Automatic Generation of Super Mario Levels via Graph Grammars

Eduardo Hauck and Claus Aranha

s1290102 Hajime Fukai

Introduction

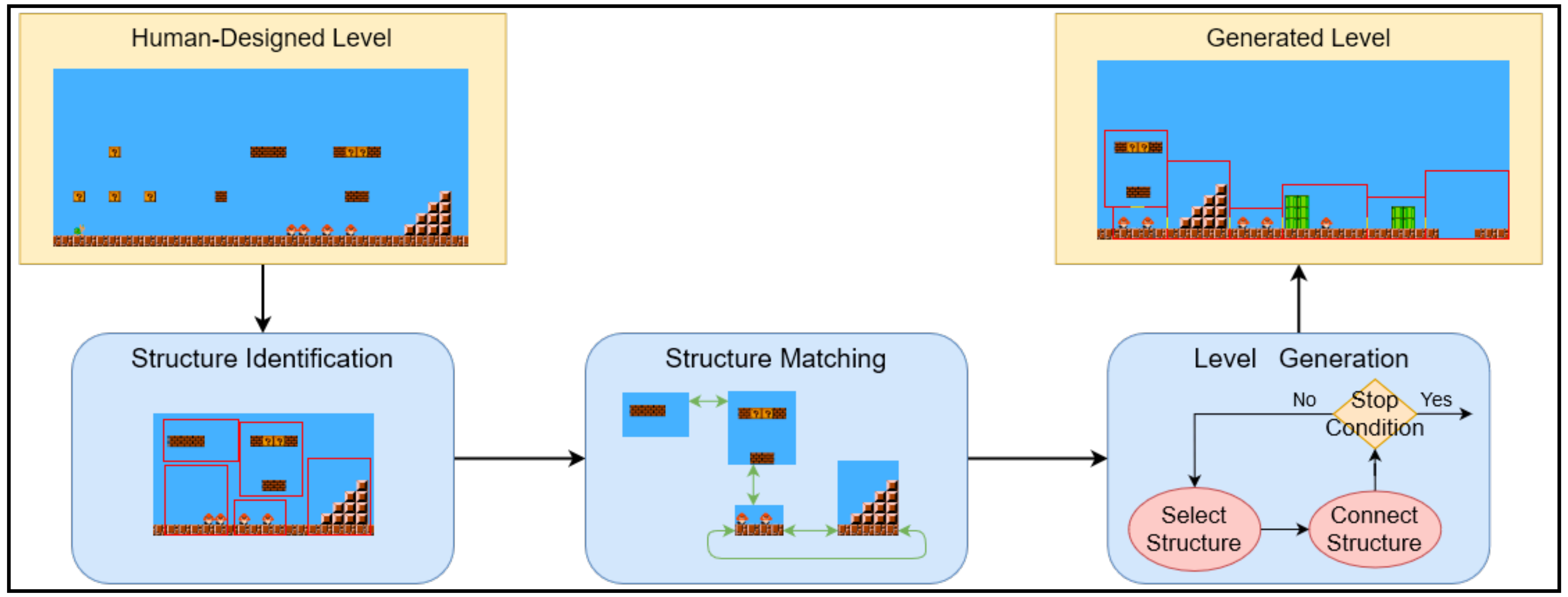
Overview of this paper

- This paper proposes a new level-generation system for Super Mario Bros
  - Procedural Content Generation (PCG)
- Adopts graph-based approach
  - A game level is modeled as a graph
- Aims to be explainable, and easy to cooperate with human designers
  - Machine Learning is **not** applied, due to its complexity
  - A designer can alter a part of generation process easily

Introduction

Outline of the proposed system

- The proposed system takes human-designed levels, as input
- Then, the system extracts and recombines patterns of the input levels
- These extracted patterns are stored in simple files. So they can be easily modified, or created
- There are three stages to generate a new level:
  1. Structure Identification
  2. Structure Matching
  3. Level Generation



Related works

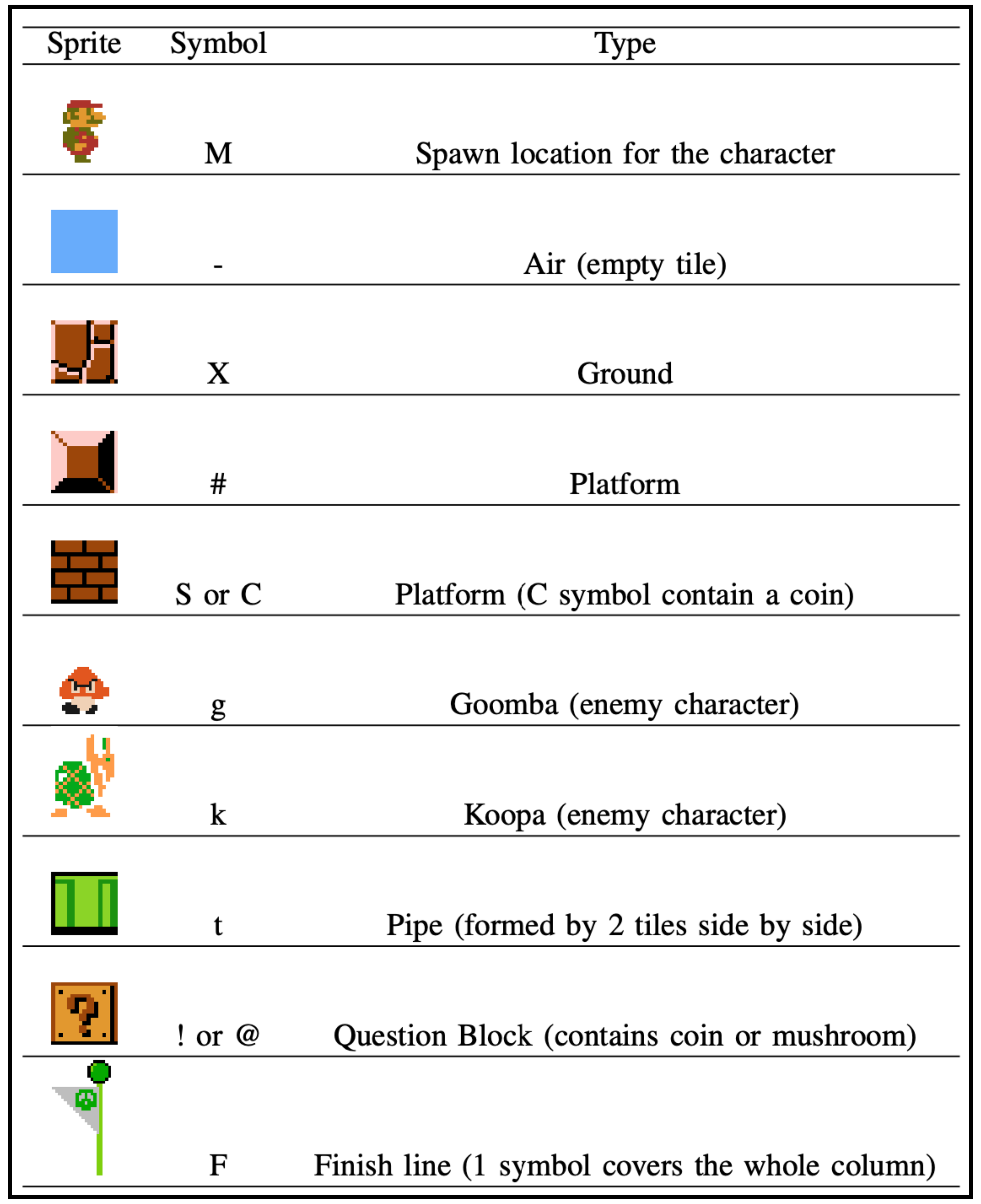
“Graph Grammars for Super Mario Bros Levels”

- S. Londoño and O. Missura “Graph Grammars for Super Mario Bros Levels”
  - This paper also features graph-grammar based approach for Super Mario Bros level generation
  - A level is represented as a directed graph
  - Introduced the concept of reachability of a player
    - If a player can navigate from a platform to another platform, reachability edge is added between them
  - No implementation detail, or analysis of the theory was provided

Related works

Mario AI Framework

- Mario AI Framework is built on top of Infinite Mario Bros, which is an open-source Super Mario Bros clone
- This framework can read level data from a text file. Each character in the text file represents corresponding sprite in the screen
- This framework is used to benchmark generated levels in this research



Sprite	Symbol	Type
	M	Spawn location for the character
	-	Air (empty tile)
	X	Ground
	#	Platform
	S or C	Platform (C symbol contain a coin)
	g	Goomba (enemy character)
	k	Koopa (enemy character)
	t	Pipe (formed by 2 tiles side by side)
	! or @	Question Block (contains coin or mushroom)
	F	Finish line (1 symbol covers the whole column)

Symbols used to represent each sprite in Mario AI Framework

Detail of the generation process

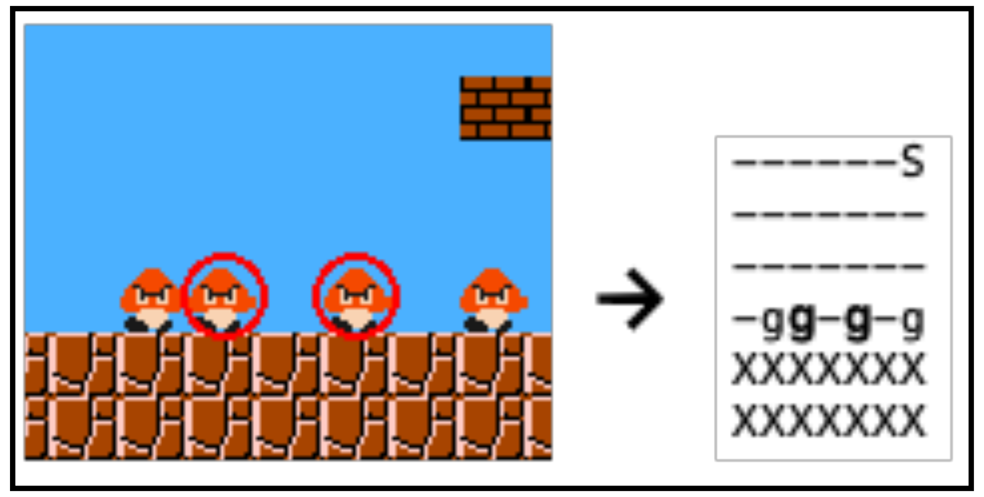
1. Structure Identification stage

- Structure Identification stage identifies structures in the input levels
    - A structure is defined to be a subsection of a level
- This stage takes three parameters
    - *L*: a set of levels
    - *n*: the minimum number of identified structures in each level
    - *d*: the base size (width and height) of a structure

Detail of the generation process

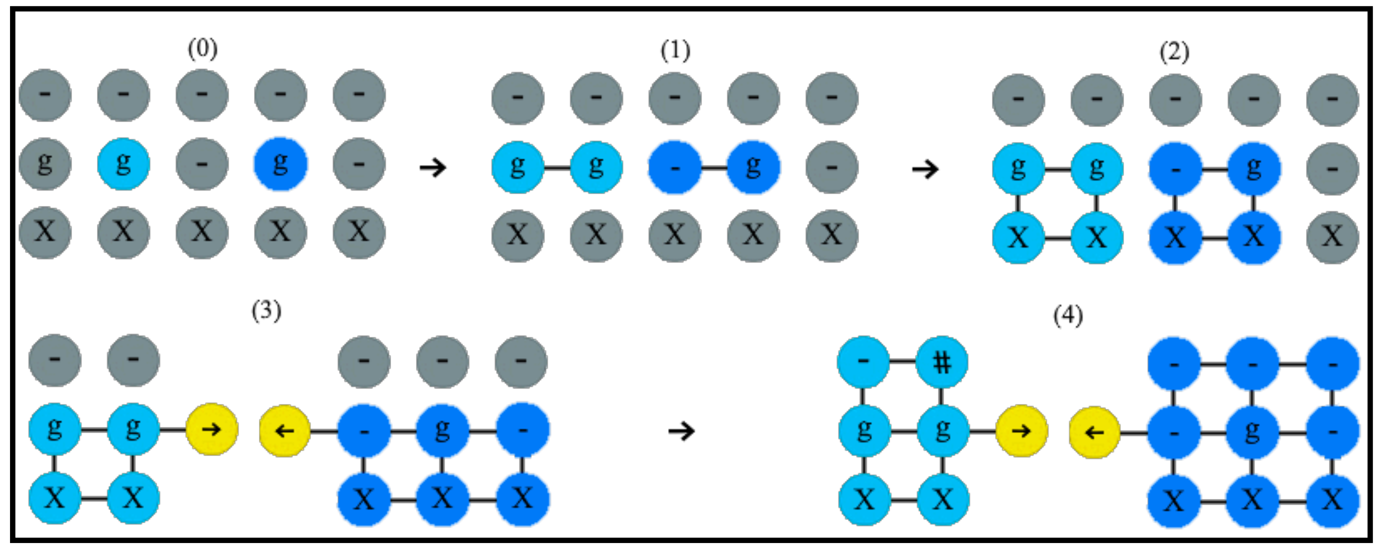
1. Structure Identification stage

- First, *n* non-air tiles in a level are selected in a way that they are equally spaced
    - *Suppression via Disc Covering algorithm* proposed by Gauglitz et al. is applied
- Then corresponding nodes are created for each selected tile
- Next, each node is expanded until reaching the size *d*
    - If two structures collide, connector nodes are added for each of the colliding structures
- Continues until all structures reach the size *d* or halt expansion due to collision



```
------S
-------
-------
-gg-g-g
XXXXXXX
XXXXXXX
```

Selected tiles



Expansion of nodes

Detail of the generation process

2. Structure Matching stage

- Structure Matching stage checks which structure pairs can be connected
  - Identifies good and bad connections among structures
- For every pair of structures *a* and *b*, these two constraints are evaluated:
  - Structural consistency: connector nodes of *a* and *b* must be toward the opposite direction
  - Reachability: the player must be able to navigate from *a* to *b*. This is verified by the concept proposed by Londoño and Missura
- The obtained list of possible connections is a set of grammar rules to execute substitutions

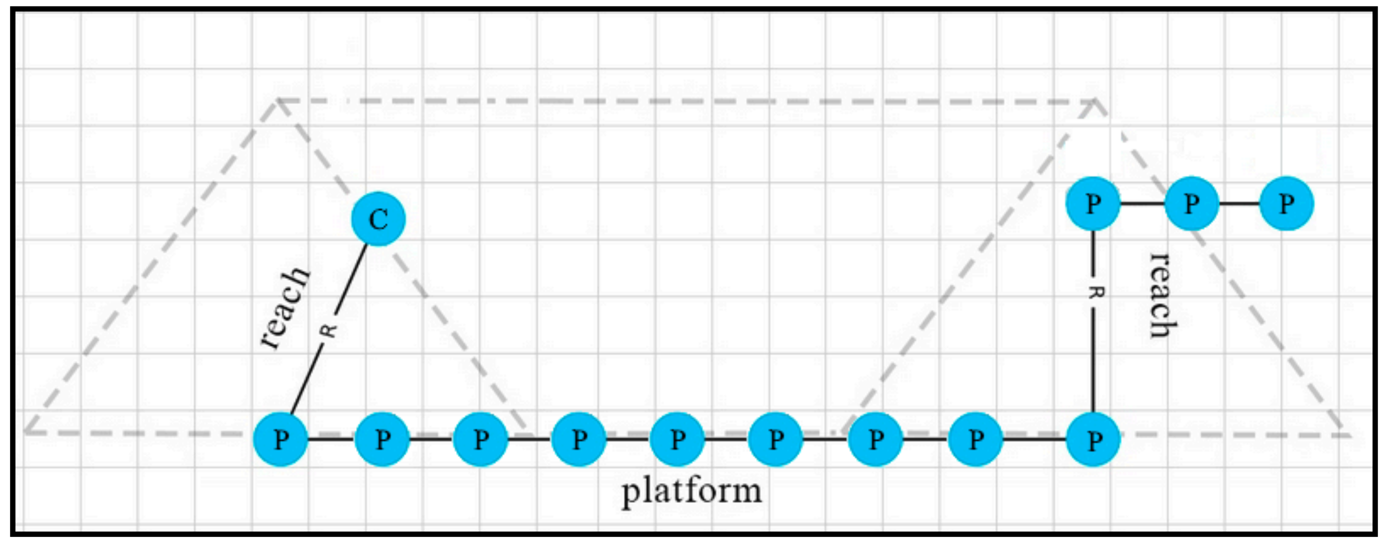


Illustration of reachable area, given the platform *P*

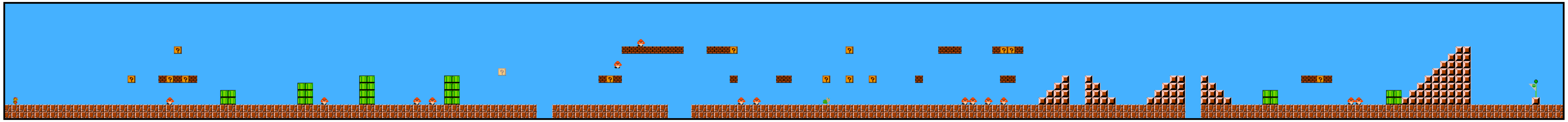
Detail of the generation process

3. Level Generation stage

- Level Generation stage actually generates a new level, based on a grammar obtained from the previous stages
- The starting point is hand-coded, so that the player can spawn at a safe location. The starting point contains one connector node
- Then, given probability distribution iteratively determines the structures to be joined to each available connector node. If any of the constraints below are violated, backtrack to the previous state
    - The availability of unconnected connector node
    - No overlap of structures

Results

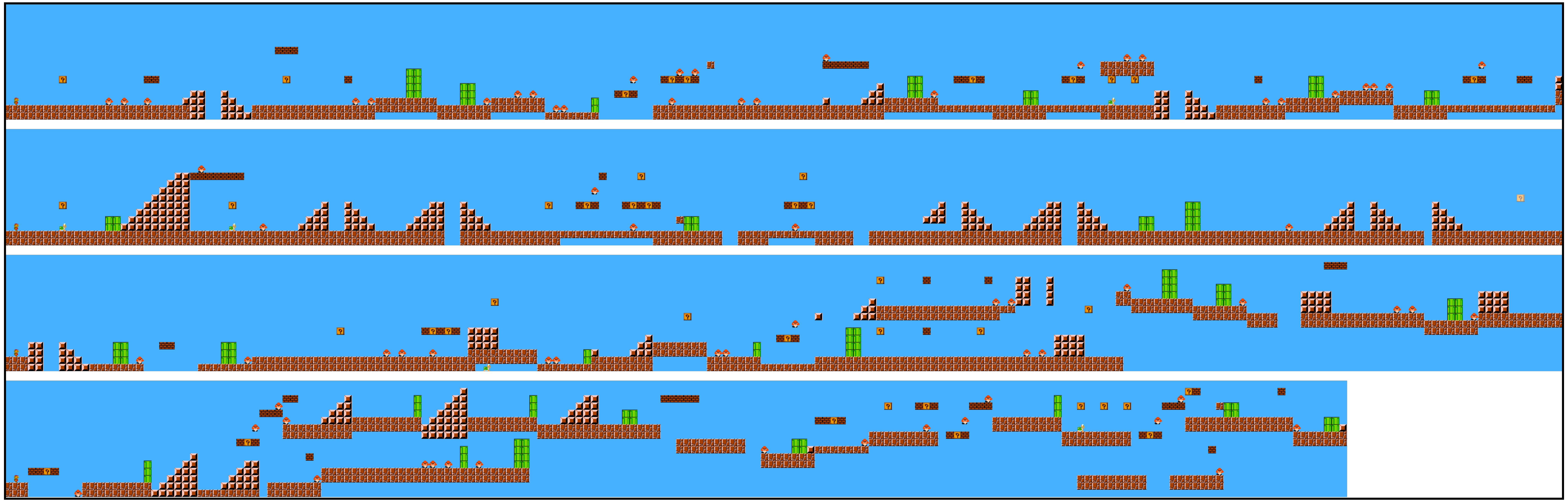
Generated levels



- This is World 1-1 and this was used as input level

Results

Generated levels

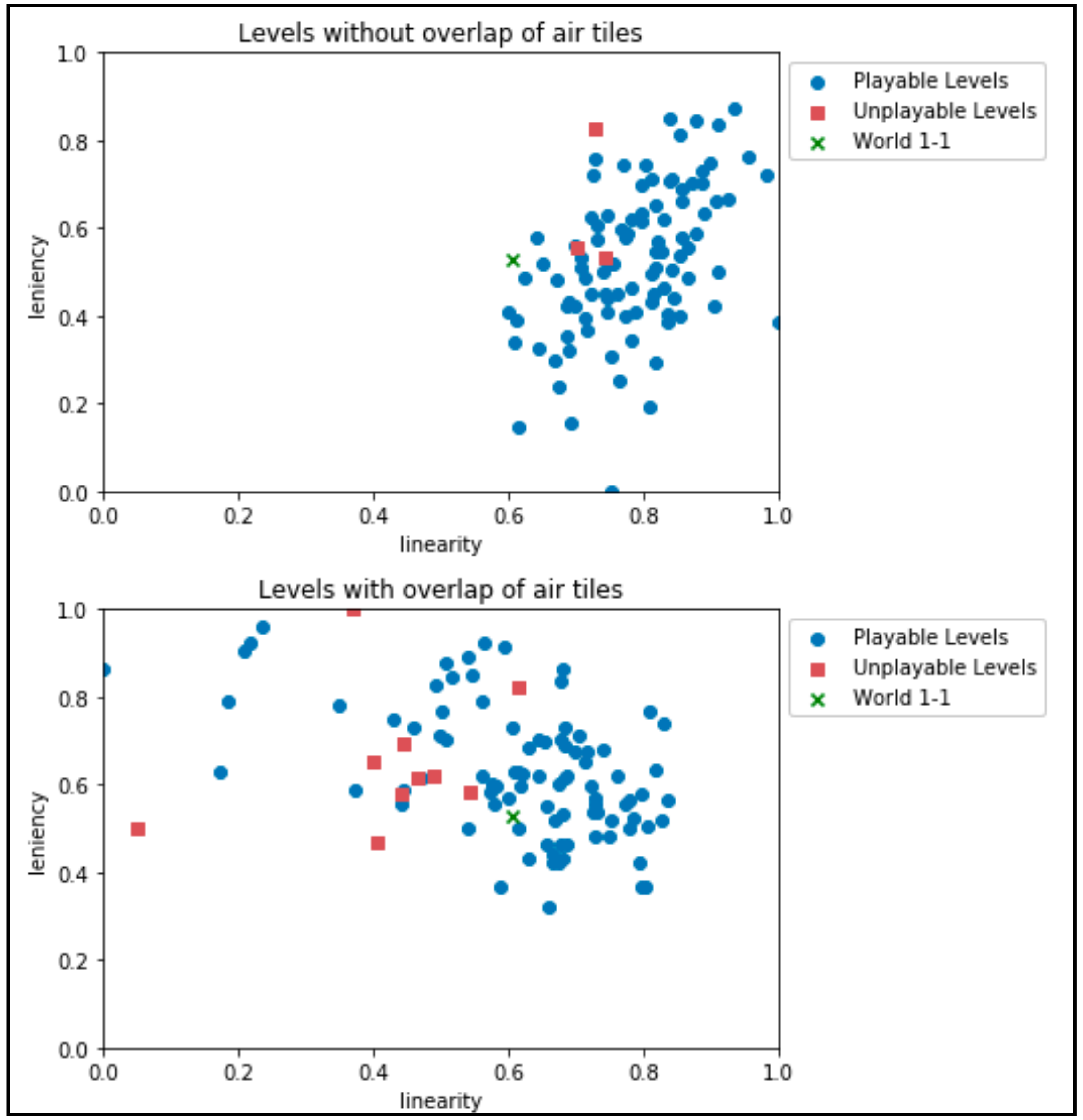


- These are the new levels generated by the system
- The two bottom levels were generated with different constraints where overlap of air tiles was allowed

Analysis

Analysis of the generated levels

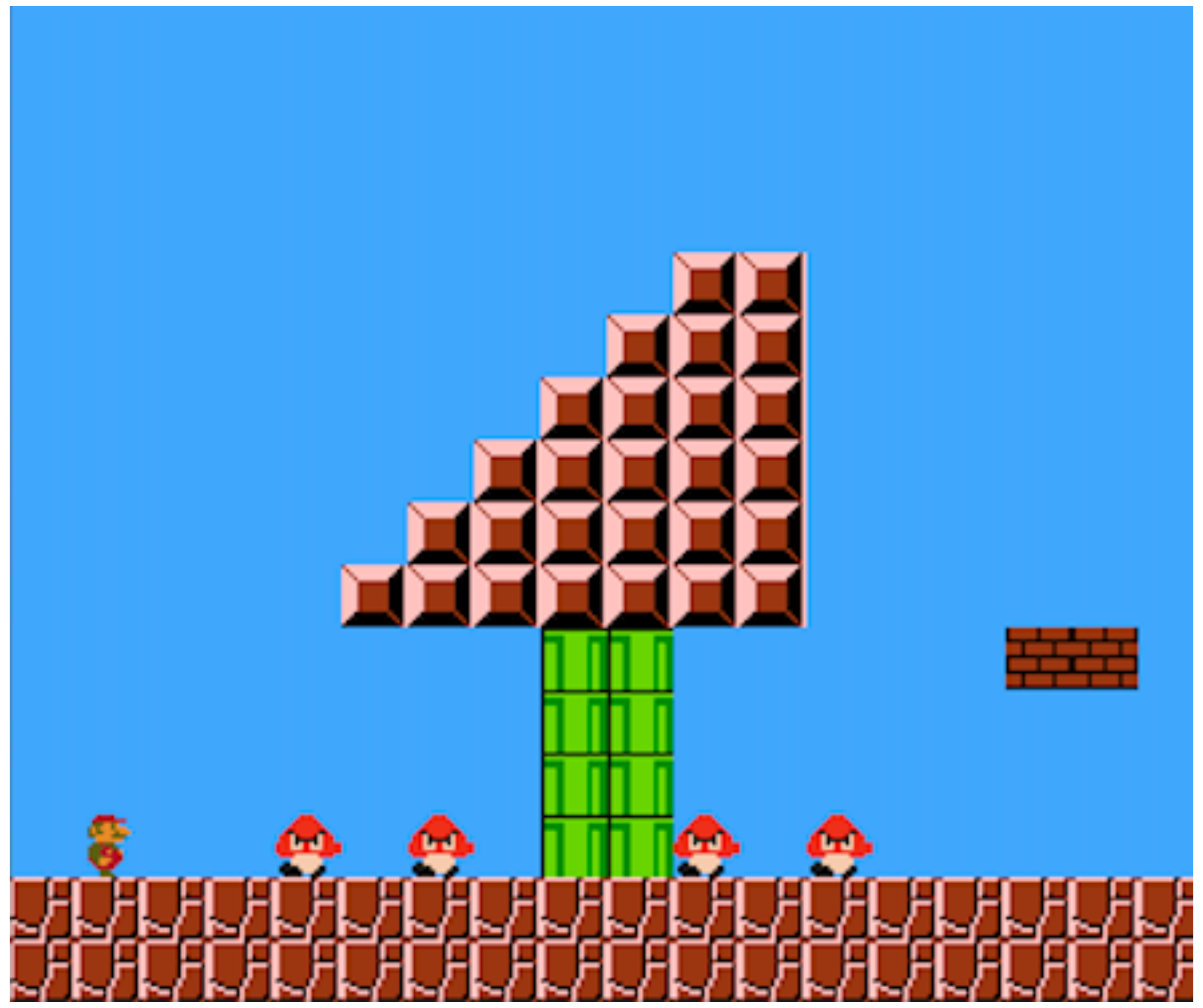
- The generated levels are analyzed by employing the leniency and linearity metrics, proposed by N. Shaker et al.
- In order to evaluate the playability of the levels, the Robin Baumgarten A* agent is employed
- When overlap of air tiles is not allowed, 3% of the levels were unplayable
- When overlap of air tiles is allowed, 10% of the levels were unplayable



Conclusions

Possibilities and limitations

- The proposed system generated playable Super Mario levels, with similar features presented in the input level
- The system allows human designers to extend the generation process easily
- However, the analysis revealed some limitations of the proposed system
  - The system generated some unplayable levels. This was caused because the reachability evaluation was incomplete and unable to model all possible scenarios



An example of unplayable generated level