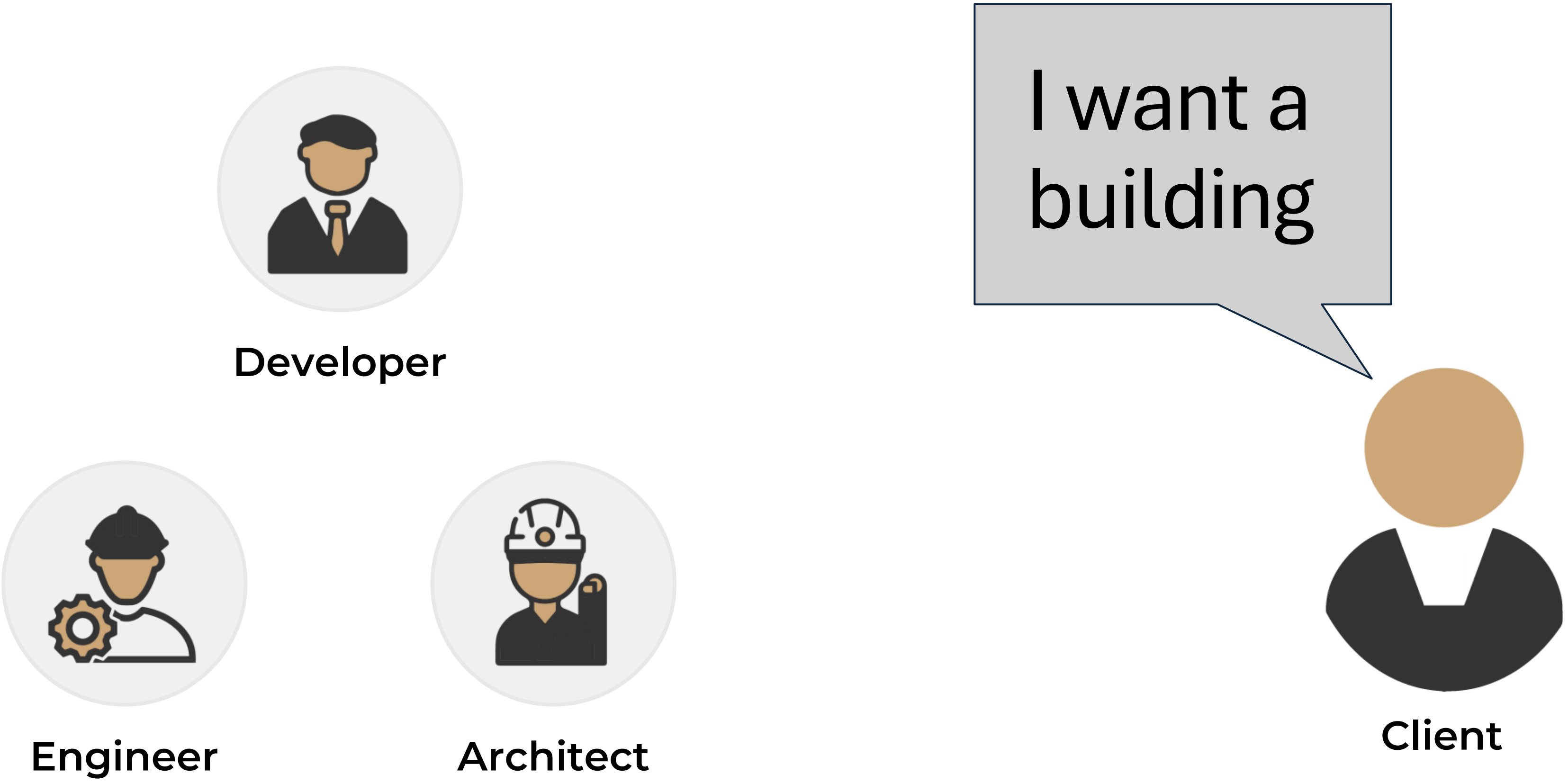
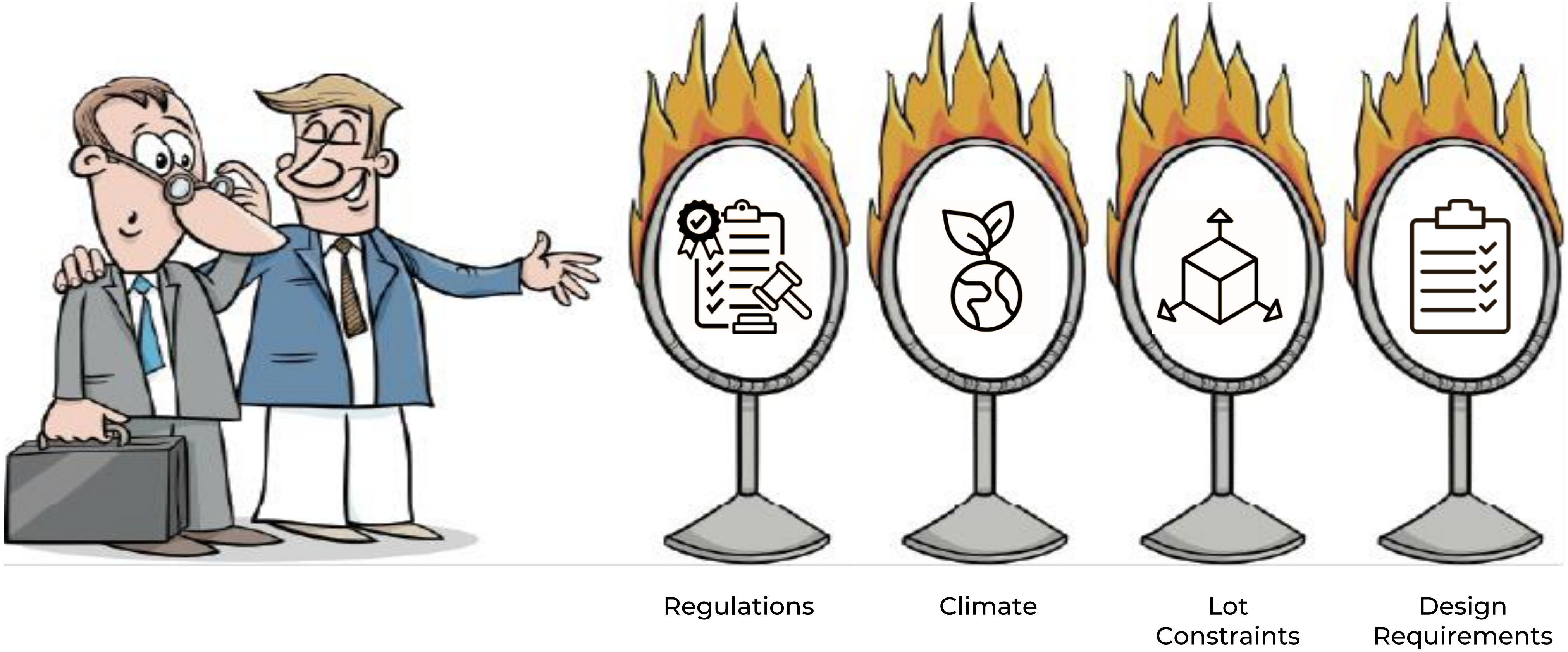
Deep Generative Design

Deep reinforcement learning for performance-based design assistance

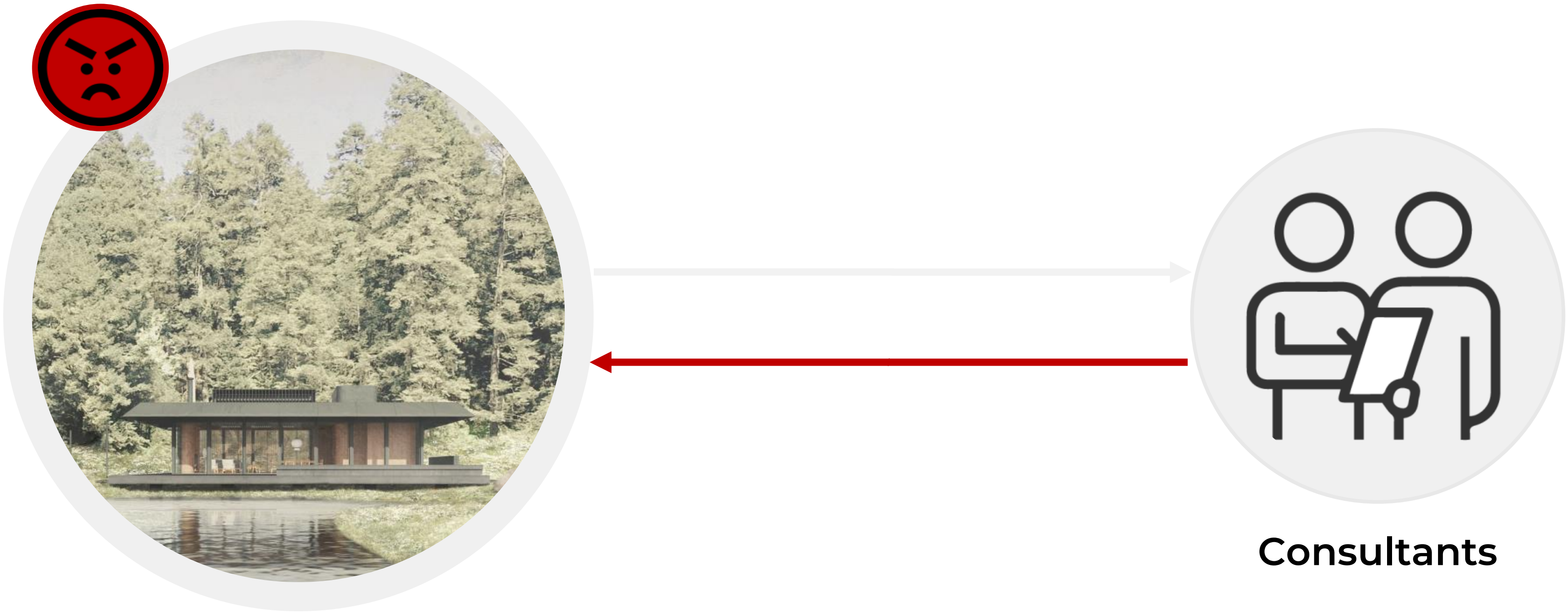
by: Jair Lemmens 4645448


Developer
Engineer
Architect
I want a building
Client



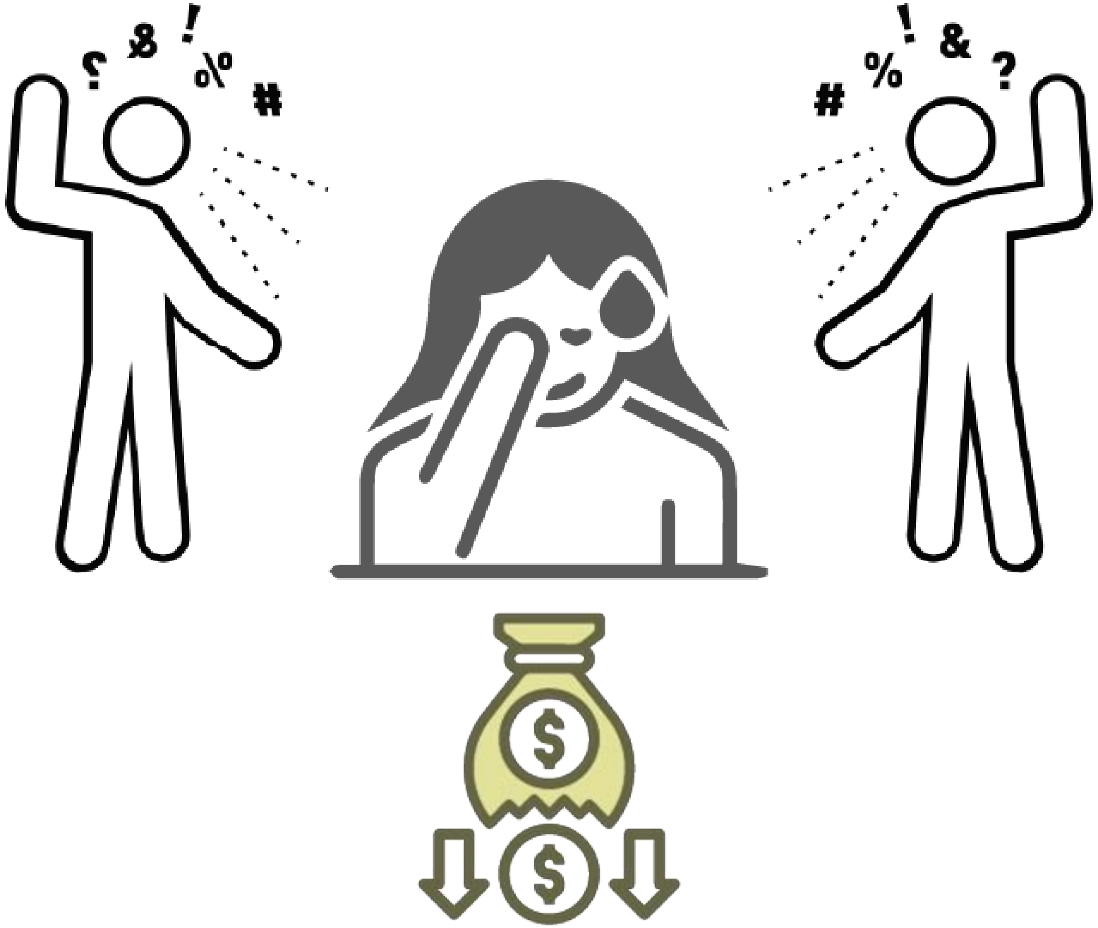
Regulations
Climate
Lot
Constraints
Design
Requirements


Consultants


Consultants

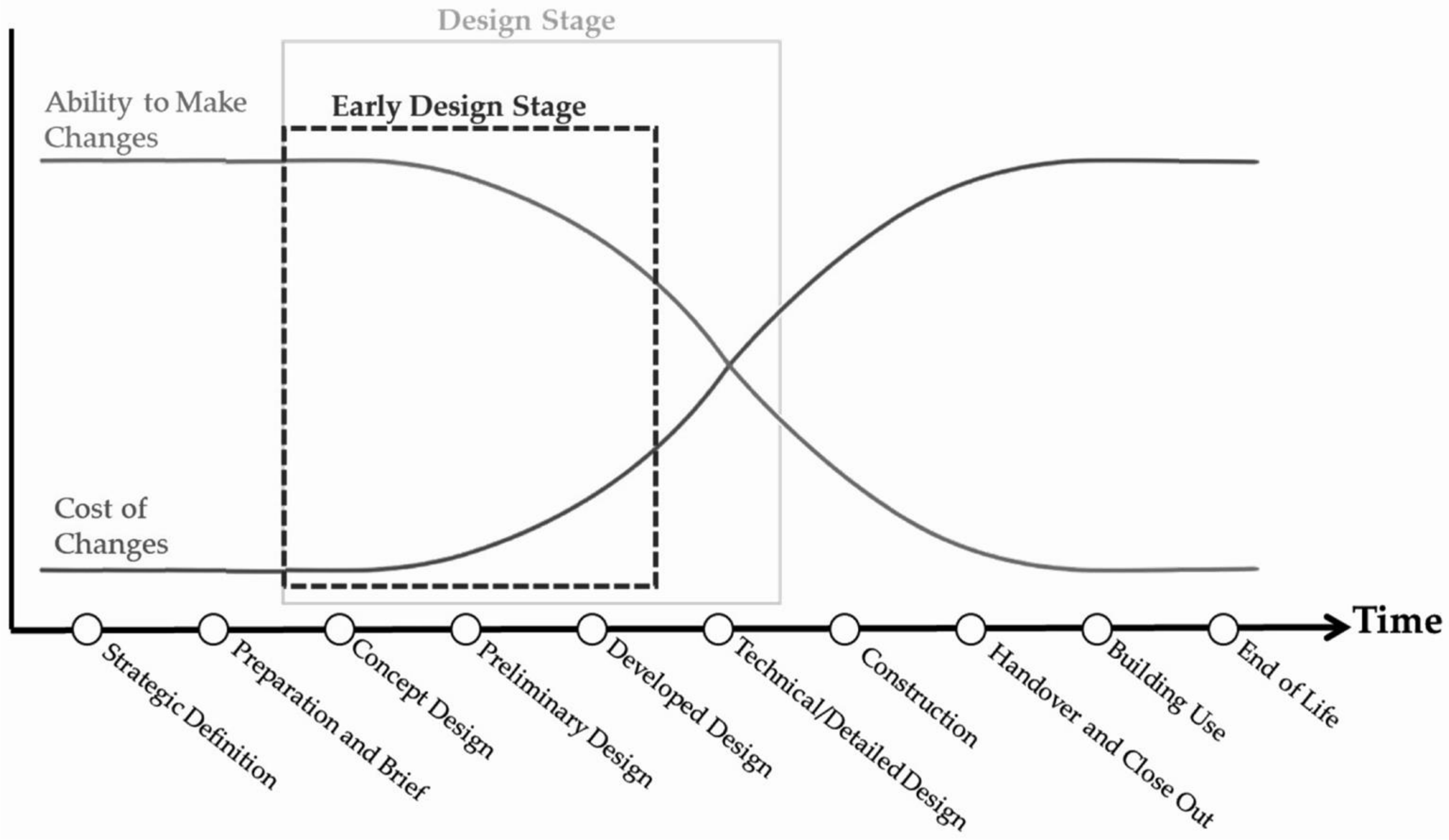
Where are the permits?

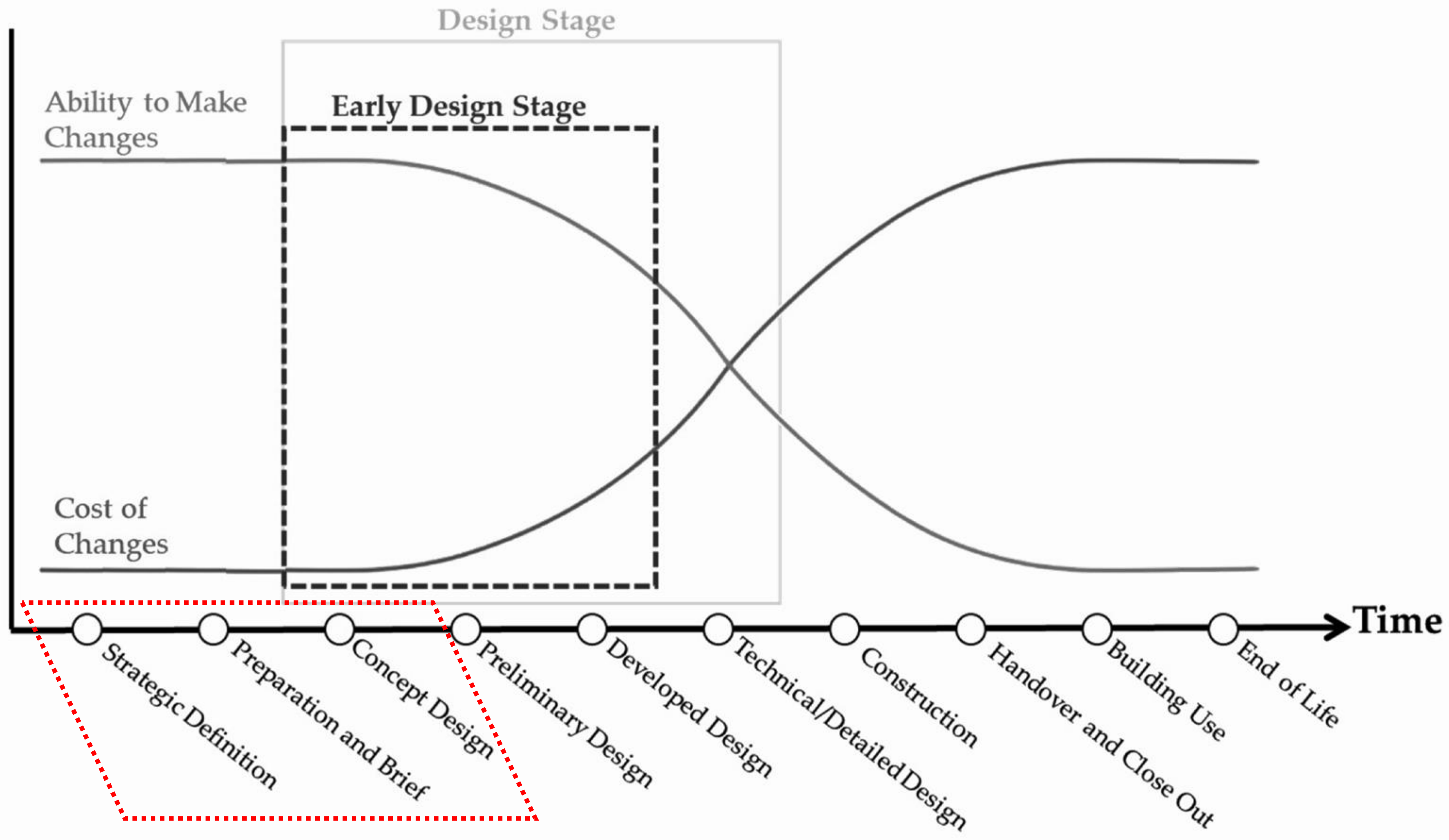
You owe me money!

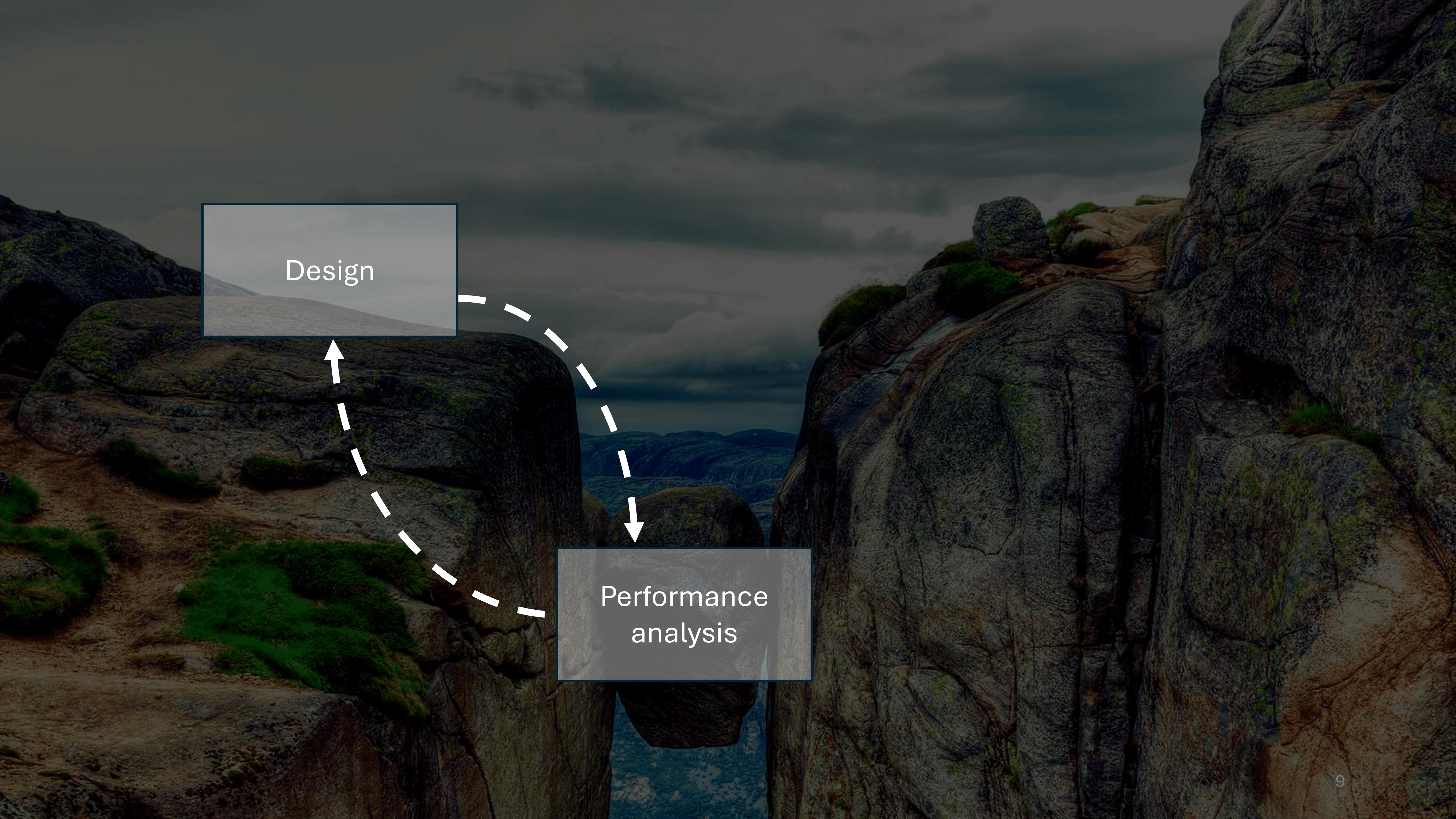


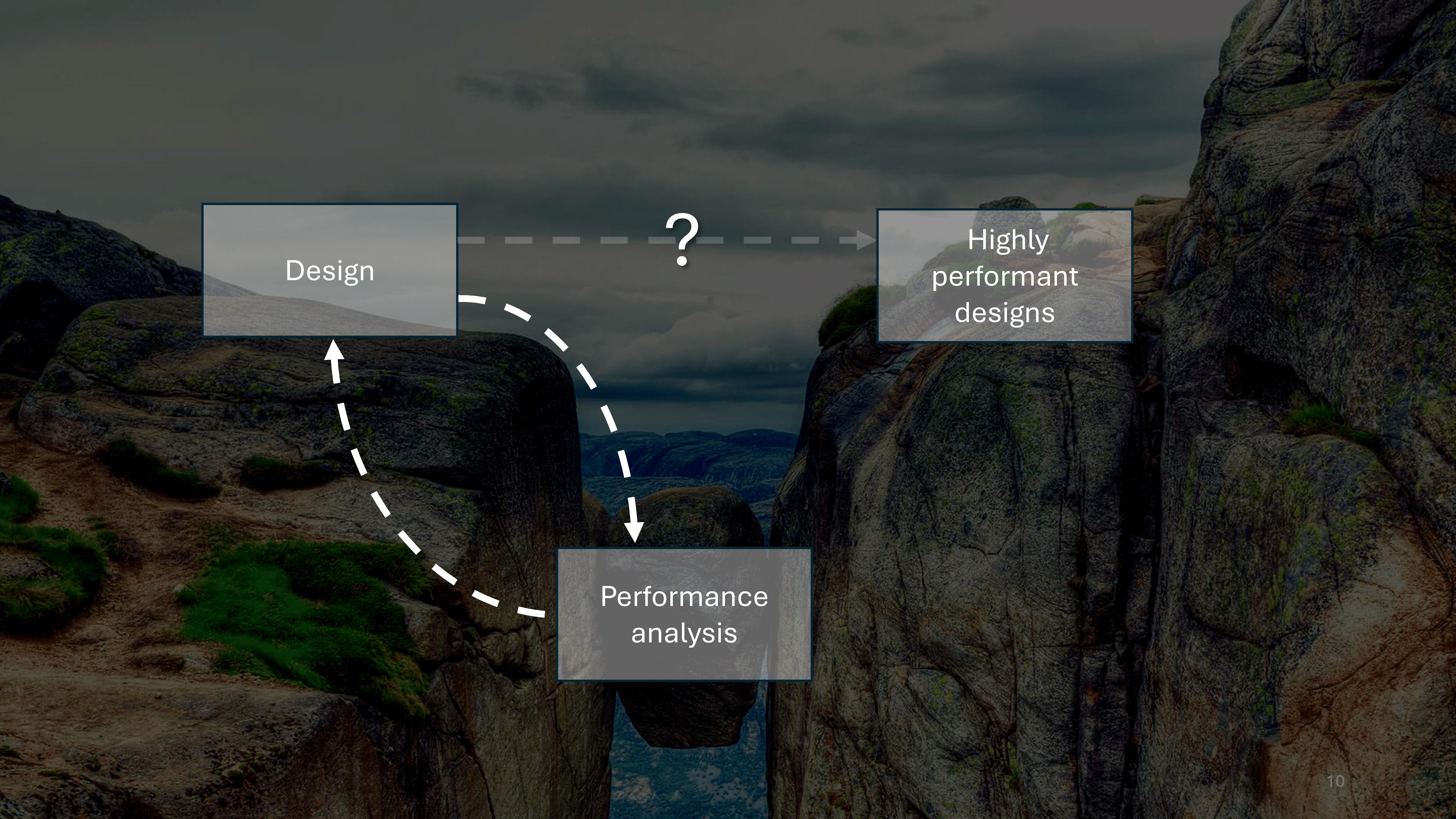
~~Achievements~~ so far:

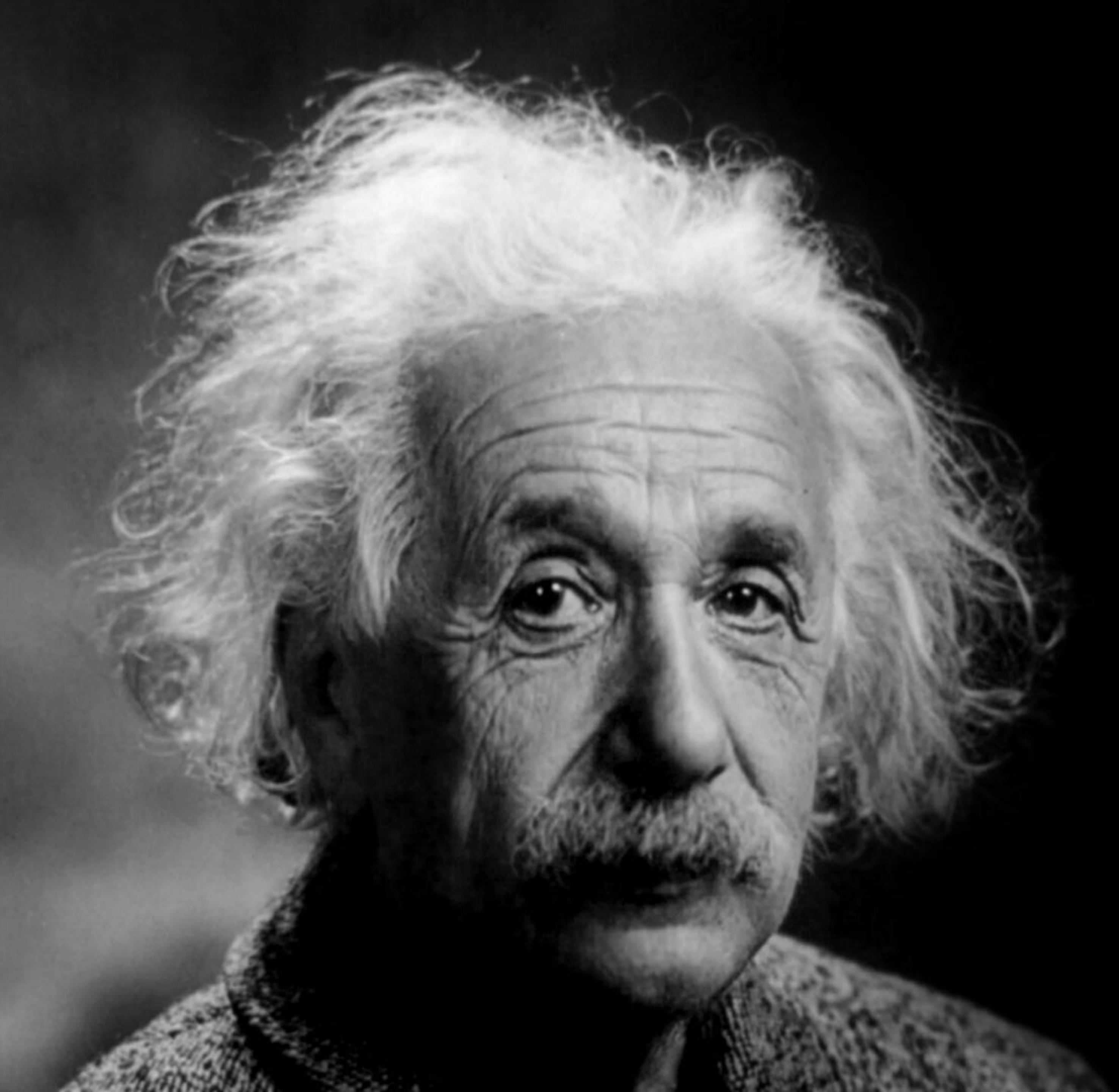
- No permits
- Vacant lot
- Employee salaries
- Consultation fees


Design Stage
Early Design Stage
Ability to Make Changes
Cost of Changes
Time
Strategic Definition
Preparation and Brief
Concept Design
Preliminary Design
Developed Design
Technical/Detailed Design
Construction
Handover and Close Out
Building Use
End of Life


Design Stage
Early Design Stage
Ability to Make Changes
Cost of Changes
Time
Strategic Definition
Preparation and Brief
Concept Design
Preliminary Design
Developed Design
Technical/Detailed Design
Construction
Handover and Close Out
Building Use
End of Life


Design
Performance analysis

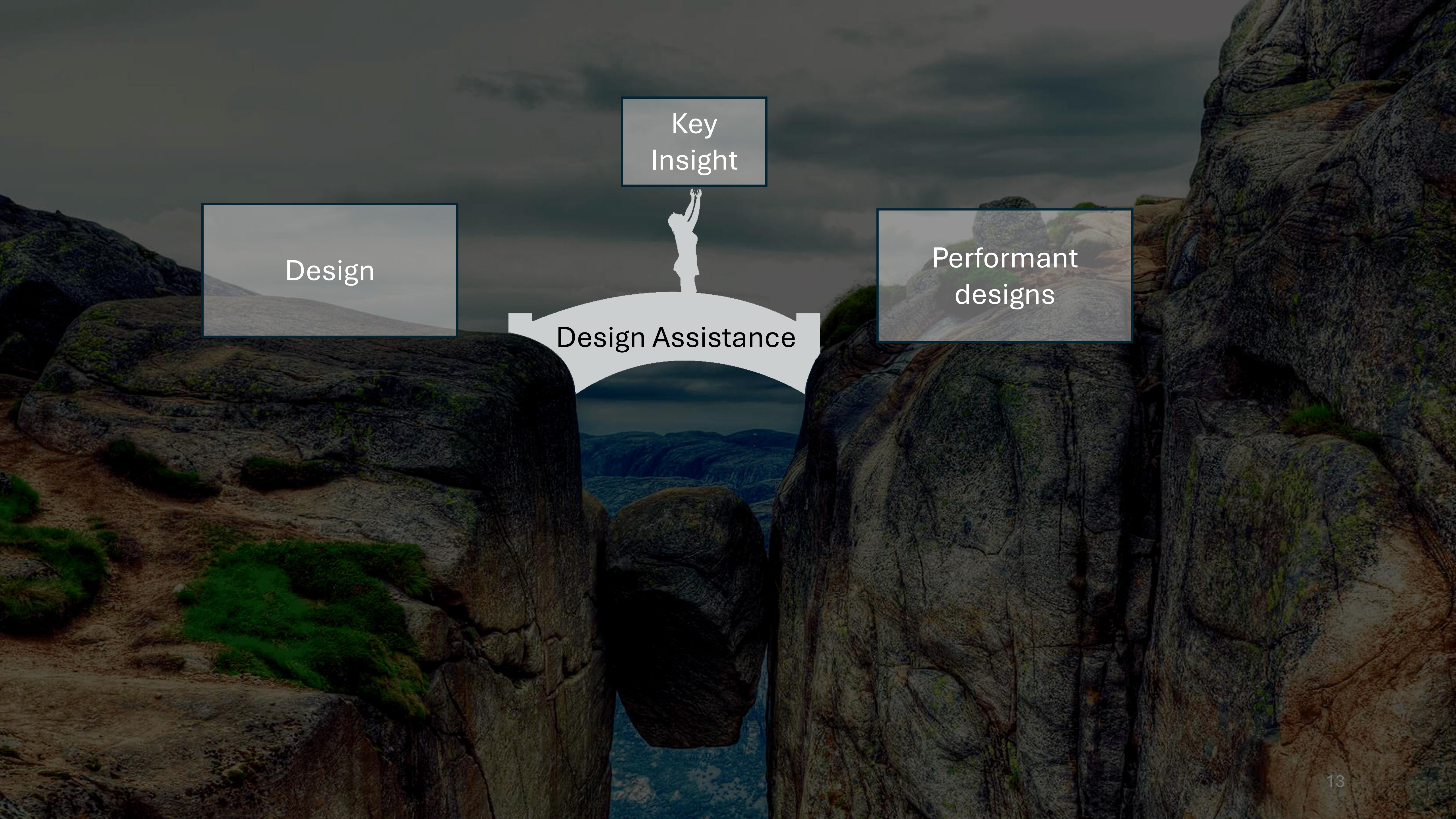

Design
?
Highly performant designs
Performance analysis



The rational mind is
a faithful servant

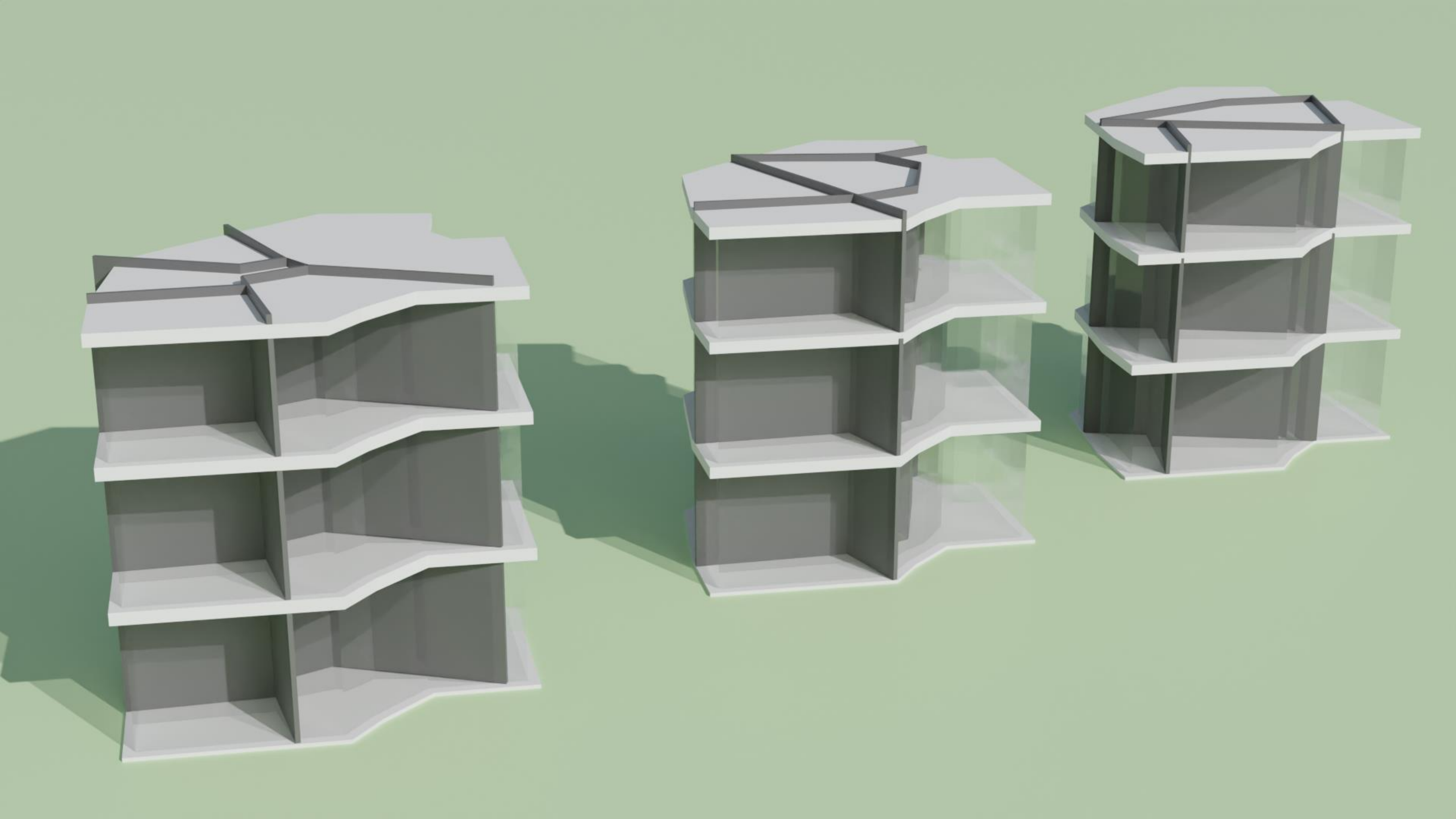
Albert Einstein


We can learn from example

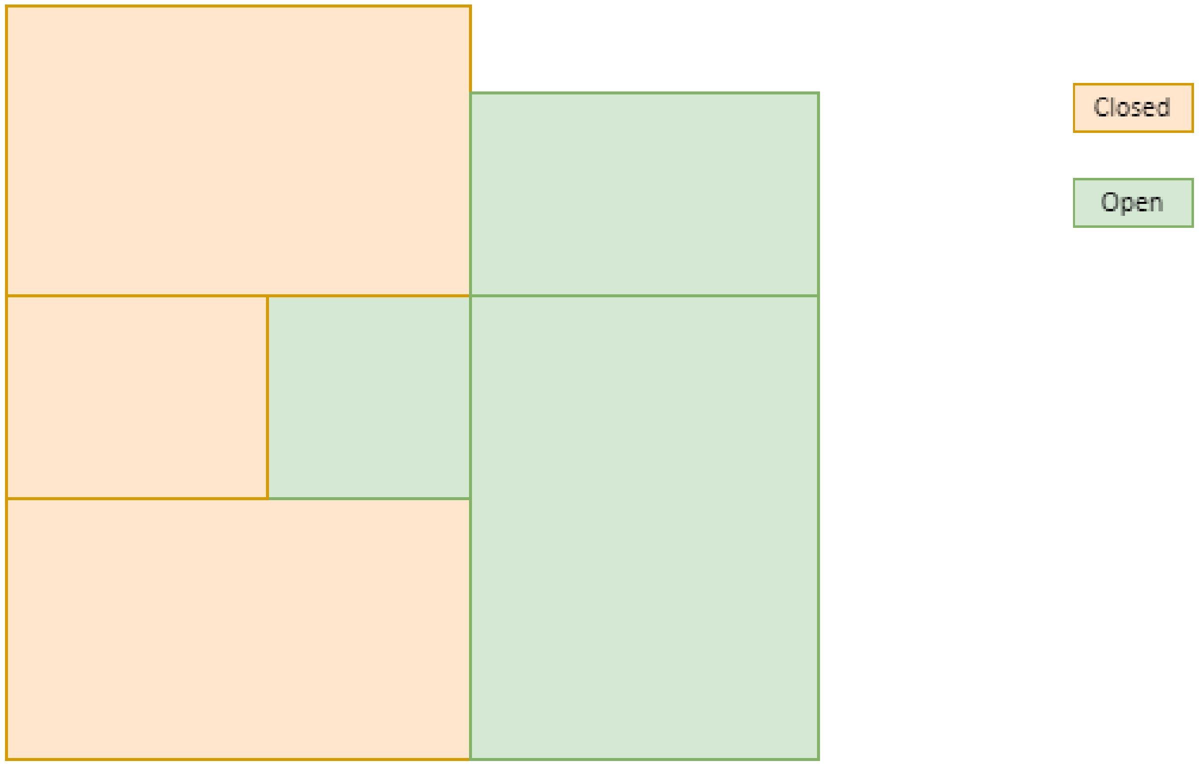

Key
Insight
Design
Design Assistance
Performant
designs

How can deep learning be used to assist in creating performance-informed floor plans?

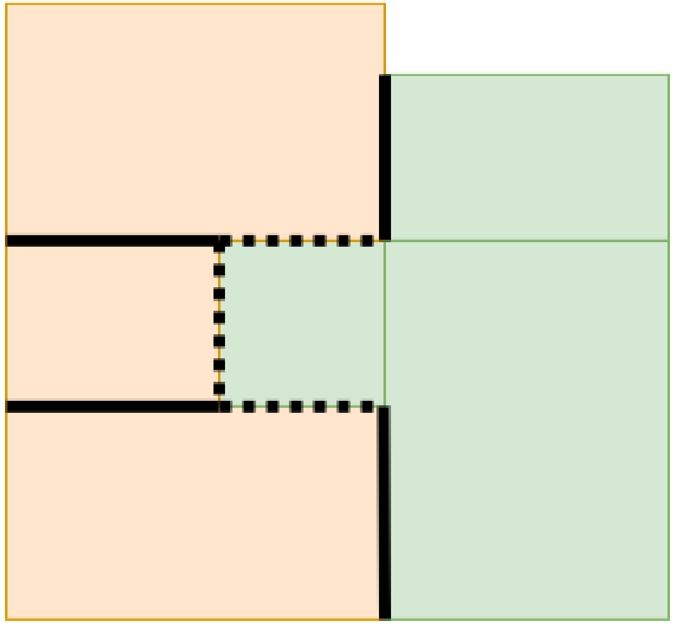
How can a collaborative, performance aware deep learning system overcome data scarcity and the creativity gap?

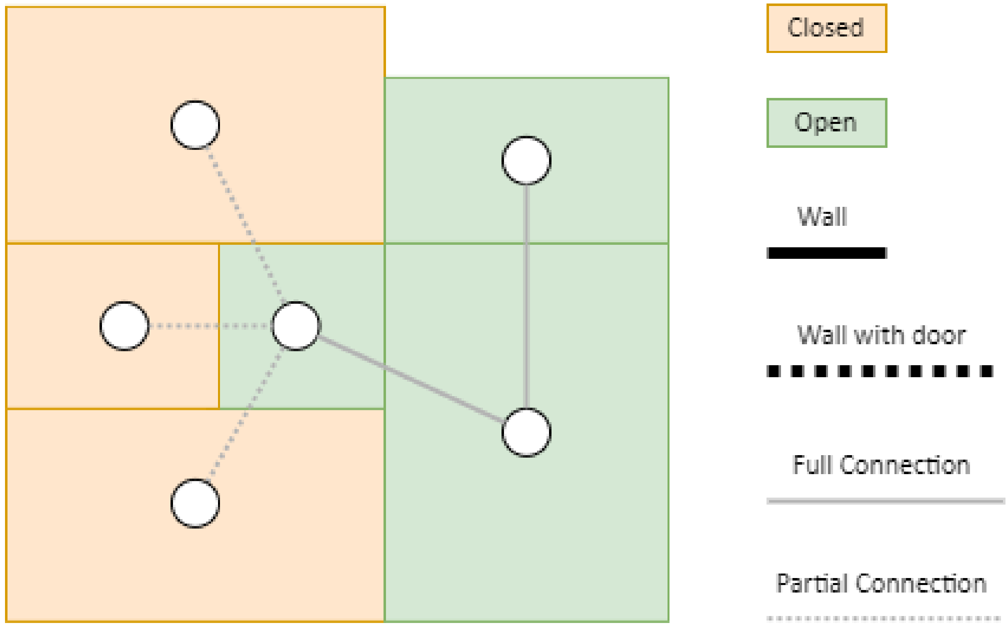


What are the heuristics of floorplan design?

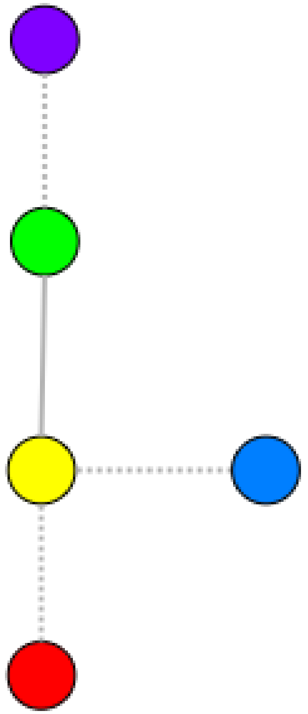


What can we extract from these plans?



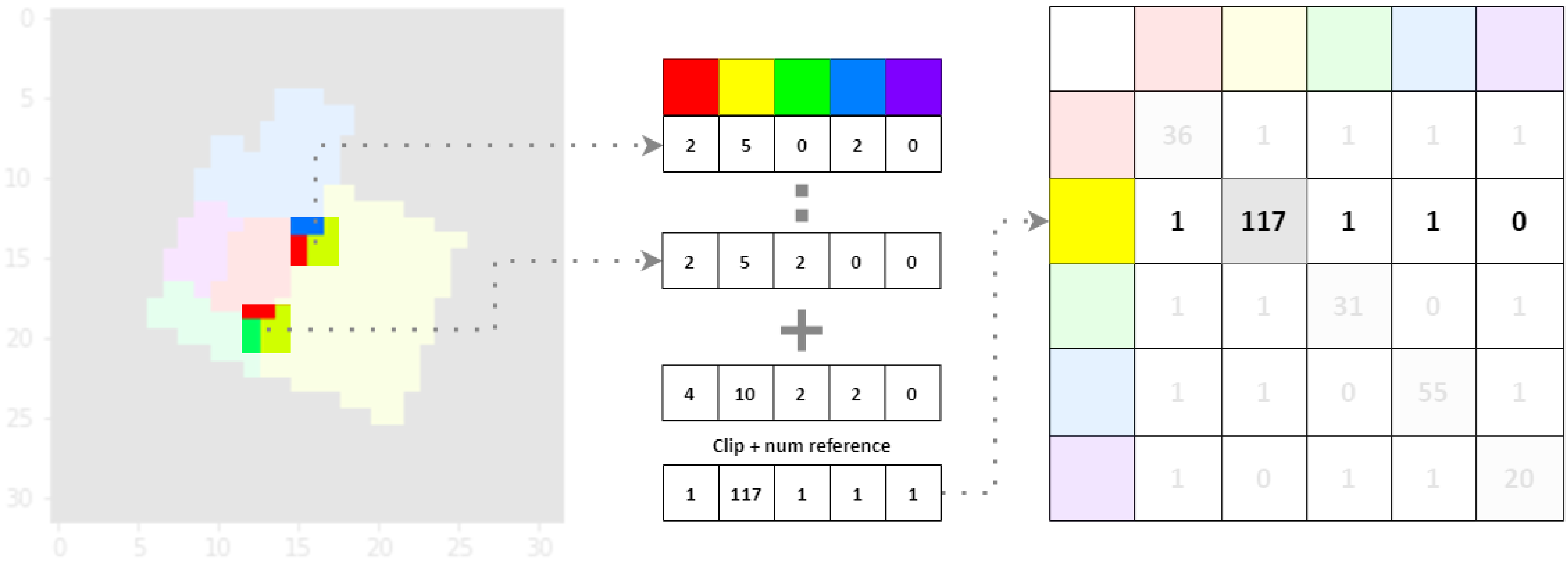


Representing the graphs

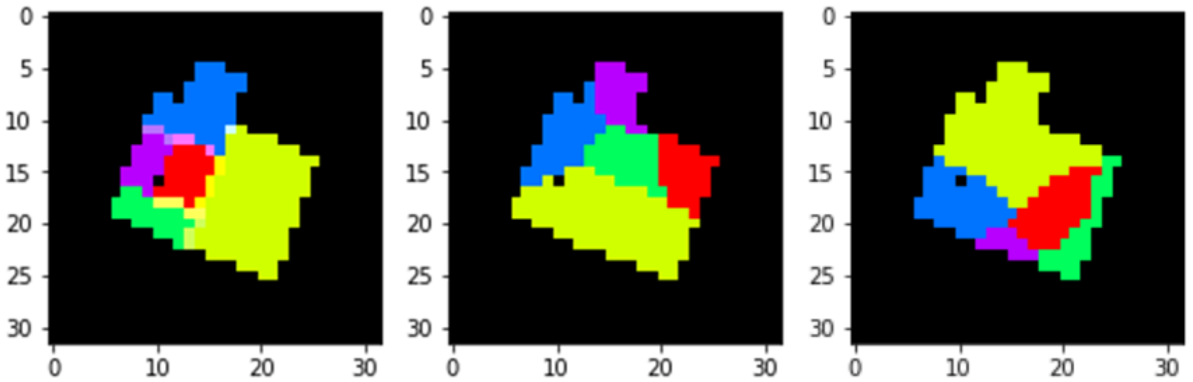


	1	1	0	0	0
	1	1	1	1	0
	0	1	1	0	1
	0	1	0	1	0
	0	0	1	0	1

Graph extraction



Assigning composition score



	36	1	1	1	1
	1	117	1	1	0
	1	1	31	0	1
	1	1	0	55	1
	1	0	1	1	20

Reference

	36	1	1	0	0
	1	117	1	1	0
	1	1	31	1	1
	0	1	1	55	1
	0	0	1	1	20

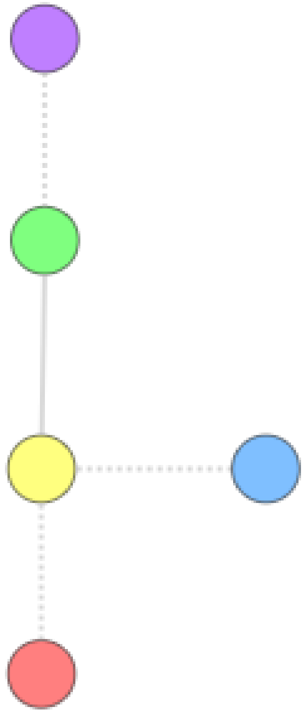
70%

	36	1	1	1	1
	1	117	1	1	0
	1	1	31	0	1
	1	1	0	55	1
	1	0	1	1	20

100%

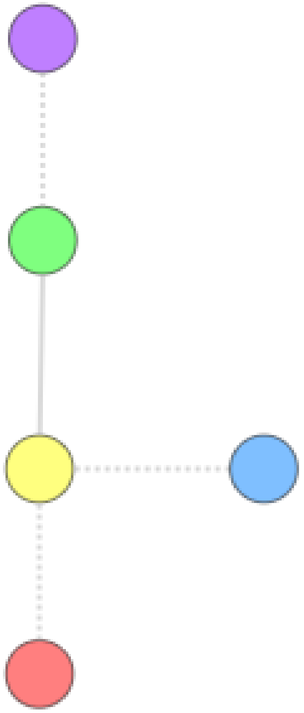
$$score_{graph} = \mathrm{mean}(\mathrm{clip}((\mathrm{graph}_{reference} - graph_{design})^2, 0,1))$$

Adding areas



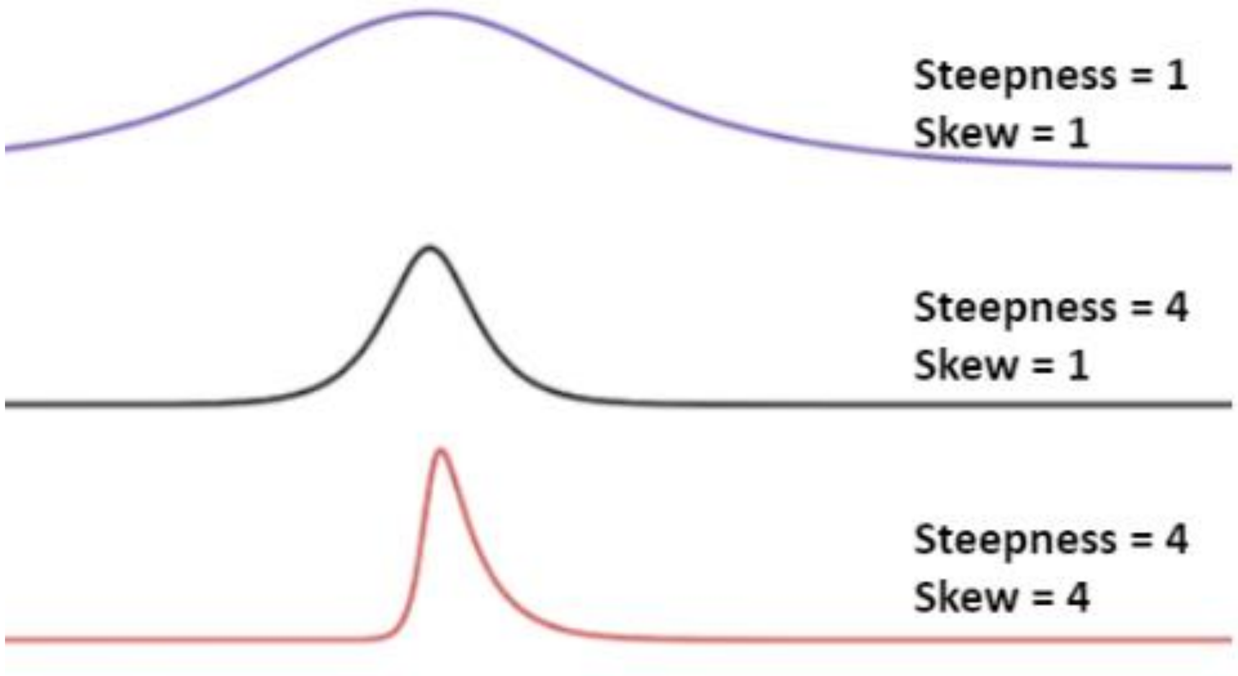
	1	1	0	0	0
	1	1	1	1	0
	0	1	1	0	1
	0	1	0	1	0
	0	0	1	0	1

Adding areas



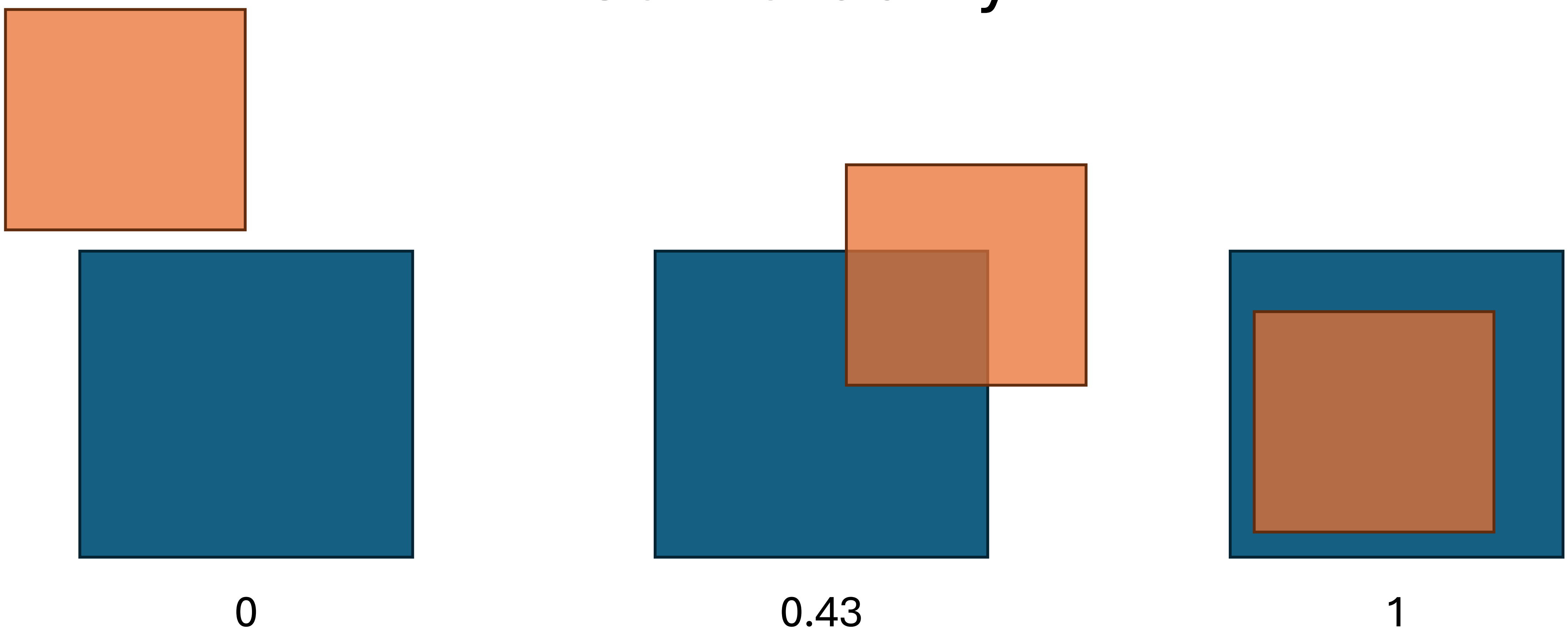
	5	1	0	0	0
	1	**12**	1	1	0
	0	1	**2**	0	1
	0	1	0	**30**	0
	0	0	1	0	**20**

Assigning area score



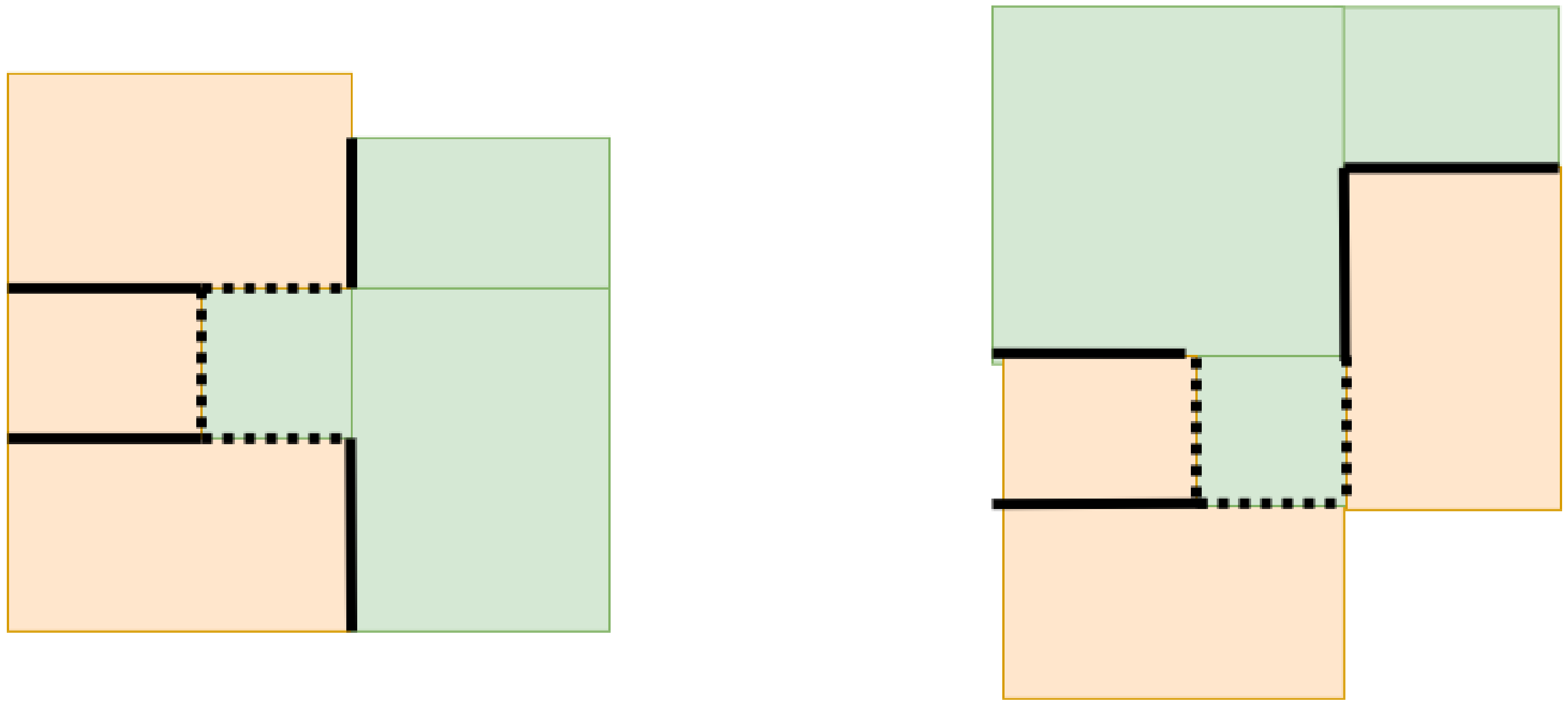
$$score_{area} = \frac{2}{e^{steepness*error} + e^{-skew*steepness*error}}$$

Controllability

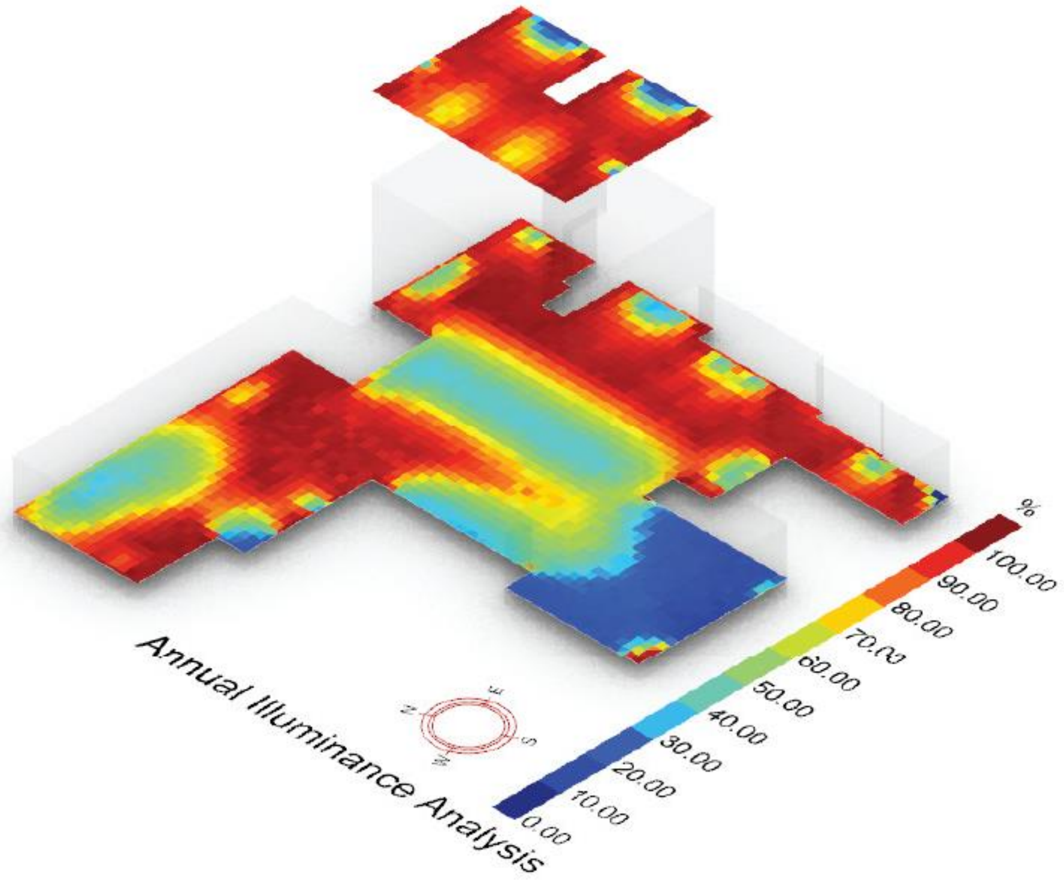


$$score_{similarity} = \frac{Drawing_{architect} \cap Drawing_{machine}}{Drawing_{architect}}$$

Which floorplan is better?

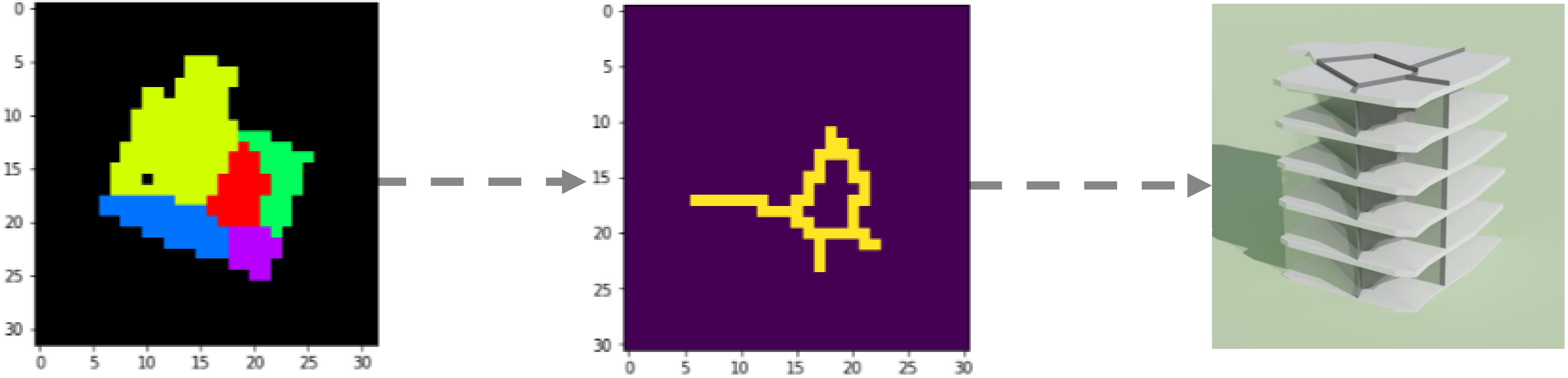


They will be differentiated by daylight satisfaction

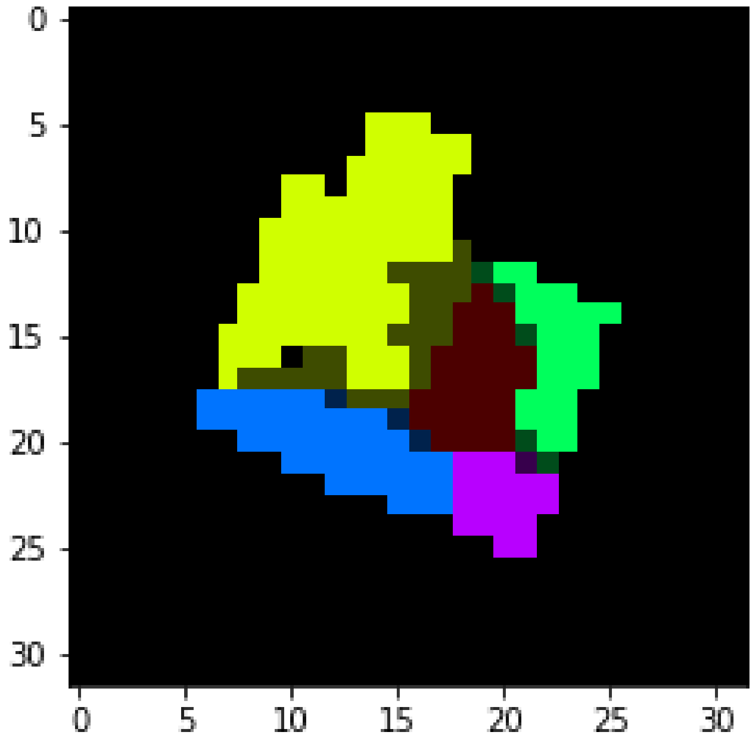


Prazmowski, Juan

A 3D model is needed

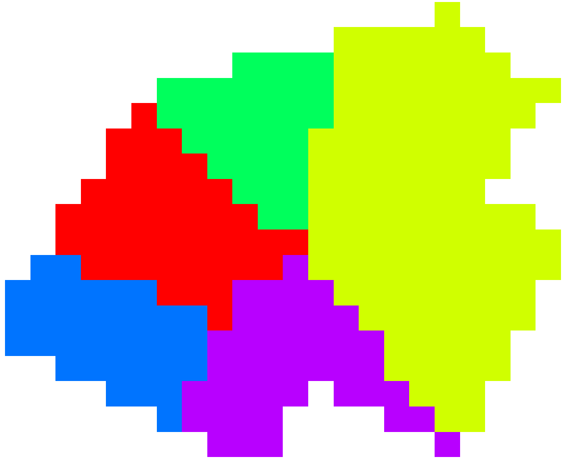


Check daylight for Breeam compliance

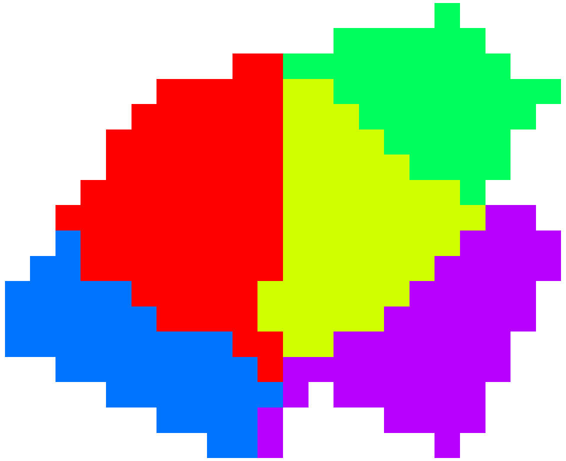


$$score_{daylight} = clip\left(\frac{sum\left(where\left(\frac{light_{pixel}}{light_{exterior}} \geq 0.03, 1, 0\right)\right)}{fraction\_required * total_{pixels}}, 0,1\right)$$

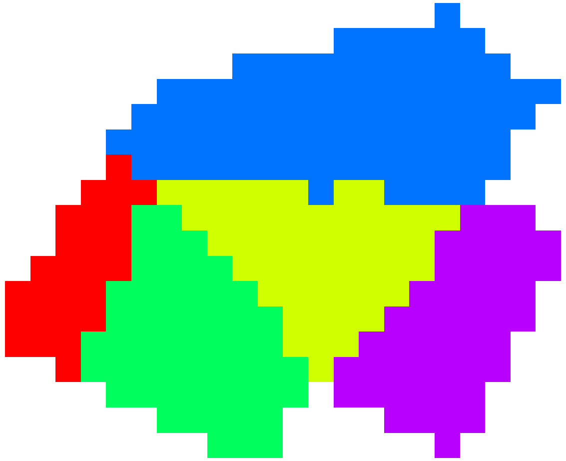
We can now assign scores to these variants



0.69

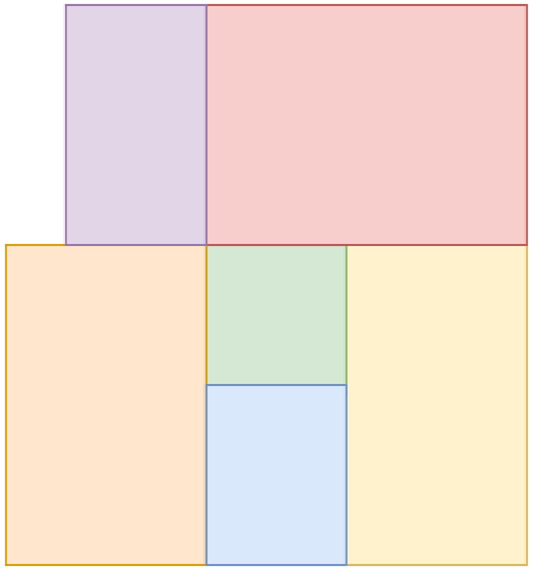


0.58



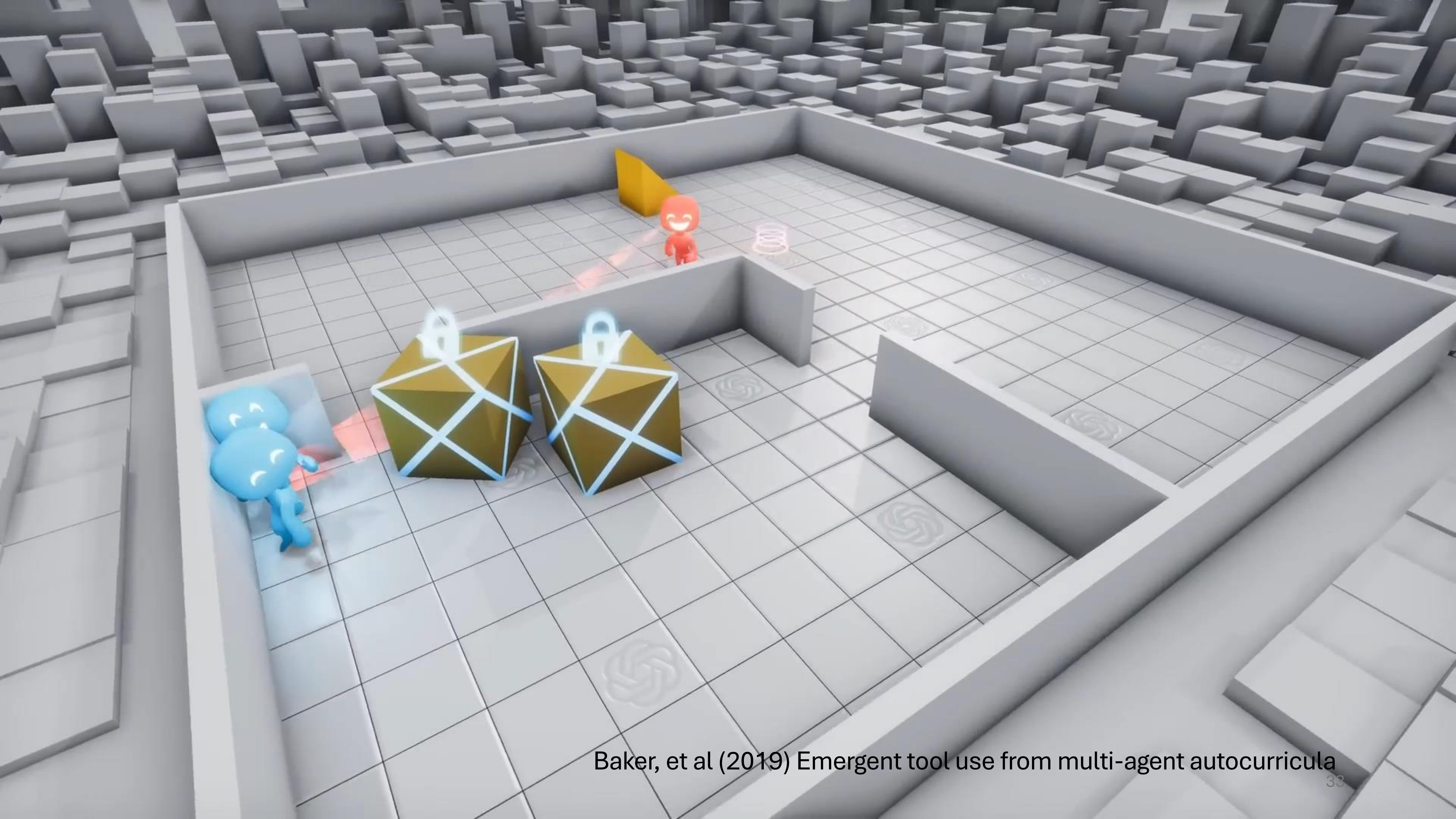
0.65

How can the machine create a symbolic floorplan?



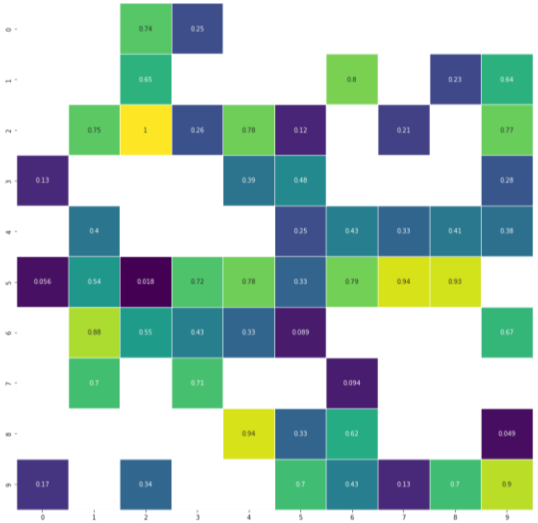
Using reinforcement learning

Mellor, et al (2019) unsupervised doodling and Painting with improved spiral

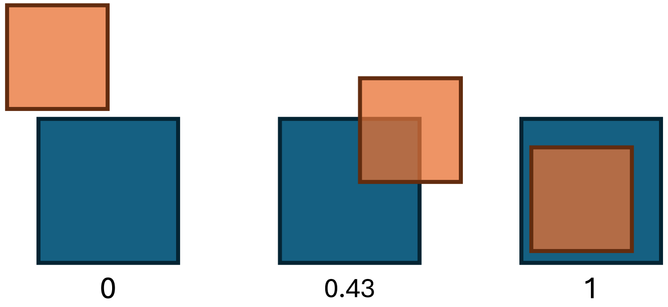


Baker, et al (2019) Emergent tool use from multi-agent autocurricula

Why?



Data scarcity

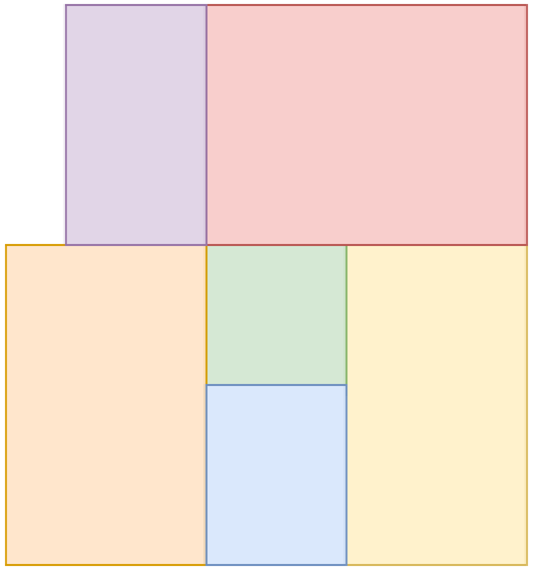


Controllability

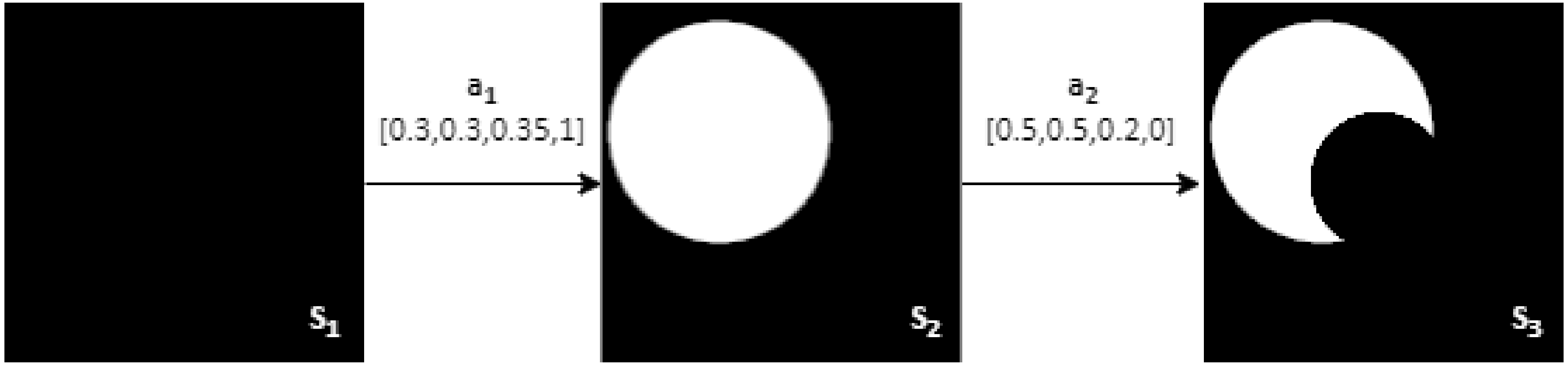
$$x^2 \frac{dy}{dx} + y^2 = xy$$

Differentiability

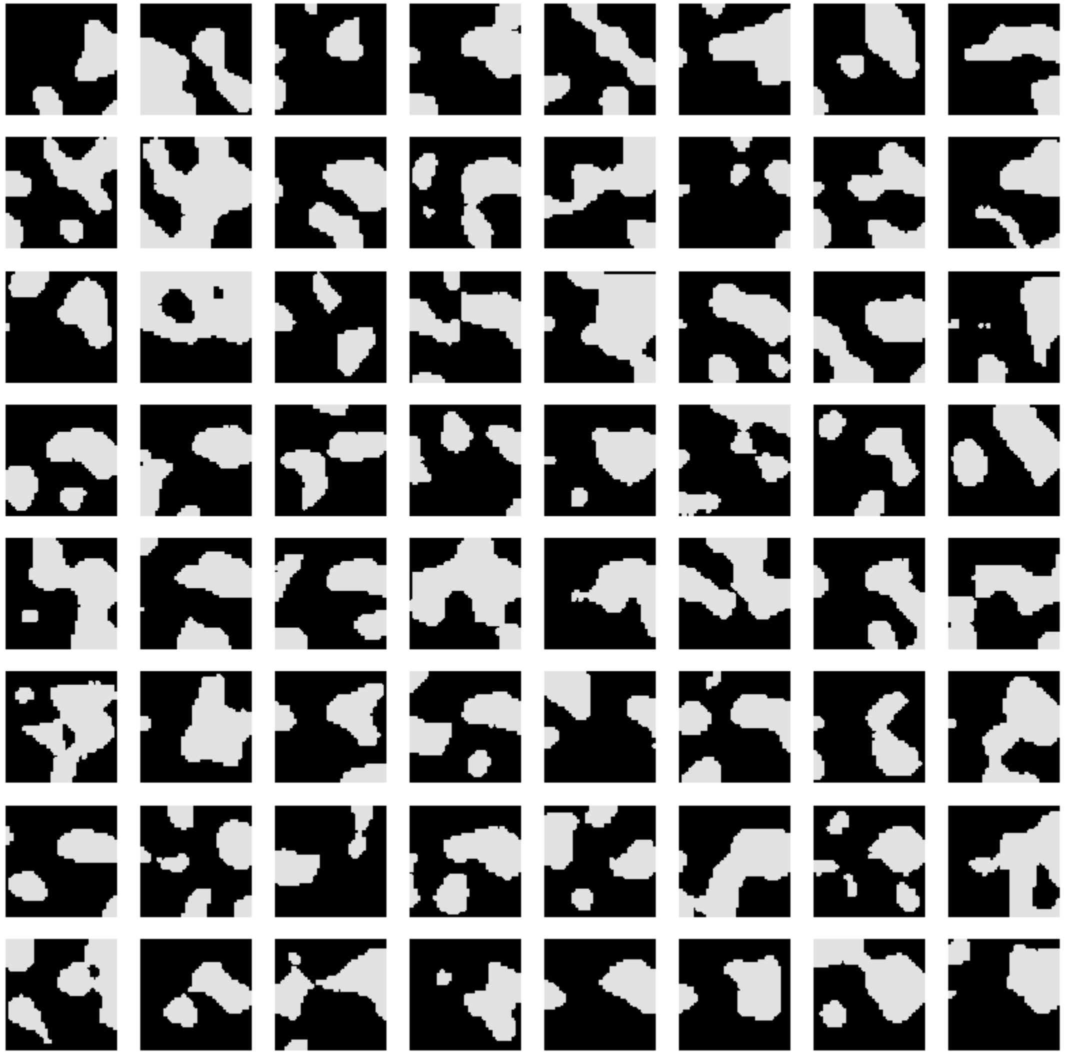
One agent per space



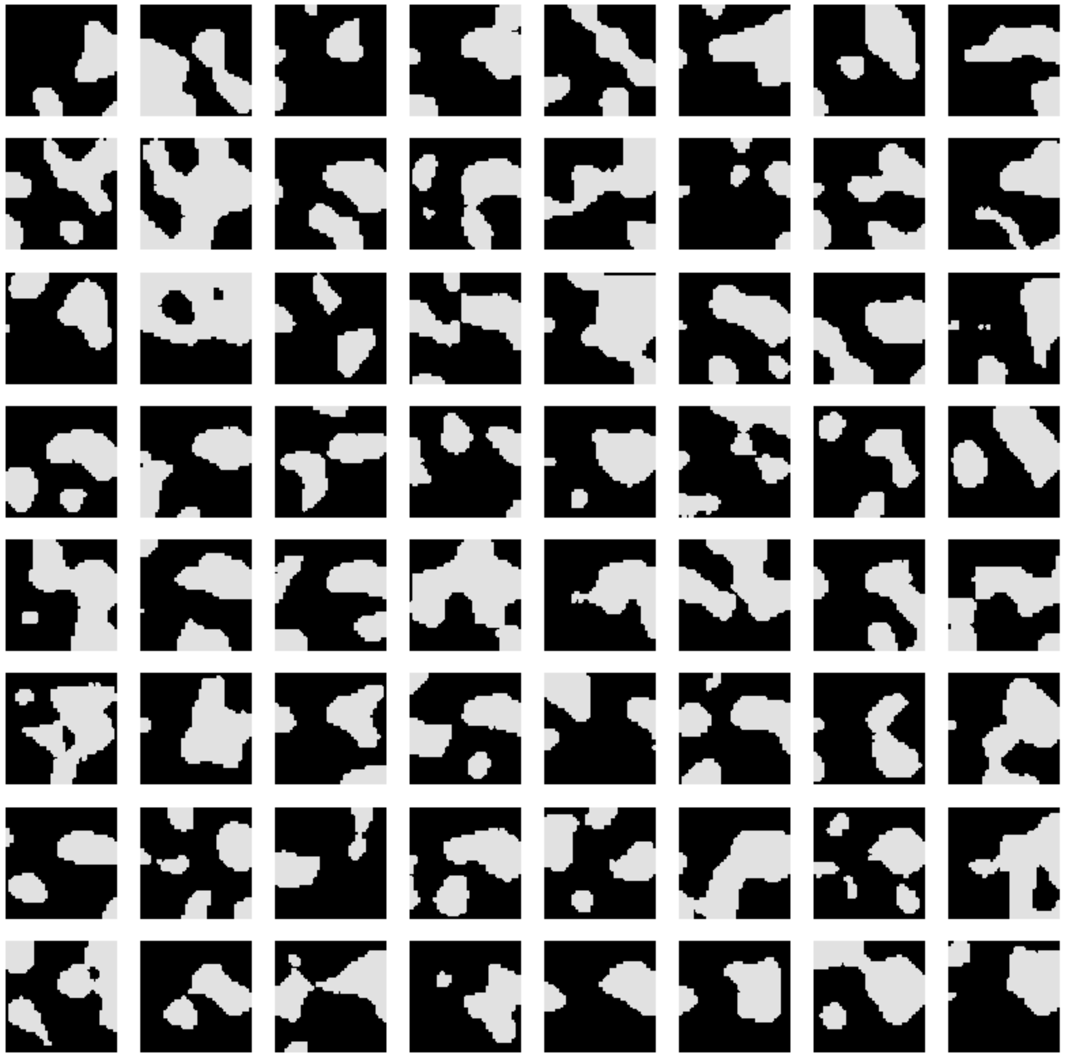
They can claim space by sequentially making moves



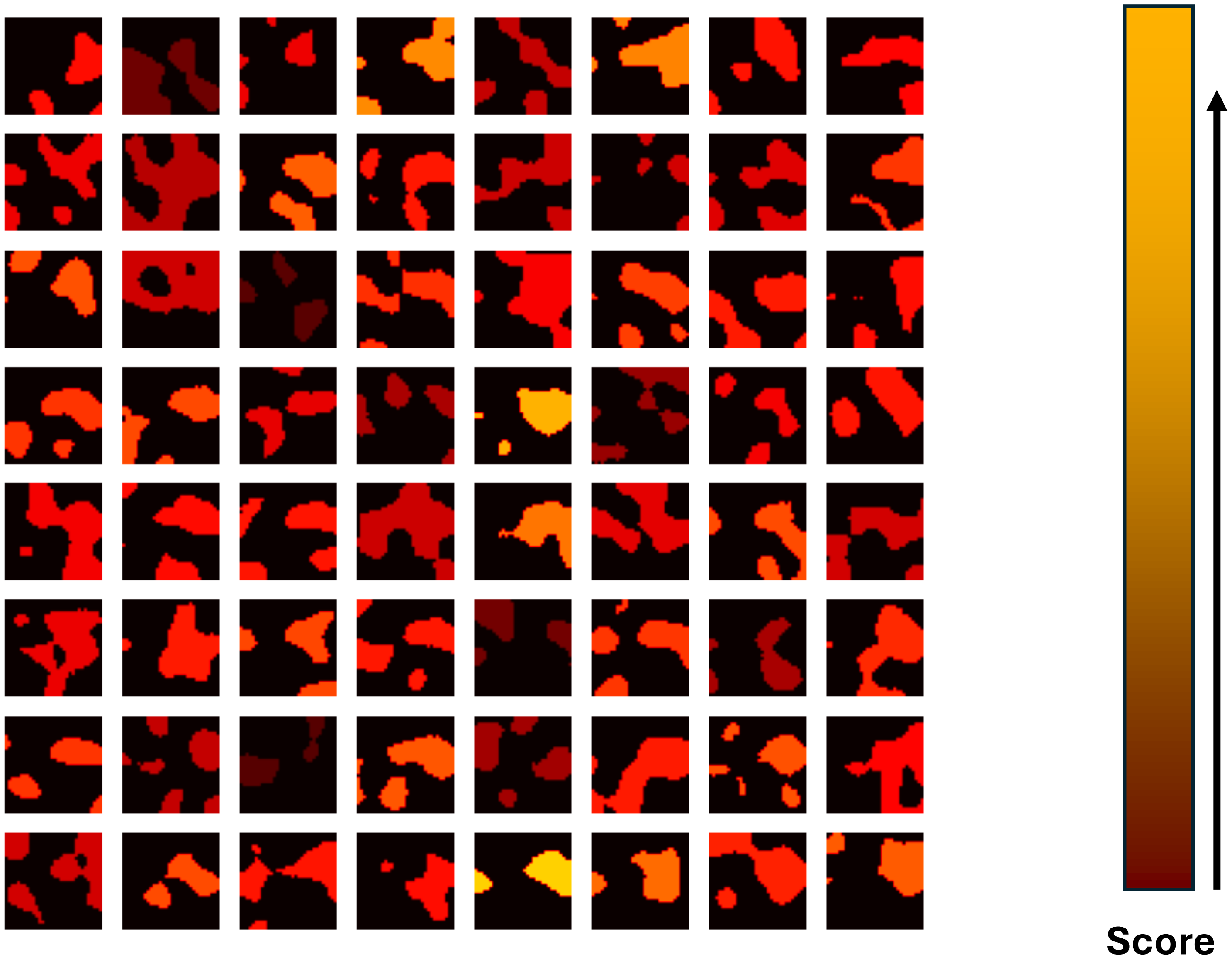
Which move should be made?



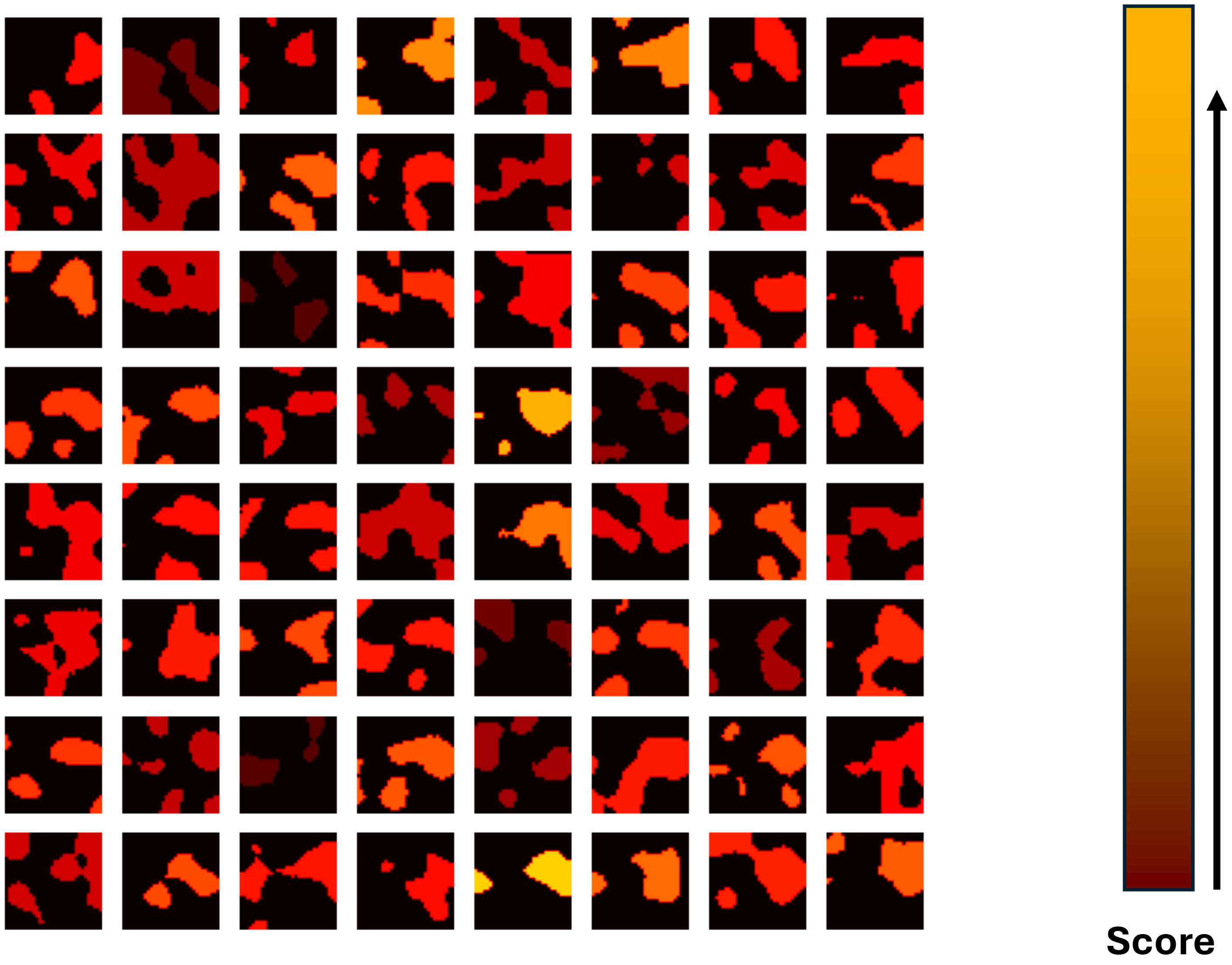
Have a look at these samples



Represent score through colour

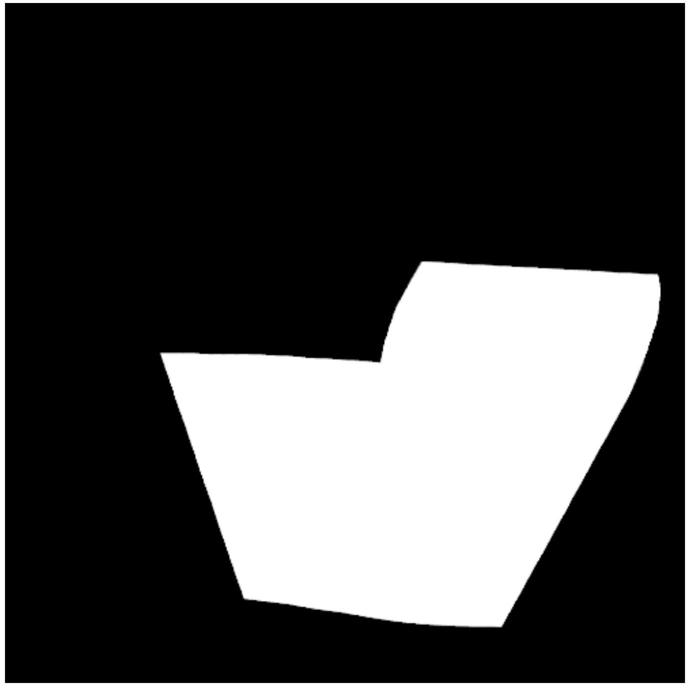


What do the good samples have in common?

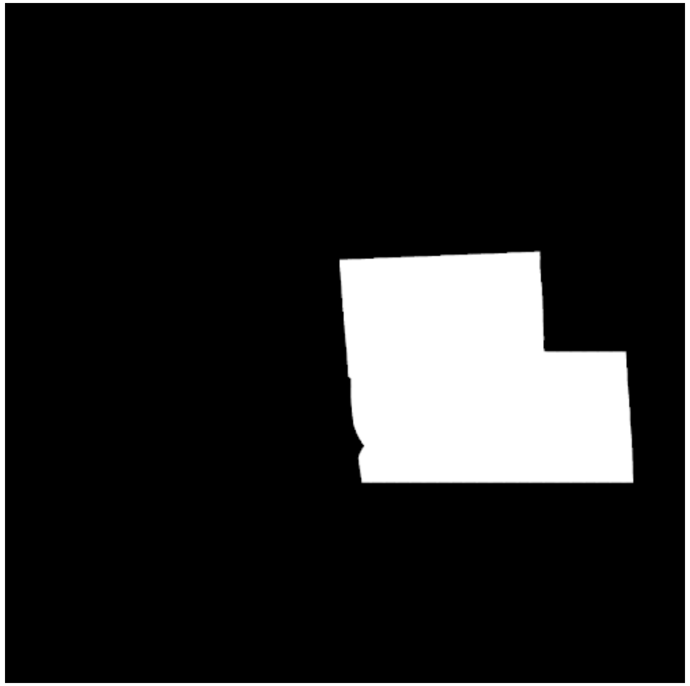


Which will be best?

A



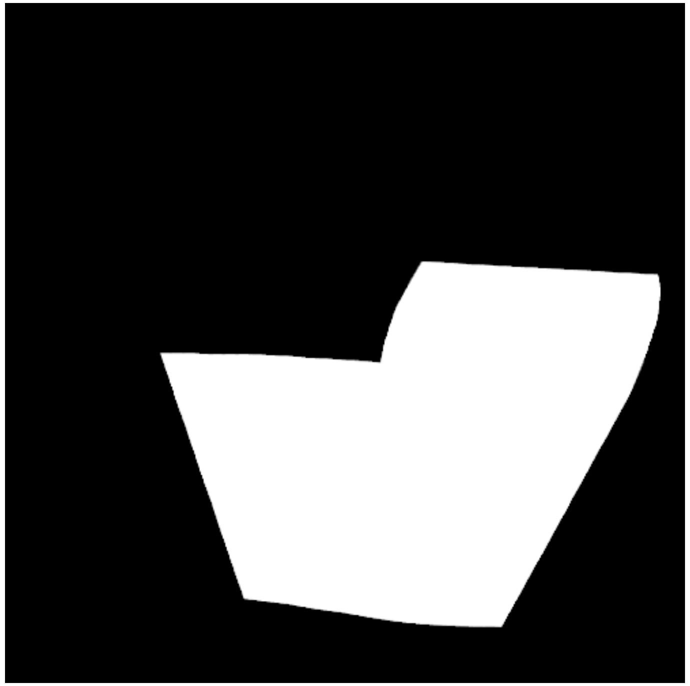
B



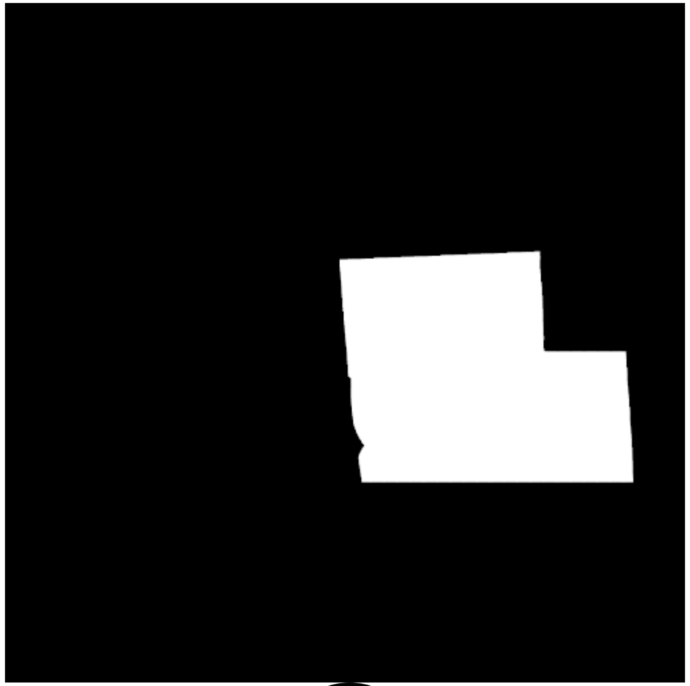
C

Which will be best?

A

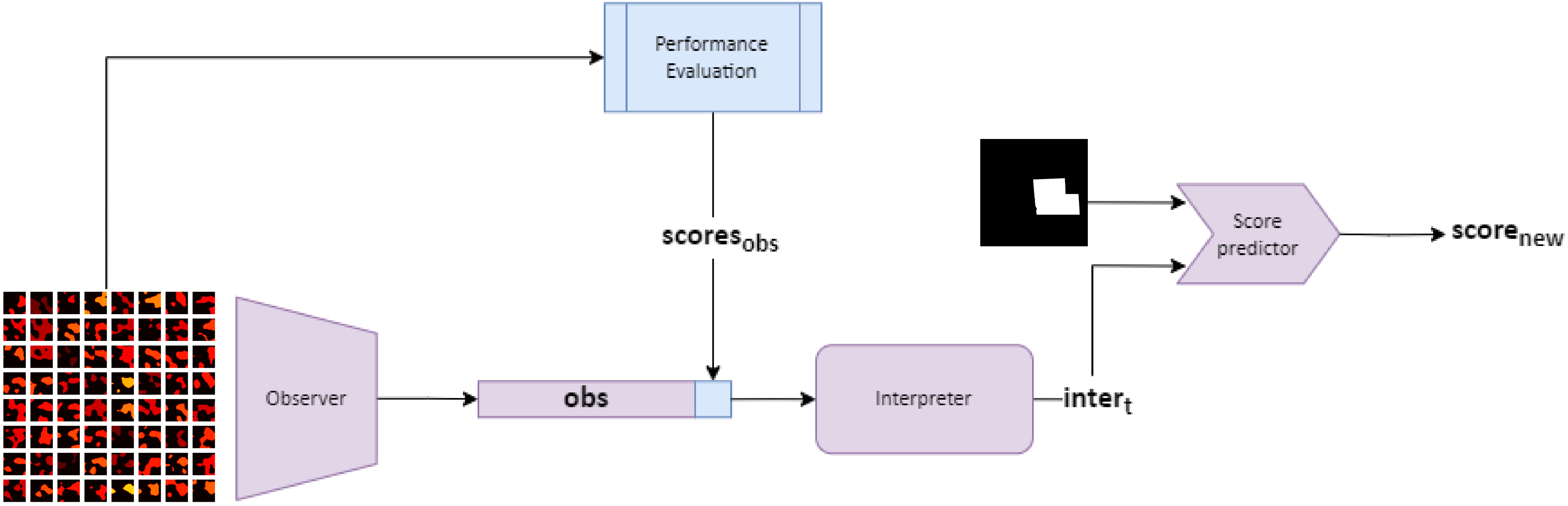


B

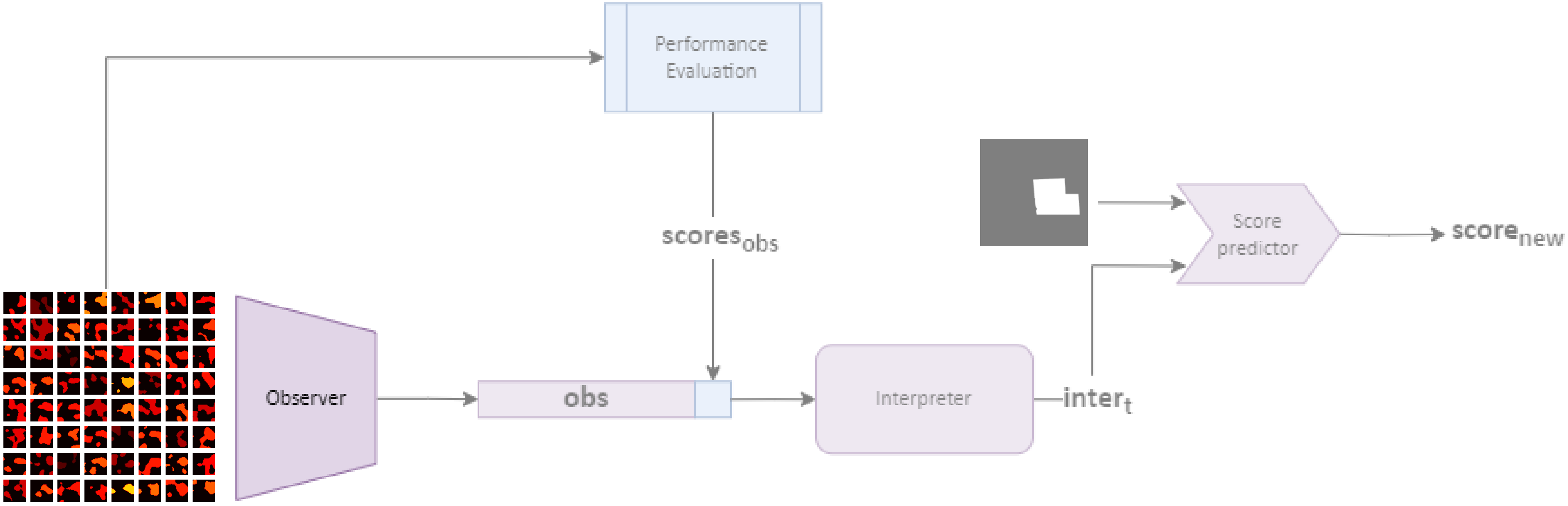


C

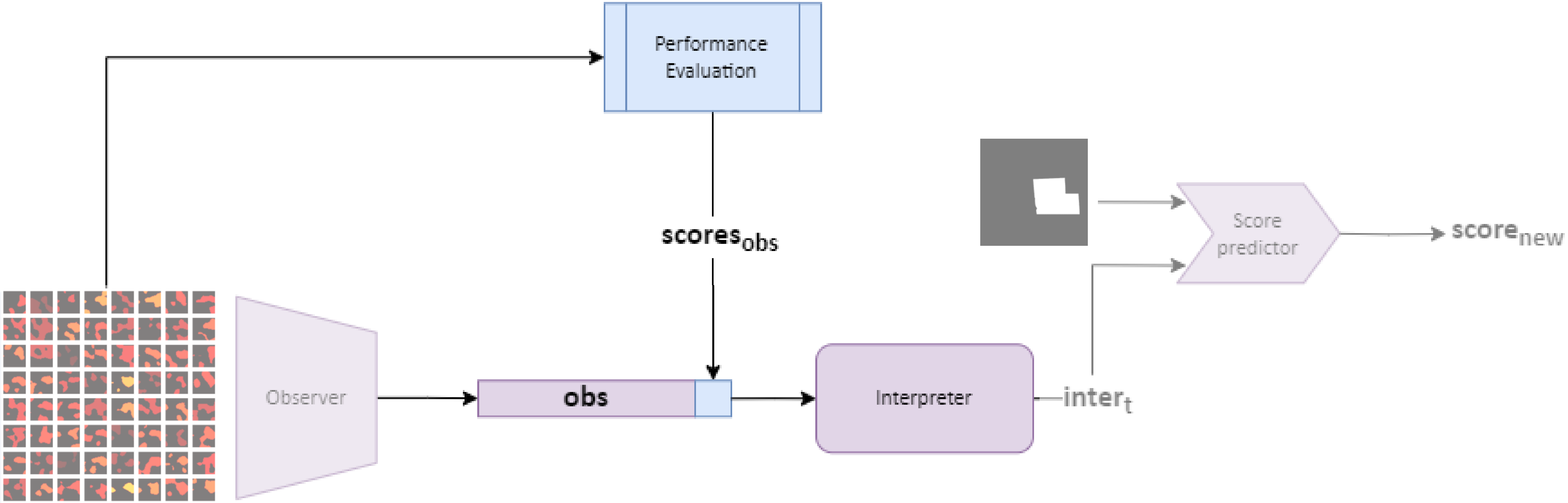
Answering this question requires three steps



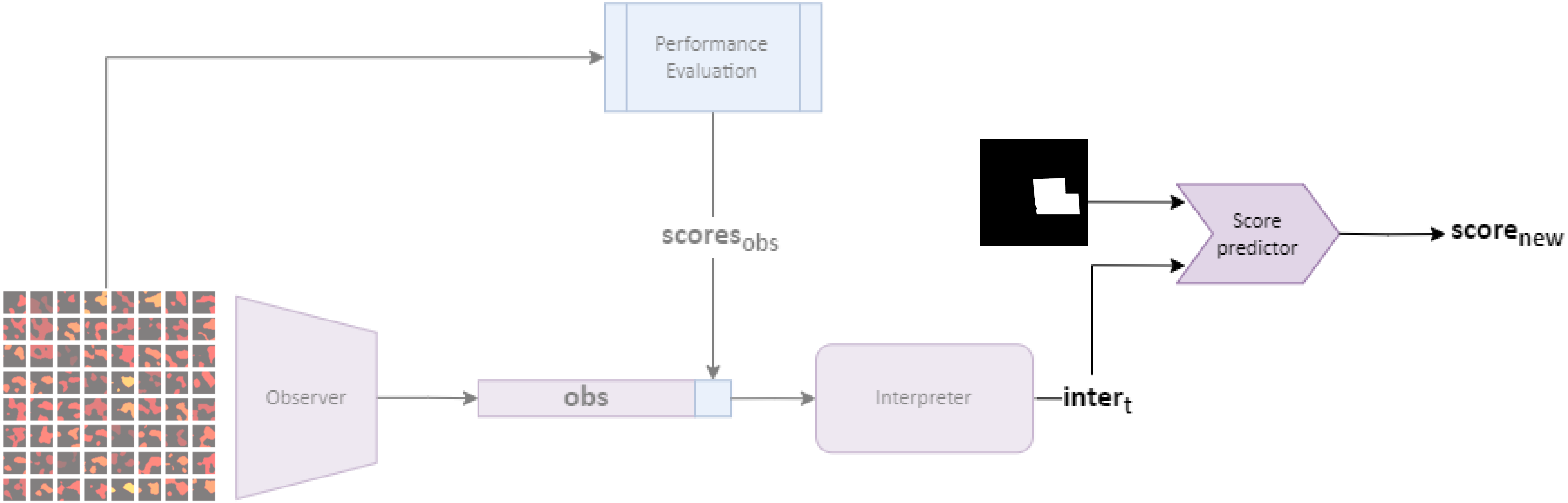
Observation



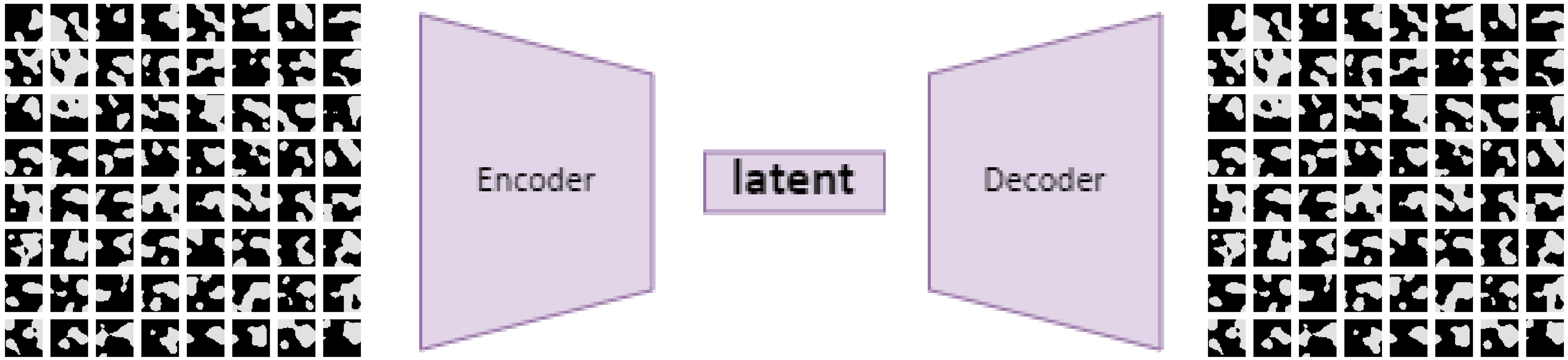
Interpretation



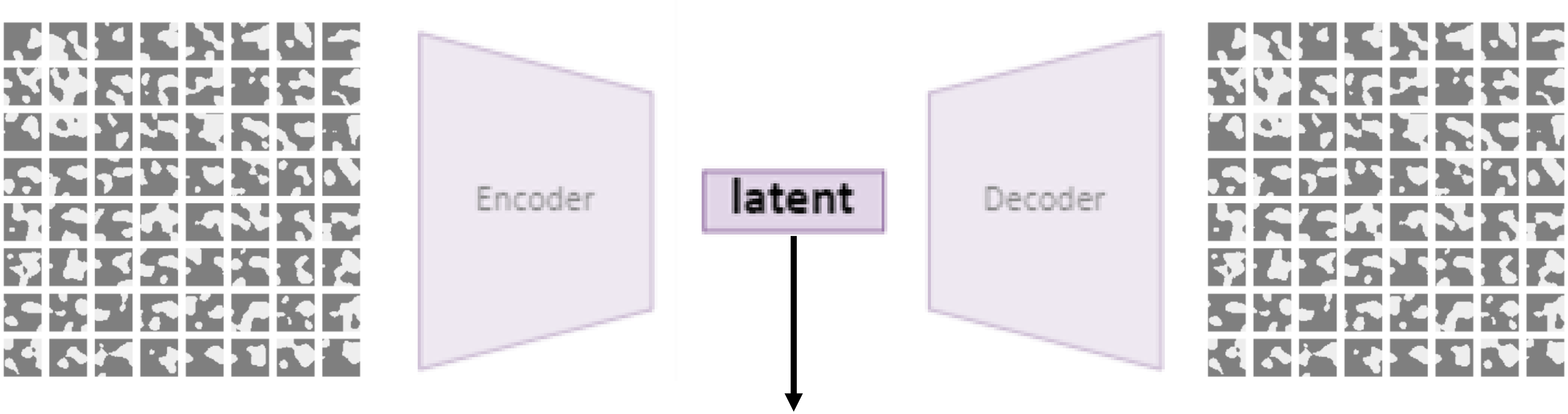
Prediction


Performance
Evaluation
scores$_{obs}$
Observer
obs
Interpreter
inter$_t$
Score
predictor
score$_{new}$

Observation: Auto Encoder

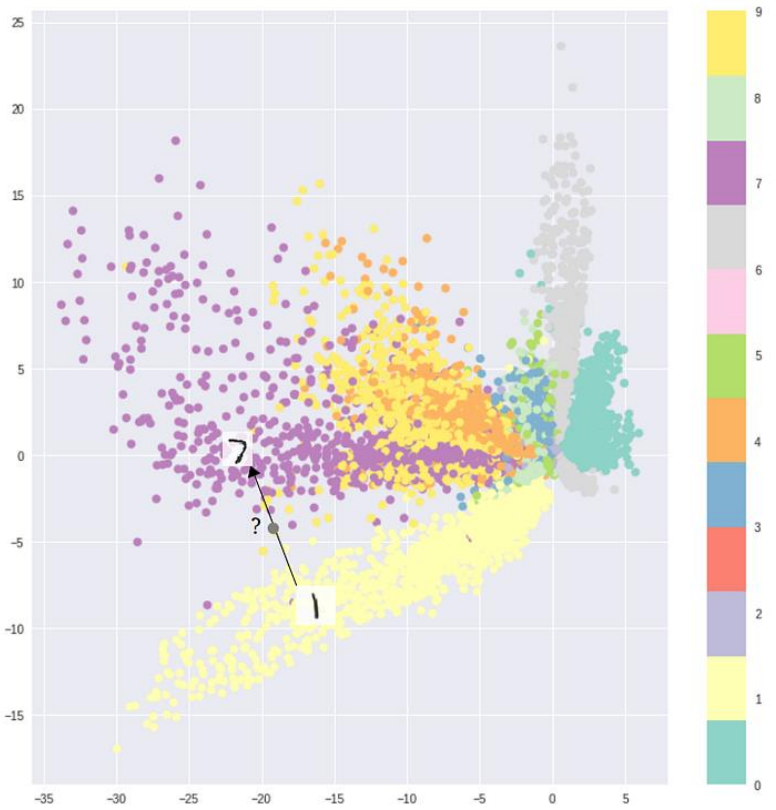


Observation: Auto Encoder

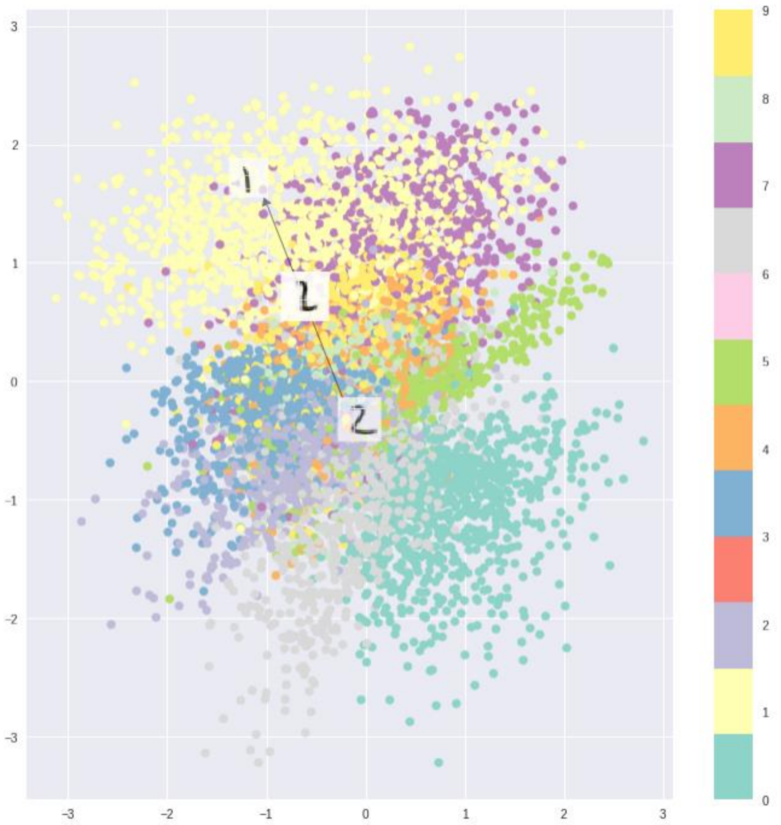


Compressed representations

Interpretability of latent space



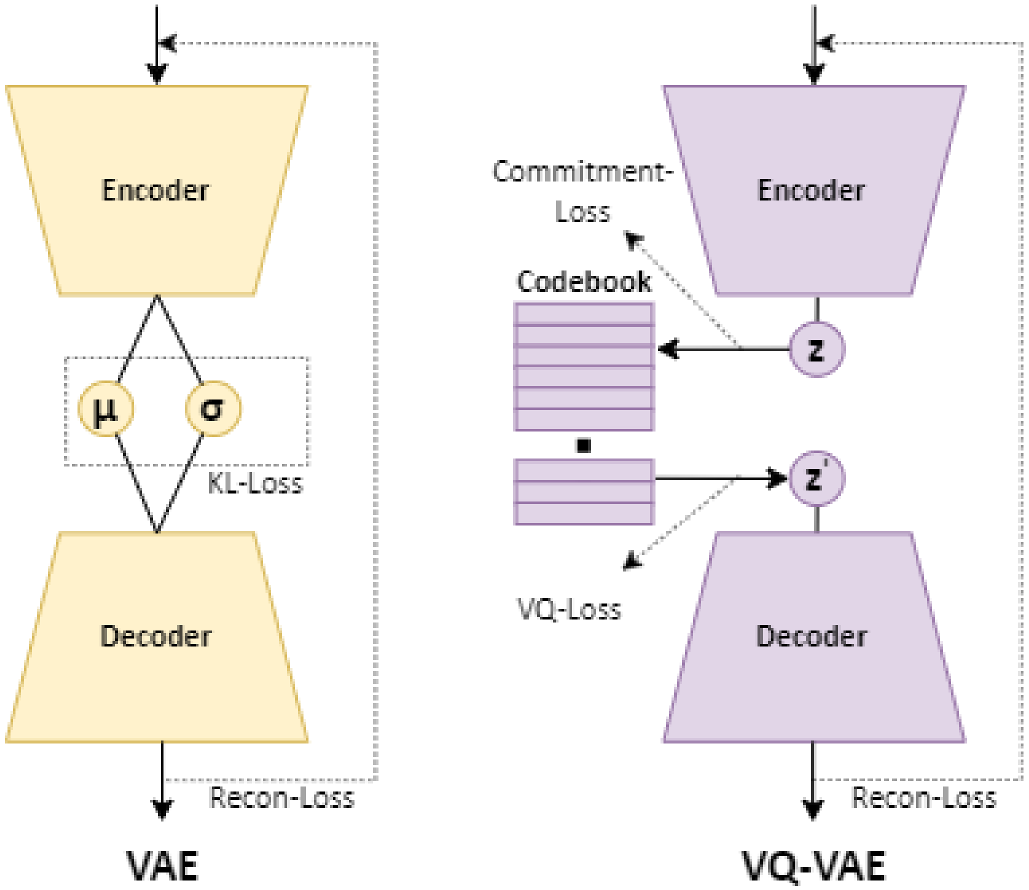
AE



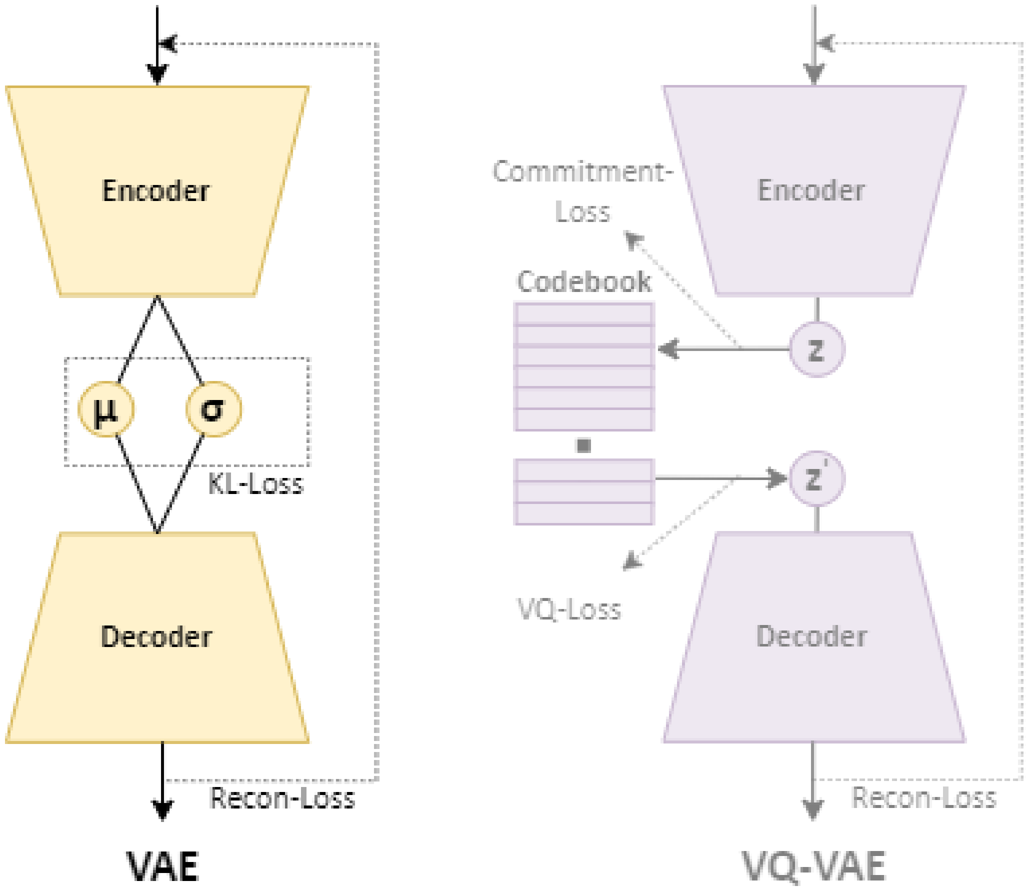
VAE

Irhum Shafkat (2018), Intuitively understanding variational autoencoders

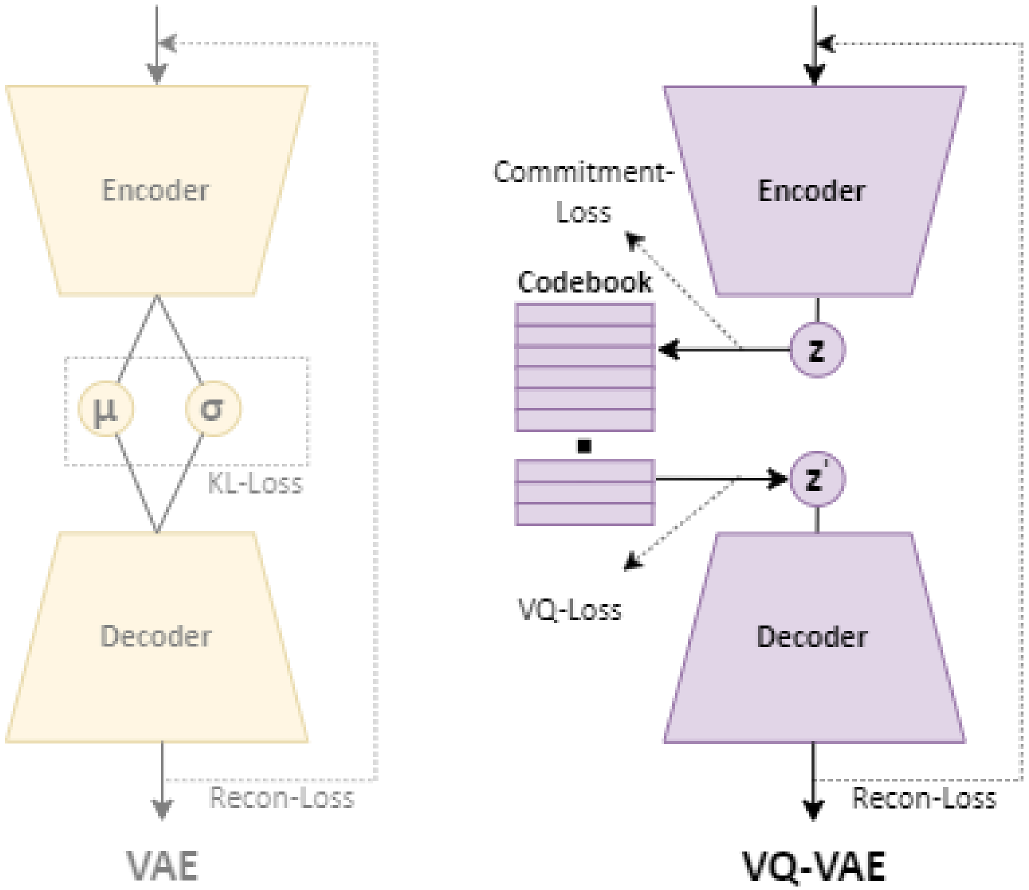
Observation: VAE vs VQ-VAE



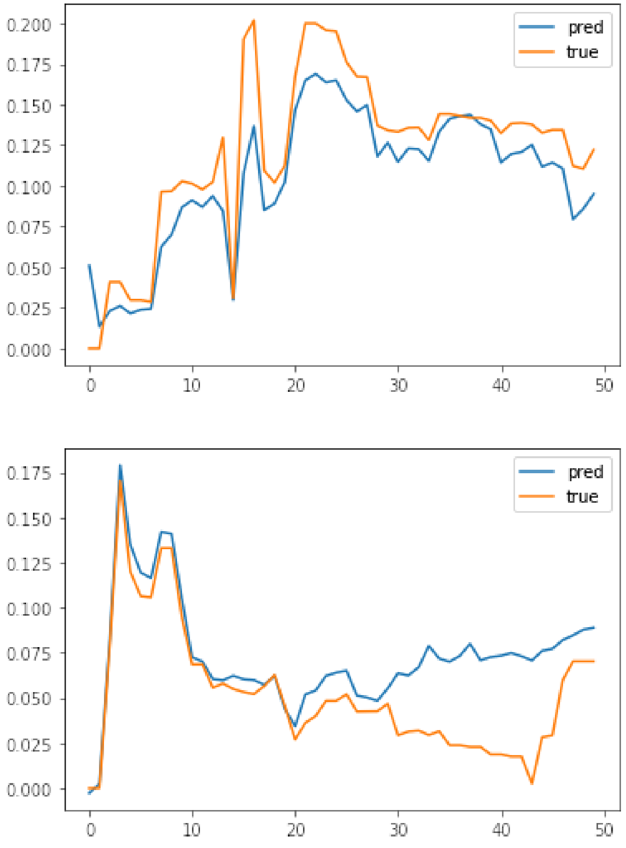
Observation: VAE vs VQ-VAE



Observation: VAE vs VQ-VAE

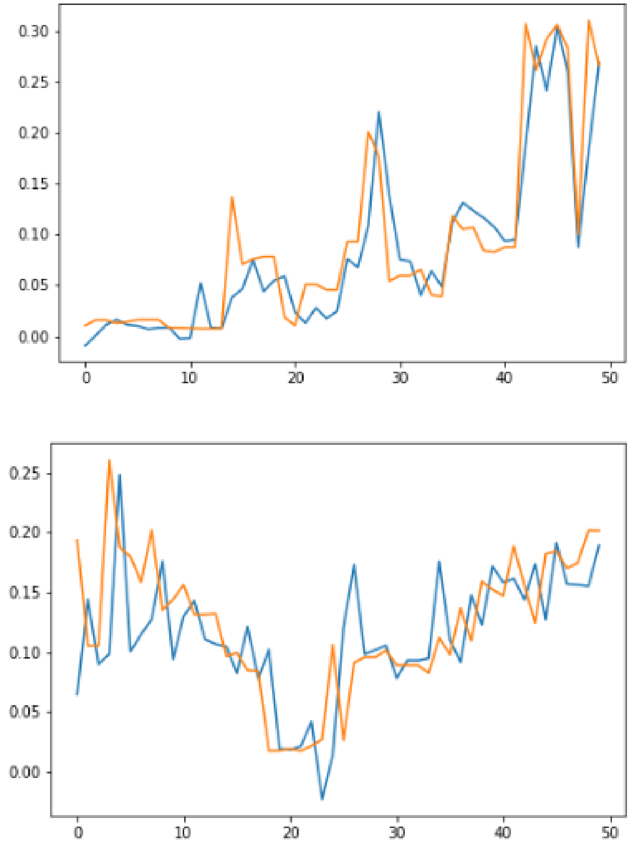


Observation: VAE vs VQ-VAE



VAE

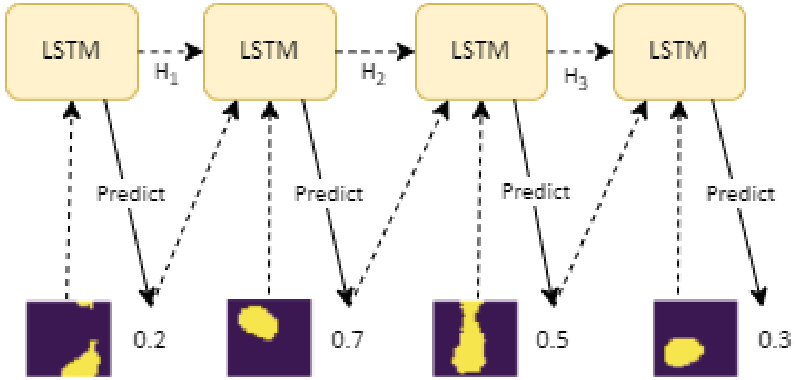
Mean error ±20%



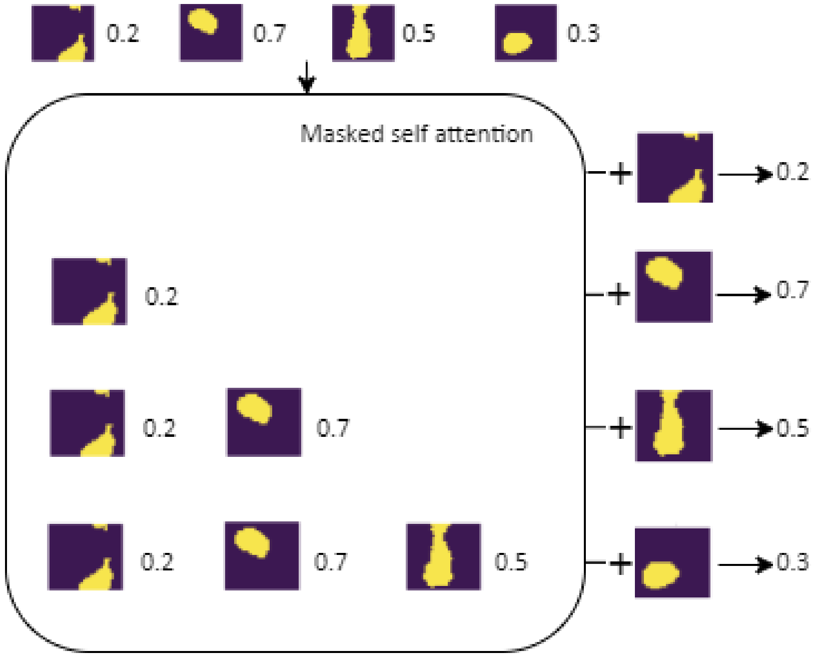
VQ-VAE

Mean error ±10%

Interpretation: LSTM vs Transformer

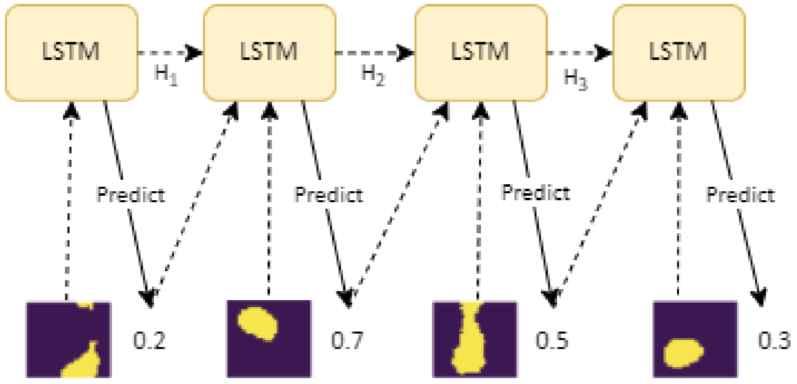


LSTM

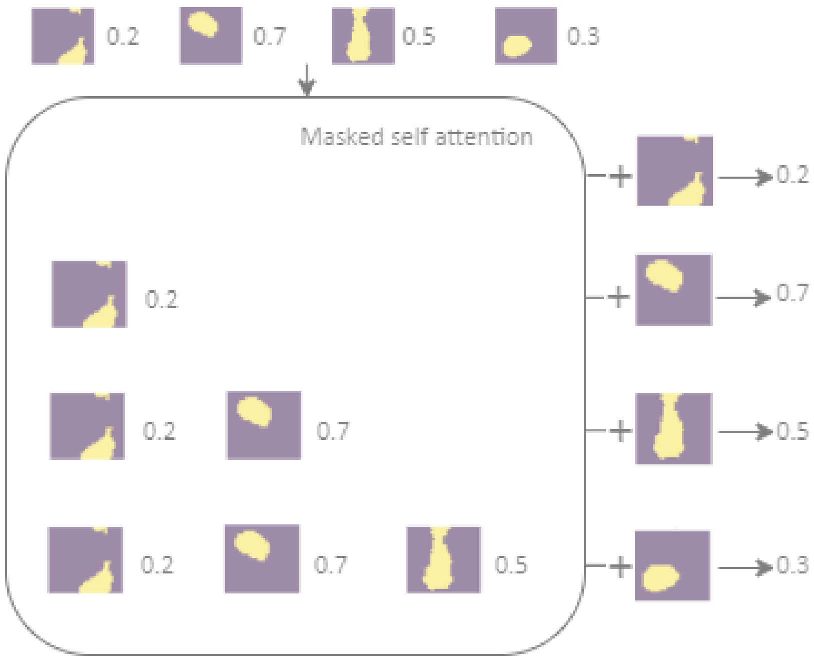


Transformer

Interpretation: LSTM vs Transformer

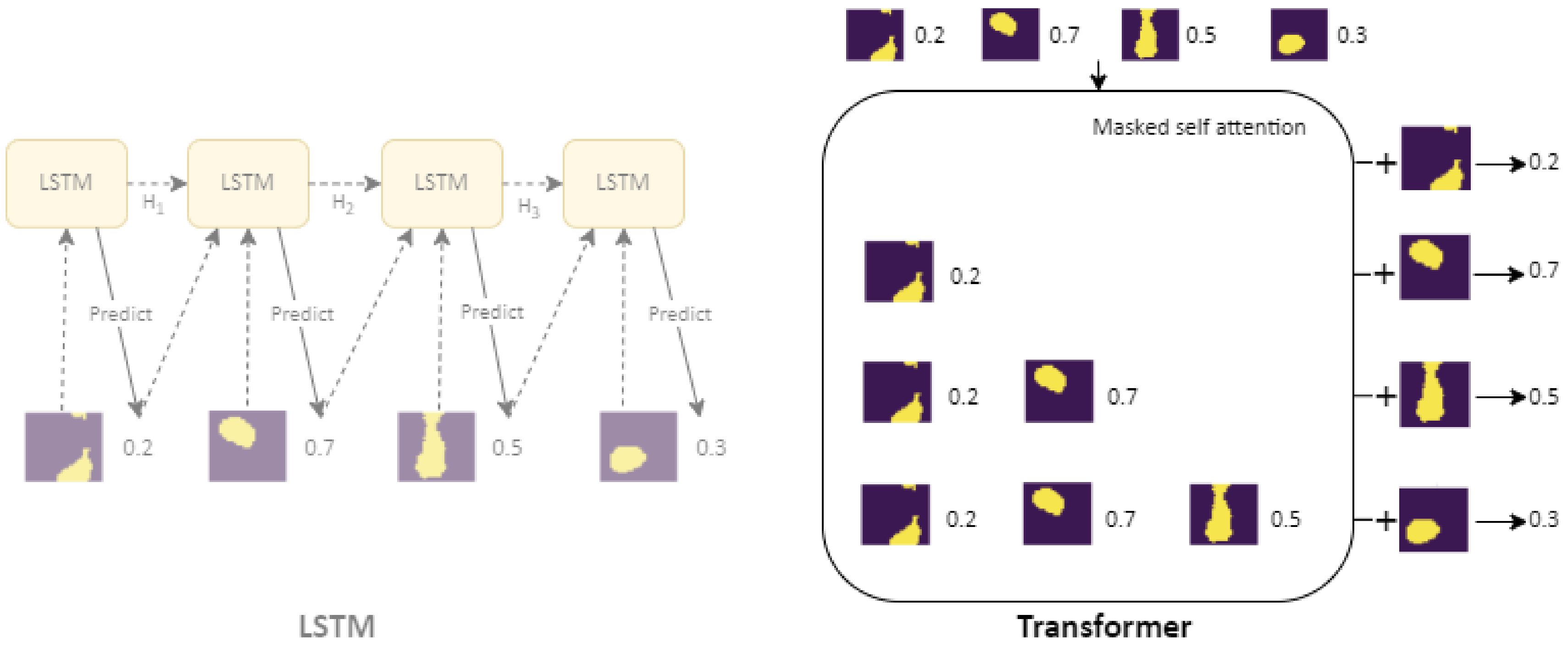


LSTM

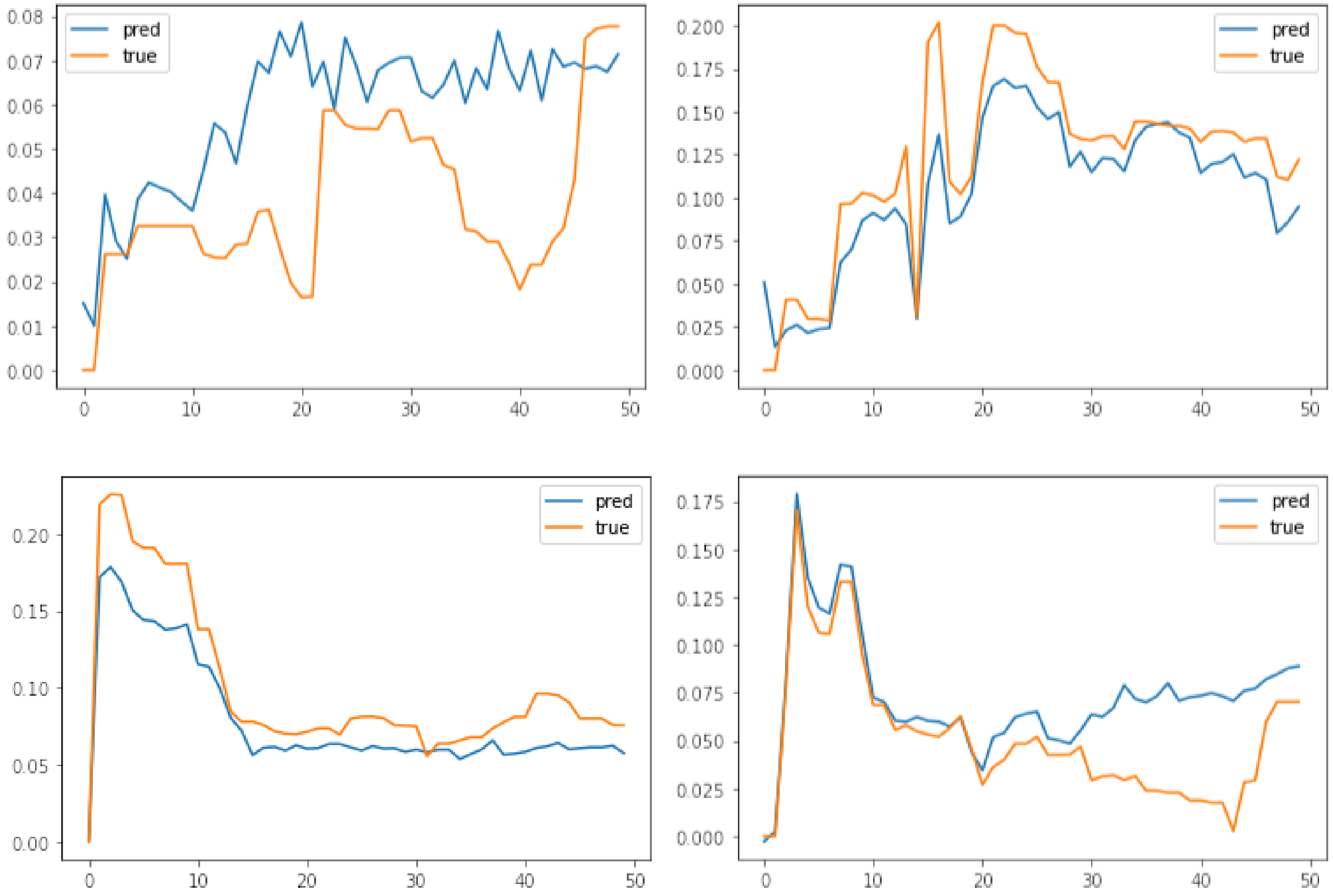


Transformer

Interpretation: LSTM vs Transformer



Interpretation: LSTM vs Transformer (with VAE)



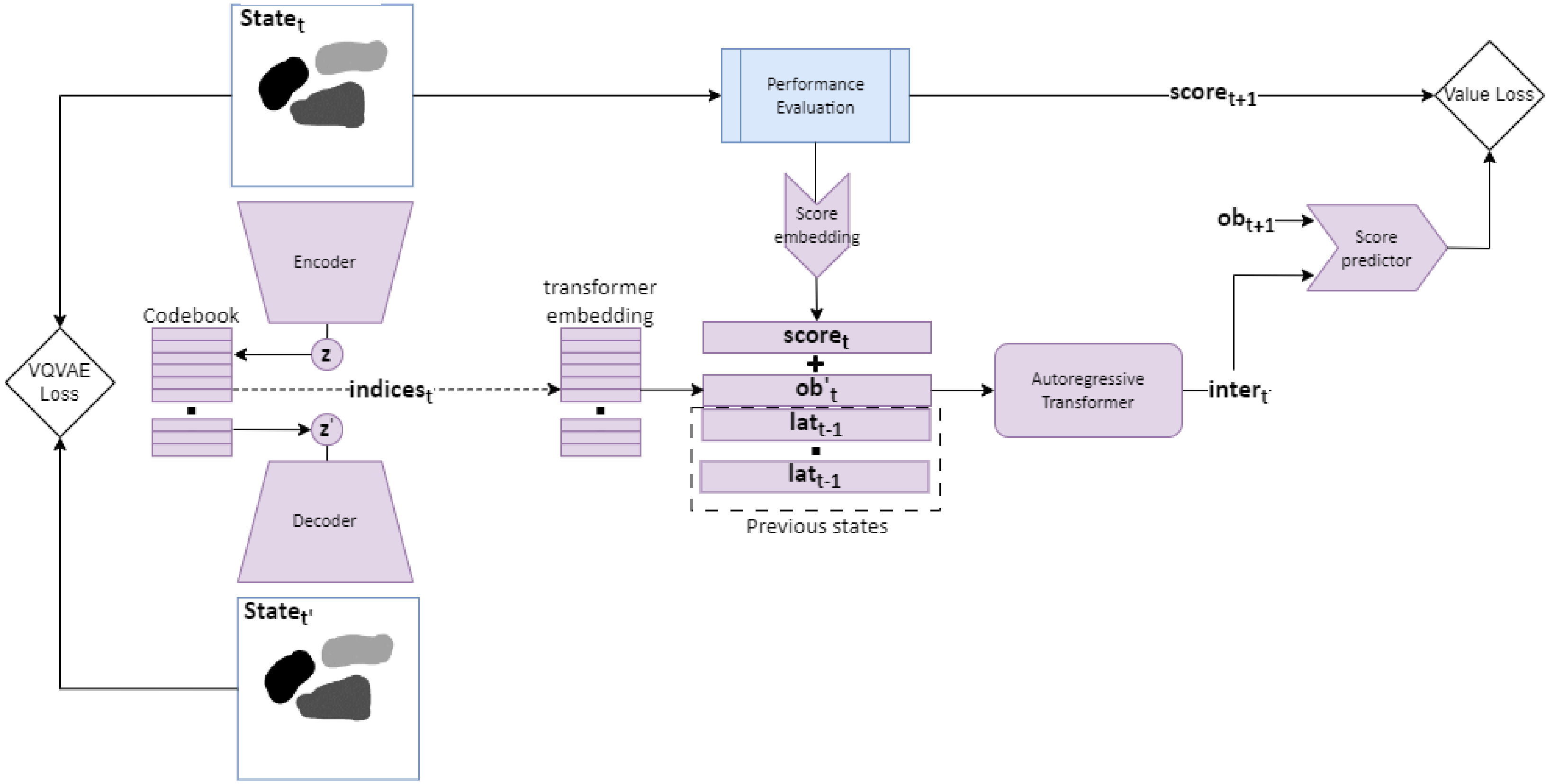
LSTM

Transformer

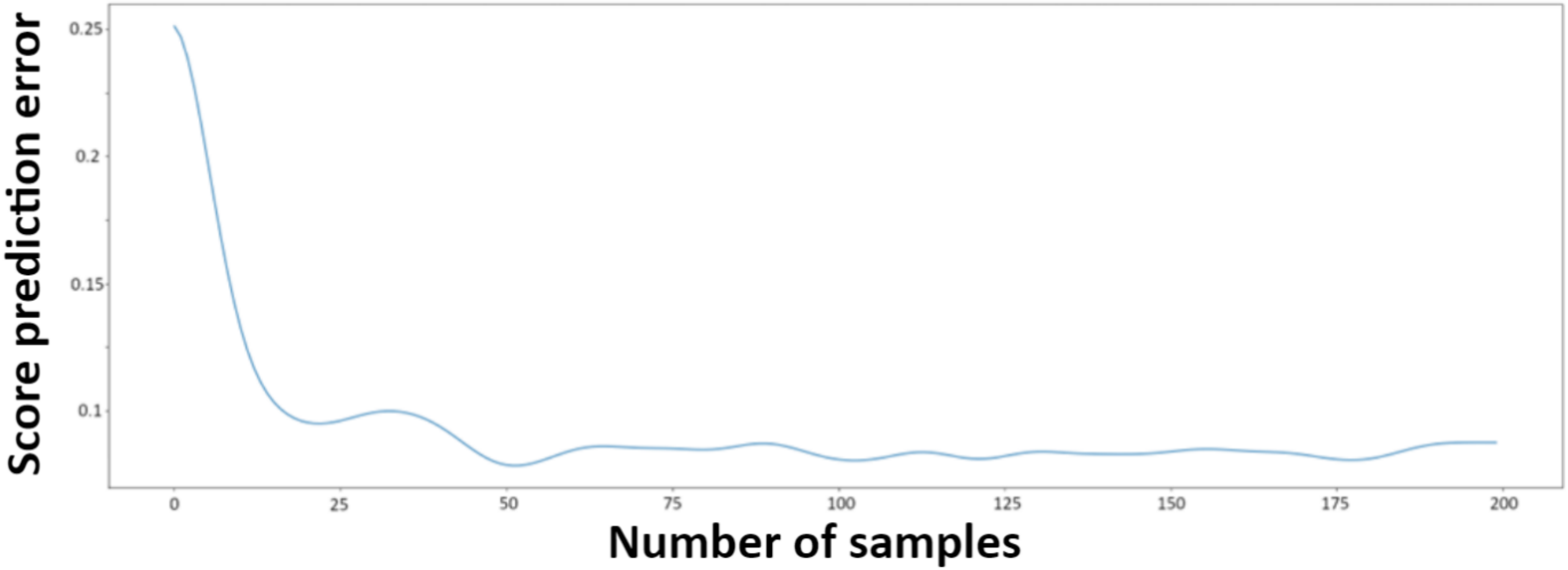
Mean error ±30%

Mean error ±20%

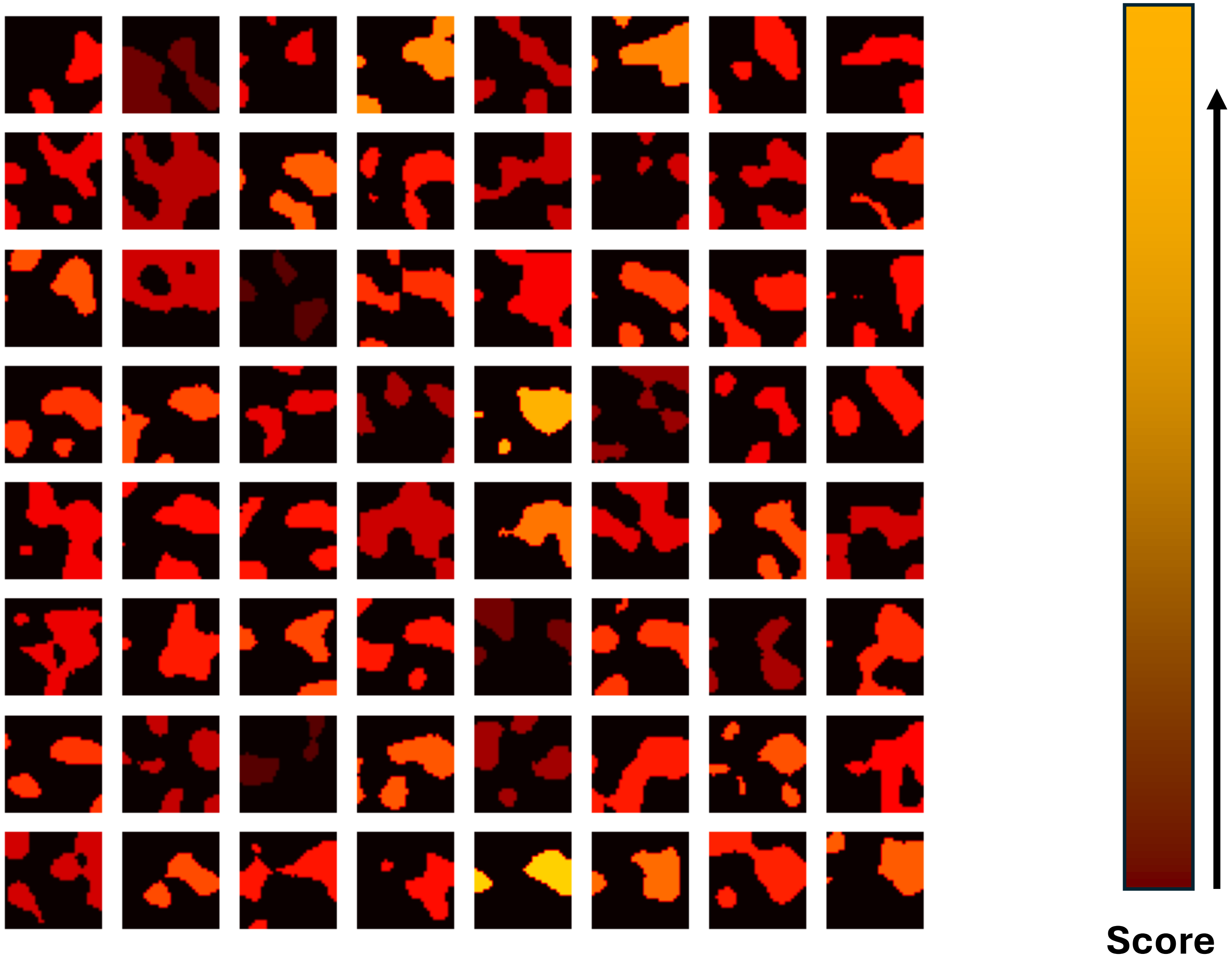
Prediction model: VQ-VAE + Transformer



Prediction: VQ-VAE + Transformer



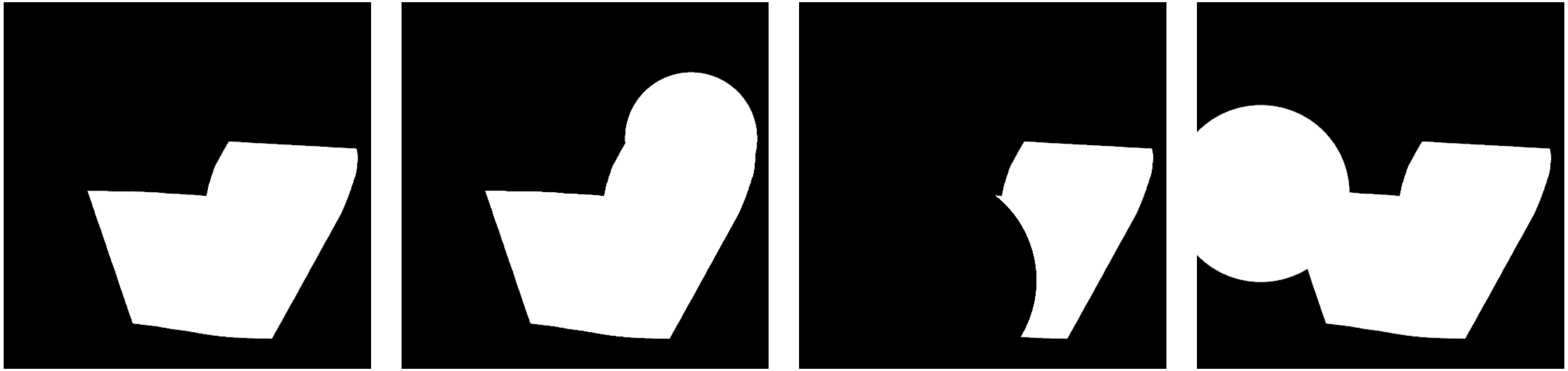
What do the good samples have in common?



If we want to improve this sample, what change should be made A,B or C?

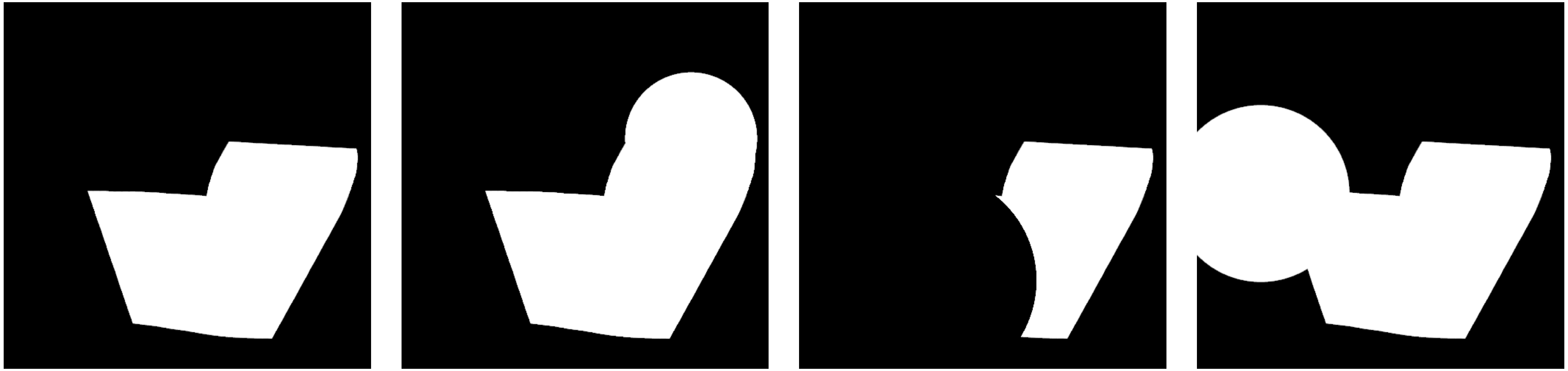
Original

If we want to improve this sample, what change should be made A,B or C?



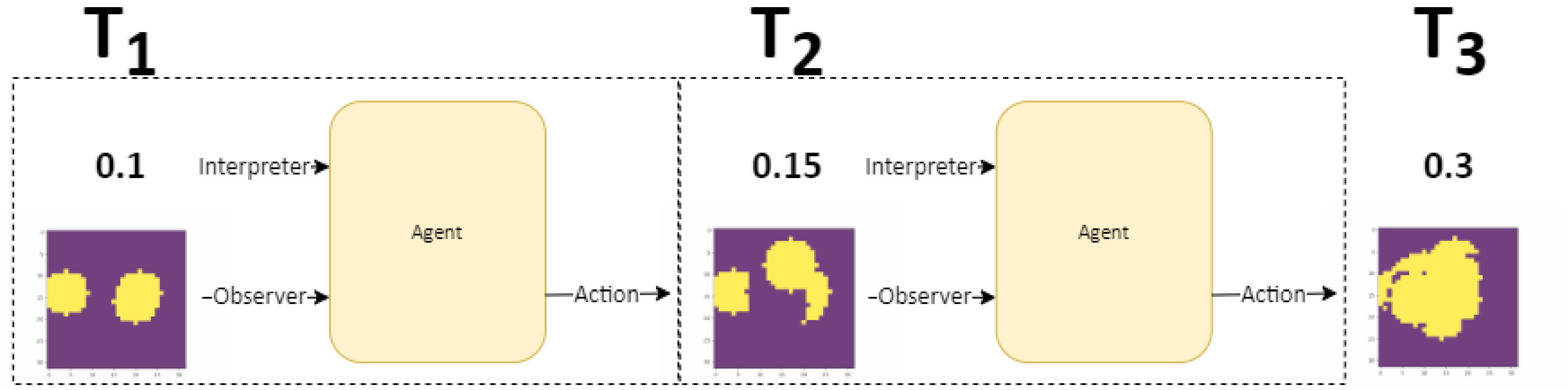
Original **A** **B** **C**

If we want to improve this sample, what change should be made A,B or C?

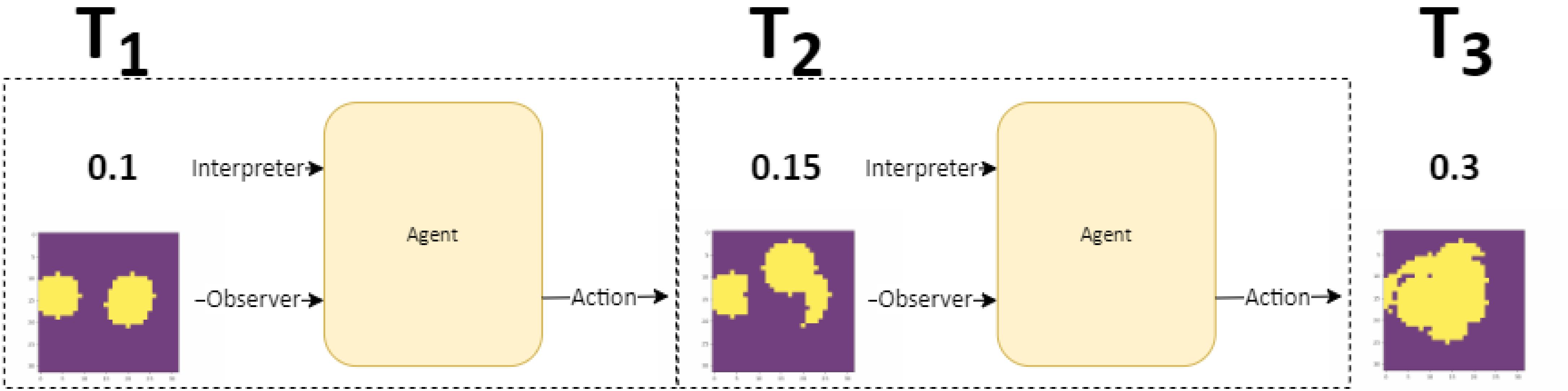


Original **A** **B** **C**

Action: Proximal Policy Optimization

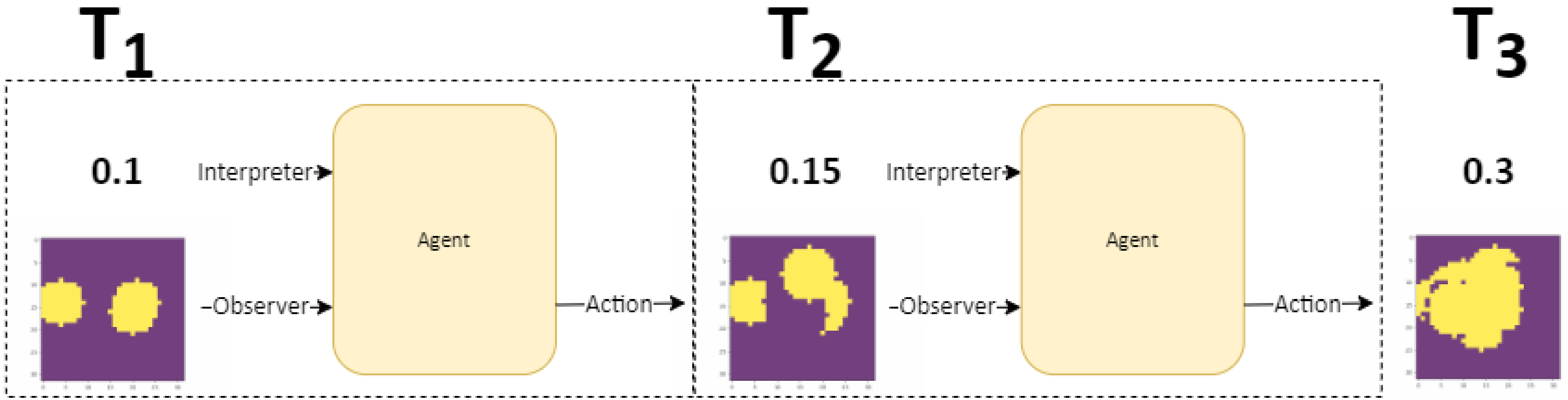


Advantage



$$\textbf{Advantage} = \text{score}_{t+1} - \text{score}_t$$

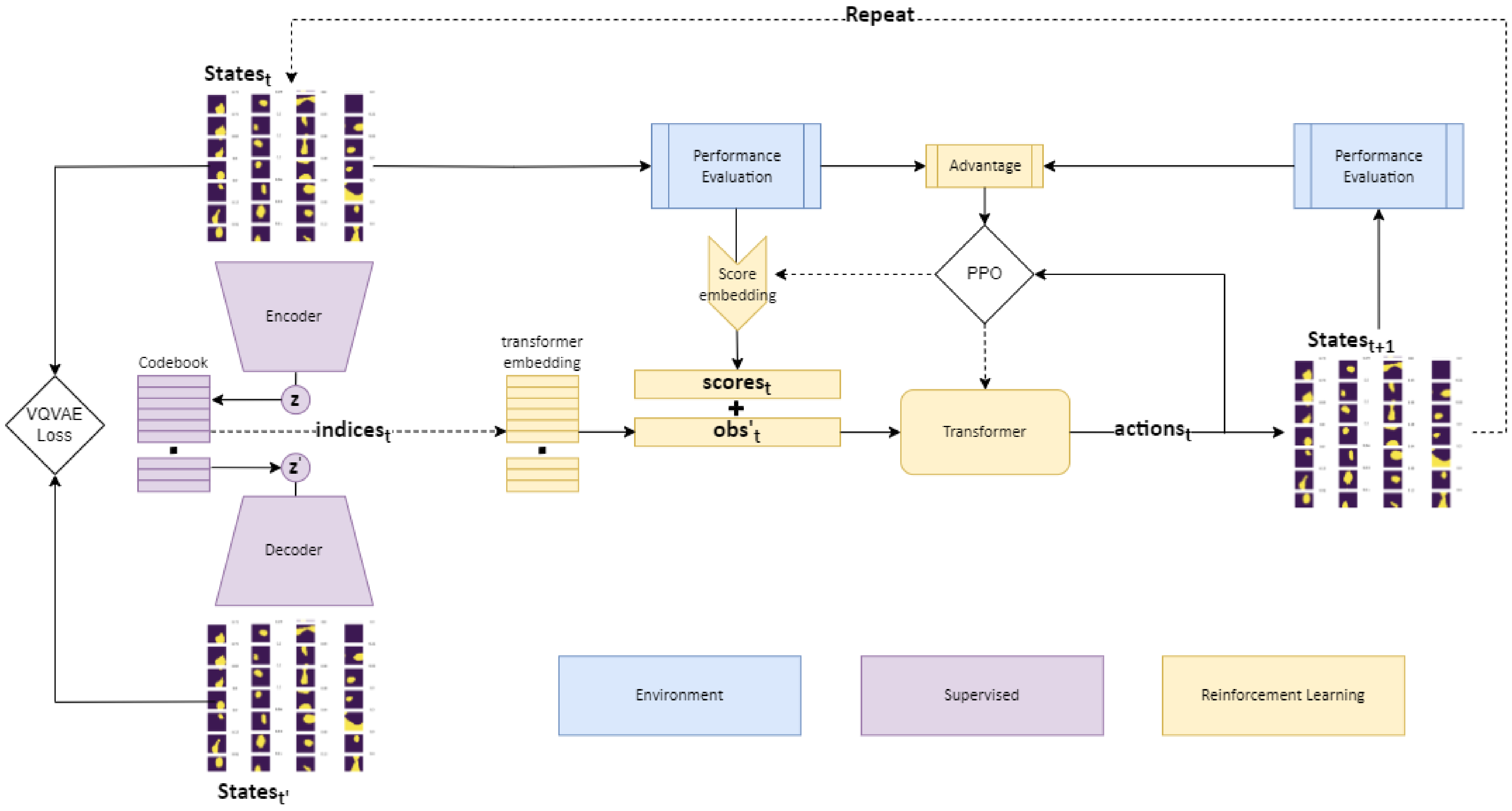
Reward



Advantage $_1$ **=0.15-0.1=0.05**

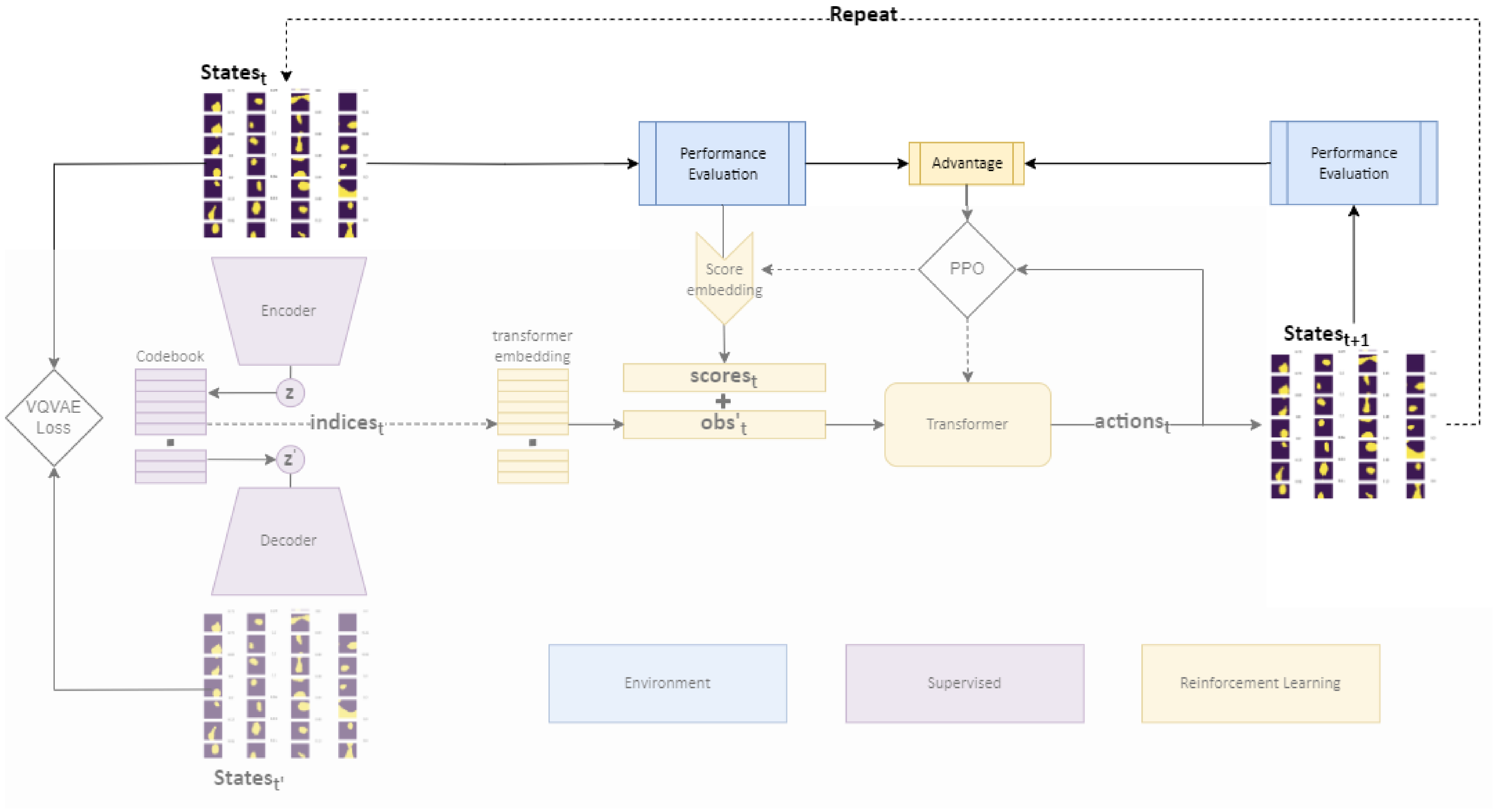
Advantage $_2$ **=0.3-0.15=0.15**

Adding PPO

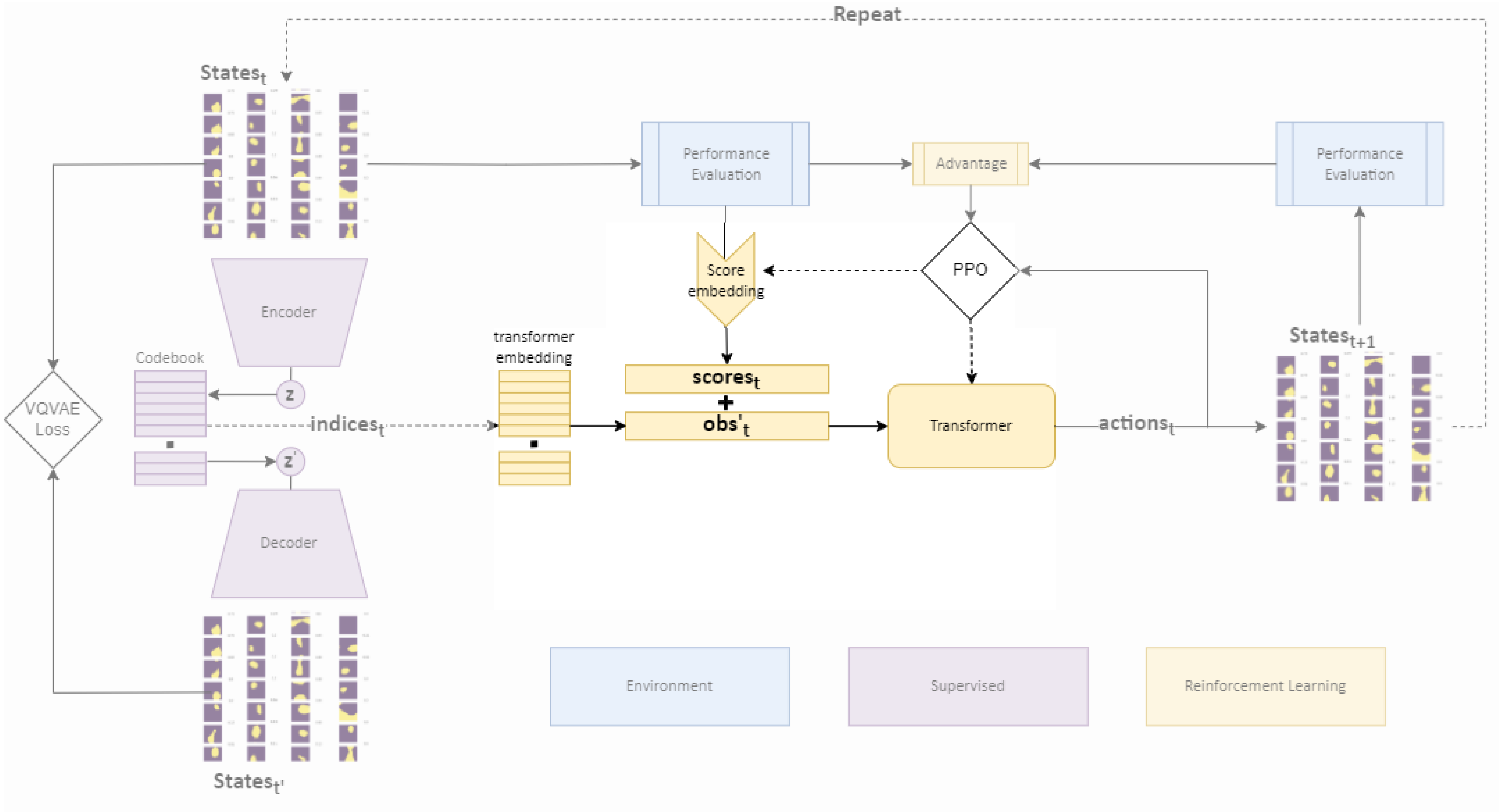


Repeat
States$_t$
Performance Evaluation
Advantage
Performance Evaluation
Score embedding
PPO
Encoder
Codebook
z
indices$_t$
transformer embedding
scores$_t$
+
obs'$_t$
Transformer
actions$_t$
States$_{t+1}$
VQVAE Loss
z'
Decoder
States$_{t'}$
Environment
Supervised
Reinforcement Learning

Computing advantage



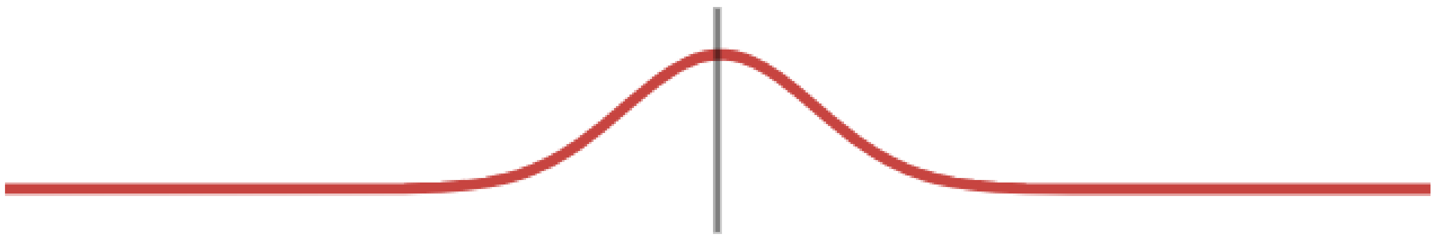
Training transformer



Increasing probability of advantageous actions

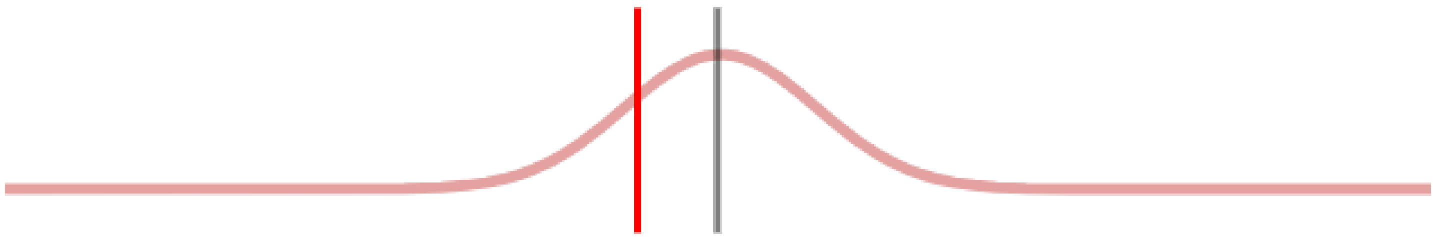
Action current policy

Increasing probability of advantageous actions



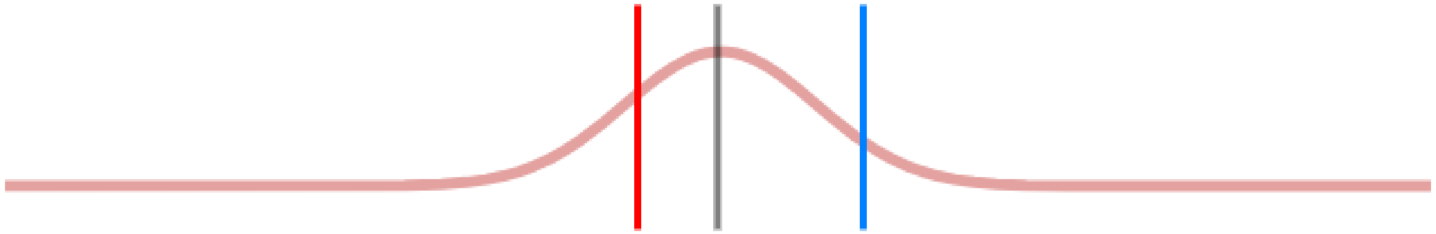
Sample action stochastically

Increasing probability of advantageous actions



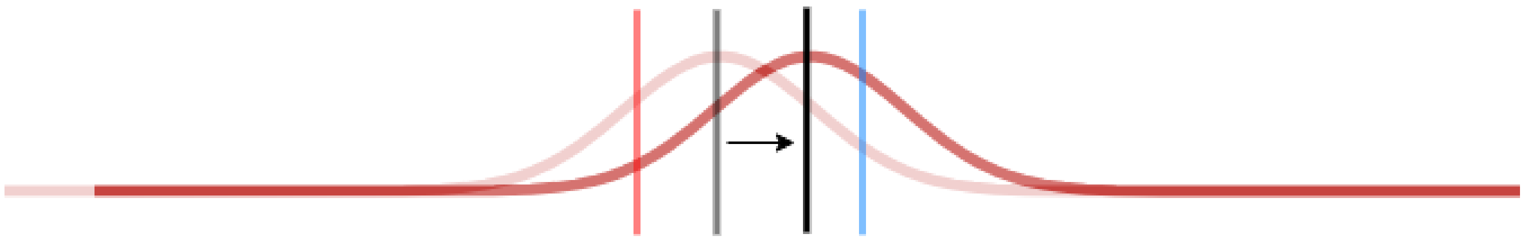
One sample is disadvantageous

Increasing probability of advantageous actions



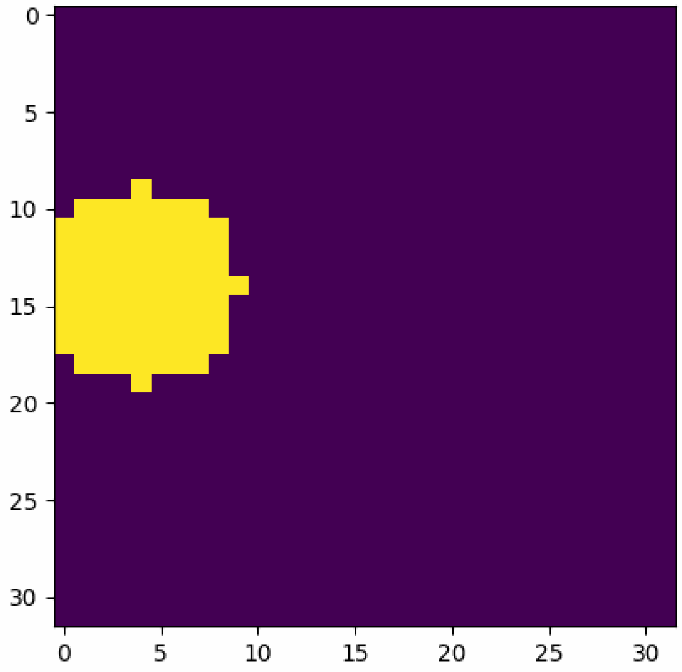
One sample is advantageous

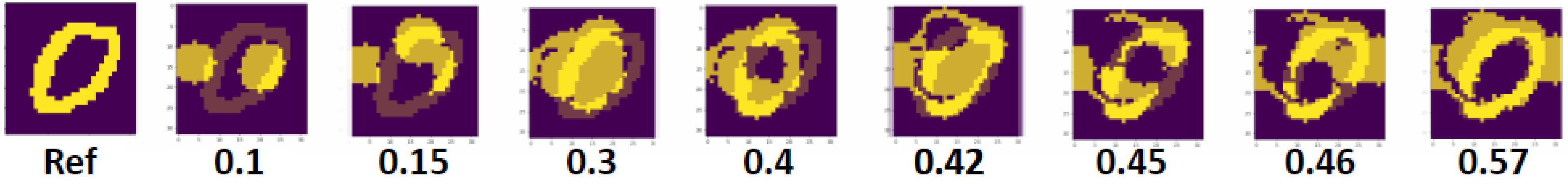
Increasing probability of advantageous actions



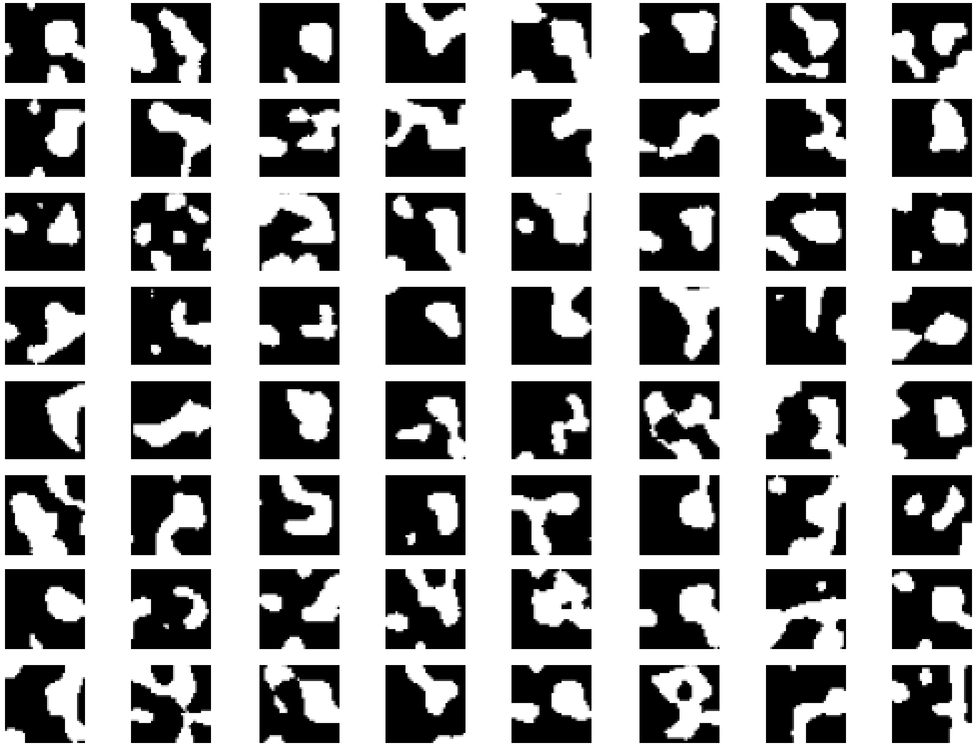
Change policy

Process visualization

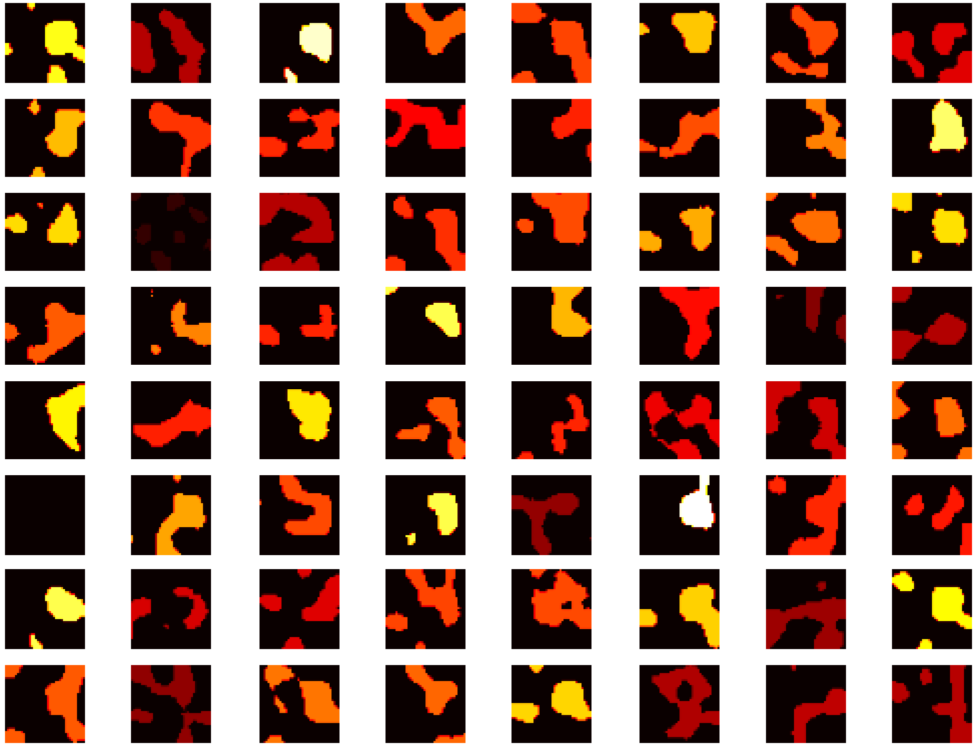




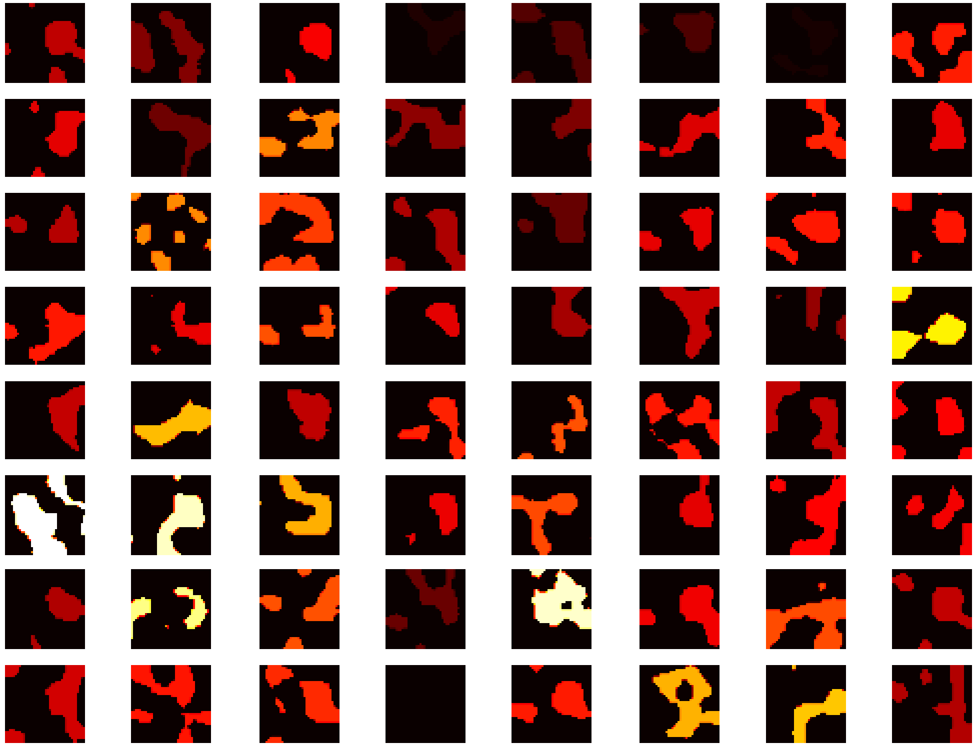
How does this work?



We again assign scores.

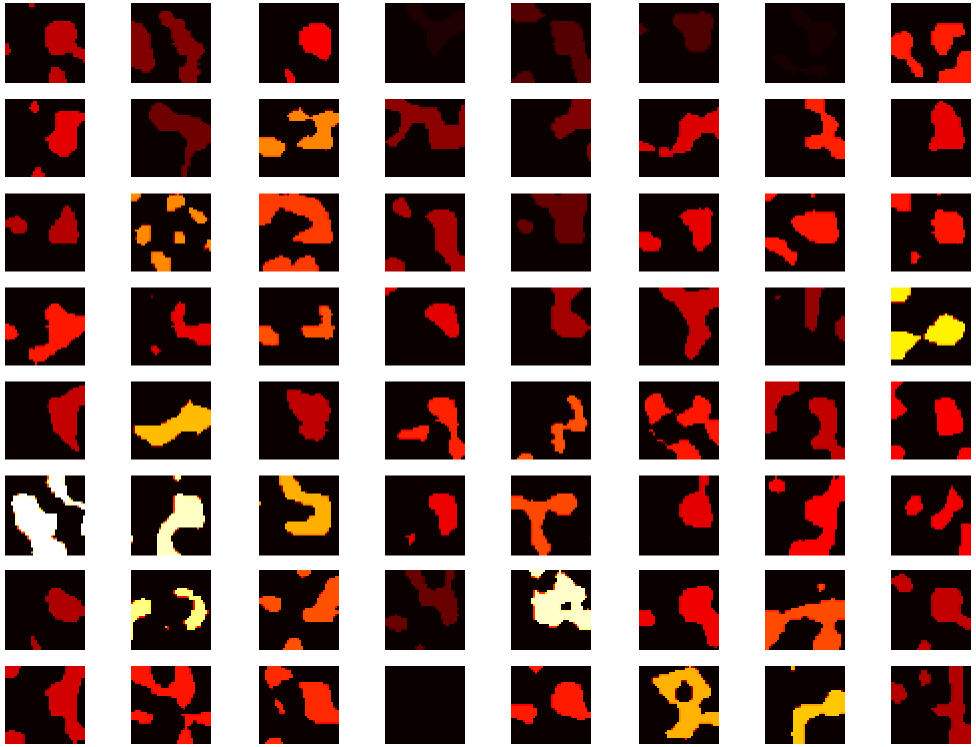


Attention.

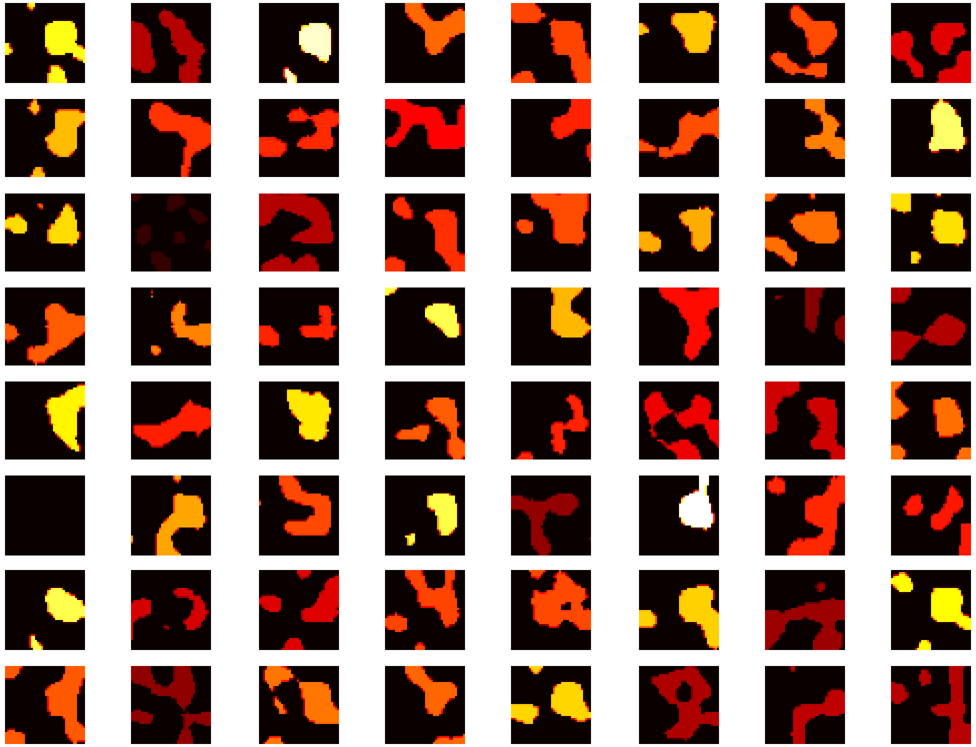


Average attention

Attention is not score

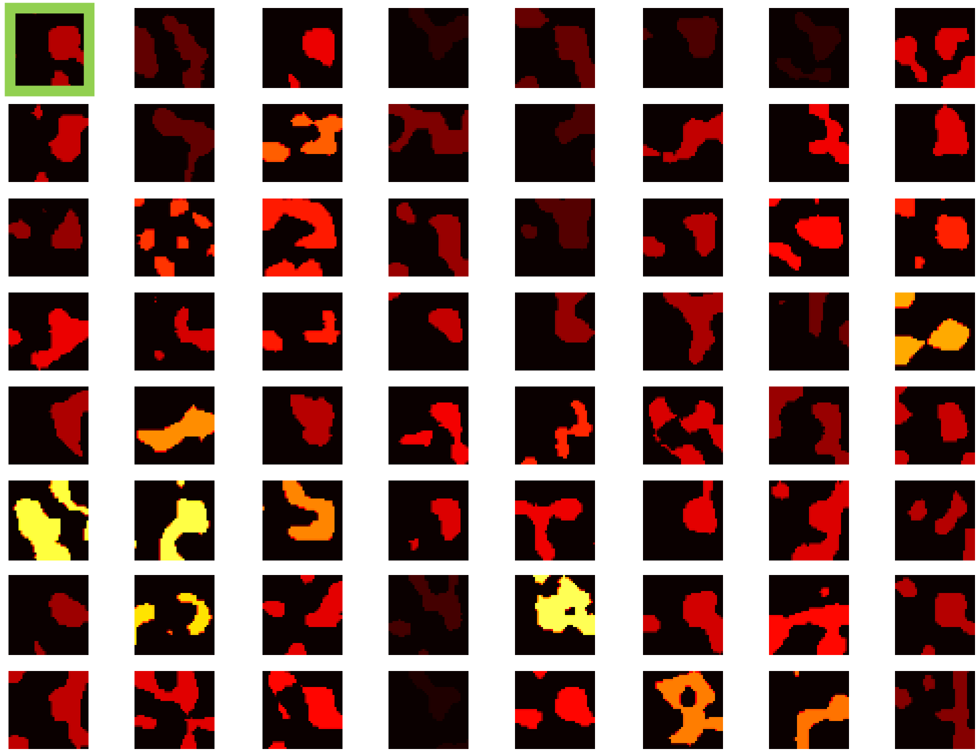


Average attention

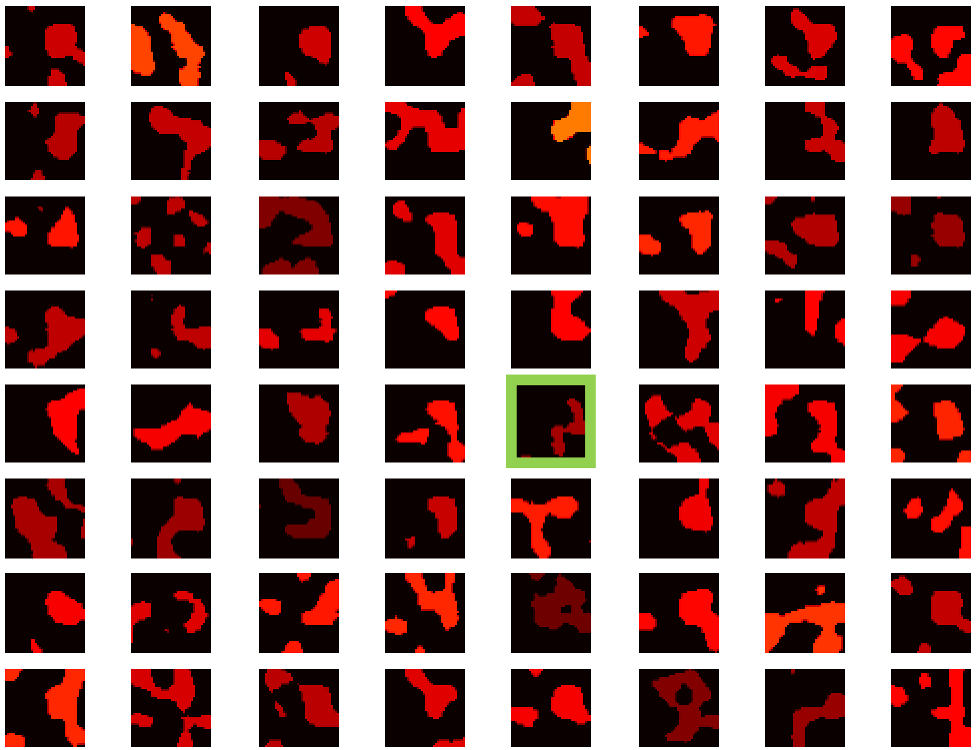


Scores

Attention map

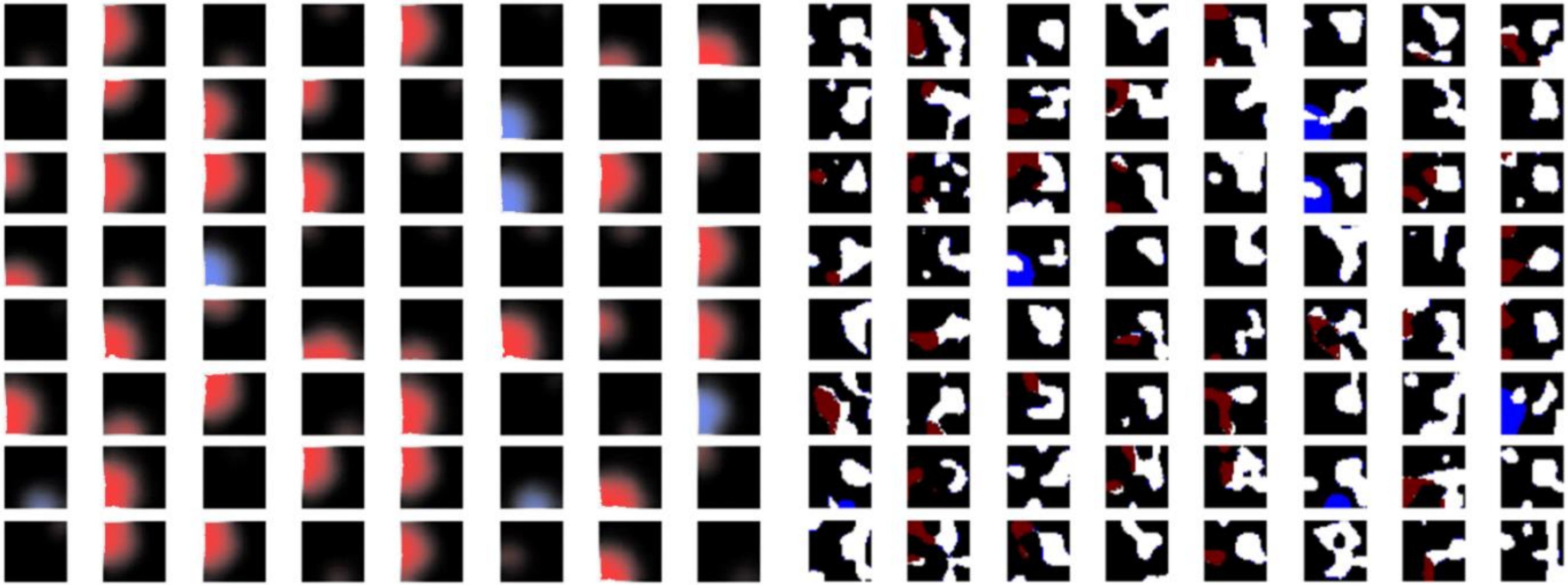


Attention sample 1



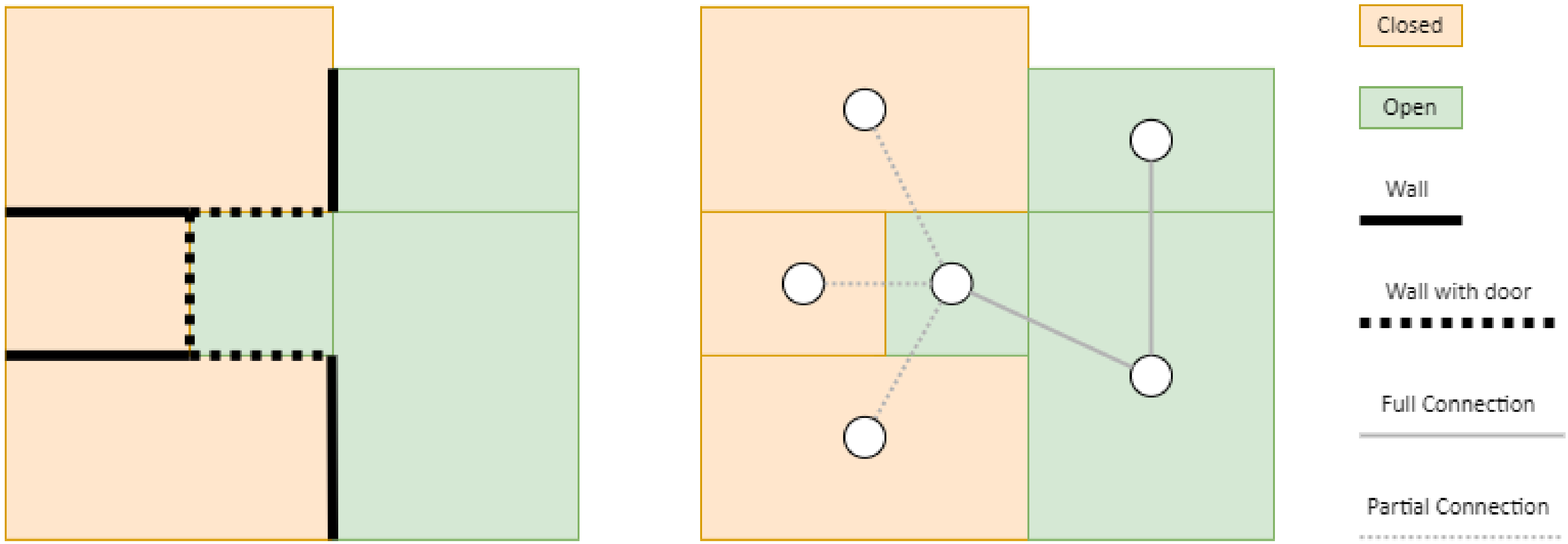
Attention sample 45

Alterations

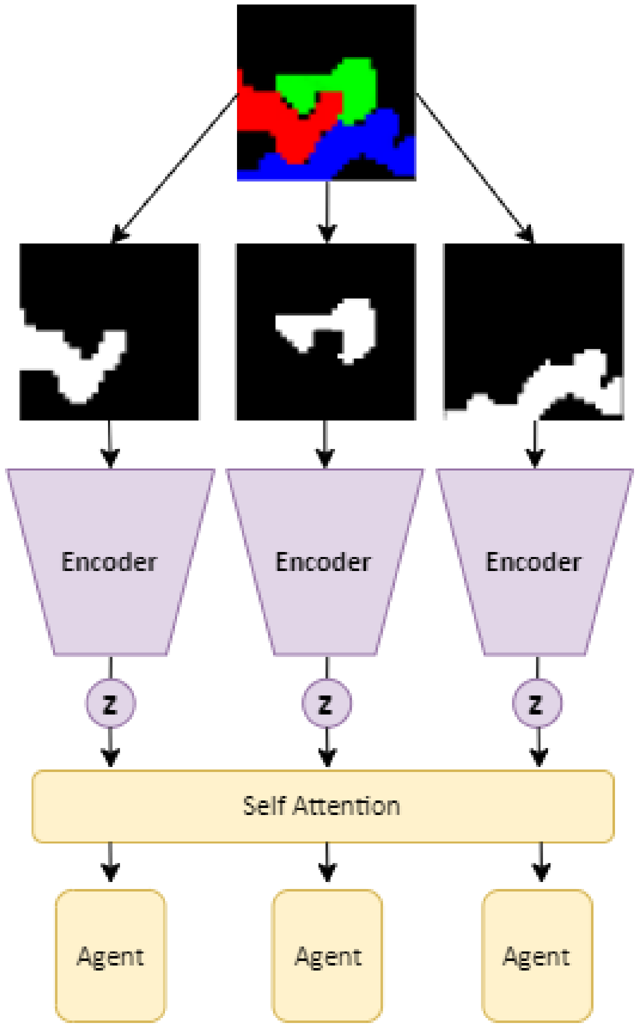


Proposed change **Result**

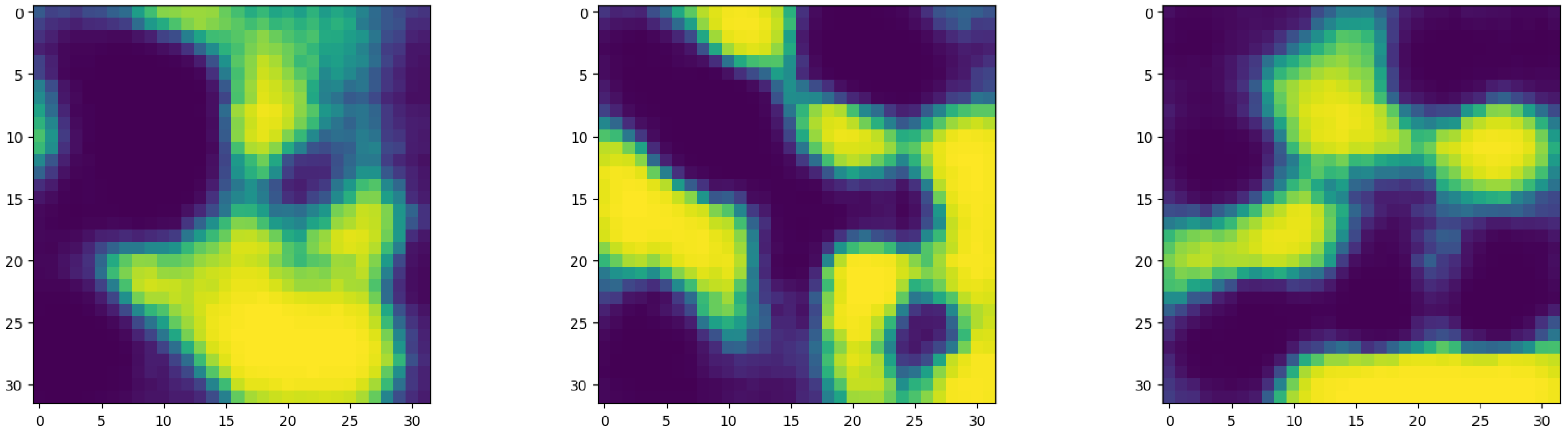
What were we doing again?!?!



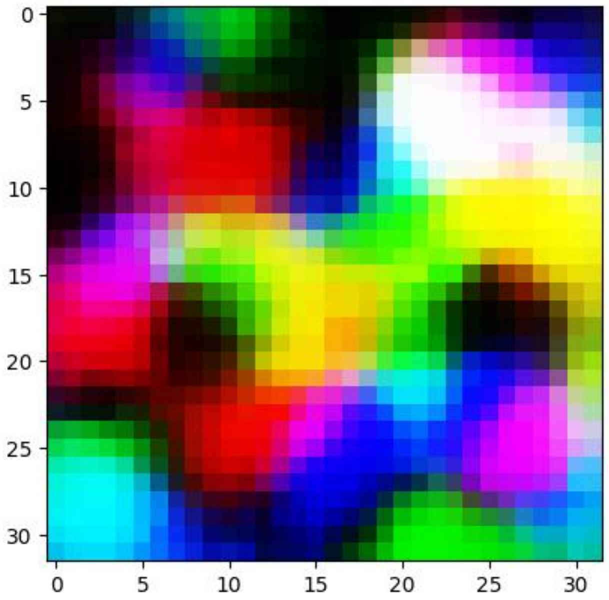
Application to samples with multiple channels



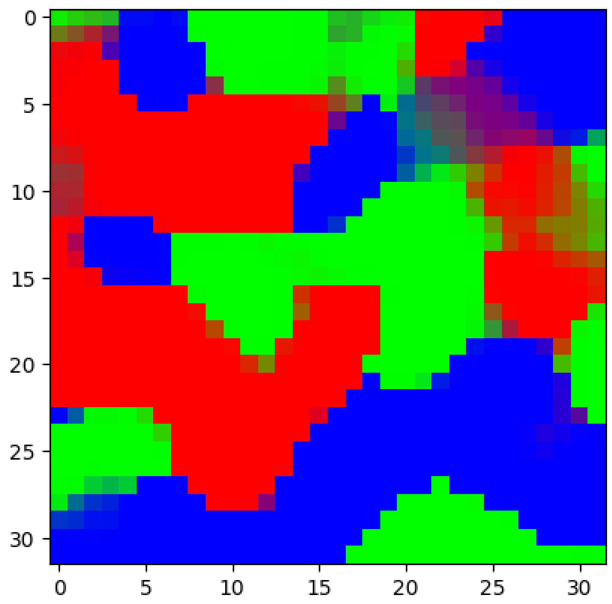
Constructing floorplans from samples



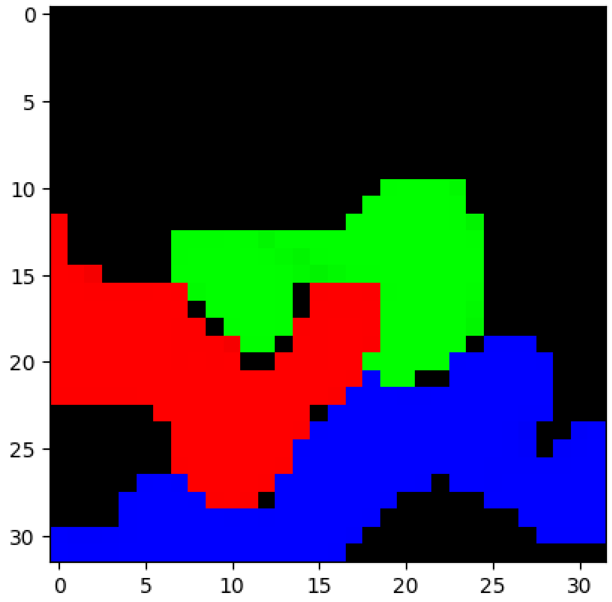
Concatenate



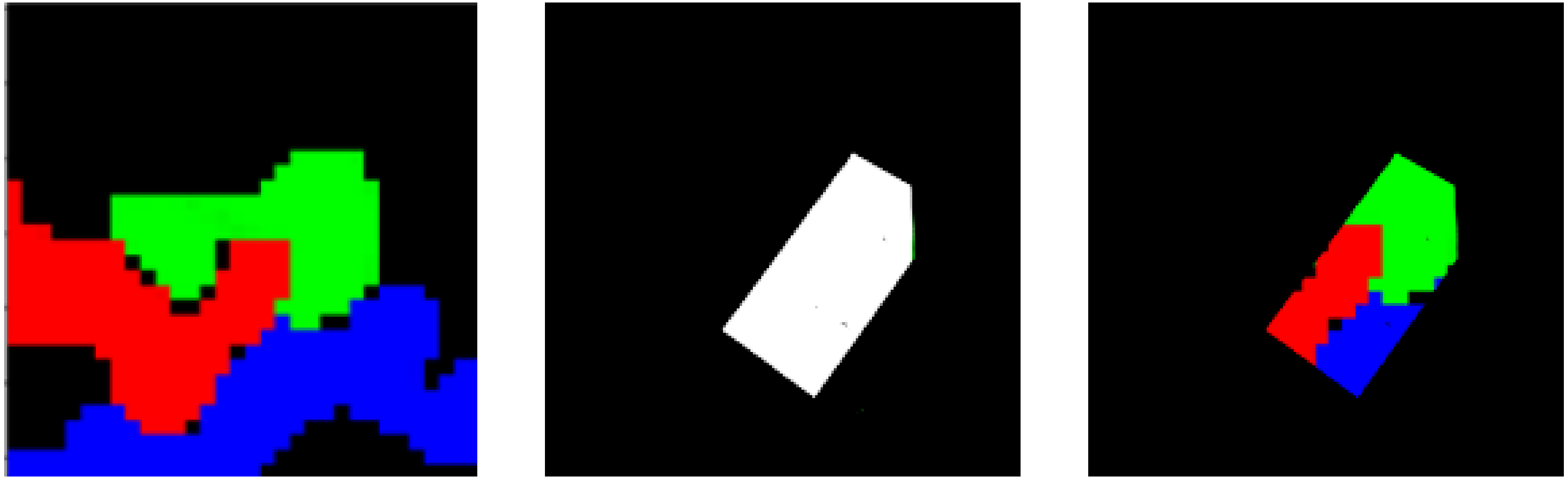
Softmax



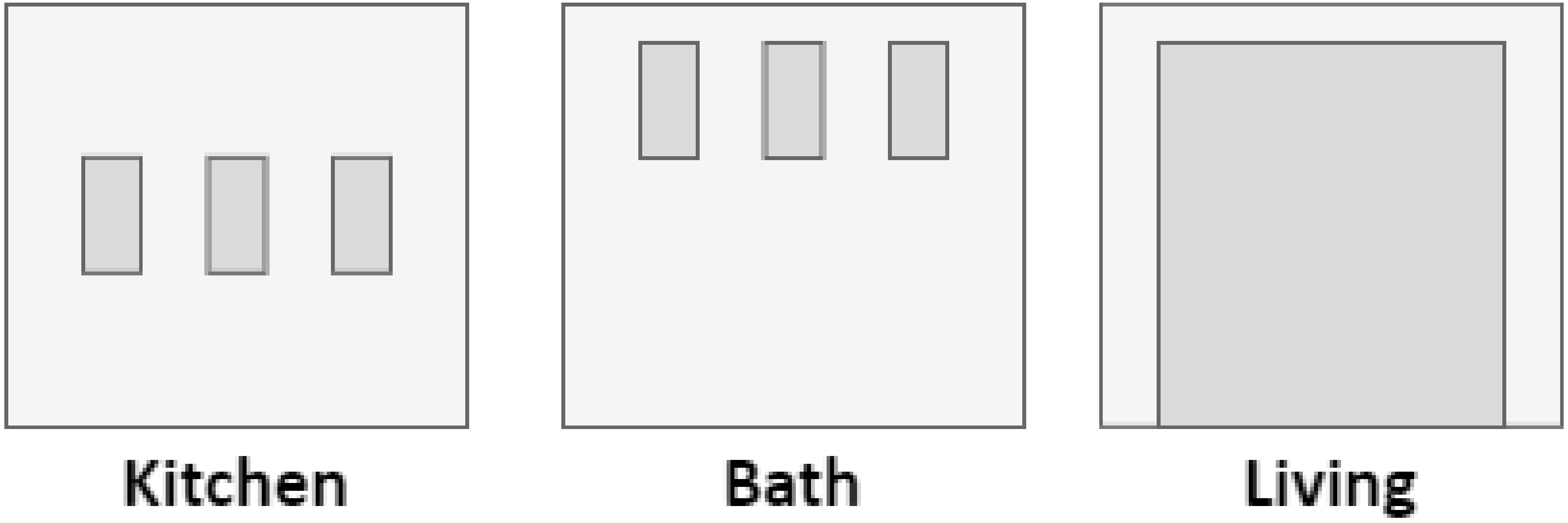
Prioritize large coherent areas



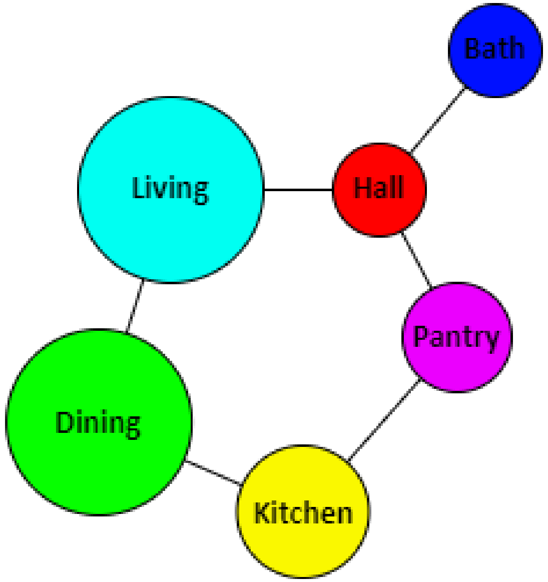
Optional mask



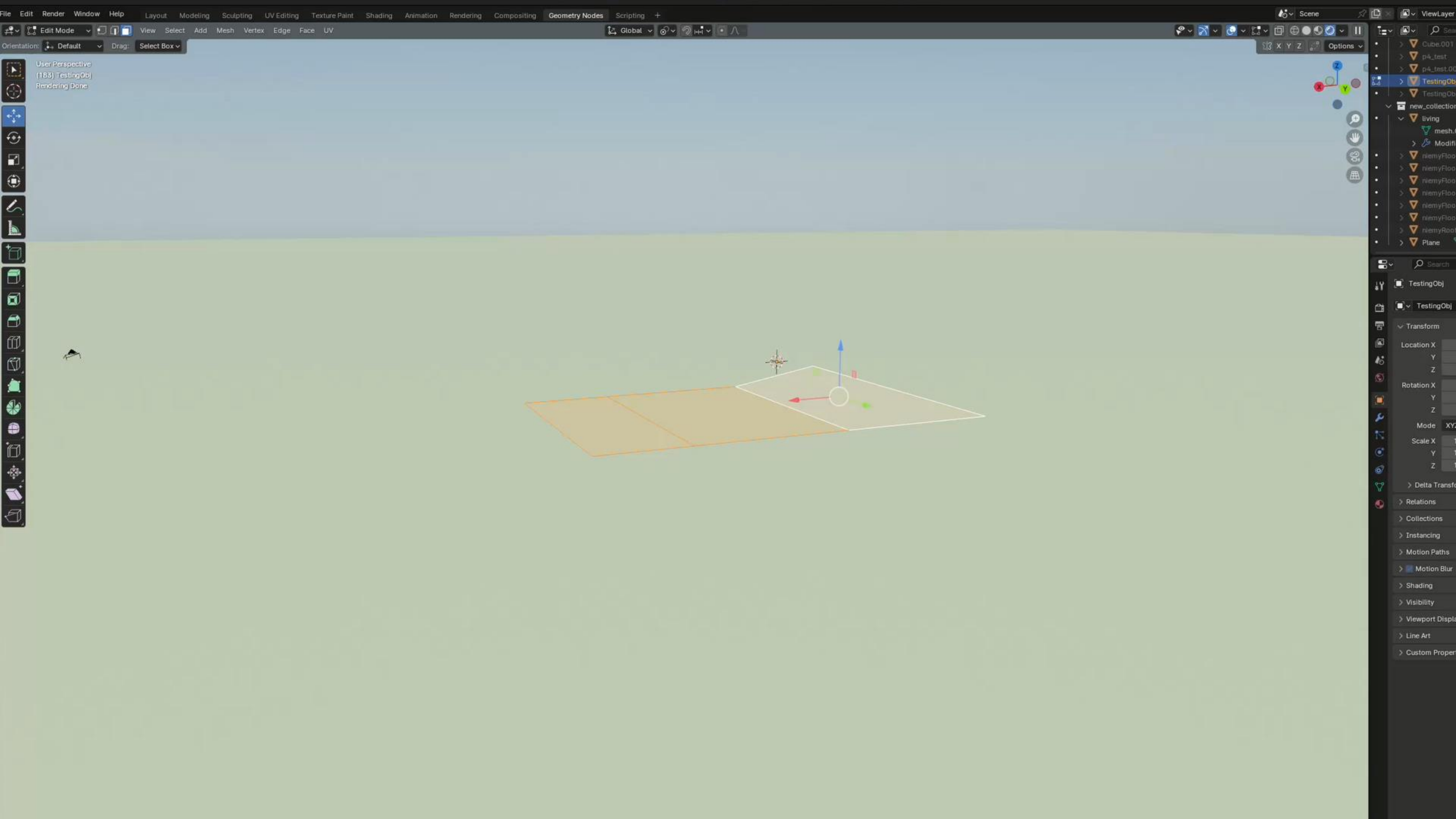
Assigning a suitable façade.



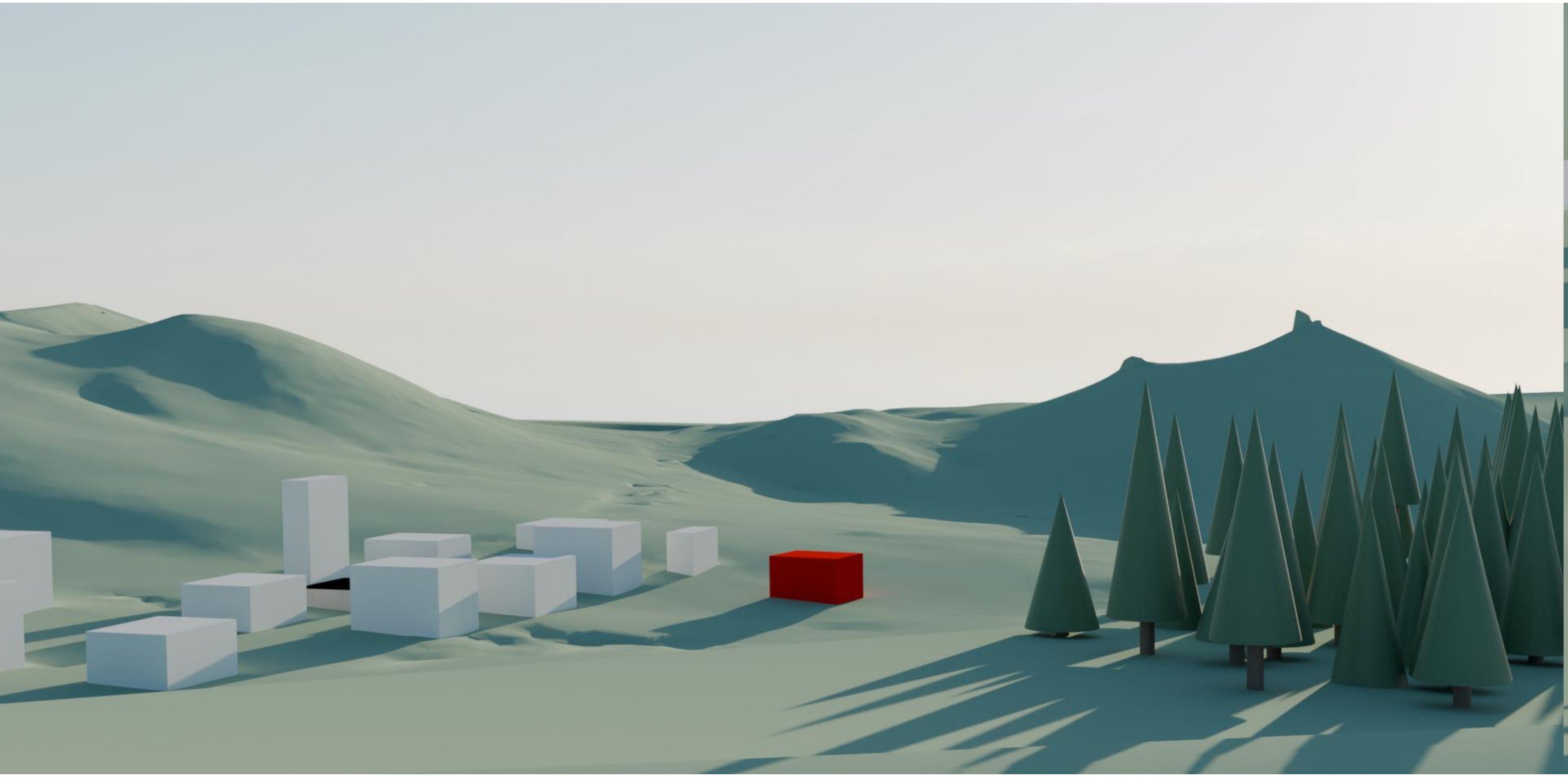
Let us design a small building.

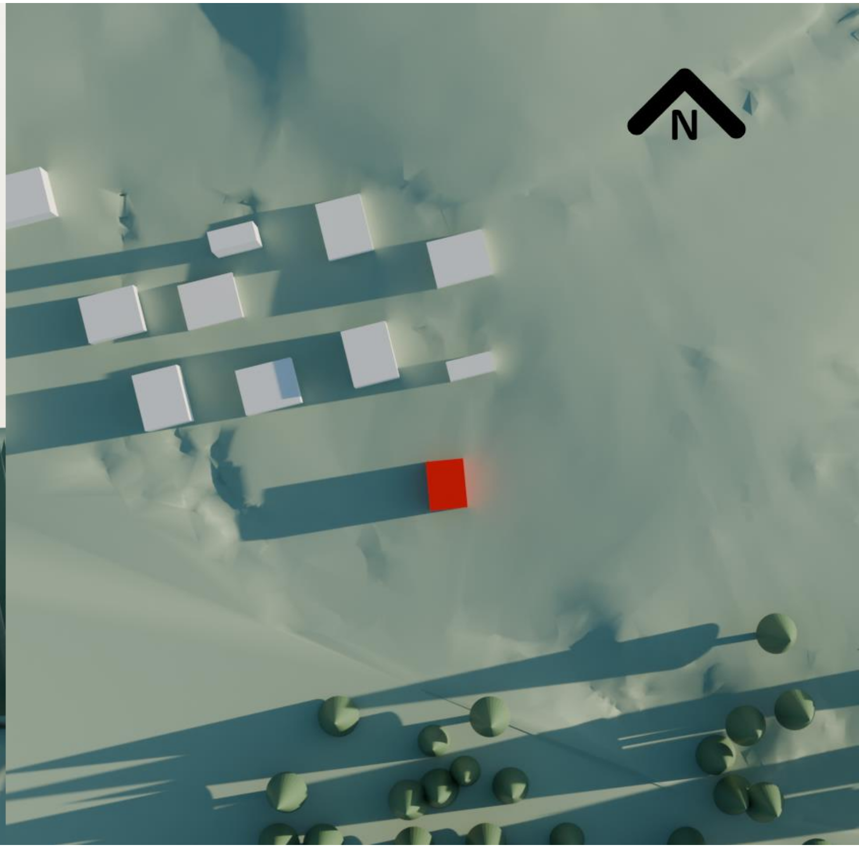


	10			1	1	1
		12	1			1
		1	28	1		
	1		1	28		
	1				2	
	1	1				10

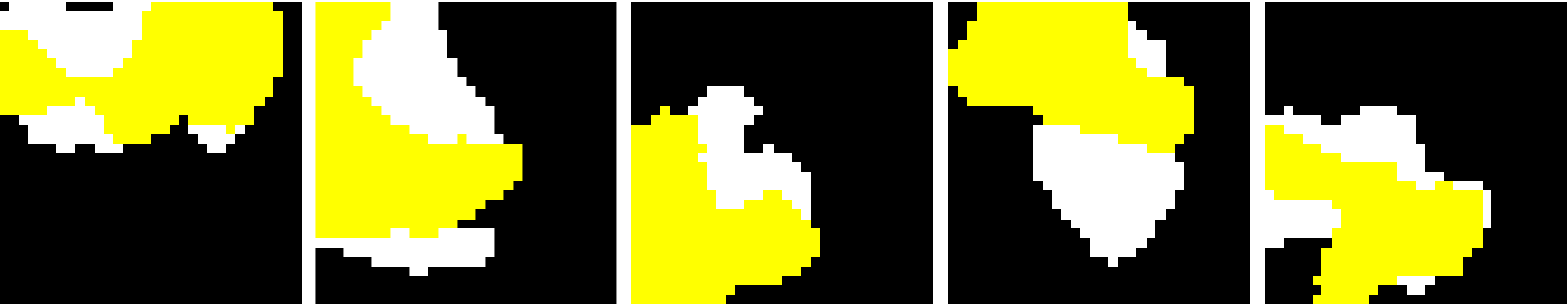

File Edit Render Window Help
Layout Modeling Sculpting UV Editing Texture Paint Shading Animation Rendering Compositing Geometry Nodes Scripting
Scene
ViewLayer
Edit Mode
View Select Add Mesh Vertex Edge Face UV
Global
Orientation: Default
Drag: Select Box
Options
User Perspective
(183) TestingObj
Rendering Done
Cube.001
p4_test
TestingOb
new_collection
living
Plane
TestingObj
Transform
Location X
Rotation X
Mode
Scale X
Relations
Collections
Instancing
Motion Paths
Motion Blur
Shading
Visibility
Line Art

What can it do without user input?

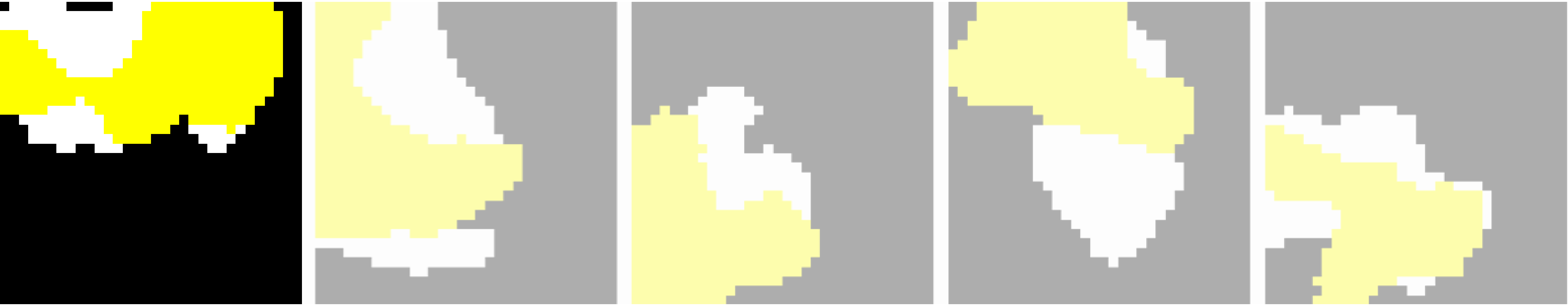




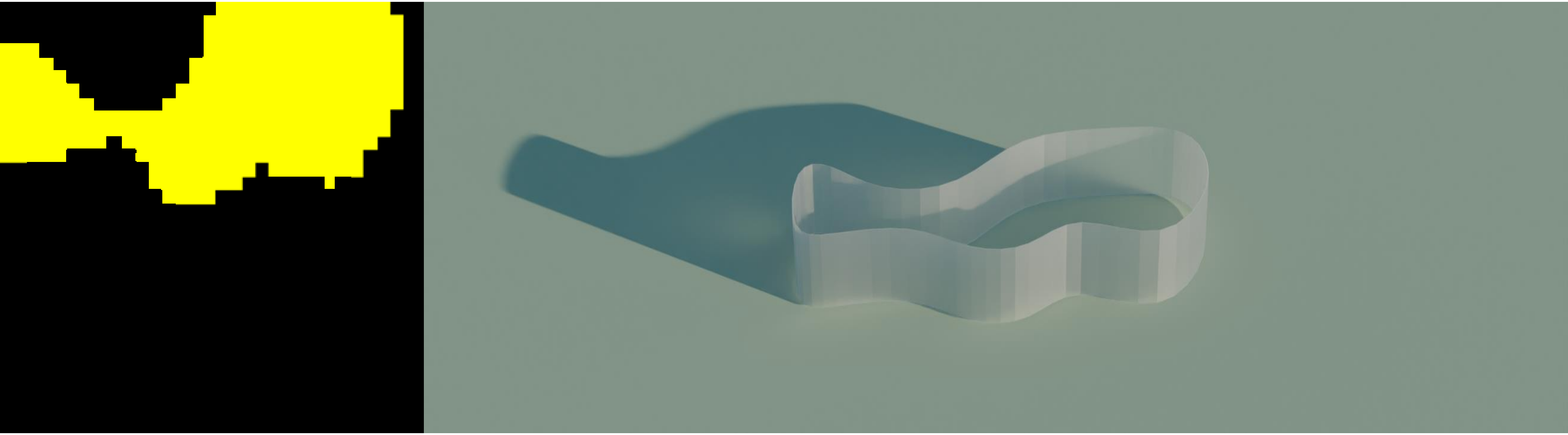
Generating building geometries.



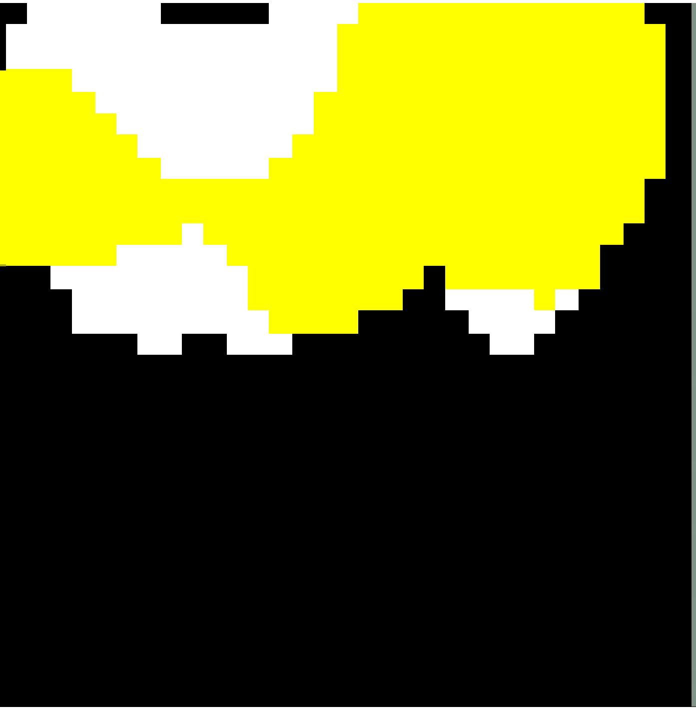
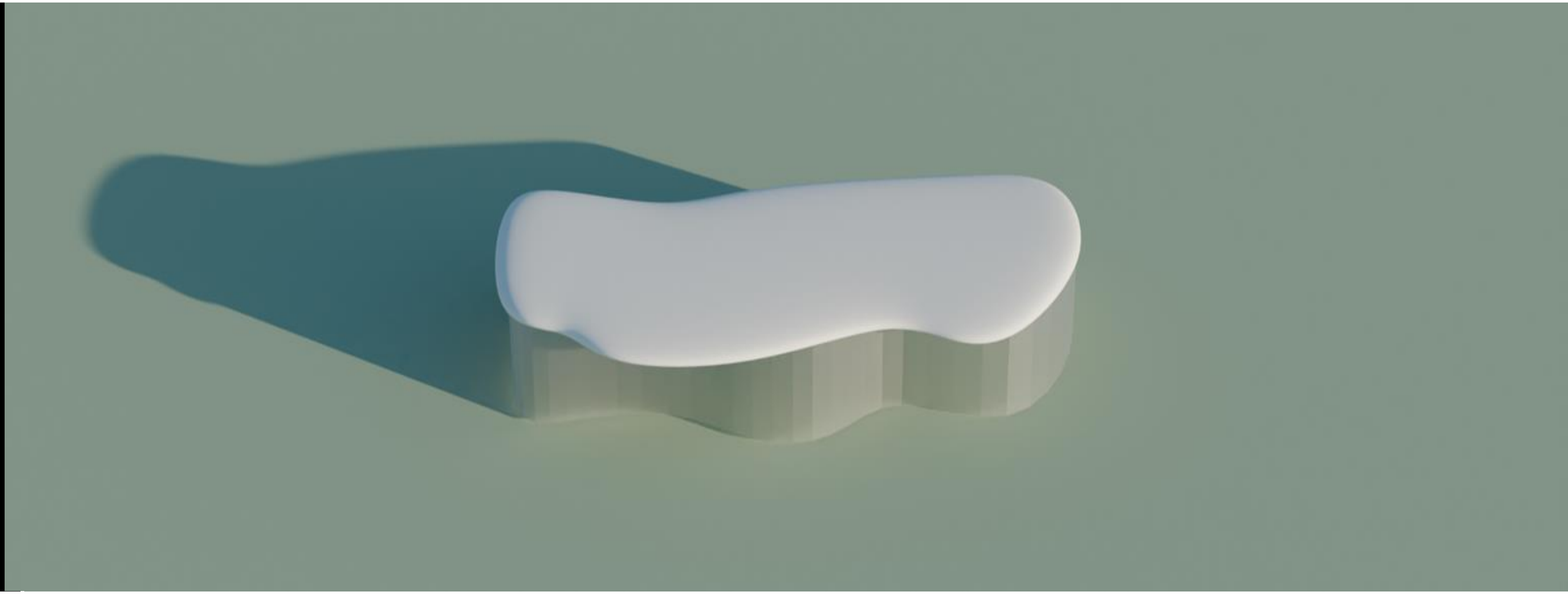
Generating building geometries.



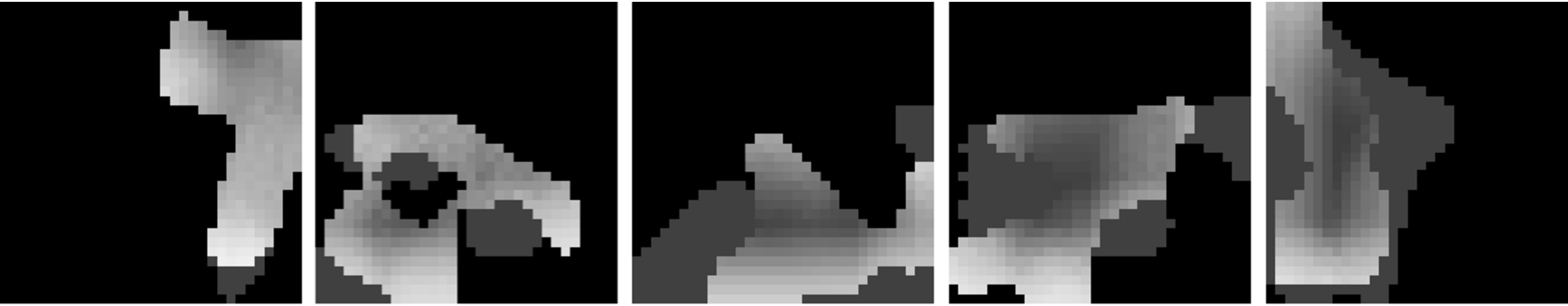
Floorspace in yellow.



Roofs in white.

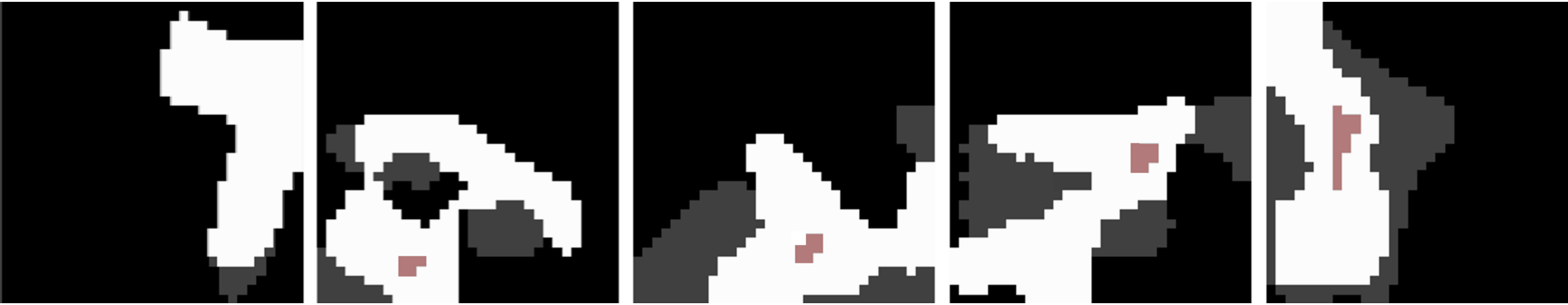



Generate solutions for daylight.

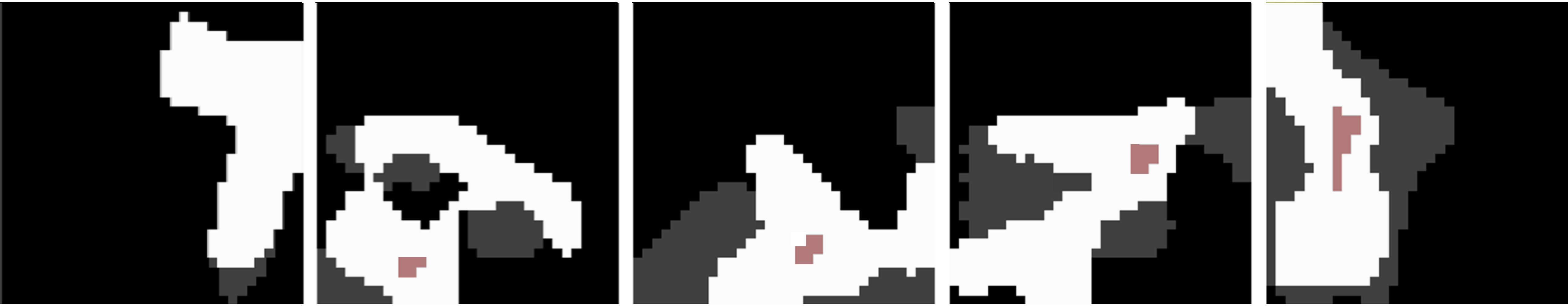


0.62 0.52 0.41 0.38 0.3

Insufficient lighting

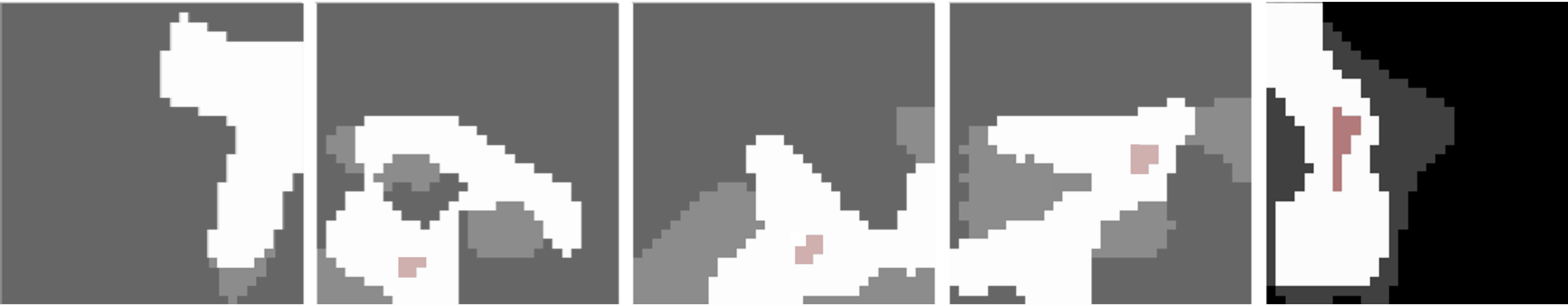


Trade-off.



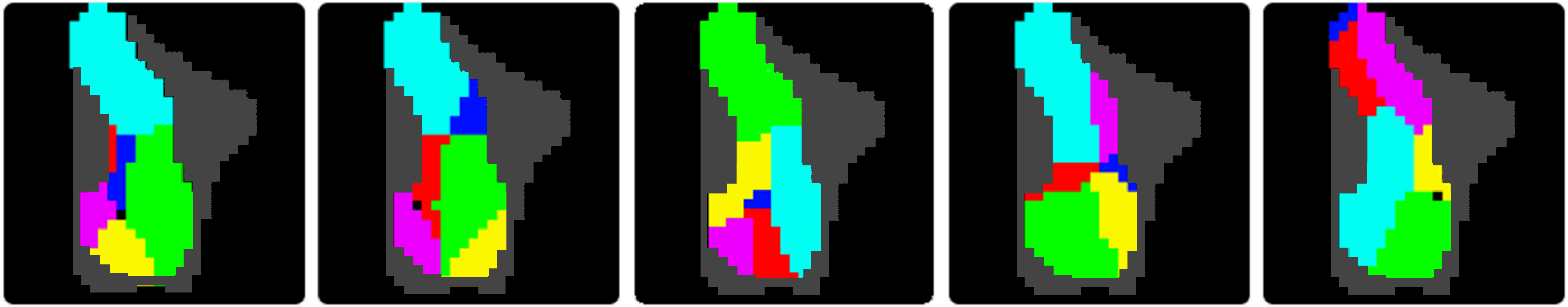
0.62 0.52 0.41 0.38 0.3

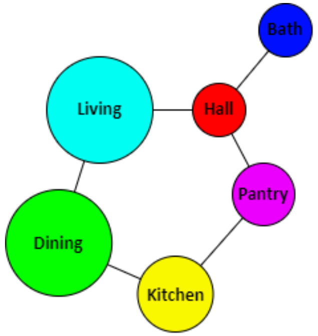
Trade-off.



0.62 0.52 0.41 0.38 **0.3**

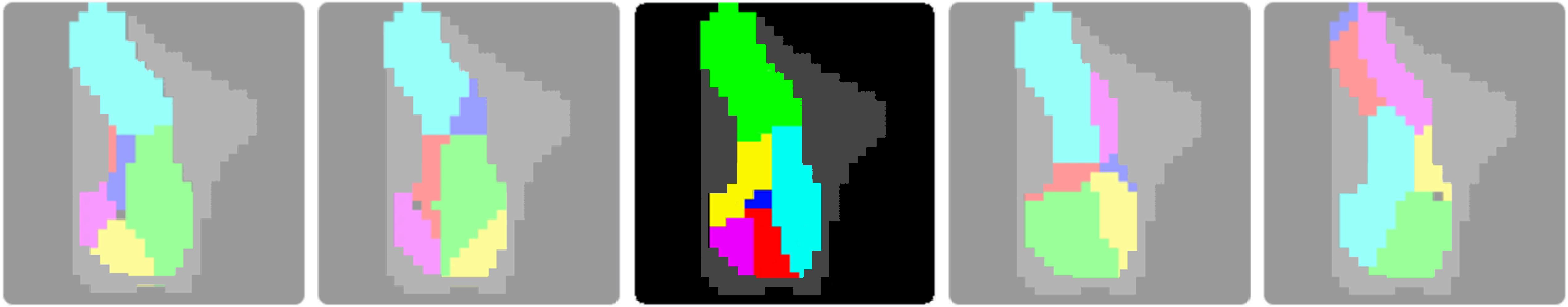
Generate five floorplans.

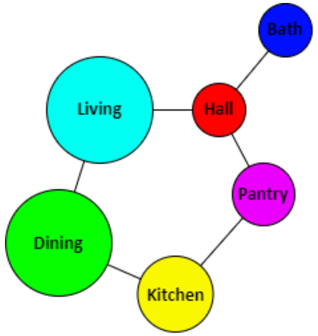




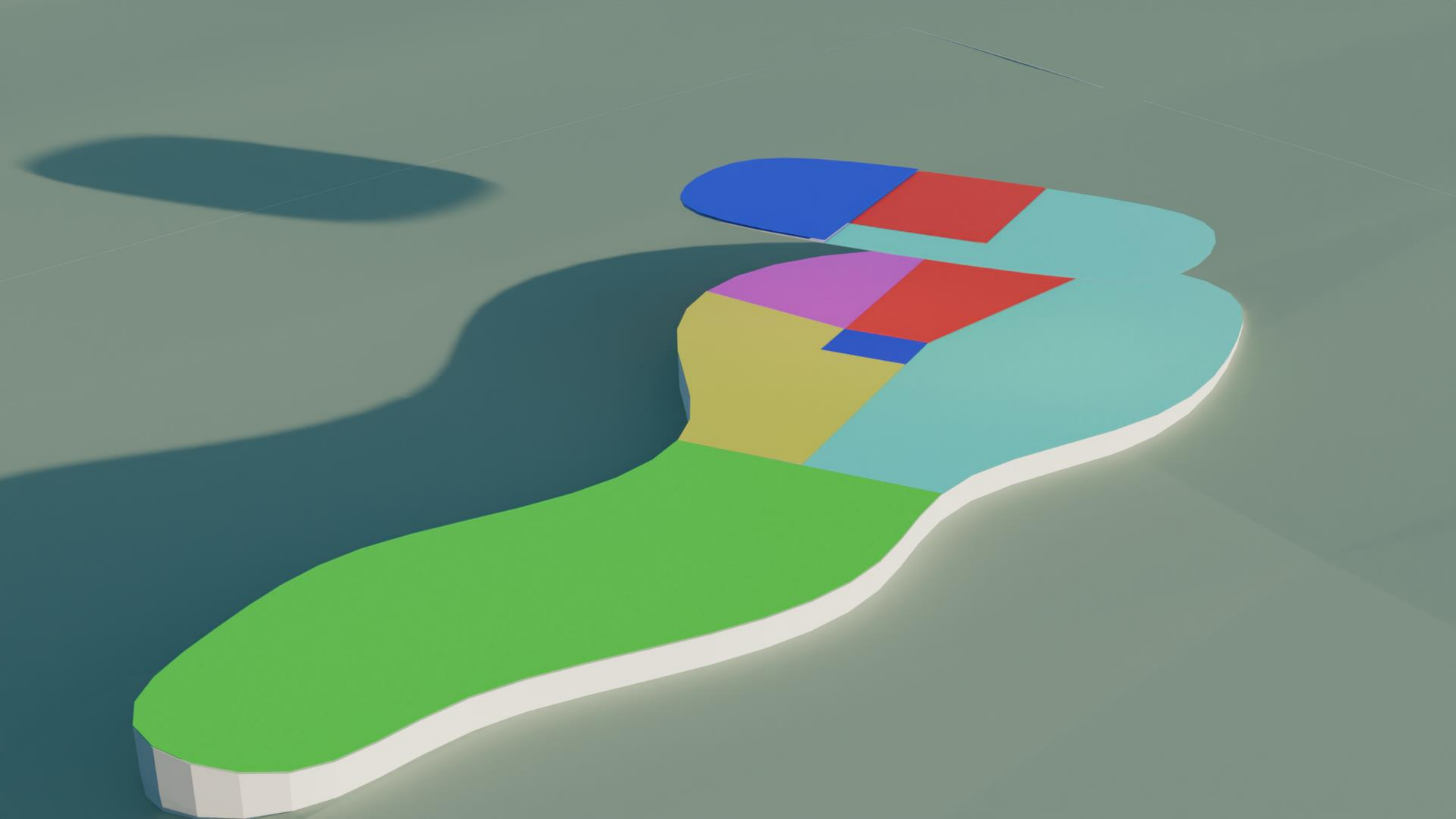
	10			1	1	1
		12	1			1
		1	28	1		
	1		1	28		
	1				2	
	1	1				10

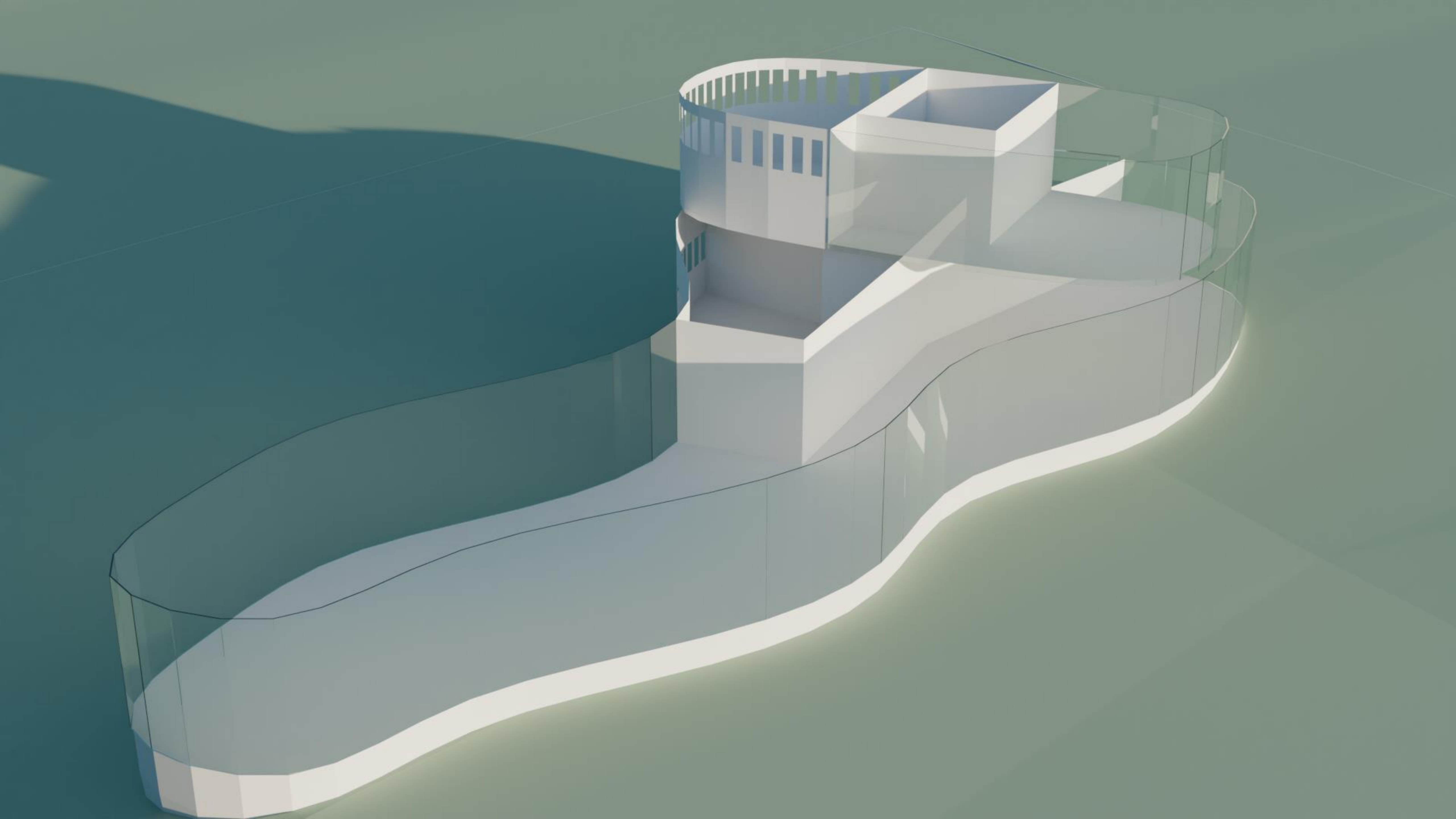
This one is nice.

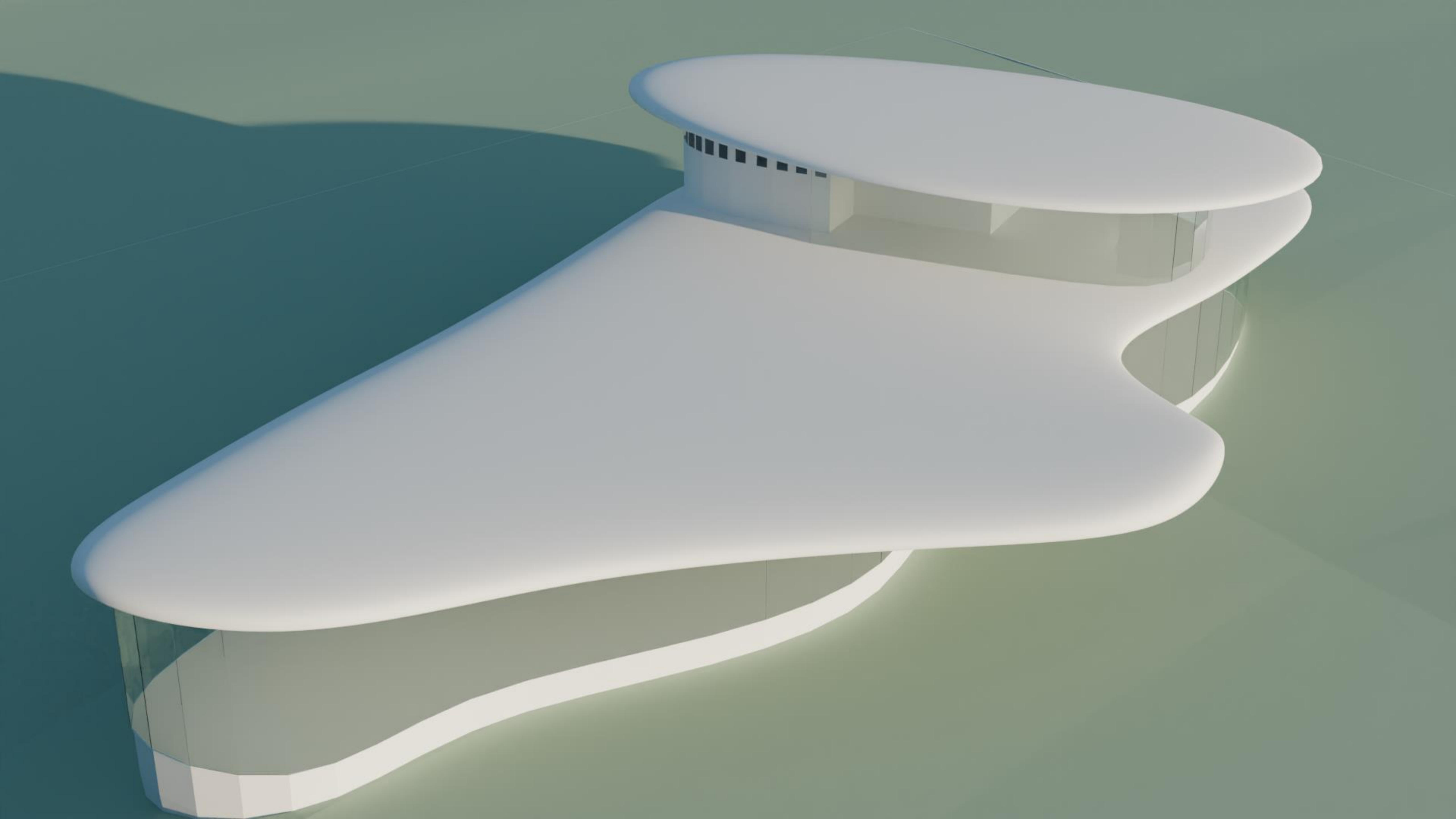


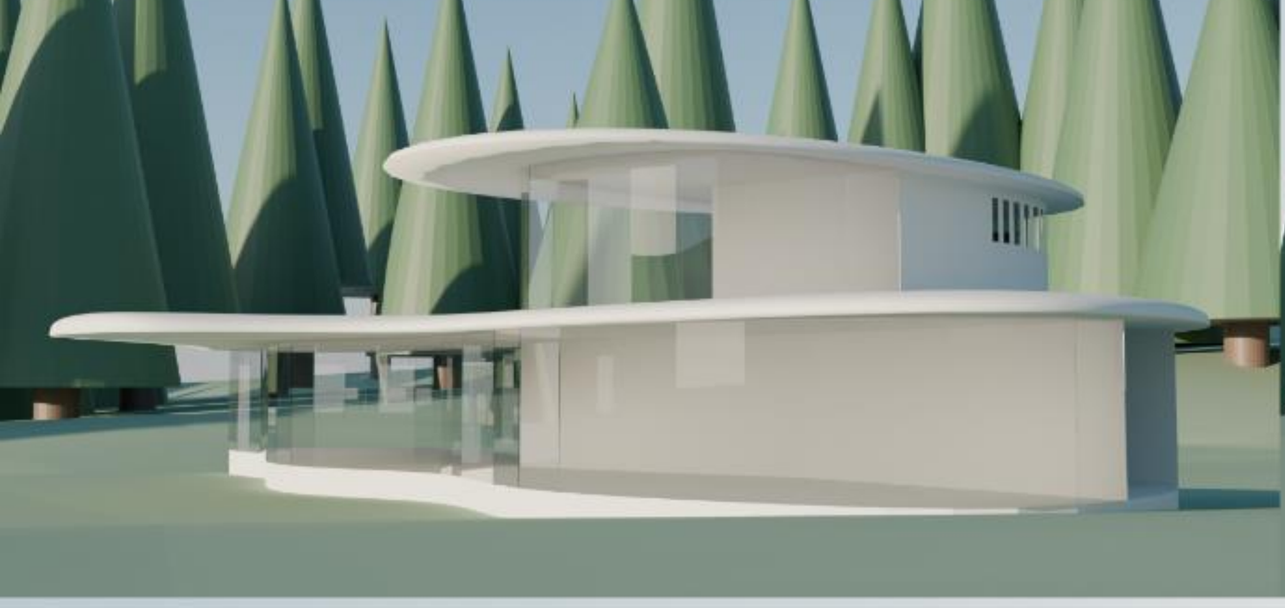


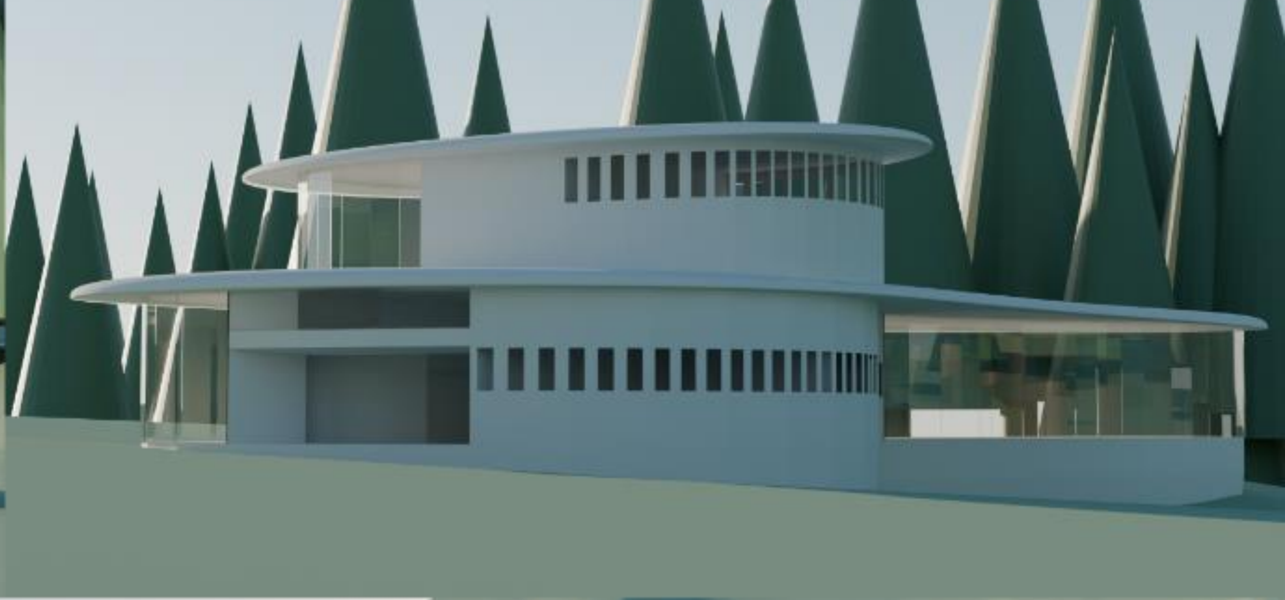
	10			1	1	1
		12	1			1
		1	28	1		
	1		1	28		
	1				2	
	1	1				10

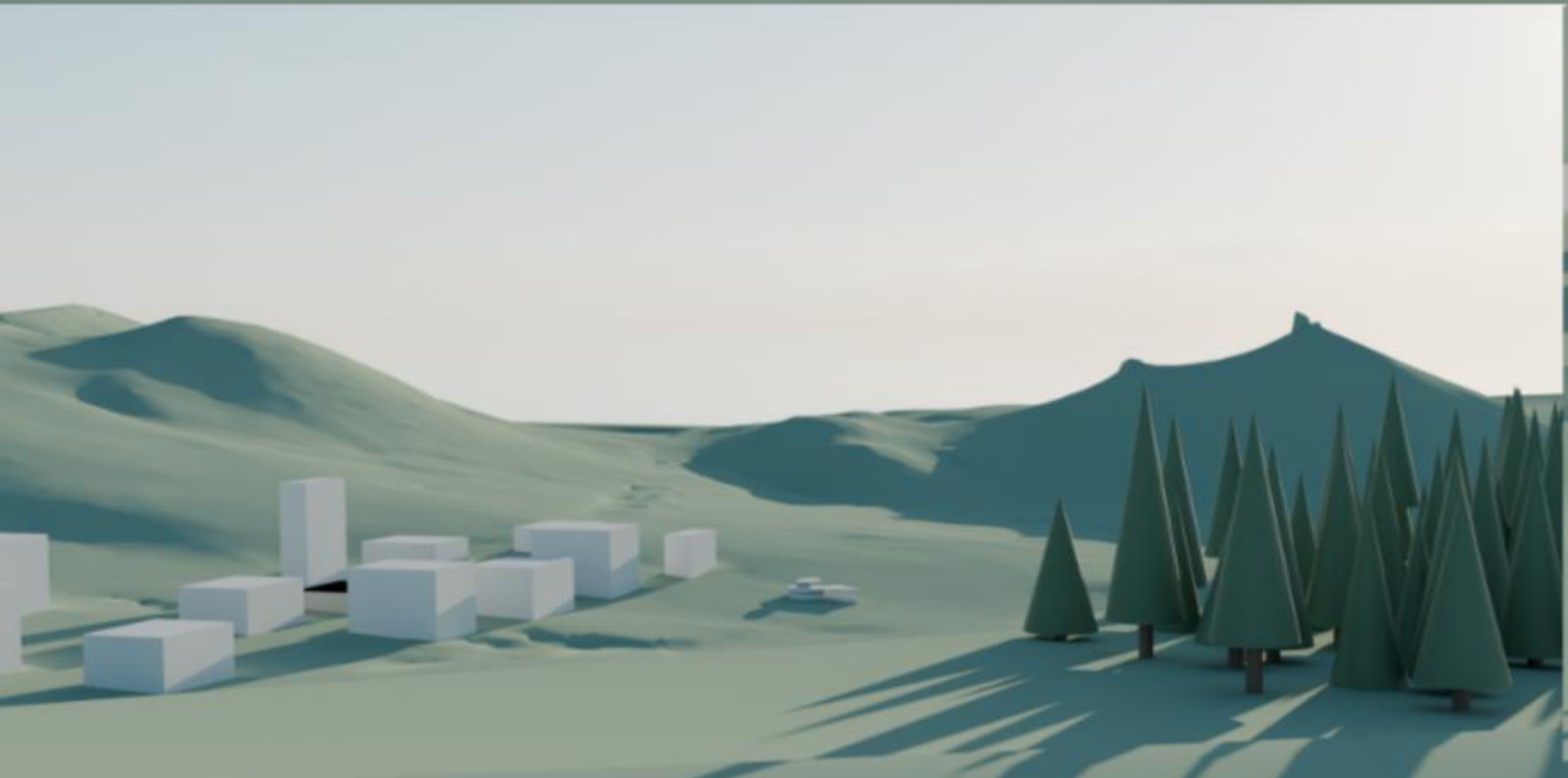


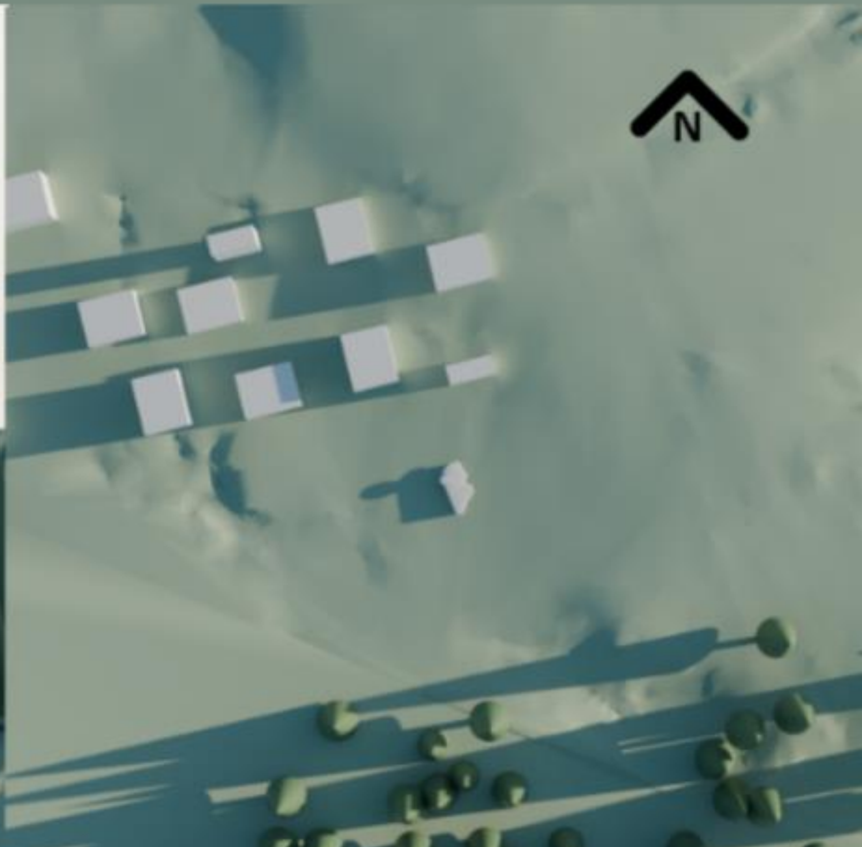




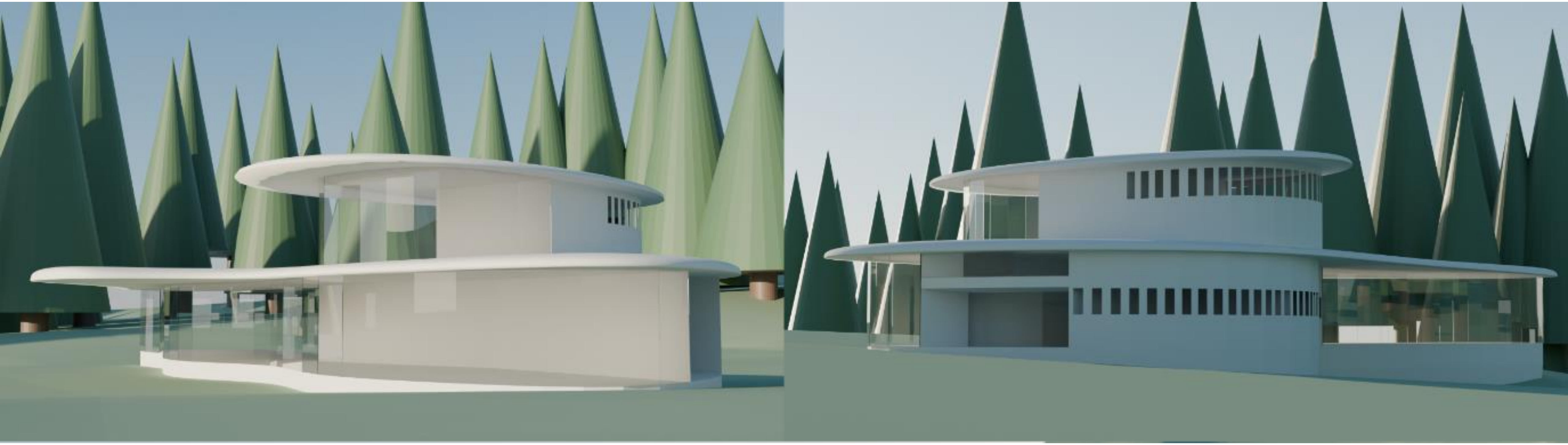






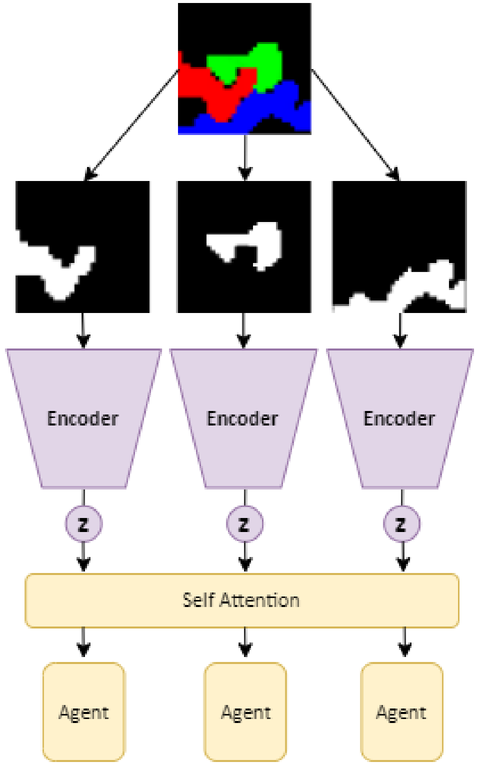

N

Why stop there?

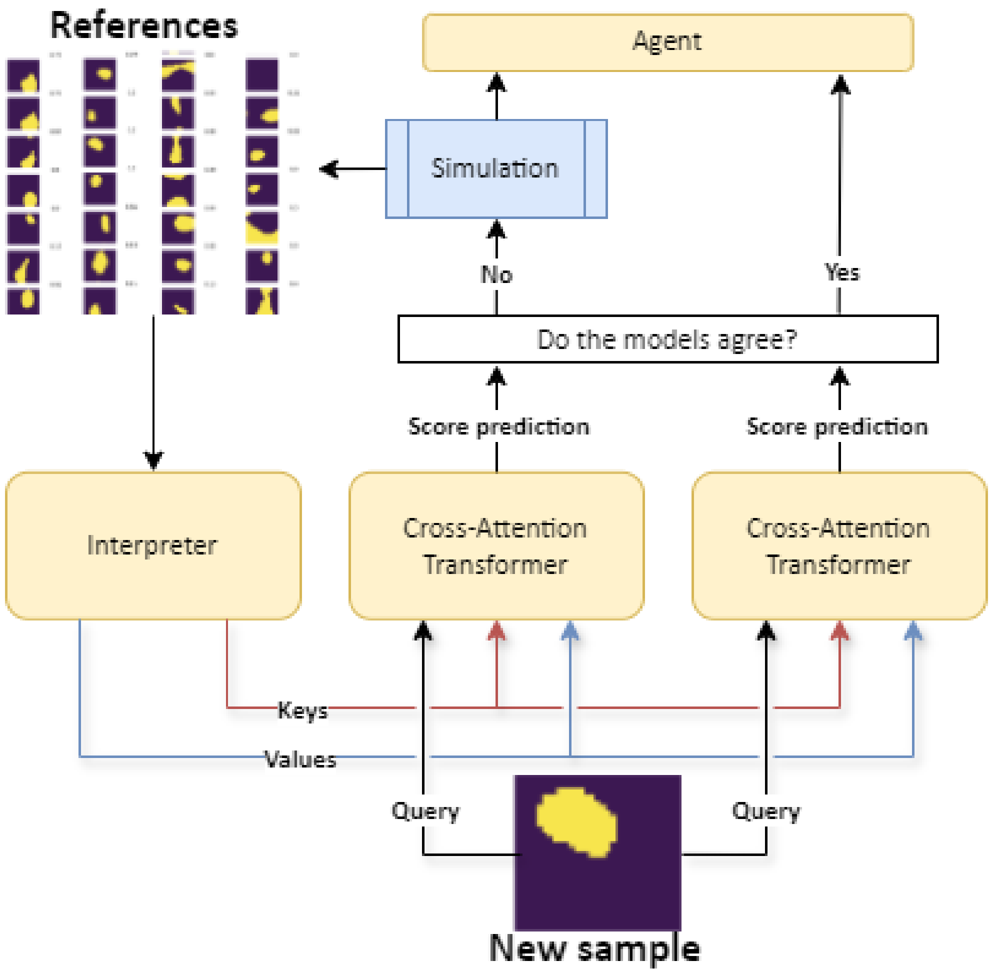


Thanks, Diederik!

Future work.


Encoder
Encoder
Encoder
z
z
z
Self Attention
Agent
Agent
Agent

Future work.


References
Agent
Simulation
No
Yes
Do the models agree?
Score prediction
Score prediction
Interpreter
Cross-Attention Transformer
Cross-Attention Transformer
Keys
Values
Query
Query
New sample

Thank you!