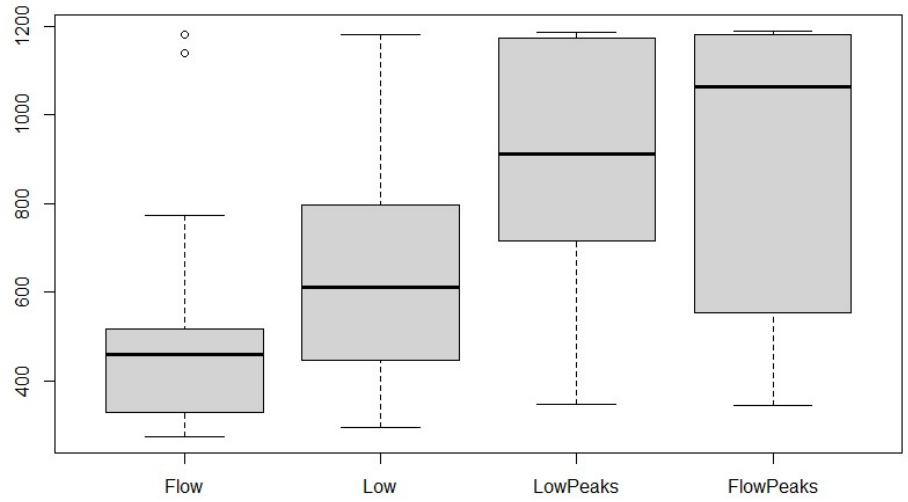


FIGURE 6.5 – Temps de jeu des participants sur la première courbe de difficulté

Nous avons ensuite effectué plusieurs tests de Wilcoxon sur ces données en testant à chaque fois le temps de jeu d'une courbe par rapport aux trois autres. Comme on peut le voir avec la table 6.1, les tests de Wilcoxon montrent que le temps de jeu moyen pour la courbe constante *moyenne* est significativement inférieur au temps de jeu moyen pour les courbes avec pics. Nous ne pouvons pas valider notre hypothèse 1, la courbe constante *moyenne* n'est pas préférée à la courbe constante *basse*. Dans cette expérience, une vision simple centrée sur le flow avec une difficulté à 0.5 n'amène pas du tout le temps de jeu le plus important.

	Basse constante	*Basse* pics	*Moyenne* constante
Basse pics	0.07350	X	X
Moyenne constante	0.09945	0.03032*	X
Moyenne pics	0.06604	0.88021	0.00784**

TABLE 6.1 – P-value des tests de Wilcoxon comparant le temps de jeu pour chaque paire de courbes

La courbe *moyenne* constante est significativement inférieure aux deux courbes avec des pics.

Notre hypothèse 2 concerne les courbes avec des pics de difficulté. Selon nous, ces courbes sont plus attrayantes que les courbes ayant une difficulté constante. Les joueurs semblent jouer plus longtemps sur les courbes à pics, comme le montre la figure 6.5, ce qui conforte notre hypothèse. Pour valider pleinement cette hypothèse, nous avons effectué un autre test de Wilcoxon en utilisant pour chaque courbe les données de temps de jeu et en regroupant les courbes ayant une difficulté constante et les courbes avec des pics. Nous obtenons une p-value $< 0{,}003$ pour ce test, rejetant l'hypothèse nulle, ce qui nous conforte dans notre hypothèse.

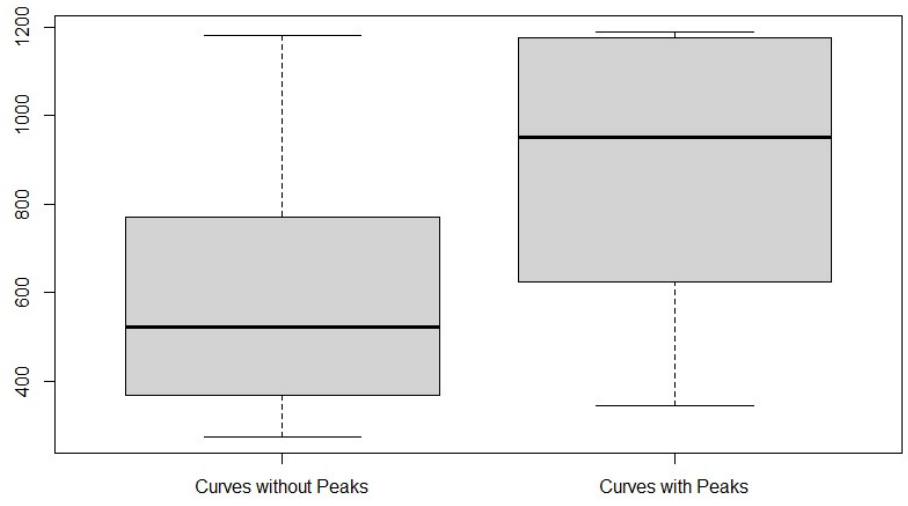


FIGURE 6.6 – Temps de jeu des participants, courbes avec pics contre courbes constantes
Les résultats sont significatifs avec une p-value du test de Wilcoxon de 0.00279. Cela signifie que les pics de difficulté ont un lien significatif avec le temps de jeu des joueurs dans notre expérimentation.

Nous avons ajouté une quatrième hypothèse : Les courbes *moyennes* sont plus apprécié que les courbes *basses*. Nous avons comparé deux à deux les courbes avec une difficulté *moyenne* et les courbes avec une difficulté *basse*. Comme nous pouvons le voir sur la figure 6.7, le résultat n'est pas celui que nous attendions. Nous avons fait un test de Wilcoxon pour vérifier si le niveau de base de la courbe de difficulté avait un impact sur le temps de jeu. Cependant, le test a donné une p-value de 0,16159, ce qui signifie que nous n'avons pas détecté de lien significatif entre le niveau de base de la courbe de difficulté et le temps de jeu des joueurs.

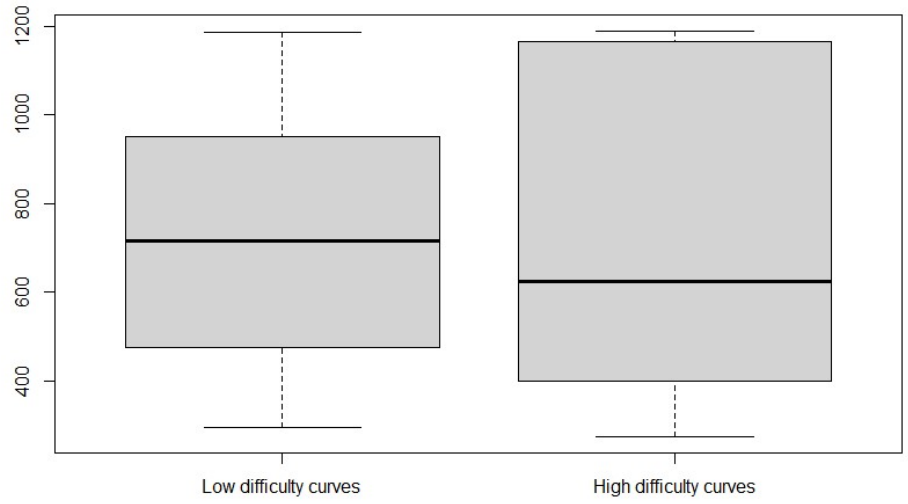


FIGURE 6.7 – Temps de jeu des participants, difficulté moyenne contre difficulté basse
Les résultats ne sont pas significatifs, avec une p-value du test de Wilcoxon de 0.16159. Cela signifie que la difficulté moyenne n'a pas de lien significatif avec le temps de jeu pour notre expérimentation.

Enfin, notre hypothèse 3 était que la courbes *moyenne* avec pics serait la plus appréciée des quatre courbes. La courbe *moyenne* avec pic est bien celle avec le temps de jeu le plus long. Cependant, nous ne pouvons pas affirmer qu'il s'agisse de la plus attrayante des quatre courbes.

Il est à noter de ces résultats que contrairement à nos attentes, la courbe *moyenne* constante ne surclasse pas les autres. Elle a obtenu la moyenne la plus basse et se détache significativement des deux courbes avec pics, mais de manière inverse à ce qu'on pouvait attendre. De plus, les courbes avec pics semblent être plus motivantes pour les joueurs que celles sans pics.

6.4.2 Données du questionnaire

Nous avons décidé de vérifier les données du GEQ afin de voir si les expériences des joueurs correspondent à nos résultats. Le GEQ comprend 7 composantes que l'on peut classer en deux catégories :

1. Composantes positives : Compétence, Immersion Sensorielle et Imaginative, Flow, Challenge, Affect positif

2. Composantes négatives : Tension/Contrariété, Effet Négatif

Concernant nos hypothèses, nous nous attendons à ce que le score dans les composantes positives les joueurs ayant une courbe avec des pics soit plus élevé que les joueurs ayant une courbe constante, ainsi qu'un score plus faible dans les composantes négatives.

En utilisant les données du GEQ, nous calculons le score de chaque composante pour chaque participant. Nous comparons chaque courbe avec les trois autres pour chaque composante. Nous obtenons des résultats significatifs sur les composantes compétence et affect positif pour les deux courbes avec pics en comparaison avec la courbe *moyenne* constante, comme le montre la table 6.2. Les participants jouant sur la courbe *moyenne* constante ont ressenti moins d'émotion positive avec une p-value de 0,0271 au test de Wilcoxon et moins de compétence avec une valeur p de 0,0485908 que les participants jouant sur la courbe de *moyenne* avec pics. Les autres résultats significatifs sont sur les composantes affect positif et compétence pour la courbe *moyenne* constante contre la courbe *basse* avec pics. Encore une fois, les participants ont ressenti moins de compétence sur la courbe *moyenne* constante, avec une p-value 0,0336 et moins d'affect positif avec une p-value de 0,0074. La table 6.2 regroupe les *location shift* pour chaque résultat significatif. Bien qu'on ne puisse pas affirmer que les courbes avec pics ont été préférées aux courbes constantes avec les résultats du questionnaire, la question se pose quant à l'utilisation de la courbe *moyenne* constante pour entraîner l'état de Flow, au vu de la baisse de qualité d'expérience significative entre cette dernière et les courbes avec pics.

Compétence	*Moyenne* constante	Affect positif	*Moyenne* constante
Basse pics	-5*	*Basse* pics	-5**
Moyenne pics	-3*	*Moyenne* pics	-3*

TABLE 6.2 – *Location shift* significative du test de Wilcoxon comparant les résultats du GEQ pour chaque couple de courbes
Pour les composantes Compétence et Affect positif, la courbe du Flow a un score significativement plus bas que les deux courbes avec des pics.

6.5 Discussion

Nos résultats ont confirmé que les participants préféraient les courbes de difficulté avec des pics de difficulté. Cependant, nos participants sont principalement des joueurs et ce sont tous des étudiants. Nous avons décidé de faire passer l'expérimentation dans une école de développement de jeux pour nous débarrasser de certains biais comme l'apprentissage de l'utilisation d'un clavier et d'une souris pour jouer à un jeu et l'apprentissage de la mécanique de base des FPS. Le résultat positif de notre expérimentation peut venir du fait que notre population cible pour l'expérimentation est proche de la population cible des jeux FPS. On peut dire que cette population préfère les courbes avec des pics de difficulté aux courbes ayant une difficulté constante. Mais nos résultats ne sont valables que pour cette population, et il faudrait répliquer cette expérimentation sur un public différent. Le public cible pourrait être les travailleurs plutôt que les étudiants. Nous voulons tester nos hypothèses sur une plus grande échelle de la population afin d'avoir vraiment une idée de la perception de la difficulté pour chaque groupe de la population.

Comme nous l'avons dit, nous avons utilisé un jeu de tir à la première personne, principalement parce que c'est un type de jeu basé sur la motricité, avec une courte période d'apprentissage, car les participants doivent pouvoir apprendre les quelques mécaniques du jeu, la mobilité du joueur, la gravité, la le taux de tir, le temps de rechargement et la carte du jeu en quelques minutes, ce qui leur permet de jouer au mieux de leur potentiel par la suite. Les jeux FPS sont connus pour être dynamiques et les joueurs de FPS sont très souvent des joueurs compétitifs. Notre résultat pourrait venir du fait que certains de nos participants sont des joueurs réguliers

de FPS, et qu'ils ont pris l'expérience comme une compétition. On ne peut pas affirmer qu'une même population de joueurs réagira de la même manière sur un jeu d'aventure comme Zelda, et tester la même population sur ce type de jeu serait un moyen de confirmer nos résultats.

De plus, le jeu que nous avons utilisé est un FPS d'arène de combat en un contre un, qui est une sous-catégorie des jeux FPS. Les joueurs apparaissent dans une petite arène avec un ennemi unique. La carte est généralement petite et il est facile de l'apprendre. Dans notre jeu, la mécanique est simplifiée, il y a qu'une seule arme et pas d'objets utilitaires. Les joueurs ne peuvent compter que sur leurs compétences et leur réactivité pour battre leur adversaire. Dans ce jeu, l'échec n'est pas punitif, car le joueur réapparaîtra instantanément après chaque échec. Comme on a pu le voir, un échec punitif peut entraîner une baisse de la motivation à jouer. Nos bons résultats peuvent donc également venir du fait que notre jeu n'est pas punitif. Si les joueurs meurent, ils ressuscitent instantanément et peuvent essayer de battre l'ennemi tout de suite. Refaire une expérimentation similaire avec par exemple un système de points qui se cumule quand le joueur bat l'ennemi, mais qui retombe a zéro s'il perd peut-être intéressant afin d'explorer cette composante de la difficulté.

6.6 Conclusion

Dans cette expérimentation, nous nous intéressons aux courbes de difficulté. De nombreux auteurs utilisent des courbes de difficulté afin d'améliorer la motivation des joueurs, mais peu étudient l'impact des courbes de difficulté sur les joueurs.

Pour notre expérimentation, nous avons proposé à une population d'étudiants en jeu vidéo de jouer à un FPS d'arène de combat. Ce jeu utilisait notre modèle d'adaptation dynamique de la difficulté afin d'adapter la difficulté au joueur. Chaque participant a joué avec l'une des quatre courbes de difficulté que nous avons dessiné.

Le résultat de cette expérimentation montre que les joueurs préfèrent les courbes de difficulté avec des pics aux courbes avec une difficulté constante. Il est à noter que nous pensions que la courbe *moyenne* constante serait plus appréciée, étant donné que cette courbe propose une

difficulté constante de 0,5, ce qui est comparable au modèle du Flow, alors que les résultats montrent alors qu'elle n'est pas du tout la plus appréciée. L'utilisation d'un modèle d'adaptation dynamique de la difficulté pour équilibrer parfaitement la difficulté pourrait réduire le temps de jeu des joueurs, et il est possible que les modèles d'adaptation dynamique de la difficulté réels créent de petites oscillations qui donnent un certain rythme au jeu. De plus, la difficulté n'est pas significative dans cette expérimentation.

Conclusion

Les travaux de recherche présentés dans cette thèse ont permis d'aboutir à un modèle de difficulté exploitable pour différents types de gameplays, capable de converger suffisamment rapidement à partir des données générées par le joueur au cours de la partie. Ce modèle nous a ensuite permis d'étudier plus précisément l'impact d'une modulation de la difficulté sur la motivation du joueur dans les jeux vidéo.

Nous avons dans un premier temps exploré l'état de l'art des recherches effectuées au sujet de la difficulté des jeux vidéo, ce qui nous a permis de mettre en lumière le manque d'un modèle capable d'ajuster la difficulté à un niveau choisi par le concepteur, tout en restant suffisamment léger pour s'affranchir d'un système de stockage centralisé des données joueurs.

Comme nous l'avons démontré, notre modèle respecte ce cahier des charges strict. En effet, *delta*-logit est capable de cibler n'importe quel niveau de difficulté, est utilisable sans données préalables et peut s'appliquer à un très grand nombre de gameplays. Le gameplay doit cependant respecter certaines contraintes : le designer doit être en mesure d'extraire des objectifs binaires poursuivis par le joueur. Plus ces objectifs seront à court terme et plus notre modèle pourra rapidement évaluer les capacités du joueur et adapter le jeu en conséquences.

Comme nous l'avons expliqué, *delta*-logit fonctionne grâce à deux algorithmes distincts. Ceci nous permet d'avoir une solution simple de repli lorsque aucune donnée n'est disponible ou que ces données ne permettent pas de profiler le joueur suffisamment précisément. Durant cette période, la plus courte possible, la difficulté reste adaptée, mais vers une valeur d'équilibre se rapprochant d'une probabilité d'échec de 50%. Dans les faits, cette valeur est rarement atteinte, car nous nous appuyons sur une courbe de difficulté croissante au début du jeu, qui nous permet de récolter suffisamment d'échantillons avant d'avoir atteint cette valeur d'équilibre. Nous pouvons ainsi rapidement activer le second algorithme, qui exploite une régression logistique pour viser une probabilité d'échec définie par la courbe de difficulté. Au prix donc d'une courte phase d'exploration et d'une vision simplifiée du gameplay, nous permettons au designer de contrôler assez rapidement et simplement sa courbe de difficulté.

Notre modèle répond ainsi particulièrement aux contraintes posées par le projet de recherche DysApp, qui a financé cette thèse. Dysapp propose de développer un ensemble de mini-jeux

capables d'entraîner et d'aider à diagnostiquer les enfants dyspraxiques. Ces mini-jeux sont construits autour d'actions complexes à réaliser par un joueur dyspraxique. Dysapp nous a offert l'opportunité de participer au design des mini-jeux. Nous avons également du développer divers prototypes d'une partie de ces jeux, ce qui a enrichi notre approche et nous a permis d'étudier l'utilisabilité de notre modèle dans différentes situations gameplay. Pour chaque mini-jeu, aussi bien sonore et rythmique qu'axé sur la planification visuelle ou motrice, nous avons su utiliser notre modèle pour adapter la difficulté des challenges proposés et proposé une solution finalement intégrée au jeu final. Nous sommes donc à la fois satisfait d'avoir pu contribuer à un projet de ce type et confiant dans la capacité de notre modèle à s'adapter à de nombreux gameplays.

Au-delà de notre capacité à utiliser notre modèle dans le cadre d'un jeu en cours de développement, nous avons cherché à valider scientifiquement notre modèle. Pour ce faire, nous avons modifié un jeu de tanks développé par Unity afin de faire jouer un joueur contre un tank contrôlé par une intelligence artificielle. Chaque tour de jeu est un succès si le joueur bat son adversaire et un échec si le joueur est vaincu. Le but de l'expérimentation était donc de faire jouer les joueurs avec le modèle qui proposait une difficulté en suivant une courbe de difficulté que nous avions dessiné pour l'occasion, commençant à une difficulté de 0,2 pendant trois tours, avec un pic a 0,5 puis un pic a 0,7. Pour les difficultés réelles visées de 0,2, 0,5 et 0,7, nous obtenons des fréquences d'échec respectives de 0,14, 0,55 et 0,74. Ces variances sont dues au fait que nous avons un paramètre de difficulté discontinu. Il s'agit du nombre de tanks ennemis qui apparaissent. Avec un θ en dessous de 0,5, le modèle ne fait apparaître qu'un seul tank. Lorsque θ est égal ou supérieur à 0,5, deux tanks ennemis apparaissent. Cela crée un pic de difficulté supplémentaire à des difficultés plus élevées, ce qui peut expliquer l'écart entre la difficulté voulue et la fréquence d'échec obtenue. Ces résultats sont satisfaisants et valident l'utilisation de notre modèle pour le projet DysApp, et ont fait l'objet d'une publication présentée lors de la conférence *International Conference on Entertainment Computing* 2018 à Arequipa, Peru.

Une fois le modèle validé, nous l'avons intégré, avec l'aide de notre partenaire Tralalère, au projet DysApp. Le jeu a ensuite été déployé en école. Cela nous a permis récolter un maximum

de données sur l'utilisation du modèle. Cependant, le temps de latence entre les nouvelles mises à jour du jeu et le déploiement de ces mises à jour en école fut beaucoup trop important. Certains bugs étaient encore présents dans le jeu lors du déploiement en école, bugs qu'il était nécessaire de corriger afin de récupérer des données cohérentes et utilisables. Et bien que la correction et la mise à jour d'une nouvelle version du logiciel côté Tralalère fut rapide, le déploiement de cette nouvelle version du logiciel fut bien plus lente dans les écoles partenaires. Ces problèmes liés au caractère multipartenaire du projet sont regrettables, mais j'ai grand espoir que le projet soit mené à terme afin de pouvoir déployer l'application en école et permettre à de nombreux élèves en difficulté d'améliorer leur scolarité.

Nous avions souhaité profiter de cette expérimentation a grande échelle afin d'expérimenter l'attrait des enfants pour la difficulté. Pour ce faire, nous avons dessiné trois courbes de difficulté, que nous avions aléatoirement assigné à l'un des trois mini-jeux disponibles sur la dernière version du logiciel du jeu, afin de voir s'il y avait un lien entre le temps de jeu et la courbe de difficulté choisie. Cependant, les problèmes de déploiement des nouvelles versions du logiciel en école nous ont forcés à monter notre propre expérimentation.

Notre seconde expérimentation, centrée donc sur l'étude des courbes de difficulté, consiste à proposer a nos participants un jeu de type shooter à la première personne avec pour chaque participant une courbe de difficulté choisie aléatoirement parmi quatre courbes possibles. Ces quatre courbes ont été modélisées à la suite de nos recherches le sujet. Les résultats de cette expérimentation nous montrent que des courbes de difficultés avec des pics de difficultés sont préférées. En effet, les joueurs ayant eu une courbe de difficulté avec des pics ont joué plus longtemps que ceux ayant eu une courbe de difficulté plate. De plus, les résultats au questionnaire montrent que pour les composantes de compétence et d'affect positif, les courbes avec des pics de difficulté ont de meilleures notes que la courbe *moyenne* constante. Ces résultats valident donc une de nos hypothèses, selon laquelle les courbes avec pics de difficulté seraient préférées par les joueurs. Mais de manière plus surprenante, il semble que la courbe *moyenne* constante soit loin d'être la courbe la plus appréciée des quatre courbes. Cette courbe correspond pourtant à une interprétation directe des principes basiques proposés par le modèle de flow, ce qui

montre la complexité de l'étude des courbes de difficulté.

6.6.1 Perspective

Les travaux de cette thèse ouvrent plusieurs pistes que j'aurais aimé suivre si j'en avais eu le temps. L'une de celle-ci consiste à pousser plus loin l'expérimentation sur les courbes de difficulté. Dans celle-ci nous nous sommes arrêté sur un type de jeu, un FPS, avec une population de participant adulte et en grande majorité habituée à jouer régulièrement. L'approfondissement de cette expérimentation peut être vue de deux façons possibles. La première serait de conserver le type de jeu, mais de modifier la population de participant afin d'avoir des enfants et des adolescents joueurs et non joueurs ainsi que des adultes non-joueurs afin de vérifier nos hypothèses sur des populations différentes. Les étudiants d'une école de jeux vidéo sont pour la majeure partie des joueurs expérimentés, ce qui peut en partie expliquer pourquoi la courbe de difficulté proposant la difficulté la plus importante ait eu le plus de temps de jeu.

Une autre direction serait de modifier le type de jeu afin de savoir si les joueurs préfèrent une difficulté différente en fonction du jeu auquel ils jouent. En effet, nous avons utilisé un FPS pour cette expérimentation. Il s'agit d'un type de jeu très compétitif et il est facile de s'ennuyer si le niveau de l'adversaire est bien en deçà du nôtre. Les jeux d'aventure, les derniers titres de la série des Mario ou des Zelda par exemple, sont des jeux plus contemplatifs où le joueur peut simplement se déplacer dans le monde, profiter de l'environnement et progresser à son rythme. Contrairement aux FPS, ce type de jeu ne pousse pas à la compétition. De plus, comme le montre Yee, en fonction de la tranche d'âge du joueur, ce dernier est plus attiré par certains attraits d'un jeu[68]. Yee montre par exemple que les plus jeunes joueurs préfèrent les jeux qui offrent de la compétition. Il est donc logique de penser que les préférences en termes de difficulté des joueurs sont également impactées par le style de jeu.

Nous avions également mis en place une autre expérimentation concernant la perception de la difficulté, mais nous n'avons pas été jusqu'au bout. Il s'agissait de faire jouer nos participants au jeu utilisé lors de la première expérimentation mais, à la fin de chaque tour, nous demandons si le participant a trouvé le tour actuel plus dur, plus simple ou équivalant au tour précédant.

L'intérêt de cette expérimentation serait de voir à quel point une personne peut percevoir des oscillations de difficulté. Cette étude permettrait de soutenir l'idée qu'un joueur a une perception assez vague de la difficulté du jeu auquel il joue. Constant montre que très souvent les joueurs surestiment leurs chances de surmonter un challenge difficile[11]. Nous nous sommes appuyés sur ces travaux afin d'intégrer une valeur d'exploration à notre modèle, dans le but d'explorer un échantillon plus large de difficulté, mais il serait utile de chiffrer réellement une erreur de précision dans la perception des différences de difficulté afin d'ajuster ce paramètre.

Bibliographie

[1] M. Zohaib, "Dynamic difficulty adjustment (dda) in computer games : A review," *Advances in Human-Computer Interaction*, vol. 2018, 2018.

[2] J. K. Olesen, G. N. Yannakakis et J. Hallam, "Real-time challenge balance in an rts game using rtneat," dans *Computational Intelligence and Games, 2008. CIG'08. IEEE Symposium On.* IEEE, 2008, p. 87–94.

[3] C. Pedersen, J. Togelius et G. N. Yannakakis, "Modeling player experience in super mario bros," dans *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on.* IEEE, 2009, p. 132–139.

[4] ——, "Modeling player experience for content creation," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, n°. 1, p. 54–67, 2010.

[5] N. Shaker, G. Yannakakis et J. Togelius, "Towards automatic personalized content generation for platform games," dans *Sixth artificial intelligence and interactive digital entertainment conference*, 2010.

[6] L. V. Carvalho, A. Moreira, V. Vicente Filho, M. Albuquerque et G. L. Ramalho, "A generic framework for procedural generation of gameplay sessions," *Proceedings of the SB Games*, 2013.

[7] P. Spronck, I. Sprinkhuizen-Kuyper et E. Postma, "Difficulty scaling of game ai," dans *Proceedings of the 5th International Conference on Intelligent Games and Simulation (GAME-ON 2004)*, 2004, p. 33–37.

[8] ——, "Online adaptation of game opponent ai with dynamic scripting," *International Journal of Intelligent Games and Simulation*, vol. 3, n°. 1, p. 45–53, 2004.

[9] G. Andrade, G. Ramalho, H. Santana et V. Corruble, "Extending reinforcement learning to provide dynamic game balancing," dans *Proceedings of the Workshop on Reasoning, Representation, and Learning in Computer Games, 19th International Joint Conference on Artificial Intelligence (IJCAI)*, 2005, p. 7–12.

[10] J. Hagelback et S. J. Johansson, "Measuring player experience on runtime dynamic difficulty scaling in an rts game," dans *Computational Intelligence and Games, 2009. CIG 2009. IEEE Symposium on.* IEEE, 2009, p. 46–52.

[11] T. Constant, G. Levieux, A. Buendia et S. Natkin, "From objective to subjective difficulty evaluation in video games," dans *IFIP Conference on Human-Computer Interaction.* Springer, 2017, p. 107–127.

[12] N. Hocine et A. Gouaïch, "Therapeutic games' difficulty adaptation : An approach based on player's ability and motivation," dans *Computer Games (CGAMES), 2011 16th International Conference on.* IEEE, 2011, p. 257–261.

[13] U. de Poitiers. (2021) Dysapp. Accessed : 2021-12-03. [En ligne]. Disponible : https://dysapp.prd.fr/

[14] T. Allart, G. Levieux, M. Pierfitte, A. Guilloux et S. Natkin, "Difficulty influence on motivation over time in video games using survival analysis," dans *Proceedings of the 12th International Conference on the Foundations of Digital Games.* ACM, 2017, p. 2.

[15] E. A. Locke et G. P. Latham, "Goal setting theory, 1990." 2013.

[16] Y. Fried et L. H. Slowik, "Enriching goal-setting theory with time : An integrated approach," *Academy of management Review*, vol. 29, n°. 3, p. 404–422, 2004.

[17] E. A. Locke et G. P. Latham, "Goal setting theory : The current state." 2013.

[18] A. Bandura, "Self-efficacy : toward a unifying theory of behavioral change." *Psychological review*, vol. 84, n°. 2, p. 191, 1977.

[19] A. Flammer, "Self-efficacy," 2001.

[20] P. A. Heslin et U.-C. Klehe, "Self-efficacy," *Encyclopedia Of Industrial/Organizational Psychology, SG Rogelberg, ed*, vol. 2, p. 705–708, 2006.

[21] J. E. Maddux et J. T. Gosselin, *Self-efficacy.* The Guilford Press, 2012.

[22] A. L. Lannie et B. K. Martens, "Effects of task difficulty and type of contingency on students'allocation of responding to math worksheets," *Journal of Applied Behavior Analysis*, vol. 37, n°. 1, p. 53–65, 2004.

[23] J. Piaget, R. Garcia, P. Davidson, P. M. Davidson et J. Easley, *Toward a logic of meanings.* Psychology Press, 2013.

[24] L. S. Vygotsky, *Mind in society : The development of higher psychological processes.* Harvard university press, 1980.

[25] E. E. Gickling et D. L. Armstrong, "Levels of instructional difficulty as related to on-task behavior, task completion, and comprehension," *Journal of Learning Disabilities*, vol. 11, n°. 9, p. 559–566, 1978.

[26] K. A. Orvis, D. B. Horn et J. Belanich, "The roles of task difficulty and prior videogame experience on performance and motivation in instructional videogames," *Computers in Human behavior*, vol. 24, n°. 5, p. 2415–2433, 2008.

[27] T. W. Britt, "The effects of identity-relevance and task difficulty on task motivation, stress, and performance," *Motivation and Emotion*, vol. 29, n°. 3, p. 189–202, 2005.

[28] B. Hughes, H. J. Sullivan et M. Lou Mosley, "External evaluation, task difficulty, and continuing motivation," *The Journal of Educational Research*, vol. 78, n°. 4, p. 210–215, 1985.

[29] J. Nakamura et M. Csikszentmihalyi, "The concept of flow," dans *Flow and the foundations of positive psychology.* Springer, 2014, p. 239–263.

[30] B. Weiner, H. Heckhausen et W.-U. Meyer, "Causal ascriptions and achievement behavior : A conceptual analysis of effort and reanalysis of locus of control." *Journal of personality and social psychology*, vol. 21, n°. 2, p. 239, 1972.

[31] G. Borg, O. Bratfisch et S. Dorni'c, "On the problems of perceived difficulty," *Scandinavian journal of psychology*, vol. 12, n°. 1, p. 249–260, 1971.

[32] D. Delignières et J.-P. Famose, "Perception de la difficulté, entropie et performance," *Science & sports*, vol. 7, n°. 4, p. 245–252, 1992.

[33] P. Vorderer, T. Hartmann et C. Klimmt, "Explaining the enjoyment of playing video games : the role of competition," dans *Proceedings of the second international conference on Entertainment computing*, 2003, p. 1–9.

[34] J. Feil et M. Scattergood, *Beginning game level design.* Thomson Course Technology, 2005.

[35] N. Lazzaro, "Why we play games : Four keys to more emotion without story," 2004.

[36] T. W. Malone, "Heuristics for designing enjoyable user interfaces : Lessons from computer games," dans *Proceedings of the 1982 conference on Human factors in computing systems.* ACM, 1982, p. 63–68.

[37] P. Sweetser et P. Wyeth, "Gameflow : a model for evaluating player enjoyment in games," *Computers in Entertainment (CIE)*, vol. 3, n°. 3, p. 3–3, 2005.

[38] R. M. Ryan, C. S. Rigby et A. Przybylski, "The motivational pull of video games : A self-determination theory approach," *Motivation and emotion*, vol. 30, n°. 4, p. 344–360, 2006.

[39] C. Klimmt, C. Blake, D. Hefner, P. Vorderer et C. Roth, "Player performance, satisfaction, and video game enjoyment," dans *ICEC*, 2009, p. 1–12.

[40] E. Adams, *Fundamentals of game design.* Pearson Education, 2014.

[41] J. Juul, "The game, the player, the world : Looking for a heart of gameness," *PLURAIS-Revista Multidisciplinar*, vol. 1, n°. 2, 2010.

[42] ——, "Fear of failing ? the many meanings of difficulty in video games," *The video game theory reader*, vol. 2, n°. 237-252, 2009.

[43] ——, *The art of failure : An essay on the pain of playing video games.* Mit Press, 2013.

[44] G. R. Loftus et E. F. Loftus, *Mind at play ; The psychology of video games.* Basic Books, Inc., 1983.

[45] C. Linehan, G. Bellord, B. Kirman, Z. H. Morford et B. Roche, "Learning curves : analysing pace and challenge in four successful puzzle games," dans *Proceedings of the first ACM SIGCHI annual symposium on Computer-human interaction in play*, 2014, p. 181–190.

[46] C. Pruett, "Defining the all-important difficulty curve," *The Journal of Education, Community, and Value*, vol. 8, n°. 1, 2008.

[47] J. T. Alexander, J. Sear et A. Oikonomou, "An investigation of the effects of game difficulty on player enjoyment," *Entertainment computing*, vol. 4, n°. 1, p. 53–62, 2013.

[48] D. Ang et A. Mitchell, "Comparing effects of dynamic difficulty adjustment systems on video game experience," dans *Proceedings of the Annual Symposium on Computer-Human Interaction in Play*. ACM, 2017, p. 317–327.

[49] O. Yasuyuki et S. Katsuhisa. (2003) Racing game program and video game device. Accessed : 2018-09-18. [En ligne]. Disponible : https://patents.google.com/patent/US7278913

[50] A. M. Colwell et F. G. Glavin, "Colwell's castle defence : A custom game using dynamic difficulty adjustment to increase player enjoyment," *arXiv preprint arXiv :1806.04471*, 2018.

[51] S. Mader, G. Levieux et S. Natkin, "A game design method for therapeutic games," dans *Games and Virtual Worlds for Serious Applications (VS-Games), 2016 8th International Conference on*. IEEE, 2016, p. 1–8.

[52] S. Demediuk, M. Tamassia, W. L. Raffe, F. Zambetta, X. Li et F. Mueller, "Monte carlo tree search based algorithms for dynamic difficulty adjustment," dans *2017 IEEE Conference on Computational Intelligence and Games (CIG)*, Aug 2017, p. 53–59.

[53] M. Ishihara, S. Ito, R. Ishii, T. Harada et R. Thawonmas, "Monte-carlo tree search for implementation of dynamic difficulty adjustment fighting game ais having believable behaviors," dans *2018 IEEE Conference on Computational Intelligence and Games (CIG)*. IEEE, 2018, p. 1–8.

[54] R. Vicencio-Moreira, R. L. Mandryk et C. Gutwin, "Now you can compete with anyone : Balancing players of different skill levels in a first-person shooter game," dans *Proceedings*

of the 33rd Annual ACM Conference on Human Factors in Computing Systems. ACM, 2015, p. 2255–2264.

[55] S. Xue, M. Wu, J. Kolen, N. Aghdaie et K. A. Zaman, "Dynamic difficulty adjustment for maximized engagement in digital games," dans *Proceedings of the 26th International Conference on World Wide Web Companion.* International World Wide Web Conferences Steering Committee, 2017, p. 465–471.

[56] M. M. Khajah, B. D. Roads, R. V. Lindsey, Y.-E. Liu et M. C. Mozer, "Designing engaging games using bayesian optimization," dans *Proceedings of the 2016 CHI conference on human factors in computing systems*, 2016, p. 5571–5582.

[57] A. Zook et M. O. Riedl, "A temporal data-driven player model for dynamic difficulty adjustment." dans *AIIDE*, 2012.

[58] M.-V. Aponte, G. Levieux et S. Natkin, "Measuring the level of difficulty in single player video games," *Entertainment Computing*, vol. 2, n°. 4, p. 205–213, 2011.

[59] E. Byrne, *Game level design.* Charles River Media Boston, 2005, vol. 6.

[60] R. Vazquez. (2011) How tough is your game ? creating difficulty graphs. Accessed : 2020-05-05. [En ligne]. Disponible : https://www.gamasutra.com/view/feature/134917/how_tough_is_your_game_creating_.php

[61] D. Strachan. (2018) Making difficulty curves in games. Accessed : 2020-09-16. [En ligne]. Disponible : http://www.davetech.co.uk/difficultycurves

[62] Frazer. (2017) Level design and difficulty curves. Accessed : 2020-09-16. [En ligne]. Disponible : http://www.teaboygames.com/2017/06/14/level-design-and-difficulty-curves/

[63] S. Miyamoto, T. Nakago et T. Tezuka, "The legend of zelda," *Nintendo : Kyoto, Japan*, 1986.

[64] J. Concato, P. Peduzzi, T. R. Holford et A. R. Feinstein, "Importance of events per independent variable in proportional hazards analysis i. background, goals, and general strategy," *Journal of clinical epidemiology*, vol. 48, n°. 12, p. 1495–1501, 1995.

[65] J. McCaffrey. (2012) Test run - coding logistic regression with newton-raphson. Accessed : 2018-09-19. [En ligne]. Disponible : https://msdn.microsoft.com/en-us/magazine/jj618304.aspx

[66] Unity. (2015) Tanks tutorial. Accessed : 2018-09-19. [En ligne]. Disponible : https://unity3d.com/fr/learn/tutorials/s/tanks-tutorial

[67] G. Toussaint, "The geometry of musical rhythm," dans *Japanese Conference on Discrete and Computational Geometry*. Springer, 2004, p. 198–212.

[68] N. Yee, "The gamer motivation profile : What we learned from 250,000 gamers," dans *Proceedings of the 2016 Annual Symposium on Computer-Human Interaction in Play*, 2016, p. 2–2.

[69] T. Constant et G. Levieux, "Dynamic difficulty adjustment impact on players' confidence," dans *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, ser. CHI '19, 2019, p. 463 :1–463 :12.

[70] W. Rao Fernandes et G. Levieux, "δ-logit : Dynamic difficulty adjustment using few data points," dans *Joint International Conference on Entertainment Computing and Serious Games*. Springer, 2019, p. 158–171.

[71] A. Gavin. (2011) Making crash bandicoot part 6. Accessed : 2018-09-06. [En ligne]. Disponible : https://all-things-andy-gavin.com/2011/02/07/making-crash-bandicoot-part-6

[72] Unity. Fps microgame. Accessed : 2020-09-15. [En ligne]. Disponible : https://learn.unity.com/project/fps-template

[73] G. Chen, S. M. Gully et D. Eden, "Validation of a new general self-efficacy scale," *Organizational research methods*, vol. 4, n°. 1, p. 62–83, 2001.

[74] A. Bandura *et al.*, "Guide for constructing self-efficacy scales," *Self-efficacy beliefs of adolescents*, vol. 5, n°. 1, p. 307–337, 2006.

[75] C. A. Holt et S. K. Laury, "Risk aversion and incentive effects," *American economic review*, vol. 92, n°. 5, p. 1644–1655, 2002.

[76] W. IJsselsteijn, Y. De Kort et K. Poels, "The game experience questionnaire," *Eindhoven : Technische Universiteit Eindhoven*, p. 3–9, 2013.

le cnam

William RAO FERNANDES

Etude des courbes de difficulté dans les jeux vidéo avec forte variabilité du niveau des joueurs et une faible quantité de donnée

HESAM UNIVERSITÉ

Résumé : Ce travail de recherche porte sur l'étude des courbes de difficulté dans les jeux vidéo ainsi que sur l'adaptation dynamique de la difficulté des jeux vidéo avec forte variabilité du niveau des joueurs et une faible quantité de données. Nous nous intéressons à l'adaptation de la difficulté car la difficulté des jeux vidéo est un facteur de plaisir et de motivation. Nous avons développé un modèle générique, utilisable dans n'importe quel jeu. De plus, notre modèle permet de cibler n'importe quel niveau de difficulté. Enfin, notre modèle fonctionne sans données au préalable. Ce modèle a été testé lors d'une expérimentation scientifique. Nous avons mené une autre expérimentation sur des courbes de difficulté. Nous avons testé quatre courbes de difficulté différentes. Ces travaux peuvent aboutir à une meilleure connaissance des besoins de chaque catégorie de joueurs grâce à un modèle simple et utilisable par une grande variété de jeux vidéo différents.

Mots clés : Jeu Vidéo, Difficultée, Courbe de Difficulté, Modèle de Difficulté, Adaptation de la Difficulté, Motivation, Modélisation de Joueurs.

Abstract : This research work focuses on the study of difficulty curves in video games as well as on dynamic adaptation of video games' difficulty with high variability in the players' level and a small amount of data. We are interested in the difficulty's adaptation because video games' difficulty is a factor of pleasure and motivation. We have developed a generic model that can be used in any game. In addition, our model can target any level of difficulty. Finally, our model works without prior data. This model has been tested in a scientific experiment. We carried out an experimentation on difficulty curves. We tested four different difficulty curves. This work can lead to a better understanding of the needs of each category of players thanks to a simple model that can be used by a wide variety of different video games.

Keywords : Video Game, Difficulty, Difficulty Curve, Difficulty Model, Difficulty Adaptation, Motivation, Player Modeling