Rube Goldberg Machines := RGM

Controllable Level Blending between Games using Variational Autoencoders
https://arxiv.org/pdf/2002.11869.pdf

- use Variational Autoen-coders (VAEs) to blend existing levels from different games into a new level
- new level combine properties of both games
- VAEs capture latent design space across games
- allows for generation of level segments that optimize functions and satisfy specific properties -> co-creative level design
- evolutionary search to evolve segments satisfying different designer-specified constraints and proportions
- co-creative level design -> human designers collaborate with procedural generators
- main issue of playability of segments
- only based on two games not multiple

-> when combining RGMs, each machine can be seen as a game, how complex should the recombined parts in EA be?

Multi-Objective level generator generation with Marahel
https://arxiv.org/pdf/2005.08368.pdf

- system to design constructive generators by searching the space of constructive level generators defined by Marahel language
- use NSGA-II, multi-objective optimization algorithm, to search for generators

-> multi-objective optimization approach, otherwise not much useful information

TOAD-GAN: Coherent Style Level Generation from a Single Example
https://arxiv.org/pdf/2008.01531.pdf

- use GANs to generate SMB levels with arbitrary size based on only a single training example level

Mario Level Generation From Mechanics Using Scene Stitching
https://arxiv.org/pdf/2002.02992.pdf

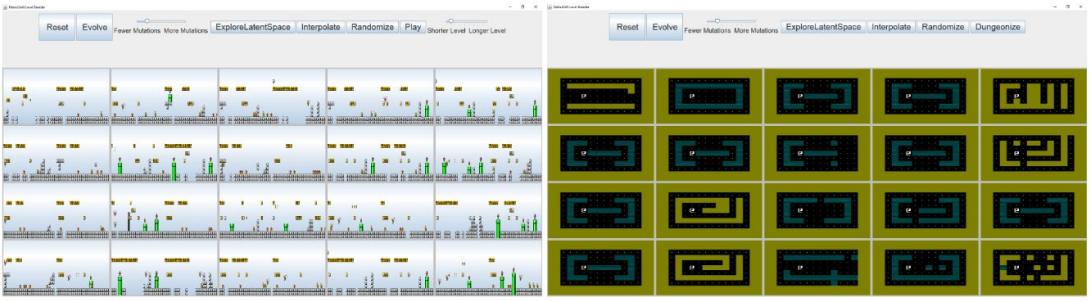
- level generation method for super mario by stitching together pre-generated “scenes”
- scenes are level parts that contain one or multiple specific mechanics
- FI-2Pop algorithm
    - feasible -> levels must be playable (constraint)
    - infeasible -> levels must match mechanics (fitness)
- Mutators
    - delete random scene
    - add random scene
    - split scene in half and replace with new left and right
    - merge, add same mechanic scene on left/right then replace original and new with one scene which has combined mechanics
    - random change
- structural diversity lower than original

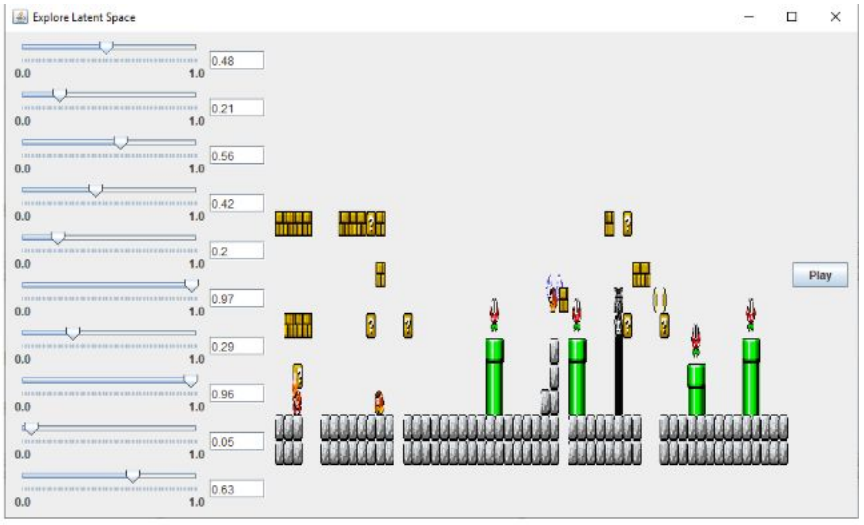
-> might be useful to split RGMs into feasible and infeasible populations

Procedural automation of general game level generation: the good, the bad and the ugly
https://www.researchgate.net/profile/Noah_Posner/publication/340630503_Procedural_automation_of_general_game_level_generation_the_good_the_bad_and_the_ugly/links/5e95ebae299bf1307997d53d/Procedural-automation-of-general-game-level-generation-the-good-the-bad-and-the-ugly.pdf

- monumental goal of AI model general solutions to perform a variety of tasks that normally demand human intelligence
- main problem of level generation -> quantization of many variables that make level generation hard
- no single generally applicable heuristic to measure levels and components
- complex to quantify “enjoyableness” of levels
- non-descriptive and subjective quality
- AI agents tests whether levels playable
- SB (Generate and Test) ranked better that constructive
- Hyper-Agent / Hyper-Heuristic approach -> choosing best agent based on the context and “learned” decision tree
- design patterns -> players ingame tasks
- use patterns for level generation for any game
- Graphs enable semantic relationship between elements of levels
- MCTS, NMCS, NRPA, GANs, unsupervised learning, FI-2Pop, noise/fractals, solver-based, grammar-based, cellular automata, LSTMs
- find accurate benchmarking methods and best construction methods

Interactive Evolution and Exploration Within LatentLevel-Design Space of Generative Adversarial Networks
https://arxiv.org/pdf/2004.00151.pdf

- interactive Latent Vector Evolution of tile-based games using GANs
- interactive evolution via selective breeding
- user sees N = 20 sample levels and selects best individuals
- 

- remaining spots filled by offspring
- user controlled mutation
- option to replace selected genomes with random ones to prevent getting stuck converging into an area
- Manual Exploration of Latent Space
- slider to change Vector
- 

- or interpolate slider (combine two levels)
- interaction with generated content -> users can play and test generated levels
- User Study
  - some users prefer direct evolution because combination is complicated
  - sophisticated users prefer combination since they can take full advantage of options
  - latent vectors not correlated to easily understandable features -> users not sure what changing latent vectors will result in, can not control the direction they want the level to go (e. g. more enemies)

- exploring latent space useful to find users personal preferences
- integrate direct editing into IEC process
- make latent dimensions correspond to level features meaningful to human user
- mixed initiative system

-> IEC could prove very useful in RGM EA

IEC
https://www.peterolsted.com/documents/PCG_Level_Gen.pdf

Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation
http://sclab.yonsei.ac.kr/courses/09EC/papers/IECsurvey.pdf

- human fatigue
- pipeline during human evaluation

Searching the Latent Space of a Generative Adversarial Network to Generate DOOM Levels
https://ieee-cog.org/2019/papers/paper_79.pdf

- apply CMA-ES to search latent space of GAN trained to generate DOOM levels
- GAN able to exploit design patterns learned from training
- CMA-ES can effectively search design space for specific contents
- combination of SB-PCG & PCGML
- indirect level encoding as input vector of previously trained GAN
- real valued vector
- levels evaluated on 7 high level features -> fitness calculated as L^2 norm of difference between vector of features and target features
- Search Process -> Population of level vectors generated -> each vector provided as input to generator network which generated image-based representation of generated level -> comput fitness -> rank population based on fitness to update CMA-ES parameters
- CMA-ES outperforms hill climbing

Automatic Generation of Dungeons for Computer Games
https://pdfs.semanticscholar.org/2502/0f8d955aee07b7dd49a3ec23b1f2a8cf1d06.pdf

Unity
https://www.youtube.com/watch?v=OR0e-1UBEOU
https://www.youtube.com/watch?v=BMVef0bAtSE&list=PLoFAh-iWxtU4ESFh3YnMvQVXlEiRdTEBP&index=2
https://www.youtube.com/watch?v=1oXr16Tdfvo

https://www.youtube.com/watch?v=qWu3HkFgqfw
https://www.youtube.com/watch?v=G8KJWONEeGo&list=PLm7W8dbdfloi6DOIQDqfTx0DjGwo6tGqm
https://www.youtube.com/watch?v=pCBqgREiSUE
https://www.youtube.com/watch?v=V29O_Q7W2ZU

Generative Adversarial Network Rooms in Generative Graph Grammar Dungeons for The Legend of Zelda
https://arxiv.org/pdf/2001.05065.pdf

- GAN creates levels
- Graph Grammar connects levels into dungeon

Genotypes:

1. Liste an GameObjects (o. Referenz zu GO) + Input / Output information -> Jedes GameObject stellt Segment dar, z. B. Dominostein Folge o. Murmelbahn

- jedes GameObject speichern zu komplex
- Mutation -> Segment ersetzen
- Crossover mit passenden input/output (Korrektheit der neuen RGM prüfen)
- Position? Kette?
- Wie Angry Birds

2. Multidimensionale Liste -> gleichzeitig ausführbare Segmente

- Parallelisierung Problem
- bei 1-D Liste in 1 GameObject mehrere Teile definieren?

3. Expression Trees -> parallel

- crossover?
- probleme bei Operatoren

4. Graph -> NEAT

- GraphPaper