PCG-SAF: Procedural Content Generation via Self-Assembling Figures for Tabletop Games

Fiona Shyne[1], Seth Cooper[1]

[1]Northeastern University

Abstract

Tangibility within tabletop games is an important factor to many gamers. Commercial games can include methods to procedurally generate tangible content using only analog components, however these are limited in their capability: they usually require manual assembly. Within nature, we find many systems that are able to "self-assemble," using the physical properties of components to arrange themselves in response to undirected motion. In this work, we use this process of self-assembly to procedurally generate tangible game figures: miniatures and dice. We iteratively designed bases (self-assembly connection points) that are selective, attractive, and adhesive. We qualitatively evaluated this design, and found that they are successfully able to self-assemble, although improvements can still be made. We have complied our work into a toolkit that hobbyists with the necessary materials can use to produce self-assembling game elements.

Keywords

procedural content generation, self-assembly, tabletop games

1. Introduction

Tabletop gaming allows a unique experience that cannot be replicated in digital gaming. The value of tangibility can be seen especially in culture around dice and miniatures. Even with digital dice, players still covet their dice collections. War game players will spend hours hand painting their miniatures [1]. Artisans can be found across internet sites such as Etsy, conventions, and fairs, selling dice, miniatures, and customized gaming pieces. Players will even go so far as to personify their game piece collections, placing dice that have rolled poorly in "dice jails" or talking about miniatures as if they were alive.

Dice also play an interesting role mechanically in tabletop games. By introducing randomness, dice and other elements, make each play-through of a game a unique experience. These random components can even be used to procedurally generate content for tabletop games. For example the Dungeon and Dragons Dungeon Master Guide provides a dice look-up table for procedurally generating combat encounters or dungeons. Games, such as Catan and Carcassonne, have players build a map by placing tiles either in a random fashion or as a result of gameplay. These tiles could be thought of as a "map grammar," the same way computational grammars have been used to generate content. Still, others such as Threadsteading [2] and eBee [3] produce physical artifacts (e.g. a quilt).

While existing methods of generating content for tabletop games lead to interesting game-play opportunities, the possibility space for tangible procedural generation is still greatly under-explored. Existing techniques largely either require manually assembly or control (e.g. Catan) or do not directly result in tangible artifacts (e.g. lookup tables). This work seeks to expand what is possible for tabletop games through Procedural Content Generation via Self-Assembling Figures (PCG-SAF). By "figure" we refer to any small, physical component of gameplay; in this work look specifically miniatures and dice.

Self-assembly is a concept that has its roots in chemistry and microbiology [4]. Fundamentally, self-assembly describes components that arrange themselves into appropriate positions through undirected motion provided by the environment. Through local interactions, different components become either connected, disconnected, or ignored, based on the physical properties of the components, and over

Joint AIIDE Workshop on Experimental Artificial Intelligence in Games and Intelligent Narrative Technologies, November 10-11, 2025, Edmonton, Canada.
shyne.f@northeastern.edu (F. Shyne); se.cooper@northeastern.edu (S. Cooper)
0009-0009-9222-5242 (F. Shyne); 0000-0003-4504-0877 (S. Cooper)

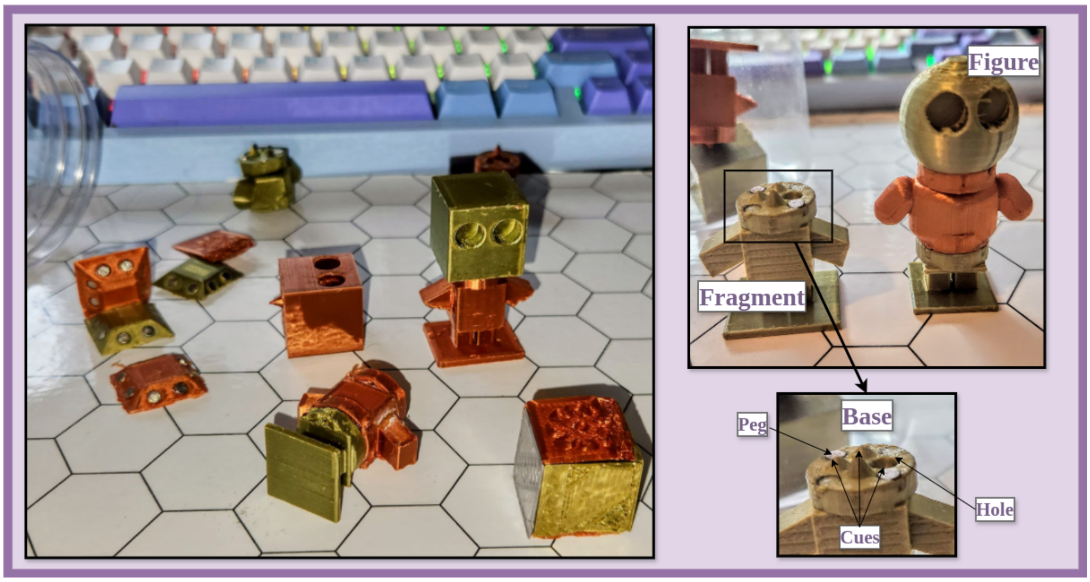

Figure 1: Final Version of Self-Assembling Figures and labeled schematic

time global structures form. Researchers in macro self-assembly have taken inspiration from chemistry, and propose that self-assembly can be a construction method for man-made structures, for example components of micro electronics [5].

This project presents a method for creating self-assembling figures for tabletop games (see Figure 1). We have taken models of game figures, specifically dice and character miniatures, and broken them into smaller fragments that can be self-assembled in a mix-and-match manner when shaken. Each fragment is given one or more binding sites, which we refer to as “bases”, that selects and is attracted to specific other bases. By having multiple candidates for each binding site, the final form of each figure is randomly determined by the motion of the player shaking a jar. This process is analogous to computational methods for procedural content generation, for example constructive methods that randomly select combinations of elements. We further demonstrate that self-assembly can not only result in procedurally generated content but procedurally generated generators, in the form of self-assembling dice.

In this paper we present a toolkit for designers to create their own self-assembling figures for tabletop games. Further, we demonstrate some of the gameplay that becomes possible with self-assembling figures, through the creation of a proof-of-concept game “Rattle, Roll, Rumble.” We evaluate the feasibility of our approach, through shake tests across several conditions. We demonstrate that these fragments are successfully able to self-assemble, although the outcomes are sometimes inconsistent. Self-assembly not only provides a fun, novel interaction method for tangible figures, it opens the door to mechanical opportunities for completely analog games. While our example game already shows new opportunities, we believe we are just on the cusp of discovering what is possible when game designers are given self-assembly as a medium for procedural content generation.

2. Related Work

2.1. Tangible Game Design

Tabletop gaming provides a unique experience that cannot be completely replicated in digital entertainment. Rogerson et al. [6] interviewed tabletop gamers and found several themes for what draws players into tabletop gaming, one of which was the materiality of the medium itself. All factors of the material quality of the game were mentioned, from the design of the box and the components to the specific smell new board game pieces have. Tabletop gaming has specific appeal to different populations. Al Mahmud et al. [7] describes how senior citizens are more likely to be drawn to tabletop games, as the themes and mechanics are similar to what they are familiar with and the interaction method is more appreciable. For many, the collection of physical components is a significant part of their hobby. Darzentas et al. [1] conducted an ethnographic study of miniature collectors within the genre of war games (such as Warhammer). To these individuals, the value of the miniatures goes beyond the game, where the collection, customization, and display of miniatures are equally important. Physical components are often more then simple vessels for gameplay, but can become beloved objects admired for their physical properties, aesthetics, or associated with previous gameplay experiences.

Developments in fabrication technology has the potential to expand what is possible for tangible game play. Bhaduri et al. [8] explores the possibility space for how 3D printers can be incorporated into gameplay. They propose that 3D printer can be utilized in a variety of ways including: creating objects that react to player input, revealing hidden objects, creating records of gameplay, or generating map tiles during gameplay. Stemasov et al. [9] explores how the act of creation, through 3D printing pens, can be incorporated into game play. Their project presents a Tabletop roleplaying game, where players craft their weapons, computational system assess these crafts, and a laser cutter can move or destroy tangible components of the game. Our work explores a new possibility space for how fabrication technology can enhance tangible gameplay. Our 3D printable components allow for the physical structure of gamepieces to be randomly constructed through shaking. This allows for different configurations to be randomly assigned during gameplay only using tangible components.

2.2. Procedural Content in Tabletop Games

Procedural content generation (PCG) describes how content in games can be created through algorithmic processes. Computational PCG has been used to create a wide variety of content, both for digital and tabletop games. Shyne and Cooper [10] preformed a literature review on computational tool for tabletop role playing games. This work found that PCG has been used to generate a variety of content for tabletop games, including narrative, maps, dungeons, and even taking the place of a game manager (GM). However, what has been explored in a completely analog capacity has been less explored.

Analog means of procedural generation have a history predating their computational counterparts. Smith [11] describes how tabletop games directed randomness to create content, before computer were able to. They describe two methods they use to accomplish this: modular components (such as map tiles) that fit together, and randomly driven algorithmic design. Guzdial et al. [12] further describes tabletop role-playing games as PCG systems, and discusses how we can use strategies in these games to inform computational systems. Brown and Scirea [13] describes a framework for how tabletop games implement PCG methods. These methods includes: random lookup tables where dice rolls determine which element is chosen, construction of maps using tiles, and procedures that determine turn-order or enemy actions. We can see these methods in a variety of modern tabletop games. Dungeons and Dragons [14], presents a series of lookup tables to generate randomized combat encounters along with rules for randomized turn order. Several games, including Catan [15], Carcassonne [16], and Heroscape [17], all include configurable map tiles where the players use a procedural process to construct the game map itself. While these techniques employ algorithmic and randomized processes, the construction of structures still have to implemented by the players. In this work we present a new method for PCG in tabletop games, where the structure is assembled automatically and randomly, using the randomized motions created through shaking.

2.3. Self-Assembly

Self-assembly describes a wide variety of processes for how individual components automatically assemble into structures, from the smallest of molecules to entire solar systems. Whitesides and Grzybowski [4] describes self-assembly to systems with the following features: systems are made up of pre-existing components, the process is reversible, and the process can be controlled through the design of its components. In a self-assembling system, information is encoded onto individual components, which are able to move freely through the introduction of energy, and components are either attracted to or repelled from each other through their individuals properties. In a micro context, components are often attracted to each other through non-covalent or weak covalent interactions, such as van der Waals bonds. On a macro scale, components are attracted using a variety of forces including magnetic and gravitational attraction. Soute et al. [18] describes how the properties of self assembly, including self-repair, self-replication, and growing and mutating new structures, are useful to large scale construction.

Self-assembling systems share a lot of similarity with computational techniques. Winslow [19], demonstrated that a specific model for self-assembly, staged self-assembly, can be represented with up to a logarithmic increase in size from polynomial context-free grammars. Klavins [20] describes how you can represent a self-assembling system using graph grammars, which can be used to reason about what the final form of a self-assembling system.

Previous self-assembling systems in the centimeter scales have used a variety of techniques. Common among these techniques is the use of magnets to attract and repel pieces. Majumder and Reif [21] provides a framework for 2D tiles that self-assemble, where the polarity of the magnets on each side of the tile determines which other tiles it attracts. Jílek et al. [22] implements a strategy similar to this, with 3D printed pieces that contain slots for inserting magnets into the sides. They also curve the edges of each piece to assisted with "guiding" adjacent components into the right position. However, this manufacturing process can be tedious and require a substantial amount of individual magnets to define different interactions. Nisser et al. [23] addresses these concerns with their design of magnetic cubes that can be "programmed" using a specialized machine that can set the polarity of magnets. On each side of these cubes is a magnetic sheet, where the polarity each individual "pixel" can be set and edited by the machine. While this process provides existing opportunities, it requires access to specialized and expensive equipment that most people, our lab included, do not have access to. Hacohen et al. [24] proposes another method for selection. In this work, each piece contains only one magnet, and selection is determined by having extruding cones (pegs), or indentations (holes). Our work uses this approach as a basis, iterating on this design, to create a toolkit that is accessible to hobbyists with moderate technical experience.

3. Design Process

Here we describe our design goals and iterative prototyping process, along with the final product and an example game created with these figures.

3.1. Design Goals

The goal of this project was to create a system to produce gameplay figures (i.e. minatures and dice) that self-assemble, in a short time frame, when being shaken by a human player in a jar. We found no existing models that fit this goal. Existing work, 1) did not have any publicly accessible models, and 2) looked at figures that self-assemble when machine generated motion is applied and only fully assemble after several minutes. This was not ideal for our goal, as we wanted players to explicitly to participate in the self-assembly process. We did find existing models, on the Thingiverse [1] website, for self-assembling components that were meant to be shaken. However, these models only allow for one repeating model to be present in the existing structure. For example one model, is supposed to represent a virus and consists of 12 identical pieces that form a dodecagon. The problem with this model is there is no method of selection, all pieces are attracted to each other.

3.2. Problem Definition

With our goal in mind we developed a set of terminology to describe different components of the project (see Figure 1 for labels). In our project a *figure*, is a collection of self-assembling *fragments* that are fully

[1]https://www.thingiverse.com/

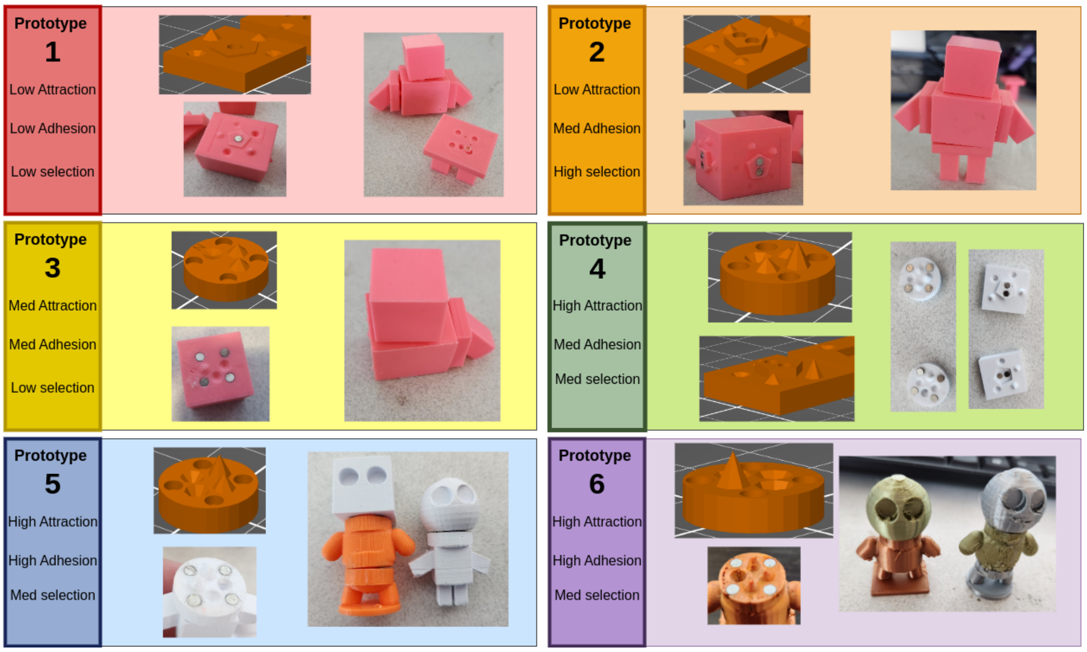

Figure 2: Progression of prototypes over time.

assembled into their final form. Each fragment contains one or more *bases*, which is what the makes the fragments attracted to other fragments. Bases have *cues*, that determine which other bases it can connect to. Cues are either indentations, called *holes*, or outward faces cones, called *pegs*. Each base has a set number of positions of which cues are placed. The length and position of each cue, determine which other base(s) it *selects*. Bases have magnets glued into them that make them *attracted* to other bases. Once two bases are attracted to each other if they are compatible (they select each other) they become connected. The strength of this connection is called its *adhesion*. The goal of this project is to design bases that are to be placed on fragments, that have the following properties:

1. **High Selection**: Bases should only be able to connect with compatible bases, in the correct orientation. For any base (A), it should only be able to select it's perfect match (A') and not other base in the set.
2. **High Attraction**: Bases should be highly attracted to each other, such that when in a moving environment, it is likely that they end up close enough to become attracted to each other.
3. **High Adhesion**: Once bases are connected to each other, they should remain connected even when continuing to be shaken.

We wanted to make this project completely open source. Currently, we have the files related to this project on an Open Science Framework project [LINK]. This project has all models produced, both from prototypes and the final version, along with the printing file (either in ultimaker or gcode format). Additionally, we have a series of video demonstrating the properties of different prototypes. For the finalized version, we have included a complete guide to modeling, printing, and assembling the self-assembling fragments. Finally, we have included complete instructions for a example use case game: "Rattle, Roll, Rumble" about constructing robot armies to take into battle.

3.3. Prototypes

In this section we describe our iterative prototyping process. Developing self-assembling game pieces was an iterative process of trial and error. We originally started with trying to replicate designs from previous academic work, and continuously found methods to improve the attraction, adhesion, and selection of printed pieces. These values were accessed qualitatively. All prototypes are shown in

Figure 2.

Prototype 1: Replication

Our first attempt at creating bases was to replicate an approach we found in previous work [24]. In this project, bases have a center shape that either sticks out or is indented. The magnet for the base is placed in the center of this component, with north facing magnets being placed in bases that stick out and south facing magnets being placed in bases that are indented. The shape of this center component is orientation specific (e.g. a pentagon), such that bases are only attracted to each other if the orientation is correct. They also had different center shapes, but we started with implementing only the pentagon shape. On the outside of the bases, there are several positions for cues. Each cue could take one of 7 lengths out of $[-3mm, -2mm, -1mm, 0mm, 1mm, 2mm, 3mm]$. Negative positions refer to hole, $0mm$ is a flat surface, and positive positions are pegs.

We could not locate the exact models used in this paper, so we started from scratch to replicate this work. In order to make it easier to modify the work, we used the programming based CAD software, OpenSCAD [2]. This program generates a 3D model from a basic scripting language. This allowed us to create a simple base program, where one can easily modify the different parameters including: number of cues, the heights of each cue, and dimensions of the base. This first prototype consisted of 5 cues, with a center hole for one $3 \times 1mm$ magnet. To determine the heights of each base in the set, we wrote a simple program that, through trial and error, returns a set of heights such that each base can only be selected by its perfect match. Selection is determined by — for each position — if the hole is at least as deep as the length of the peg in the pair (as long as the pair have alternative polarity). This program randomly generates height sets and only adds them to final list if they are not selected by any other base in the set.

This program and the OpenSCAD file were used to create a set of 3 bases and their perfect matches (for 6 total bases). To test out the bases, a simple robot model was created on the TinkerCAD [3] software. This model was broken up into 5 fragments, one head, two arms, a torso, and one pair of legs. The bases were attached to the robot fragments such that the head can select the top of the torso, the torso can select an arm on either side of its body, and the legs can select the bottom of the torso. Note that in this model the arms are interchangeable and use the same base. We printed these fragments at our university Makerspace, on a Ultimaker S3.

This first prototype had many flaws. First, the indentation of the center shape was too shallow to be useful. Orientation was enforced, but through the cues and not the center shape. While this worked out in our sample set, we made no assumption that bases could be attracted to each other in different orientations in our height generation program. Therefore, it would be easy to accidentally create a set of bases where one base is attracted to a wrong base at an alterative orientation. Second, the sole $3 \times 1mm$ magnet was not nearly strong enough to attract fragments together. Even if you placed the fragments together, they would fall apart easily, leading to low adhesion as well. Third, we printed the legs with the base facing down which lead to fatal printing flaws on the base. From this we determined that bases should only be printed on their side or facing up. Lastly, the magnets were extremely difficult to place as the magnet holes were only marginally larger then the magnets themselves.

Prototype 2: Increasing Center Height

Our second prototype addressed many of the problems of the first prototype. We increased the height of the center component, added a second magnet to each base, and made the magnet holes larger. These changes, particularly the center height, had an substantial impact on selection and adhesion. Given the distance between the magnets and face of the fragments, bases were only attracted to each other if the fragments were in the exact correct orientation. While there was some evidence the fragments would self correct their position, this was fairly uncommon and manual correct was often needed. Additionally, the distances between magnets enforced the selection property of the cues. Bases could

[2]https://openscad.org/
[3]https://www.tinkercad.com/dashboard

only be connected if the pegs were fully enclosed in the holes. The fragments in this version also had much higher adhesion. Fragments that were gently placed on each other could sustain light shaking (compared to previously where even gravity would disconnect fragments). If the fragments are slightly pressed together, adhesion improved greatly and significant shaking was needed to separate fragments.

However, bases still had low attraction, despite the additional magnet. Since shaking demonstrated no attraction, we tested how attracted fragments were by gently dropping them on top of each other. Fragments could connect when dropped from a few millimeters away, and at least close to the right orientation. But any further distance, or greater orientation differences, and the fragments demonstrated no attraction.

Prototype 3: Removing Center

For the next prototype, we attempted to improve attraction by both removing the center component and placing 4 magnets on each base. Now, to enforce orientation, we used the polarity of the magnets. In the “north facing” base, only three of the magnets are north facing with one magnet being south facing (the “south facing” base is the inverse). With this set up, bases in the wrong orientation can only be attracted to 2 out of the 4 magnets.

This new design did have improvements on attraction. We could drop the fragments from a greater height and still get connections. However, attraction was still not great enough for any self-assembly to occur when shaken. Without the center shape, orientation was not strictly enforced, and fragments were attracted to each other even when the bases were wrong. Additionally, the center shape seemed to have a large effect on adhesion, as without them the pieces could only tolerate very light shaking.

Prototype 4: Larger Magnets

After several prototypes with limited success, we turned back to existing self-assembly work as inspiration. Instead of looking through academic papers, we search through open source 3D modelling websites, such as Thingiverse. Available models on these websites had two different forms, the first being a 12 sided polygon to imitate a “virus” [4] and a seaweed like figure [5]. Both of these use copies of the same model and have no selection criteria. However, we decided to print the virus model as inspiration for this project. This virus model guided us in two ways: first it reminded us of a dice which lead to us creating a 6 sided dice model for our example game (see Figure 1), and second it used $3 \times 1.5mm$ magnets instead of $3 \times 1mm$. While this initially didn’t seem like it would make a big difference, it ended up having a substantial impact on attraction.

For the next prototypes, we decided to reduce printing time by just printing the bases and not the entire robot model. We printed both bases with and without center components, this time with the larger magnets. While the larger magnet did increase attraction for the base with the center component, it was still insufficient to successfully self-assemble. The base without a center component, while having lower selection and adhesion, was substantially more attractive. It was at this point of the process where we realized that attraction was more important than selection or adhesion for this project. If we wanted fragments that would quickly self-assemble when being manually shaken, we would want them to be as attractive as possible. While selection is still important, with the small number of bases we are testing on, it was not as critical.

Prototype 5: Simple Robot

Once we narrowed down on the base configuration, we continued to test them on the robot fragments. To make the process easier we started with a two fragment robot: one head fragment and one combination torso and leg fragment. We also constructed two forms for the robots, an angular body (called “Kiki”) and a smooth body (called “Boba”), inspired by a famous experiment [25]. These two forms used the same bases, so a Kiki head could be attached to a Boba body and vice versa. This simpler figure, along with highly attractive bases, were substantially more successful. We were able to get the

[4] https://www.thingiverse.com/thing:2834219
[5] https://www.thingiverse.com/thing:5353951

fragments to self assemble when shaken, although it could sometimes take a couple of minutes for the fragments to assemble.

With the success of the two part robot, we constructed a three part robot: one head fragment, one torso fragment, one leg fragment. Like before, we made these pieces for both the Boba and Kiki models. The head and torso attach with same base (A and A') as the two-part robots, such that same heads are compatible between the two and three part robots. The torso and legs attach with a different base (B and B').

It was the through the introduction of the second base pair that we noticed the issues with this new base configuration. While shaking the three parts robots we noticed the legs (with base B') becoming attached to the head (with base A) at certain orientations. The attraction between bases was so high that fragments could become connected both at the wrong orientation (where only two magnets can connect), and when a hole is $1mm$ too shallow for the peg.

Prototype 6: Minor Revisions

Since we realized that attraction was very important, instead of fixing selection by lowering attraction (such as introducing the center shape) we made the following adjustments:

1. Construct cue height sets such that they cannot be selected at incorrect orientations
2. Make the difference between cue heights at least $2mm$ instead of only $1mm$

While this has some effect on the number of bases that can be in a system, at the scale we were working with (only a few bases needed) this was not a problem. We modified our cue algorithm to check for orientation differences, and we changed our cue heights to $[-4m, -2mm, 0, 2m, 4mm]$.

With these changes we came to our final design. While they are certainly be improvements to make in the future, these fragments are highly attractive and adhesive and have adequate selection.

3.4. Self-Assembly Toolkit

Through finalizing our design, we can present a toolkit for creating custom self-assembling figures. The process follows a series of steps. All materials are available on OSF [6]:

1. (OPTIONAL) **Construct a cue height set**: Using our Python program, generate a list of cue heights for base pairs. This program works by randomly generating N lists of size M (our bases uses $M = 4$). For each list it generates, it checks if the cues could be selected by any other cue currently in the list, at any orientation. If there are not other bases that could select it, that list is added. These values are printed out for the user.
2. (OPTIONAL) **Construct base models**: Using our OpenSCAD program, copy and paste the cue lists from the python program for each base. This will generate a STL file containing both that base (on the left) and its perfect match (on the right). The left base will be the north facing base, and the right will be the south facing base.
3. **Create your 3D Model**: Using modelling software of choice (we used TinkerCAD), create a model for your 3D figure.
4. **Add bases to your model**: You can either use bases that were constructed in steps 1 and 2, or use the pre-built bases provided in the kit. Start by breaking your model into however many fragments you desire. At each joint where you want a base, create a hole that is the size of the base ($16mm \times 16mm \times 1mm$). Then place the base in the hole and merge the objects together. Do this for all fragments. Make sure that bases are pairs of each other where you want them to connected.
5. **Print the models**: If you do not have a 3D printer, check for local Makerspaces in libraries, universities, or community centers. Alternatively, there are print by demand services that will ship fragment to you. When printing, make sure you use a fine detail such as a $0.1mm$ layer height. Also assure that models are orientated such that the bases are facing up or at a 90% angle

[6]https://osf.io/8mk54/

(not facing the build plate). We used an Ultimaker S3 and a Prusa M4 using PLA filament, but other configurations should work.

6. **Glue magnets in**: Orient a base such that it is facing up. If this is a north facing base, glue (using super glue) one south facing $3mm \times 1.5mm$ magnet in the upper $left$ corner. Glue north facing magnets in the remaining four holes. For a south facing bases, glue one north facing magnet in the upper $right$ and south facing magnets everywhere else.
7. **Shake and play**: Place all fragments in a small plastic jar, and shake until fragments are self assembled. Reference the evaluation section for more detailed shaking information.

3.5. Proof of concept: Rattle, Roll, Rumble

To demonstrate the possibilities that emerge with self-assembling figures, we created an example game: "Rattle, Roll, Rumble." Full game instructions and materials are available on the OSF project.

This is a two player game, where each players controls a robot army. At the beginning of the game both players place a pre-determined amount of robot fragments in a jar and shake the jar for a set time. After this time, any robot that is fully assembled becomes a part of your army. Each part of the robot gives you different abilities. For example a "Kiki" torso gives you a short and long range attack, while a "Boba" torso give that robot a mid range attack and a healing ability. Robots also get bonuses based on the color of the fragment.

In addition to self-assembling the robot fragments, players also self-assemble the dice they will use in the game. Each side of the dice has a different color, associated with a different type of attack/defense. After assembling robots and dice, the players take turns taking over their robots and trying to defeat the other robot army. The authors tested Rattle, Roll, Rumble in a small playtest. This playtest demonstrated the fun potential of self-assembly, particularly the excitement of watching the figures assemble. However, there are still several flaws, including game design issues including poor balancing and a repetitive game-loop along with some physical problems, particularly that the dice came apart and had to be manually re-assembled. We recognized several opportunities for iteration of this game including using simulations to assist with balancing and incorporating self-assembly throughout the game (as self-assembly is currently just used in the game set-up).

4. Evaluation

We evaluate how well our fragments self-assemble through a series of tests preformed by the first author. These tests varied by temperature (i.e. intensity of shakes), duration, and number of copies. These tests confirm that this is a feasible design for self-assembly, although there is some concern of fatigue for the user.

4.1. Methodology

To test how well the figures self assemble, the first author performed 5 trials for each condition. The fragments were shaking in an transparent $3'' \times 3.5''$ jar. During each trial they measured how many full connections (two fragments coming together correctly), complete figures (all fragments of a figure become assembled), and partial or false connections. Partial connections are where two matching fragments connect in the wrong orientation, and false connections are partial connections between two fragments that do not match. False connections are weaker then partial connections, which are weaker then full connections.

Each trial varied by several conditions. The first condition was to test the different models we created:

1. Two part robot: robot model with a head and torso fragments
2. Three part robot: robot model with a head, torso, and leg fragments
3. D6 dice: six sided dice with 6 identical fragments

Next we varied the temperature (how intense the shakes are) between low and high temperature. We decided to use a loose interpretation for these terms, as this will eventually be given to an end user who will have to interpret written directions. In the low condition the first author shook the jar just enough for the fragments to move, but where the the fragments had little vertical movement. In the high condition the jar was shaken enough for relatively constant movement of the fragments in all directions. We also varied how long we shook the jar between 30 seconds to 90 seconds, and how many copies of each fragment were added (1, 2, or 3). Since the dice figures assembled faster, we only tested these up to 60 seconds.

Videos and outcomes of all trials are available on Open Science Framework (linked above).

4.2. Results

The results from the trial are shown in Table ??. Below we talk about the effect of the different conditions.

Time

The more time the jar is shaken, the more figures are able to form. Across all robot conditions, 6% of trials were able to form at least one figure in 30 seconds, 25% in 60 seconds, and 32% in 90 seconds. However, increasing the time too much may also have negative effects. In the 90 second trial, connections would often form and break before the time was up. Also the 90 seconds, especially with the high temperature, was tiring for the first author to complete.

Copies

Increasing the number of copies also increased the chances of complete figures forming. For trials with only 1 copy, 10% assembled complete figure compared to 22% for two copies, and 37% for three copies. There are some issues with increasing the number of copies. The additional fragments can interfere with the assembly process, either by breaking connections or creating partial and false connections. These partial and false connections most often break apart quickly, but prevent true connections from forming for those fragments. These problems are potentially mitigated by having a larger jar when introducing more copies.

Temperature

For the robot figures, the lower temperature was more successful. In the low temperature 31% of trials resulted in a completed figure, compared to 14% for high temperature. However, both high and low temperatures have benefits. In high temperatures, the bases are more likely to interact and partial or false connections are easier to break. However, bases that do interact are less likely to connect and connected bases are more likely to break. In contrast, in low temperatures there are less interactions between fragments, but fragments that do interact are more likely connect and stay connected.

Robot Models

Somewhat unsurprisingly, the 2-part robot had more trials that resulted in complete figures then the 3-part robot. For the 2-part robot, 32% of trials resulted in complete figures, compared to 13% for the 3-part robot. That being said, the 3-part robot had more connections, with an average of 0.85 connections per trial compared to 0.4 connections in the 2-part robot. This is slightly more then we would expect from the fact that the 3-part robot had twice as many bases that could connect.

Dice Models

The dice fragments behaved very differently from the robot models. The dice, having 8 dice on a smaller volume space, where highly attractive. This meant that complete figures were assembled at higher rates, however this also meant there was an increase of partial connections. This was such a problem that the low temperature was often unable to break away partial connections, leading to less complete figures. This was even more evident in the 3 copy condition, where the extra fragments distrusted the assembly process. Overall the dice consistently form at a higher rate, with all trials with

2-Part Robots					
Temp.	Time	Had Fig.	Full Conns.	Partial Conns.	False Conns.
One Copy					
low	30	1	1	0	0
low	60	1	1	0	0
low	90	3	3	0	0
high	30	0	0	0	0
high	60	1	1	0	0
high	90	0	0	0	0
Two Copies					
low	30	0	0	0	0
low	60	3	5	0	0
low	90	3	3	0	0
high	30	1	1	0	0
high	60	0	0	0	0
high	90	2	2	0	0
Three Copies					
low	30	1	1	0	0
low	60	4	5	0	0
low	90	4	6	0	0
high	30	1	1	0	0
high	60	3	4	0	0
high	90	1	2	0	0

3-Part Robots					
Temp.	Time	Had Fig.	Full Conns.	Partial Conns.	False Conns.
One Copy					
low	30	0	1	0	0
low	60	0	1	0	0
low	90	0	2	0	0
high	30	0	0	0	0
high	60	0	1	0	0
high	90	0	0	0	0
Two Copies					
low	30	0	3	1	0
low	60	2	8	0	0
low	90	2	10	0	1
high	30	0	0	0	0
high	60	0	0	0	0
high	90	0	1	0	0
Three Copies					
low	30	0	5	3	2
low	60	3	10	4	0
low	90	1	7	1	2
high	30	1	7	1	1
high	60	0	7	0	0
high	90	3	14	0	0

Dice					
Temp.	Time	Had Fig.	Full Conns.	Partial Conns.	False Conns.
One Copy					
low	30	0	12	4	0
low	60	2	28	0	0
high	30	2	26	0	0
high	60	4	24	0	0
Two Copies					
low	30	0	38	15	0
low	60	2	50	15	0
high	30	5	53	11	0
high	60	5	54	8	0
Three Copies					
low	30	0	8	59	0
low	60	3	57	19	0
high	30	1	72	20	0
high	60	5	80	10	0

Table 1
Experimental results. The jar was shaken 5 times per conditions. The Had Figure column is the number of trials were a full figure formed; full connections, partial connections, and false connections sum the total amount of each type of connection across trials.

high temperatures that ran for at least 60 seconds being able to assemble a complete figure. The high temperature produced a complete figure at 30 second when 2 copies were present.

Usability

While we did not test this formally, we did consider how usable these fragments are as actual game fragments. The robot fragments are certainly stable enough to be used, although breakages may occur. These breakages should be trivial to repair. The dice are able to be rolled without breaking, although they have to be treated delicately. Harsh throws, or tools like dice towers, would easily break apart the dice. These breakages might be harder to repair, as it requires the player to remember where each side goes.

5. Discussion

5.1. Value of Toolkit

In this work we present a toolkit for making self-assembling figures for tabletop gaming. This toolkit is more accessible then previous work in this area. Designers who want to use this system would need to have some technical knowledge, but don't need to have extensive CAD or programming language experience. The assembly process is fairly easy, although it does require some fine motor control and understanding of magnetism. Further, the materials to make a self-assembling system are relatively inexpensive. Designers do need access to a 3D printer, but these are becoming increasingly accessible in home and communities environments. We used an inexpensive variety of 3D printing filament, along with inexpensive magnets and super glue. Overall this toolkit is approachable to the hobbyist population.

The results of our evaluation show that the figures are able to self-assemble. However, we have not been able to find a condition that consistently results in fully formed figures, especially for the 3-part robot. This could indicate that improvements need to be made to the system. For example, we could consider having a single large magnet in the center of the base. This could improve adhesion, along with reducing the possibility for partial connections at one of the magnets at the edge. We could also continue to experiment to find the ideal way for the fragments to self assemble. For example we haven't yet looked at the way the jar was shaken, nor the size of the jar. It is possible, if not likely, that shaking for self-assembly is a skill that can be improved overtime. While conducting the experiments, the first author did notice themselves "trying" to get the fragments to self assemble. Moving the jar such that the fragments that needed to connect are closer together, or subtle slowing the shaking motion to avoid connections breaking apart.This level of skill could be undesirable, if the designer wants the outcome to be completely random, in which case, an opaque jar may help reduce the level of skill. However, the introduction of shaking skill could be considered an additional game mechanic to learn.

5.2. Possibility Space

We have demonstrated two uses for PCG-SAF: randomly constructed miniatures and dice. In our example game, we determined mechanics that could be associated with each fragment of the robot. This means that every self-assembled figure will not only look different, but have different effects on gameplay. We had a simple formula for how figures should form, with each having any head, body, or torso. However, designers could choose to have finer control over this process. For example, a torso could accept only certain legs or heads. This can be used to ensure that incompatible mechanics don't occur. Future designers could also consider figures without set number of pieces in them. For example, some arms could attach to both the torso and a weapon, while other arms only attach to the torso. Alternatively, fragments could allow creating structures that range widely in size like the "seaweed" mentioned above. We noticed several times where fragments could partially connect. This could either be seen as undesirable, or could be an opportunity to introduce more game mechanics (e.g. special mechanics for a "two-headed" robot). The self-assembled dice bring further possibility for not only procedurally generating content, but random generators as well. Board games have long used customized dice, but this allows for dice to both be specific to a player and be randomly assigned. Future designers could consider different dice sizes, or even experimenting with the physical properties of the dice such as weighting the sides differently.

However, these use cases are only a narrow range of what could be possible with a PCG-SAF system. Along with figures and dice, designers could consider using self-assembly to construct game boards, maps, or dungeons. Mechanics that have you drawing from a deck, could be replaced with self-assembling a figure and interpreting the results. PCG-SAF has a lot of parallels to computational methods for PCG. Self-assembly could be considered as a type of grammar, which has been used for PCG previously [26, 27, 28]. One could consider how to translate past work on grammars for PCG into a PCG-SAF system. Further, PCG-SAF focuses on local interaction, which is similar in concept to PCG algorithms such as Wave Function Collapse [29]. It seems plausible that a Wave Function Collapse like algorithm could be implemented using PCG-SAF; possibly in 3D with voxel-like fragments that assemble into buildings or dungeons. Self-assembly begins to bridge the gap between computational PCG and completely analog PCG. Further, the range of what all is possible with a PCG-SAF system is not yet known. By presenting this concept to computer scientists and game designers, we are likely to discover many existing new opportunities for how to use PCG-SAF.

5.3. Limitations

As preliminary work, there are several limitations to this project. We only tested our bases with a simple figure, including only two different base sets in total. We also introduced rules for a game that could use these figures, but only conducted a small internal playtest of the game. Further, our evaluation showed that we were not able to consistently produce full-assembled figures. This is likely to improve through re-design of the bases, but it may point out some of the inherit limitations with this process. There is likely a low ceiling for how complex a system can became before it is infeasible in this context, although we don't yet know where this ceiling is.

There are also several ethical considerations to this work. The first is that the process of shaking can cause irritation or fatigue for players. This risk increases the more complex a system is. There are also some minor dangers to assembling the pieces. Using super glue, especially in such small spaces, can cause damages. We have frequently gotten super glue on our hand, which caused slight pain and discomfort but no serious injuries. Lastly, we want to acknowledge the plastic waste that was produced in working on this project. While we tried to only print the pieces that we needed, we did produce plastic waste through failed prototypes, print errors, and support material. The impact of this could be mitigated through the choice of filament. We used PLA, which is both the easiest to work with and recyclable and biodegradable [30, 31]. Additionally, PLA doesn't emit toxins when printing like material such as ABS. However, future work should consider how to reduce such waste and whether the artifacts produced are worth the environmental cost of producing them.

5.4. Future Work

Our evaluation demonstrated that there is still progress to made on the design of the bases. Particularly it seems there needs to improvements on attraction and adhesion. Along with containing the design process, future work should consider testing on larger and more complex systems. Understanding the limits on what can be accomplished is important for considering the design of game mechanics. We are interested in implementing a simulator to allow more rapid iteration on designs without the use of additional plastic, as well as exploring different materials.

Future work should also consider how to translate computational PCG methods into PCG-SAF systems. There seems to be several analogies between computational methods, such as grammars or Wave Function Collapse, that could be translated into an analog system. Further, game designers could be consulted to consider how these systems can be utilized. While we introduced our initial ideas of how to use PCG-SAF systems, board game designers would have a wider idea of what could be considered. Designers should also be consulted to determine what would be required of these system to be successfully integrated into tabletop games.

6. Conclusion

Despite the many advances in digital entertainment, tabletop games remain a popular medium. Researchers and designers should continue to investigate how to expand experiences that do not require digital components. In this work, we presented a method of procedurally generating content for tabletop games using self-assembling figures. This has the possibility to expand what is possible in tabletop games. While improvements can still be made, we demonstrate the it is feasible to procedurally construct figures and dice using self-assembly. To promote the development of this work, we provide a toolkit that allows researcher, designers, and hobbyists to explore this new medium for procedural content generation.

Declaration on Generative AI

During the preparation of this work, the author(s) used Writefull in order to: Grammar and spelling check. After using this tool/service, the author(s) reviewed and edited the content as needed and take(s) full responsibility for the publication's content.

References

[1] D. P. Darzentas, M. A. Brown, M. Flintham, S. Benford, The Data Driven Lives of Wargaming Miniatures, in: Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, ACM, Seoul Republic of Korea, 2015, pp. 2427–2436. URL: https://dl.acm.org/doi/10.1145/2702123.2702377. doi:10.1145/2702123.2702377.

[2] L. Albaugh, A. Grow, C. Liu, J. McCann, G. Smith, J. Mankoff, Threadsteading: Playful interaction for textile fabrication devices, in: Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, 2016, pp. 285–288.

[3] C. Pearce, G. Smith, J. Choi, I. Carlsson, ebee: Merging quilting, electronics & board game design, in: Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, 2016, pp. 3877–3880.

[4] G. M. Whitesides, B. Grzybowski, Self-Assembly at All Scales, Science 295 (2002) 2418–2421. URL: https://www.science.org/doi/full/10.1126/science.1070821. doi:10.1126/science.1070821, publisher: American Association for the Advancement of Science.

[5] R. Bogue, Self-assembly: a review of recent developments, Assembly Automation 28 (2008) 211–215.

[6] M. J. Rogerson, M. Gibbs, W. Smith, "I Love All the Bits": The Materiality of Boardgames, in: Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems, ACM, San Jose California USA, 2016, pp. 3956–3969. URL: https://dl.acm.org/doi/10.1145/2858036.2858433. doi:10.1145/2858036.2858433.

[7] A. Al Mahmud, O. Mubin, S. Shahid, J.-B. Martens, Designing and evaluating the tabletop game experience for senior citizens, in: Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges, NordiCHI '08, Association for Computing Machinery, New York, NY, USA, 2008, pp. 403–406. URL: https://dl.acm.org/doi/10.1145/1463160.1463205. doi:10.1145/1463160.1463205.

[8] S. Bhaduri, J. G. Ortiz Tovar, S. K. Kane, Fabrication Games: Using 3D Printers to Explore New Interactions for Tabletop Games, in: Proceedings of the 2017 ACM SIGCHI Conference on Creativity and Cognition, C&C '17, Association for Computing Machinery, New York, NY, USA, 2017, pp. 51–62. URL: https://dl.acm.org/doi/10.1145/3059454.3059463. doi:10.1145/3059454.3059463.

[9] E. Stemasov, T. Wagner, A. Askari, J. Janek, O. Rajabi, A. Schikorr, J. Frommel, J. Gugenheimer, E. Rukzio, DungeonMaker: Embedding Tangible Creation and Destruction in Hybrid Board Games through Personal Fabrication Technology, in: Proceedings of the CHI Conference on Human Factors in Computing Systems, ACM, Honolulu HI USA, 2024, pp. 1–20. URL: https://dl.acm.org/doi/10.1145/3613904.3642243. doi:10.1145/3613904.3642243.

[10] F. Shyne, S. Cooper, Computational tools for table-top role-playing games: A scoping review, in: Proceedings of the 20th International Conference on the Foundations of Digital Games, 2025, pp. 1–14.

[11] G. Smith, An Analog History of Procedural Content Generation, in: Foundations of Digital Games, 2015.

[12] M. Guzdial, D. Acharya, M. Kreminski, M. Cook, M. Eladhari, A. Liapis, A. Sullivan, Tabletop Roleplaying Games as Procedural Content Generators, in: International Conference on the Foundations of Digital Games, ACM, Bugibba Malta, 2020, pp. 1–9. URL: https://dl.acm.org/doi/10.1145/3402942.3409605. doi:10.1145/3402942.3409605.

[13] J. A. Brown, M. Scirea, Procedural Generation for Tabletop Games: User Driven Approaches with Restrictions on Computational Resources, in: P. Ciancarini, M. Mazzara, A. Messina, A. Sillitti, G. Succi (Eds.), Proceedings of 6th International Conference in Software Engi-

neering for Defence Applications, Springer International Publishing, Cham, 2020, pp. 44–54. doi:10.1007/978-3-030-14687-0_5.
[14] J. Crawford, J. Wyatt, R. J. Schwalb, B. R. Cordell, Player's handbook, Wizards of the Coast LLC, 2014.
[15] K. Teuber, The settlers of Catan, board game, 1995.
[16] K.-J. Wrede, Carcassonne, board game, 2000.
[17] Hasbro, Heroscape, board game, 2004.
[18] I. Soute, T. Vacaretu, J. D. Wit, P. Markopoulos, Design and Evaluation of RaPIDO, A Platform for Rapid Prototyping of Interactive Outdoor Games, ACM Trans. Comput.-Hum. Interact. 24 (2017) 28:1–28:30. URL: https://dl.acm.org/doi/10.1145/3105704. doi:10.1145/3105704.
[19] A. Winslow, Staged self-assembly and polyomino context-free grammars, Natural Computing 14 (2015) 293–302. URL: https://doi.org/10.1007/s11047-014-9423-z. doi:10.1007/s11047-014-9423-z.
[20] E. Klavins, Programmable Self-Assembly, IEEE Control Systems Magazine 27 (2007) 43–56. URL: https://ieeexplore.ieee.org/document/4272328/. doi:10.1109/MCS.2007.384126.
[21] U. Majumder, J. H. Reif, A Framework for Designing Novel Magnetic Tiles Capable of Complex Self-assemblies, in: D. Hutchison, T. Kanade, J. Kittler, J. M. Kleinberg, F. Mattern, J. C. Mitchell, M. Naor, O. Nierstrasz, C. Pandu Rangan, B. Steffen, M. Sudan, D. Terzopoulos, D. Tygar, M. Y. Vardi, G. Weikum, C. S. Calude, J. F. Costa, R. Freund, M. Oswald, G. Rozenberg (Eds.), Unconventional Computing, volume 5204, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 129–145. URL: https://link.springer.com/10.1007/978-3-540-85194-3_12. doi:10.1007/978-3-540-85194-3_12, series Title: Lecture Notes in Computer Science.
[22] M. Jílek, L. Bertl, M. Kulich, L. Přeučil, Centimeter-Scaled Self-assembly of a Chessboard Pattern, in: O. Gusikhin, K. Madani, J. Zaytoon (Eds.), Informatics in Control, Automation and Robotics, Springer International Publishing, Cham, 2022, pp. 410–431. doi:10.1007/978-3-030-92442-3_21.
[23] M. Nisser, Y. Makaram, F. Faruqi, R. Suzuki, S. Mueller, Selective Self-Assembly using Re-Programmable Magnetic Pixels, in: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022, pp. 12659–12666. URL: https://ieeexplore.ieee.org/document/9981879/?arnumber=9981879. doi:10.1109/IROS47612.2022.9981879, iSSN: 2153-0866.
[24] A. Hacohen, I. Hanniel, Y. Nikulshin, S. Wolfus, A. Abu-Horowitz, I. Bachelet, Meshing complex macro-scale objects into self-assembling bricks, Scientific Reports 5 (2015) 12257. URL: https://www.nature.com/articles/srep12257. doi:10.1038/srep12257.
[25] V. S. Ramachandran, E. M. Hubbard, Synaesthesia–a window into perception, thought and language, Journal of consciousness studies 8 (2001) 3–34.
[26] C. Jemmali, C. Ithier, S. Cooper, M. S. El-Nasr, Grammar based modular level generator for a programming puzzle game, in: Proceedings of the Experimental AI in Games Workshop, 2020, p. 7.
[27] G. S. Tows, Imagegram: image grammar for procedural generation, 2009. URL: https://zaratustra.itch.io/imagegram.
[28] J. M. Font, R. Izquierdo, D. Manrique, J. Togelius, Constrained level generation through grammar-based evolutionary algorithms, in: G. Squillero, P. Burelli (Eds.), Applications of Evolutionary Computation, Lecture Notes in Computer Science, Springer International Publishing, Cham, 2016, pp. 558–573.
[29] M. Gumin, Wavefunctioncollapse, https://github.com/mxgmn/WaveFunctionCollapse, 2016.
[30] N.-A. A. B. Taib, M. R. Rahman, D. Huda, K. K. Kuok, S. Hamdan, M. K. B. Bakri, M. R. M. B. Julaihi, A. Khan, A review on poly lactic acid (pla) as a biodegradable polymer, Polymer Bulletin 80 (2023) 1179–1213.
[31] M. R. Hasan, I. J. Davies, A. Pramanik, M. John, W. K. Biswas, Potential of recycled pla in 3d printing: A review, Sustainable manufacturing and service economics 3 (2024) 100020.