

From Tiles to Jumps: Optimizing Wave Function Collapse Window Size for Playable Platformer Generation

Trevor Truesdell
Trinity University
San Antonio, Texas, USA
ttruesde@trinity.edu

Erik Rankin
Trinity University
San Antonio, Texas, USA
erankin@trinity.edu

Britton Horn
Trinity University
San Antonio, Texas, USA
bhorn@trinity.edu

Abstract—Wave Function Collapse (WFC) has emerged as a popular procedural content generation algorithm for tile-based game levels, yet little is known about how its structural parameters affect the playability of generated content. This paper investigates the relationship between WFC window size, the spatial extent of patterns captured during learning, and level playability in *Super Mario Bros.* We systematically evaluate nine window sizes ranging from 1×1 to 14×6 across 15 original source levels. Levels are assessed using A* agents, expressive range analysis, and human trials. A one-way ANOVA reveals that window size significantly affects agent completion percentage, with completion rates increasing from 32% at 2×2 to nearly 100% at 14×6 . Results show that mid-size windows (4×4 to 5×5) balance playability and diversity, while larger windows capturing jump heights (6×6) achieve near 100% completion. Human player trials ($n = 25$) confirm that large window levels are rated comparably to original levels on playability, enjoyment, coherence, and replayability, while smaller window levels score significantly lower. Additionally, agent completion rate is a strong predictor of human progress. This work provides the first systematic investigation of WFC structural parameters on functional playability and player experience, offering actionable guidance for PCG practitioners, and demonstrating window size as an intuitive designer control mechanism.

Index Terms—procedural content generation, wave function collapse, level design, constraint satisfaction

I. INTRODUCTION

Procedural Content Generation (PCG) enables automatic creation of game levels, environments, and assets through algorithms rather than manual authoring. By reducing development costs and supporting replayability, PCG allows designers to focus on higher-level creative decisions while maintaining variety and scalability [1].

The *Wave Function Collapse* (WFC) algorithm has gained attention for generating visually coherent content from small example inputs [2]. Inspired by quantum mechanics, WFC iteratively constrains tile placement according to learned adjacency rules, preserving stylistic and structural consistency [3], [4]. This balance between creativity and constraint has made WFC accessible for a wide range of applications. Despite this

progress, controlling WFC’s output remains challenging when balancing diversity with playability and design intent.

Tile-based platformers such as *Super Mario Bros.* provide ideal testbeds for evaluating PCG algorithms [5]. Each level encodes implicit design rules, such as platform spacing, enemy placement, and reward timing, that directly influence playability. Previous work has applied WFC to replicate Mario’s stylistic features or introduce design-level constraints [3], [6], [7], but most efforts focus on aesthetic similarity rather than functional playability, leaving questions about how specific WFC parameters affect player experience.

As PCG systems mature, designers require tools offering both creative flexibility and predictable results. Research has explored embedding constraints at the tile or rule level to steer generation [6], [8]; however, little attention addresses how WFC’s *structural parameters*, such as window size, influence coherence and playability. These algorithmic properties represent an underexplored but powerful form of designer control.

WFC *window size* defines the spatial extent of patterns extracted during learning, determining contextual information captured per tile. Smaller windows produce varied outputs but risk incoherence; larger windows preserve structure but reduce diversity. Although previous work implicitly varied window size when scaling to large or hierarchical maps [9]–[11], no study has systematically examined how this parameter affects level playability in concrete gameplay contexts.

This paper investigates the relationship between WFC window size and level playability in *Super Mario Bros.* We hypothesize that larger windows yield more playable levels by encoding broader spatial dependencies, but that this benefit plateaus once the window captures the physical constraints (jump height/length) governing platform reachability.

This paper makes the following contributions:

- The first systematic investigation of WFC window size effects on generated *Super Mario Bros.* level playability.
- A reproducible evaluation pipeline combining simulation-based playability testing with structural and aesthetic similarity measures.
- Evidence that WFC window size functions as an intuitive, physics-grounded control parameter for balancing

playability and generative diversity.

- Player trials ($n = 25$) confirming large window levels match human-designed originals in playability, enjoyment, coherence, and replayability, and that agent completion strongly predicts human completion ($r_s = 0.582$).

II. RELATED WORK

In this section, we outline prior work in WFC theory, designer control mechanisms, hybrid optimization approaches, and PCG evaluation methodologies.

A. WFC Foundations and Extensions

While similar to Merrell’s Model Synthesis [12], WFC was popularized by Gumin’s implementation [2], which generates locally-coherent content by extracting and satisfying adjacency constraints between tile patterns. Karth and Smith [13] provided theoretical grounding, positioning WFC as constraint satisfaction through arc consistency and entropy-driven search. Critically, they identified pattern size as fundamental to WFC’s behavior, noting trade-offs between diversity and coherence, though without empirical investigation of effects on playability. Lux documents WFC’s evolution into a widely-adopted tool across game development and research [4].

Researchers have extended WFC with design-level constraints [7], automatic rule inference [6], practical tooling for production pipelines [14], and mixed-initiative interfaces [8]. These works emphasize making generation parameters accessible to designers while maintaining local pattern coherence.

Cooper’s Sturgeon [15] represents a more general approach, combining example-learned patterns with explicit designer-specified constraints. Where Sturgeon gives designers direct control by specifying exact tile counts, reachability guarantees, or structural properties, WFC’s window size operates as a single implicit parameter governing pattern granularity. We offer a complementary perspective on designer control through structural tuning rather than explicit constraint specification.

B. Scalability and Hybrid Approaches

Hierarchical and nested WFC variants address scalability by operating at multiple pattern scales simultaneously [9]–[11], [16]. While these approaches implicitly vary pattern size across generation phases, they do not systematically examine how this affects functional quality.

Hybrid approaches combine WFC with optimization methods to achieve global objectives. [17] embed WFC within genetic algorithms to optimize novelty and complexity while maintaining pattern validity. [3] use reinforcement learning to guide tile selection toward more playable levels. [18] similarly demonstrate that evolutionary methods can balance local coherence with global goals, suggesting additional guidance may be necessary for specific objectives.

C. Evaluation in the Mario AI Framework

For evaluation, the Mario AI Framework established A*-based agents as effective proxies for measuring level playability [19], [20]. [21] used neural networks to predict human rat-

ings of level quality, while [22] applied CNNs to predict difficulty, enjoyment, and aesthetics of Mario levels. [23] proposed theory-based quantitative metrics bridging qualitative design analysis and empirical evaluation. [24]–[26] demonstrated that AI agents can model player affective states and play styles, establishing that agent-based metrics can approximate aspects of human experience.

Structural analysis methods complement agent-based testing. [27] introduced expressive range analysis for characterizing generator diversity. [28] benchmarked procedural Mario generators using linearity, leniency, density, and similarity metrics, which we adopt in our evaluation.

III. WAVE FUNCTION COLLAPSE ALGORITHM

The Wave Function Collapse (WFC) algorithm is a constraint-based procedural generation method that produces locally-coherent, tile-based content from example inputs [2]. Despite its name, WFC is fundamentally a constraint satisfaction algorithm that iteratively resolves tile placements by maintaining and propagating adjacency constraints [13].

A. Algorithm Overview

WFC operates in two phases: an extraction phase that determines patterns and constraints from example input, and a generation phase that produces new content by satisfying these extracted constraints.

1) *Pattern Extraction Phase*: Given an input example (either a bitmap image or a set of tiles with explicit adjacency rules), WFC first extracts all possible $M \times N$ patterns, where M is width, N is height, and together define the *window size*. For each position, the algorithm captures a rectangular window of tiles and records: the pattern, its frequency, and which other patterns appear adjacent to it in each cardinal direction. These adjacency relationships form the constraints that govern generation.

The window size parameter is critical to WFC’s behavior. Smaller windows capture local tile relationships but may miss broader structural patterns, potentially leading to incoherent outputs. Larger windows preserve more context and structure but reduce pattern variety. This trade-off between local diversity and global coherence is central to our investigation.

2) *Generation Phase*: The generation phase initializes an output grid where each cell is in a *superposition* state, potentially containing any tile from the input. The algorithm then iteratively: (1) selects the cell with lowest *entropy* (fewest tile options), (2) collapses it to a single tile weighted by frequency, (3) propagates constraints to neighboring cells to maintain adjacency consistency, and (4) repeats until all cells are determined or a contradiction occurs.

B. Model Variants and Configuration

WFC has two primary variants: the *overlapping model* for bitmap textures with smooth transitions, and the *simple tiled model* for discrete tile-based content with explicit adjacency rules. We employ the simple tiled model as game levels consist of discrete tiles with strict adjacency constraints.

Standard WFC implementations support *periodicity* (wrapping edges) for seamless textures and *rotations/reflections* to augment pattern variety. However, for platformers, these features break critical design constraints. For example, periodicity allows the ground to appear at any y position, while rotations and reflections can flip staircases or place ground tiles on ceilings. We disable both to preserve platformer logic.

IV. METHODS

All WFC adjacency rules are derived from the 15 original *Super Mario Bros.* levels provided by the Mario AI Framework¹, which converts text-based level representations into playable game levels. Generated levels match the dimensions of their source level, which vary in width but are uniformly 16 tiles tall. Original levels fall into three categories whose structural differences produce markedly different WFC outputs:

- Above ground (see Figure 1a): Levels 1, 4, 5, 7, 9, 11, 12, 14, and 15.
- Underground: Levels 2 and 8.
- Mushroom (see Figure 1e): Levels 3, 6, 10, 13.

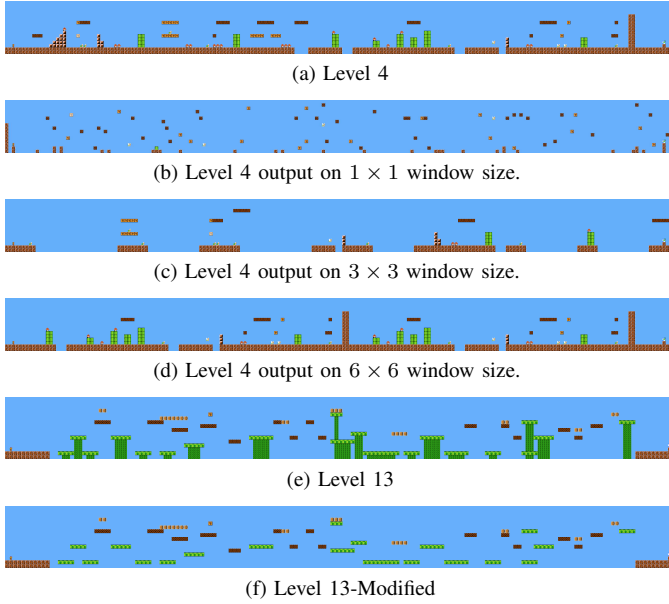


Fig. 1: Level visualizations for Level 4 (a-d), 13 (e), and 13-Modified (f). (b) is an example of near-random output with window size 1×1 and input from Level 4.

A. Window Sizes

We analyze 9 distinct window sizes in our generation and tests in the form $M \times N$ to test the hypothesized increase in level similarity and playability over increased window sizes. Sizes include: 1×1 , 2×2 , 3×3 , 4×4 , 5×5 , 6×6 , 14×2 , 14×6 , and 1×16 . Square windows form the core progression as they treat both axes symmetrically, providing an unbiased single-parameter sweep.

In order to test the impact of physics-based window sizes on playability of levels, we make use of window sizes 14×6 and 14×2 . 14×6 represents empirically measured maximum possible jump height and length of Mario in the Mario AI Framework. Since a 14×6 window size decreases the amount of unique patterns to the point of consistently producing the original source level, we incorporate 14×2 to represent the jump length of Mario while reducing vertical constraints and 6×6 to represent jump height while reducing horizontal constraints, thus increasing the amount of unique patterns.

Following [29], who showed that vertical column patterns effectively represent linear levels, we include a 1×16 window, where 16 equals the level height of all original levels in the framework.

B. Original WFC Modifications

To increase the functionality of WFC generations on text-based Mario Levels, we make several modifications to the original bitmap-based algorithm.

1) *Edge Constraints*: To restrict periodicity under valid adjacencies, we introduce edge constraints. All boundary tiles within original levels are constrained to their respective boundary in the output. For example, if a tile is only found on the left edge in the original level, it can only appear on the left edge in the output. Tiles appearing on both the boundary and interior may appear in both locations. Interior-only tiles are similarly prohibited from edge positions. Without these constraints, edge tiles placed in the interior would cause generation failures due to missing adjacencies

2) *“All” Levels*: To evaluate WFC over multiple inputs, “All” levels source tiles and adjacencies from all 15 original levels, allowing tiles from any input level to appear in the output, as long as they follow adjacency rules. When producing “All” levels, Mario and Finish tiles, each carrying singular adjacencies, commonly caused contradictions when drawn from separate levels. Additionally, ambiguous enemy placement further over-constrained generation. To address this, we apply a ‘similar tiles’ function, removing all enemy, Mario, and finish characters before pattern extraction, such that tiles identical under this reduction share adjacencies. During generation, any tile from the similar set is selected randomly.

3) *Mario and Finish Tiles*: Often during generation, Mario and Finish tiles appear in widely varying x locations, sometimes causing the start to precede the finish or contain multiple start/finish locations. In order to ensure that a single Mario and Finish tile is placed at the respective ends of the level, we constrain their generation in the output. For “All” levels, this is done by forcing the Mario and Finish tile to their respective X and Y offset from their place in the original. For all other levels we force the Mario and Finish tile to be in the exact same location in the input as the output.

4) *Decoration Removal*: To determine whether purely aesthetic tiles affect WFC output, we modified Level 13—a mushroom room level that commonly produces near-identical outputs—by removing its decorative mushroom platform supports. Unlike decorative tiles in other source levels, the mushroom

¹<https://github.com/amidos2006/Mario-AI-Framework>

supports fall directly on the player’s traversal path, making their removal the minimal structural intervention. This allows us to assess how non-functional level aspects influence WFC’s constraint structure and output diversity (see Figures 1e and 1f).

C. Analysis

To analyze a wide range of output, we generate 1000 levels for each of the nine window sizes and 17 source level combinations (15 original, Level 13-Modified, and “All”), producing a total of 153,000 levels. We then analyze level playability with A* agents, structure with Expressive Range Analysis, and player experience with human participant evaluation.

1) *Playability*: We make use of the MFF A* agent suite [30], which provides multiple A* variants, to test the playability of WFC-produced levels. We ran all variants in the Mario AI Framework and report the highest completion percentage achieved by any agent for each generated level. Playability is defined as the horizontal percent of level completion achieved by the best-performing agent. Importantly, the suite completes all original levels. To test whether window size significantly affects level completion percentage, we conducted a one-way Analysis of Variance (ANOVA) with window size as the independent variable (9 levels) and agent completion percentage as the dependent variable across generated levels. Post-hoc analysis with Tukey’s HSD test was performed to identify specific window size comparisons with significant differences in completion rates.

2) *Expressive Range Analysis*: To assess the similarity of produced levels with the original, we calculate normalized edit and compression distances. For edit distance, we convert each text-based level into a single continuous string as done in [31] and calculate the levenshtein distance. Compression distance represents the distance between size of files when they are compressed. Our implementation utilizes the standard gzip algorithm for compression as in [32]. It should be noted that the representation of levels can significantly alter the compression distance metric [33]. For “All” levels, distances are computed relative to Level 1, as no single source level exists for comparison.

We analyze the structure of generated levels using linearity, density, and leniency as described by [28]. Linearity is measured as the R^2 value of platform positions where each platform is defined as a continuous line of standable tiles with no solid tiles directly above. A platform ends when it reaches air or a standable tile one unit up (e.g. a pipe or stair). We use the center of each platform as a single data point for its position. Density measures the number of standable platforms at each vertical y position. We normalize density to compare relative structural characteristics across levels. We calculate Leniency by assigning a value to each obstacle occurrence, with negative values for gaps, average gap length, and enemy count, and positive values for platforms and ground. We then normalize the raw values across levels to establish relative difficulty among outputs.

3) *Human Trials*: Twenty-five university students (ages 18–24) were recruited to assess player experience of WFC-generated content. Participants self-reported a median platformer experience of 3 (1 = never played, 5 = play frequently). Each participant played 10 study levels drawn from a pool of 45 levels spanning five conditions: 1×1 anomaly, small window (2×2 and 3×3), medium window (4×4 and 5×5), large window (6×6 and 14×6), and original levels. Levels were assigned via a randomized shuffle, balanced across all 25 participants to ensure even coverage of all 45 levels. Within each WFC condition, levels span the full range of agent completion rates observed for that window size. Nine original *Super Mario Bros.* source levels, each achieving 100% A* completion, provide a human-designed baseline. Participants were informed that the study investigated procedural content generation, but were not told which individual levels were procedurally generated or the method used.

Prior to the main session, each participant completed one warm-up level from the original source set to acclimate to the Mario AI Framework’s controls, which differ slightly from the original NES release. Participants then played each of 10 assigned levels in a unique randomized order. Participants were given up to 5 attempts per level.

Horizontal completion percentage was recorded for each level. Following each level, participants were asked five questions via structured interview using a 7-point Likert scale (1 = strongly disagree, 7 = strongly agree): playability (“I was able to make progress through this level.”), enjoyment (“I enjoyed playing this level.”), coherence (“This level felt intentionally designed (not random).”), challenge (“This level was appropriately challenging (not too easy or too hard).”), and replayability (“I would want to play more levels like this one.”). These measures capture both structural quality (playability, coherence) and broader player experience (enjoyment, challenge, replayability).

V. RESULTS

Window size significantly affected agent completion percentage ($F(8, 152991) = 11092.12, p < 0.001$). Table I presents descriptive statistics for each window size group. Tukey’s HSD post-hoc tests for multiple comparisons found that mean completion percentages were significantly different ($p < .001$) between all pairwise window size comparisons except 5×5 vs. 1×16 and 6×6 vs. 14×6 .

Completion rates generally increased with window size, from 32% (2×2) to 99.9% (6×6 , 14×6). The 1×1 window is an exception at 75.6% completion despite capturing minimal spatial context, outperforming 2×2 , 3×3 , and 4×4 . The vertically-oriented 1×16 window achieved 87.9% completion, closely matching the 5×5 window, while the horizontally-oriented 14×2 window reached 96% completion.

A. Structural and Similarity Metrics Across Window Sizes

Figure 2 shows mean completion percentage, edit distance, compression distance, density, leniency, and linearity across all source levels and window sizes.

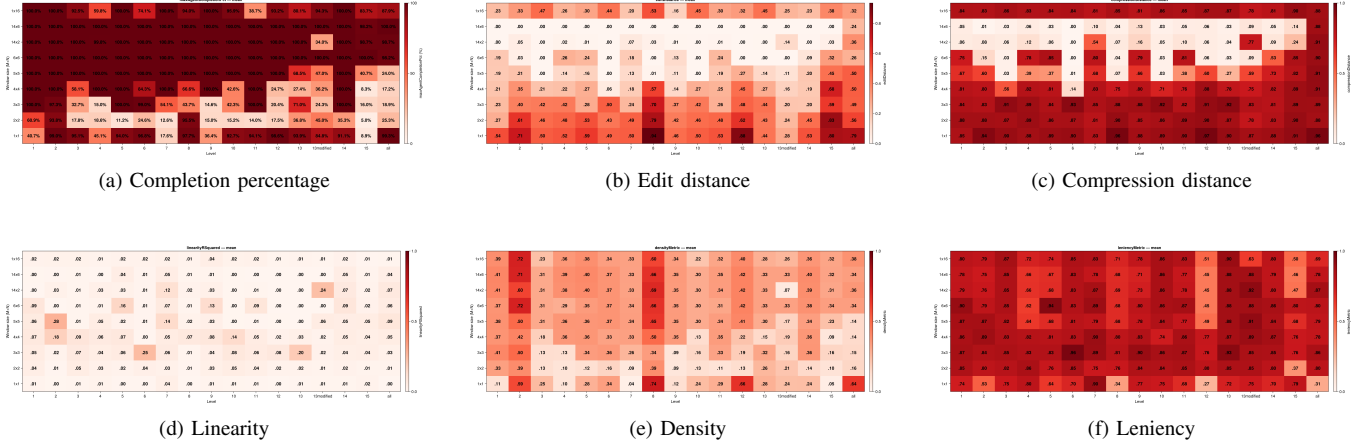


Fig. 2: Heatmaps for mean agent completion percentage, edit distance, compression distance, linearity, density, and leniency (left-right, top-bottom) over all levels and window sizes. Window size on Y-Axis with source level on X-Axis in order: 1×1 , 2×2 , 3×3 , 4×4 , 5×5 , 6×6 , 14×2 , 14×6 , 1×16 . Darker hues indicate higher values.

TABLE I: Agent completion percentage averages and variances by window size. Each row includes all levels generated from every source, including “All.”

Window Size	Average	Variance
1×1	0.756	0.141
2×2	0.320	0.110
3×3	0.558	0.161
4×4	0.686	0.130
5×5	0.871	0.074
6×6	0.999	<0.001
1×16	0.879	0.077
14×2	0.960	0.024
14×6	0.999	<0.001

Agent completion percentage (Figure 2a) shows strong dependence on both window size and source level. Larger windows generally produce higher completion for most source levels. Mushroom levels show reduced completion rates on small and medium window sizes relative to other single-source levels. The multi-source “All” condition consistently achieves lower completion than single-source levels across all window sizes, and only achieves 100% completion at 14×6 .

Edit distance (Figure 2b) decreases as window size increases for most source levels. The 14×6 window produces near-zero edit distances for single-source output. The “All” levels show higher edit distances than single-source levels across all window sizes (computed relative to Level 1). Compression distance (Figure 2c) similarly decreases as window size increases, approaching zero for 14×6 windows across single-source levels.

Density (Figure 2e) varies by source level, with Levels 2 and 8 showing higher density values across all window sizes. Density generally increases with window size, yet the 1×1 window produces higher density values for levels 2, 8, 12, and “All” compared to other levels. Linearity (Figure 2d) shows R^2 values predominantly below 0.25 across all source-window

combinations, with few exceptions. Leniency (Figure 2f) varies by source level with no consistent pattern.

B. Relationship Between Metrics and Playability

Expressive Range Analysis (Figure 3) reveals systematic relationships between structural metrics and agent completion. Completion is highest where density is low (5.5% at 0.004) and increases to 98% at density 0.4, above which levels consistently achieve near-perfect completion. Completion increases slightly with linearity from 70.6% at $R^2 = 0.1$ to 98% at $R^2 = 0.45$, with most levels falling below $R^2 = 0.25$. Completion decreases as compression distance increases (93.7–100% below 0.4, falling to 57.7% at 0.9) and as edit distance increases (99.9% at 0.004 to 34–70% between .36 and .75). Completion generally increases with leniency but non-monotonically (Figure 3e).

C. Human Trials

Table II summarizes median and max progress and subjective ratings by condition. Human progress differed significantly across conditions (Kruskal-Wallis $H(4) = 112.82, p < .001$). The 1×1 anomaly condition produced the lowest median progress (7%) and max progress (83%), followed by small window (17%, 97%), medium window (37%, 100%), large window (47%, 100%), and original (47%, 100%). No participant fully completed any 1×1 anomaly or small window level. Full completion occurred in 24%, 12%, and 4% of trials for medium window, large window, and original conditions respectively. Pairwise comparisons (Mann-Whitney U , Bonferroni corrected) revealed significant differences between the 1×1 anomaly condition and all others, and between small window and medium window, large window, and original conditions (all $p < .001$). Medium window, large window, and original conditions did not differ significantly from one another. Agent completion rate was a strong positive predictor of human progress ($r_s = 0.582, p < .001$).

TABLE II: Median and max human trial outcomes by condition ($n = 50$ level-play trials per condition across 25 participants). Progress columns show median and maximum traversal percentage. Likert scales are 1–7 (medians).

Condition	Progress (%)		Play- ability	Enjoy- ment	Coher- ence	Chall- enge	Replay- ability
	Median	Max					
1×1 Anomaly	7	83	1.0	2.0	2.0	2.0	2.0
Small Window	17	97	1.5	3.0	3.0	3.0	2.0
Medium Window	37	100	4.0	5.0	4.0	4.0	3.0
Large Window	47	100	5.0	5.0	5.0	4.0	5.0
Original	47	100	5.0	5.0	5.0	5.0	6.0

Kruskal-Wallis tests found significant condition effects on all five Likert dimensions: playability ($H = 82.18$), enjoyment ($H = 76.96$), coherence ($H = 42.44$), challenge ($H = 48.17$), and replayability ($H = 65.78$), all $p < .001$. Linear mixed-effects models with random intercepts per participant and original levels as the reference condition found that large window levels did not significantly differ from original levels on playability, enjoyment, coherence, or replayability. Large window levels were rated significantly lower than original levels on challenge only ($\beta = -0.753, p = .025$). All other WFC conditions rated significantly lower than original levels on all five dimensions (all $p \leq .004$). Agent completion rate significantly predicted playability ($\beta = 3.24$), enjoyment ($\beta = 2.72$), challenge ($\beta = 1.44$), and replayability ($\beta = 2.54$), all $p < .001$. Coherence was not significantly predicted by agent completion ($\beta = 0.47, p = .312$). Presentation order had a small positive effect on playability ratings ($r_s = +0.186, p = .003$) but did not significantly affect other outcomes.

VI. DISCUSSION

WFC window size significantly affects both computational playability and human player experience, with effects that interact with source-level characteristics and physics constraints.

A. Window Size and Playability

The progression of agent completion rates across window sizes reveals a clear but non-monotonic pattern. The 1×1 window presents an intriguing anomaly, achieving 75.6% completion despite capturing only individual tile adjacencies, considerably outperforming the 2×2 (32%), 3×3 (56%), and 4×4 (69%) windows that encode more spatial context. As window size increases to 5×5 (87%) and 6×6 (99.9%), playability improves substantially ($p < 0.001$), plateauing at larger physics-informed sizes with 14×2 achieving 96% and 14×6 reaching 99.9%. The density heatmap (Figure 2e) reveals that 1×1 windows produce unusually dense levels for certain source levels (notably 2, 8, 12, and “All”), and this density compensates for structural incoherence (see Figure 1b for an example output). A* agents navigate these successfully not due to good design structure, but because platform abundance provides sufficient forward progress, even if the resulting levels might frustrate human players. The 2×2 window emerges as the worst performer, capturing enough

context to break out of 1×1 ’s propensity for highly dense levels while lacking structural coherence for deliberate design.

B. The Physics Constraint Plateau

Our hypothesis predicted playability would plateau once window sizes captured Mario’s movement constraints. Agent completion results strongly confirm this with agent completion percentage reaching near-perfect levels at 6×6 (99.9%), suggesting that capturing jump height (6 tiles) is sufficient. The comparison between 14×2 (96%) and 6×6 (99.9%) illuminates this further. While 14×2 captures full horizontal jump span, its 2-tile vertical constraint limits encoding of jump arcs and vertical platform arrangements, indicating vertical relationships are more fundamental to platformer playability. This makes intuitive sense as gaps rarely span maximum jump distance (requiring perfect timing), but vertical relationships are explicitly constrained by fixed jump height.

The 1×16 window (87.9% completion) further validates the importance of vertical structure. As a pure vertical n-gram approach capturing complete 16-tile columns, it encodes ground-to-platform distances, enemy heights, and vertical obstacles while allowing horizontal freedom. [29] demonstrated vertical n-grams effectively represent linear levels; our results validate this within WFC. That 14×2 (96%) outperforms 1×16 (87.9%) suggests horizontal context adds meaningful value, yet complete vertical information can largely compensate.

However, near-zero edit distances across all level inputs for 14×6 windows reveal that over-constrained generation often reproduces source levels nearly exactly. While effectively guaranteeing playability, this severely limits generative diversity, representing a fundamental trade-off when selecting window size. Including all levels as source noticeably improved the edit distance metric for 14×6 , yet this condition shows markedly lower completion rates across window sizes, indicating greater diversity at the cost of playability. This suggests that while large windows can over-constrain generation, providing a more diverse set of source patterns can mitigate this by enabling pattern blending across levels with otherwise incompatible adjacency rules. This highlights the importance of source selection alongside window size to balance playability and diversity.

C. Similarity, Structure, and Playability

The inverse relationship between completion percentage and both edit and compression distance confirms that more faithful reproduction generally yields more playable outputs. Density proves a strong predictor of completion, particularly in the 0.35-1.0 range where Q1-Q3 values remain tightly bounded near 100%. This aligns with established design principles: moderate platform density provides challenge through meaningful gaps while maintaining clear forward paths. Linearity shows weaker but positive relationships with completion, expected for Mario-style platformers featuring relatively linear progression despite vertical variation.

Removing purely decorative mushroom platform supports from Level 13 produced unexpected results, with edit distance

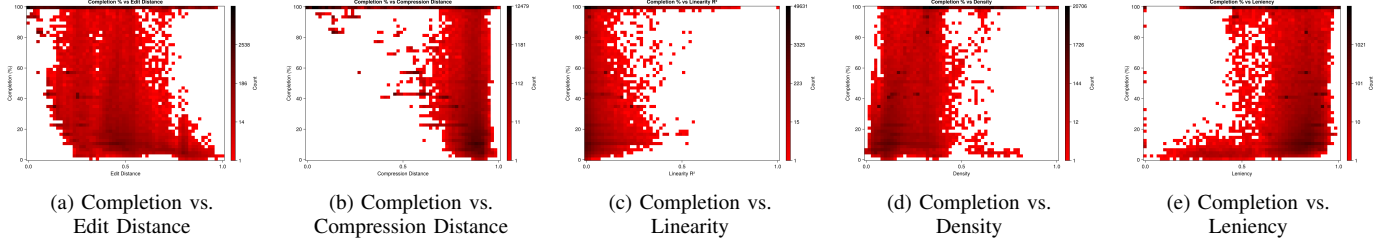


Fig. 3: Heatmaps of each metric against agent completion percentage across all generated levels. Completion on Y-Axis, with the X-axis containing edit distance, compression distance, linearity, density, and leniency (left-right).

decreasing at nearly all window sizes (except 14×2). Compression distance was unchanged at smaller window sizes but increased slightly above 5×5 , including at 14×6 , the window that motivated the modification. Agent completion improved at 2×2 , 4×4 , and 1×16 but decreased, sometimes drastically so, at most other window sizes. These results suggest that decorative tiles contribute both tile-type variety and adjacency context that shapes pattern coherence.

D. Human Trials

Human results validate the agent-based findings while revealing where they diverge. The strong correlation between agent completion and human progress ($r_s = 0.582$, $p < .001$) confirms A* completion as a useful proxy for human traversability. Large window levels matched original levels on playability, enjoyment, coherence, and replayability, mirroring the agent-based plateau at 6×6 .

The 1×1 anomaly illustrates the limits of agent-only evaluation. Despite achieving 75.6% agent completion, these levels received the lowest human ratings across all five dimensions (median playability 1.0/7). Human players perceived the structural incoherence that A* agents sidestep through dense platform abundance, motivating multi-metric evaluation.

Agent completion predicted playability, enjoyment, challenge, and replayability ratings but not coherence ($\beta = 0.47$, $p = .312$). Coherence appears to capture spatial logic and design consistency that traversal metrics cannot measure. Large window levels were rated significantly lower than original levels only on challenge ($\beta = -0.753$, $p = .025$), likely because near-uniform completion rates at 6×6 and 14×6 reduce perceived difficulty relative to original levels with deliberate difficulty progression.

E. Implications for Designer Control

Window size directly maps to spatial concepts, offering an intuitive control mechanism. Rather than specifying complex constraints, designers simply ask “How much context should patterns capture?” Effective window sizes correspond to the spatial scale of gameplay-critical mechanics. For platformers, this means capturing movement capabilities such as jump heights and traversal distances. Too small fails to encode functional constraints; too large over-constrains generation, sacrificing diversity. Thus, designers should match window

size to their production needs where immediately playable outputs require larger windows, and outputs destined for human curation benefit from smaller windows’ increased novelty.

Using a diverse multi-source input increases generation novelty (as evidenced by higher edit distances at 14×6) but substantially reduces playability at smaller window sizes. Designers prioritizing diversity can pair large windows with multi-source inputs to partially offset over-constraint, since the “All” condition achieves near-perfect completion at 14×6 while maintaining greater structural variety than any single-source configuration.

The strong correlation between agent completion and human progress ($r_s = 0.582$) supports using agent-based testing for rapid, large-scale screening before committing to human playtesting. However, the 1×1 anomaly illustrates that levels with high agent completion but low structural coherence may still disappoint human players. A multi-metric approach can better identify levels that are both traversable and coherent prior to human evaluation.

F. Limitations

Agent-based evaluation measures functional traversability rather than subjective experience. The agent’s success on dense but incoherent 1×1 levels illustrates this gap. Our human trials partially address this, though the small sample ($n = 25$) and university-age participants limit generalizability. Additionally, our human trial study used only single-source WFC levels as including the “All” condition would substantially expand an already minimal sample.

This study examines only *Super Mario Bros.* with a specific tile representation. Physics-based window sizing principles should generalize to other platformers, but optimal values must be determined per game. In general, minimum window dimensions should span the game’s fundamental movement primitives—jump height, traversal distance, and any other movement-defining abilities—as these define the spatial scale at which functional patterns must be captured. For games with power-ups or variable movement states, dimensions should reflect the most capable state to ensure generated levels remain traversable across all conditions. We also evaluate each window size independently, and hierarchical or adaptive approaches may achieve better playability-diversity trade-offs.

WFC generation failure (contradiction) rates were not tracked. Instead, levels producing contradictions were regenerated, but the frequency of regeneration attempts is unknown.

VII. CONCLUSION & FUTURE WORK

We present the first systematic investigation of WFC window size on functional playability of generated platformer levels, validated by both agent-based evaluation and human player trials. Across nine window sizes and 153,000 generated levels, completion rates range from 32% (2×2) to 99.9% (6×6 , 14×6), with a clear plateau once windows capture Mario's jump height. Human trials confirm this pattern as large window levels were rated comparably to original *Super Mario Bros.* levels on playability, enjoyment, coherence, and replayability, while smaller window levels scored significantly lower.

Serving as an intuitive designer control dial, larger windows promote immediate playability while smaller windows allow for novelty with post-generation curation. The 1×1 anomaly highlights the limits of agent-only evaluation. Dense but incoherent levels are completable by A* but rated poorly by humans, underscoring the need for multi-metric evaluation.

Future work should explore hierarchical and adaptive WFC using multiple window sizes simultaneously, and apply this methodology to other game genres to test whether physics-based window sizing principles generalize.

REFERENCES

- [1] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 9, no. 1, pp. 1–22, 2013.
- [2] M. Gumin, "mxgmn/WaveFunctionCollapse," Aug. 2016.
- [3] M. Babin and M. Katchabaw, "Leveraging Reinforcement Learning and WaveFunctionCollapse for Improved Procedural Level Generation," in *Proceedings of the 16th International Conference on the Foundations of Digital Games, FDG '21*, (New York, NY, USA), pp. 1–8, Association for Computing Machinery, Oct. 2021.
- [4] M. Lux, "Procedural Content Generation - The Open Source Success Story of Wave Function Collapse," *SIGMultimedia Rec.*, vol. 15, p. 6:1, Dec. 2024.
- [5] J. Togelius, S. Karakovskiy, J. Koutník, and J. Schmidhuber, "Super mario evolution," in *2009 IEEE Symposium on Computational Intelligence and Games*, pp. 156–161, IEEE, 2009.
- [6] D. Cheng, H. Han, and G. Fei, "Automatic generation of game levels based on controllable wave function collapse algorithm," in *International Conference on Entertainment Computing*, pp. 37–50, 2020.
- [7] A. Sandhu, Z. Chen, and J. McCoy, "Enhancing wave function collapse with design-level constraints," in *Proceedings of the 14th International Conference on the Foundations of Digital Games, FDG '19*, (New York, NY, USA), pp. 1–9, Association for Computing Machinery, Aug. 2019.
- [8] T. S. L. Langendam and R. Bidarra, "miWFC - Designer Empowerment through mixed-initiative Wave Function Collapse," in *Proceedings of the 17th International Conference on the Foundations of Digital Games, FDG '22*, (New York, NY, USA), pp. 1–8, Association for Computing Machinery, Nov. 2022.
- [9] Y. Nie, S. Zheng, Z. Zhuang, and J. Togelius, "Nested Wave Function Collapse Enables Large-Scale Content Generation," *IEEE Transactions on Games*, vol. 16, pp. 892–902, Dec. 2024.
- [10] M. Beukman, B. Ingram, I. Liu, and B. Rosman, "Hierarchical Wave-Function Collapse," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 19, pp. 23–33, Oct. 2023.
- [11] S. Alaka and R. Bidarra, "Hierarchical semantic wave function collapse," in *Proceedings of the 18th International Conference on the Foundations of Digital Games*, pp. 1–10, 2023.
- [12] P. C. Merrell, *Model synthesis*. PhD thesis, University of North Carolina at Chapel Hill, 2009.
- [13] I. Karth and A. M. Smith, "WaveFunctionCollapse is constraint solving in the wild," in *Proceedings of the 12th International Conference on the Foundations of Digital Games, FDG '17*, (Hyannis, Massachusetts), pp. 1–10, Association for Computing Machinery, Aug. 2017.
- [14] A. Newgas, "Tessera: A Practical System for Extended WaveFunctionCollapse," in *Proceedings of the 16th International Conference on the Foundations of Digital Games, FDG '21*, (New York, NY, USA), Association for Computing Machinery, 2021.
- [15] S. Cooper, "Sturgeon: tile-based procedural level generation via learned and designed constraints," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, vol. 18, pp. 26–36, 2022.
- [16] R. Christie, B. Kitchen, W. Tumilowicz, S. Hooper, and B. C. Wünsche, "Procedurally Generating Large Synthetic Worlds: Chunked Hierarchical Wave Function Collapse," in *2024 39th International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pp. 1–6, Dec. 2024.
- [17] R. Bailly and G. Levieux, "Genetic-WFC: Extending Wave Function Collapse With Genetic Search," *IEEE Transactions on Games*, vol. 15, pp. 36–45, Mar. 2023.
- [18] A. Karpetskiy, P. Napieralski, and D. Szajerman, "Hybrid Procedural Level Generation Using Wave Function Collapse and Genetic Algorithms," in *Computational Science - ICCS 2025 Workshops* (M. Paszynski, A. S. Barnard, and Y. J. Zhang, eds.), (Cham), pp. 111–118, Springer Nature Switzerland, 2025.
- [19] S. Karakovskiy and J. Togelius, "The Mario AI Benchmark and Competitions," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 6, pp. 89–92, June 2012.
- [20] J. Togelius, S. Karakovskiy, and R. Baumgarten, "The 2009 Mario AI Competition," in *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1–8, 2010.
- [21] A. Summerville and M. Mateas, "Super Mario as a String: Platformer Level Generation via LSTMs," *arXiv preprint arXiv:1603.00930*, 2016.
- [22] M. Guzdial, N. Sturtevant, and B. Li, "Deep Static and Dynamic Level Analysis: A Study on Infinite Mario," in *Proceedings of the 3rd Experimental AI in Games (EXAG) Workshop*, 2016.
- [23] A. Canossa and G. Smith, "Towards a Procedural Evaluation Technique: Metrics for Level Design," in *Proceedings of the 10th International Conference on the Foundations of Digital Games*, 2015.
- [24] C. Pedersen, J. Togelius, and G. N. Yannakakis, "Modeling player experience for content creation," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 2, no. 1, pp. 54–67, 2010.
- [25] C. Holmgård, A. Liapis, J. Togelius, and G. N. Yannakakis, "Evolving Personas for Player Decision Modeling," in *Proceedings of the IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 1–8, 2014.
- [26] C. Holmgård, M. C. Green, A. Liapis, and J. Togelius, "Automated Playtesting with Procedural Personas through MCTS with Evolved Heuristics," *IEEE Transactions on Games*, vol. 11, no. 4, pp. 352–362, 2019.
- [27] G. Smith and J. Whitehead, "Analyzing the Expressive Range of a Level Generator," in *Proceedings of the Workshop on Procedural Content Generation in Games, PCGames '10*, pp. 4:1–4:7, 2010.
- [28] B. Horn, S. Dahlsgog, N. Shaker, G. Smith, and J. Togelius, "A Comparative Evaluation of Procedural Level Generators in the Mario AI Framework," in *Proceedings of the 9th International Conference on the Foundations of Digital Games*, 2014.
- [29] S. Dahlsgog and J. Togelius, "Procedural Content Generation Using Patterns as Objectives," in *Proceedings of EvoGames, part of EvoStar*, pp. 325–336, 2014.
- [30] D. Šosvald and J. Gemrot, "Super mario a-star agent reloaded," in *2025 IEEE 37th International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 1308–1315, IEEE, 2025.
- [31] S. Dai, X. Zhu, N. Li, T. Dai, and Z. Wang, "Procedural level generation with diffusion models from a single example," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 10021–10029, 2024.
- [32] N. Shaker, M. Nicolau, G. N. Yannakakis, J. Togelius, and M. O'Neill, "Evolving levels for super mario bros using grammatical evolution," in *2012 IEEE Conference on Computational Intelligence and Games (CIG)*, pp. 304–311, IEEE, 2012.
- [33] H. Schaa and N. A. Barriga, "Evaluating the expressive range of super mario bros level generators," *Algorithms*, vol. 17, no. 7, p. 307, 2024.