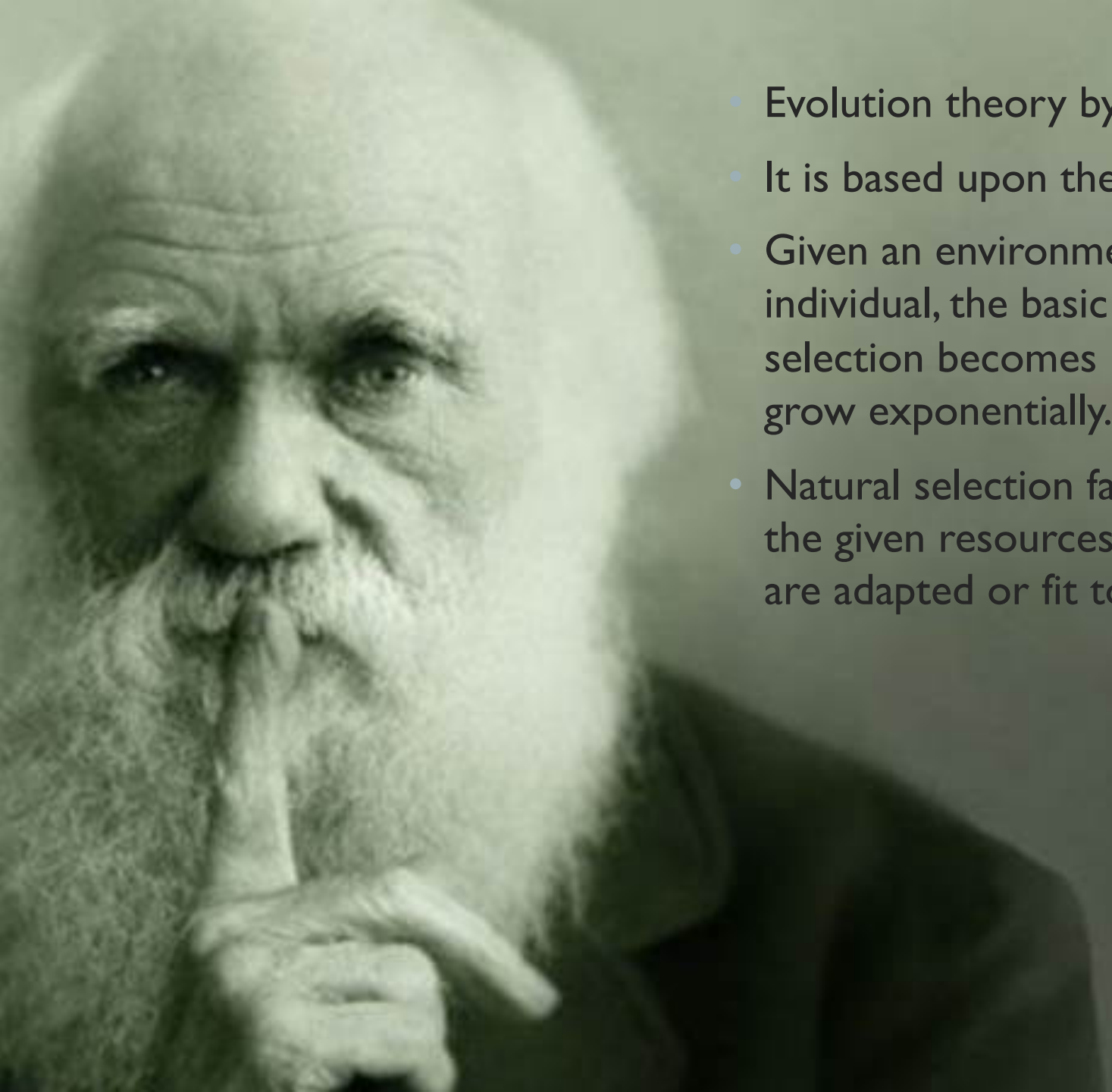
POLITECNICO
MILANO 1863

IMAGE AND SOUND
ISPG
PROCESSING GROUP

CREATIVE PROGRAMMING AND COMPUTING

Evolutionary creative systems



- Evolution theory by Charles Darwin
- It is based upon the idea of natural selection.
- Given an environment that can host only a limited number of individual, the basic instinct of individuals to reproduce, selection becomes inevitable if the population size is not to grow exponentially.
- Natural selection favours those individuals that compete for the given resources most effectively, in other words, those that are adapted or fit to the environmental conditions best.

EVOLUTION THEORY

- Darwinian evolutionary theory is based on three main principles:
  - **Heredity:** If creatures live long enough to reproduce, then their traits are passed down to their children in the next generation of creatures
  - **Variation**: There must be a variety of traits present in the population : find a way to introduce variation (evolution)
  - **Selection:** There must be a mechanism by which some members of a population have the opportunity to be parents and pass down their genetic information and some do not. This is typically referred to as "survival of the fittest." - *The faster gazelles are more likely to escape the lions and are therefore more likely to live longer and have a chance to reproduce and pass their genes down to their children.*

EVOLUTIONARY COMPUTING

- The evolutionary process is a good way to produce new solutions
- This make it suitable:
  - To explore the solution space of a problem in a non deterministic way
  - For creative applications

- The idea to use Darwinian principles for automated problem solving was conceived as early as 1948 by Turing. By the 1960's the first computer experiment had been executed on "optimization through evolution and recombination"

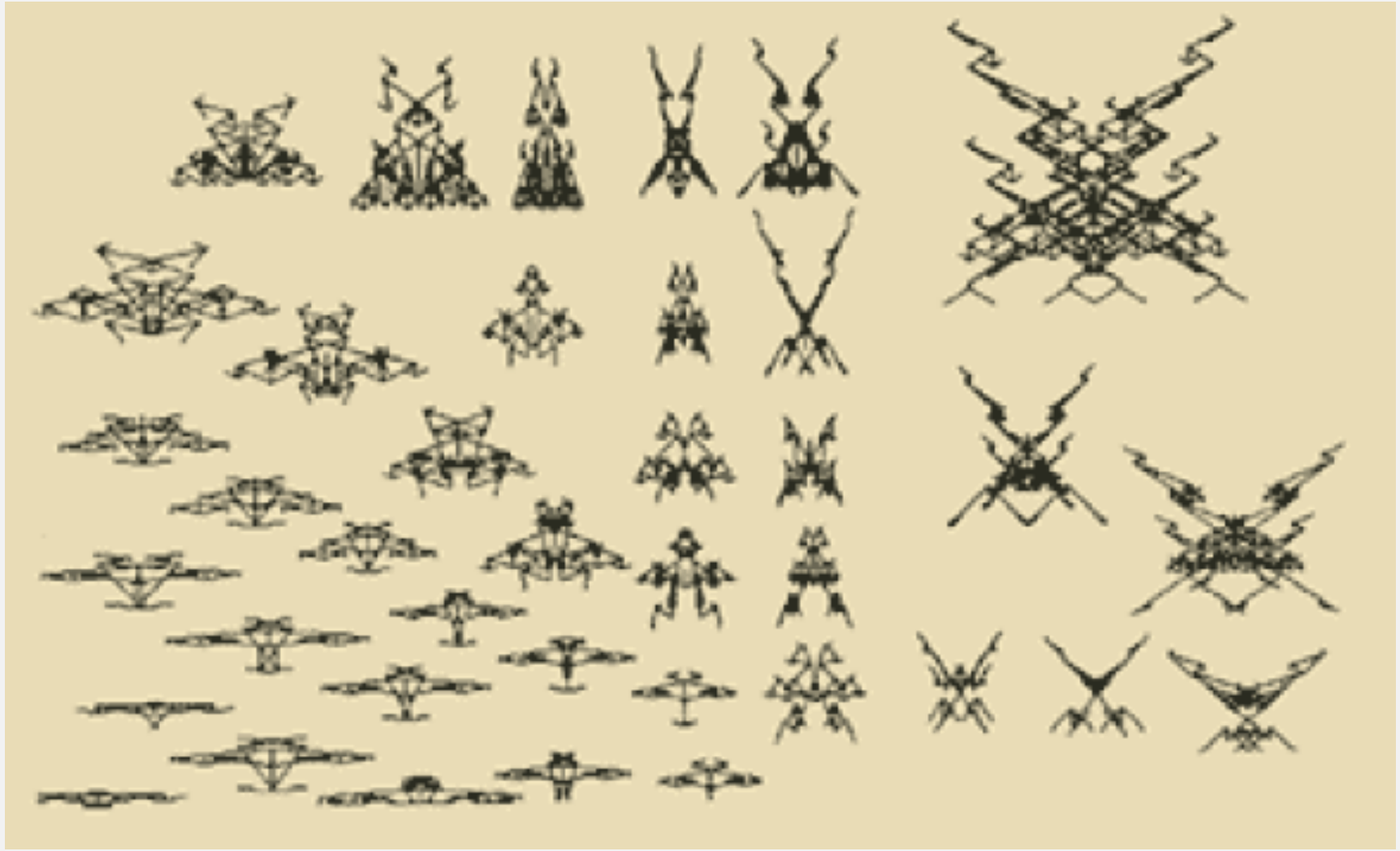
- In modern day:
  - **Evolutionary computing** is a family of algorithms for global optimization by applying evolutional theory: heredity, variation, selection

EVOLUTIONARY ART

- Evolutionary art is part of Generative art
- The first ides of Evolutionary art has been around even since the time of Wolfgang Amadeus Mozart.
- In his work "**Musikalisches Würfelspiel**" 1757 dice were used to select musical sequences from a numbered pool of previously composed phrases

- The evolutionary art started with the **Richard Dawkins** book **The blind Watchmaker**, published in 1986, first mentioned the idea for a program which could evolve 'virtual creatures' or biomorphs
- It uses evolutionary computing to create a drawing of all kinds of creatures after which the user indicates which he or she liked best and the program evolves a new set of creatures based on the choices of the user. The user could keep going until they found a biomorph to their liking.

EVOLUTIONARY ART



Richard Dawkins - The blind Watchmaker

EVOLUTIONARY ART

- **Evolutionary art:** the artist does not do the work of constructing the artwork, but rather lets a system do the construction. In **evolutionary art**, initially generated art is put through an iterated process of **selection** and **modification** to arrive at a final product

- The members of a population undergoing artificial evolution modify their form or behaviour over many reproductive generations in response to a selective regime

https://www.youtube.com/watch?v=x-S9iOF0uMI

- In **interactive evolutionary art** the selective regime may be applied by the viewer or the artist explicitly by selecting individuals which are aesthetically pleasing

- **The blind Watchmaker** is an example of interactive evolutionary art

GENETIC ALGORITHMS

GENETIC ALGORITHMS (GA)

- It is a metaheuristic inspired by the process of natural selection that belongs to the larger class of evolutionary algorithms (EA).
- **John Holland** introduced genetic algorithms in **1960** based on the concept of Darwin's theory of evolution; afterwards, his student **David E. Goldberg** extended GA in **1989**
- It is an random-based optimization techniques -> given a function, GA is a way to find the best solution into the solution space
- This is done using Darwin's theory of natural evolution based on Heredity, Variation, Selection

FOUNDATIONS

- **Population** (or population pool) is the set of elements that will evolve or die
- It is a set solution to the problem
- A **chromosome** (or DNA) is a single solution to the problem

E.g. Goal is **to be or not to be**

Possible population

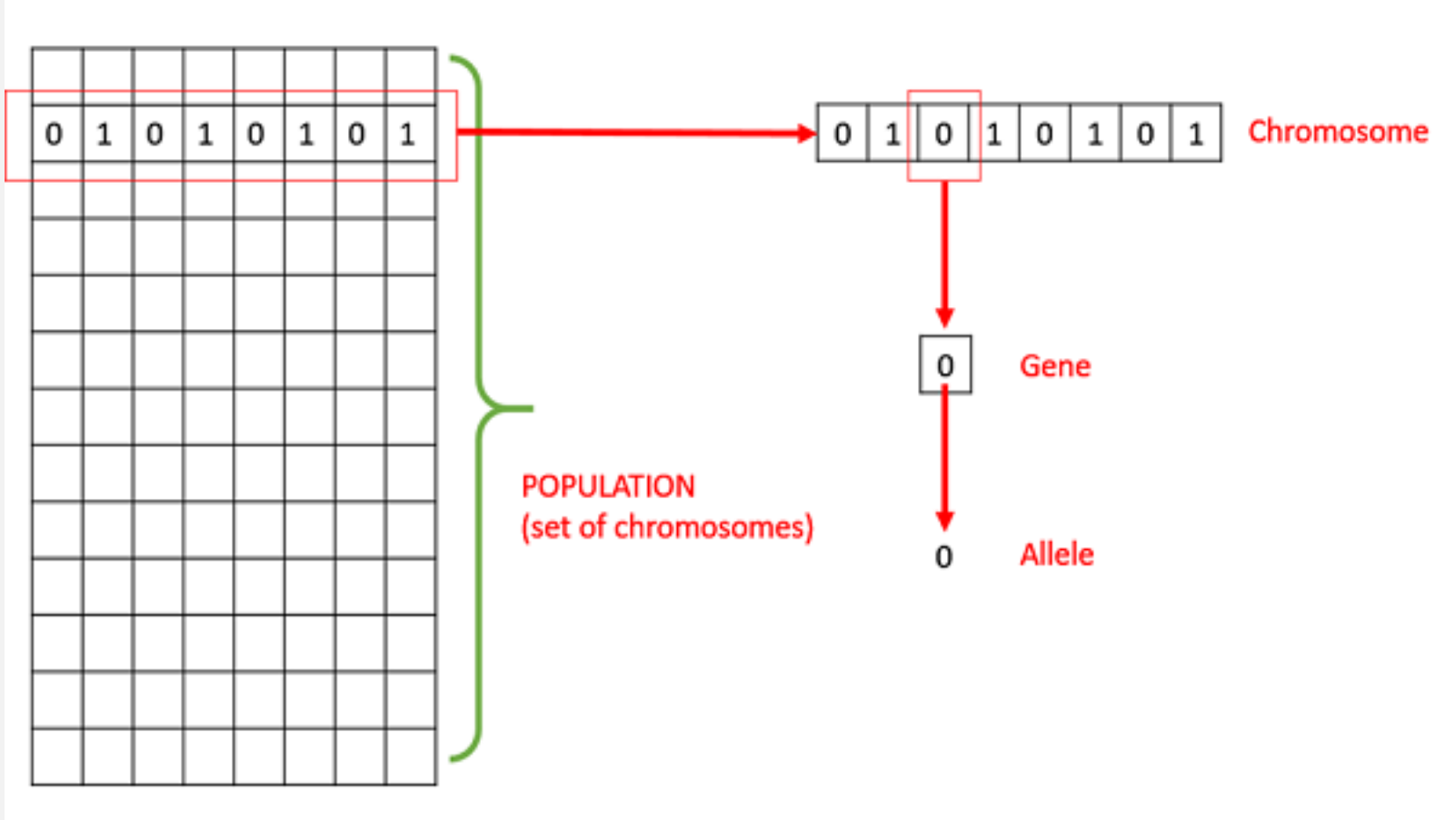
A,B,C are Chromosomes

```
A: to be or not to go
B: to be or not to pi
C: xxxxxxxxxxxxxxxxxxbe
```

- In order to guarantee **variation** the population should be large enough

FOUNDATIONS

- **Gene:** is one element position of a chromosome
- **Allele**: It is the value a gene takes for a particular chromosome.



FOUNDATIONS

- In order to guarantee to converge the set of **Alleles** should be enough expressive to represent the final solution

- Example:
  - Goal: generate the the word CAT
  - Alleles: R,T,A,M,N
  - Initial population: RAT, TAT, MAN
  - Is not power enough to generate CAT

FOUNDATIONS

- Each element in the population has a **genotype** which is the element in the computation space. The chromosomes are represented in a way which can be easily understood and manipulated using a computing systems
- Genotype is pass down from generation to generation

- Each element in the population has a **phenotype** which is the element in the actual real world solution space in which solutions are represented in a way they are represented in real world situations.

Genotype	Phenotype
int c = 255;	
int c = 127;	
int c = 0;	

FOUNDATIONS

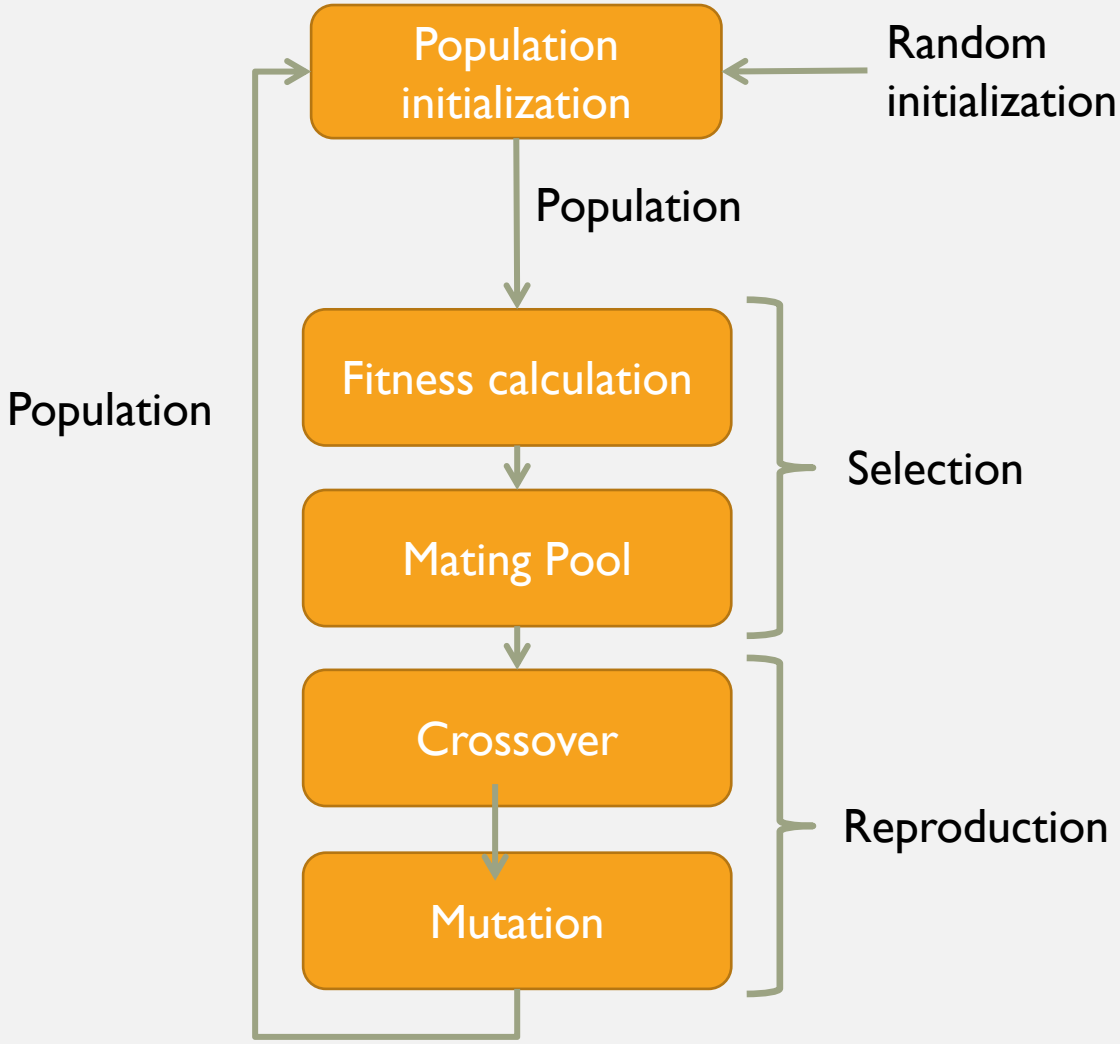
- The distinction (genotype VS Phenotype) is key on how we will use genetic algorithms:
- What are the objects in your world?
- How will we design the **genotype** for your objects **(the data structure to store each object's properties)** as well as the **phenotype (what are you using these variables to express?)**

- For simple problems, the phenotype and genotype spaces are the same

- In monkey-typing example is that there is no difference between genotype and phenotype: ***chars***

- **Fitness Function**: given a chromosome, it determines how close it is to the desired answer

FOUNDATIONS



2° STEP: SELECTION

- **Selection**: evaluate the population and determine which members are fit to be selected as parents for the next generation
- Two steps: **evaluate fitness** and **create a mating pool**

- **Evaluate fitness**
- Each element in the population is compared with the target through a **fitness function**
- One the main issue in GA is to properly design the fitness function

Example:
Goal: generate the the word CAT
In our case:-> **fitness = the number of correct characters**

DNA	Fitness
hut	1
car	2
box	0

2° STEP: SELECTION

- **Create a mating pool**
- Once the fitness has been calculated for all members of the population, we can then select which members are fit to become parents and place them in a mating pool.

- There are several different approaches:
  - **Elitist**: the two best scored chromosomes in the population will be the parents -> do not guarantee enough variation in the future population
  - **Elitist %**: keep the top 50% of the population -> the high-scoring top chromosomes would have the same chance of being selected as a parent as the ones toward the middle – risk to have the 50% most similar element in the population
  - **Probabilistic**: fitness value are normalized (range 0-1) to be transformed into probabilities. Parents are selected using the computed probability distribution

2° STEP: SELECTION

Element	Fitness	Normalized Fitness	Expressed as a Percentage
A	3	0.3	30%
B	4	0.4	40%
C	0.5	0.05	5%
D	1.5	0.15	15%
E	1	0.1	10%

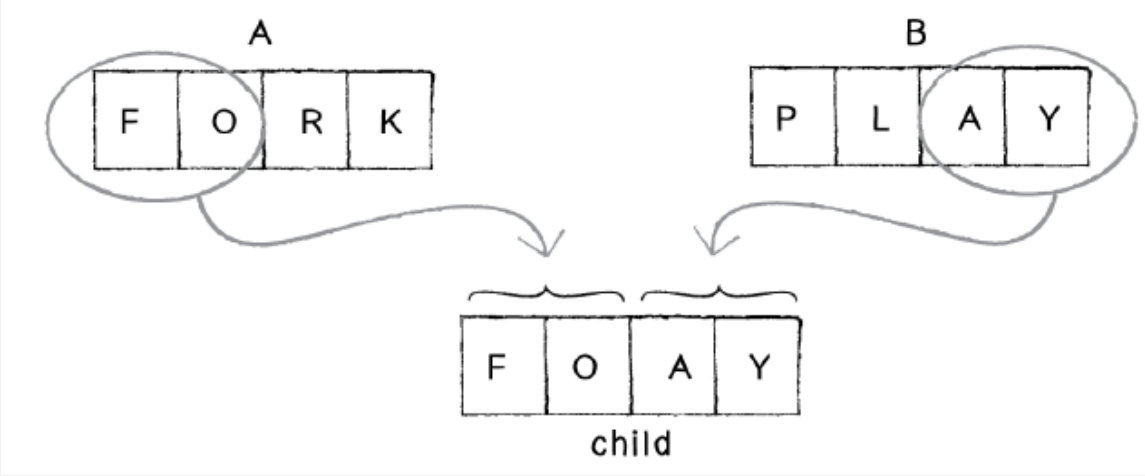
- It guarantees that the highest-scoring elements will be most likely to reproduce.
- It does not entirely eliminate any variation from the population.

3° STEP: REPRODUCTION

- New population is generated from selected elements (parents)
- Children inherit properties from their parents
- Two steps: **crossover** and **mutation**

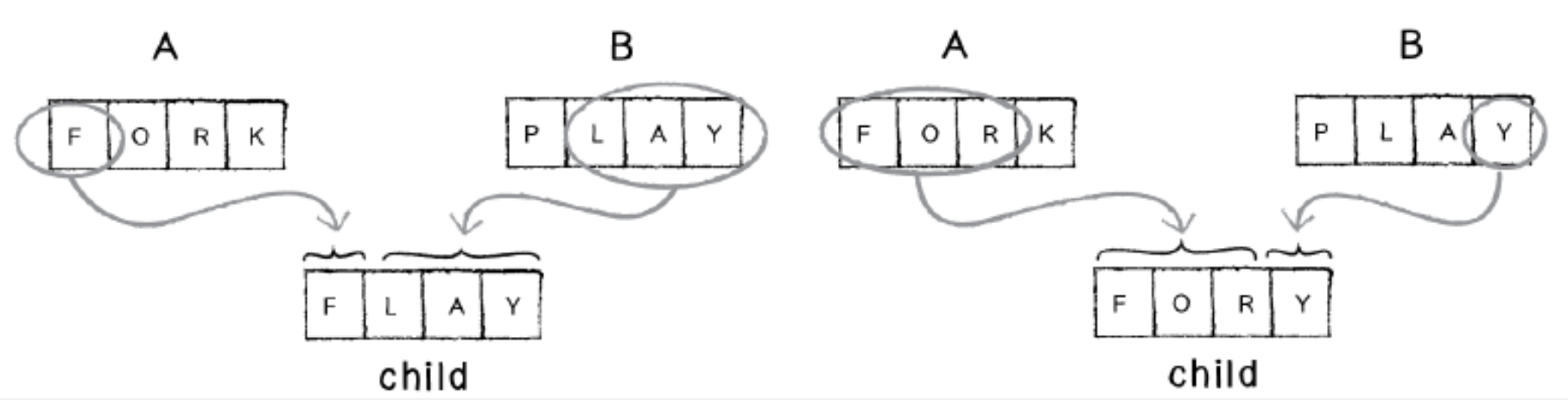
- **Crossover**
- Crossover involves creating a child out of the genetic code of two parents
- Given two parents:
  - **A: FORK**
  - **B: PLAY**

Half and half

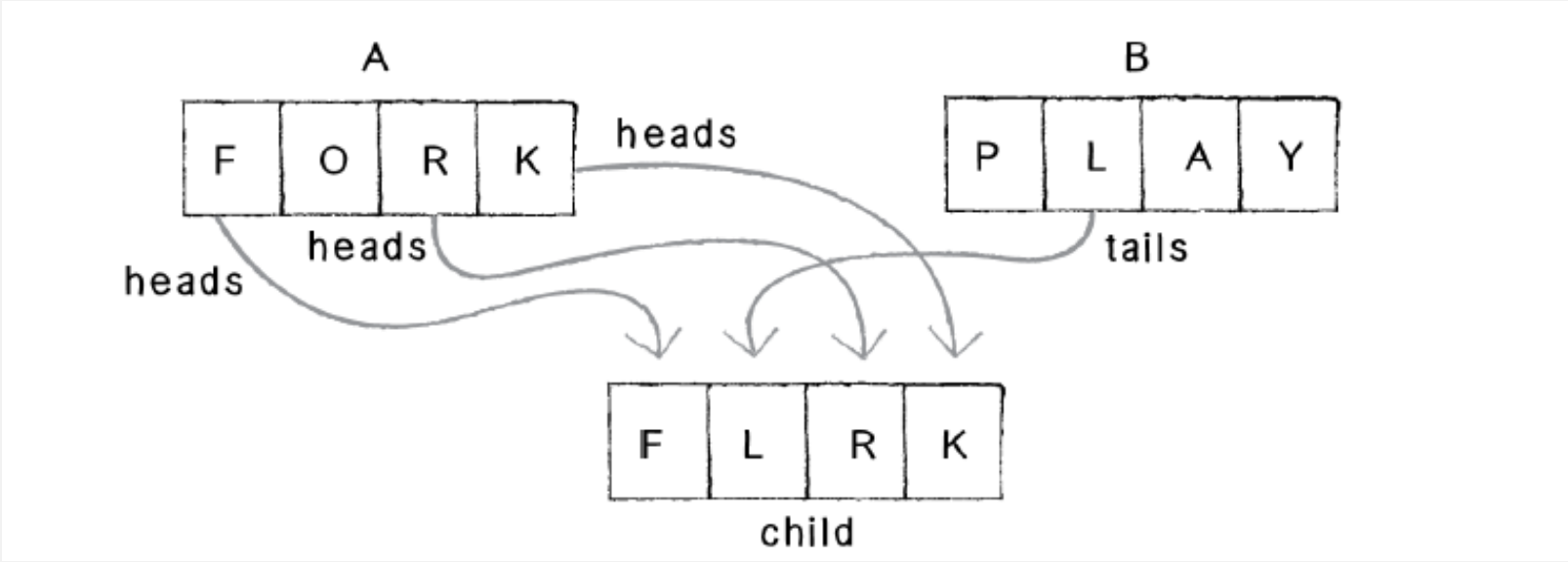


3° STEP: REPRODUCTION

Random splitting point



Coin-flipping approach

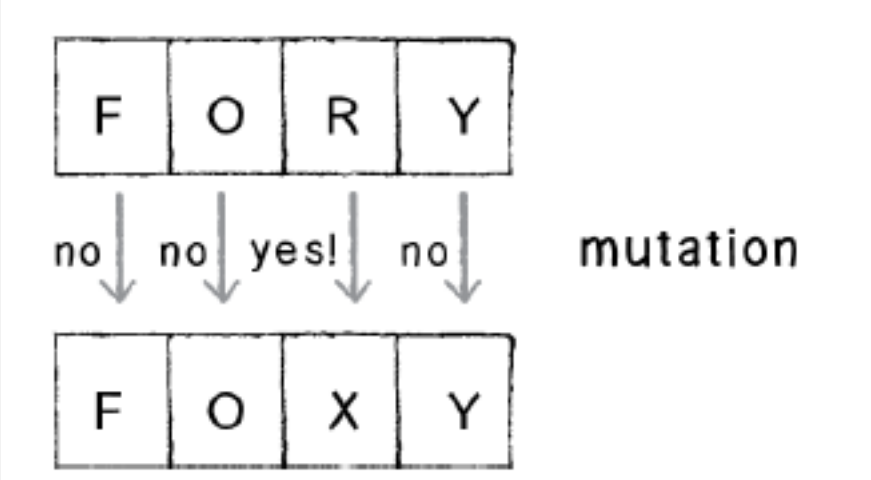


3° STEP: REPRODUCTION

- **Mutation**
- Allows to introduce further mutations into new generation
- It is described in terms of a rate of a gene to mutate

- If we have a **mutation rate** of 1%, this means that for each character in the phrase generated from crossover, there is a 1% chance that it will mutate.
- The mutated elements is replaced with a randomly generated one

High mutation rate (such as, say, 80%) would negate the evolutionary process itself. If the majority of a child's genes are generated randomly, then we cannot guarantee that the more "fit" genes occur with greater frequency with each successive generation.



SUMMARY

- SETUP:
  - Step 1: Initialize. Create a population of N elements, each with randomly generated DNA.
- LOOP:
  - Step 2: Selection. Evaluate the fitness of each element of the population and build a mating pool.
  - Step 3: Reproduction. Repeat N times:
    - a) Pick two parents with probability according to relative fitness.
    - b) Crossover—create a "child" by combining the DNA of these two parents.
    - c) Mutation—mutate the child's DNA based on a given probability.
    - d) Add the new child to a new population.
  - Step 4. Replace the old population with the new population and return to Step 2.

GA-BASED SYSTEM DESIGN

- Some design key-points:
  - **Key #1: Population size and mutation rate**
  - **Key #2: The fitness function**
  - **Key #3: Genotype and Phenotype**
    - choose how to encode the properties of your system
    - what are you trying to express, and how can you translate that expression into a bunch of numbers?
    - what's the chromosomes ? What are the alleles ?

"INFINITE MONKEY THEOREM"

- A monkey hitting keys randomly on a typewriter will eventually type the complete works of Shakespeare (given an infinite amount of time)
- The problem with this theory is that the probability of said monkey actually typing Shakespeare is so low that even if that monkey started at the Big Bang, it's unbelievably unlikely we'd even have Hamlet at this point

The infinite monkey theorem is an optimization problem

- In our case -> arrive at the phrase "**to be or not to be that is the question**"

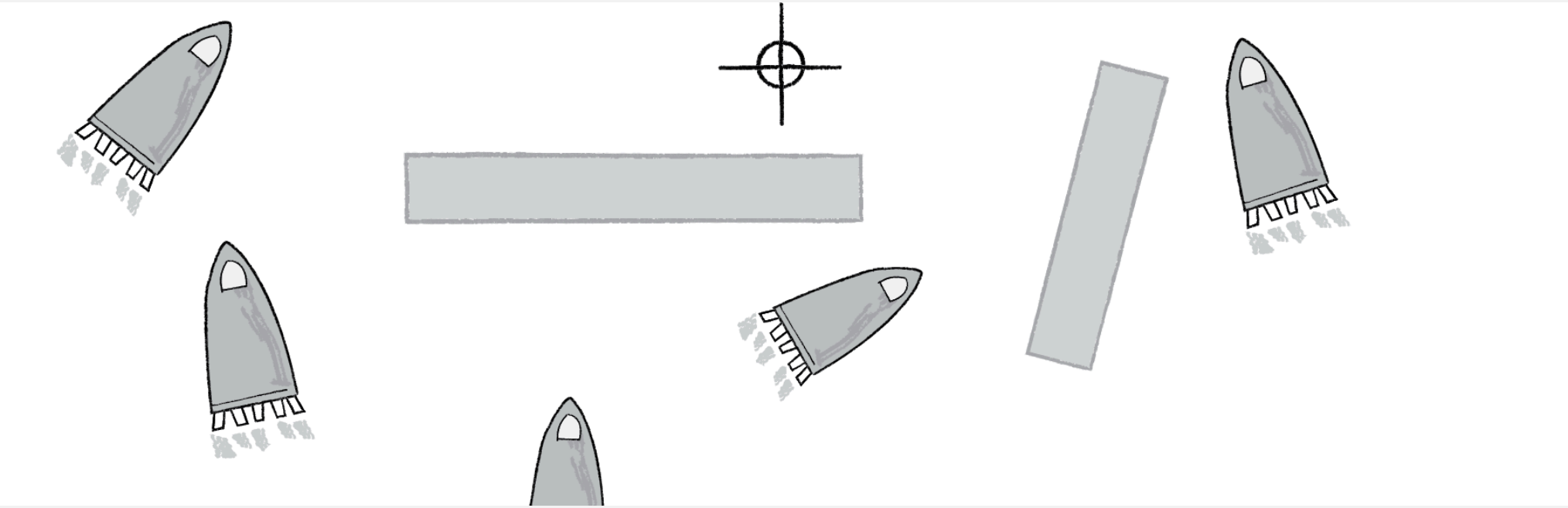
“INFINITE MONKEY THEOREM”

- Genotype: **chars**
- Phenotype: **chars**
- Fitness function: **% of correct chars**
- Chromosomes: **phrase**
- Population size: **150**
- Mutation rate: **1%**
- Crossover method**: midpoint**

Shakespeare

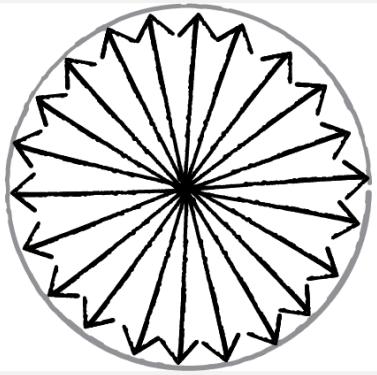
SMART ROCKETS

- **Goal**: a population of rockets launches from the bottom of the screen with the goal of hitting a target at the top of the screen (with obstacles blocking a straight line path)



SMART ROCKETS

- Population size: **100 rockets**
- Mutation rate: **1%**
- Fitness function: **Fitness is inversely proportional to distance: the smaller the distance, the greater the fitness; the greater the distance, the smaller the fitness**
- Genotype: **directions of the rocket**
- Phenotype: **the rocket**



```
PVector v = new PVector(random(-1,1),random(-1,1));
```

SMART ROCKETS

- Chromosomes: **sequence of frames of the rockets**
- The rockets has a life cycle that corresponds to a path to reach the target (# of frames)
- The path is a sequence of directions
- Once the path is completed so the chromosome is completed

- Hence, the number of genes in the chromosome is the number of frames of the rocket

SmartRockets

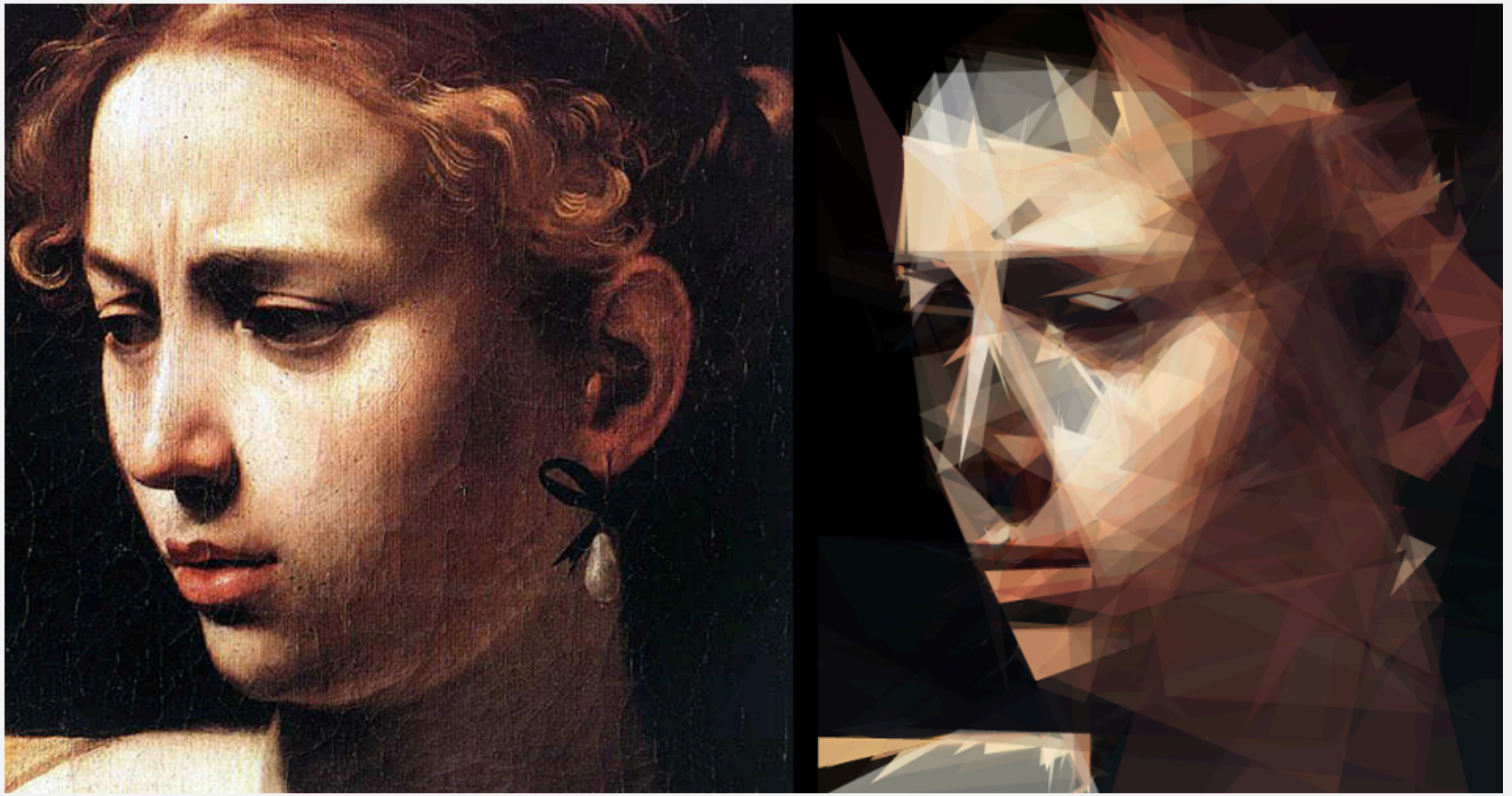
SmartRocketsObstacles

PAINTING

https://www.youtube.com/watch?v=iV-hah6xs2A

https://www.youtube.com/watch?v=Tza09kC6Xnc

https://www.youtube.com/watch?v=rGt3iMAJVT8



The genotype and the phenotype will highly influence the style

A tool
https://chriscummins.cc/s/genetics/

PAINTING

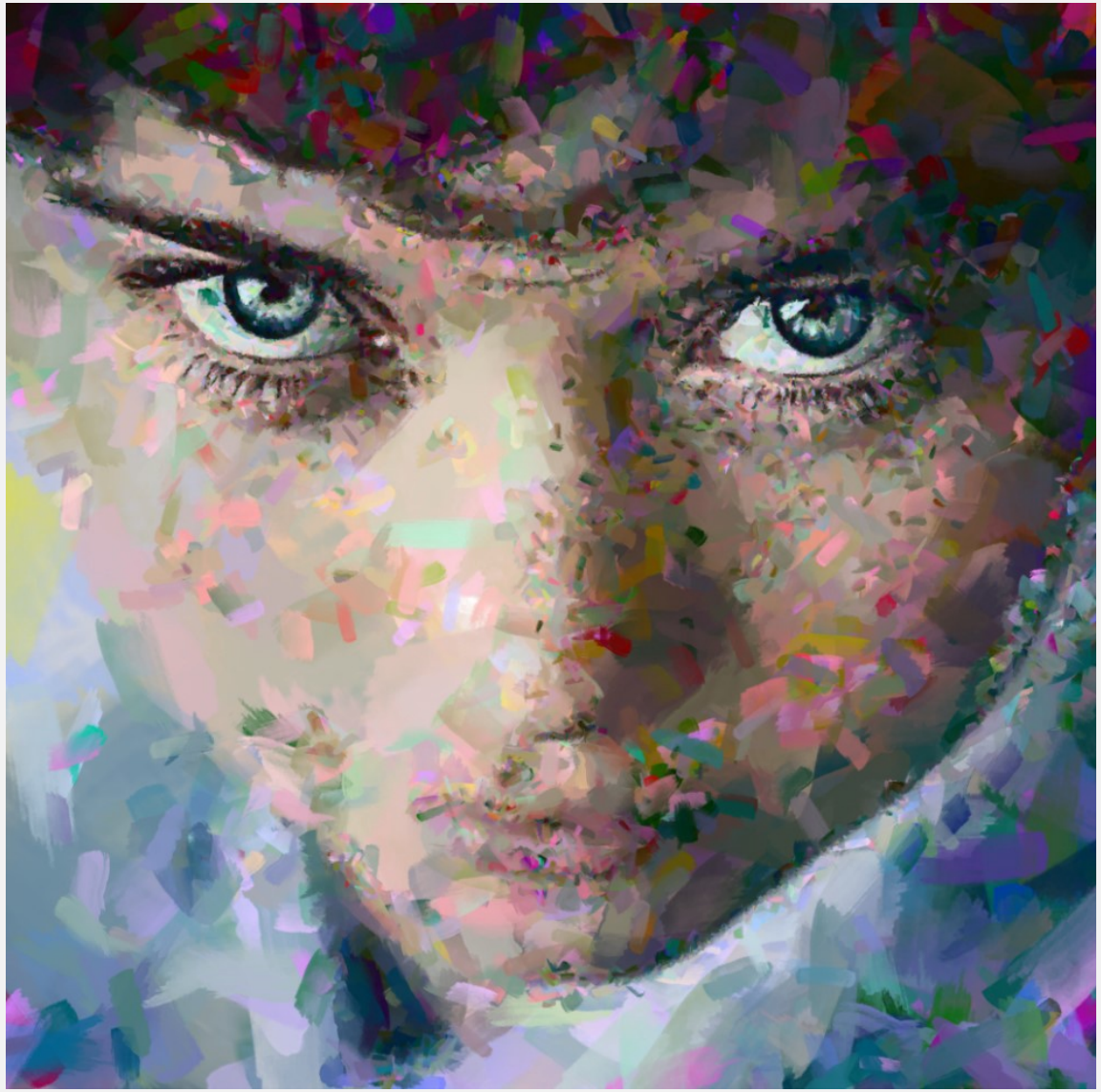


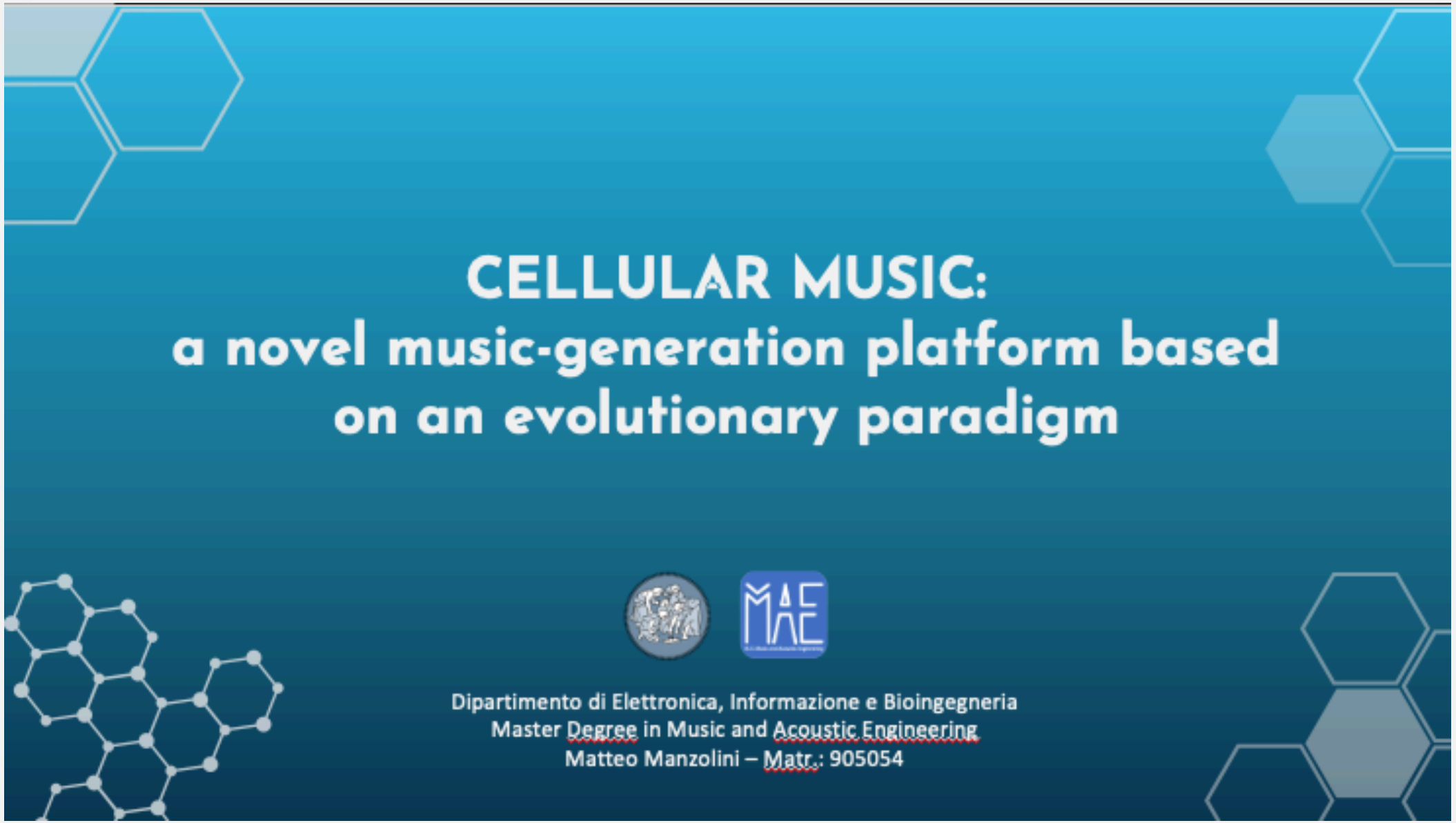
IMAGE EVOLUTION

- **Goal**: Given a limited number of brushes placed on the canvas, how must we position, scale, rotate and colour these brushes, to create a result which is visually as close to the original image as possible?
- **Genotype**: position, rotation, scale, colour and texture of the brush
- **Phenotype**: a brush
- **Gene**: position, rotation, scale, colour and texture of the brush
- **Chromosome**: a collection of 120 brushes
- **Fitness function:** pixel-to-pixel difference between target image and generated image

IMAGE EVOLUTION

The genotype and the phenotype will highly influence the style

CELLULAR MUSIC



CELLULAR MUSIC

- A generative space to put in relation music compositions and generative rules defined by the composer
- Inspired by microbiology
- The model emulates the life cycle of cellular microorganisms bred in Petri Capsule
- We insert some genetic evolution paradigm into the composition

CELLULAR MUSIC

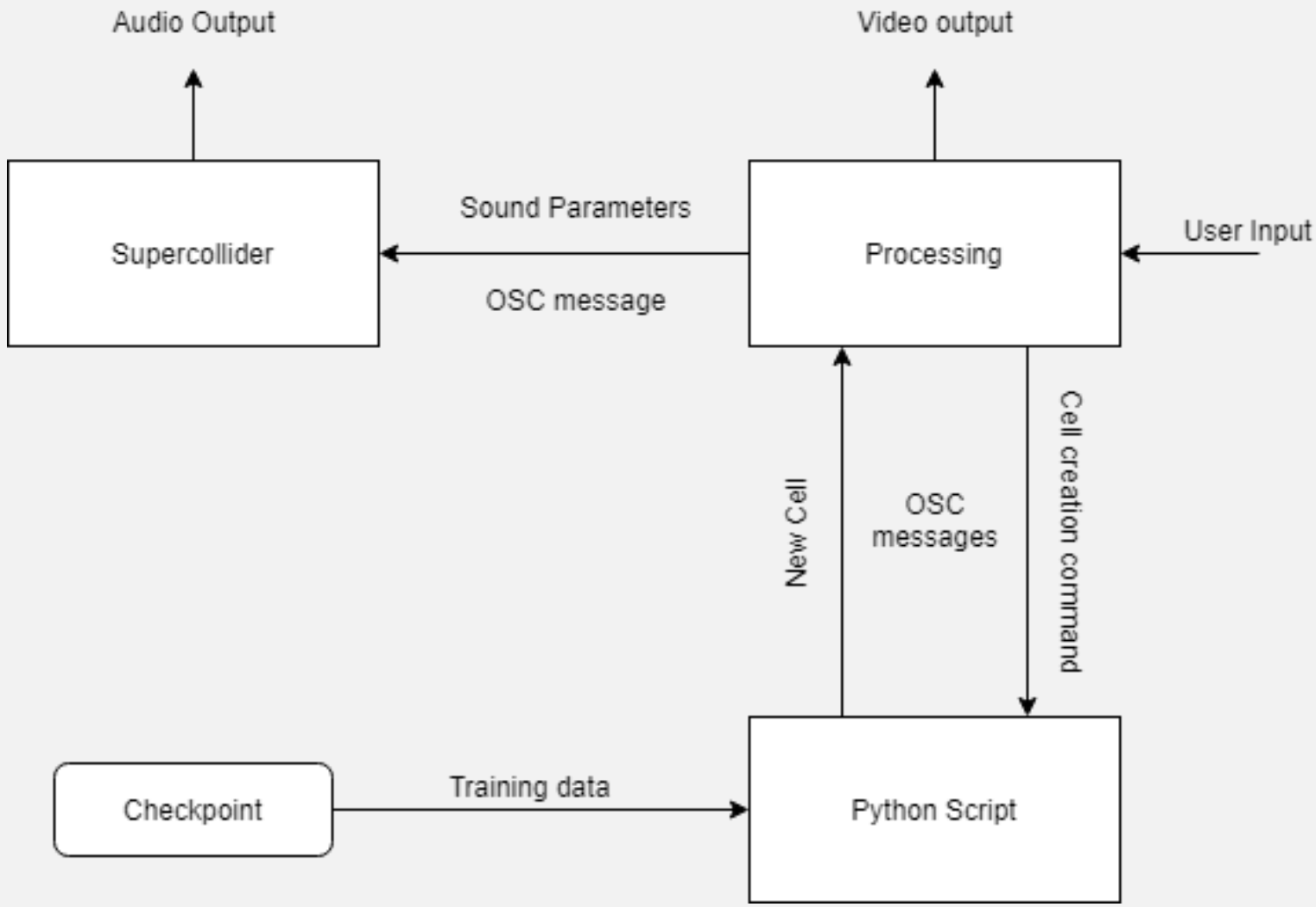
- **Cells**
  - Melody
  - Note
  - Melodic cell: sequence of note that can compose a melody
  - They live in a habitat
- **Habitat**
  - Bidimensional space where a bass note plays. More habitats create and ecosystem
- **Fitness function**:
  - Compatibility measure between cells or cells and an habitat through a set of composition rules defined by the composer

- Evolution in implemented through a GA

CELLULAR MUSIC

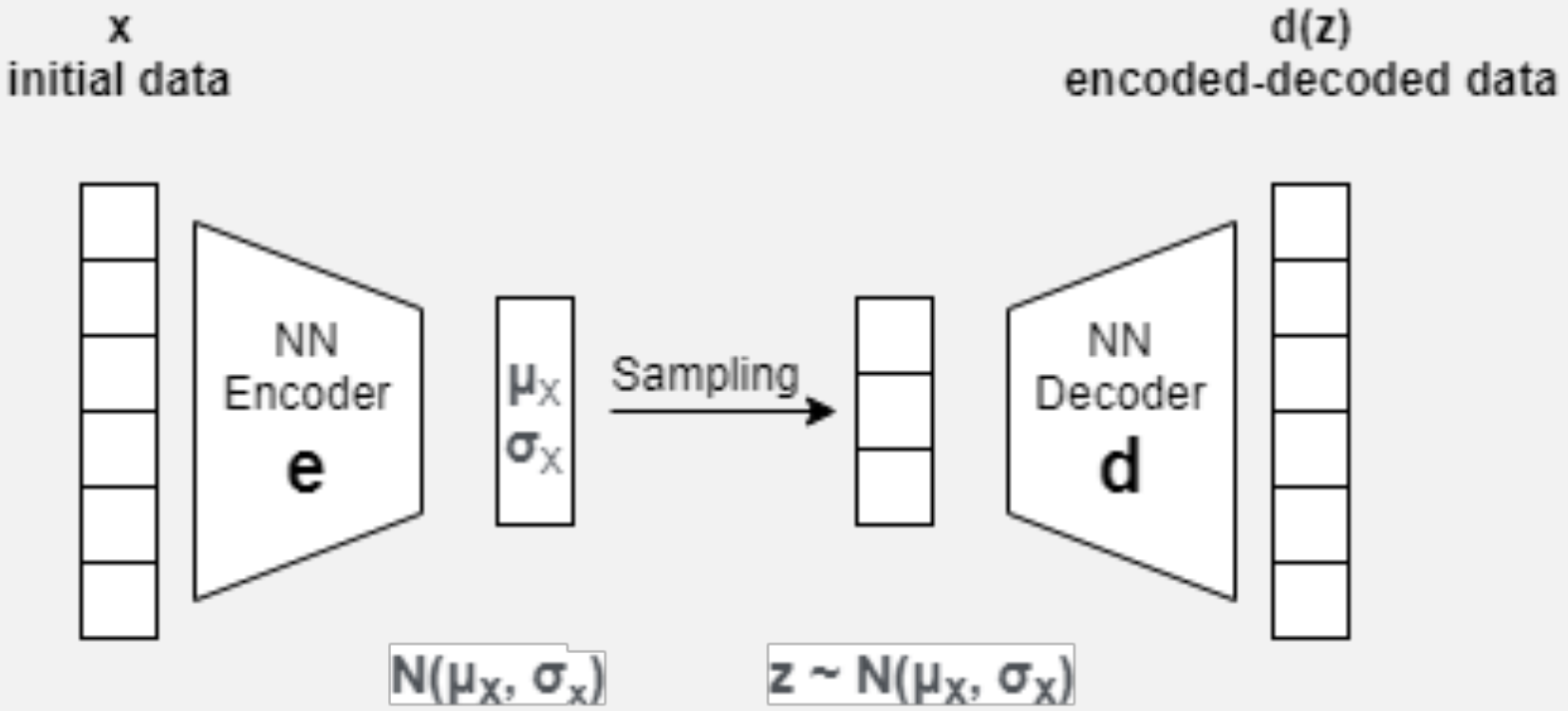
- GA leads the position of the cell in the ecosystem –> at each time each cell will try to optimize its position in the ecosystem according to the compatibility with the habitat and the cells around
- More evolutive rules are added:
  - Cells with high compatibility and close in the space can be
    - concatenated
    - deployed
    - recombined
  - Cells can split
  - Cell can grow

CELLULAR MUSIC



CELLULAR MUSIC

- Cells can be generated by the composer of automatically thanks to musicVAE



VIDEO

INTERACTIVE GENETIC ALGORITHMS

INTERACTIVE GA (IGA)

- Classical GA can be applied when a target is clear and the fitness function can be properly design
- This is not the case of creative applications
- An interaction by the human (or by another system) can overcome the absence of the fitness function
- Interactive selection refers to the process of evolving through user interaction
- The fitness values assigned by users (or another system)

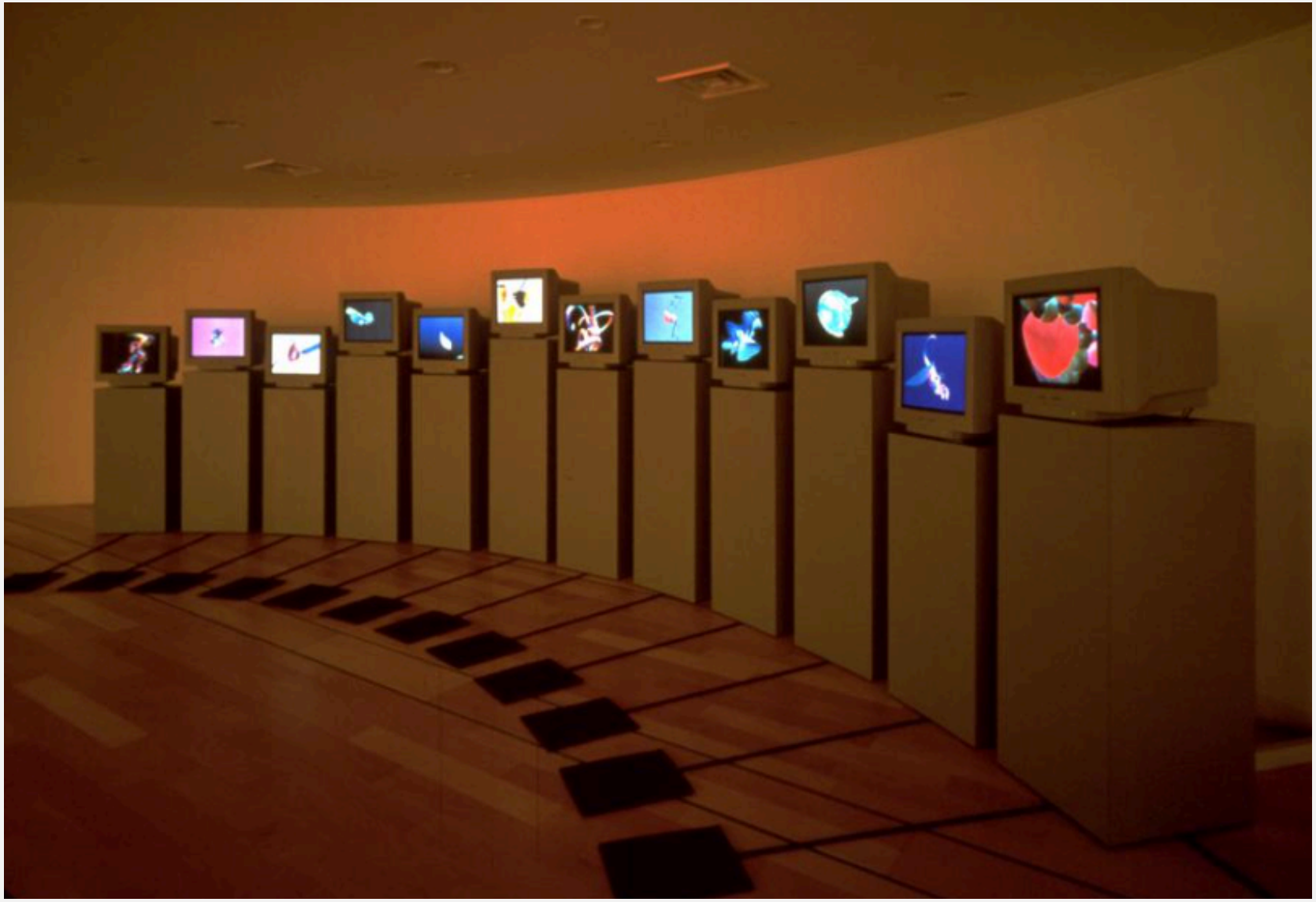
- The first important IGA-based art – **Galapagos by Karl Sims**

INTERACTIVE GA (IGA)

- Galapagos by Karl Sims
- Installed in the Intercommunication Center in Tokyo in 1997 – twelve monitors displaying computer-generated images
- **Population**: Images displayed in the monitors
- These images evolve over time, following the genetic algorithm steps of selection and reproduction.

- In front of each monitor is a sensor on the floor that can detect the presence of a user viewing the screen. The **fitness** of an image is tied to the length of time that viewers look at the image

INTERACTIVE GA (IGA)



INTERACTIVE GA (IGA)

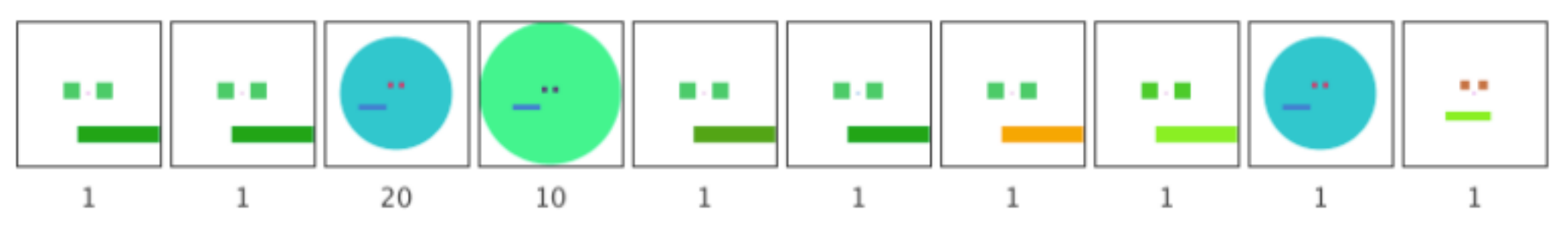
- Genotype (DNA) is a complex set of shapes and their parameters (dimensions, colours, textures)

EXAMPLE: EVOLVE EMOTICON

- Each face have a set of properties: head size, head colour, eye location, eye size, mouth colour, mouth location, mouth width, and mouth height
- The face's **chromosomes** is an array of floating point numbers between 0 and 1, with a single value for each property
- Each genes is then map on a specific parameter space

color eyecolor =color(dna.genes[4],dna.genes[5],dna.genes[6])

EXAMPLE: EVOLVE EMOTICON

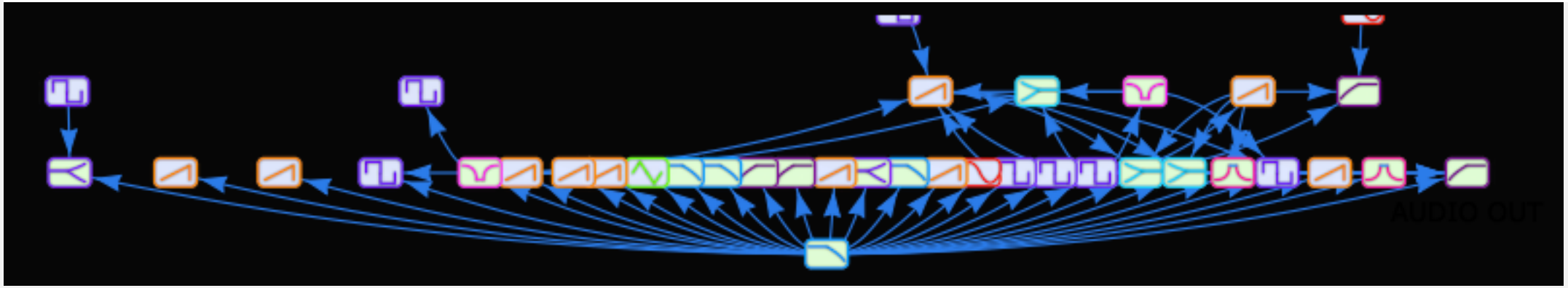


- The user can place the mouse pointer over preferred emoticons
- The time the mouse pointer remains on a emoticon will be the probability value of the fitness function

Faces interactive selection

EXAMPLE: EVOSYNTH

Name	Description	Normalised range
u	ugen type	oscillator/ filter
s	ugen subtype	sine, square, sawtooth, triangle / lowpass, highpass, bandpass, lowshelf, highshelf, allpass, notch
x	position	0-1
y	position	0-1
θ^1	connection arc angle	0-2π
θ^2	connection arc angle	0-2π
r	arc radius	0-1
p^1	initial value 1	
p^2	initial value 2	
i	connection target	0-no. in ports on target

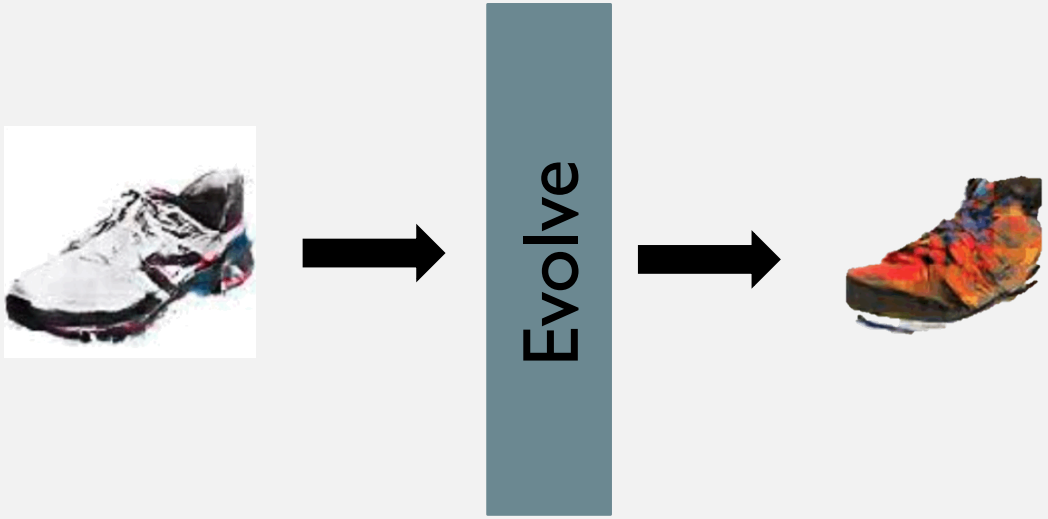


http://www.yeeking.net/evosynth/

DEEP INTERACTIVE EVOLUTION

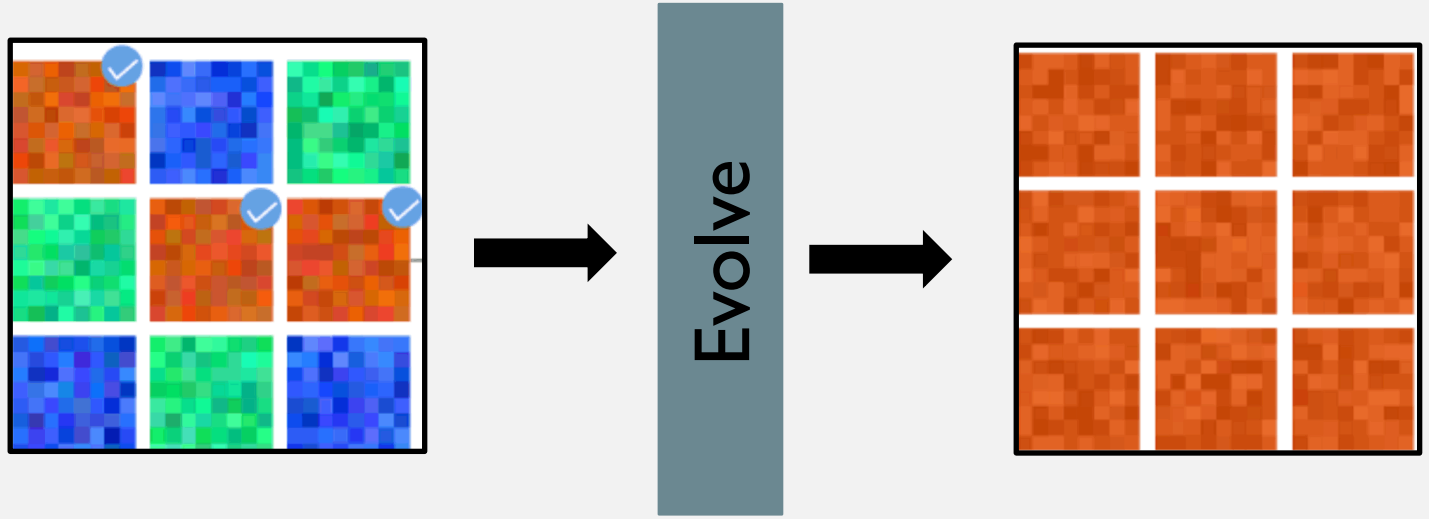
DEEP INTERACTIVE EVOLUTION

- **Combine Deep Generative models and Evolutionary Computation**
  - How can a model take user preferences into account during the generation process?



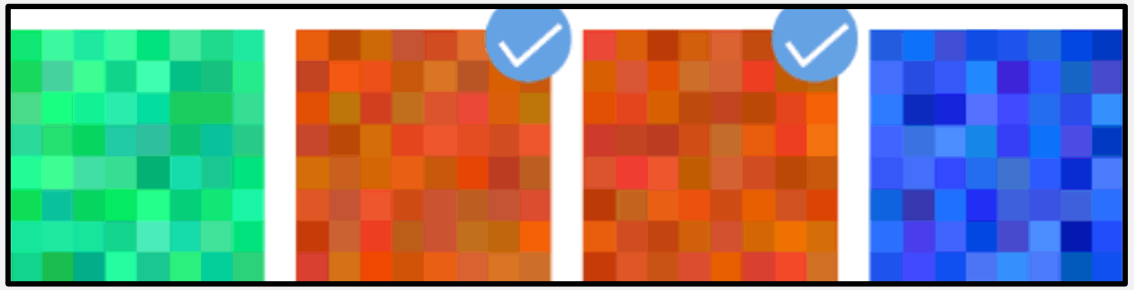
DEEP INTERACTIVE EVOLUTION

- **Genetic Algorithms**
  - The system that we want does the following:
    1. *Start with a batch of random samples*
    2. *Ask the user to select the preferred ones*
    3. *Generate samples with features similar to those of the selected samples*
    4. *Repeat 2-3 until a sample has all desired attributes*



DEEP INTERACTIVE EVOLUTION

- **Genetic Algorithms**
  - We are performing an optimization process
  - *Fitness function is not differentiable, since it is user-defined!*



$$\rightarrow \quad [0, 1, 1, 0]^T$$

  - Selected choices can be represented as a vector of 1's and 0's

  - No back-propagation is possible −> we resort to Genetic Algorithms

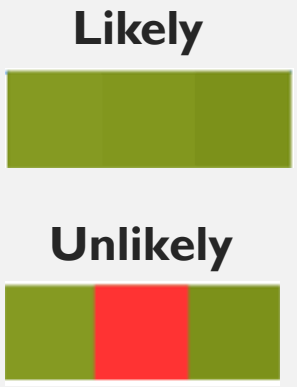
DEEP INTERACTIVE EVOLUTION

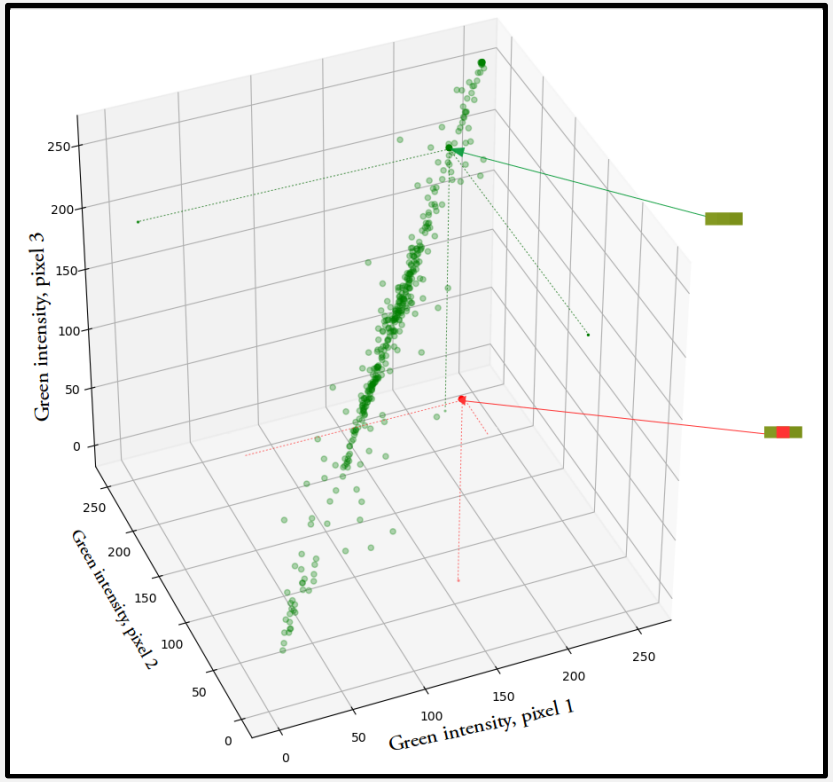
- **Interactive Evolutionary Computation(IEC)**
  - genetic optimization in response to user inputs
  - Genetic algorithms, combine features of the best samples (+ mutation) to produce better ones

  - Hard to define what is a *feature*
    - If each single pixel is a *feature*:

DEEP INTERACTIVE EVOLUTION

- Genetic algorithm require features to be as much independent as possible
- For most data types (e.g. images, sound, text) this is not true
- E.G. pixels:



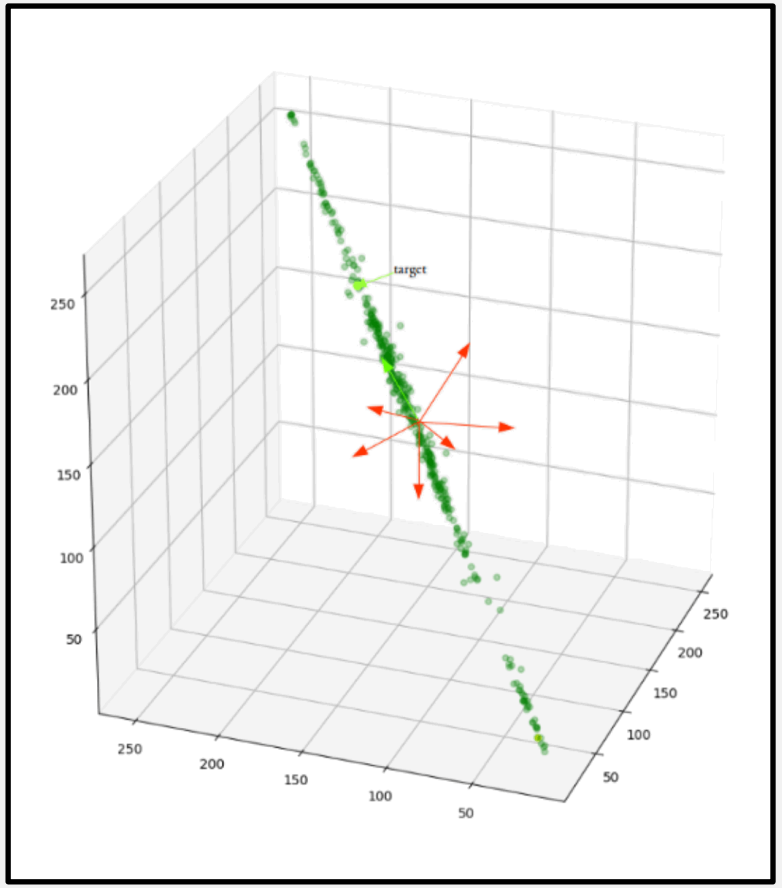


DEEP INTERACTIVE EVOLUTION

- Most Data lie on a small subspace of the total space used to represent

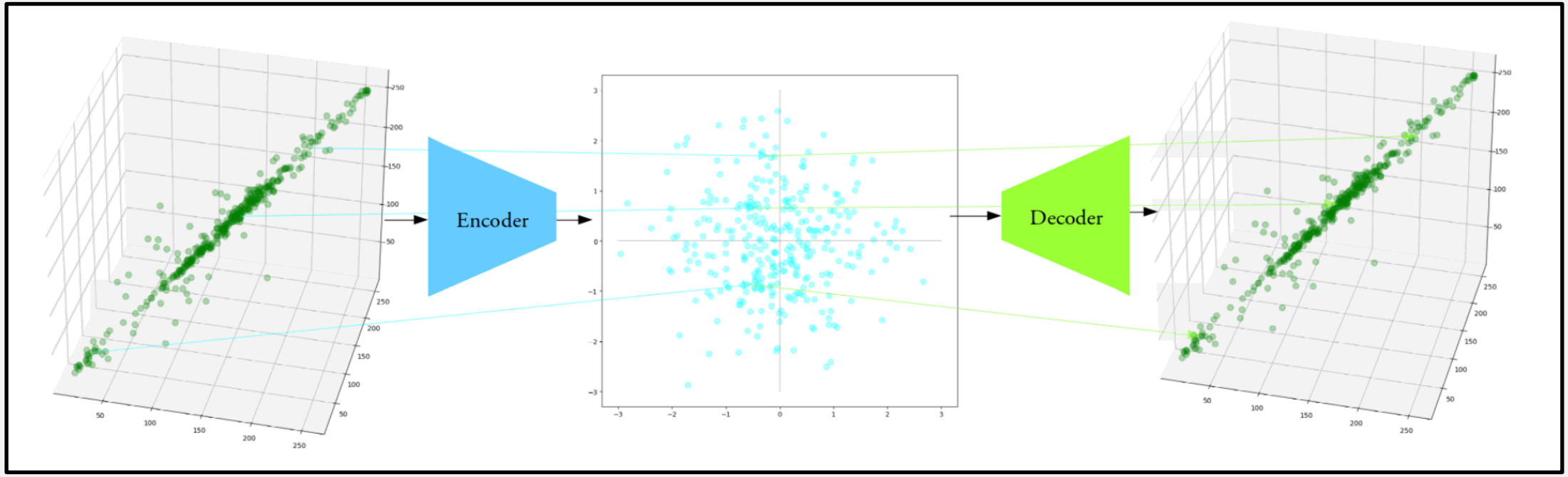
  it –> manifold

- Only this data represents information that would be considered realistic
- Genetic algorithms randomly explore the search space
- Human judgement-based approach is not going to converge in a reasonable amount of time



DEEP INTERACTIVE EVOLUTION

- **Idea**
  - *Optimize the latent vectors of generative models instead of the Raw data*

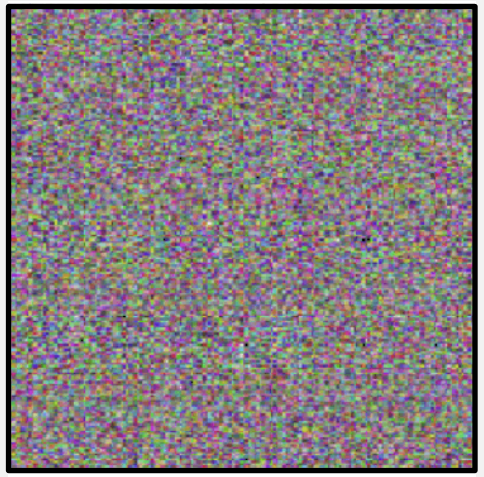


  - Merge IEC with Deep Generative Models
  - General Solution

DEEP INTERACTIVE EVOLUTION

- **Idea**
  - Latent vectors are typically sampled from a gaussian distribution
  - Components are independent of each other
  - Decoder learns to map certain regions of latent space to realistic saples

Sampling from latent space



Sampling from input space

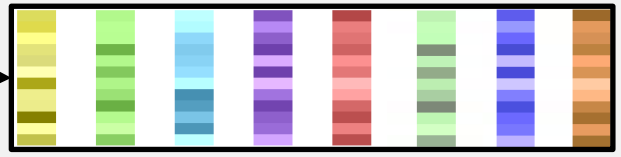
- *One can freely recombine features and add random noise to them, while still generating realistic data*

DEEP INTERACTIVE EVOLUTION

- ***Algorithm***

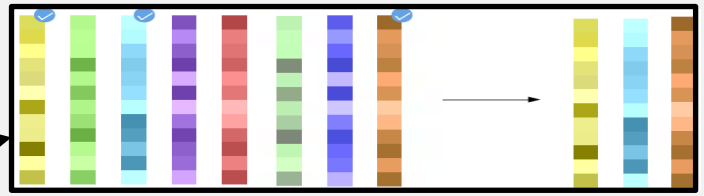
A. Sample

- Sample n random vectors from distribution



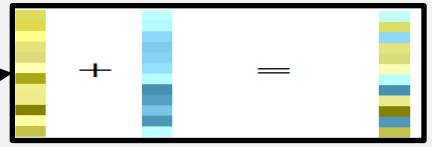
B. Generate

- Repeat the following:



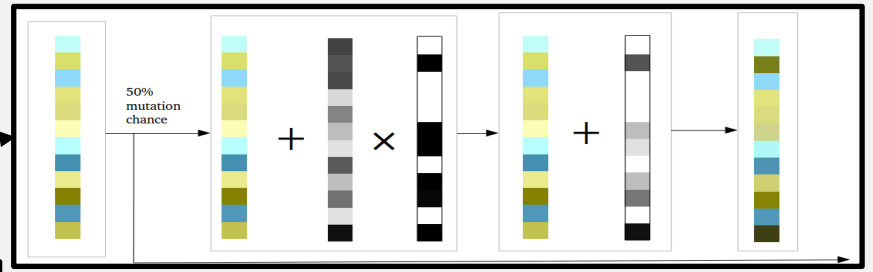
i. Selection

- user selects best samples



ii. Crossover

- Crossover between selected latent vectors

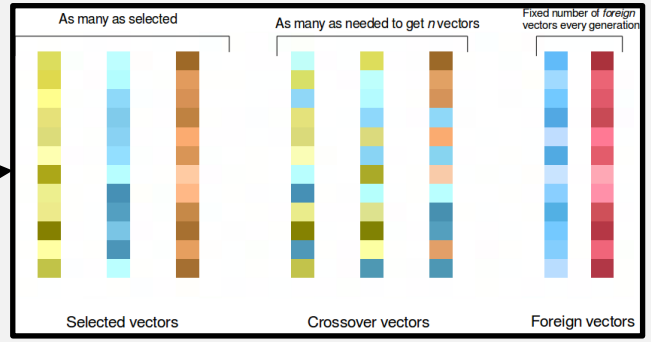


iii. Mutation

- Mutate all latent vectors

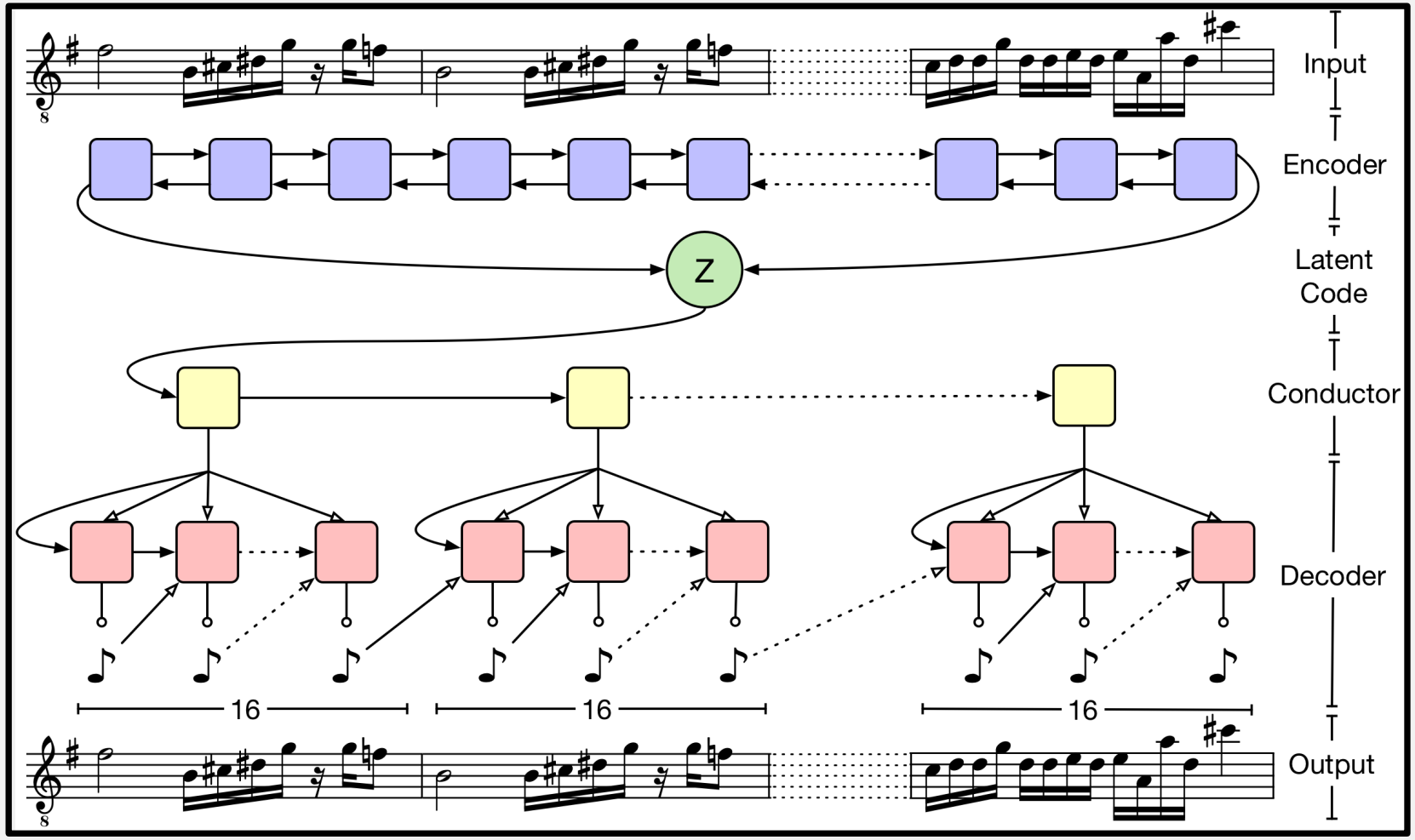
iv. Foreign Vectors

- Randomly generated vectors



DEEP INTERACTIVE EVOLUTION

- [MusicVAE]
  - Hierarchical autoencoder
  - **Encoder**
    - Bidirectional RNN
  - **Decoder**
    - Autoregressive RNN
- Latent code passed to a conductor RNN, outputs a new embedding for each bar of the output
- Note RNN generates. Each bar independently

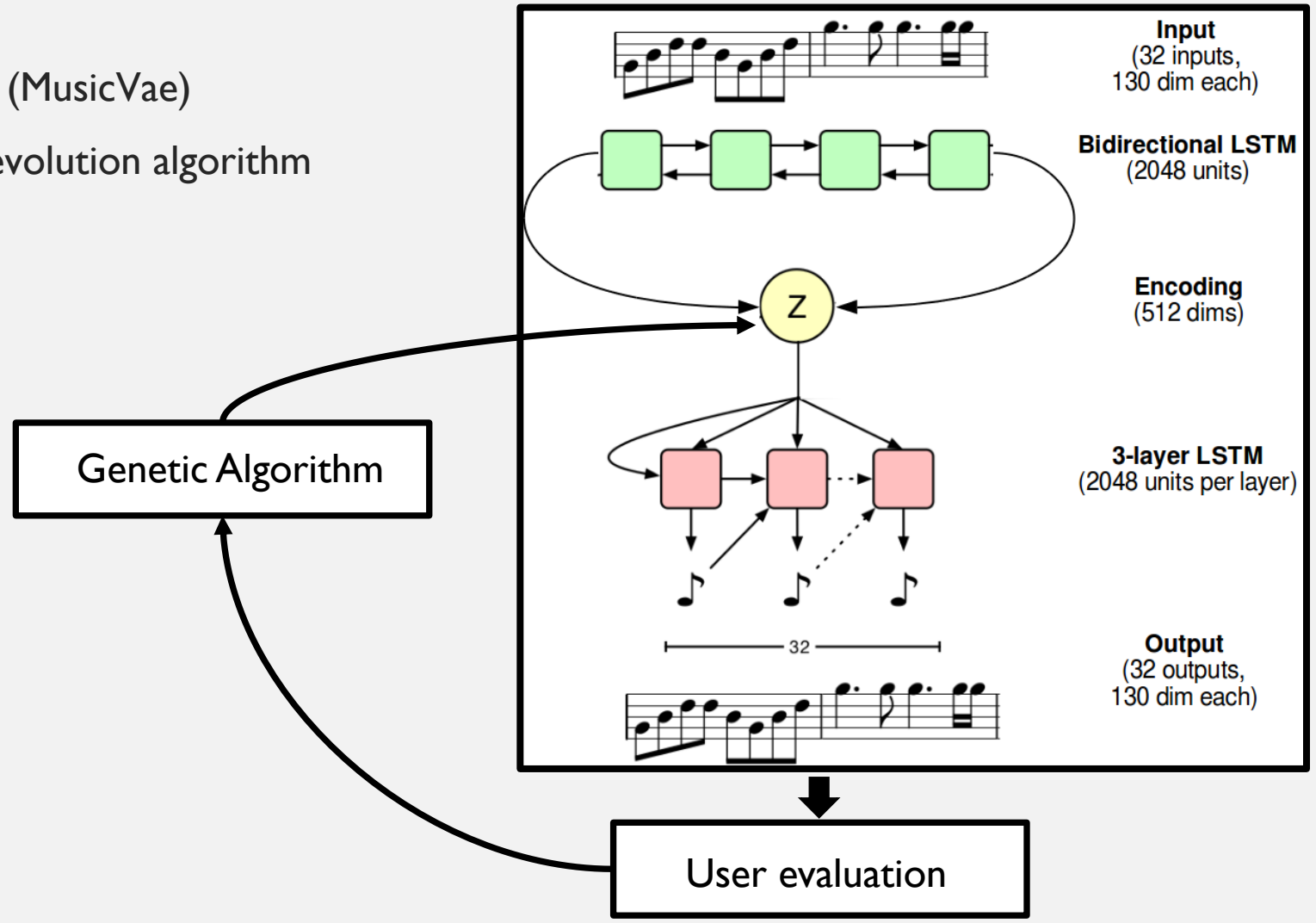


DEEP INTERACTIVE EVOLUTION

- **How to do Music Generation?**
  - Train a generative model on music data (MusicVae)
  - Apply previously described interactive evolution algorithm

  - E.G.
    - Start

    - 8 Generations later



MATERIALS

- **References**

- **Further readings**

REFERENCES

- **Evolutionary Ecosystem**
  - Nature of Code Book Chapter on Genetic Algorithms

- **Deep Interactive Evolution**
  - Original Paper
  - Blog Post
  - **MusicVAE**
    - Original Paper
    - Blog Post
    - Sound Synthesis

- **Genetic Algorithms with Supercollider**
  - Sound Synthesis