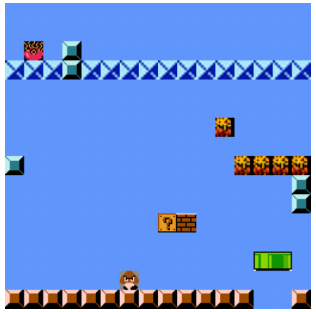
Generating and Blending Game Levels via Quality-Diversity in the Latent Space of a Variational Autoencoder

Anurag Sarkar and **Seth Cooper**

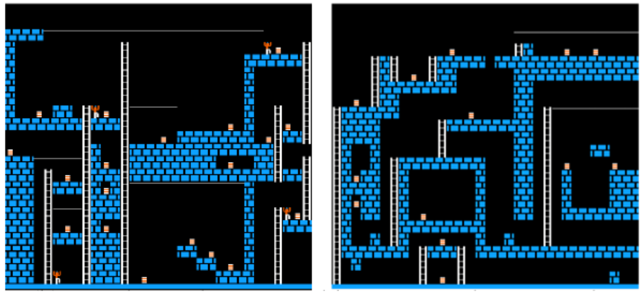
Northeastern University

Motivation

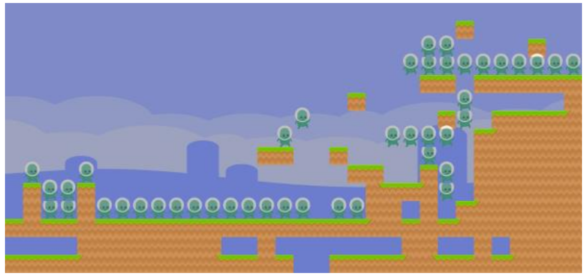
- Recent works have used variational autoencoders (VAEs) for generating and blending levels for several games
  - Due to generation via sampling, standard VAEs can't easily produce diverse content controllably



Sarkar, Yang and Cooper, 2019



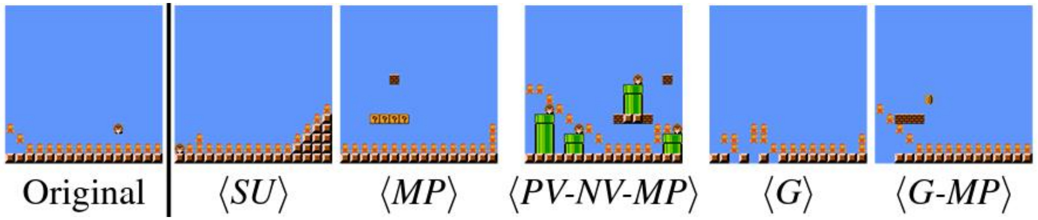
Thakkar et al., 2019



Sarkar et al., 2020

Motivation

- Recent works have used variational autoencoders (VAEs) for generating and blending levels for several games
    - Due to generation via sampling, standard VAEs can't easily produce diverse content controllably



Sarkar, Yang and Cooper, 2020

- Conditional VAEs
    - Use labels to control outputs
    - Require labeled data
    - Generation by modifying randomly sampled vectors via labels rather than exploring search space

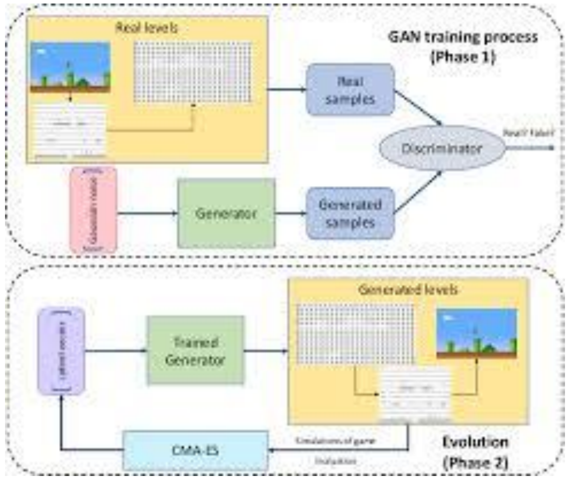
Sarkar and Cooper, 2021

Motivation

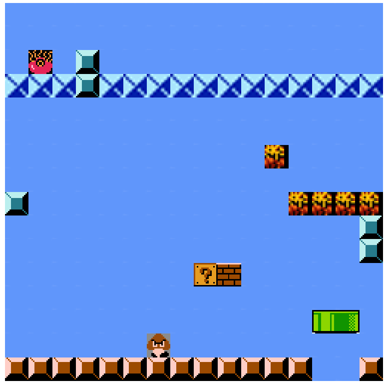
- Recent works have used variational autoencoders (VAEs) for generating and blending levels for several games
  - Due to generation via sampling, standard VAEs can't easily produce diverse content controllably

- Conditional VAEs
  - Use labels to control outputs
  - Require labeled data
  - Generation by modifying randomly sampled vectors via labels rather than exploring search space

- Latent Variable Evolution
  - Find optimal vectors in latent space
  - Produces single optimal solution



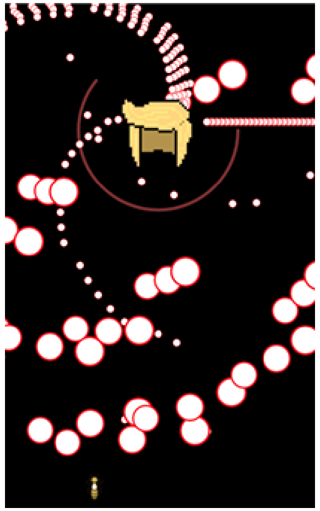
Volz et al., 2017



Sarkar, Yang and Cooper, 2019

Motivation

- Recent works have used variational autoencoders (VAEs) for generating and blending levels for several games
    - Due to generation via sampling, standard VAEs can’t easily produce diverse content controllably
- Conditional VAEs
    - Use labels to control outputs
    - Require labeled data
    - Generation by modifying randomly sampled vectors via labels rather than exploring search space
- Latent Variable Evolution
    - Find optimal vectors in latent space
    - Produces single optimal solution
- Quality-Diversity (QD) methods (e.g. MAP-Elites) designed to produce diverse content in 1 evolutionary run



Khalifa et al., 2018

Khalifa et al., 2019

Alvarez et al., 2019

Charity et al., 2020

Motivation

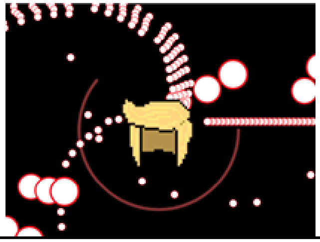
- Recent works have used variational autoencoders (VAEs) for generating and blending levels for several games
  - Due to generation via sampling, standard VAEs can't easily produce diverse content controllably

- Conditional VA
  - Use labels
  - Require la
  - Generatio labels rath

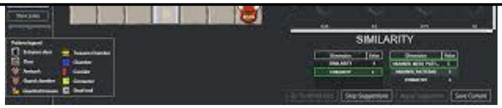
- Latent Variable
  - Find optimal vectors in latent space
  - Produces single optimal solution

- Quality-Diversity (QD) methods (e.g. MAP-Elites) designed to produce diverse content in 1 evolutionary run

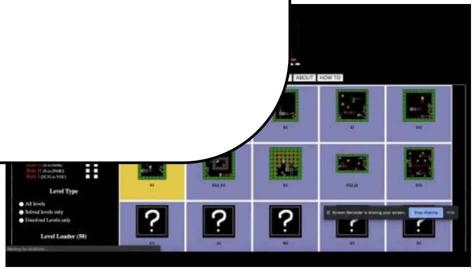
VAE+MAP-Elites to generate and blend a diverse range of levels



2019



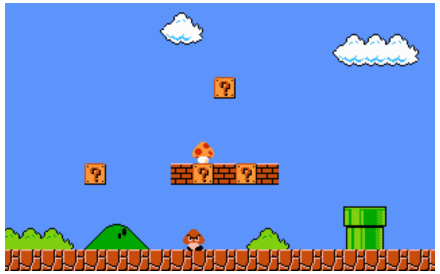
Alvarez et al., 2019



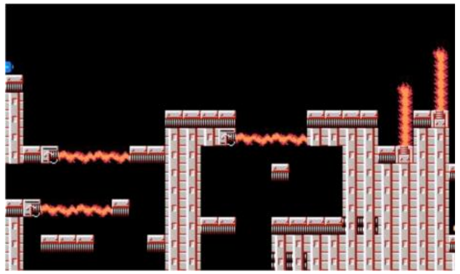
Charity et al., 2020

Approach

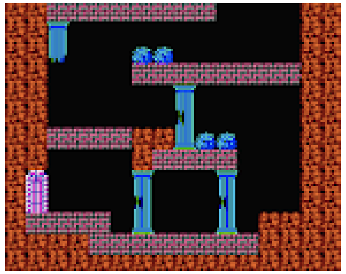
- Two-step approach
    - Train VAE on game levels from VGLC
        - 5 platformers individually
        - Mario+KI+Mega Man (Blend Elites)
        - 16x16 segments



Super Mario Bros. (SMB)

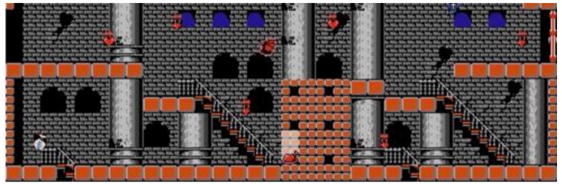


Mega Man (MM)



Kid Icarus (KI)

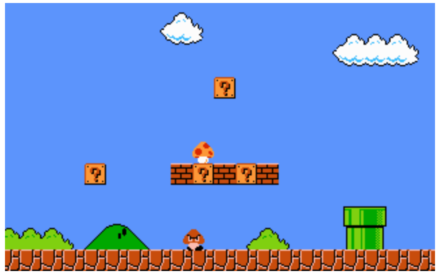
Ninja Gaiden (NG)



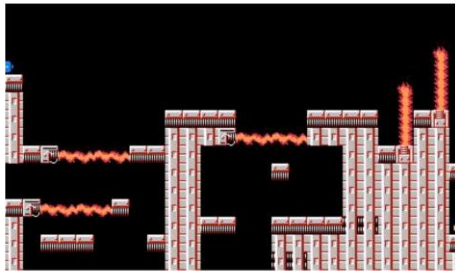
Castlevania (CV)

Approach

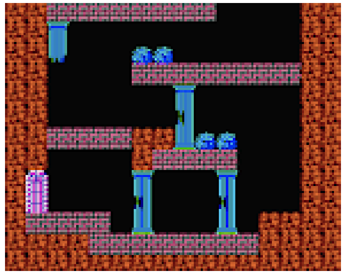
- Two-step approach
  - Train VAE on game levels from VGLC
    - 5 platformers individually
    - Mario+KI+Mega Man (Blend Elites)
    - 16x16 segments
  - Run MAP-Elites using the VAE latent space as search space



Super Mario Bros. (SMB)

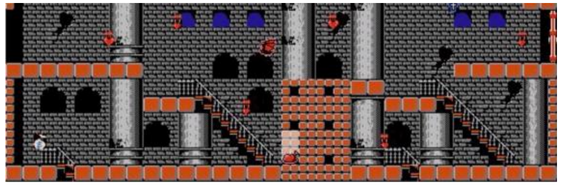


Mega Man (MM)



Kid Icarus (KI)

Ninja Gaiden (NG)



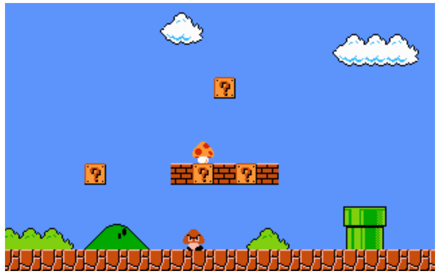
Castlevania (CV)

Approach

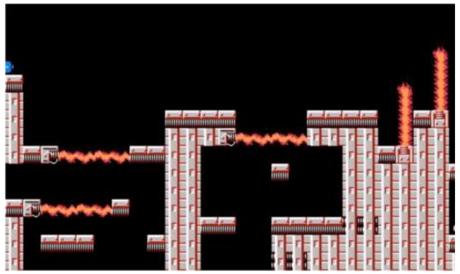
- Two-step approach
  - Train VAE on game levels from VGLC
    - 5 platformers individually
    - Mario+KI+Mega Man (Blend Elites)
    - 16x16 segments

  - Run MAP-Elites using the VAE latent space as search space

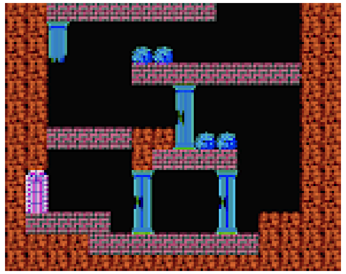
- Contributions
  - Hybrid PCGML+PCGQD approach combining VAEs and MAP-Elites for level generation/blending
  - First use of MAP-Elites for generating levels of Kid Icarus, Mega Man, Castlevania, Ninja Gaiden
  - Blend-Elites i.e. use of MAP-Elites for blending



Super Mario Bros. (SMB)

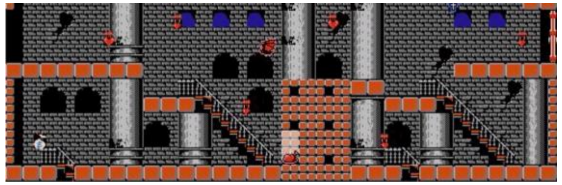


Mega Man (MM)



Kid Icarus (KI)

Ninja Gaiden (NG)



Castlevania (CV)

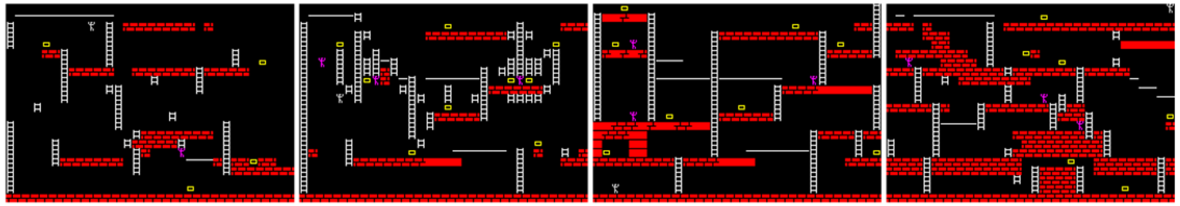
Approach

- Two-step approach
  - Train VAE on game levels from VGLC
    - 5 platformers individually
    - Mario+KI+Mega Man (Blend Elites)
    - 16x16 segments

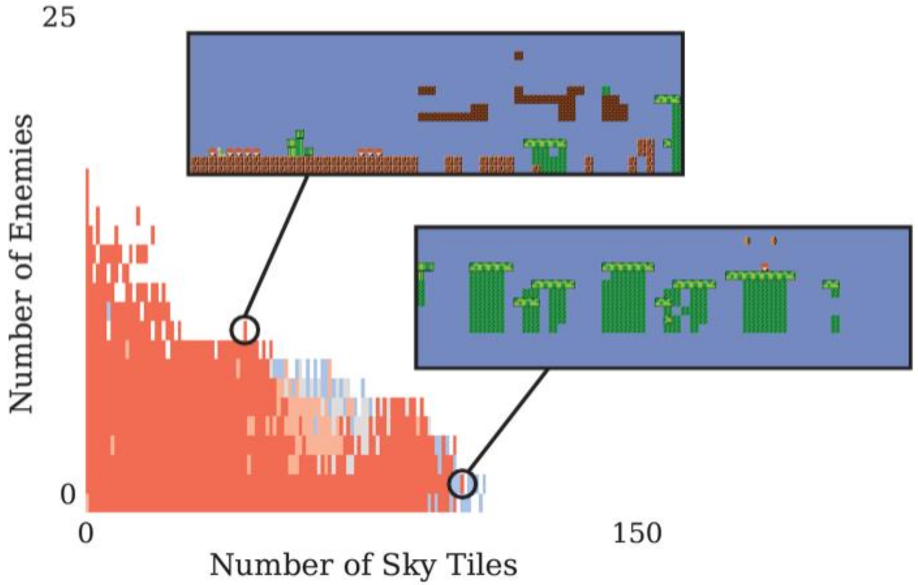
  - Run MAP-Elites using the VAE latent space as search space

- Contributions
  - Hybrid PCGML+PCGQD approach combining VAEs and MAP-Elites for level generation/blending
  - First use of MAP-Elites for generating levels of Kid Icarus, Mega Man, Castlevania, Ninja Gaiden
  - Blend-Elites i.e. use of MAP-Elites for blending

- Latent Space Illumination → process of running MAP-Elites in a learned latent space



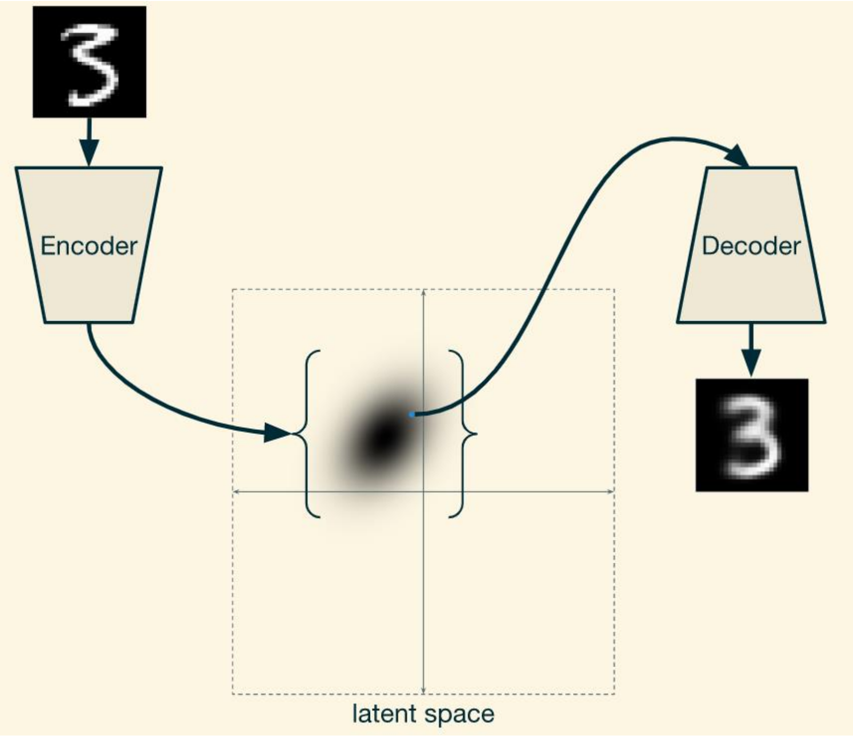
Steckel and Schrum, 2021

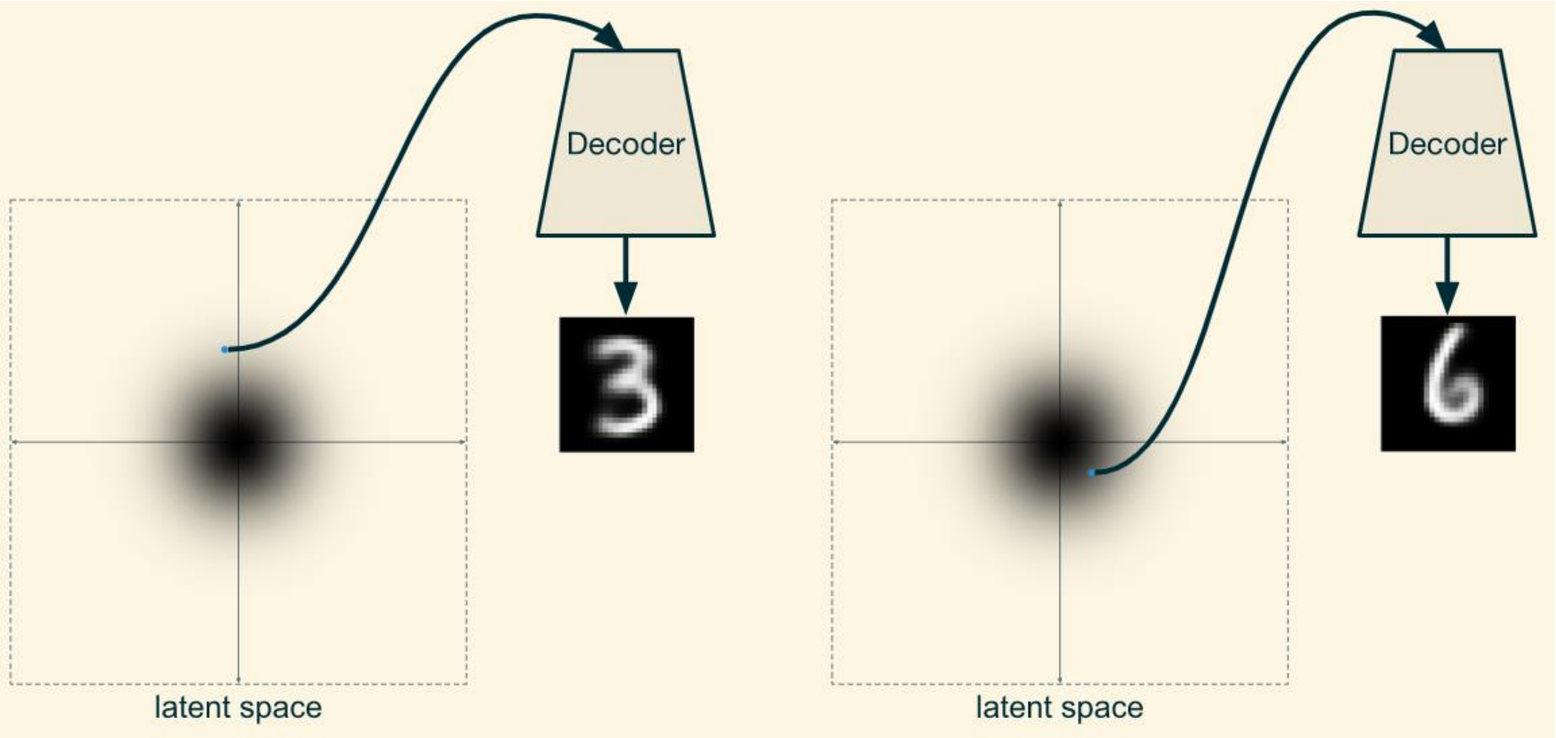


Fontaine et al., 2021

Variational Autoencoder (VAE)

- Latent variable generative models that learn continuous latent spaces
  - Encoder → input data to latent space
  - Decoder → latent space to reconstructed data
- Enables generation via sampling the latent space
- Latent space can serve as a continuous search space for evolution
- Learns genotype-to-phenotype mapping

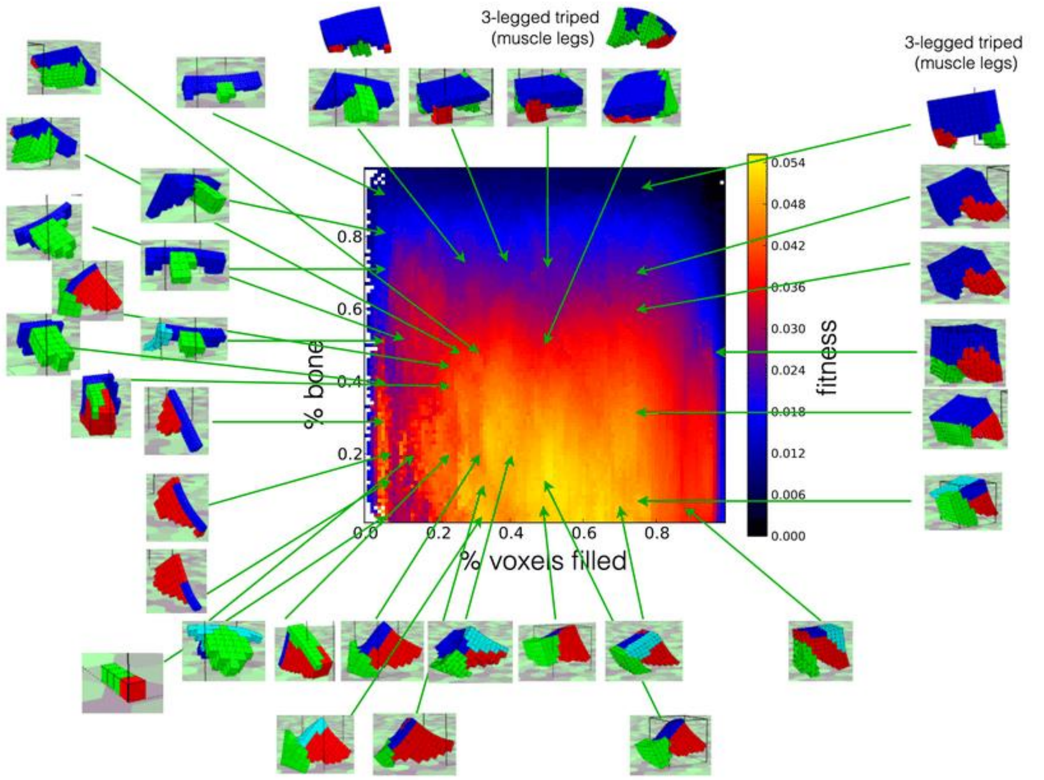




source: jdykeman.github.io/ml/2016/12/21/cvae.html

MAP-Elites

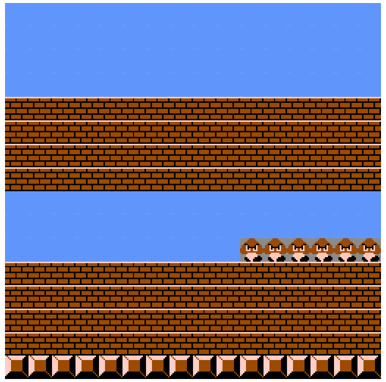
- Quality-diversity (QD) evolutionary algorithm that divides search space into cells based on behavior characteristics (BCs)
- Each cell corresponds to a different region of the behavior space
- Returns locally optimal solution in each cell based on a fitness function
- For games, fitness usually defined in terms of playability with BCs capturing level properties



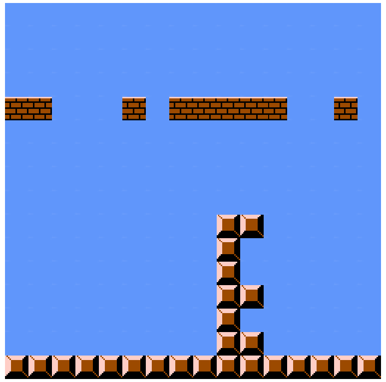
Mouret & Cully, 2015

Behavior Characteristics (BCs)

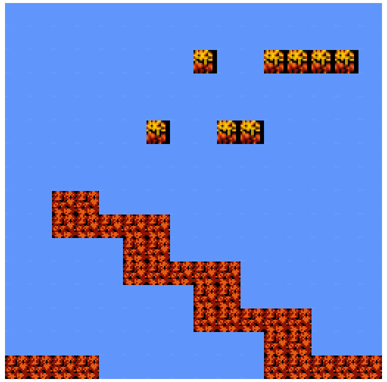
- Three sets of BCs
- Density-Nonlinearity (DE-NL)
  - Density - # tiles in a segment that aren't background or path tiles (range: [0,256])
  - Nonlinearity - how well segment's topology follows a straight line (range: [0,64])
  - Archive: 257x65 = 16,705 cells
- Symmetry-Similarity (SYM-SIM)
  - Symmetry of a segment along both axes (range: [0,256])
  - Similarity of a generated segment compared to segments in training data ([0,32])
  - Archive: 257x33 = 8,481 cells



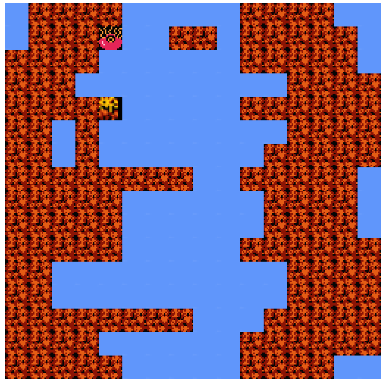
SMB
High DEN, Low NL



SMB
Low DEN, High NL



KI
Low SYM, High SIM



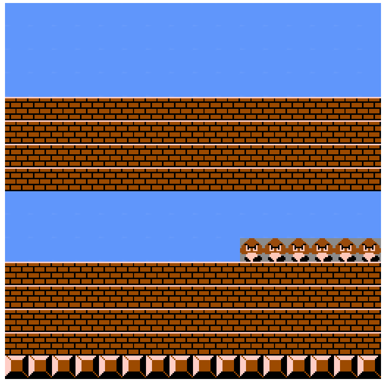
KI
High SYM, Low SIM

Behavior Characteristics (BCs)

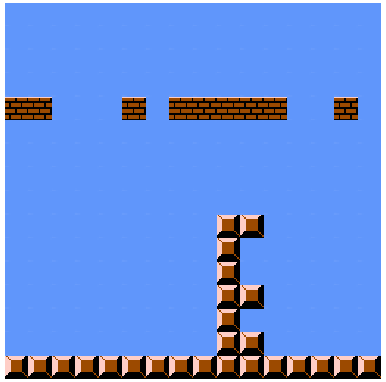
- Three sets of BCs

- Density-Nonlinearity (DE-NL)
  - Density - # tiles in a segment that aren't background or path tiles (range: [0,256])
  - Nonlinearity - how well segment's topology follows a straight line (range: [0,64])
  - Archive: 257x65 = 16,705 cells

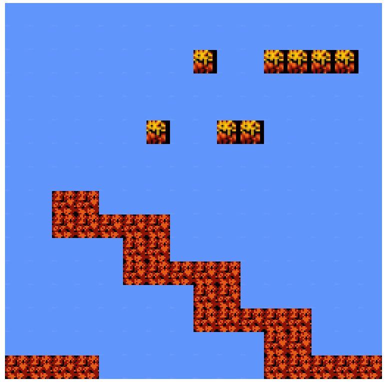
- Symmetry-Similarity (SYM-SIM)
  - Symmetry of a segment along both axes (range: [0,256])
  - Similarity of a generated segment compared to segments in training data ([0,32])
  - Archive: 257x33 = 8,481 cells



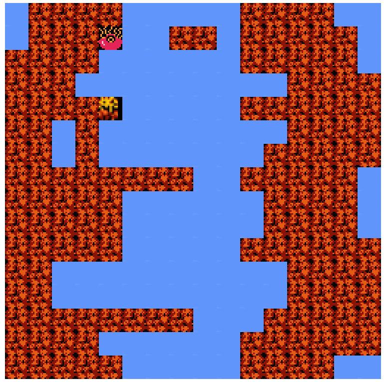
SMB
High DEN, Low NL



SMB
Low DEN, High NL



KI
Low SYM, High SIM



KI
High SYM, Low SIM

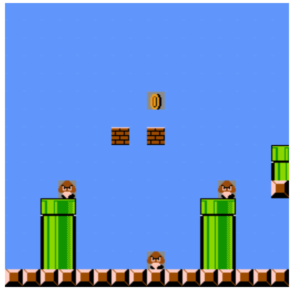
Behavior Characteristics (BCs)

- Game Elements
  - Can MAP-Elites discover segments containing different combinations of elements?

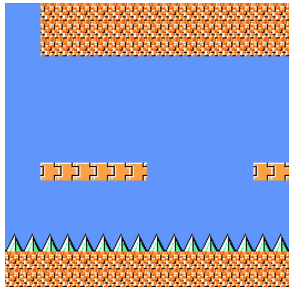
  - Used different archives for each game as type of elements differ across games

  - Each cell represented by an N-digit binary number
    - N – number of elements considered for that game
    - 0/1 indicating absence/presence of corresponding element
    - Archive: 2^N cells

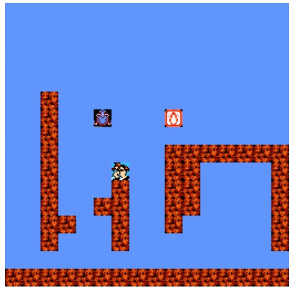
  - Values of N
    - SMB, MM, NG – 5
    - KI – 4
    - CV – 7
    - Blend-Elites - 9



SMB - <11011>
< Enemy, Pipe, ?-Mark, Coin, Breakable>



MM - <10010>
<Hazard, Doors, Ladders, Platforms, Collectables>



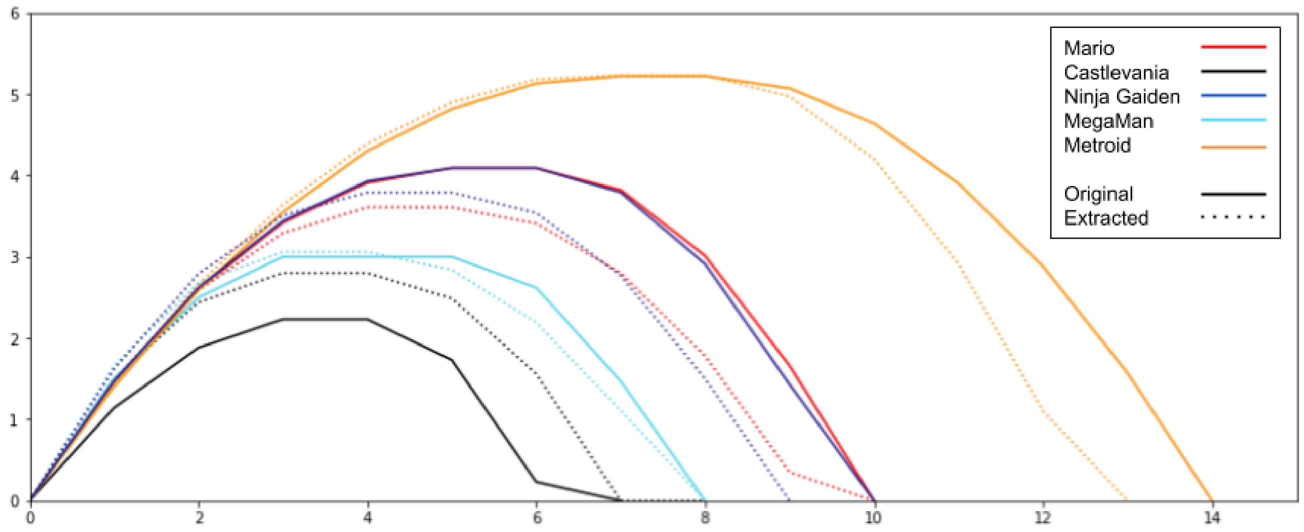
NG - <10011>
< Enemy, Animal, Ladder, Weapon, Collectable>

Fitness

- Playability as determined by game-specific A* agents tuned using jump arcs for respective games

- Fitness value: how far in the segment an agent can progress normalized from 0-1
  - SMB and CV: only horizontal progress
  - KI: only vertical direction
  - MM and NG: both horizontal and vertical directions

- Blend-Elites (SMB-KI-MM) – playability of a segment tested by running each agent and setting fitness to highest among the values



Summerville et al., 2020

VAE-MAP-Elites

Algorithm 1 VAE-ME

$archive \leftarrow$ create empty cells based on behavior characteristics
Randomly sample pop_size latent vectors
Assign each vector to cells in $archive$ (using method below)
for $i \leftarrow 1$ to $num\_generations$ **do**
  $z_1, z_2 \leftarrow$ randomly select 2 occupied cells in $archive$
  $z_{child} \leftarrow$ mutate(crossover(z_1, z_2)) with probability p
  $segment \leftarrow Decoder(z_{child})$
  $c \leftarrow GetCell(segment)$
  $score_{child} \leftarrow objective(segment)$
  if $archive_c$ is empty **then**
    Add $z_{child}, score_{child}$ to $archive_c$
  else
    $z_c, score_c \leftarrow archive_c$
    if $score_{child} > score_c$ **then**
      Replace $z_c, score_c$ with $z_{child}, score_{child}$ in $archive_c$
    end if
  end if
end for

Experiments

- 6 domains x 3 BCs → 18 separate experiments
- For each experiment
  - 100,000 generations
  - Mutation probability of 0.3
  - VAE latent size of 32

Experiments

- 6 domains x 3 BCs → 18 separate experiments

- For each experiment
  - 100,000 generations
  - Mutation probability of 0.3
  - VAE latent size of 32

- Evaluation
  - Test for playability, compare behavior spaces
  - QD-Score – sum of fitness values for all occupied cells in the archive
  - Coverage – percentage of archive cells that are occupied at the end of the run
  - Optimality – percentage of occupied archive cells with optimal fitness value

Experiments

- 6 domains x 3 BCs → 18 separate experiments

- For each experiment
  - 100,000 generations
  - Mutation probability of 0.3
  - VAE latent size of 32

- Evaluation
  - Test for playability, compare behavior spaces
  - QD-Score – sum of fitness values for all occupied cells in the archive
  - Coverage – percentage of archive cells that are occupied at the end of the run
  - Optimality – percentage of occupied archive cells with optimal fitness value

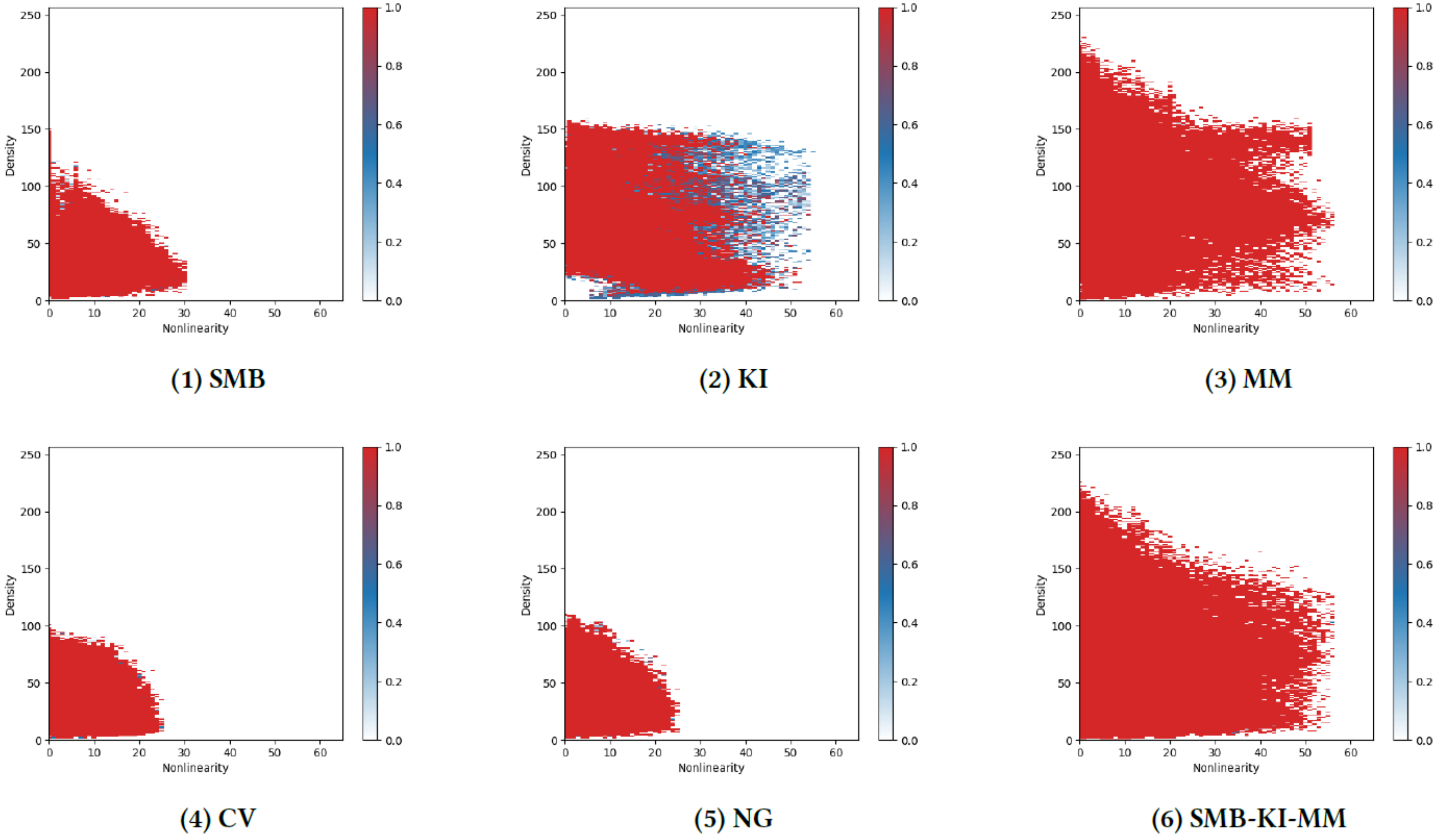
  - Blend-Elites (SMB+KI+MM)
    - Identify regions of archive where certain agents and/or combinations of agents do better → may suggest certain regions blend certain games
    - For each cell, track which agents completed a segment assigned to that cell

Results

	Density-Nonlinearity			Symmetry-Similarity			Game-Elements		
	QD-Score	Coverage	% Optimal	QD-Score	Coverage	% Optimal	QD-Score	Coverage	% Optimal
SMB	2341.81	14.03	97.48	2726.00	32.14	98.97	32.00	100.00	100
KI	5631.81	41.91	67.4	2660.31	32.48	94.22	16.00	100.00	100
MM	7245.94	48.29	89.26	5239.06	62.9	97.75	30	93.75	100
CV	1849.38	11.24	97.97	1353.63	16.17	97.81	104.00	81.25	100
NG	1955.94	11.92	97.69	1237.13	14.8	98.41	32.00	100.00	100
Blend-Elites	8267.31	49.64	99.66	5262	62.22	99.72	455.00	88.87	100

- QD-Score and Coverage for both Density-Nonlinearity and Symmetry-Similarity
  - Blend-Elites, MM, KI > SMB > CV and NG
- QD-Score and Coverage for Game Elements
  - SMB, KI and NG > MM and Blend-Elites > CV
- In most cases, if a solution was found for a cell, then it was also optimal

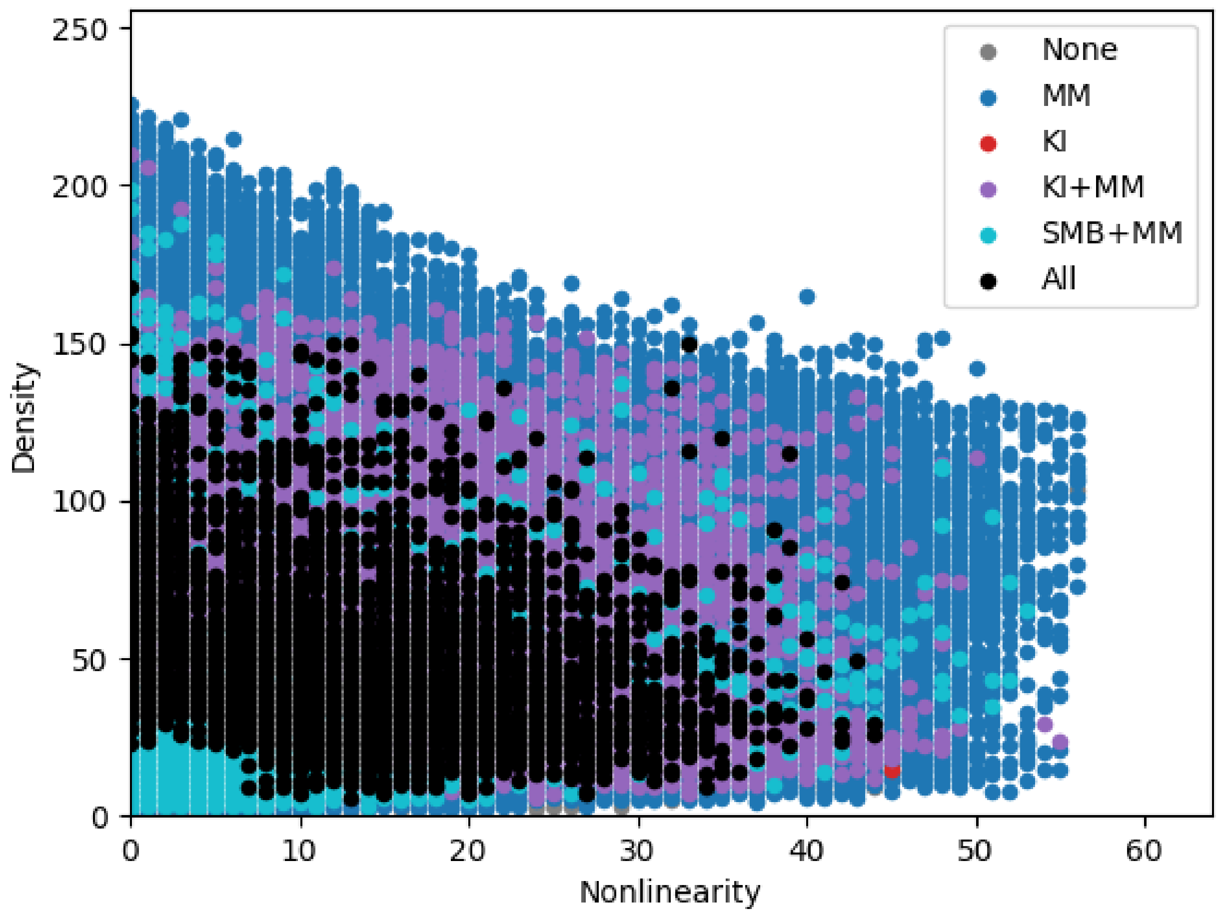
Results

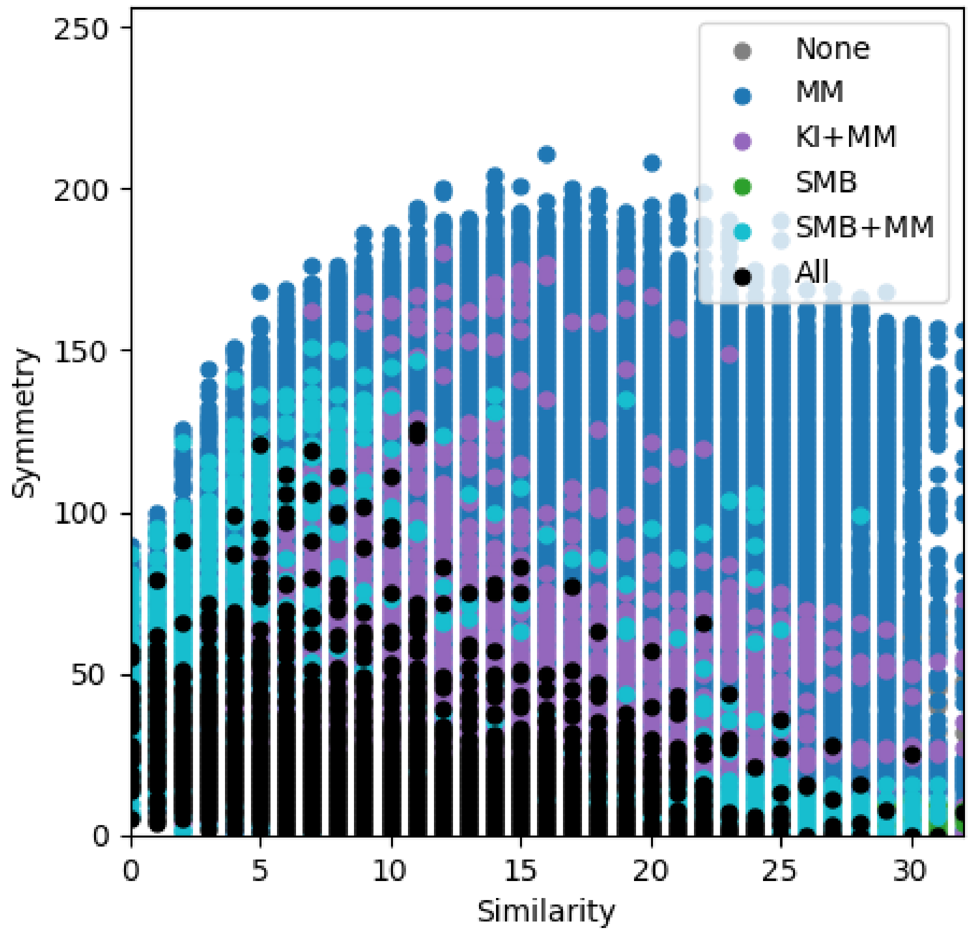


Archives for Density-Linearity

- Lower coverage for SMB, CV and NG than MM, KI and Blend
  --- SMB, CV, NG levels tend to be less dense and more open
- More capable agent, higher playability
  --- MM can move in both directions
  --- KI only upward movement
- Blend-Elites archive roughly intersects the regions covered by the 3 games individually

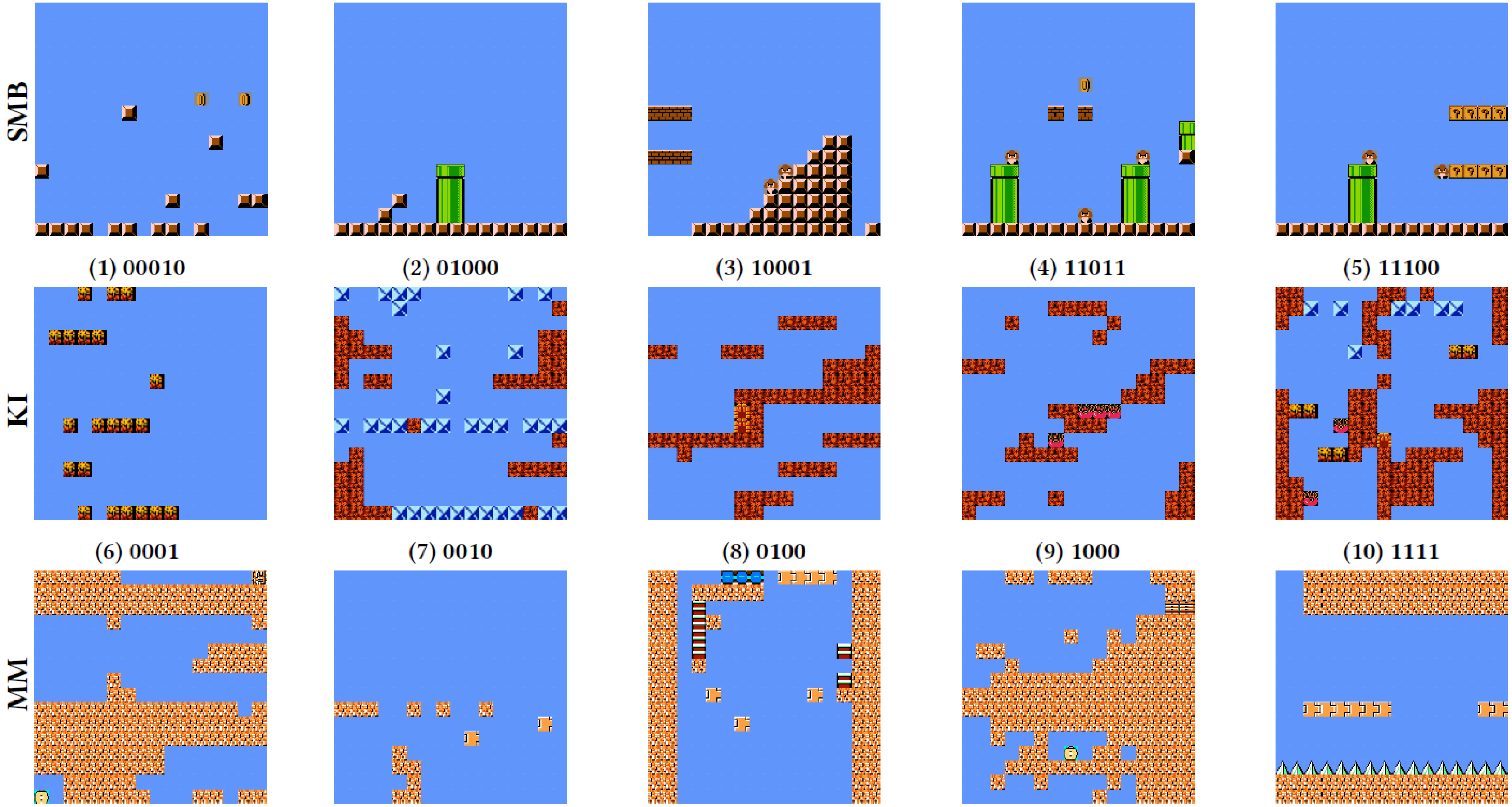
Results





Archive of tile-based BCs for Blend-Elites with each cell colored based on the agents that completed a segment assigned to that cell.

Results

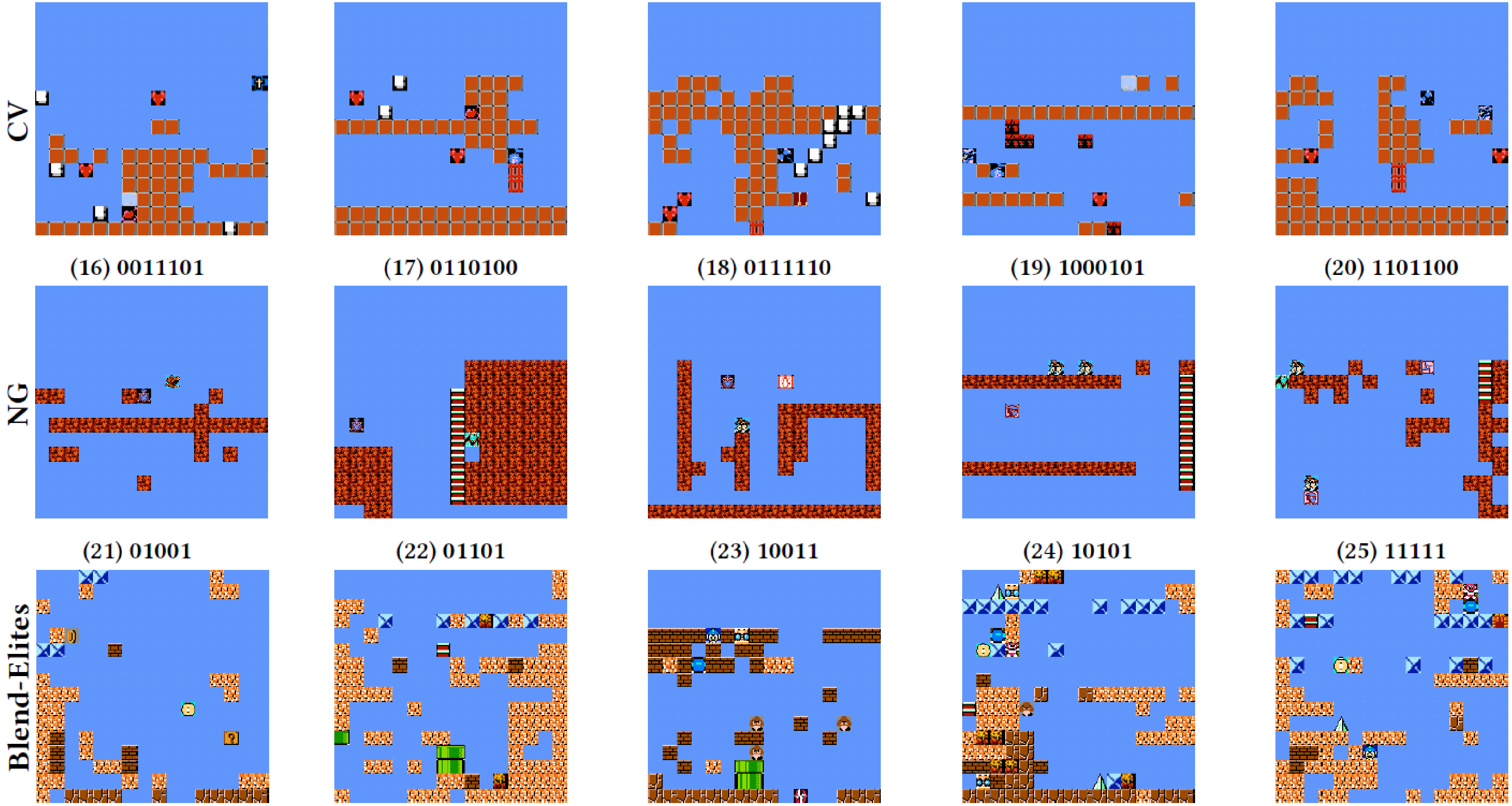


(1) 00010 (2) 01000 (3) 10001 (4) 11011 (5) 11100

(6) 0001 (7) 0010 (8) 0100 (9) 1000 (10) 1111

(11) 00001 (12) 00010 (13) 00111 (14) 01001 (15) 10010

Results



(16) 0011101 (17) 0110100 (18) 0111110 (19) 1000101 (20) 1101100

(21) 01001 (22) 01101 (23) 10011 (24) 10101 (25) 11111

(26) 000011101 (27) 001100111 (28) 100101011 (29) 101001111 (30) 111001101

Conclusion

- Combined VAEs with MAP-Elites for generating and blending game levels
- Generated and blended diverse range of playable levels
- Identified regions that blend specific combinations of games

Future Work

- Study other QD algorithms when combined with VAEs
- Variations of MAP-Elites + advanced VAE models
- User studies and playtests to study perception of diversity of generated levels
- Incorporate MAP-Elites into ML-based co-creative and automated design tools

Future Work

- Study other QD algorithms when combined with VAEs
- Variations of MAP-Elites + advanced VAE models
- User studies and playtests to study perception of diversity of generated levels
- Incorporate MAP-Elites into ML-based co-creative and automated design tools

Contact

Anurag Sarkar
Northeastern University
sarkar.an@northeastern.edu