LSAA 2024

Connecting to Tomorrow

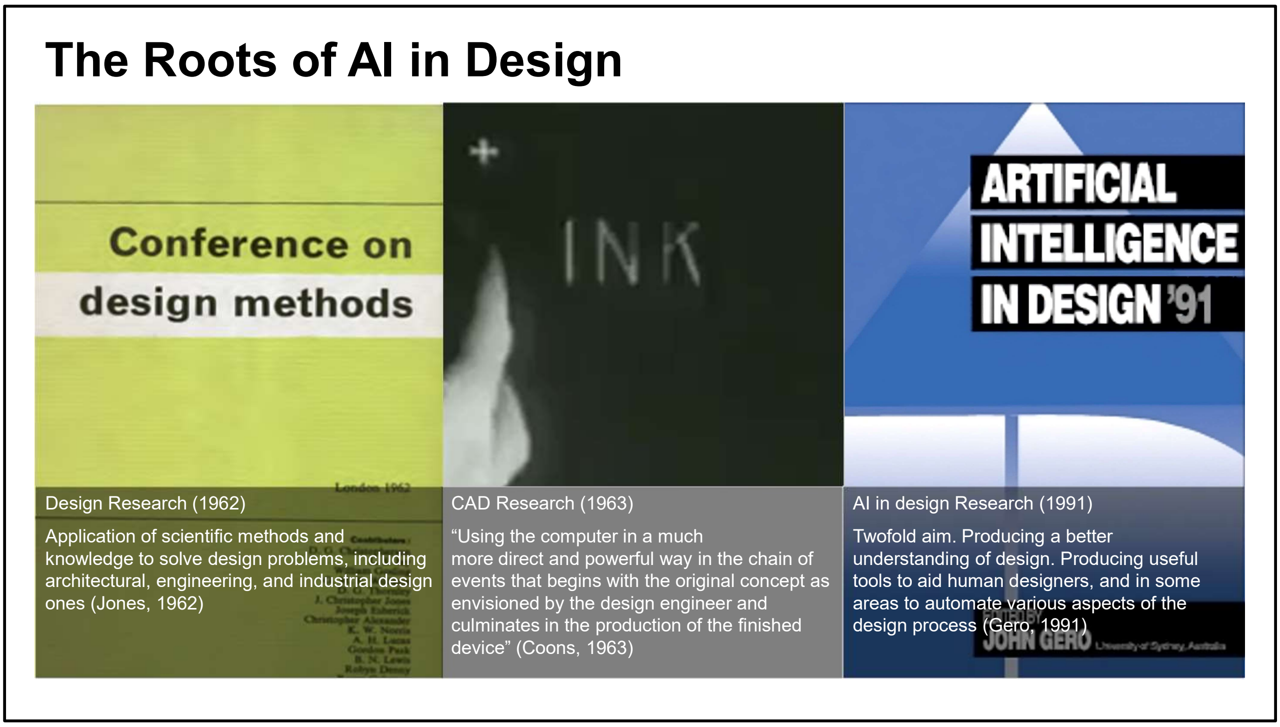
Lightweight Structures Association of Australasia

Collaboration scenarios between architects and artificial intelligence

Alberto PUGNALE

13 September 2024

THE UNIVERSITY OF MELBOURNE



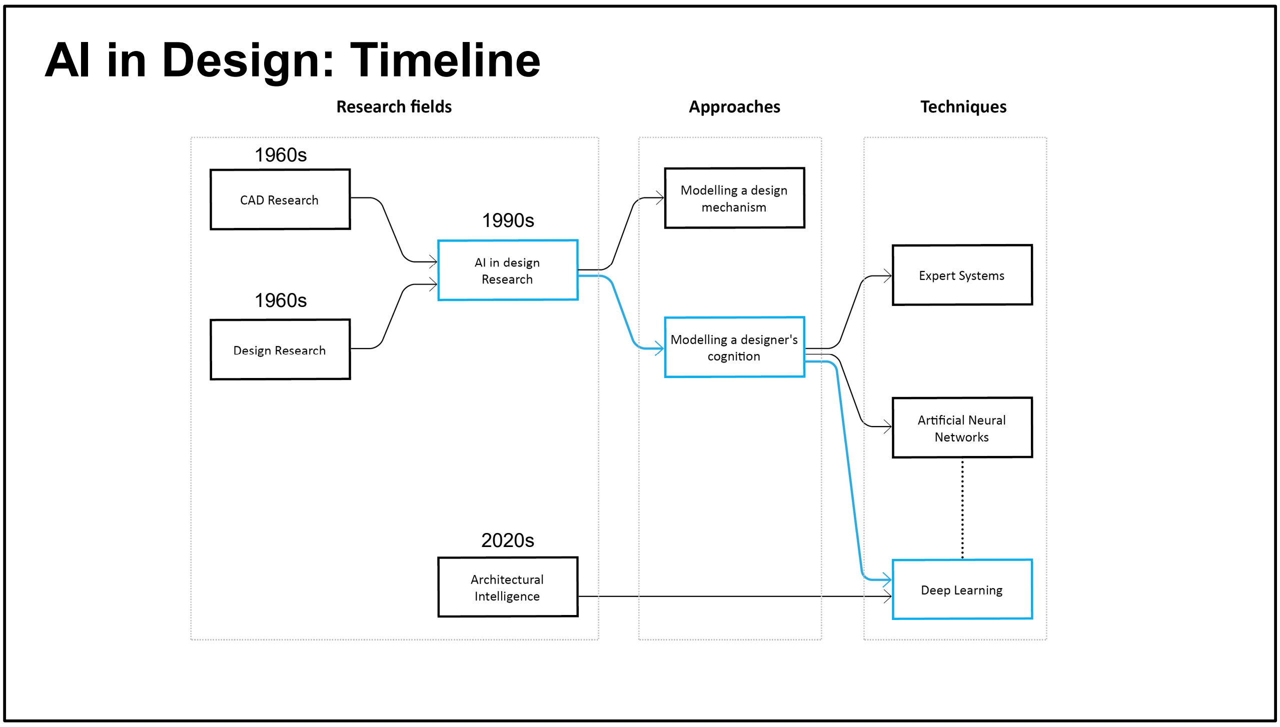
This presentation aims to show you at least three ways to simulate mechanisms of a human designer’s cognition to create artificial forms of intelligence, AIs with which we can interface and collaborate. However, I’d like to clarify that when we put together these slides, we thought that the overarching purpose of the presentation should have also been to provide you with an overview of the history of AI in the specific field of design, and how the concepts around AI in design have evolved and mutated over the decades.

Exploring the past to better understand the present is always a surprisingly good way to discover that what we aim to achieve nowadays was already theorised 50 or more years ago, generally anticipating the technological developments that actually allowed us to apply the theory.

On this slide, you can see three key moments in the development of AI in Design. On the left, the 1962 conference on design methods was one of the first attempts to conduct research in the design disciplines following a scientific approach: academic working on design methods were among the first to apply decision-making and problem-solving strategies to investigate the solution of design problems. The same academics were also the first to suggest that computers could have facilitated the formalisation, and therefore the resolution, of such problems.

Computer-Aided Design or CAD Research emerged as a field of study in the same years. Here the focus was on the development of human-machine interfaces to support a variety of design processes and workflows, from conceptual sketching to design production. I think we can all agree on considering SketchPAD, developed in 1963 at the MIT, as the first Graphic User Interface and the first CAD software, from which all current programs derive.

On the right, you can see AI appearing in the title of these 1991 conference proceedings, and the reason for discussing AI in the context of CAD Research is that academics working in this stream started from the hypothesis that computers should possess some mechanisms of human cognition in order to act as design assistants. This interpretation of Computer-Aided Design is probably the most sophisticated one as research in this field aimed at better understanding design and design processes to create tools that could automate the resolution of design tasks.



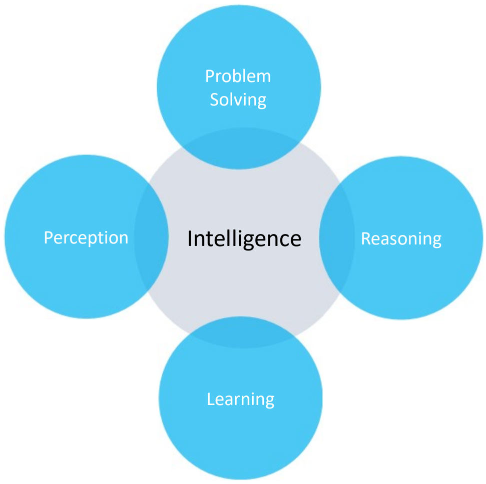
To summarise, the complex and intricate timeline of AI research in design can be seen as the convergence between the CAD research and the design research of the 60s, which evolved into a stream renamed as Artificial Intelligence in Design in the 90s. Before this period, AI was primarily seen as a way to automate tasks, and didn't involve understanding and modelling the designer's cognition, as opposed to classical optimisation, which models design mechanisms. We'll come back to that in a few slides.

Nowadays, when we use the term Artificial Intelligence, we generally refer to Artificial Neural Networks and Deep Learning, therefore to systems that are based on "learning".

What is Intelligence?

Intelligence measures an agent’s ability to achieve goals in a wide range of environments
(Legg and Hutter, 2007)

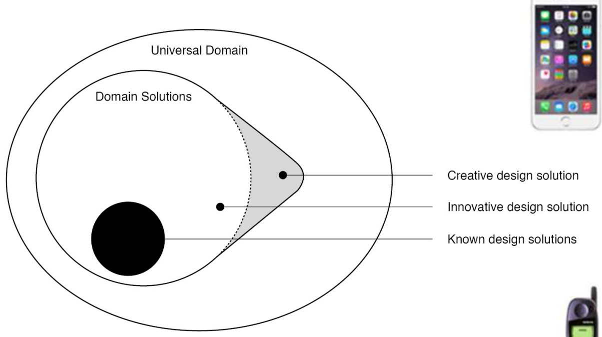
- The notion of ‘agent’ implies that intelligence is an emerging property of any system – biological or artificial – capable of ‘achieving goals’
- The last part of the definition concerns the universality of intelligence: an intelligent agent needs to demonstrate its ability to solve a large variety of problems.



The term “learning” seems to imply a certain degree of “intelligence”, and “intelligence” itself seems to imply a certain degree of “creativity”. Before we proceed any further, it’s worth highlighting that some researchers developed entire literature reviews on these aspects. For example, in 2007, Legg and Hutter analysed 70 definitions of the term intelligence, and they came up with their own, we could say, general and definitive definition. In their opinion, intelligence measures an agent’s ability to achieve goals in a wide range of environments. This ability is expressed through a set of skills, on the slide illustrated with the blue circles –

learning, reasoning, problem-solving and perception.

Not only automation: the quest for creativity

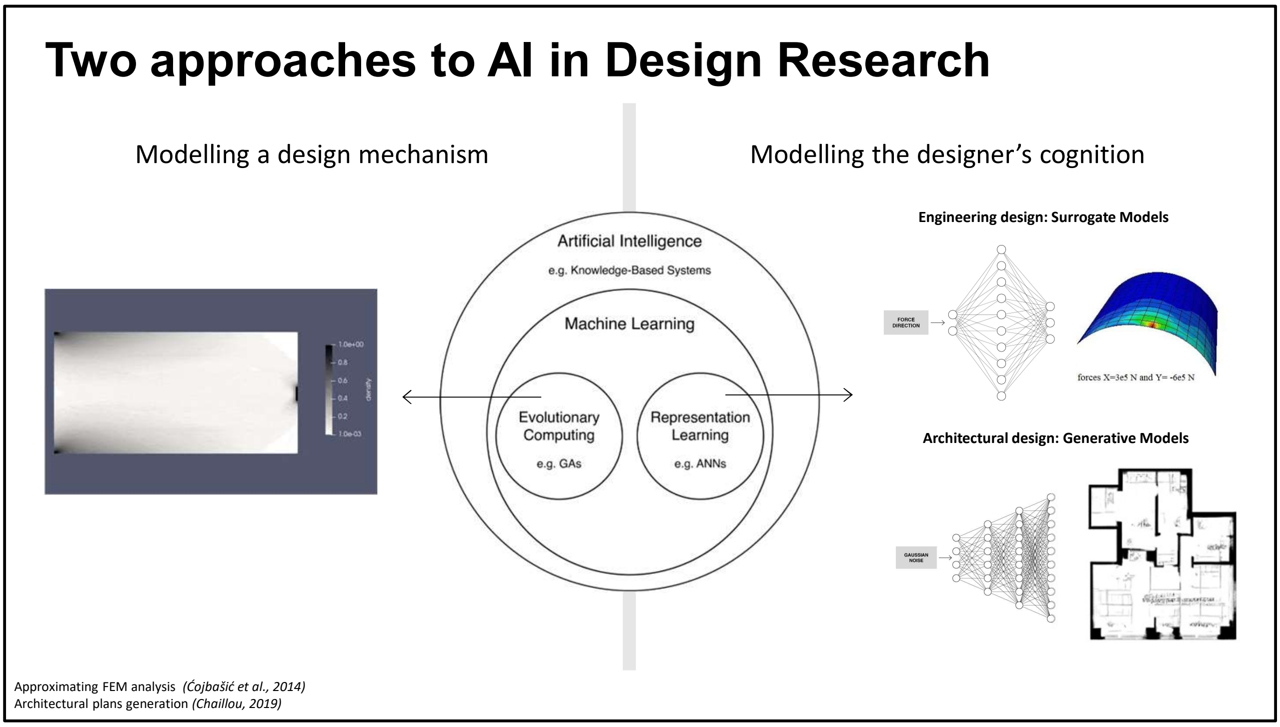


Relationship between known designs, innovative designs, and creative designs based on different extensions of the solution space (Rosenman and Gero, 1993)

But why would we want to develop an “intelligent” CAD system? We want computers to be less passive as part of the design process; we want them to be able to support a designer’s creativity and imagination. On this topic, a key researcher was, and still is, John Gero, who used to lead the Design Computing Unit in Sydney for many years. In his opinion, “creativity” could be defined as the ability to move beyond the design space of known solutions, or in other words, towards a broader domain, which he called the universal domain.

This slide shows a simple example of what we mean by the space of known solutions, and even innovative solutions within the same solution domain. It’s a simplified example of the design evolution of mobile phones. Now, without aiming to do a comprehensive review of mobile phone design evolution, we know that, up to a certain point, all designs included a keyboard and a screen. The most innovative designs were based on foldable phones, larger screens or complete keyboards, but the real creative solution only arose from a big conceptual jump, from moving beyond the space of known solutions: we don’t need the keyboard, we don’t need buttons, we don’t need a separate screen. The phone is the screen, which is a touch device, and such a device does everything else, including being a keyboard. From 2007 onwards, we could argue that all phone designs have been developed within this “new” space of known solutions.

Going back to John Gero’s work, this conceptual formalisation of the research on AI in design, allowed his group to imagine computational processes that could simulate creativity. We won’t get into those, but there are four: combination, mutation, analogy and first principles.



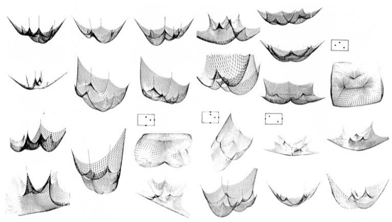
To complete the contextualisation of our research on AI in Design, we’d like to close this initial set of slides by distinguishing between two main design approaches that are currently followed to implement AI models in design applications. On the left, you can see evolutionary computing aimed at modelling a specific design mechanism. The example you see on the slide is a topology optimisation application, where the goal is to reduce the use of material starting from specific boundary conditions, and therefore, from a well-defined family of possible design solutions. If we follow the original definition of Artificial Intelligence and the sub-group of Machine Learning strategies, topology optimisation as well as Genetic Algorithms are part of AI research in Design. On the right, you can see the opposite approach, which involves modelling the designer’s cognition, therefore some aspects of human intelligence, rather than a specific design mechanism.

Common examples of this approach include surrogate models, for example, in which structural solutions are designed by an AI without performing actual calculations, or where apartment plans are designed by an AI starting from a given perimeter. Both strategies are based on a certain experience acquired by the model in order to design.

An important difference between the approach on the left versus the approach on the right is the way in which we, human designers, can interact with these systems. While we communicate with optimisation algorithms mainly through the definition of design variables, we can interact with the models on the right through visual outputs or natural language.

Simulating three learning mechanisms

Expertise



Exploration of multiple configurations of hanging chain models

Clustering and combining information related to a single knowledge domain

Simulating the designer's acquired experience

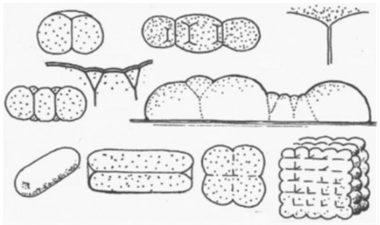
Playfulness

Playing with Froebel blocks

Interacting with objects with no boundaries

Simulating how a child learns by playing

Analogical Reasoning

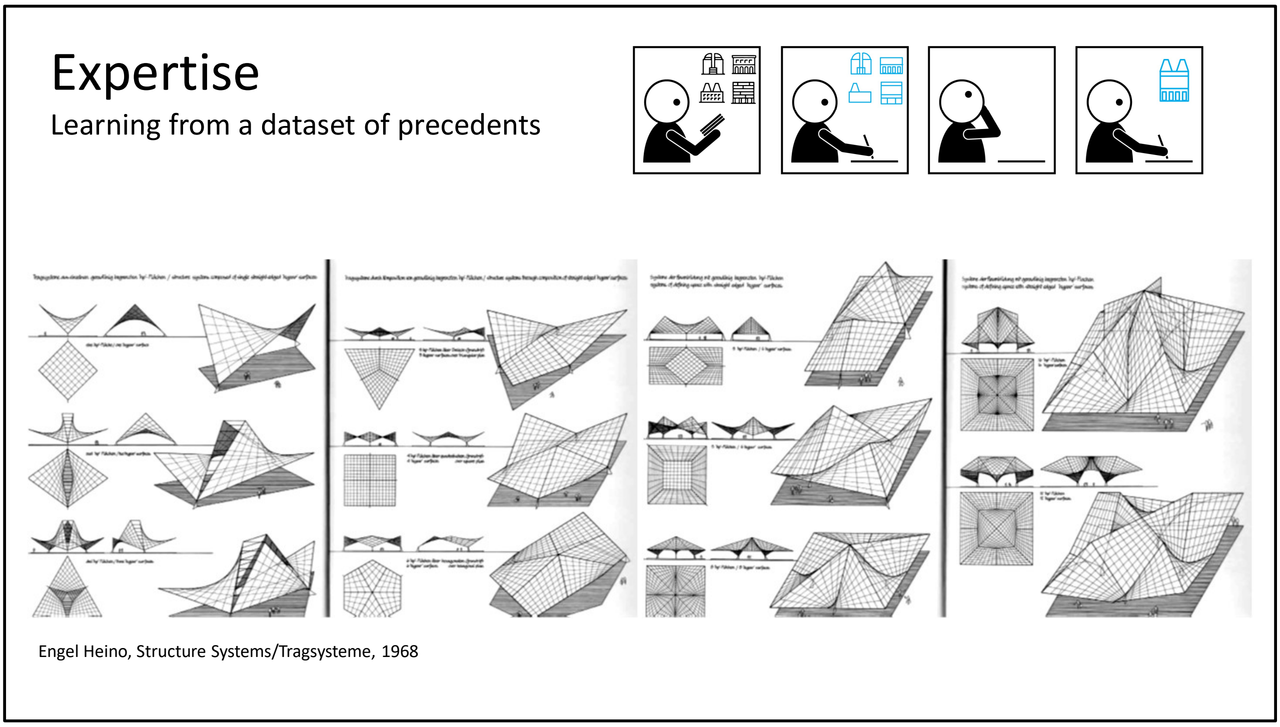


Construction of analogies between biology and architecture

Transferring information between multiple knowledge domains

Simulating how designers find inspiration from other fields

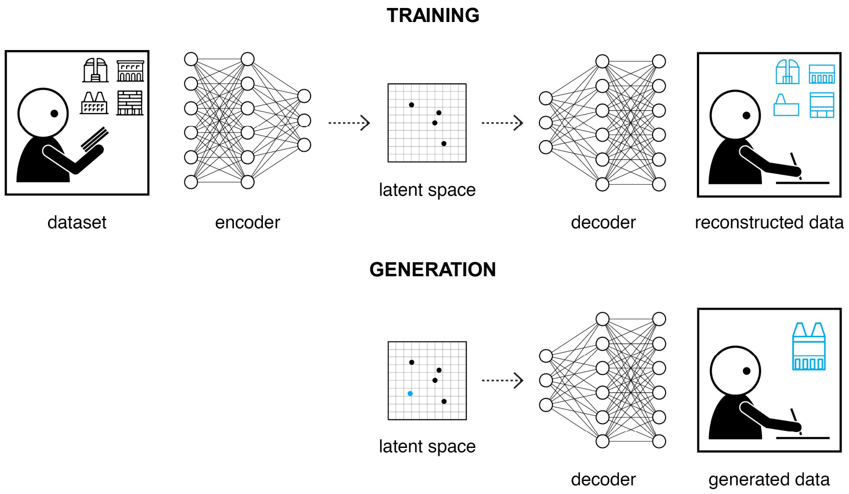
And this is the core of Gabriele's thesis: three applications that simulate different human learning mechanisms with AI. Design expertise, which we apply on a regular basis by relying on previous knowledge and design precedents. Playfulness, which is a principle that goes back to the hopefully happy years of our childhood, and is simulated with AI through Reinforcement Learning. The last mechanism is analogical reasoning, which is also the most ambitious for us, and involves transferring information between different but relatable knowledge domain. Many architects and engineers use this approach on a regular basis, and before passing the ball to Gabriele, I’ll just mention Frei Otto as an obvious example. The stage is your Gabriele, to discuss the AI applications.



I’ll start by introducing The first learning mechanisms on Expertise, which is here intended as the ability to extract meaningful design features from studying and analysing design precedents.

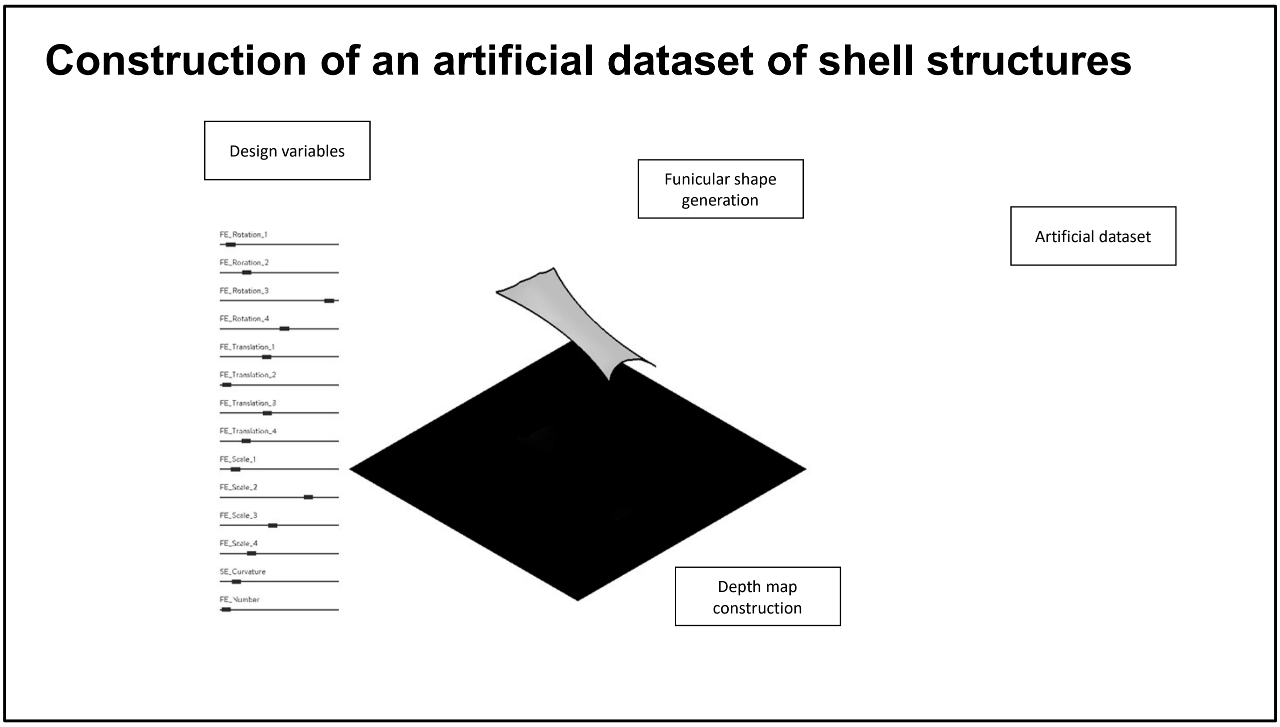
In the context of structural design, expertise allows an engineer to reimagine known structural typologies by recombining their knowledge of notable projects. In this slide, for example, you can see several variations of structures made of hypars that were elaborated by Engel Heino starting from projects by felix candela.

Generating Design Spaces with Variational Autoencoders



The process of learning from design precedents can be simulated by an AI technique named Variational autoendoer. This model features two components. The encoder learns to compress the information of a dataset of precedents into a low dimensional space (latent space), where similar input data are close to one another. The decoder learns to reconstruct the original input from its encoded representation.

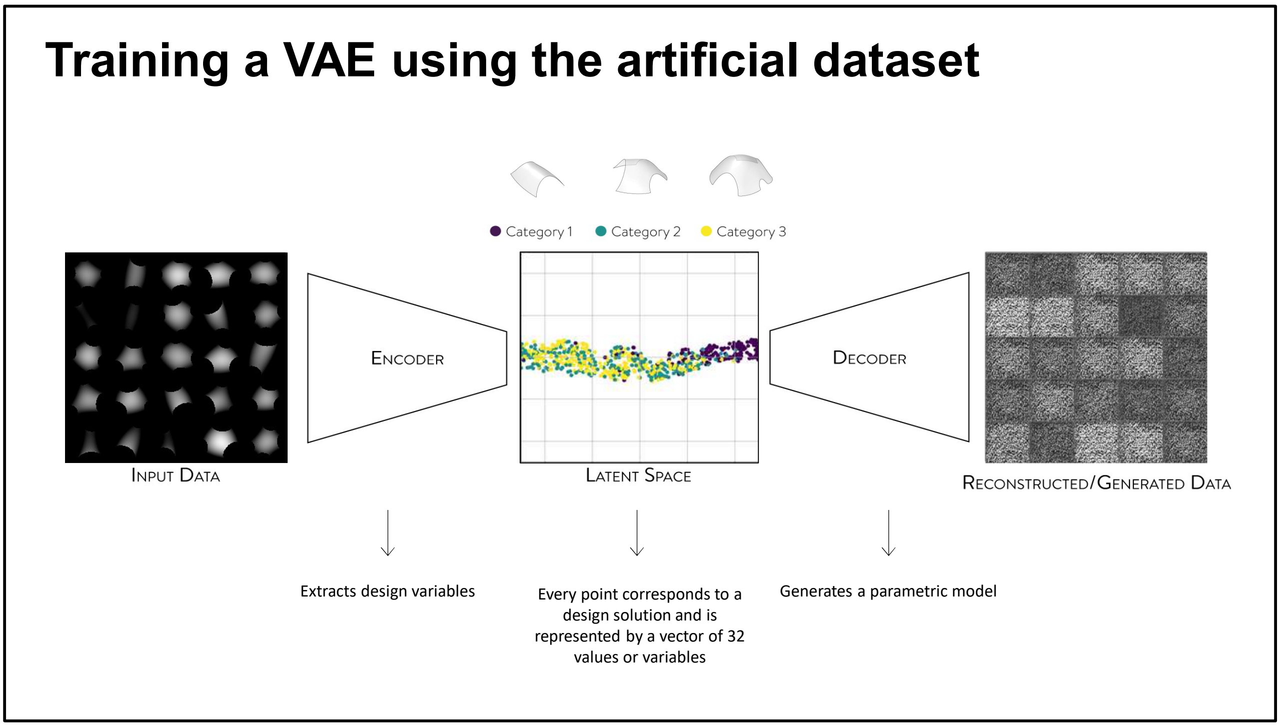
Once trained, the latent space can be sampled to generate new data that recombine the features of the analysied design precedents.



Let’s begin with the first application, in which the AI model was trained using a dataset of shell structures obtained with an ordinary parametric model.

This animation shows how the dataset was constructed through the parametric model. First, the algorithm generates a random combination of variables which control the boundary curves of the shell. These curves are then used as input for the dynamic relaxation algorithm to construct a 3D model of a funicular geometry. Finally, the 3D model is converted into a depth map and the image saved on the local hard drive.

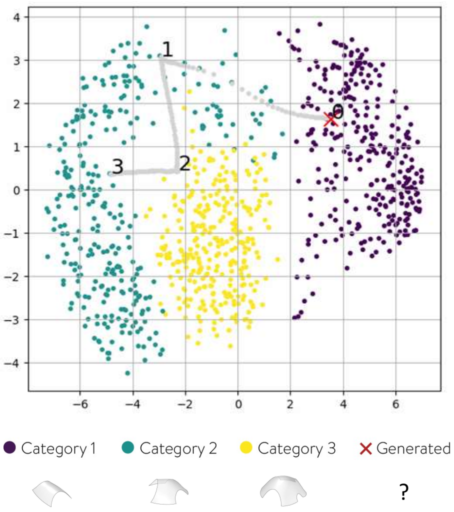
We used this process to generate 800 3d models. the dataset included three categories of shell structures: the first one with 2 openings or free edges, the second one with 3 openings, and the third one with 4 openings.



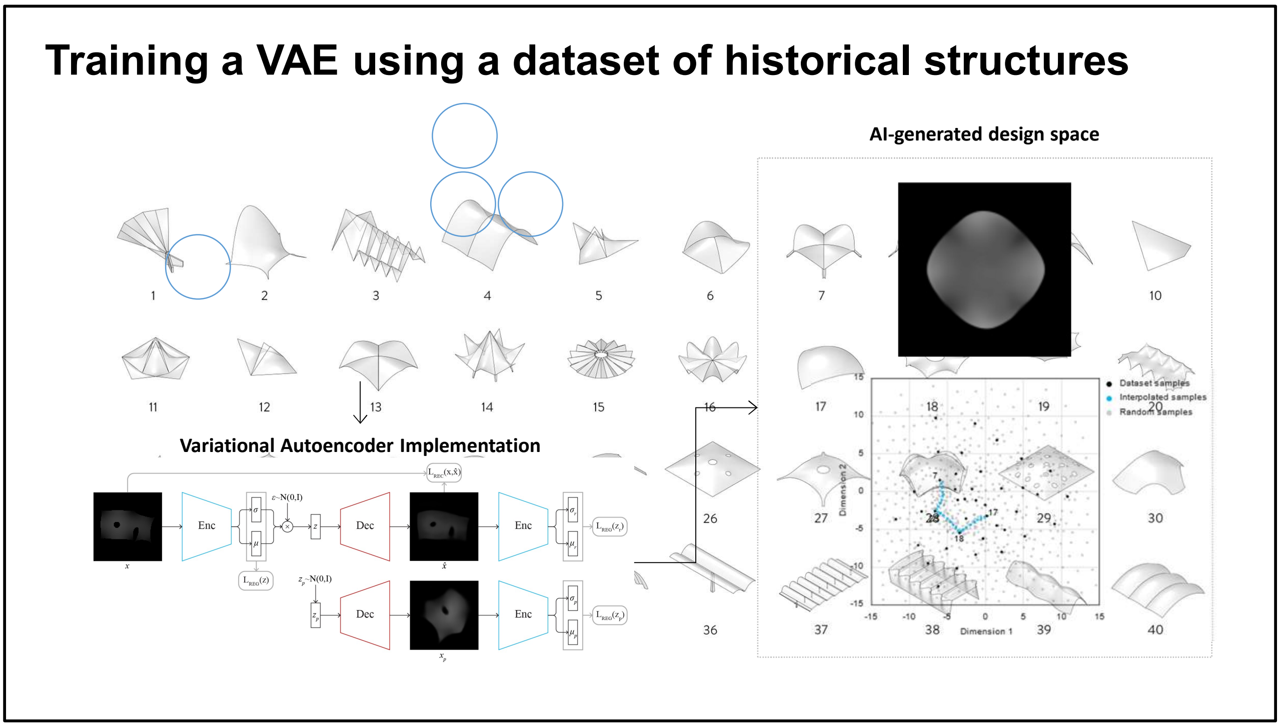
After constructing the dataset we trained the Variation Autoencoder. Training works this way: firstly, we fed the model with random data samples taken from the dataset, which are shown on the left. Then, we let the model perform two different tasks: number 1, compress the input data, that means extracting 'design variables', those shown in the graph at the center of the diagram, and number 2, decompress the 'design variables' to reconstruct the original input, which is shown on the right. Over several iterations, the Variational autoencoder learned to extract increasingly more useful variables and to reuse them to reconstruct increasingly finer details of the input data.

The animation at the centre illustrates the position of the combinations of variables describing all the 800 depth maps populating the dataset during each training iteration. Points colour represents the category of the 3D model represented by the depth map, that is shells with two, three or four openings. It can be seen that although the model has no information about these categories, it successfully learnt meaningful variables which embed information about the number of openings. This is demonstrated by the formation of three clusters in the latent space. An additional property of this space is that it can be sampled to generate new data.

Exploration of the AI-generated design space



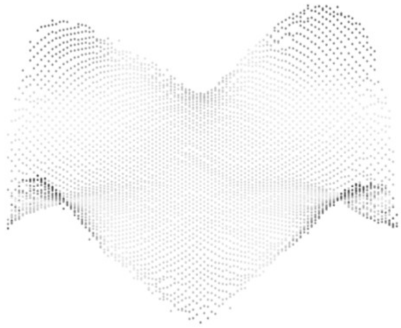
This slide shows an implementation of the trained model for the generation of new designs. On the left you can see a representation of the latent space with four numbers, which identify the position of 4 designs taken from the dataset. The variables describing these designs are linearly interpolated to create new combinations of variables, which are shown as grey dots along linear paths. These new combinations of variables are then decompressed into new designs, shown on the right, which are not part of the initial dataset.



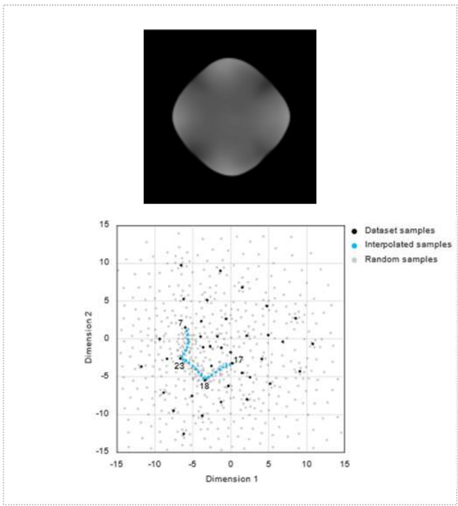
Having confirmed the ability of AI to extract design variables from a dataset of forms and recombine these variables to generate new designs, we wanted to see if AI could also be trained using a dataset of real projects. We developed a dataset of 40 shell and tensile structures designed by famous architects and engineers. We produced a 3D model for each project and converted it into a depth map to train the AI model.

After training we asked the model to interpolate between 4 different designs, thus generating hybrid structural forms that recombine the features characterising the projects that populated the dataset. This task was more challenging than the previous application, as each one of the four projects represent a unique structural typology.

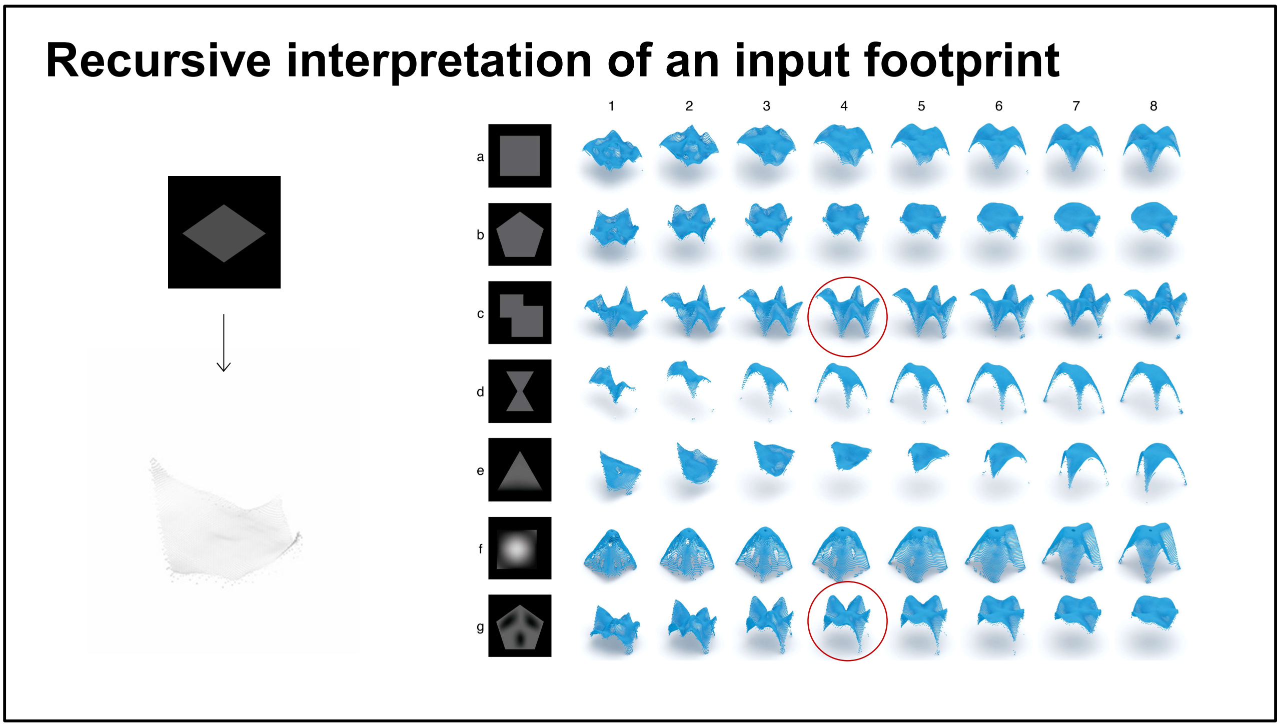
From depth maps to 3D models



AI-generated design space

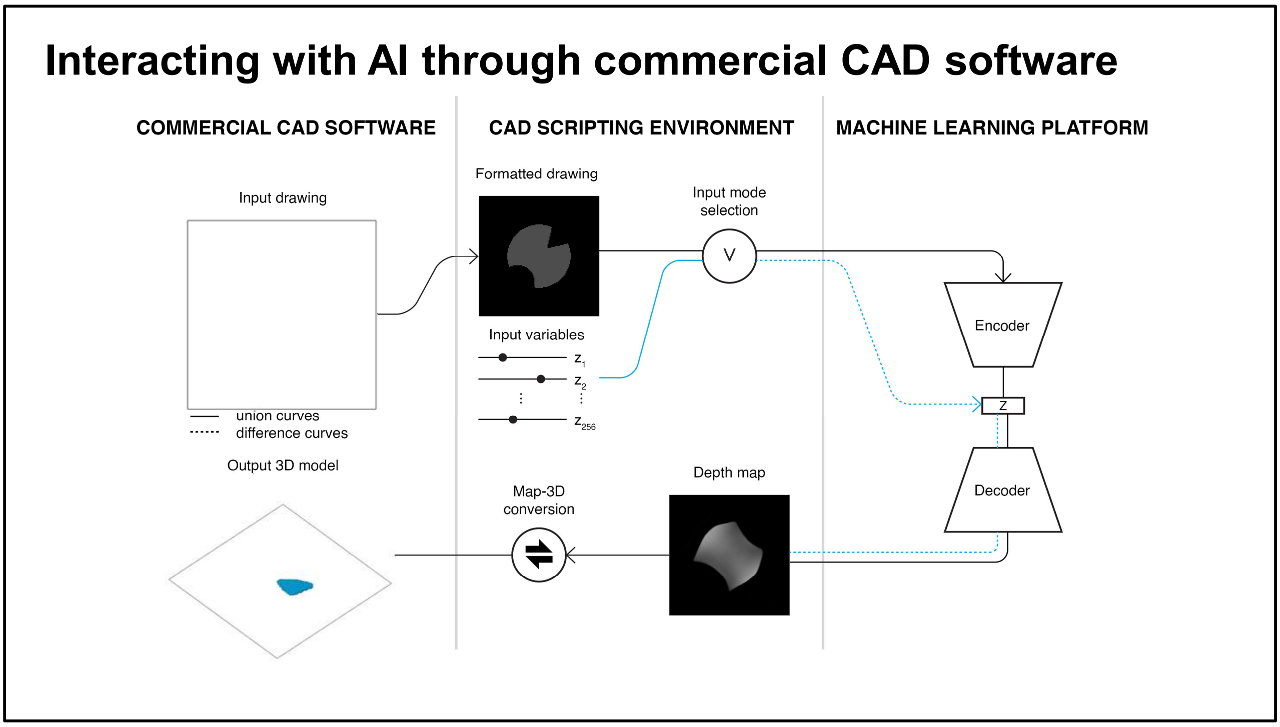


This slide shows an animation of the interpolation process in 3D. The animation was obtained by converting the depth maps into point clouds, demonstrating that depth maps embed all the information necessary to reconstruct 3D models and are therefore a suitable data format for analysing and generating shell and tensile structures with standard AI model architectures.

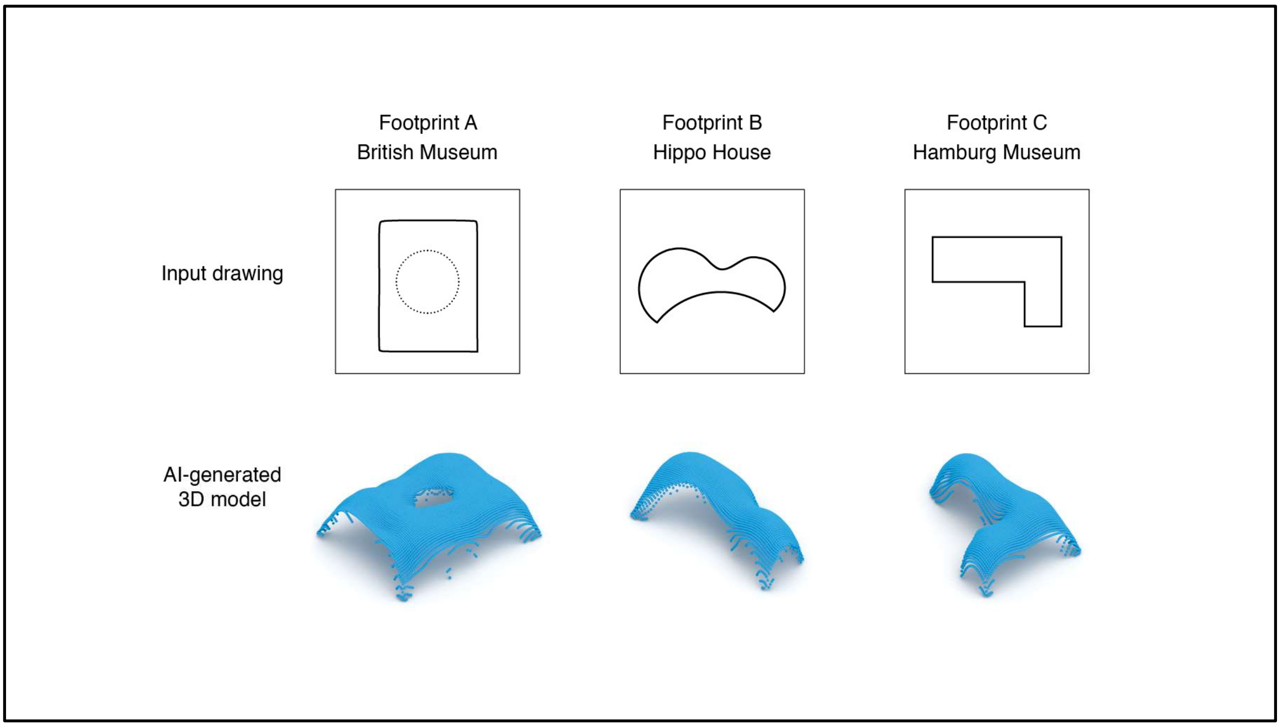


The AI model can also interpret inputs recursively, which allow generating multiple variations of a design form. This slide shows that starting from a single footprint the AI model could generate a series of 3D forms that the designer can analyse for further elaborations. The higher the number of interpretation the closer the resemblance between the new forms and those that populate the dataset.

In some cases we observed that the model could come up with solutions that are very different from the projects populating the dataset. For example, The solutions highlighted with a red circle show a shell with a large cantilever and a shell with a weavy opening.



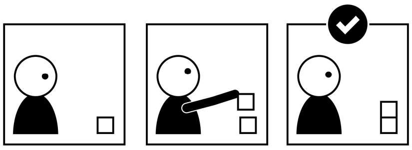
The AI model was then integrated within CAD software, which allowed to perform the two tasks of generation and interpretation more interactively. The slide show the three nodes managing the exchange of information between the designer and the AI model, whereby a input sketch is transformed into a 3D point cloud in real-time.

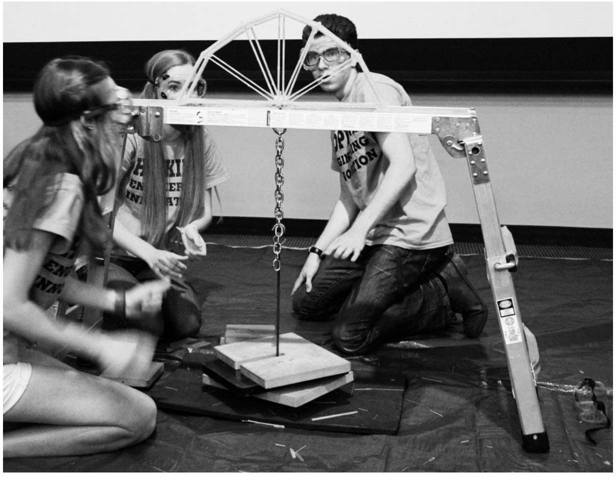


We tested the applicability of this interface to address real world design problems through a benchmark involving redesigning iconinc buildings, which were not included in the training dataset. Here we asked the model to produce an alternative design for the Hyppo House shell. The model was fed with a 2D image representing the building footprint and asked to produce several variations through recursive interpretation. The final design, shown on the right, is a shell form with multiple openings and supports. It represents an alternative option to the original design which is characterised instead by a continuous support along the edge.

Playfulness

Reinforcement Learning



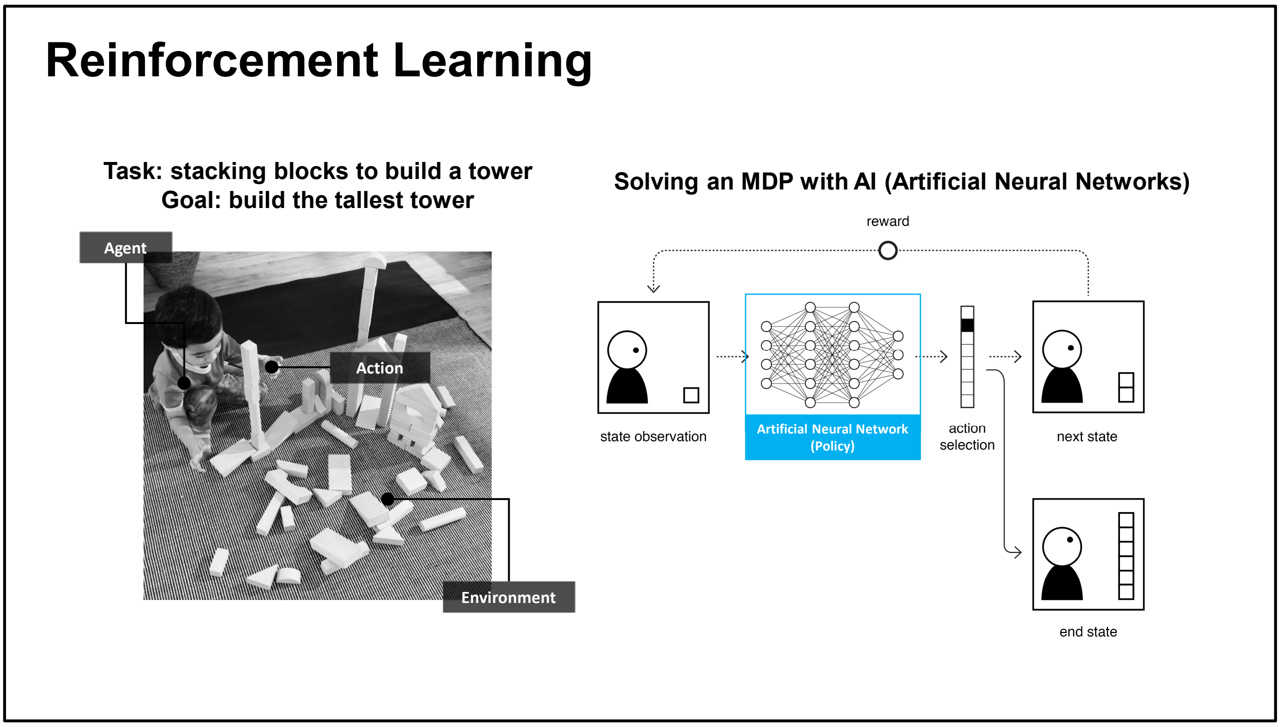


Students from the Johns Hopkins University testing a spaghetti bridge. *Image credits: Scott Calvert*

Let’s now introduce the second learning mechanisms, Playfulness, which involves learning through a process of free exploration.

In design, playfulness can be related to pedagogical activities like model-making, though which student learn by doing. On the left you see a simple reciproclar structures assembled from matches. To produce a model of this kind students only need to know the basic principles of reciprocal structure. Then they can explore and test different patterns by manipulating objects with their hands.

On the right you see another example in which students test scale models of bridges assembled from spaghetti. This is an exercise that stimulate a student’s creativity, and allow them to learn the relationship between form and performance in bridge design.



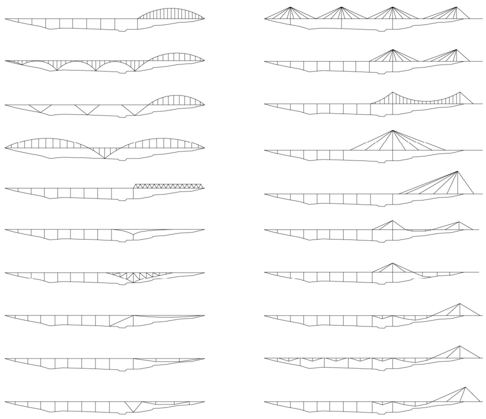
To understand how a learning mechanism based on playfulness can be simulated with AI, let’s consider a simple scenario in which a kid learns to build a tower by stacking blocks vertically.
Building the tallest tower can be considered the goal of the playing activity, during which the kid interacts with the environment by observing, picking and stacking blocks.

“This process is sequential, which means that the kid will have to perform multiple actions before completing the tasks. Moreover, these actions must be performed in the right order to accomplish the goal.

In the first iterations the kid will not know how the balance the blocks, which would make the tower collapse. After several attempts, however he will learn to pick the right block and position it in a way that a stable tower is produced. He will learn to do so through a process of reinforcement, that is by repeating those actions that lead to the achievement of a reward (in this case achieving a tall tower) and discarding the actions that made the tower collapse.
”

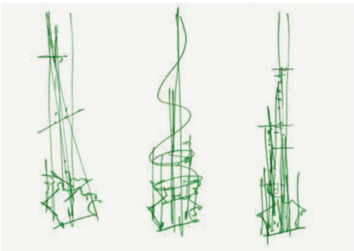
This sequential process can be modelled with AI, using a Reinforcement Learning approach. The problem in this case is learning a function that maps observations of an environment into actions, which are performed to accomplish a goal. The learning process does not rely on external datasets, but on the experience accumulated during training. In other words, the model learns from its own mistakes, similarly to a kid.

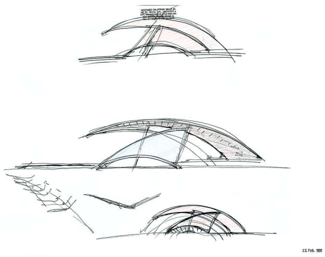
Application: learning to design a bridge structure



Alternative structural schemes developed by Jörg Schlaich
for the Obere Arben crossing.

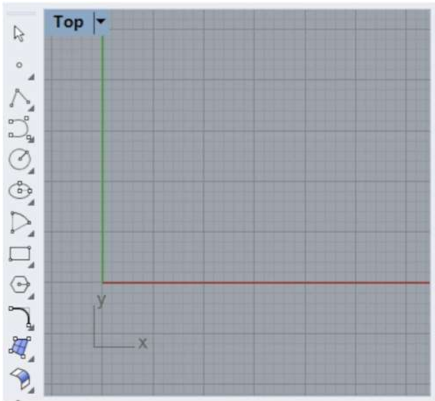
Application: learning to design a bridge structure

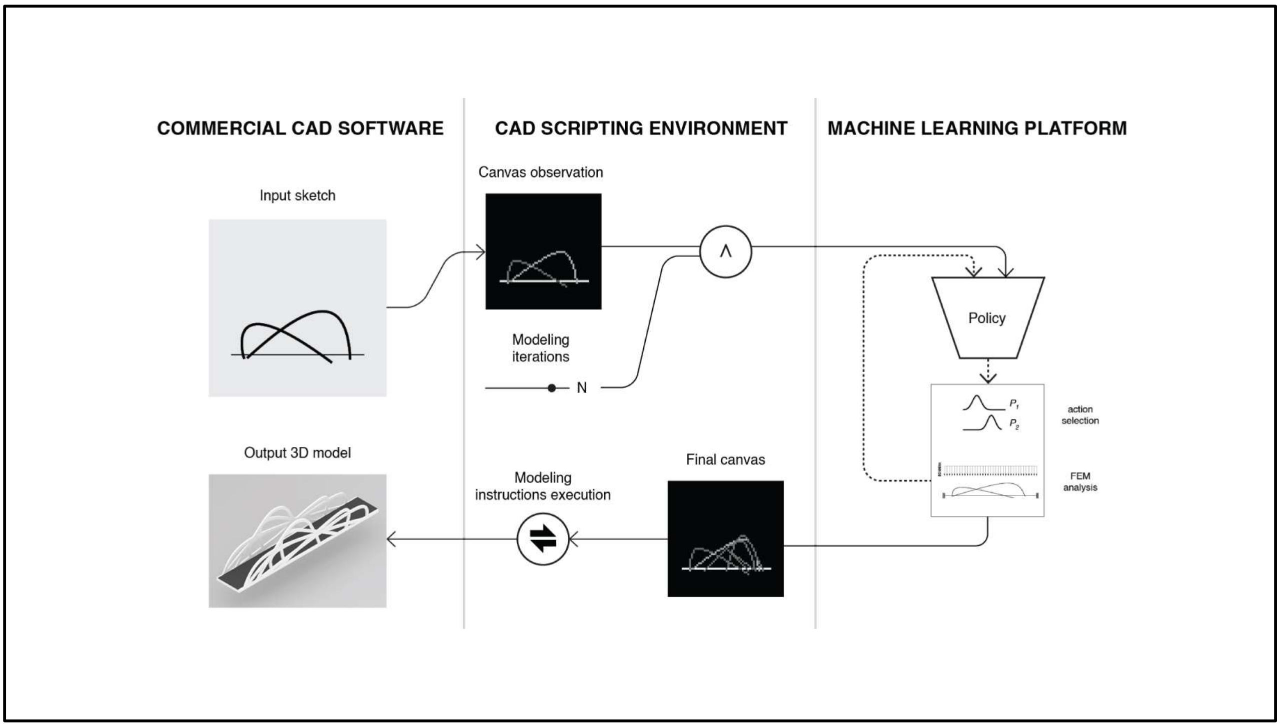




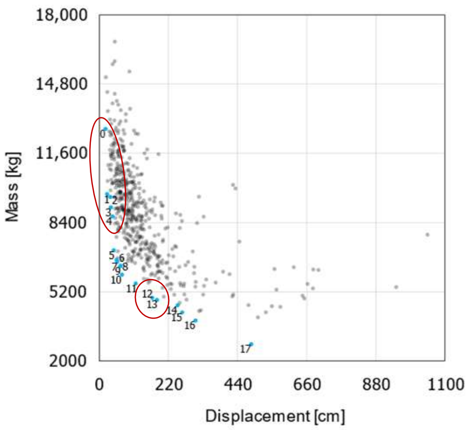
Conceptual sketches for The Shard (Renzo Piano, 2012)
Section studies for the Auditorio de Tenerife (Santiago Calatrava, 2003)

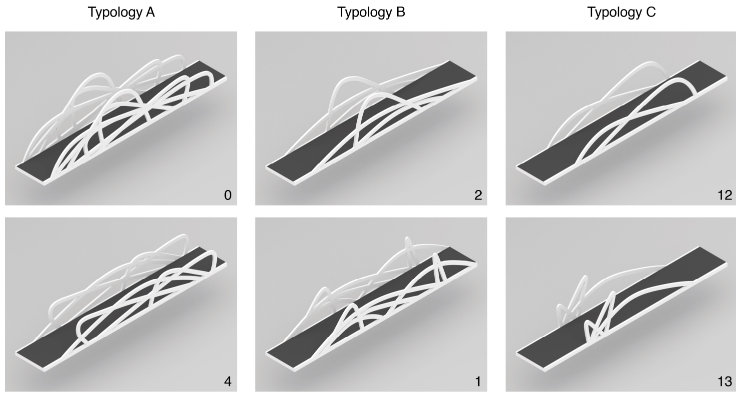
Updating the action space:
drawing by executing 2D CAD commands




COMMERCIAL CAD SOFTWARE
CAD SCRIPTING ENVIRONMENT
MACHINE LEARNING PLATFORM
Input sketch
Canvas observation
Modeling
iterations
N
Policy
action
selection
FEM
analysis
Output 3D model
Modeling
instructions execution
Final canvas

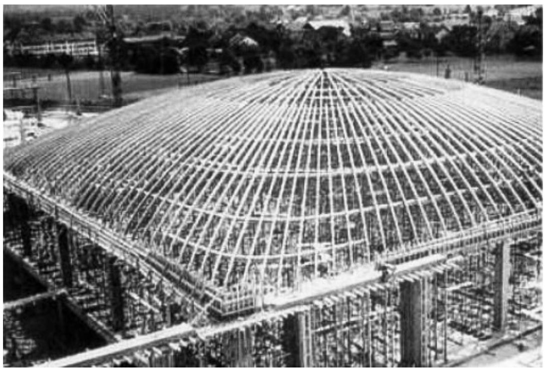
Using the trained agent: comparing the generated designs in a performance space





Analogical Reasoning

Learning from conditional exploration

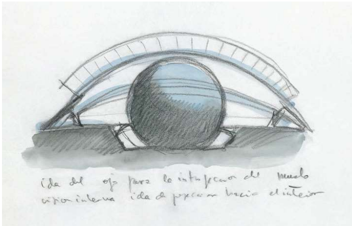


Chilton J., *Heinz Isler*, 2000

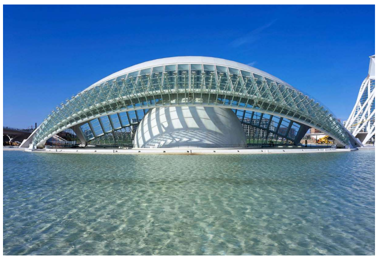
We are now at the third and last application for today's presentation.

We define analogical reasoning as the "ability to learn through a process of exploration that is conditioned by non-architectural or non-design-related forms. For example, engineer Heinz Isler is well known for his pneumatic form-finding method that was developed to design thin concrete shells. The physics behind the shape of a pillow clearly inspired the development of this method, in which the analogy is not purely formal, but goes deep into the structural behaviour of an inflated pillow.

Analogical reasoning in architecture: bio-informed design



UAE Pavilion at Dubai Expo (Calatrava, 2020)



L'Hemisfèric (Calatrava, 1998)

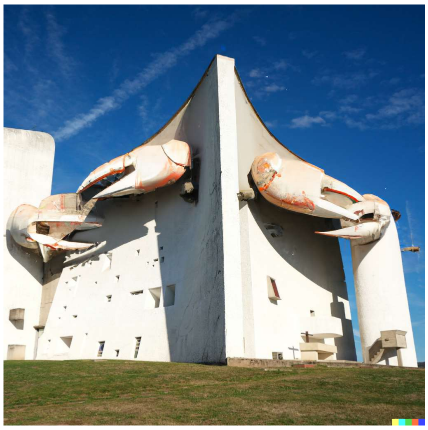
Notre Dame du Haut (Le Corbusier, 1954)

This point is particularly important when we look at bio-inspired designs. In most cases, the analogy between biology and architecture is purely formal. It's only in rare cases that a biological model or system is used as the source of inspiration but not necessarily its original shape.

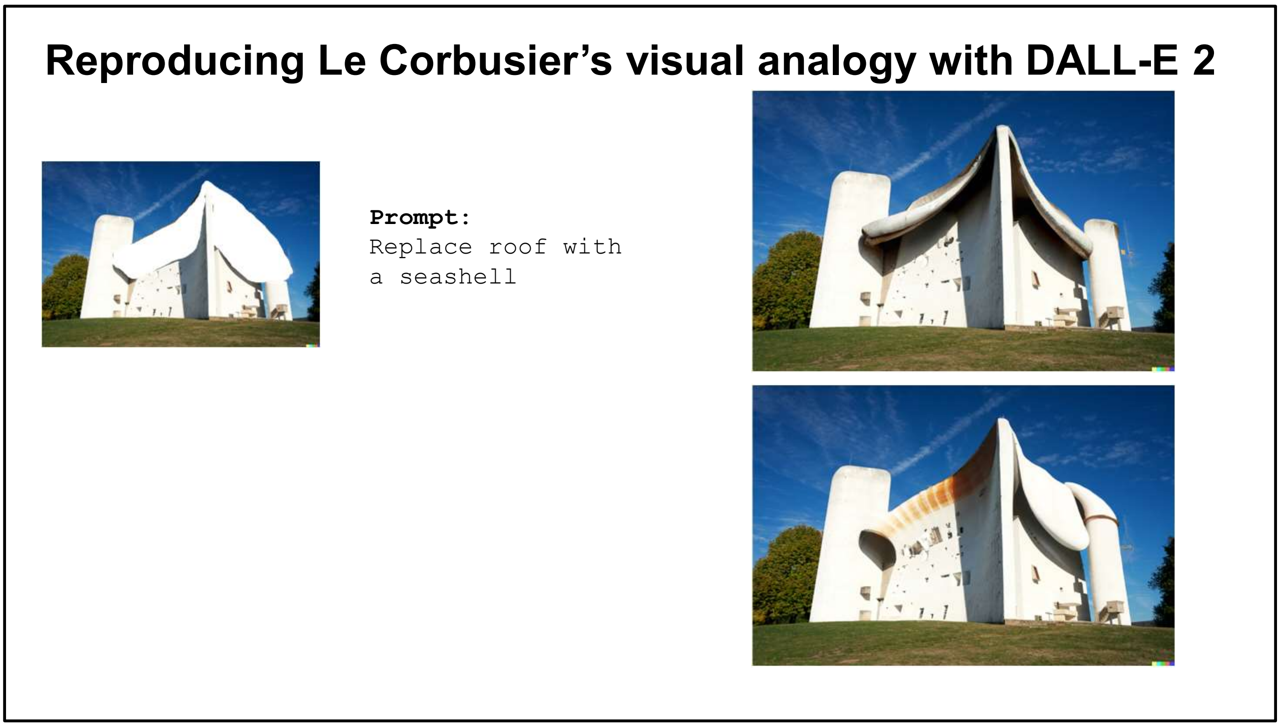
Let's have a look more in detail at the project on the right, Ronchamp by Le Corbusier.

Reproducing Le Corbusier’s visual analogy with DALL-E 2

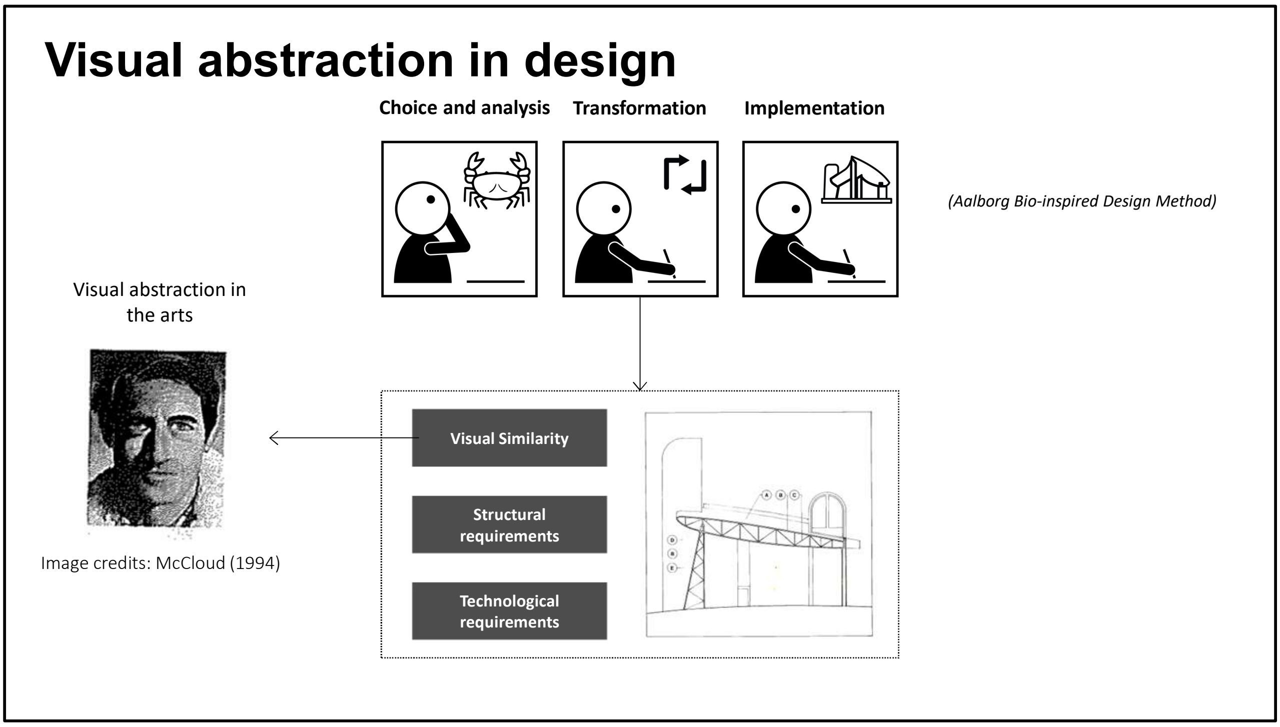
Prompt:
Replace roof with
a crab shell



If we use generative AI, for example Dall-E, to reproduce the design of Ronchamp’s roof only, we end up having a fairly literal analogy. The AI didn’t design a crab shell shaped roof, it simply replaced the original roofing system with crab shells!



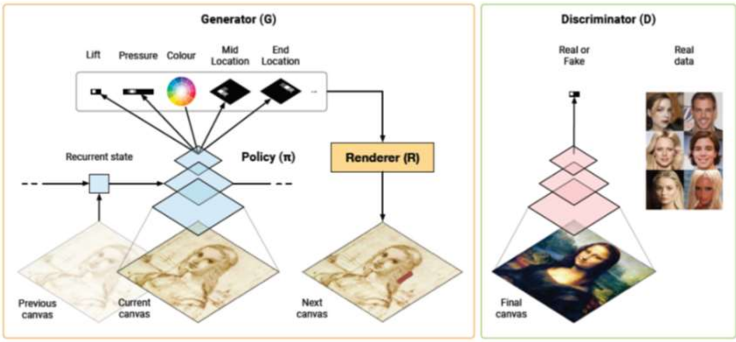
Changing the textual prompt to seashell provided us with these results. Even though the AI did redesign the roof effectively this time, by better blending the visual features of Le Corbusier's design with biological forms, from a design point of view, the images illustrate a biomorphic approach rather than a biologically informed approach to design.



To understand why Dall-E came up with these results, we need to look into the principles of visual abstraction, which is what Le Corbusier used. Visual abstraction is a process that reduces the complexity of a visual representation to enhance those features that are semantically relevant.

While in visual arts this process is primarily visual, as the name would suggest, in architectural design, performing visual abstraction requires one to develop a proposition from visual features, but also from structural, functional and other technical requirements of the biological source.

Synthesising Programs for Images using Adversarial Reinforced Learning (SPIRAL)



Ganin et al. (DeepMind, 2018)

Ganin et al. used this approach to train an AI model named “Synthesising Programs for Images using Reinforced Adversarial Learning” – SPIRAL – to generate visual abstractions of 2D images.

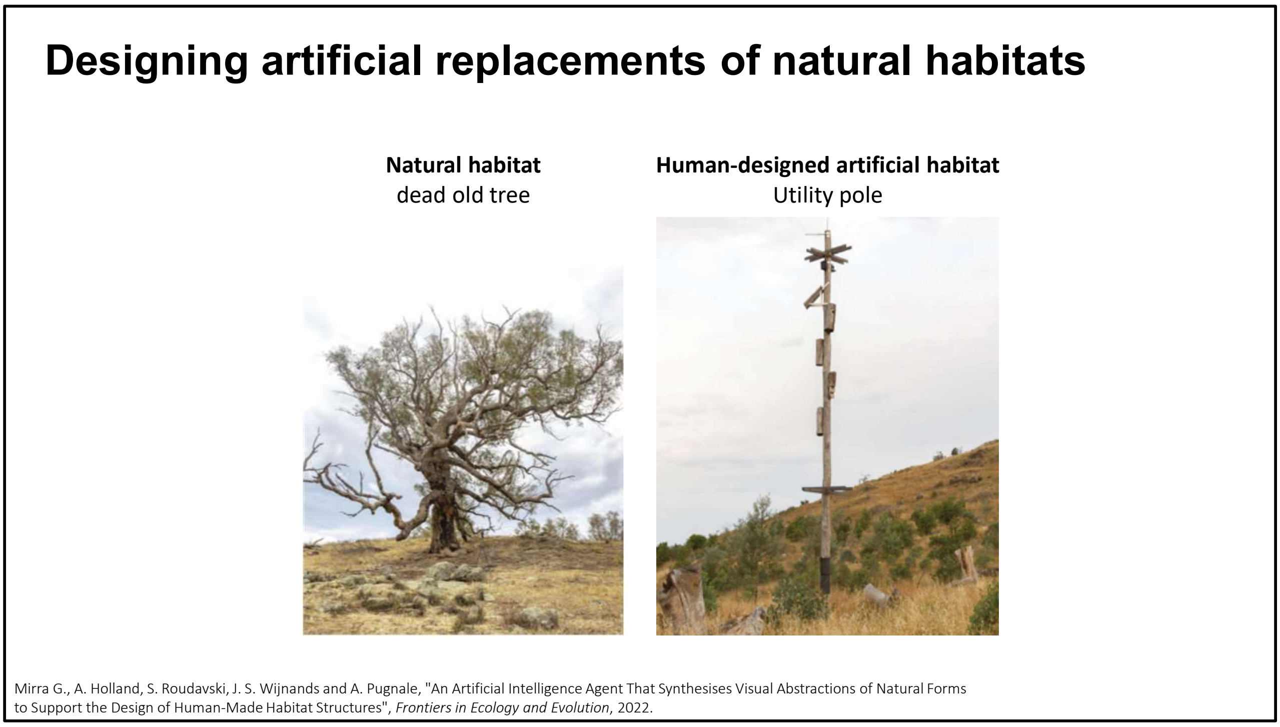
In SPIRAL, humans do not supply the agent with information about human drawings, and the agent must develop a strategy by trial and error.

Computer scientists achieved these objectives for 2D images applications by integrating an AI agent with a GAN discriminator. Ganin et al. (2018) used this approach to train an AI model named “Synthesising Programs for Images using Reinforced Adversarial Learning” – SPIRAL – to generate visual abstractions of 2D images. This application tasked the AI agent with learning a set of drawing actions to reproduce the key features of an image dataset within a drawing software.

Unlike previous applications of AI for inverse graphics (see section “Artificial Intelligence for Inverse Graphics”), in SPIRAL, humans do not supply the agent with information about human drawings, and the agent must develop a strategy by trial and error. An additional component – a GAN discriminator – produces a similarity metric that informs the agent about the quality of the synthesised images.

The SPIRAL model and its upgrade SPIRAL++

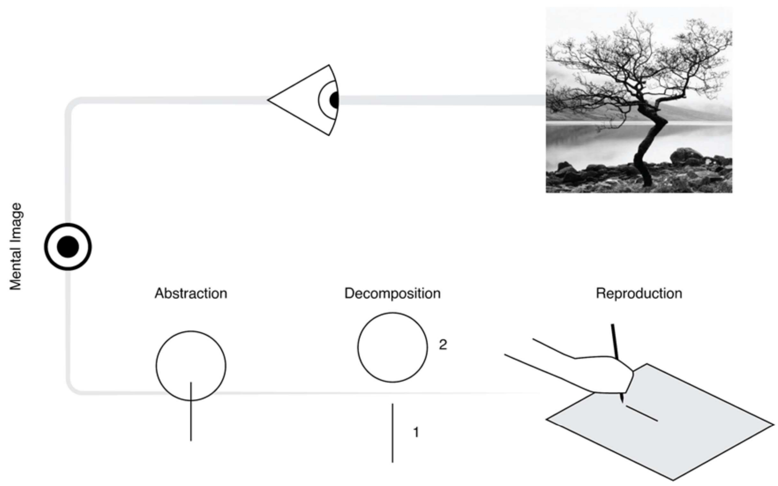
To date, studies applied SPIRAL and SPIRAL++ to generate human faces, handwritten digits, and images representing 3D scenes. There are no current implementations of SPIRAL for 3D data to our knowledge. Despite that, we selected this SPIRAL model for further development because, unlike GANs, it can autonomously decide the number and characteristics of the features to reproduce in a synthetic visual abstraction. Furthermore, unlike current applications for inverse graphics, it does not require a dataset of drawing instructions and supports fine control of the abstraction process through the specification of constraints. Later in the article, we describe how our version of SPIRAL can extract visual features from a 3D dataset of tree forms and synthesise visual abstractions of such forms.



Our application uses the Markov Decision Process to design artificial replacements for dead old trees, which would obviously require us to wait for decades if we wanted to replace them with other trees. This application was developed with other colleagues, Stanislav Roudavski and Alexander Holland, and our aim was to abstract those visual features of natural trees that birds find appealing in order to select their nesting place. The goal was to come up with something better than the utility pole that you can see on the right on this slide, which is the solution for deforested areas at the moment.

Analogical reasoning: abstracting natural forms

1. Which (and how many) visual features of natural trees should be reproduced in an artificial tree?
2. How to produce artificial trees that maintain the structural complexity of natural trees while being easy to build?

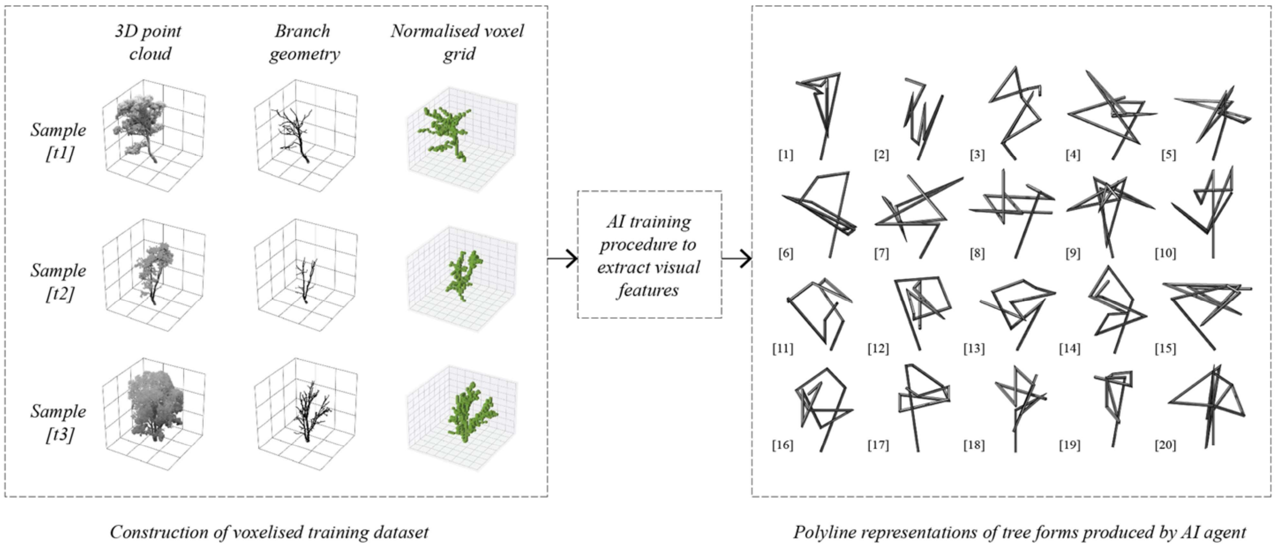


Mirra, G., Holland, A., Roudavski, S., Wijnands, J. S., & Pugnale, A. An Artificial Intelligence Agent That Synthesises Visual Abstractions of Natural Forms to Support the Design of Human-Made Habitat Structures. Frontiers in Ecology and Evolution. (2022).

Through the analysis of literature on AI, we found that there are no models that are specifically designed to synthesise visual abstractions in 3D.

Since many bird species and other living organisms live in large old trees and the number of such trees is diminishing, research on the design and construction of effective human-made replacements is a significant international priority [1, 2, 3, 4]. In 2022, Mirra et al. [1] developed an Artificial Intelligence (AI) agent trained to produce line sketches of large old trees to capture and reproduce their characteristic features in a simplified form. This AI agent produces visual abstractions of tree geometries used to train it, and such abstractions retain features that animals look for when choosing nests or perch locations.

Ai-driven synthesis of visual abstractions



Mirra, G., Holland, A., Roudavski, S., Wijnands, J. S., & Pugnale, A. An Artificial Intelligence Agent That Synthesises Visual Abstractions of Natural Forms to Support the Design of Human-Made Habitat Structures. Frontiers in Ecology and Evolution. (2022).

Figure 1 illustrates the workflow used by Mirra et al. [1] to generate visual abstractions of tree forms. This method involves preparing a 3D dataset of natural trees, training an AI agent to extract geometric features from such models and producing simplified representations that preserve the visual features of the original geometries. The AI model generates these visual abstractions by tracing line segments.

The training dataset comprised high-resolution 3D point clouds [2], from which the researchers first isolated the most relevant features – the points representing the branch geometry – and then clustered and converted them into line segments. This data format was translated into a 32x32 voxel representation to suit AI training requirements: the goal was to minimise the computational cost of geometry processing while preserving essential information about the structural complexity and mass distribution of the original trees (Figure 1, on the left). At each iteration of reinforcement learning, the AI agent observed samples from the dataset of voxelised tree forms and attempted to reproduce their geometries by drawing lines within a 3D canvas. The AI agent was constrained to use a maximum of 10 lines for each attempt, forcing it to generate simplified representations – or visual abstractions – of natural trees. Figure 1, on the right, shows 20 visual abstractions generated by the AI agent during the last training iterations.

The researchers analysed the performance of these 3D polylines using two metrics – the Perch Index and the Complexity Index – and discovered that their AI-generated forms were more similar to natural canopy structures than the reference human-designed trees used for the comparison. Moreover, many of the generated solutions contained diverse canopy shapes and branch distributions.

How do we translate the
AI-generated outputs
into a structural system
(an artificial habitat)?

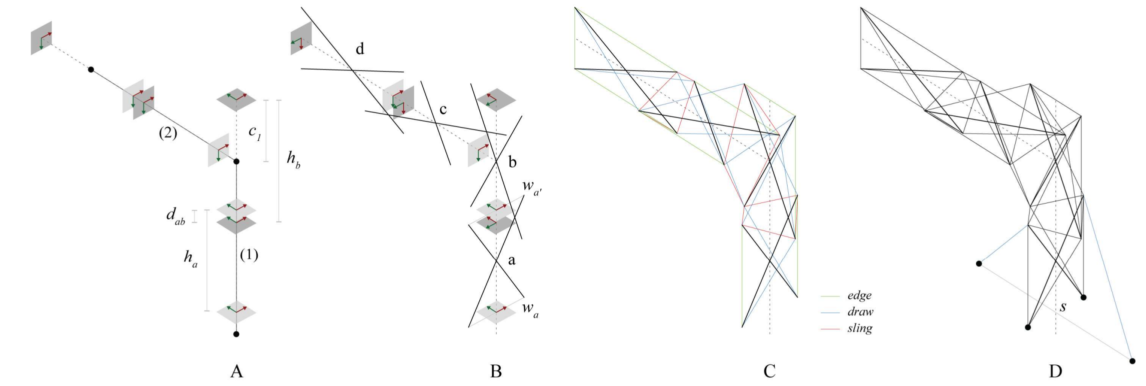
"X-piece", Kenneth Snelson, 1948

Figure 1 illustrates the workflow used by Mirra et al. [1] to generate visual abstractions of tree forms. This method involves preparing a 3D dataset of natural trees, training an AI agent to extract geometric features from such models and producing simplified representations that preserve the visual features of the original geometries. The AI model generates these visual abstractions by tracing line segments.

The training dataset comprised high-resolution 3D point clouds [2], from which the researchers first isolated the most relevant features – the points representing the branch geometry – and then clustered and converted them into line segments. This data format was translated into a 32x32 voxel representation to suit AI training requirements: the goal was to minimise the computational cost of geometry processing while preserving essential information about the structural complexity and mass distribution of the original trees (Figure 1, on the left). At each iteration of reinforcement learning, the AI agent observed samples from the dataset of voxelised tree forms and attempted to reproduce their geometries by drawing lines within a 3D canvas. The AI agent was constrained to use a maximum of 10 lines for each attempt, forcing it to generate simplified representations – or visual abstractions – of natural trees. Figure 1, on the right, shows 20 visual abstractions generated by the AI agent during the last training iterations.

The researchers analysed the performance of these 3D polylines using two metrics – the Perch Index and the Complexity Index – and discovered that their AI-generated forms were more similar to natural canopy structures than the reference human-designed trees used for the comparison. Moreover, many of the generated solutions contained diverse canopy shapes and branch distributions.

From visual abstractions to tensegrity configurations



Step A: construct pairs of planes to orient the kite modules

The polyline segments are extended by distance c, corresponding to half the height of the modules located at the corner (module b) and the end of the polyline (module d).

Step B: construct the axes of the struts that define the kite modules

Rotate every second pair of planes by 90 degrees about their Z-axis. Construct the end points of the struts along the X-axis of the reference planes.

Step C: generate a network of cables

(1) edge lines, defining the sides of each kite module; (2) draw lines, connecting two consecutive modules (3) sling lines, connecting and suspending consecutive modules.

Step D: add two additional tension lines to stabilise the structure and anchor it to the ground

Variable s controls the distance between the anchoring points.

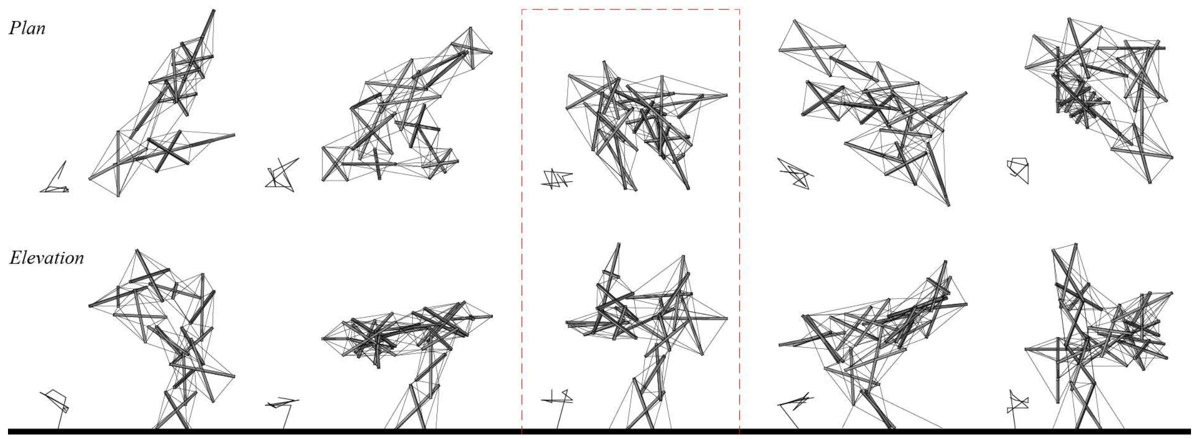
Automated design strategies for tensegrity structures based on modules with three or more compressed elements are usually designed from a starting reference surface or polyhedric geometry [10, 11]. Instead of using a three-strut module, common for tensegrity towers, we implemented a simpler quasi-tensegrity structural system inspired by Kenneth Snelson's X-piece sculpture of 1948. Acknowledged as the first built tensegrity, this structure consists of two X-shaped compressed elements and 14 cables in tension. One X module is called a kite. Snelson later described this and other tensegrity systems in his 1965 US3169611A patent [12].

This tensegrity configuration facilitated the exploration of stacking modules that deviate from the vertical direction of growth.

Step A consists of constructing pairs of planes to orient the kite modules. The initial polyline segments are extended by a fixed distance c, corresponding to half the height of the modules located at the corner (module b) and the end of the polyline (module d). This step also involves calculating the number of additional plane pairs that can be placed along each segment. The variable h determines the spacing between the planes, whereas depth d adjusts the relative positioning of consecutive plane pairs. For the 2-segment L-shaped polyline shown in Figure 2, this process

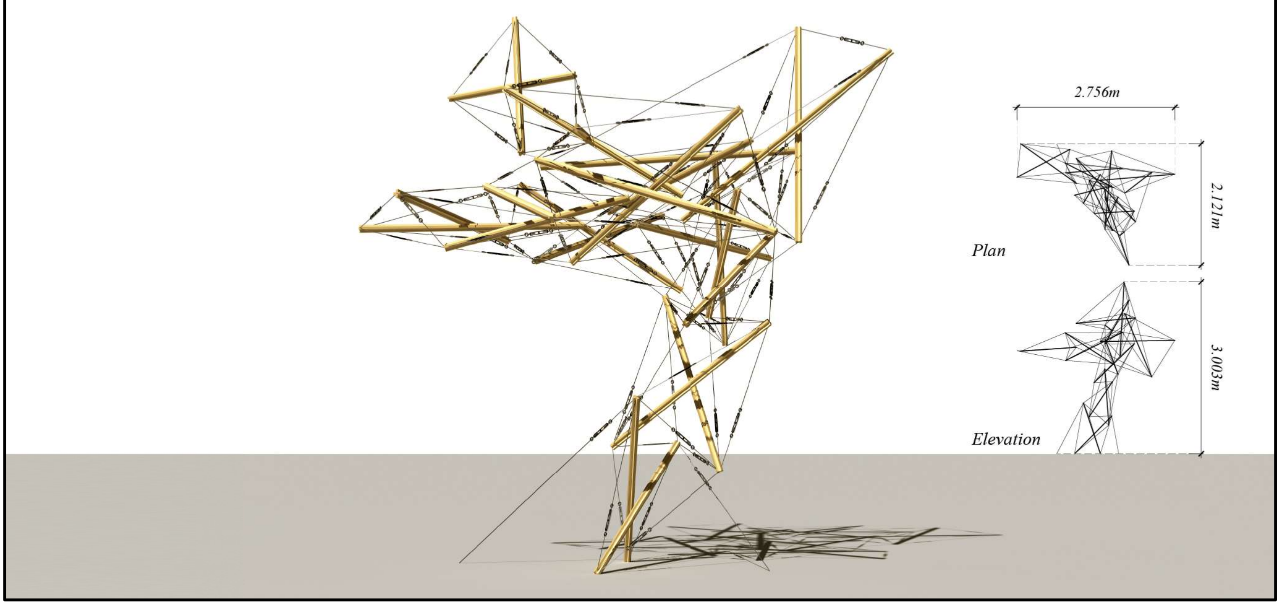
results in constructing one additional plane pair per segment, corresponding to modules a and c.
Step B involves constructing the axes of the struts that define the kite modules. We rotate every second pair of planes by 90 degrees about their Z-axis to orient the kites correctly. For each module, we then construct the start and end points of the struts along the X-axis of the reference planes. We created the variable w to control the distance between these points and define the width of kite modules.
Step C generates a network of lines that represent cables. We construct three types of tension lines, following Snelson’s kite design: (1) edge lines, defining the sides of each kite module; (2) draw lines, connecting two consecutive modules and pulling them towards each other; (3) sling lines, also connecting and suspending consecutive modules.
Step D adds two additional tension lines to stabilise the structure and anchor it to the ground. Variable s controls the distance between the anchoring points.

Design options

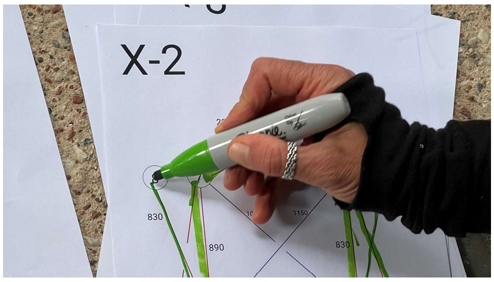


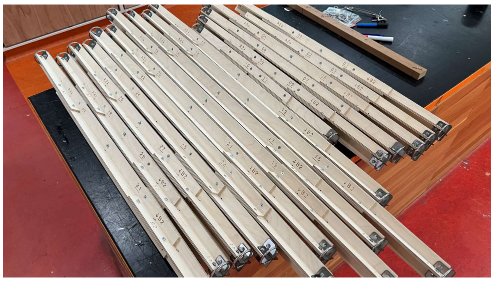
This procedure allowed us to rapidly create 3D models of tensegrity systems from the 20 AI-generated visual abstractions defined in Mirra et al. [1]. We selected visual abstractions prioritising single trunk and branch-like structures as main features. We also selected AI-generated polylines with long horizontal lines, as these configurations are more suitable for perching.
Using these criteria, we shortlisted solutions 6, 7, 8, 15 and 17, as shown in Figure 3. We selected solution 8 for the pavilion design because it consists of several near-horizontal kite elements.
While our pavilion design is a prototype of an artificial habitat structure, it should also be seen as a creative output. Other tensegrity topologies or structural systems could work equally well. However, the proposed design is a significant advancement over existing structures, such as utility poles.

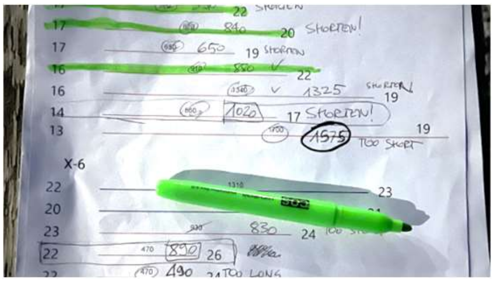
Final design

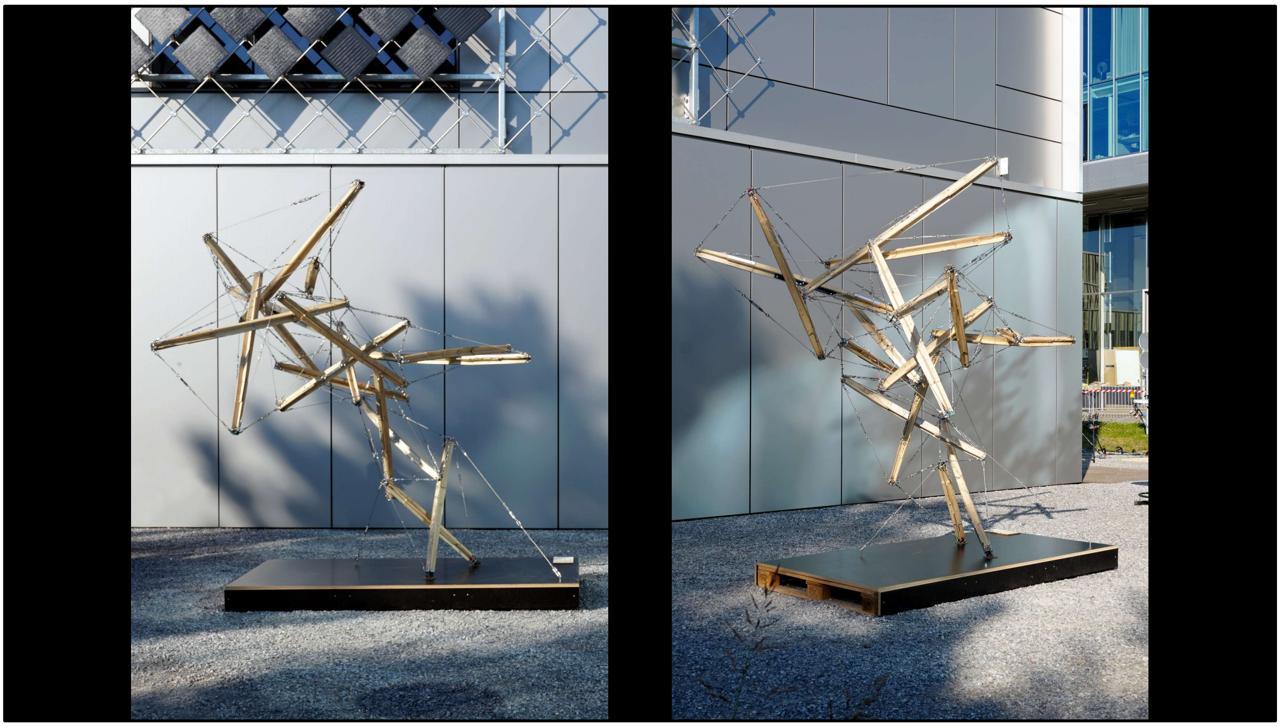


Packing list for the IASS 2024

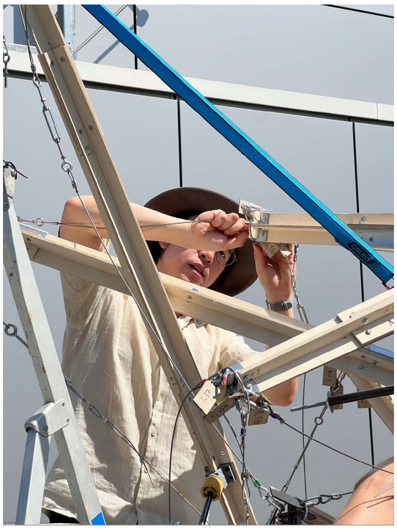






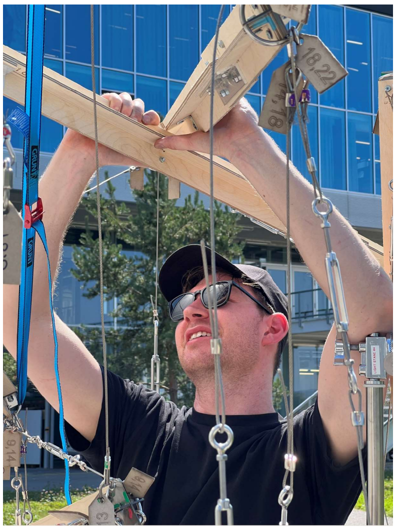


Assembly team in Zurich



Michael

Gabriele



Jack

Assembly team in Melbourne

Matthew
(and Daniel)

Michael 2

Sofia

LSAA Lightweight Structures Association Australasia Inc.

Lightweight Structures Association Australia