

Escape from Wano

Autors: Jia Long Ji Qiu, You Wu

Videojocs: Exercici 3D

Tutor: Jesús Alonso Alonso

Curs 2022/23 (Q2)



**UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH**

Facultat d'Informàtica de Barcelona

FIB

Índex

1. Cliff Hopper.....	3
2. Descripció del Projecte.....	4
2.1 Estructura del nivell.....	4
2.2 Jugador.....	4
2.3. Càmera.....	5
2.4. Blocs.....	6
2.5. Punxes.....	7
2.6. Tolls de fang.....	8
2.7. Serres.....	8
2.8. Ninjes.....	9
2.9. Blocs inestables.....	10
2.10. Monedes.....	11
2.11. God Mode.....	12
3. Apartat artístic.....	13
3.1. Models 3D i Animacions.....	13
3.2. Música i So.....	14
4. Metodología.....	15
5. Conclusions.....	17
6. Bibliografia.....	18

1. Cliff Hopper

Cliff Hopper és un joc arcade individual per dispositius mòbils, compatible amb Android i iOS, que va ser desenvolupat i publicat per Mana Cube el 22 de febrer de 2017. L'equip de desenvolupament estava format per Maxime Bénéteau, Philippe Desgranges, Ridha El Bekri, Nicolas Godement i Sylvain Hellio. Va rebre la seva última actualització l'any 2019.

Actualment, aquest joc ja no es troba a portals de descàrregues oficials com la Play Store o la App Store, motiu pel qual és difícil saber-ne el número de descàrregues i quina acollida ha tingut entre els usuaris. Tampoc té cap web oficial i de comunitat d'usuaris.

En quant al gameplay, Cliff Hopper consisteix en un joc runner de plataformes on el jugador ha de evitar diferents obstacles mentre avança en un nivell generat de manera procedural. Gameplay tràiler: <https://www.youtube.com/watch?v=BBe2Qni7JNl>



Figura 1. Logo de Cliff Hopper

2. Descripció del Projecte

Per aquest projecte l'objectiu ha sigut fer un joc el més fidel possible al joc original, Cliff Hopper, amb un petit afegit d'originalitat. S'ha desenvolupat amb el motor gràfic Unity.

Les mecàniques són gairebé les mateixes: el nivell consisteix en un camí finit estructurat en zig-zag. El jugador sempre corre en línia recta fins que arriba a marcadors situats al final de cada segment, on podrà canviar de direcció. El jugador també pot saltar allà on sigui necessari, i té fins a dos salts seguits, un des de terra i l'altre a l'aire. Tant el canvi de direcció com el salt s'efectuen amb la tecla d'espai. Al llarg d'aquest nivell es trobarà amb diferents obstacles i trampes que haurà d'evitar: punxes, tolls de fang, serres, ninjes que tiren shurikens i blocs inestables. També hi ha monedes que podrà recollir. L'objectiu del joc és, per tant, arribar al final del nivell, on es trobarà amb una barca que permetrà al personatge escapar de la regió de Wano.

2.1 Estructura del nivell

Com ja s'ha explicat anteriorment, el nivell consisteix en un camí en zig-zag, on anomenem “segment” les parts que començen amb un canvi de direcció i acaben amb un altre, i son variables en quant a llargada.

Internament, el camí es traça en el plànor XZ, i a mesura que avança en el sentit de les X i Z positives, l'altura Y baixa gradualment. Així doncs, cada segment pot tenir una sola direcció: o bé en X o bé en Z.

2.2 Jugador

El jugador és una entitat que es mou en línia recta a una velocitat constant. Es controla amb la tecla espai i pot saltar (o fer un doble salt) i fer un canvi de direcció.

Quan fa un canvi de direcció, el seu cos rota 90° en sentit horari o antihorari dependent de la nova direcció, i se li aplica un offset gradualment que el desplaça al centre del segment.

Qualsevol col·lisió amb les punxes, les serres i els shurikens farà que salti al buit i mori. Quan es troba a sobre d'un toll de fag, es ralenteix i té menys potència de salt. Pot agafar les monedes que es troba quan s'apropa a aquestes.

En quant a animacions, el jugador per defecte comença corrent i quan efectua un salt canvia la animació segons si ha sigut el primer o el segon. Una vegada torna al terra, segueix corrent. També canvia la seva postura quan mor i quan arriba al final del camí.

2.3. Càmera

El paper de la càmera és molt important en els jocs runner com aquest, ja que ha de permetre al jugador una bona visualització tant del que hi ha com el que hi haurà, per tal de què es pugui pensar amb antelació com actuar de cara als propers segments. Per tant, el comportament de la càmera s'ha pensat amb molta cura, ja que és un factor decisiu que pot marcar la diferència entre una bona i mala jugabilitat.

Per començar, com que aquest joc té una projecció dimètrica, la càmera ha d'estar ajustada amb una projecció ortogonal i rotada amb els angles corresponents. En aquest cas, la càmera està girada 30° en l'eix X i 225° en l'eix Y.

En quant al moviment de la càmera, es va decidir el següent comportament: el desplaçament vertical ha de poder mantenir al jugador a la mateixa altura respecte a la pantalla, mentre que el desplaçament horitzontal ha de mantenir el segment a on es troba el jugador centrat a la pantalla. Per aconseguir-ho, el desplaçament s'ha definit amb un SmoothDamp de Unity, on el target varia segons els càlculs pertinents.

El càlcul de la posició del target es pot dividir en dos passos. Sota la premisa de què el jugador només es pot moure en el plànol XZ en sentit positiu:

D'una banda, per situar el target T' al mateix nivell que el jugador P , primer s'ha de substituir el punt P a la recta $z = -x + c$ per treure c , de manera que $c = z + x$. Seguidament, com que en un principi volem situar T' enmig del plànol, fem una intersecció amb la recta $z = x$, resultant en un sistema de coordenades que, resolent per substitució, obtenim $T' = (c/2, c/2)$.

D'altra, per obtenir finalment la posició al target que realment ens interesa, T , s'ha de calcular un offset que desplaçi la posició T' sobre la mateixa recta $z = -x + c$ de manera que des de la càmera es vegi el segment on es troba el jugador P de manera centrada. Per aconseguir aquest offset, primer s'ha d'obtenir el punt mitjà del segment actual que es pot calcular fàcilment amb les posicions inicial i final del segment. El que s'ha de calcular tot seguit és la recta que passa per aquest punt mitjà, que tindrà la forma $z = x + d$, que s'haurà de substituir amb el punt mitjà per calcular $d = z - x$. Calculant ara la intersecció entre les rectes $z = x + d$ i $z = -x + c$, obtenim que $T = ((c-d)/2, (c+d)/2)$. Finalment, l'offset es pot calcular amb la diferència $T - T'$, i es va aplicant un offset diferent per cada segment a mesura que el jugador avança.

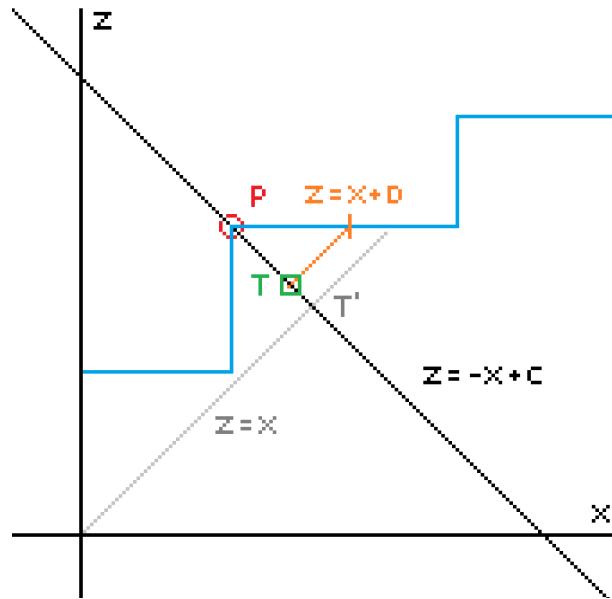


Figura 2. Esquema del càlcul de la posició del target T

2.4. Blocs

Les entitats per on camina el jugador son els blocs, els quals poden ser cubs si mantenen la mateixa altura, o un romboedre en cas de què hi hagi un desnivell (escales).

Els cubs tenen una rotació aleatòria sobre si mateixos al començament del nivell, per oferir una certa variació al mapa. A més a més, alguns cubs tenen a sobre i prem la tecla espai, provocant d'aquesta manera un canvi de direcció. Aquest marcador consisteix en un plànor 2D que pot tenir dues textures: desactivat i activat.

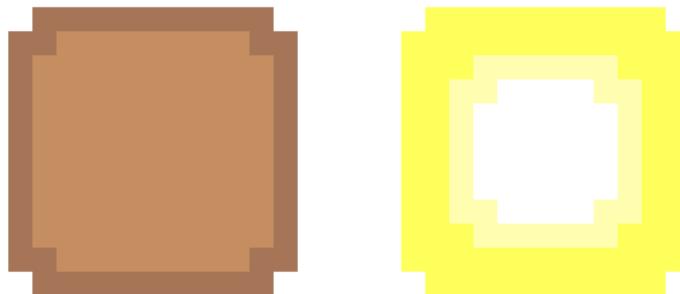


Figura 3. Textures 2D del marcador.

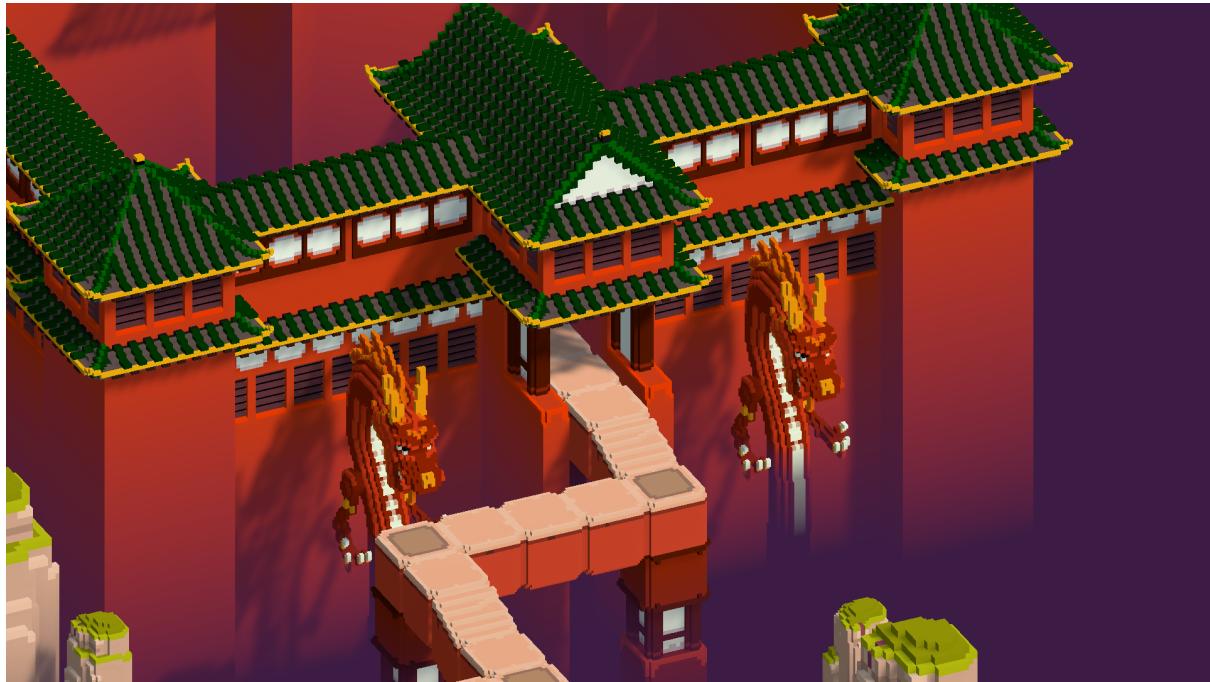


Figura 4. Pantalla principal del joc

2.5. Punxes

Els blocs amb punxa s'activen quan el jugador està a sobre. L'activació consisteix en una elevació molt ràpida que fereix al jugador. Passat un breu temps, el bloc baixa lentament fins situar-se a la seva posició inicial.

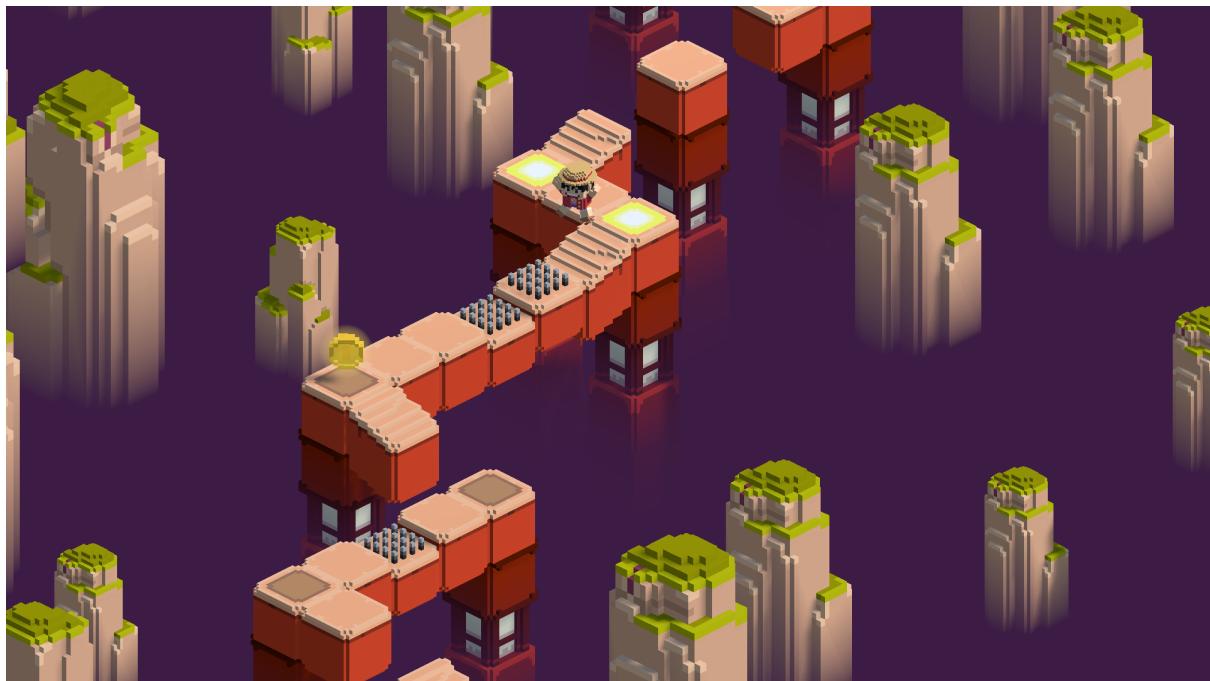


Figura 5. Punxes

2.6. Tolls de fang

Els tolls de fang alenteixen el jugador quan és a sobre, fent que l'avanç sigui més lent i el salt més curt.

Els blocs a on hi ha tolls de fang tenen un buit on es sitúa una textura 2D que canvia periòdicament i en bucle entre tres textures diferents, simulant d'aquesta manera un moviment circular.

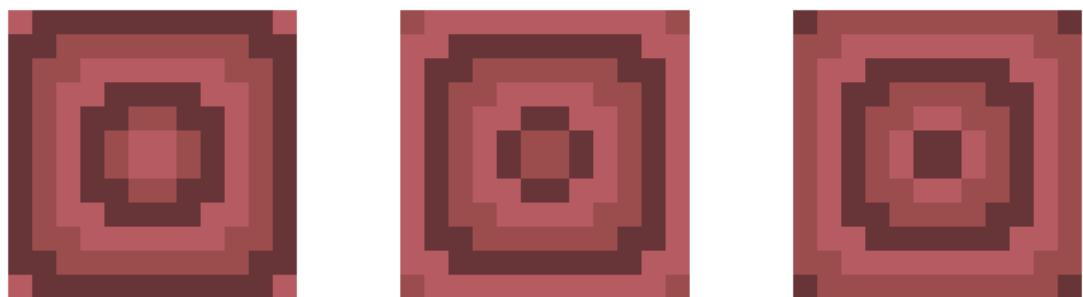


Figura 6. Textures 2D d'un toll de fang

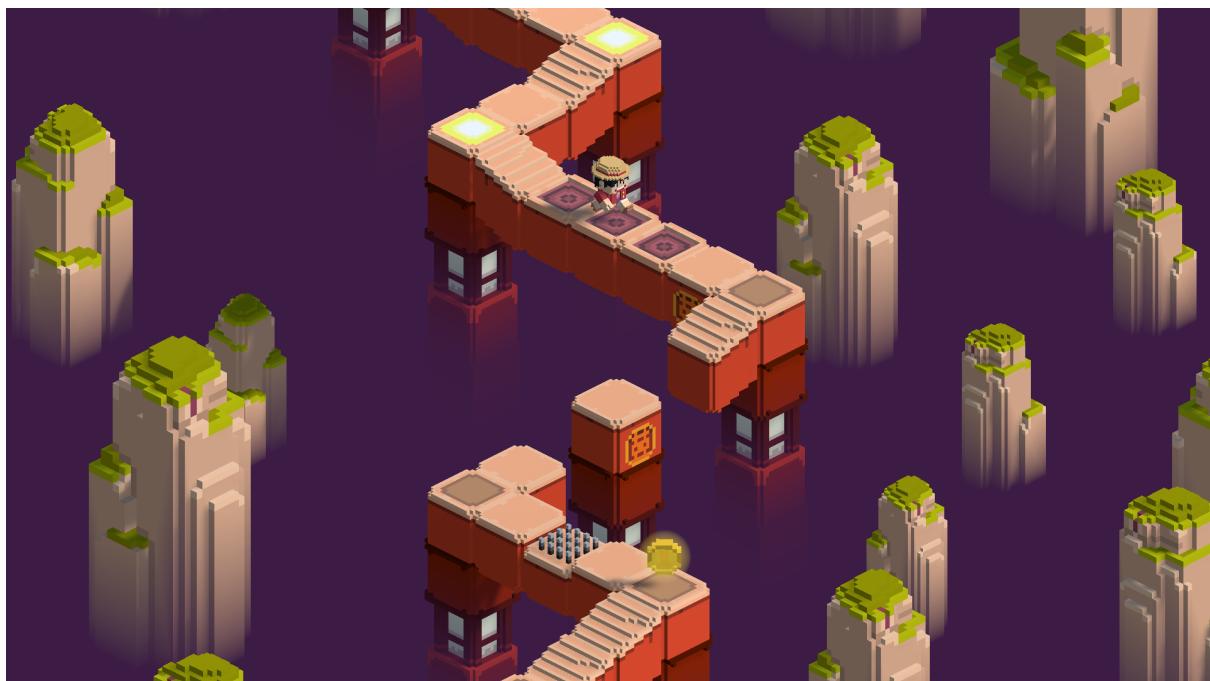


Figura 7. Tolls de fang

2.7. Serres

Les serres son discs que es desplacen lentament amb una velocitat constant sobre un rail, segons com estiguin orientades. Mentre es desplaçen, també giren sobre sí mateixes en l'eix corresponent, a un angle fixe de 90° que, amb l'ajuda del model 3D, crea un efecte de rotació constant.

Les serres poden desplaçar-se independentment d'on estiguin, ja que només se'ls hi ha d'especificar una direcció i una distància màxima de desplaçament, resultant així en un moviment d'anada i tornada constantment sobre el mateix recorregut. Els rails son, per tant, un element decoratiu que serveix per fer saber al jugador on comencen i acaben aquests recorreguts.



Figura 8. Serres

2.8. Ninjes

Els ninjes son NPCs que es sitúen apartats del camí zig-zag, esperant l'arribada del jugador per llençar-li un shuriken.

Quan un ninja valora que ja pot procedir a llençar un shuriken, dóna un petit salt i acte seguit fa un gest amb un braç dret. El shuriken avança en línia recta (X o Z negatives) en la direcció a la que estigui orientat el ninja i amb una velocitat constant, rotant sobre si mateix i deixant un trail de color blanc. El jugador podrà evitar-lo donant un salt en el moment adequat.



Figura 9. Ninja llençant un shuriken

2.9. Blocs inestables

Hi ha dos tipus de blocs inestables, els actius i els reactius. Els blocs actius son els que cauen quan detecten que el jugador s'apropa. Els blocs reactius, en canvi, estan vinculats a altres blocs inestables i cauen quan aquest altre bloc cau. D'aquesta manera, es poden crear reaccions en cadena on un conjunt de blocs cau un darrere l'altre.

Aquests blocs tremolen durant un breu període de temps, donant d'aquesta manera un feedback al jugador per fer-li saber que alguna cosa passarà amb aquest bloc. Concretament, els blocs actius tremolen de manera lateral (en direcció X o Z, perpendicular a la direcció del segment al qual es troben) mentre que els reactius ho fan verticalment.

Per evitar canvis sobtats, quan els blocs arriben a una certa altura, començen a desaparèixer gradualment creat un efecte de fade out.

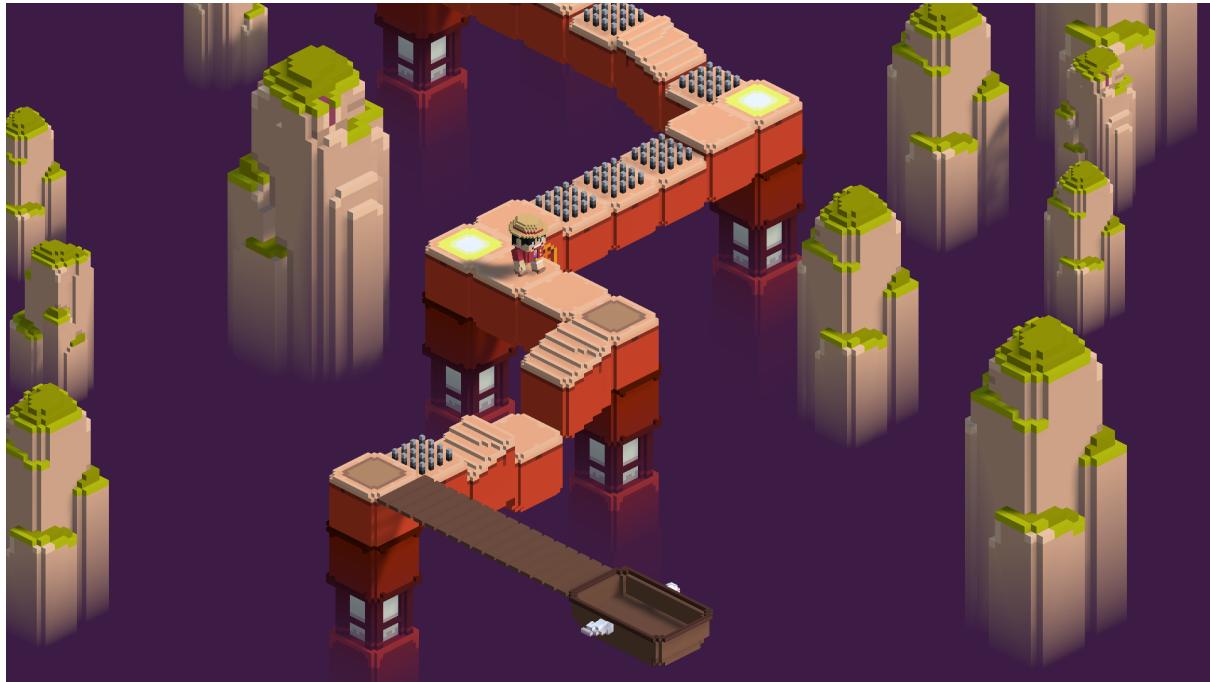


Figura 10. Blocs inestables que estan caient

2.10. Monedes

Les monedes son l'únic col·leccionable que el jugador pot agafar. Tenen forma de disc i giren constantment sobre sí mateixos, amb un halo de llum representant la brillantor que emet.

Quan el jugador recull una moneda, aquesta es mou cap amunt i desapareix, representant que ha sigut agafada.

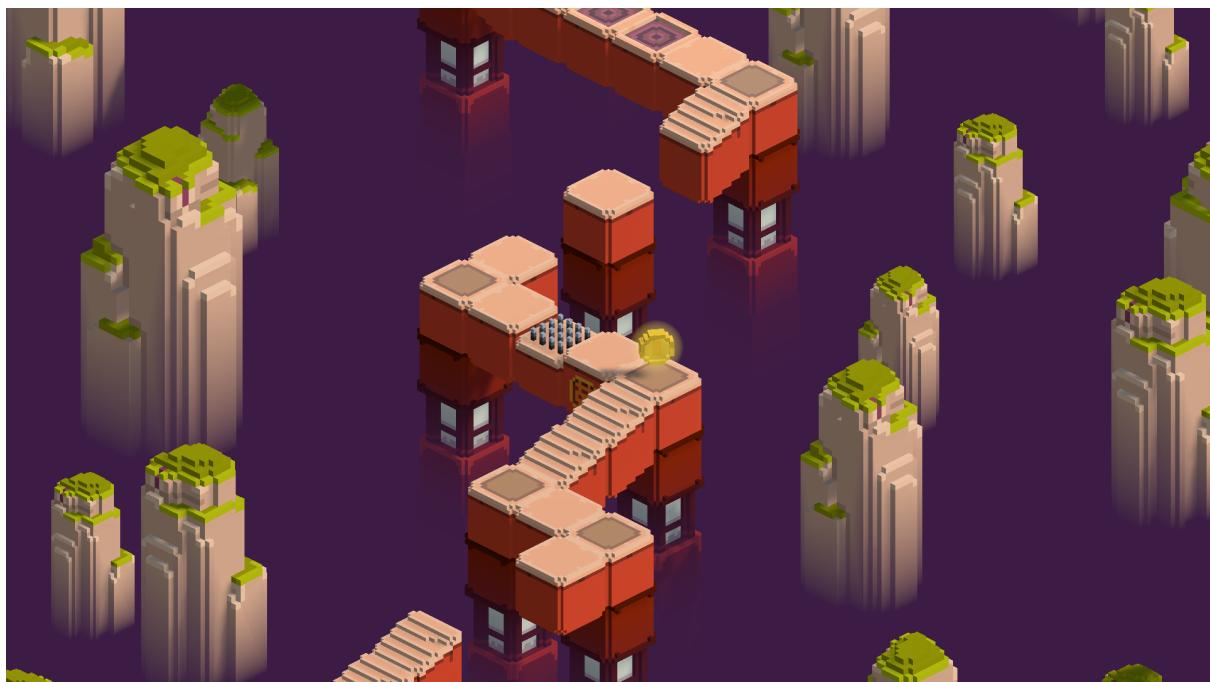


Figura 11. Moneda

2.11. God Mode

El God Mode es pot activar amb la tecla ‘G’. Quan s’activa, el jugador salta i gira allà on sigui necessari. Tots els obstacles mantenen el seu comportament original a excepció de les serres. Com que les serres tenen un grau de complexitat més elevat a l’hora d’esquivar-les (posició imprèdictible, properes a caselles buides...), s’ha decidit ocultar-los amb el God Mode, i tornen a aparèixer una vegada es desactiva.

Internament, hi ha una col·lisió invisible que només es detecta amb el God Mode activat allà con el jugador hagi de prèmer la tecla espai per girar o saltar.

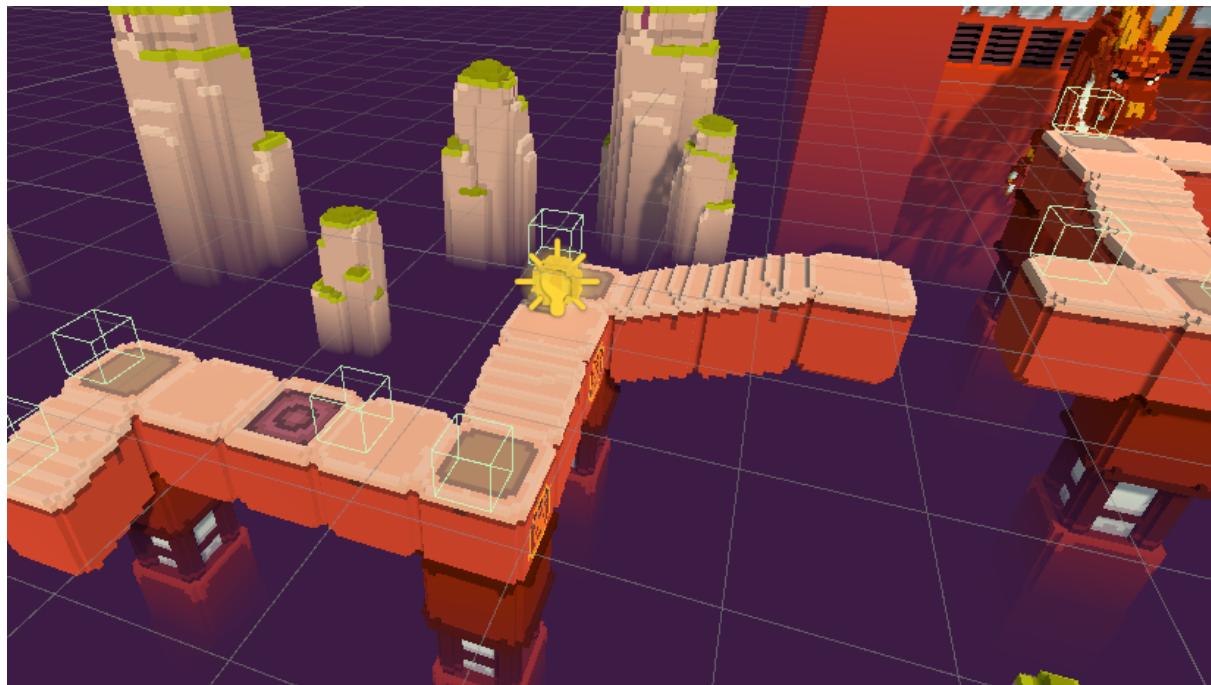


Figura 12. Col·lisions del God Mode

3. Apartat artístic

Per l'apartat artístic s'ha de donar crèdit sobretot a Mana Cube, ja que molts models 3D i efectes de so s'han extret del joc original, mantenint el mateix aspecte o bé fent una modificació per adaptar-lo a aquesta versió per escriptoris.

3.1. Models 3D i Animacions

Tots els models s'han fet manualment amb el programa MagicaVoxel. S'ha optat per fer servir aquest programa ja que és molt senzill crear models 3D de voxel art. Això ha permès una total llibertat a l'hora de donar vida al joc, ja que es poden fer ampliacions segons sorgeixen idees sobre diferents mecàniques del joc, mantenint una coherència entre tots els models, cosa que no hagués sigut possible si s'hagués optat per afagar uns assets ja fets.

A algunes textures se'ls ha aplicat un shader personalitzat per aconseguir un efecte de fog molt semblant al videojoc Monument Valley, i és una modificació del shader que es proposa en aquest [article](#) publicat per Spennerino que, en comptes de fer el gradient segons les coordenades a l'espai del món, es fan segons la coordenada a l'espai de l'objecte, ja que el que es volia aconseguir era el mateix efecte pel mateix model però situat en diferents altures.

Per aquells models amb col·lisió que tenen una forma diferent a les formes predefinides al Unity, s'ha fet servir el package ProBuilder. Concretament, s'ha utilitzat per definir la col·lisió dels blocs d'escala.

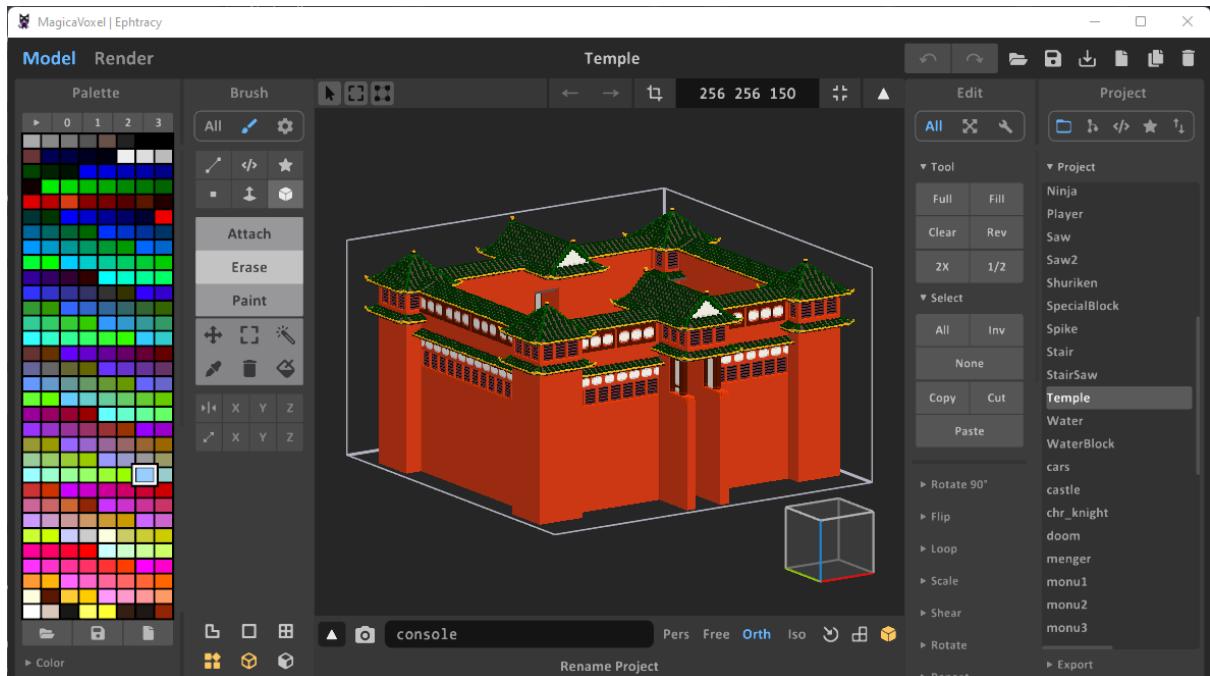


Figura 13. Interfície del MagicaVoxel

Per fer les animacions s'ha fet servir el programa VoxEdit. Aquest programa és compatible amb MagicaVoxel i també es molt fàcil de fer servir. Resumidament, per fer una animació basta amb situar-se en diferents timelines i articular el model, obtenint d'aquesta manera la interpolació desitjada. Es pot exportar com un arxiu DAE i es pot importar directament a Unity.

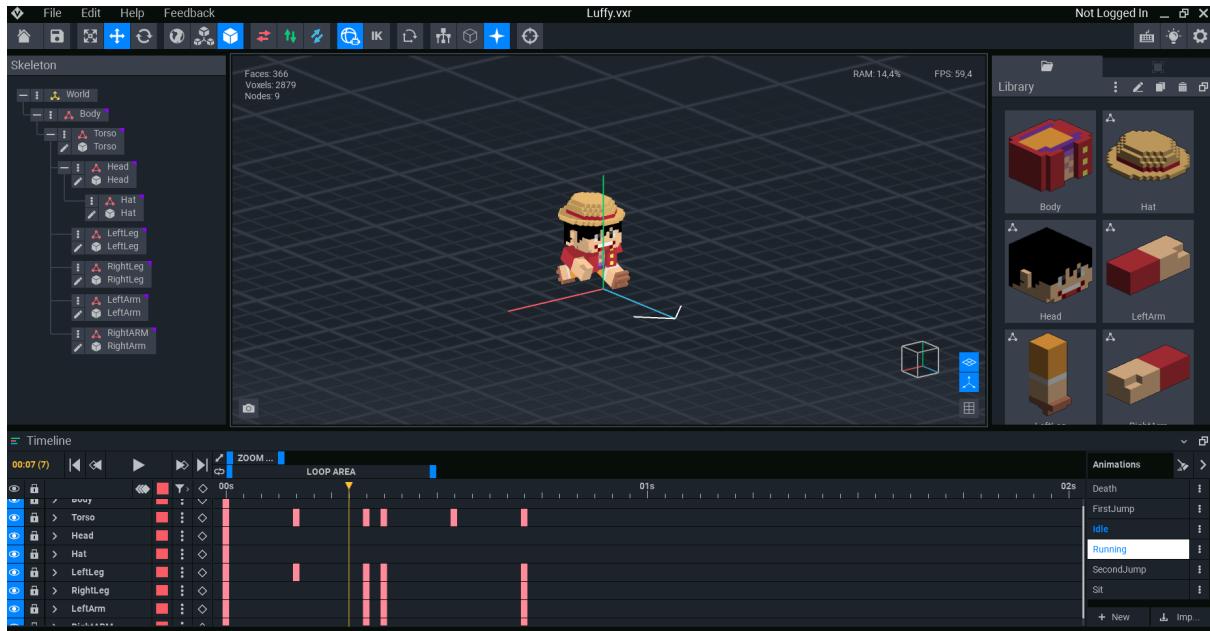


Figura 14. Interfície del VoxEdit

3.2. Música i So

La música que s'ha fet servir és una banda sonora de la sèrie One Piece, concretament el tema "[Opening Curtains, Closing Curtains](#)". Aquesta soa quan el jugador inicia la partida i comença a còrrer.

Pel que fa als efectes de so, alguns s'han extret del joc original, en concret els sons de salts, activació del marcador, trepitjar la tolla de fang, agafar una moneda i caiguda de blocs inestables. Els sons que fa el personatge (crits i rialles) s'han extret del protagonista de la sèrie One Piece. El so que es fa quan el personatge cau al buit s'ha extret del Minecraft. La resta de sons: de serra, del shuriken (llançament i col·lisió) i de punxes s'han extret d'internet i s'han modificat lleugerament.

4. Metodología

En la realització d'aquest projecte, l'objectiu principal és aconseguir repartir les tasques de forma independent, a través de reunions per compartir el progrés i prendre decisions. Inicialment, no sabíem ben bé com repartir aquest treball, ja que ens va semblar complicat dividir un joc com aquest en els quals hi ha una dependència important entre els objectes de l'escena.

Setmana 1 (24/04 - 30/04)

La primera setmana l'hem dedicat a entendre i descobrir els mecanismes utilitzats en el joc original, mitjançant la jugada de diverses partides de Cliff Hopper per observar el moviment del personatge, els obstacles que s'enfronten i les interaccions que es produeixen. A més, hem buscat comprendre individualment com funciona Unity mitjançant tutorials per poder desenvolupar el joc i hem dividit el treball.

A un principi vam proposar un dels membres s'encarregués del base del joc i l'altre del disseny, però final vam veure que la càrrega de treball era bastant desequilibrat, i vam decidir dividir-la en dues parts: un s'encarregarà principalment del personatge i l'altra del cub que persegueix el personatge. D'aquesta manera, podrem avançar en el treball de forma paral·lela.

Setmana 2 (01/05 - 07/05)

Durant la segona setmana hem estat codificant codi paral·lelament. Hem estat comunicant-nos a través de xarxa social per saber els progressos i com avança cada un. Per unificar el codi que hem escrit, teníem pensat de crear un repositori a Github, però aquest no es permet els fitxers Assets, i vam decidir utilitzar el Drive i Discord per transferir el codi i unificar-lo de forma manual.

Setmana 3 (08/05 - 14/05)

Tenint en compte que ja teníem un codi base del jugador, aquesta setmana vam decidir crear la mapa pel qual el jugador haurà de córrer, juntament amb els obstacles que apareixen en el camí. Vam estar modificant el codi del jugador i de la capsula per tal que tinguin cura d'aquests obstacle i el disseny de la mapa.

Setmana 4 (15/05 - 21/05)

A la quarta setmana ens hem dedicat a efectes del joc i les interaccions que es duen a terme quan jugador es col·lisioni amb un obstacle o monedes, també hem revisat les capses de col·lisions per tal que el jugador faci una reacció de manera correcta. També hem acabat les rotacions del cub, però aquest no estava ben fixat, ja que després de la rotació es produïa una petita desviació i caldia arreglar-lo.

Setmana 5 (22/05 - 28/05)

A la setmana 5 vam estar refinant els detalls del jugador amb l'escena i el cub, i començar a incorporar sons i músiques, també hem redimensionat les capses de col·lisions de les rajoles que estan als blocs de la cantonada de tal forma el jugador pugui fer el canvi de direcció correctament, també ens hem posat d'acord amb el disseny de la interfície d'usuari.

Setmana 6 (29/05 - 04/06)

Durant l'última setmana abans de l'entrega, vam estar intentant fixar els errors que produïa la capsa, però finalment no hem pogut resoldre, i vam decidir no incorporar-lo dins del joc. De l'altra banda hem estat terminant les tasques no finalitzades a la setmana anterior: les sons, la música i interfície d'usuaris. També hem estat progressant la memòria, i revisar el joc que no hi hagi cap error durant el joc i entre canvis d'estats, un cop revisat hem gravat el vídeo de demo.

A continuació [Figura 15], es mostra un diagrama de Grantt on podeu observar la distribució temporal de cada tasca respecte al temps total del projecte.

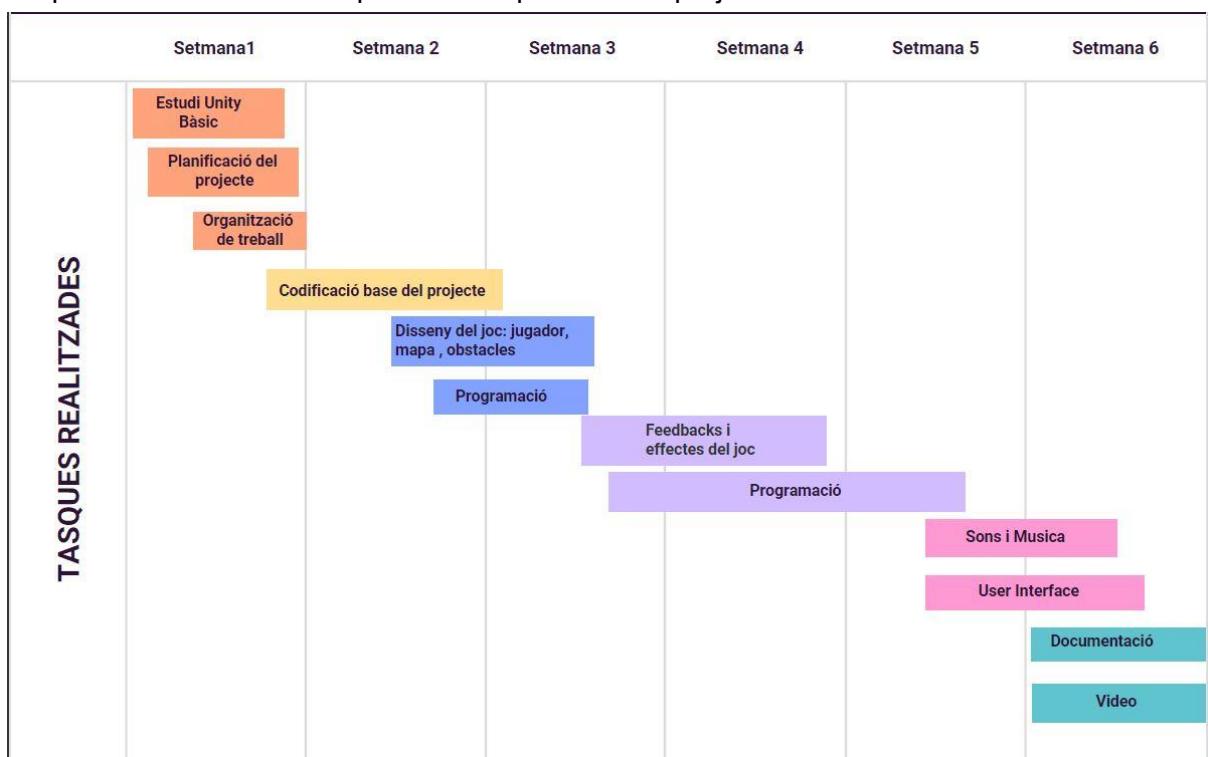


Figura 15. Diagrama de Grantt

5. Conclusions

El grau de complexitat d'aquest projecte, tot i havent fet servir Unity i no haver-ho fet "from scratch", ens ha semblat més elevat que el joc 2D.

Aquest és el primer videojoc 3D que hem fet, també cal dir Unity és una nova eina per nosaltres, tot i no ser un motor complicat d'adquirir, hem hagut de dedicar temps per adaptar-nos-hi. No obstant això, és una eina poderosa que ha permès obtenir resultats notables. En comparació amb el nostre projecte anterior, l'ús d'aquest motor ens ha estalviat temps en diversos àmbits. Per exemple, en el cas de les caixes de col·lisions, en el joc de dues dimensions havíem de crear les nostres pròpies caixes per a cada tipus d'enemic i adaptar-les segons les diferents animacions.

Com a conclusió del projecte podem afirmar que el desenvolupament de jocs requereix un gran esforç i la implicació de diverses disciplines. Aquest ha estat un dels projectes més significatives que hem realitzat fins ara. Hem de destacar que l'assignatura ens ha agradat molt, ja que ha estat una experiència increïble que ens ha permès ampliar el nostre coneixement i experiència en el camp del desenvolupament de videojocs.

6. Bibliografia

“Isometric Camera in Unity”

<https://www.youtube.com/watch?v=optfGhipg-w>

Vídeo que ensenya com configurar una càmera isomètrica (que en realitat és dimètrica) a Unity.

“Simple Character Controller in Unity”

<https://www.youtube.com/playlist?list=PLBcfp6HMOJwzDcdCzoAx3jKm7slcBXJZ>

Sèrie de vídeos que ensenya com configurar un jugador senzill amb un controlador a Unity.

“Cómo exportar de Magicavoxel a Unity”

<https://www.youtube.com/watch?v=ojNBO291tFA>

Vídeo que ensenya com exportar un model de voxel art creat amb Magicavoxel i animat amb Voxedit i importar-ho a Unity.

“Cómo animar un personaje de Magicavoxel con Voxedit”

<https://www.youtube.com/watch?v=HuQ-sXiSvT4&t=2s>

Vídeo que ensenya com animar un personatge de Magicavoxel amb Voxedit.

“Position Based Gradient Shader”

<https://spennerino.wordpress.com/2017/06/20/gradient-shader/>

Article que parla de com configurar un shader per obtenir un efecte de fog similar al joc Monument Valley.

“Unity: Scripting API”

<https://docs.unity3d.com/ScriptReference/>

Documentació de l'API de Unity.

“ Rolling a cube - Unity Tutorial”

https://www.youtube.com/watch?v=V9rVZ9mf0uA&t=5s&ab_channel=Tarodev

Vídeo que ensenya com rotejar un cub a Unity

“Unity Panel Manager”

<https://www.youtube.com/watch?v=texonivDsy0>

Vídeo que ensenya com interactuar diferents panels de Canva a Unity

“The Easiest Way to Make a High Score in Unity”

https://www.youtube.com/watch?v=Y7GjVFFSMuI&t=79s&ab_channel=BMo

Vídeo que ensenya com actualitzar i guardar la màxima puntuació obtinguda en diferents partides a Unity

“AUDIO MIXERS In Unity”

https://www.youtube.com/watch?v=pbuJUaO-wpY&t=510s&ab_channel=KapKoder

Vídeo que ensenya com configurar la música i els sons utilitzant AudioMixers mitjançant un Silder a Unity