

Generació procedimental 3D mitjançant l'algorisme Wave Function Collapse basat en l'aprenentatge probabilístic de patrons

Luis Adrián Gómez Batista

30 de juny de 2026

Resum – Aquest Treball de Fi de Grau presenta una eina desenvolupada a Unity (C#) per a la generació procedimental d'estructures 3D mitjançant l'algorisme Wave Function Collapse (WFC). La solució adapta i optimitza l'algorisme per a entorns volumètrics mitjançant un sistema de sockets estructurals, parametrització configurable i mecanismes d'aprenentatge probabilístic que permeten generar entorns coherents a partir de models de referència. El resultat és una eina flexible, escalable i reutilitzable per a la creació automàtica d'entorns virtuals tridimensionals.

Paraules clau – aprenentatge probabilístic, estructures 3D, generació procedimental, parametrització, sistemes de sockets, Wave Function Collapse.

Abstract – This Final Degree Project presents a Unity (C#) tool for the procedural generation of 3D structures using the Wave Function Collapse (WFC) algorithm. The solution adapts and optimizes the algorithm for volumetric environments through a structural socket system, configurable parameterization, and probabilistic learning mechanisms that guide generation from reference models. The result is a flexible, scalable, and reusable tool for the automatic creation of coherent three-dimensional virtual environments.

Keywords – 3D structures, parameterization, probabilistic learning, procedural generation, socket systems, Wave Function Collapse.



1 INTRODUCCIÓ I CONTEXT

LA Generació Procedimental de Contingut (PCG) es defineix com un conjunt de tècniques algorítmiques destinades a la creació automàtica de contingut digital mitjançant regles formals i processos estocàstics [1]. En lloc de dependre del modelatge manual, els sistemes PCG combinen primitives geomètriques —cel·les de gralla, blocs tridimensionals o patrons elementals— segons restriccions predefinides, de manera que l'estructura global emergeix de la seva composició. Aquest enfocament s'ha consolidat com una tècnica àmpliament utilitzada en la indústria del videojoc i les simulacions d'alta fidelitat, on es requereix escalabilitat sense incrementar proporcionalment els costos de producció.

Des d'un punt de vista tècnic, una de les eines més re-

levantes per a la generació de formes naturals és el **soroll coherent**, una família de funcions matemàtiques dissenyades per produir variació gradual i contínua en l'espai. A diferència del soroll aleatori pur, el soroll coherent garanteix que valors propers en l'espai també siguin propers en valor, la qual cosa el fa especialment adequat per imitar la irregularitat orgànica dels entorns naturals. La tècnica més estesa dins d'aquesta família és el Perlin Noise [2], que presenta continuïtat espacial i permet generar patrons continus que representen superfícies complexes com el relleu de muntanyes. A més, la majoria de sistemes PCG es basen en mecanismes deterministes controlats mitjançant llavors (*seeds*) [1], que garanteixen resultats reproduïbles conciliant variabilitat i consistència. Un exemple representatiu és No Man's Sky [3], on la coherència de l'univers no depèn d'un disseny manual exhaustiu, sinó de regles aplicades a les funcions generadores. No obstant això, aquests mètodes presenten limitacions importants quan es requereix el compliment de restriccions estructurals estrictes o patrons arquitectònics consistents, fet que motiva l'ús d'algorismes més expressius com el Wave Function Collapse.

- E-mail de contacte: luis.adrian.gomez.batista@gmail.com
- Menció: Computació
- Treball tutoritzat per: Robert Benavente Vidal (Departament de Ciències de la Computació)
- Curs 2025/26

2 ESTAT DE L'ART

L'any 2016, Maxim Gumin va publicar el Wave Function Collapse (WFC) [4], un algorisme inspirat en la síntesi de textures per exemplificació i en conceptes de la mecànica quàntica. En el seu mode *Overlapping*, el sistema extreu automàticament tots els patrons locals de $N \times N$ cel·les presents en una imatge d'exemple i genera noves estructures on tots els patrons observats es respecten. En el mode *Simple Tiled*, treballa amb peces predefinides i regles de compatibilitat explícites entre mòduls, oferint un control més exhaustiu sobre el resultat a costa d'una configuració manual de les relacions d'adjacència. La popularitat de l'algorisme va créixer ràpidament gràcies a recursos divulgatius com la demo interactiva d'Oskar Stålberg [5] i tutorials tècnics com els de Sebastian Lague [6] i Robert Heaton [7], que van facilitar la seva adopció en el desenvolupament de videojocs independents.

La transició del WFC a dominis tridimensionals ha estat abordada per diversos desenvolupadors, constituint un dels fronts més actius dins la recerca en PCG. Marian Kleinberg va documentar una de les implementacions de referència més rellevants [8], generant ciutats procedimentals mitjançant una adaptació del mode *Simple Tiled* al volum tridimensional. De forma complementària, Boris the Brave va aprofundir en els mecanismes de propagació de restriccions del WFC, concretament l'algorisme AC-3 (*Arc Consistency 3*) [9], estudiant com aquest permet gestionar de forma eficient les restriccions en graelles volumètriques. Aquests treballs constitueixen les referències pràctiques més rellevants per a l'extensió del WFC a tres dimensions i serveixen de base per a les implementacions posteriors.

3 MOTIVACIÓ I REPTES DEL PROJECTE

Les implementacions tridimensionals existents del WFC depenen majoritàriament de regles d'adjacència definides manualment, un enfocament altament propens a errors i amb una escalabilitat pràcticament nul·la en catàlegs de peces complexos. A diferència d'aquestes aproximacions, el present treball proposa una hibridació entre el mode *Simple Tiled* i la filosofia d'aprenentatge de l'*Overlapping*: el sistema analitza automàticament una escena de referència tridimensional, n'extreu la distribució probabilística de peces i adjacències, i utilitza aquesta informació per orientar estadísticament les generacions futures sense perdre la flexibilitat que ofereix el catàleg de mòduls discrets amb sockets semàntics.

Per aconseguir-ho, cal afrontar tres reptes principals. En primer lloc, el volum de l'espai de generació creix de forma cúbica i cada cel·la interacciona amb sis veïns, fet que intensifica la propagació de restriccions i augmenta el cost computacional de forma significativa. En segon lloc, l'expansió de l'espai d'estats incrementa la probabilitat de conflictes lògics que obliguen l'algorisme a recórrer a mecanismes de retrocés (*backtracking*), amb un impacte directe sobre el rendiment. En tercer lloc, la definició i gestió automatitzada de les regles d'adjacència en 3D —sense dependre de configuracions manuals— constitueix un problema obert que aquest treball aborda com a objectiu central.

4 OBJECTIUS DEL PROJECTE

Per tal d'abordar els reptes computacionals i de complexitat estructural associats a l'extensió de l'algorisme WFC a entorns tridimensionals, aquest treball es divideix en una sèrie d'objectius orientats a donar resposta a les problemàtiques identificades.

4.1 Objectius principals

- **Implementació del prototip WFC 3D:** Desenvolupar un entorn funcional capaç de traslladar la lògica de l'algorisme a un espai volumètric, executant el procés de col·lapse i la propagació de restriccions en sis direccions sobre una graella tridimensional de cel·les.
- **Investigació de l'aprenentatge probabilístic per a la síntesi de models:** Estudiar i implementar mètodes d'extracció de patrons a partir de models 3D existents amb l'objectiu d'automatitzar la generació de regles d'adjacència. Es busca que l'algorisme aprengui la distribució probabilística dels elements d'entrada i pugui generar noves estructures tridimensionals que, tot i ser inèdites, mantinguin la coherència i l'estètica del model original.

4.2 Objectius secundaris

- **Disseny d'un sistema d'etiquetatge modular (sockets):** Definir un protocol de nomenclatura i categorització per a les cares dels models 3D (sockets), amb l'objectiu d'estandarditzar les regles de compatibilitat. Això permet simplificar la validació d'adjacències i facilitar la integració de nous models sense ambigüitats, assegurant la connectivitat lògica entre peces.
- **Anàlisi i optimització del rendiment computacional:** Investigar i aplicar estratègies d'optimització algorísmica per reduir la càrrega de processament en entorns tridimensionals. Es posarà èmfasi en la millora de la gestió de dades i en la reducció del cost de propagació, garantint temps d'execució viables per a volums complexos.
- **Desenvolupament d'un sistema de parametrització funcional:** Dissenyar una estructura de dades i variables de control que permetin ajustar el comportament de l'algorisme des del codi. Aquest sistema facilitarà la configuració de paràmetres crítics com la llavor (*seed*), les probabilitats de col·lapse i els límits de propagació, fent el sistema adaptable a diferents escenaris sense modificar la lògica interna.

5 METODOLOGIA

Per al desenvolupament d'aquest projecte s'adoptarà una metodologia iterativa, basada en la implementació progressiva dels diferents components del sistema. Aquest enfocament permet validar cada funcionalitat a mesura que s'incorpora, detectar errors de coherència estructural i analitzar l'impacte computacional de les restriccions pròpies de l'algorisme WFC en entorns tridimensionals.

5.1 Eines i tecnologies utilitzades

El projecte s'implementa en C# sobre Unity [10], que actua com a motor principal per a la visualització i la depuració del sistema en temps real. El control de versions es gestiona amb Git i GitHub. Per al disseny dels mòduls s'utilitza OpenSCAD per la seva naturalesa paramètrica, complementat amb actius de Kenney [11]; els models es processen a Blender abans d'exportar-los a Unity com a *prefabs*. Tot el procés s'acompanya de documentació tècnica progressiva que formarà part de la memòria final.

5.2 Metodologia algorísmica del WFC

El nucli metodològic del projecte es basa en el funcionament intern de l'algoritme Wave Function Collapse (WFC). En la seva forma base, l'espai de generació es representa mitjançant una graella de cel·les on cada unitat es troba inicialment en un estat de superposició, contenint totes les configuracions possibles compatibles amb el sistema.

L'execució de l'algoritme es desenvolupa de manera iterativa seguint tres fases principals. En primer lloc, es realitza la fase d'**observació**, en la qual es calcula l'entropia de totes les cel·les per seleccionar aquella amb menor nombre d'estats compatibles. Aquesta estratègia permet reduir la probabilitat d'arribar a configuracions irresolubles.

A continuació, es duu a terme la fase de **col·lapse**, moment en què la cel·la seleccionada fixa un únic estat definitiu d'entre totes les opcions possibles, seguint una distribució de probabilitats associada als patrons disponibles.

Aquest cicle es repeteix fins que la graella queda completament resolta o fins que es detecta una contradicció. Per il·lustrar aquest funcionament, a la Figura 1 es mostra el pseudocodi base del cicle principal i del procés de propagació, els quals seran posteriorment adaptats a un entorn tridimensional.

Un cop definida la lògica formal de l'algoritme mitjançant el pseudocodi, resulta útil observar com aquestes operacions es tradueixen visualment durant l'execució. La Figura 2 sintetitza de manera gràfica el comportament iteratiu del *Wave Function Collapse* a partir d'un exemple extret de la demo interactiva desenvolupada per Oskar Stålberg [5].

En primer lloc, es mostra el **conjunt de peces possibles**, és a dir, el domini de patrons que l'algoritme pot utilitzar per construir l'entorn. Cada peça defineix restriccions d'adjacència estrictes que determinaran quines combinacions són vàlides durant la generació.

A continuació, es representa l'**estat inicial de superposició màxima**, en què totes les cel·les de la graella contenen simultàniament totes les opcions compatibles. La línia de connexió visual indica que cadascuna d'aquestes cel·les no conté una peça definitiva, sinó el conjunt complet de possibilitats representat a l'esquerra, equivalent a un "zoom conceptual" del domini intern de cada posició de la graella.

Seguidament, es produeix el **primer col·lapse d'una cel·la**, moment en què l'algoritme selecciona una posició concreta i redueix el seu estat a una única configuració definitiva. Aquesta decisió desencadena immediatament la **propagació de restriccions**, eliminant de les cel·les adjacents totes aquelles peces incompatibles amb la nova selecció. Com a conseqüència, l'espai de possibilitats es redueix

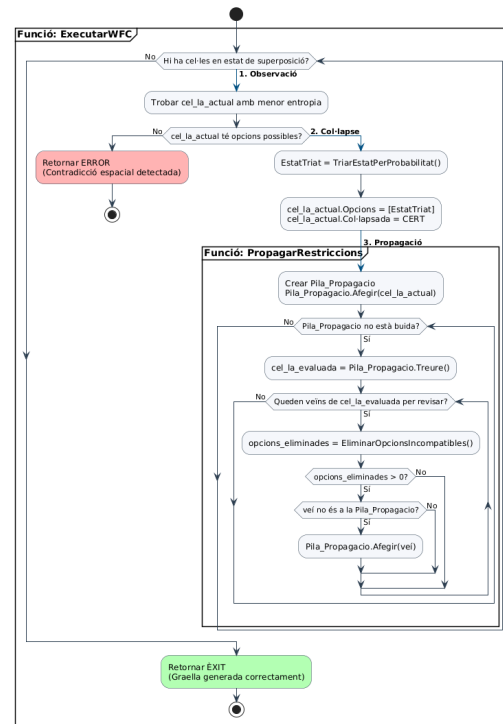


Fig. 1: Representació lògica del funcionament base del Wave Function Collapse. Aquest pseudocodi il·lustra la teoria bidimensional estàndard, la qual s'adaptarà estructuralment i s'optimitzarà en C# per suportar la càrrega computacional de l'entorn tridimensional del projecte.

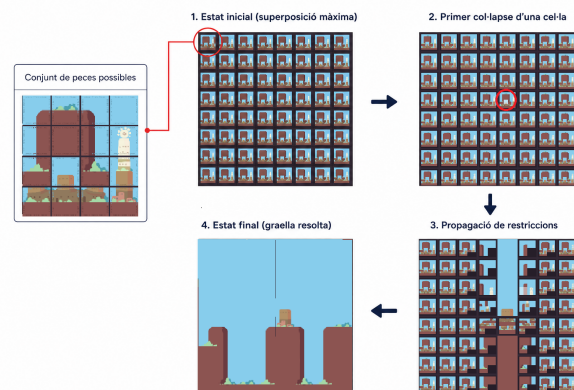


Fig. 2: Representació visual del procés iteratiu del Wave Function Collapse (WFC).

progressivament i l'entropia global del sistema disminueix.

Finalment, després de múltiples iteracions de col·lapse i propagació, la graella arriba a un **estat completament resolt**, en què totes les cel·les han estat assignades respectant estrictament les regles d'adjacència. El resultat és una composició global coherent obtinguda a partir d'un conjunt relativament reduït de regles locals.

6 PLANIFICACIÓ

El projecte es desenvoluparà al llarg de 19 setmanes amb una dedicació total estimada de 300 hores, seguint una estructura de desenvolupament progressiu des de l'anàlisi inicial fins a l'avaluació final. La planificació s'organitza en

cinc fases principals:

Fase 1: Anàlisi i configuració (45h). Aquesta fase se centra en la base teòrica i la preparació de l'entorn de desenvolupament.

- **Revisió de l'estat de l'art (25h):** estudi de la generació procedimental de contingut i de l'algorisme Wave Function Collapse, amb especial atenció a limitacions i extensions a 3D.
- **Configuració de l'entorn (20h):** instal·lació de Unity, control de versions amb Git i definició de l'arquitectura inicial del sistema i estructures de dades.

Fase 2: Prototip WFC 3D (65h). Desenvolupament del nucli funcional de l'algorisme.

- **Implementació de la graella i primitives (15h):** creació de l'espai tridimensional i dels mòduls visuals bàsics.
- **Lògica central del WFC (35h):** implementació de l'entropia de Shannon, col·lapse de cel·les i propagació de restriccions amb cua de processament.
- **Validació inicial (15h):** proves en graelles reduïdes i depuració amb eines visuals.

Fase 3: Aprenentatge probabilístic (80h). Introducció de comportament estadístic basat en models d'entrada.

- **Extracció de patrons (30h):** anàlisi d'un model 3D per obtenir freqüències i relacions entre peces.
- **Integració probabilística (35h):** substitució de la selecció uniforme per un sistema ponderat.
- **Ajust del sistema (15h):** calibratge de pesos i validació amb múltiples seeds.

Fase 4: Optimització i usabilitat (65h). Millora del rendiment i flexibilitat del sistema.

- **Sistema de sockets (30h):** substitució de regles complexes per etiquetatge de connexions [9].
- **Optimització (20h):** reducció de consum de memòria i millora del rendiment.
- **Configuració del sistema (15h):** exposició de variables com seed i probabilitats.

Fase 5: Avaluació i redacció final (45h). Validació del sistema i tancament del projecte.

- **Proves de rendiment (15h):** anàlisi amb Unity Profiler i tests de càrrega.
- **Redacció de la memòria (20h):** elaboració del document final amb resultats i arquitectura.
- **Preparació de la defensa (10h):** creació de presentació i material audiovisual.

El conjunt de tasques inclou també el temps dedicat a tories i documentació contínua, garantint la coherència entre desenvolupament tècnic i redacció acadèmica. L'Annex 1 mostra la distribució temporal mitjançant un diagrama de Gantt, on es visualitza la seqüència de les 19 setmanes de treball i la dependència entre fases.

7 DESENVOLUPAMENT

L'estratègia metodològica aplicada ha mantingut el seu caràcter iteratiu i incremental, prioritant la validació contínua de cada component abans d'escalar en complexitat. El procés d'implementació tècnica i la resolució d'obstacles algorísmics han derivat en evolucions significatives respecte al plantejament inicial, les quals han permès assolir un sistema més robust i autònom. Aquest apartat descriu els aspectes tècnics més rellevants del desenvolupament, des del disseny dels mòduls fins a la implementació de l'algorisme i els mecanismes d'aprenentatge probabilístic.

7.1 Disseny dels mòduls paramètrics

Davant la manca d'actius gratuïts adequats per al projecte, s'han dissenyat 32 mòduls propis amb temàtica de canonades industrials. Cada peça parteix d'un node central esfèric des del qual neixen braços cilíndrics que acaben en un connector de tipus "mascle" (vegeu la Figura 3), cosa que permet identificar visualment de forma immediata si la connectivitat entre mòduls és correcta. El conjunt cobreix totes les configuracions necessàries: tubs rectes, colzes, unions en T, creus de 4, 5 i 6 braços i terminals.



Fig. 3: Exemples de mòduls paramètrics dissenyats per al sistema.).

7.2 Sistema de connectivitat i sockets semàntics

Per gestionar les restriccions d'adjacència entre mòduls, s'ha implementat un sistema d'identificadors compartits basat en una nomenclatura determinista i bidireccional. Cada interfície de contacte entre dues peces queda descrita per un únic codi amb la sintaxi `c{idxMenor}{dirMenor}_{idxMajor}{dirMajor}`, on `idx` és l'índex numèric del mòdul i `dir` és la direcció de la cara (n, s, e, o per als eixos horitzontals, i u, d per a dalt i baix). Això es tradueix en cadenes com `c14n.26s`. Els descriptors s'ordenen sempre pel valor numèric menor primer, de manera que ambdues cares d'una mateixa interfície compatible reben exactament el mateix identificador independentment de l'ordre de processament.

Per garantir l'eficiència durant les iteracions crítiques de col·lapse, es va descartar la comparació directa de cadenes de text a causa del seu elevat cost computacional. En el seu lloc, el sistema aplica una funció de dispersió (*hash*) posici-

onal determinista en base-31, formalitzada segons l'Equació 1:

$$H(S) = \left(\sum_{i=0}^{n-1} \text{ord}(S[i]) \cdot 31^{n-1-i} \right) \pmod{m} \quad (1)$$

on $S[i]$ representa el caràcter en la posició i -èssima de la cadena, $\text{ord}(c)$ retorna el valor numèric ASCII del caràcter c , i m és un mòdul primer de gran magnitud. Aquest algorisme garanteix que una mateixa cadena es tradueixi invariablement en el mateix enter de 32 bits en qualsevol dispositiu i sessió.

7.3 Visió general de l'arquitectura

El sistema desenvolupat s'estructura en quatre blocs funcionals seqüencials, cadascun responsable d'una fase concreta del flux de treball. La Figura 4 recull de forma visual tots els components implementats i les dependències entre ells, i es recomana consultar-la paral·lelament a la lectura d'aquest apartat.

El primer bloc comprèn les eines responsables del **diseny i exportació d'actius**. OpenSCAD s'utilitza per al modelatge paramètric de les peces, garantint la precisió geomètrica dels connectors. L'aplicació *WFC Pipeline*, desenvolupada en Python, automatitza la conversió dels models a un format compatible amb Unity mitjançant Blender, corregint l'orientació dels eixos i eliminant la necessitat d'intervenció manual en aquest procés.

El segon bloc gestiona la **configuració i inicialització** del catàleg de peces. El component *PrefabAutoSetup* processa el fitxer `sockets.txt`, que conté les regles de connexió entre cares, i crea automàticament tots els *prefabs* del sistema injectant-hi els identificadors de connector de cada cara. El component *WFCPrefabLoader* s'encarrega posteriorment de vincular tot el catàleg al motor de generació amb una única operació, evitant la configuració manual de cada actiu.

El tercer bloc constitueix el **pipeline d'aprenentatge probabilístic**. A partir d'una escena de referència construïda manualment per l'usuari, el component *WFCResultDataLogger* analitza les relacions espacials entre els mòduls de la referència, obtenint les freqüències d'aparició de cada peça i les relacions de veïnatge per direcció. Aquestes dades alimenten dos exportadors: el *WFC.SocketsExporter*, que genera el fitxer `sockets.txt` amb les regles d'adjacència vàlides, i el *WFCDataExporter*, que serialitza la distribució probabilística apresada en el fitxer `WFCRuleSet.json`, recollint la probabilitat de cada peça veïna possible per a cadascuna de les sis direccions de l'espai.

El quart bloc constitueix el **motor de generació WFC** i és el nucli computacional del sistema. El component *WFCDataLoader* carrega el model probabilístic en memòria i exposa els pesos i probabilitats al motor de col·lapse. Prèviament a qualsevol generació, el *WFCAnalyzer* realitza un diagnòstic de la configuració de sockets, detectant inconsistències que poguessin derivar en contradiccions durant l'execució. Finalment, el *WFCManager* actua com a nucli de l'algorisme: governa el col·lapse cel·la per cel·la, executa la propagació de restriccions en les sis direccions de l'espai i integra el context

probabilístic après per orientar la generació cap a l'estil del model de referència.

7.4 Aprenentatge probabilístic i model de generació

El sistema ofereix dues modalitats per definir les regles d'adjacència. La primera és la **configuració manual**, on l'usuari assigna directament els identificadors de socket a cada cara de cada peça. La segona, i principal aportació del projecte, és l'**aprenentatge automàtic** a partir d'un model de referència: l'usuari construeix una escena tridimensional d'exemple a Unity i el sistema n'extreu les relacions d'adjacència, generant automàticament els fitxers `sockets.txt` i `WFCRuleSet.json` que guiaran totes les generacions futures.

Extracció de la informació estadística

El component *WFCResultDataLogger* escaneja tots els mòduls del volum de referència i extreu dues estructures de dades: la freqüència global d'aparició de cada tipus de peça i una taula d'adjacència que recull, per a cada peça i cadascuna de les sis direccions de l'espai, la freqüència amb la qual cada tipus de veí hi ha aparegut. A partir d'aquestes dades, el *WFCDataExporter* genera el fitxer `WFCRuleSet.json`, on cada peça emmagatzema la probabilitat condicional de cada veí possible per a cada direcció, expressada com $P(\text{veí} \mid \text{peça, direcció})$.

Integració del context probabilístic durant la generació

Durant la generació, el *WFCDataLoader* carrega el fitxer `WFCRuleSet.json` en memòria i exposa els pesos i probabilitats al motor de col·lapse, aplicant un llindar mínim de 0,01 per evitar bloquejos derivats de l'absència de mostres en combinacions poc freqüents.

Al llarg del sistema s'utilitza la interpolació lineal entre dos valors a i b regulada pel paràmetre $\omega \in [0, 1]$, definida com:

$$\text{Lerp}(a, b, \omega) = (1 - \omega) \cdot a + \omega \cdot b \quad (2)$$

El càlcul del pes final de cada peça candidata es desenvolupa en dos passos. En primer lloc, el pes base s'obté interpolant entre el pes assignat manualment des de l'Inspector i el pes global après del fitxer JSON, regulat pel paràmetre *influènciaAprenentatge* $\omega \in [0, 1]$:

$$\text{pes_base} = \text{Lerp}(\text{pes_inspector}, \text{pes_JSON}, \omega) \quad (3)$$

En segon lloc, es calcula el modificador contextual a partir de les probabilitats dels veïns ja col·lapsats. Per evitar la degradació numèrica que produiria la multiplicació directa de probabilitats properes a zero, s'utilitza la mitjana geomètrica en formulació logarítmica:

$$\text{MG} = \exp\left(\frac{1}{n} \sum_{i=1}^n \ln(P_i)\right) \quad (4)$$

Aquest modificador s'incorpora al pes base mitjançant una interpolació lineal també regulada per ω :

$$\text{pes_final} = \text{pes_base} \times \text{Lerp}(1, \text{MG}, \omega) \quad (5)$$

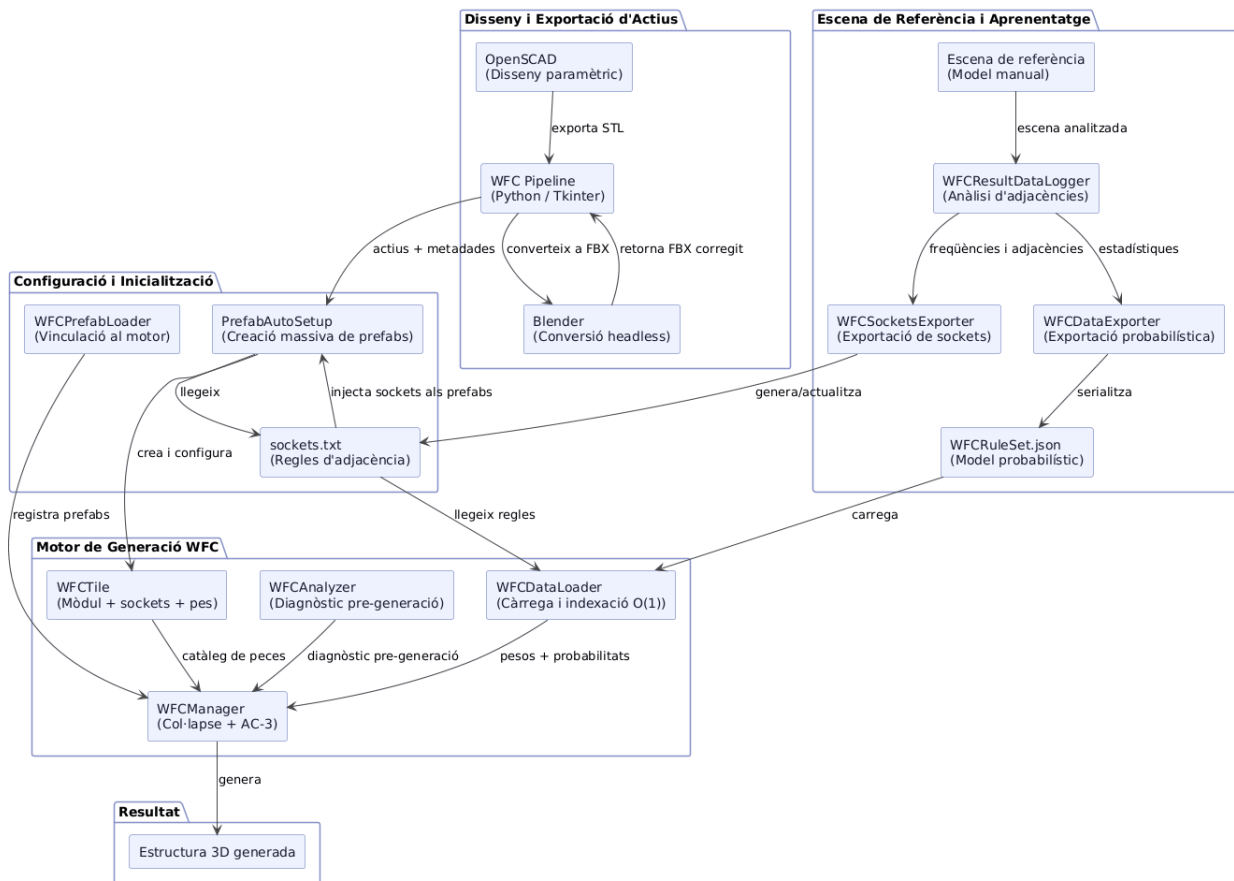


Fig. 4: Diagrama general de l'arquitectura...

Amb $\omega = 0$ el sistema opera únicament amb pesos manuals, sense cap influència del model après; amb $\omega = 1$ la generació replica al màxim l'estil i la distribució del model de referència.

7.5 Control manual de pesos i parametrització

Un cop generat el `WFCRuleSet.json`, l'usuari pot modificar manualment el pes de qualsevol peça des de l'Inspector de Unity sense alterar la lògica interna. Aquesta capacitat permet orientar la generació cap a estils concrets incrementant o reduint la presència de determinats mòduls de forma directa i previsible, tal com es valida experimentalment a l'apartat de resultats (vegeu la Figura 9).

A més del pes de les peces, el sistema exposa un conjunt de paràmetres configurables des de l'Inspector que permeten ajustar el comportament de l'algorisme sense modificar el codi font:

- **Seed:** valor inicial del generador pseudoaleatori. Una mateixa seed amb la mateixa configuració de peces produeix sempre exactament el mateix resultat, garantint la reproduïbilitat dels experiments.
- **Mida de la graella:** dimensions $X \times Y \times Z$ de l'espai volumètric de generació, que determinen directament el cost computacional del procés.
- **influènciaAprenentatge (ω):** lliscador en el rang $[0, 1]$ que regula el pes del model probabilístic après

sobre la selecció de peces. Amb $\omega = 0$ el sistema opera amb pesos manuals; amb $\omega = 1$ replica al màxim l'estil del model de referència.

- **maxIteraciones i maxPasosPropagacion:** límits de seguretat que impedeixen bucles infinits en cas de contradiccions, activant el mecanisme de *backtracking* de forma controlada.
- **Profunditat de backtracking:** nombre màxim de *snapshots* que el sistema pot emmagatzemar i de retrocessos permesos abans d'executar un reinici complet de la generació.

Per fer el sistema accessible i escalable, s'han desenvolupat dues eines integrades directament a l'editor de Unity que automatitzen el flux de treball entre el disseny geomètric i la generació procedimental. El component `PrefabAutoSetup` processa el fitxer `sockets.txt` i crea de forma massiva tots els *prefabs* del catàleg, injectant-hi els identificadors de connector ja convertits a enters mitjançant la funció de *hash* descrita anteriorment. El component `WFCPrefabLoader` escaneja la carpeta d'actius del projecte i vincula automàticament tots els *prefabs* disponibles al motor de col·lapse amb un sol clic.

8 RESULTATS

La validació d'un sistema de generació procedimental basat en el Wave Function Collapse (WFC) requereix una anàlisi que cobreixi tant la viabilitat tècnica com la capacitat expressiva de l'algorisme. A més de la correcció geomètrica,

cal assegurar que el sistema manté un rendiment adequat i que permet un cert control artístic sobre el resultat.

Aquesta secció es divideix en dos blocs principals. El primer avalua el comportament del sistema des d'un punt de vista tècnic, tot mesurant el cost computacional i l'estabilitat de l'algorisme a mesura que augmenta la complexitat de les malles. El segon analitza el mòdul d'aprenentatge probabilístic integrat i la seva capacitat per reproduir les característiques estadístiques dels models de referència, incloent-hi proves de modificació directa de pesos per valorar el grau de control sobre el resultat final.

8.1 Anàlisi de rendiment computacional i escalabilitat

Per quantificar el cost computacional del sistema, s'ha dissenyat un test de rendiment consistent en l'execució de l'algorisme WFC 20 vegades consecutives per a diferents mides de malla tridimensional, des d'un volum de $5 \times 5 \times 5$ cel·les fins a $15 \times 15 \times 15$. Per a cada mida s'han registrat els temps mitjans d'execució i la desviació estàndard. Les proves s'han realitzat en un entorn Windows amb un processador Intel Core i5 de 10a generació i 16 GB de memòria RAM DDR4, configuració de referència de gamma mitjana.

A mesura que augmenta la mida de la malla, el temps d'execució mostra un creixement clarament no lineal. Tal com es pot observar a la Figura 5, en les primeres dimensions el sistema manté temps de processament estables, però a partir de valors intermedis el cost augmenta de manera significativa. Aquest comportament s'explica pel creixement cúbic del volum de dades, ja que un increment lineal de les dimensions implica un augment exponencial del nombre de cel·les a processar.

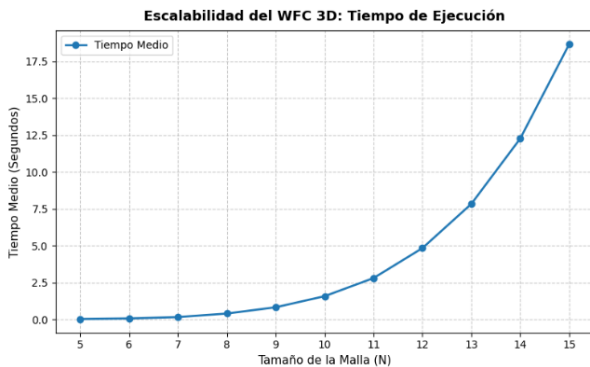


Fig. 5: Gràfic d'escalabilitat del temps d'execució de l'algorisme WFC per a malles tridimensionals de $5 \times 5 \times 5$ a $15 \times 15 \times 15$ cel·les, mostrant el creixement no lineal del cost computacional.

S'ha observat un punt crític a partir de malles de $23 \times 23 \times 23$, on Unity deixa de respondre durant les proves d'estrès i el sistema acaba congelant-se completament, indicant que aquest lílindar supera la capacitat de processament en temps real de l'entorn utilitzat. Les proves de $15 \times 15 \times 15$ s'han utilitzat com a límit pràctic per a les proves d'estrès, ja que en aquesta mida el sistema manté temps d'execució considerables però viables dins d'un entorn experimental. Per a casos on la velocitat no sigui un requisit, com ara la generació prèvia de terrenys o objectes destinats a ser exportats, el

sistema permet treballar amb malles considerablement més grans sense aquesta restricció.

L'estabilitat del sistema s'ha analitzat mitjançant la desviació estàndard dels temps d'execució en cada configuració. Com es pot observar a la Figura 6, la desviació augmenta progressivament amb la mida de la malla, però es manté sempre en valors baixos. El valor màxim registrat és de 0,27 s per a les malles de $N = 12$, mentre que en la resta de configuracions la variació es manté per sota d'aquest lílindar i tendeix a estabilitzar-se en valors inferiors a 0,20 s en el volum de $N = 15$.

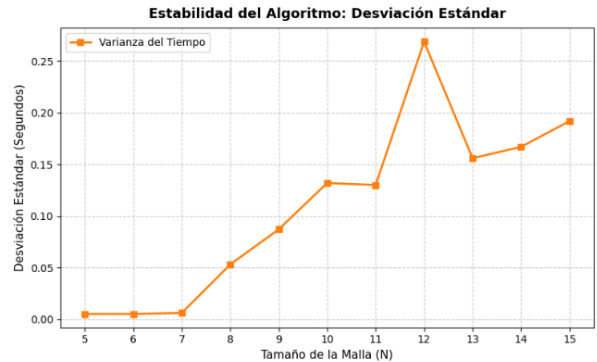


Fig. 6: Gràfic d'estabilitat temporal: desviació estàndard dels temps d'execució per a cada mida de malla, evidenciant la robustesa del sistema en condicions de regles ben definides.

Aquest comportament indica que el sistema és robust quan les regles estan ben definides. En escenaris amb restriccions més estrictes o amb una escassetat de peces més marcada, els processos de retrocés (*backtracking*) poden incrementar la variabilitat dels temps de manera significativa; no obstant això, en entorns ben definits i amb connexions coherents entre peces, com és el cas d'aquest experiment, el sistema es comporta de manera altament estable i previsible.

8.2 Anàlisi de l'aprenentatge probabilístic i fidelitat distributiva

Per quantificar l'impacte del paràmetre d'aprenentatge probabilístic, s'ha dissenyat un experiment a partir d'una estructura de referència específicament seleccionada per fer clarament visibles totes les peces amb orientació vertical i la peça que representa l'absència de connexió. Un cop generada, aquesta configuració s'ha guardat com a *WFCRuleSet* i s'ha utilitzat com a fitxer JSON de referència per al procés d'aprenentatge probabilístic.

A partir d'aquest *RuleSet*, s'han realitzat 10 generacions independents per als valors del paràmetre d'aprenentatge $\omega \in \{0, 0.1, 0.25, 0.5, 1\}$. Aquest paràmetre, amb rang $[0, 1]$, determina el grau d'influència de la distribució de peces del *RuleSet* sobre la generació final: quan $\omega = 1$ el sistema tendeix a replicar la distribució original; quan $\omega = 0$ la generació es comporta de forma completament independent.

8.2.1 Anàlisi qualitativa

Les captures de la Figura 7 mostren com, per a valors baixos del paràmetre d'aprenentatge, la generació presenta una distribució més aleatòria de peces. A mesura que ω augmenta, s'incrementa progressivament la presència de canoades verticals dins de la malla, fins al punt que l'estructura final s'aproxima cada cop més al patró de referència. Aquest comportament evidencia la capacitat del sistema per incorporar la informació estadística continguda en el *Rule-Set*, modulant de manera controlada la influència del model d'origen sobre la generació final.

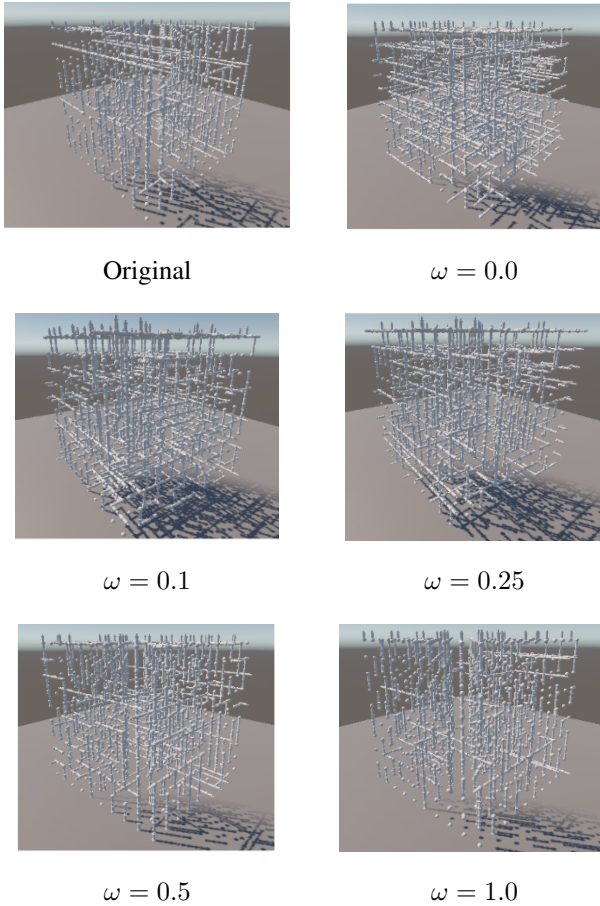


Fig. 7: Resultat visual de la generació procedimental

8.2.2 Anàlisi quantitativa de la distribució de peces

Aquest experiment s'ha centrat en la comparació del nombre i la freqüència d'aparició de les diferents peces en cadascuna de les 11 generacions realitzades, emprant la generació original com a referència (representada als gràfics amb una barra negra). Per tal d'evitar una saturació de figures, s'han inclòs únicament els casos extrems, $\omega = 0$ i $\omega = 1$, ja que són els que millor il·lustren el comportament del sistema.

Amb $\omega = 0$ les distribucions mostren una tendència clarament homogènia, ja que el sistema no incorpora la informació del model de referència i les probabilitats d'aparició tendeixen a ser més uniformes. En canvi, amb $\omega = 1$ la distribució de peces s'alinea de manera molt precisa amb la del model de referència, amb una freqüència molt similar a la de l'estructura original.

Aquesta tendència es confirma a la Figura 8, on es comparen els dos casos extrems del paràmetre d'aprenentatge. S'hi pot observar clarament com el cas $\omega = 0$ presenta una distribució més homogènia i independent, mentre que el cas $\omega = 1$ mostra una correspondència molt directa amb la distribució original.

8.2.3 Índex compost de fidelitat (CFI)

Per a una validació més robusta i menys dependent de la simple observació de freqüències, s'ha dissenyat un Índex Compost de Fidelitat (*Composite Fidelity Index*, CFI) que quantifica la similitud entre la distribució de referència i les execucions generades.

El CFI es construeix a partir de quatre mètriques normalitzades en $[0, 1]$:

- **Coefficient de Bhattacharyya:** mesura el grau de solapament entre les dues distribucions.
- **Correlació de Pearson:** avalua si la forma general de la distribució és semblant.
- **Correlació de Spearman:** es centra en l'ordre relatiu de les peces, independentment dels valors exactes.
- **TVS (*Total Variation Similarity*):** mesura les diferències absolutes entre distribucions, penalitzant les desviacions excessives.

El sistema distingeix dos nivells d'anàlisi: el CFI global, que avalua la similitud entre les distribucions completes de peces, i el CFI d'adjacències, que aplica la mateixa mètrica a les relacions entre peces i direccions. El resultat final es combina de la manera següent:

$$CFI = 0.6 \cdot CFI_{\text{global}} + 0.4 \cdot CFI_{\text{adjacències}} \quad (6)$$

Aquesta ponderació dona més pes a la coherència global de la distribució de peces, alhora que manté un pes significatiu sobre les restriccions locals, essencials per a la consistència estructural del sistema generatiu.

Els resultats obtinguts es resumeixen a la Taula 1. Amb $\omega = 0$ s'obté una fidelitat mitjana de 0,4865, reflectint una generació essencialment independent del model de referència. A mesura que augmenta el paràmetre, l'índex creix de manera consistent fins arribar a una mitjana de 0,8005 per a $\omega = 1$, amb pics individuals de fins a 0,8222. Aquesta evolució confirma de manera quantitativa que el factor d'aprenentatge controla de forma directa i estable la fidelitat de la generació respecte al model de referència.

TAULA 1: RESUM DE L'ÍNDEX DE FIDELITAT (CFI) RESPECTE A LA REFERÈNCIA ORIGINAL PER A DIFERENTS NIVELLS D'INFLUÈNCIA D'APRENENTATGE.

INFLUÈNCIA	MÍNIM	MÀXIM	MITJANA
0,0	0,4254	0,5358	0,4865
0,1	0,6184	0,6858	0,6557
0,25	0,6859	0,7461	0,7188
0,5	0,7100	0,7814	0,7464
1,0	0,7893	0,8222	0,8005

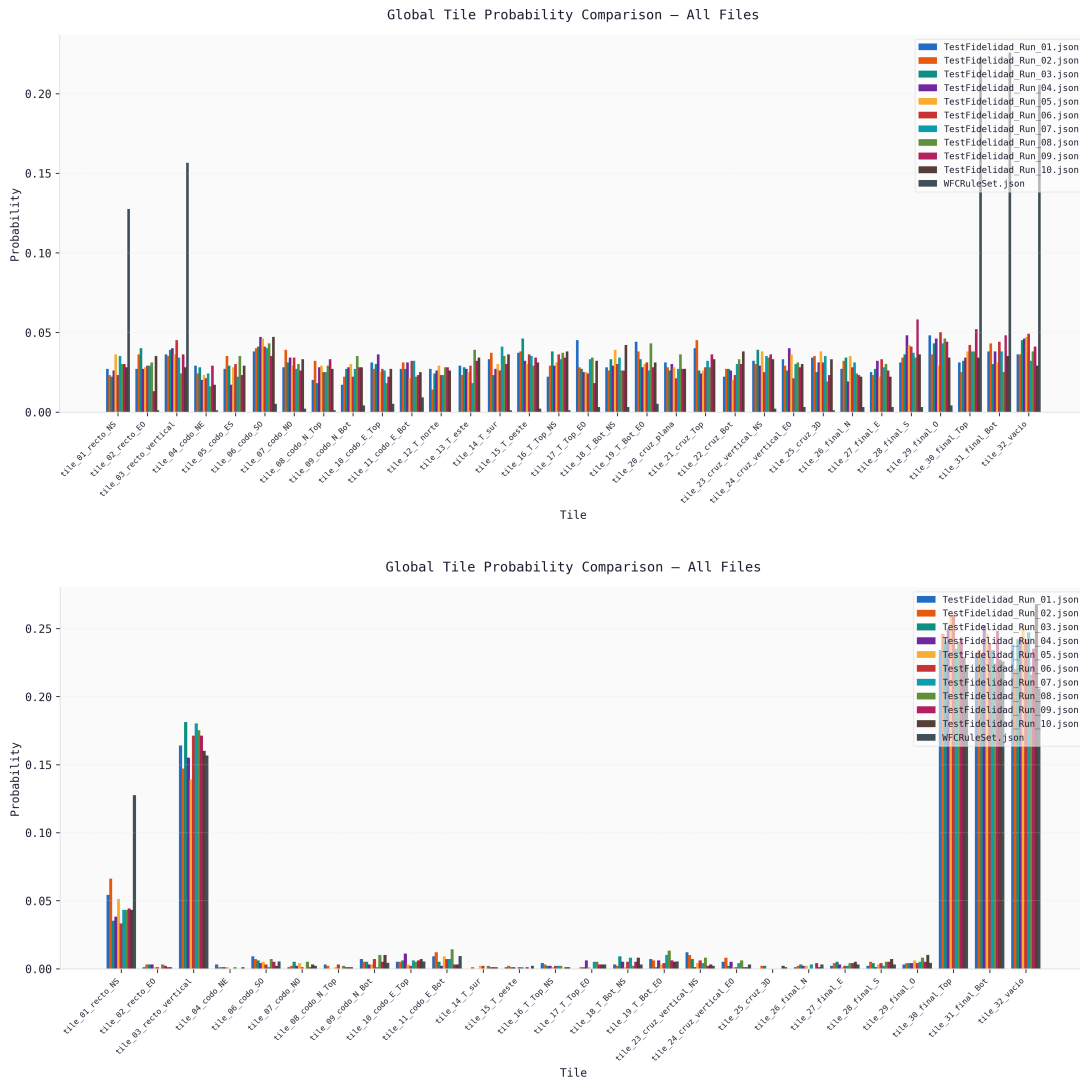


Fig. 8: Comparativa de probabilitat global de peces amb una influència d'aprenentatge de [0,1]

8.3 Validació del control probabilístic mitjançant la manipulació de pesos

Com a prova final, s'ha validat el sistema de modificació de pesos per comprovar la sensibilitat del model a canvis directes en la parametrització. Partint de la distribució base (WFCRuleSet_00), s'ha generat una nova versió (WFCRuleSet_01) incrementant la probabilitat de les últimes sis peces del catàleg (tile_27 a tile_32).

Tal com es pot observar a la Figura 9, aquest ajust produeix un augment clar en la freqüència d'aparició d'aquestes peces respecte al cas base, generant un pic destacat en la distribució. Aquest resultat reforça que el sistema respon de manera directa als canvis de pesos, permetent orientar la generació de forma controlada i previsible.

9 CONCLUSIONS

El projecte confirma la viabilitat de combinar el Wave Function Collapse amb l'aprenentatge probabilístic en entorns volumètrics, assolint la totalitat dels objectius definits. S'ha consolidat un prototip funcional a Unity amb propagació AC-3 en les sis direccions de l'espai i un sistema de *sockets* semàntics que garanteix la connectivitat estructural

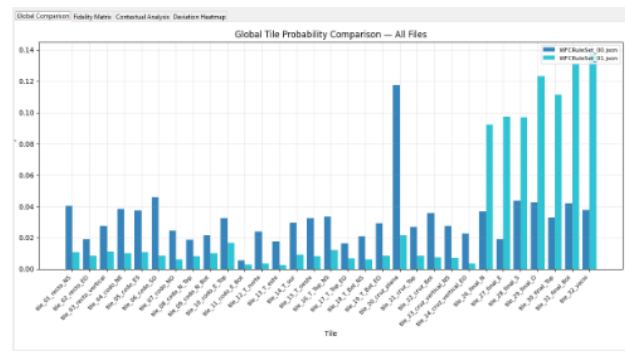


Fig. 9: Comparació de probabilitats globals entre la distribució base (WFCRuleSet_00) i la distribució modificada (WFCRuleSet_01), mostrant el pic provocat per l'augment manual de pesos de tile_27 a tile_32.

entre mòduls. El mòdul d'aprenentatge probabilístic permet orientar la generació cap a l'estil d'un model de referència sense reproduir-lo exactament, preservant la variabilitat. Les optimitzacions algorítmiques implementades han eliminat els bloquejos previs de l'editor, donant lloc a una eina funcional, escalable i parametritzable per a la generació procedimental tridimensional.

9.1 Limitacions detectades

La qualitat estadística del model après depèn de la diversitat de la mostra de referència, fet que pot comprometre la fidelitat de l'aprenentatge en catàlegs poc variats. El *hash* base-31, tot i ser molt eficaç, presenta una probabilitat de col·lisió no nul·la que podria generar conflictes en projectes amb un volum massiu d'identificadors. Finalment, el sistema no és capaç de generar malles superiors a $23 \times 23 \times 23$ unitats.

9.2 Línies de treball futur

Es proposen dues extensions principals. La primera és la generació dinàmica en *runtime* per blocs, que permetria col·lapsar únicament els sectors rellevants a mesura que l'usuari es desplaça, facilitant la síntesi d'entorns pràcticament infinits amb un consum de memòria optimitzat. La segona és la integració de Xarxes Neuronals de Grafs (GNN) [12], que permetrien analitzar patrons de forma més específica i generar estructures molt més similars al model de referència del que permet l'anàlisi probabilística actual.

AGRAÏMENTS

Vull agrair al meu tutor, Robert Benavente, la seva guia i acompanyament al llarg de tot el treball. També vull donar les gràcies als meus amics Eduard i Adrián, per haver-me acompanyat i recolzat durant tota la carrera i, molt especialment, durant aquests últims mesos de treball.

ÚS DE LA IA GENERATIVA

Durant l'elaboració d'aquest projecte s'han utilitzat eines d'IA generativa com a suport puntual en tasques de correcció lingüística, millora de la redacció i organització inicial de determinats apartats de la documentació. Aquestes eines s'han emprat exclusivament com a assistència en el procés d'escriptura, amb l'objectiu de millorar la claredat del text i l'estructura del document. El contingut tècnic, el desenvolupament de l'algorisme, les decisions de disseny i la definició dels objectius del projecte corresponen íntegrament al treball original de l'autor.

REFERÈNCIES

- [1] Level Up Game Dev Hub, "Generación de contenido procedural: una aproximación amigable," Blog de Level Up, 2022. [En línia]. Disponible a: <https://www.levelup-gamedevhub.com/blog> [Consultat el: 11 abr. 2026].
- [2] K. Perlin, "Perlin Noise," Wikipedia, gen. 2026. [En línia]. Disponible a: https://en.wikipedia.org/wiki/Perlin_noise [Consultat el: 11 abr. 2026].
- [3] Hello Games, "No Man's Sky," Pàgina oficial del joc, 2016. [En línia]. Disponible a: <https://www.nomanssky.com> [Consultat el: 11 abr. 2026].
- [4] M. Gumin, "WaveFunctionCollapse," Repositori de GitHub, 2016. [En línia]. Disponible a: <https://github.com/mxgmn/WaveFunctionCollapse> [Consultat el: 11 abr. 2026].
- [5] O. Stålberg, "Wave Function Collapse (Interactive Demo)," oskarstalberg.com. [En línia]. Disponible a: <http://oskarstalberg.com/game/wave/wave.html> [Consultat el: 11 abr. 2026].
- [6] S. Lague, "Coding Adventure: Wave Function Collapse," Canal de YouTube Sebastian Lague, 2019. [En línia]. Disponible a: <https://www.youtube.com/watch?v=2SuvO4Gi7uY> [Consultat el: 11 abr. 2026].
- [7] R. Heaton, "The Wavefunction Collapse Algorithm explained very clearly," robertheaton.com, Des. 2018. [En línia]. Disponible a: <https://robertheaton.com/2018/12/17/wavefunction-collapse-algorithm/> [Consultat el: 11 abr. 2026].
- [8] M. Kleineberg, "Infinite procedurally generated city with the Wave Function Collapse algorithm," marian42.de / GitHub, oct. 2018. [En línia]. Disponible a: <https://github.com/marian42/wavefunctioncollapse> [Consultat el: 11 abr. 2026].
- [9] B. the Brave, "Arc Consistency Explained," Boris the Brave Blog, 2021. [En línia]. Disponible a: <https://www.boristhebrave.com/2021/08/30/arc-consistency-explained/> [Consultat el: 11 abr. 2026].
- [10] Unity Technologies, "Manual de l'usuari i Scripting API," Documentació oficial d'Unity, 2026. [En línia]. Disponible a: <https://docs.unity3d.com/Manual/index.html> [Consultat el: 11 abr. 2026].
- [11] Kenney, "Assets," Llibreria de recursos 3D modulars, 2026. [En línia]. Disponible a: <https://kenney.nl/assets> [Consultat el: 11 abr. 2026].
- [12] B. Sanchez-Lengeling, E. Reif, A. Pearce, i A. B. Wiltschko, "A Gentle Introduction to Graph Neural Networks," Distill, 2021. [En línia]. Disponible a: <https://distill.pub/2021/gnn-intro/> [Consultat el: 17 maig 2026].

APÈNDIX

.1 Diagrama de Gantt

