LEARNING FRAME BY FRAME

A material-first generative structural design assistant for repurposed bicycle frames

The transition towards the circular economy requires strategies that keep materials in use. Within the built environment, however, current practices remain focussed around linear economic principles, and repurposing as a practice remains relatively under-investigated in current research. This thesis operates under the presumption that there is potential in repurposing as a meaningful circular economic practice, as many objects and materials can serve functions for which they are not inherently designed.

Designing with irregular reclaimed stock of any kind is a complex and creatively demanding task. Computational design can support this task, but remains entrenched in top-down parametric approaches, which impose a target design model onto the constrained stock, often warranting cutting or modification. As an alternative, this thesis proposes material-first computational design as a general concept for bottom-up generative approaches. Drawing on frameworks of design and computational design processes, it positions the algorithm as the guiding theme of the design process and the stock as its point of departure. Critically, it mandates that stock is used without cutting or modification.

Current bottom-up research, however, still relies on cutting or refabrication to reconcile irregular elements. In recent years, material-first approaches which do not cut stock have come to fruition through the application of reinforcement learning. These approaches remain largely limited to reuse, as opposed to repurposing. This thesis therefore applies material-first computational design to a constrained stock of repurposed objects.

To this end, a methodology is formulated based on generative principles to create structural assemblies from repurposed objects. A compact set of rules is created on which a reinforcement learning agent can be trained. These rules are explored through an analogical practice known as 'design games'. They are then applied to a design scenario, through which a design algorithm is formulated. The algorithm places objects sequentially, and a set of passive logics completes the structure with connections and tension elements. The algorithm is first validated via genetic optimization and then used to train reinforcement learning agents with Proximal Policy Optimization.

The genetic optimization proved insufficient for structural optimization when exposed to the full action space, though it outperformed the trained agents in a more constrained action space. The trained agents, on the other hand, generate a more diverse set of designs in much quicker succession, are capable of interacting with a partially completed assembly, and generalize relatively well to other extractions from the database on which they were trained, though they struggle with more radically shaped elements.

A4 VERSION - Date of Defense: **04-11-2026**

Delft University of Technology (TU Delft)

Faculty of Architecture and the Built Environment

Msc Architecture, Urbanism and Building Sciences

Track: Building Technology

STUDENT	Rens Jansen | 4859537
FIRST MENTOR	Dr. Gabriele Mirra
SECONDARY MENTOR	Dr. Charalampos Andriotis
EXTERNAL	Dr. Rebecca Hartwell

Material-First Computational Design:

Design practices which employ computational means to create systems or models which generate form through a chosen stock of materials, rather than a superimposed target design model, and which seeks not to modify these materials, but ideally leverage the qualities which makes the stock unique and varied.

Terminology

1.	CE	|	Circulair Economy
2.	BE	|	Built Environment
3.	CD	|	Computational Design
4.	PD	|	Parametric Design
5.	GD	|	Generative Design
6.	AD	|	Algorithmic Design
7.	RL	|	Reinforcement Learning
8.	MDP	|	Markov Decision Processes
9.	MLP	|	Multi-Layer Perceptrons

I would like to thank

Dr. Gabriele Mirra

Dr. Charalampos Andriotis

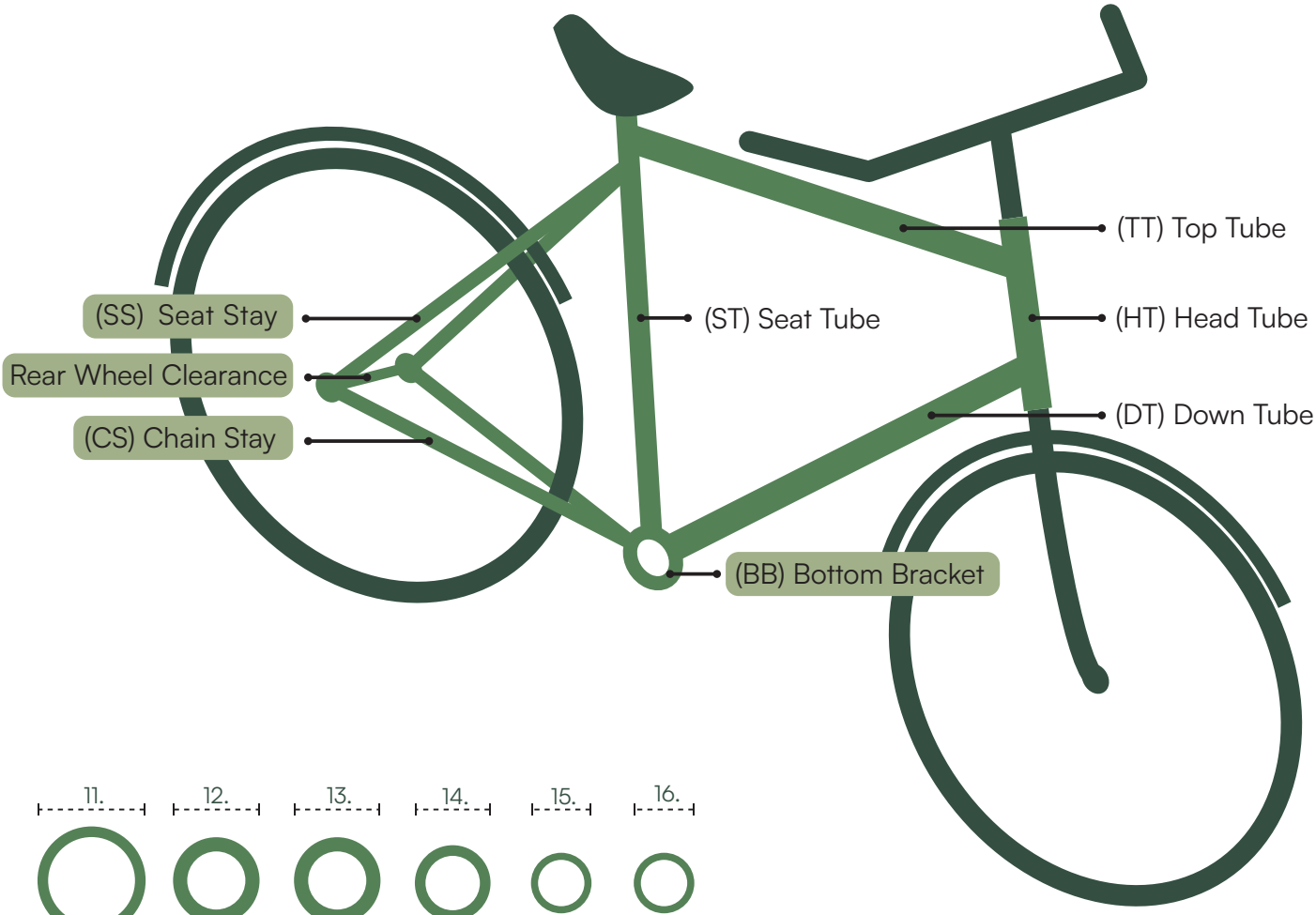
Dr. Rebecca Hartwell

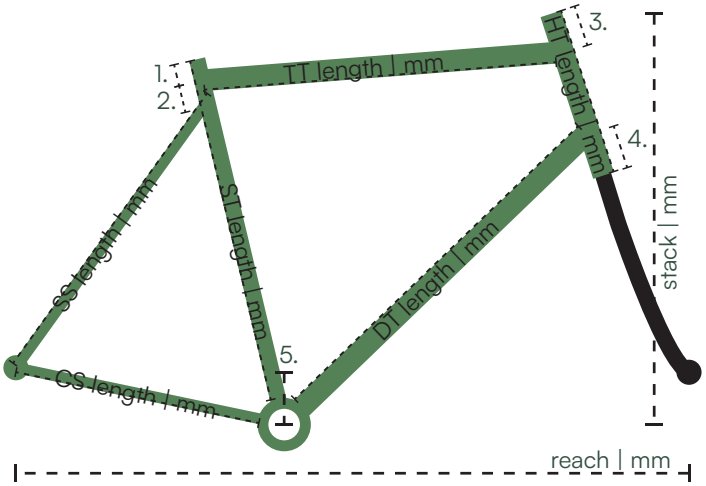
For their time, insight, enthousiasm, and above all, patience.

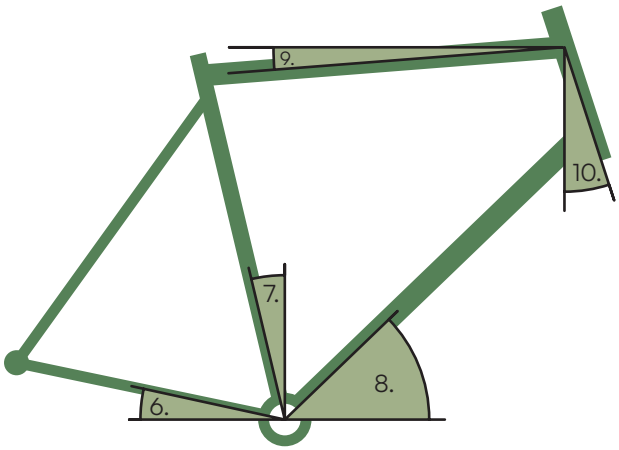
This thesis is dedicated to my grandfather,

who fueled my fascinations,

without whom I would not have chosen this career path.







1. seat tube extension
2. seat stay offset
3. head tube upper extension
4. head tube lower extension
5. bottom bracket drop
6. chain stay angle
7. seat tube angle
8. down tube angle
9. top tube angle
10. head tube angle
11. head tube outer diameter
12. down tube outer diameter
13. top tube outer diameter
14. seat tube outer diameter
15. chain stay outer diameter
16. seat stay outer diameter

1.0 Introduction 6
The Circular Economy 7
The Built Environment 9
The Generative Potential 11
Reinforcement Learning 13
Case Studies - Material and Design 14
Problem Statement 15
Value Proposition: Material-First Computational Design 15
2.0 Research Questions 16
Main Research Question 17
Sub-questions 17
3.0 Manifesto of Material-First Computational Design 18
Chapter Objective 19
3.1 Theoretical Framework 19
. Design processes in general 19
. Computational design processes 20
. Computational Design Typologies 21
. Material-first computational Design 21
. Design Games and Shape Grammars 22
3.2 Methodology 24
. Design Games Setup 24
. Setup — Objects 24
. Setup — Connections 26
. Shape Grammar Formalism 26
3.3 Design Game Results 27
. The Pyramid 27
. The Star 28
. The Tube 29
. Final Generative Principle 30
3.4 Intermediary Conclusions 30
4.0 Material Case Study: Bicycle Frames 32
Chapter Objective 33
4.1 Theoretical Framework 33
. Geometrical Qualities 33
4.2 Methodology 34
. Parametric Dataset 34
. Extracted Dataset 35
. Extracted Dataset Construction and Validation 35
. Extracted Dataset Simplification 36
. Extracted Dataset Analysis and Filtering 37
. Extracted Dataset Selection 37
4.4 Intermediary conclusions 38
5.0 Design Case Study: Cycling Bridge 40
Chapter Objective 41
5.1 Theoretical Framework 41
. Structural aspects of bridge design 41
. Structural Criteria 42
5.2 Methodology 43
. Case Study Selection 43
. Self-weight Optimization 45
5.3 Intermediate Conclusions 46
6.0 Material First Generative Algorithm: Overview 48
Chapter Objective 49
6.1 Methodology 49
. Main Functionality 49
. Structural Post-Processing 52
. Materialization 54
. Performance Metrics 54
. Scripting Setup 54
6.2 Preliminary Conclusions 55
7.0 Genetic Validation 56
Objective 57
7.1 Theoretical Framework 57
7.2 Methodology 58
. Full Action Space Variant 58
. Constrained Action Space Variant 59
. Structural Optimization Variant 60
7.3 Results 60
. Full action space 61
. Constrained Action Space 62
. Example Solutions 64
7.4 Preliminary Conclusions 64
8.0 Reinforcement Learning Application 66
Chapter Objective 67
8.1 Theoretical Framework 67
8.2 Method 70
. The Environment 70
. The Action Space 70
. The Observation Space 71
. Extractor Setup 71
. The Reward Function 72
. Penalty Functions 73
. Hyper parameters 73
. Evaluating success 73
8.3 Results 74
. Non-FEA Agents 74
. FEA Agents 75
. Visual Analysis 77
. Interactive Capacity 77
. Generalized Capacity 78
8.4 Preliminary Conclusions 78
9.0 Conclusions 80
10.0 Discussion 84
Positioning in the Current Landscape 85
The Knowledge Gap 85
Contextualizing the results 85
Next Steps 85
11.0 Reflection 88
12.0 References 90
13.0 Appendix 94

1.0 INTRODUCTION

The Circular Economy

Carbon expenditure is set to double by 2050, and countries are struggling to prevent a rise in global temperatures below 1.5 degrees Celsius, as specified by the Paris Climate Agreement of 2015. This limit was intended to stave off the worst of the potential climate disasters but is now very likely to be crossed. (UNFCC, 2016;UNEP, 2025; Statista RD, 2025)

Ongoing efforts to reduce carbon emissions and general waste production have been on the rise throughout the later half of the 20th century, notably spurred on by The Limits of Growth, published in 1972 by the Club of Rome (Meadows et al., 1972). One such effort predates this monumental report: The Circular Economy (CE). The first known academic description of this concept can be traced back to 1966, though a more elaborated model wasn't developed until 1989 (Leising et al., 2018). Currently, most researchers follow the formulation of the CE as given by the Ellen MacArthur Foundation (EMF):

"*A circular economy is an industrial system that is restorative or regenerative by intention and design.*" (EMF, 2013).

Through the EMF, the concept of the Circular Economy has gained major traction - due to their publications and online platform. Circular Economy practices are not singularly focused on carbon emissions; The CE seeks to remedy losses in biodiversity, reduce the consumption of finite resources and cull the production of waste materials (EMF, 2026). Circular Economy models often refer to the practice of Closing the Loop. This idea is arguably linked to - or based on - the concept known as Cradle-to-Cradle. The term, first coined by Walter Stahel, describes a set of practices where materials are never considered to be waste but should always be reused at the highest possible quality. Current practices, where materials are simply discarded, can be antithetically described as Cradle-to-Grave. Stahel also sketches an economical vision which prioritizes the selling of services, rather than products, as far back as 1976 (EMF, 2023). In short, the CE described a set of practices which emphasize the reduction of consumption and reuse of materials to prevent them becoming waste.

Perhaps one of the most widely known circularity concepts, the R-Strategies exists as a framework by which circularity practices can be characterized. Derived from Lansink's Ladder, Potting et al. (2017) provide a comprehensive formulation of the R-Strategies in their policy report: *Circular Economy: Measuring innovation in the product chain.* The R-Strategies are often structured along the 'R-Ladder', where each strategy is listed in sequence of priority, see **Table 1.1.2**. This order of priorities is an important notion to critically assess CE practices, however, it is by no means a universal rule. Whichever strategy has the best net-positive impact changes on a case-by-case basis and is

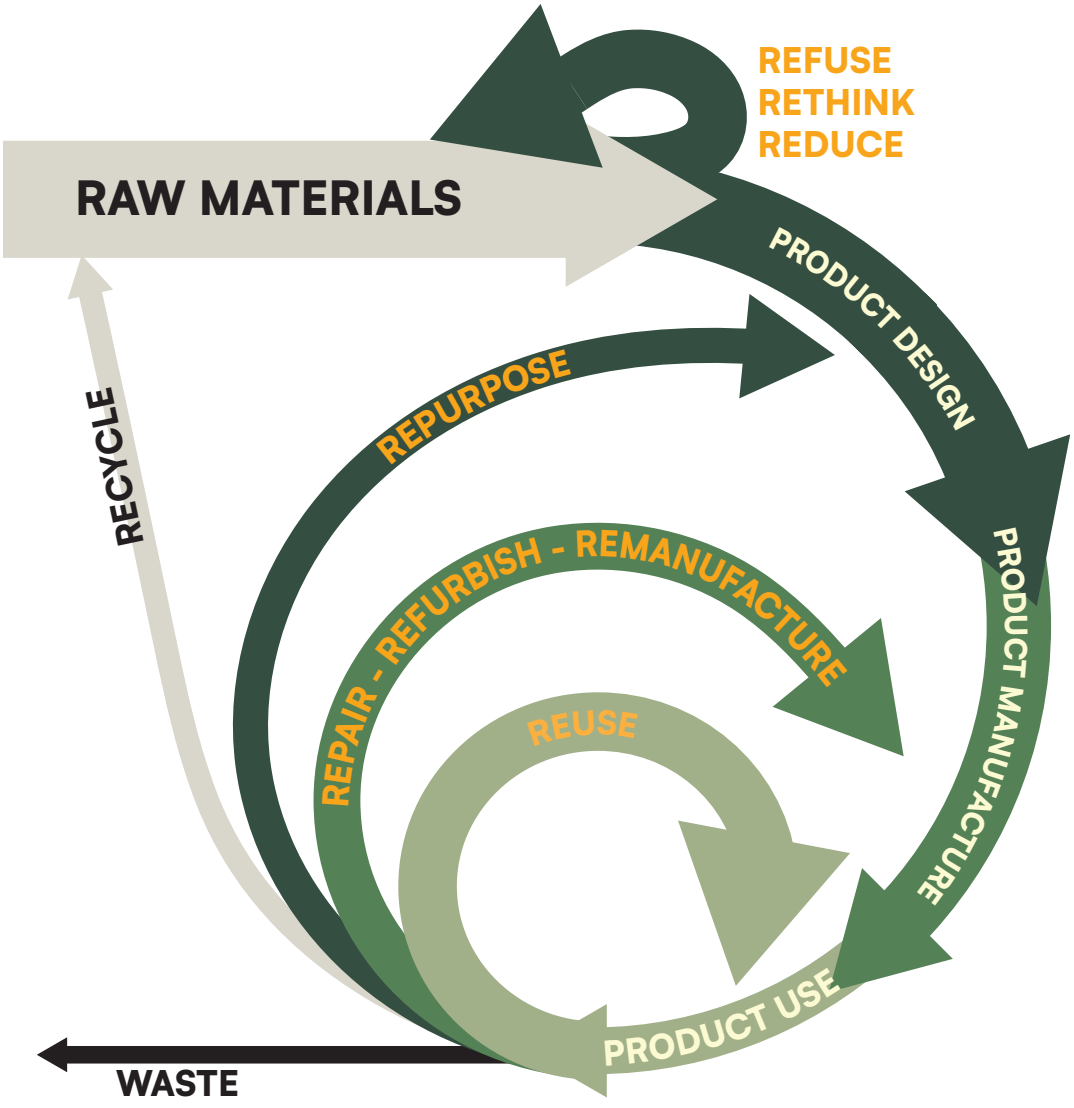


Figure 1.1.1 - *The Circular Economy* - Own work

sensitive to a number of contexts. Kishna & Prins (2024) provide a meaningful elaboration on the R-Strategies with comprehensive definitions for each one, which are also included in **Table 1.1.2**.

The ten practices outlined can be further divided into three overarching strategies: Narrowing, slowing and closing (of the loop). It is important to firstly address the closing strategies. Whilst closing strategies are most implemented across industries (Potting et al., 2017), closing is a misnomer, as these strategies do not complete the loop, they terminate it. In their article *Targets for a Circular Economy*, Morseletto (2020) points out that recycling and recovering (burning) of waste materials permanently alter or destroy the materials and should not be considered true CE targets.

In **Figure 1.1.1** we can recognize the narrowing strategies as those practices which occur prior to the actual loop. These practices are aimed at shrinking the amount of (raw) materials and energy expenditures needed in the first place to produce a thing or service. These strategies generally bring about the strongest impact, as they pertain to a much more fundamental reconsideration of a given production or service practice.

The slowing strategies, which occur inside the loop itself, seek to extend the use of products and parts thereof for as long as possible, ideally indefinitely, by reintroducing them into the loop at different points in time. Perhaps counter intuitive, the smallest loop in the figure, the re-use loop, is the most important one: Using a product as long as possible without having to expend energy to repair or modify it. Characteristically, the other slowing strategies reintroduce the product or part thereof in the production process itself, requiring additional energy expenditure to (re-)process them. An oddity in this are re-purposing strategies, which do not seek to reintroduce the used product inside their own production chain, but rather seeks a new purpose for it, a new product altogether. Morseletto (2020) points out that it is difficult to assign pragmatic targets for re-purposing as a practice, due to a variety of concerns:

Re-purposing detaches the 'repurposer' from the originator of the product. It does not promote a vertically integrated design-production-use pipeline. Due to this, the traceability of the product and components can be lost. Furthermore, the scale of these projects is often artisan, and whilst many parts can be easily re-purposed for a wide range of products, not many products can be re-purposed in their entirety.

However, the most interesting concern which Morsoletto points out, at least in regard to this thesis, is the dependency on the creativity of the 'repurposer'. Whilst creative problem

Overarching Strategy	Circularity Strategy	Definition
Narrow	R0 - Refuse	Making a product redundant by abandoning its function, or replacing that function with a radically different product
	R1 - Rethink	Intensify product use by sharing products or manufacturing multi-functional products
	R2 - Reduce	Manufacture products more efficiently, or make their use more efficient
Slow	R3 - Reuse	Reuse of a product in the same function
	R4 - Repair	Repair and maintenance of a product for use in its original function
	R5 - Refurbish	Upgrade or modernise a product
	R6 - Remanufacture	Using parts of a disposed product in a new product with the same function
	R7 - Repurpose	Using a disposed product or its parts in a new product with a different function
Close	R8 - Recycle	Processing and reuse of materials
	R9 - Recover	Recover energy from materials

Table 1.1.2 - *R-strategies* - Adapted from Kishna & Prins (2024)

solving can, arguably, be deployed in any of the R-Strategies, re-purposing is perhaps most deeply entrenched in it. It concerns itself with the creation of a new product, but not from raw materials, but from other products, which are rarely as easy to give shape to.

Products have complex qualities which are designed for a purpose, they have intention, innately. However, that does not mean that these products can only serve their intended purposes. Re-purposing can become viable when a product has designed qualities which can fulfil other purposes outside of those for which they were intended. This concept is referred to as non-inherent qualities throughout this thesis.

In some industries it can be difficult to imagine what other purpose a product can serve. Whilst parts of a toaster oven, for example, could be readily re-purposed for other products, this practice is more closely related to re-cycling and is far less ideal. However, considering an industry such as the built environment, re-purposing practices can be recognized in a different light, due to the scales of magnitude inherent to it.

"Circular Economy" AND "Buildings" AND ("Reuse" OR "case" OR "technique")	
Benachio et al. (2020)*	166 results
Current date**	3084 results

* Article submitted in 2019 | ** 17/10/26

Table 1.1.3 - *Repeating Scopus Keyword Search of Benachio et al. (2020)* - Benachio et al. (2020) / Own Work

The Built Environment

The built environment accounts for approximately 44% of raw material consumption world-wide as well as 20% of carbon expenditure annually (UNEP, 2024). In both regards it is the single largest sector. Furthermore, construction materials such as steel and concrete, most commonly used in structural systems, account for 18% of the carbon expenditure inside the built environment. Currently, there are no to-scale low embodied carbon alternatives for these materials (UNEP, 2025). This is indicative of a growing need for CE practices which have the potential to reduce carbon expenditure in a scalable manner.

In their systematic literature review, Benachio et al. (2020) identified an exponential increase in publications related to Circular Economy practices in the built environment since 2017. Similarly, Illankoon & Vithanage (2023) identified a growing interest in the topic by the industry at large. Repeating the search strategies outlined by Benachio et al. now yields well over 2000 results, compared to the 262 papers reported at the time, suggesting that this exponential trend has continued in recent years, see **Table 1.1.3**.

Throughout the different literature reviews, material re-use is consistently identified as a major subset of CE research inside the sector. Notably, more recent publications also denote a growing rise in information technologies, computational design and lifecycle assessment (Benachio et al., 2020; Illankoon & Vithanage, 2023; Munaro & Tavares, 2023).

Ghisellini et al. (2018) points out that CE practices in the build environment are much more concentrated around the closing strategies, which is consistent with the overall trends noted by Potting et al. (2017). As previously stated, closing strategies are not ideal for the CE. In reality, they are much more aligned with take-make-dispose or cradle-to-grave practices, also known as the linear economy.

Considering what barriers currently exist towards the implementation of the narrowing and slowing strategies, Benachio et al. (2020) points to a major knowledge gap between research and practice, especially in regards to standardization.

To look more critically at the current state of research and practice, Ghisellini et al. (2016) points out that proper life-cycle analysis for the construction industry needs to consider CE principles across different scales. These can be commonly described as the macro-, meso- and micro-scales.

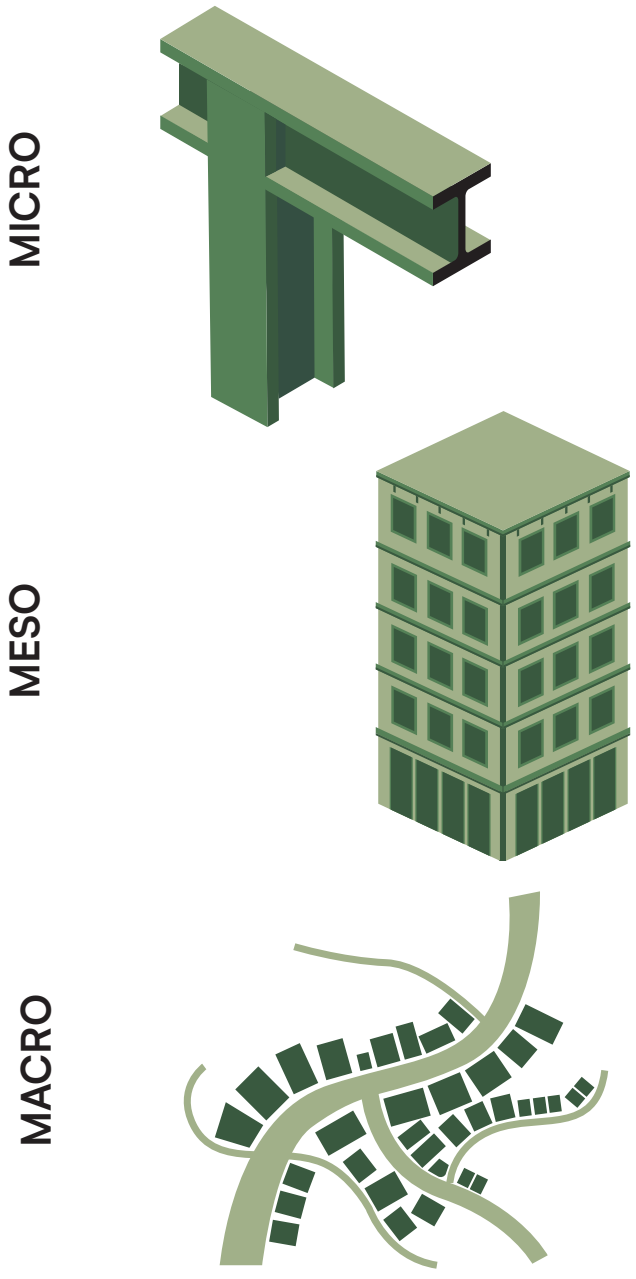


Figure 1.1.4 - *Scales in the BE* - Own work

These are the scales of magnitude referred to in the previous section. They bring forth CE considerations unique to the built environment, highlighting important nuances between the different R-Strategies. Take for example an I beam, recovered from the demolition of an office building. Now, it is being re-used for the construction of a residential unit in an adjacent municipality. If we analyse this practice on the micro-scale, we can classify it as proper re-use of a product. However, when we look towards the meso- and macro-scales, where the product is considered as a building or even an entire urban region, the practice is much more aligned to re-manufacturing or even re-purposing.

The distinction that the different scales provide are significant, because the CE model indicates that re-manufacturing or re-purposing practices, compared to re-use, incur a higher energy cost, which can be plainly recognized: Storage and transportation. Several systematic literature reviews on CE inside the built environment underline the significance of this aspect, stating that the success of slowing strategies are highly dependent on the local availability of resources (AlJaber et al., 2023; Ghisellini et al., 2018; Ginga et al., 2020; Illankoon & Vithanage, 2023).

Furthermore, multiple literature reviews indicate a growing interest in designing for modularity and demountability (AlJaber et al., 2023; Benachio et al., 2020; EMF, 2013; Illankoon & Vithanage, 2023; Kalmykova et al., 2018). These kinds of research have the potential to address the transportation concerns. If demountable and modular products are readily utilized within a given region, it would greatly increase the local access to materials which can be reused for new constructions at a relatively low energetic cost. However, these kinds of research are rethinking practices which seek to prevent products or materials from ending up in a landfill by redesigning them from the root. It is very important kind of research to pursue, however, what do we do with the products and materials which are already there?

Willskytt et al. (2016) defines repurposing as an open-loop practice which should only occur if the proposed material has suffered degradation beyond the point where the preferable strategies would be viable. Given the current reality, this thesis would disagree with the sentiment. Whilst the world at large should aim to develop closed systems by redesigning the way we make new products and materials, in an effort to enable slowing strategies for as long as possible, re-purposing should be an important complimentary effort to address the current reality, to start closing the linear economy. If a product is being over-produced, if the market for it has changed, if materials are simply there and no-one is taking them, repurposing can be used to create a new CE loop, reintroducing said material into the economy, potentially indefinitely. Whilst it is preferable to reintroduce a given material inside the relevant sector, if it viable, it could also be introduced into another. As such, this thesis aims to look at re-purposing and the difficulties innate to it, with the pre-conceived notion that it can be a very viable strategy, provided the product has good non-inherent properties. Through Morseletto (2020) the following considerations remain critical: Repurposing tends to be low-scale and is dependent on the creativity of the 'repurposer'.

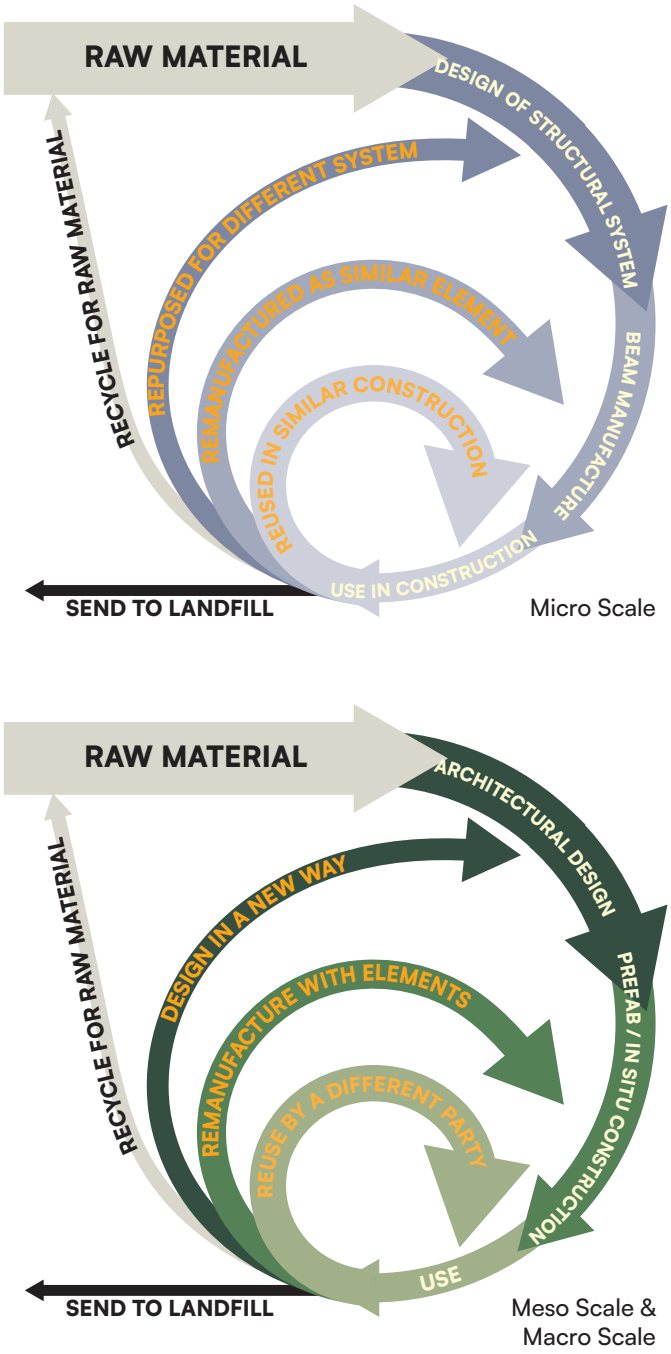


Figure 1.1.5 - *R-Strategies Across Scales* - Own work

Looking at the slowing strategies at large, many barriers remain the same. There is a lack of standardization of practice, a gap between theory and application, and generally speaking, these kinds of materials and products are characterized by irregularity due to their previous application: They have suffered degradation, have been cut to specific sizes and often have been designed without demountability in mind, leading to further cutting of the stock.

Even when the goal is to reuse or remanufacture, to design with these kinds of products and materials is an innately complicated and creatively demanding task. This is why, in part, there is a noticeable increase of research into the development of informational technologies. Leveraging the advances in computational design can help to promote these kinds of practices and expand them to a wider market.

The Generative Potential

In their paper on algorithmic circular design, Huang et al. (2021) recognizes the challenge inherent to designing with reuse in mind. The designer has to devote significant time, creativity and flexibility to work with a geometrically and structurally limited stock of parts. They characterize two predominant methods to address this through computational design: Top-down, where an intended design is imposed on a constrained stock of materials, and bottom-up, where the constrained stock informs the formulation of the design. In other words: Manifesting designs in spite of the stock or manifesting designs based on the stock.

Top-down design can be characterized by a 'target design model'. A preconceived design concept which guides the creation process and the computational algorithms formulated for them. Meanwhile, bottom-up design can be characterized by rules of aggregation, ways in which elements are procedurally combined to form the eventual design. These rules find their basis in shape grammars and making grammars as defined by Stiny & Gips (1971) and Knight & Stiny (2015) respectively (Huang et al., 2021).

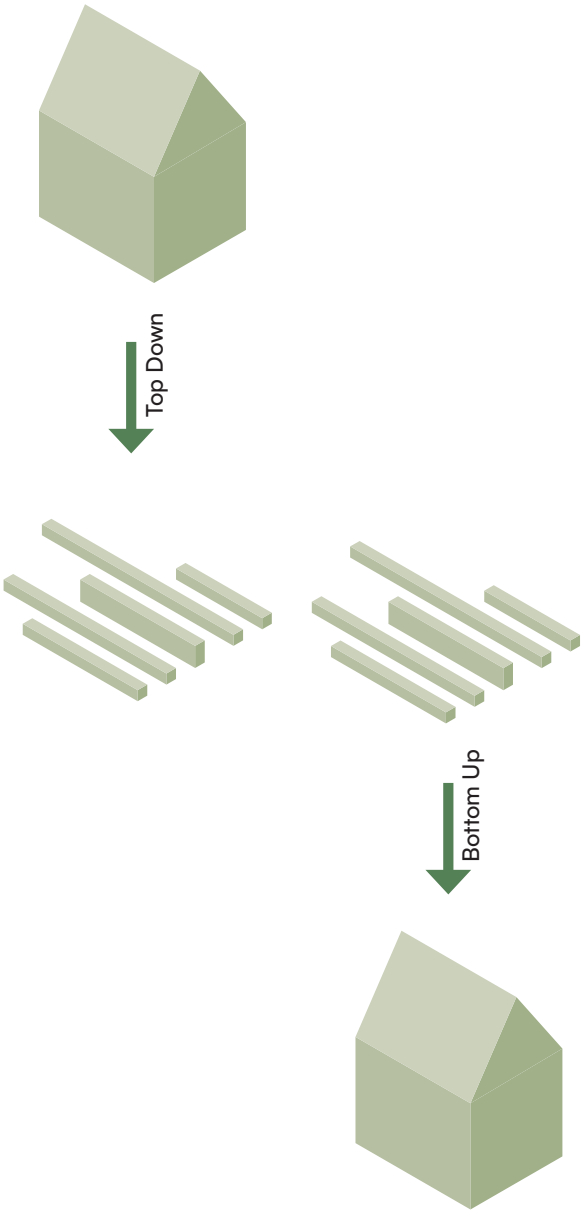


Figure 1.1.6 - *Top-down & Bottom-up*
Own work

Whilst the division between top-down and bottom-up design of Huang et al. (2021) relates specifically to circular design, the same distinction can also be seen in computational design (CD) at large. Caetano et al. (2020) put this best in their formulation of three typologies of CD: Parametric (PD), generative (GD) and algorithmic (AD) design. Parametric design is defined by parameters and rules which constrain them, whilst generative and algorithmic design are defined by a set of instructions which are repeated. The latter two typologies are much more iterative, often have fewer parameters and characteristically tend to show 'unexpected behaviour'. The three typologies are not entirely separate from one another, and whilst algorithmic design is generally seen as a subset of generative design, it can also be considered to sit between parametric and generative, to a degree, see **Figure 1.1.7**.

Going even further and contextualizing these general CD practices into design practice at large, the fundamental difference between parametric, generative and algorithmic design becomes even more pronounced. For this, we can look towards the five generic elements of the design process which are outlined by Van Dooren et al. (2014) - most importantly the guiding theme. This aspect of the design process can be described as a set of ideas which guide the design process across the scale of time, from the abstracted onset to the final explicit design, providing a means of convergence.

One important distinction between computational design and design at large is the parametric quality inherent to it. At the end of the design process, there is not one explicit design, but rather a theoretical range of possible designs. A unique and powerful quality, as the computational design can potentially be applied to a different set of circumstances and still produce an acceptable end result. This theoretical range can be described as the 'design space' or 'solution space'.

Looking then at the different CD practices outlined by Caetano et al. (2020), with the guiding theme in mind, the differences between the typologies becomes very pronounced.

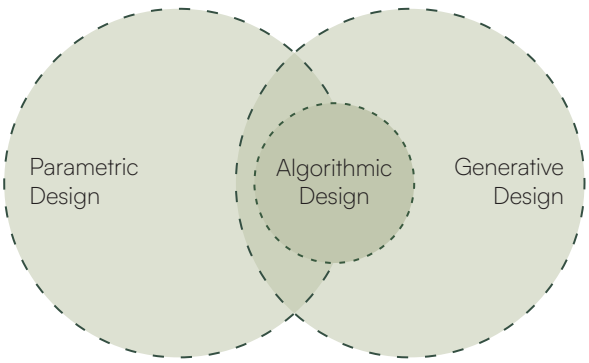


Figure 1.1.7 - *Three CD Typologies*
Adapted from Caetano et al. (2020)

Parametric Design

Schematizing PD processes, see **Figure 1.1.8**, it can be argued that the guiding theme exists as the 'target design model' outlined by Huang et al. (2021). Gradually, the parameters and rules which constrain them produce an increasingly complicated but narrowed set of potential solutions, informed by the design intentions. Theoretically, the solution space is infinitely large still, but the range of diversity is relatively limited, and the difference between solutions can be very small. As such, Mirra & Pugnale (2021), offer a valuable critique of expansive parametric design spaces, noting that these multi-dimensional search spaces are hard to search, forcing the designer to significantly narrow down the design space towards a much more specific target design model. In PD processes, the parameters symbolically represent aspects of the design (Caetano et al., 2020) and as such, the process can be characterized as highly linear and hierarchical.

Generative Design

Schematizing GD produces a very different picture, see **Figure 1.1.8**. The guiding theme does not exist as an intended target design, because GD is not fundamentally informed by it. Rather, GD exists as a set of rules which are iterated. The intention behind these rules originates from a design language, often formalized through shape-grammars. As such, GD does not produce a range of acceptable designs, but a range of potential designs. The process arguably widens over time, new grammars do not constrain the potential end result but diversify it. This is contradictory to the expectations of the guiding theme, but does highlight a critical aspect of generative design in two ways: For one, it illustrates that the design space is much wider in scope, the differences between different solutions much greater. However, it also introduces a kind of unexpected behaviour. It is often difficult to steer generative designs to an intended target design, because it is not fundamentally defined by it. This is what Caetano et al. (2020) refer to as '*unexpected behaviour*'. The second way in which generative design subverts our expectations of the guiding theme can be explained through what Caetano et al. (2020) classifies as '*a lack of traceability*', which is perhaps best explained through an example:

Take a parametric design of a house. There are parameters for the floor-to-ceiling height, the thickness of the walls, and the angle of the roof. Changing one of these parameters directly changes one or a set of aspects of the design. Indeed, their labels are often elaborate, it is clear what each one does, because the rules constrain them to change a very specific aspect. Now, look at a generative design for a brick wall, where each brick is placed based on independent instructions for each separate

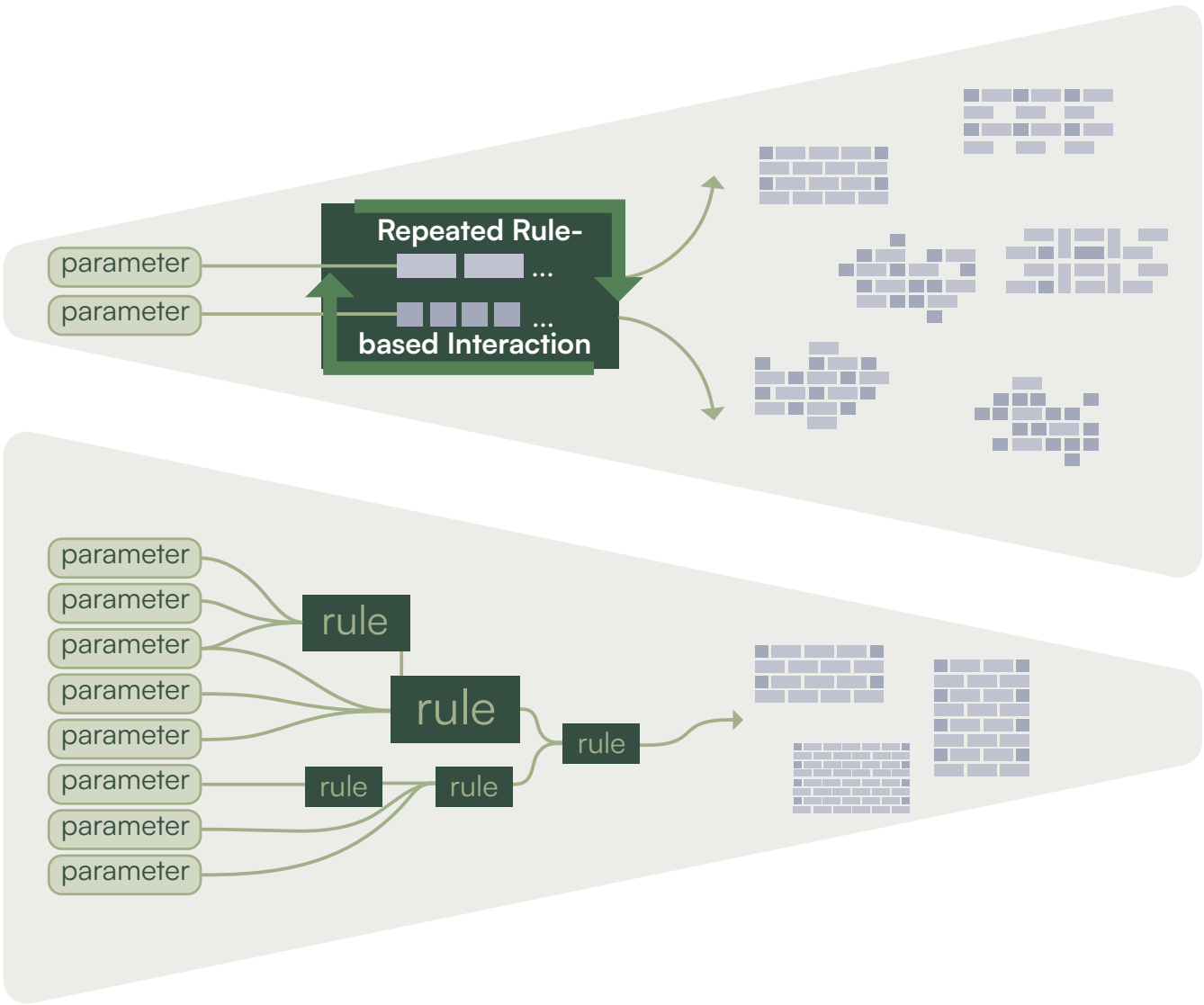


Figure 1.1.8 - *Schematizing Parametric and Generative Design* - Own work

brick, depending on the current state of the wall. Change one instruction, on brick is placed at a slightly different angle, and the entire wall would change fundamentally. Each consecutive action that occurs after this one now has a profoundly different effect, and it is difficult if not impossible to define exactly what part of the design each parameter has influence over. This is the difficulty with GD. However, contextualizing it inside re-use based design, it is also what makes it so powerful. Each brick has an immense sense of agency, it 'gets a say' over how the entire wall is constructed. The shape and qualities of a material determine the design, rather than the design determining the material.

Algorithmic Design

Algorithmic design is generally considered as a subset of generative design, though to a degree it can also be considered as a hybrid of parametric and generative design. The distinguishing quality of algorithmic design is that there is a correlation between the input parameters and the output model, providing the 'traceability' which generative design lacks. In essence, the parameters of an AD process inform the designer as to what they can do. In GD processes, it is not a given that the program used to generate solutions reflects the design model itself.

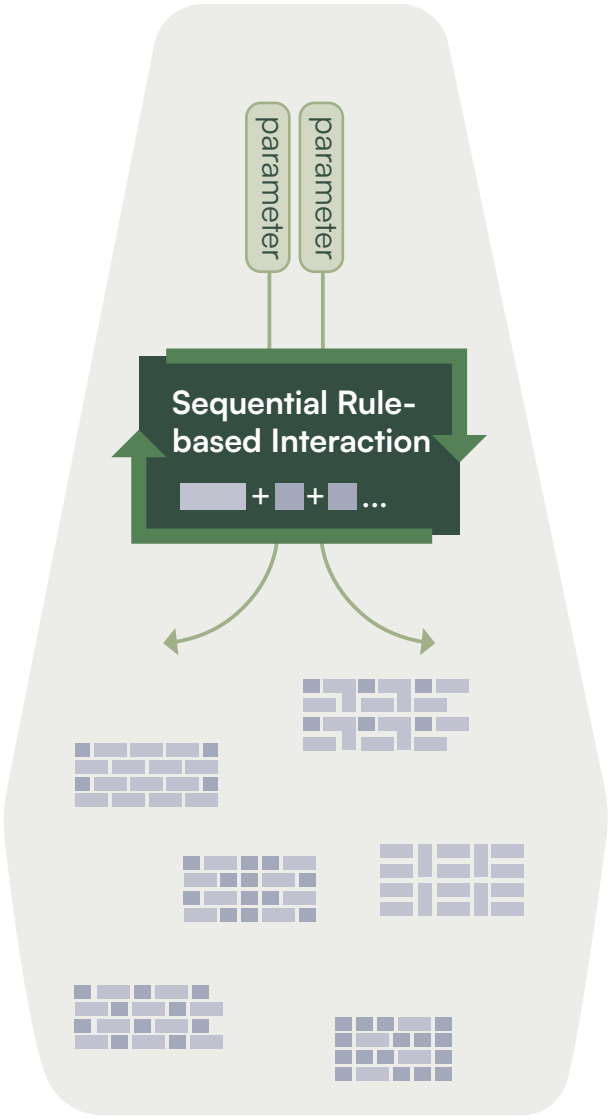


Figure 1.1.9 - *Schematizing Algorithmic Design*
Own work

Taking another look then at the example of a brick wall, a simple change through which the generative variant becomes an algorithmic one would be to place each brick in sequence. Instead of each brick interacting with the wall at the same time step, each brick interacts over a linear progression. As such, it becomes possible to correlate the influence which one brick has over the wall at large. Notably, algorithmic processes are generally expected to be more predictable, better to guide by the designer to a desired solution, though the solution space is arguably narrower than it is with generative design in general.

The critical point in the assessment of these different typologies is not only to characterize them, but to highlight a key difference between PD practices and GD/AD practices specific to reuse. These practices are commonly known as stock-constrained design models. Parametric design demands a negotiation between the target design model and the stock of materials. A common solution is modify the reuse stock, either via cutting the materials to a more agreeable shape, or by processing them into modular assemblies. Generative and algorithmic design don't require this outright, as the materials determine the design, not vice-versa.

Computational design of any subset produces considerable search spaces. To navigate these spaces, a common approach is to apply optimization algorithms. Contemporary methods often make use of mixed-linear integer programming, greedy search algorithms and perhaps most prevalent are genetic algorithms. However, these applications are typically better suited to linear design processes, such as parametric models. The issue at large, with genetic algorithms especially, is attributable to the parametric interdependency inherent to GD and AD models (Davidor, 1991). An increasingly more prevalent alternative to these optimization method comes in the form of Reinforcement Learning (RL).

Reinforcement Learning

Machine learning (ML) is a major subset of artificial intelligence at large. Inside the discipline, three main typologies can be recognized: Supervised learning, unsupervised learning and reinforcement learning.

Supervised and unsupervised learning concern themselves with data prediction. They take an input dataset and try to develop an understanding of it. The categorical difference between them is that supervised learning provides the intended outcome a priori. The model is told what prediction should match to what input, and the task is to emulate this. Unsupervised learning isn't told anything in terms of good or bad but instead attempts to find the underlying structure of the data, based on some kinds of similarity. Both function with the innate understanding that there is some hypothesis which can describe the data, and their goal is to find it.

The potential, the main reason and power of these models, is in their capacity to generalize. Provided with sufficient training, the models can work on entirely new datasets, provided a degree of similarity with the training data.

Reinforcement learning does not require a dataset a priori. The task is not to predict labels, or to cluster data, it is not tasked with finding an underlying hypothesis, but rather, a policy. A kind of behaviour, a system of actions, which yields positive results inside an environment. Early examples of RL are therefore often game-like. Models are trained to perform well at old Atari games, for example.

What makes reinforcement learning so interesting is the ability to comprehend actions over time. In GA it can be asserted that non-linear action spaces and highly interdependent parameters are detrimental to the success of the optimization. RL, on the other hand, thrives in these kinds of conditions.

Just like other ML typologies, the ultimate potential of RL is to generalize. The policy, given enough training, is capable of working on different contexts. The model can be extremely quick acting, working in a fraction of the time that GA optimization takes and provide results of comparable or improved quality. This unlocks another potential, not innate to the more traditional optimization algorithms: The ability to create an interactive design tool for early design exploration.

Research into reinforcement learning for stock-constrained design is still in the stages of infancy, with few publications having been found throughout the research process that focus on this. Furthermore, no publications have been found which seek to apply reinforcement learning to a constrained stock of repurposed elements.

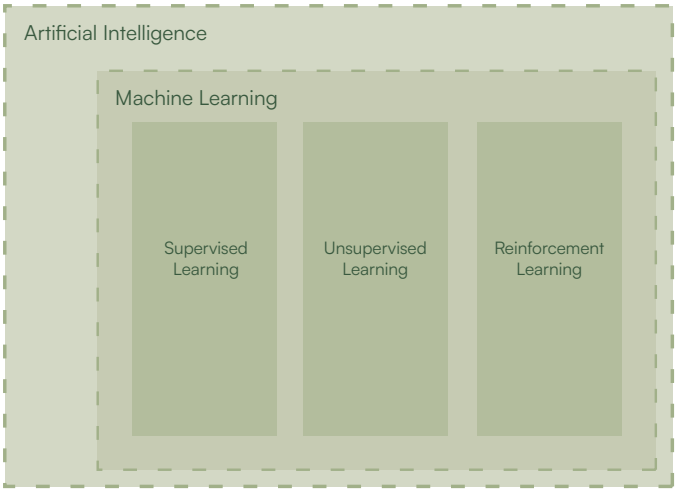


Figure 1.1.10 - *Machine Learning and its Subsets*
Own work

Case Studies - Material and Design

The value of reuse practices has been well established, and the potential for repurposing practices has been identified. Computationally, the approach to these practices is often formalized as stock-constrained design problem. However, this raises an as of yet unaddressed question: What stock?

To this end, bicycle frames have been selected. The idea for this comes from the paper of Nian et al. (2022) who give a functional demonstration of a modular system which utilizes old bicycle frames to make construction elements. This material becomes significantly more interesting considering the location in which the case study will be implemented: The Netherlands.

In the Netherlands, bicycle frames are an abundant resource. Cycling account for roughly 27 percent of all transportation of individuals in the Netherlands (CBS, 2023). About 804.000 new bicycles have been purchased in 2023, with the total estimate of bicycles in the Netherlands being at 24.1 million (Fietsberaad Crow, 2024b). This means that there are more bicycles than residents in the Netherlands itself. The amount of bicycles being discarded each year is more difficult to put to terms, which points at a knowledge gap common in material waste streams, however, each year a significant number of bicycles are removed by municipalities, over 200.000 as of 2018, whilst other sources but the estimated number of bicycles collected by their respective owners between 10 and 50 percent (BNR, 2018; Fietsberaad Crow, 2020.) This points to a massive and steadily producing waste stream of material. Furthermore, bicycle have interesting structural properties: They are usually made from materials often used in structural design, such as stainless steel and aluminium, and utilize triangular structures to spread the weight of the user. However, bicycles come in a very large variety of shapes and sizes, making it an exceptionally unpredictable stock which would be difficult to design with.

Bicycles, and the frames themselves especially, make for an interesting case of re-purposing practice, as the local availability might make them less carbon intensive to adapt, as transportation is a common barrier for CE adoption. They exhibit structural qualities which might be leveraged to serve a new but valid purpose, and the unpredictable nature of the stock makes for a compelling case to use generative design principles and reinforcement learning.

That leaves the question of what the material is supposed to achieve, structurally. Again, looking at the local vernacular, an interesting opportunity presents itself: Cycling bridges. A common hallmark of the Dutch infrastructural system, these bridges make for a compelling case study due to the stringent structural criteria they posses, with relatively large live and dead loads imposed on them. Naturally, there is also a more intuitive appeal to creating infrastructure from the objects which make use of it, providing a powerful design model to encapsulate a value proposition, to be defined after the problem statement.

Problem Statement

Having outlined the need for Circular Economy practices in the built environment, contextualized the different practices across several scales, analysed the potential of both computational design at large as well as reinforcement learning optimization, and lastly conceiving of potential material source and design model, the problem statement is given as follows:

Stock-constrained design problems are often approached via parametric models, whilst generative models offer a more direct consideration of the stock elements, but remain under-investigated and prove difficult to optimize by more traditional means.

This problem statement is consequently based on the following sub-statements:

1. *Carbon expenditure is set to double by 2050, making the 1.5 degree rise in global temperatures likely to be exceeded.*

2. *The built environment accounts for a significant subset of carbon expenditures worldwide.*

3. *Structural design accounts for a significant percentage of this subset.*

4. *Circular Economy practices offer potential solutions but are currently not widely adopted in the built environment, and if so, are often concentrated around the closing strategies.*

5. *Through analysis of CE practices over the different scales, a common barrier is identified in research: Transportation of reuse stock.*

6. *Repurposing is identified as a potential remedy to transportation as a barrier, enabling access to a wider spectrum of local materials, but is currently under investigated as a practice.*

7. *Computational design is an often leveraged and powerful tool to enable reuse practices, but parametric approaches often warrant the modification of stock.*

8. *Generative and algorithmic principles are identified as methodologies which do not require stock modification outright, but are generally more difficult to steer and optimize via more traditional algorithms.*

9. *Reinforcement learning is identified as a powerful alternative to traditional optimizations, but the research into these practices is sparse.*

Value Proposition: Material-First Computational Design

With the problem statement in mind, this final sections serves to briefly outline a common term which has been used throughout this thesis: Material-first computational design. This concept is defined as:

Design practices which employ computational means to create systems or models which generate form through a chosen stock of materials, rather than a superimposed target design model, and which seeks not to modify these materials, but ideally leverage the qualities which makes the stock unique and varied.

The reason to formulate this definition is two fold: For one, it encapsulates the core concept of this thesis, and secondly, there was no term identified throughout literature which encompassed all the aspects which are deemed critical to this concept. It arguably is founded in the bottom-up approach sketched out by Huang et al. (2021), but is elaborated by including the desire to leverage unique stock qualities as a possible advantage, rather than a barrier. It is also more attuned to repurposing practices, under the notion that stocks of materials can possess qualities which makes them fit for more than their designed function.

2.0 RESEARCH QUESTIONS

This section serves to outline the research questions for this thesis, and to provide an overview of the approach taken to answer them.

Main Research Question

How can a constrained stock of repurposed elements be utilized for structural designs through a generative algorithm, implemented as an interactive design tool which enables material-first design solutions?

Sub-questions

1. How can material-first generative principles be formalized from the basis of a constrained stock of repurposed elements?

2. How can a dataset be constructed to represent a constrained stock of repurposed elements for the purposes of material-first structural and generative design?

3. How can the material-first generative principles be applied to a given design objective and which criteria need to be satisfied?

4. How is the material-first generative algorithm constructed from the generative principles, dataset and design objective?

5. Can a genetic optimizer successfully utilize the material-first generative algorithm, therefore validating the functionality of the underlying principles given the provided material and design context?

6. Can a reinforcement learning agent successfully utilize the material-first generative algorithm to produce optimized design solutions given the provided material and design context?

To answer these research questions, the following overarching methodology has been outlined, visualized in **Figure 2.1.1**:

Chapter 3 deals with the formulation of the material-first principles. It does so through a literature framework to further contextualize material-first computational design in design at large. A method is described which consists of the practice of 'Design Games' - a concept borrowed from Mirra (2022). Through these design games, the final generative principles are created.

Chapter 4 deals with the building of a dataset by which to represent a constrained stock of repurposed elements, with the aim being to construct a dataset with a meaningful degree of accuracy to an actual real-world stock and to process said dataset for the creation of a design algorithm, yielding a geometrical and material representation of the stock.

Chapter 5 deals with the design case itself. The principles of chapter 3 are purposefully developed without a preconceived target design model. This chapter produces a target design model, without consideration of the generative principles. The goal is to conceive of structural requirements and design parameters.

Chapter 6 deals with the development of the actual design algorithm. This is where the generative principles and the target design model meet. It is a method section only, explaining how the algorithm works, and why it has been constructed in the way that it has been.

Chapter 7 consists of the genetic validation of the algorithm. This step is borrowed from the work of Mirra & Mueller (2025). The goal is to verify if the algorithm can converge in the first place, prior to applying reinforcement learning. Additionally, this chapter provides a means to specifiy the design parameters through multi-objective optimization of the structural performance, demonstrating which design configurations are feasible.

Chapter 8 lastly consists of the application of the reinforcement learning model. The trained agents are tested across three categories: Structural performance, capacity for generalized application and capacity for interaction with a user.

After this, general conclusions are drawn and the results are discussed.

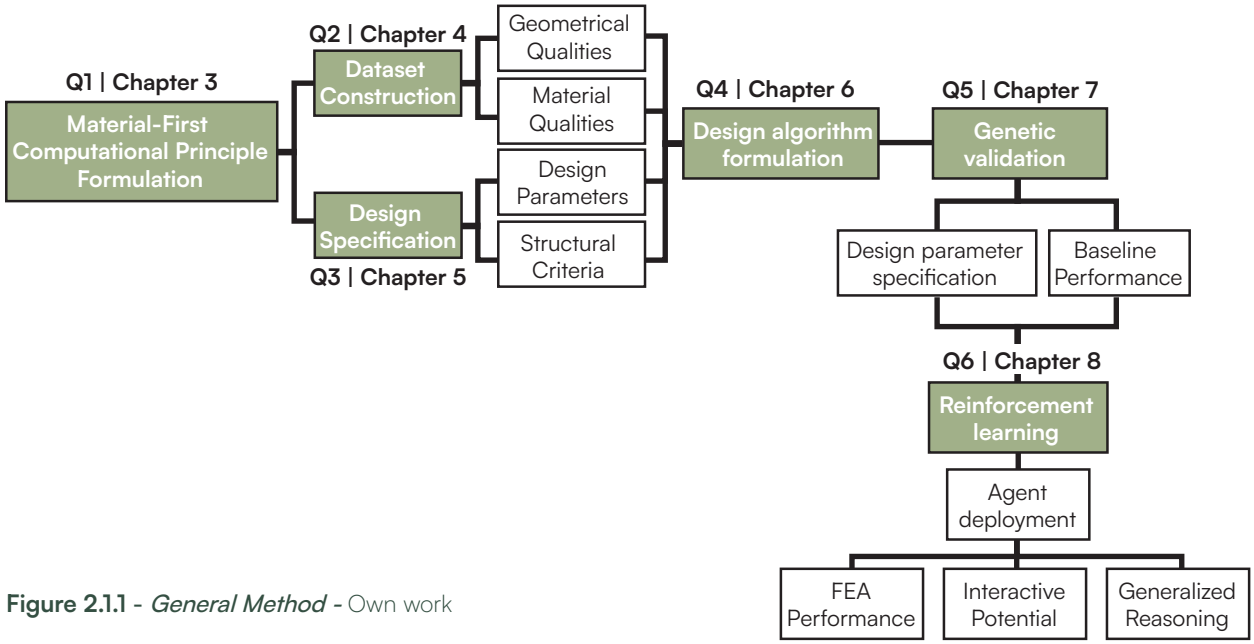


Figure 2.1.1 - *General Method* - Own work

3.0 MANIFESTO OF MATERIAL-FIRST COMPUTATIONAL DESIGN

Whilst it is a relatively minor element in the scope of the research itself, the objective of 'what' to design and how to arrive at said objective is an important consideration and an integral part of this thesis. As such, this chapter serves to outline the design philosophy that is woven throughout the development of this project. Through literature review a coherent theoretical framework is proposed for the practice of 'material-first design'. The methodology will then outline a practice known as 'design games', through which some initial experimentations have been performed, outlined in a brief results section. Due to the time constraints of the thesis, these experiments are currently an open-ended tangent. Still, their discussion remains important, as considerable time has been invested in these experiments. As such, it has been decided to include them here, in a more informal manifesto, rather than into the main body of the thesis.

Chapter Objective

The objective of this chapter is firstly to define the fundamental design concept woven throughout this thesis: Material-first computational design. This concept is outlined through the theoretical framework. Beyond this, the theoretical framework also introduces several aspects through which the actual material-first computational design concept can be translated to an actual design practice. The method section will outline a kind of experimentation process, design games, which can be employed to conceive of different kinds of design practices. These practices are made explicit in terms of shape grammars. A variety of these shape grammars are then discussed in the results section, after which the final shape grammar to be developed in this thesis is proposed.

Relevant Research Questions

How can material-first generative principles be formalized from the basis of a constrained stock of repurposed elements?

3.1 Theoretical Framework

The theoretical framework of this chapter serves a few key objectives. It firstly outlines design processes in general and consequently contextualizes them in both material-first design practice and computational design practice. Following that, previous literature is addressed to underline the current practices and trends, common pitfalls and precedents of these kinds of design practice. This will also help to underline the relevance of the topic in general and fortifies the need for these kinds of research and design projects.

Design processes in general

To formalize a coherent theoretical framework for material-first design through computational means, it is necessary to first provide a formalism which accounts for design practice at large. Van Dooren et al. (2014) provides such a formalism — their work targets the design process from an educational perspective, as such, their focus lies on making the process itself explicit and communicable. This framework has been selected for this chapter due to this explicit quality. In addition, the framework is inherently based on design in the built environment, architecture specifically, providing meaningful elaboration specific to this field, which further underlines the relevance as it pertains to this thesis. The framework of van Dooren et al. (2014) consists of five key elements: Experimentation, guiding theme, domains, frame of reference and laboratory of experimentation. **Figure 3.1.1**, adapted directly from their publication — schematizes the five elements into a top-level design workflow that moves from the onset to the finalized design from left to right.

Experimentation is described as a process of exploring and deciding. Van Dooren characterizes experienced designers as

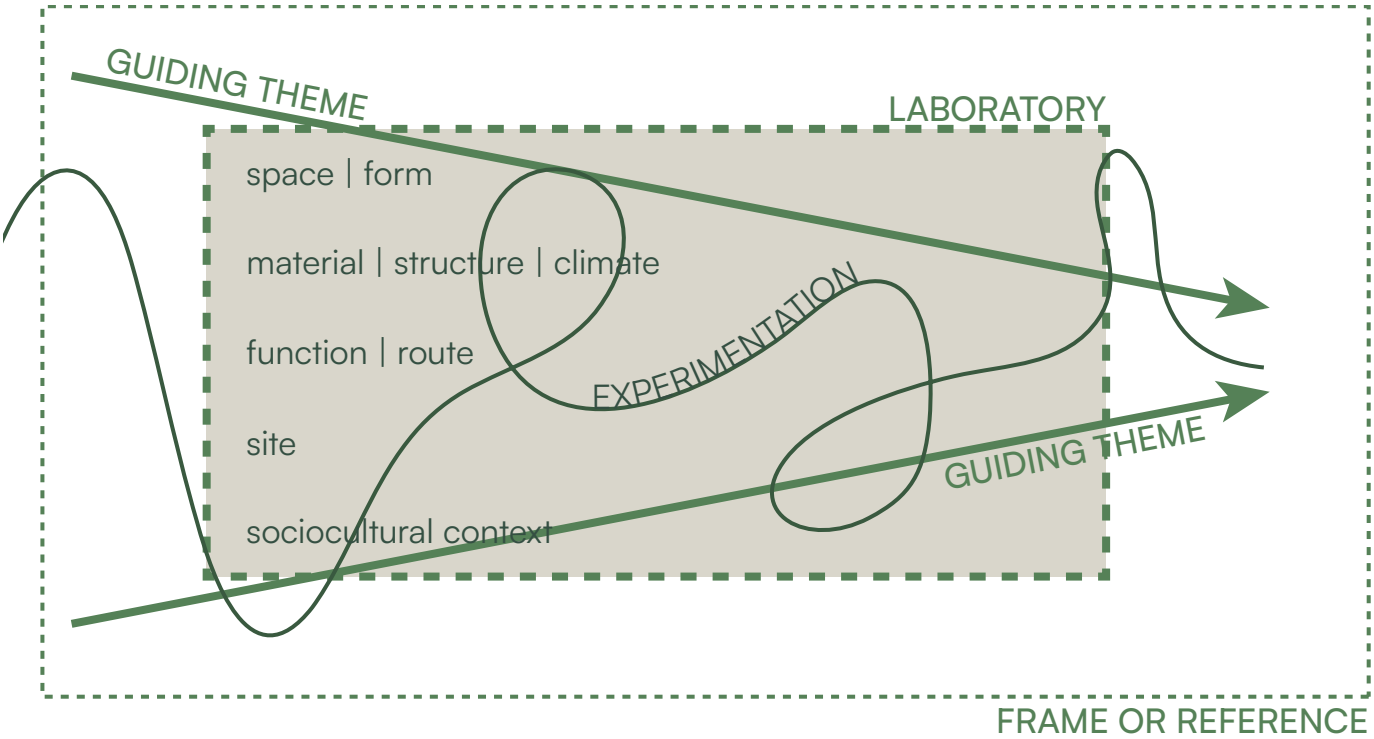


Figure 3.1.1 - *The Five Generic Elements in Scheme* -
Adapted from van Dooren et al. (2014)

being able to seemingly diverge and converge simultaneously, weighing alternatives and choosing from them. Experimentation is a quantitative process, requiring variants to be tested — but it is also a quantitative process, requiring metrics to test against.

How do these metrics come to be, how does the designer decide where to start their experimentation, and how do they steer the process beyond? Perhaps the most immediate factor is the guiding theme. Van Dooren et al. (2014) describes this concept as '*an inspiring direction or order*'. Critically, the guiding theme is not something which is known on the outset of the design process, it is something which develops throughout the process itself. It is the way through which the designer sees or frames the design problem. It is the act of exploring design qualities at the onset and to elaborate these qualities throughout the process, turning them into aspects of the designed object.

The guiding theme develops through experimentation, which occurs inside a field of work, which is the third element of van Dooren's framework: Domains. Van Dooren et al. (2014) put this in terms of thinking and communicating about the concrete world, using words. These words, these ideas, require categorization. Domains provide said categorization. The authors assert that design occurs across domains, seeking to make a statement which encompasses them. They also provide specific domains for the built environment, also indicated in **Figure 3.1.1**.

As mentioned, experimentation raises an important question of which metrics to test against, and where to start and steer the experimentation process itself. The guiding theme is a major consideration, but the frame of reference is perhaps equally significant. This is the fourth element of the framework, the authors describe the frame of reference, or library, as a set of precedents or references. These are made explicit in diagrams and sketches, but more abstractly it can also be recognized as a '*repertoire of patterns*'. This repertoire develops throughout the career and life of a designer, it is something which they bring into a process a-priori, but which also expands and changes with every design process they undertake.

The last element is the laboratory. It is arguably the most significant element to consider throughout this manifesto. In the same way that a scientist requires a suite of tools, a laboratory, to conduct their experiments, so does a designer need a laboratory to conduct theirs. The laboratory of design is considered as the visual language of images. Van Dooren et al. (2014) considers the laboratory to more explicitly consist of sketching and modelling. However, as will become apparent, perhaps this definition is too limiting in the context of computational design.

Computational design processes

As comprehensive as the framework of van Dooren is, it does not consider how their five elements are represented in a computational design process. The critical next step in building the framework for material-first computational design is to therefore make this connection between the design process in general and the computational design process specifically. As mentioned, the laboratory is perhaps the most fundamental aspect to consider in this regard. Computational design does not sketch in the traditional sense, neither on paper nor on the computer screen. Van Dooren states that the laboratory of design, as a visual language, runs parallel to the process of using words, arguing '*Both languages capture meaning, and are to an extent, vague.*' - van Dooren et al. (2014). This is where computationally design diverges fundamentally from 'traditional' design.

The laboratory of computational design is coded. The language is based on logic, and whilst it can capture meaning, it is not vague. Before moving further, it should be asserted that computational design is not fundamentally divorced from 'vagueness' nor from the traditional process outright. A qualified computational designer still needs to frame the design situation, still need an inspiring order and still requires a degree of subjectivity to derive at said order. Regardless, the difference in the laboratory carriers a significant ripple effect that permeates the entirety of the generic elements of van Dooren et al. (2014), and as such, the framing of the material-first computational design concept.

The research of Lee & Ostwald (2020) provides key insights into computational design processes. In their article, the researchers consider decision-making in design as a cognitive process, in which design alternatives are generated and evaluated — a definition closely aligned with the experimentation process that van Dooren et al (2014) defines. Throughout their in-depth protocol analysis, the authors identify three decision-making processes in parametric design specifically, classified as 'conclusive', 'confirmative' and 'simulative' decision-making processes. In the conclusive process, the question is framed as 'Do the solutions meet the design goals?'. If not, the designer uses the algorithm to generate new variants, and tests them again. This process is characterized as forward-incremental. Critically, van Dooren et al. (2014) outlines a misconception in design processes among students, wherein they perceive the design process as purely forward-incremental. They often stop considering alternatives once they have an idea, and do not perceive the design process as iterative. This is perhaps similar to the conclusive process in parametric design. It is no longer a matter of re-evaluating the more fundamental concepts but of finding different configurations that fit within the concepts already developed. Ostwald & Lee similarly characterize the conclusive DMP as one of lower creative thinking. Perhaps more interesting then, are the confirmative and simulative processes. The fundamental difference between these and the conclusive process, is that the algorithm itself is under scrutiny. The confirmative process assesses the development of the algorithm itself, whilst the simulative process questions the design solutions in it's entirety. Both processes are linked to higher creative thinking by Ostwald & Lee (2020).

There is a link to be made between the algorithm of the computational design process and the guiding theme of the design process outright. Both of these concepts can be characterized as the vehicle of coherency. The guiding theme exists as an inspiring order and the algorithm is a very literal order. Their likeness is further underscored by the fact that both van Dooren et al. (2014) as well as Ostwald & Lee (2020) consider that the reinvestigation of the guiding theme and algorithm respectively is a positive or creative quality in the design process.

A notion exists amongst designers which argues that computational design restricts creative thinking, but the comparison above can argue that this is not the case. The fundamental flaw of this notion stems perhaps from viewing a computational design process only at the penultimate stage, where the algorithm is finalized and only parameters are still adjusted. If proper analysis of a general design process merits consideration of the guiding theme and how it is developed, than proper analysis of computational design processes merits consideration of the algorithm and how it is developed. However, before doing so, the different subsets of computational design should be analysed and material-first design should be contextualized.

Computational Design Typologies

Throughout relevant literature, terms such as 'parametric design' or 'algorithmic design' tend to be used interchangeably. This creates some confusion as to what the terms entail specifically. The lack of generalized computational definitions led Caetano et al. (2020) to put forward a formalization of these terminologies. Their framework highlights key differences between the types of computational design processes. The authors put forward three different types of computational design (CD): Parametric design (PD), generative design (GD), and its subset, algorithmic design (AD).

Caetano et al. (2020) defines parametric design (PD) as an approach which describes a design symbolically through parameters. Often, the correlation between the parameters and the resulting design configuration is highly visible, but this isn't necessarily the case. In a sense, it could be argued that the design exists as the 'sum' of its parameters, though the consideration of parameters through the algorithm can be highly complex.

Generative design (GD) is characterized as utilizing algorithms which act more independently or autonomously than PD. As such, there is a more well-defined starting point and exit condition. The algorithm iterates on a set of rules or instructions until a certain set of conditions are satisfied. GD processes can generate highly complex design outputs from fewer starting parameters, due to the ability to iterate many times. It is much less linear, compared to PD, but the trade-off comes in what the authors describe as a 'degree of non-traceability'. It is more difficult to correlate the input parameters to the resulting design.

Algorithmic design (AD) is a subset of GD which specifically remedies the lack of traceability between the input parameters and the output designs. This can be achieved in a variety of ways, but generally it can be expected that there is a trade-off between GD and AD implementations. GD can behave unexpectedly, whether this quality is desirable is subjective. However, an AD process will generally be more predictable and steerable, at a loss of unexpected behaviour.

The merit of one typology is not objectively better than another, this is dependent on the goal of the designer. If they seek to produce a full-scale and elaborated design, such as in multidisciplinary BIM projects, then PD might be preferred, offering a very direct control between different inputs, different disciplines, different domains. However, the designer might work on a project which is indifferent to a specific design scenario, building a tool for design, rather than a complete design outright. This is where the capabilities of GD become more desirable - simple inputs, complex output. But even then, the designer might want a more granular level of control over their tool. In this case, AD might be preferred, prioritizing a more dialectical process between designer and tool over the autonomy of said tool.

Material-first computational Design

The term material-first is somewhat arbitrarily chosen, as there's many different names used throughout literature to describe design processes which start with the building blocks, rather than the building. In their literature review on computational circular design, Huang et al. (2021) talk about bottom-up and top-down design approaches. They consider top-down processes as rooted in parametric design, starting with "*a 'target' design concept model*", from which the parametric algorithm is written. Again, this can be linked to design at large via the guiding theme - van Dooren et al. (2014) describe how the guiding theme can start from the abstract and develop of time to a concrete but still open 'form language'. However, van Dooren et al. (2014) state that the inverse is also possible, starting from concrete forms or ideas which are translated into other aspects. This is very close to the bottom-up approach described by Huang et al. (2021), which they categorize as the aggregation of available material objects into architectural assemblies. These processes are highly iterative, sequentially placing elements, which can be recognized as an algorithmic design process through the framework of Caetano et al. (2020). Huang et al. (2021) merits further consideration, however, because they discuss computational design in terms of circularity, and introduce a critical concept in doing so: Material availability.

There is a considerable difference in material-first design which assumes a material is homogeneous and in near-limitless supply, and one which does not. If a material has dependable and consistent sizing, then the form language can exist as a direct expression of those qualities: Pattern generation. However, under the consideration of circularity it is difficult if not fundamentally counter-intuitive to consider materials as dependable as readily available. Rather the opposite becomes true. Materials are harvested from previous projects, waste streams or even completely different fields of industry. As such, creating assemblies from these stocks is perhaps more closely aligned with the act of creating a mosaic than of building patterns.

Huang et al. (2021) develop their own method and design tool through the top-down parametric lens. They produce a design algorithm for the creation of geodessic domes from linear wooden elements with a variety of lengths. The model contains two characteristics which are often found in parametric projects of this specific nature: There is a stock matching and stock cutting trade-off. Consequently, their algorithm produces Pareto optimal solutions which use both a smaller number of the actual stock items, up to only 2%, and often require the cutting of

said items to a certain degree. This is a common drawback, also apparent in Brütting et al. (2020) and Bukauskas et al. (2017), which the authors also cite. In both these researches, the cutting waste becomes an explicit parameter for optimization. Whilst the percentage of material which is cut falls below 10% in both researches, it still means that the materials 'erode', impeding reuse further down the time-line. In publications as recent as Favilli et al. (2025), the issue of material cut-offs is still a primary barrier. Brütting et al. (2020) also allows for a combined usage of reuse stock and virgin stock — and through an LCA can assert that, depending on the context, mixed usage actually creates the lowest environmental impact. However, this does operate under a narrower perception of what constitutes locally available material, as their research pertains to reuse alone.

Part of the philosophy underlining this thesis is to take a wider look at local materials, beyond what is sitting in the built environment itself. It is not the objective to reuse objects from the build environment to fulfil a mostly similar function, but to repurpose materials from outside the build environment to fulfil a new one. This does make the material question more complicated still, because the materials considered are not innately designed for the new purpose which is assigned to them. An example of this is the research of Nian et al. (2022) who use a parametric model to create modular truss assemblies from bike frames, a piece of research which was very influential to this thesis at large.

However, Nian et al. (2022) as well as all other research addressed above is part of the top-down parametric design method. What is yet to be discussed is the application of the opposite, the bottom-up algorithmic design method. This approach is implicitly defined by the material, and due to the iterative nature of algorithmic approaches, stock matching or cutting need not be a design parameter by default. It is not an effort of compromise between material reuse and a target design model, but an effort of creating designs directly from the material which is available. This is the essence of material-first design. Not to mediate between design and circularity, but to enable design through circularity. The variance of stock is no longer a barrier to overcome, but an advantage. However, contemporary research into these bottom-up generative design methods does not necessarily achieve this.

In *Matter as Met*, Eschenbach et al. (2023) develop a generative design method using mixed linear-integer programming to create combinatorial assemblies from a stock of concrete elements, sourced from the built environment. They assert that most contemporary parametric design and fabrication processes as stuck in a linear-economic "*digital-real*" process chain, and use their generative method to propose a "*real-digital*" process instead. However, their method is heavily reliant on a common theme: Refabricating stock into modular assemblies. Modularity is a way through which the variance of stock can be surmounted. Still, the drawback of this is that stock is still effectively cut, not leveraging the variance as a positive.

Other researches, such as Rossi & Tessmann (2019) formalise a kind of approach that is more in line with the exploitation of variance, rather than the absolving of it. They propose a voxel-based method to aggregate heterogeneous materials into larger assemblies. The work of Allner et al. (2019) uses similar methods to aggregate natural materials with a high degree of irregularity - tree forks. In the final stages of assembly, a mesh-relaxation step is performed to ensure connections between the different forks. Both of these researches make use of the Grasshopper plug-in Wasp, developed by Andrea Rossi (2017) for their doctoral dissertation. This tool is capable of iteratively combining a stock of varied elements, either in an arbitrary fashion or under the prescription of certain rules or design gestures. However, the work of Allner et al. (2019) highlights especially how there is a need to mediate between the form and the final design still, cutting the stock to a degree in order to have a consistent and interconnected shape.

This highlights a real research gap: Generative design principles naturally align with material-first design, due to the ability to iteratively build a structure from a stock of varied elements, but most contemporary research does not resolve the issue of modification, lessening the degree to which the stock variance is a quality to be exploited instead of just a barrier to overcome.

The value-proposition of this manifesto is this: Material-first generative design processes can be used to create assemblies of reused or repurposed materials which are integrated with very little cutting or refabrication. However, little contemporary research exists which achieves this outside of a specific field of knowledge: Reinforcement learning. As was previously stated, assembling structures from a varied stock of components is more comparable to the arranging of a mosaic than it is to the building of patterns. This kind of practice is characteristically intuitive for people, but not easily translatable to the logic of computers. To make assemblies such as these requires reasoning, an ability to estimate, predict and anticipate. This is why reinforcement learning has a high level of potential for the application of material-first generative design, and why previous researchers in this field have succeeded in creating assemblies directly and entirely from a varied stock of elements. However, these researches have been kept back for the section on reinforcement learning itself.

As was stated before: If proper analysis of a general design process merits consideration of the guiding theme and how it is developed, than proper analysis of computational design processes merits consideration of the algorithm and how it is developed. Having now outlined and contextualized material-first computational design as a generative practice with a high-potential application in reinforcement learning, it now leaves the question of how to conceive of the actual generative design principle - which is a kind of design process in its own right. This requires a methodological approach which is well attuned to the specific nature of generative design development.

Design Games and Shape Grammars

This final section of the manifesto's theoretical framework serves to outline the methodological basis by which the actual generative principle is conceived, or differently put: Designed. The practical question is this: How is a generative design principle actually established?

In their generic principles, van Dooren et al. (2014) recognizes that a guiding theme can be establishes from two perspectives: An overarching abstract design concept on the one hand, and through the concrete and material world of things on the other. The authors recognize the latter practice as the development of a design language. Huang et al. (2021) makes a narrower assertion yet, stating that bottom-up computational design is founded in the concept of shape grammars, a common concept throughout generative design literature at large.

Shape grammars are a type of codification first proposed by George Stiny and James Gips well before the advent of the household computer (Stiny & Gips, 1971). A given shape grammar consists of a tuple with four elements (G. Stiny, 1980):

1. A set of shapes (S). Shapes are defined by maximal line representation, meaning the fewest amount of non-intersecting lines which can describe the shape accurately.

2. A set of symbols (L). These can become labels for a given shape when associated with a point on said shape. Combining the shapes and labels create the set (S, L).

3. A set of rules (R), formalized as (α) -> (β). It is stated that (α) must be part of the set (S, L) whilst (β) may also be what is known as the empty labelled shape.

4. An initial shape, (I). This shape must be part of the set (S, L).

The advantage of these kinds of codifications is that they allow for generative design principles to be communicable and explicit. Recalling that the laboratory of design serves in part to make experimentation explicit through a visual language, shape grammars aid to structure said visual language when it comes to generative principles.

Before a generative principle can be applied to a design scenario, it is required to define the principle itself. Borrowing from Caetano et al. (2020), it can be asserted that generative design consists of the repeated application of a set of rules. Whilst the aforementioned shape grammars provide a formal definition of said rules, it does not provide a method to attain them.

Conceiving of these rules requires a process of experimentation, which itself occurs inside a laboratory of design. This laboratory therefore needs to support the testing and evaluating of rules based on a constrained stock. Moreover, the intent is to leverage the unique elements of said stock, meaning that the laboratory needs to reflect said variance as well. As such, the laboratory becomes constrained, imposing limitations on what the designer can and cannot do.

Take the laying of a mosaic as a comparison for this kind of laboratory. The designer is constrained by the available pieces, which are each shaped in a unique fashion. It would be difficult to argue that the designer conceives of a set of rules which fully accounts for every possible shape at their disposal. Instead, the designer is able to leverage something unique: Intuition. The goal of the experimentation process is not to conceive of a set of rules which fully accounts for the variance of the available stock, this would be an incredibly complex endeavour. Instead, the idea is to create a smaller set of rules and to then train a reinforcement learning agent on said rules to manifest a kind of intelligence, perhaps similar to human intuition.

The reason to include this aspect now, is that it further informs the way in which the laboratory is shaped. Ideally, it would not only support the testing of rules, but it would also provide an analogue for reasoning, to better conceive of a set of rules which can be utilized in the reinforcement learning section.

The work of Mirra (2022) provides the fundamental approach to the laboratory of design. In their doctoral dissertation, Mirra outlines three learning mechanisms for the training of artificial intelligence design applications: Expertise, playfulness and analogical reasoning. Expertise can be likened to the frame of reference, the capacity to learn from precedents. Playfulness can be seen as a pure kind of experimentation, based on the development of human intelligence through learning by playing. Analogical reasoning deals with the transference of knowledge from one domain to the application in another.

The second learning mechanism, playfulness, is especially relevant to this thesis. It is concerned with the emulation of unconstrained play, experimenting without a preconceived outcome. The generative principles to be determined should to a degree exist outside of an intended design. The goal is to discover the potential use behind a shape or form, not to prescribe a usage on it. Mirra (2022) goes on to formulate a method of experimentation, which they call: *Design games.*

Design games are an open ended practice, and as the author denotes, the name itself is a bit of a misnomer. Games are intended to follow specific rules, whilst design games purposefully do not. Rather, design games are intended to be as unconstrained as possible to provide a space in which constraints can be tested.

Before moving on to the methodology of this chapter, it pays to briefly outline the findings of the theoretical framework. Firstly, design processes at large are outlined through the generic elements of van Dooren et al. (2014), where the laboratory is flagged already as an especially important concept to define for the purposes of this thesis. Through Ostwald & Lee (2020) the comparison is made between the guiding theme and algorithmic development between computational design processes and design processes at large. Computational design is more accurately defined in the subsets of parametric, generative and algorithmic design via Caetano et al. (2020) and further contextualized in circular design via the top-down and bottom-up approaches described by Huang et al. (2021). Material-first design, as it is considered in this thesis, is characteristically generative and bottom-up. However, current research indicates a knowledge gap - most contemporary bottom-up computational design research performs some kinds of stock cutting still, with the exception of reinforcement learning researches, which offer a powerful solution to address this gap. The reason for this, is that generative design based on a varied and constrained stock is a kind of endeavour which arguably requires some kind of intelligence, likened to the intuition of humans in practices such as the making of mosaics.

It is determined that the conception of the design rules should not be in an effort to fully address the potential variance of a stock. Instead, a smaller set of rules can be formulated on which a reinforcement learning agent can feasibly be trained. Conceiving of these rules requires a process of experimentation inside a kind of design laboratory which supports the testing of rules, hence the significance of this generic element. Shape grammars offer a potential formalism to describe the rules, but do not provide a means of to attain them. For this, design games as outlined by Mirra (2022) are introduced as an open-ended kind of design practice which supports the testing of rules.

3.2 Methodology

Having outlined the practice of material-first design by generative computational means, the following method section will address not how to write said generative algorithm, but how to derive at the more fundamental rules, before having to write any actual code. For this purpose, design games have been outlined in the theory, as well as shape grammars. The methodology will go into detail regarding the specific setup of the design games and provide an example of the shape grammar formalism.

Design Games Setup

Design games are, in essence, an analogue to the digital generative process. There are several beneficial learnings to be derived from them, but the primary objective is to have an unconstrained and open-ended kind of experimentation without any notion of a pre-conceived design. The question is not how to achieve an objective, but instead: what kind of objective might be achieved? The practice also helps to prove the validity of a design strategy prior to the development of the code, simulate a child's mode of playing as a legitimate mechanism for the development of artificial intelligence and to produce an AI agent that can interact with the human designer throughout the design process.

Setup — Objects

The stock that is considered for this thesis, as explained in the introduction, consists of bicycle frames. However, this immediately raises the question of how to represent the stock in a physical way and which criteria need to be formulated to achieve a successful experimentation. For the sake of accessibility, 3D printing was chosen to create scale models of bike frames. Having this preconceived notion, the following criteria have been defined through several iterations of trial and error:

1. The bicycle frames must be appropriately sized as to be easy to manipulate but not to become too fragile.

2. Variance between frames can be created through randomization across defined domains, but the proportions must remain realistic.

3. The frames may be joined via a connector. This connector must form a decently rigid connection, but should also be removable with relative ease.

4. The frames must be printed to minimize overhang on the one hand and delamination on the other.

Each of these criteria will be addressed in further detail. The aspect of size and scale was narrowed down rather immediately. The choice was made to print each frame at a 1:15 scale, which proved a good balance between the rigidity and the ease of handling. Of the 80+ frames which have been printed at this scale, ~10% of the frames have failed in some capacity. The filament used was PLA, with a nozzle size of 0.4 millimetre and a layer height of 0.2 millimetre. Printing occurred in batches of 32 frames, oriented as shown in **Figure 3.2.3**, with the aid of tree-type supports. This orientation proved to be most optimal with regards to the overhang, requiring fewer supports. The

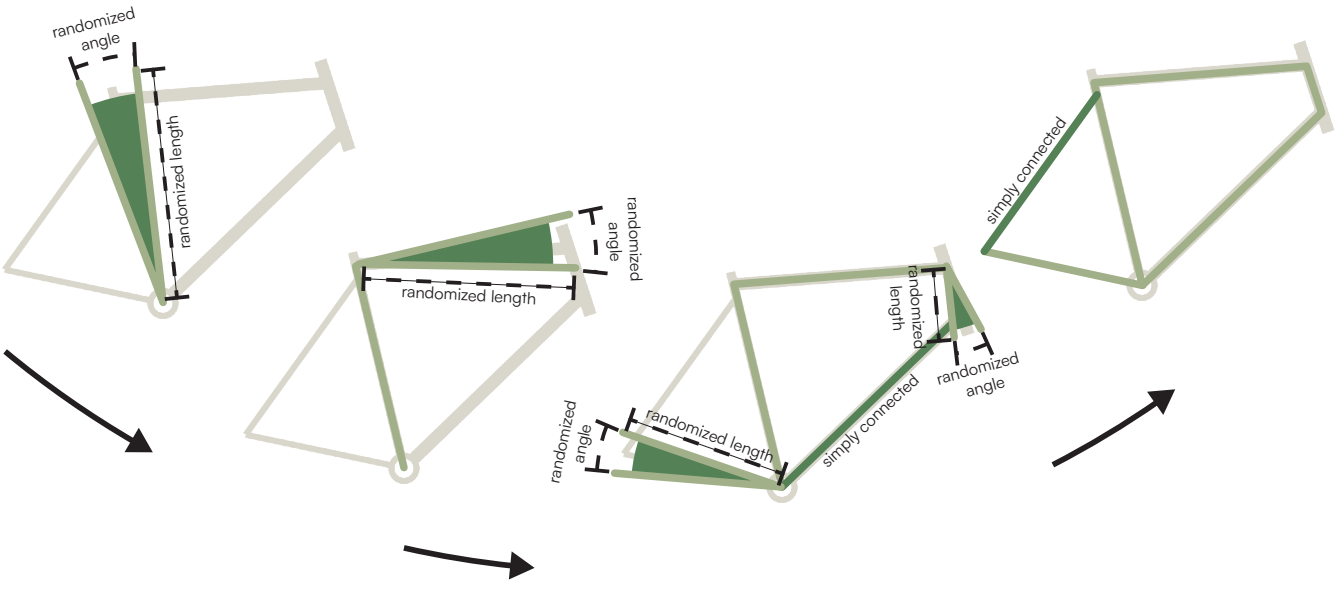


Figure 3.2.1 - *Flowchart of Randomized Bike Frame Generation* - Own work

disadvantage is that most tubes have layer lines which are in the normal direction, rather than the tangent — which makes them more prone to delamination. Improvements could be made if a different filament or scale was used, however, PLA is relatively biodegradable, as opposed to stronger plastics like ABS.

A Grasshopper script was written for the creation of the digital models, in such a way that the output could be directly exported to the chosen slicing software. The setup is illustrated in the **Flowchart 3.2.1**. As listed in the criteria, the variance between frames can be created through randomization, but it is key to maintain realistic proportions. The baseline shape for the model was determined through the use of the basic dimensions of an on-road bicycle frame model as determined by Cheng et al. (2016), The research of Cheng et al. (2016) is specifically targeted at multi-objective optimization for permanent deformation reductions in the frame itself, however, the baseline model provides a good starting point for this chapter, as the accuracy of the representation is focused on variance, but not yet on the degree of accuracy in emulating real-world stock.



Figure 3.2.3 - *Printing Orientation for Frames* - Own work

As the **Flowchart 3.2.1** highlights, the randomization process occurs in an iterative fashion, where first the seat tube is generated within a chosen boundary for both length and angle, then the top tube is generated in similar fashion, after which the head tube is generated and the down tube is simply used to connect the head tube back to the bottom bracket. The chain stay is then also generated using bounds for angle and length, and the seat stay is again used to form the connection to the seat tube. This method ensures that certain conditions are met, such as the top tube always having a +90-degree angle relative to the seat tube, as it is rarely seen that the top tube has an angle fewer than 90 degrees relative to the seat tube. In this way, the proportions of the bike frame are maintained. The domains have been set rather arbitrarily, using visual confirmation to ensure that the estimated ranges remain reasonable. Again, this was deemed an acceptable approach because the focus of this process was to ensure variance, rather than a highly realistic representation.

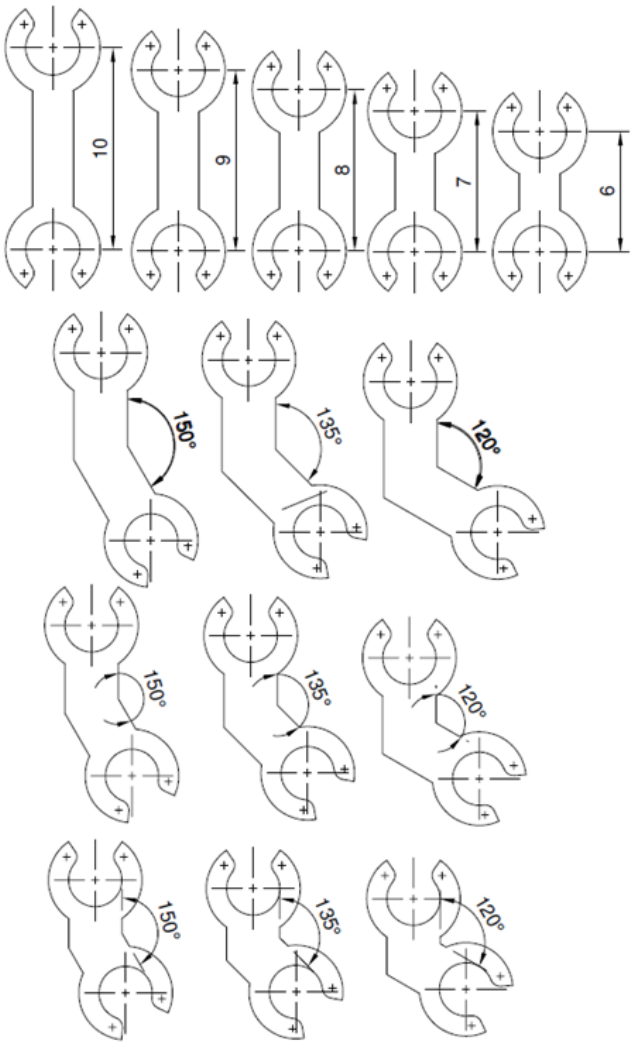


Figure 3.2.4 - *Varieties of Connector Elements* - Own work
* units are in millimeters

Setup — Connections

Initially the thickness of the different tubes was also randomized, though a multiplication constant was used to preserve the general hierarchy, as head tubes are rarely thinner than the down tubes, for example — however, this was only used in the first batch of frames, where the connections were achieved with the use of removable adhesive putty. This connection proved difficult, not permitting load-transfer and easily disconnecting. As such, the choice was made to keep the diameters consistent across the various frames, and to replace the use of adhesive putty with a discrete connection. The design of this connection is based on the hands of a Lego figure. Critically, Lego uses a process of injection moulding with ABS plastic which means that the material is much more isotropic and can deform more elastically compared to 3D printed PLA (*The LEGO® Moulding Philosophy*, n.d.). As such, the designed connections will lose gripping power over the frequency of use. In the design games, connections typically lost their gripping power after 3-4 uses. The modelling for these parts was done using Fusion3D, keeping most dimensions as adjustable parameters. **Figure 3.2.4** shows the variety of connectors which were created using this file. There was also an attempt to design a ball and socket connection, but this proved difficult to achieve due to the small scale.

Shape Grammar Formalism

Shape grammars are used throughout the application of the design games to formulate the actual rules being tested. In writing, shape grammars can often become highly mathematical and complex definitions, which isn't deemed necessary for the purposes of this thesis. Instead, a visual approach is taken to represent the rules, adapted from the approach outlined by Stiny & Gips (1971) in their conference article. **Figure 3.2.5** has been included here to provide an example of the visual representation of shape grammars in this thesis. In this specific example, the initial set of shapes is defined as both a bike frame and a connecting element, this is not always the case.

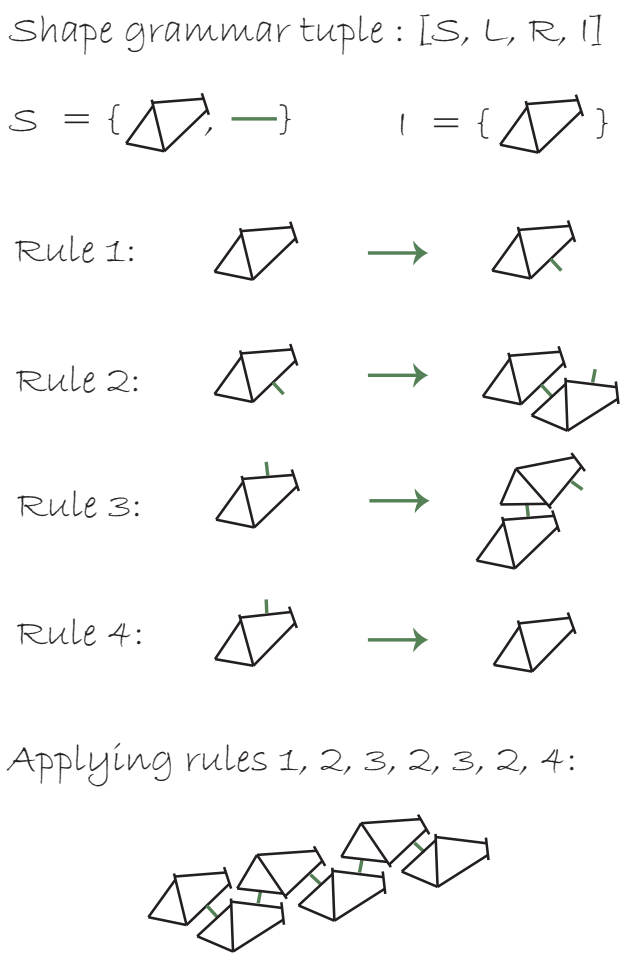


Figure 3.2.5 - *Shape Grammar Example* - Own work

Note that there is no label provided to differentiate between different frames in the example, this is a conscious decision. Experimentation proved it difficult to assign rules based on the specific variance of a bike frame. The selection of one frame over another is highly contextual and to a degree intuitive, and difficult to translate to a set of rules. The rules themselves make use of IDs for elements on the bike frame, outlined below in **Figure 3.2.6**.

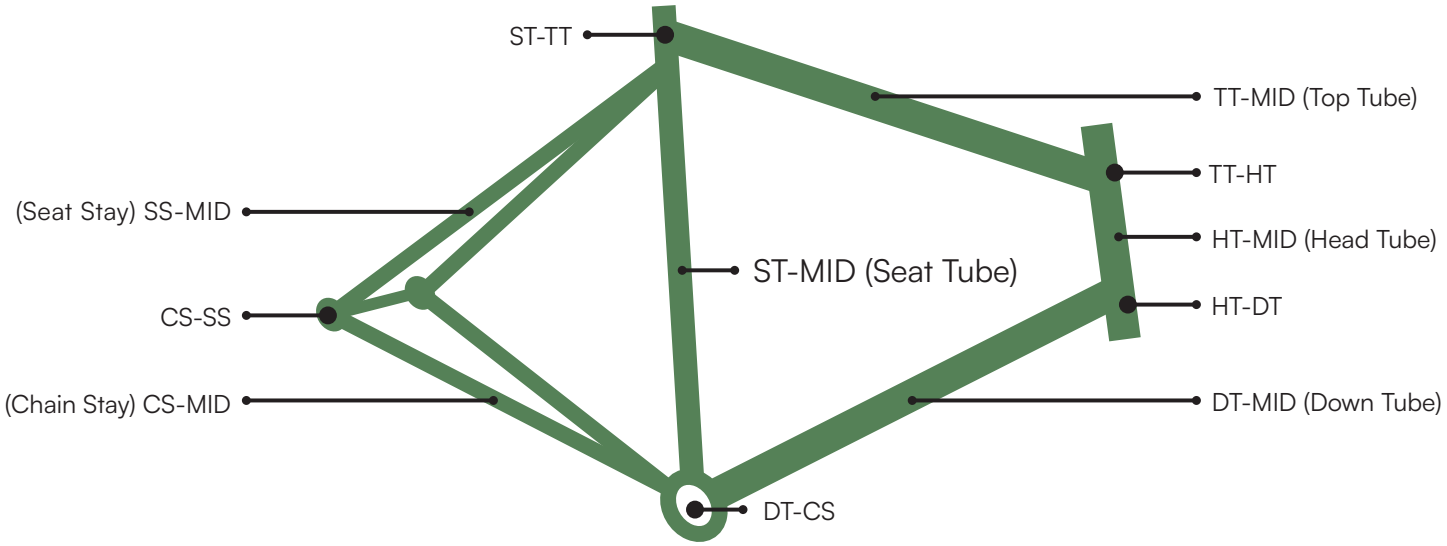


Figure 3.2.5 - *IDs Used in Shape Grammars* - Own work

3.3 Design Game Results

The following section presents the experimentations in through the design game practice outlined in the theory and method sections. Critically, the outcome of these experiments have not directly resulted in the determination of the generative principle, which was the objective of these experiments. The reason for this is simply the time constraint imposed by the thesis timeline, but it is further expanded upon with some early experiments in shape deformation which underscores the inherent complexity of 3D environments for design objectives such as these. Each experiment has been assigned a title, which is in essence arbitrary, but has been deliberately chosen to omit the potential application, as to not impose a specific design objective.

The Pyramid

The very first result of the experimentation process, the pyramid shape is a rather simple form with limited potential for expansion. **Figure 3.3.1** illustrates the shape grammar used to achieve this configuration. The starting shape is defined not as a single frame, as might be expected, but as a combination of three frames. This shape has a clear value trade-off. One the one hand, using only three frames results in a sturdy triangular configuration. However, it was also noted that the specific angular relationship between the top tube and down tube — between which the connection is made — makes the closing of the structure by means of a simple connection more difficult. Expanding the shape by inserting additional frames, in essence increasing the 'faces' of the 'pyramid', makes the connection easier to achieve, but impeded the rigidity of the final structure.

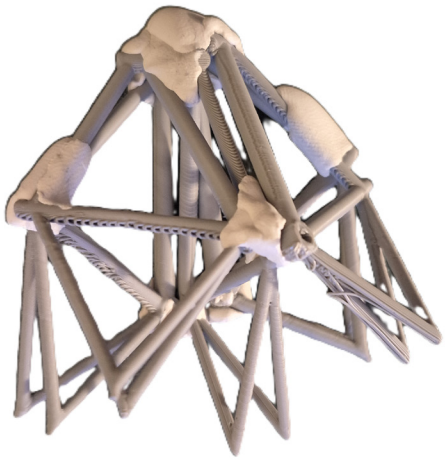


Figure 3.3.2 - *Pyramid Shape* - Own work

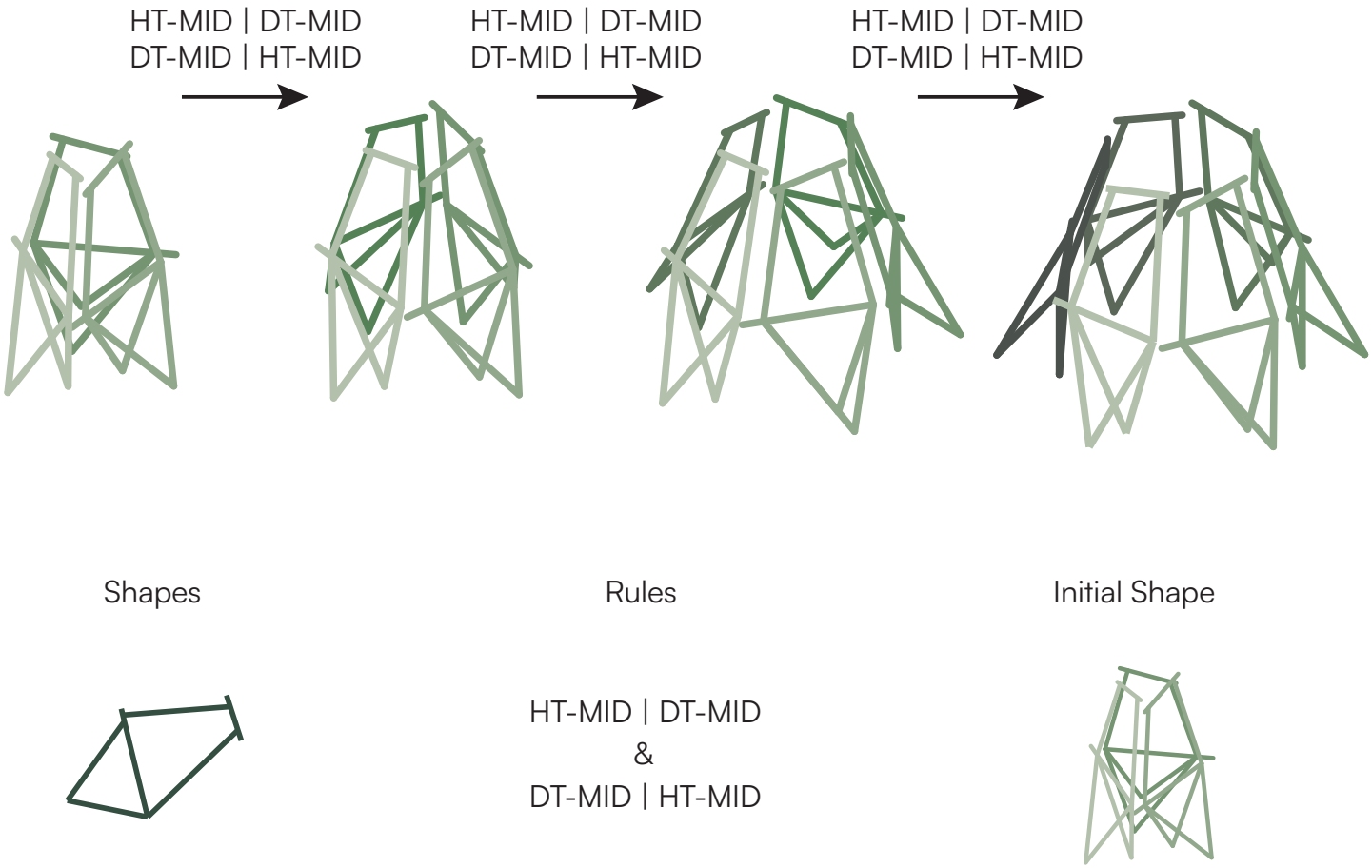


Figure 3.3.1 - *Pyramid Shape Grammar* - Own work

The Star

The star typology is created from two different shapes, both consisting of four individual frames, but connected by different means. The first shape relies on connections between the top tube and seat tube. The second shape does not rely on the use of the prescribed connections, but instead makes use of the bottom bracket and the head tube, which could in practice be clamped together via the recess of the bottom bracket. The overarching idea is that the two shapes could interface to create a larger 2D pattern, which is possible when assuming that the frames are true parallelograms, as in **Figure 3.3.3** — however, practical experimentation showed that this is not the case for the actual shape. Furthermore, these kinds of patterns introduce an intermediary step prior to building them, which is to construct an initial base shape. This further complicates the workflow, as the designer or algorithm first needs to build a combined object from the stock prior to placing it in the larger pattern.

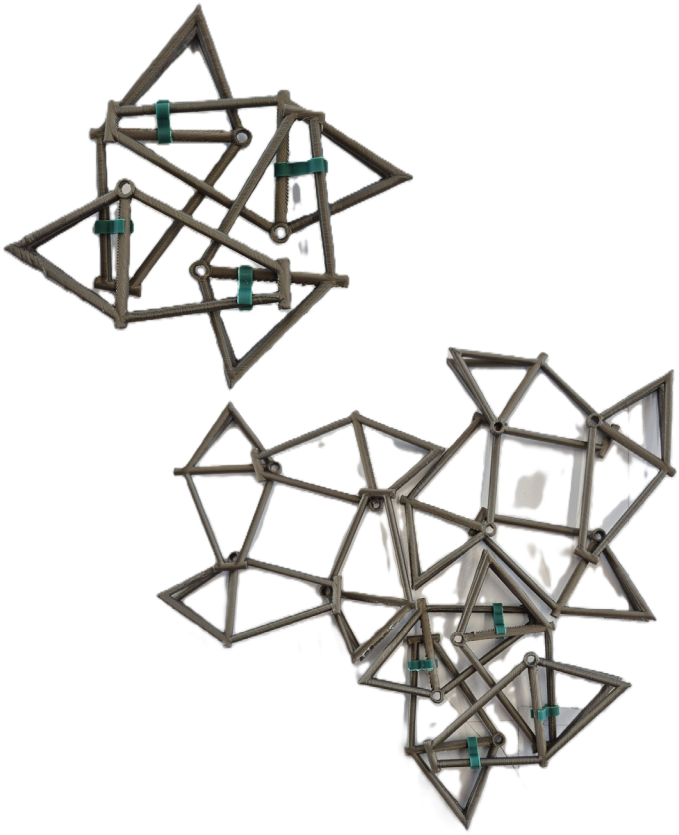


Figure 3.3.4 - *Star Shape* - Own work

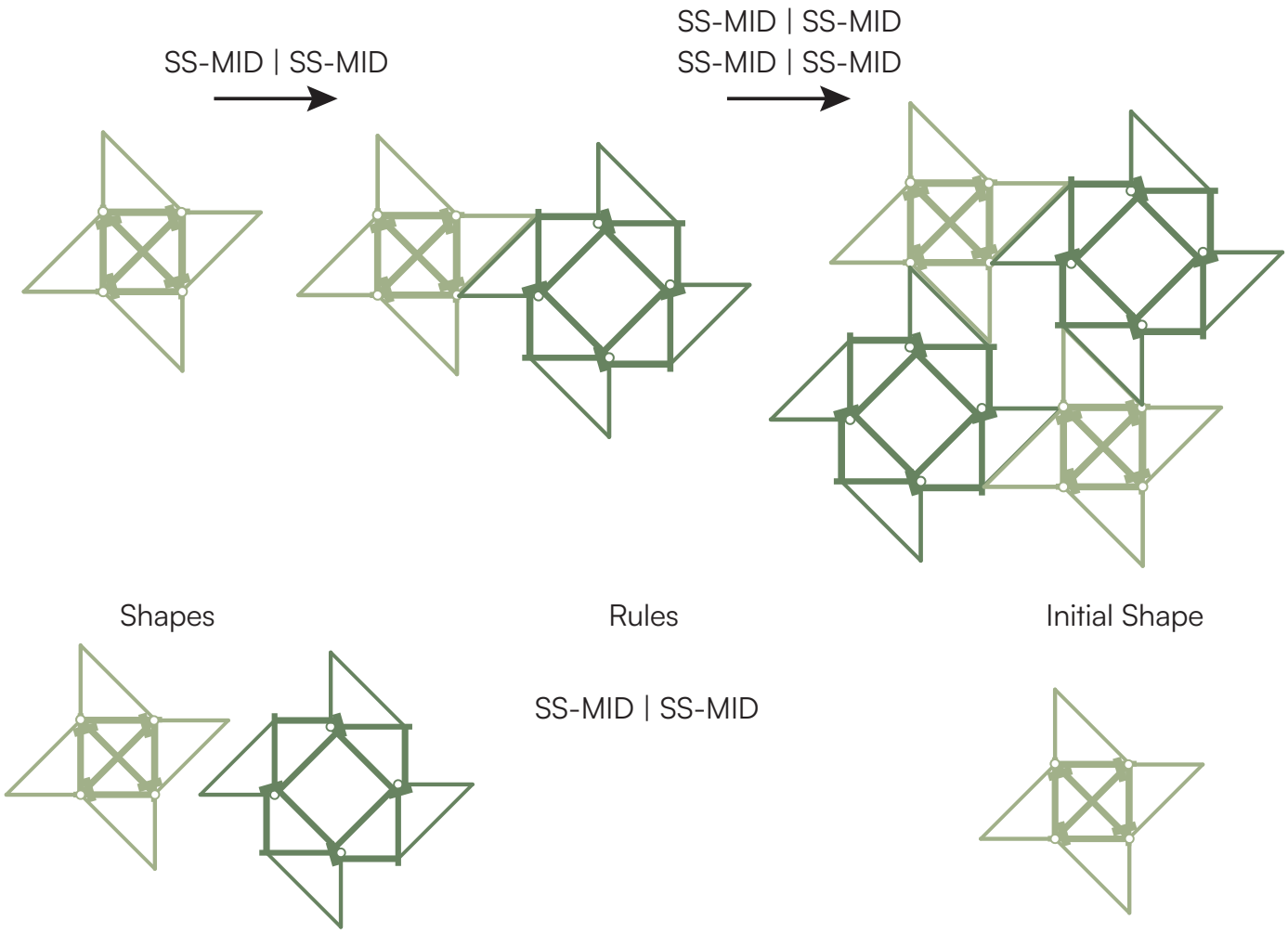


Figure 3.3.3 - *Star Shape Grammar* - Own work

The Tube

Perhaps most investigated in these design games is the tube concept. It has a similar basic principle as the pyramid typology, where the amount of faces can be increased. Similar to the star pattern, the grammar could be constructed from assemblies, rather than individual frames. However, it would also be possible to construct the grammar from individual frames, treating it as an inverted unwrapping procedure, exemplified in **Figure 3.3.5.**

This configuration saw more extensive development compared to the previous two, most notably performing several form-finding tests via Kangaroo, a plugin for Grasshopper. These results have not been included for sake of scope.

This brings the results of the design games to an early closure, as the general experience proved that 3 dimensional structures prove very difficult to intuitively assemble. This exercise is intended in part as a analogue for reasoning, to predict how difficult a given task might be for an RL agent to accomplish. With 3D configurations being quite difficult to create and extend, and 2D configurations with assembled shapes being similarly difficult too, the decision was made to adapt a much less complicated 2D principle.



Figure 3.3.6 - *Tube Shapes* - Own work

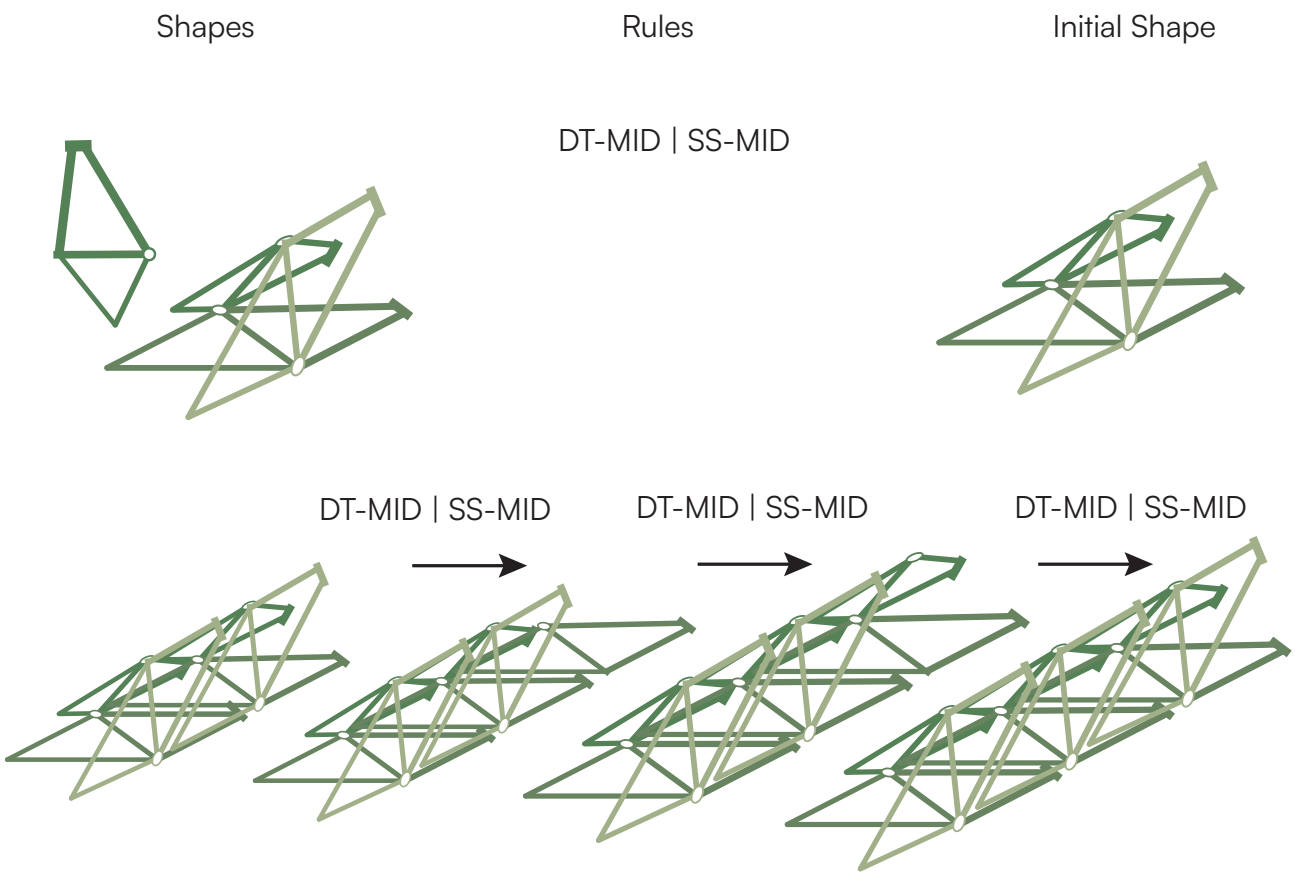


Figure 3.3.5 - *Tube Shape Grammar* - Own work

Final Generative Principle

The final generative principle has been simplified significantly compared to the experiments of the design games. To this end, the grammar rules have been limited to the aligning of frames via the midpoints of the tubes, keeping the elements in parallel to each other. The grammar consists of four elements:

1. Frame selection
2. Candidate orientation
3. Target orientation
4. Mirroring

Figure 3.3.7 has simplified the grammar slightly, omitting the selection of the frame. Furthermore, the elements are treated as purely 2D, meaning that the chain stays and seat stays have been collapsed into two elements, instead of four.

Intermediary Conclusions

Concluding on the results of the design games proves difficult, in part because of the more subjective nature of the practice, but more importantly because the practice did not yield a result to be used in the rest of the thesis directly. Instead, the generative principle was conceived in an effort to keep the scope of the thesis workable. That being said, the theoretical framework has served to further contextualize the material-first computational design principle in design practices at large, providing a more elaborate design philosophy, which has been a guiding theme for this thesis at large.

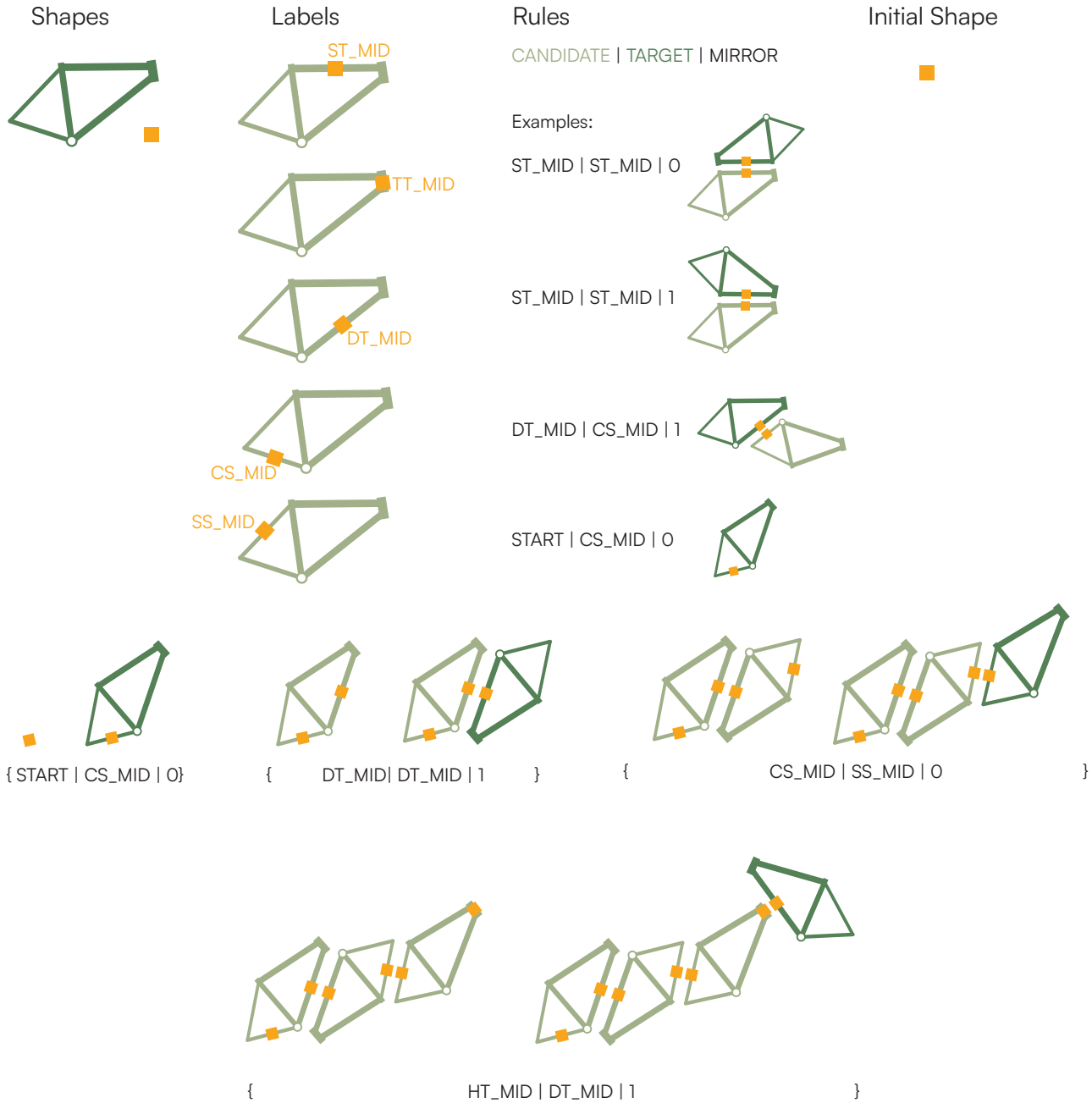


Figure 3.3.7 - *Final Shape Grammar* - Own Work

4.0 MATERIAL CASE STUDY: BICYCLE FRAMES

The case study of this thesis is split into two separate elements: The material case study and the design case study. The manifesto served to contextualize the concept of material-first design in both general and computational design processes, recognizing that the material-first concept best aligns with generative design. It is as such characterized through two distinct design phases: Designing of the generative principles, a kind of form language derived from the material - and applying said principles to an actual design scenario. First material, then design. As such, the following two chapters serve to elaborate on the case study via these two elements.

The design games of the manifesto served to provide a means to conceive of the principles, working from a 1:15 set of bike frames to engage in free-form experimentation without a prescribed target design. These scale models relied much heavier on arbitrary randomization rather than scientific literature, which was deemed acceptable to jump-start the design process. As such, this chapter serves to close the gap, building datasets with a higher degree of research to support them, to then utilize said datasets for the actual development of the design algorithm.

Chapter Objective

This chapter specifically deals with the theory and method required to build representative datasets of the materials, so that these can be used in the development of the design algorithm. The goal is to construct a dataset which reflects the geometrical and structural qualities of bicycle frames. As had been asserted in the introduction, the quantity and quality of the waste stream flow of bicycles in the Netherlands is not publicly available. As such, the decision has been made in this thesis to focus instead on research within the field of bicycle frame design to find relevant data by which an approximate representative dataset might be constructed.

Relevant Research Questions

How can a stock library be formalized to represent a constrained stock of radically reused elements for the purposes of a structural generative design system?

4.1 Theoretical Framework

The theoretical framework of this chapter will focus on relevant scientific literature by which the representative dataset might be constructed, regarding the geometrical and structural qualities — which are the primary aspects required for the dataset to be used in a material-first design principle and more practically, the actual generative algorithm. The research of Nian et al. (2022) is considered fundamental in this thesis. Their research investigates the value proposition of waste materials that possess convenient structural qualities in the use for structural design. Through a research process which encompasses literature review and the formulation of a framework for adaptive reuse of waste materials they eventually select bicycle frames as a case study. Notably, the general design intend diverges from this thesis, as they seek to create modular assemblies from the frames — still, their research and the sources they provide offer immense value to this thesis.

Geometrical Qualities

One of the key metrics which the researchers of Nian et al. (2022) do not readily provide is the geometrical qualities of their bicycle frames. They have created their own frame topologies through the conduction of surveys and interviews which are not made available. As such, other literature was considered and adapted for the purposes of this thesis.

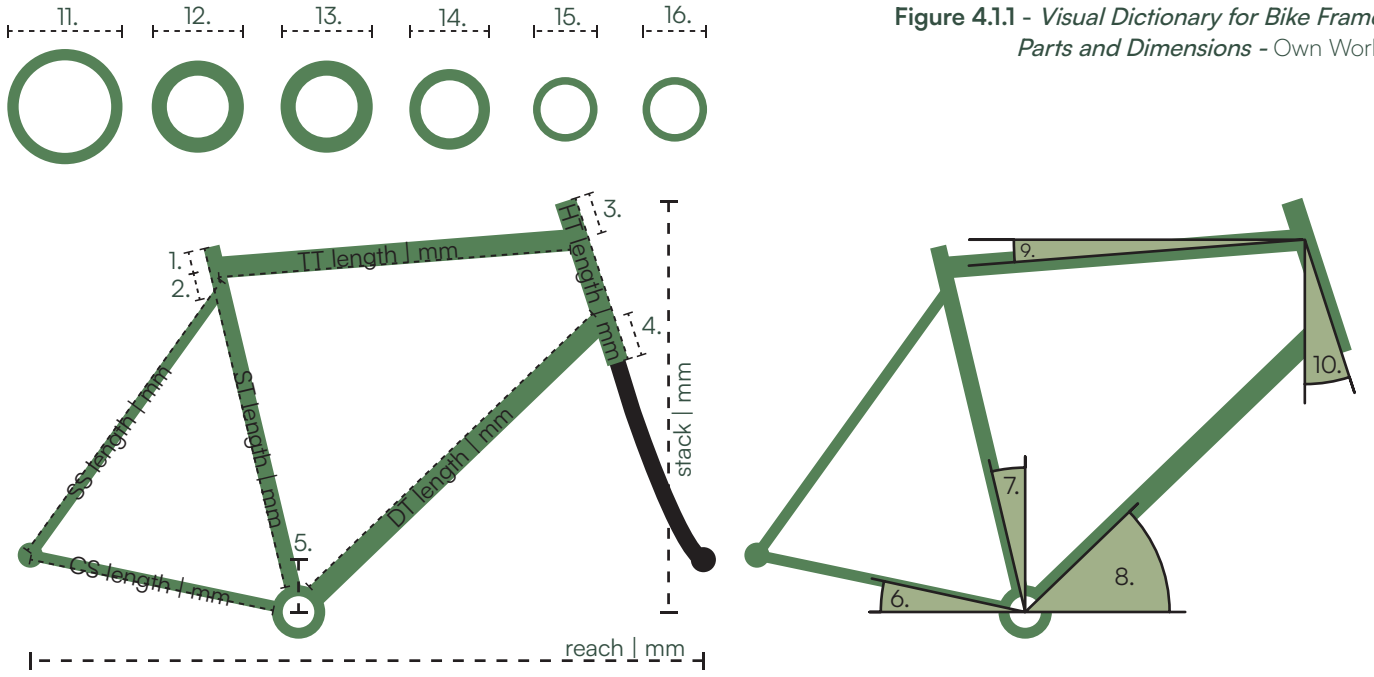


Figure 4.1.1 - *Visual Dictionary for Bike Frame Parts and Dimensions* - Own Work

1. seat tube extension
2. seat stay offset
3. head tube upper extension
4. head tube lower extension
5. bottom bracket drop
6. chain stay angle
7. seat tube angle
8. down tube angle
9. top tube angle
10. head tube angle
11. head tube outer diameter
12. down tube outer diameter
13. top tube outer diameter
14. seat tube outer diameter
15. chain stay outer diameter
16. seat stay outer diameter

The initial intent of this thesis was to create a mock dataset using randomization of length and angle parameters based on scientific literature. The first source for this was the research of Cheng et al. (2016) for the creation of the 3D printed bike frames, which has already been discussed in the manifesto methodology. This method was later refined with the parametric finite element analysis research of Covill et al. (2014), who employ a similar method of randomization and critically supply bounding domains for said randomization, see **Figure 4.1.2**, in turn based on a historical catalogue of frame geometries. Unfortunately, this catalogue is not accessible. Besides this, the domains provided are also incomplete, notably lacking the length of the head tubes and top tubes.

The parametric randomized approach proved valuable in the earlier stages of the research and design process, but was eventually supplanted by the dataset of Regenwetter et al. (2022), as it arguably makes for a more compelling source for the specific application of this thesis, being generative design and reinforcement learning. The dataset is part of the research project FRAMED, in turn a successor of the research project BIKED, both part of the MIT DeCodeLab. What makes this dataset compelling is where it comes from. The authors of FRAMED were able to extract a large swath of data from the open-source program BikeCad, a software platform specifically made for the design of bike frames, by both industry professionals and amateurs alike. Furthermore, both FRAMED and BIKED are aimed at the application of machine learning. BIKED is first and foremost geared towards the creation and validation of a dataset of 4500 bike frames, but also includes the training of multiple classifiers to help machines understand bicycle design (Regenwetter et al., 2021). FRAMED takes the same dataset into a different direction, demonstrating how Automated Machine Learning can be utilized to create surrogate models which can be utilized for engineering design problems (Regenwetter et al., 2022). The latter research critically provides a dataset which contains measurements for the bike frames themselves, including the thicknesses of the tubes and the materials they consist of.

4.2 Methodology

Having outlined the literature used to construct the different datasets, the following section will outline how the actual bike frame datasets have been constructed and importantly, which concessions have been made in an effort to keep the scope appropriate. This section is structured along the two main datasets which have been developed — firstly, the parametric dataset which is predominantly based on the domains provided through Covill et al. (2014). This dataset has mostly been used throughout the early stages of the development of the generative algorithm. After this section, the extracted dataset from the FRAMED database of Regenwetter et al. (2022) will be discussed in greater detail, as this is the dataset which has been used for the actual development and testing of the formalized generative algorithm and reinforcement learning routine. This chapter will not contain a results section, as the relevant research question is answered through the methodology itself. **Figure 4.1.1**, also present in the visual dictionary before the introduction chapter - has been added here to illustrate the different dimensions which will be mentioned in this section.

Parametric Dataset

The initial approach to the construction of the bike frame dataset was to develop a parametric model which relies on domains within which randomization is used to create different shape typologies. Critically, these domains are interdependent. There are certain implicit design rules when it comes to road-bicycle topology to which the research has been constrained. For example, the top tube will almost always have a positive angle when measured against the wheelbase (also known as the 'reach'). Independent randomization of variables would prevent adherence to these design rules. As such, an approach was taken where each element is constructed consecutively from the previous element, as illustrated in **Figure 4.2.1**. Critically, the down tube and seat stays are not randomized in this process, but are instead used to 'close the loop', ensuring that the frame is properly connected at each junction.

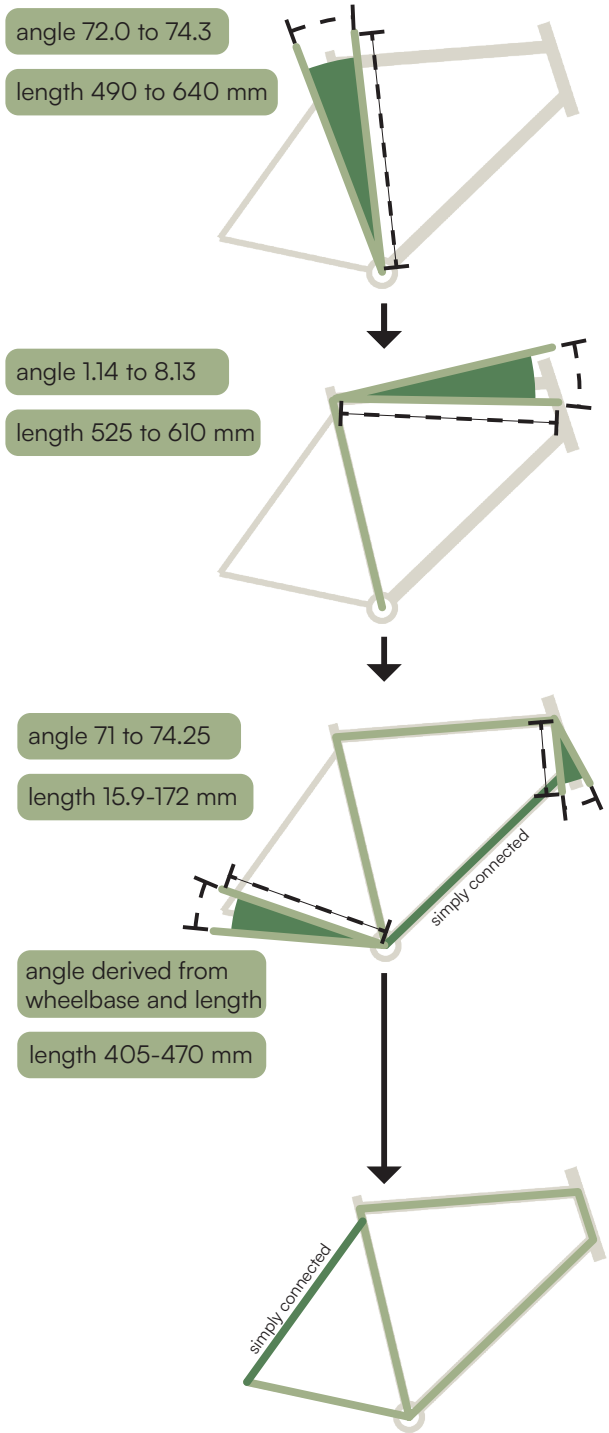


Figure 4.2.1 - *Workflow of Parametric Dataset* - Own Work

The domains for this randomization have been adapted from those which Covill et al. (2014) utilized in their own parametric model construction. The sample size from which these domains have been constructed is relatively low, 82 frames, and the database from which these frames have been extracted are not publicly accessible. Furthermore, the domains do not provide head tube lengths or top tube angles, which the authors also note. Initially, the head tube and top tube domains were supplanted using an arbitrary range imposed on the head tube length provided through Cheng et al. (2016), but were later replaced using a 95% mean analysis drawn from the FRAMED dataset of Regenwetter et al. (2022). **Table 4.2.2** contains the relevant normal distributions.

Extracted Dataset

Whilst the generative algorithm, more accurately the generative principles were developed using the parametric model, the choice was made to switch over to an extracted dataset from the FRAMED database of Regenwetter et al. (2022) at the onset of the genetic validation and reinforcement learning development. The FRAMED database provides immediate access to a large number of frames. However, the dataset is build from open source software with designs from both industry professionals and amateurs. As such, it is critical that the dataset is properly filtered and validated. Both of these aspects represent a few concerns which need to be addressed in order to extract a subset which is geometrically feasible and can be said to align with real-world design typologies.

The FRAMED dataset is provided as a csv containing measurements of each frame. Not every measurement has been required for the intended construction, which will be addressed throughout this section.

Extracted Dataset Construction and Validation

Not every measurement of every tube is provided by the authors, but a sufficient amount of them are present to construct the frames, which has been achieved using the parametric design software Grasshopper inside Rhino3D. The specific workflow has been outlined in **Figure 4.2.3**. The steps are explained below.

1. The first tube which is constructed is the seat tube, for which most measurements are available. It has been assumed that the seat tube measurement is taken from the centre of the bottom bracket.

2. Secondly the down tube is constructed. The authors of FRAMED have been so kind as to provide their own validation script, from which the approach could be verified. The down tube construction requires that the vertically projected length of the head tube is calculated, which, when subtracted from the stack, provides the vertically projected down tube length. From this, the angle of the down tube can be calculated using the length of the tube, which is provided.

Variable	95% Mean
Head Tube Length	15.924 mm to 172.284 mm
Top Tube Angle	1.1441 to 83.1370 degrees

Table 4.2.2 - *Logarithmic 95% Mean Distributions for head tube and down tube* - extracted from Regenwetter et al. (2022)

3. Having then determined the point where the down tube and head tube meet, the head tube itself can be constructed, for which all measurements are already provided. The top tube is assumed to sit between the upper extensions of both head tube and seat tube, which has also been checked against the software from which the dataset is drawn, BikeCad.

4. The chain stay can be calculated simply using the measurements which have been provided. The seat stay is assumed to sit between the edge of the chain stay and the seat tube extension of the seat tube. It has been assumed that the seat stay extension is taken from the total length of the seat tube.

Not every frame provided in the csv manages to converge under the prescribed method. Regenwetter et al. (2022) also denotes that there are invalid frames in the dataset, and provide a means to filter them. However, post filtering of these frames, the construction process used in this thesis has ran into several other frames which did not successfully converge. The filter of the FRAMED authors disqualifies 392 frames from the database, whilst a further 170 frames have been found insufficient in the construction method used in this thesis. The process of validation has been outlined in Flowchart **4.2.5**.

Extracted Dataset Simplification

The method used for construction already builds a simplified representation of each frame, as the data does not provide certain figures, such as the shape of the tubes, which are therefore assumed to be circular. However, as will become apparent in the chapter of the generative algorithm, the eventual design space has been kept two-dimensional, which warrants an additional step of simplification, as the seat stays and chain stays actually protrude out of plane, and sit on either side of the wheelbase. The decision was made to instead simplify the shape, having only one chain stay and seat stay. The effective cross-sectional area of these elements was doubles to approximate the actual cross-section, using the formula below, where OD = outer diameter and t = thickness.

$$\mathrm{OD}_{\mathrm{new}} = \sqrt{2 \cdot \mathrm{OD}_{\mathrm{old}}^2 - (\mathrm{OD}_{\mathrm{old}} - 2t_{\mathrm{old}})^2}$$

$$t_{\mathrm{new}} = \frac{\mathrm{OD}_{\mathrm{new}} - (\mathrm{OD}_{\mathrm{old}} - 2t_{\mathrm{old}})}{2}$$

Furthermore, the decision was made to move the chain stay and top tube connections with the seat tube to coincide with each other. This was done for the sake of further shape simplification, guaranteeing that most of the frames can be categorized as a convex polygon, though there are exceptions. **Figure 4.2.6** highlights the transition from the fully-fledged frame to the reduced representation which is used throughout all of the computational efforts post algorithm definition.

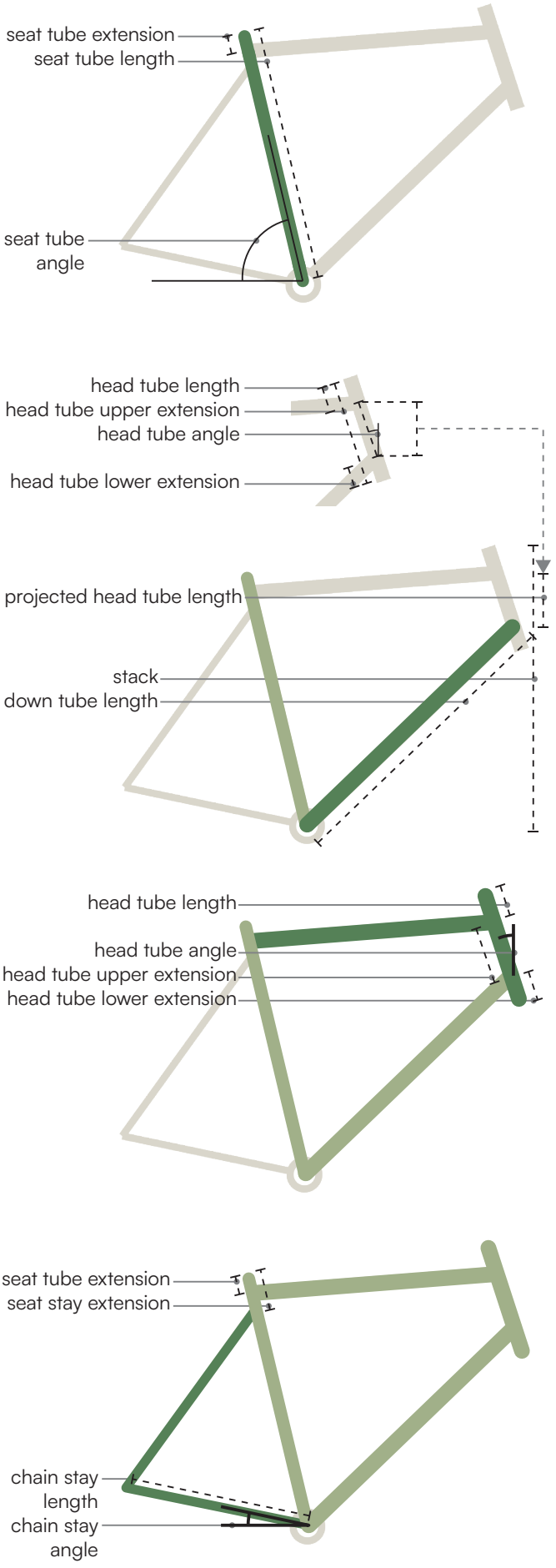


Figure 4.2.3 - *Extraction Process of Framed Dataset* - Own Work

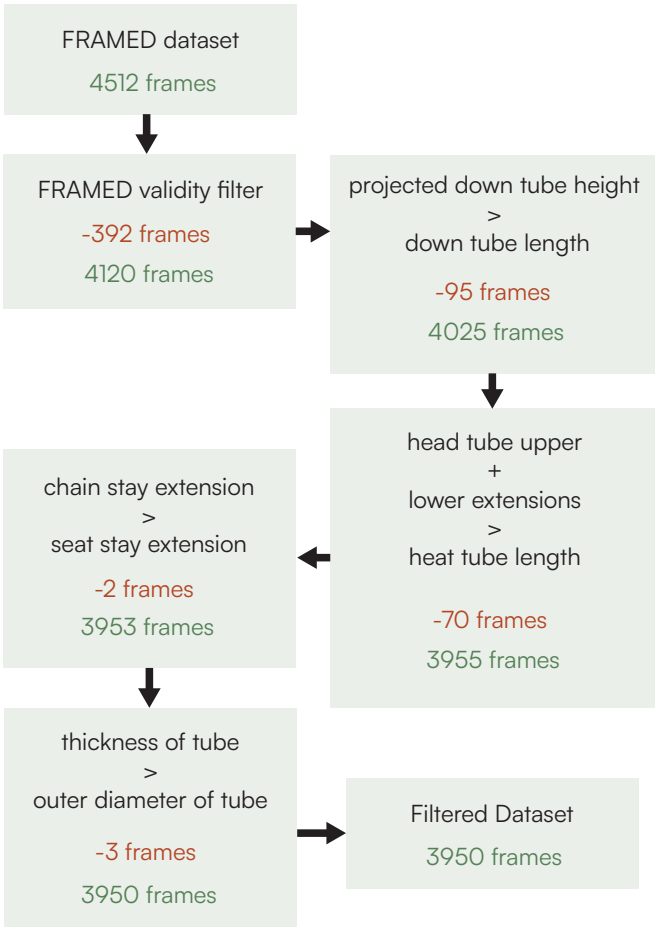


Figure 4.2.4 - *Dataset Validation Filter* - Own Work

Extracted Dataset Analysis and Filtering

The validation process produced a dataset of 3950 different entries capable of converging to a geometrically feasible bike frame. However, visual inspection shows that notable percentage of the geometries appear entirely unrealistic. As noted in the theoretical framework, the dataset is drawn from both industry professionals as hobbyists — which explains the existence of these oddities. **Figure 4.2.5** is provided to highlight two of these outliers. This raises the question of how to filter these frames to improve the degree to which the dataset reflects real world frame designs. Initially, the oddities were filtered through visual assessment. However, it proved more efficient and arguably more consistent to instead filter the outliers through a normal distribution analysis, followed by a 95% mean segmentation. **Figure 4.2.8** highlights the process in further detail. Not every measurement could be well represented through a normal distribution, those being the cross-sectional areas and notably the head tube. As such, these were represented with a logarithmic distribution. For a frame to pass the 95% mean filter, each of the measurements used to construct it have to lie within their respective 95% bounds. The process eliminated a further 839 frames from the database.

The filtering is performed on the simplified geometry, not on the direct data of FRAMED. This does raise another concern, however: Duplication. A considerable number of frames only differ in measurements which have not been accounted for or have been purposefully simplified. This produces a considerable number of identical frames. These have been filtered out, maintaining only one frame per set of duplicates, guaranteeing a unique dataset. This process filtered out another 451 frames.

Extracted Dataset Selection

The processes described above produced a geometrical dataset which is validated, filtered for outliers and ensures each entry is unique. This leaves the question of how to retrieve a subset from this dataset. Throughout the development of the algorithm, datasets were needed of only 25-50 frames. A simple random selection introduces a risk, because of the relatively large ratio of frames available to frames needed. The risk being that the subset does not accurately represent the diversity of the dataset. To address this, a k-means clustering was performed on the

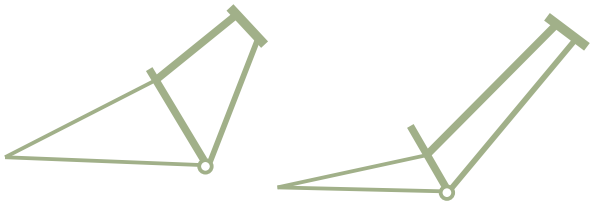


Figure 4.2.5 - *Example Outliers* - Extracted from Regenwetter et al. (2022)

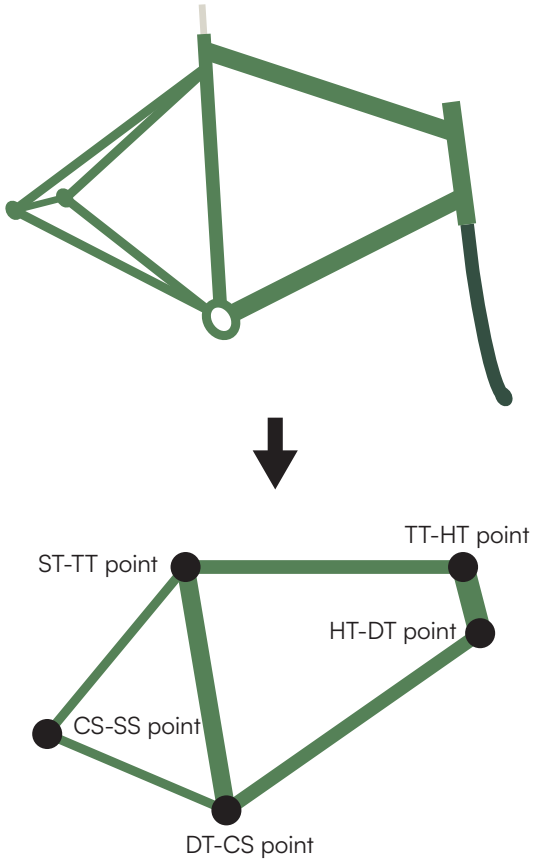


Figure 4.2.6 - *Frame Geometry Simplification* - Own Work

complete dataset. This was also accomplished via Grasshopper, using the Lunchbox plugin. Each frame was discretized as a tuple of 4 vectors, oriented between the bottom bracket and the 4 other points. K-means clustering is an unsupervised machine learning method which requires that the user defines the number of clusters on the outset. Several cluster sizes have been tested, but visual analysis proved a good distribution of shape typologies using only 3 clusters. **Figure 4.2.7** provides a small selection from each cluster.

From this point, all that is needed is for a specific number of frames to be given as an input parameter, at which point random reduction is performed on each cluster to retrieve a number of frames, which is in turn calibrated according to the respective sizes of each cluster. This ensures that the final subset is proportionally aligned with the clusters, and as such ensures that the variance discovered through the analysis is maintained in the subset.

Intermediary conclusions

The research question relevant to this chapter was given as:

How can a stock library be formalized to represent a constrained stock of radically reused elements for the purposes of a structural generative design system?

The 'how' of this research question can effectively be broken down into two aspects: How to represent a stock of elements and how to represent said elements for generative design.

An important metric for the first aspect is the degree to which the dataset accurately represents a real-world stock. Due to scope and lack of publicly available data, the degree of accuracy is difficult to quantify. Still, measures have been taken to best extract a representative subset of frames from the datasets which are available, notably through a filtering of outlying datapoints and a machine learning analysis of the constructed geometries. Whilst two datasets have been constructed, parametric and extractive, the prior is built on real world data but said data is not accessible. Only domains constructed by other authors from a small sample size of 82 frames are made available. The extractive dataset is directly drawn from a much larger dataset, but not necessarily representative of actual design principles within the manufacturing industry, though it does not rely on constrained randomization. Instead, each frame reflects an actual design made by an individual.

Considering then the second aspect, being the question of how to represent the stock elements for a generative system, the decision was made to simplify the geometries to only five points. The largest shift beyond this is in the way that the frames have been flattened into two-dimensional representations. The consequence of this being that the seat stays and chain stays are represented as only one line each, whilst doubling the cross-sectional area of said elements.

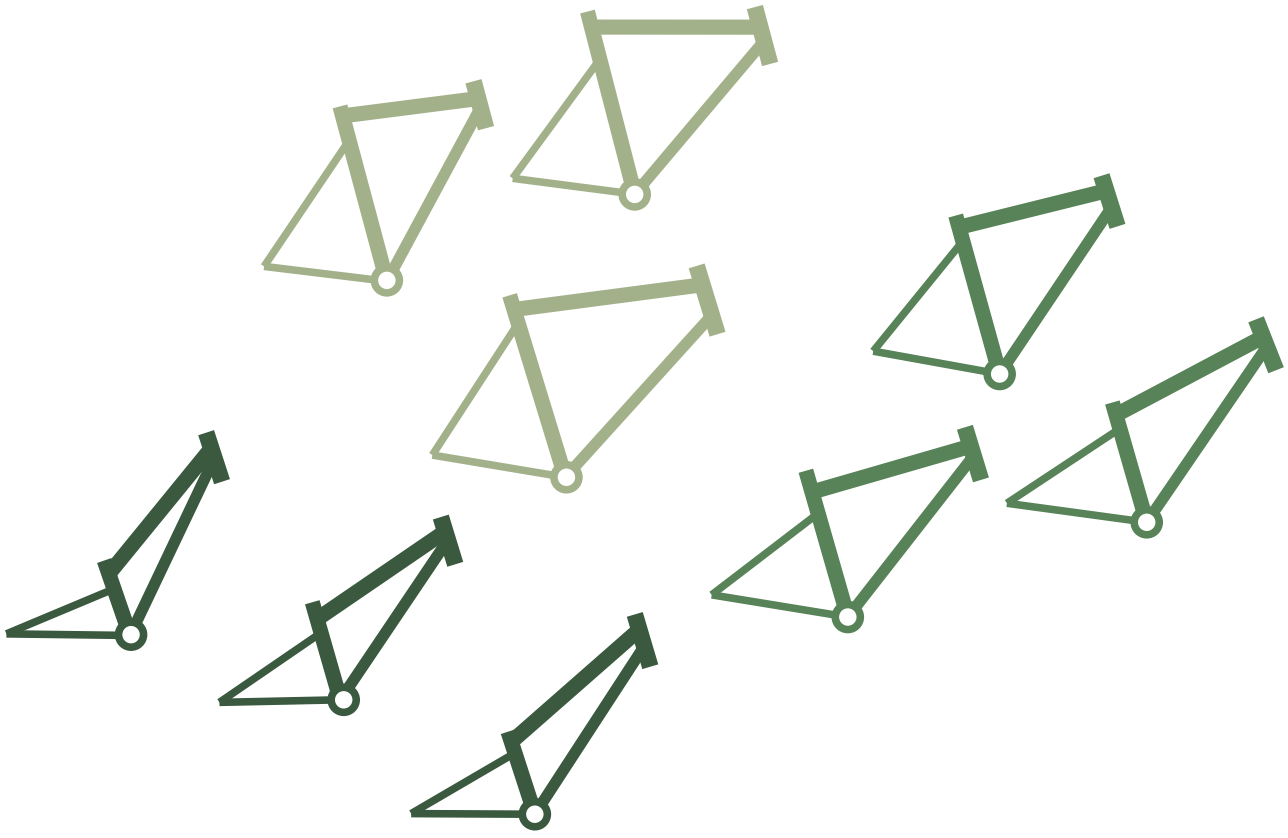


Figure 4.2.7 - *K-Cluster Examples* - Own Work

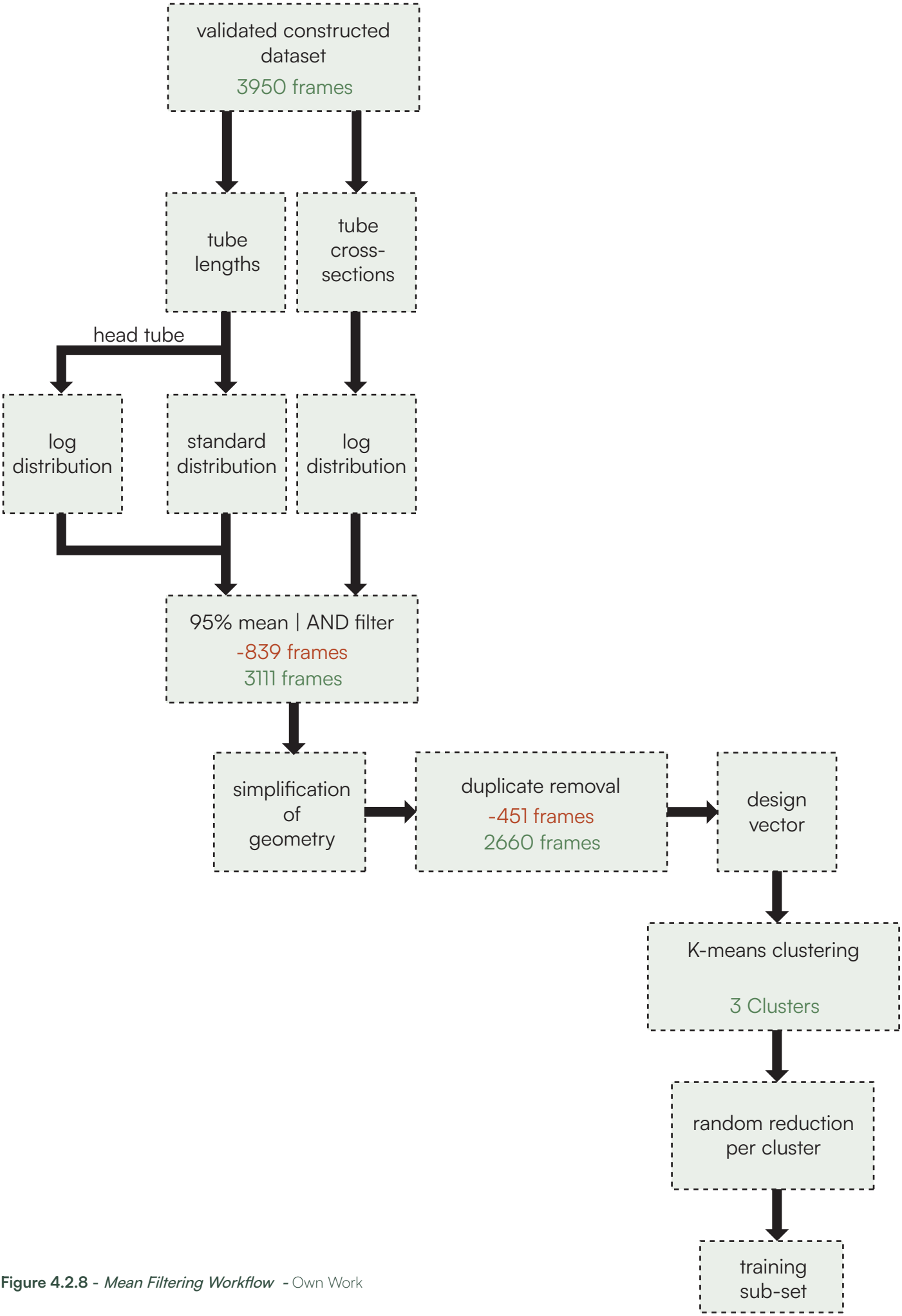


Figure 4.2.8 - *Mean Filtering Workflow* - Own Work

5.0 DESIGN CASE STUDY: CYCLING BRIDGE

Having established a generative principle as well as a coherent dataset to reflect the stock of bike frames, the final aspect to be addressed prior to formulating the generative algorithm itself is that of the design objective.

The practice of material-first generative design was outlined in the manifesto along several analytical perspectives. Relevant to his chapter is the perspective of Huang et al. (2021) who segment computational design practices for material reuse into two categories: Bottom-up and top-down. The material-first generative design practice outlined in the manifesto — and which is the arguable objective of this thesis — is a bottom-up approach.

Huang et al. (2021) argue that top-down practices are defined by a target design model, an intended objective around which the computational approach is formed. Bottom-up is instead defined by the aggregation of individual elements to build shape. Nevertheless, it would be difficult to argue that bottom-up approaches do not require a target design model outright. A design objective provides validation; it is prove of a potential application. It brings something entirely theoretical into a world which has consequences which merit consideration. It provides criteria which not only allow for the evaluation of the design principles, but which are required to turn an abstract concept into an actual object.

More pragmatically put: A design objective needs to be submitted to the algorithmic principles in order to both make a thing and evaluate it. This is the core of this chapter, to supply an actual design objective with a context drawn from real-world considerations. As the introduction already states, the chosen design objective of this thesis is to design a supporting structure for a cycling bridge. The selection of this objective is arbitrary in certain sense, as the generative principles aren't conceived to fulfil a specific design objective. As such, the cycling bridge idea was chosen not for the sake of the generative principles, but for requirements that such a design objective brings forward, providing an interesting and decently complex context to validate the core concept.

Chapter Objective

The goal of this chapter is to define all aspects required to apply the generative principles to the creation of a cycling bridge support structure and to define the conditions which need to be met for success. These aspects are divided into the dead load, live load and performance criteria. The theoretical framework is used to outline the selected case study, from which the basic dimensions are defined, the live loading requirements and material assumptions. The methodology explains how the non-supporting structures such as the deck of the bridge have been pre-optimized to reduce the dead load on the support structure. There is no results section, as this chapter functions predominantly as a theoretical section. Chapter 6 will outline the full generative algorithm, where the application of the theoretical basis described here will become apparent.

Relevant research questions:

How can the material-first generative principles be applied to a given design objective and which criteria need to be satisfied?

5.1 Theoretical Framework

The theoretical framework is segmented into the following sections: Firstly, a brief overview of design typologies pertaining to cycling bridges is provided through which several potential case studies are outlined. The second section consists of the structural aspects, divided into the loading of bridges by external forces and the criteria which exist to validate structural performance. Lastly, sources on material properties are provided in preparation of the method section.

Structural aspects of bridge design

As has been stated, this section will address all aspects which can be considered for the structural performance and evaluation of cycling bridges. As will become apparent in the methodology, not every metric will be considered for the sake of both scope and expediency, the latter pertaining specifically to the genetic validation and reinforcement leaning aspects of the thesis. The aspects are divided into the loads which could be enacted on the structure and the criteria with which to validate the performance of said structure.

The Dead Load

Two kinds of loads can be said to act on any given structural design: The dead load and the live load. The dead load, also known as the permanent load, refers to the mass of the structure. It is insufficient to only consider the weight of load bearing elements — aspects such as utility ducts, signage and handrails need also be considered. Dead loads typically also warrant consideration for permanent deformations, for example creep. However, these aspects are not included in the scope to reduce the computational load and simplify the design scenario.

The live load — vertical forces

The live load refers to every external force which is enacted on the structure, such as traffic, pedestrians or wind. These loads are defined in the Eurocode, specifically NEN-EN 1991-2, which deals with traffic loads on bridges and other civil engineering works. In section 3.1.1.8 a footbridge is defined as a bridge mainly intended to carry pedestrian and/or cycle-track loads, however, road traffic loads can be considered — for example to provide access to service vehicles or emergency services. Table 4.1 of the code goes on to provide three load models:

1. The uniformly distributed load q_{fk}
2. The concentrated load q_{fwk}
3. The service vehicle load q_{serv}

The uniformly distributed load q_{fk} is intended to simulate

crowd loading of the bridge. Section 7.3.2 provides a formula by which the load is calculated:

$$q_{\mathrm{fk}} = 2.0 + \frac{120}{L + 30}, \quad 2.5 \leq q_{\mathrm{fk}} \leq 5.0\,\mathrm{kN/m^2}$$

Wherein L is the length of the unfavourable part of influence surface in metre.

The concentrated load q_{fwk} is intended to simulate maintenance work, and should be taken as a 10kN force action on a square surface of 10 by 10 centimetre. It should not be considered if a service vehicle is specified.

The service vehicle load q_{serv} is intended to simulate the temporary presence of a service vehicle on the structure. The specific load can be dependent on the actual service vehicle, but if not specified, the load specified in 7.6.3 should be taken, which is intended to simulate the accidental presence of an unspecified vehicle on the structure. This load is defined as a two-axle group of 80kN and 40kN respectively. However, if the right precautions are taken in the design of the bridge the service vehicle load may also be discounted outright, provided that the bridge is inaccessible to vehicles outright.

The live load — horizontal forces

Beyond these three traffic load models, there is also a load model specified for horizontal forces in section 7.4. This force is applied on the pavement level in the direction of the bridge deck. The load should be taken as the greater of two values, given as either 10% of the total load corresponding to the uniformly distributed load or as 60% of the total weight of the service vehicle.

Beyond this, the grouping of loads under section 7.5 states that the horizontal forces should be considered simultaneously with the uniformly distributed load or the service vehicle load. The concentrated load is considered as an isolated action where no horizontal loads need to be considered.

Lastly, there is the consideration of accidental loads, of which there are two: Firstly the accidental vehicle load, already described for q_{serv}, and the secondly the collision of traffic passing underneath the bridge. The NEN states that footbridges are generally more sensitive to traffic collisions compared to roadbridges and recommends the use of a road restraint system and additional clearance compared to road bridges to avoid possible collisions.

ULS and SLS safety checks

NEN-EN 1990-1 specifies two limit states against which a given structure should be assessed. These are known as the serviceability limit state (SLS) and ultimate limit state (ULS). The SLS is intended to verify the integrity of the structure under the expected usage, more focused on the assurance of comfort and appearance. The ULS is intended to verify the structure under excessive forces, focused on the assurance of safety. Both limit states introduce partial safety factors (gamma), which are to be used on the loading models. These partial safety factors can be derived from table A.2.10 of the aforementioned NEN-EN 1990-1, and have been adapted in **Table 5.1.1**. To calculate the design value, the formula provided by 8.12 of NEN-EN 1990-1 is provided below:

$$\sum F_{\mathrm{d}} = \sum_i \gamma_{\mathrm{G},i} G_{\mathrm{k},i} + \gamma_{\mathrm{Q},1} Q_{\mathrm{k},1} + \sum_{j>1} \gamma_{\mathrm{Q},j} \psi_{0,j} Q_{\mathrm{k},j} + \left(\gamma_{\mathrm{P}} P_{\mathrm{k}}\right)$$

F_d represents the actual value of the design action

$\Upsilon_{g,i}$ is the partial factor of the permanent action i

$G_{k,i}$ is the characteristic value of permanent action i

$\Upsilon_{q,1}$ is the partial factor of leading variable action 1

$\Psi_{0,1}$ is the combination factor for leading variable action 1

$Q_{k,1}$ is the characteristic value of leading variable action 1

$Q_{k,j}$ is the characteristic value of accompanying action j

$\Psi_{0,j}$ is the combination factor for action j

$\Upsilon_{Q,j}$ is the partial factor of accompanying action j

P_k is the characteristic value of a prestressing force

Υ_p is the partial factor of a prestressing force

Action	SLS	ULS
Self-weight of the structure	1.0	1.50
Additional self-weight	1.0	1.50
Variable traffic loads	1.0	1.35

Table 5.1.1 - *ULS & SLS Factors* - From NEN-EN 1990-1:2025+A1:2026

Structural Criteria

There are three kinds of failure modes against which a given structure needs to be verified, these are known as:

1. Buckling, a sudden failure characterized by a sudden out of plane displacement in a member of the structure.

2. Excessive displacement, where the loading under the serviceability limit state exceeds a given maximum.

3. Structural collapse, where the materials enter the plastic stage of deformation under the ultimate limit state.

For the determination of the maximum displacement, a rule of thumb formula can be utilized:

$$\delta_{\max} \leq \frac{L}{400}$$

To check against structural collapse, it is required to assess the fiber stress imposed on the elements, checking that they do not exceed the tensile and compressive strengths of the material.

Buckling is not dominated by the strength of materials, but rather the stiffness. The intent is to check that the members do not suffer sudden lateral instability under the application of a compressive force before reaching the compressive strength — or yield strength — of the material. For slender columns, this can be checked using Euler's Buckling Formula:

$$N_{\mathrm{cr}} = \frac{\pi^2 EI}{L_{\mathrm{cr}}^2}$$

Previous Research

As it pertains to computational design of bridge structures, the master thesis of S. van Baalen (2017), Topology Optimized Pedestrian Bridge, proved most influential. Whilst their objective differed significantly from this thesis, their work greatly aided in the formalization of structural criteria and topologies of bridge design.

5.2 Methodology

The methodology section of this chapter consists of three parts. Firstly, the selection of the case study. Following that, the method of optimization by which the dead loads are determined. Lastly, a full overview of the design criteria and parameters.

Case Study Criteria

The final generative principles are limited to sequential placement of elements, as such, the potential truss configurations by which the bridge may be supported will all be slim. As such, it is decided to create arched trusses, to leverage geometrical shape for increased

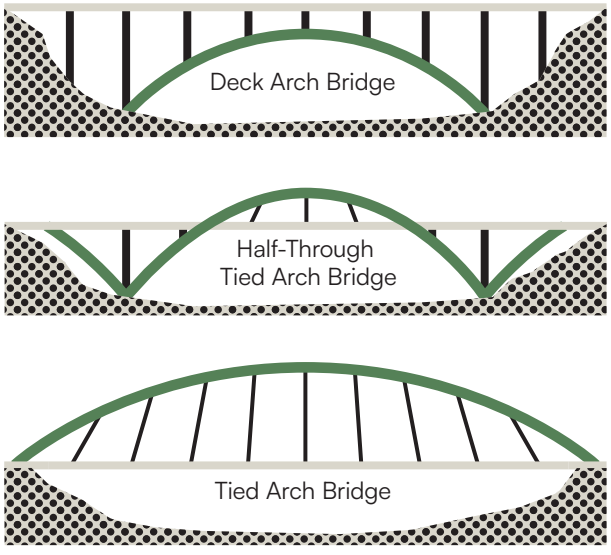


Figure 5.2.1 - *Different Arch Configurations* - Adapted from prior master thesis of S. van Baalen, 2017

There are three main typologies by which this can be accomplished, highlighted in **Figure 5.2.1**. The tied arch bridge is a kind of suspension bridge. Half-through tied arch bridges act as a kind of hybrid which reduces the required construction height.

The deck arch bridge, lastly, supports the bridge structure from underneath through compression. For this thesis, the decision was made to use a deck arch bridge. This configuration affords some flexibility, as it permits the use of multiple arches side by side without interfering with the deck structure above. Potential case studies are not inherently limited to deck arch bridges, as modern designs often achieve massive spans which are unrealistic for the scope of this thesis.

An important consideration which should be taken into account is the traffic flow underneath the bridge. If need be, the design might could be segmented into multiple smaller arches, requiring additional foundations, If this is the case, the traffic flow should not be inhibited.

Case Study Selection

Each bridge which has been taken into consideration will be outlined briefly, leading up to the final bridge which has been chosen as the actual case study object.

The pedestrian footbridges Arsenaal Delft.

This project, realized by Cepezed, consists of two bridges connecting ArsenaalDelft to the Oude Delft and Lange Geer. The bridges span 13 and 16 metres respectively, with a relatively steep incline to provide sufficient clearance for thoroughfare underneath. The designs are intended to be 'invisible', making use of sleek prefabricated elements, from materials also used in the fabrication of helicopters, a reference to the military past of the location. The bridges are supported via hidden helical piles along the water's edge, as to not touch the historical canal bank. Whilst the spans are potentially achievable, the narrow tolerances for thoroughfare access, combined with the expected increase in built height for the frame trusses makes this case study promising but complex.

Circular Footbridge Almere Haven

This footbridge is part of a circular design project 'Circulaire Brug'. Covering a span of 24 meters and a width of 1.5 meters, the bridge utilizes a light-weight wooden construction for the deck and supports, relying on hardwood which was been extracted through urban mining from the local area. However, the width of the bridge is likely not sufficient for bicycle traffic. S. van Baalen (2017) provides a coherent breakdown of the required width for cycling bridges to support two-way traffic, which comes down to 2,4 meters given that the bridge is not sloped. Furthermore, the bridge does cross an active waterway, though it is likely only used for small recreational boats. The design was realized by Arc2 Architects.

ZigZagBrug Arnhem

This bridge comes from the same 'Circulaire Brug' design project. As with the Circular Footbridge Almere Haven, the wood has been harvested through urban mining. The span of this bridge is shorter, standing at 16 metres total, however, the width stands at 2.5 metres, passing the minimal width requirement for two-way cycling traffic. As such, this bridge has been selected to be adapted for the case study.

Structural Criteria Specification

Having determined the case study - the Zigzag bridge - a selection is made from the structural criteria pertaining to pedestrian bridge design as highlighted in the theoretical framework. Because the bridge is not accessible by car and the waterway is not in use, the service vehicle load model and the accidental collision load model are not taken into account. Beyond this argumentation, these models also make the structural analysis much more complicated, as the horizontal forces would be difficult to incorporate into the 2D generative principles which have been established. In a similar vein, the wind forces have also been disregarded. Beyond this, the concentrated load q_{fwk} Is also discounted, as most previous research pertaining to reinforcement learning of structural assemblies do not incorporate such a load, which is further highlighted in chapter 8. This leaves the following considerations:

1. The self-weight of the structure

2. The uniformly distributed load model q_{fk}

The imposed vertical load is calculated via the formula for live traffic loading provided in the theory section. To determine the loading length L, the unfavorable part of the influence surface

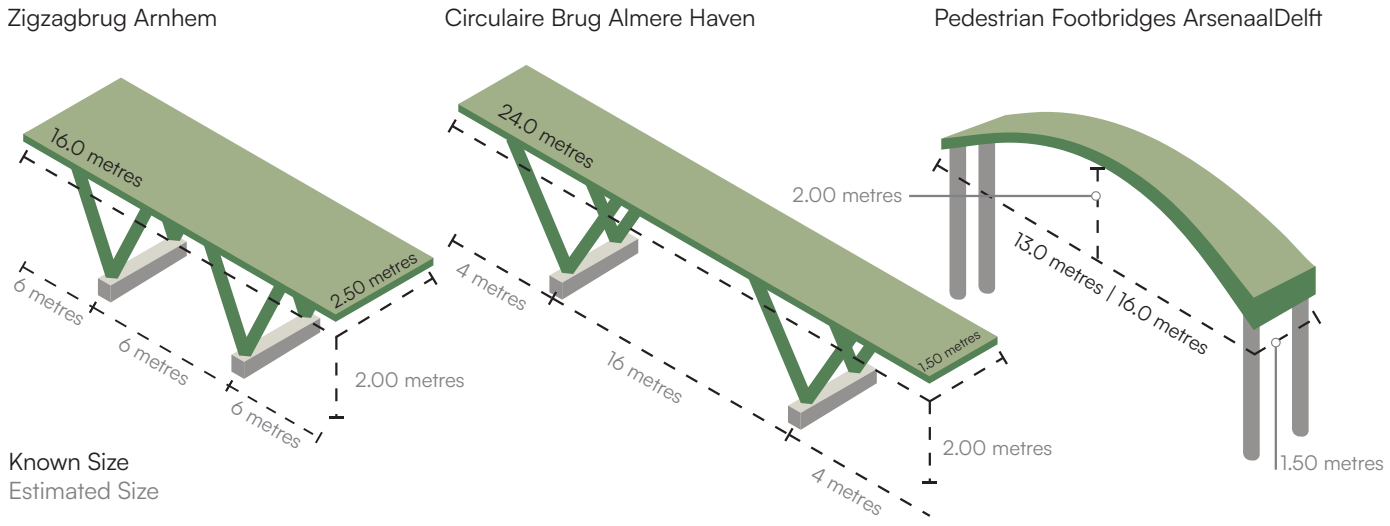


Figure 5.2.5 - *Dimensions Known and Assumed of Potential Case Studies* - Own work

must be determined. This consideration changes depending especially on the number of parallel arches. For an arch count above 2, one of the central beams should be considered, as these will carry the largest live load. If the arches run along the full length of the deck, a live load is yielded of 4.608 kN/m2. If instead multiple arches are used in sequence, then the number quickly exceeds the 5.0kN/m2 maximum. Therefore, the 5.0 kN/m2 figure has been assumed throughout the rest of the thesis. The live load is applied under ULS conditions, meaning that the load must be multiplied with the safety factor of 1.35, yielding 6.75 kN/m2 at the ULS criterium.

Self-weight Optimization

Based on the available data, the Zigzag bridge is known to consist of the following structural elements:

1. Hardwood materialization of the deck
2. Longitudinal beams underneath the deck of the bridge
3. A bridge deck consisting of slats fastened onto the longitudinal beams
4. A railing along both edges of the deck
5. Horizontal tension crosses

However, exact dimensions, outside of the span and the width, are not known. The spacing of the supports has been estimated via the number of railing elements, but this is not a confirmed dimension. In an effort to specify a set of possible sizes, a structural analysis has been performed, utilizing Grasshopper inside Rhino3D, in combination with the FEA software Karamba3D. To this end, the deck construction has been adjusted with the following assumptions: An additional central arch is needed to spread loads more evenly across the arches and the transfer of the deck load to the arch happens

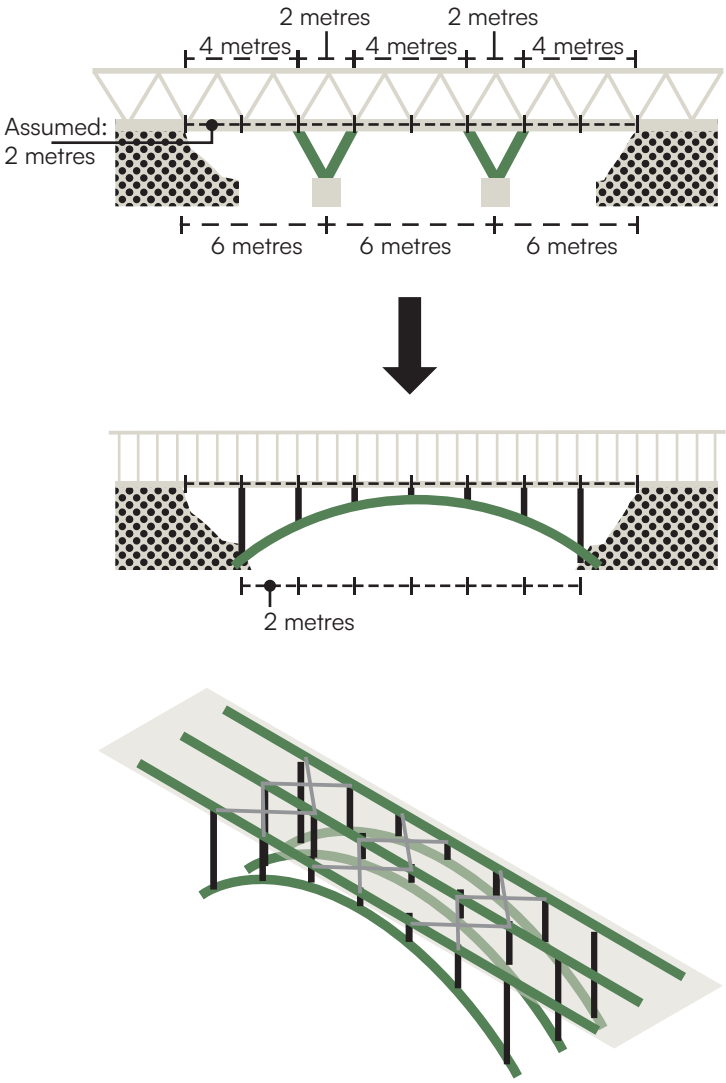


Figure 5.2.6 - *Translating Case Study to Design Principle* - Own work

Element	Material	Weight	Dimensions	Strength	Dead Load
Main Beams	C18 Hardwood	518.2 kg/m3	H: 200 mm W: 100 mm	1.80 kN/cm2 Tensile -1.80 kN/cm2 Compressive	0.17 kN/m
Slats			H: 50 mm | W: 75 mm Spacing: 5 mm		0.05 kN/m2
Railing Horizontals			70*70 mm		0.11 kN/m
Railing - Verticals			80*80 mm		
Staves	CHS 114.3x8	21 kg/m	OD: 114.3 mm Thickness: 8 mm	2.35 kN/cm2 Tensile -2.35 kN/cm2 Compressive	0.08 kN/m
Horizontal Bracing	CHS 88.9x4	8.38 kg/m	OD: 88.9 mm Thickness: 4 mm		0.02 kN/m

Table 5.2.7 - *Material qualities and dimensions of Karamba deck model* - Own work

via staves spread at 2 metre intervals. The railings have been simplified to slat railing. The live load was assumed at the ULS state, as 6.35 kN/m2, applied evenly to the slats of the deck.

Figure 5.2.6 illustrates the make-up of the adjusted configuration. **Table 5.2.7** contains the assumed material properties, which have been extracted from Karamba3D.

From this, an averaged load per square metres has been extracted, and rounded upward to a total of 0.50 kN/m2. The choice to calculate an average load per square metres is due to the fact that the longitudinal and parallel number of arches is yet unknown, and as such, the actual dimensions could be smaller or larger depending

5.3 Intermediate Conclusions

Action	SLS	ULS
Total Live Load	5.00 kN.m2	6.75 kN/m2
Total Dead Load	0.50 kN/m2	0.75 kN/m2

Table 5.2.8 - *Final Dead Load and Live Load* - Own work

This chapter set out to define the design objective to which the material-first generative principles are applied. A bottom-up approach still needs a target design, as it gives the criteria to build and evaluate a design. The support structure of a cycling bridge was chosen as that objective. The choice was not made to suit the generative principles, but for the real-world requirements it brings.

The theoretical framework covered the loads acting on a bridge and the criteria for checking its performance. The loads consist of the dead load and the live load, the latter following the load models in NEN-EN 1991-2. Performance is assessed through the ultimate and serviceability limit states defined in NEN-EN 1990:1, and against three potential failure modes: buckling, excessive displacement and structural collapse.

The methodology then applied this framework to a concrete case. A deck arch bridge was chosen as the typology, as it suits the sequential placement of elements. Of the three candidate bridges reviewed, the ZigZagBrug in Arnhem was selected, with a span of 16 metres and a width of 2.5 metres. The service vehicle, accidental, horizontal, wind and concentrated loads were excluded, leaving the self-weight and the uniformly distributed load. The live load is taken as 5.0 kN/m², or 6.75 kN/m² at ULS. The deck structure was pre-optimised in Karamba3D, giving a dead load of 0.50 kN/m², 0.75 kN/m2 at ULS.

Together, these define how the generative principles are applied to the design objective and which criteria must be met. Chapter 6 shows how the principles are put into practice.

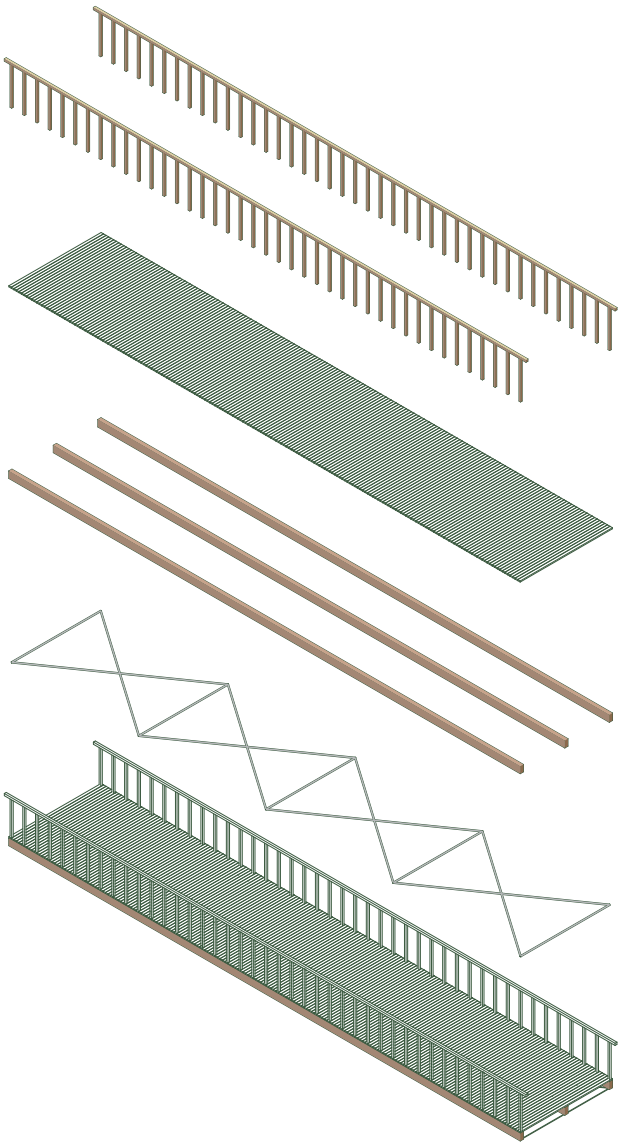


Figure 5.2.9 - *Axonometry from Karamba3D model* - Own work

6.0 MATERIAL FIRST GENERATIVE ALGORITHM: OVERVIEW

The previous three chapters served to build the foundations on which the complete design algorithm can be developed: Through the manifesto, a generative principle has been determined free of any target design model. The material case study has been used to then close the gap between an initial estimated dataset and a final, rigorous dataset based on the FRAMED database of Regenwetter et al. (2022). The design case study has provided both context and criteria for verification.

Chapter Objective

The objective of this chapter is to translate the generative principles to an actual design algorithm by applying said principles to a design context. This chapter is a methodology only, as the required theoretical background has already been discussed in the chapters preceding it.

Relevant Research Question

How is the material-first generative algorithm constructed from the generative principles, dataset and design objective?

6.1 Methodology

The methodology section will be divided into the following subsections: Firstly, the application of the generative principles to the design context. Following that, the processing required to take the designed output to a fully-fledged structural object. Lastly, the performance requirements are restated briefly, as these have already been discussed in chapter 5, but should still be included to provide a complete overview.

Regardless of implementation, reinforcement learning or genetic optimization, the bounding space is determined in the same way. This space is defined two-dimensionally using an x-axis and a z-axis. Objects are always positioned in the fully positive quadrant.

Main Functionality

The generative principles, free standing of any target design, were previously illustrated without the consideration of stock variance. The shape grammar, consisting of the shapes, labels, rules and initial shape was outlined as shown in **Figure 6.1.1**. Rules consist of three elements: Target orientation, candidate orientation and mirroring. Target refers to the immediate previous frame in the sequence, whilst candidate refers to the new frame added at the given step.

Three subsets of bicycle frame geometries have been created using the extracted dataset described in chapter 4, consisting of 25, 35 and 50 frames respectively. For the purposes of the algorithm overview, the assumption is made that the 35-frame set is used. This introduces a fourth element to the rule definition: Frame selection.

The Design Gesture

Applying the generative principle into the design case study introduces a set of new considerations, first of which is the design gesture. The design gesture effectively introduces an a target design model, though it is a very abstract one: A simple one-directional curve. The shape of this curve is dictated by the structural parameters described in chapter 5, being the

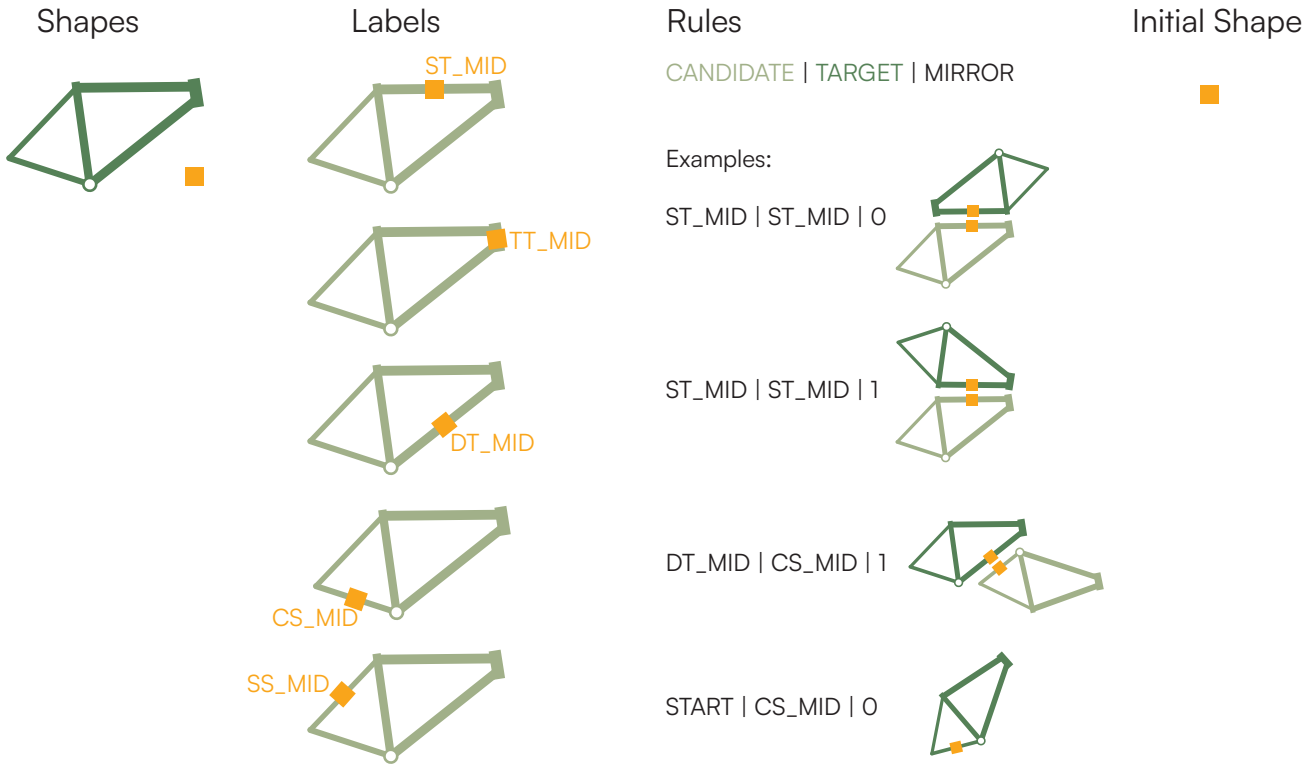


Figure 6.1.1 - *Generative Principle as Defined in the Manifesto* - Own work

longitudinal and perpendicular arch division, see **Figure 6.1.2**. For the main functionality only the longitudinal count is required.

Incorporating the variance of the stock and the design gesture into the generative principles yields the main algorithmic functionality, outlined in **Figure 6.1.3**.

Valid Placement

The stock variance introduces a new concern, consumption. Stock elements which have been placed in previous steps should no longer be usable in later ones. Whether or not the action space allows for the selection of previously consumed frames, the placement is denied outright, resulting in the step terminating prior to the frame actually being committed, moving on to the next. Beyond this, intersections are also guarded against, meaning that the placed frame cannot coincide with any of the points of the previous frame, nor can the lines of either frame intersect. Intersections are always possible through the action space - however, the placing of an intersecting frame is denied outright, resulting likewise in the termination of the step without committing the frame. **Figure 6.1.4** contains a flowchart outlining the main functionality's logic.

Initial Placement

There is one alternate situation not yet adressed, which is the placement of the first frame. For this, five possible target orientations are generated through the followings steps:

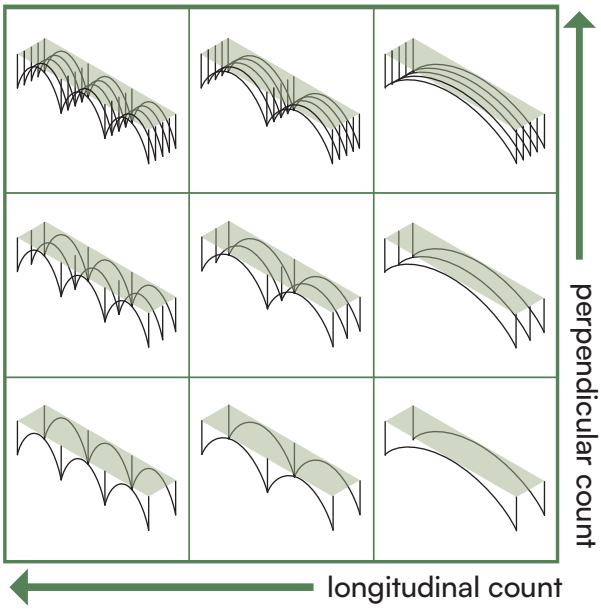


Figure 6.1.2 - *Arch count variations* - Own work

0. The selected frame is temporarily placed onto the design gesture, aligned as in figure 6.1.5 and drawing a line.

1. The first orientation plane is created perpendicular to this line.

2. & 3. The first plane is rotated 30 degrees in either direction

4. & 5. The chain stay and seat stay middles are added as the final two potential planes

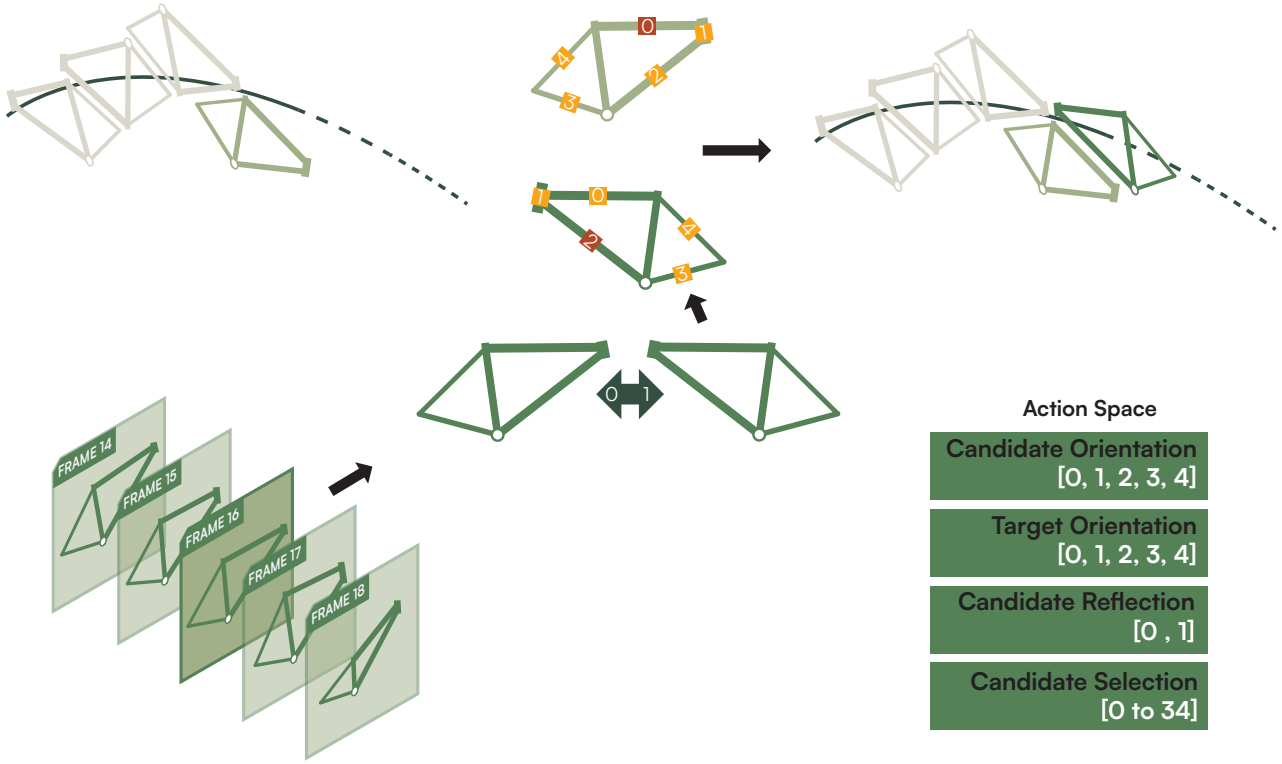


Figure 6.1.3 - *General Concept Algorithm* - Own work

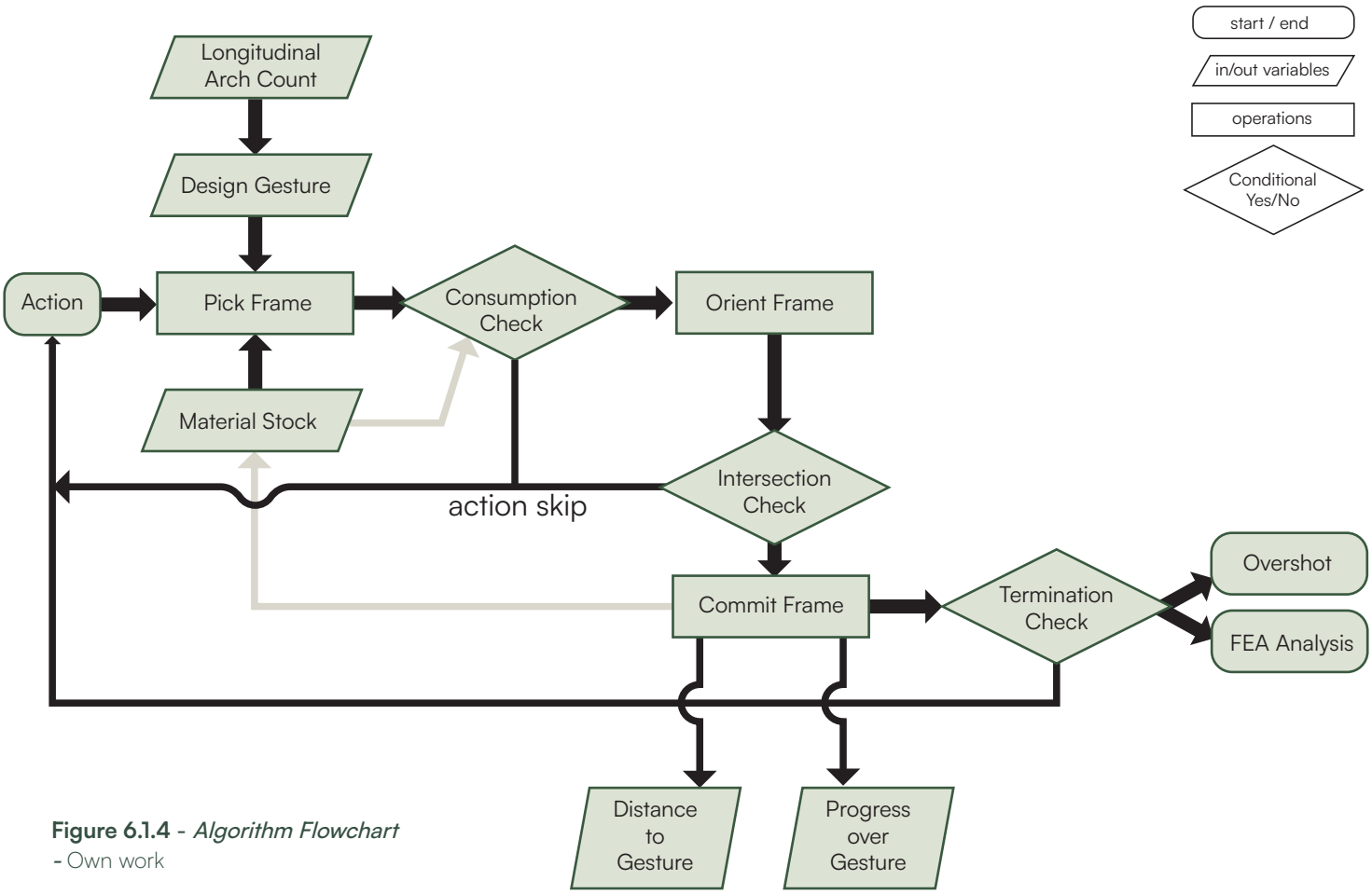


Figure 6.1.4 - *Algorithm Flowchart* - Own work

Metrics

The incorporation of the design gesture introduces certain metrics — which are utilized to shape optimization goals and reward structures in the later chapters. Two metrics have formulated: The distance metric and the progress metric. These metrics are calculated on each step in the algorithm.

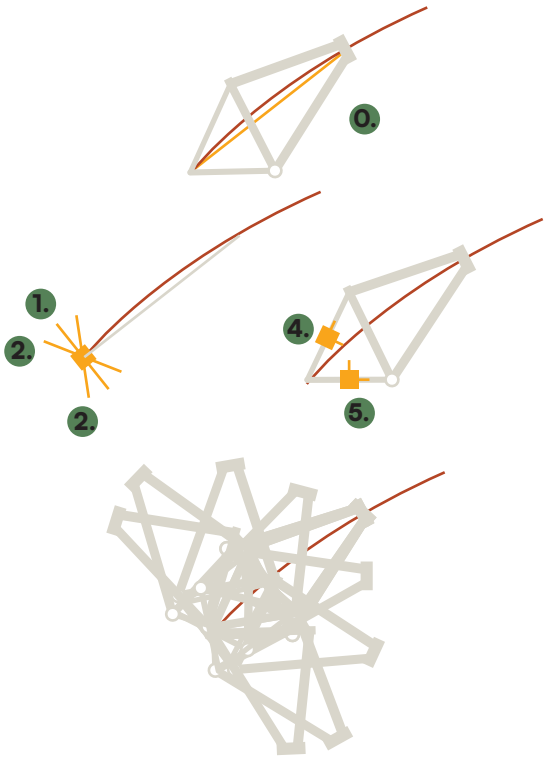


Figure 6.1.5 - *Initial Placements* - Own work

Centroid Determination

Both metrics require the calculation of the centroid c_s at step s of the placed frame, which is achieved via a mean calculation over the coordinates of each of the frame's edge points $P_{s,n}$ expressed as coordinates in an xz-plane.

$$\mathbf{c}_s = \frac{1}{5}\sum_{n=1}^{5}\mathbf{P}_{s,n}, \quad \mathbf{P}_{s,n} = \left(x_{s,n}, z_{s,n}\right)$$

Distance Metric

Following this, the centroid c_s is projected onto the design gesture curve. For the distance metric d_s, the Euclidian distance between centroid and each curve sample point g_i is calculated, yielding the closest point dg_s.

$$\mathbf{dg}_s = \underset{i\in\{0,\ldots,N\}}{\arg\min}\|\mathbf{g}_i - \mathbf{c}_s\|$$

$$d_s = \sqrt{\left(x_{\mathrm{dg},s} - x_{\mathrm{c},s}\right)^2 + \left(z_{\mathrm{dg},s} - z_{\mathrm{c},s}\right)^2}$$

Progress Metric

The progress metric calculates how far along the design gesture a given frame is. It does so by taking the index i of closest point dgs to produce a parameter value t between 0 and 1. The progress metric only returns a new value if the current value exceeds the previous value - it only reports true progress.

$$t_s = \max\left(t_{s-1},\ \tfrac{i}{N}\right)$$

$$\mathbf{dg}_s = \mathbf{g}_i, \quad t_0 = 0, \quad t_s \in [0,1]$$

Termination

The design gesture introduces a new placement check, the termination check. This check also relies on the utilization of the centroid of the placed frame and calculates the distance between it and a known termination vicinity parameter d_{term}. If the distance between the frame's centroid c_s and the end point of the curve P_{end} is determined to be smaller than d_{term}, the loop terminates, yielding V_s. However, there are two possible alternative termination states. The first alternative is known as the bounding overshot — in this case, the frame is outside of the termination vicinity, but has exceeded the bounding space of the curve, defined using a prependicular projection at the end of the curvature τ_{end}, yielding O_s. The second alternative state is the termination overshot, which means that the centroid is inside the d_{term} but has overshot τ_{end} still, yielding E_s

$$V_s \Leftrightarrow \|\mathbf{c}_s - \mathbf{p}_{\text{end}}\| < d_{\text{term}}$$

$$O_s \Leftrightarrow \left(\mathbf{c}_s - \mathbf{p}_{\text{end}}\right) \cdot \hat{\tau}_{\text{end}} > 0$$

$$\hat{\tau}_{\text{end}} = \frac{\mathbf{p}_{N-1} - \mathbf{p}_{N-2}}{\|\mathbf{p}_{N-1} - \mathbf{p}_{N-2}\|}$$

$$T_s \Leftrightarrow V_s \vee O_s$$

$$C_s \Leftrightarrow V_s$$

$$E_s \Leftrightarrow O_s \wedge \neg V_s$$

The last termination state E_s is an optional condition, intended to be used in design scenarios which require narrow tolerances. For the purposes of this thesis, this state has generally been left disabled.

Structural Post-Processing

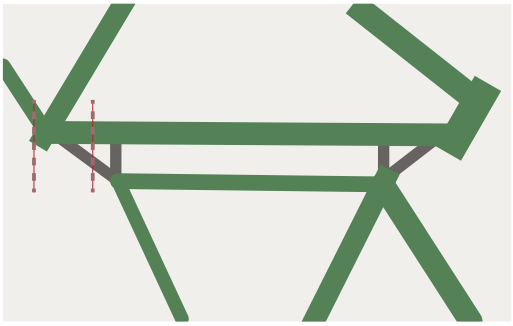
The main functionality encompasses the interaction between the entity, be it designer or optimization algorithm — and the design algorithm at large. The functionalities described below are not part of the interaction, meaning that these aspects occur autonomously upon achieving true termination. These aspects ensure that the structure is 'complete' from a structural design perspective. The following aspects are discussed to this end: The placement of connections, the determination of supports, the determination of loads and the placement of tension cables.

The Connections

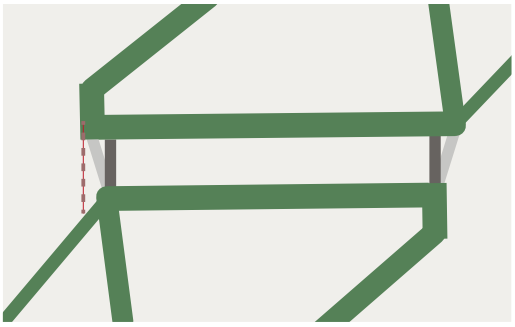
The connections represent a complicated practical challenge, one which has been largely omitted from the research to keep the scope manageable. As such, the practical manner of connecting is not determined. Instead, the connections are simplified to estimate the flow of the forces through the structure. The distance between frames is determined by a variable known as the connection offset, which is the heartline distance, not accounting for the tube diameters. There's two potential connections which can be made between two elements, the type of connection is ruled by two variables: The connection distance limit and the minimum connection distance. Connection type 1, as shown in **Figure 6.1.6** is generated if the distance between the edge points is within the connection distance limit, and if the distance between the longer side's edge points to the projected points is larger than the minimum connection distance. The reason for using this double connection is to minimize the presence of forces perpendicular to either tube. If the minimum connection distance is not exceeded, it means that the tubes are of approximate equal length, resulting in connection type 2. If however the connection distance limit is exceeded, the resulting connection, type 3, is likely problematic. For the materiality of the connections, the choice was made for a hollow tube section and steel, the specific measurements and material characteristics are provided in **Table 6.1.8**.

The Supports

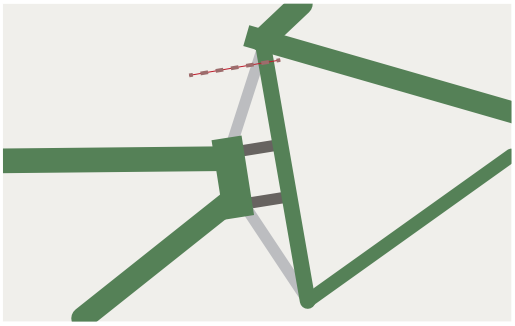
Two points are extracted from the structure and classified as supports. The first point, extracted from the first frame placed, is classified as a pin. The second point, extracted from the last placed frame, is classified as a roller. This is done to ensure that the tension cables, discussed further down the line, are always in tension under vertical loading. The lowest point on a given frame is used for the support.



> minimum connection distance
< connection distance limit



< minimum connection distance



> connection distance limit

Figure 6.1.6 - *Connection Logic* - Own work

The Loading Points

The structure is vertically loaded for the structural performance. This load is predominantly determined by the perpendicular arch count, as this defines the tributary width. From this, a line load is determined. After this, the points which will be loaded need to be determined. This is dependent on a two variables: The load division and the load indices. The prior determines how many points will be loaded, the latter determines which of the 5 points of the frames are allowed to be loaded in the first place. Only the first and third point, being the points which make up the seat tube, are used for the indices — unless otherwise stated. The load division is set to 5 in every instance.

To determine which points are loaded, the available points are checked for intersections with elements above them via a vertical ray. Having filtered out the points which do not intersect, the best spaced remaining points are selected for loading, see Figure 6.1.7.

Tension Cable Placement

The logics outlined above complete the structure, and permit for the application of structural analysis. However, early investigations highlighted the need for an additional element to aid in the transfer of the vertical forces. For this, the decision was made to place tension cables along the bottom of the structure. As stated before, this logic operates autonomously. The logic is ruled by the tension count, tension threshold and tension both ways parameters. The functionality is illustrated in Figure 6.1.7. First, the points on each frame below the curvature are isolated. Starting from the first point, a set of lines is drawn. The longest non-intersecting line wins, and from the end point of said line, a new set of lines is drawn. The number of lines drawn is determined by the tension count. The threshold filters out lines which are too close or too far. If tension both ways is enabled, the logic repeats a second time, now moving in reverse point order, to add more tension cabled along the bottom. In this case, cables which coincide with those placed in the first direction are discounted.

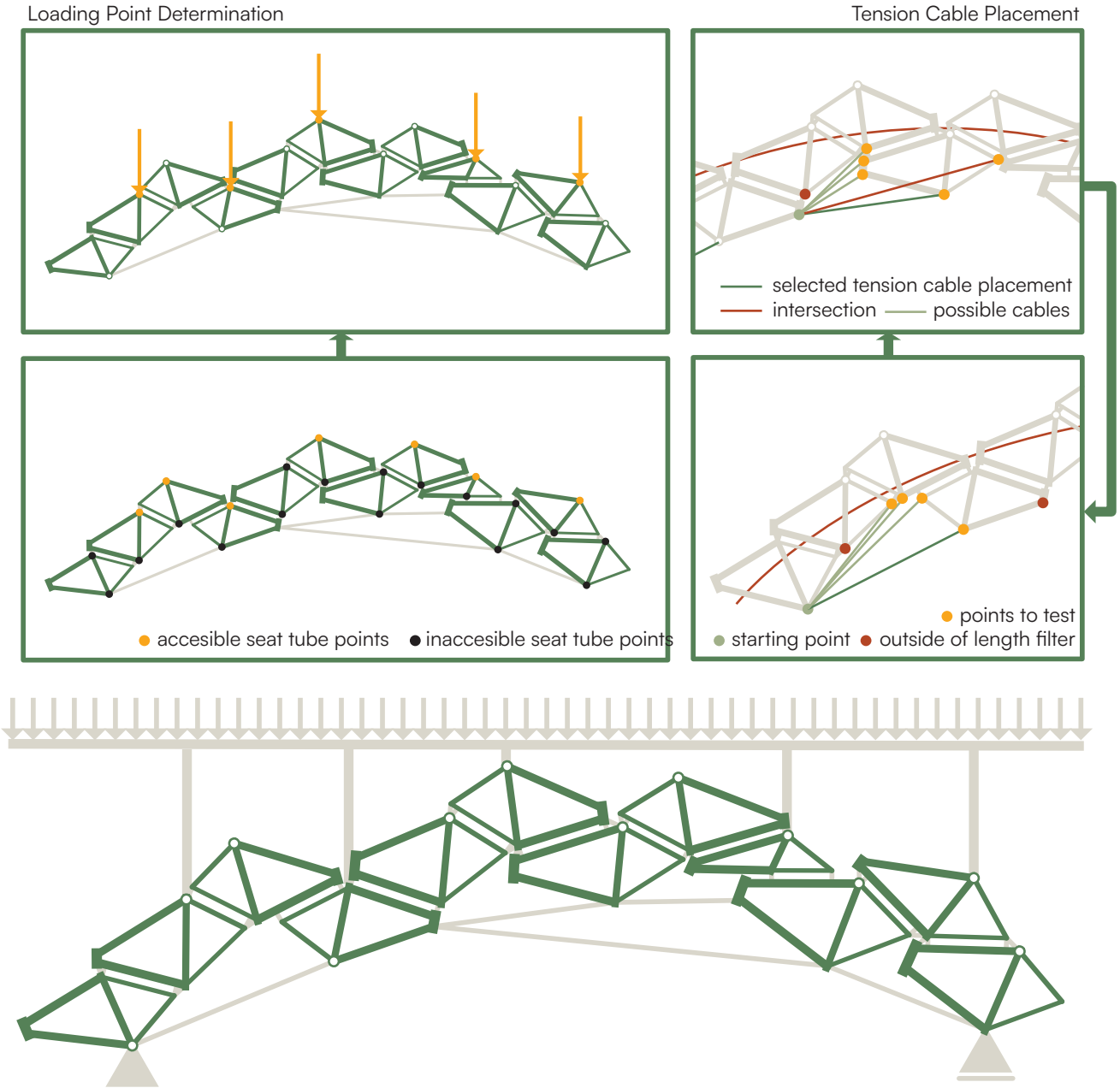


Figure 6.1.7 - *Loading Point Determination and Tension Cable Placement* - Own work

Materialization

For the materialization of the different elements of the design algorithm, Table 6.1.8 has been provided. For the materials of the bike frames, the assumption made for AISI 4130 steel - mirroring the approach of Regenwetter et al. (2022). The material properties have been attained via Granta Edupack. For the tension cables, properties have been extracted from the Ronstan Steel Architecture. The compressive strength for the tension cables is given as 0. For the connections, the properties of S355 steel have been assumed from Karamba3D.

Performance Metrics

Two sets of performance metrics have been created for the optimization and structural verification. For the optimization, the partial FEA scoring is used, reflecting the criteria for which both genetic algorithm and reinforcement learning will actually optimize. The full FEA score is intended to check the generated designs in a more extensive manner. The metrics have been included in **Figure 6.1.9**.

Scripting Setup

Over the course of the thesis, the codebase for the reinforcement learning routine saw a major overhaul, and this was consequently translated back to the genetic validation as well. The core of the algorithm is contained in the utilities and class scripts. The different variables for design are contained inside the internal variables file. Both the reinforcement learning and genetic algorithm scripts make use of the core codebase, with some additional functionalities for either. **Figure 6.1.10** illustrates the setup.

Metric	Condition	Partial FEA	Full FEA
Maximum Deformation	≤ 37.50 mm | 1 arch	Y	Y
	≤ 17.50 mm | 2 arches	Y	Y
	≤ 10.83 mm | 3 arches	Y	Y
Maximum Frame Tensile Stress	≤ 62.10 kN/cm^2	Y	Y
Maximum Frame Compressive Stress	≤ 48.30 kN/cm^2	Y	Y
Maximum Connection Tensile Stress	≤ 35.50 kN/cm^2	N	Y
Maximum Connection Compressive Stress	≤ 35.50 kN/cm^2	N	Y
Maximum Cable Tensile Stress	≤ 13.65 kN/cm^2	N	Y
Maximum Cable Compressive Stress	≤ 0.0 kN/cm^2	N	Y

Table 6.1.9 - *Partial and Complete FEA Check Conditions* - Own work

Element	Sizing	E-Modulus	Shear Modulus	Tensile Strength	Compressive Strength
Bike frames	Variable	210000 N/mm2	8076.00 kN/cm2	62.10 kN/cm2	48.30 kN/cm2
Connections	30.00 mm outer diameter, 5 mm thickness	210000 N/mm2	8076.00 kN/cm2	35.50 kN/cm2	35.50 kN/cm2
Tension cables	18.00 mm diameter	127000 N/mm2	4884.00 kN/cm2	13.65 kN/cm2	0.00 kN/cm2

Table 6.1.8 - *Properties for the Different Elements* - Own work

GA core contains the environment for the genetic optimization, which is then imported into a calling script inside a Grasshopper file which makes use of Karamba3D for the structural evaluation. Baseline GA utilities contains a few additional functionalities needed for the genetic routine.

BikeBuilder env contains the scripts for the RL environment, which are fed through the training script, policy and extractor. The training diary file was created to ensure that all internal variables, environment keyword arguments and policy keyword arguments are recorded upon training completion. FEA was achieved using OpenSeesPy, and the performance was found to be comparable to the Karamba model of the genetic modules.

Preliminary Conclusions

This chapter set out to translate the generative principles into a design algorithm by applying them to the design context. It drew on the generative principle from the manifesto, the dataset from the material case study and the criteria from the design case study.

The main functionality extends the shape grammar with the variance of the stock. Each rule consists of target orientation, candidate orientation, mirroring and, with the introduction of the 35-frame dataset, frame selection. The design gesture introduces an abstract target design in the form of a one-directional curve. Placement is guarded by two checks: consumed frames cannot be reused, and intersecting frames are denied. In both cases the step ends without committing the frame. A separate procedure generates the five possible target orientations for the first frame. On each step, the centroid of the placed frame is used to calculate the distance and progress metrics, which shape the optimisation goals and rewards in later chapters. The same centroid determines termination, with true termination occurring within the termination vicinity and two overshoot states as alternatives.

Upon true termination, structural post-processing completes the structure autonomously. Connections are simplified to estimate the flow of forces, with three connection types governed by the connection distance limit and the minimum connection distance. A pin is placed at the lowest point of the first frame and a roller at the lowest point of the last frame. Five loading points are selected from the seat tube points that are not obstructed from above. Tension cables are placed along the bottom of the structure to aid the transfer of vertical forces. The frames are materialised as AISI 4130 steel, the connections as S355 steel, and the cables based on the Ronstan Steel Architecture data. Partial and full FEA scores are defined for optimisation and verification respectively.

Together, these describe how the material-first generative algorithm is constructed from the generative principles, dataset and design objective. This answers the chapter's research question.

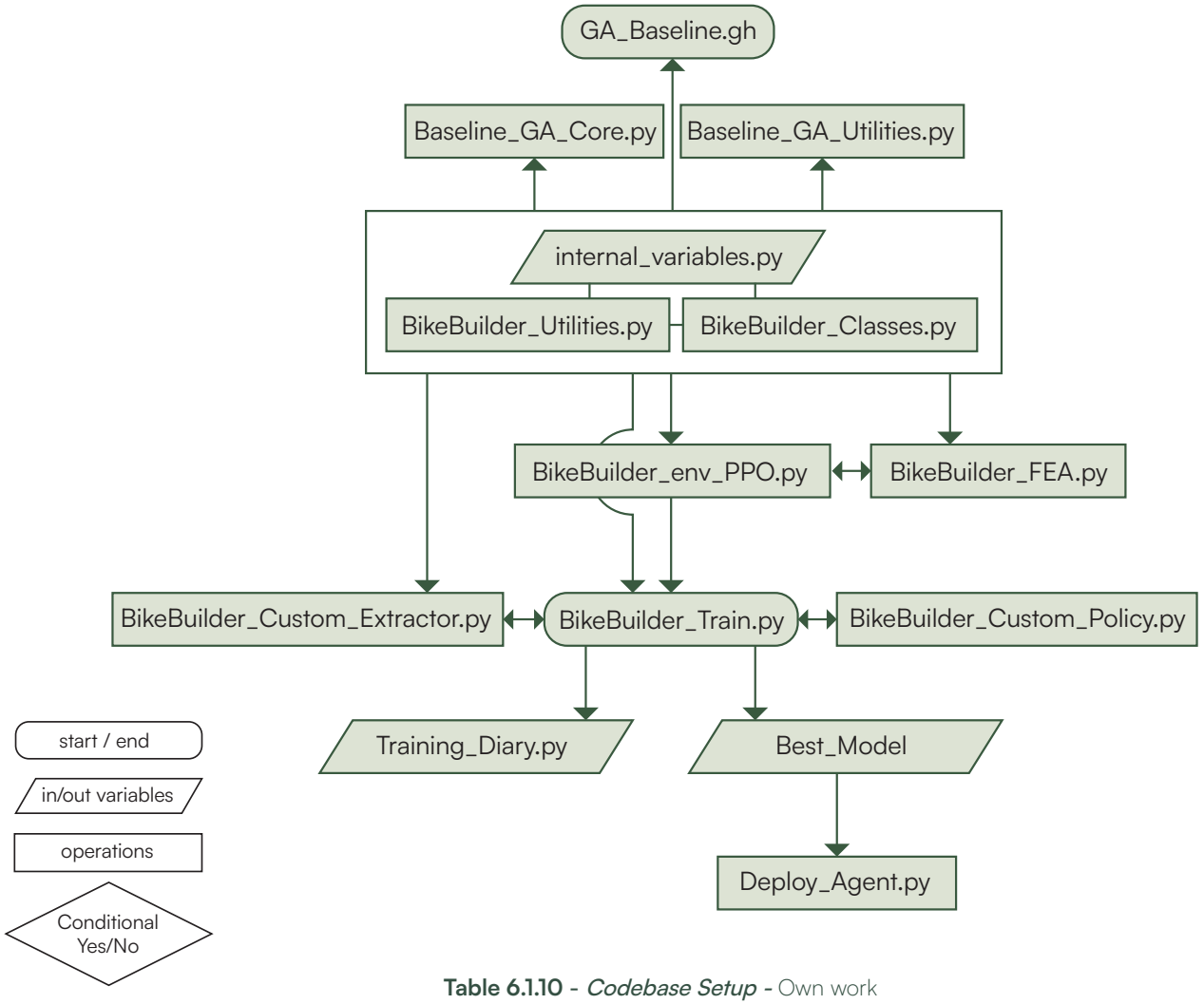


Table 6.1.10 - *Codebase Setup* - Own work

7.0 GENETIC VALIDATION

Having outlined the design algorithm in full, one critical step remains prior to the training of a reinforcement learning agent, which is to validate that the design algorithm can conceivably converge with a more rudimentary optimization algorithm. For this, a genetic algorithm has been selected, due to the common usage of said algorithms in contemporary computational design practices. This chapter will consist of a theoretical framework, methodological approach and results.

Objective

The main objective of this chapter is to validate the design algorithm. Doing so requires several steps. Firstly, it has to be assessed if the algorithm is capable of convergence regardless of structural performance — meaning that the genetic algorithm is capable of reaching the termination condition described in the previous chapter. When this is accomplished, the next step is to assess if the algorithm can optimize the structural performance, however: The structural variables defined in chapter 5, being the arch count in the longitudinal and perpendicular directions, are as of yet not specified. As such, a multi-objective optimization is performed to derive at the right variable specification. After this, the final optimization routine is performed which will in turn provide a performance baseline against which the reinforcement learning deployment might be tested.

Relevant Research Question

Can a genetic optimizer successfully utilize the material-first generative algorithm, therefore validating the functionality of the underlying principles given the provided material and design context?

7.1 Theoretical Framework

Genetic algorithms (GA) were first conceived of by John Holland in the 60's, first publishing his findings in his 1975 book Adaptation of Natural and Artificial Systems (Holland, 1975). As the name implies, the principles of genetic algorithm are based in biology — a not uncommon trope in machine learning at large, of which genetic algorithms are arguably a subset. The method revolves around a population of possible solutions which are scored via a type of reward signal v. These solutions are generated by a set of parameters and accompanying logics. The algorithm begins with stochastically selecting values for each parameter, and through a cycle of updates it progressively finds narrower rangers for each parameter which yield improved reward signals. The parameters in essence represent an action space, but in the context of GAs are often referred to as the search space, whilst the parameters are referred to as 'genes'. The reward signal is often called the 'fitness value'.

The inherent value of GAs is evident from the fact that the algorithms can find optimal solutions in vast and multi-dimensional action spaces, and is capable of doing so through evolutionary principles. For this, GAs commonly make use of two operators which dictate the sampling of parameter settings over the course of optimization. The first is the crossover operator which samples parameter settings, described by Holland (1975) as 'alleles'. It combines these alleles to yield new solutions. The second is the mutation operator. In essence, this operator introduces a degree of randomization within a sampled allele. Holland (1975) denotes that this operation is often misrepresented, seen as a vehicle through which new solutions are generated. Instead, the role of the mutation operator is to ensure that the crossover operator has access to a wider range of parameter samples, alleles.

The crossover and mutation operators compliment each other, the goal being to balance these values so that the solver does not get stuck in local optima. This is a significant concern with genetic optimization. The process starts with an immense search space and through initially stochastic sampling attempts to find the optimal settings of the parameters which make up this search space — but the process cannot guarantee that the settings identified by the algorithm are actually the best possible ones, and furthermore, as the algorithm naturally selects smaller and smaller ranges, it cannot be known if it has taken the right direction.

Beyond local optima, generative design introduces a second concern, known as epistatic variance, first introduced by Davidor (1991). Genetic optimization revolves around crossovers, where different parameters of different solutions, i.e. different alleles from different genes, are combined to form new solutions — genes. This works well if each allele is said to have an independent contribution to the final gene. This rings true for most parametric design procedures, where parameters are said to symbolically represent real design values, such as the height or thickness of a wall. However, generative and algorithmic design processes have a degree non-traceability (Caetano et al., 2020), parameters combine in an iterative sequence, and the role of parameter(t) at step t is to a degree defines by the impact of parameter(t-1) and each parameter that preceded before that one. Parameters are therefore said to be interdependent. This makes it difficult for genetic algorithms to navigate these types of processes, because the role of a single allele is not consistent in each solution.

It is not impossible to use genetic optimization for epistatic design problems, as Mirra & Mueller (2025) demonstrates: Before committing to a reinforcement learning application to their design problem, the authors first prototype the environment via a multi-objective optimization using 10 variable 3-tuples, and demonstrate through this that the design problem is solvable.

7.2 Methodology

The genetic validation has been performed using a combination of Python scripts, Grasshopper, Karamba3D for the finite element analysis and Wallacei for the genetic optimization. The first versions of the design algorithm were implemented with genetic optimization in mind. However, once the reinforcement learning codebase had been developed, a new variant was made, using the same internal variables, utility functions and data classes as the reinforcement learning agent uses. Chapter 6 contains a top-level overview of the codebase to illustrate this visually, see **Figure 6.1.10** on page 55. This was done to ensure that the genetic optimization routine and reinforcement learning routine are as closely aligned as possible.

The methodology for the genetic validation is reliant on three different steps. However, the approach through which the genetic validation is performed is mostly consistent with each step. This approach has been visualized in **Figure 7.1.1**. The key difference in each approach is the consideration of the action space.

Full Action Space Variant

The full action space variant has been indicated in **Figure 7.1.1** with teal. This variant of the genetic optimization is most closely aligned with the reinforcement learning algorithm, reliant on the same action space. As such, the action space is directly represented through four gene pools. The length of each gene pool reflects the maximum number of allowed steps, 35. This yields a total action space of 3.208 e^113. Critically, the fitness function is different from the other optimization, as the intent was never extended beyond achieving termination. As such, the fitness function consists of four elements:

The first two elements are drawn from the progress and distance metrics outlined in chapter 6, reiterated here: The distance metric is taken from the centroid of a given frame to the closest point on the design gesture. The progress of a given frame is calculated by attaining parameter t_s on the design gesture and comparing it to the previous placement, t_{s-1}.

Fitness Function for Distance

Each frame's distance metric dg_s is normalized through a prescribed bounding range [0,500]. onto a normalized range [0,1]. The sum of these normalized values $\mathcal{F}_{d,s}$ represents the total fitness score $\mathcal{F}_d$ for distance. Wallacei is only capable of minimization, which works well for this metric.

$$\mathcal{F}_d = \sum_{s=1}^{S} \mathcal{F}_{d,s}, \qquad \mathcal{F}_{d,s} = \frac{\text{clip}(d_s, 0, 500)}{500}$$

Fitness Function for Progress

The progress metric t_s is utilized firstly to identify whether or not the placement of a frame has yielded net positive advancement along the curve, and is a value between [0,1]. For this fitness function, the distance is calculated between the centroid of the current frame and the one preceding it. This value is normalized over a range of [0,1500] - this range has been calculated from a combinatory calculation to find the furtherst progression possible given the stock of items.

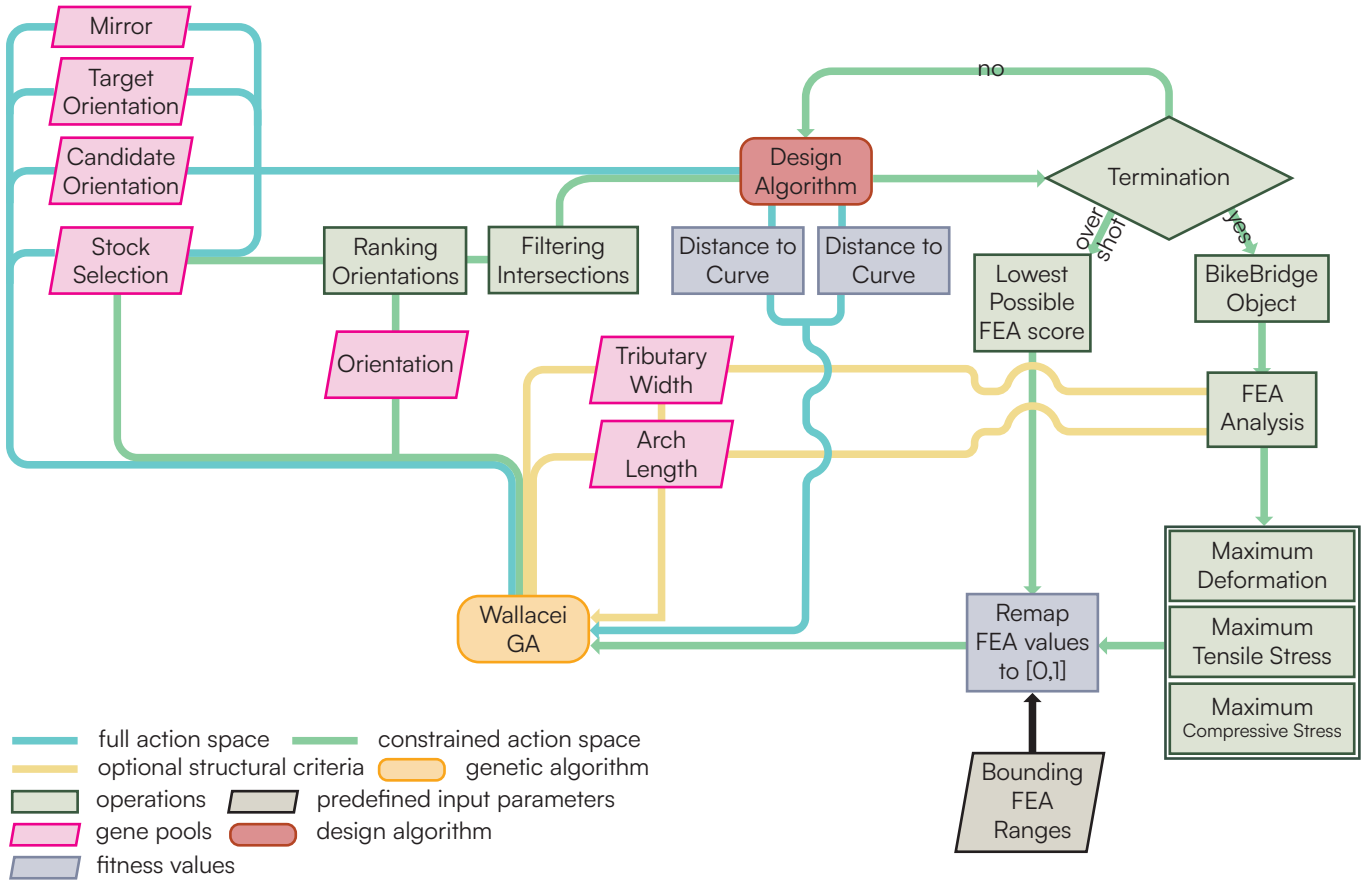


Figure 7.2.1 - *Connection Logic* - Own work

Critically, this is a value to be maximized, which Wallacei does not support, therefore the values are remapped to an inverted range so that 1500 approaches 0., yeilding $\mathcal{F}_{\mathrm{p,s}}$, The sum of these values is the complete progress fitness score $\mathcal{F}_{\mathrm{p}}$.

$$\mathscr{F}_p = \sum_{s=1}^{S} \mathscr{F}_{p,s}, \quad \mathscr{F}_{p,s} = 1 - \frac{\mathrm{clip}(\|\mathbf{dg}_s - \mathbf{dg}_{s-1}\|, 0, 1500)}{1500}$$

Fitness Function for Intended Behaviour

The latter fitness functions are designed to deal with the specific ways in which the genetic algorithm has shown to diverge from the intended behaviour. The third function is the placed frames function $\mathcal{F}_{\mathrm{ps}}$. It is a flat integer value that reflects the number of frames which have been placed correctly, it is not normalized but it is inverted, so that higher numbers of placed frames yield a score closer to 0. The reason that this value is not normalized, is that the previous two metrics are summed, and this metric is not. As such, the balance between the metrics is better preserved without normalization.

$$\mathscr{F}_{\mathrm{pf}} = \text{maximum step} - S$$

Fitness Functions for Termination

The last fitness metric is the termination metric. This metric is an important one, as it can drastically change behaviour. There are two possible termination conditions: Either the last placed is within the termination threshold, calculated via Euclidean distance, or the last placed frame has exceeded the design gesture's bounding space, overshooting. Three different fitness functions have been tested to address this possible behaviour. Firstly, a flat score of 30 is given if the agent fails to terminate or overshoots. This score is reduced to 0 if the genetic algorithm achieves true termination, meaning it is within the termination threshold.

$$\mathscr{F}_{\mathrm{term}} = \begin{cases} 0 & \text{if terminated} \\ 30 & \text{otherwise (including overshot)} \end{cases}$$

Secondly, a more aggressive signal has been tested, where the termination function is not considered as a separate fitness value, rather, the previous three fitness values are set of 0 is termination is achieved. However, this signal dissuades the GA from further maximizing the other fitness values once termination is achieved.

$$\mathscr{F}_d, \mathscr{F}_p, \mathscr{F}_{\mathrm{pf}} = \begin{cases} 0 & \text{if terminated} \\ \mathscr{F}_d, \mathscr{F}_p, \mathscr{F}_{\mathrm{pf}} & \text{otherwise (including overshot)} \end{cases}$$

The third function is more granular, instead scoring for the Euclidean distance of the last placed frame to the end point on the design gesture $\mathrm{G_N}$. This distance is then remapped from a range based on the X component of the bounding space, representing in essence the furthest possible deviation, to a range of 0 to 30, reflecting the flat score previously described. As such, the GA receives different feedback with each iteration, instead of a binary score.

$$\mathscr{F}_{\mathrm{term}} = 30 \cdot \frac{\mathrm{clip}(\|\mathbf{c}_s - \mathbf{g}_N\|, 0, x_{\mathrm{bounds}})}{x_{\mathrm{bounds}}}$$

Each value, $\mathcal{F}_{\mathrm{d}}$, $\mathcal{F}_{\mathrm{p}}$ and $\mathcal{F}_{\mathrm{term}}$ are used as separate fitness functions, creating a multi-objective optimization.

Constrained Action Space Variant

The second variation of the genetic validation method was aimed at a reduction in size of the action space. This was achieved by collapsing the mirror, target and candidate orientation subsets into a single action — labelled as 'orientation' in figure 7.1.1. The genetic algorithm picks from this pool as well as the stock pool, which remains unchanged. It is no longer able to decide outright how a frame is placed. Instead, each possible placement 0 of the selected frame is calculated, filtering out non-progressing frames and scoring the remaining for progress and distance to the design gesture.

$$S = \{o \in O : t(s) - t(s-1) > 0\}$$
$$R(o) = F_d(o) + F_p(o)$$
$$\text{ranking:} S \text{sorted by} R(o), \text{descending}$$

The placement options are then ranked. The genetic algorithm can then select from the highest scoring orientations, depending on the range of the orientation gene pool. Intersections are also filtered out, after the ranking step is performed. Any frame which does not pass the intersection check is removed and replaced by the next best orientation, provided that this orientation does pass the intersection check. The ranking is maintained, however. If fewer options remain than the size of the orientation pool allows for, and the algorithm tries to pick an orienation that does not exist because of this, the action is skipped.

The fitness values are determined through the finite element analysis step, provided that the design algorithm has yielded termination. All tests were performed on non-strict termination, meaning that the last frame is allowed to exceed the bounding space of curve, provided that the centroid of said frame within the terminal threshold, which was set to 300 millimetres on all iterations. If terminated, a first-order FEA analysis is performed which yields three critical variables: Maximum deformation, maximum tensile stress and maximum compressive stress. These are consequently remapped to a reward function, using prescribed bounding ranges.

$$\mathscr{F}_{\mathrm{deform}} = \frac{\mathrm{clip}(\delta_{\max}, 0, 400)}{400}, \quad \mathscr{F}_{\mathrm{stress}} = \frac{\mathrm{clip}(\sigma_{\max}, 0, 2000)}{2000}$$

A reciprocal function is used to ensure that the reward increases most near the threshold for allowable deformation and fibre stresses, yielding the following formula:

$$\mathscr{F}_{\mathrm{recip}} = R_0 + \left(R_1 - R_0\right) \frac{F^{-k} - v_1^{-k}}{v_0^{-k} - v_1^{-k}}$$

Where R represents the reward range, and v represents the value range. k is a variable, however, throughout testing it has been set to 0.010 for deformation and 0.015 for stress. This yields three fitness values in total, for deformation, compression and tension respectively.

Structural Optimization Variant

The last variation on the genetic validation method is the same as the constrained action space variant, with the addition of two possible variations, one of which is expressed through a gene as well.

Longitudinal Arch Length Testing

The first consideration is the longitudinal arch length, effectively: Is one arch used to support the entire length of the bridge deck, or are multiple arches used? To this end, two variants have been tested. The full arch length, 15 metres accounting for the space of the foundations, and half the length, 7 metres, again accounting for the space of the foundations.

Perpendicular Arch Length Testing

The amount of perpendicular arches essentially dictates the tributary width, and in turn, the load which is exerted onto the arch. To this end, variants are to be tested ranging from 2-9 arches. For this, a multi-objective optimization is performed, with an additional gene introduced which controls the tributary width. **Table 7.2.2** contains the different widths tested. A fitness function is additionally required, to motivate the optimizer to try smaller arch counts. This function is defined as follows:

$$\mathscr{F}_{\text{trib}} = 6\frac{(n+1)^{-k} - 7^{-k}}{1 - 7^{-k}}, \quad n \in \left\{0, \dots, 6\right\}$$

Arch Count	Tributary Width	Gene (integer)
2	1.250 metres	0
4	0.833 metres	1
5	0.625 metres	2
6	0.500 metres	3
7	0.416 metres	4
8	0.317 metres	5
9	0.312 metres	6

Table 7.2.2 - *Arch counts and Tributary widths* - Own work

7.3 Results

The following section contains an overview of the key findings of the genetic validation. Firstly, the full action space variant is assessed through a variety of approaches. After this, the constrained action space is evaluated, Utilizing it to specify the yet undetermined longitudinal and perpendicular arch counts.

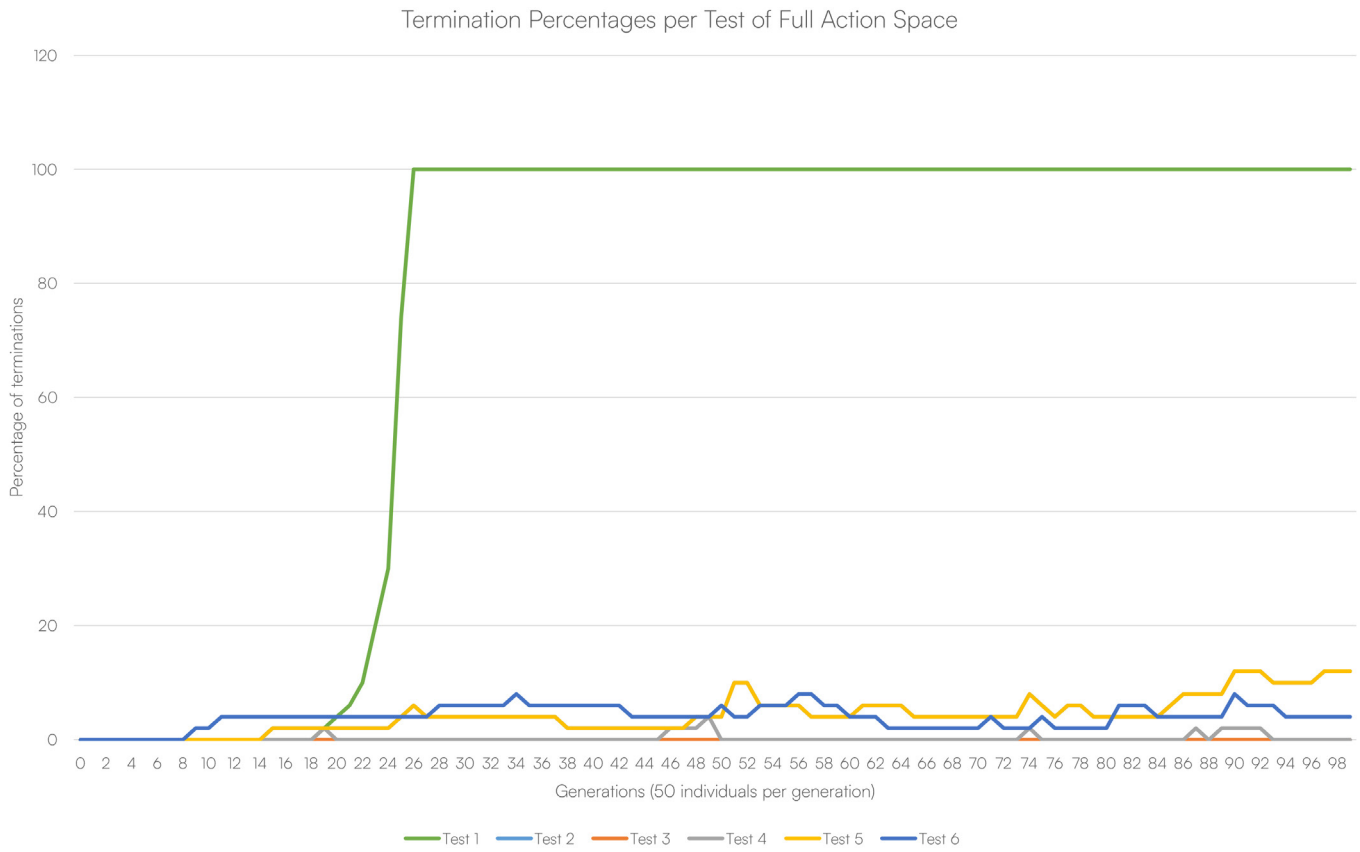


Table 7.3.1 - *Termination Percentages for different reward setups* - Own work

Full action space

The figure below, **Figure 7.3.1**, contains an overview of the different tests which have been conducted on the full action space variant. These tests consist of the following:

1. In test 1, the aggressive fitness function is used from the methodology, setting each fitness value to 0 if the algorithm terminates. Whilst the graph appears successful, in actuality, the algorithm is no longer incentivised to improve, yielding structures with little regard for the distance or progress functions, see figure **7.3.2**.

2. In test 2, the flat termination penalty has been used - yielding a fitness score of 30 for failing to terminate, 0 if the agent succeeds.

3. In test 3, the granular fitness function is used for termination, yielding progressively lower scores the closer that the final frame is to the end point of the curve.

4. In test 4, no termination reward is given. This is done to assess if the GA might terminate through the signals for distance, progress and placed frames alone.

5. Test 5 is a repeat of test 2 with a higher penalty, from 30 to 90.

6. Test 6 is a repeat of test 3, with a higher penalty, from 30 to 90.

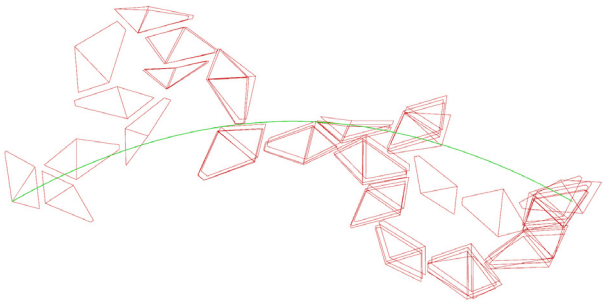


Figure 7.3.2 - *Example of test 1 geometry* - Own work

Most promising is test 2, the flat penalty. Increasing said penalty to 90 yielded no difference in the optimization routine. Test 3 yielded no termination, however, repeating it with a higher maximum penalty of 90 in test 6 did yield termination. However, none of these tests proved to have termination rates steady enough to warrant moving further to the FEA analysis, as this would further complicate the fitness objective. Still, being the most promising, test 2 was extended from 100 total generations to 300 total generations, to assess if the termination rate would further improve over long spans of training, but as **Figure 7.3.3** indicates, the rate of termination plateaus.

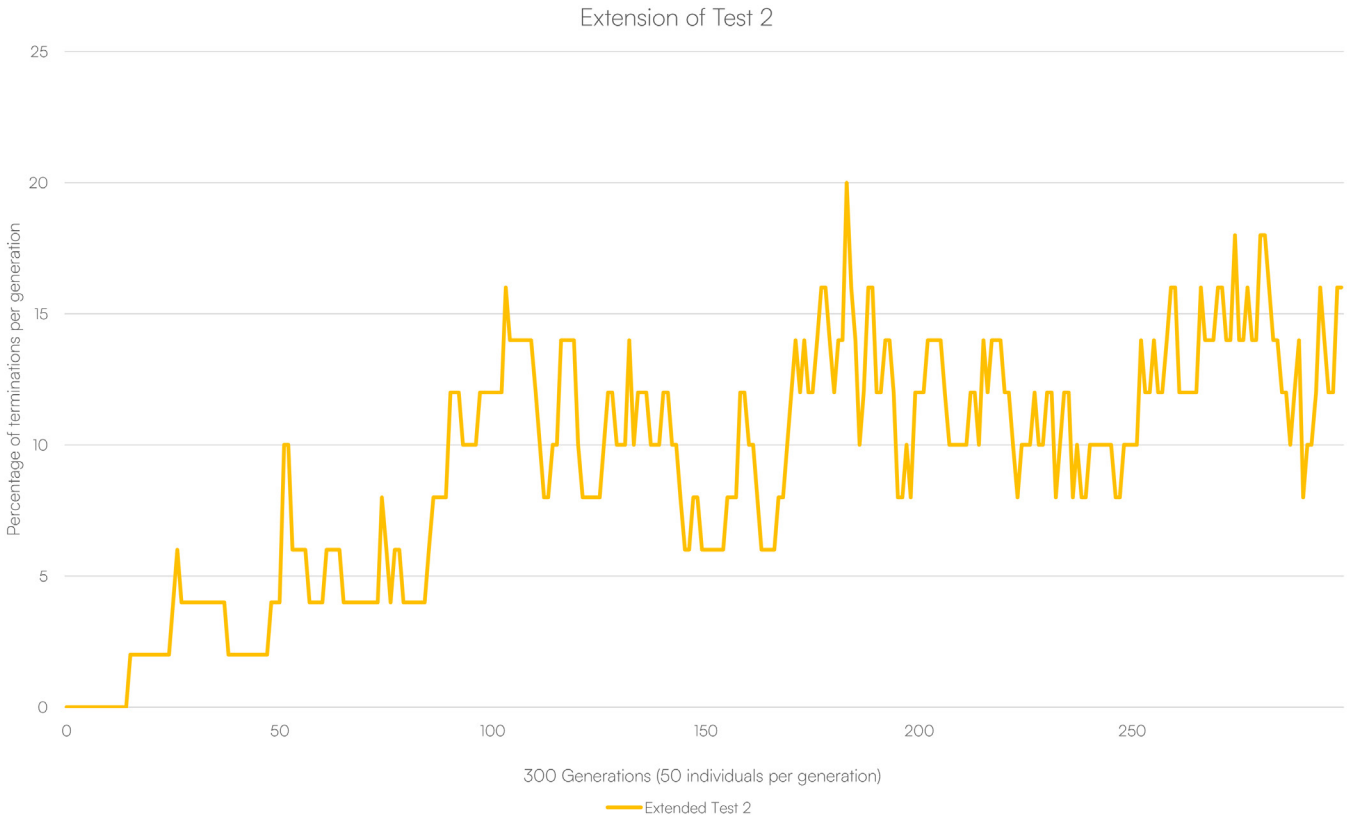


Table 7.3.3 - *Extended run of test 2* - Own work

Constrained Action Space

The constrained action space variant was implemented to fix the issue of the termination rate, so that the GA would be more likely to perform actual structural optimization. Furthermore, the constrained variant has also been used to specify the structural design metrics, being the number of arches placed longitudinally to the deck as well as perpendicularly underneath it, to which end **Figure 7.3.4** has been included, borrowed from the generative algorithm section. Firstly, the determination of the longitidunal count is highlighted, and secondly the perpendicular count.

Longitudinal count Testing

For the longitudinal count, two options have been considered: An arch which covers the full length of the bridge deck, and an arch which covers half of the length. The full length arch quickly proved unfeasible. To this end, **Figure 7.3.5** has been included which shows the progression of the maximum deformation over an optimization routine of 300 total generations, with 50 individuals per generation. The perpendicular arch count has been fixed to the highest number under consideration, 9 arches. The size of the orientation pool has been set to 5. Testing on the shorter, feasible arch length revealed that this pool yielded the second highest number of solutions, only being surpassed by the variant without any orientation selection, but the choice for 5 orientations was consciously made, as to not limit the potential solution space.

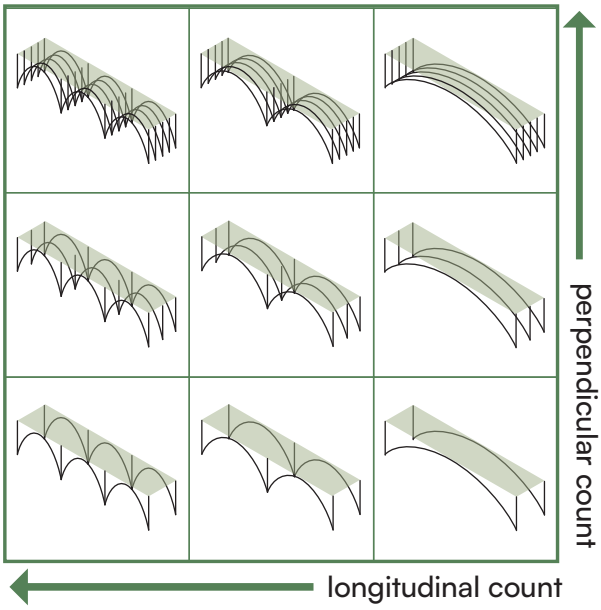


Figure 7.3.4 - *Arch count variations* - Own work

Longitudinal Arch Specification

As the routine fails to meet the minimum requirement for deformation, which for a span of 15 metres is 3.75 centimetre, the first of the structural design parameters is specified: Two longitudinal arches are to be used. Accounting for the spacing of the foundations, this yields an arch length of 7 metres per arch.

	50 pool	25 pool	10 pool	5 pool	2 pool	no pool	Total
2 arches	0	0	0	0	0	0	0
4 arches	0	0	295	667	0	131	1057
5 arches	0	0	1	0	0	161	162
6 arches	0	0	0	0	225	26	251
7 arches	0	449	431	738	0	572	2190
8 arches	0	0	0	0	0	0	0
9 arches	0	0	2	0	0	756	758
Total	0	449	729	1405	225	1886	

Table 7.3.5 - *Full FEA variants generated* - Own work

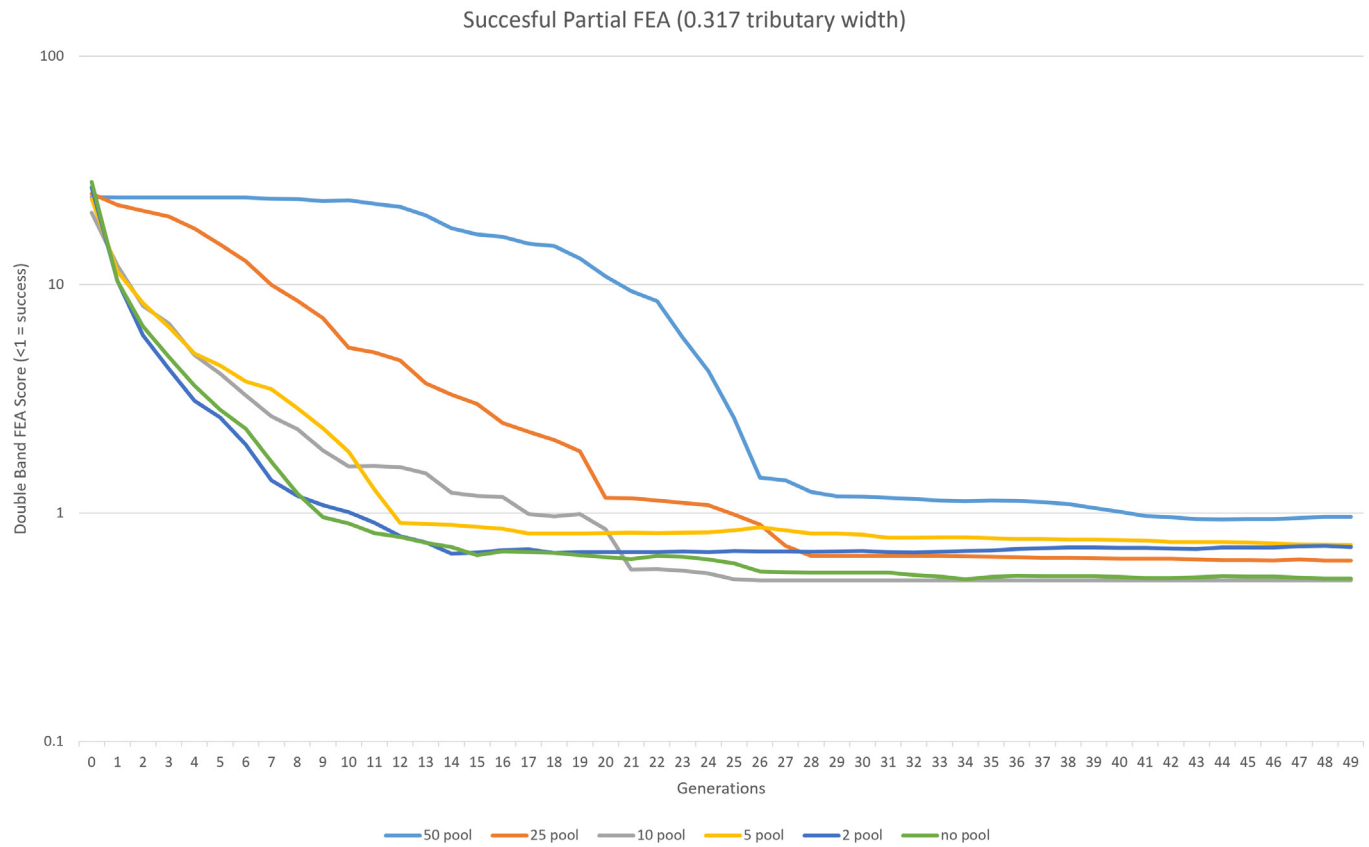


Figure 7.3.7 - *8 Arches partial FEA scores* - Own work

Perpendicular Count Testing

For the perpendicular arch count, the decision was made to also test the size of the ranked orientation pool, to ensure that the final results reflect a large spectrum of testing variations. To this end, 7 different multi-objective optimization were performed, with the fitness functions for stress, deformation and tributary width included. **Table 7.3.4** contains the results of this test, where each entry reflects the amount of solutions which passed the full FEA criteria, meaning that the fibre stresses for the connections and cables are also included. Each run was performed for 200 generations of 50 individuals. As is evident, the no pool variant yielded the largest amount of valid solutions.

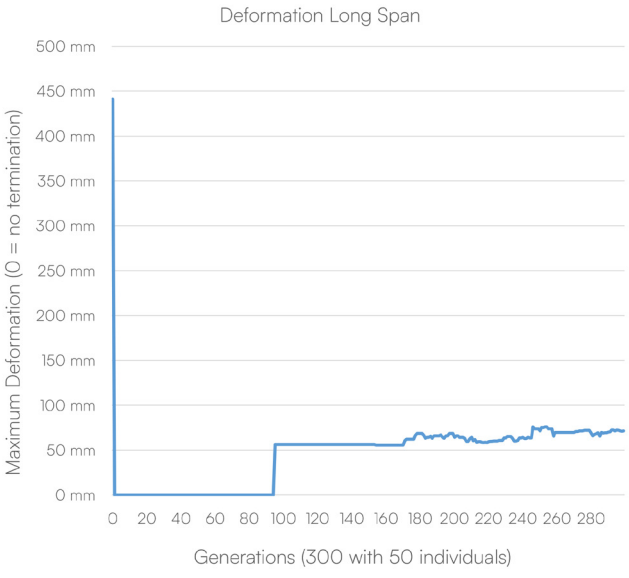


Figure 7.3.6 - *Long Span Displacement* - Own work

However, this variant is also the most constrained of the action spaces. The second most optimal variant is the 5 pool, where the algorithm could select from the 5 best scoring placement options.

The table does seem to imply that the 50 pool yielded no terminations whatsoever, but this is not true. Fixing the tributary to 0.317 (8 arches) and running every pool without this additional parameter yields the graph of **Figure 7.3.7**, which maps the partial FEA performance via a unity score given as:

$$
\begin{aligned}
\eta_\delta &= \frac{\delta_{\max}}{17.5} \\
\eta_t &= \frac{\max(0, \sigma_{\max})}{621} \\
\eta_c &= \frac{\max(0, -\sigma_{\min})}{483} \\
S &= \begin{cases} \dfrac{\eta_\delta + \eta_t + \eta_c}{3} & \text{if all} \eta_i \leq 1 \\ 1 + \displaystyle\sum_i \max\Big(0, \eta_i - 1\Big) & \text{otherwise} \end{cases}
\end{aligned}
$$

Any number below 1.0 meets all three criterias, whilst numbers grow exceedingly larger than 1 if any of the criterias is violated. What this graph shows especially, is that the 50 pool is capable of yielding structurally adequate results, but less effectively so than the other orientation pools. Furthermore, the 10 pool is actually shown to yield slightly better performance compared to the 5 pool, even if the previous analysis shows that it finds said solutions less frequently.

Each previous run was performed for 300 generations to provide ample runtime to yield optimal solutions. However, as the charts of **Figures 7.4.6**, **7.4.7** and **7.4.8** indicate, the algorithm reaches a plateau in performance well before even 50 generations. For these charts, an arch count of 6 was used with an orientation pool of 10 - furthermore highlighting that, whilst **Table 7.3.4** showed no solutions generated with these settings, that it is very much possible to do so, the multi-objective optimization simply discounted the 'allele' which responded to said arch count.

Example Solutions

The settings used for the charts to the right were also iterated with different seeds, and **Figures 7.4.9**, **7.4.9** and **7.4.10** contain samples from the pareto-front solutions of these seeds. This highlights a behavioural characteristic of genetic optimization, especially as it pertains to application potential as a tool for design. The optimizer tends to find a narrow bandwidth of settings which produce optimal results, and the variety amongst the optimal results is quite sparse.

7.4 Preliminary Conclusions

The intent of this chapter was to confirm the hypothesis that the generative algorithm could feasibly be optimized. As such, it was asserted that it is possible to terminate under the full action space, but that the frequency of this was relatively sparse, which does not indicate a good potential for actual FEA structural optimization. What did proof to be effective was to limit the potential selections, replacing the target orientation, candidate orientation and mirror Boolean inputs with a simple ranked list to be selected from. Regarding the design parameters, it has been determined that a full span is not feasible, but a half span of 7 metres is. Regarding the arch count, it has been asserted that at least 4 arches are required to create structures which pass the complete FEA check. Whilst it is not a crucial element of this chapter, analysing the results of even a relatively short optimization routine highlights the lack of variation between optimal solutions. Ideating on this, it becomes evident that the optimization does not make for a very immediate design tool.

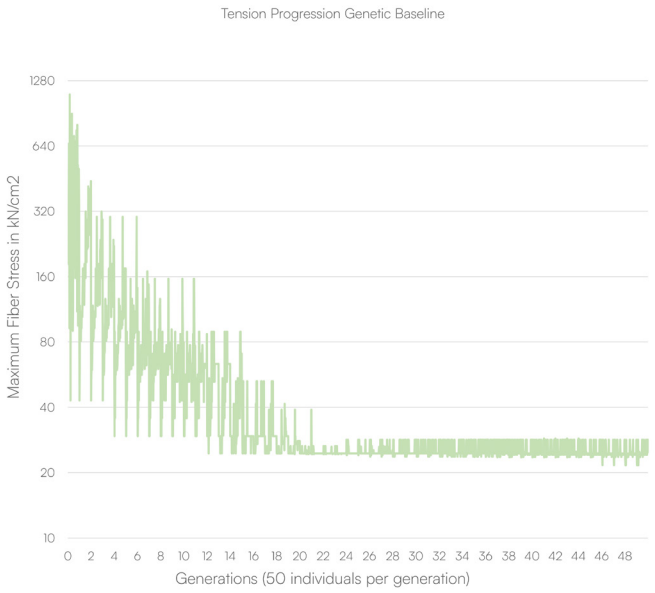


Figure 7.3.8 - *Tension throughout optimization* - Own work

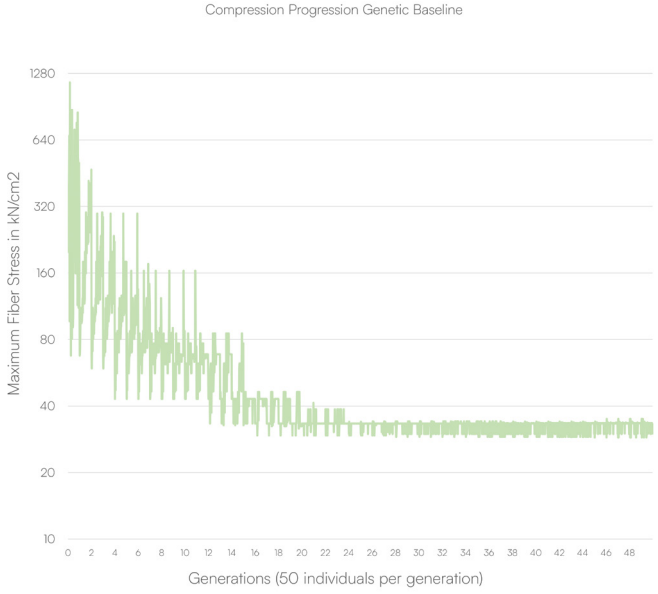


Figure 7.3.9 - *Compression throughout optimization* - Own work

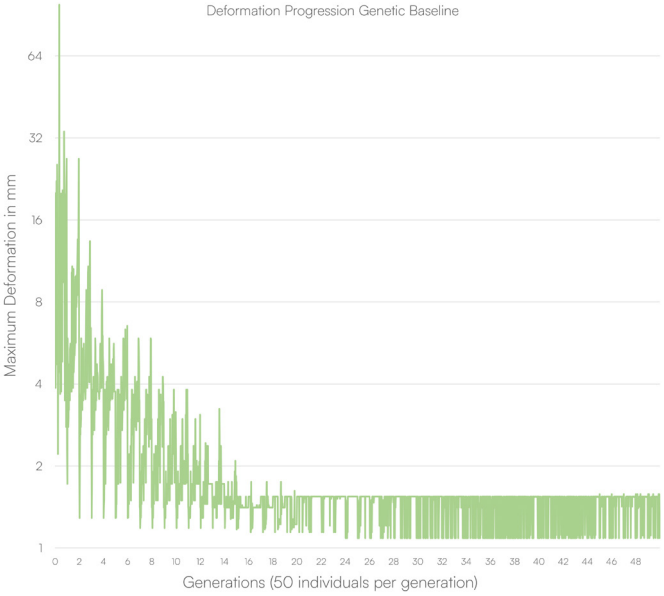


Table 7.3.10 - *Deformation throughout optimization* - Own work

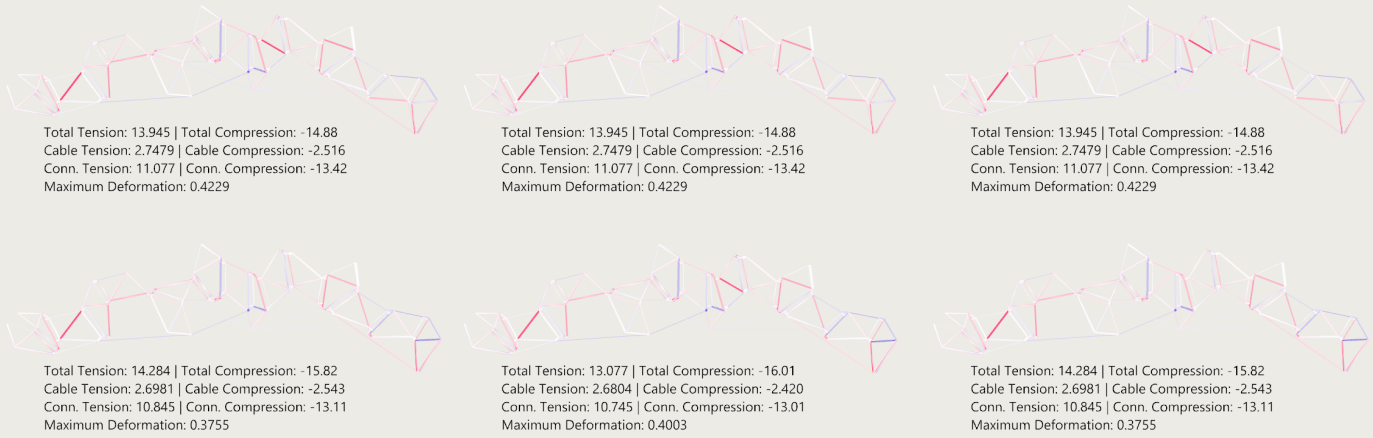


Figure 7.3.11 - *Examples generations seed 0 | 6 arches | 10 orientation options* - Own work

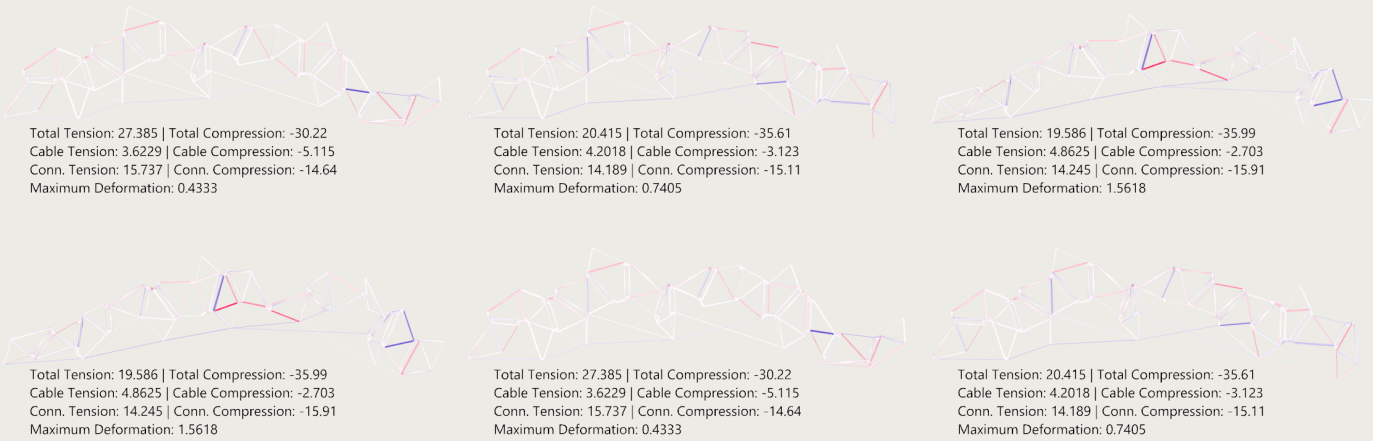


Figure 7.3.12 - *Examples generations seed 3 | 6 arches | 10 orientation options* - Own work

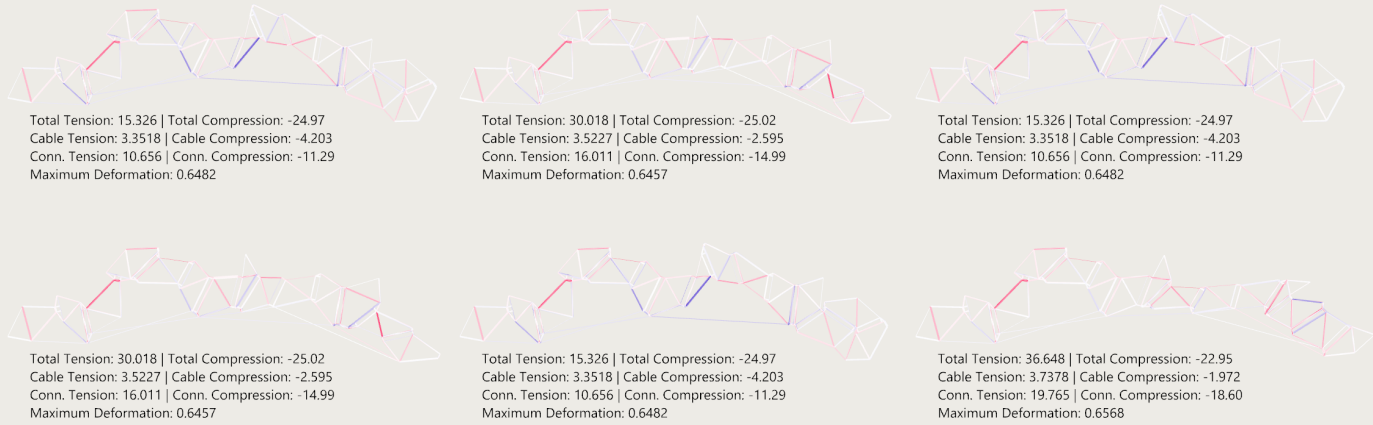


Figure 7.3.13 - *Examples generations seed 42 | 6 arches | 10 orientation options* - Own work

8.0 REINFORCEMENT LEARNING APPLICATION

Having validated the design algorithm via genetic optimization, additionally specifying the parameters of the design gesture and defining a baseline FEA performance, the design algorithm can now be used to train a reinforcement learning agent. This chapter will consist of a theoretical framework outlining reinforcement learning at large as well as relevant prior research. The methodology will describe how the design algorithm has been adapted into a reinforcement learning environment with the aim of utilizes Proximal Policy Optimization (PPO) as the method by which the agents are trained. The results will highlight the most promising agents, comparing the FEA performance which they have achieved against the baseline, testing capacity for generalized thinking and potential for user interaction.

Chapter Objective

The goal is to train an agent to optimize structural assemblies based on the design gesture and FEA criteria. This introduces a variety of metrics by which the agent is to be validates. Firstly, as with the genetic validation, the agent needs to be able to terminate with a meaningful frequency. Secondly, the agent needs to be able to optimize the structural performance, ideally passing the structural verification criteria outlined in chapter(x). If this is successful, the agent can be tested for generalized application, meaning that the training set of frames is replaced with a different dataset. The agent will be tested for both termination frequency and FEA performance under the new dataset. The last criterium deals with interactive capacity. This is achieved by submitting a variant of the environment to the agent with a partially complete structure. The agent is then similarly tested for ability to converge and FEA performance, though the latter will also be impacted by the partially complete structure, intended to represent the input of a designer, and therefore an interactive process.

Relevant Research Question

Can a reinforcement learning agent successfully utilize the material-first generative algorithm to produce optimized design solutions given the provided material and design context?

8.1 Theoretical Framework

Machine learning is commonly divided into three subsets: Supervised, unsupervised and reinforcement learning. Understanding the differences between these three typologies can be illustrated via the consideration of training data and general objective.

In supervised learning, the training data typically consists of input and output pairs (x,y), and the objective is to generalize to unseen inputs drawn from the same distribution, which is to say, a wider dataset from which the training data was drawn. In simpler terms: The algorithm is trained on inputs which have known outputs and is then tested to see if it can produce the correct output based on a new, unseen input. It achieves this via a supervising signal, hence the name. As such, supervised learning requires both a training dataset as well as a validation dataset a-priori.

Unsupervised learning, however, does not require a validation dataset. The goal is not to match inputs with the likeliest outputs, but to find underlying structures in the input data itself.

Reinforcement learning takes a significant departure from either previous subset. The optimizer is not directly told which input belongs to which output, as with supervised learning. Kaebling (2019) described the following interaction sequence:

1. Learner observes input (s)i
2. Learner generates output (a)i
3. Learner observes reward (r)i
4. Learner observes input (s)i+1
5. ...

For supervised and unsupervised learning, the goal objective is to derive at a hypothesis which represents the data with meaningful accuracy. In reinforcement learning, the goal is instead to derive at a policy. The policy does not directly predict values or patterns as with supervised and unsupervised learning, but rather predicts states of a system, and how valuable those states are.

For this, consider the Markov Decision Process, or MDP. An MDP is a variation of a state machine, a process which is described in terms of potential sequences of states. Kaebling (2019) defines a state as all that you need to know about a system to predict all possible future trajectories. An MDP is formally defined as a tuple of 5 elements $[S,A,T,R,y]$:

1. S is a finite or infinite set of possible states
2. A is a finite or infinite set of possible actions
3. $T : S \times A \times S \rightarrow \mathbb{R}$ Is a transition model where:

$$T(s,a,s') = Pr(S_t = S' | S_{t-1} = s, A_{t-1} = a)$$

4. R: $S \times A$ is a reward function, where $R(s,a)$ specifies how desirable it is to be in state s and to take action
5. $\gamma \in [0,1]$ is a discount factor

The transition function of an MDP is stochastic, meaning that it defines a probabilistic distribution over the next possible states given the previously observed state and the action taken.

Beyond this, there is the consideration of finite and infinite horizon MDPs. With a finite-horizon MDP, the total number of steps that the agent has with the MDP is known, which in turn impacts how the agent estimates the value states. If the agent is tasked with navigating a maze with multiple possible routes to the goal post, knowing it can only move a certain number of steps would greatly impact which route the agent takes. Typically, the approach to optimizing finite-horizon MDPs is to determine an optimal action-value function, Q.

More commonly, MDPs are considered as having infinite horizons, in which case the agent does not know when the

game will be over. This prevents the definition of an optimal action-value function, as the agent cannot predict how many steps it has left to complete. Instead, the approach is to define a discount factor Y, the goal being to predict the expected infinite horizon discounted value, as opposed to the expected horizon undiscounted value via the aforementioned optimal action-value function. The discount factor essential reduces the value of an expected future state, depending how far it is removed from the current state (Kaebling, 2019).

Whilst having outlined the basic principles of reinforcement learning through the Markov Decision Process, it is not yet addressed how a reinforcement learning agent is actually trained.

The basic principle is to create an environment and to then run a reinforcement-learning optimization algorithm inside said environment. Whilst the sequence described by Kaebling (2019) would imply that the 'learner' is passively observing, the optimization algorithm generally does interact, via the selection of actions, with the environment. As training progresses, the algorithm continually updates the policy, also commonly labelled as 'agent'. It picks an action from a probability distribution which is initially stochastic, but said distribution ideally trends towards deterministic over the course of training.

Neural Networks

Most contemporary reinforcement learning methods, and indeed machine learning methods at large, make use of neural networks, a specific kind of network architecture based on the neurons of biology. These networks consist of layers of neurons, the basic units which receive inputs. Typically, a neural network is comprised of an input layer, hidden layers and an output layer. Critically, these networks iteratively refine their sequence of input-process-output through forward and backward propagation. What constitutes good output is dependent on the application (*Introduction To Neural Networks*, 2026). Neural networks are commonly used in image recognition, predictive speech and complex data processing.

Reinforcement Learning Algorithms

One of the more foundational algorithms in reinforcement learning is Q-learning, originally developed by Christopher Watkins. Q-learning is a model-free algorithm, meaning that the algorithm does not create its own model of the environment. It is a tabular learning algorithm, revolving around a structured dataset known as the Q-table. This table maps all possible state action pairings repeatedly, converging to optimal action-values through the following formula:

$$Q\left(S_t, A_t\right) \leftarrow Q\left(S_t, A_t\right) + \alpha\left[R_{t+1} + \gamma \max_a Q\left(S_{t+1}, a\right) - Q\left(S_t, A_t\right)\right]$$

Watkins & Dayan (1992) demonstrate the functionality of Q-learning under the presumption that all actions are repeatedly sampled in all states, provided that the action-values are represented discretely, i.e. as real numbers. Q-Learning works well if all state-action parings are known. As such, it commonly struggles with more complex MDPs, critically those which feature higher-dimensional action spaces. More recent usage of Q-Learning is performed via Deep Q-Learning, or DQN, which utilizes deep-learning neural networks which act as approximators for each state action pairing. The advantage of this is that the feasible size of the state space is considerably larger than that of traditional Q-learning (*Deep Q-Learning in Reinforcement Learning*, 2026).

Actor-Critic architecture is a different approach towards reinforcement learning problems, reliant on temporal difference and trial-and-error learning methods. In essence, this method revolves around separate memory structures which independently represent the policy and the value function (Sutton & Barto, 2014). The actor refers to the policy network, which undertakes actions and receives the new state. It is tasked with predicting the likelihood that an action leads to a given state. The critic refers to the value network, which is independently tasked with predicting the value of a given state. Generally speaking, actor-critic architecture is better suited to multi-dimensional action spaces, compared to Q-Learning and DQN.

A contemporary algorithm which has seen a strong rise in popularity is Proximal Policy Optimization, or PPO. Developed by Shulman et al. (2017), the algorithm makes use of stochastic gradient ascent to update the policy over the course of training. The major difference with previous algorithms comes in the form of a new objective function which allows for multiple epochs of minibatch updates, opposed to just one gradient update per data sample. Shulman et al. (2017) demonstrate that their algorithm outperforms other policy gradient methods through a variety of benchmarks, ranging from robotic locomotion to Atari game playing.

Proximal Policy Optimization works by having the actor network interact with the environment for a pre-determined number of steps, defined by the n_step hyperparameter, collecting a rollout buffer containing the transition tuple.

$$[s_t, a_t, r_t, V(s_t), \log \pi(a_t \mid s_t), \text{terminated}, \text{truncated}]$$

On rollout completion, a bootstrapping step is performed if the last step collected did not produce a termination or truncation. After this, the temporal-difference error is calculated over each transition in reverse order. From this, a set of advantages (A) is calculated for each transition (t), indicating the value of said transition per the reward signal.

$$\delta_t = r_t + \gamma V(s_{t+1}) - V(s_t)$$

$$\hat{A}_t = \sum_{l=0}^{\infty} (\gamma\lambda)^l \delta_{t+l}$$

This process is known as the Generalized Advantage Estimation or GAE. Once complete, the update phase commences, which is where the potential of PPO becomes most pronounced.

$$r_t\left(\theta\right) = \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_{\text{old}}}(a_t \mid s_t)}$$

Firstly, the rollout buffer is shuffled and partitioned into batches, dictated by the hyperparameter batch_size. For each batch, the probability ratio $r_t(\theta)$ is calculated using both the distributions of the old and new policy network.

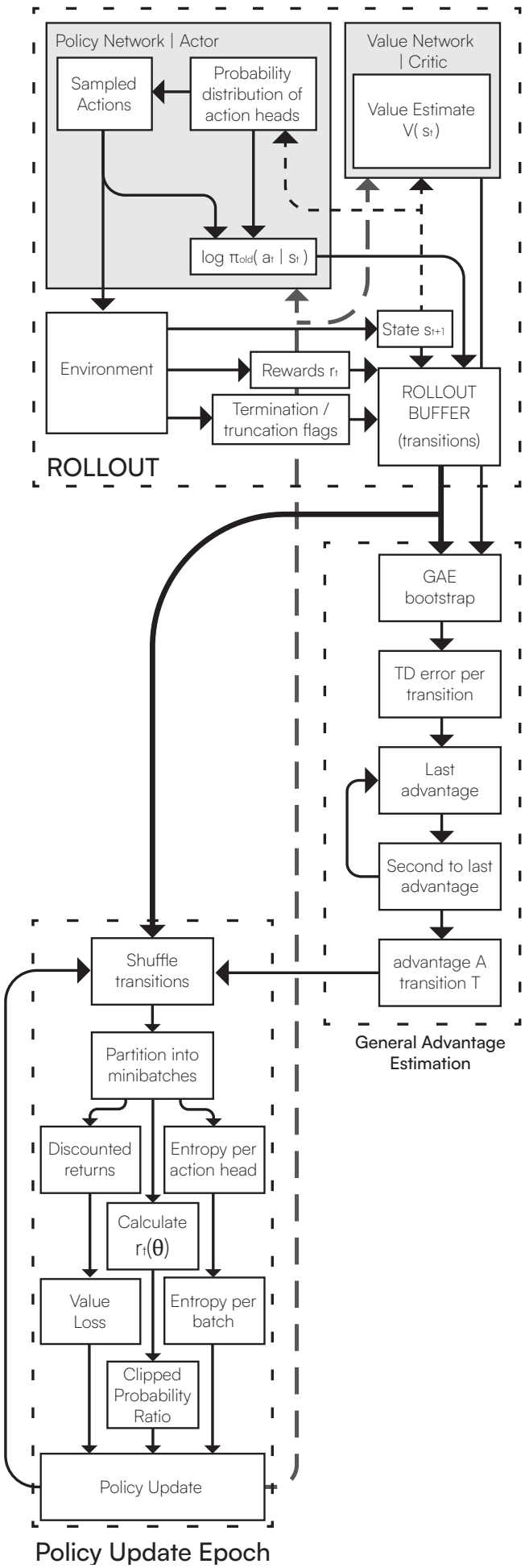


Figure 8.1.1 - *PPO overview flowchart* - Own work

After this, the policy loss is calculated via a clipping ratio, dictated by hyper parameter clip_range. This step ensures that the new policy does not diverge too strongly from the existing one.

$$L^{\mathrm{CLIP}}\left(\theta\right) = \hat{\mathbb{E}}_t \left[\min \left(r_t\left(\theta\right) \hat{A}_t,\ \mathrm{clip}\left(r_t(\theta), 1-\epsilon, 1+\epsilon\right) \hat{A}_t \right) \right]$$

At the same time, the value estimates under the updated policy are processed using the discounted return, utilizing the advantages calculated in the GAE step, after which the value loss is computed.

$$G_t = \hat{A}_t + V\left(S_t\right)$$

$$L^{\mathrm{VF}}\left(\theta\right) = \hat{\mathbb{E}}_t \left[\left(V_\theta(S_t) - G_t\right)^2 \right]$$

Lastly, the entropy per transition is calculated as the sum of the entropy of each action head, after which the entropy for the entire minibatch is computed.

$$H_h\left(S_t\right) = -\sum_{a \in \mathscr{A}_h} \pi_\theta^h\left(a \mid S_t\right) \log \pi_\theta^h\left(a \mid S_t\right)$$

$$H\left(S_t\right) = \sum_{h=1}^{4} H_h\left(S_t\right)$$

$$H_{\mathscr{B}}\left(\theta\right) = \frac{1}{|\mathscr{B}|} \sum_{t \in \mathscr{B}} H\left(S_t\right)$$

From these three elements, being the policy loss, value loss and entropy per minibatch, the total loss is calculated and used to update the policy. This happens iteratively for the total number of minibatches. This constitutes one epoch. The entire process is repeated, starting by re-shuffling the rollout buffer, for the number of epochs prescribed.

The strength of PPO is especially pronounced in the utilization of several epochs of policy updates combined with the aforementioned clipping ratio, leading to more efficient and more effective training.

One metric which requires some elaboration is that of the explained variance. This metric is often used in research to illustrate the degree of comprehension that a given reinforcement learning agent portrays. It is calculated as the difference between the predicted values and the actual values (returns).

$$\mathrm{EV} = 1 - \frac{\mathrm{Var}[G_t - V(S_t)]}{\mathrm{Var}[G_t]}$$

Values which approach 1.0 indicate that predictions match the actual return values, values which dip below 0.0 indicate that the value function is effectively incapable of estimating the return value.

Relevant Studies

The following section outlines previous research in the realm of reinforcement learning, further narrowed by the scope of material-first computational design. As was stated in the manifesto, outside of the realm of machine learning, there is little to no identifiable prior research which specifically implements generative design principles without stock cutting of some kind.

Ororbia & Warn (2022) conceive of an RL application utilizing Q-learning formalized through an MDP. The action space is represented through shape grammars. The goal is to optimize planar truss configurations with binary cross-sectional areas, providing four numerical examples to highlight the possible application value. The authors went on to extend their framework to a higher-dimensional state space, where each state is no longer representable via direct tabular learning. As such, they implemented DQN, showing a capacity to learning optimized policies which produce high performing designs frequently (Ororbia & Warn, 2023).

Mirra & Mueller (2025) has been highly influential to this thesis. They utilize the design games concept, of Mirra (2022), to abstract the design problem for AI training. The goal of their research was to train an agent using PPO to generate optimal cantilevered trusses from a constrained and diverse stock of linear elements. The workflow consists of the iterative placement of two stock elements, to form triangles. The structures were evaluated for structural performance at each step, applying a 10kN load to each free node. Similar to this thesis, they validated their method firstly via genetic optimization prior to the application of reinforcement learning. For the observation space, the authors made use of a convolutional neural network (CNN) to represent the bounding space as well as 1D vectors to represent each of the 20 stock elements, processed via a multi-layered perceptron (MLP). The authors made use of reward system consisting of penalties for intersections and excessive triangle angles, a base reward for the forming of valid triangles, two conditional rewards for the unsupported span and use of supports, and lastly a bonus reward for new nodal placements in the right support zone provided that only 2 supports exist. Their training progressed over 500.000 total steps, with a steadily climbing explained variance that eventually settled around 0.80.

The second research, in similar vain of the research of Mirra & Mueller (2025), is the master thesis of K. Sørensen (2024). Their goal is also to generate truss structures through the iterative placement of linear elements, forming triangles. Where Sørensen provides a stronger correlation to this thesis is in their use of a design gesture, a flat curve which the structure is intended to generate along. Furthermore, the author also performs the FEA analysis on structural completion, opposed to the stepwise approach of Mirra & Mueller (2025). They further diverge in the treatment of the stock observation, which is performed through binning the stock of elements into subsets of similar length. They furthermore provide the agent with a lookahead observation on the design gesture, given as three points spaced out in front of the current state of assembly. The trained agents are evaluated for their ability to follow the curvature and the maximum displacement. There is no indication as to the explained variance provided.

The work of Rajaskar et al. (2024) takes another approach to the question of structural evaluation. Whilst the aforementioned studies all make use of finite element analysis, the authors of this study use graphic statics instead. Furthermore, like Ororbia & Warn (2022), the authors make use of shape grammars to represent the action space. The authors argue that the advantage of graphic statics over FEA is that the system guarantees structurally equilibrated designs at each step. Rajaskar et al. (2024) solve their approach via Q-Learning.

Now, each of the aforementioned studies makes use of linear elements to assemble the structures. In this aspect, the focus of this thesis makes for a large shift, as there are no identifiable studies in current literature which seek to implement a reinforcement learning algorithm on a stock of elements which is not only non-linear but which also characteristically does not have a consistent cross-section for each sub element.

8.2 Method

The following section will outline how the reinforcement learning was approached. The choice was made to implement the Proximal Policy Optimization algorithm due to the high potential application for higher-dimensional action spaces. Scripts were written in Python, using most notably the Gymnasium and StableBaselines3 libraries.

The method will be structured along the following aspects: The action space, followed by the environment setup, followed by the reward functions, followed by the observation space, followed by the neural network setup, followed by the hyper-parameter setup.

The Environment

The environment at large can be thought of as a game, featuring actions which can be taken, which yields changes to the environment and additionally yields rewards or penalties. The environment for this thesis has been created via the Gymnasium template, existing as a Python file. **Figure 8.2.1** contains an overview of the environment.

The bounding space is defined as an XZ plane ranging from [0,8000] in the X direction and [0,2000] in the Z direction.

Each episode is constrained to a maximum of 35 steps. Provided with a high percentage of termination it is unlikely that the agent requires this many steps, however it does provide more steps in earlier stages when placement would be skipped through either reuse or intersection with previous frames.

The frames and areas are shuffled on each episode, as early training revealed that the agent will otherwise learn based on the index of each frame as opposed to the actual perception of the frame.

The Action Space

The final shape grammar provided in the manifesto informed the formulation of the design algorithm at large. The shape grammar exists in part as a set of rules. In the genetic validation

these rules were initially directly represented in the full action space variant, but this setup did not yield adequate termination frequency. For the reinforcement learning section, the rules are again directly represented, at least initially. This is done via the spaces subpackage of Gymnasium, defining a multi-discrete action space as a tuple of [35,5,5,2] — 35 reflecting the amount of frames placeable, 5 and 5 reflecting the candidate and target orientations, and 2 reflecting the mirroring Boolean.

The Observation Space

Observations are an integral part of reinforcement learning. The algorithm does not learn from the reward signal directly, instead it learns first and foremost to predict the next state based on an action and a previous state. On top of that it also learns the value of said state. The observation space for this thesis has been formulated through a series of continuous vectors between 0 and 1:

6. The guide curve (design gesture) is a 2D vector, shaped as [100,2] consisting of 100 sample points expressed through x and z values. These points are normalized through the bounding space.

7. The stock of frames is expressed as a 3D vector, shaped as [35, 5, 2] reflecting 35 frames in the stock, each expressed with 5 points which are in turn expressed with 2 coordinate values. For the normalization each frame is refitted using a common bounds of [630,630].

8. The placed frames are also given as a 3D vector, also shaped as [35,5,2] but note that 35 does not reflect the number of stock items here, but the maximum step per episode. At the onset of an episode each sub-vector is represented with a sentinel value of 0, when a frame is placed, the respective slot is filled with the coordinates of said frame, normalized to the bounding space, as with the curve.

9. The stock mask is a binary array with shape [35,] reflecting the number of stock items available. At the onset of an episode, this array is filled with 1s. When a stock item is placed into the bounding space, the respective stock mask is set to 0.

10. The areas of the stock reflect the cross-sectional areas of each tube, each tube is normalized independently across the stock, so that a given head tube, for example, is normalized by a bounding range derived from the largest and smallest head tube in the stock.

11. The progress along the curve is given as 1D vector, shaped [1,]. This number reflects a point on the curve, with 0 as the start of the curve and 1 as the end.

12. The episode progress is also given as a 1D vector, shaped [1,]. It reflects the current timestep, normalized to the maximum number of time steps.

Whilst the complete observation is of considerable size, with 1147 total components, it is important to state that a considerable portion of these are passive throughout the process, such as the stock, guide curve and areas. Furthermore, the placed frames observation is padded with sentinel values, and is unlikely to exceed the max step.

Extractor Setup

The observation vectors are not perceived directly by the value and action heads of the algorithm. They are processed by a set of neural networks prior to this. To this end, the choice was made to process the observations for the frames, placed frames and areas using an initial smaller neural network, the intent being to have a set of weights which are adjusted independently of which time step or stock order. As such, each placed frame, stock item and area item are processed through an independent MLP layer, after which each of these are pooled together

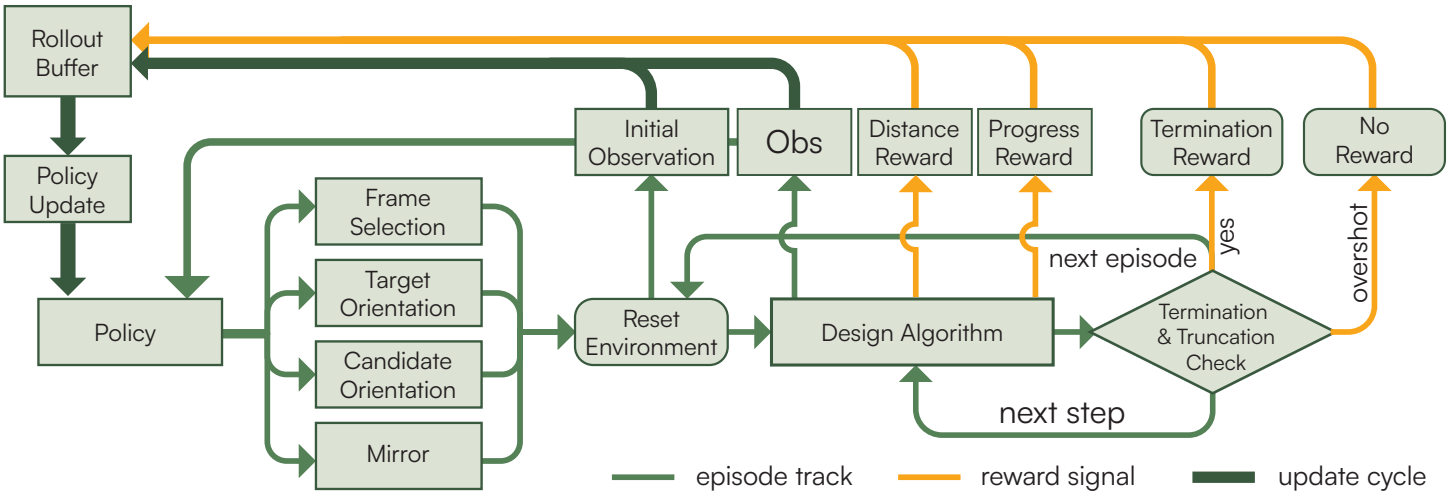


Figure 8.2.1 - *RL Flowchart* - Own work

with an additional max-pooling step. **Figure 8.2.2** reflects the configuration of the total extractor. The stock masking observation is additionally fused into the stock observation, to provide a stronger link between stock and consumed frames.

The Reward Function

The reward function is built from four aspects, the distance reward, progress reward, termination reward and the FEA reward. Each will be illustrated here. The progress and distance metric have a weight, w1 and w2, to control the balance between them.

The distance reward is based on the distance metric of the design algorithm, being the Euclidean distance between a frame's centroid and the closest point on the design gesture. This value is mapped to a clipped range, yielding a negative reward up to -1.0 at placements further than 500 millimetre from the design gesture, and up to 1.0 at a distance of 0.

$$R_d = \max\left(w_1\left(1 - \frac{d_s}{500}\right), -1\right)$$

The progress reward is based on the progress metric of the design algorithm. It does not yield a penalty, only a positive reward if the placed frame is further along the design gesture compared to the previous one. It makes use of a scaling function which takes the current progression, which is expressed as a value between 0 and 1, and turns this into a multiplier k_s so that progression becomes increasingly more valuable as the episode progresses, to more strongly encourage termination.

$$\kappa_s = 1.2(1 + t_s)$$

$$R_p = \begin{cases} w_2 \kappa_s \frac{\|\mathbf{dg}_s - \mathbf{g}_{Nt_{s-1}}\|}{1500} & \text{if} t_s > t_{s-1} \\ 0 & \text{otherwise} \end{cases}$$

There are three different terminal states, highlighted in the design algorithm section, reiterated here. The first state is the overshot, which is when a frame exceeds a bounding line defined perpendicularly to the end of the curve. In the RL environment, this yields a termination without reward. This is done to improve training efficiency, preventing episodes from extending further unnecessarily. This does not yield a penalty.

The second state is true termination, when the frame is placed within the bounding circle of the end of the curve, which has a diameter of 300 millimetres. In this case, a flat reward is yielded of 3.0, this value is calibrated to approximately 150% of the reward yielded for perfect placement.

$$R_{\text{term}} = \begin{cases} 3 & \text{if terminated} \\ 0 & \text{otherwise (including overshot)} \end{cases}$$

The third state possible state is called strict termination, and it is dependent on a Boolean, and therefore not always enabled. In this case, the agent needs to place a frame both in the bounding circle and not exceed the overshot domain. If it fails to do so, as with the overshot state, no penalty is given but the episode is terminated.

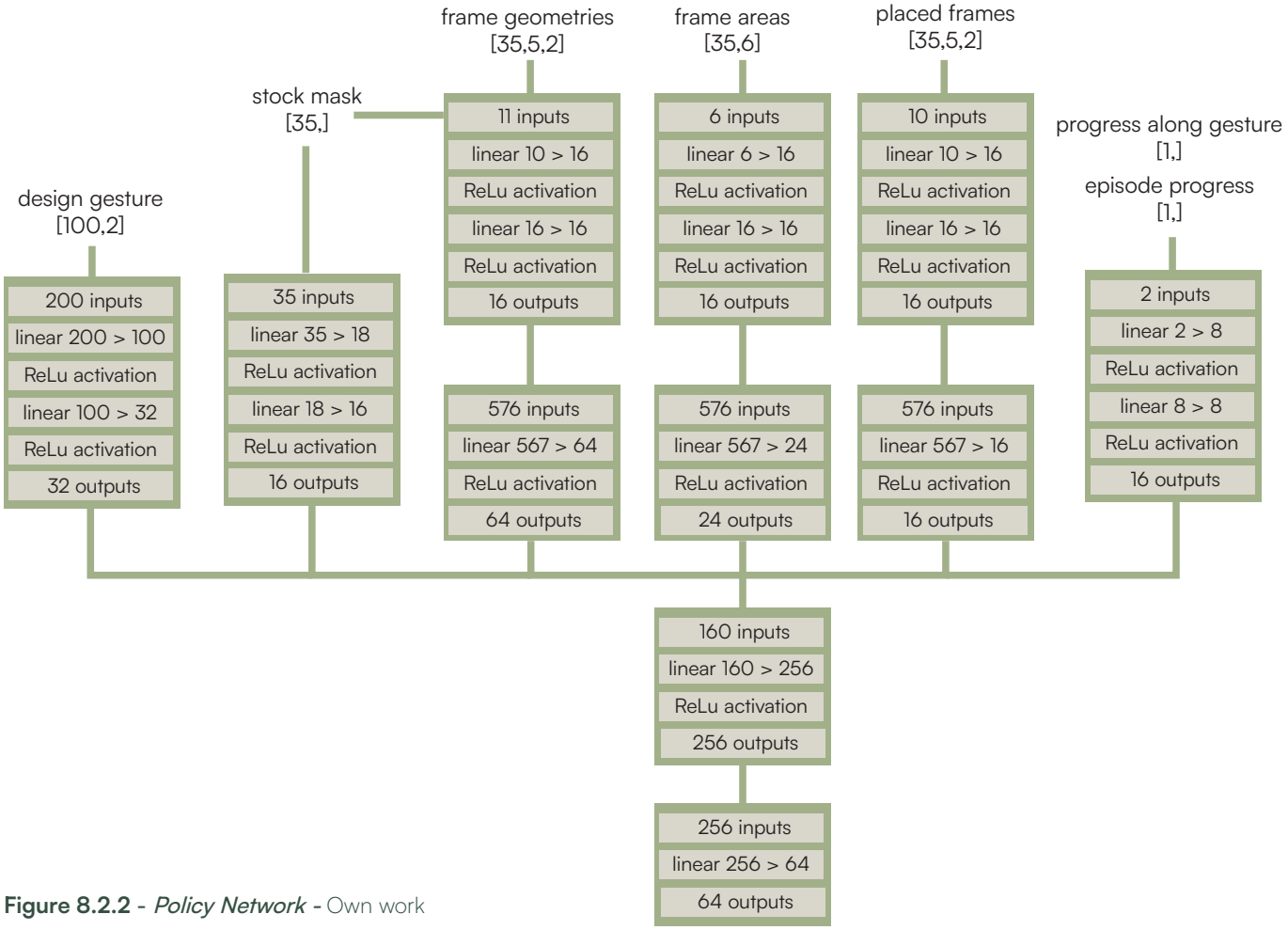


Figure 8.2.2 - *Policy Network* - Own work

The FEA reward is shaped in similar fashion to the GA validation, consisting of three subcomponents: Deformation, maximum fibre compression and maximum fibre tension. For the latter two, only the frames are considered. Each variable is remapped to a reciprocal function, described below:

$$R_\delta = R_{\max} \frac{\delta_{\max}^{-0.01} - 300^{-0.01}}{0.2^{-0.01} - 300^{-0.01}}$$

$$R_{\text{tens}}, R_{\text{comp}} = R_{\max} \frac{\sigma_{\max}^{-0.15} - 2500^{-0.15}}{10^{-0.15} - 2500^{-0.15}}$$

FEA never yields a penalty, only a reward. However, the reciprocal function ensures that at the reward only becomes significant, and increases more strongly, as the values approach the required performance targets, which are listed in the design algorithm section.

Agents are trained only on the total deformation and the frame stresses. However, they are further evaluated on the connection and cable stresses on deployment.

Penalty Functions

There are certain states which are prevented outright through the design algorithm, being the reselection of a previously consumed frame and the intersection with a previously placed frame. When either occurs, the placement is skipped, yielding no changes in the observation space, except for the episode progression. To further dissuade the algorithm from picking these actions, a penalty score is returned. For both reselection and intersection, this penalty score has been set to -2.0, constituting the lowest possible reward at a given step.

Reselection proved especially difficult to learn, however. If the penalty was increased over the intersection penalty, the agent would in turn struggle to learn non-intersecting behaviour. Attempts were made to set the relevant stock observation to 0, instead of the binary stock mask, but this did not yield particular improvement. As such, the decision was made to mask the placed frames through the action space. A relatively common practice in reinforcement learning, the process removes the specific action from the probability distribution — preventing the agent from selecting it outright.

Hyper parameters

PPO contains a swath of possible hyperparameters. In this thesis, most have been kept to their default values, with a few exceptions, listed below.

The n_step determines the size of the rollout buffer. However, if training is run with multiple parallel environments, the actual rollout buffer size is dictated by the n_step multiplied by the number of parallel environments. To improve the training speeds, the decision was made to use CUDA to train on the graphics processor instead of the CPU, using dummy vectorized environments. This does not mean that the environments run in parallel, but it does improve performance still. What proved to increase training speeds even further was to use a larger batch size, as it reduces the number of gradient updates, which proved to be the largest bottleneck for training speed. Larger batch sizes are generally also better suited to complex environments such as this. As such, the following hyperparameters were found to be most beneficial, yielding steady convergence at higher speeds:

1. 16 parallel dummy environments
2. N_step 256
3. Batch size 128

This makes for a rollout buffer of 4096 steps, which are then divides into 16 batches, yielding a total of 160 gradient updates after 10 epochs, compared to 320 updates in the default setup, but notably performed on a doubled batch size, yielding the same step-to-update ratio, only sorted differently.

Evaluating success

To assess the abilities of the agents, a variety of metrics are used to test them.

Firstly, the termination rate is intended to assess the ability of the agent to navigate the environment. This is generally quantified as the percentage of episodes per rollout which terminate, not accounting for the overshot episodes.

Secondly, the adherence to the design gesture is taken to evaluate how well the given agent is able to follow the prescribed curve over the course of training. This is quantified through the mean distance reward per episode. Values which approach 1.0 indicate close adherence to the design gesture.

Thirdly, the FEA performance is outlined along the three given subsets, being the deformation, fibre compression and tension. These values are quantified as the average per episode performance across a given rollout batch. Deployed agents are further checked on the fibre stresses of the connections and tension cables.

Beyond these immediate metrics, the agents are further tested for two qualities: Generalization and interactivity.

Generalization is tested by submitting the agent to a different random subset of the larger FRAMED database of Regenwetter et al. (2022). The performance is quantified by running 500 episodes of the agent acting deterministically and comparing the percentage of episodes which meet the full set of FEA criteria against deployment of the agent on the trained dataset.

Interactivity is tested by submitting the agent to a partially completed environment to assess whether it is able to complete the design gesture, and if so, if the completed design meets the performance requirements.

8.3 Results

The results section is divided into four aspects. Firstly, the convergence rate of the design algorithm is tested without FEA analysis, in similar vein to the genetic validation. Secondly, a variety of agents trained with FEA analysis are outlined. Thirdly, the aspect of generalized thinking is analysed, and lastly the interactive capacity of the FEA trained agents is tested.

Non-FEA Agents

As with the genetic validation, the design algorithm must first be tested for ability to converge in general, quantified through the rate of true termination - meaning that the agent successfully places a frame inside the bounding area known as the termination threshold - a circle of with a radius of 300 millimetres. As **Figure 8.3.1** shows, the initial agent struggled to reach frequent terminations. The reason for this is mostly related to the penalty actions, where placement of a frame yields either an intersection or makes use of a previously consumed frame - in both cases the action is skipped, yielding a penalty of -2.0. It was discovered that rebalances of the penalties for reuse and intersections mostly shifted which penalty was better understood by the agent, leading to a reduction in the rate of frame reselection whilst increasing the frequency of intersections, and vice-versa. The appendix contains additional training data which demonstrates this effect. There were also multiple variations of observational masking attempted, most notably testing between the separate binary mask observation and zeroing the stock observation upon reselection, but neither made a significant improvement to the termination rate. These training datasets are also accessible in the appendix.

What proved to make a major difference, however, was not to mask the observation, but to mask the action. As the charts of **Figure 8.3.1** indicate, this change led to a significant spike in the termination rate at around 1,000,000 steps, which generally managed to stabilize well in the FEA trainings. This did raise a concern regarding the general learning comprehension of the agent, however, the NON-FEA variants generally managed a high degree of explained variance throughout training, but with a notable initial dip once the termination rate started to increase, see **Figure 8.3.2**.

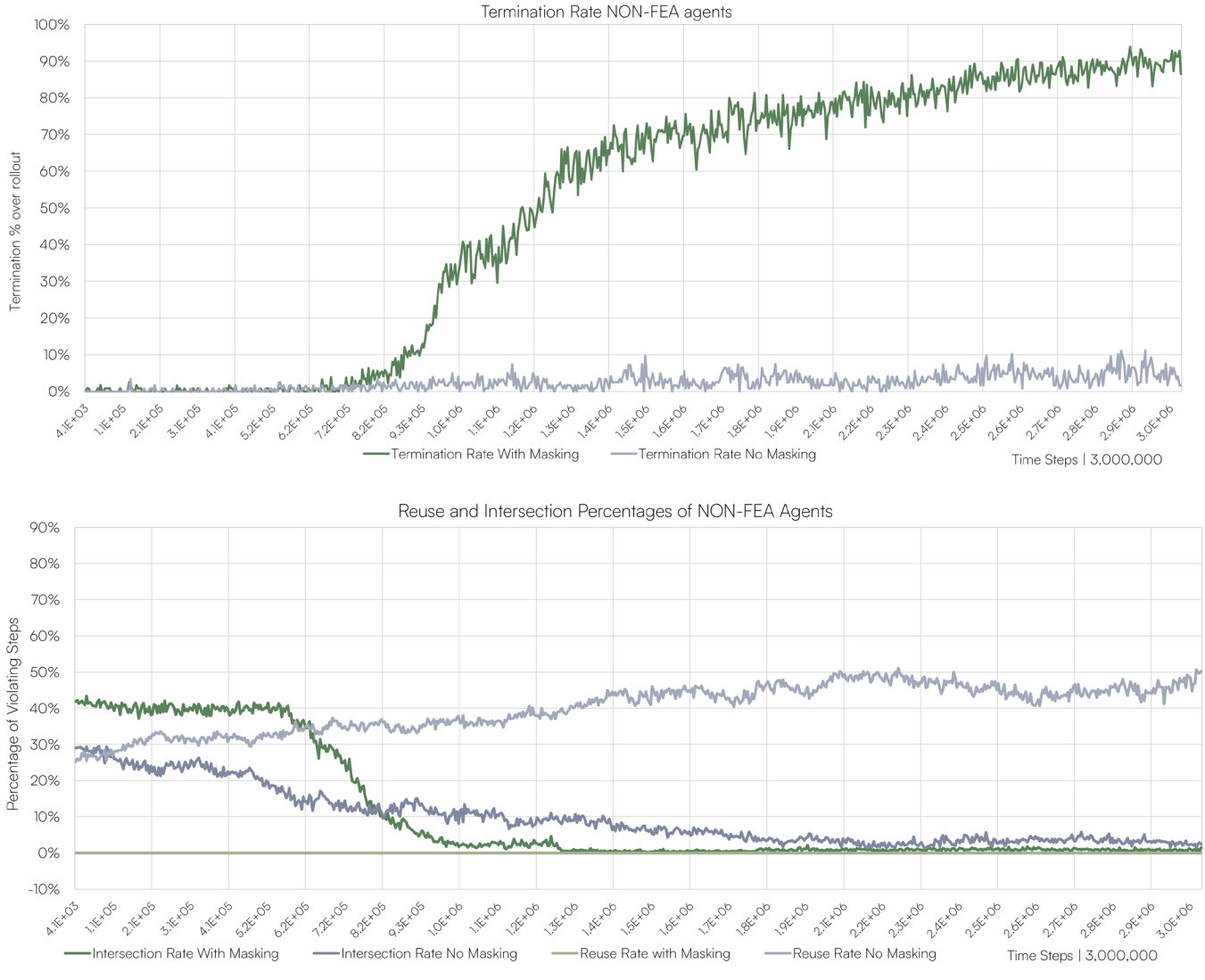


Figure 8.3.1 - *Termination, intersections and reuse percentages of NON-FEA agents* - Own work

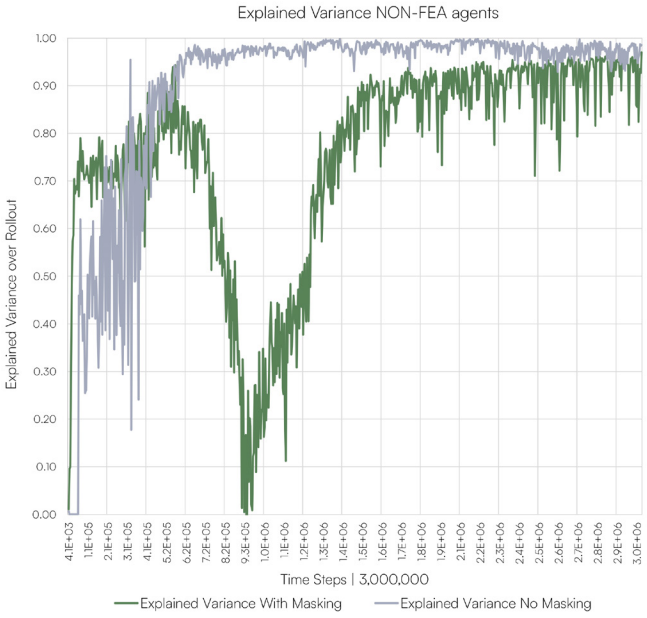


Figure 8.3.2 - *Explained Variance NON-FEA agents* - Own work

FEA Agents

For the FEA agents, two main agents have been outlined. These agents have been trained on the settings as prescribed through the methodology. The tributary width has been set to 0.50 metres, six arches, as it is the median of the values tested in the genetic validation. The difference between the two agents resides in the use of entropy. It was found that the initial agent, agent 1, plateaud to scores just above the minimum requirements for both deformation and fibre stress values, being 17.50 millimetres, 62.1 kN/cm2 and 48.3 kN/cm2 for deformation, tension and compression respectively. Whilst agent 1 demonstrates the ability to find a policy which can optimize for structural performance, the extended plateau of the training data suggested that it was exploiting too early. As such, a second training was performed with an entropy coefficient of 0.01. It was found through testing that values beyond 0.015 degraded termination as a whole. Generally, agent 2 manages marginal improvements over agent 1 in terms of FEA performance.

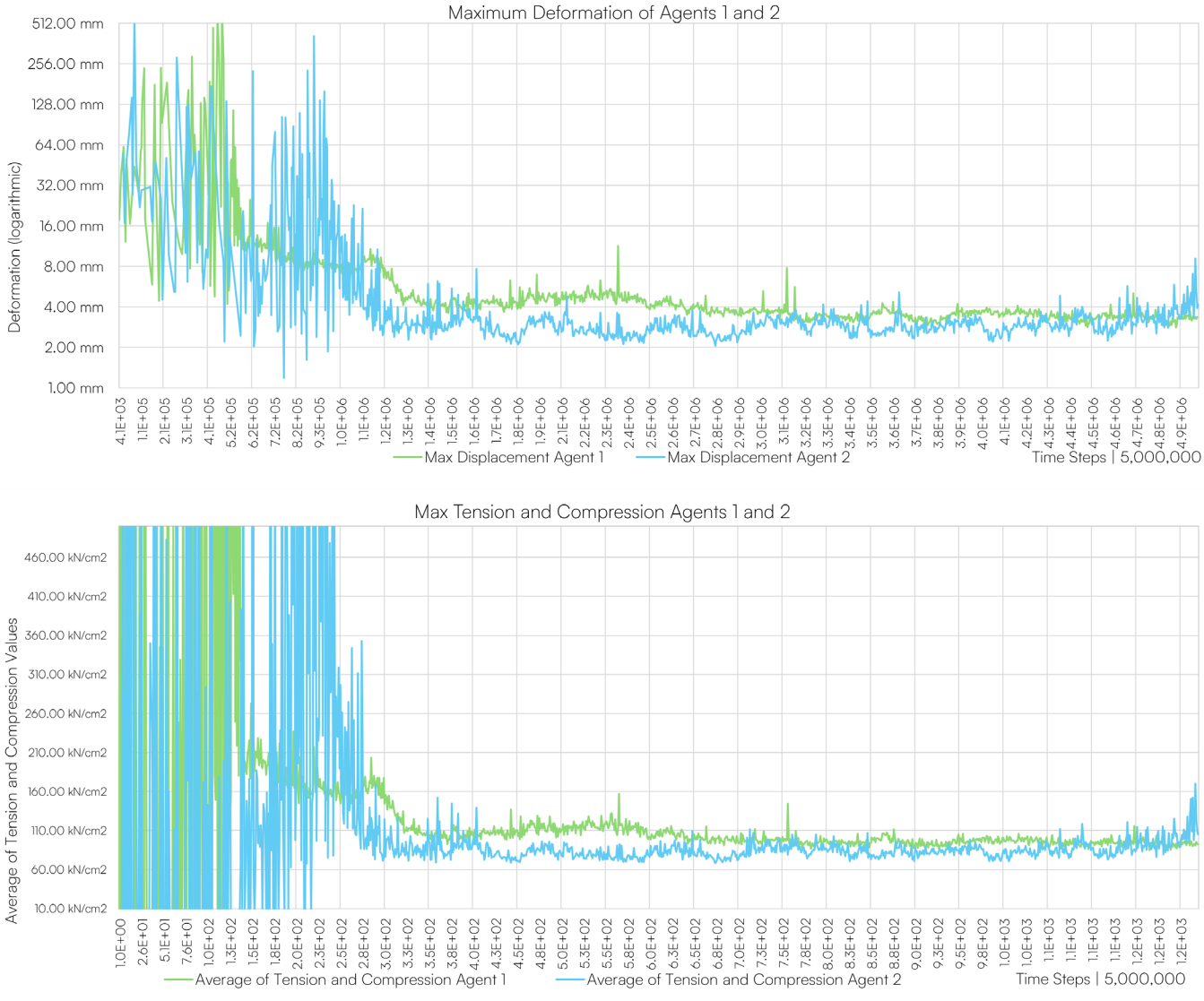


Figure 8.3.3 - *Deformation, tension and compression scores of Agent 1 & Agent 2* - Own work

Critically, neither agent demonstrates a high degree of explained variance, which suggests that neither has formed a policy which consistently predicts the structural performance, as the non-FEA agent manages to both hold a high termination rate and an explained variance around 0.90, meanwhile, agent 1 and 2 hold an explained variance around 0.4, which also osscilates rapidly across the different update cycles, see **Figure 8.3.6**.

FEA Agent Deployment

To assess the performance of the trained agents, each is deployed into the environment with the same settings as with which they were trained, including the shuffling of stock on each step. The agents were evaluated for 1000 episodes, operating deterministically - meaning it picks the best determined action at each observation, as opposed to drawing from a distribution.

Agents are evaluated for the entire set of criteria, meaning: Total deformation, maximum tensile stress and compressive stress of the frames, connections and tension cables. The tension cable criteria proved to be especially difficult to enforce, and as the agents were not directly rewarded for said criteria, a second evaluation is also done, only checking the requisite deformation, tensile and compressive stress of the frames themselves. Beyond agents 1 and 2, several other variants have been included in **Table 8.3.7**. The tributary width is still set to 0.50 metres, six arches.

Agent 2 remains to be the best performing one, though agent 5, being an alternate seeding of agent 4, displays similar performance.

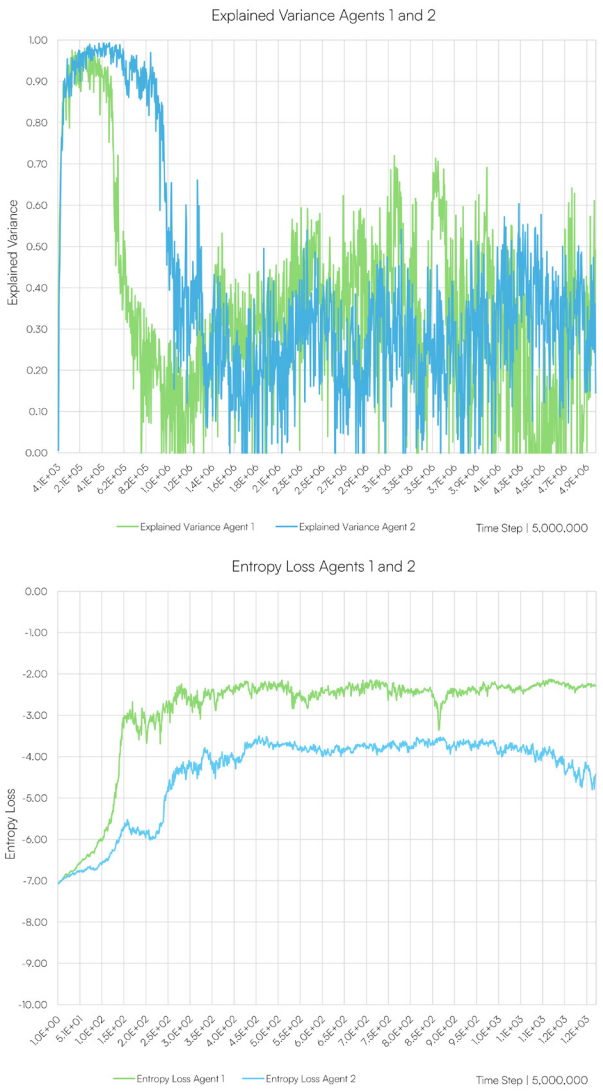


Figure 8.3.4 - *Explained Variance, Entropy Loss - Agents 1 & 2* - Own work

Agent	Settings	Termination Percentage	FULL FEA	PARTIAL FEA
Agent 1	Methodology Settings 5 million total steps	85.2%	0.0%	0.2%
Agent 2	Entropy coefficient of 0.010 5 million total steps	57.0%	4.7%	11.5%
Agent 3	FEA Maximum reward lowered (30>24) 3 million total steps*	68.2%	0.9%	1.3%
Agent 4	FEA Maximum reward lowered (30>24) Entropy coefficient of 0.015 5 million total steps	63.4%	1.1%	5.2%
Agent 5	Alternative seed of agent 4 8 million total steps	85.4%	4,4%	10.5%
Agent 6	Alternative seed of agent 4 8 million total steps	79.4%	0.2%	7.5%

Table 8.3.5 - *Deformation, tension and compression scores of Agent 1 & Agent 2* - Own work

Visual Analysis

To assess the generated geometries visually, a set of both agent 2's and agent 5's valid geometries has been outlined here and on the next page. Whilst agent 5 performs slightly worse in terms of how frequently it manages to generate FEA compliant designs, it does appear to follow the perscribed curvature much more closely. Agent 2, meanwhile, portrays a tendency to iteratively connect frames between the down tube and seat stay, this might be advantageous if the difference between the frames produces the right curvature, but in other cases, such as the geometry generated by agent 2 in the bottom left corner, the curvature ends up sitting inversly to the design gesture.

Interactive Capacity

To demonstrate the potential interactive behaviour, agent 2 and agent 5 have been tasked with completing a partial structure, consisting of 3 or 6 placed frames, derived from one of the GA optimized structures. **Table 8.3.10** contains the performance of both agents, tested for 1000 episodes. Agent 5 notably did not manage to terminate. Agent 2 did, however, manage to pass the complete FEA check for the state with 3 frames placed, and a partial FEA check for the 6 placed frames, see **Figures 8.3.8** and **8.3.9**.

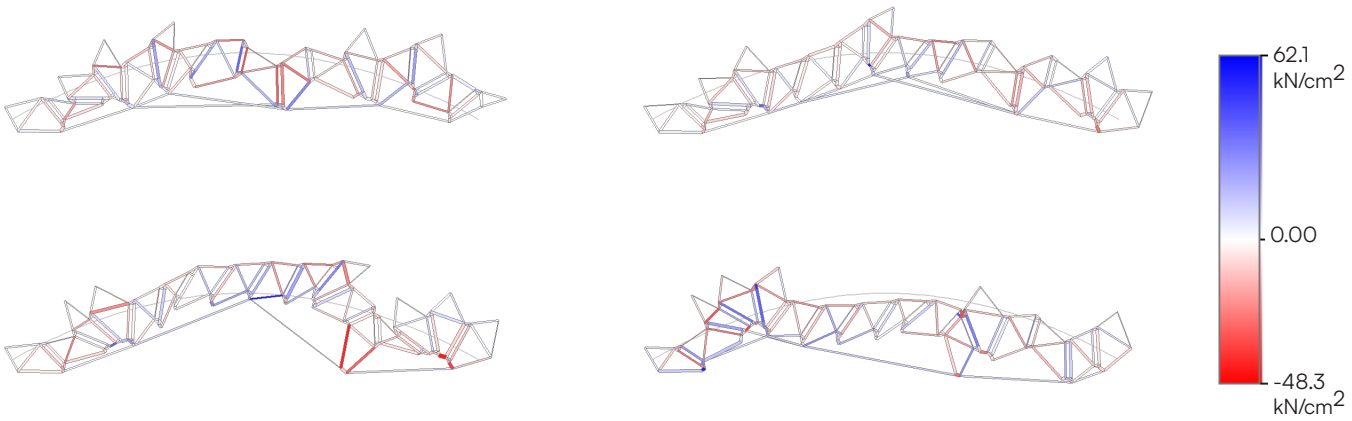


Figure 8.3.6 - *Visual Examples from Agent 5* - Own work

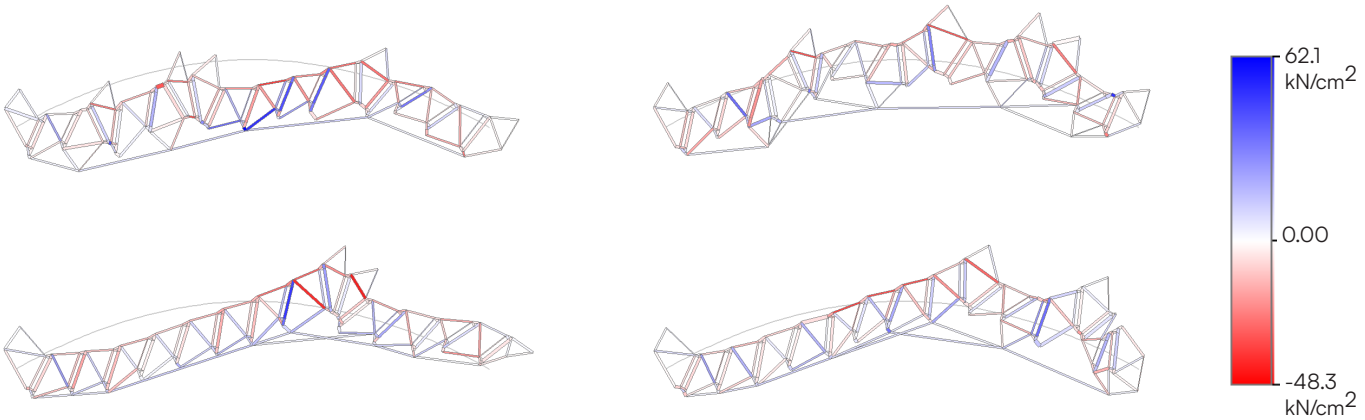


Figure 8.3.7 - *Visual Examples from Agent 2* - Own work

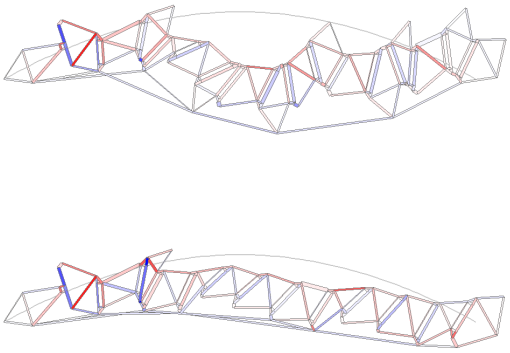


Figure 8.3.8 - *Visual Examples Partial Completion Agent 2, Three Frames Placed* - Own work

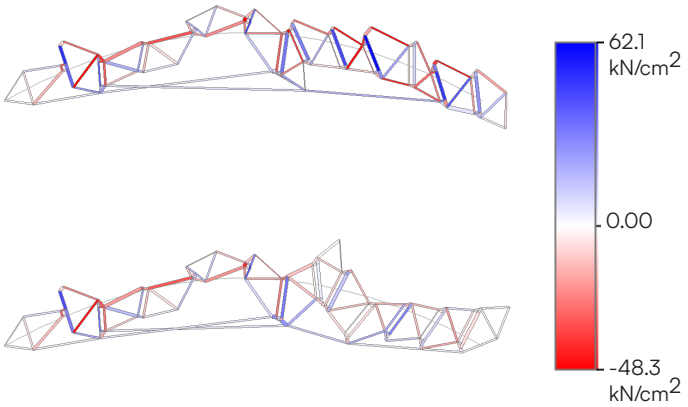


Figure 8.3.9 - *Visual Examples Partial Completion Agent 2, Six Frames Placed* - Own work

Generalized Capacity

To assess the degree to which the agents are capable of generalizing to data on which they have not been trained, each agent is ran for 1000 episodes with a variety of different datasets, each one still being extracted from the larger FRAMED dataset. The first two tests are conducted with random extraction seeds but otherwise the method of extraction is unchanged. The other three tests are conducted on datasets which have been extracted from only one of the K-Means clusters.

As **Table 8.3.16** shows, the agents perform quite well on the first four tests, but struggle most with the final cluster test. This cluster is arguably the most different compared to the previous two, see **Figure 8.3.11**.

Agent	% Terminated	% Valid Partial FEA	% Valid Full FEA
Agent 2 3 Frames	57.2%	16.8%	5.6%
Agent 2 6 Frames	53.4%	4.7%	0.0%
Agent 5 3 Frames	0.40%	0.0%	0.0%
Agent 5 6 Frames	1.70%	0.0%	0.0%

Table 8.3.10 - *Performance for Partial States* - Own work

Preliminary Conclusions

This chapter set out to train a reinforcement learning agent that optimises structural assemblies based on the design gesture and FEA criteria. The theoretical framework outlined reinforcement learning, its main algorithms and PPO in particular. Prior research applied reinforcement learning only to linear elements, not to a stock of non-linear elements with varying cross-sections.

The methodology adapted the design algorithm into a Gymnasium environment trained with PPO. The action space represents the rules of the shape grammar, and the reward combines the distance, progress, termination and FEA rewards, with penalties for reselection and intersection. Masking consumed frames in the action space proved decisive for reaching a high termination rate. The FEA agents learned to optimise structural performance, with agent 2 performing best. However, an explained variance of around 0.4 suggests that neither agent formed a policy that consistently predicts structural performance. Agent 2 completed partial structures in the interactive tests, and the agents generalised well to four of the five unseen datasets.

Together, these results address the chapter's research question on whether a reinforcement learning agent can use the material-first generative algorithm to produce optimised designs.

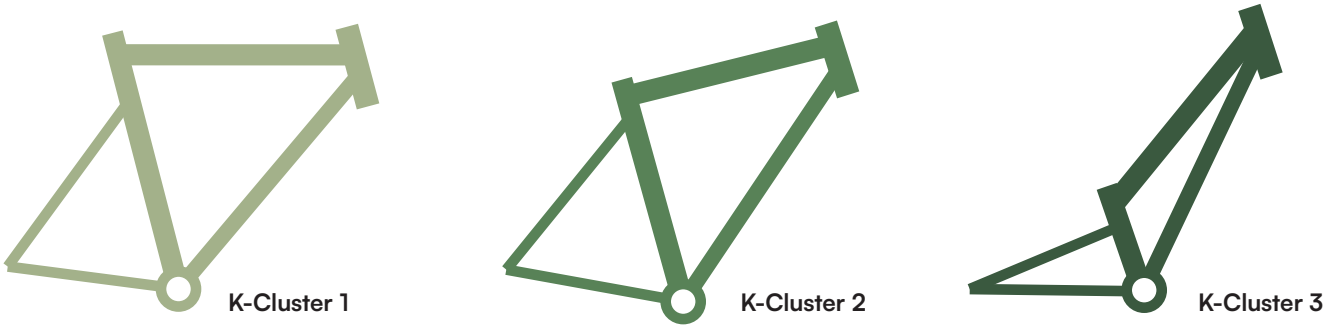


Figure 8.3.11 - *K-Cluster Examples* - Own work

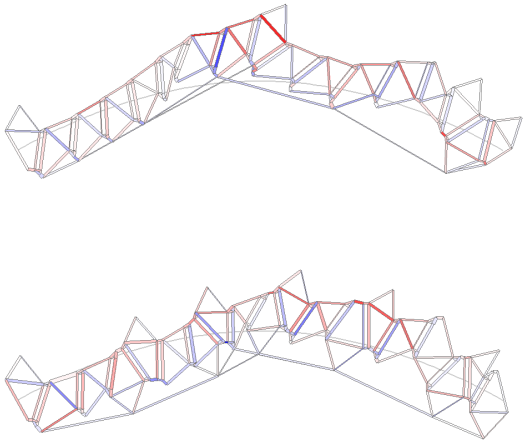


Figure 8.3.12 - *Visual Examples Alternative Seeding Agent 1* - Own work

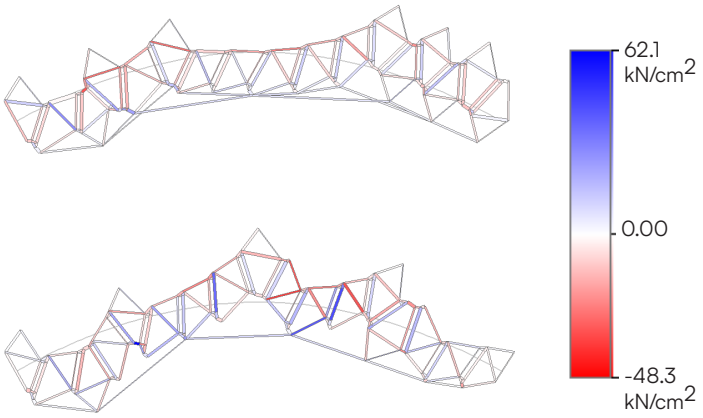


Figure 8.3.13 - *Visual Examples Cluster 1 Agent 1* - Own work

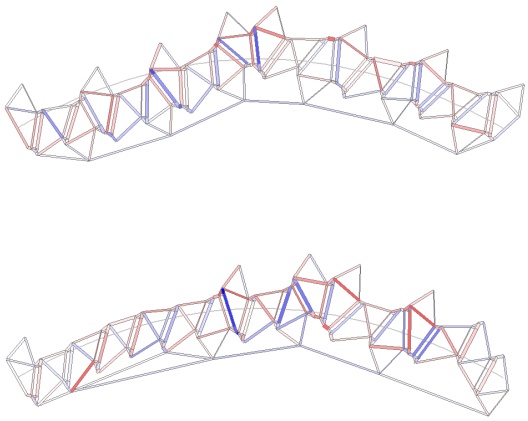


Figure 8.3.14 - *Visual Examples Cluster 2 Agent 1* - Own work

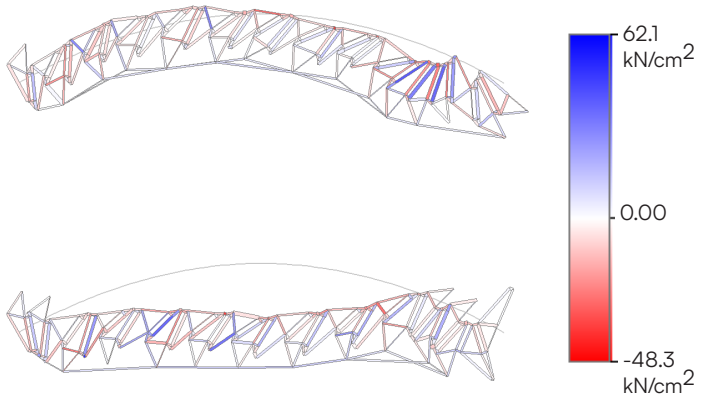


Figure 8.3.14 - *Visual Examples Cluster 3 Agent 1* - Own work

	% Terminated	Agent 2 Full FEA	Agent 2 Partial FEA	% Terminated	Agent 5 Full FEA	Agent 5 Partial FEA
Baseline	57.0%	4.7%	11.5%	85.4%	4.4%	10.5%
Alternate Seed 1	61.5%	8.8%	12.1%	86.4%	5.6%	12.7%
Alternate Seed 2	62.2%	4.2%	11.5%	85.0%	5.3%	11.7%
Cluster 1	69.9%	5.2%	8.5%	82.9%	8.2%	13.3%
Cluster 2	66.5%	2.5%	7.6%	92.3%	7.1%	17.8%
Cluster 3	2.1%	0.1%	0.2%	21.1%	0.0%	0.2%

Figure 8.3.16 - *Generalized Performance* - Own work

9.0 CONCLUSIONS

Drawing on the findings presented in each chapter, the report's key conclusions can now be formulated. To this end, the overarching method flowchart has been provided below, as well as the research questions themselves. Each sub question will be addressed, after which an answer is provided for the main research question, which is given as follows:

How can a constrained stock of repurposed elements be utilized for structural designs through a generative algorithm, implemented as an interactive design tool which enables material-first design solutions?

Manifesto of Material-First Computational Design

The research began with a question of what to design. The selection of both a material case study and design case study, being bicycle frames and cycling bridges respectively, were given as a predetermined scope. As such, a general design target could be argued: Structural assemblies. Still, it does not provide an answer as to how these assemblies can be formed from the constrained stock of elements. This led to the formulation of the following research question:

1. How can a material-first generative principles be formalized from the basis of a constrained stock of repurposed elements?

In part, the answer to this question is derived from the method, rather than the results. To this end, the manifesto of material-first design illustrated a concept which relies on generative design practices to create structural assemblies which do not make concessions to the particular sizing of the stock. With this in mind, design games were utilized, a concept borrowed from Mirra (2022). The intent was to emulate a '*child's mode of playing*' to then conceive of possible structural configurations without prescribing a preconceived target design. Through the theoretical framework, the use of shape grammars was determined as a formal basis to then define the generative principles.

In the end, the results of the design game experiments mostly served to outline the combinatory complexity of the stock considered. 3D implementations quickly turned into complex and unintuitive puzzles. For the sake of scope, the decision was therefore made to greatly reduce the complexity by adhering to a 2D principle based on perpendicular alignment of different frames along the centre points of their limbs, formalized as a shape grammar.

Material Case Study: Bicycle Frames

Having determined the base generative principles, the next step was to construct a coherent dataset by which to approximate a real-world stock. To this end, the following sub question was formulated:

2. How can a dataset be constructed to represent a constrained stock of repurposed elements for the purposes of material-first structural and generative design?

Two datasets were constructed. The first dataset utilized a randomized parametric model, whilst the second relied on the extraction of relevant data from a comprehensive database - FRAMED, of Regenwetter et al. (2022). This database was validated and filtered by a 95% mean of each element to remove frames of unrealistic proportions. Prior to the randomized subset extraction, the remaining database was partitioned via a K-mean clustering analysis, to preserve the diversity of shapes. 3 clusters have been created for this. The FRAMED dataset was most heavily used throughout the process.

For the structural aspects, certain assumptions had to be made. Each tube was treated as a hollow circular cross-section, and the materials were assumed through literature. Most significantly, the chain stay and seat stay elements were flattened in an effort to represent the frames as two-dimensional, doubling their cross-sectional area. For the generative design application, the frames were further simplified to 5 key points, so that they may be represented as a tuple of [5,2], each point expressed in X and Z coordinates.

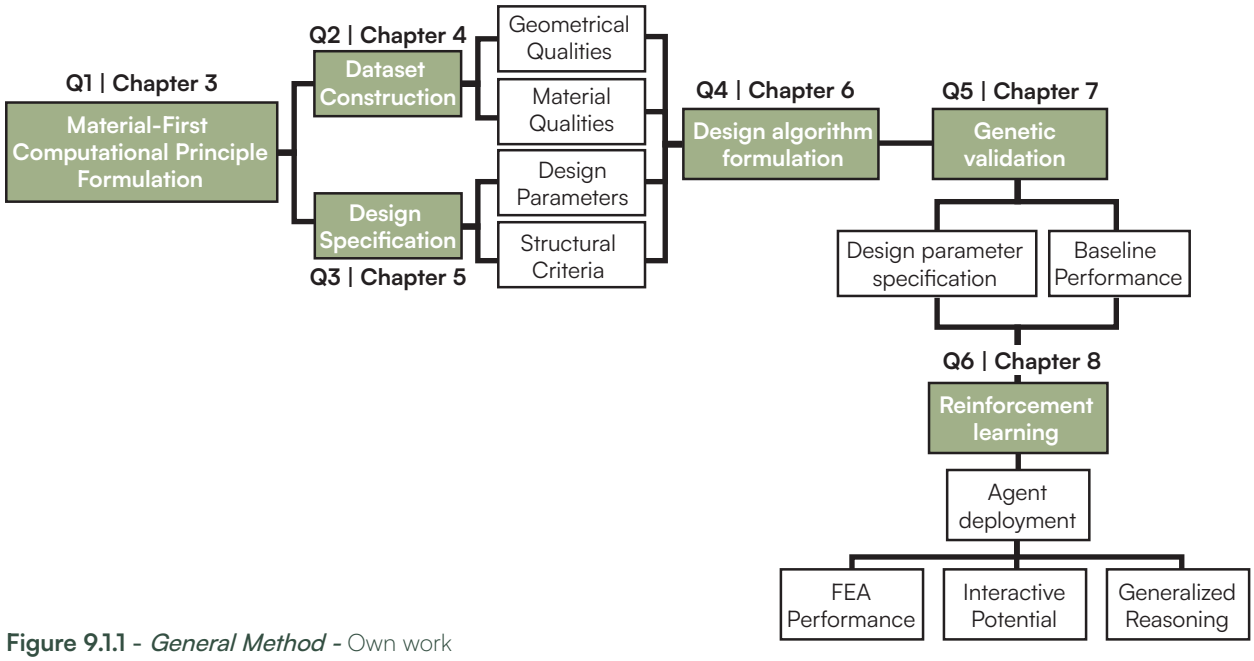


Figure 9.1.1 - *General Method* - Own work

Design Case Study: Cycling Bridge

Having established a coherent and elaborated dataset, the next step was to conceive of the design application itself, based on the preconceived scope of making a cycling bridge. To this end, the following sub-question was formulated:

3. How can the material-first generative principles be applied to a given design objective and which criteria need to be satisfied?

The decision was made to create arches using the defined generative principles of chapter 2, through which the deck of the cycling bridge could be supported. This introduced two design variables: The longitudinal arch count and the perpendicular arch count. These variables needed to be specified in later chapters. The design case itself was narrowed to the ZigZag bridge in Arnhem, which provided a manageable span over a non-active waterway, so that it would also be possible to use multiple smaller arches to support the bridge if need be. The dead load was specified through a optimization process through an FEA model of the bridge deck, to prevent excessive sizing of the different components. The load was simplified and rounded up to be 0.50 kN/m2. The live load was determined through the NEN-EN 1991-2 as 5.0 kN/m2. Applying the ULS conditions from NEN-EN 1990-1 yielded a total distributed load of 7.50 kN/m2. Of the three potential failure modes, only excessive deformation and structural collapse have been considered, buckling, however, was not.

Design Algorithm

Having conceived of the design intent, the next objective was to then conceive of the complete design algorithm, taking into account the generative principles, stock of materials and the intended design outcome. This raised the following question:

4. How is the material-first generative algorithm constructed from the generative principles, dataset and design objective?

To enable this, a design gesture was introduced in the form a flat curve, oriented in an XZ bounding area. A multidimensional action space was defined through the shape grammar formalism, as well as two metrics to monitor and eventually score the quality of frame placements. Passive logics were introduced to connect the frames, determine the support points and placement of tension cables to aid with the distribution of forces. Structural metrics were constrained to the testing of vertical forces, and a set of criteria was given for to guard against excessive deformation and structural collapse. This was further expanded into two FEA checks, partial and full. The partial check reflects the metrics for which the algorithm is actually optimized, the full check also considers the performance of the connections and tension cables.

Genetic Validation

Prior to committing to a reinforcement learning model, the algorithm was first tested through genetic optimization. The intent being to verify that the design algorithm is capable of termination in the first place and if it can be optimized for structural performance in the second. Beyond this, the genetic validation also served to specifiy the design parameters of longitudinal and perpendicular arch counts, providing a baseline of feasible structural performance to measure the RL agents against. The research question of this chapter was formulated as follows:

5. Can a genetic optimizer successfully utilize the material-first generative algorithm, therefore validating the functionality of the underlying principles given the provided material and design context?

The algorithm was initially tested with the full action space represented. However, the search space that this configuration yielded proved to difficult to navigate even at extended optimization sessions. A variety of fitness functions were tested, but the algorithm only ever managed a termination rate of 16% in a given generation. To this end, an alternate routine was created which made use of a pre-selected valid and ranked set of possible orientations at each step, limiting the action space to frame selection and picking one of the possible valid orientations. This increased the termination rate sufficiently to enable actual structural optimization. A comprehensive set of multi-objective optimizations were performed with varying search space sizes, enabled by the number of ranked orientations of which the algorithm could select. The optimizer proved adapt at navigating the search space, even if the ranked selection was entirely unconstrained. Furthermore, each multi-objective optimization was also tasked with achieving the lowest number of perpendicular aches. From this, it was determined that at least 4 arches would be needed to yield adequate full FEA performance. The longitudinal arch count was tested with the highest number of perpendicular arches, with the search space that proved most optimal from the previous tests, which were conducted on a longitudinal arch count of 2. However, the testing revealed that less than 2 arches was not feasible, yielding structural performance well below even the partial FEA check.

Reinforcement Learning Application

Having validated the functionality of the design algorithm and the design parameters specified, the final step of the research consisted of the application of reinforcement learning, under the following sub question:

6. Can a reinforcement learning agent successfully utilize the material-first generative algorithm to produce optimized design solutions given the provided material and design context?

The training was performed using the Proximal Policy Optimization, making use of a custom extractor to embed the observations prior to submitting them to the main policy perceptrons. As with the genetic algorithm, it was firstly assessed if agents would be able to terminate outright. This proved difficult, as the policy struggled to stave of both the reusing of elements and placing them without causing intersections. As such, the decision was made to filter out previously placed frames from the action probability distribution, which yielded a stable rise in the termination rate, approaching 80% of rollout episodes after around 3 million steps of training. The explained variance showed an initial dip as the termination rate increased, but managed to remain fairly stable around 0,90 over the course of training.

Training of the FEA agents revealed a plateau in structural performance after 1-2 million total timesteps, whilst the explained variance was more prone to fluctuations and remained below 0.50 generally. That being said, the agents did manage to reach near 90% termination rates and notably agents 2 and 5 proved capable of passing the full FEA check, with 4.7% and 4.4% of episodes passing said check. However, the agents only managed this performance with a perpendicular arch count of 6, whilst the genetic baseline was able to achieve performance at arch counts of 4. The main difference between agents 2 and 5 compared to the others was in the usage of an entropy coefficient to encourage more exploration over the course on training.

The agents were also submitted to a partially completed environment to assess the capacity for interactions. Whilst these results are tentative, agent 2 managed to pass the full FEA check in an environment with 3 frames placed, whilst achieving the partial check in an environment with 6. Agent 5, notably, did not.

The agents were lastly tested for generalized reasoning capacity, by submitting different extractions from the stock dataset to them. Both agents performed close to baseline for the alternate extractions, with the exception of the extraction drawn from the third K-means cluster, which notably differed most strongly from the others.

Final Conclusions

How can a constrained stock of repurposed elements be utilized for structural designs through a generative algorithm, implemented as an interactive design tool which enables material-first design solutions?

Coming back to the main research question, the method and results described above yielded a generative algorithm which managed to achieve the creation of an interactive design tool capable of producing structurally acceptable design solutions via the implementation of Proximal Policy Optimization. Genetic optimization proofed to yield better structural performance, though the action space was much more constrained compared to the reinforcement learning variant.Furthermore, the trained FEA agents proved capable to a degree of interaction with a partially complete environment, could generalize to the dataset as a whole and yielded a more diverse set of solutions in a fraction of the time. These design solutions adhere to the material-first principle, they do not cut stock, they do not mediate with a target design model, but instead express designs directly from the constrained stock itself.

As such, this thesis has answered its main research question by formulating a novel approach to generative design with a constrained stock of repurposed elements, demonstrating potential for generating diverse solutions which adhere to structural criteria without making concessions to the material itself.

10.0 DISCUSSION

Positioning in the Current Landscape

This thesis set out to test whether a constrained stock of repurposed bicycle frames could be turned into structural designs through a generative algorithm. The algorithm was to be optimised first through genetic optimisation and then through reinforcement learning. The results answer this partially. The generative principle works. Both optimisers can produce structures that pass the structural criteria. However, neither does so reliably enough to be called a mature design tool. This discussion places these findings within the research outlined in the manifesto and the reinforcement learning chapter, examines what they mean for material-first design, and outlines the steps needed to move the work forward.

The manifesto placed this thesis within the bottom-up approach described by Huang et al. (2021). There, the stock informs the design, rather than a target design model being imposed on the stock. Most of the research reviewed in that chapter falls on the other side of this divide. The work of Brütting et al. (2020), Bukauskas et al. (2017) and Huang et al. (2021) themselves are parametric and top-down. In each case, stock cutting appears as a parameter to be minimised rather than eliminated. Furthermore, the bottom-up research of Eschenbach et al. (2023) and Allner et al. (2019) relies on some degree of refabrication to make the constrained stock of elements fit. Nian et al. (2022), whose work prompted the choice of bicycle frames, sought to create modular assemblies from them in an effort to overcome the variance of the stock.

Against this background, the central contribution of this thesis is that the frames are placed whole. They are not cut, trimmed or matched to a predefined module. This answers the gap identified in the manifesto: generative design aligns naturally with material-first principles, but contemporary work rarely avoids modification of the stock. However, research in the field of reinforcement learning proves to be an exception to this.

The reinforcement learning chapter reviewed a growing body of work on generative structural design models enabled through reinforcement learning. These researches are the inarguable precedent for a thesis like this one, reflecting the essence of material-first design. Ororbia & Warn (2022, 2023) described a design workflow as a Markov Decision Process solved through Q-learning, and later on DQN. Rajaskar et al. (2024) implemented graphic statics, as opposed to FEA, showing a promising alternate route towards a similar goal, one in which each step of the process mandates that the design remains in equilibrium. Mirra & Mueller (2025) trained a PPO agent to build a cantilevered truss, with a structural feedback signal at every time step. Sørensen (2024) is perhaps most closely aligned to this thesis, however, as they made use of a design gesture to create a truss configurations.

Where the thesis arguable differs, is in the consideration of the stock. Each of the aforementioned RL studies made use of 1D linear elements, some with only a variable length but otherwise consistent structural qualities. This thesis made use of 2D non-linear elements with a varied set of structural and dimensional qualities.

The Knowledge Gap

This thesis shares its shape-grammar formalism with Ororbia & Warn, its algorithm and stock-constrained premise with Mirra & Mueller, and its design gesture and terminal evaluation with Sørensen. Where it departs from all of them is the nature of the elements.

The true knowledge gap inherent to this thesis is two-fold but expressable either way through the consideration of the constrained stock:

For one, there is very little literature which is concerned with repurposing practices through generative computational design. Whilst the aforemetioned research in reinforcement learning aligns with the material-first computational design concept, it does not specifically consider repurposing.

Going further, the second knowledge gap resides not in the circular economic practice, but in the specific geometrical and structural qualities of the stock concerned. The shift from a linear element to a 2D shape with inconsistent material qualities greatly increases the complexity, especially in reinforcement learning. Whilst a linear element might be expressed through length and cross section inputs alone, the bicycle frames of this study were represented with 16 different inputs, even though the representation was already considerably simplified.

Contextualizing the results

This difference helps to contextualize the results. Whilst the non-FEA agents managed to achieve high rates of termination with a high rate of explained variance, the FEA-agents notably dropped in the second aspect. They struggled to predict the next state dependently, at this shows in the deployment. Even the best FEA agents managed only sub 10% adequate results over a span of 1000 episodes. Arguably, the agents have not learned a good model of the structural performance. It shows further in the visual evaluations, as the agents demonstrates a strong preference to repeat the same connections over and over. This is likely due to the fact that these connections yield the best transfer of forces. Still, the custom policy helped to improve performance, most strongly by funneling each observation through a separate MLP, which would indicate some capacity to estimate the structural performance. However, it is likely that the agents have tapped into a 'lucky' policy, as they show extended performance plateaus across training from which they do not recover, and none of the agents have managed to outperform the genetic baseline.

Next Steps

There are many potential avenues by which this research could continue. As such, a set of these has been outlined below. There is no preceived order of importance, but these steps do reflect where I would continue next.

Feature Engineering

The practice of feature engineering in reinforcement learning generally relates to the way in which the different aspects of the environment are observed by the algorithm. Sørensen (2024) provides some fascinating ways in which this is achieved, and these methods could prove to unlock genuine increases in the performance of the agents. Firstly, Sørensen (2024) represents their design gesture much differently from this thesis, using a lookahead window of three points along the curve, as opposed to representing the entire curve as a passive observation. In this way, the entire design gesture can be represented in just 6 inputs, as opposed to the 200 inputs used currently. Secondly, Sørensen (2024) made the choice to bin the stock of items, grouped by length, instead of representing each item as a separate object with a separate action. This could prove useful in several ways. For one, the current agents demonstrate difficulty with the comprehension of the third cluster of stock elements. This clustering could be used to represent the actions, reducing both the action space significantly and aiding with the comprehension of the variety of stock elements. Secondly, by providing bins of elements rather than separate ones, the agents might be more adept at learning that items cannot be reused, as they do not have to learn this for each item separately.

Connection Typologies

The connections between the frames were purposefully simplified in this thesis, as they represent a complicated design problem in their own right. The question of how to go about connection frames with not only varying diameters of tubes, but also varying cross sectional shapes, is a difficult one to answer. Nevertheless, if a full fabrication pipeline would be researched, then this aspect is perhaps most critical. There is potential in additive manufacturing, for example. However, it would perhaps be even more interesting to have non-custom connections, which make use of clamping or sleeving to flexibly be implemented but maintain a good transference of forces.

Connections are especially critical when the bicycle frames are considered in a 3D perspective, as the chain stay and seat stay have been flattened in this thesis to simplify the geometry. This would also warrant further testing from a structural perspective.

Stepwise FEA

Mirra & Mueller (2025) approach the reward signal for their algorithm differently, rewarding for structural performance on every step, as opposed to the end of episode reward of this thesis. This makes for a much more receptive and dense reward signal, which could potentially improve the performance of the agent as well as the explained variance metric. However, this would mean that the objective needs to be structured differently, either as a cantilevered truss or as a kind of bridge-building operation.

Life Cycle Analysis

The thesis operates under the preconceived notion that repurposing as a practice could prove to be less carbon intensive than other practices, provided that the materials are more readily accessible to the design location. However, without an elaborated fabrication pipeline it would be difficult to actually assess whether or not this is true. Brütting et al. (2020) asserted in their own research that depending on the actual case, virgin materials might prove more carbon efficient than reuse practice. This illustrates that there are a variety of concerns which need to be addressed if one is to prove that a design method such as this would actually have a net-positive on carbon expenditures in the built environment. By conceiving of a potential fabrication pipeline, an LCA might be performed to assess if the design strategy described in this thesis would actually have a real world benefit.

Database Re-evaluation

One major knowledge gap flagged in the introduction is on the waste stream flow of bicycle frames in the Netherlands. There are no clear quantities known, nor in what state the bicycles are when they are disposed of. As such, the stock has been represented via the FRAMED database, but these designs do not reflect industry standards necessarily. A potential avenue to pursue would be to look at point-cloud scanning or image detection to conceive of a real-to-digital pipeline.

Structural Analysis

On the side of the structural analysis, a variety of additional metrics could still be implemented for the optimization process. For one, buckling has not been taken into account for this thesis. This does raise further questions about how to consider the clamped length of the arches. It could be argued that the deck provides sufficient rigidity, reducing the likelihood of buckling, but this should be investigated.

Beyond just buckling, a variety of loads could also be further included, such as the point load, service vehicle load, horizontal load, collision load and snow load described by NEN-EN 1991-2. Again, the horizontal loads warrant consideration of the structure as a whole, rather than the simplified 2D model.

Agent deployment on a 3D structure

A more immediate potential next step resides in the interactive capacity of the agents, and the design intent as a whole. Whilst an agent can only generate 2D arches, multiple agents could feasibly be deployed into a 3D environment to generate the entire bridge structure. Furthermore, instead of having a designer interact with the agents, it could also be possible to have a genetic algorithm interact with them, optimizing the placement of a few frames and having the agent complete the structure instead. This would greatly reduce the search space for the algorithm itself.

Design Tool: Intuitiveness

Currently, the design tool does not exist in a proper sense. Whilst the basic functionality has been demonstrated, to actually translate the agents to a proper program to be used by a designer requires more consideration. How would the interface work? How intuitive is the interaction? How much say do designer and agent actually get over the final shape?

11.0 REFLECTION

The original conceived scope of the thesis was considerably larger than the final result — this is attributable to both personal ambition and lack of insight. The latter specifically referring to the insight into reinforcement learning, which has consumed about half of the time invested into the thesis, also considering the extension which was taken. This is perhaps a good avenue via which to reflect on the specific nature of the research process. At the original A3 date, the option was offered to take an extension, the intent being to improve and indeed finalize the report, and therefore receive an evaluation which better reflected the size of this research project. What had happened, in truth, is that I was still running agent trainings very close to the deadline. I was not satisfied with the results, and felt I had a poor grasp on my own codebase. As such, too little priority was taken for the report itself, and the state in which it was delivered was not acceptable. Furthermore, it was not until the A3 presentation itself that my tutors informed me of a major logical fault in my training approach — the lack of randomization between training episodes, which caused every agent I had trained to 'cheat' in essence. Now, it should be clearly stated that this was completely my own fault, and as such, it does beg the question of how it came to be that I had missed this quite fundamental step. This brings me to my main subject which I have chosen to discuss in my reflection: The use of LLMs in (undergraduate) research, such as I have used it.

Large Language Models in Computational Design Projects

I want to clearly convey my approach to writing, and the degree to which I make use of Large Language Models. My philosphy with the usage of LLMs is to only ever reduce. What this means, is that I never permit an LLM to produce text for me outright, only ever to provide summaries of my own work, or the occasional suggestion on how to phrase something. Even then, I find that LLMs often both miss the mark when they write summaries, but even more worrisome, are capable still of fabricating falsehoods. As such, I rarely actually use LLMs for writing. I believe writing is a critical skill, especially so in the research process. It is not merely a tool to document or communicate; it is also a kind of dialogue with the self. Often times, it is only when we write thoughts down that we discover aspects not previously considered. In the way that van Dooren et al. (2014) talks about sketching as a language for design, so goes the same for writing as a language through which to think.

That being said, there has been a different part of my research process where I did use LLMs extensively: The coding and development of the reinforcement learning routine. Large language models are like genius friends who always want to help, but who can also lie, misunderstand and altogether become misaligned with your own intentions. But beyond that, large language models lack certain basic comprehensions that we take for granted. Artificial intelligence at large seems often like a facsimile of our own intelligence, yet time and time again we become aware that for each of the incredible strengths that it has, it also bares major weaknesses. A major example which I have noticed is the lack of basic spatial comprehension. My entire thesis revolves around spatial comprehension, and how to manifest something like it in an RL agent. It is not something that large language models understand well, and whenever the conversation came to a more high-level discussion of the thesis, or if the model chose to visualize something we would be discussing, it would become readily apparent how little it understood the basic generative design concept. This is something I am thankful for, because it does keep me sharp. I cannot overtly trust or expect that the LLM will comprehend prompts which lack explicit instructions on how data should actually be treated.

Nevertheless, LLMs are like rocket fuel to research such as this, for better and for worse. I honestly do not think I could have managed this level of scope without access to large language models, and yet I've also come to notice how tunnel-visioned we can become because of them. In retrospect, I think that the LLM I used gave me too much confidence at points in the research where I would have otherwise asked for help. There have been many choices I made through intuition in the creation of my code base. I wonder if I would have taken the same approach if there wasn't an LLM to do as I instruct it, but another person with their own insights and perspective. LLMs are a double-edged sword, and we don't yet know which side cuts deeper.

Contextualizing in Other Research

This research is unique in several aspects. There aren't many master theses in the built environment which focus on reinforcement learning, let alone ones which take such a radical reuse concept as the basis for it. As far as I can tell, mine is the first of its kind. I don't state this as a positive, but rather as something which makes it difficult to reflect on the process in the context of the current research approaches. The closest ones which I could identify were the research of Mirra & Mueller (2025) and Sørensen (2024). Both of these authors used Proximal Policy Optimization to create structural assemblies from finite and diverse sets of elements — but their elements were effectively one-dimensional, represented with three or fewer scalars. My elements were two-dimensional, each object effectively expressed via ten scalars for size and six for cross-sectional areas. If I could start this thesis again, I would put more focus on the observation engineering, to represent my state space more discretely, especially as it comes to the elements themselves.

Reflecting on the Extension

During my extra time, I decided to redo my entire codebase from scratch. I had even hoped to include more aspects than I finally managed, but I am very glad that I took this route, instead of simply finalizing my report. Again, this comes back to the use of large language models. The old codebase felt rushed, poorly understood. Too many decisions I had subconsciously made for the sake of progress, and this extra time allowed me to take a calmer look at everything. I am very thankful towards my tutors for their insights, patience and genuine enthusiasm, but I am perhaps most thankful that they gave me this extra time. I now feel much more confident and proud of the work I have been able to document in this report, making the conclusion of my efforts much more satisfying.

12.0 REFERENCES

Admin, R. (2018, April 9). Structural Cable Options — Stainless Steel & Galfan Coated Steel. Ronstan Tensile Architecture. https://www.ronstantensilearch.com/structural-cable-options/

AlJaber, A., Martinez-Vazquez, P., Baniotopoulos, C., AlJaber, A., Martinez-Vazquez, P., & Baniotopoulos, C. (2023). Barriers and Enablers to the Adoption of Circular Economy Concept in the Building Sector: A Systematic Literature Review. Build-ings, 13(11). https://doi.org/10.3390/buildings13112778

Allner, L., Kroehnert, D., & Rossi, A. (2019). Mediating Irregularity: Towards a Design Method for Spatial Structures Utilizing Naturally Grown Forked Branches (p. 445). https://doi.org/10.1007/978-3-030-29829-6_34

Ansys Granta EduPack | Software for Materials Education. (n.d.). Retrieved 18 October 2026, from https://ansys.synopsys.com/products/materials/granta-edupack

Benachio, G. L. F., Freitas, M. do C. D., & Tavares, S. F. (2020). Circular economy in the construction industry: A systematic literature review. Journal of Cleaner Production, 260, 121046. https://doi.org/10.1016/j.jclepro.2020.121046

BNR. (2018, September 24). Meeste verwijderde fietsen worden nooit meer opgehaald. bnr.nl. https://www.bnr.nl/nieuws/mobiliteit/10354902/meeste-verwijderde-fietsen-worden-nooit-meer-opgehaald

Brütting, J., Vandervaeren, C., Senatore, G., Temmerman, N., & Fivet, C. (2020). Environmental impact minimization of re-ticular structures made of reused and new elements through Life Cycle Assessment and Mixed-Integer Linear Program-ming. Energy and Buildings, 215, 109827. https://doi.org/10.1016/j.enbuild.2020.109827

Bukauskas, A., Shepherd, P., Walker, P., Sharma, B., & Bregulla, J. (2017). Form-Fitting Strategies for Diversity-Tolerant De-sign.

Caetano, I., Santos, L., & Leitão, A. (2020). Computational design in architecture: Defining parametric, generative, and algo-rithmic design. Frontiers of Architectural Research. https://doi.org/10.1016/j.foar.2019.12.008

CBS. (2023). Hoeveel fietsen inwoners van Nederland? [Webpagina]. Centraal Bureau voor de Statistiek. https://www.cbs.nl/nl-nl/visualisaties/verkeer-en-vervoer/personen/fietsen

Cheng, Y., Lee, C., & Tsai, M.-T. (2016). Multi-objective optimization of an on-road bicycle frame by uniform design and com-promise programming. Advances in Mechanical Engineering, 8. https://doi.org/10.1177/1687814016632985

Circulaire brug ZigZagBrug Arnhem—Arc2 architectuur. (n.d.). Retrieved 18 October 2026, from https://www.bruggenarchitectuur.nl/projecten/voetbrug/zigzagbrug-arnhem/

circulaire voetbrug Almere Haven—Arc2 architectuur. (n.d.). Retrieved 18 October 2026, from https://www.bruggenarchitectuur.nl/projecten/circulair/circulaire-voetbrug-almere-haven/

Covill, D., Begg, S., Elton, E., Milne, M., Morris, R., & Katz, T. (2014). Parametric Finite Element Analysis of Bicycle Frame Geometries. Procedia Engineering, The Engineering of Sport 10, 72, 441—446. https://doi.org/10.1016/j.proeng.2014.06.077

Davidor, Y. (1991). Epistasis Variance: A Viewpoint on GA-Hardness. In Foundations of Genetic Algorithms (Vol. 1, pp. 23—35). Elsevier. https://doi.org/10.1016/B978-0-08-050684-5.50005-7

Deep Q-Learning in Reinforcement Learning. (2026, July 7). GeeksforGeeks. https://www.geeksforgeeks.org/deep-learning/deep-q-learning/

EMF. (2013). Towards the circular economy Vol. 1: An economic and business rationale for an accelerated transition (To-wards the Circular Economy).

EMF. (2023, March 23). Schools of thought that inspired the circular economy. https://www.ellenmacarthurfoundation.org/schools-of-thought-that-inspired-the-circular-economy

EMF. (2026). Circular economy introduction. https://www.ellenmacarthurfoundation.org/topics/circular-economy-introduction/overview

Eschenbach, M. B., Wagner, A.-K., Ledderose, L., Böhret, T., Wohlfeld, D., Gille-Sepehri, M., Kuhn, C., Kloft, H., & Tessmann, O. (2023). Matter as Met: Towards a Computational Workflow for Architectural Design with Reused Concrete Components. In C. Gengnagel, O. Baverel, G. Betti, M. Popescu, M. R. Thomsen, & J. Wurm (Eds), Towards Radical Regeneration (pp. 442—455). Springer International Publishing. https://doi.org/10.1007/978-3-031-13249-0_35

Favilli, A., Laccone, F., Cignoni, P., Malomo, L., & Giorgi, D. (2025). Optimizing Free□Form Grid Shells with Reclaimed Elements under Inventory Constraints. Computer Graphics Forum, 44. https://doi.org/10.1111/cgf.70047

Fietsberaad Crow. (2024a, January 24). Steeds meer fietsen worden verwijderd—Fietsberaad. Fietsberaad Crow. https://www.fietsberaad.nl/Kennisbank/Er-worden-steeds-meer-fietsers-verwijderd

Fietsberaad Crow. (2024b, November 7). Aantal fietsen blijft stijgen, aantal kilometers vlakt af—Fietsberaad. Fietsberaad Crow. https://www.fietsberaad.nl/Kennisbank/Aantal-fietsen-blijft-stijgen,-aantal-kilometers-v

Ghisellini, P., Cialani, C., & Ulgiati, S. (2016). A review on circular economy: The expected transition to a balanced interplay of environmental and economic systems. Journal of Cleaner Production, Towards Post Fossil Carbon Societies: Regenera-tive and Preventative Eco-Industrial Development, 114, 11—32. https://doi.org/10.1016/j.jclepro.2015.09.007

Ghisellini, P., Ripa, M., & Ulgiati, S. (2018). Exploring environmental and economic costs and benefits of a circular economy approach to the construction and demolition sector. A literature review. Journal of Cleaner Production, 178, 618—

643. https://doi.org/10.1016/j.jclepro.2017.11.207

Ginga, C., Ongpeng, J., & Daly, M. (2020). Circular Economy on Construction and Demolition Waste: A Literature Review on Material Recovery and Production. Materials, 13. https://doi.org/10.3390/ma13132970

Guan-feng, L., Yang, J., Li-feng, F., Xing, X., Di, W., & Xi-kang, Z. (2023). Constitutive model of 25CrMo4 steel based on IPSO-SVR and its application in finite element simulation. Materials Today Communications, 35, 106338. https://doi.org/10.1016/j.mtcomm.2023.106338

Huang, Y., Alkhayat, L., De Wolf, C., & Mueller, C. (2021). Algorithmic circular design with reused structural elements: Meth-od and tool. 457—468. https://doi.org/10.35789/fib.PROC.0055.2021.CDSymp.P056

Illankoon, C., & Vithanage, S. C. (2023). Closing the loop in the construction industry: A systematic literature review on the development of circular economy. Journal of Building Engineering, 76, 107362. https://doi.org/10.1016/j.jobe.2023.107362

Introduction To Neural Networks. (2026, August 31). GeeksforGeeks. https://www.geeksforgeeks.org/deep-learning/neural-networks-a-beginners-guide/

Kalmykova, Y., Sadagopan, M., & Rosado, L. (2018). Circular economy — From review of theories and practices to develop-ment of implementation tools. Resources, Conservation and Recycling, Sustainable Resource Management and the Circu-lar Economy, 135, 190—201. https://doi.org/10.1016/j.resconrec.2017.10.034

Kishna, M., & Prins, A. G. (2024). Monitoring circularity strategies. Principles for application by PBL (No. 5586). © PBL Neth-erlands Environmental Assessment Agency.

Knight, T., & Stiny, G. (2015). Making grammars: From computing with shapes to computing with things. Design Studies, Special Issue: Computational Making, 41, 8—28. https://doi.org/10.1016/j.destud.2015.08.006

Lee, J., & Ostwald, M. (2020). Creative Decision-Making Processes in Parametric Design. Buildings, 10, 242. https://doi.org/10.3390/buildings10120242

Leising, E., Quist, J., & Bocken, N. (2018). Circular Economy in the building sector: Three cases and a collaboration tool. Journal of Cleaner Production, 176, 976—989. https://doi.org/10.1016/j.jclepro.2017.12.010

Meadows, D. H., Meadows, D. L., Randers, J., & Behrens, W. W. I. (1972). The Limits to Growth | Dartmouth Libraries. Universe Books. https://doi.org/10.1349/ddlp.1

Mirra, G. (2022). Expertise, playfulness, analogical reasoning: Three learning mechanisms to train Artificial Intelligence for design applications [Dissertation (external), University of Melbourne]. https://findanexpert.unimelb.edu.au/scholarlywork/1756533-expertise--playfulness--analogical-reasoning--three-learning-mechanisms-to-train-artificial-intelligence-for-design-applications

Mirra, G., & Mueller, L.-M. (2025). A Reinforcement Learning Agent for Enhancing the Reuse of Structural Components in New Buildings.

Mirra, G., & Pugnale, A. (2021). Comparison between human-defined and AI-generated design spaces for the optimisation of shell structures. Structures, 34, 2950—2961. https://doi.org/10.1016/j.istruc.2021.09.058

Morseletto, P. (2020). Targets for a circular economy. Resources, Conservation and Recycling, 153, 104553. https://doi.org/10.1016/j.resconrec.2019.104553

Munaro, M. R., & Tavares, S. F. (2023). A review on barriers, drivers, and stakeholders towards the circular economy: The construction sector perspective. Cleaner and Responsible Consumption, 8, 100107. https://doi.org/10.1016/j.clrc.2023.100107

Nederlande Norm. (n.d.). NEN-EN 1990-1:2025+A1:2026 (Eurocode—Grondslagen van het constructief en geotechnisch ontwerp—Deel 1: Nieuwe constructies).

Nederlandse Norm. (n.d.). NEN-EN 1991-2:2025 (Eurocode 1: Actions on structures — Part 2: Traffic loads on bridges).

Nian, S., Pham, T., Haas, C., Ibrahim, N., Yoon, D., & Bregman, H. (2022). A functional demonstration of adaptive reuse of waste into modular assemblies for structural applications: The case of bicycle frames. Journal of Cleaner Production, 348, 131162. https://doi.org/10.1016/j.jclepro.2022.131162

None, N. (2017). Topology Optimised Pedestrian Bridge. https://repository.tudelft.nl/record/uuid:6184bd62-63d2-4929-ae94-7f549149f5a5

Ororbia, M. E., & Warn, G. P. (2022). Design Synthesis Through a Markov Decision Process and Reinforcement Learning Framework. Journal of Computing and Information Science in Engineering, 22(021002). https://doi.org/10.1115/1.4051598

Ororbia, M. E., & Warn, G. P. (2023). Design Synthesis of Structural Systems as a Markov Decision Process Solved With Deep Reinforcement Learning. Journal of Mechanical Design, 145(061701). https://doi.org/10.1115/1.4056693

Piker, D. (2013). Kangaroo: Form Finding with Computational Physics. Architectural Design, 83, 136—137. https://doi.org/10.1002/ad.1569

Potting, J., Hekkert, M. P., Worrell, E., & Hanemaaijer, A. (2017). Circular Economy: Measuring innovation in the product chain.

Rajasekar, V., Lee, J., Gencsoy, A., & Rycke, K. D. (2024). AI-guided generation of reticular structures by integrating reinforcement learning with shape grammars.

Regenwetter, L., Curry, B., & Ahmed, F. (2021). BIKED: A Dataset for Computational Bicycle Design with Machine

Learning Benchmarks. Journal of Mechanical Design, 1—19. https://doi.org/10.1115/1.4052585

Regenwetter, L., Weaver, C., & Ahmed, F. (2022). FRAMED: An AutoML Approach for Structural Performance Prediction of Bicycle Frames (arXiv:2201.10459). arXiv. https://doi.org/10.48550/arXiv.2201.10459

Rossi, A. (2023). Mediated Assemblies: An Open Source Software Approach to Combinatorial Design and Fabrication. https://doi.org/10.26083/tuprints-00024039

Rossi, A., & Tessmann, O. (2017). Aggregated Structures: Approximating Topology Optimized Material Distribution with Discrete Building Blocks.

Rossi, A., & Tessmann, O. (2019). From Voxels to Parts: Hierarchical Discrete Modeling for Design and Assembly. In L. Coc-chiarella (Ed.), ICGG 2018—Proceedings of the 18th International Conference on Geometry and Graphics (pp. 1001—1012). Springer International Publishing. https://doi.org/10.1007/978-3-319-95588-9_86

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., & Klimov, O. (2017). Proximal Policy Optimization Algorithms (arXiv:1707.06347). arXiv. https://doi.org/10.48550/arXiv.1707.06347

Sørensen, K.-J. I. (2024). From Waste to Structure: A Deep Reinforcement Learning Approach to Circular Design [Massa-chusetts Institute of Technology]. https://hdl.handle.net/1721.1/157362

Statista Research Department. (2025, October 28). Global waste generation—Statistics & facts. Statista.Com. https://www.statista.com/topics/4983/waste-generation-worldwide/

Stiny, G. (1980). Introduction to shape and shape grammars. Environment and Planning B: Planning and Design, 7(3), 343—351. https://doi.org/10.1068/b070343

Stiny, G., & Gips, J. (1971). 'Shape Grammars and the Generative Specification of Painting and Sculpture'. In IFIP Congress (Vol. 71, p. 1465).

Sutton, R. S., & Barto, A. G. (2014). Reinforcement Learning: An Introduction (Second edition, in progress). MIT Press.

The LEGO® moulding philosophy. (n.d.). LEGO® History. Retrieved 18 October 2026, from https://www.lego.com/en-us/history/articles/e-the-lego-moulding-philosophy

The Paris Agreement, United Nations COP 21 (2016). https://unfccc.int/documents/184656

Tomaszewski, T. (2021). Fatigue life analysis of steel bicycle frame according to ISO 4210. Engineering Failure Analysis, 122, 105195. https://doi.org/10.1016/j.engfailanal.2020.105195

UNEP. (2024). SCP-HAT Database v3.0 [Data set]. https://scp-hat.org/data-download/

UNEP. (2025). Not just another brick in the wall: The solutions exist - Scaling them will build on progress and cut emissions fast. (Global Status Report for Buildings and Construction). United Nations Environment Programme. https://doi.org/10.59117/20.500.11822/47214

UNEP. (2025, May 11). The world is likely to exceed a key global warming target soon. Now what? UNEP. https://www.unep.org/news-and-stories/story/world-likely-exceed-key-global-warming-target-soon-now-what

van Baalen, S. (2017). TOPOLOGY OPTIMISED PEDESTRIAN BRIDGE.

van Dooren, E., Boshuizen, E., van Merriënboer, J., Asselbergs, T., & van Dorst, M. (2014). Making explicit in design educa-tion: Generic elements in the design process. International Journal of Technology and Design Education, 24(1), 53—71. https://doi.org/10.1007/s10798-013-9246-8

Voetgangersbruggen arsenaaldelft. (n.d.). Cepezed - Nederlands. Retrieved 3 October 2026, from https://www.cepezed.nl/projecten/voetgangersbruggen-arsenaaldelft/

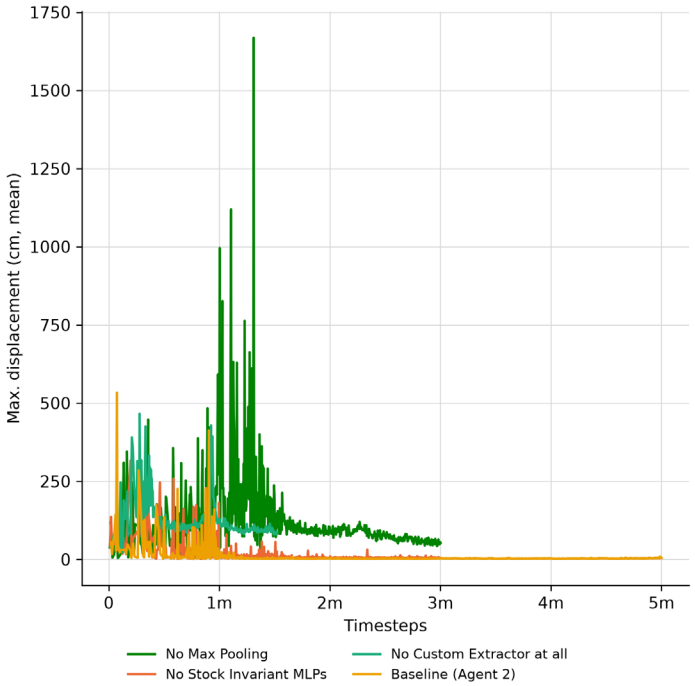
Watkins, C. J. C. H., & Dayan, P. (1992). Q-learning. Machine Learning, 8(3), 279—292. https://doi.org/10.1007/BF00992698

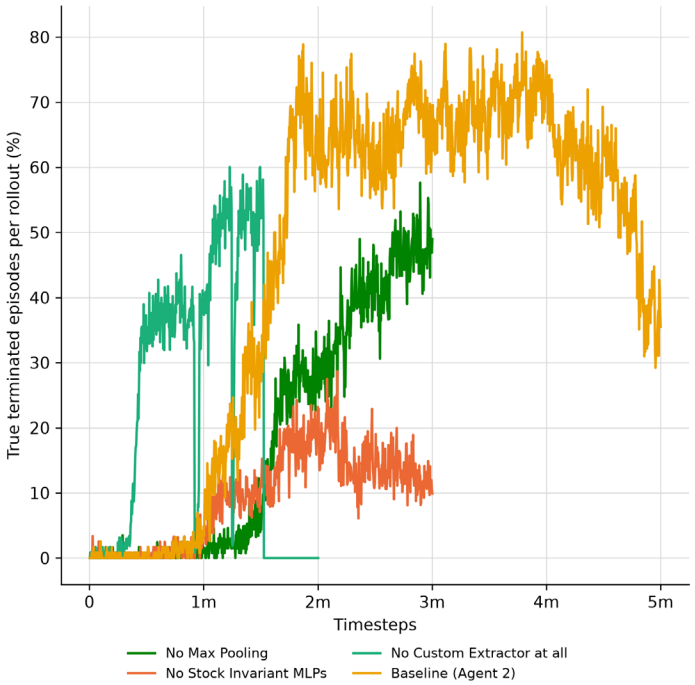
Willskytt, S., Böckin, D., André, H., Söderman, M. L., & Tillman, A. (2016). Framework for analysing resource-efficient solu-tions. https://www.semanticscholar.org/paper/Framework-for-analysing-resource-efficient-Willskytt-B%C3%B6ckin/6b27d1904dfbb40882d7b598af1cf6109b803831

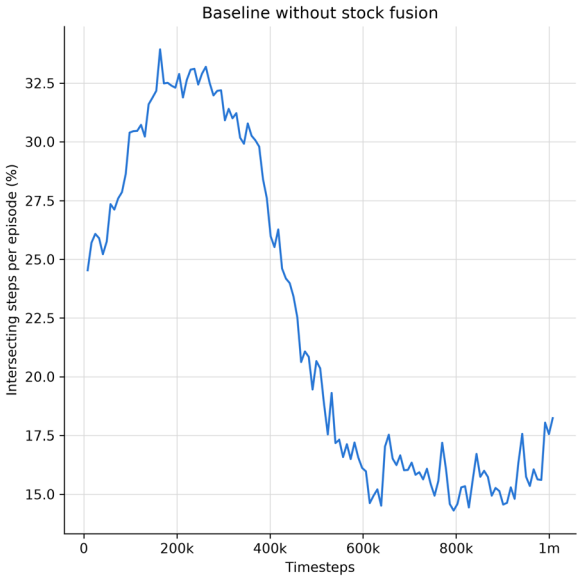
13.0 APPENDIX

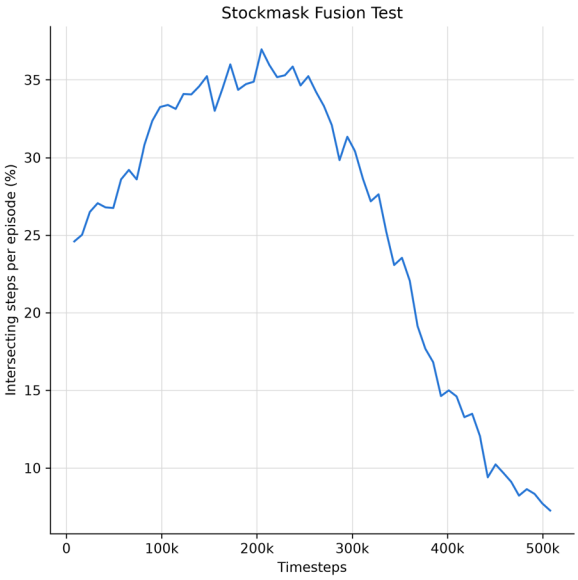
For insight into the code used in this thesis, please see the repository at:

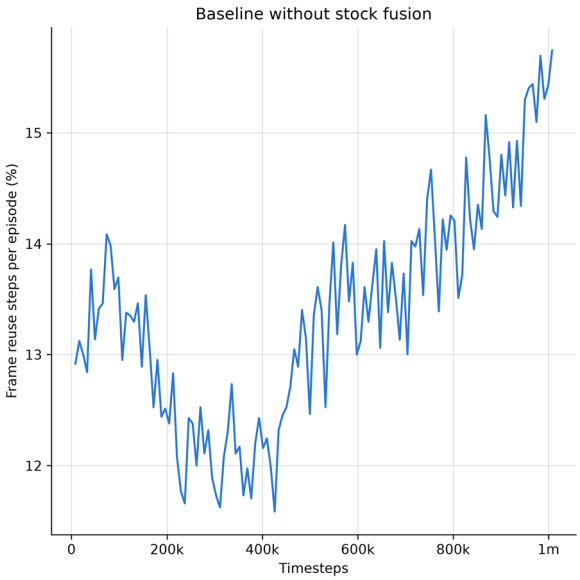
https://github.com/renshjansen-source/REP-STRUC-RL

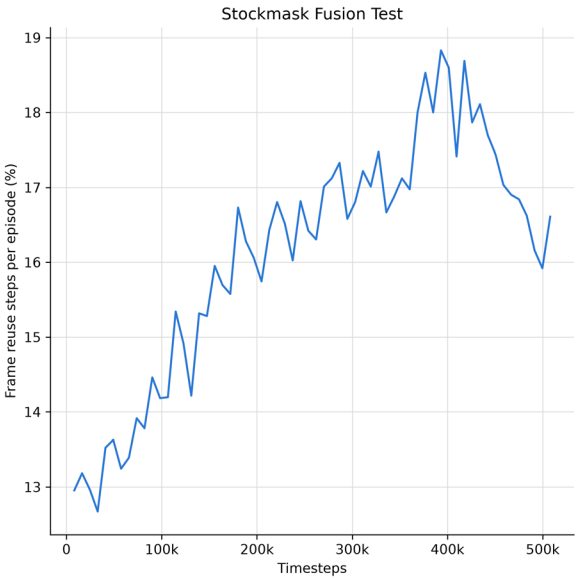


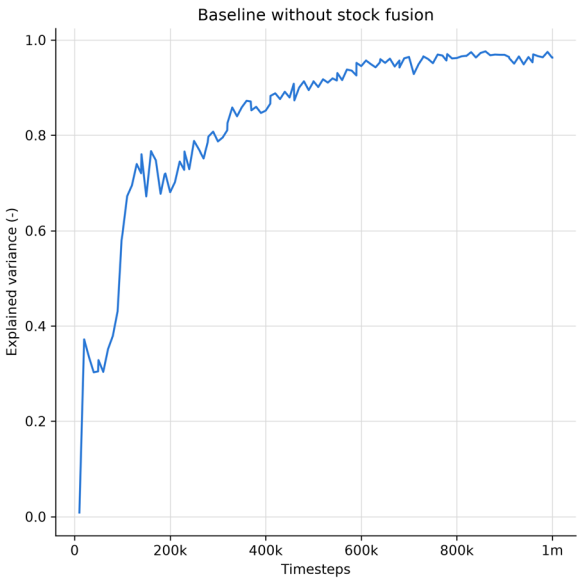


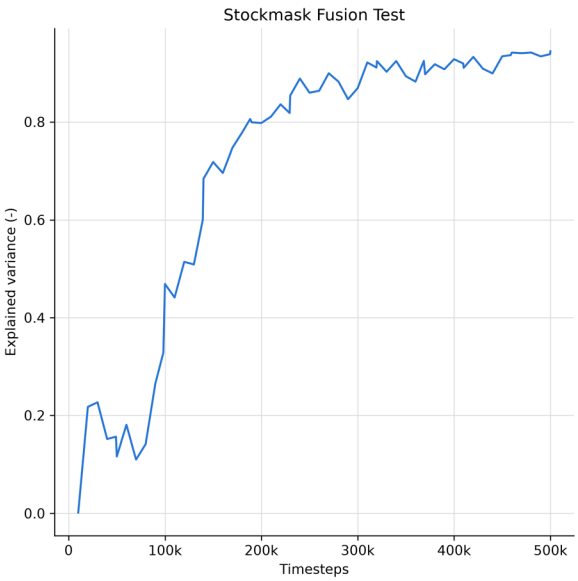

Baseline without stock fusion
Intersecting steps per episode (%)
32.5
30.0
27.5
25.0
22.5
20.0
17.5
15.0
0
200k
400k
600k
800k
1m
Timesteps

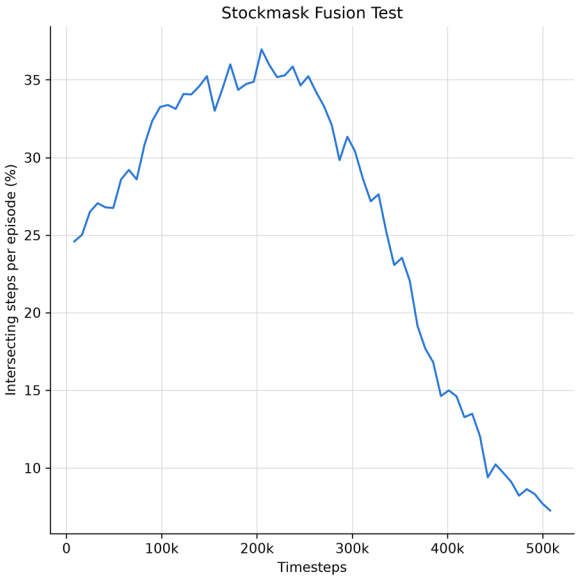

Stockmask Fusion Test
Intersecting steps per episode (%)
35
30
25
20
15
10
0
100k
200k
300k
400k
500k
Timesteps

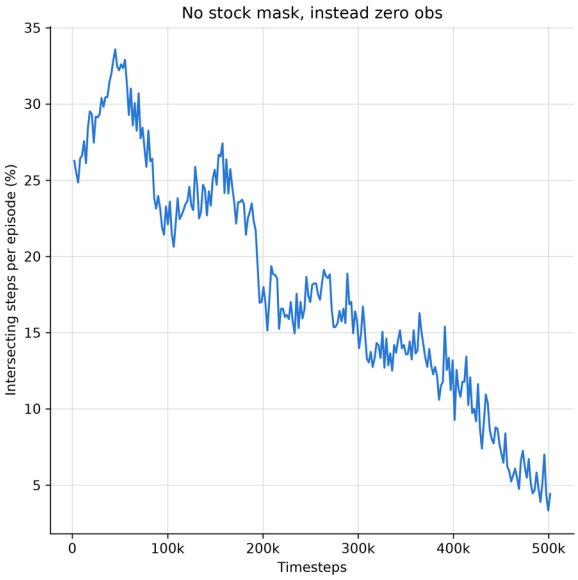

Baseline without stock fusion
Frame reuse steps per episode (%)
15
14
13
12
0
200k
400k
600k
800k
1m
Timesteps

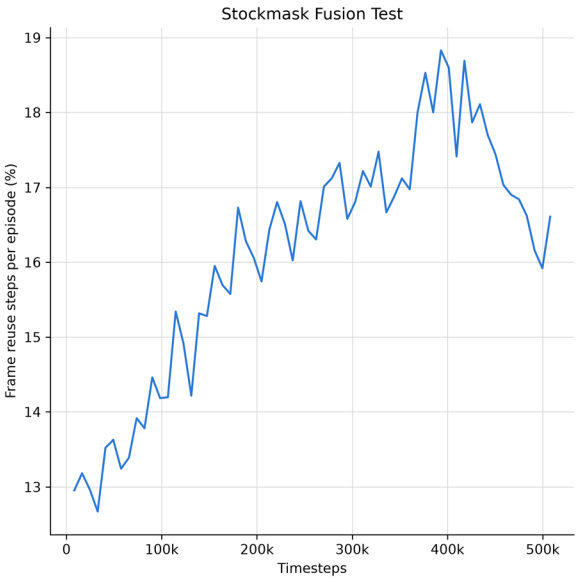

Stockmask Fusion Test
Frame reuse steps per episode (%)
19
18
17
16
15
14
13
0
100k
200k
300k
400k
500k
Timesteps

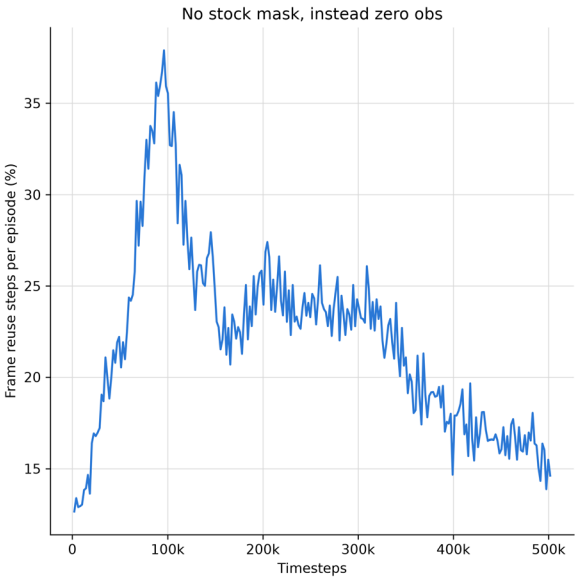

Baseline without stock fusion
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps

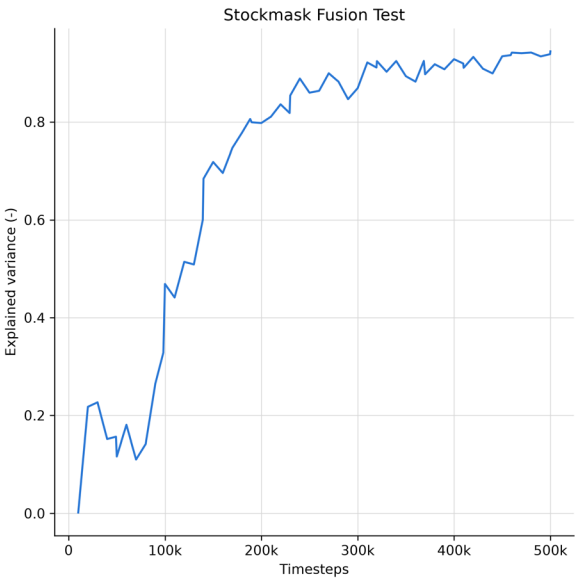

Stockmask Fusion Test
Explained variance (-)
0.8
0.6
0.4
0.2
0.0
0
100k
200k
300k
400k
500k
Timesteps

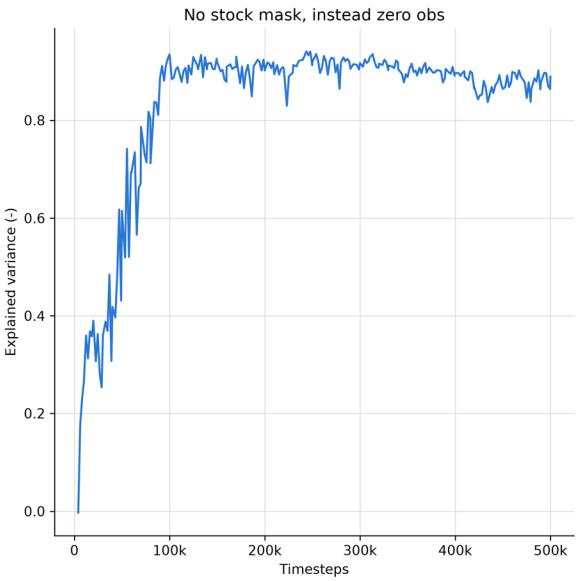

Stockmask Fusion Test
Intersecting steps per episode (%)
Timesteps

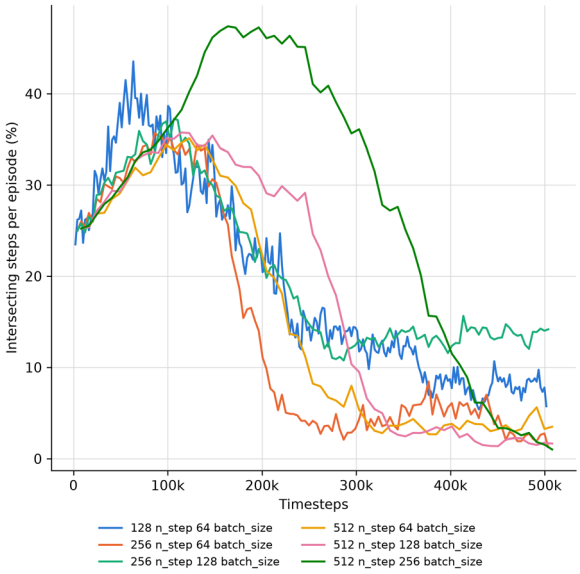

No stock mask, instead zero obs
Intersecting steps per episode (%)
Timesteps

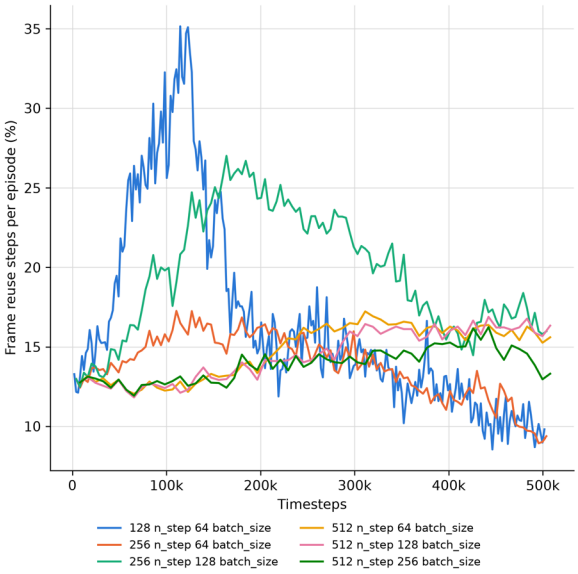

Stockmask Fusion Test
Frame reuse steps per episode (%)
Timesteps

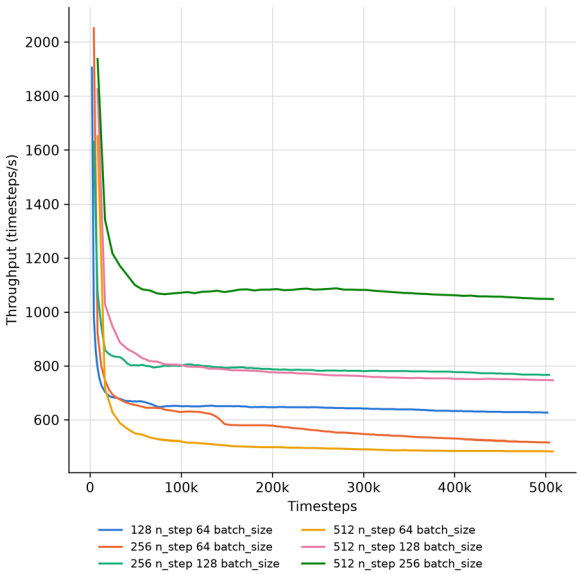

No stock mask, instead zero obs
Frame reuse steps per episode (%)
Timesteps

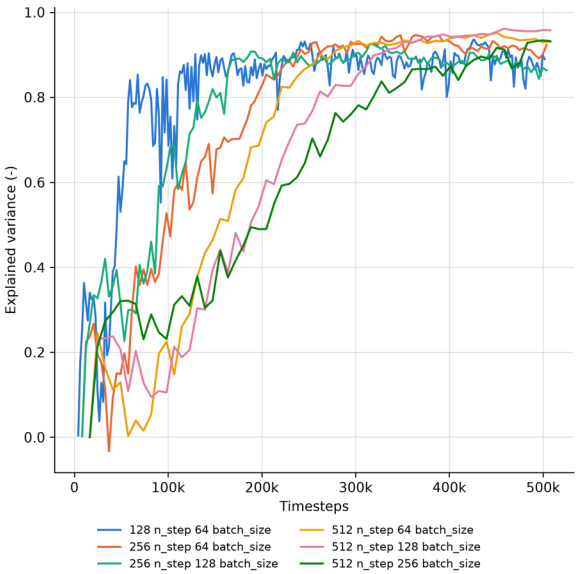

Stockmask Fusion Test
Explained variance (-)
Timesteps

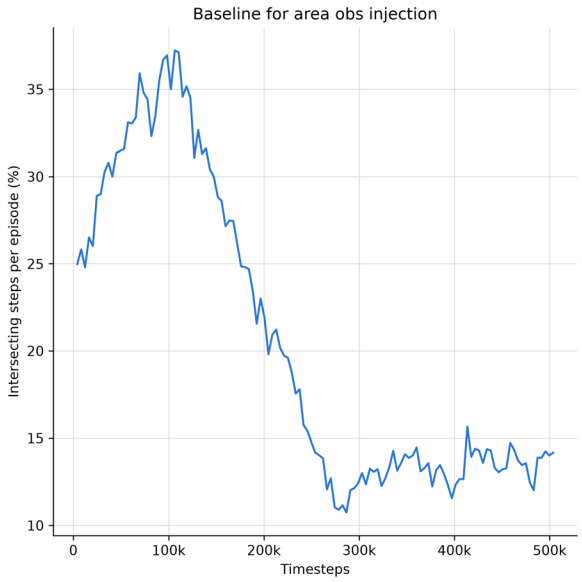

No stock mask, instead zero obs
Explained variance (-)
Timesteps

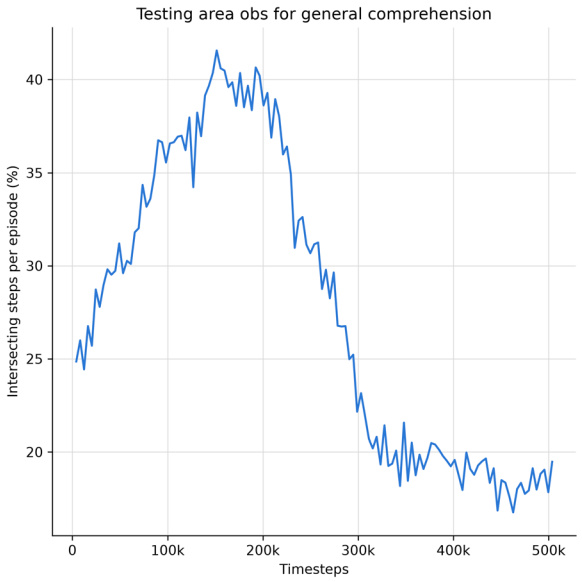

Intersecting steps per episode (%)
0
10
20
30
40
0
100k
200k
300k
400k
500k
Timesteps
128 n_step 64 batch_size
256 n_step 64 batch_size
256 n_step 128 batch_size
512 n_step 64 batch_size
512 n_step 128 batch_size
512 n_step 256 batch_size

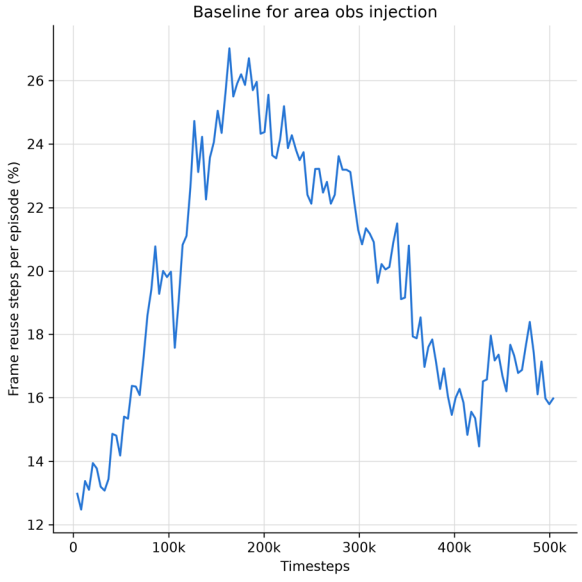

Frame reuse steps per episode (%)
10
15
20
25
30
35
0
100k
200k
300k
400k
500k
Timesteps
128 n_step 64 batch_size
256 n_step 64 batch_size
256 n_step 128 batch_size
512 n_step 64 batch_size
512 n_step 128 batch_size
512 n_step 256 batch_size

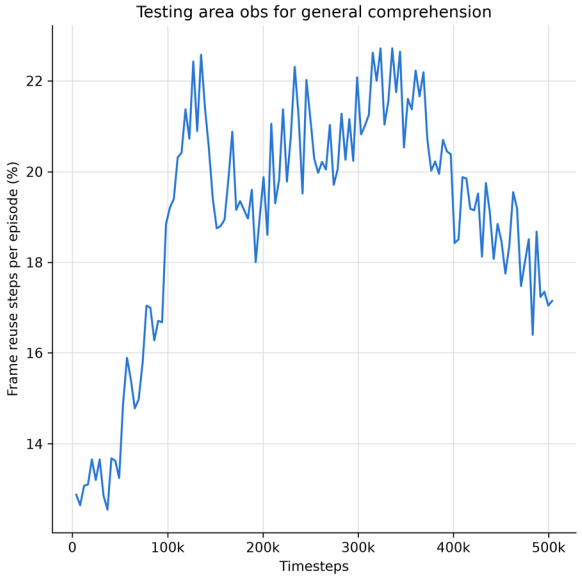

Throughput (timesteps/s)
600
800
1000
1200
1400
1600
1800
2000
0
100k
200k
300k
400k
500k
Timesteps
128 n_step 64 batch_size
256 n_step 64 batch_size
256 n_step 128 batch_size
512 n_step 64 batch_size
512 n_step 128 batch_size
512 n_step 256 batch_size

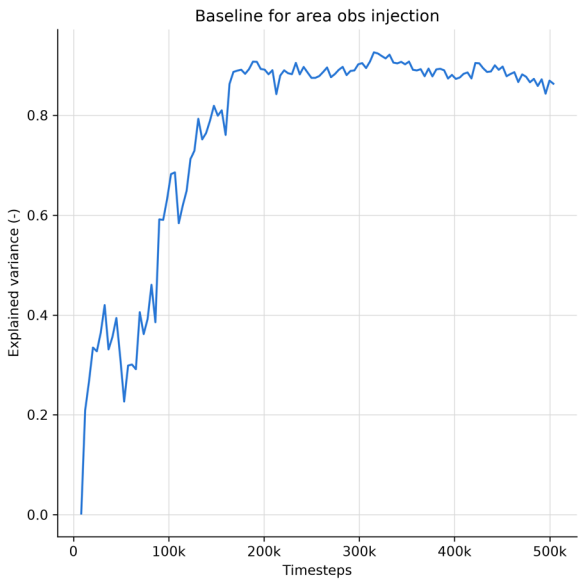

Explained variance (-)
0.0
0.2
0.4
0.6
0.8
1.0
0
100k
200k
300k
400k
500k
Timesteps
128 n_step 64 batch_size
256 n_step 64 batch_size
256 n_step 128 batch_size
512 n_step 64 batch_size
512 n_step 128 batch_size
512 n_step 256 batch_size

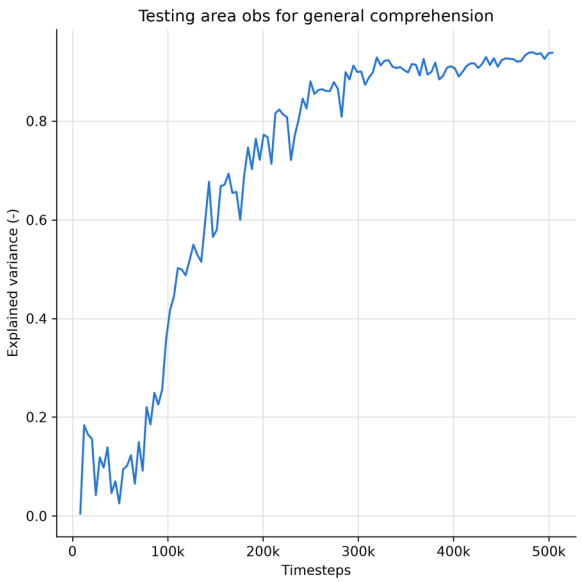

Baseline for area obs injection
Intersecting steps per episode (%)
Timesteps
0
100k
200k
300k
400k
500k
10
15
20
25
30
35

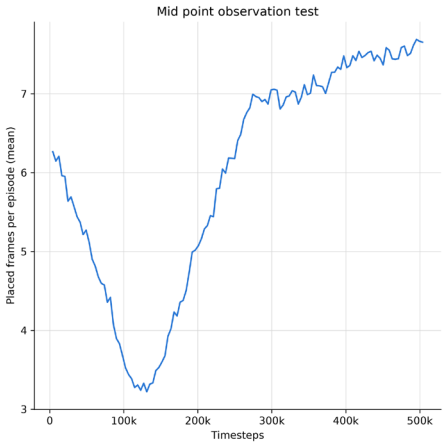

Testing area obs for general comprehension
Intersecting steps per episode (%)
Timesteps
0
100k
200k
300k
400k
500k
20
25
30
35
40

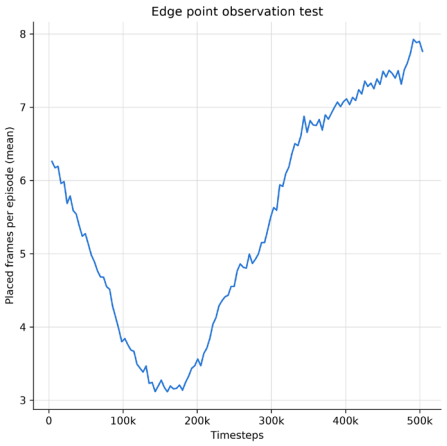

Baseline for area obs injection
Frame reuse steps per episode (%)
Timesteps
0
100k
200k
300k
400k
500k
12
14
16
18
20
22
24
26

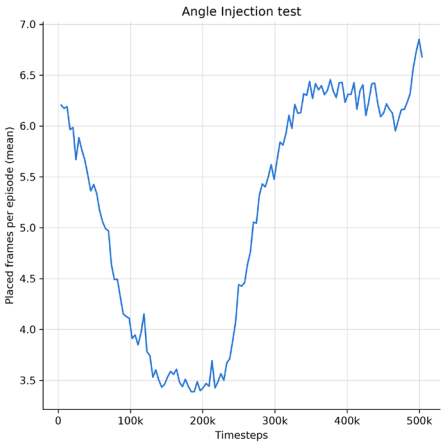

Testing area obs for general comprehension
Frame reuse steps per episode (%)
Timesteps
0
100k
200k
300k
400k
500k
14
16
18
20
22

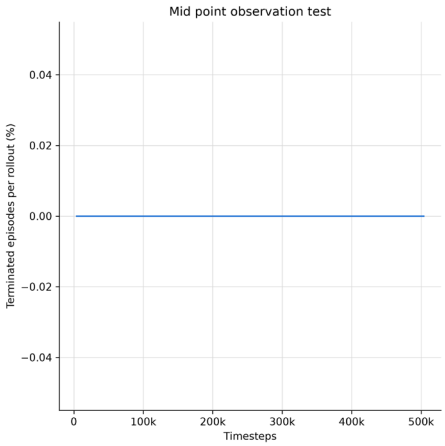

Baseline for area obs injection
Explained variance (-)
Timesteps
0
100k
200k
300k
400k
500k
0.0
0.2
0.4
0.6
0.8

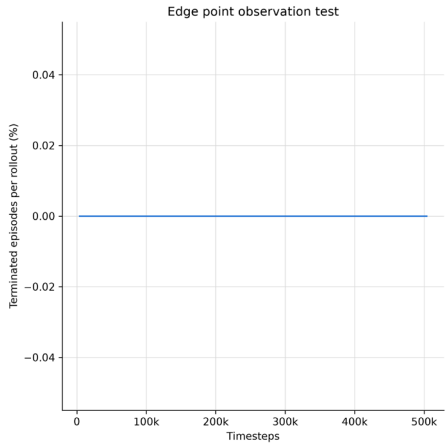

Testing area obs for general comprehension
Explained variance (-)
Timesteps
0
100k
200k
300k
400k
500k
0.0
0.2
0.4
0.6
0.8

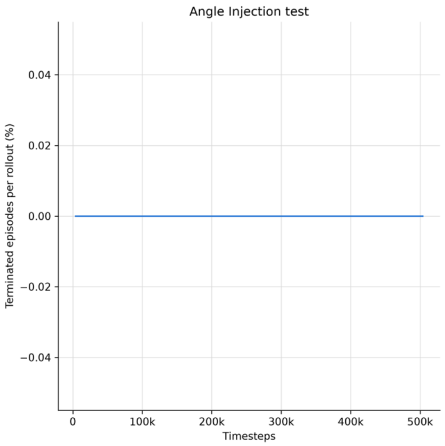

Mid point observation test
Placed frames per episode (mean)
Timesteps

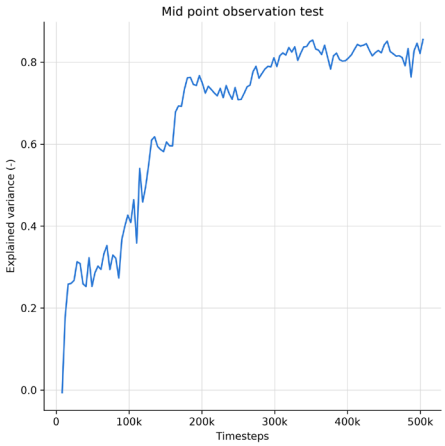

Edge point observation test
Placed frames per episode (mean)
Timesteps

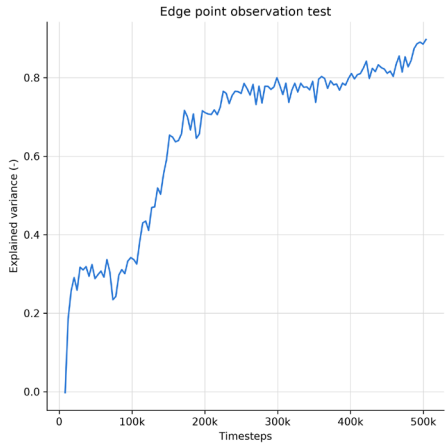

Angle Injection test
Placed frames per episode (mean)
Timesteps

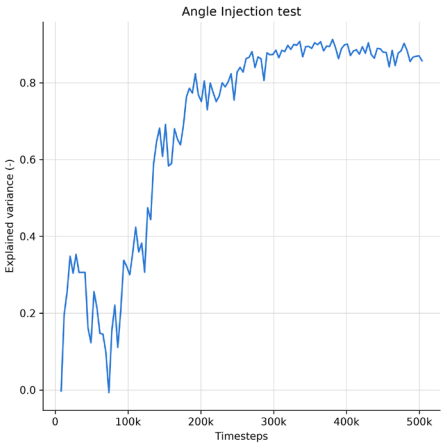

Mid point observation test
Terminated episodes per rollout (%)
Timesteps

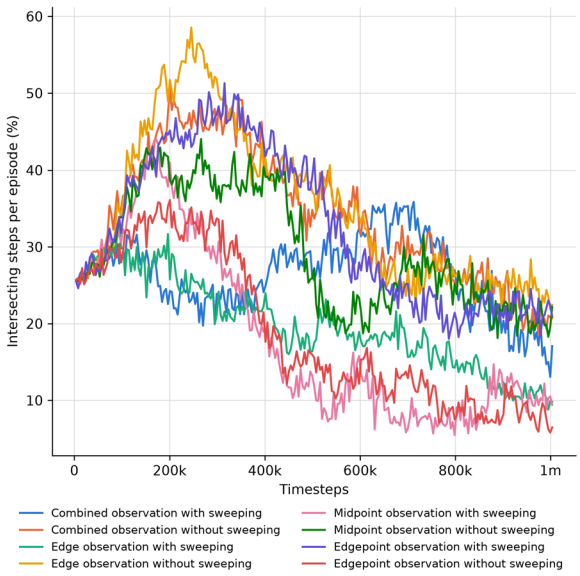

Edge point observation test
Terminated episodes per rollout (%)
Timesteps

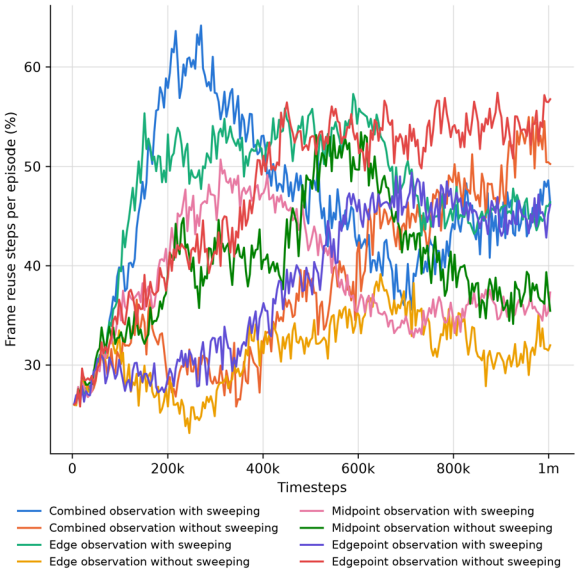

Angle Injection test
Terminated episodes per rollout (%)
Timesteps

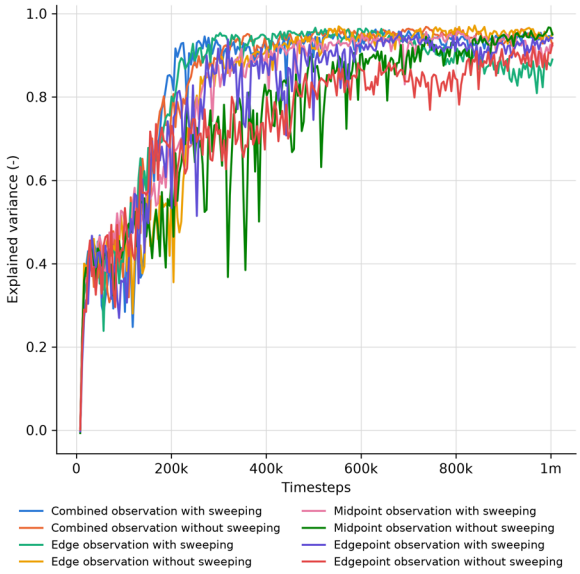

Mid point observation test
Explained variance (-)
Timesteps

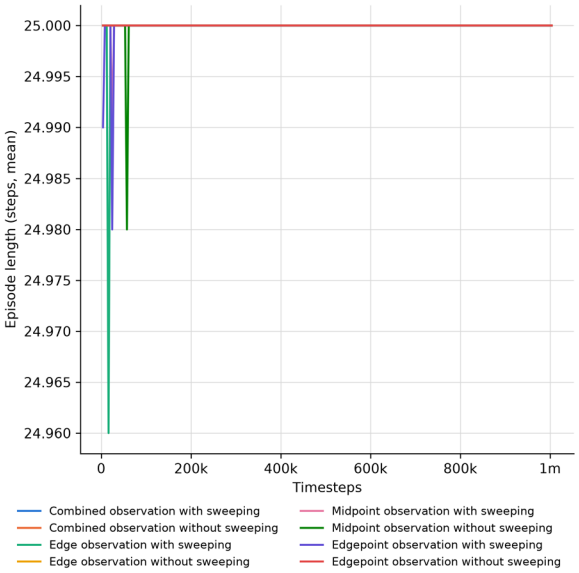

Edge point observation test
Explained variance (-)
Timesteps

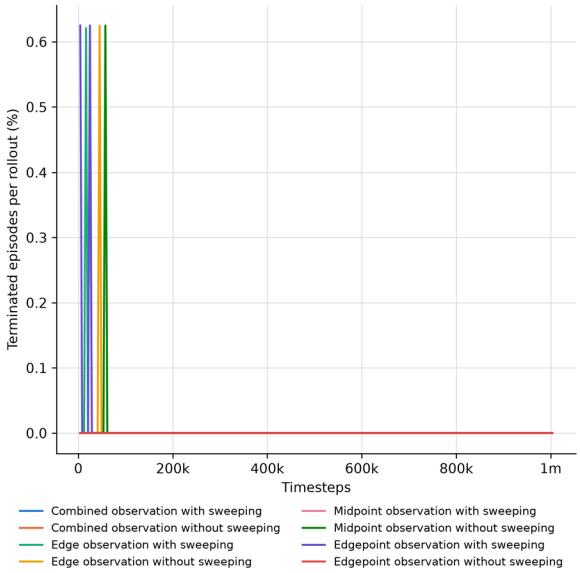

Angle Injection test
Explained variance (-)
Timesteps

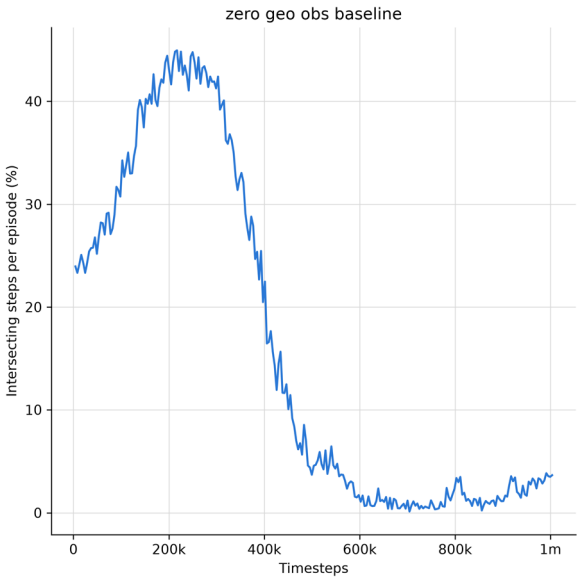

Intersecting steps per episode (%)
60
50
40
30
20
10
0
200k
400k
600k
800k
1m
Timesteps
Combined observation with sweeping
Combined observation without sweeping
Edge observation with sweeping
Edge observation without sweeping
Midpoint observation with sweeping
Midpoint observation without sweeping
Edgepoint observation with sweeping
Edgepoint observation without sweeping

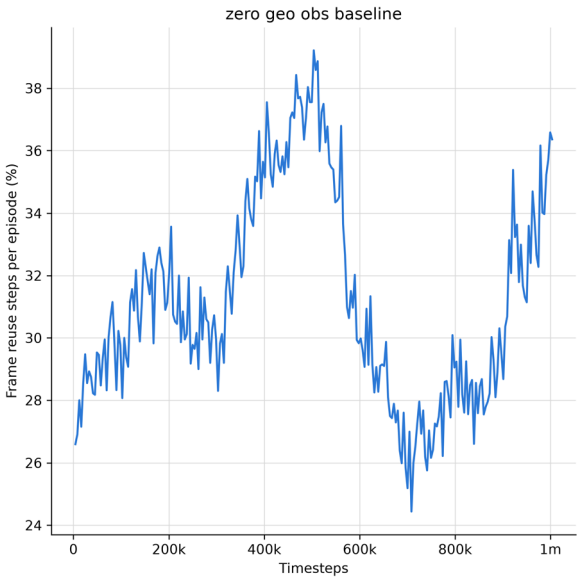

Frame reuse steps per episode (%)
60
50
40
30
0
200k
400k
600k
800k
1m
Timesteps
Combined observation with sweeping
Combined observation without sweeping
Edge observation with sweeping
Edge observation without sweeping
Midpoint observation with sweeping
Midpoint observation without sweeping
Edgepoint observation with sweeping
Edgepoint observation without sweeping

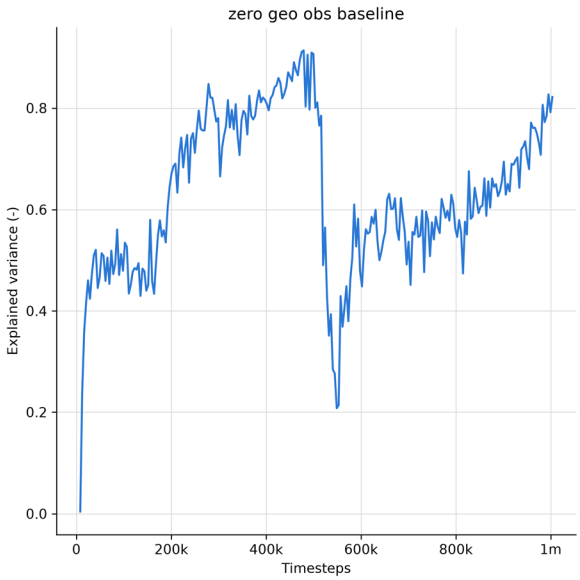

Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps
Combined observation with sweeping
Combined observation without sweeping
Edge observation with sweeping
Edge observation without sweeping
Midpoint observation with sweeping
Midpoint observation without sweeping
Edgepoint observation with sweeping
Edgepoint observation without sweeping

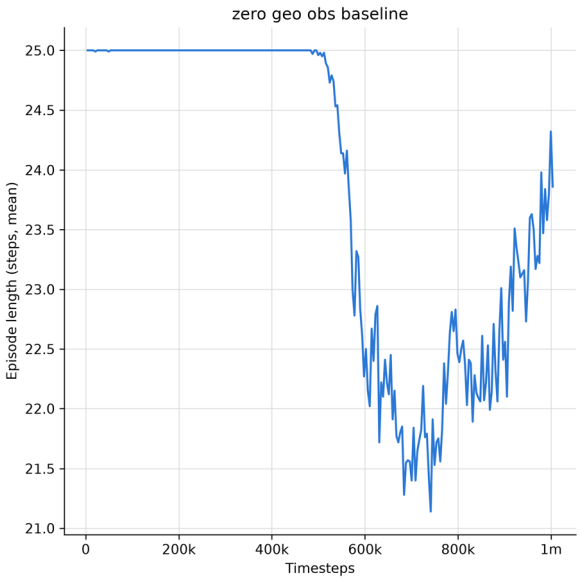

Episode length (steps, mean)
25.000
24.995
24.990
24.985
24.980
24.975
24.970
24.965
24.960
0
200k
400k
600k
800k
1m
Timesteps
Combined observation with sweeping
Combined observation without sweeping
Edge observation with sweeping
Edge observation without sweeping
Midpoint observation with sweeping
Midpoint observation without sweeping
Edgepoint observation with sweeping
Edgepoint observation without sweeping

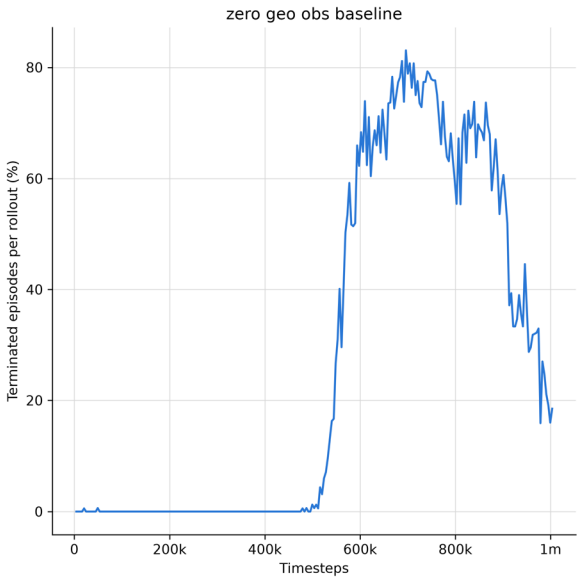

Terminated episodes per rollout (%)
0.6
0.5
0.4
0.3
0.2
0.1
0.0
0
200k
400k
600k
800k
1m
Timesteps
Combined observation with sweeping
Combined observation without sweeping
Edge observation with sweeping
Edge observation without sweeping
Midpoint observation with sweeping
Midpoint observation without sweeping
Edgepoint observation with sweeping
Edgepoint observation without sweeping

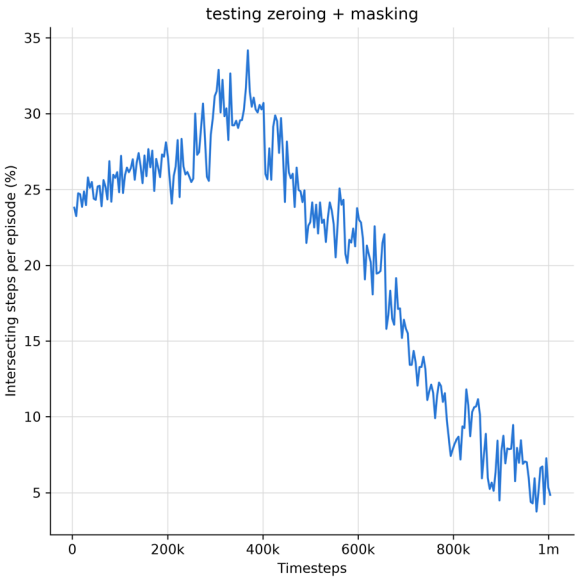

zero geo obs baseline
Intersecting steps per episode (%)
0
10
20
30
40
0
200k
400k
600k
800k
1m
Timesteps

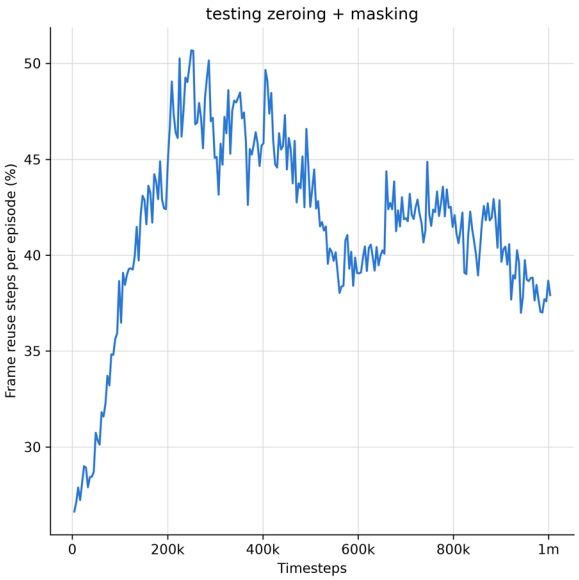

zero geo obs baseline
Frame reuse steps per episode (%)
24
26
28
30
32
34
36
38
0
200k
400k
600k
800k
1m
Timesteps

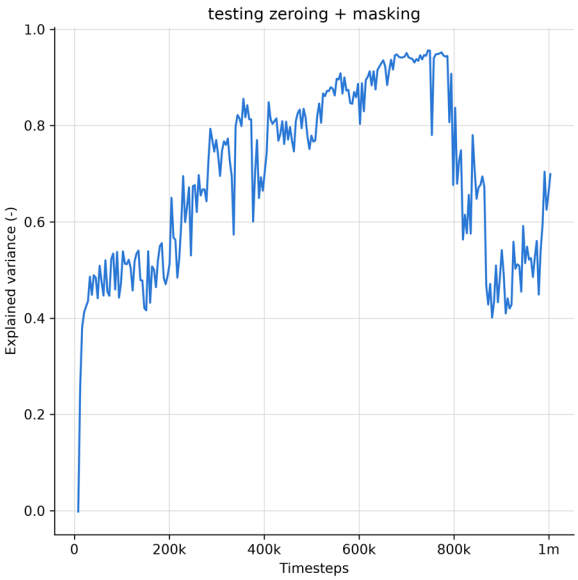

zero geo obs baseline
Explained variance (-)
0.0
0.2
0.4
0.6
0.8
0
200k
400k
600k
800k
1m
Timesteps

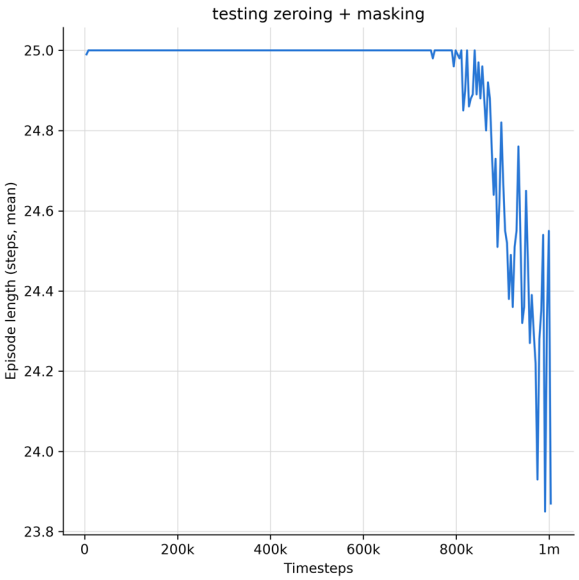

zero geo obs baseline
Episode length (steps, mean)
21.0
21.5
22.0
22.5
23.0
23.5
24.0
24.5
25.0
0
200k
400k
600k
800k
1m
Timesteps

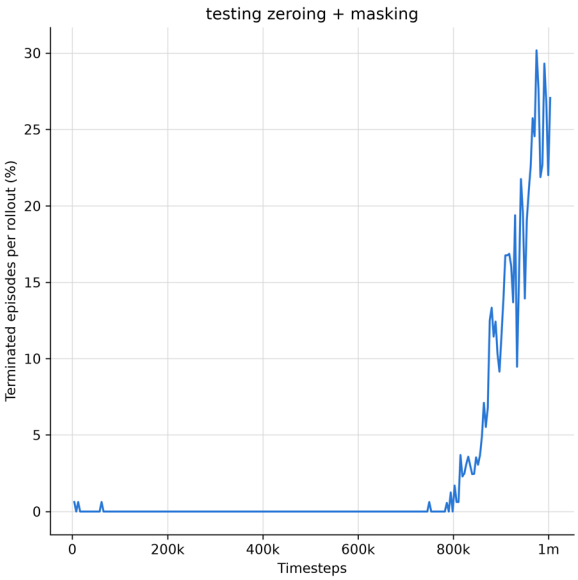

zero geo obs baseline
Terminated episodes per rollout (%)
0
20
40
60
80
0
200k
400k
600k
800k
1m
Timesteps

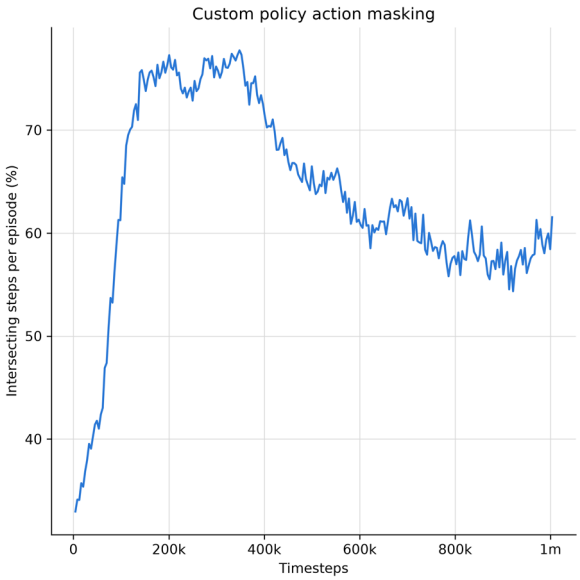

testing zeroing + masking
Intersecting steps per episode (%)
35
30
25
20
15
10
5
0
200k
400k
600k
800k
1m
Timesteps

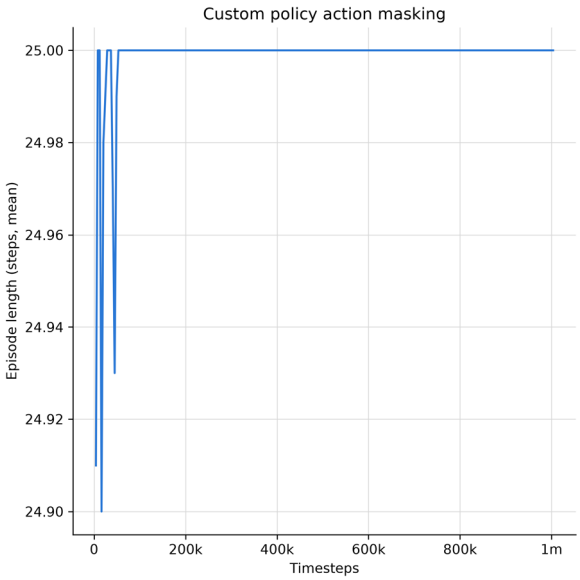

testing zeroing + masking
Frame reuse steps per episode (%)
50
45
40
35
30
0
200k
400k
600k
800k
1m
Timesteps

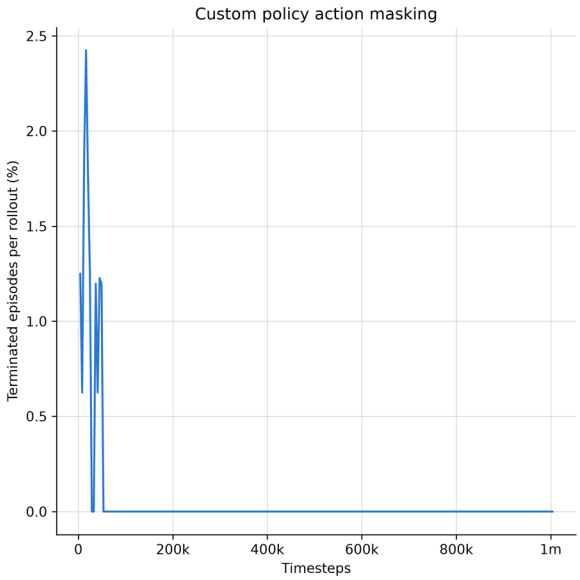

testing zeroing + masking
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps

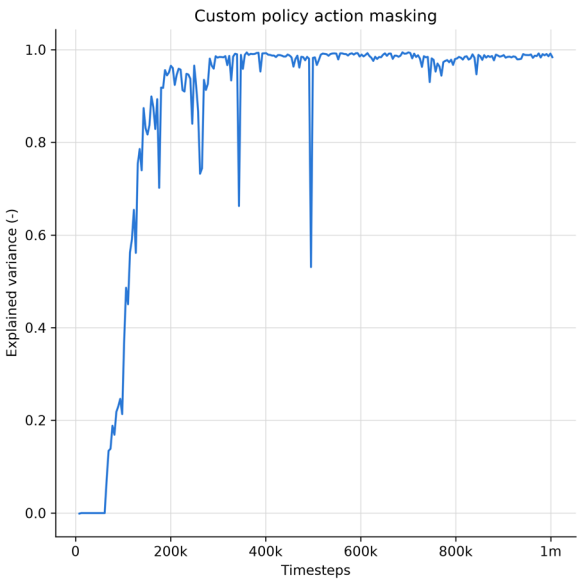

testing zeroing + masking
Episode length (steps, mean)
25.0
24.8
24.6
24.4
24.2
24.0
23.8
0
200k
400k
600k
800k
1m
Timesteps

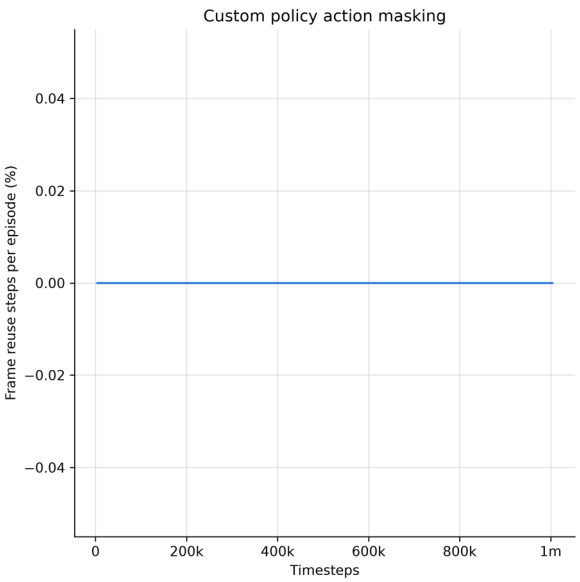

testing zeroing + masking
Terminated episodes per rollout (%)
30
25
20
15
10
5
0
0
200k
400k
600k
800k
1m
Timesteps

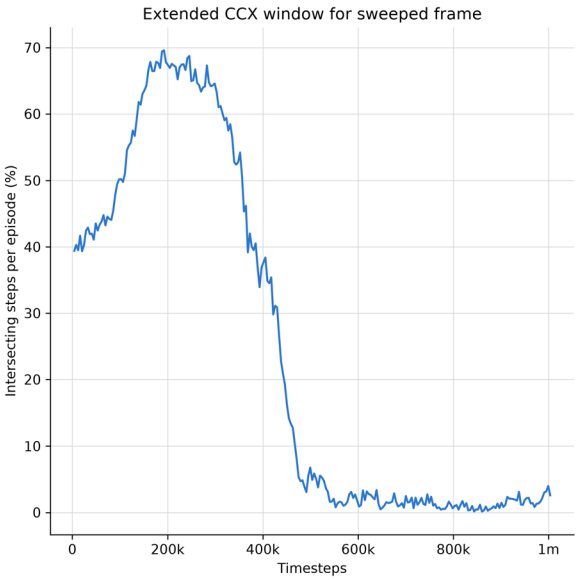

Custom policy action masking
Intersecting steps per episode (%)
70
60
50
40
0
200k
400k
600k
800k
1m
Timesteps

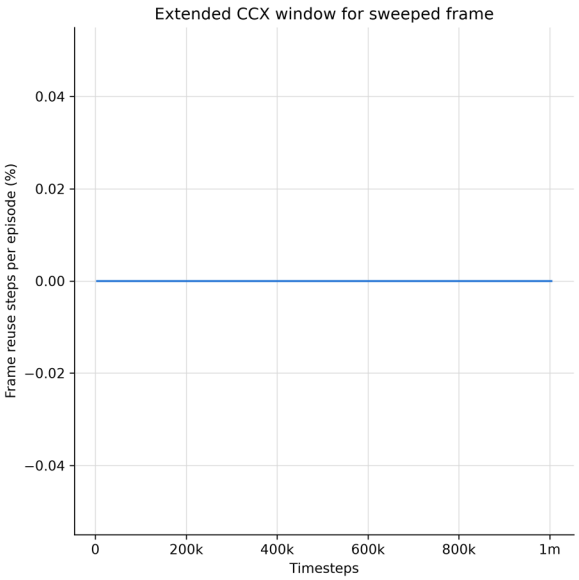

Custom policy action masking
Episode length (steps, mean)
25.00
24.98
24.96
24.94
24.92
24.90
0
200k
400k
600k
800k
1m
Timesteps

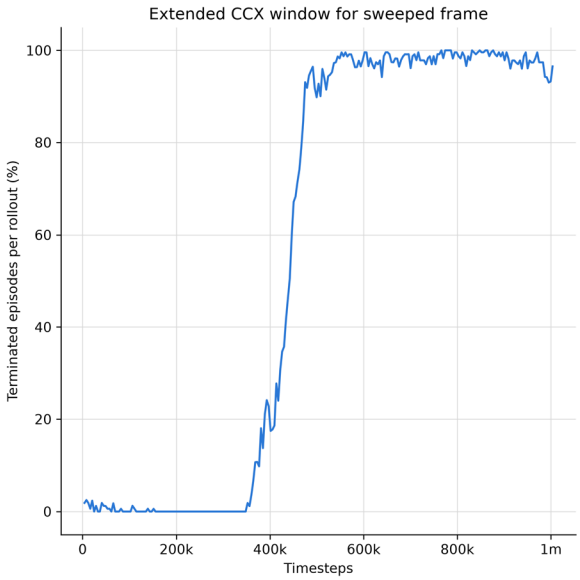

Custom policy action masking
Terminated episodes per rollout (%)
2.5
2.0
1.5
1.0
0.5
0.0
0
200k
400k
600k
800k
1m
Timesteps

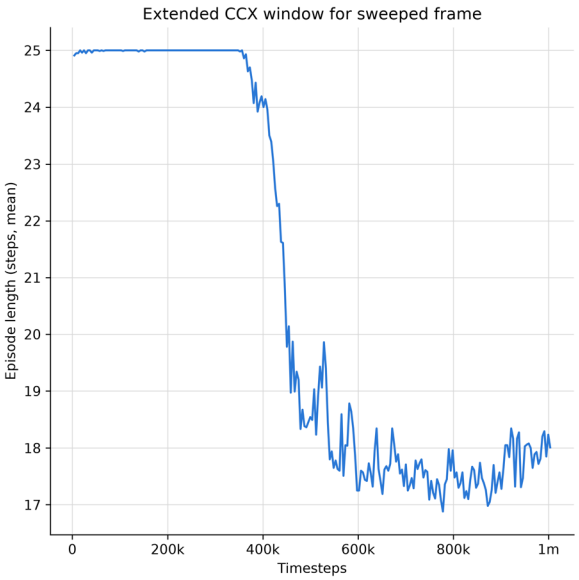

Custom policy action masking
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps

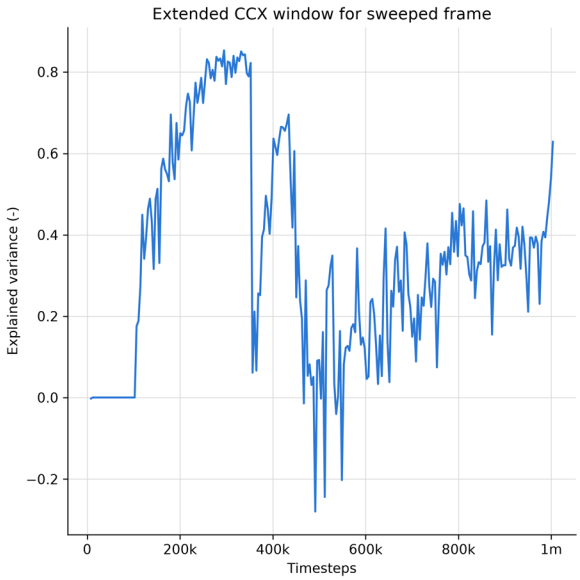

Custom policy action masking
Frame reuse steps per episode (%)
0.04
0.02
0.00
−0.02
−0.04
0
200k
400k
600k
800k
1m
Timesteps

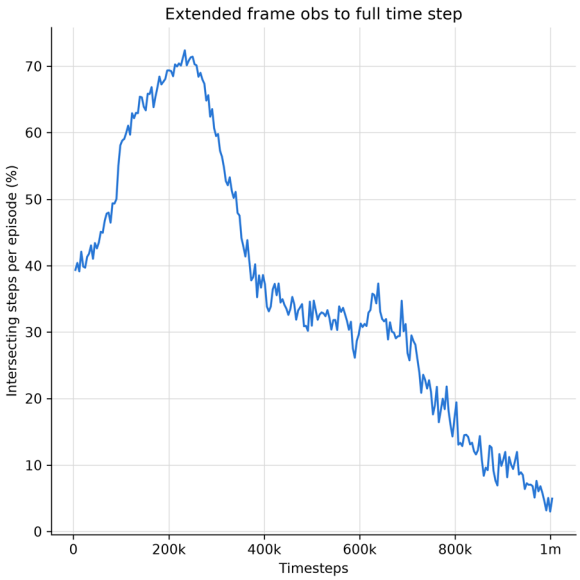

Extended CCX window for sweeped frame
Intersecting steps per episode (%)
70
60
50
40
30
20
10
0
0
200k
400k
600k
800k
1m
Timesteps

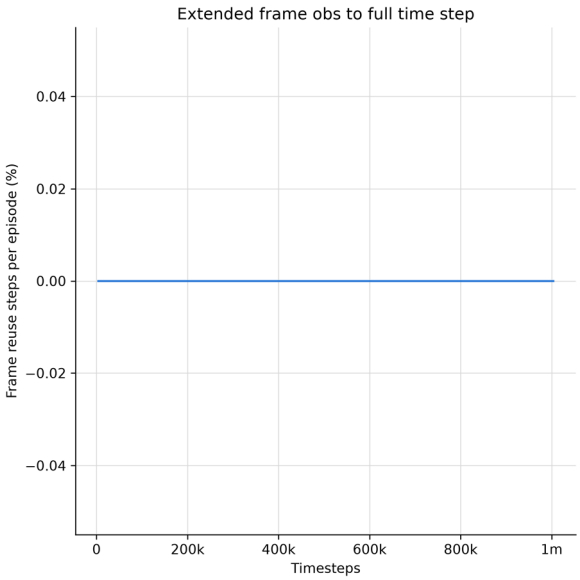

Extended CCX window for sweeped frame
Frame reuse steps per episode (%)
0.04
0.02
0.00
−0.02
−0.04
0
200k
400k
600k
800k
1m
Timesteps

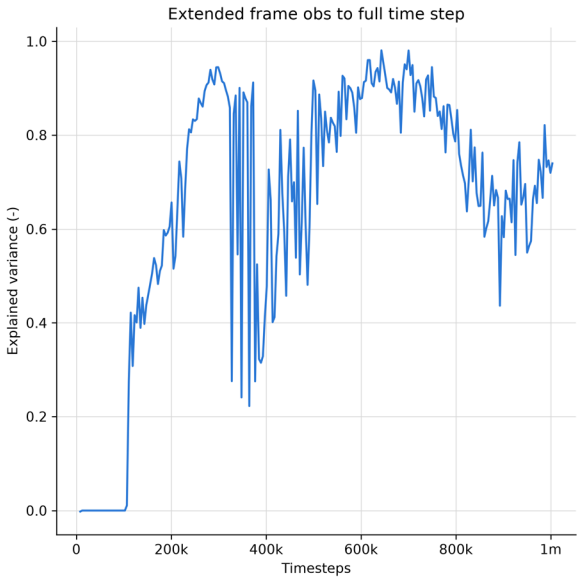

Extended CCX window for sweeped frame
Terminated episodes per rollout (%)
100
80
60
40
20
0
0
200k
400k
600k
800k
1m
Timesteps

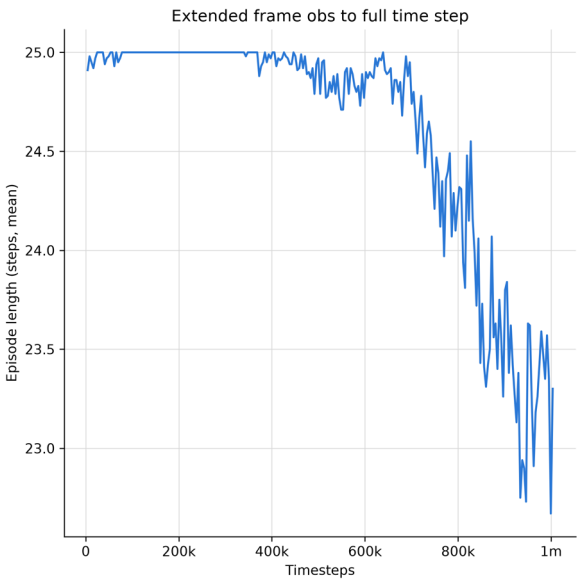

Extended CCX window for sweeped frame
Episode length (steps, mean)
25
24
23
22
21
20
19
18
17
0
200k
400k
600k
800k
1m
Timesteps

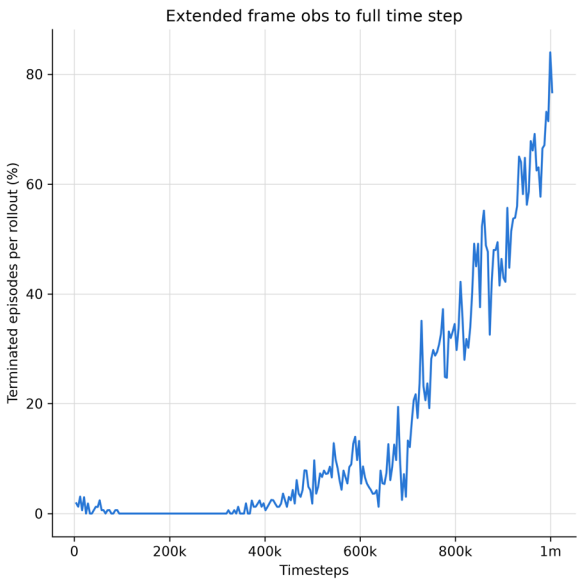

Extended CCX window for sweeped frame
Explained variance (-)
0.8
0.6
0.4
0.2
0.0
−0.2
0
200k
400k
600k
800k
1m
Timesteps

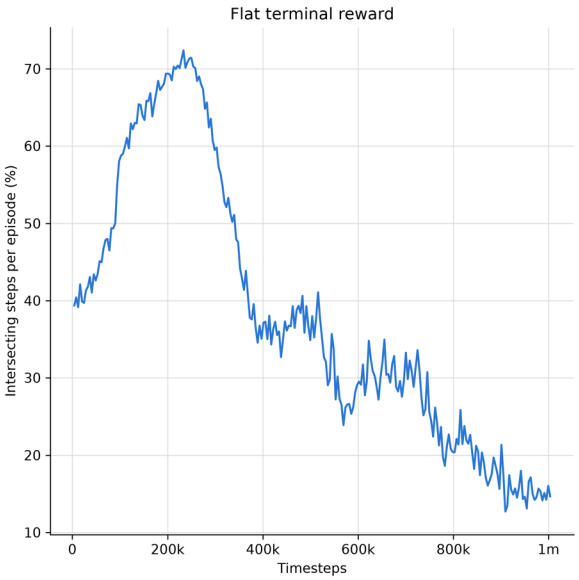

Extended frame obs to full time step
Intersecting steps per episode (%)
70
60
50
40
30
20
10
0
0
200k
400k
600k
800k
1m
Timesteps

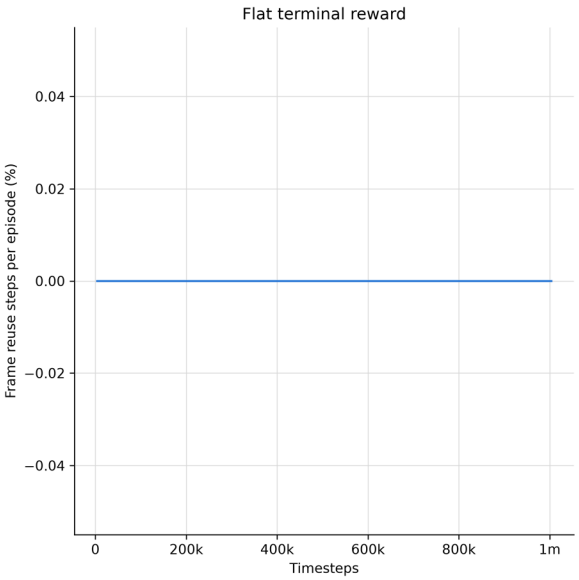

Extended frame obs to full time step
Frame reuse steps per episode (%)
0.04
0.02
0.00
−0.02
−0.04
0
200k
400k
600k
800k
1m
Timesteps

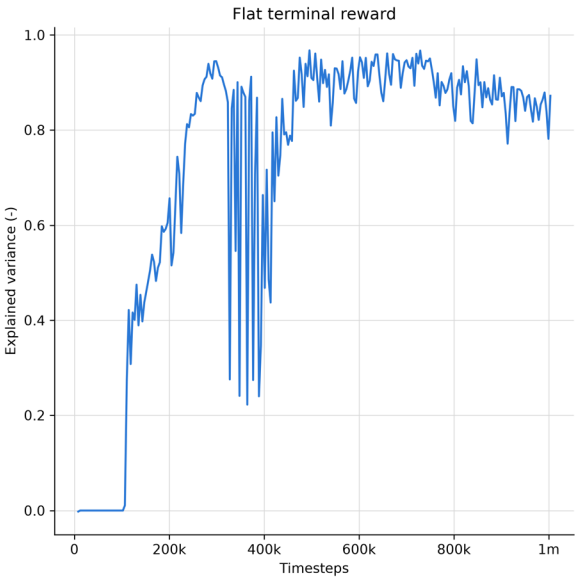

Extended frame obs to full time step
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps

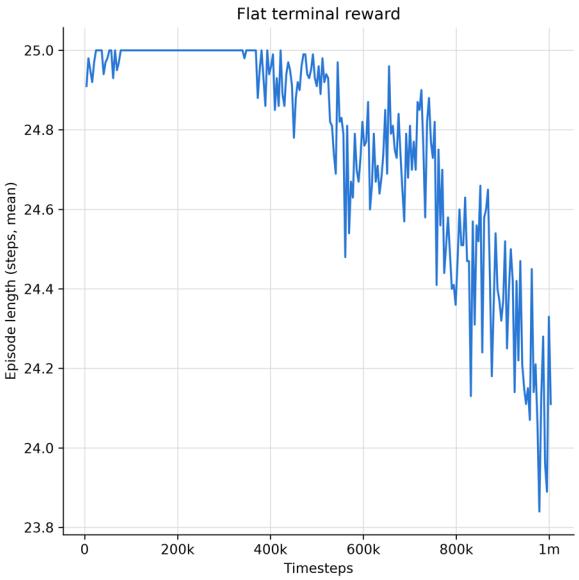

Extended frame obs to full time step
Episode length (steps, mean)
25.0
24.5
24.0
23.5
23.0
0
200k
400k
600k
800k
1m
Timesteps

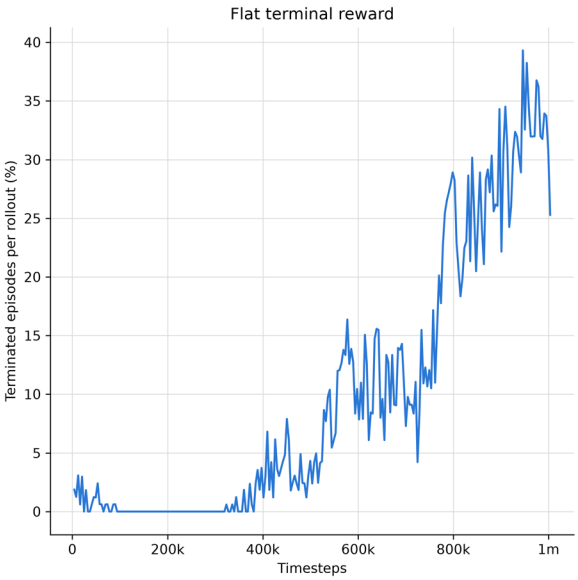

Extended frame obs to full time step
Terminated episodes per rollout (%)
80
60
40
20
0
0
200k
400k
600k
800k
1m
Timesteps

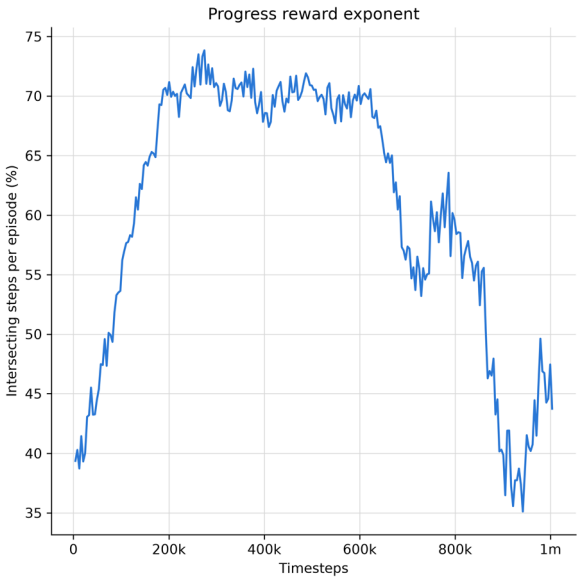

Flat terminal reward
Intersecting steps per episode (%)
70
60
50
40
30
20
10
0
200k
400k
600k
800k
1m
Timesteps

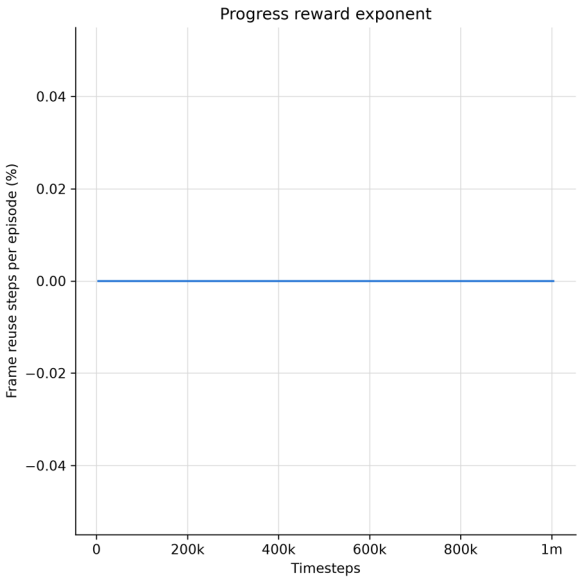

Flat terminal reward
Frame reuse steps per episode (%)
0.04
0.02
0.00
−0.02
−0.04
0
200k
400k
600k
800k
1m
Timesteps

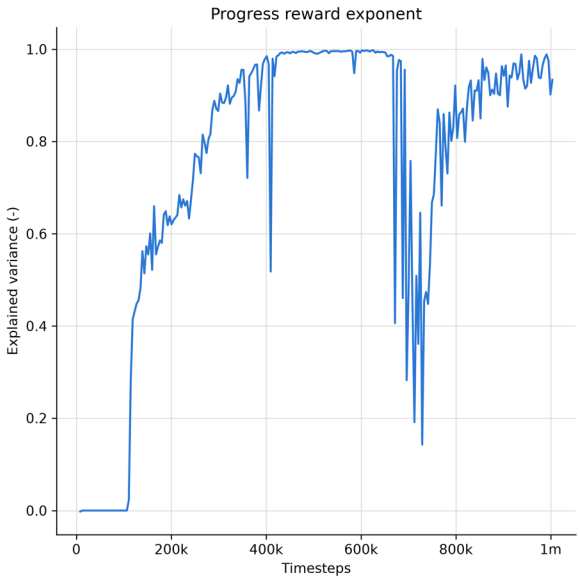

Flat terminal reward
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps

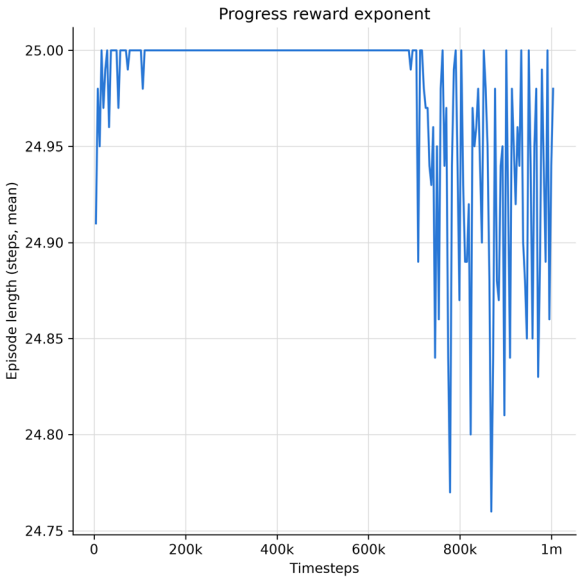

Flat terminal reward
Episode length (steps, mean)
25.0
24.8
24.6
24.4
24.2
24.0
23.8
0
200k
400k
600k
800k
1m
Timesteps

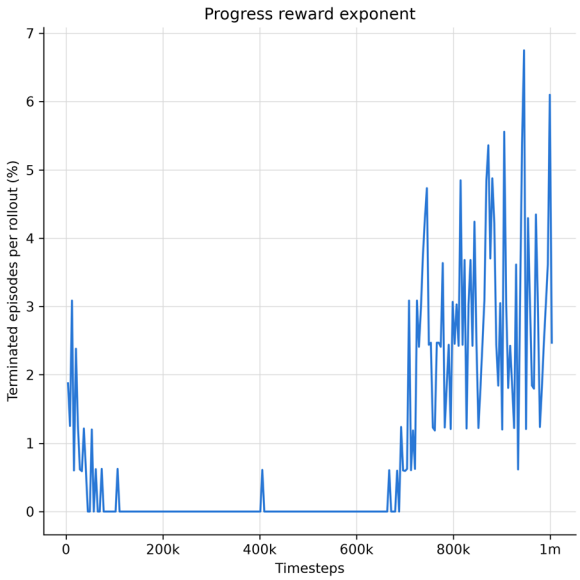

Flat terminal reward
Terminated episodes per rollout (%)
40
35
30
25
20
15
10
5
0
0
200k
400k
600k
800k
1m
Timesteps

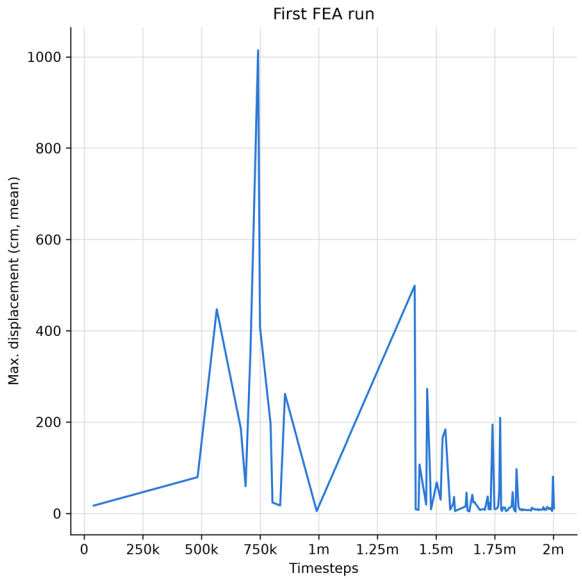

Progress reward exponent
Intersecting steps per episode (%)
75
70
65
60
55
50
45
40
35
0
200k
400k
600k
800k
1m
Timesteps

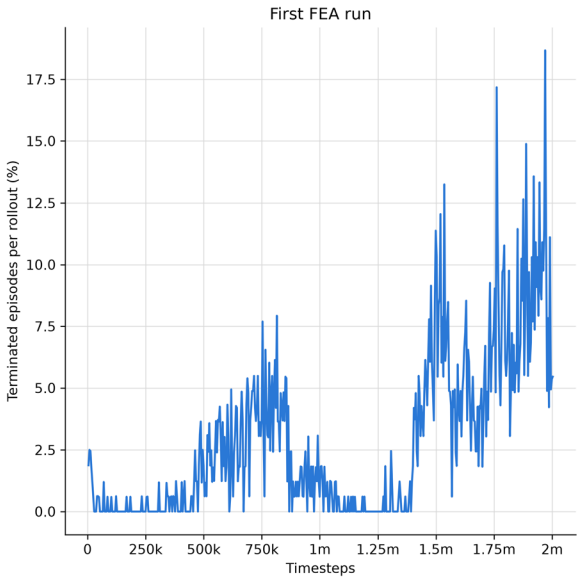

Progress reward exponent
Frame reuse steps per episode (%)
0.04
0.02
0.00
−0.02
−0.04
0
200k
400k
600k
800k
1m
Timesteps

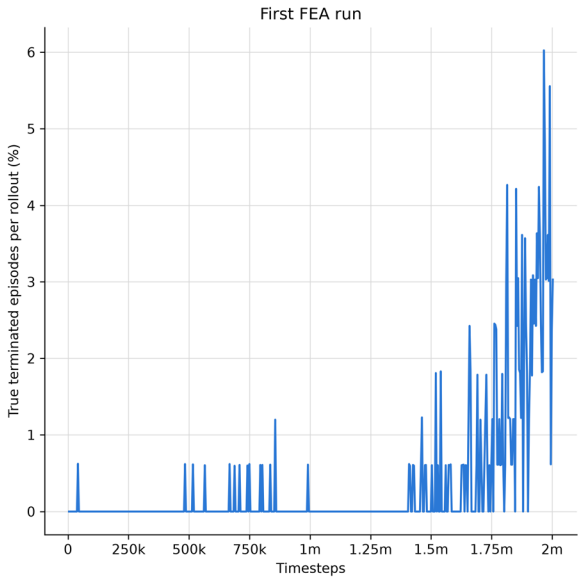

Progress reward exponent
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
200k
400k
600k
800k
1m
Timesteps

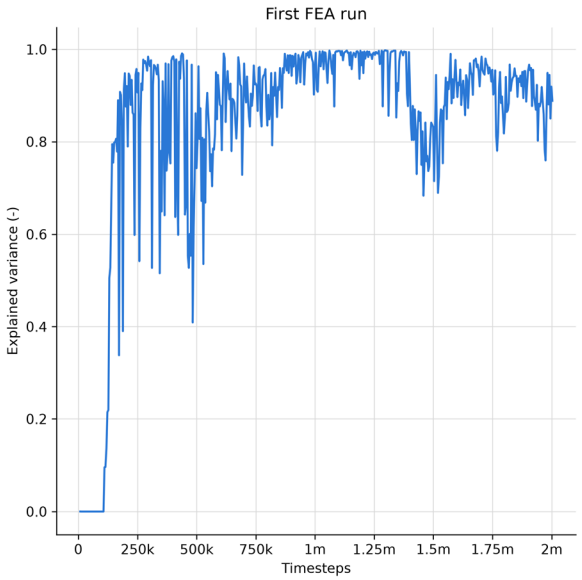

Progress reward exponent
Episode length (steps, mean)
25.00
24.95
24.90
24.85
24.80
24.75
0
200k
400k
600k
800k
1m
Timesteps

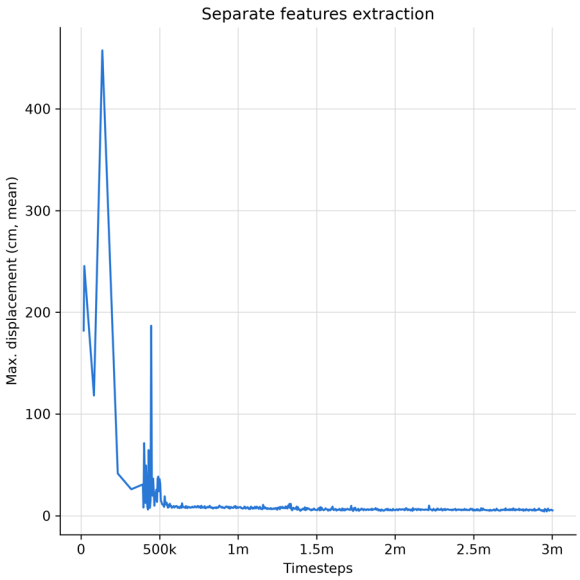

Progress reward exponent
Terminated episodes per rollout (%)
7
6
5
4
3
2
1
0
0
200k
400k
600k
800k
1m
Timesteps

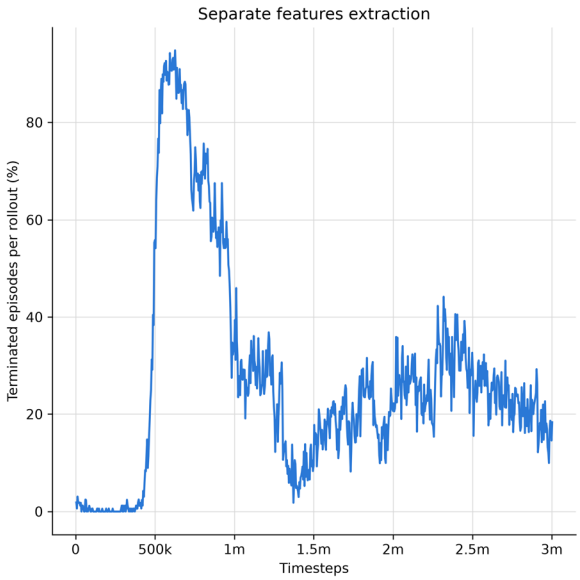

First FEA run
Max. displacement (cm, mean)
1000
800
600
400
200
0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps

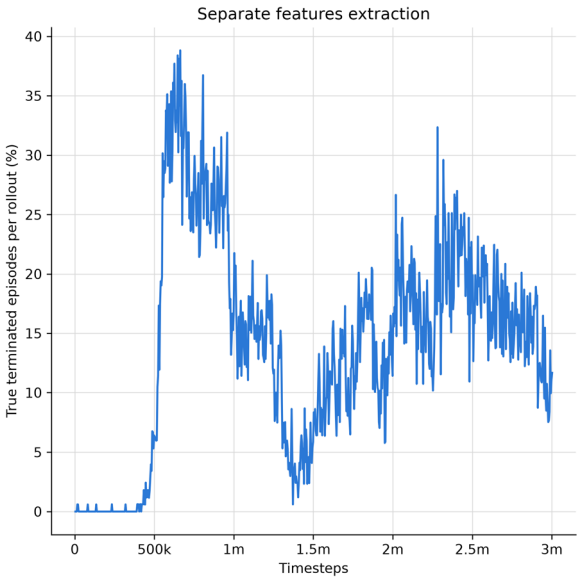

First FEA run
Terminated episodes per rollout (%)
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps

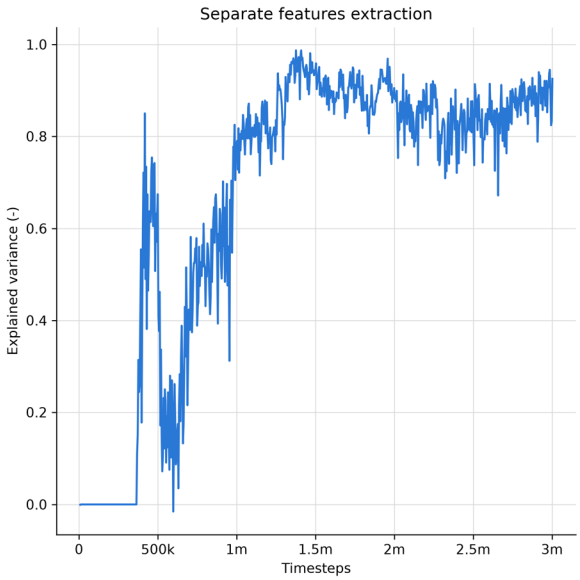

First FEA run
True terminated episodes per rollout (%)
6
5
4
3
2
1
0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps

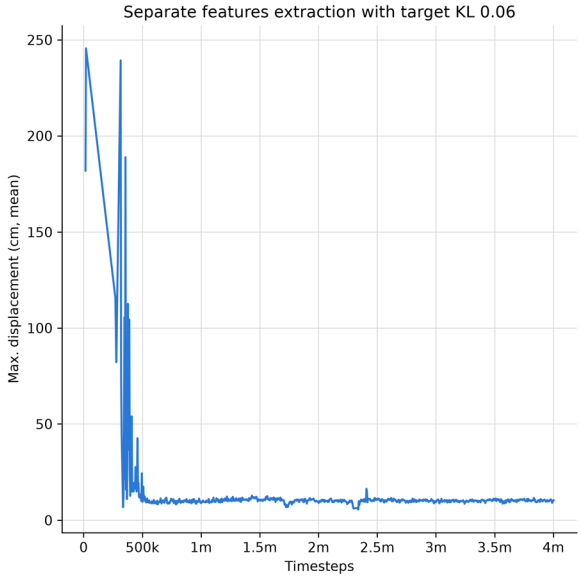

First FEA run
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps

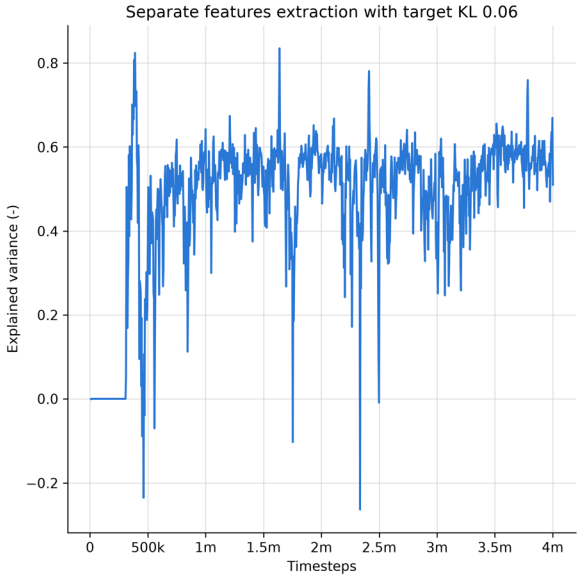

Separate features extraction
Max. displacement (cm, mean)
0
100
200
300
400
0
500k
1m
1.5m
2m
2.5m
3m
Timesteps

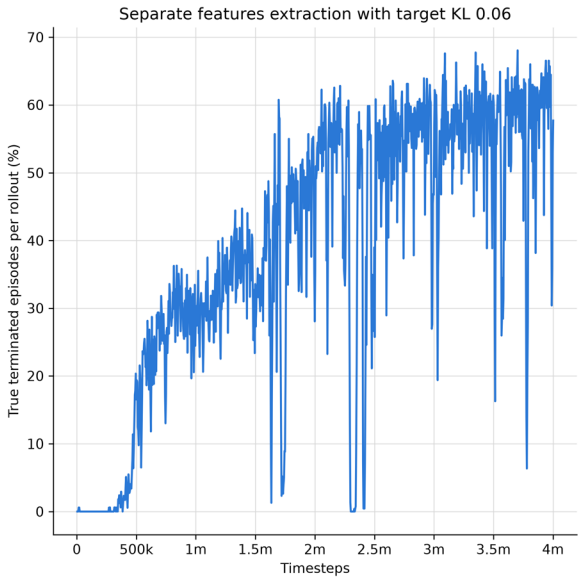

Separate features extraction
Terminated episodes per rollout (%)
0
20
40
60
80
0
500k
1m
1.5m
2m
2.5m
3m
Timesteps

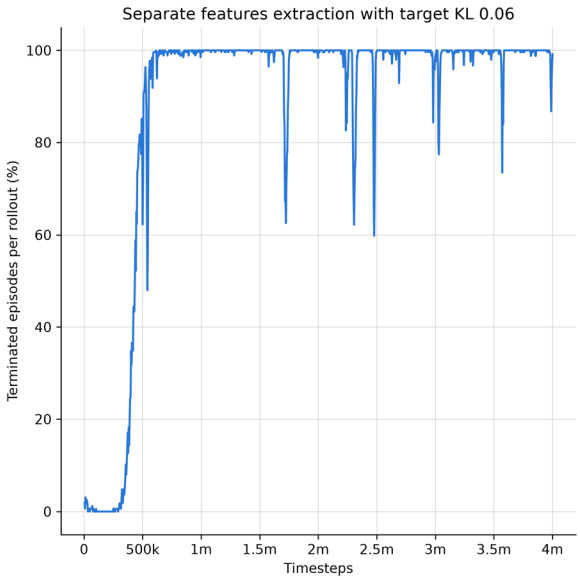

Separate features extraction
True terminated episodes per rollout (%)
0
5
10
15
20
25
30
35
40
0
500k
1m
1.5m
2m
2.5m
3m
Timesteps

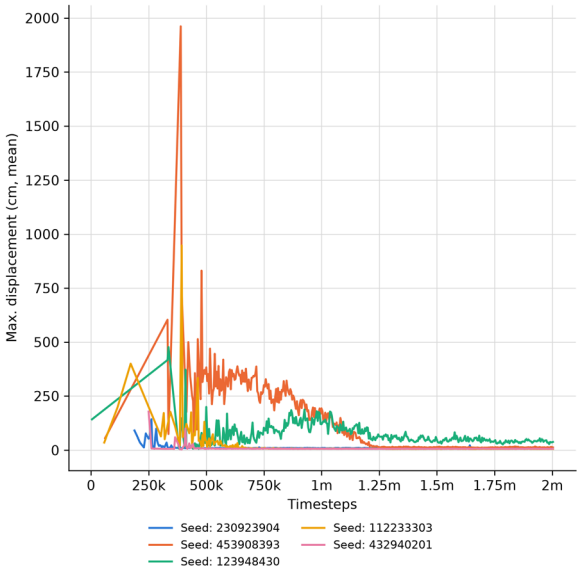

Separate features extraction
Explained variance (-)
0.0
0.2
0.4
0.6
0.8
1.0
0
500k
1m
1.5m
2m
2.5m
3m
Timesteps

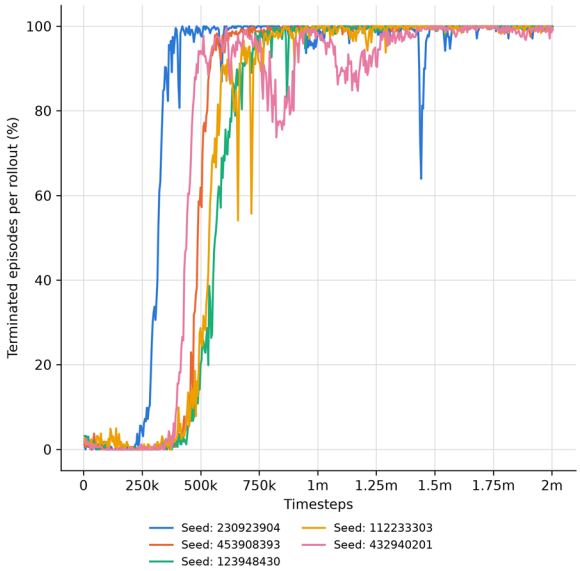

Separate features extraction with target KL 0.06
Max. displacement (cm, mean)
Timesteps

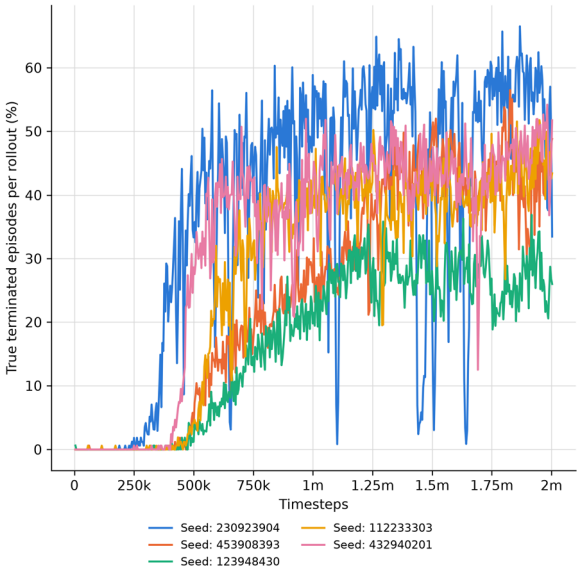

Separate features extraction with target KL 0.06
Explained variance (-)
Timesteps

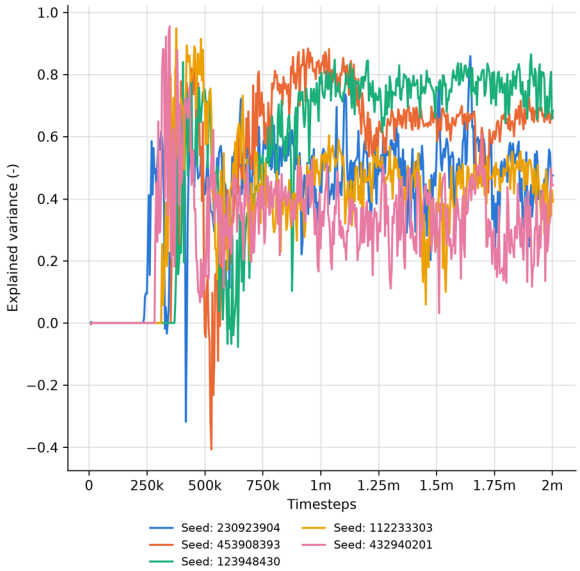

Separate features extraction with target KL 0.06
True terminated episodes per rollout (%)
Timesteps

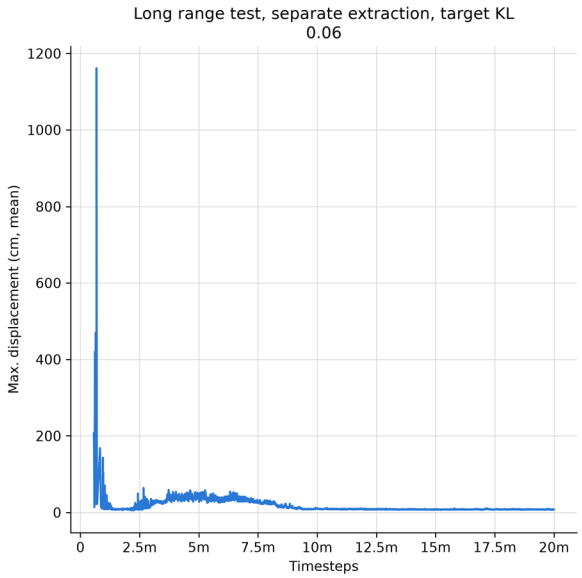

Separate features extraction with target KL 0.06
Terminated episodes per rollout (%)
Timesteps

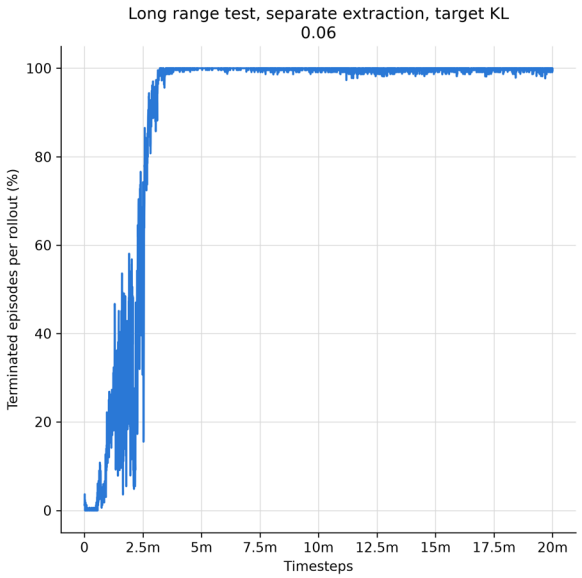

Max. displacement (cm, mean)
2000
1750
1500
1250
1000
750
500
250
0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps
Seed: 230923904
Seed: 453908393
Seed: 123948430
Seed: 112233303
Seed: 432940201

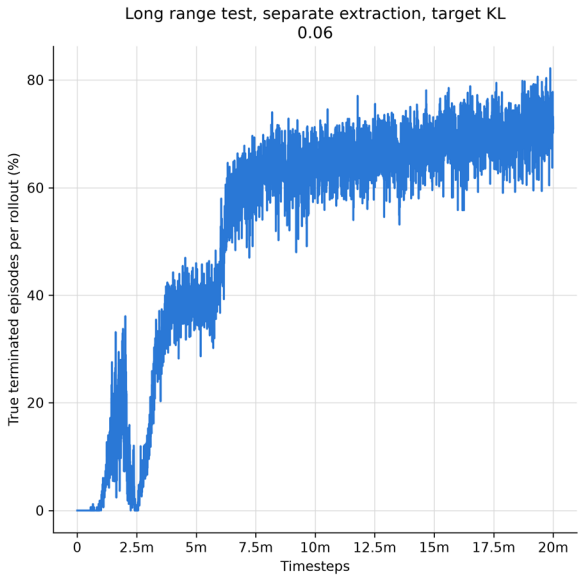

Terminated episodes per rollout (%)
100
80
60
40
20
0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps
Seed: 230923904
Seed: 453908393
Seed: 123948430
Seed: 112233303
Seed: 432940201

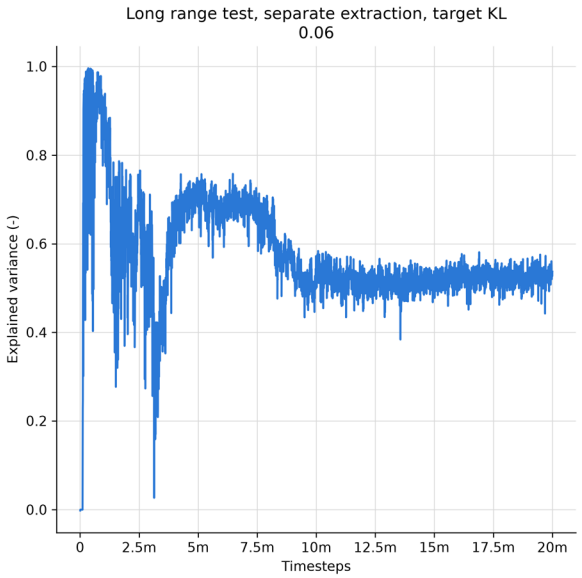

True terminated episodes per rollout (%)
60
50
40
30
20
10
0
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps
Seed: 230923904
Seed: 453908393
Seed: 123948430
Seed: 112233303
Seed: 432940201

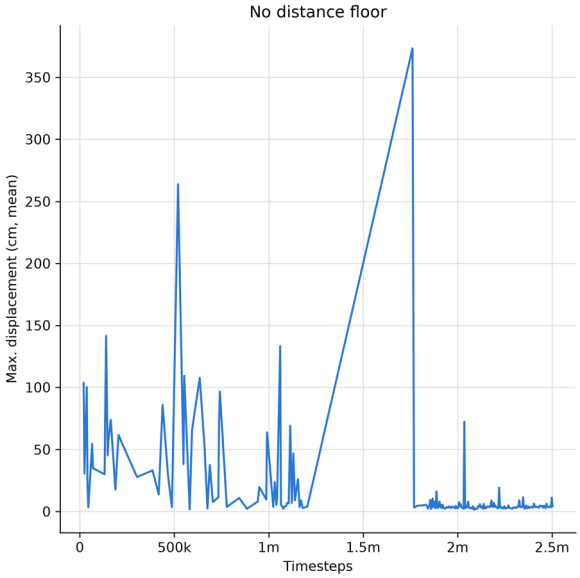

Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
250k
500k
750k
1m
1.25m
1.5m
1.75m
2m
Timesteps
Seed: 230923904
Seed: 453908393
Seed: 123948430
Seed: 112233303
Seed: 432940201

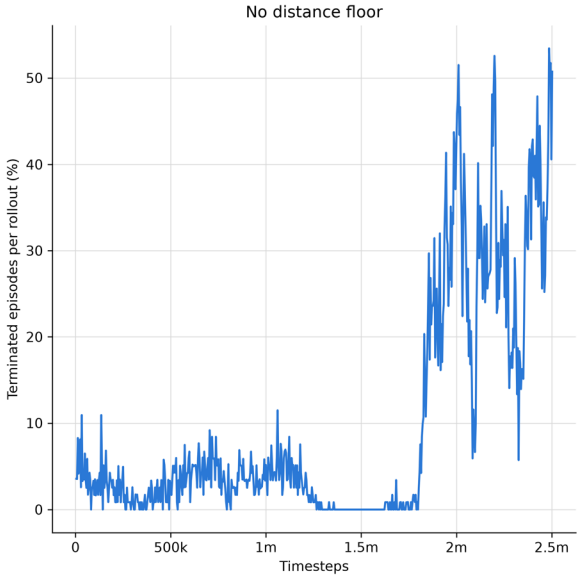

Long range test, separate extraction, target KL 0.06
Max. displacement (cm, mean)
1200
1000
800
600
400
200
0
0
2.5m
5m
7.5m
10m
12.5m
15m
17.5m
20m
Timesteps

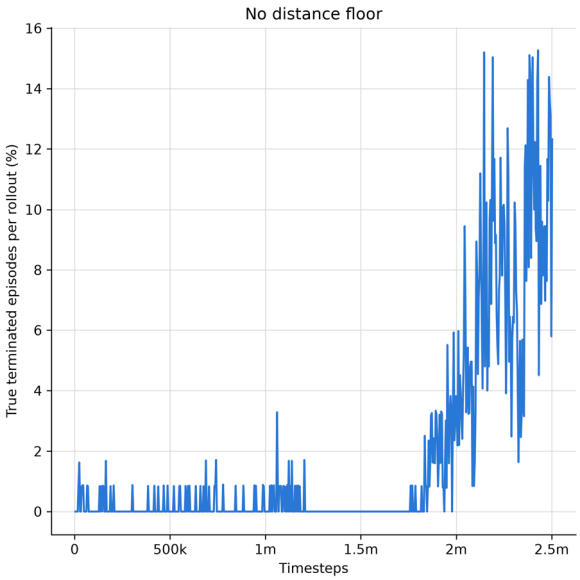

Long range test, separate extraction, target KL 0.06
Terminated episodes per rollout (%)
100
80
60
40
20
0
0
2.5m
5m
7.5m
10m
12.5m
15m
17.5m
20m
Timesteps

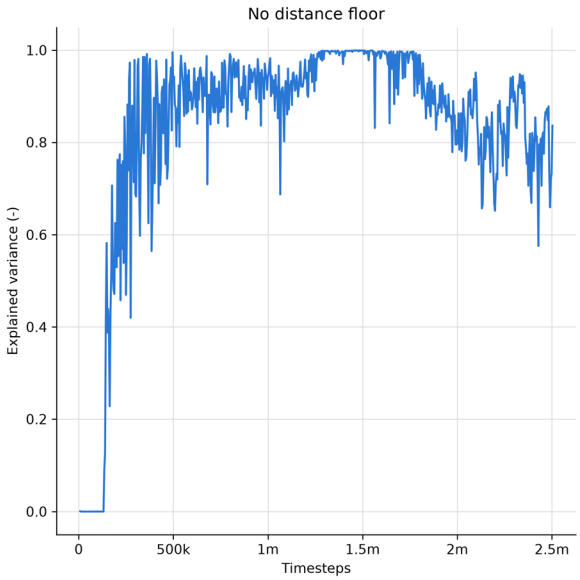

Long range test, separate extraction, target KL 0.06
True terminated episodes per rollout (%)
80
60
40
20
0
0
2.5m
5m
7.5m
10m
12.5m
15m
17.5m
20m
Timesteps

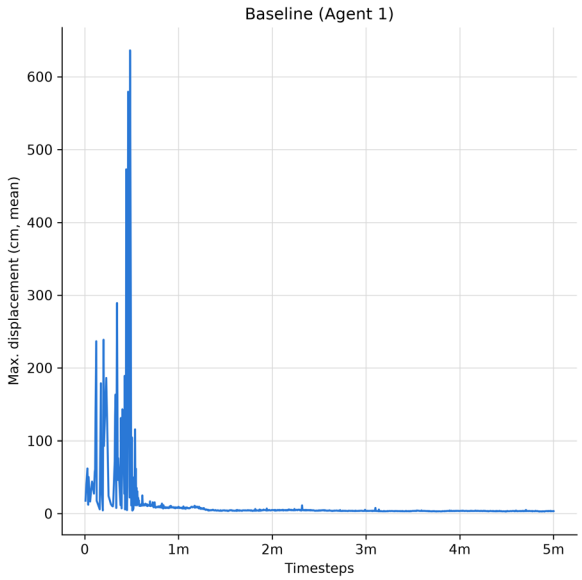

Long range test, separate extraction, target KL 0.06
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
0
2.5m
5m
7.5m
10m
12.5m
15m
17.5m
20m
Timesteps

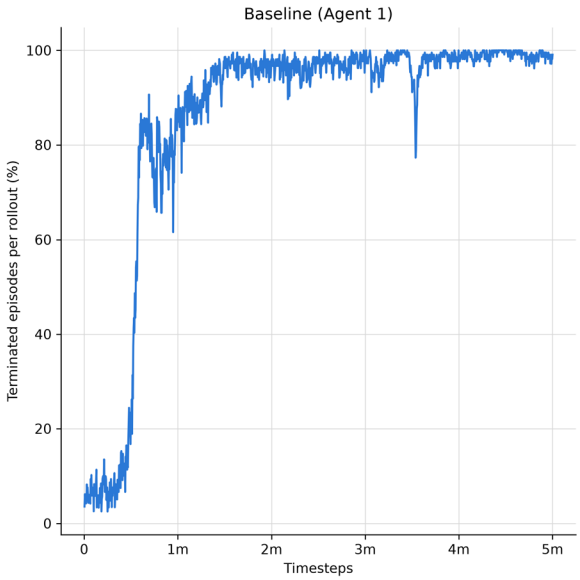

No distance floor
Max. displacement (cm, mean)
0
50
100
150
200
250
300
350
0
500k
1m
1.5m
2m
2.5m
Timesteps

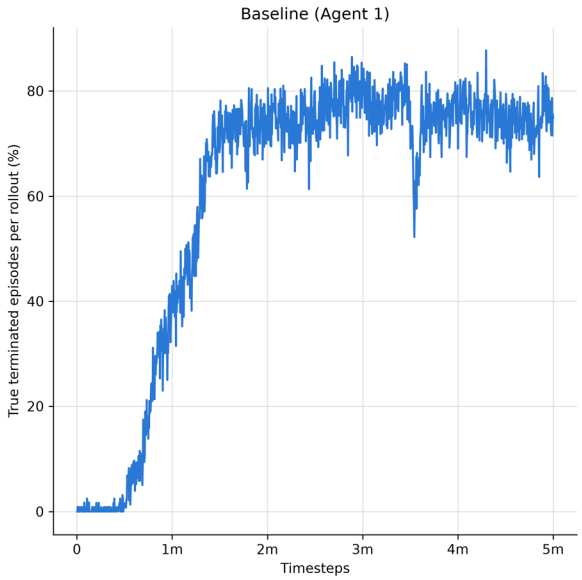

No distance floor
Terminated episodes per rollout (%)
0
10
20
30
40
50
0
500k
1m
1.5m
2m
2.5m
Timesteps

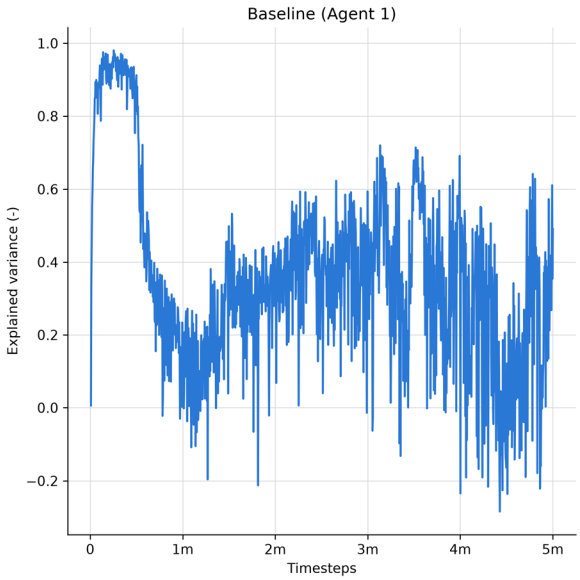

No distance floor
True terminated episodes per rollout (%)
0
2
4
6
8
10
12
14
16
0
500k
1m
1.5m
2m
2.5m
Timesteps

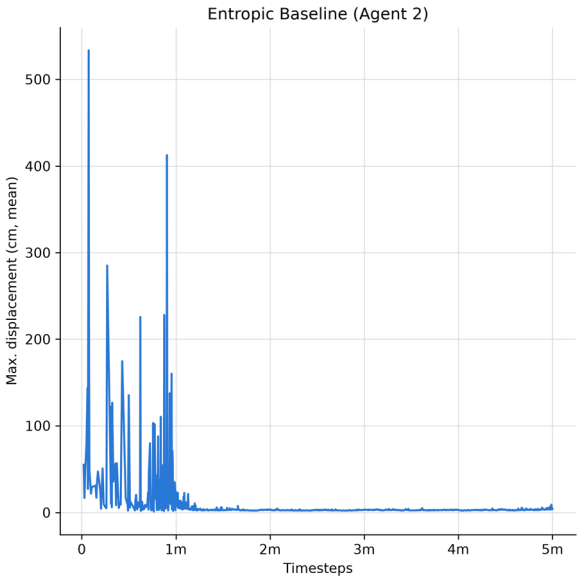

No distance floor
Explained variance (-)
0.0
0.2
0.4
0.6
0.8
1.0
0
500k
1m
1.5m
2m
2.5m
Timesteps

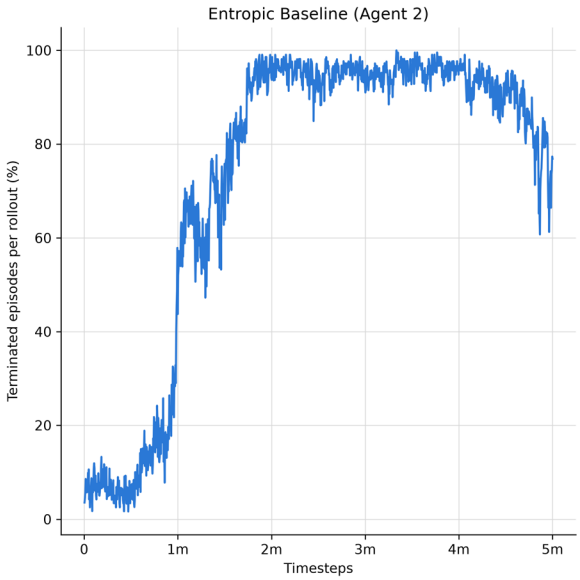

Baseline (Agent 1)
Max. displacement (cm, mean)
0
100
200
300
400
500
600
0
1m
2m
3m
4m
5m
Timesteps

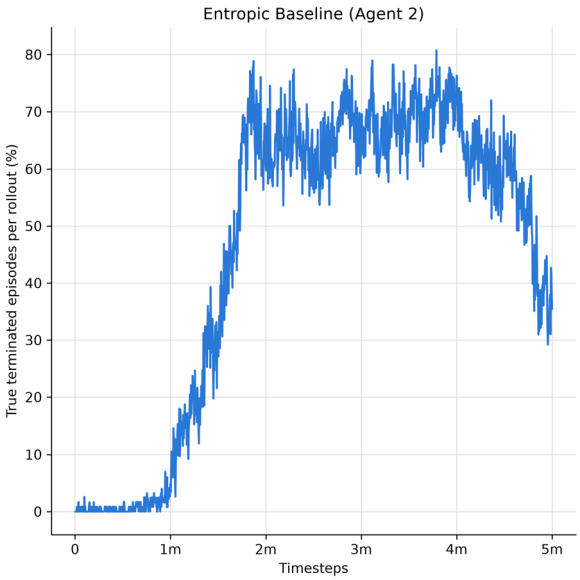

Baseline (Agent 1)
Terminated episodes per rollout (%)
0
20
40
60
80
100
0
1m
2m
3m
4m
5m
Timesteps

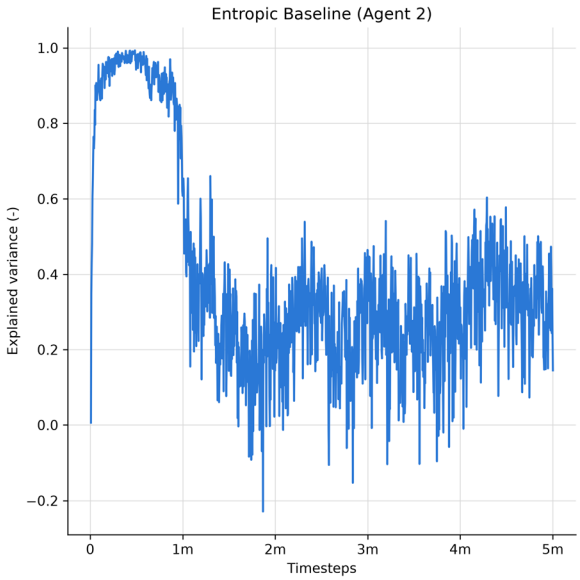

Baseline (Agent 1)
True terminated episodes per rollout (%)
0
20
40
60
80
0
1m
2m
3m
4m
5m
Timesteps

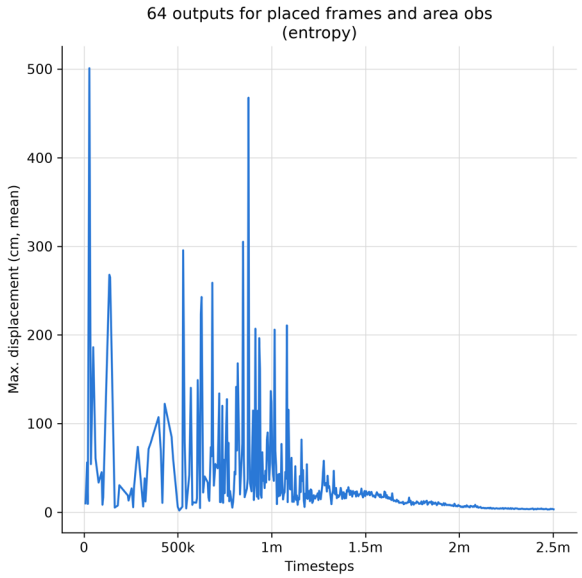

Baseline (Agent 1)
Explained variance (-)
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
0
1m
2m
3m
4m
5m
Timesteps

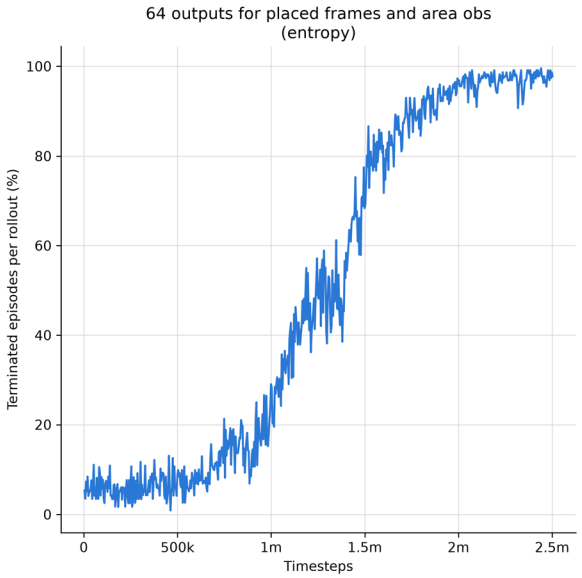

Entropic Baseline (Agent 2)
Max. displacement (cm, mean)
0
100
200
300
400
500
0
1m
2m
3m
4m
5m
Timesteps

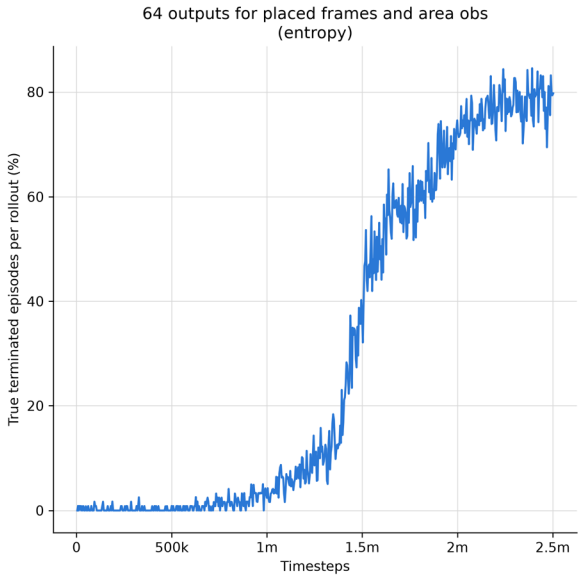

Entropic Baseline (Agent 2)
Terminated episodes per rollout (%)
0
20
40
60
80
100
0
1m
2m
3m
4m
5m
Timesteps

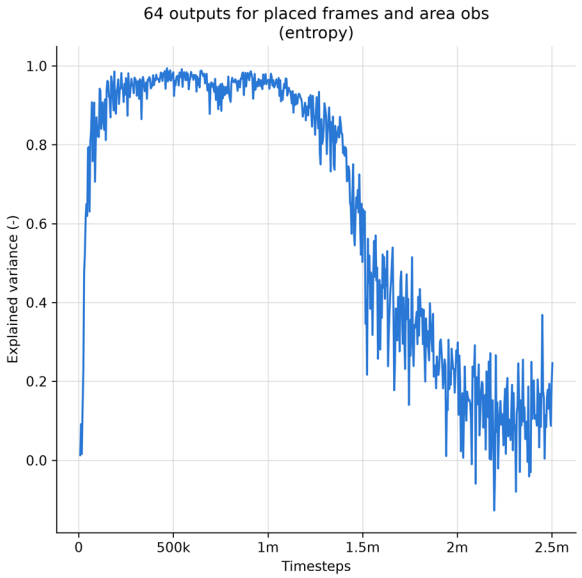

Entropic Baseline (Agent 2)
True terminated episodes per rollout (%)
0
10
20
30
40
50
60
70
80
0
1m
2m
3m
4m
5m
Timesteps

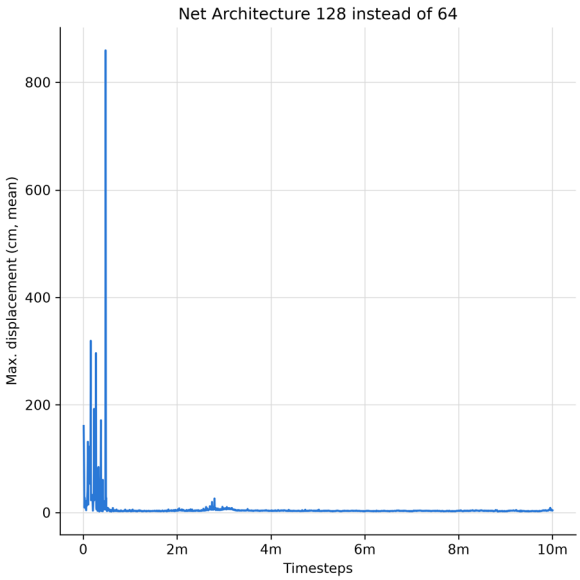

Entropic Baseline (Agent 2)
Explained variance (-)
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
0
1m
2m
3m
4m
5m
Timesteps

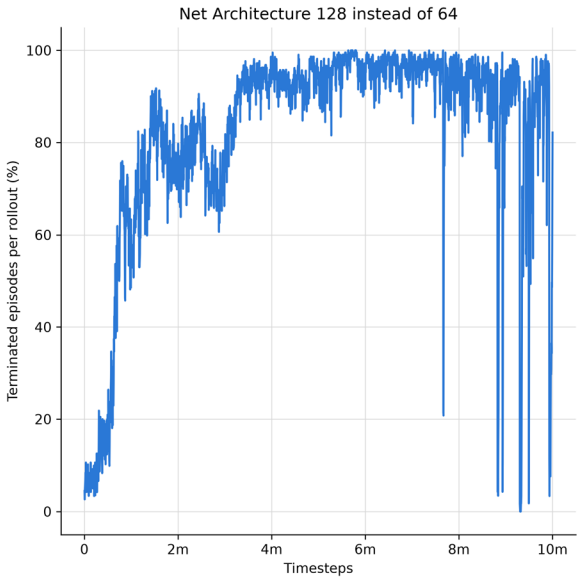

64 outputs for placed frames and area obs
(entropy)
Max. displacement (cm, mean)
0
100
200
300
400
500
0
500k
1m
1.5m
2m
2.5m
Timesteps

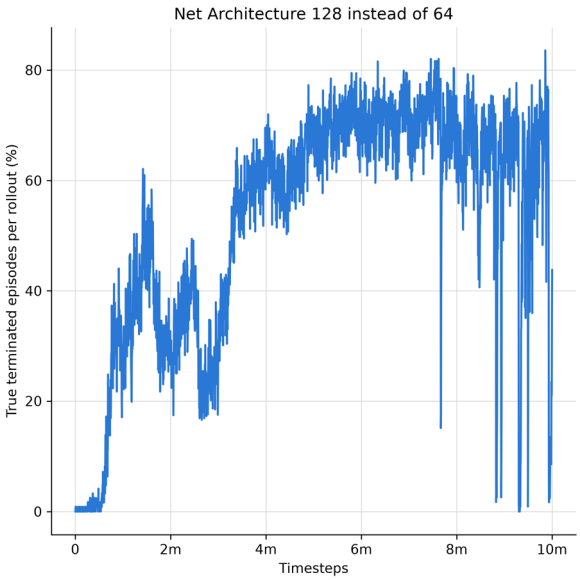

64 outputs for placed frames and area obs
(entropy)
Terminated episodes per rollout (%)
0
20
40
60
80
100
0
500k
1m
1.5m
2m
2.5m
Timesteps

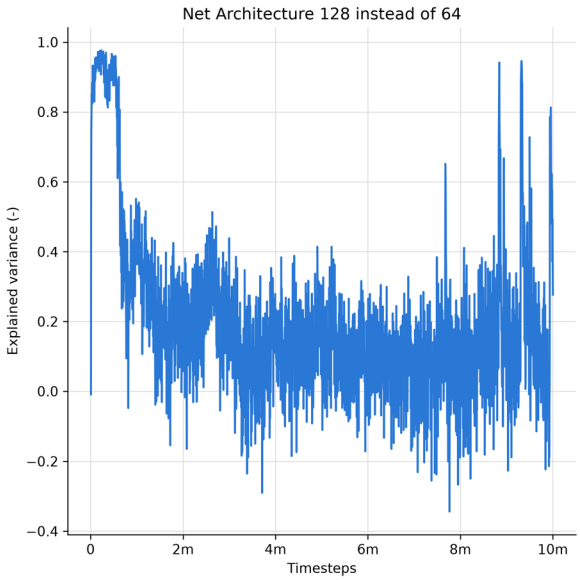

64 outputs for placed frames and area obs
(entropy)
True terminated episodes per rollout (%)
0
20
40
60
80
0
500k
1m
1.5m
2m
2.5m
Timesteps

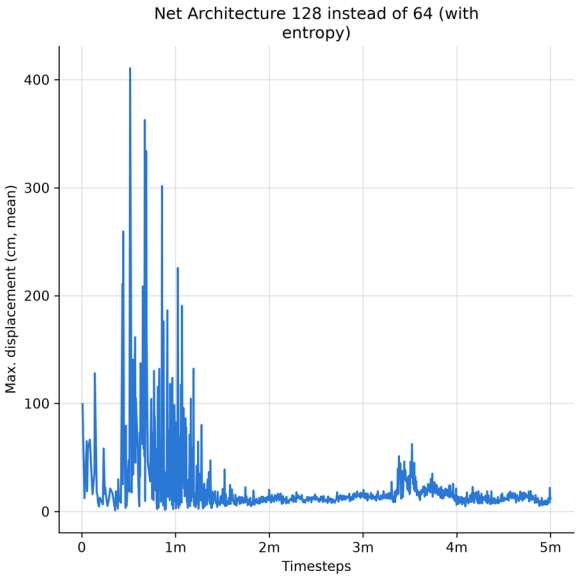

64 outputs for placed frames and area obs
(entropy)
Explained variance (-)
0.0
0.2
0.4
0.6
0.8
1.0
0
500k
1m
1.5m
2m
2.5m
Timesteps

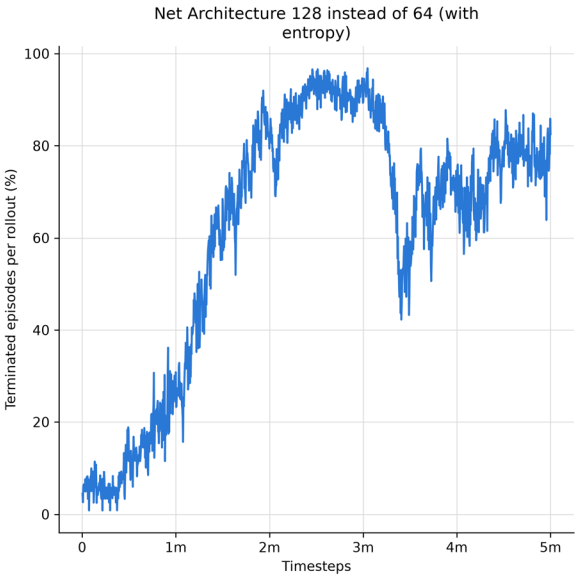

Net Architecture 128 instead of 64
Max. displacement (cm, mean)
800
600
400
200
0
0
2m
4m
6m
8m
10m
Timesteps

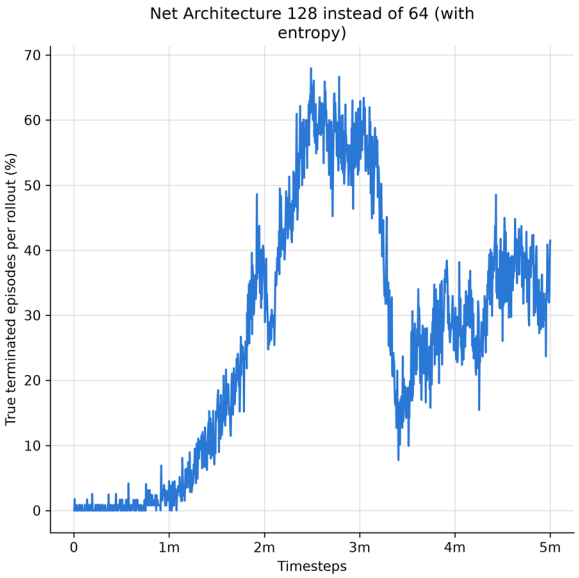

Net Architecture 128 instead of 64
Terminated episodes per rollout (%)
100
80
60
40
20
0
0
2m
4m
6m
8m
10m
Timesteps

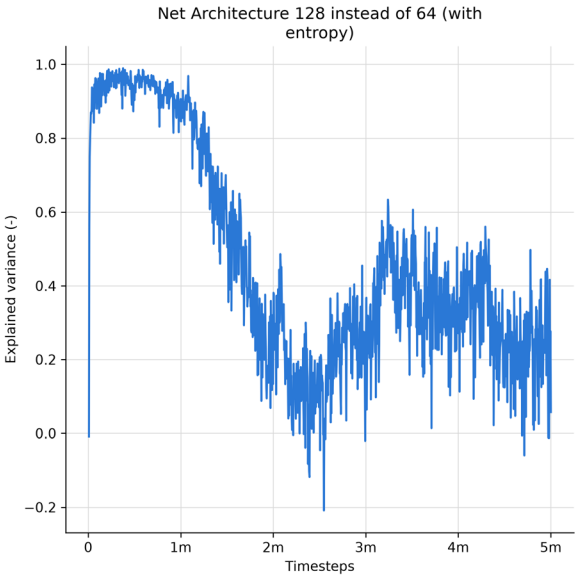

Net Architecture 128 instead of 64
True terminated episodes per rollout (%)
80
60
40
20
0
0
2m
4m
6m
8m
10m
Timesteps

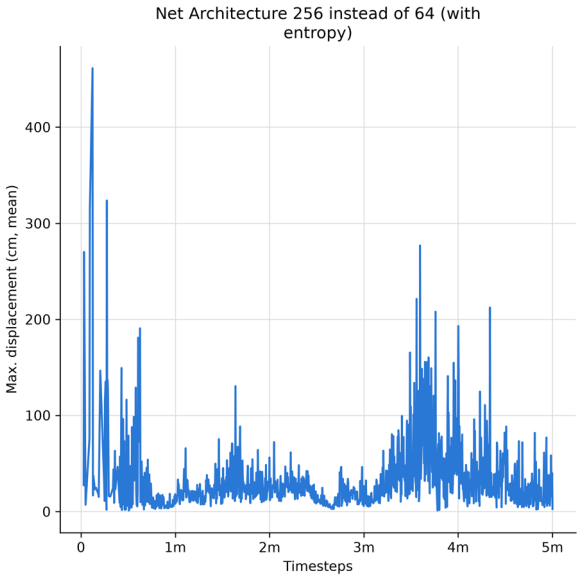

Net Architecture 128 instead of 64
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
2m
4m
6m
8m
10m
Timesteps

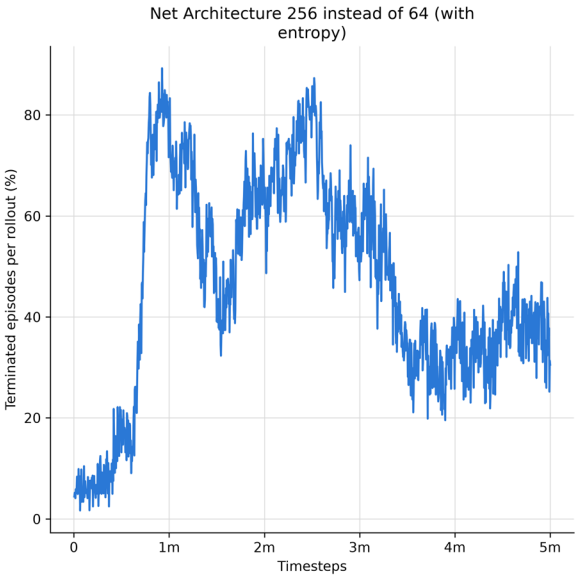

Net Architecture 128 instead of 64 (with entropy)
Max. displacement (cm, mean)
Timesteps
0
100
200
300
400
0
1m
2m
3m
4m
5m

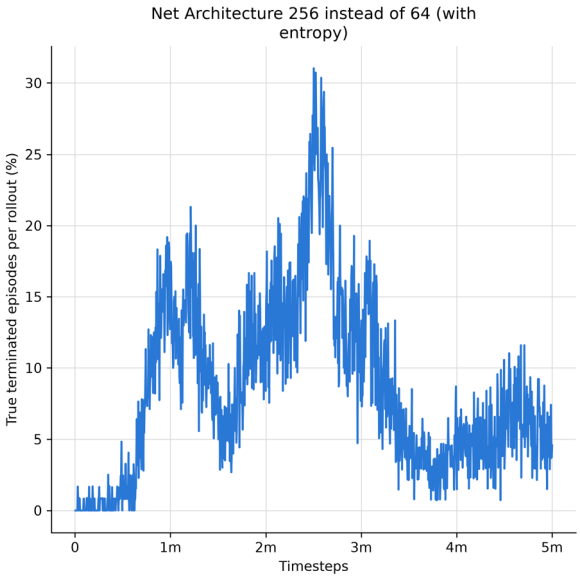

Net Architecture 128 instead of 64 (with entropy)
Terminated episodes per rollout (%)
Timesteps
0
20
40
60
80
100
0
1m
2m
3m
4m
5m

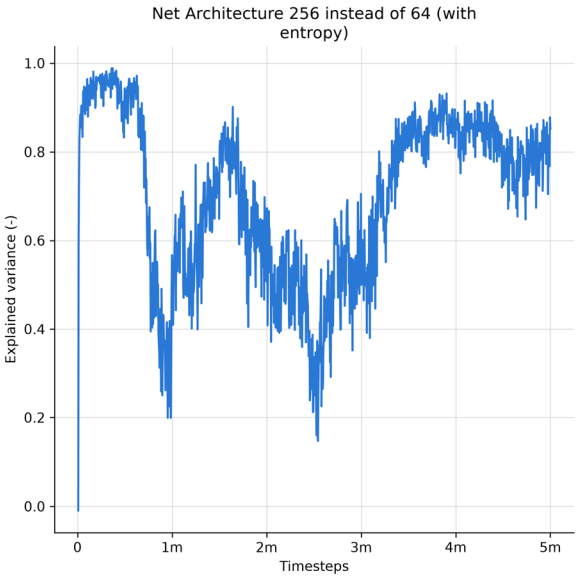

Net Architecture 128 instead of 64 (with entropy)
True terminated episodes per rollout (%)
Timesteps
0
10
20
30
40
50
60
70
0
1m
2m
3m
4m
5m

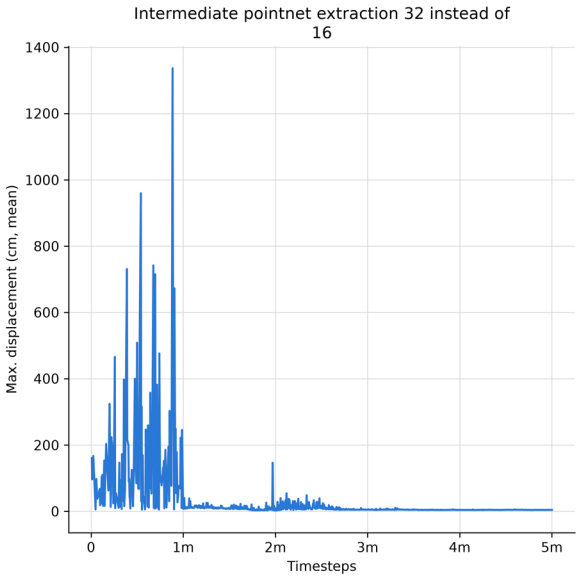

Net Architecture 128 instead of 64 (with entropy)
Explained variance (-)
Timesteps
−0.2
0.0
0.2
0.4
0.6
0.8
1.0
0
1m
2m
3m
4m
5m

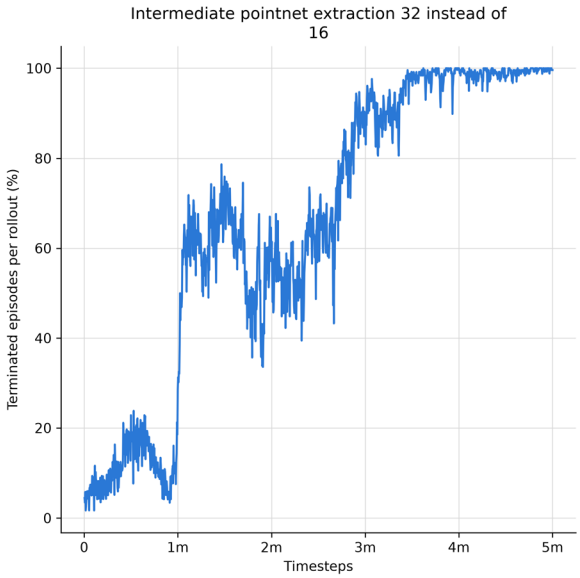

Net Architecture 256 instead of 64 (with entropy)
Max. displacement (cm, mean)
0
100
200
300
400
0
1m
2m
3m
4m
5m
Timesteps

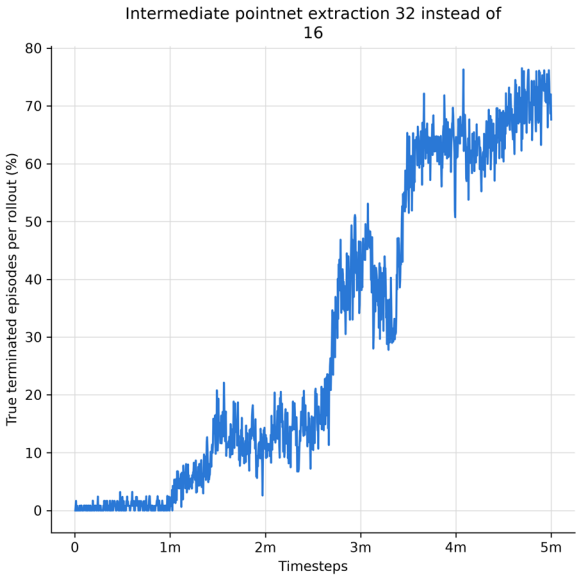

Net Architecture 256 instead of 64 (with entropy)
Terminated episodes per rollout (%)
0
20
40
60
80
0
1m
2m
3m
4m
5m
Timesteps

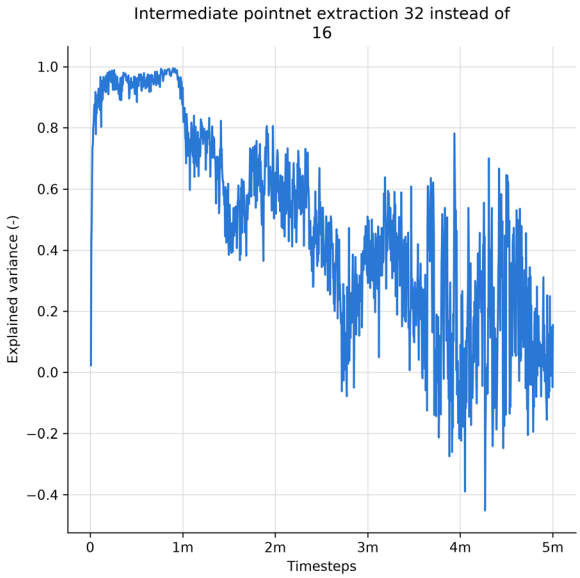

Net Architecture 256 instead of 64 (with entropy)
True terminated episodes per rollout (%)
0
5
10
15
20
25
30
0
1m
2m
3m
4m
5m
Timesteps

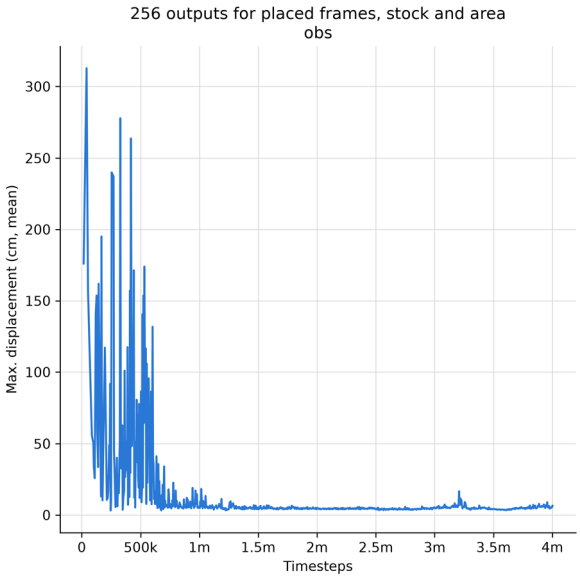

Net Architecture 256 instead of 64 (with entropy)
Explained variance (-)
0.0
0.2
0.4
0.6
0.8
1.0
0
1m
2m
3m
4m
5m
Timesteps

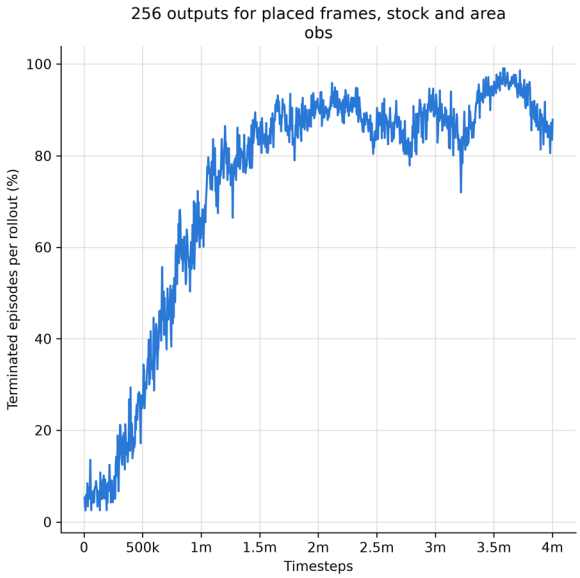

Intermediate pointnet extraction 32 instead of 16
Max. displacement (cm, mean)
1400
1200
1000
800
600
400
200
0
0
1m
2m
3m
4m
5m
Timesteps

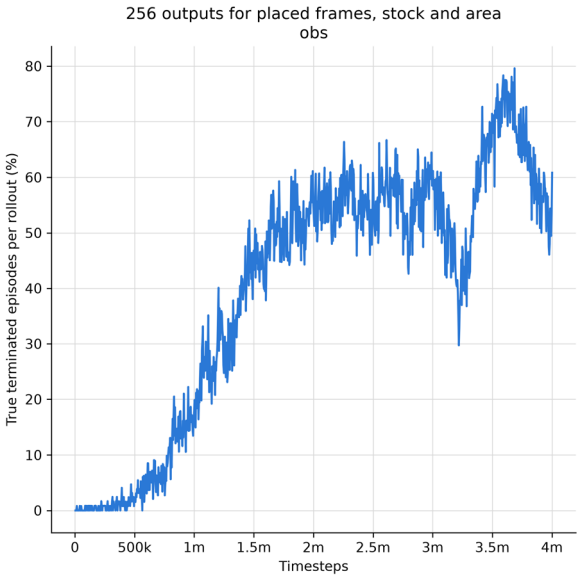

Intermediate pointnet extraction 32 instead of 16
Terminated episodes per rollout (%)
100
80
60
40
20
0
0
1m
2m
3m
4m
5m
Timesteps

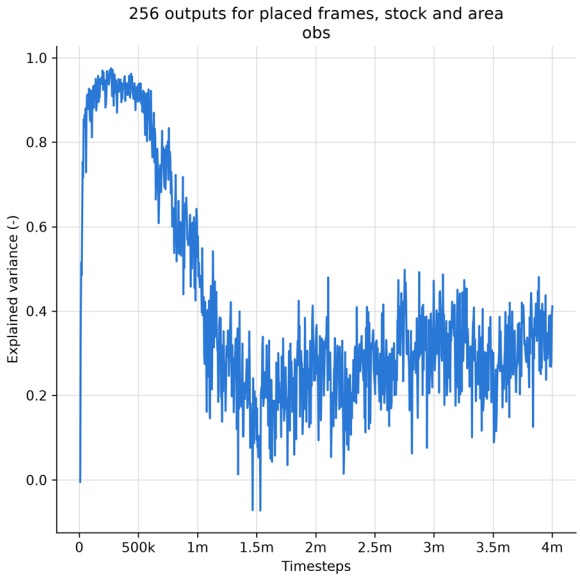

Intermediate pointnet extraction 32 instead of 16
True terminated episodes per rollout (%)
80
70
60
50
40
30
20
10
0
0
1m
2m
3m
4m
5m
Timesteps

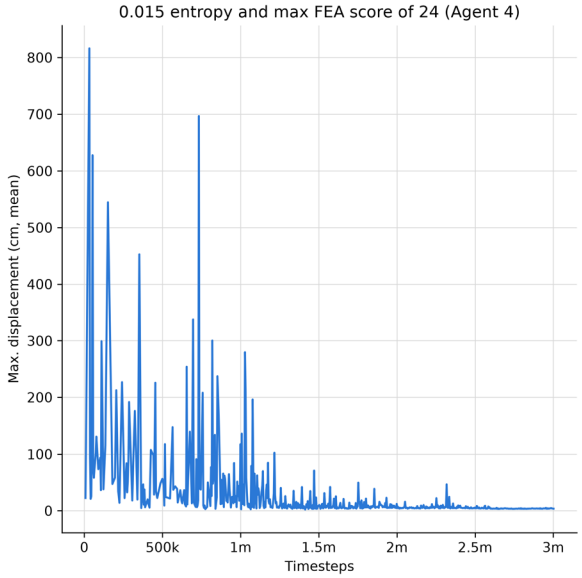

Intermediate pointnet extraction 32 instead of 16
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
0
1m
2m
3m
4m
5m
Timesteps

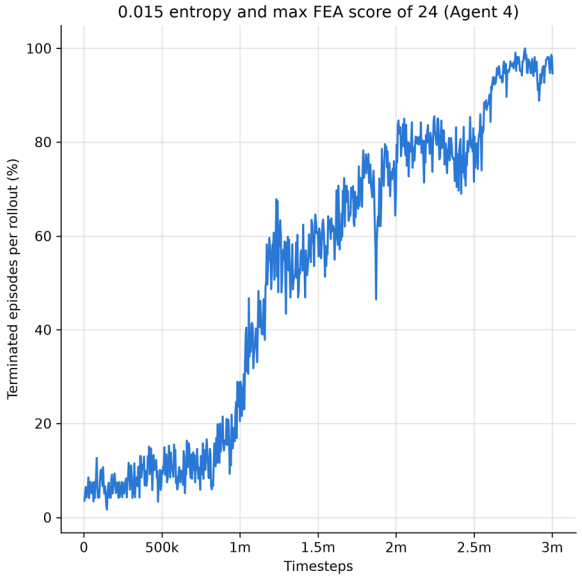

256 outputs for placed frames, stock and area obs
Max. displacement (cm, mean)
Timesteps
0
50
100
150
200
250
300
0
500k
1m
1.5m
2m
2.5m
3m
3.5m
4m

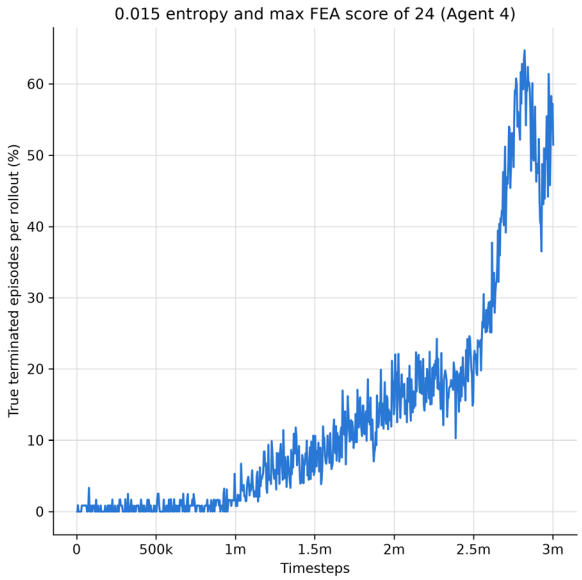

256 outputs for placed frames, stock and area obs
Terminated episodes per rollout (%)
Timesteps
0
20
40
60
80
100
0
500k
1m
1.5m
2m
2.5m
3m
3.5m
4m

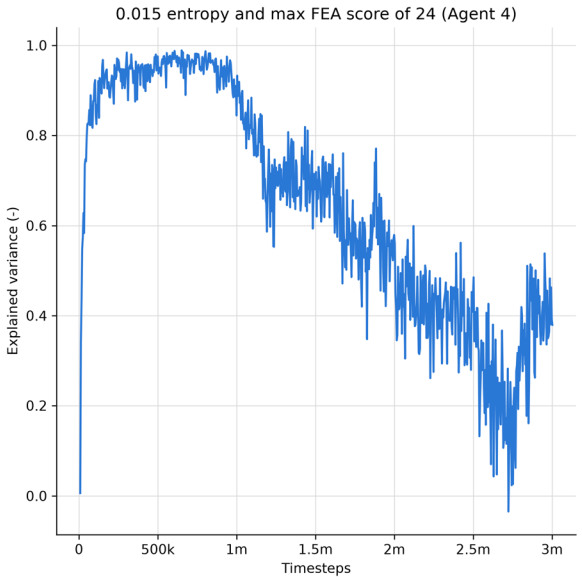

256 outputs for placed frames, stock and area obs
True terminated episodes per rollout (%)
Timesteps
0
10
20
30
40
50
60
70
80
0
500k
1m
1.5m
2m
2.5m
3m
3.5m
4m

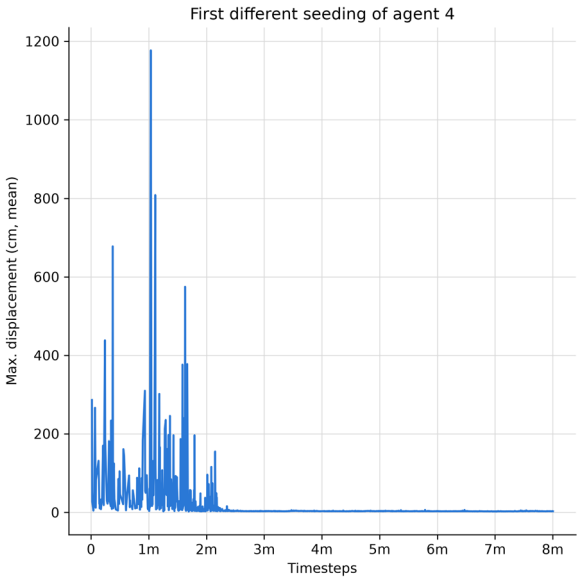

256 outputs for placed frames, stock and area obs
Explained variance (-)
Timesteps
0.0
0.2
0.4
0.6
0.8
1.0
0
500k
1m
1.5m
2m
2.5m
3m
3.5m
4m

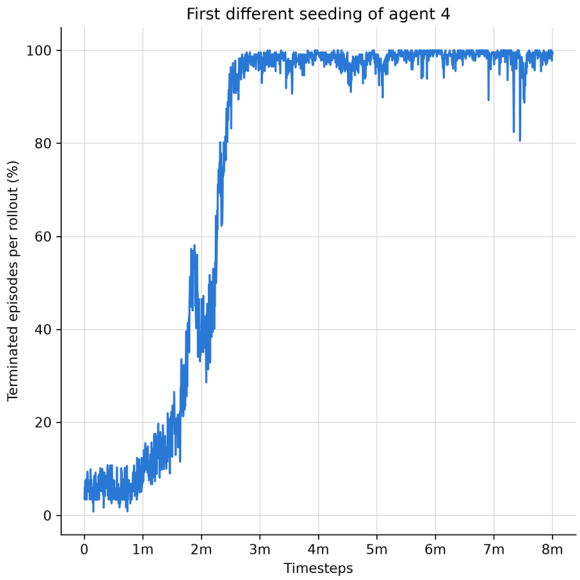

0.015 entropy and max FEA score of 24 (Agent 4)
Max. displacement (cm, mean)
Timesteps
0
500k
1m
1.5m
2m
2.5m
3m
0
100
200
300
400
500
600
700
800

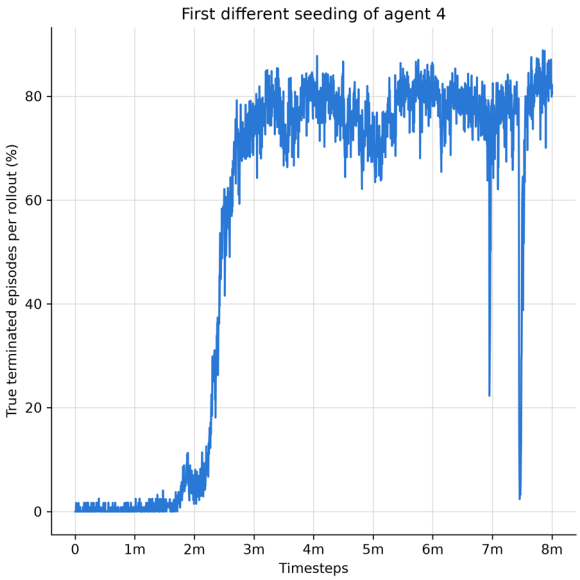

0.015 entropy and max FEA score of 24 (Agent 4)
Terminated episodes per rollout (%)
Timesteps
0
500k
1m
1.5m
2m
2.5m
3m
0
20
40
60
80
100

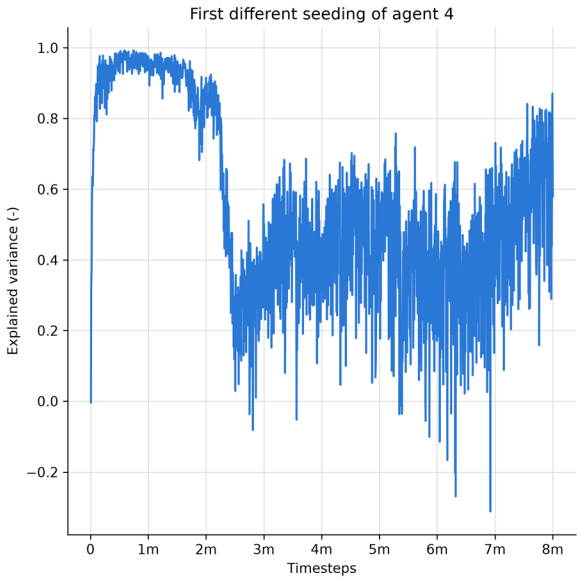

0.015 entropy and max FEA score of 24 (Agent 4)
True terminated episodes per rollout (%)
Timesteps
0
500k
1m
1.5m
2m
2.5m
3m
0
10
20
30
40
50
60

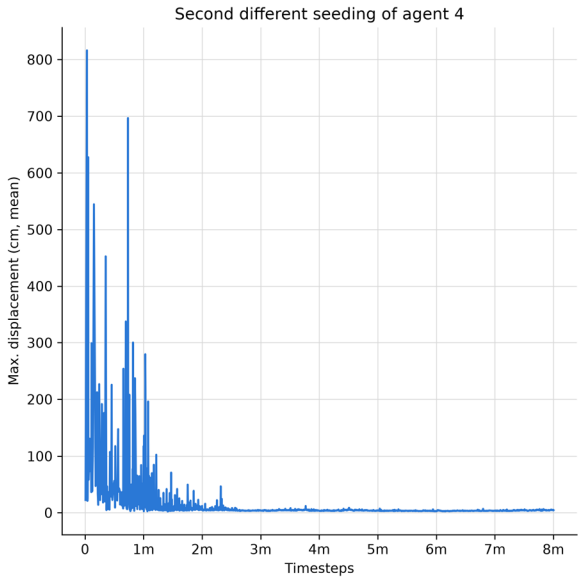

0.015 entropy and max FEA score of 24 (Agent 4)
Explained variance (-)
Timesteps
0
500k
1m
1.5m
2m
2.5m
3m
0.0
0.2
0.4
0.6
0.8
1.0

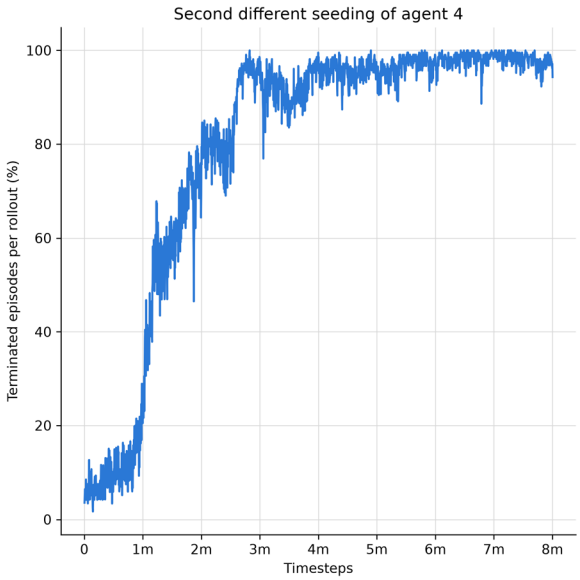

First different seeding of agent 4
Max. displacement (cm, mean)
1200
1000
800
600
400
200
0
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps

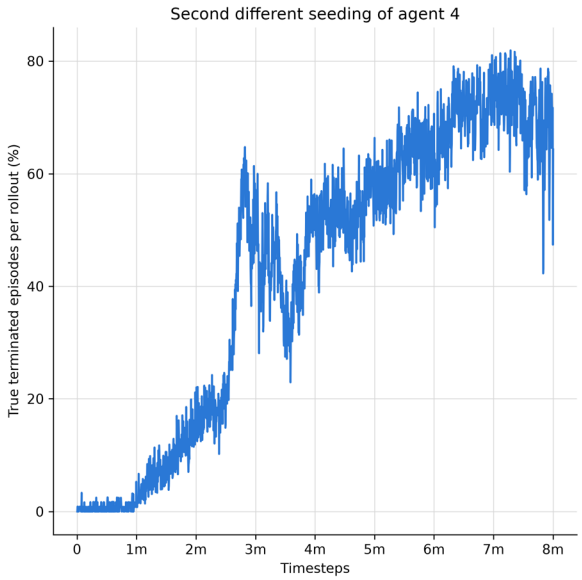

First different seeding of agent 4
Terminated episodes per rollout (%)
100
80
60
40
20
0
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps

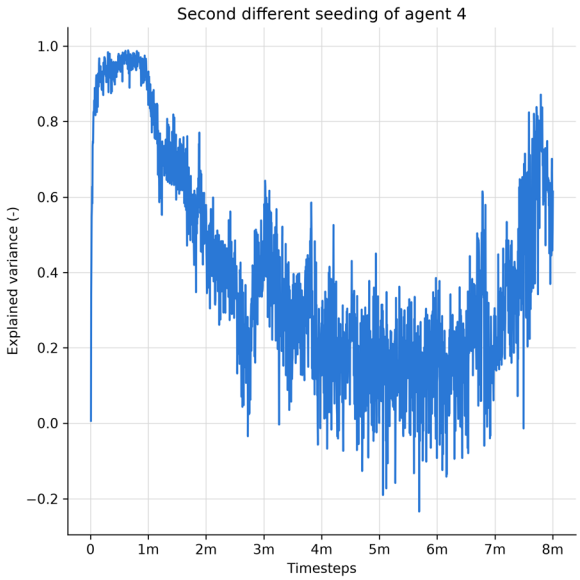

First different seeding of agent 4
True terminated episodes per rollout (%)
80
60
40
20
0
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps


First different seeding of agent 4
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps


Second different seeding of agent 4
Max. displacement (cm, mean)
800
700
600
500
400
300
200
100
0
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps


Second different seeding of agent 4
Terminated episodes per rollout (%)
100
80
60
40
20
0
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps


Second different seeding of agent 4
True terminated episodes per rollout (%)
80
60
40
20
0
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps


Second different seeding of agent 4
Explained variance (-)
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
0
1m
2m
3m
4m
5m
6m
7m
8m
Timesteps