Square Jump: joc de plataformes amb nivells de dificultat adaptativa

Jordi Morillo Montes

Resum—Aquest article presenta un algorisme generador de nivells adaptatius per a un joc de plataformes en 2D anomenat Square Jump. El joc proposat consisteix en un senzill joc de tipus *runner* on un personatge amb aparença de quadrat corre constantment i el jugador ha de fer-lo saltar o lliscar per evitar diferents obstacles. El joc té un generador de nivells que crea nivells on la dificultat s'ajusta en funció del rendiment del jugador. En altres paraules, el generador de nivells permet la creació de nivells dissenyats específicament per a les pròpies habilitats del jugador. Això s'aconsegueix utilitzant primer agents com a representacions de les característiques reals del jugador humà i després ajustant la dificultat del següent nivell presentat al jugador. El treball presentat en aquest article proporciona una solució per a un desenvolupament més eficient dels videojocs. Amb l'àmplia varietat de jugadors que podem trobar actualment, cadascun d'ells amb una experiència diferent en els jocs, Square Jump és un clar exemple d'un joc dirigit a usuaris amb diferents nivells d'habilitats i preferències.

Paraules clau— Dynamic Difficulty Adjustment, intel·ligència artificial, joc de plataformes, videojocs

Abstract— This paper presents an adaptative level generator algorithm for a 2D platformer game called Square Jump. The proposed game consists of a simple runner game where a character with the appearence of a square is constantly running and the player must make it jump or slide to avoid different obstacles. The game has a level generator that creates levels in which the difficulty is adjusted based on the player's performance. In other words, the level generator allows the creation of levels specifically designed for the player's own abilities. This is achieved by first using agents as representations of real human player's characteristics and then adjusting the difficulty of the next level presented to the player. The work presented in this paper provides a solution for a more eficient development of videogames. With the wide variety of players we can find nowadays, each of them with different experience in games, Square Jump is a clear exemple of a game addressed to users with differing levels of skills and preferences.

Index Terms— Artificial intelligence, Dynamic Difficulty Adjustment, platform games, videogames

1 Introducció

QUAN algú juga a un joc, el seu objectiu principal és divertir-se. Els desenvolupadors de videojocs han de dissenyar els seus jocs perquè siguin divertits. Però, què fa que un joc sigui divertit? El concepte de la diversió en els jocs ha estat objecte d'estudi de nombroses investigacions durant els darrers anys. Hi ha una sèrie de teories en l'àmbit de la psicologia i també dels videjocs centrades en descriure com és l'experiència de jugar a un joc. Gairebé tots coincideixen en què els videojocs són avorrits quan són massa fàcils i frustrants si són massa difícils. Tenint en compte que hi ha jugadors de tots els nivells d'experiència, que van des dels que mai no han jugat a cap joc fins a competidors professionals d'eSports [1], crear un joc adreçat per a tots ells es converteix en una tasca que no és gens trivial.

Fixar un nivell de dificultat únic per a cada jugador no és una solució possible. Una alternativa és permetre al jugador triar un nivell de dificultat. Aquest mètode és el més utilitzat avui en dia, però té alguns inconvenients, com ara tenir un nombre limitat de nivells de dificultat que no s'ajustin realment a l'habilitat del jugador i, en conseqüència, el jugador pot sentir que no progressa i acaba abandonant el joc. Afortunadament, durant l'última dècada han aparegut tècniques de Dynamic Difficulty Adjustment (DDA), una àrea de recerca emergent, que té com a objectiu desenvolupar un sistema automatitzat que manté al jugador compromès i desafiat en tot moment. En aquest treball es farà servir alguna d'aquestes tècniques de DDA per fer que els nivells del nostre joc tinguin dificultat adaptativa.

Una altra motivació d'aquest projecte és el fet de poder generar nivells de forma automàtica amb Procedural Level Generation (PLG) sense necessitat de fer-los a mà [2]. La generació dels nivells tindrà en compte la dificultat ajustada mitjançant DDA, de manera que cada nivell serà construït amb un nivell de dificultat adaptat a les habilitats del jugador. A més, els nivells seran tan divertits com els creats a mà per un dissenyador de nivells i podran generar un nombre infinit de nivells diferents només variant els paràmetres que canvien el procés de generació.

- *E-mail de contacte: jordi.morillo@e-campus.uab.cat.*
- *Menció realitzada: Tecnologies de la informació.*
- *Treball tutoitzat per: Ana Oropesa Física.*
- *Curs 2020/2021.*

Square Jump pretén ser un joc desenvolupat amb la combinació d'algorismes Dynamic Difficulty Adjustment i Procedural Level Generation perquè qualsevol jugador, sigui quin sigui el seu nivell d'experiència, se senti desafiat i es diverteixi jugant el màxim temps possible.

2 OBJECTIUS

L'objectiu general d'aquest projecte és que l'experiència d'un usuari al jugar a Square Jump es vegi gratificada per la seva modulació de dificultat adaptativa. Al mateix temps, el projecte consta d'altres objectius específics que s'enumeren a continuación:

1. Abans de la fase d'estudi, obtenir els factors de dificultat que influeixen a fer que un joc sigui interessant per a un usuari mitjançant una investigació sobre tres estudis que tractin sobre la teoria de la diversió. Aquest punt ajudarà a entendre com afecta el nivell de dificultat en l'experiència d'un usuari a l'hora de jugar a un videojoc i, a més a més, permetrà definir de forma més exacta la motivació del projecte.

2. Abans de la fase de desenvolupament, obtenir un llista dels quatre jocs del gènere autorunner més populars del mercat juntament amb l'estratègia per adaptar la dificultat que segueixen. Aquest objectiu servirà per entendre quines són les característiques i mecanismes més utilitzats per escollir el nivell de dificultat en els autorunners. D'aquesta manera es podrà començar a pensar sobre com serà el joc que es crearà.

3. Triar quina estratègia de dificultat adaptativa es farà servir mitjançant un estudi dels conceptes Dynamic Difficulty Adjustment i Procedural Level Generation. Es triarà l'estratègia abans del començament de la fase de desenvolupament.

4. El quart objectiu serà assolir la finalització del desenvolupament del joc mitjançant la creació del personatge, l'escenari i la mecànica principal, a més de 5 obstacles que el personatge haurà de superar. Això es farà seguint els passos de la metodologia Kanban de gestió de projectes, complint els terminis de lliuraments i desenvolupament de cada mòdul segons les dates indicades en el diagrama de Gantt. S'estima que la implementació del joc estigui feta abans de la fase del disseny d'un agent que modeli el comportament del jugador.

5. El cinquè objectiu consistirà en l'aplicació d'algorismes genètics per construir un sistema que sigui capaç d'adaptar la dificultat del joc mitjançant la modelació de les característiques d'un jugador. Aquest objectiu s'haurà d'assolir abans de la fase de proves del joc.

6. Analitzar l'eficàcia i funcionament del Square Jump mitjançant una enquesta realitzada a un mínim de 15 persones, on s'obtindran dades del grau de satisfacció dels usuaris amb el nivell de dificultat del joc. Aquesta tasca haurà d'estar feta abans de l'obtenció de resultats finals del projecte.

3 METODOLOGIA

La metodologia escollida per dur a terme el projecte ha estat Kanban [3] aplicada a Trello [4], ja que la combinació d'aquestes dues tècniques permet organitzar les tasques del projecte de manera efectiva i fer un seguiment dels objectius proposats.

3.1 Kanban

Kanban [3] és una metodologia àgil amb l'objectiu de gestionar de manera general com es van completant les tasques d'un projecte. El terme Kanban [3] és una paraula japonesa que significa "targetes visuals", on Kan és "visual" i Ban correspon a "targeta". Per tant, podem deduir que la metodologia utilitza targetes per a gestionar la realització de determinats processos i tasques.

L'aplicació del mètode Kanban [3] implica la generació d'un tauler de tasques que permet millorar el flux de treball i assolir un ritme sostenible [5]. Originalment, per crear el tauler es feia servir una pissarra blanca que es dividia en columnes i files. Cada columna representa l'estat en què es troben les tasques i les files corresponen als diferents tipus d'activitats que s'han de dur a terme. En general, un tauler Kanban [3] es divideix en tres seccions bàsiques:

- TO DO: és la fase "Per fer", que correspon a les tasques pendents per realitzar. Les tasques s'aniran afegint en funció de com es desenvolupi el projecte i les necessitats de cada moment.

- DOING: és la tasca "En procés", que són les tasques que estan en procés i pendents per tancar.

- DONE: és la tasca "Fet", que són les tasques acabades i testejades del projecte.

3.2 Trello

Com s'ha mencionat anteriormente, originalmente el tauler Kanban [3] es construïa sobre una pissarra blanca on s'enganxaven els post-its que representaven les tasques. Avui en dia, però, existeixen múltiples solucions de targetes Kanban [3] digitals més practiques i accesibles a nivel global que són perfectes tant per als equips remots com per als equips que desenvolupen la seva activitat en el lloc on es realitza el projecte. Una d'aquetes solucions és Trello [4].

Trello [4] és una aplicació basada en el mètode Kanban [3] que serveix per gestionar projectes de forma col·laborativa mitjançant taulers virtuals compostos de llistes de tasques en forma de columnes. És perfecta per a

la gestió de projectes, ja que es poden representar diferents estats de les tasques i compartir-les amb totes les persones que formin part del projecte. Com que disposa d'una versió gratuïta i es tracta d'una aplicació molt simple i intuïtiva, s'ha decidit fer servir Trello [4] per al projecte. Un altre factor que ha ajudat a prendre la decisió és que ja s'havia utilitzat Trello [4] prèviament en altres assignatures del grau com ara Gestió de Projectes o Enginyeria del Software.

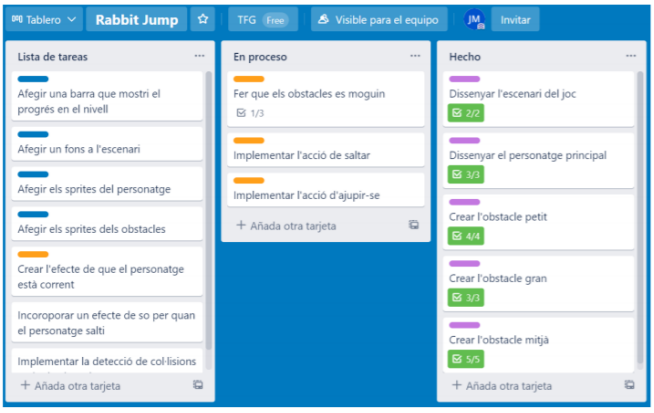


Fig. 1. Exemple d'un taulell Trello.

4 Planificació

El projecte es divideix en dos grans blocs. El primer consisteix a investigar, aprendre i entendre més sobre el tema per crear una visió general del procés a seguir. Això implica llegir molts blogs, fórums i articles rellevants sobre la utilització d'algorismes de Dynamic Difficulty Adjustment i Procedural Level Generation. El segon bloc (i el principal) será desenvolupar el joc de plataformes amb un sistema que ajusti la dificultat dels nivells de forma dinámica en funció del rendiment del jugador.

5 Teories de la diversió

Entendre què és el que fa que un joc sigui divertit no és una tasca gens fàcil. Al llarg dels darrers anys, s'han proposat diverses teories i s'han dut a terme moltes investigacions sobre aquest àmbit. Abans de desenvolupar qualsevol joc, és necessari comprendre com de divertit i atractiu pot ser el joc per al futur públic. És per això que en aquesta secció s'identificaran els conceptes clau de la diversió en els jocs mitjançant l'estudi de tres teories diferents proposades per Malone [6], Csikszentmihaly [7] i Ralph Koster [8].

5.1 Teoria motivacional de Malone

L'any 1981, Thomas Malone [6] va presentar una teoria sobre la motivació intrínseca en el context del disseny de jocs d'ordinador [9]. Amb la seva investigació, Malone buscava com potenciar la motivació intrínseca per a l'aprenentatge, el desig d'aprendre per si mateix, sense necessitat de recompensa o càstig. Va observar que els jocs aconsegueixen motivar els jugadors i involucrar-los en la resolució de problemes i el pensament crític. Segons ell, la motivació intrínseca es crea per tres qualitats: el repte, la fantasia i la curiositat [13].

Repte: cada joc ha de tenir una sèrie d'objectius, que poden ser personals per al jugador o poden ser generats pel mateix joc per mantenir al jugador compromès. Com que el resultat del joc és incert (ja que l'usuari no sap si assolirà els seus objectius), això fa que el jugador es mantingui compromès i motivat en tot moment. Un repte òptim no ha de ser ni massa difícil ni massa fàcil.

Fantasia: Malone defineix la fantasia com les "imatges mentals" que els jugadors creen a partir de la interacció amb l'entorn. Les fantasies més efectives dels jocs són aquelles que estan més orientades a estimular la imaginació del jugador, com per exemple la interpretació d'un mapa en un joc d'aventures o esbrinar com vèncer a un oponent.

Curiositat: Per a l'èxit de la creació d'un joc, són importants dos tipus de curiositat: la sensorial i la cognitiva. La curiositat sensorial s'activa per l'estètica del joc (la seva aparença, els sons, el feedback, l'autèntica creació d'un món o esdeveniment). La curiositat cognitiva s'activa presentant oportunitats perquè el jugador pugui millorar els seus coneixements. Quan es dissenya un joc basat en aquest marc teòric, els jugadors estan més motivats per jugar i aprendre

5.2 Teoria del flux de Csikszentmihalyi

El doctor en psicologia Mihaly Csikszentmihalyi [7] va formular la teoria del flux [11] o la teoria de l'experiència òptima el 1975 en un article publicat al "Journal of Humanistic Psychology".

L'estat de flux és un "estat en el qual la persona es troba completament absorta en una activitat per al seu propi plaer, durant la qual el temps vola i les accions, pensaments i moviments se succeeixen unes a les altres sense pausa". Aquesta descripció de l'experiència del flux és idèntica al que experimenten la majoria de persones quan juguen a un joc que els agrada, moment en què perden la noció del temps i executen accions de manera automàtica. Per tant, en un videojoc és important fer coincidir la dificultat amb les habilitats del jugador amb l'objectiu de fer que el jugador assoleixi aquest estat de flux. Per contra, si la dificultat d'un joc és massa baixa, els jugadors s'avorririen i, si fos massa alta, es frustrarien.

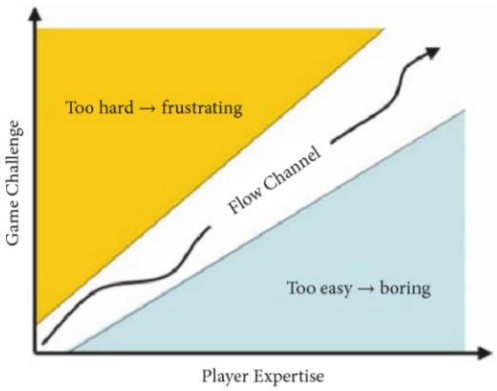


Fig. 2. Gràfica que relaciona el nivell de dificultat amb les habilitats de l'usuari.

A més a més, a mesura que avança el joc, el jugador adquireix noves habilitats i aprèn a com jugar, per la qual cosa és important anar augmentant progressivament la dificultat d'un joc per mantenir un estat de flux. Els desenvolupadors fan servir la tècnica del flux per avaluar la "diversió" en els jocs i adaptar-los per proporcionar als jugadors una experiència òptima. Tanmateix, mesurar la dificultat d'un joc i les habilitats del jugador per proporcionar un estat de flux no és una tasca senzilla.

5.3 Teoria de la diversió de Raph Koster

El 2003, Ralph Koster va escriure un llibre titulat *Theory of Fun for Game Design* [12], on ensenyava als dissenyadors de videojocs una nova manera de crear i millorar els seus dissenys per incorporar el màxim grau de diversió. Per a ell, la diversió es descriu com una recompensa per aprendre. Segons el seu punt de vista, els jocs han de presentar patrons i reptes que promoguin l'aprenentatge i el jugador ha de rebre feedback perquè pugui conèixer en tot moment quin és el seu progrés en el joc. Tanmateix, un cop après un patró, el joc ja no és divertit. Per tant, per mantenir l'interès per un joc, el jugador ha d'estar aprenent i constantment experimentant nous reptes.

6 EL GÈNERE DELS *AUTO RUNNERS*

Donada l'àmplia varietat de gèneres que podem trobar en els jocs actuals, és important dissenyar nivells que es corresponguin amb els criteris imposats per un gènere de joc en particular. A efectes d'aquest projecte, el joc que es crearà estarà basat en el gènere dels auto runners.

Els auto runners [13] són jocs de plataformes en què el personatge avança contínuament per un món generat de manera procedimental. Normalment, els auto runners presenten un conjunt de controls que es limiten a fer que el personatge salti, s'ajupi, llisqui o ataqui. El principal objectiu és arribar el més lluny possible abans que el personatge mori en xocar contra un obstacle. A mesura que el personatge corre per la pista, la velocitat del joc augmenta lentament i el jugador ha d'intentar recollir monedes, ítems que li proporcionin poder i també ha d'evitar obstacles o atacar a enemics mitjançant una combinació dels controls esmentats anteriorment. La simplicitat d'aquests tipus de jocs fan que siguin molt atractius per a les plataformes mòbils. Exemples d'aquest gènere són Canabalt [14] (2009), Temple Run [15] (2011) i JetPack Joyride [16] (2011).

Fig. 3. JetPack JoyRide del 2011.

El gènere dels auto runners [13] prové de jocs de plataformes 2D dels anys noranta. En aquella època, jocs com Super Mario [17] estaven de moda. Es tractava d'un joc senzill, on el jugador controlava tots els aspectes del protagonista principal (Mario), inclosos els moviments, els salts i els atacs que podia fer el personatge.

Amb l'aparició dels telèfons intel·ligents i les tauletes des del 2007, els hàbits dels consumidors van evolucionar fins al punt que també va canviar la manera de jugar. Els usuaris es van acostumar als jocs casuals, on les sessions de joc són minuts en lloc d'hores. Per satisfer aquesta demanda, els desenvolupadors van haver de simplificar els jocs de plataformes 2D.

Una de les variacions que van fer va consistir a reproduir l'experiència de plataformes 2D al dispositiu mòbil, sense controls de moviment. Els desenvolupadors van fer que el personatge principal corregués constantment i els jugadors només tenien el control de saltar. També va aparèixer un altre gènere, els endless runners, que eren molt semblants als auto runners, però eren jocs teòricament infinits on no hi havia nivells.

Donat que l'objectiu d'aquest projecte és dissenyar un joc amb un sistema que generi nivells de dificultat adaptats al rendiment del jugador, s'ha decidit que el joc que es crearà serà un auto runner [13] molt simple. Consistirà en un joc 2D on el personatge correrà constantment esquivant quatre tipus d'obstacles: un de petit, un de mitjà i un de gran, a més d'un quart obstacle que podrà aparèixer en tres alçades diferents. El jugador només tindrà la possibilitat de fer que el personatge salti o s'ajupi.

7 DYNAMIC DIFFICULTY ADJUSTMENT

Dynamic Difficulty Adjustment [18] és un terme general utilitzat per referir-se als mètodes que alteren la jugabilitat d'un joc per adaptar-se al rendiment del jugador. DDA també es coneix com a Dynamic Game Balancing (DGB) o Dynamic Game Difficulty Balancing. Més concretament, és el procés del canvi automàtic de paràmetres, escenaris i comportaments en un joc en temps real en funció de les habilitats i capacitats del jugador, per tal d'evitar avorrir el jugador (si el joc és massa fàcil) o fer que se senti frustrat (si és massa difícil). L'objectiu principal del DDA és mantenir l'interès de l'usuari des del principi fins al final, proporcionant un bon nivell de desafiament durant tot el nivell. No hi ha un algorisme universal per aplicar DDA, perquè sempre està dissenyat en funció del tipus de joc [19].

La implementació de qualsevol sistema DDA es basa fonamentalment en dues tècniques. La primera és un mecanisme que mesura el rendiment del jugador en el joc i la segona és una tècnica per ajustar la dificultat del joc.

Per avaluar el rendiment del jugador, tots els jocs poden incloure variables que indiquin l'estat actual del jugador i proporcionin informació sobre el seu rendiment, la relació d'èxit i fracàs, i sobre com està sent el seu procés d'aprenentatge. Les variables seleccionades dependran del joc específic en què s'implementa el sistema DDA. S'ha de tenir en compte que un nombre elevat de variables donarà lloc a un ajust de dificultat més precís, però

també consumirà més memòria, mentre que un nombre baix de variables pot resultar en un ajust pobre. La quantitat de variables escollides dependrà de la complexitat del joc, però en la seva majoria els estudis amb un nombre de tres a cinc variables són suficients. En el cas de l'auto runner que es crearà, les variables escollides per avaluar el rendiment del jugador seran les següents: número d'obstacles superats, número de salts realitzats, número de vegades que el personatge s'ajup, número d'intents per superar el nivel, distancia recorreguda i temps viu.

Amb les variables anteriors escollides i analitzades, és possible adaptar la dificultat del joc per tal que coincideixi amb les habilitats del jugador. Alguns elements que es poden manipular per implementar Dynamic Difficulty Adjustment en un auto runner són: la velocitat del personatge, la freqüència dels obstacles, el tipus d'obstacle i la freqüència dels enemics (no en aquest joc).

Com més velocitat tingui el personatge, menys temps de reacció té el jugador, cosa que el fa el joc més difícil. El mateix passa amb la freqüència d'obstacles, si hi ha més obstacles, la probabilitat xocar contra alguna cosa augmentarà. Pel que fa al tipus d'obstacles, la mida dels obstacles determinarà com de complicats de superar són, sent l'obstacle més petit el més fàcil de sortejar i el més gran el més difícil. L'obstacle que es pot presentar a tres alçades diferents també té el seu valor de dificultat afegit, ja que el personatge té tres maneres diferents d'esquivar-lo: si es presenta arran de terra, haurà de saltar; si està a mitjana alçada, podrà saltar o ajupir-se; i si està a la màxima alçada, no haurà de fer res. Es tracta d'un obstacle que pot causar confusió respecte a quina acció dur a terme per superar-lo, per la qual cosa és també el més difícil.

Així doncs, per crear un sistema de Dynamic Difficulty Adjustment en el Square Jump primer s'estudiarà el rendiment del jugador mitjançant l'anàlisi de paràmetres del joc com ara el número d'obstacles superats, el número de salts realitzats, el número de vegades que el personatge s'ajup, la distància total recorreguda i el temps que el personatge està viu. Tot seguit, es crearà un agent amb algorismes genètics, els paràmetres de rendiment del qual siguin semblants al del jugador real. Aquest agent se sotmetrà a una sèrie de nivells generats procedimentalment i, finalment, el nivell presentat a l'usuari serà aquell que hagi resultat més desafiant per a l'agent. La dificultat dels nivells es podrà modificar mitjançant la manipulació dels paràmetres de velocitat del personatge, freqüència dels obstacles i tipus d'obstacle.

7.1 Algorismes genètics

Per crear l'agent que modeli el comportament del jugador es faran servir algorismes genètics [20]. Els algorismes genètics [20] són una tècnica d'optimització basada en la cerca que s'inspira en els principis de la genètica i la selecció natural. En aquests tipus d'algorismes, tenim un grup o una població de possibles solucions al problema donat. Aquestes solucions se sotmeten a una recombinació i mutació (com en la genètica natural), produint nova descendència, i el procés es repeteix durant diverses generacions fins que es troba una solució òptima.

8 Procedural Level Generation

Procedural Content Generation (PCG) [21] és un mètode àmpliament utilitzat en el món dels videojocs que consisteix en la creació de contingut de forma automàtica mitjançant l'ús d'algorismes. Tot i que PCG es pot utilitzar per desenvolupar diverses funcions d'un joc, aquest projecte se centrarà en l'ús de PCG per a la creació de nivells en els videojocs. Concretament, es farà servir Procedural Level Generation (PLG) [22], que és una tècnica que utilitza algorismes de Procedural Content Generation per generar l'entorn específic d'un nivell d'un joc.

Donat que PLG crea contingut algorítmicament sense necessitat de la intervenció humana, contribueix a fer un ús efectiu dels recursos i permet l'ampliació de la jugabilitat. Tenir un joc on els nivells es construeixen sols per si mateixos en comptes d'haver-ho de col·locar tot manualment estalvia molt de temps i treball. A més, fa que el joc sigui rejugable perquè crea ambients nous que el jugador pot experimentar.

Actualment, hi ha diversos algorismes de Procedural Level Generation que s'utilitzen en el desenvolupament de videojocs. Cadascun té una manera única de generar contingut automàticament i es poden avaluar des de diferents perspectives: els nivells es poden jugar (és a dir, és posible acabar-los)? Els nivells generats són divertits de jugar? Quina velocitat té el generador? Pot generar nivells en temps real? Com de variats són els nivells generats? Es veuen tots el mateix? Els nivells són estèticament agradables? Es veuen naturals? El generador requereix intervención humana?

Tots aquests aspectes afecten directament la utilitat del generador. Per exemple, si un generador produeix bons nivells, però tots tenen el mateix aspecte, no és un generador molt útil. Els generadors que es mencionen en aquesta secció es poden classificar en tres categories diferents: els generadors basats en agents (o constructius), generadors basats en la cerca i una combinació dels anteriors (generadors compositius) [23].

8.1 Generadors basats en agents

Els generadors basats en agents [23] poden ser molt senzills. Aquest enfocament implica la utilització d'un agent que simula el moviment del personatge en el joc (córrer, saltar i ajupir-se). Consisteix a deixar que l'agent es mogui aleatòriament per un nivell buit mentre es va enregistrant tot el seu moviment. En funció del camí enregistrat, s'afegeixen plataformes, obstacles i altres objectes al nivell que es vol crear. Aquest enfocament compleix fàcilment l'objectiu que el nivell es pugui jugar. Tot i això, es posa en dubte la diversió que l'usuari final pot experimentar mentre juga. Per aquest motiu, s'acostumen a utilitzar altres tècniques o es combinen els generadors basats en agents amb altres mètodes [23].

8.2 Generadors basats en la cerca

Els generadors basats en la cerca es basen en una idea completament diferent. Com el seu nom indica, el principi fonamental és buscar entre un conjunt de nivells existents i triar aquell que sigui millor segons un criteri donat [23]. D'aquesta manera, produeix un contingut més refi-

nat que altres algorismes procedimentals on intervé més el factor de l'atzar. Per aconseguir-ho, fa ús d'un cicle generació-prova, on el contingut es genera per primera vegada i després s'avalua mitjançant una sèrie d'iteracions. Durant cada iteració, el contingut no només s'accepta o es rebutja, sinó que utilitza una funció per classificar cada nivell generat amb un valor de fitness. El nivell amb millor valor de fitness és el que s'utilitza per crear més nivells a la següent iteració. Per tant, l'algorisme cada vegada produeix nivells amb un valor de fitness més alt.

En general, el problema d'aquesta tècnica és la velocitat de creació dels nivells, ja que els algoritmes evolutius utilitzats són computacionalment molt intensos. El que és interessant, però, és que hi ha una manera molt natural de millorar els resultats (o obtenir bons resultats més ràpidament). És possible utilitzar simplement nivells creats per humans com a població inicial i si la fitness function està ben entrenada, aleshores els nivells construïts a sobre aquests nivells també seran bons i seran diferents.

8.3 Generadors compositius

Un generador de nivells compositiu [23] és un generador que combina les dues tècniques mencionades anteriorment, on el generador basat en agents és responsable de la creació dels nivells i el generador basat en la cerca tria quin és el millor nivell. L'agent simplement es mou per un nivell inicialment buit i a partir del seu moviment es col·loquen els obstacles que fan que el nivell sigui jugable. Un cop es tenen un cert nombre de nivells creats, s'avaluen cadascun d'aquests nivells amb una fitness function per determinar el nivell més apte.

Per al joc Square Jump es farà servir aquest tipus de generador. Primer, el model del jugador se sotmetrà als nivells creats amb el generador basat en agents. Tot seguit, amb el generador basat en la cerca, s'escollirà quin ha estat el nivell on el model ha obtingut un millor resultat. Finalment, el nivell resultant serà el nivell al qual s'haurà d'enfrontar el jugador.

9 Disseny i implementació del joc

La implementació d'aquest projecte s'ha dut a terme mitjançant la utilització d'HTML5 [24] i JavaScript [25]. Concretament, s'ha fet servir Canvas [26], un element HTML incorporat en HTML5 que permet la generació de gràfics dinàmicament per mitjà de JavaScript. L'element Canvas [26] només és un contenidor per a gràfics i s'ha de fer servir JavaScript per dibuixar-los. Canvas té diversos mètodes per dibuixar línies, rectangles, cercles, text i afegir imatges.

Un joc típic de plataformes en 2D implica que un jugador controla un personatge (avatar del joc) i va completant una sèrie de nivells on ha de complir un objectiu específic. El joc desenvolupat fins al moment inclou un petit personatge amb forma de rectangle vermell que ha d'evitar quatre tipus d'obstacles diferents (en aquest cas, representats amb rectangles de color blau). Consta de 9 nivells diferents, cadascun amb un nivell de dificultat calculat a partir del rendiment del jugador en el nivell anterior. Al principi del joc, el jugador només té disponible el nivell 1 i la resta de nivells estan bloquejats. Una vegada completa el primer nivell, es desbloqueja el segon nivell que, al seu torn, també desbloqueja el tercer nivell en completar-se. Així successivament fins que el jugador desbloqueja els 9 nivells. Un indicador a la part superior esquerra de la pantalla mostra el nivell que està jugant el jugador en aquell moment. La figura 4 il·lustra l'aparença general del joc:

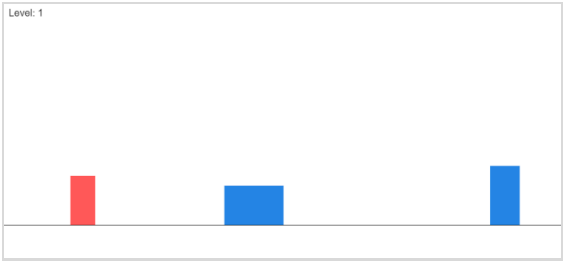
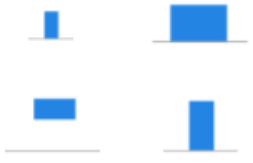


Fig. 4. Joc utilitzat per aquest projecte, on es pot veure el personatge principal (en vermell) i dos obstacles (en blau).

El joc desenvolupat no té cap condició de pèrdua; el jugador simplement continua jugant fins que es completen tots els nivells i després s'acaba el joc. Per completar un nivell de joc, el jugador ha de superar el nombre total d'obstacles de què disposa el nivell.

La resta de components del joc (avatar del joc, accions que pot dur a terme el jugador, obstacles, etc.) es detallen a continuació:

Avatar del joc 	L'avatar del joc representa el personatge controlat pel jugador. Es tracta d'un simple rectangle de color vermell que té la capacitat de saltar o ajupir-se.
Obstacles 	En un joc de plataformes 2D, els obstacles s'utilitzen sovint per frenar el progrés o causar danys a l'avatar del joc. En aquest joc, hi ha quatre tipus d'obstacles diferents. Per una banda, hi ha tres obstacles que estan a terra i poden presentar alçades i amplades diferents. També hi ha un quart obstacle volador que pot aparèixer en tres alçades diferents.
Gravetat 	Quan comença la partida, el personatge cau del cel i s'atura quan arriba a terra. Aquest efecte, a banda d'afegir més espectacularitat al joc, serveix perquè el jugador es faci una idea de com funciona la gravetat i ho tingui en compte a l'hora de calcular el moment en què ha de saltar.

Saltar 	Prement la tecla de l'espai o la fletxa apuntant cap amunt, el jugador pot fer que el personatge salti per esquivar els obstacles. Depenent del temps que premi la tecla, el salt pot ser gran o petit. Un toc curt farà que el personatge se separi una mica del terra per tornar ràpidament a la seva posició inicial, mentre que un toc més llarg provocarà que el personatge s'elevi més i estigui més temps suspès a l'aire.
Ajupir-se 	Prement la tecla de la fletxa apuntant cap a baix, el jugador pot fer que el personatge s'ajupi per esquivar els obstacles. Quan això passa, el rectangle canvia de posició vertical a horitzontal.
Detecció de col·lisions 	Quan el personatge col·lisiona amb algun obstacle, el joc ho detecta i s'atura la partida. Es mostra una finestra modal que dóna l'opció al jugador de repetir el nivell o tornar al menú principal.

Fig. 5. Taula resum amb les principals funcionalitats del joc.

10 SISTEMA PER AJUSTAR LA DIFICULTAT

10.1 Descripció del procés general

L'objectiu d'aquest treball de fi de grau és ajustar dinàmicament la dificultat dels nivells als quals el jugador s'ha d'enfrontar. Per aconseguir-ho, es necessita una manera d'avaluar el rendiment del jugador i una manera d'actualitzar els paràmetres que determinen la dificultat de cada nivell. El sistema resultant permet aplicar Dynamic Difficulty Adjustment en el Square Jump i altres jocs similars del mateix gènere.

Una visió general del procés de generació de nivells adaptatius seguit en aquest projecte és la següent:

1. Per a cada nivell, mesurar el rendiment del jugador mitjançant el càlcul del temps mitjà en segons que triga a completar el nivell.

2. Amb algorismes genètics, crear aleatòriament una població inicial de jugadors i evolucionar-la fins que un jugador sigui capaç de jugar al joc sense cometre cap error. El millor individu de cada generació serà un personatge amb una habilitat específica per jugar al joc que podrem fer servir com a model o representant del jugador real.

3. Sotmetre cada model obtingut en el pas anterior a diferents nivells i registrar el temps en segons que està viu cada model en aquests nivells.

4. Quan el jugador completa un nivell, cercar el model més semblant a l'estil de joc del jugador (comparant el temps que ambdós han estat vius en el nivell) i fer que el pròxim nivell presentat al jugador sigui aquell en què el model hagi mostrat un major rendiment. En altres paraules, aquell nivell on el model ha estat capaç de sobreviure més temps.

10.2 Creació del model del jugador

Com s'ha mencionat a l'apartat anterior, la creació dels models o representants del jugador s'ha dut a terme mitjançant l'ús d'algorismes genètics [27]. Un algorisme genètic és una heurística de cerca que s'inspira en la teoria de l'evolució natural de Charles Darwin. Aquest algorisme reflecteix el procés de selecció natural on els individus més forts són seleccionats per a la reproducció i produir descendència de la propera generació.

El procés de selecció natural comença amb la selecció dels individus més forts d'una població. Aquests individus produeixen descendència que hereta les seves característiques i que s'afegiran a la propera generació. Si els pares tenen una bona forma física, la seva descendència serà millor que els pares i tindrà més possibilitats de sobreviure. Aquest procés continua iterant-se i, al final, es trobarà una generació amb els individus més forts.

Aquesta noció es pot aplicar per a un problema de cerca, considerant un conjunt de solucions per a un problema i seleccionant-ne el millor.

Es consideren cinc fases en un algorisme genètic:

1. Població inicial: el procés comença amb un conjunt d'individus que s'anomena població. Cada individu és una solució al problema que es vol resoldre. Un individu es caracteritza per tenir un conjunt de paràmetres (variables) coneguts com a gens. Els gens s'uneixen en una cadena per formar un cromosoma (solució). En el cas d'aquest projecte, la població inicial serà de 10 individus i els cromosomes contindran l'equació que potencialment pot resoldre el problema.

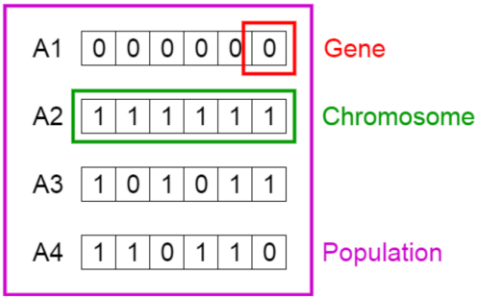


Fig. 6. Població, cromosomes i gens.

2. Funció d'avaluació: la funció d'avaluació generalment és la funció objectiu, és a dir, és el que es

vol arribar a optimitzar. Per al problema que ens ocupa, la funció d'avaluació és la que representa la decisió de salt del jugador. Aquesta decisió es pren a partir de 3 *inputs*, que són la velocitat del joc, l'amplada de l'obstacle i la seva distància al jugador. L'objectiu de l'algorisme genètic és optimitzar els coeficients (Ws) d'aquestes 3 variables en l'equació, juntament amb el bias.

$$Decision = w1 * speed + w2 * width + w3 * distance + bias$$

3. Selecció: la idea de la fase de selecció és seleccionar els individus més aptes i deixar-los passar els gens a la següent generació. Se seleccionen dos parells d'individus (pares) a partir dels quals es crea la seva descendència. Els pares escollits han estat els dos últims jugadors que han sobreviscut en el joc. La seva descendència se substitueix pels dos primers jugadors a morir per obtenir la següent generació.

4. Crossover: el crossover és la fase més significativa en un algorisme genètic. Per a cada parella de pares a combinar, es tria un punt de crossover a l'atzar entre els gens. Per exemple, considerem que el punt de crossover és 3 com es mostra a continuació:

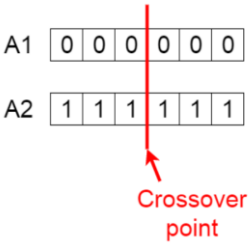


Fig. 7. Punt de crossover.

Els descendents es creen intercanviant els gens dels pares entre ells fins que s'arriba al punt d'encreuament.

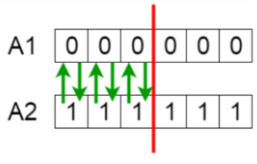


Fig. 8. Intercanvi de gens entre pares.

La nova descendència s'afegeix a la població.

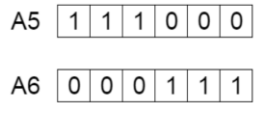


Fig. 9. Nova descendència.

5. Mutació: de la nova descendència obtinguda, alguns dels seus gens poden ser sotmesos a una mutació, és a dir, es poden introduir canvis aleatoris en algunes solucions. Això es fa per mantenir la diversitat dins de la població i evitar la convergència prematura.

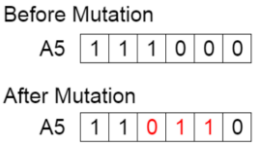


Fig. 10. Mutació: abans i després.

Aquests passos es repeteixen fins que les solucions obtingudes siguin acceptables o satisfacin el significat d'"òptim", que dependrà del que s'està intentant aconseguir. En el cas d'aquest projecte, l'algorisme genètic s'aturarà quan un jugador sigui capaç de jugar al joc de manera autònoma.

10.2.1 Representació d'una generació en Square Jump

La implementació dels algorismes genètics aplicada al projecte d'Square Jump s'ha dut a terme mitjançant la utilització de 10 individus que s'han anat evolucionant fins que un d'ells era capaç de jugar al joc de manera autònoma. Cada individu (cromosoma) s'ha representat com un *array* de quatre posicions: les tres primeres per als coeficients w1, w2 i w3 i l'última per al bias. Aquests valors s'han combinat amb els 3 *inputs* (velocitat, amplada i distància) de l'obstacle que s'acostava tal com s'indica a la funció de decisió. Si el resultat era inferior a 0, el jugador saltava. A la figura 11 es pot veure un exemple de població inicial i la seva representació.

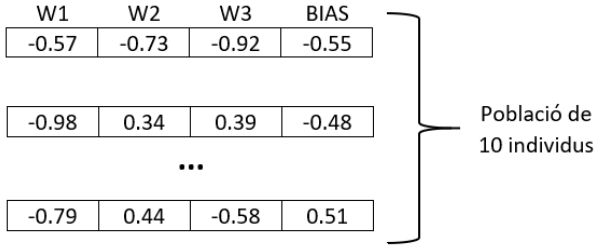


Fig.11. Representació de la població inicial.

Al començament, els pesos w1, w2 i w3 i el bias s'inicialitzen aleatòriament. Una vegada tots els jugadors moren, en el procés de selecció s'escullen com a individus més aptes els dos últims en morir. Aquests dos individus són els pares a partir dels quals es crea la propera generació. El següent pas és triar un punt de crossover a l'atzar i intercanviar els gens dels pares entre ells. Dels cromosomes resultants, es muta un gen per afegir més diversitat a la població. Finalment, es construeix la nova generació reemplaçant els dos pitjors jugadors de la generació anterior (els primers a morir) pels dos nous cromosomes obtinguts. A la figura 12 es mostra el procés de selecció, crossover i mutació descrit anteriorment.

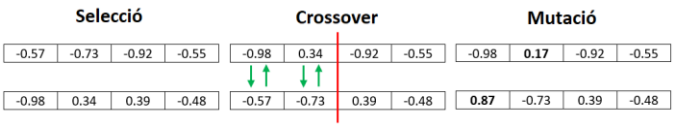


Fig. 12. Representació gràfica dels processos de selecció, crossover i mutació duts a terme durant el projecte.

10.3 Generació dels nivells

Amb els algorismes genètics s'han obtingut un total de 50 generacions. De cada generació, s'ha escollit el millor individu com a possible representant del jugador. L'individu de la generació 50 no és infal·lible, ja que es creu que amb una velocitat molt alta del joc acabaria morint en intentar passar per sota d'un obstacle que voli a mitja altura. No s'ha pogut comprovar perquè la velocitat del joc incrementa gradualment i s'hauria d'esperar força temps fins a assolir una velocitat tan elevada. A efectes pràctics, però, no interessa que el model sigui perfecte, donat que el desitjable és disposar d'un model amb les fortaleses i debilitats que podria tenir un jugador real.

Pel que fa a la dificultat dels nivells, aquesta s'ajusta mitjançant la modificació de 3 paràmetres: la velocitat del joc, la distància entre els obstacles i el nombre total d'obstacles. Cada paràmetre és un nombre enter positiu que determina la quantitat desitjada en el nivell del joc. Per generar els nivells, s'han fet servir diferents combinacions d'aquests paràmetres. Per a la velocitat, s'han agafat valors que van des dels 5 fins als 30 i s'incrementen de 5 en 5. La distància entre els obstacles es regula amb un temporitzador que quan arriba a 0 crea un obstacle que es mou cap al jugador. Depenent el nivell, el temporitzador pot començar en un valor que va des de 200 fins a 40 (per cada nivell va disminuint de 20 en 20). El nombre d'obstacles pot ser de 10 fins a 50 i avança de 10 en 10. Combinant tots els valors possibles que poden prendre els paràmetres, s'obtenen 270 nivells diferents.

El següent pas ha estat provar els 50 models als 270 nivells generats i registrar-ne el temps que cada model ha estat viu en el nivell. Això implica fer un total de 13.500 proves. Donat que no s'ha trobat cap manera d'accelerar el procés, s'han emprat 2 dies per registrar les dades que es necessitaven. Es tracta d'un temps força significatiu, però necessari per obtenir la informació dels models. Com a resultat de l'operació, de cada model s'ha registrat el nivell que s'estava provant, la generació a la qual pertany, el temps viu i els paràmetres (velocitat, distància entre obstacles i nombre d'obstacles) corresponents al nivell.

10.4 Elecció dels paràmetres

Un cop obtinguda la informació dels models i registrat el temps mitjà que ha trigat el jugador a superar el nivell, només queda relacionar el jugador amb un model per assignar-li els paràmetres del pròxim nivell. La informació dels models s'emmagatzema en un objecte de tipus JSON [28], un format de text compatible amb JavaScript [25] que es fa servir per a l'intercanvi de dades. Per tal de fer l'explicació més entenedora, però, aquesta informació es representarà amb una matriu que té la forma que es pot veure en la figura 13. Les columnes de la matriu corresponen als models i les files als nivells.

	Model 1	Model 2	Model 3	Model 4	Model 5
Nivell 1 (s: 5, d: 200, n: 10)	7	10	9	9	9
Nivell 2 (s: 5, d: 200, n: 20)	9	7	10	9	9
Nivell 3 (s: 5, d: 200, n: 30)	9	9	9	7	7
Nivell 4 (s: 5, d: 200, n: 40)	6	10	6	7	9
Nivell 5 (s: 5, d: 200, n: 50)	7	7	9	7	6

Fig. 13. Matriu que relaciona els models (columnes) amb cadascun dels nivells (files) i els seus paràmetres (s: speed, d: distance, n: number of obstacles). Cada casella de la matriu correspon al temps que el model ha estat viu en el nivell.

Quan un jugador completa un nivell, es busca a la matriu aquell model amb el temps més proper al temps mitjà que ha emprat el jugador per superar el nivell. Això és, de la fila corresponent al nivell que s'acaba de completar, s'extreu aquell model que disposi d'un temps similar al del jugador. En cas d'haver-hi més d'un model amb el mateix temps, se seleccionarà de manera aleatòria un d'ells.

En el moment en què se selecciona el model que representa millor el jugador, s'ha de cercar el nivell on el model ha sobreviscut més temps. En altres paraules, de la columna del model (que representa tots els nivells on ha estat provat el model), s'escull la posició amb un temps més elevat. Si hi ha més d'una posició amb el mateix temps, s'escollirà una de les posicions candidates de manera aleatòria. Si el nivell resultant d'aquesta selecció aleatòria ja ha estat jugat, es triarà el següent nivell.

Aquesta és una aproximació bàsica per triar el nivell més adient per al jugador. Tanmateix, el resultat no és l'esperat, ja que el canvi entre els diferents nivells és mínim i tots els nivells són molt semblants. Per tal de fer l'ajust de dificultat més notable, s'ha afegit un altre component que afecta la tria del millor nivell. Si el jugador completa el nivell en 5 intents o menys, a l'hora de triar el nivell on el model té un temps més elevat només es tenen en compte nivells on la velocitat és major a la del nivell actual. En cas contrari, si el jugador necessita més de cinc intents per superar el nivell, la cerca es fa amb nivells on la velocitat sigui menor. D'aquesta manera la velocitat serà diferent a cada nivell i el jugador té la percepció que la dificultat ha canviat.

11 Resultats

L'objectiu del projecte era implementar Dynamic Difficulty Adjustment amb Procedural Content Generation en un joc de plataformes en 2D anomenat Square Jump. Un jugador jugaria el joc durant un temps, el sistema d'ajust de dificultat dinàmic calcularia el rendiment del jugador i després, amb l'ajuda del generador de contingut, es generarien nivells segons el rendiment del jugador.

Per tal d'avaluar l'efectivitat del sistema de DDA implementat, s'ha realitzat una enquesta a un total de 20

persones a les quals es demanava que omplissin un breu formulari després de jugar un mínim de 3 nivells del joc. El formulari constava de 7 preguntes relacionades amb la seva experiència jugant a videojocs i amb les sensacions que tenien després d'haver jugat a Square Jump, juntament amb un quadre de text on l'enquestat podia deixar un comentari sobre el joc. El qüestionari complet es pot veure a l'apèndix A1. Les respostes es detallen a continuació:

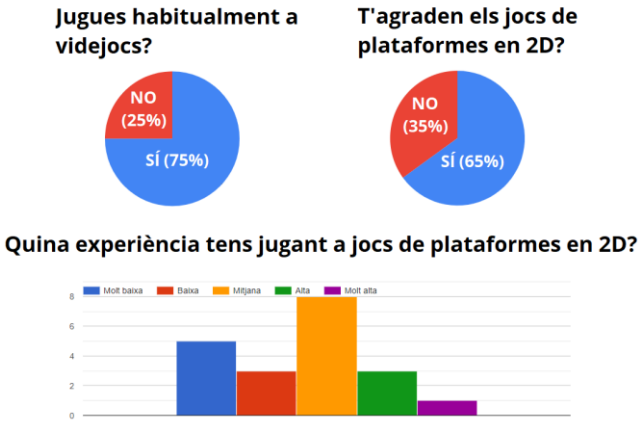


Fig. 14. Gràfica que mostra els resultats corresponents a les tres primeres preguntes de l'enquesta.

Les tres primeres preguntes de l'enquesta tenien com a objectiu conèixer el tipus de jugador que jugava a l'Square Jump. Es volia saber quina era l'experiència de l'enquestat amb els videojocs en general i amb els jocs de plataformes en 2D. Com es pot veure a la figura anterior, la gran majoria d'enquestats jugaven habitualment a videojocs i els agradaven els jocs de plataformes en 2D. Tot i això, la seva experiència jugant a jocs de plataformes era variada, amb un clar predomini de persones que tenien una experiència mitjana i algunes de molt baixa. Aquesta varietat de jugadors és interessant de cara a les pròximes preguntes, ja que podrem descobrir si el joc realment adapta la seva dificultat independentment de l'experiència i les habilitats de l'usuari.

La resta de preguntes de l'enquesta estaven enfocades a obtenir informació sobre els pensaments i impressions dels usuaris després de jugar a l'Square Jump. L'anàlisi d'aquestes preguntes, però, s'ha fet de forma individual, ja que abastaven temes que van des de com havien trobat de divertit l'Square Jump fins a quin era el seu nivell jugant a videojocs.

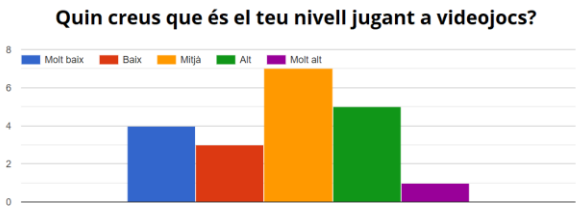


Fig. 15. Gràfica dels resultats de la pregunta 4 de l'enquesta.

A la figura 15 es pot veure que la majoria de persones enquestades reconeixien tenir un nivell mitjà-alt jugant a videojocs. També hi havia un grup més reduït de 6 persones que afirmaven tenir un nivell baix o molt baix. De nou, com més variades les respostes en aquesta pregunta, millor, perquè es podrà comprovar si s'assoleix l'objectiu del projecte.

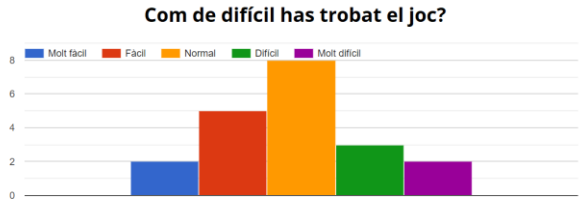


Fig. 16. Gràfica dels resultats de la pregunta 5 de l'enquesta.

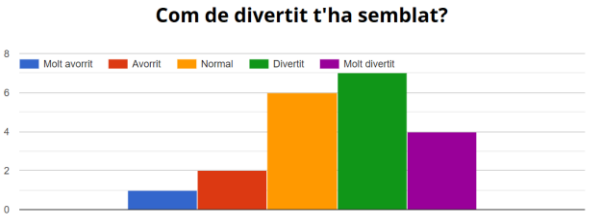


Fig. 17. Gràfica dels resultats de la pregunta 6 de l'enquesta.

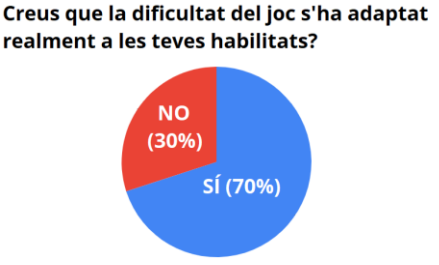


Fig. 18. Gràfica dels resultats de la pregunta 7 de l'enquesta.

A la figura 16 es pot observar que la majoria de jugadors han trobat que el joc té una dificultat "normal", la qual cosa és positiva perquè el joc no és massa fàcil ni massa difícil. Pel que fa a la figura 17, es pot veure que el joc ha estat divertit per a molts dels enquestats i el 70% creuen que el joc ha adaptat la dificultat a les seves habilitats (figura 18). A la secció de comentaris, un dels comentaris més repetits era que el joc tenia molts obstacles.

12 Conclusions

La solució presentada en aquest estudi proporciona un joc on cada nivell compta amb un gran nombre d'obstacles. Això és degut al fet que s'utilitza el temps viu en un nivell per comparar el rendiment del jugador amb el dels models. Un temps viu elevat significa que el jugador té les habilitats suficients per a no morir aviat en el nivell, la qual cosa es tradueix en un bon rendiment. Tanmateix, els nivells on hi ha un gran nombre d'obstacles també provoquen que el temps necessari per superar-lo sigui alt i en conseqüència els nivells presentats al jugador acaben

tenint 50, 40 o 30 obstacles.

Al llarg del projecte han sorgit diferents problemes pel que fa a la creació de les diferents funcionalitats. El desenvolupament dels algorismes genètics, per exemple, va suposar una major duració de la planejada en un principi. El motiu va ser que un dels *inputs*, el de la posició X de l'obstacle que s'apropa al jugador, no es carregava correctament i en conseqüència els agents saltaven aleatòriament sense aprendre després d'evolucionar de generació en generació.

El llenguatge escollit no és el millor perquè normalment, en els jocs, els aspectes més habituals com ara la detecció de col·lisions o altres tasques de renderització, solen requerir càlculs intensius utilitzant la GPU. Amb JavaScript, atès que ens limitem a fer servir el navegador, la potència de càlcul no és tan potent com altres motors i tecnologies de renderització de jocs. A més a més, si el joc està basat única i exclusivament en la part del client, qualsevol usuari pot manipular variables i cridar funcions del joc a través de la consola i, per tant, fer trampes molt fàcilment. Per evitar això s'hauria d'incloure un servidor encarregat de gestionar els valors vertaders de les variables. Pel que fa a l'Square Jump, però, l'ús de JavaScript i HTML5 Canvas ja és suficient perquè mostra de manera molt visual l'evolució dels algorismes genètics i permet dissenyar gràfics fàcilment per a qualsevol joc.

Per últim, afegir també que degut a la facilitat de trobar dades sobre el canvi de dificultat en els jocs, s'ha arribat a la conclusió que el Dynamic Difficulty Adjustment és un mètode bastant popularitzat i acceptat, per la qual cosa segurament en el futur se seguirà fent servir.

13 Treball futur

A causa de les limitacions de temps i la quantitat de treball necessari, ha quedat espai per a noves millores i la possibilitat d'afegir noves funcionalitats. Alguns treballs futurs per a aquesta implementació podrien ser:

- Afegir *sprites*. El joc té una aparença força simple i seria interessant afegir alguns *sprites* per fer-lo més atractiu visualment parlant. En el cas dels obstacles terrestres, una imatge fixa ja seria suficient per donar-los l'aparença d'algun objecte (unes roques, per exemple). Pel que fa al personatge principal i a l'objecte volador, seria interessant que disposessin d'una animació de córrer i de volar respectivament. Per al fons, es podria fer que de tant en tant passessin núvols de forma fixa però posició aleatòria. Són canvis força simples, però el joc ja tindria una altra aparença.

Fig. 19. : Exemples d'*sprites* que es podrien fer servir en el joc.

- Com que els usuaris que han jugat al joc s'han quedat amb ganes de més, s'ha pensat que es podria crear una nova versió del joc amb dues pistes. Cada pista tindria un personatge que es controlaria de manera independent, fet que augmentaria significativament la dificultat del joc.

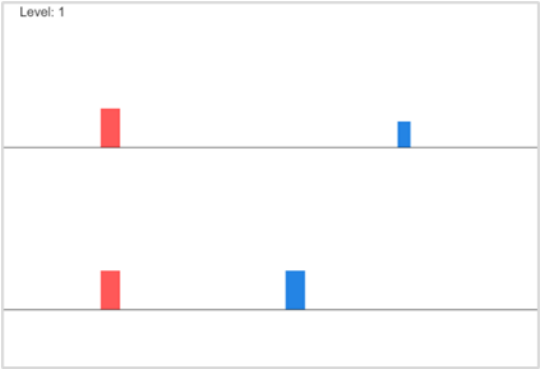


Fig. 20. Aparença general del joc amb dues pistes.

- Una altra possible millora seria comprar un domini web per comercialitzar el joc. El joc comercialitzat tindria més nivells i ja es vendria amb les dues pistes perquè els jugadors haguessin de jugar a dues mans.

Agraïments

Per acabar, vull agrair tota l'ajuda que he rebut per part de la meva tutora de treball de fi de grau, Ana Oropesa, gràcies a ella he après diferents tecnologies que no coneixia.

També vull donar les gràcies als meus amics i familiars, que m'han ajudat a definir el disseny i les funcionalitats del joc a través dels seus comentaris i consells.

Referències

[1] The Ultimate Resource for Video Game Design. 2021. What Are the 6 Different Types of Gamers?. [online] Available at: <https://www.gamedesigning.org/gaming/gamer-types/> [Accessed 12 March 2021].

[2] raywenderlich.com. 2021. Procedural Level Generation in Games Tutorial: Part 1. [online] Available at: <https://www.raywenderlich.com/2637-procedural-level-generation-in-games-tutorial-part-1> [Accessed 4 March 2021].

[3] Gilibets, L., 2021. Qué es la metodología Kanban y cómo utilizarla. [online] Thinking for Innovation. Available at: <https://www.iebschool.com/blog/metodologia-kanban-agile-scrum/> [Accessed 28 February 2021].

[4] Expertosnegociosonline.com. 2021. ▷ Trello. Qué es, Para Qué Sirve y Cómo Funciona. [online] Available at: <https://www.expertosnegociosonline.com/que-es-trello-para-que-sirve/> [Accessed 28 February 2021].

[5] Kanban Software for Agile Project Management. 2021. ¿Qué es un tablero Kanban? | Kanbanize. [online] Available at: <https://kanbanize.com/es/recursos-de-kanban/primeros-

pasos/que-es-tablero-kanban> [Accessed 28 February 2021].

[6] Cci.mit.edu. 2021. Thomas W. Malone | MIT Center for Collective Intelligence. [online] Available at: <https://cci.mit.edu/malone/> [Accessed 25 April 2021].

[7] Verywell Mind. 2021. A Biography of Positive Psychologist Mihaly Csikszentmihalyi. [online] Available at: <https://www.verywellmind.com/mihaly-csikszentmihalyi-biography-2795517> [Accessed 25 April 2021].

[8] Raph's Website. 2021. About Raph. [online] Available at: <https://www.raphkoster.com/about-raph/> [Accessed 25 April 2021].

[9] Playable. 2021. Malone's theory of gaming.. [online] Available at: https://deangroom.wordpress.com/2015/03/19/malones-theory-of-gaming/#:~:text=Mal one%20argues%20that%20intrinsic%20motivation,skills%20req uired%20for%20the%20instruction> [Accessed 16 March 2021].

[10] Learning Theories. 2021. Intrinsically motivating instruction (Malone) - Learning Theories. [online] Available at: <https://www.learning-theories.com/intrinsically-motivating-instruction-malone.html> [Accessed 16 March 2021].

[11] Instituto Europeo de Psicología Positiva. 2021. Teoría del Flow o Experiencia Óptima: El Tiempo Vuela - IEPP. [online] Available at: <https://www.iepp.es/teoria-del-flow/> [Accessed 18 March 2021].

[12] Google Books. 2021. Theory of Fun for Game Design. [online] Available at: <https://books.google.es/books/about/Theory_of_Fun_for_Game_Design.html?id=GQpQAAAAMAAJ&redir_esc=y> [Accessed 18 March 2021].

[13] Glitchwave. 2021. Auto runner - Glitchwave. [online] Available at: <https://glitchwave.com/games/genre/auto-runner/> [Accessed 20 March 2021].

[14] Play.google.com. 2021. [online] Available at: <https://play.google.com/store/apps/details?id=fishnoodle.canabalt&hl=en> [Accessed 25 April 2021]

[15] Play.google.com. 2021. [online] Available at: <https://play.google.com/store/apps/details?id=com.imangi.templerun&hl=en> [Accessed 25 April 2021].

[16] Play.google.com. 2021. [online] Available at: <https://play.google.com/store/apps/details?id=com.halfbrick.jetackjoyride&hl=en> [Accessed 25 April 2021].

[17] Nintendo of Europe GmbH. 2021. Super Mario Bros.. [online] Available at: <https://www.nintendo.es/Juegos/NES/Super-Mario-Bros--803853.html> [Accessed 25 April 2021].

[18] En.wikipedia.org. 2021. Dynamic game difficulty balancing - Wikipedia. [online] Available at: <https://en.wikipedia.org/wiki/Dynamic_game_difficulty_balancing> [Accessed 23 March 2021].

[19] Medium. 2021. More Than Meets the Eye: The Secrets of Dynamic Difficulty Adjustment. [online] Available at: <https://remptongames.medium.com/more-than-meets-the-eye-the-secrets-of-dynamic-difficulty-adjustment-bd35d6a08c82> [Accessed 23 March 2021].

[20] Tutorialspoint.com. 2021. Genetic Algorithms - Introduction - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/genetic_algorithms/genetic_algorithms_introduction.htm> [Accessed 27 May 2021].

[21] 2021. [online] Available at: <https://www.researchgate.net/publication/228620622_What_is_Procedural_Content_Generation_Mario_on_the_borderline> [Accessed 25 March 2021].

[22] Gamasutra.com. 2021. Procedural Level Generation in Unity for M.E.R.C. (part 1 of 2). [online] Available at: <https://www.gamasutra.com/blogs/GrahamDavis/20170130/290326/Procedural_Level_Generation_in_Unity_for_MERC_part_1_of_2.php> [Accessed 25 March 2021].

[23] Tobice.cz. 2021. [online] Available at: <https://www.tobice.cz/publications/level-generation-techniques-for-platformer-games.pdf> [Accessed 25 March 2021].

[24] W3schools.com. 2021. Introduction to HTML. [online] Available at: <https://www.w3schools.com/html/html_intro.asp> [Accessed 25 April 2021].

[25] W3schools.com. 2021. What is JavaScript. [online] Available at: <https://www.w3schools.com/whatis/whatis_js.asp> [Accessed 25 April 2021].

[26] W3schools.com. 2021. HTML Canvas. [online] Available at: <https://www.w3schools.com/html/html5_canvas.asp> [Accessed 17 April 2021].

[27] Medium. 2021. Introduction to Genetic Algorithms — Including Example Code. [online] Available at: <https://towardsdatascience.com/introduction-to-genetic-algorithms-including-example-code-e396e98d8bf3> [Accessed 23 May 2021].

[28] W3schools.com. 2021. What is JSON. [online] Available at: <https://www.w3schools.com/whatis/whatis_json.asp> [Accessed 23 May 2021].

APÈNDIX

A l'apèndix es complementarà la informació de l'informe amb més dades sobre la realització del projecte.

A1. ENQUESTA

1. Jugues habitualment a videojocs? *

	Sí	No
Selecciona una opció:	◯	◯

2. T'agraden els jocs de plataformes en 2D? *

	Sí	No
Selecciona una opció:	◯	◯

3. Quina experiència tens jugant a jocs de plataformes en 2D? *

	Molt baixa	Baixa	Mitjana	Alta	Molt alta
Selecciona una opció:	◯	◯	◯	◯	◯

4. Quin creus que és el teu nivell jugant a videjocs? *

	Molt baix	Baix	Mitjà	Alt	Molt alt
Selecciona una opció:	◯	◯	◯	◯	◯

5. Com de difícil has trobat el joc? *

	Molt fàcil	Fàcil	Normal	Difícil	Molt difícil
Selecciona una opció:	◯	◯	◯	◯	◯

6. Com de divertit t'ha semblat? *

	Molt avorrit	Avorrit	Normal	Divertit	Molt divertit
Selecciona una opció:	◯	◯	◯	◯	◯

7. Creus que la dificultat del joc s'ha adaptat realment a les teves habilitats? *

	Sí	No
Selecciona una opció:	◯	◯