Language and Tool Support for 3D Crochet Patterns:

Virtual Crochet with a Graph Structure

Klara Seitz, Jens Lincke, Patrick Rein, Robert Hirschfeld

Technische Berichte Nr. 137

des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam

Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam

Technische Berichte des Hasso-Plattner-Instituts für
Digital Engineering an der Universität Potsdam | 137

Klara Seitz | Jens Lincke | Patrick Rein | Robert Hirschfeld

Language and Tool Support for 3D Crochet Patterns

Virtual Crochet with a Graph Structure

Universitätsverlag Potsdam

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der
Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind
im Internet über http://dnb.dnb.de/ abrufbar.

Universitätsverlag Potsdam 2021
http://verlag.ub.uni-potsdam.de/

Am Neuen Palais 10, 14469 Potsdam
Tel.: +49 (0)331 977 2533 / Fax: 2292
E-Mail: verlag@uni-potsdam.de

Die Schriftenreihe **Technische Berichte des Hasso-Plattner-Instituts für Digital Engineering an der Universität Potsdam** wird herausgegeben von den Professoren des Hasso-Plattner-Instituts für Digital Engineering an der Universität Potsdam.

ISSN (print) 1613-5652
ISSN (online) 2191-1665

Online veröffentlicht auf dem Publikationsserver der Universität Potsdam
https://doi.org/10.25932/publishup-49253
https://nbn-resolving.org/urn:nbn:de:kobv:517-opus4-492530

Zugleich gedruckt erschienen im Universitätsverlag Potsdam:
ISBN 978-3-86956-505-7

Abstract

Crochet is a popular handcraft all over the world. While other techniques such as knitting or weaving have received technical support over the years through machines, crochet is still a purely manual craft. Not just the act of crochet itself is manual but also the process of creating instructions for new crochet patterns, which is barely supported by domain specific digital solutions. This leads to unstructured and often also ambiguous and erroneous pattern instructions. In this report, we propose a concept to digitally represent crochet patterns. This format incorporates crochet techniques which allows domain specific support for crochet pattern designers during the pattern creation and instruction writing process. As contributions, we present a thorough domain analysis, the concept of a graph structure used as domain specific language to specify crochet patterns and a prototype of a projectional editor using the graph as representation format of patterns and a diagramming system to visualize them in 2D and 3D. By analyzing the domain, we learned about crochet techniques and pain points of designers in their pattern creation workflow. These insights are the basis on which we defined the pattern representation. In order to evaluate our concept, we built a prototype by which the feasibility of the concept is shown and we tested the software with professional crochet designers who approved of the concept.

Zusammenfassung

Häkeln ist eine weltweit verbreitete Handarbeitskunst. Obwohl andere Techniken, wie Stricken und Weben über die Zeit maschinelle Unterstützung erhalten haben, ist Häkeln noch heute ein komplett manueller Vorgang. Nicht nur das Häkeln an sich, sondern auch der Prozess zur Anleitungserstellung von neuen Häkeldesigns ist kaum unterstützt mit digitalen Lösungen. In dieser Arbeit stellen wir ein Konzept vor, das Häkelanleitungen digital repräsentiert. Das entwickelte Format integriert Häkeltechniken, wodurch wir den Prozess des Anleitungschreibens für Designer spezifisch für die Häkeldomäne unterstützen können. Als Beiträge analysieren wir umfassend die Häkeldomäne, entwickeln ein Konzept zur Repräsentation von Häkelanleitungen basierend auf einer Graphenstruktur als domänenspezifische Sprache und implementieren einen projektionalen Editor, der auf der besagten Graphenstruktur aufbaut und weiterhin die erstellten Anleitungen als schematische Darstellung in 2D und 3D visualisiert. Durch die Analyse der Domäne lernen wir Häkeltechniken und Schwachstellen beim Ablauf des Anleitungserstellens kennen. Basierend auf diesen Erkenntnissen entwickeln wir das digitale Format, um Anleitungen zu repräsentieren. Für die Evaluierung unseres Konzepts, haben wir einen Prototypen implementiert, der die Machbarkeit demonstriert. Zudem haben wir die Software von professionellen Häkeldesignern testen lassen, die unsere Herangehensweise gutheißen.

Contents

1 Introduction

According to *myboshi*, one of Germany's most well-known websites offering patterns and wool for crochet and knitting, over 45 thousand patterns were purchased and about 700 finished crocheted hats were ordered on their site in 2019[1]. This demonstrates the existing demand for patterns for self reproduction, which is even higher than that of finished items, which may originate from the fact that handmade products are expensive and patterns rather cheap. All over the world exist such marketplaces to share and sell patterns or the finished handmade items. But all over the world, patterns are shared in non-uniform formats and varying qualities.

While techniques like knitting and weaving are not a purely handmade craft anymore, crochet still is. Crochet relies on designers creating patterns with clear and correct instructions which are then sold to customers for self reproduction. Designers not only compete for their creative ideas but also for their quality of patterns. A customer who pays for a pattern wants to be sure that the instructions are clear and the result will be as expected. But due to the fact that crochet is a manual process there are not many programs which facilitate the workflow of designers which is mostly still pen and paper based.

This work proposes a concept which enables crochet designers to create patterns using a software system which understands the domain of crochet. Thus, each pattern is based on a crochet specific digital representation which, on the one hand, allows automatic visualization based on connections and types of stitches and, on the other hand, export to any other format such as human readable instructions in various languages or machine code for a future crochet machine. Thereby the outreach of a single pattern can be theoretically enlarged to any crocheter in the world and the chance of mistakes in instructions is reduced to a minimum.

1.1 Current Workflow and Representation of Crochet

Crochet is the manual process of creating fabric with a crochet hook and a single thread. A crocheter uses the hook to make specific knot patterns with the yarn. Each type of knot is called a stitch. Stitches can be connected to nearly any place on the crochet fabric which allows the creation of flat and also three-dimensional arbitrary shapes. Unlike knitting, which uses two or more needles and a single thread, crochet is a purely handmade craft. Currently no machines exist to industrially produce crocheted fabric.

[1]Courtesy of Thomas Jaenisch, executive director of myboshi, data received on 5-25-20.

This work is based on the aforementioned crochet charts. Such charts show the orientation and position of stitches of flat patterns. Designers mostly draw them by hand, or use a program which can supply them with stitch fonts and guidelines to place the symbols representing stitches on a 2D canvas. The resulting stitch chart image can be used directly as a template for reproduction. But if the pattern is 3D or the chart alone is not enough to reproduce the design, the designer has to fall back to using written or video recorded pattern instructions.

1.2 Existing Pattern Instruction Formats

Crocheting is an old handcraft which follows clear rules. Yet, no exact, formal way of writing instructions for a pattern has been defined. Therefore, crocheters face an unnecessary barrier when crocheting, learning to crochet or designing a new pattern. Similarly, the designers can not simply focus on designing creative, new patterns; instead they face the pressure to create clear instructions with the least amount of mistakes.

Crocheters typically follow a set of instructions in order to produce crocheted items. The designers coming up with new patterns usually have to go stitch by stitch on a trial-and-error basis to find out how to form the desired shape. Crocheting is a very ordered process: Stitches are placed one by one and if there was a mistake at an earlier point everything up until that point has to be undone to be able to fix or change that part. Finished designs can either be sold as the resulting crocheted product or as instructions for reproduction. Crochet patterns exist in various forms as shown in Figure 1.1. Such instructions can be how-to videos or more often texts with diagrams called 'crochet chart' or pictures of the actual crochet process. The pattern instructions can then be used by other crocheters to reproduce the item. Instruction videos (Figure 1.1 (a)) are mostly very repetitive and everyone follows a different structure. Textual instructions (Figure 1.1 (b)) often lack accurate explanations as every designer comes up with their own notation. Many instruction texts include accidental errors such as skipping steps or incorrect sums of stitch number, all of which can lead to confusion when trying to reproduce the pattern. The previously mentioned crochet charts can be used to diagrammatically depict flat patterns. It makes use of worldwide known, standardized symbols to represent stitches. This eliminates the issue of instruction language, as it exists in written and video instructions, and thus makes the same pattern directly understandable by any crocheter across the world. The clarity of such a chart depends on how accurately it is drawn. An example chart can be seen in Figure 1.1 (c). Each symbol represents a specific stitch type and the position and rotation of each symbol is relevant to indicate the order of the stitches and to show which stitches are connected to each other.

The way that designers currently build such charts is by far not as exact as one might expect. Many designers draw the charts by hand and incorporate a photo of it into their pattern instructions. If any programs are used, they are mostly generic drawing programs or text editors.

There exist also some crochet specific programs which try to tackle the issue of positioning the stitch symbols. They support the designers by supplying the stitch symbols and snapping guidelines to facilitate placement. Yet, the stitches are never aware of any connections between them and thus are rather a more specialized drawing program. Find more detailed descriptions and examples of such programs in Chapter 6.

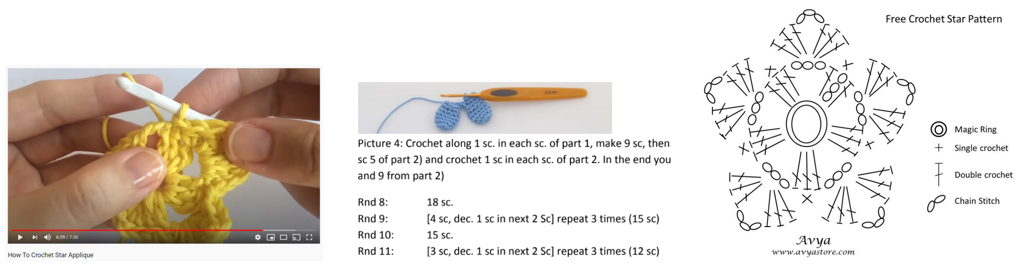
A big advantage of the technique of crochet is its ability to create any 3D shapes in a single piece. The previously mentioned crochet charts, however, are only able to represent 2D patterns as they are always drawn on a flat canvas. Thus, all the 3D patterns are currently explained in manually written instructions and progress pictures or shown by videos.

All in all, there is no standard way of describing crochet patterns, neither for 2D nor 3D patterns. Internationally known visual representations for 2D patterns exist but there is no program or method to draw such diagrams unambiguously. Furthermore, designers write and structure textual instructions very differently from each other and use their own abbreviations which also does not yield a unified pattern instruction style. Through the limited digital support and unstructured writing process, crochet patterns are generally very error prone.

1.3 Lack of Crochet Machines

At the moment no machines exist which are able to industrially produce fabric based on the technique of crochet. Currently, any industrially created fabric is made by weaving or knitting machines. Crocheted clothing, figures or other items are handmade.
While there is currently no process available to automate crochet to create entire projects, there have been attempts to imitate crochet or automate specific stitch types.

For an art project [25] a machine has been created which is able to make a single line of chain stitches. Also, there are so-called 'crochet machines' [17, 24] which are able to knit in a way which imitates the texture of crochet. Similarly, there are crochet gallon machines[46] which, despite the name, are also use a knitting method called



Picture 4: Crochet along 1 sc. in each sc. of part 1, make 9 sc, then sc 5 of part 2) and crochet 1 sc in each sc. of part 2. In the end you and 9 from part 2)

Rnd 8:	18 sc.
Rnd 9:	[4 sc, dec. 1 sc in next 2 Sc] repeat 3 times (15 sc)
Rnd 10:	15 sc.
Rnd 11:	[3 sc, dec. 1 sc in next 2 Sc] repeat 3 times (12 sc)

(a) (b) (c)

Figure 1.1: Crochet Instruction Types: (a) Video of the Crochet Process (b) Written Instructions with the help of Images (c) Crochet Chart[2]

'warp knitting' which builds chains similar to crochet. Additionally, there a sewing machine exists [12] which is able to do a crochet border stitch which imitates the way it would look when crochet is used to work around the border of a fabric. Lastly, the most recent machine [1] is an actual crochet machine with limited capabilities. It can be used to crochet flat, rectangular fabric. The machine can produce chain stitches and single crochets. It cannot adapt the amount of stitches within a pattern which reduces the output to rectangles. Also, it allows only the most common insertion point to be used.

When compared to knitting machines, crocheting with that crochet machine is a very slow process. Handmade knitting and crocheting takes about the same time to create as each stitch has to be made one after the other. But when knitting with a machine the structure of knitted fabric can be turned into an advantage as whole rows can be knitted at once. Crochet machines cannot use the same principle, as crochet stitches require the previous stitch to be completed. Stitches in knitting only require the stitch of the previous row to be completed when starting a new one.

The main reason why there are no industrial crochet machines is rooted in the multitude of stitch insertion points. In crochet, stitches can be inserted in various points, for each stitch there are multiple places where the hook could be inserted. Additionally, any other spaces between stitches can also be used. Read more about stitch insertion points in Chapter 2. To build a machine, which supports any insertion point, either many place holders have to be positioned which might get in the way of each other, or the manual crocheting technique could be imitated. The latter would require for example a robotic arm which has the ability to 'see' or 'feel' the insertion points similar to a human. Grimmelsmann et al.[19] have analyzed the options in detail in order to build their crochet machine[1]. Current knitting machines, on the other hand, can take advantage of a single loop per knitting stitch where the insertion options are limited. Koch shows how knitting can be easily represented through basic code or even binary[26].

As currently there is no digital and formal way to describe crochet pattern instructions and the structure for a machine is very complex, crochet machines still do not exist.

1.4 Digitization of the Crochet Workflow

In this report, we propose to lay the foundation for a digitally supported crochet workflow from creating crochet patterns to the finished item. Figure 1.2 shows the current workflow from idea to final object. Designers start by trial-and-error shaping until the desired shape has been created. Then, rough charts are drawn or notes taken to keep track of the steps. Next, final instructions are written, drawn or recorded.

[2]Video source for pattern 'How to crochet star applique' : www.youtube.com, source of the pattern 'jip the owl': www.ravelry.com, source of the chart: www.avyastore.com.

Those instructions can be given to other crocheters who then reproduce the design by crocheting with hook and yarn.

In this report, we present a concept to build digital crochet representations for 2D and 3D patterns. Such representations can be created by using a visual editor which will replace the current step of manually drawing or editing crochet charts with generic text or photo software. Our approach allows the creation of 3D patterns additionally to the already existing known charts of flat patterns. The underlying structure of the pattern representation enables stitches to be connected in any way that they could in the real world. Thus, any crochetable 2D and 3D shape can be specified using our format. For visualization, an editor can present the pattern in a 3D environment using the already widely known crochet symbols. Such charts could then be uniformly exported to any text or visual format.

In the future, our pattern representation and editor can be used to further digitize the whole workflow. Taking advantage of the digital representation, patterns can be transformed into various formats or generated from other inputs such as 3D designs. In the Chapter 7, we present these ideas in more detail.

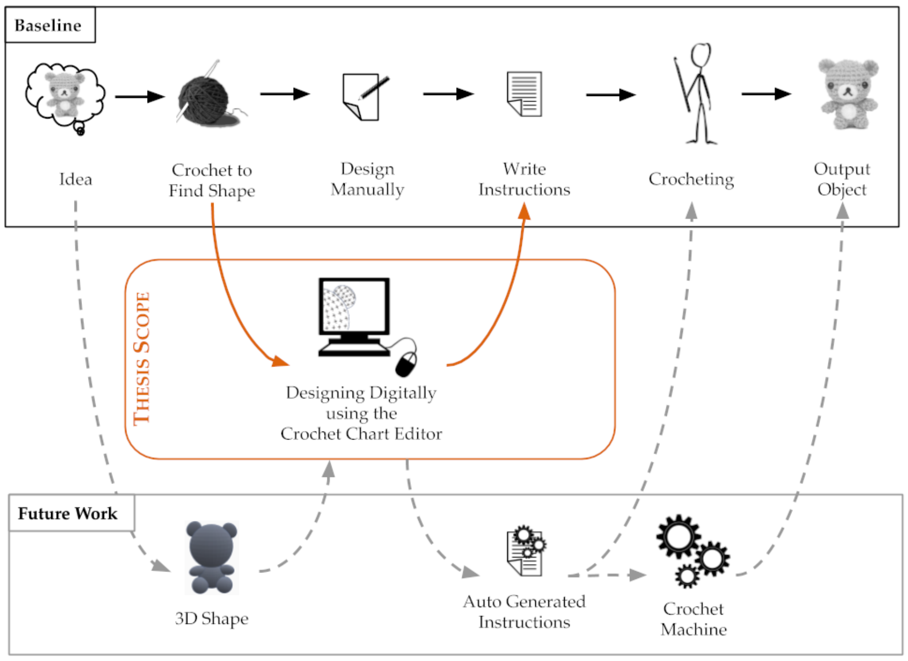


Figure 1.2: Overview of the current workflow as baseline and how our approach and further concepts can be applied

1.5 Representing Patterns Using a Graph Structure

In order to prepare the digitization of the crochet workflow, we built the basis by defining a machine readable representation of crochet patterns. Important characteristics of crochet are represented in a graph structure where a set of edges and a node represent stitches. We identified three types of stitches by grouping them by their amount of connections to other stitches and insertion points. Each stitch has at least one edge which connects to the previous stitch. By following these edges, we can traverse the whole graph in the order needed for reproduction. Using the graph structure, all crochet methods, as explained in Chapter 2, can be represented which allow building patterns of arbitrary shapes. In our prototype, this graph structure is used to visualize and layout the pattern in 2D and 3D environments.

1.6 Potential of Domain Specific Support

The main benefit of our approach is that crocheting gets domain specific digital support. This results not only in clear and unambiguous crochet charts but it also comes along with an exact, crochet specific internal representation of a pattern. Thus, the program has knowledge about the crochet domain and patterns can be analyzed and computations performed on them, such as validation or auto-completion. Additionally, patterns can be transformed into other helpful representations such as textual instructions in any language, visualizations of the output object as chart or simulated object with a crochet texture or even into machine code which could drive a future crochet machine to crochet an object from the pattern.

For professional crochet designers the proposed system improves the design process and simplifies the instruction generation. When designing a pattern digitally the slow trial-and-error shaping can be skipped and the desired shape can be reached without constantly undoing and redoing crocheted parts. Designers could focus solely on connecting stitches one after the other without thinking about placement, rotation and alignment which improves productivity. The shapes can be previewed in the 3D editor and proportions could be directly compared. Additionally, designers can focus more on creativity than precise explanations because the instructions could be generated from the digital pattern. Like this, designers can also reach a broader audience, as the instructions can be generated in any language or shared as internationally known charts. Computations on the patterns, such as auto-completion of patterns and editing functions, improve efficiency by utilizing fully the advantages of the digitization of the crochet pattern representation.

Crocheters who seek clear instructions for reproducing crochet patterns themselves can rely on the standardized patterns. The way we implemented our approach in a prototype, every pattern is always crochetable as the program only allows making valid stitch connections.

When taking into consideration the ideas for a future digital workflow, anyone, despite not knowing crochet, could also design simple crochet patterns by 3D modeling the item. The generated instructions could ultimately be sent to a future crochet machine or to a crocheter for production. This widens the reach of crochet to a broad audience.

All in all, the main advantages of such digital pattern representations and its 3D editor include that patterns are digitally represented by a machine readable format which allows automatic computations and transformations.

1.7 Contributions

In this report, we present three main contributions: crochet domain analysis, digital crochet pattern representation and a prototype of a 3D editor enabling the use and creation of patterns using said representation. Chapter 2 gives a thorough introduction to the techniques of crochet. Furthermore, we present a survey and two qualitative interviews with professional crochet designers which give insight into the current workflow from creating to publishing a new crochet design. Based on that knowledge, we were able to develop a comprehensive crochet pattern representation and build a prototype as tool for the designers. Our approach to digitally represent crochet patterns is presented in Chapter 3. We built a graph structure to be used as graphical domain specific language, visualizing the pattern as a crochet chart, in a projectional editor.

As a proof-of-concept, we built such an editor for the pattern representations. The process of creating a chart with this software resembles the way an actual item would be crocheted with a hook and yarn. Instead of inserting the hook into a stitch to start a new stitch, in the editor a node is clicked while the desired stitch type is selected. The program visualizes and layouts the graph structure using the standard crochet symbols in 2D and 3D environments and allows creation and basic editing of new crochet patterns. We show further implementation details in Chapter 4 and evaluated the approach and software in Chapter 5.

Our concept is the basis for any of the future steps to the digitization of the whole crochet workflow. The representation of the stitches and their connections within the graph allow calculations on the pattern. The graph can be visualized in a 2D and 3D environment which shows the feasibility of using known crochet charts for three-dimensional objects. The graph structure includes the directional information as needed for pattern instructions. The graph allows traversal along the stitches in the order of instructions. The structure could already be used to transform it into machine code for automatic execution or to generate a simple set of human readable instructions. Thus, our prototype shows the feasibility of the concept and the vision of the whole digital workflow.

1.8 Outline

In the following, we present the structure of this report and give a short summary of each chapter. This introduction chapter (Chapter 1) focuses on outlining the problem identified in the domain of crochet patterns and the contribution of this work. In Chapter 2 we give a more thorough background on the relevant knowledge about crochet. How we shaped the structure of the underlying crochet representation and concepts which can be built on top of it are presented in Chapter 3. The technology stack and other development specific solutions for our prototype are discussed in Chapter 4. The proof-of-concept prototype has been evaluated by professional crochet designers, compared to their current workflow and the range of representable patterns has been demonstrated in Chapter 5. Chapter 6 compares existing crochet languages and editors to our approach. Further ideas for the continuation of the project are laid out in Chapter 7.

2 Crochet: Analysis of Techniques and the Design Workflow

In Chapter 1 we gave an overview of the domain and main contributions of this report. Since crochet may not be known well to all of the readers, we start by giving a thorough introduction to the technique of crochet in this chapter. This background information is already part of the contribution as domain analysis and relevant as a basis for the other contributions. Here, we will present the method of crochet and list main issues of the current pattern creation workflow which we learned through a survey and interviews with professional crochet designers. These encountered issues are the main motivation and basis for developing our digitized, formal representation of crochet patterns. The approach will be discussed later on in Chapter 3.

2.1 The Technique of Crochet

Crochet is the manual process of creating fabric with a crochet hook and a single thread. A crocheter uses the hook to make specific knot patterns with the yarn. Each type of knot is called stitch. Stitches can be connected to nearly any place of the crochet fabric which allows the creation of flat and also three-dimensional shapes.

2.1.1 Crochet Tools

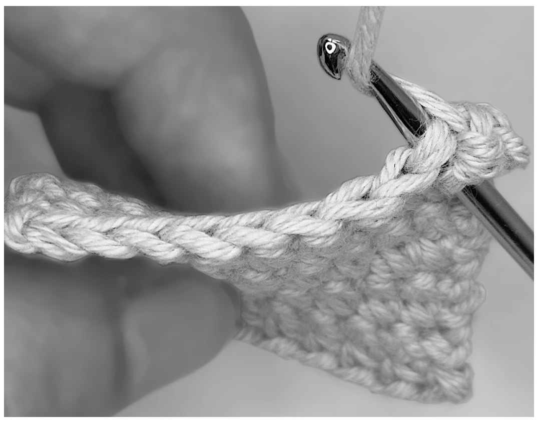


Figure 2.1: Crochet hook inserted in fabric, ready to start the next stitch

Crocheting uses a single hook and yarn as seen in Figure 2.1. The hook is required to pull yarn through tight spaces. The tip of the hook is usually tapered which helps to pass through small spaces in the crocheted fabric. Once the head of the tool has passed the fabric, the hook is used to get a hold of the thread and to pull it back through the fabric. This causes a loop of yarn to be wrapped around the hook. As long as a project has not been finished, there is always one loop around the hook. While stitches are being made, multiple loops can be on the hook.

Just as yarn can be of various thicknesses, there are also crochet hooks of different sizes. Yarn and hook should be chosen accordingly. Working with a big hook and thick yarn requires lewer stitches to reach a certain size of fabric compared to a small hook and thin yarn. When the size of a crochet pattern needs to be exact, such as when making clothing, typically the designer suggests to start with a swatch, a small crocheted piece of a fixed number of rows and stitches, to check the output size. The designer mentions the size that the swatch should have for the final piece to be of the correct size. An incorrect size of the swatch can indicate generally two things: either the hook is too small and therefore a bigger one is needed (or vice versa) or the crocheter influenced the size by applying a looser to tighter tension than others.

2.1.2 The Basis of Crochet: Stitches

A piece of crochet fabric consists of stitches. The way the stitches are arranged and which stitch types are used influence the shape. In crochet there are a lot of different types of stitches. Many basic stitches have been internationally defined[13]. But options to create new stitches or stitch compounds are unlimited. Stitches vary in height, their insertion point and connections to other stitches.

2.1.2.1 Texture Variation through Stitch Types

Table 2.1 shows a list of the most basic stitches, as defined internationally[13]. Each stitch creates a unique texture to the resulting fabric. We are listing their names in American and British English, for reference also in German, and show their corresponding symbols which represent them visually. In this report, we only reference stitches by their American name. *Chain stitches*, as shown in Figure 2.2 (a), are created by pulling yarn through the yarn loop on the hook. For a *Slip Stitch*, depicted in Figure 2.2 (b), the hook is inserted into the previously created fabric, then a loop of yarn is pulled through the fabric and also through the loop on the hook. Any of the *other stitches* are also started by inserting the hook into the previously created fabric and a loop of yarn is pulled through. Then, depending on the stitch type this process can be repeated multiple times to create more loops on the hook. To finish the stitch, the yarn is pulled through some or all of the loops on the hook until only one loop remains. In Figure 2.2 (c), this process is shown for the single crochet stitch.

2.1.2.2 Stitch Insertion Points or Places to Start a New Stitch

Generally, any place in between strands of yarn of the crocheted fabric where the crochet hook can pass through, are valid insertion points to start a new stitch. Over

Table 2.1: List of base stitches

American Name	**British Name**	**German Name**	**Symbol**
chain stitch	chain stitch	Luftmasche	⬭
slip stitch	slip stitch	Kettmasche	•
single crochet	double crochet	Feste Masche	+
half double crochet	half treble crochet	Halbes Stäbchen	⊤
double crochet	treble crochet	Stäbchen	Ŧ
treble crochet	double treble crochet	Doppelstäbchen	‡
double treble crochet	triple treble crochet	Dreifach Stäbchen	⧧

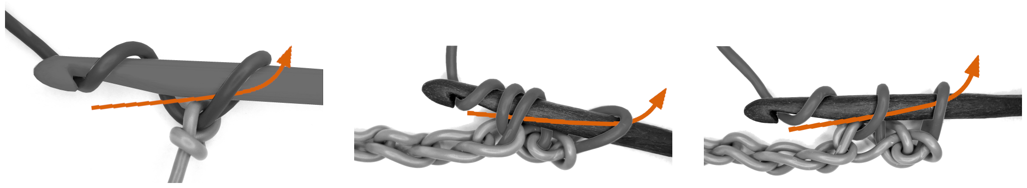

(a) Chain Stitch (b) Slip Stitch (c) Single Crochet

Figure 2.2: Yarn flow shown for three stitches as example

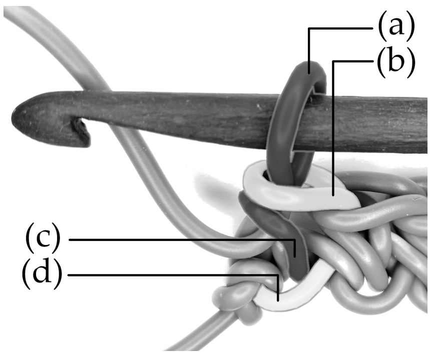


Figure 2.3: Structure of a stitch: (a) Active loop on hook (b) Top loop of the stitch (c) Stitch Post (d) Top loop of insertion stitch

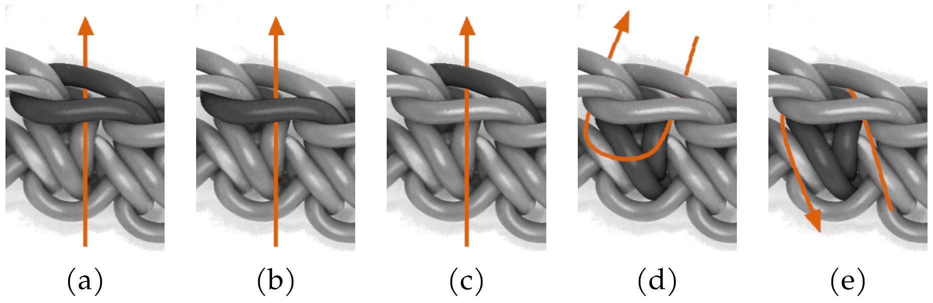


Figure 2.4: The five most common stitch insertion points: (a) Under the two top loops (b) Under the front loop (c) Under the back loop (d) Around the front of the post (e) Around the back of the post

time, specific points have been established as standard entry points. Figure 2.3 shows the structure of a single crochet as example. Figure 2.3 (a) marks the single loop on the hook which indicates the highlighted stitch has been completed and a new one can be started. Figure 2.3 (b) marks the top loop of the stitch. This loop is split in half by the loop which sits on the hook, Figure 2.3 (a). When not indicated otherwise, the hook is inserted under both parts of the top loop. Two other options which arise from this are to insert the hook only under the front half of the loop or only under the back half. These three options are shown in Figure 2.4 (a), (b) and (c).

The second part of the highlighted stitch is marked by Figure 2.3 (c) which is called *post* and Figure 2.3 (d) marks the loops of another stitch where the current one is inserted into. A stitch post can vary in height depending on the type of stitch. Here, the most commonly used insertion points are around the post. Either a stitch is called *Backpost* stitch which indicates the hook is to be passed from the back around the front of the post, as shown in Figure 2.4 (d). Or the stitch is called *Frontpost* stitch which indicates the hook should insert from the front of the fabric around the back of the post. The latter is shown in Figure 2.4 (e).

Different entry points are used by designers to create different textures and shapes. The aforementioned insertion points are the most common ways to insert into any stitch, but depending on the stitch type there can exist additional commonly used entry points.
Furthermore, some stitch combinations create free spaces between the stitches. Such holes are also frequently used to insert the hook into to start a new stitch. Most commonly, such holes are created through a chain of chain stitches in the middle of a pattern. Instead of inserting into each chain stitch separately, which can be quite fiddly, the hook can be inserted directly under the line of chain stitches.

2.1.3 Methods to Start and Shape the Fabric

In the following we will give an overview of different methods commonly used in crochet which influence the shape of the outcome.

2.1.3.1 Starting Techniques

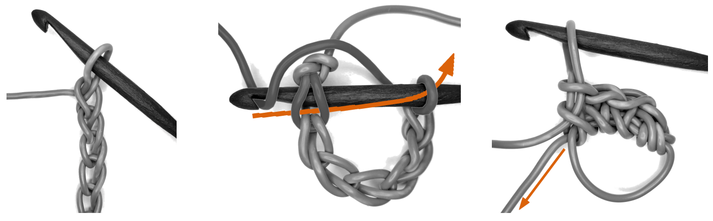

(a) Chain Line (b) Chain Circle (c) Magic Ring

Figure 2.5: Three starting techniques

A project can start off with either a ring or with a line of stitches. The most common three methods are shown in Figure 2.5. A line of stitches is mostly used to start something flat and rectangular. A ring can be used to create something flat and round or three-dimensional.

To begin with a line of stitches, several chain stitches are made after another until the desired length is reached. To continue, stitches are worked into the chain stitches. This starting technique is shown in Figure 2.5 (a).

From a line of chain stitches a slip stitch can be used to connect the ends of the line to form a ring, as shown in Figure 2.5 (b). This ring is called *Chain Circle* and is the first option to start a round project. Further stitches are worked either into the chain stitches or into the hole between them, at the center of the ring.

The other way of creating a ring is called *Magic Ring*, displayed in Figure 2.5 (c). Here, the initial stitches are worked around a loop of yarn. Once all desired stitches have been placed into the ring, the yarn can be pulled to reduce the loop size to a minimum. A ring of chain stitches cannot be adjusted in size once it has been closed which usually leaves a hole in the fabric, as shown in Figure 2.6 (a). With the magic ring, this hole can be reduced to be almost invisible, seen in Figure 2.6 (b).

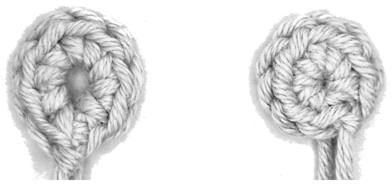
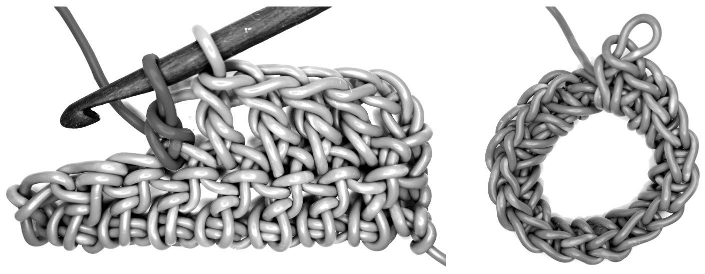

(a) (b)

Figure 2.6: Comparison of two starting methods: (a) The chain circle leaves a hole in the fabric (b) The magic ring allows closing the fabric tightly

2.1.3.2 Crocheting in Rows and Rounds



(a) Rows (b) Rounds

Figure 2.7: Examples of crocheting in rows (a) and rounds (b)

Depending on the desired shape and design, stitches can be crocheted in rows or in rounds, as shown in Figure 2.7. A pattern can switch from one to the other method at any time. The row method, shown in Figure 2.7 (a), works stitches one by one up until a certain point. There the fabric is turned and stitches are worked again one by one into the other direction. When working in rounds, shown in Figure 2.7 (b), the fabric does not have to get turned and stitches are worked continuously in the same direction.

The method of crocheting in rows requires making so called *Turning Chains* at the end of the row before turning the fabric and continuing in the opposite direction as before. Turning chains are made to raise the working height to the height of the following stitches to come. Figure 2.8 (a) shows the differences in height of six different stitches starting from the right with slip stitches (smallest) to double treble crochets (longest). Turning chains consist of a number of chain stitches. The amount is defined by the height of the stitch which the turning chains replace. Figure 2.8 (b) shows a line of four chain stitches, which is the equivalent amount of chains to reach the height of a double treble crochet.

As an example for turning chains, we might want to crochet two rows of ten single crochets each. The amount of chain stitches of the turning chain is one because the single crochet is about as high as one chain stitch. Therefore, at the end of the first row we crochet one chain stitch, before starting the second row.

For the method of crocheting in rounds there are two possibilities for transitioning between each round. One method is called *Spiral Rounds* or also *Continuous Rounds* which does not end the current round but rather directly continues crocheting in the same direction. The other method, called *Joined Rounds*, finishes each round with a slip stitch into the first stitch of the round. Then, similar to the turning chain at the beginning of a new row, the new round is started by the amount of chains necessary to reach the height of the next stitch. Instead of turning the work the following stitches are worked into the stitches of the previous round in the same direction as before.

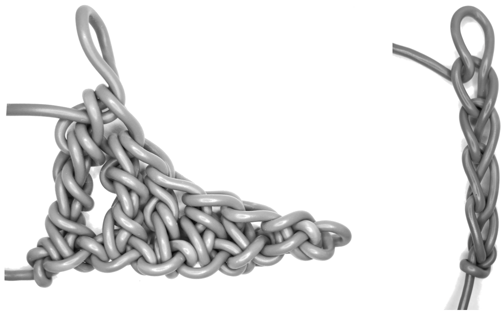

(a) (b)

Figure 2.8: Depending on the height of a stitch, a new row is started by levelling up to it by working an appropriate amount of chain stitches. (a) Six stitches of different heights, from left to right: double treble crochet, treble crochet, double crochet, half double crochet, single crochet, slip stitch (b) Four chain stitches which build up to the same height as the double treble crochet

2.1.3.3 Increasing and Decreasing Stitches To Adapt the Width

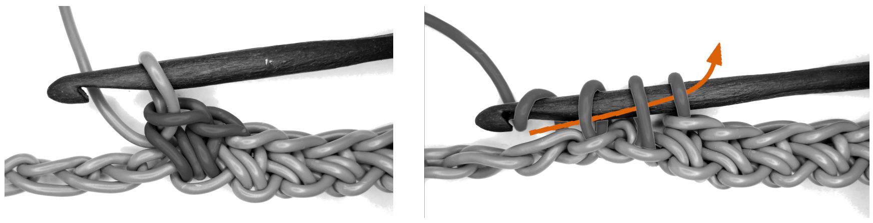

(a) Increase (b) Decrease

Figure 2.9: The width can be adapted through changing the stitch count of the row or round. (a) Working two or more stitches into the same insertion point is called increase (b) Connecting two or more insertion point to work only one stitch is called decreasing

The methods which are mainly responsible for reaching any shape are *Increase* and *Decrease*. Their use also leads to the vast possibilities of creating 3D shapes. When using the increase method, the amount of stitches in a row or round are rising and when decreasing, the amount is falling. The more stitches, the longer the row or wider the round. When using the increase method, shown in Figure 2.9 (a), multiple stitches are worked into the same space. As opposing operation, the decrease method, shown in Figure 2.9 (b), crochets several stitches as one which reduces the amount of stitches for the next round. Alternatively, some stitches can be skipped to reduce the number of stitches per row or round, as well. When decreasing, unfinished stitches are worked into subsequent spaces which lastly are finished as if they were a single stitch. Normally, the last move to finish a stitch is pulling the yarn through the loops on the hook. Only at the end of a decrease operation all unfinished stitches are closed off by pulling the yarn through all loops on the hook.

2.1.4 Internationally Known Diagramming System to Visualize Patterns

Stitch names vary between languages. Even British and American English sometimes have different terms, as seen previously in the list of basic stitches (Table 2.1). To eliminate such inconsistency, a diagramming system has been developed which uses an international standard notation where stitches are represented by a defined symbol[13]. This system is currently only used to represent flat patterns. Find an example chart in Figure 2.10.

Such charts can display clearly which stitches are used for a pattern. They show which stitch is worked into which, by rotating the stitch symbols to point to the insertion stitch. This can be observed in Figure 2.11, where Figure 2.11 (a) and (b) show the exact same three stitches: two chain stitches and a half double crochet stitch where the latter is worked into one of the chain stitches. The half double crochet stitch in Figure 2.11 (a) points with its lower end towards the left of the two chain stitches. This indicates that it is crocheted into that same chain stitch. Figure 2.11 (b) shows the half double crochet stitch rotated towards the right chain stitch where it is supposed to be crocheted into.

Exact entry points can be represented by additional symbols. Figure 2.12 shows an example for indicating a front post or back post insertion of a double crochet stitch. This type of insertion has been explained previously in Section 2.1.2.2.

Rounds or rows can additionally be kept visually apart by alternating the color of stitches every row or round, which was not done in this example chart. All symbols are arranged to mirror the resulting shape of the pattern on a 2D canvas.

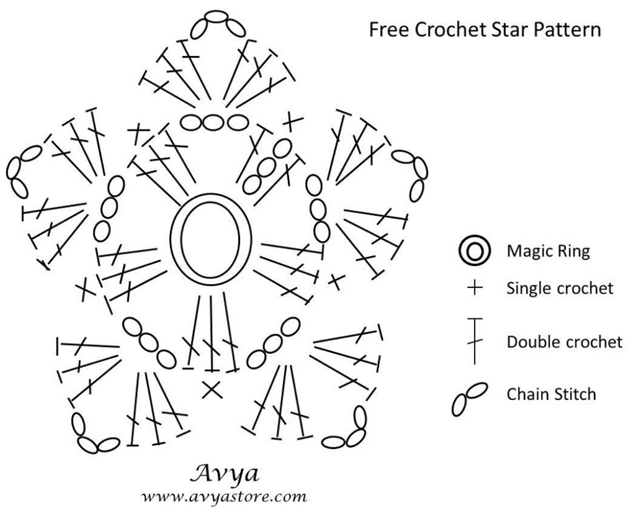
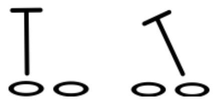
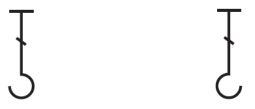


Figure 2.10: Example of a crochet chart for a flat star. Each symbol represents a stitch type as the legend shows.[1]



(a) (b)

Figure 2.11: In crochet charts, which stitch inserts into which is visualized through rotation of the symbols. (a) The half double crochet stitch marked as a T-shape point to the left chain stitch where it inserts into (b) Here, the half double crochet is meant to insert into the right chain stitch



(a) Front Post (b) Back Post

Figure 2.12: Special insertion points can be indicated through an addition to the basic stitch symbol. (a) Front post insertion is shown by adding a left facing hook under the symbol for the stitch, in this case a double crochet (b) For the back post the hook faces the right

The specification of this diagramming system is a great way to define clear instructions. But currently, the process of creating such a diagram is very manual. This means that in order to build a crochet chart for a pattern, the designer needs to manually place, move and rotate stitches until they look right. It is not an easy or exact process which leads to imprecise and ambiguous charts and cancels the initial intent of defining such standardized diagrams.

Additionally, there are some aspects of a pattern which cannot be represented with such a pattern. There is no symbol which indicates if stitches have to be crocheted into a chain stitch or under a chain stitch. The way to indicate where to start or which stitches belong to a row or round is not standardized and often forgotten.

2.2 The Perspective of Crochet Designers

In order to gain more insights on the crochet pattern creation process, we conducted a survey with professional crochet designers. The survey was distributed digitally to the designers of the German crochet company *myboshi*[2], who allowed us to

[1]Source of the image: www.avyastore.com.

[2]myboshi offers a website where crocheters can buy tools, yarn and pattern instructions or finished crochet hats. Designers share their instructions on the website for the users to buy.

contact their designers through a group on Facebook. Out of almost 200 members 25 responded and filled out our survey.

Additionally, we visited and interviewed two of the survey participants about their work to get a closer look at their workflow during the pattern creation process.

2.2.1 Survey about the Design Workflow

In the survey we asked six questions about different topics regarding the process of writing, testing and user feedback on their own crochet patterns. For each question we gave a set of possible answers. All but the last question allowed the participant to write additional answers in the form of free text so as to not restrict the outcome and miss important factors. All free-form answers have been condensed to the most frequent answers. Find an overview of the question topics and most frequent answers in Figure 2.13.

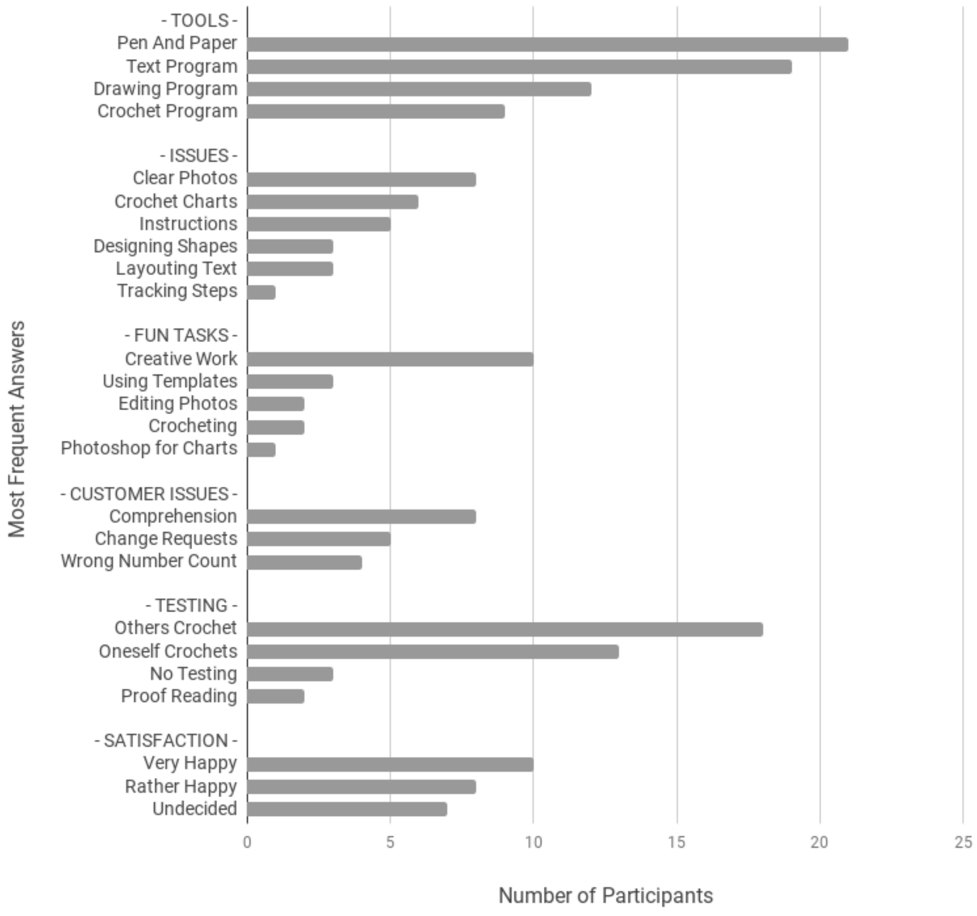


Figure 2.13: Answers given in the survey grouped by the questions

Tools for Pattern Creation We asked which tools were used starting from the moment a new idea for a pattern comes up until writing the final version of the instructions of that pattern. Almost all designers declared using pen and paper and text editors. While about half of them additionally use drawing programs, only nine of the designers use crochet programs to create charts. This shows that the workflow of creating new pattern instructions is very manual. Pen and paper seems almost always to be part of the process just like text programs. Only about a third of the designers use domain specific programs to create crochet charts.

Main Issues During the Workflow We asked what the most time-consuming and bothersome tasks are during the pattern creation workflow. Here, the participants did not agree on a few pressing issues. Some mentioned that creating the instructions can be hard, especially taking detail photos for step-by-step explanations, instruction formulations and layout. Others missed a good program for creating crochet charts which is not only available for English-speakers. And before instructions can be written down nicely, designers found the process of designing specific shapes and tracking steps while designing complicated. These answers show that clear instructions are vital to crochet designers and require carefully written instructions and clear images. There does not seem to be much support from crochet specific programs for any of these tasks.

Enjoyable Tasks During the Design Process We asked what the most enjoyable tasks are during the pattern creation workflow. Most designers agreed that being able to let their creativity run free is the main advantage of designing own patterns. Other things mentioned were crocheting itself, creating charts with *Photoshop*, editing product photos and using self made templates for pattern instruction writing.

It seems that creativity and well working tools and templates which facilitate the workflow to allow focusing on creativity are most important for these crochet designers.

Most Typical Issues with Customers We asked which were issues that customers had when contacting the designers about one of their designs. Mostly, the customers seem to contact the designer because of comprehension issues with the pattern instructions. But often, designers added that these questions could be solved easily by rephrasing the part that was unclear to the customer. Some customers also ask for help to change the pattern, either in size, yarn or towards other preferences. According to the designers, these requests usually need time-consuming, individual consideration. Another point is that customers reach out to the designers when the instructions had missing or wrong numbers such as the amount of stitches per row. This verifies that clear and correct instructions are very important. We can also see that little errors through oversight or unclear formulations cannot currently be evaded completely and lead to misunderstandings for customers.

Testing Process We asked how the designers are testing their instructions before making them available for purchase to their customers. When a pattern is done and instructions are ready, most designers have them tested by a group of other crocheters. They crochet the whole pattern to test for mistakes and unclear explanations. Another option, done by about half the participants, is to test the pattern themselves by crocheting it again. Only very few do not test their patterns or only rely on having patterns proofread. Again, we can observe that the whole process is very manual as there is no simple and direct way to validate a pattern.

Overall Satisfaction with the Design Process We asked how content the designers are with the current process of creating pattern instructions for a new design. Participants were asked to vote on a scale of 1 (unsatisfied) to 5 (very satisfied). More than two thirds of the designers voted satisfied (4) or very satisfied (5). The others were neither satisfied nor unsatisfied (3). Many designers mentioned previously in free-form text that they already built a well working, yet manual workflow for themselves, including templates for writing instructions. And in the previous question about the most enjoyable part, many designers were happy to be able to be creative in their designs. We can only assume that these factors led many designers to choose a high satisfaction value on this question. Nevertheless, many problems have been identified which, when solved, would drastically improve their workflow and lead to fewer inquiries from customers.

2.2.2 Two Qualitative Interviews

The survey gave us a general overview of the workflow of crochet designers. To gain more detailed insight into and a more thorough understanding of the workflow and verify the previously found issues, we visited and interviewed two of the participants about their daily work as a professional crochet designer. One designer specializes in creating three-dimensional objects, the other in flat fabric.

2.2.2.1 I Sell Experiences, Not Just Pattern Instructions

The first crochet designer that we interviewed mainly designs crochet hats. That means that most of the patterns result in a 3D object. The designer led us through the typical design process from the idea to the final design, talked about writing instructions and how crochet programs could support instruction writing.

Initially, with an idea in her head, the designer works on creating a miniature version of the hat she wants to make. In this process the yarn, proportions, structure and various patterns are tested to find out if they work well together. Nothing is written down yet. Later, a full size version of the hat is created based on the miniature hat. While this happens, the designer documents the exact pattern of stitches and already takes pictures for clarifying the step-by-step instructions later. Getting from an idea to a full size hat can take up to two weeks.

When writing the whole instruction document, she uses a text editor to transfer the handwritten notes into digital step-by-step instructions. She says she sometimes enjoys finding proper formulations as these help to personalize the instructions. According to the designer, she "sells experiences, not just instructions". But she tries to make the instructions not too complicated or else the customers would not crochet the designs.

When writing instructions, she would preferably write everything using pen and paper. Her instructions usually do not include any crochet charts as she is mostly making three-dimensional objects. But whenever she does need a crochet chart, she draws them by hand and includes a photo of it into the instruction document. If she were to use a program to help her with writing instructions, it would need to be structured in a logical way, according to the process of designing a crochet pattern. It should show the rough shape of the output, otherwise the designer might think that something went wrong. If the program could also output the instructions as text, it should consider specific crochet wordings in different languages. Simple translations do not suffice. While she herself does not enjoy using the computer, she did observe in the community that there is no good digital support for pattern creation. According to her "Everyone needs a proper crochet program."

2.2.2.2 I Just Don't Want to Enter Any Codes

The other crochet designer with whom we were able to get an interview, mainly creates colorful shawls. These are flat pieces of fabric, usually in the shape of a triangle. This designer explained how she creates her instructions, what is relevant in crochet charts to her and how crochet programs helped her so far and what is still missing.

First, the designer gets inspired by colorful wool and its textures. With an idea in her mind she sometimes starts crocheting immediately and only writes some notes to remember the types of stitches used in each row. Other times she begins by drawing a rough crochet chart by hand. In a chart the proportions and symmetry can be previewed before starting to crochet. In her instructions she always adds a chart and additionally a text for crocheters who do not know how to read the former. Furthermore, she includes detail pictures of the crocheted fabric when she thinks it helps the understanding. Finished pattern instructions are tested by a small group of people by reproducing the pattern.

According to the designer, a crochet chart should be self-explanatory and can be used as pattern instructions by itself. To make a chart clear she colors the rows alternating with black and red. She marks repeating parts within rows in yellow to highlight them. The charts also help her and her customers to imagine the shape and texture of the outcome of a pattern. The shape does not have to be completely exact but should match the final objects' shape. In other crochet charts from other designers she sometimes misses the clarity. Insertion points are not always clearly marked by rotation of the stitch symbols or explained in the text. She prefers very strictly defined instructions over loosely defined ones where the crocheter needs to take guesses. The crocheter always can deviate from the pattern later, after being presented with clear instructions.

Very often, the designer draws the charts by hand but she also uses the program *CrochetCharts by StitchWorks*. With this program she prefers creating rectangular charts as there is no good basis to start working in a triangle. Moving, rotating and mirroring stitches to reach the correct position is tiring to her. She is also missing a range of stitch types or the option to create her own. She would prefer to draw the charts with the computer rather than by hand but she is sure that she does not "want to enter any codes" to create a chart. When presented with the idea that crochet charts could also be used to represent 3D patterns she was surprised. It were hard to imagine but sounded useful. But she knows most customers like to print the patterns to have something tangible to keep track while crocheting. She found it difficult to imagine a solution to print 3D charts or make them otherwise tangible.

2.3 Summary

In this chapter, we presented the main techniques of crochet and analyzed issues in the current workflow of designers. The most important aspects of crochet for this report are that crochet is made up of successive stitches which can vary in type, height and insertion points; A pattern can be started using row or round techniques but they can be switched during the pattern arbitrarily; And also through the decreasing and increasing methods, the creation of arbitrary 2D and 3D shapes is possible.

Thanks to the survey and qualitative interviews conducted with professional crochet designers, we learned about several issues in their workflow of creating new patterns and what is most important to them. We found out that the whole process is still very manual. Mostly, designers use pen and paper, generic text editors or drawing programs to write their instructions or draw charts. Additionally, they test their patterns by having it crocheted or proofread. There exist no support to allow a validation of patterns which lead to customers reaching out to designers about comprehension questions which could easily be solved by clear formulations, correct instructions and numbers. Designers seem to enjoy most using their creativity to newly-create designs. They look for tools and workflows which do not impede their creativity.

All of the insights gained in this chapter are the basis for our further work on the digital representation of 2D and 3D crochet patterns, discussed in Chapter 3.

3 Digitalized Patterns: Graph Structure as Representation and Its Potential

Crochet pattern designing currently is a very manual process, as we found out in Chapter 2. The goal of this work is creating digital representations of crochet patterns which support crochet designers writing unambiguous patterns. We want to define patterns so that a computer is able to process them while being aware of the crochet domain. Our approach is to represent patterns as a graph structure which is constrained to connections existing in crochet. The graph tracks all links between and the order of stitches. Thus, we enable designers to crochet digitally. Instead of inserting the needle in a stitch to start a new one, a new node is created and connected to the previous and insertion node.

Our graph structure acts as domain specific language (DSL) which can be used to build crochet patterns. Thereby, it is not just human-readable but can also be processed by a program. If you will, you might even say it can program the human to crochet a pattern. But the raw graph structure is not an ideal representation for direct editing by domain experts, such as crochet designers. Therefore, we suggest to visualize the graph for editing purposes and thus, create a visual DSL. Refer to Chapter 4 about how we implemented the concept as visual DSL in a projectional editor.

In this chapter, we pick up the crochet methods which were specified in Chapter 2 to analyze what is needed to build a digital representation of crochet patterns. Furthermore, we discuss choices which shaped the concept of our approach. Next, we present the graph structure for pattern representation and how it can be used for the methods explained in Chapter 2. Finally, we present concepts which can be applied to the process of designing, thanks to the digital representation of patterns.

3.1 Basis for Digitized Patterns

In order to design a crochet pattern format which is able to represent arbitrary shapes, we first lay out the methods from Chapter 2, which need to be covered.

The pattern format needs to be unambiguous. If, in the future, there were a crochet machine it could follow the pattern precisely to deterministically reproduce the output object.

The techniques which must be representable by the pattern format are the following:

* Different Stitch Types
  Impacts height and connection points and is relevant for instructions

* Differences in Stitch Height
  Impacts the layout of the final design
* Crocheting in Rows vs. in Rounds
  Impacts layout and are relevant for instructions
* Increasing vs. Decreasing Stitches
  Impacts the shape and layout of the final design
* Insertion Points vs. Insertions into Holes
  Impacts texture and is relevant for instructions
* Different Starting Techniques
  Impacts texture and shape and is relevant for instructions
* Different Techniques to Change from One to the Next Row or Round
  Impacts texture and is relevant for instructions
* Connection Points of a Stitch
  Is needed for layout and keeping track of the order of stitches
* Order of Stitches
  Is relevant for knowing the order for instructions

By supporting all previously mentioned techniques, we assure that the pattern can represent any shape which can be created through crochet. The main advantages are that the pattern is unambiguously specified, can represent any, even three-dimensional, shape and can be processed by a program. Later on in this chapter, we will address the possibilities in more detail.

3.2 Considerations for Building a Pattern Representation

There are various options to create a representation for patterns. First, we will discuss possible underlying data structures and present which one we use for this approach. In the following, we give an overview of possible paths which can be taken to gather pattern data and represent, visualize and interact with digitized patterns.

3.2.1 Digital Representation Format

In order to define a pattern format, we needed to decide on a data structure. This format dictates the way we will be able to work with it, how the crochet methods are implemented and the flexibility we have when it comes to transformation and calculations on the data.
The considered options, summarized in Table 3.1, were text file formats, data structures such as arrays or linked lists and a graph structure.

First, we considered using a **text file format** as it is close to current written instructions. Guided by the format of the text file, instructions could be written

Table 3.1: Representation format considerations

	Pro	**Contra**
Textfileformat (xml/json/txt/...)	Direct mapping from pattern format to storage format. Similar to current written instructions.	Difficult to do calculations on sets of instruction codes.
Tabular data structure (2D Matrix, 3D Matrix)	Simple representation format. Allows easy design of pixelart-like patterns.	Restricts free-form crochet possibilities to rectangular shapes. Changing grid size or shape not supported. Only indirect interconnection between cell entries.
Linked list as data format	Keeps track of instruction order.	Only allows linear connections, no direct interconnectivity.
Graph as data format	Represents interconnectivity well. Traversal allows calculations and analysis.	High space consumption.

in clear notation to unambiguously represent patterns. While it is benefiting to have a structure which can be written and stored in the same format, it does not easily support running calculations.

Another idea is to represent patterns as **tables** where each cell stands for a stitch. This is already used for knitting and very basic types of crochet. When using **two-dimensional arrays** only flat objects can be represented, so **three-dimensional arrays** could be the solution for designing 3D objects. But while the grid size cannot be refined for each layer, the representation range is restricted to rectangular and other angular objects. Each row would need to have an adjustable amount of entries or many cells would be empty and data would need to be rearranged in the array when a smaller refinement is needed.

The data structure of a **linked list** gives more flexibility and allows creation of arbitrary structures while keeping track of instruction order. With crochet, as we said earlier, arbitrary shapes can be built since stitches can be connected to any insertion point in the whole pattern. But lists supports rather only linear connections which means the representation of interconnectivity suffers when using this format.

Finally, we considered using a **graph**. By nature a graph represents interconnectivity well while also allowing traversal. On a graph we can run calculations on a pattern for analysis or transformations. But because of possibly high interconnectivity between nodes, a graph can have higher space consumption than the alternatives.

For this report we chose to use a graph to represent crochet patterns as it allows to represent arbitrary shapes, traversal and calculations on patterns.

3.2.2 Pattern Data Source

As currently all crochet patterns are written manually, there is no direct way of gathering the data automatically. Therefore, we took into account different options such as manual input of instructions, analysis of existing patterns, analysis of crochet needle movements while crocheting and modeling shapes and converting them into crochet instructions. Find an overview in Table 3.2.

Table 3.2: Data source considerations

	Pro	**Contra**
Manual input	Input is immediately in the desired data format.	No usage of existing patterns. Requires crochet knowledge.
Pattern Analysis	Reuse existing patterns. No active participation from community required.	Extract information from unstructured, possibly faulty or incomplete data.
Needle Movement Analysis	Crocheting yields pattern directly.	No usage of existing patterns. Requires pattern to be fully crocheted.
3D Modeling	Requires no crochet knowledge. Editing resulting shape instead of instructions.	No usage of existing patterns. 3D modeling experience needed.

Manual input requires a crochet designer to actively enter the data for each pattern. Existing patterns can not be leveraged automatically but the data we get from manual input would be exactly in the format we want for further steps.

Pattern analysis could take advantage of the vast amount of existing patterns. Thus, no crocheter or designer would need to actively give input. But, as we explained in Chapter 2, current patterns can be of various unstructured data formats such as videos, text and pictures and might contain mistakes due to oversight or ambiguous formulations. Analysis of this type of data requires a lot of resources, for training an algorithm, considering multiple data formats and taking decisions manually.

The idea to **analyze the needle movement** while crocheting requires not only many resources for gathering a representative amount of data and training an algorithm but also the design and creation of a needle which can track movement. When achieved, it would allow crochet to lead directly to pattern instructions without changing the context. Existing patterns could not automatically be leveraged, any pattern would need to be crocheted as a whole to be registered.

Modeling a crochet object in 3D could enable even non-crocheters to invent new designs as the output could be edited directly without requiring any knowledge about the low level instructions. On the other hand, it required crocheters to have 3D modeling knowledge to be able to create patterns using this method. A lot of effort would have to go into the transformation of such shapes into crochet patterns. In order to create and edit these shapes, they would have to be transformed into a digital representation, such as the one we propose in this report.

This, and the further considerations do not affect our concept but is relevant to any software using it. For our proof-of-concept, we chose to use manual input.

3.2.3 Possible Transformations of the Pattern Format

Previously, we decided to use a graph as underlying representation of crochet patterns. As discussed in Chapter 2, crochet designers typically work with visual tools or use text to create pattern instructions. In Table 3.3, we discuss possible visual representations to be presented to the user, starting by computer readable serializations and aiming towards the actual crocheted result of a pattern.

In between options are text, as it is already known to designers, plain graph visualization of nodes and edges, domain specific adaption of the graph to represent crochet charts and a computed, realistic model.

A **serialization** is required to store patterns but not suitable as visual representation. While it stores all instructions and relevant information about a pattern, it does not give a clear overview of the structure which could help the user navigate, read and understand the pattern.

Instruction text is currently most widely used by designers to specify patterns. Nevertheless, there has not been developed a standard format, yet. The underlying graph representation needs to be transformed into well structured human readable text. Additionally, if multiple languages should be supported, the transformation process will have to be adapted to each of them.

Table 3.3: Visual representation considerations

	Pro	**Contra**
Serialization	Same format as for storage of pattern.	Not visual. Not intelligible at first glance.
Instruction Text	Known, widely used format by designers.	No existing standards for text structure. Generation of well structured, human readable text is complex.
Plain Visualization of the Graph Structure	Direct mapping to underlying structure.	Unknown representation to designers. Not intelligible at first glance.
Graph as Chart	Known, widely used format by designers. Straightforward mapping to underlying structure.	Exact layout of graph is highly relevant but complex.
Realistic Model	Direct mapping of real pattern outcome. Exact preview of pattern result.	Instructions not visually discernible.

A **plain visualization of the graph structure** can be a simple straightforward visualization of the data structure. To designers, the presentation of nodes and edges is currently unknown as representation. The instructions of the pattern are not visually mapped to the graph elements, thus making it difficult for designers to read and work with the visualization.

Visualizing the **graph as a crochet chart** clearly maps the instructions to the graph structure. Already known stitch symbols can be used to be laid over the nodes and edges. Thereby, designers can work with a familiar representation. But in order for it to be well readable, the graph requires a correct layout which also mirrors the shape of the actual crochet output and keeps appropriate distances between stitch symbols.

Ultimately, a **realistic model** could be computed to show the yarn flow with proper textures and shapes of the pattern outcome. Such a representation can be the exact preview of a pattern and can help designers imagine the result and compare shapes and textures. However, the recognizability of instructions is lost through such a transformation.

In our proof-of-concept prototype, we will visualize the graph as crochet chart.

3.2.4 Approaches to Layout Visual Pattern Transformations

When using a visual representation, such as the crochet chart graph, there are different options to layout the visualization.
As summarized in Table 3.4, we discuss options such as letting users manually decide the layout, providing grids or other snapping guides, using a generic layout algorithm which automatically computes positions and a domain specific layout algorithm which takes advantage of domain knowledge to calculate the layout.

Table 3.4: Graph visualization considerations

	Pro	**Contra**
Manually Positioning Nodes	Full control over layout.	Requires user to think about layout. No automatic even distribution.
Snapping Guidelines	Full control over layout. Supports user in placing nodes.	Requires user to think about layout. No automatic even distribution.
Using a Layout Algorithm	Calculates Layout Automatically. Lets the user focus on the pattern. Correct layout helps imagine the output.	Graph Layout Algorithms usually face NP-hard problems and are based on finding approximate solutions. Wrong or unclear layouts may make a user think something is wrong with the pattern.
Layout Algorithm Based on Domain Knowledge	Precise and realistic layout.	May require intensive and complex structural analysis.

When giving the user the power to **manually layout** the graph, nodes could be dragged to the wanted position. But this option does not let the designer focus on creating the pattern. The nodes are not distributed automatically and the user has to make an effort to give an even shape to the pattern.

The usage of **snapping guidelines** could be an improvement to the manual layout. Users can take advantage of grids and other guides to help position stitches evenly. Yet, the user still cannot solely focus on creating the pattern and has to manually input the layout.

A **graph visualization algorithm** would allow the user to focus completely on creating a pattern while the layout additionally supports the process. Additionally, it created an even distribution of nodes or stitches, which lead to comparable results, even from differing authors. A visualization of the output can also help users understand how the pattern will turn out. On the other hand, layout algorithms for graphs typically face hard to solve problems which is why many algorithms are merely approximations. This can lead to incorrect or confusing visualizations which counteract the mentioned upside of automatic layouts.

Therefore, it is relevant which algorithm is used. Most of the algorithms are built to display large unstructured data. Connections of such data can be unpredictable. But, in our case we know that the structures can be arranged in 3D space since the patterns lead to a real world object.

Common graph layout algorithms include force-directed, topological, tree and space-filling approaches. The last three seem less appropriate as our data is not structured as trees or contain specifically structured sub-graphs. Force-directed node-link layout algorithms seem like a trivial way to layout graphs while naturally spacing nodes.

Using domain knowledge to calculate the graph layout results in a more precise visualization compared to using a generic algorithm. Additionally to using structural knowledge about crochet, the user can give further indications to the algorithm, such as specifying which side of the fabric is supposed to be the inside. To realize this approach, structural analysis would be required which demands large calculation effort and time.

For our prototype, we will use a generic force-directed algorithm to layout the pattern graph to the user for editing.

3.2.5 Interacting with and Editing the Digital Pattern

Another aspect to consider when using our proposed pattern representation is the method by which users can create and edit them. We present (see Table 3.5) the following options: plain command line interface (CLI), a domain specific programming language, a graphical user interface (GUI) and directly designing patterns through 3D modeling.

While a **CLI** may be fast, uses fewer resources and provides high flexibility through commands, it is not suitable for crochet designers. As directly stated in an interview (Chapter 2), designers do not want to enter commands or use interfaces close to

Table 3.5: User interactions considerations

	Pro	Contra
Command Line Interface	Requires few resources compared to using GUI. High flexibility for user through commands.	No immediate visualization while editing a pattern. Flat learning curve. Not intuitive to use.
Programming Language	Abstraction of crochet methods and structures. High flexibility for user through commands.	No immediate visualization while editing a pattern. Flat learning curve. Unfamiliar tool to crochet designers.
GUI with Buttons executing commands	User interaction is decoupled from graph layout.	Offering all possibilities as buttons may overload the interface. Similar to entering codes which is not intuitive and has a flat learning curve.
Click Graph to Add Stitches	Similar to crochet: click resembles insertion of crochet hook.	Requires visual contact to next insertion point which gets harder in 3D.
Modeling Shapes	Requires no crochet knowledge. Editing resulting shape instead of instructions.	3D modeling experience needed.

programming languages. A CLI does not offer instant visualization or an intuitive user interface and the learning curve is flat.

Similarly, a **programming language** provides high flexibility but is not a common tool in the crochet design community.

Instead of typing in commands, we could offer an interface with **buttons to execute the most basic commands**. The visualization could update at the same time as commands are given but the interaction would be decoupled from it. Just as crochet has many different methods that can be used, the amount of commands that should be offered through buttons might overload the interface. This process is still very similar to entering commands, now only with the hidden console and strapped from flexibility due to fixed amounts of buttons.

An option which resembles the process of crochet the most is to allow **direct interaction with the graph** to input data. Clicking a node could add a new stitch. While this process is intuitive, it can restrict efficiency when patterns get large and confusing.

Designing patterns through **3D modeling** allows working directly on the resulting shape of the pattern. Thus, no crochet knowledge is required but 3D modeling experience is needed.

In our implementation, we will allow interaction directly with the graph. Stitches will be able to be added through clicking on nodes.

3.3 Graph Structure as Underlying Format

As discussed earlier in this chapter, we chose to use a graph to internally represent crochet patterns to leverage its interconnectivity and ability of traversal. Node and edge combinations represent stitches with their connections. By traversing the graph in a specific manner, the general path of the yarn through all stitches can be followed from the beginning to the end of the pattern. Each node represents possible entry points for the hook.

The structure of the graph is designed to be able to represent all of the main crochet characteristics and methods, previously explained in Chapter 2.

In the following, we present how crochet is represented by the structure of our graph.

First, we will establish how stitches and their connections can be modeled. Then, we present how the established graph structure can be used to model the crochet methods explained in Chapter 2.

3.3.1 Representing Stitches in a Graph

Before we can talk about how to represent any crochet methods, we need to model stitches, the smallest entity of a crochet pattern, in a graph. Previously in this chapter, we defined a basis on which we build our representation. This included the representation of stitch types, their height, connection points, insertion points and order of stitches. In the following, we explain how we can model that in the graph.

First, we group the standard stitches presented in Chapter 2 by the way they connect to neighboring stitches. We show the equivalent of each defined stitch type modeled in the graph. Finally, we show how insertion points are marked and additional spaces can be modeled as node to allow insertion.

3.3.1.1 Differentiating between Stitch Types

Chapter 2 presented main stitches such as the chain stitch, slip stitch, single crochet, double crochet and more. We analyzed the way that all of these stitches connect to each other and identified a pattern which groups all stitches into three categories. For reference, we have included again the figure from Chapter 2 which shows three example stitches in their making, see Figure 3.1.

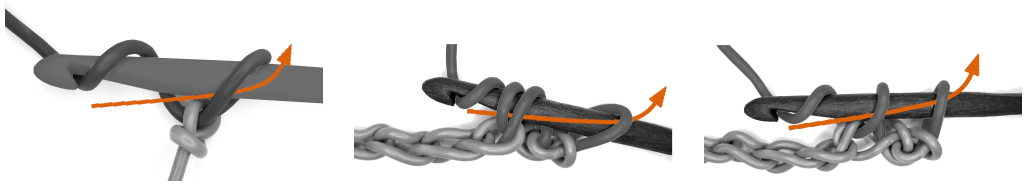

(a) Chain Stitch (b) Slip Stitch (c) Single Crochet

Figure 3.1: As a recap, the yarn flow is shown again for three stitches, each representing one group of stitches

The **chain stitch** (Figure 3.1 (a)), only raises the current working height. It does not connect to any insertion point but allows insertion in or under that stitch for other upcoming stitches.

A **slip stitch** (Figure 3.1 (b)) merely connects two points while not gaining any height or creating any new insertion points.

All the other stitches presented in Chapter 2, such as the **single crochet** (Figure 3.1 (c)), always raise the working height. The amount depends on the type of stitch, they insert into an existing insertion point and create a new one.

This distinction is directly applicable to a graph structure. Insertion points are represented as nodes and connections between stitches are modeled as edges. On the nodes we label the type of stitch and any other information relevant to the pattern such as height, color, etc. On edges we label what this connection models. In Figure 3.2, we show the representation of the chain stitch, slip stitch and the single crochet as representative for its group.

The **chain stitch**, shown in Figure 3.2 (a), is modeled by a single node and link. The node represents the new insertion point that was created and the link is the connection to the previous stitch. To allow traversal and to keep the order of the stitches we mark this link as 'previous'.

The **slip stitch**, shown in Figure 3.2 (b), is modeled by a single link. There is no node, as a slip stitch does not create any new place for insertion. Since the connection made by a slip stitch merely connects two points and is not marking any previous node, it is simply labeled with the name of the stitch 'slip stitch'.

The other stitches, like the **single crochet** (Figure 3.2 (c)), are modeled by one node and two edges. The node marks the insertion point and one of the links represents the connection to the previous node, similarly to the link of the chain stitch. The other link is labeled as 'insert' and represents the connection between the current stitch, in this case the single crochet, and the insertion point. On insert edges we label the type of insertion (explained in Chapter 2), such as both loops, back loop only, front post, etc.

As an example, Figure 3.3 shows how such a structure maps to crocheted fabric. In Figure 3.3 (a) the first two stitch types are shown and in Figure 3.3 (b) the usage of the slip stitch to connect two chain stitches is presented.

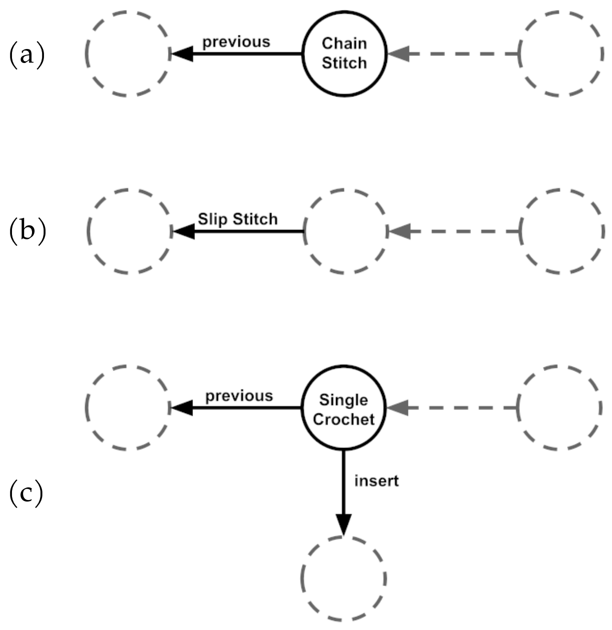


Figure 3.2: Representation of each stitch group in the graph structure: (a) The chain stitch only gains height and does not insert into any stitch (b) The slip stitch connects to stitches (c) The last group, represented by the single crochet stitch, gains height and inserts into another stitch

By handling stitches as a set of nodes and edges, we give the ability to traverse the graph along the stitch connections. Using the edges labeled 'previous', we can move along the path of stitches in the order they have to be worked in a pattern. The 'slip stitch' labeled edge is a special form of a 'previous' edge. It can only exist when the node it is connected to already existed when making that connection. This most often will represent the end of a row or round in a pattern. For traversal it means that when encountering a node with an out-going previous and in-going slip stitch edge, first the slip stitch edge is followed. When encountering the same node again, then the previous edge is followed.

3.3.1.2 Supporting Any Stitch Insertion Points

As just presented, insertion points in stitches are represented through nodes and the insertion edge specifies which stitch connects to it and into which exact point.

But sometimes, it is not enough to specify a stitch to insert into. In crochet practice it is very common to not insert into any chain stitches but rather under the whole stitch. And when there are multiple chain stitches in a row an opening is created

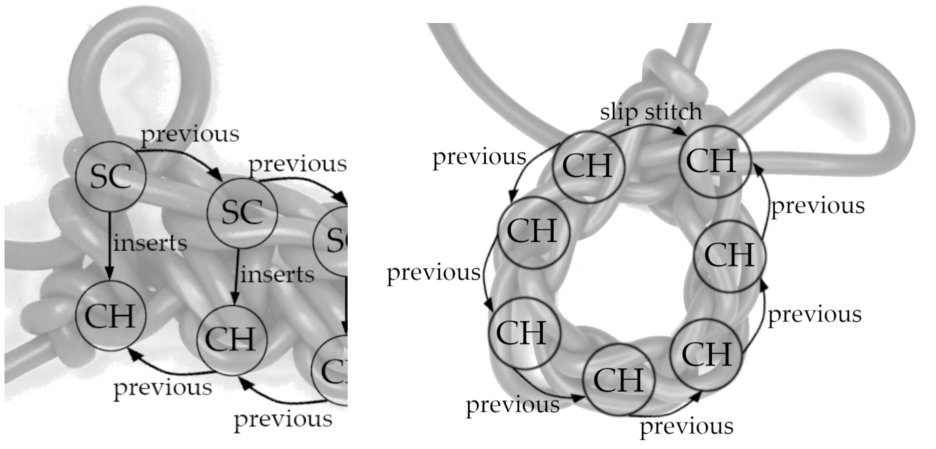


Figure 3.3: Example graph structure mapped onto crochet fabric. (a) Clipping of a two-row pattern. The first row consists of chain stitches (CH), the second of single crochets (SC) (b) Chain Round of seven chain stitches (CH) and closed using a slip stitch

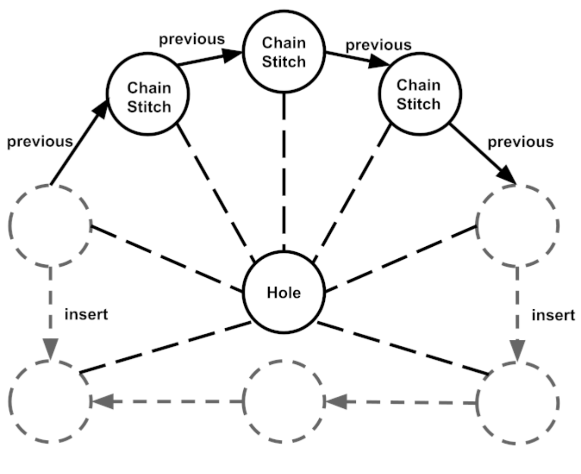


Figure 3.4: Representation of holes as additional insertion points in the graph structure. This example allows the insertion into the space under a line of chain stitches

where stitches can be inserted into in the same manner. This technique is currently not represented in standard crochet charts. Either it is specified in a text along with the chart or the chart reader has to take a guess which version might have been preferred by the designer.

To remove this ambiguity and allow direct definition of such insertion points, we represent holes as nodes, since they are also used as insertion points like any of the other nodes. This hole node is connected through edges to all of the other nodes surrounding the hole. A visualization of the graph structure for holes is shown in Figure 3.4.

3.3.2 Modeling Crochet Techniques in the Graph Structure

By defining stitches and their connections, we have already established a basic structure for the graph. With this, we already covered many points from the previously defined basis in Section 3.1. In the following, we explain how we can model the remaining crochet methods which are the key to allow the creation of arbitrary flat and three-dimensional shapes. As such we present pattern starting techniques, crocheting in rows or rounds, changing from one row or round to the next and increasing and decreasing stitches.

3.3.2.1 Starting a Pattern

As presented in Chapter 2, there are three options of starting a pattern, such as a line or round of chain stitches or a magic ring. In continuation, either the row or round based method can be used. Since a magic ring is an insertion point, it is modeled through a single node. No starting stitches have out-going edges, as there are no previous stitches to connect to. Further stitches can be connected to the start stitch as described previously, according to the type of stitch.

3.3.2.2 Working in Layers

Typically, crochet patterns are structured by rows or rounds. We unite these two terms into one: *layers*. In the graph, we do not need specific changes in structure to represent either a row or a round. Which method you use solely depends on where you continue inserting stitches after finishing one row or round. Therefore, we merged these terms into one since a pattern can also change anytime from crocheting in rows to in the round. Depending on the connections made, you can imitate one or the other method. The graph tracks the layer number of each stitch as property in the node.

3.3.2.3 Starting a New Round

Finishing and starting a round can be done using either the method of continuous rounds or joined rounds. When crocheting continuously, see Figure 3.5, the first stitch of the new round (№5) inserts into the first stitch of the previous round (№1). When crocheting joined rounds, see Figure 3.6, the last (№4) and first (№1) stitch of the current round are joined with a slip stitch. Then, the new round is started

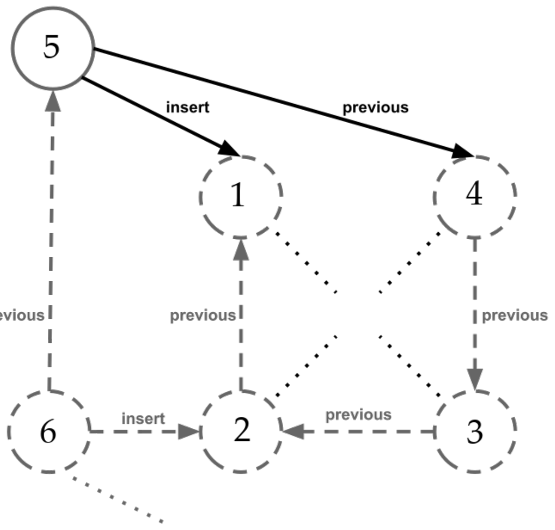


Figure 3.5: Representation of continuous rounds in the graph structure. Rounds are not closed, the next one is directly continued instead

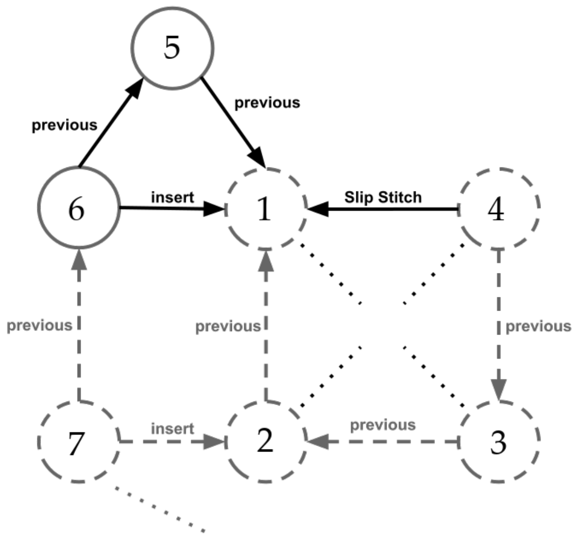


Figure 3.6: Representation of joined rounds in the graph structure. Before starting a new round the previous one is closed using a slip stitch

by a chain stitch (№5), to gain height for the next stitch and then the round can be continued with stitches of the third group, such as the single crochet.

3.3.2.4 Starting a New Row

In order to start a new row you proceed by crocheting any amount of chain stitches to raise the height to the working height. The working height is defined by the following stitch to come. Then, the pattern can be continued by starting to insert into the last stitch of the previous row. Like this, the rows are worked alternating from left to right and vice versa.

3.3.2.5 Increasing and Decreasing Stitches

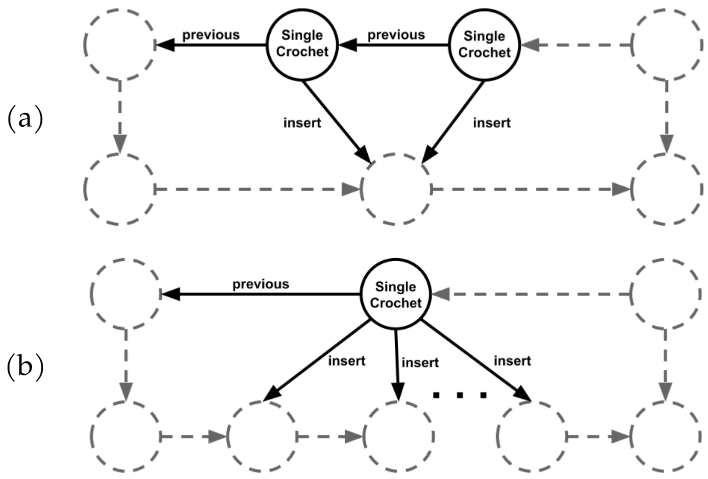


Figure 3.7: Representation of Increase (a) and Decrease (b) Methods in the Graph Structure

In order to allow the representation of the increasing and decreasing techniques, we allow multiple incoming insertion edges and multiple out-going insertion edges for the nodes. If a node has exactly one out-going insertion edge, such as the single crochets in Figure 3.7 (a), this stitch is an increasing stitch. The node that receives these insert edges is actually only being increased if it has more than one incoming insert edges. Otherwise, there is neither an increase or decrease happening, as the number of stitches on that layer, compared to the previous layer, is not altered by this stitch. Figure 3.7 (b) shows a single crochet used to decrease multiple stitches at once. The amount of stitches for that layer are reduced by the amount of out-going insert edges subtracted by one.

3.4 Advantages of the Digitization of Patterns

The graph structure we just defined allows representing any pattern which can be created through crochet techniques explained in Chapter 2. This structure is the basis which enables further concepts to be applied to patterns. When designers currently 'digitized' their patterns, they meant typing text with a text editor or drawing an image using a generic drawing program. All of these representations were digital and editable, but they were not based on a standard format and programs were not aware of the crochet domain or any crochet connections. Instead of editing lines of text or positioning images on a canvas, a program which uses our graph structure for representing crochet patterns can now support the designer in crochet specific ways. Crochet concepts such as rows, rounds, stitches and insertion points are incorporated in the graph structure we designed. In the following, we present concepts which we see as direct consequence to the digitization of crochet patterns. First, we analyze the meaning for the crochet domain, when patterns are digitally represented. Second, we show how common edit functionalities can now be applied to patterns in the same way as they are known from text or other editors. Last, we show domain specific calculations and analyzes which can, among others, support the editing of crochet patterns.

3.4.1 Impact on the Crochet Domain

In Chapter 2, we conducted a survey and interviewed designers. One of the findings was that writing correct pattern instructions is difficult, as the writing and testing process is manual and many mistakes are made by oversight. The concept we proposed tackles these problems. Each instruction is represented in the graph structure. In manually written instructions, designers often included the count of stitches per row or round in addition to the stitch instructions. This duplication is a typical source for erroneous information. With a pattern representation, like our graph structure, these clarifications do not need to be given and could even be calculated automatically by a program. Other findings were, that customers sometimes have troubles understanding the formulations used by designers in their patterns to describe certain parts. Thanks to the stitch connections modeled directly in the graph, all instructions are clearly represented. In order to present such digitized and exact patterns as human-readable instructions to customers, the graph can be visualized like the already known crochet charts or clear and correct written instructions could be generated automatically. In the crochet domain, crochet charts are currently only used for 2D patterns as they can be drawn on a flat canvas. Our graph can not only represent any arbitrary shape but it can also be used to visualize them by rendering the graph in a 2D or 3D environment.

3.4.2 Editing of Patterns

Similarly to the possibilities of editing text in a text editor, our representation can be edited as well. Concepts such as **selecting** nodes and edges or **undo and redo** can be implemented in a straightforward manner. But the concept of **copy and paste** will require further calculations and decisions to make it work for the crochet domain. A selection of a pattern might need to be adapted for insertion at a different position in the graph. In crochet patterns, especially when using the increase or decrease methods, a whole layer cannot be simply duplicated, like you would duplicate a line in text. Here, a program would need to find domain specific solutions and handle various functions separately, such as layer duplication and exact selection copies.

As explained in Chapter 2, apart from the basic stitches there are many possibilities of combining them into a new stitch. For example, working five double crochet stitches into a single stitch is called shell stitch[13]. When designers currently draw crochet charts using a program, they use an image for each of the possible stitch combinations. Instead, using our pattern representation, partial patterns can be defined which specify stitch combinations. Such predefined combinations can then be used within a whole pattern. For designers, this gives the flexibility of creating their own combinations and they do not require an extensive and confusing list of all stitch combinations. Such **partial patterns** consist of a set of nodes and edges which represent the stitches which make up the combined stitch. Additionally, nodes are included as place holder, which specify the connection points to surrounding stitches in a pattern. An example for a shell stitch of five double crochet stitches as partial pattern is given in Figure 3.8. When such a combined stitch is added to a pattern, the connections are closed by matching the place holder nodes to existing nodes of the pattern.

Such partial patterns, which represent a combined stitch are just one example of **modularity** of patterns. One pattern could consist of multiple parts which define separate pieces of the whole item. The connections between the parts are either specified through stitch connections by matching the place holder stitches or by adding additional properties which define the type of connection, for example sewing or crochet connection. When positioning such parts in a visual environment, they could snap to typical locations, such as the center of a piece of fabric to specify the place of connection.

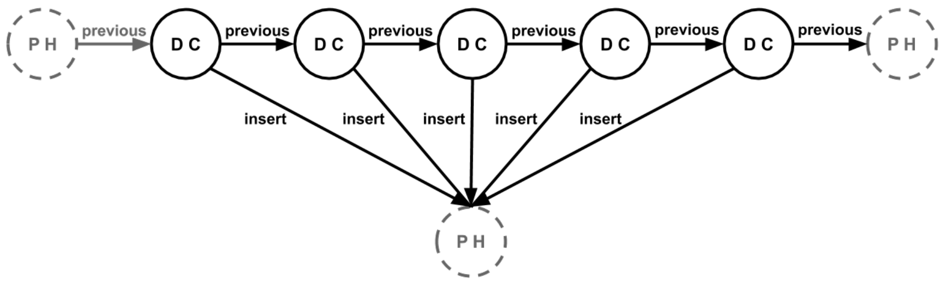


Figure 3.8: Representation of partial patterns in the graph structure. With the example of a shell stitch of five double crochet (DC) stitches. Place holder nodes (PH) mark open connection points

Another typical feature of text editors is **auto-completion** of words or constructs. Since the graph structure has an awareness of rows or rounds, it is possible to analyze the pattern of the current layer and then repeat it until the end of the layer.

Similarly, to facilitate the editing process of large, repetitive patterns, partial patterns can be used to design one main stitch structure which could automatically be completed over a given amount of rows or rounds.

3.4.3 Verification and Analysis of Patterns

Any crochet pattern which uses the methods from Chapter 2 can be represented using our graph structure. Vice versa, any graph structure should be able to be crocheted. Depending on the edit methods given to create and modify the pattern, incorrect states may be allowed in the process. In our proof-of-concept presented in Chapter 4, we are limiting the user interface to allow only creating valid connections and thus, always building a crochetable graph. In other scenarios, a constraint solver would be required to test the graph structure for validity. Being able to prove the correctness of a pattern is an important factor for customers, as they can be sure that the pattern is **verified** and that they can crochet it without encountering any impossible or ambiguous structures.

Furthermore, the graph **structure could be analyzed**. Since our representation supports arbitrary 3D patterns, the designer might be interested in knowing if a pattern is actually flat or of what shape will be the result. This requires information about each stitch type, its height, space towards other stitches and physical behavior of yarn.

Apart from mathematically calculating the curvature of pattern surfaces, the graph structure can also be used as input for a layout mechanism. This way, charts can be drawn automatically and designers do not need to manually position and rotate stitch symbols. Thanks to the internal representation of structures, the connections between stitches are given which can be used to automatically rotate stitch symbols to point towards the correct insertion points.

Another applicable concept for graph analysis is the **identification of repetition**. Patterns usually consist of many repetitive parts. If a program could directly identify them, they could be handled as one entity, for example for written instruction generation to condense the text. Alternatively, patterns could also be compared with each other to **detect overlaps**. If one pattern features the same repetitive stitch structures as another, it could be an indication of fraud. Since, currently, there exist no international standards for textual instructions, it is not easily detectable if parts or the whole pattern are actually stolen from another designer. Further, a pattern could be **quantified**, for example by difficulty, duration and yarn usage.

3.4.4 Transformation of Patterns

As discussed earlier, our underlying graph data structure is not the ideal format to present to designers and customers. The graph can be transformed into various other formats. For **visualization** it can be rendered as a chart in a 2D or even 3D environment.

Written **instructions could be generated** from the graph structure. The analysis of repetitions in the graph can be used to write clear and condensed instructions in any language for presentation to customers. If there existed a crochet machine, the graph could be transformed into machine code which would instruct the machine step-by-step what to do in order to reproduce a pattern. Until a real, fully functional crochet machine exists, there could be a program which simulates a crochet machine or a human crocheter. It could use the instructions to calculate the exact shape, texture and yarn behavior for finally displaying the completely crocheted object and the steps to reach that output. This could be rendered into instructional videos or used are output shape preview

Apart from transforming the pattern into various formats, the instructions could be transformed into **variations of the same pattern**. If there is a way to determine the difficulty of a pattern, it could be simplified. Or if a pattern is of a specific size, such as piece of clothing, it could be generated for a larger or smaller size. The instructions for the design could be calculated for crocheting it in reverse, from the end to the beginning. Or, when a pattern consists of multiple pieces which need to be crocheted or sewn together to join them, an algorithm could identify a path that creates the same shape, using the same stitches, but in a different order. This could find a path to crochet the whole item at once, without the need of separation into individual parts.

3.5 Summary

In this chapter, we presented our concept for a digital crochet pattern representation. It is based on a graph structure which incorporates and is constrained to all crochet methods explained in Chapter 2. This graph structure acts as a domain specific language which can be used to represent arbitrary crochet patterns. In Chapter 2, we identified issues in the current crochet pattern designing workflow which are tackled by this approach. Through our graph representation of patterns, we provide a format which removes ambiguity from pattern instructions and also allows analyzing patterns for mistakes which currently could easily be overlooked due to the manual writing and testing procedure. Additionally, we support even three-dimensional structures, whereas current crochet pattern charts were limited to flat objects. Our approach is the basis for the digitization not only of crochet patterns but of crochet designers' whole workflow.

4 Proof-of-Concept: Implementing a Projectional Editor for Digital Crochet Patterns

Previously, in Chapter 3, we designed a theoretical approach to represent crochet patterns as graph structure. Additionally, we suggested various concepts which can be built thanks to the digitization of patterns. In this chapter, we present a prototype[1] of a projectional editor as proof-of-concept which is based on our graph structure, uses manual input of pattern data, visualizes patterns as crochet charts in 2D and 3D, makes use of a generic force-based layout algorithm and allows users to work with the visual representation using a graphical user interface with point-and-click interaction.

4.1 Technology Decisions for Building the Prototype

Before starting the development of the prototype, we discussed whether to build a desktop or web application, see Table 4.1. Since we wish to enable visualization and automatic layout of patterns, we investigated possible supporting tools for our chosen platform, see Table 4.2. Finally, we will give an overview of the components of the prototype.

When building a 3D application, a typical starting point is to build a desktop application since the *OpenGL* interface can be used to render graphics. Additionally, such an application is not dependent on internet connection which could be a bottleneck for performance. On the other hand it requires installation on each device, it is restricted to the hardware available on the device it is installed on and to offer it to any platform, different implementations of the whole program are needed.

When building a **web application** anyone with the access to internet and a browser can access the program, independent of device and hardware. Typically, one implementation can run on any browser, but there might be some differences in how a browser handles the styles or some functions may not exist, which can require partial browser-specific solutions.

For this report we chose to build a web application. Not only does this allow us to make the prototype easily available to testers during the evaluation period but in general it will keep the barriers low, especially for non-tech savvy crochet designers.

[1]The source code is hosted under MIT license on `https://github.com/klaraseitz/LayerwiseCrochet/`.

Table 4.1: Platform considerations

	Pro	**Contra**
Desktop Application	Can use OpenGL for efficient 3D rendering. Not dependent on internet connection.	Requires installation. Restricted to local hardware. Different platforms require different implementations.
Web Application	Accessible to anyone with internet connection and browser. One version can run on many browsers. Not dependent on local hardware.	Requires network connection. Browser specifics might need different implementations.

For building our single page application, we used the JavaScript framework *Vue.js*[40]. Other considered frameworks were *AngularJS*[23] and reactjs[38], but they are more complex and powerful than needed. Using a framework to simplify building clear, usable designs was an important step to keep the time minimal for user interface design and allow focusing on the implementation of functions.

Table 4.2: JavaScript library considerations for graph visualization

	2D	**3D**	**Interactive**	**Physics**	**Free**
Syncfusion[36], *GoJS*[32], *yFiles*[41], *KeyLines*[11], *ZoomCharts*[45]	✔	✗	✔	✗	✗
Graph Dracula[34], *Sigma.js*[3]	✔	✗	✔	✗	✔
Vis.js[6], *D3*[8], *Cytoscape.js*[15], *2D Force Graph*[5]	✔	✗	✔	✔	✔
Three.js[10], *3D Force Graph*[4]	✔	✔	✔	✔	✔

In order to visualize the graph structure which represents crochet patterns, we need a library for JavaScript which supports visualizing graphs in a 2D and 3D environment. Additionally, this visualization has to allow interaction so that the user can edit the graph directly. Furthermore, we preferably look for an automatic graph layout which does not solely rely on manual placement of nodes. For that, we require physics support which would allow automatic distribution based on forces. For JavaScript exist many libraries to visualize graphs. Mostly, the visualizations are meant for data analysis but could also be used in our case. Table 4.2 lists various possible libraries and marks whether they fulfill the previously specified requirements. Additionally, we show if the library is a commercial product or freely available. Finally, we chose the 3D Force-Directed Graph[4] which uses Three.js and WebGL for rendering and a variant of D3[8] as physics engine[7]. With this library, we can provide an interactive environment for visualizing and editing the pattern graph. The underlying physics engine enables automatic layout based on the force layout algorithm by Dwyer[14].

In our prototype, we implemented a switch from the 2D to the 3D environment. Here, we used, next to the 3D Force-Directed Graph[4], the 2D Force-Directed Graph[5] which is a variation written by the same author and offers the same interface.

In overview, we built a single page application using vue.js for the browser. We are using a library which supports the visualization of graphs in a 2D and 3D environment using physical forces. To keep the whole application as simple as possible we did not use any server structure or database to save and load patterns. Instead, we serialize the graph structure into a JSON text file, as it is already supported by the 3D graph library, and save it to the local storage of the browser for switching between dimensions without losing progress and enabling downloading it to the computer.

4.2 Implementation of the Graph Structure

Chapter 3 explained the general structure of the graph we use to represent crochet charts. In the following, we will have a closer look at the properties of nodes and edges in our implementation which enable us to keep track of certain values and are used to separate between the different stitch types.

4.2.1 Properties of Nodes

The nodes specify the type of stitch and keep track of the layer on which it was created. For easy traversal, we added a reference to the previous node, next node and a list of nodes where the current stitch inserts into. A boolean logs whether the node is an increasing stitch or a decreasing stitch. This is used for the visualization of the stitch symbols. Finally, we track if the node was used to start a layer. This is currently only used for visualizing the start of the row or round but later could be used for automated row or round creation or completion.

In the case that a new entry point or insertion hole is defined, we add a node to represent that point. In this case, another property saves the list of surrounding nodes. This is currently used for the calculation of the position of the node as it should always be located in the middle of all surrounding nodes.

The following lists all properties of a node:

type (*string*)
Name of specific stitch type.

layer (*integer*)
Number of layer at which the node was created.

start (*boolean*)
Specifies whether node is the first node of its layer.

previous (*node uuid*)
References previous node.

inserts (*array of node uuids*)
References all nodes into which this stitch inserts into.

next (*node uuid*)
References next node; That node, in turn, references the current one as previous node.

isIncrease (*boolean*)
Specifies whether this node represents an increase or decrease stitch.

surroundingNodes (*array of node uuids*)
References all nodes which were marked as surrounding nodes; Empty array if this node does not represent a hole.

uuid (*string*)
Uniquely identifies the node.

4.2.2 Properties of Edges

The edges track which nodes they connect. Additionally, we track whether the link is an insertion link or a slipstitch, which is a special type of stitch.

The following lists all properties of an edge:

source (*node uuid*)
References the node from which this edge originates.

target (*node uuid*)
References the node which this edge connects to.

inserts (*boolean*)
Specifies whether this edge represents an insertion connection.

slipstitch (*boolean*)
Specifies whether this edge represents the special stitch type 'slip stitch'.

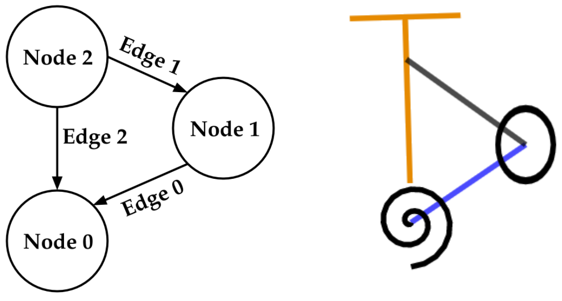
4.2.3 Graph Format for Storage

The library 'Force-Directed Graph' which we use for the graph visualization, can read JSON files as input. Therefore, we are using the same format to save and load the graph data. The expected format by the library is an object with a property 'graphData' which contains a list of edges, named as 'links', and nodes. Further data, which we use to reconstruct the history and which marks the latest status of the pattern, is saved in the same file.

Listing 4.1: Condensed graph serialization in JSON Format

```
{
   ”graphData”:{
      ”nodes”:[
         {  ”type”:”mr”,
            ”layer”:0,
            ”index”:0, ...},
         {  ”type”:”ch”,
            ”layer”:1,
            ”index”:1, ...},
         {  ”type”: ”hdc”,
          ”layer”: 1,
          ”index”: 2, ...}
      ],
      ”links”:[
         {  ”source”:”1”,
            ”target”:”0”,
            ”inserts”:false,
            ”slipstitch”:false,
            ”index”: 0},
         {  ”source”: ”2”,
          ”target”: ”1”,
          ”inserts”: false,
          ”slipstitch”: false,
          ”index”: 1},
         {  ”source”: ”2”,
          ”target”: ”0”,
          ”inserts”: true,
          ”slipstitch”: false,
          ”index”: 2}
      ]
   },
   ”currentNode”:{ ... },
   ”numLayers”:1,
   ”history”:{ ... }
}
```

For better readability and overview, in Listing 4.1 a reduced version of an exported JSON file of a graph is shown. The represented pattern contains three nodes: a magic ring, a chain stitch and a half double crochet. The first two are connected with a single edge. The half double crochet has an edge to the chain stitch and an 'insert' edge towards the magic ring. Figure 4.1 (a) shows the schematic visualization of the graph from the serialized JSON and Figure 4.1 (b) presents the actual output that this pattern shows in our prototype.



(a) (b)

Figure 4.1: Comparison of the plain graph structure to the visualization as Crochet Chart: (a) Nodes are connected with Edges (or Links) as listed in the serialized JSON format (b) The magic ring and chain stitch symbol are drawn on top of their respective nodes, the half double crochet is drawn across its insert edge to visualize where it inserts into

4.3 Visual Projection of Digital Patterns

In Chapter 3, we gave an overview of different possibilities to visualize digitized crochet patterns. In our prototype, we visualize patterns as crochet charts. We use the known stitch symbols to represent stitches visually and hide the underlying graph structure.

As presented and explained in Chapter 2, an international standard[13] for drawing patterns as crochet charts does exist. We use the same **standard symbols** to represent stitches in our prototype to take advantage of already defined and widely accepted symbols. The importance of the usage of such images is demonstrated in Figure 4.2. The images and their rotation, shown in Figure 4.2 (a) allow the reader to gather information about stitch types, methods, connectivity and shape at one glance, while the other visualization, shown in Figure 4.2 (b), only represents connectivity and shape.

Current crochet charts are only used to represent two dimensional patterns. Our digital crochet pattern representation supports arbitrary patterns, also of three-

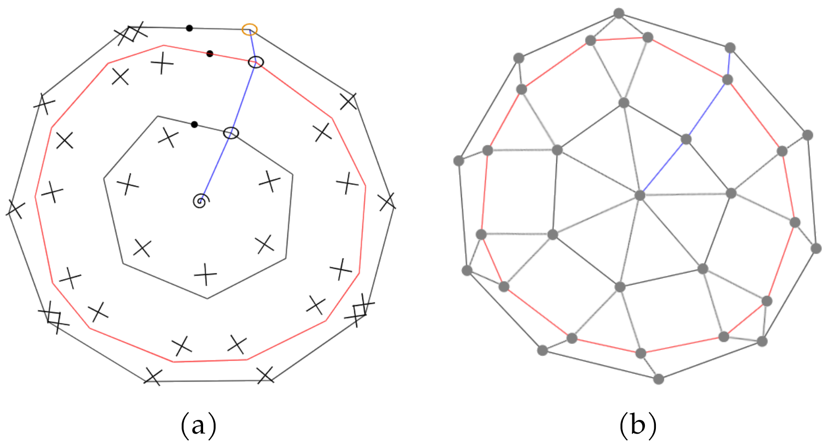


Figure 4.2: The Effect of Stitch Symbols: (a) Compared to the Plain Graph Visualization (b). In both, shape and connections can be seen but only the symbols additionally inform about stitch types, methods and insertions

dimensional shapes. Therefore, we needed to find a way to visualize the known **stitch symbols in a 3D environment**. We saw two possible options: using the known flat images or designing new appropriate three-dimensional objects to replace each symbol. To keep the visualization familiar to designers and for simplicity, we chose to keep 2D symbols for representation of stitches in 3D.

4.4 User Interactions and Edit Functions

According to the feedback we gathered from professional crochet designers, current crochet chart software is hard to use or does not support most crochet techniques. Also, the programs still require users to manually place and rotate stitch symbols. From the interviews it was clear, that the optimal crochet pattern application should be visual, without any written commands or entering codes. Therefore, we decided to enable interaction directly on the graph, visualized as crochet chart. Graphs typically allow various operations such as selecting nodes, removing nodes and edges and adding new ones. Our prototype is limited in its amount of possible operations. We had to build the editor from scratch as there were no frameworks available to assist the creation of domain specific graph editors to support crochet methods. In the user interface we offer the most basic interactions which are necessary for this proof of concept prototype, such as adding connections and undo and redo. Technically speaking, any operations that can be made on text could be directly done on our graph as any pattern is currently serialized to a JSON text file, but this is not a useful interface for crochet designers.

Our interface is based on **point and click** interaction. While a stitch type is selected, new stitches are added by clicking the desired insert stitch in the visual crochet chart representation. The view can be moved or rotated by right- or left-clicking the background and dragging. The rotation interaction only exists while the application is in the 3D viewer mode.

The user is given the possibility to create and **edit the graph** with the editor. For simplicity, the limitations are very similar to actual limitations when crocheting. Designers can add stitches to the last stitch, remove the last stitch and add another one. If any mistake has been made at the beginning of the design everything else after that has to be undone to reach the erroneous position and correct it. As presented in Chapter 3, further, more generic edit functions are desired, to allow modifying small parts of a pattern without removing and reconstructing it. Due to the fact that we are not using a generic graph editor to be able to offer domain specific functions, adding the support for other common methods was out of the scope for this prototype and also irrelevant for the demonstration of our concept. For faster creation of a pattern, we offer a basic version of an auto-completion function. It allows to copy the last X actions and redoes these actions starting from the next available stitch.

We also allow selecting stitches for the specification of a new insertion point. This way, holes in the fabric can be specified as nodes which can be used as connection point for new stitches. This is an addition to the known crochet charts, where such insertion positions did not have any representation. To specify the insertion into a hole, the designer would currently mention it in a text alongside a crochet chart image. As presented in Chapter 3, there exist many more possible concepts that could be applied to further enhance the editing experience.

Listing 4.2: Calculating rotation angle for the stitch symbols on the insertion edge

```
/* angle between link and y-axis
   it calculates the smaller angle between the lines */
let angle = new Vector(0, 1, 0).unitAngleTo(linkVec);
if(link.inserts){
  [...]
    /* translates context to the center of the link before
       rotation enabling rotation around the center of the image */
    ctx.translate(centerX, centerY);
    if(linkVector.x < 0){
        ctx.rotate(Math.PI + angle);
    } else{
        /* the big angle should have been used, alternatively
           we rotate counterclockwise using the small angle. */
        ctx.rotate(Math.PI - angle);
    }
    // flips stitch symbol if current stitch is a decrease
    if(!link.source.isIncrease){
        ctx.rotate(Math.PI);
    }
    ctx.translate(-x, -y);
    stitchCanvas.draw(link.source.type, ctx, x, y, color);
    [...]
}
```

4.5 Implementation Challenges

One challenge was to design the graph structure, another is to enable users to work with it. In the following, we present several tasks which we tackled in order to provide the previously listed functionalities to the editor. For visualization, we needed to ensure, that the images are correctly rotated. By tracking the issued commands, we can offer undo and redo functionalities. As a demonstration of possibilities for automatic support, we built a simple function for repeating a pattern of stitches.

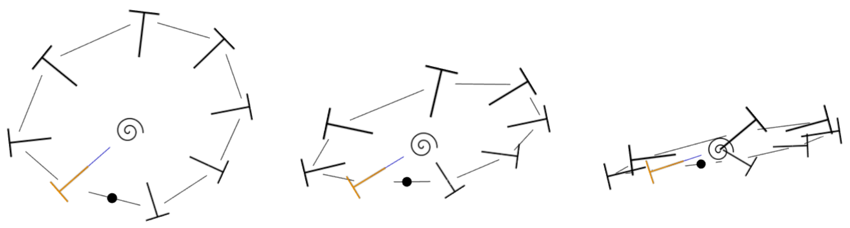


Figure 4.3: Stitch symbols face the user despite different view angles

4.5.1 Orienting Stitches Towards their Insertion Point

Earlier, we discussed that we want to include existing stitch symbols as visualization of stitches in our graph. Just like in the internationally known crochet charts, it is not only relevant where these symbols are positioned but also how they are oriented. The top of the symbol indicates the rough place of insertion for another stitch. The bottom points towards the position where the current stitch is inserted into. For our application, we built the symbols ourselves in SVG format for the 3D view and for the 2D view in a similar way using HTML5 Canvas Paths. But drawing these flat images in a 3D environment leads to the fact that from certain angles the images cannot be seen at all because we look at them from the side. Since the image has no depth, there is nothing to display. Three.js, the basis of the 3D force graph visualization library we used, supports sprites which are 2D images which can be used in 3D scenes but always face the camera. This way, no matter from what angle we look at the graph, the images always face the user. The stitch orientation, which is relevant to crochet charts, can still be maintained by rotating the sprite so that the images are indicating the direction of an 'insert' edge. Listing 4.2 shows how we rotate the paths for the 2D view. The process is very similar for the 3D view as there, the images are also only shown on a 2D plane, always facing the camera, thanks to the concept of sprites. The main challenge was to calculate the correct angle. Two crossing, non perpendicular, lines have two different angles between them. Typically, the smaller

Listing 4.3: Command pattern to track the command history to enable undo and redo functionality

```
--- graphMixin.js ---
addChain(previousNode){
    /* calls the execute function of the
       newly added AddChain Command */
    let actions = commandTracker.execute(new CommandAddChain(
        previousNode, this.graphLayers)
    );
    // applies changes to graph structure
    this.handleAction(actions);
}
--- Command.js ---
function Command(normalAction, undoAction, parameterObject) {
    this.execute = normalAction;
    this.undo = undoAction;
    this.values = parameterObject;
}
// this specific command inherits from the generic Command
export function CommandAddChain(previousNode, layer) {
    Command.call(this,
                 // action called on redo
                 addChain,
                 // action called on undo
                 removeChain,
                 // relevant data for replaying and undoing actions
                 {previousNode, layer}
    );
}
CommandAddChain.prototype = Object.create(Command.prototype);

--- Actions.js ---
export function addChain() {
    let node = new Node("ch", this.values.layer, false,
                        this.values.previousNode.uuid,
                        null, null, true, this.uuid);
    this.uuid = node.uuid;
    let link = new Link(node.uuid, this.values.previousNode.uuid);
    this.values.previousNode.next = this.uuid;
    /* indicates which information has to be
       updated on the graph structure */
    return {
        currentNode: node,
        newNodes: [node],
        newLinks: [link],
        updateNodes: [this.values.previousNode]
    }
}
```

angle is used for calculations which we could use if we only had to align the stitch symbol with the link line. But additionally, we needed the symbol to be oriented in a specific direction. Therefore, we needed to separate the two cases and distinguish when to use the larger or smaller angle to rotate the sprite. In Figure 4.3 the result can be observed by viewing the same pattern from different angles.

4.5.2 Keeping Track of History to Allow Correcting Mistakes

In order to offer the undo and redo functionality we keep track of actions done by the user. Possible options to solve this, include either keeping a list of commands issued by the user through edit functionality or saving the state of the program before a new command is run. The first approach can be solved using the *Command Pattern*[16] and the latter by using the *Memento Pattern*[16]. For our use case, we chose the command pattern, as each action is very minimal and the state barely changes. Using the memento pattern would result in large space consumption and redundancy. Listing 4.3 shows excerpts from multiple files which orchestrate the command pattern. The first snippet shows the usage, where a new command is created and a commandTracker keeps track of it. In the second snipped it can be observed how we defined the commands which can be tracked. Each command is inheriting from a generic Command object that saves actions which are run when executing, undoing or redoing the command. Additionally, we keep track of relevant data needed for the actions. Lastly, an example action is shown which adds a chain stitch to the graph. Since we do not have access to the graph structure itself from within the commands, we return actions which are run on the graph data to actually update the structure as seen in the first snippet, in the 'handleAction' function.

Listing 4.4: Traversal of the graph for finding the pattern to repeat for the auto-completion function

```
handleAutoComplete(numStitches, numRepetitions) {
  // find the stitches to repeat:
  let stitchesToRepeat = this.getPreviousStitches(
                                this.currentNode,
                                numStitches);
  /* order stitches so that we know relevant info and
     know how many stitches go into the same stitch */
  let orderedStitches = this.orderStitches(stitchesToRepeat);

  // determine in which direction the user crocheted
  let isForwardDirection = this.determineDirection();
  let nextStitch = this.getNextStitchToInsert(this.currentNode,
                                            isForwardDirection);
  for(let i = 0; i < numRepetitions; i++) {
    // repeating all stitches in order
    for(let k = 0; k < orderedStitches.length; k++) {
      orderedStitches[k].forEach(stitch => {
        let actions;
        if(stitch.isIncrease){
            actions = commandTracker.execute(
                new CommandAddStitch(
                    this.currentNode,
                    nextStitch, stitch.type,
                    this.graphLayers
                )
            );
        } else {
            actions = commandTracker.execute(
                new CommandAddDecreasingStitch(
                    this.currentNode,
                    nextStitch
                )
            );
        }
        this.handleAction(actions);
      });
      nextStitch = this.getNextStitchToInsert(
                            this.currentNode,
                            isForwardDirection
                        );
    }
  }
}
```

4.5.3 Repeating a Pattern of Stitches Automatically

Chapter 3 explained the graph structure we designed for representing patterns and we also advertised that it could be used to traverse the pattern for adding smart domain specific functionalities. We implemented a simple version of an auto complete function to showcase this.

The function allows to repeat a set of previously added stitches for a specified amount of times. The user inputs how many of the previous stitches should be repeated. The responsible function receives this value in the variable 'numStitches'. Also, the amount of repetitions of that set of stitches is given by 'numRepetitions'. For repeating that set of stitches the specified amount of times, we do the following steps, also shown in Listing 4.4:

1. Retrieve the list of stitches to be repeated: Traverses the graph backwards by 'numStitches' steps.
2. Order the retrieved stitches: Reverses the list of stitches and organizes it by insertion points.
3. Determine crochet direction: Analyzes the last few stitches to determine if they were crocheted in the same or opposite direction as the previous row.
4. Find the first stitch where the repetitions should start; makes use of the determined direction.
5. Repeat 'numRepetitions' times: add the stitches to the graph according to the ordered list, starting at the calculated start point and moving into the determined direction.

4.6 Summary

As a proof-of-concept for the digitization of crochet patterns, we built a prototype which uses the graph structure we designed as underlying representation of crochet charts. The graph is visualized and can thus be used for editing like a visual domain specific language. Our editor allows basic functions such as adding stitches to create a pattern and undo and redo for editing it, which imitates the real crochet process. To show the potential of our concept, we demonstrated the visualization of arbitrary patterns in a 3D environment using the already known stitch symbols to draw a three-dimensional crochet chart. Additionally, we implemented a basic auto-completion function which demonstrates the graph traversal and shows how domain knowledge can be used to repeat a given pattern.

5 Concept Validation: Comparison of the Current Workflow to the Proposed Solution

In this report, we presented a concept for digital creation of crochet instructions for 2D and 3D patterns, as explained in Chapter 3. As a proof of concept, we built a prototype which allows the creation of such digital instructions. It uses our graph structure design to represent crochet patterns and offers an interface to allow fundamental interaction with the visual domain specific language. Details about implementation can be found in Chapter 4. The developed concept and prototype aim to tackle the issues mentioned in Chapter 2. In this chapter, we will present an evaluation of the concept on the basis of the prototype, regarding its feasibility and usability. First, we will establish the properties of the current pattern writing process as baseline which we can compare our evaluation findings to. Second, we present examples which show the flexibility of the graph structure to represent arbitrary patterns. In the last part, we involved professional crochet pattern designers in a small qualitative study. The six designers were each given the same tasks within our prototype and were able to give further feedback and comments.

5.1 Manually Drawing a Pattern Chart

The workflow to create crochet pattern instructions is currently very manual. Even the programs which exist to support chart drawing are mainly based on a manual workflow. As a baseline, to compare the new workflow of our prototype to, we use the pen and paper based process of drawing charts. Otherwise, all the different programs would have to be considered separately with their individual learning curve and range of functionalities. For more details about the current pattern creation process refer to Chapter 2.

5.1.1 Manual Pattern Writing Process

The current pen and paper based process for writing crochet charts is to draw the symbols manually on a piece of paper. Often squared paper is used, taking advantage of the grid lines for even spacing. When written instructions are created, the instructions are written per row or round. First the number of the row or round is indicated, followed by instructions in the format of a number and the name of the stitch. And it is typical to find the number of total stitches at the end of the instructions for a row or round. One line could look like this:

$$\text{Row 15: } 4(1\text{sc } 3\text{hdc } 1\text{sc}) \Rightarrow 20$$

This would describe how to crochet row 15 of a pattern: A set of instructions has to be repeated 4 times. That set of instructions being to crochet first one single crochet, then three half double crochets, and finally again one single crochet, each into one stitch of the previous row. At the end 20 stitches were made in that row. While written instructions typically show structures like this, the exact format and abbreviations vary strongly depending on the author.

5.1.2 Drawing a Pattern Manually with Even Layout

To confirm the struggle of manually drawing a crochet chart, we performed a simple self test. We took a pattern in written instructions[1] producing a flat object, a heart tag, made out of 222 stitches. The task was to draw that pattern on a sheet of paper as a chart.

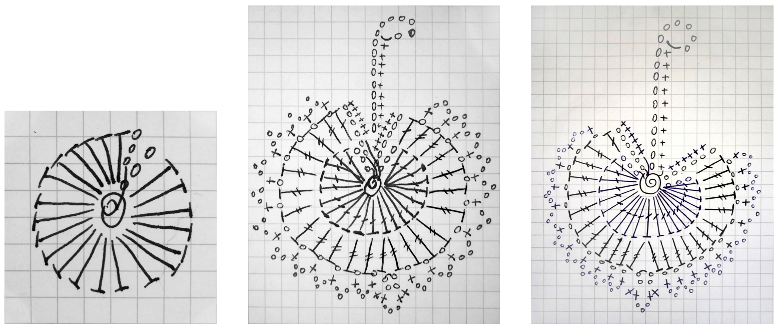
Listing 5.1: Written instructions for flat heart with a crocheted hanger

```
Start with a magicring.
1. Rd.: 3ch, 21dc, 3ch, slst into ring

2. Rd.: 6ch, 1dc, 1ch, 10(1tr, 1ch), into next stitch:
        (1tr, 4ch, slst into first of the four ch, 1tr),
        1ch, 10(1tr, 1ch), 1dc, 6ch, slst into ring

3. Rd.: 6sc around the 6ch from the previous round,
        sc into ch between dc and tr of the previous round,
        continue  10(3ch, 1sc), into the ring of 4ch of the
        previous round: (2sc, 3ch, 2sc), 1sc into ch
        between the next two tr, continue 10(3ch, 1sc),
        6sc around the 6ch of the previous round, slst into ring.

For the loop to hang the heart:
        16ch, slst into 6th last ch,
        10sc into the ch, slst into ring.
```

To draw the chart for the pattern, given by the written instructions in Listing 5.1, we needed three tries. Figure 5.1 shows the various intents. In Figure 5.1 (a) you can see that at a first try we were able to place all 21 stitches of the first round in a circle but there was no space for the start of the next rounds. In the second try, in Figure 5.1 (b), all stitches were fitted into the round but they were not evenly spaced.

[1]This pattern is a modified version from Karina Méri `https://youtu.be/NXFtWaqfsB0` (visited 2020-06-30).

Finally, in Figure 5.1 (c), the whole pattern was drawn fairly evenly and alternating rounds were differentiated through color. To achieve better clarity and uniformity, the drawing would have to be repeated again. This clearly shows that the manual process of drawing chart is very time consuming. This task did not even include any creative work of coming up with a new idea and pattern but it rather only depicts what had already been written down before.



(a) (b) (c)

Figure 5.1: Evaluation results of manual chart drawing. Three attempts were needed to reach an acceptable layout. More would be required to achieve symmetry and uniformity.

5.2 Showcase of an Example Pattern Using the Prototype

In the following, we show how the prototype can be used to create a simple 3D pattern of a bowl-like shape. This shows, the concept has been applied to a functioning editor system and can be used to create digital crochet instructions.

Figure 5.2 presents the interface of the prototype. Functionalities are separated into four spaces. In the middle, beneath all menus is the canvas which displays the visualized pattern while editing. On the top are the main file related actions which allow creating, opening and saving a pattern and also undo and redo functions. On the left are view related functions which can center the view, toggle the visibility of edge lines, hide previous rows and change the view from 2D to 3D. On the right are positioned all relevant actions needed to input data for a pattern. This includes a list of stitches which can be selected by click and several functions. There, the stitch mode can be toggled from increasing to decreasing. Rows or rounds can be auto completed and new insertion holes specified. And to keep track of the amount of layers, a counter can be updated every time a new row or round is started.

A new pattern can be started by selecting one of three techniques. Figure 5.3 (a) a magic ring, Figure 5.3 (b) a line of chain stitches or Figure 5.3 (c) a round of chain

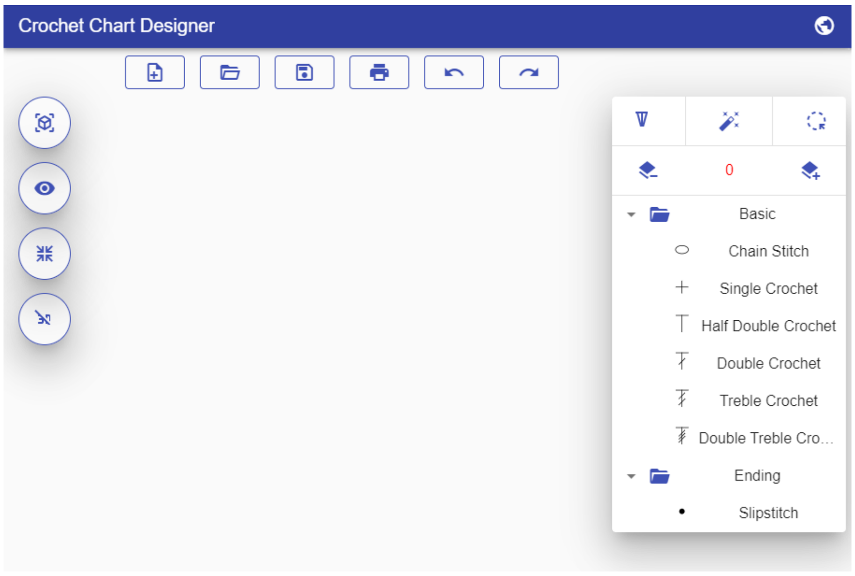


Figure 5.2: Application Interface

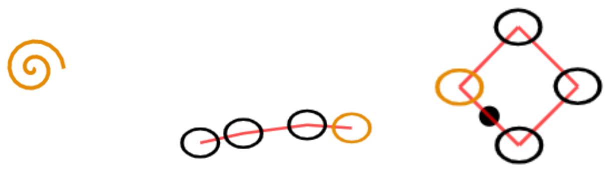


Figure 5.3: Visualization of the three starting methods: (a) magic ring, (b) line of chain stitches, (c) chain round

stitches. When choosing a line or round of chain stitches, the user can decide the amount of chain stitches to start with.

In the following, we will present an example workflow which starts with a magic ring and continues working in joined rounds. In our example, we first added a chain stitch and then additionally five single crochets by first selecting the type and then clicking the magic ring. The result is shown in Figure 5.4 (a). To close the round with a slip stitch, we change the selection to slip stitch and click on the chain stitch. This connects the current stitch, the single crochet marked in yellow with the first stitch of the round, the chain stitch. The outcome is shown in Figure 5.4 (b).

Note that now the slip stitch is not highlighted as current stitch, but the chain stitch is. This is because a slip stitch is a stitch type which merely connects two points. Therefore, it is displayed in the middle of the connecting edge. We just finished a round, therefore we raise the layer counter by one. In the next round, we use another type of stitch called half double crochet which is taller than single crochet stitches.

A typical way to grow the diameter of crochet fabric is to use the increase method. Here, we want to double each stitch with each two double crochets. To increase a

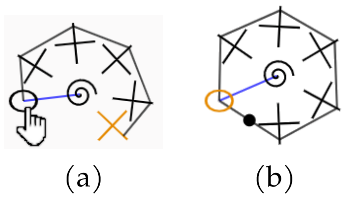


Figure 5.4: Visualization of Before (a) and After Closing (b) the First Round. The connecting slip stitch is added through click on the chain stitch.

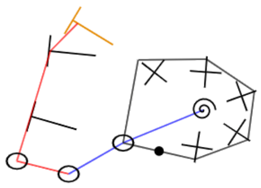


Figure 5.5: Visualization of an Increase Using Half Double Crochets

stitch from the previous row, we can click it multiple times which will insert the according amount of new stitches. In our case, we add two double crochets into a single crochet, shown in Figure 5.5. This pattern now needs to be repeated across the whole round into each of the single crochets of the first round. Instead of manually inserting each and every stitch of this repetitive pattern, we can use the auto complete function which allows repetition of the last X stitches Y times. For this pattern, we want to repeat the two increasing half double crochets into the remaining four single crochets.

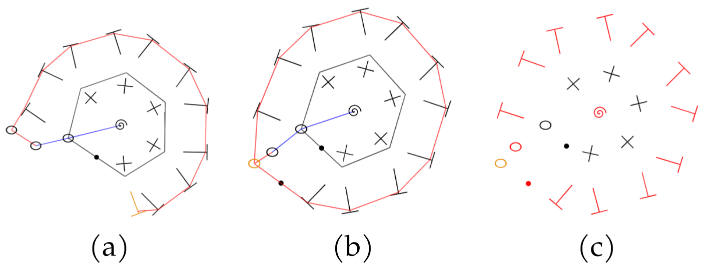


Figure 5.6: Result of the Auto-completion Function Shown with and without Edges. (a) State after the auto-completion is completed (b) Round is closed with a slip stitch, edges are still shown (c) Edges are hidden to resemble the result of current crochet charts

Figure 5.6 (a) shows the outcome of the auto complete function. Then, the round can be closed again by using a slip stitch, seen in Figure 5.6 (b). After finishing these two rounds, we can observe the color scheme used in the visualization. The first round is colored in black as it is an odd layer number and the second round is red. Blue edges mark the beginning of a layer. Using the toggle button with an eye symbol on the left-hand side of the interface, we can hide the edges of the graph which leads to a view more similar to the internationally known crochet charts, see Figure 5.6 (c). In the second round, we enlarged the diameter of the round using the increase technique. This time, we want to reduce it by using the decrease technique. For that, two stitches, this time double crochets, are crocheted into one stitch. First, a normal increasing stitch is placed into the first stitch and then, either using right click, or switching the increase-decrease mode and clicking left on the next stitch results in a decrease of those two stitches into one, shown in Figure 5.7. To finish this final round, we repeat this pattern until the end and finish again with a slip stitch. Through first increasing and later decreasing, we built a three-dimensional shape, like a bowl. Thanks to the three-dimensional environment, we can view the resulting pattern chart from various angles, as shown in Figure 5.8.

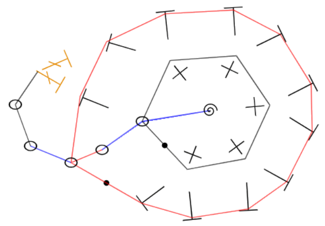


Figure 5.7: Visualization of a decrease using double crochets

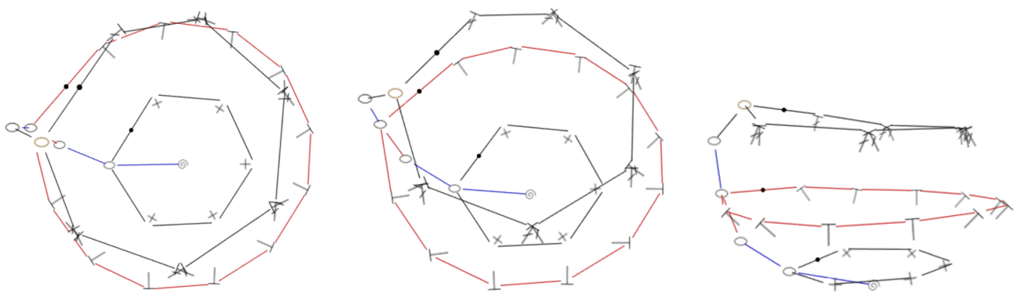


Figure 5.8: 3D view of a pattern from three angles

Creating Custom Insertion Points

If we had not started with a magic ring, we could also have used a round of chains, shown earlier in Figure 5.3. To continue from there, you can crochet into any of the chain stitches. But in crochet practice this starting technique is usually continued by crocheting into the hole which is formed by the ring of chain stitches instead.

To realize the same technique, in the prototype a new stitch insertion point can be defined by selecting the surrounding nodes or stitches.

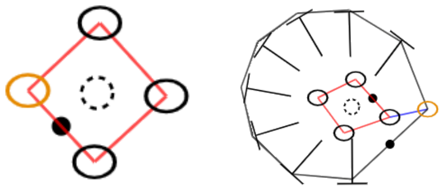


Figure 5.9: Visualization of Additional Insertion Points. (a) A hole was added in between all chain stitches by selecting them (b) In the next round half double crochets use that hole as insertion point

After confirming the selection a new node is displayed as a dotted circle in the middle of the selected nodes, as shown in Figure 5.9 (a). This new node represents the new insertion point and can be used just like any other nodes that represent stitches. An example where multiple half double crochet stitches are worked into the hole is shown in Figure 5.9 (b).

5.3 Patterns Demonstrating Usage of Crochet Methods

The previous walk-through has shown that generally creating digital patterns is possible using our concept and resulting prototype. In the following, we present further example patterns created with the same software to demonstrate the variety of patterns that can be represented. Alongside each chart we also display the actual crocheted output of the same pattern. This shows, that the created patterns are not only valid and can be crocheted but also allow a comparison between the layout of the pattern in the system and the shape of the actual crocheted output.

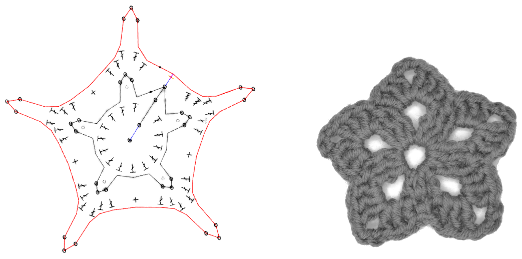
The first four examples show the variety of shapes that can be created. Table 5.1 example 1 shows the star that will later also be the basis to the user testing with crochet designers. This star, the heart in Table 5.1 example 2 and the oval in Table 5.1 example 3 are all built using the round method, explained in Chapter 2. The triangular shawl in Table 5.1 example 4 uses rows instead. All the patterns are displayed with an even but wide spread layout. The similarity to the actual crocheted output is not always exactly the same but the general shape is perceptible.

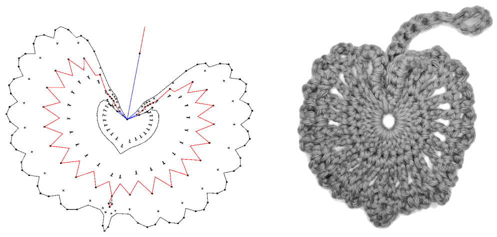
The other examples all show 3D patterns. The ball in Table 5.1 example 5 uses the spiraling round method. Table 5.1 example 6 used the oval shape as a basis and

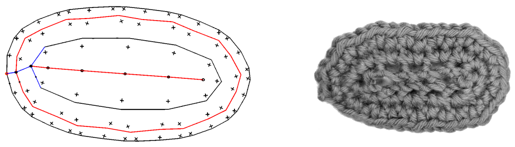
[2]Adaptation of Triangular Shawl 'Fareth' © Morben Design / Jasmin Räsänen.

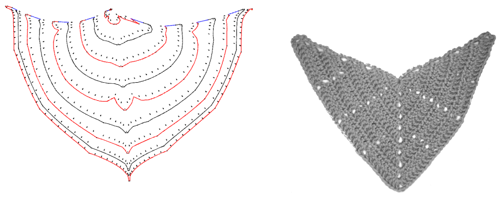
[3]Pattern from myboshi: `https://www.myboshi.net/4seasons/aomori/` (visited 2020-29-06).

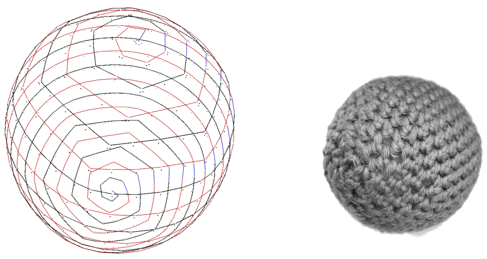
Table 5.1: Seven example patterns demonstrating the use of the different crochet methods

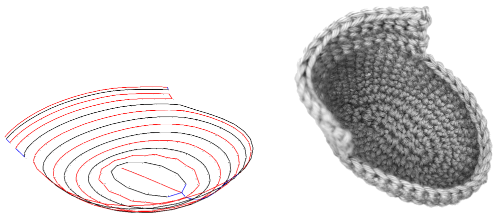

Example 1: Flat Star

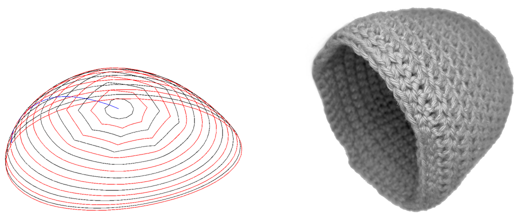

Example 2: Flat Heart Hanger


Example 3: Flat Oval Shape


Example 4: Triangular Shawl[2]


Example 5: Ball


Example 6: Moses Basket


Example 7: myboshi Hat 'Aomori'[3]

after continuing with joined rounds switched to a row based method to only rise half of the basket further. Finally, Table 5.1 example 7 shows a complete hat based on the instructions of the myBoshi hat called ‘Aomori’. The shapes of the digitized 3D patterns are similar to the actual output. Only the hat from Table 5.1 example 7 and the basket from Table 5.1 example 7 are more shallow than the actual crochet result but the basic shape remains recognizable.

5.4 Feedback from Professional Crochet Designers

In order to get feedback from the actual target user group, we conducted a series of six qualitative interviews in German with professional crochet designers, one male and five female. All of the participants had previously participated also in the survey presented in Chapter 2. To make sure every feature is discussed we provided a set of tasks to each participant and asked them to complete the tasks while providing their thoughts on the process. All interviews were conducted via online videoconferencing. The participants got the access link to the software and shared their screen while working on the tasks.

The interviews were structured by three main tasks. First, we presented an example of a crochet chart and asked to go through the whole pattern in their head and identify flaws or strengths of this chart. Second, the participants were asked to reproduce that pattern from the first task with the prototype using the 2D view. Third, we asked to design a bowl or ball using the prototype and the 3D view. Before giving access to the software, we first presented an introductory video[4] showing main interactions with the prototype.

5.4.1 Evaluation of an Existing Chart

For the first task we showed the image of a crochet chart of a flat star pattern, shown in Figure 5.10. This pattern was specifically chosen to be small, to keep the time minimal when discussing it, and to have ambiguous parts in it. We did not modify the pattern, it is a real example of a chart freely available on the internet. Our assumptions were, that the chart seems very clear and well drawn but has three weak points which are imprecisely notated and give unclear instructions for reproducing the design.

Figure 5.11 (a) highlights the last stitch, a double crochet, of the first round, which we believe is missing an indication of how to close the round.

Figure 5.11 (b) marks the six double crochets which are part of each point of the star. In the drawing each double crochet symbol is not evenly positioned and rotated which shows again the difficulty of manually creating a layout for pattern charts. Because of no clear aim of each of the symbols, we believe it is not clear where to insert them.

[4]`https://www.youtube.com/watch?v=GTiUSk4bKU4` (visited 2020-06-30).

Figure 5.11 (c) highlights the single crochet at the end or beginning of the second round respectively. Similarly, to the end of the first round, it seems unclear how to finish the second one.

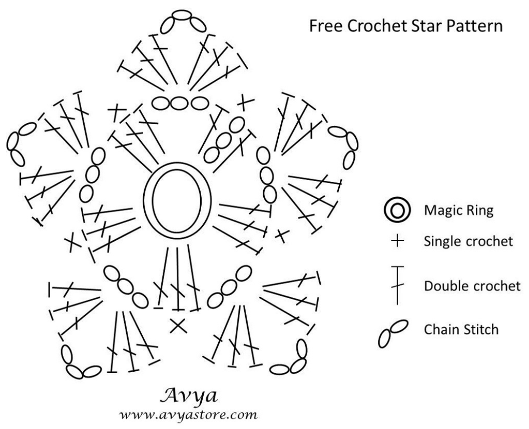


Figure 5.10: Crochet chart used as example for evaluation through the designers

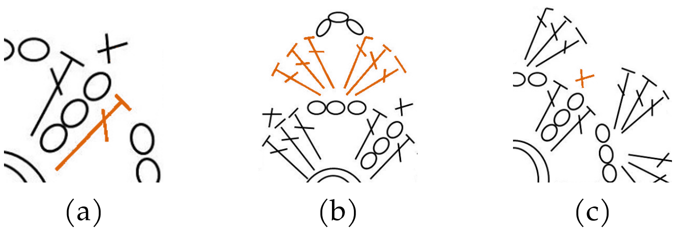


Figure 5.11: Three extracts from the chart showing expected problem zones. (a) Unspecified closing method for the first round (b) Insertion point missing for the six double crochets on each point of the star (c) Unspecified closing method for the last round

In order to validate our hypotheses, we prepared two tasks. First, we asked whether the chart seemed obvious to the participant and then, the designers were tasked to orally follow the chart stitch by stitch, comment on any encountered flaws and suggest solutions.

As to the first task, all participants immediately told us that the given crochet chart is very well arranged. Only two participants mentioned a missing slip stitch to finish the last round and another one would have wanted an indicator for the beginning of each round. After going through all the instructions several issues were identified by the participants. The main issues where how to close the first and the second round and where to crochet the 6 double crochet stitches into of each point of the star. For closing the first round the participants came up with two options.

A slip stitch could connect to the third chain stitch for joined rounds or the single crochet of the next round could be worked into the third chain stitch for a spiral round. Two participants preferred using spiral rounds and the others joined rounds. Most participants immediately decided to crochet the six double crochets of the star points into the hole under the chain stitches as they feel this to be easier and the double crochet symbols do not point at any specific chain stitch. Only one participant wanted to crochet into specific chain stitches but was not sure which double crochet should go best into which chain stitch. For the end of the second and final round of the pattern in the chart, the participants were all missing a slip stitch to close the round. Two also suggested that instead, the designer might have wanted the ends to be sewn together.

In this task, we were able to observe how professional crochet designers initially were thinking that the crochet chart was very clear and were able to identify the expected issues only at a closer look. This shows clearly that even small patterns such as this star, already can be flawed even though the stitches look well arranged and lead to believe that everything is correct.

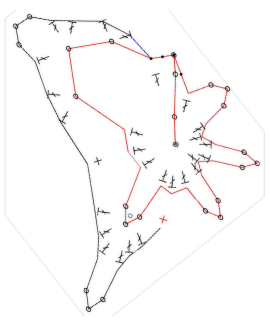
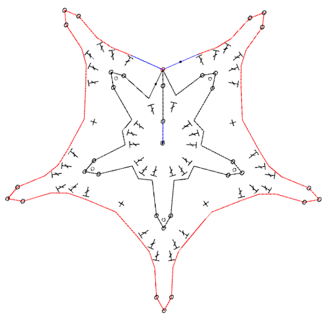
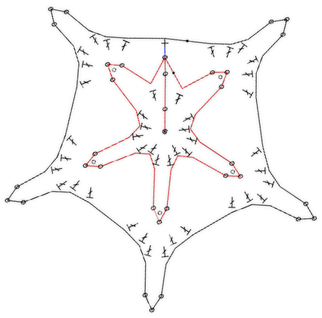
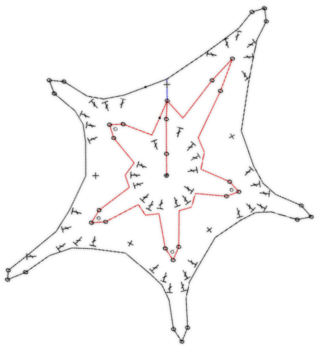
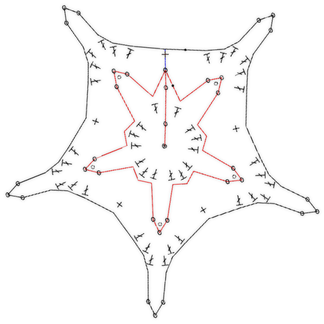
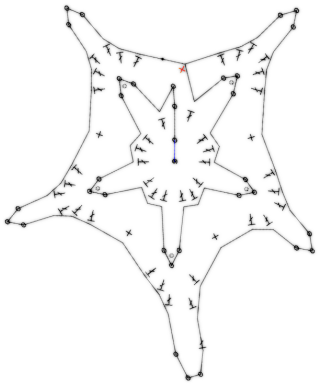
5.4.2 Building a 2D Pattern with the Prototype

Before starting the second task we showed the introduction video mentioned earlier. Then we gave access to the prototype by providing the participants with a link to open on their own computer on a browser. For this task, we asked them to recreate the previously discussed chart (Figure 5.10) using the prototype, its 2D view and the function to add a new insertion point. Since several aspects of the chart were not clear according to the designers after the first task, we let them choose how to handle the points in question when reproducing it.

Thanks to the introduction video, the participants were already familiar with the general interactions within the editor. Nevertheless, three participants struggled with how to add stitches. Instead of selecting a stitch type and clicking where they wanted to insert a stitch, they tried drag and drop or double clicking the stitch type first. After all of the participants understood the way to add stitches, they commented that they found it logical and easy. When one designer realized that stitches have to be added in order, just like in real crocheting, the designer mentioned that usually, when using a drawing program the stitches are not added in order. According to that participant, it is nice to be able to add the stitches in order, but in a drawing program it was easier to build an even design when first starting by placing some stitches to mark the shape. But through the automatic layout, this strategy is no longer needed in the prototype.

After adding several stitches, four designer enjoyed that there was an automatic layout. According to them it helped speed up the process and eliminated the fiddly manual layout process. However, almost all mentioned that the distances between the stitches were too big, especially between one specific stitch type, the chain stitch. This resulted to them in unexpected layouts and not completely even and symmetric rounds. The other reason why the layout is confusing, according to four participants, is that the whole design rotates while a new stitch is added which leads to different

Table 5.2: Resulting charts of the 2D patterns created by the participants

		
Participant A	Participant B	Participant C
		
Participant D	Participant E	Participant F

positions of the stitches each time. This makes it almost impossible to add multiple stitches fast after another. Other four designers had the issue that the layout of the stitches resulted in a clockwise stitch order. Any new stitch would appear to the right of the previous stitch but in crochet they work counter-clockwise. After being told that it is possible to drag the stitches to make them go into the right direction they were satisfied with the layout again.

During this task, the designers were asked to use the function of adding a new insertion hole to crochet the double crochets of the star points into the space under the three chain stitches of the first row. All of the designers liked the existence of the feature, as this adds the possibility to specify something that previously was not representable in charts.

In Table 5.2, you can see the results of the participants recreating the example crochet star design using the prototype. Each result is marked with a letter from A to F which indicates the designer who created it. All designers, except participant A, were able to reproduce the whole design successfully. All the designs vary a bit as the designers were free to choose how to handle the three ambiguous instruction parts which were identified during the first task. Participant A did not finish the star due to the confusion from the movement of the whole design when adding new stitches. Nevertheless, each of the tested features were used and therefore, we were still able to get feedback from this participant.

5.4.3 Testing the 3D Environment for Pattern Creation

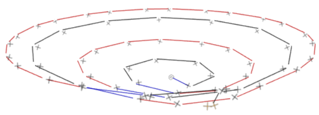
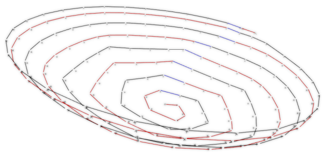
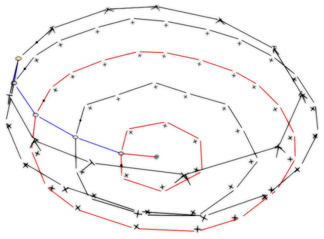
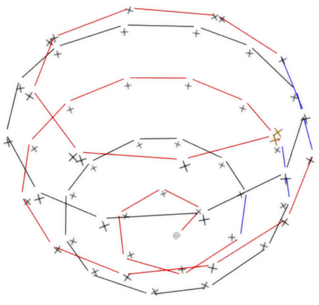
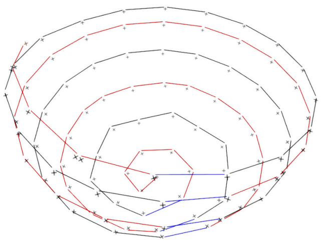
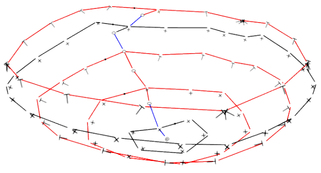
For the final task, the participants were asked to switch from the 2D view to the 3D view in the prototype. There, the task was to build a new pattern of a bowl shaped object using the increasing and decreasing methods and testing the auto complete function. The specifics of the pattern, such as which stitch types and amounts to use were left completely open to the designers.

Already at the first glance, after switching to the 3D view, five designers were amazed at the possibilities they imagined that visualizing a pattern in 3D could allow. Many mentioned the advantages for amigurumi patterns (3D figures) where the real shape can be previewed and adapted before even starting to crochet. One designer was thinking specifically about being able to compare proportions of multiple parts to one another.

During this task, the designers were asked to use the auto complete functionality. Two designers initially had some trouble trying to understand what numbers to enter for the auto-completion dialog. But after the successful auto-completion of at least one round all designers were positive about this feature and said that this can save a lot of time compared to crocheting slowly stitch by stitch.

After digitally crocheting several rounds, the designers were asked to use the decrease function. Three participants were confused by the increase and decrease mode switch. One was suggesting that the stitches to decrease should be selectable just like for the insertion point. After a successful decrease operation three of the designers liked the method and visualization of a decreasing stitch. One other

Table 5.3: Resulting charts of the 3D patterns created by the participants

 Participant A	 Participant B	 Participant C
 Participant D	 Participant E	 Participant F

designer mentioned to normally skip a stitch instead of decreasing by crocheting multiple stitches together as one.

With a growing pattern in the editor many designers had issues with finding a good view of the current workspace to continue to click the right stitches. The navigation in the 3D space is new to all of them. This led to the fact that most designers got confused about where to place the next stitch as the design grew. The designers seemed to have lost the overview quickly about what they had already done and what else they wanted to do still in this and the following rounds.

Overall, some designers mentioned they were missing a display of what they had done so far, such as the count of stitches per layer and a clear marking of the layer numbers. Furthermore, they tended to forget to update the layer number and one designer suggested that the program could try to remind the user of the layer change. Additionally, the designers were asking to build their own repeatable patterns, a set of stitches which can be chosen similar to the stitch types and added by a single click.

In Table 5.3, you can see the results of the participants building a 3D pattern for a bowl shape. As the designers were free to choose their preferred method and stitch types all designs vary from each other, but all resemble a bowl shape successfully. All participants were able to try the functions of auto-completion and decreasing stitches for their design.

5.5 Summary

In our report, we presented an approach to creating digital crochet instructions for 2D and 3D patterns of any shape. In this chapter, we initially demonstrate the difficulties that arise when faced with the task to manually draw and layout a crochet chart. Furthermore, we evaluate a prototype which we built as a proof of concept of our approach. We show the usage of the software by giving an example walk-through. Then, we presented a variety of further patterns which were created using the prototype, taking advantage of many different crochet methods. Finally, we present the feedback of professional crochet designers who tested our prototype by building two patterns with it themselves.

We assumed that drawing a chart manually with an even layout is complicated and time consuming, and that such manually created instruction charts can contain ambiguous parts which lead to confusion and incomplete instructions. Our self test of drawing such a chart and the interviewed professional crochet designers were able to confirm our assumptions. When testing the prototype, the designers had to clearly specify how to handle the ambiguous parts of a given pattern which therefore resulted in varying charts.

From the example patterns built using our prototype, we were able to observe that the generic layout algorithm we chose for the prototype results in similar shapes to the actual crochet output. Some designers were mentioning various benefits of such a shape preview, as it could, for example, show the effect that different methods have on the resulting form. On the other hand, the designers still saw room for improvement of the layout of the crochet chart, such as that some stitches are rendered with too large spaces between each other which impedes the readability and breaks any symmetry. Other general feedback was given regarding the user interaction where the usage of the decrease mode was often misunderstood or the confusion and inconvenience from the rotation of the rendered chart while adding stitches.

The designers enjoyed using the further functionalities such as auto-completion, as it could save time and defining and using holes as insertion points which is not supported by current charts. All in all, we see that the approach is applicable to the domain, and the main user group, professional crochet designers, reacted positively to the concept.

6 Related Work

This report presents the prototype of a projectional editor to create formal crochet patterns in the form of crochet charts. This chapter presents related concepts and approaches to this domain. We represent crochet charts internally using a graph structure but currently other representations are used such as text and images which will be presented here. Further, we compare existing methods to create visual representations. Lastly, we will give an overview of the limited research done on or using crochet.

6.1 Crochet Languages

Crochet instructions mainly make use of the human language. Most patterns are written as text or the steps are explained in a video. But when using natural language there exist multiple possible formulation for any instruction. Therefore, such instructions can be very ambiguous. Other barriers when using instruction text are the used language and the language specific notations that are employed. As an example, already American and British English use different notations for the same type of stitch. In the following, we present approaches which try to avoid the ambiguity and limits of natural language.

6.1.1 Crochet Charts

Crochet charts are a way to use visuals to convey the structure and instructions of a crochet pattern. Such a chart consists of schematic crochet symbols [13] which represent stitches. For each stitch type exists a specific symbol which is standardized and understood across the globe. Find an example chart in Figure 6.1. Such charts can display clearly which stitches are used for a pattern. They show which stitch is worked into which by rotating the stitch symbols to point to the inserted stitch. Exact entry points can be represented by additional symbols. Rounds or rows can be visually kept apart by alternating the color of stitches every row or round or by numbering them. All symbols are arranged to mirror the resulting shape of the pattern.

The specification of this diagramming system is a good approach to define clear instructions. While there exists a list of standardized crochet terms and symbols, there does not exist one specific workflow to create such charts. When all symbols are well arranged, the chart can represent a pattern very precisely. Due to the lack of a consistent workflow and tools for creation, most charts have flaws which lead to ambiguous images. Such an example can already be seen in the example chart

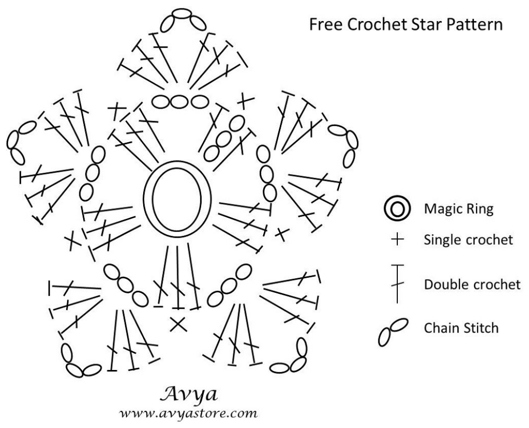


Figure 6.1: Example of a crochet chart for a flat star.[1]

Figure 6.1. As demonstrated in Chapter 5, this chart contains flaws that are not obvious at first glance.

6.1.2 Crochet Graphs

If a crochet pattern is creating a flat and rectangular shape it can be visualized using a grid, just like pixel art. Such a grid is known as a *Crochet Graph*. Each grid represents a set of stitches, which is repeated across the whole pattern and usually consists of only one or two stitch types. On the grid the individual cells can be filled with different colors indicating the yarn color that should be used to crochet that cell, an example is shown in Figure 6.2 (a). Alternatively, when only two colors are used, it can also indicate whether the cell should be crocheted using one set of stitches or another, see Figure 6.2 (b).

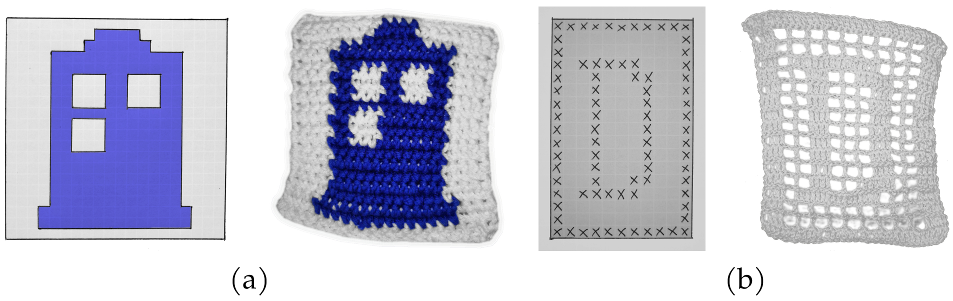


Figure 6.2: Examples of crochet graphs as pattern description and their results. (a) The color of the cells indicate which yarn color to use, all cells are worked using the same stitch type (b) Filled cells indicate to use a different stitch type than empty cells

[1]Source of the image: www.avyastore.com.

As already mentioned, this visualization technique is only used for flat, rectangular patterns with limited stitch types. This representation gives a clear overview of the final outcome and shows instructions clearly. Nevertheless, we do not use this approach for our concept, as it is limited in representable shapes and stitch types.

6.1.3 Crochet Coordinate System 'Berliner Häkelschrift'

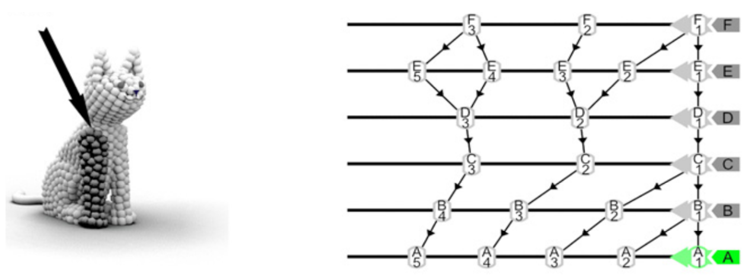


Figure 6.3: Example of Arnim Schachtschabel's 'Berliner Häkelschrift' which uses a coordinate system for visualization of the instructions

A completely different approach to specifying a pattern, in this case for 3D figures, is the so-called Berlin Crochet Language *Berliner Häkelschrift* developed by Arnim Schachtschabel[31]. Figure 6.3 shows an example where this system is used to represent the leg of a crocheted cat. It marks how stitches of 3D objects are connected in a 2D coordinate system. This system supports the representation of rows and rounds but is limited to using only one stitch type–single crochets. Rows are read alternating from right to left and rounds are always read from right to left. This mirrors the directions in which crocheting is done. The diagrams automatically are generated from 3D models. The main advantage of this system is that three-dimensional patterns can be represented in a two-dimensional visualization. Our approach does not use this type of visualization because it does not mirror the shape of the crocheted outcome and it is limited in the amount of representable stitch types.

6.2 Programs for Editing Visual Crochet Representations

Currently, designers have different options to create a visual representation of their pattern. Even though some programs exist to support the process, many designers prefer to draw crochet charts by hand on a sheet of paper to later include a photo of it in their instruction document. The reason for this can be explained by looking at the constraints of existing programs which we will present now. Table 6.1, at the end of this section, gives an overview of the discussed tools and programs and additionally lists the options of written instructions and hand drawn charts for comparison.

6.2.1 Crochet Stitch Fonts

In order to allow designers to use any image or text editors, stitch fonts have been designed, for example by hookinCrochet[22] and adriprints[2]. They supply a variety of symbols used for crochet charts. The users can then import the font into their favorite editor and arrange the symbols any way they desire. The advantage of such fonts is that the symbols are consistent across the whole chart. On the other hand, any possible stitch type combination typically uses its own font icon which leads to needlessly large icon lists. The way that such fonts are used, depends on the program that the designers use. Possible programs would be text editors, image editors, specific crochet programs and anything else that can read and import a font. When using common text or image editors the imported font symbols have to be arranged manually. Some programs might support alignment or even distribution but without domain knowledge the created charts cannot be validated, auto completed or corrected.

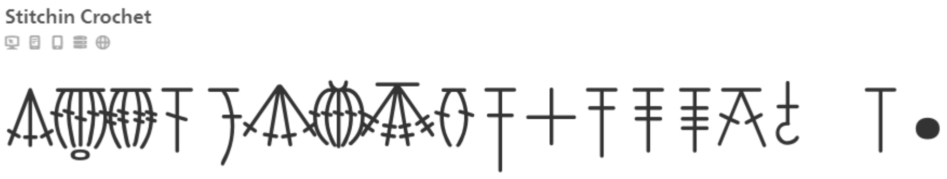


Figure 6.4: A subset of stitch symbols provided by stitchin font

6.2.2 StitchFiddle

The web based program *Stitch Fiddle*[9] allows the creation of crochet graphs as explained earlier. To create crochet graphs, the user can choose the size of the base grid and fill cells with colors, as shown in Figure 6.5 (a). Additionally, it allows creating free-form crochet charts where the user is supplied with a variety of crochet stitch symbols which can be arranged in any way on a blank canvas, see Figure 6.5 (b).

The process of creating crochet charts using Stitch Fiddle is just as manual as the usage of stitch font symbols in any other image editor without any guidelines to support their arrangement.

6.2.3 Crochet Graph Designer by HookinCrochet

Another software to create crochet graphs is *Crochet Graph Designer*[21], shown in Figure 6.6. More than creating generic squared cell grids, this software can differentiate between different stitch sets used per pattern for each cell. The stitch types effect the generated cell size and aspect ratio. A picture can be used as basis for the crochet graph and the software calculates how the image is split into a grid,

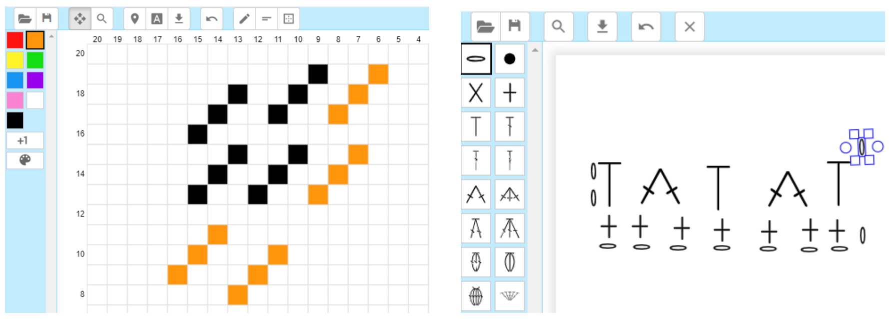
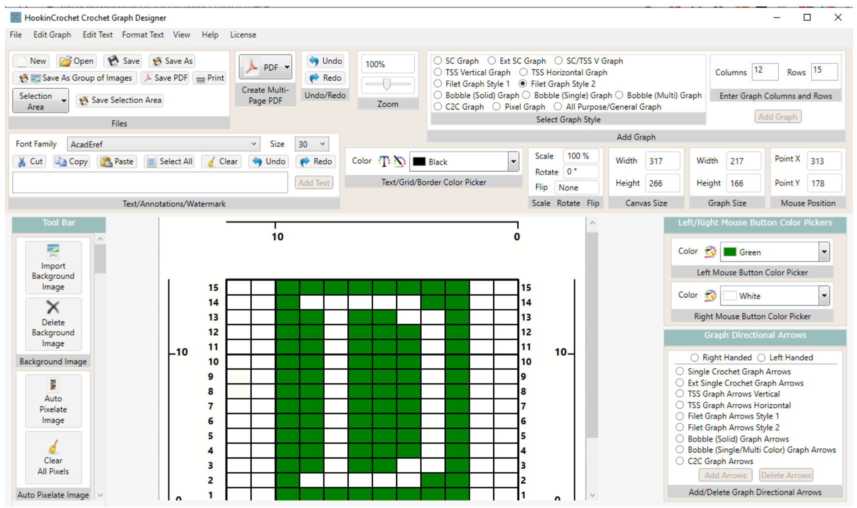


Figure 6.5: *Stitch Fiddle*

using one color for each cell. After the graph has been built, it can also be exported into a document including written instructions and the grid of the crochet graph. Even if this program may have more advanced functions, it still mainly works with crochet graphs which are limited in representable shapes and stitch types.



Zoom: 100%
[✓ Filet Graph Style 2]
Columns 12 Rows 15
Font Family AcadEref Size 30
Color Black
Scale 100 % Rotate 0 ° Flip None
Width 317 Height 266
Width 217 Height 166
Point X 313 Point Y 178
Color Green
Color White

Figure 6.6: *Crochet Graph Designer*

6.2.4 CrochetCharts by Stitchworks

CrochetCharts is a specialized editor[33] which focuses on creating crochet charts, see Figure 6.7. Stitch fonts can be imported and the stitch symbols are manually placed onto a canvas. The user can add circular or rectangular snapping grids. This supports the evenly spaced distribution of stitch symbols across the canvas. The stitches and arranged stitch selections can be rotated manually but also mirrored and rotated by a given amount of degrees which simplifies the creation of larger, repetitive patterns.

This software allows users to create charts using a stitch font and supports alignment with snapping lines. Due to the limited amount of snapping grid shapes, only limited pattern shapes can be assisted. Despite the supporting functions for rotation and positioning the process is still very manual, the pattern cannot be verified and does not have any internal representation of the actual stitch connections. The result of the software is an image, just like when using generic graphical editors.

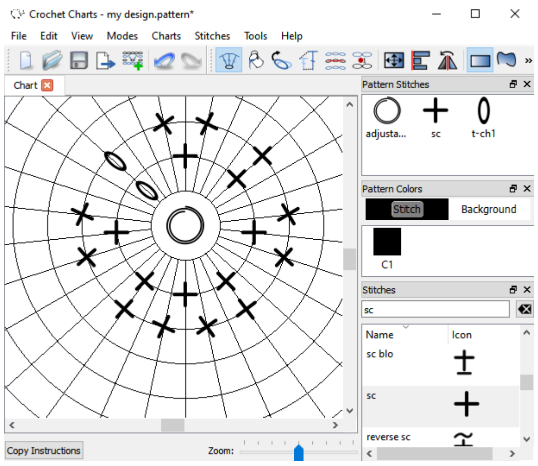


Figure 6.7: *CrochetCharts*

6.2.5 Crochet Chart Software 'My Crochet'

Another desktop app called *MyCrochet* from 2005 (see Figure 6.8) tried a similar approach to *CrochetCharts*. It already supplied the user with round and rectangular grids to help position the stitches. But instead of snapping to the lines, the stitches were placed by indicating the starting and the end point of the stitch. Across the line the stitch symbol positions and scales to fit its length.

Table 6.1: Overview of existing crochet chart and graph tools

	Pro	**Contra**
Writing Text Instructions	High flexibility. Arbitrary 2D and 3D patterns.	No standardized structure. Not visual.
Analog manual drawing	High flexibility. No limits in stitch symbols.	Difficult to evenly distribute and uniformly draw symbols. Only 2D patterns.
Stitch Fonts	Clear, reusable symbols for stitches. Usable in any text or image editor.	Many combinations need separate symbols. Only 2D patterns.
Stitch Fiddle	Chart designer can import stitch fonts.	No placement support. Only 2D patterns.
Crochet Graph Designer	Takes stitch space consumption into consideration.	Only specific types of 2D patterns.
Crochet Charts	Stitch fonts can be imported. Provides snapping guidelines.	Manual positioning. Only 2D patterns.
MyCrochet	Supports placement and rotation.	Limited amount of stitch symbols. Manual positioning. Only 2D patterns.
My Crochet Designer	Available to mobile devices. Supports placement and rotation.	Manual positioning. Only 2D patterns.

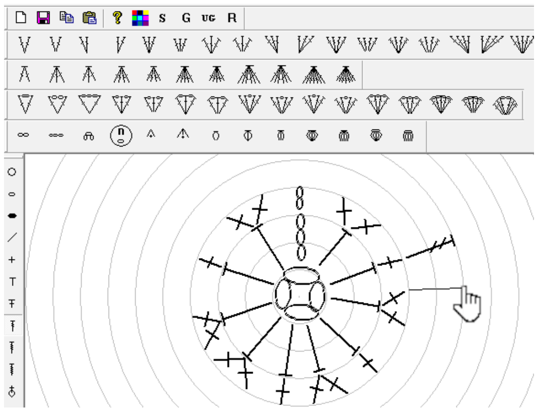


Figure 6.8: *MyCrochet* Desktop Application

6.2.6 Crochet Chart App 'My Crochet Designer'

As mobile application we only found *My Crochet Designer*[27], which allows placing, rotating and scaling stitch symbols on a canvas, see Figure 6.9. The canvas can also show circles or grids as guidelines where the stitches do not snap to. Additionally, a single circle or line can be overlaid across the canvas to serve as snapping support.

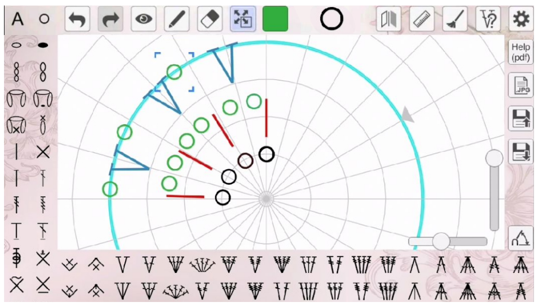


Figure 6.9: *My Crochet Designer*

6.3 Existing Research in the Field of Crochet

Crochet is a completely manual craft and more used as hobby than for the textile industry. This might be the reason why little research has been done is this field. Most crochet knowledge is shared through less official channels, such as blog entries[2]

[2]As an example, here, a blogger elaborates on the best way to crochet a sphere `https://mspremiseconclusion.wordpress.com/2010/03/14/the-ideal-crochet-sphere/` (visited 2020-06-30).

or tutorial videos. Nevertheless, there have been some attempts to represent and generate crochet or use it for visualization of concepts from other fields. Furthermore, hardware solutions for crochet problems have been built. Their patents are listed here, as well.

Zaharieva-Stoyanova et al. have already encountered the same issues we found regarding the digital representation of crochet patterns. Their work is aimed at preserving patterns for crochet and knit. As such, they built a serialization for knitting patterns using XML[44] and used it as basis to represent two-dimensional crochet patterns[43]. Their format incorporates the stitch symbols, used in crochet charts, and their position and rotation on the chart canvas. Zaharieva-Stoyanova et al. also designed a method to define compound stitches which can be formed through a combination of the basic stitches[42].

Street et al. present an algorithm to generate random, flat crochet patterns and produces a 2D crochet chart image as output[35] and also show an approach to generate free-form patterns where parts are represented through designed symbols. Their goal was to encourage crocheters to build new and unconventional structures. Another approach to generate crochet is shown by Gilbert[18] who generates radiating crochet patterns which can create sweaters.

Another crochet variant, interlocking crochet, to create flat reversible patterns, has been formalized using a grid system by Wildstrom et al.[39] and creation of symmetries and their combinations in patterns have been explored.

Apart from research about crochet patterns there have been several patents published to build crochet hardware. A limited crochet machine[1, 19] has been built and also a special crochet hook[28] was invented which supports the crocheter to keep track of the amount of finished stitches and rows.

Other fields made use of crochet and its ability to create arbitrary patterns. Crochet was used to visualize the nature of chaotic systems[29, 30] and hyperbolic planes[20, 37].

6.4 Summary

In this chapter, we presented existing crochet representations, tools, programs to support writing patterns and further research regarding crochet. We were able to observe that despite the attempts to standardize crochet representations, a tabular notation or a coordinate system, all of these approaches can not represent arbitrary two- and three-dimensional patterns. The diagramming system, crochet charts, also lack clear rules to unambiguously specify patterns through charts.

The list of tools and programs shows the limits of creating such pattern charts or graphs. Any tabular representation is even limited to solely rectangular 2D patterns of a small range of stitches. The charts that can be created using the existing tools and programs are able to represent arbitrary flat patterns but all of the outputs are always unstructured and plain images. None of the programs can support analysis, verification or auto layout of patterns and thereby, do not mitigate the original issue of ambiguous or erroneous instructions through mistakes or oversight.

In comparison, our concept and prototype also use the known stitch types and symbols but do not use the structure of crochet graph tables or plain images. Our underlying graph structure allows the incorporation of domain knowledge. Thereby, only the basic stitch types and not all possible combinations are required because compounds are positioned automatically and validations and auto-completions can be supported.

7 Future Work: Enabling an Entirely Digital Workflow

This report presented a concept and prototype to build digital crochet instructions for arbitrary 2D and 3D patterns. In the Chapter 1 we already presented an overview of the whole current process and possible additions and alternative workflows. The initial approach, explained in Chapter 3, allows digitizing crochet pattern instructions which lays the foundation for further concepts.

As shown in Figure 7.1, we see three main points which can be used to expand the concept developed in this report and which can lead to a new ideal path to creating and reproducing a pattern which ultimately enables the design of crochet patterns without requiring any crochet domain knowledge.

As a preceding step to the digital crochet pattern editor developed for this report, the overall shape of a new design could be modeled using 3D software. Such a model would then have to be transformed into basic crochet instructions based on the graph format presented in Chapter 3. This allows building arbitrary shapes without crochet domain knowledge and replaces the need to crochet on a trial-and-error basis to find a way to produce a certain shape.

In such a scenario, the digital crochet editor, as the prototype we presented in this report, would then be used to adapt the model to the users' preferences and modify the structure on a more detailed level instead of creating models from scratch.

Taking advantage of the underlying formal crochet representation, the editor could transform the instructions into various formats. One format could be the written instructions, as they are currently used. Another could be of a video tutorial style or other visualizations to help crocheters to reproduce the patterns.

A final step could be the transformation of the instructions into machine code for automatic reproduction of the design. While crochet machines do not currently exist and probably will not exist for a longer time, as explained in Chapter 1, there could be a virtual crochet machine. Such a machine would calculate the exact yarn flow according to the pattern instructions. The output of such calculations could be used for said tutorial videos, step-by-step pictures and any previews of shape and textures.

If implemented as a whole, this new workflow could not only transform the work of crochet designers but also the textile industry. Automatic creation of crochet instructions from 3D shapes would allow anyone to take advantage of the versatility of crochet. Crocheters can enjoy clear and uniform instructions from any digitally designed pattern. And with the creation of a crochet machine, crochet could compete with knitting in the textile industry.

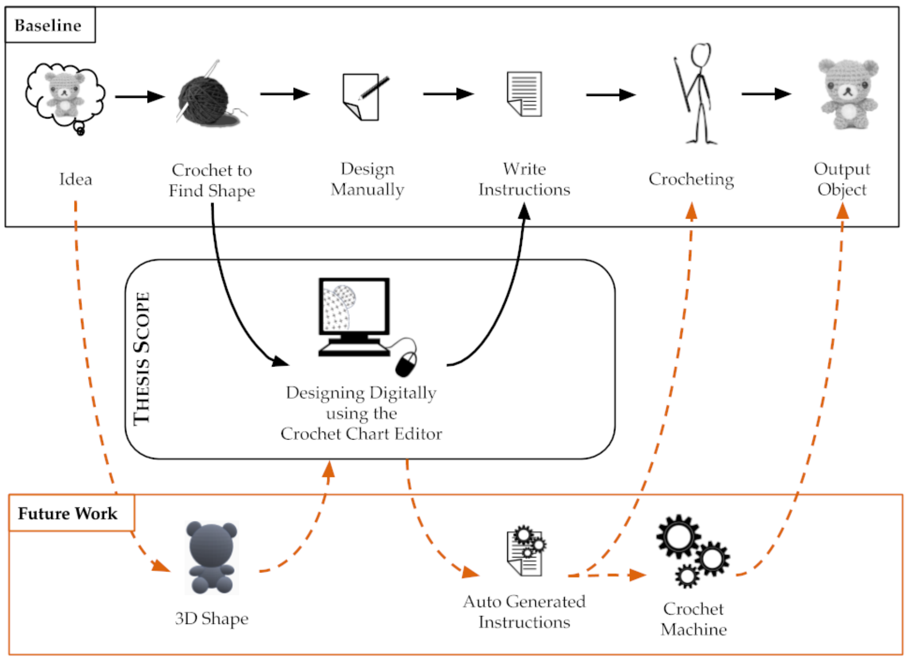


Figure 7.1: Overview of the incorporation of additional concepts into the current workflow feasible thanks to the foundation given in this report

8 Conclusion

In this report, we present an approach to support the crochet domain and its pattern writing process digitally by developing a domain specific language based on a graph structure which incorporates crochet techniques. We show the exemplary implementation of the designed graph structure as visual domain specific language used in a projectional editor for the domain of crochet. Apart from crochet, other domains could benefit similarly from their own digital representation which has an understanding of the basic concepts of the domain. This work could be the starting point for a framework which generalizes the steps we had to take in order to develop the structure, support visualization and editing in a projectional editor.

A thorough domain analysis shows that crochet is a very manual process, not just the handicraft itself but also the pattern writing process for crochet designers. Our developed graph structure is able to represent arbitrary, two and three-dimensional crochet patterns because it incorporates all explained crochet methods. We see tremendous potential for the digitization and support of the design process of crochet patterns. The digital and structured format enables analysis, verification and domain specific support for patterns. Our implementation and the evaluation interviews with professional crochet designers confirm the feasibility of our concept. We demonstrate that three-dimensional charts can be built, instructions represented unambiguously and used for calculations.

During the development process of our domain specific pattern representation as graph and a respective projectional editor, we realized that there were many steps which could be generalized into a framework to support such an approach for any domain. We manually developed the design of a graph structure as domain specific language but this could be generally supported through tests whether the structure is traversable or satisfies other properties. The visualization of the graph was straightforward and achieved good but not perfect results. Tools could allow enhancing the layout regarding the domain specific requirements. Similarly, visualization layers on top of the basic graph structure could be designed for various projections available to the editor. For our prototype, we calculated the necessary angles and positions of symbols to be shown on edges, while a tool could hide away such low level steps. A framework incorporating such tools could enable a straightforward workflow to build a domain specific language based on a graph and generate an editor allowing all common modifications to graphs with the constraints to the domain. For the crochet domain, our approach is the first step towards a more digital workflow thanks to a domain specific solution. If in the future, a framework simplifies the process of building visual domain specific languages and editors, other domains will benefit as well.

Bibliography

[1] N. G. A. Ehrmann J. Fiedler. *Crochet Machine*. Patentnumber: WO002018114025A1. URL: `https://depatisnet.dpma.de/DepatisNet/depatisnet?action=bibdat&docid=WO002018114025A1` (visited on 2020-05-19).

[2] Adriprints. *Stitchin*. URL: `https://www.fonts.com/font/adriprints/stitchin` (visited on 2020-06-01).

[3] G. P. Alexis Jacomy. *sigmajs*. URL: `http://sigmajs.org/` (visited on 2020-06-03).

[4] V. Asturiano. *3D Force-Directed Graph*. URL: `https://github.com/vasturiano/3d-force-graph` (visited on 2020-06-03).

[5] V. Asturiano. *Force-Graph*. URL: `https://github.com/vasturiano/force-graph` (visited on 2020-06-03).

[6] A. B.V. *vis.js*. URL: `https://visjs.org/` (visited on 2020-06-03).

[7] M. Bostock, V. Ogievetsky, and J. Heer. "D^3 data-driven documents". In: *IEEE transactions on visualization and computer graphics* 17.12 (2011), pages 2301–2309.

[8] M. Bostock. *D3.js*. URL: `https://d3js.org/` (visited on 2020-06-03).

[9] S. de Bruijne. *Stitch Fiddle*. URL: `https://www.stitchfiddle.com/` (visited on 2020-06-01).

[10] R. Cabello. *three.js*. URL: `https://threejs.org/` (visited on 2020-06-03).

[11] cambridge-intelligence. *The Keylines Toolkit*. URL: `https://cambridge-intelligence.com/keylines/` (visited on 2020-06-03).

[12] M. S. M. Company. *Merrow 18-E Crochet Stitch Industrial Sewing Machine*. URL: `https://www.merrow.com/crochet-sewing-machines/18e` (visited on 2020-05-19).

[13] C. Y. Council's. *www.YarnStandards.com*. URL: `https://www.craftyarncouncil.com/standards/crochet-chart-symbols` (visited on 2020-05-27).

[14] T. Dwyer. "Scalable, Versatile and Simple Constrained Graph Layout". In: *Computer Graphics Forum* 28.3 (2009), pages 991–998. DOI: `10.1111/j.1467-8659.2009.01449.x`.

[15] M. Franz, C. T. Lopes, G. Huck, Y. Dong, O. Sumer, and G. D. Bader. "Cytoscape.js: a graph theory library for visualisation and analysis". In: *Bioinformatics* 32.2 (Sept. 2015), pages 309–311. ISSN: 1367-4803. DOI: `10.1093/bioinformatics/btv557`. eprint: `https://academic.oup.com/bioinformatics/article-pdf/32/2/309/6689178/btv557.pdf`. URL: `https://doi.org/10.1093/bioinformatics/btv557`.

[16] E. Gamma, R. Helm, R. Johnson, and J. M. Vlissides. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1st edition. Addison-Wesley Professional, 1994.

[17] R. Gangi. *Warp knitting/crochet warp knitting machine*. US Patent 4,761,973. Aug. 1988.

[18] M. Gilbert. *Crochet Pattern Generator*. Apr. 2018. URL: `https://mattgilbert.net/projects/crochet_pattern/` (visited on 2020-06-30).

[19] N. Grimmelsmann, C. Döpke, S. Wehlage, and A. Ehrmann. "The largest crocheting machine in the world". In: *Melliand International* 25 (June 2019), pages 99–100.

[20] D. Henderson and D. Taimina. "Crocheting the hyperbolic plane". In: *The Mathematical Intelligencer* 23 (June 2001), pages 17–28. DOI: `10.1007/BF03026623`.

[21] HookinCrochet. *HookinCrochet Crochet Graph Designer Software*. URL: `http://www.hookincrochet.com/` (visited on 2020-06-01).

[22] HookinCrochet. *Symbols Basics Font Software*. URL: `http://www.hookincrochet.com/` (visited on 2020-06-01).

[23] N. Jain, A. Bhansali, and D. Mehta. "AngularJS: A modern MVC framework in JavaScript". In: *Journal of Global Research in Computer Science* 5.12 (2014), pages 17–23.

[24] M. James. *Warp knitting machine*. US Patent 1,924,649. Aug. 1933.

[25] E. G. Johanna Riedl. *Luftmaschen Häkelmaschine*. 2014. URL: `https://www.youtube.com/watch?v=LFyey3wCnNU` (visited on 2020-05-19).

[26] C. Koch. *The Computer Science of Knitting*. URL: `https://christinalk.github.io/blog/2017/02/06/knitting-programming` (visited on 2020-06-30).

[27] langaler1@gmail.com. *My crochet designer*. URL: `https://play.google.com/store/apps/details?id=com.dm.CrochetDesigner&hl=en` (visited on 2020-06-01).

[28] B. Moore, C. Moore, and T. Hayford. *Counting crochet hook*. US Patent App. 16/516,081. Jan. 2020.

[29] H. Osinga and B. Krauskopf. "Crocheting the Lorenz manifold". English. In: *Mathematical Intelligencer* 26.4 (Sept. 2004), pages 25–37. DOI: `10.1007/BF02985416`.

[30] H. M. Osinga and B. Krauskopf. "How to crochet a space-filling pancake: the math, the art and what next". In: *system* 2 (2014), page 3.

[31] A. Schachtschabel. *Berliner Häkelschrift*. URL: `http://www.schachzabel.de/` (visited on 2020-05-28).

[32] N. Software. *GoJS*. URL: `https://gojs.net/latest/index.html` (visited on 2020-06-03).

[33] S. Software. *Crochet Charts*. URL: `http://stitchworkssoftware.com/` (visited on 2020-06-01).

[34] J. P. Strathausen. *Dracula Graph Library*. URL: `https://www.graphdracula.net/` (visited on 2020-06-03).

[35] K. A. Street et al. "Generative crochet: using computational methods to augment handicraft". PhD thesis. 2018.

[36] Syncfusion. *HTML5/JavaScript Diagram Library*. URL: `https://www.syncfusion.com/javascript-ui-controls/js-diagram` (visited on 2020-06-03).

[37] D. Taimina. *Crocheting adventures with hyperbolic planes*. AK Peters/CRC Press, 2009.

[38] J. Walke. *reactjs*. URL: `https://reactjs.org/` (visited on 2020-06-30).

[39] D. J. Wildstrom et al. "Symmetries of Intermeshed Crochet Designs". In: *Bridges 2019 Conference Proceedings*. Tessellations Publishing. 2019, pages 203–210.

[40] E. You. *Vue.js*. URL: `https://vuejs.org/` (visited on 2020-06-01).

[41] yworks. *yFiles for HTML*. URL: `https://www.yworks.com/products/yfiles-for-html` (visited on 2020-06-03).

[42] E. Zaharieva-Stoyanova and D. Beshevliev. "Digital Representation of Crochet Symbols Sets". In: *Digital Presentation and Preservation of Cultural and Scientific Heritage*. Volume 8. Bulgaria: Institute of Mathematics and Informatics – BAS. 2018.

[43] E. Zaharieva-Stoyanova and S. Bozov. "Application of XML-based Language for Digital Representation of Crochet Symbols". In: *Digital Presentation and Preservation of Cultural and Scientific Heritage* VII (2017), pages 181–190.

[44] E. Zaharieva-Stoyanova and S. Bozov. "Portable knitting format-XML-based language for knitting symbols description". In: *Proceedings of the 16th International Conference on Computer Systems and Technologies*. 2015, pages 252–259.

[45] zoomcharts. *JavascriptCharts*. URL: `https://zoomcharts.com/en/` (visited on 2020-06-03).

[46] L. O. Zorini and M. C. Miino. *Crochet galloon machine*. US Patent 7,251,960. Aug. 2007.

List of Figures

ISBN 978-3-86956-505-7
ISSN 1613-5652