Computational Illusion Knitting

ANONYMOUS AUTHOR(S)
SUBMISSION ID: 755

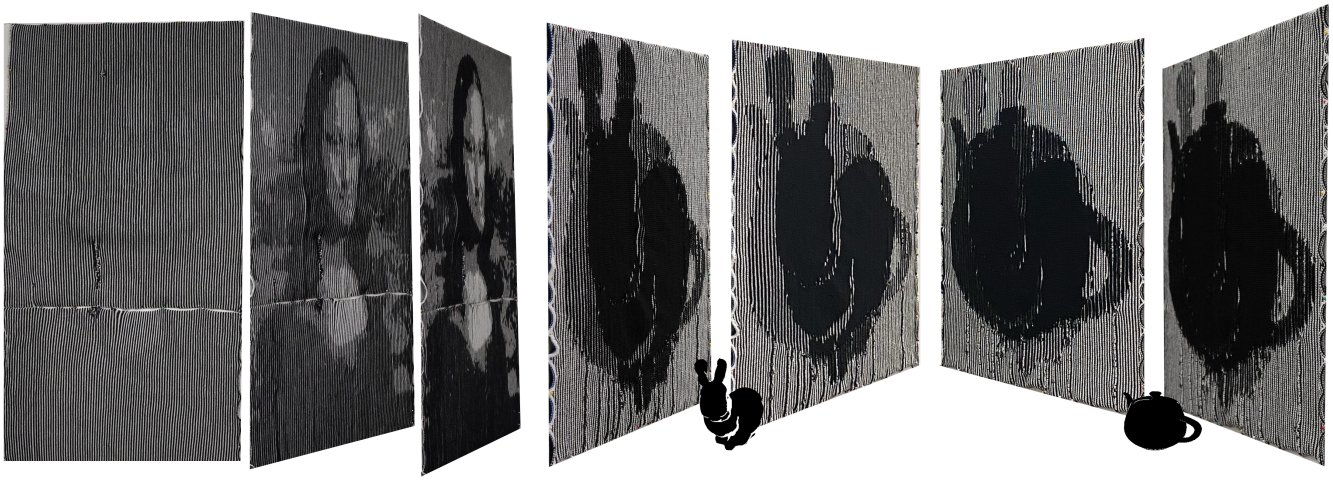


Fig. 1. We propose a framework for computational illusion knitting that enables simple production of traditional and completely novel illusion designs, such as patterns with more than one embedded feature image, and show the fabricated designs generated with our method. On the left, a Mona Lisa appears when viewed from the side. On the right, the Stanford bunny appears from one side, and the Utah teapot from the other.*

Illusion-knit fabrics reveal distinct patterns or images depending on the viewing angle. Artists have manually achieved this effect by exploiting "microgeometry", i.e., small differences in stitch heights. However, past work in computational 3D knitting does not model or exploit designs based on stitch height variation. This paper establishes a foundation for exploring illusion knitting in the context of computational design and fabrication. We observe that the design space is highly constrained, elucidate these constraints, and derive strategies for developing effective, machine-knittable illusion patterns. We partially automate these strategies in a new interactive design tool that reduces difficult patterning tasks to familiar image editing tasks. Illusion patterns also uncover new fabrication challenges regarding mixed colorwork and texture; we describe new algorithms for mitigating fabrication failures and ensuring high-quality knit results.

CCS Concepts: • **Computing methodologies** → **Graphics systems and interfaces**; • **Applied computing**;

Additional Key Words and Phrases: illusion knitting, machine knitting, knitting, fabrication

ACM Reference Format:
Anonymous Author(s). 2023. Computational Illusion Knitting. 1, 1 (May 2023), 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

XXXX-XXXX/2023/5-ART $15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Researchers have developed a wide range of tools to support diverse knitting applications, including the design of garments, plush toys, soft actuated objects and interfaces, and sensing technology [Albaugh et al. 2019, 2021; Kaspar et al. 2021; Luo et al. 2021, 2022; Narayanan et al. 2018]. All these applications depend on understanding how knit stitches can be combined to define geometric features and surface appearance. However, one important aspect of knitting has been overlooked: certain stitch combinations can induce small-scale surface structure, or "microgeometry", by varying stitch heights. Properly controlled, this microgeometry can produce fascinating visual effects when the fabric is viewed from different angles.

Indeed, hand-knitting artists have been exploiting stitch height differences for decades, developing a technique known as *illusion knitting* where embedded images are only visible when the fabric is viewed from a shallow angle [Harmon 2011; Hoxbro 2004; Nakamura et al. 1982; Stoller 2004; WoollyThoughts 2023]. An example of such an illusion is the Mona Lisa in Figure 1. Traditionally, illusion knit patterns hide a single image, take dozens of hours to design, and dozens more hours to manually knit. These techniques suggest a much broader and more diverse space of view-dependent effects that could be generated by exploiting microgeometry.

This paper lays the groundwork for computational illusion knitting by developing techniques for design space exploration and effective fabrication. We present new analysis and representation tools for knitting, as current knitting abstractions fail to capture microgeometry or account for view-dependent visual effects [Hofmann et al. 2019; Narayanan et al. 2018; Popescu et al. 2018]. Furthermore, while microgeometry effects have been investigated in

*The yarn broke during knitting of the Mona Lisa; we will reknit it for the revised manuscript.

various fabrication techniques, knitting poses a new fundamental challenge: isolating geometric effects is not straightforward. The interconnected nature of continuous yarns in knitting, which generates sequential stitches, makes it challenging to achieve specific local geometric variations without causing significant repercussions in other parts of the fabric.

Our key insight lies in defining a design space for illusion knitting that incorporates an understanding of constraints that enable microgeometry effects while ensuring machine manufacturability. We combine an understanding of the feasible space of knitted microgeometry with an analysis of artistic practices for creating illusions and use it to develop a tool that enables exploring microgeometry effects within the fabric.

Our design tool for illusion knitting has three main advances. First, we develop a microgeometry-aware renderer to accurately predict the appearance of illusion designs when viewed from different angles. Our renderer is fully differentiable and simulates the effects of the microgeometry, and therefore can be directly used for design optimization. We notice, however, that optimization alone does not give users full control of the result, as evidenced by discrepancies with single-view illusions produced by artists. To provide additional control over the pattern, we also developed a new interactive design tool that incorporates strategies for effective illusions we extracted from an analysis of the knit illusion craft (e.g., using striped patterns and large color regions). Finally, we define new knitting compiler passes to mitigate failures arising from the novel mixing of colorwork and texture central to illusion knitting.

We demonstrate the effectiveness of our computational tools to simplify the generation of high-quality single-view illusions, reducing it to familiar image editing tasks. We show how these designs can be manufactured on industrial v-bed knitting machines. Finally, we also demonstrate how our tools enable entirely novel illusion knits, such as a new "double-view" illusion that embeds *two* images within a single knit fabric, seen in Figure 1.

2 BACKGROUND

Knitting Terminology. A *stitch* is the basic knit unit formed by pulling a loop of yarn through an existing loop. Stitches are typically arranged in a grid structure; rows of loops are called *courses*, and columns of loops are called *wales*. The direction in which the yarn is being pulled through each loop, either front-to-back or back-to-front, determines the shape of the stitch, resulting in a *knit* stitch or a *purl* stitch. There exist many different arrangements of knits and purls, which are called *textures*. Full specifications of knit designs are called *patterns*. We provide a more thorough introduction to knitting in Supplemental Material Section 1.

Illusions. Illusion knitting exploits occlusions between neighboring courses created by particular arrangements of knits and purls. Traditional illusion knitting uses two colors, c_1 and c_2, and alternates them every two courses, i.e., two rows of c_1 followed by two rows of c_2 to create alternating course-length stripes. Each set of two *physical* courses creates one *logical* row composed of a mix of *flat* stitches and *bump* stitches (visible in Figure 2). Logical flat stitches are formed by a combination of one knit stitch pulled through another knit stitch from the previous row. Logical bump stitches are formed by a combination of one purl stitch pulled through a knit stitch from the previous row. Subsequent sections specify illusion designs in terms of logical patterns and stitches.

(a) A top-down view of a traditional single-view pattern.

(b) A close up of the microgeometry of single-view patterns. On the left side, white is knit; on the right side, purled, and vice-versa for black.

Fig. 2. Detailed view of a single-view illusion pattern; the knitted version is depicted in Figure 3a.

When viewed head-on, every stitch is visible, resulting in a simple striped image. However, when viewed from a shallow (closer to 0 degrees) angle, bump stitches can occlude flat stitches in the subsequent row**. Given a fixed two-color $M \times N$ image A where $A_{i,j}$ denotes the pixel at row i and column j, traditional illusion design techniques yield a $2M \times N$ logical pattern P satisfying the following constraints:

$$\begin{aligned} \texttt{color}(P_{2i,j}) = c_1 \ \wedge \ \texttt{color}(P_{2i+1,j}) &= c_2 \\ A_{i,j} = c_1 \iff \texttt{stitch}(P_{2i,j}) &= \texttt{BUMP} \\ A_{i,j} = c_2 \iff \texttt{stitch}(P_{2i+1,j}) &= \texttt{BUMP} \end{aligned}$$

We use `color()` and `stitch()` to indicate the color and stitch type for each element of a design. These single-view illusion constraints are always satisfiable for any input image A. This technique was used to generate the design for the Mona Lisa on the left of Figure 1. To manufacture a logical design P, we first lower it to a physical design Π according to the following rules:

$$\begin{aligned} \texttt{color}(\Pi_{2i,j}) = \texttt{color}(\Pi_{2i+1,j}) &= \texttt{color}(P_{i,j}) \\ \texttt{stitch}(\Pi_{2i,j}) &= \texttt{KNIT} \\ \texttt{stitch}(\Pi_{2i+1,j}) &= \texttt{PURL if } \texttt{stitch}(P_{i,j}) = \texttt{BUMP else KNIT} \end{aligned}$$

Note that a physical design Π will again have twice as many rows as the logical design P it was lowered from — artists have compensated for the resulting distortion by only considering every other row of the input image A, but in this work, we instead treat such corrections as simple, familiar image editing tasks on the input A.

Given a physical design, existing knitting compilers can be used to generate knitting commands for industrial knitting machines to manufacture the target fabric. Finally, we note that while traditional single-view illusion knits are designed for viewing from one side, viewing from the other side yields a similar effect with only slight variations at borders.

3 RELATED WORK

Fabricated View-Dependent Effects. View-dependent effects play on our visual perception via lighting and shadows. They utilize self-occlusion or self-shadowing to change the visibility of parts of the object to show different images at different viewing angles. In computer graphics, prior work has studied how to fabricate objects

**Rows with adjacent bump stitches can also create an "average color" effect.

with these artistic effects using different fabrication methods. Some works focus on achieving a specific appearance of a surface under a certain lighting condition or environment through optimizing microfacets [Papas et al. 2011; Regg et al. 2010; Schwartzburg et al. 2014; Snelgrove et al. 2013; Weyrich et al. 2009] or lenticular structures [Zeng et al. 2021]. Many fabrication techniques and materials have been used for different applications, including Lego bricks [Mitra and Pauly 2009], 3D printing [Alexa and Matusik 2011, 2012; Bermano et al. 2012; Peng et al. 2019], and more specifically, UV printers [Perroni-Scharf and Rusinkiewicz 2023; Sakurai et al. 2018] and printing with magnetic materials [Pereira et al. 2017]. Also a kind of view-dependent effect, illusion knitting relies on precisely occluding certain stitches – the microgeometry in a knitted object – to embed images in different viewing angles.

Computational Knitting. Several recent papers explore computational tools for knitting design and fabrication. Some focus on how to represent knit objects for simulation [Wu et al. 2018; Yuksel et al. 2012]. Albaugh et al. [2023] also studies the grain of machine-knit fabric. Others focus on tools for creating knitting instructions given a 3D mesh input, enabling a wide variety of stuffed animals, garments, and other 3D shapes in both hand knitting [Igarashi et al. 2008; Wu et al. 2019] and machine knitting [Jones et al. 2022; Kaspar et al. 2021; McCann et al. 2016; Narayanan et al. 2018; Popescu et al. 2018]. Nader et al. [2021] take in 3D meshes and generate knitting instructions, but their focus is on a machine-agnostic, hardware-independent intermediate representation that further facilitates compilation to diverse fabrication methods. Knitting design methods are also developed for some specific applications, including actuated soft objects, pneumatic devices, and conductive interfaces and sensing devices [Albaugh et al. 2019, 2021; Liu et al. 2021; Luo et al. 2021, 2022]. In addition to achieving a certain shape in knitting, the design of textures is also an important topic of research. Narayanan et al. [2019] provide a tool that allows easy stitch-level editing of textures and colorwork while providing machine knittability guarantees. Hofmann et al. [2020] examine knit textures, which originate in the tension of the yarn between loops, or the loop connections themselves. They create a language, KnitSpeak, to describe the generation of these textures, following from the natural, ad-hoc pseudo-DSL used by hobbyists, and compile, knit, and examine many example textures.

Although colorwork is a well-known facet of computational knitting, existing work [Lin and McCann 2021; Lin et al. 2023, 2018; McCann et al. 2016; Narayanan et al. 2019] only provide insights on how to schedule patterns with either colorwork or texture. No past work has focused on exploring and enabling the design space spanning a combination of colorwork and texture, which is a necessary component of illusion knitting, and potentially a new tool in the democratized designer's toolkit.

Illusion knitting. Illusion knitting, also known as shadow knitting, has been explored by knitters for the past few decades, particularly in garments [Harmon 2011; Hoxbro 2004; Nakamura et al. 1982; Stoller 2004] and in a variety of complex and detailed pieces [WoollyThoughts 2023] in exhibitions in 2010. Current illusion knitting artworks typically hide a single image or pattern and appear to be stripes of two colors from a head-on view. There are also design tools and tutorials for charting illusion patterns [orochi 2006]. However, little work explores beyond the traditional paradigm that hides a single image within stripes. Several books include patterns that change colors between different macro regions [Hoxbro 2004; Nakamura et al. 1982], but they also use the same two-color alternating illusion strategy, and no resources discuss multiple feature images in one knit illusion, much less the challenges of making such designs machine knittable. In this paper, we establish the first foundation for understanding the broader space of illusion knitting and develop methods for hiding more than one image, using arbitrary colors, and mitigating fabrication challenges from knitting machines.

(a) The same pattern knit with colors switching each course (top) or each wale (bottom).

(b) Examples of knit-purl patterns with undesirable microgeometry.

(c) The height of a 1-wide purl bump is greatly diminished by its legs being pulled under the next stitch.

Fig. 3. Pattern choices that affect occlusion.

4 A DESIGN SPACE FOR ILLUSION KNITTING

The body of artistic work in illusion knitting suggests new opportunities for controlling appearance in computationally designed knits. Our first step is to understand this space, the constraints necessary to ensure fabricability, and how to simulate illusion effects.

4.1 Understanding Illusion Effects

Our goal is to enable designers to freely experiment with diverse color and texture variations to create novel viewing-angle-dependent knit effects. However, challenges arise when considering the impact of fabrication and yarn tension. What happens when colors change within a row? Can more than two colors be used? For instance, knitting a densely random pattern of purls and knits with two colors yields no illusion effects due to yarn tension (Figure 3b).

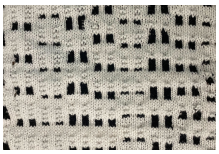
As a first step we identify the simplest atomic unit we can represent that isolates the occlusion effect: a 2x2 physical stitch block. Through a series of experiments, we observe a few important properties. First, frequent color changes lead to very thick fabric, reducing the height difference between knits and purls (Figure 3a). Second, arbitrary patterns may manifest emergent and unpredictable effects due to subtle tension changes (Figure 3b). These two properties are important, but we consider them soft constraints, existing on a continuum between seeing the illusion to seeing a jumble of colors. Third, one purl stitch surrounded by two knits have a greatly reduced height, as the knit stitches pull the ends of the bump down (as opposed to a longer run of purls) (Figure 3c). This property we consider a hard constraint, and we constrain patterns such that all raised regions are at least 2 logical bumps wide. These microgeometry constraints enable patterns that have the necessary consistently-tall ridges to induce an illusion.

4.2 Differentiable Renderer

We developed a differentiable renderer for illusion knitting that encodes these fundamental constraints. Our renderer enables immediate design preview and gradient-based optimization of illusion patterns to make target images. We make two simplifying assumptions in this renderer that allow us to write antialiased pixel colors in an analytic closed form, enabling autodifferentiation to properly handle stitch occlusion. The first simplification is to model logical stitches as cuboids of fixed color and height. The second is that the camera is orthographic with respect to the width. These assumptions ensure that viewing rays are parallel to knit wales and that they can only intersect with the leading or top faces of each stitch cuboid within one wale. The 3D rendering problem is thus reduced into a single 2D problem for each column of stitches, where the cuboid faces become line segments.

We differentiably render each wale once by analytically computing the contribution of each line segment S_i onto each pixel P_i accounting for occlusion by earlier segments $S_{j<i}$: $\left|\left(S_i \setminus \bigcup_{j<i}\left(S_j\right)\right) \cap P_i\right|$. Here S_k and P_k are the projections of the $k-$th stitch and pixel into image space. A differentiable closed form expression for this is given in Supplemental Material .

The result of our orthographic assumption is that the output image is always rectangular, even though a true camera image would be trapezoidal. We chose this because human perception naturally corrects for perspective when viewing an illusion knit from the side, and it also aids when automatically fitting an illusion knit pattern to a target image because they are both rectangular.



To validate our simplified renderer and tune its parameters, we knit a test pattern, shown inset, with various configurations of raised purls of one color obscuring varying length runs of knits a second color. We measured the angle at which the obscured lines are fully obscured and used this to calculate the length-to-height ratio of purl stitches over knit stitches. Using these computed parameters, we rendered the same test patterns to verify that the illusion effect was preserved. The same pattern can be used to tune parameters for a new setup, for example, when using a thicker yarn or different gauge needles. Finally, we compare rendered output to knit output across many test cases, and verify the results.

4.3 Global Pattern Optimization

Since our renderer is differentiable, it enables direct optimization. Figure 8 (c) illustrates an example of optimizing a single-view pattern using gradient descent and MSE loss with a blur filter. While the colors are fixed as stripes, the heights are allowed to vary during optimization. The challenge with optimization, however, is that it does not capture the domain expertise that artists use to create illusions. To address this, we leverage the design space representation and the developed renderer in constructing a design tool for creating knit illusions more directly.

Fig. 4. A rendered prediction of this single-view illusion, and the knitted result. Note the small pixels of black on the white hand, suggesting fingers, and how they are not visible in the knitted image.

5 DESIGN TOOL

We developed a new interactive design tool for illusion knitting based on (1) a survey of traditional (single-view) illusion patterns and (2) a formal understanding of the constraints necessary for entirely new kinds of illusions, e.g., embedding multiple images.

5.1 Insights from Artistic Illusion Knitting

Our review of traditional illusion patterns showed that designers employ three major strategies for effective illusions. First, designers generally do not use dithering to achieve color variations. We believe this is due to the relative coarseness of knit stitches; good dithering effects would require significantly larger fabrics to provide the required resolution. Additionally, when considering perception, scattered points tend to blend into the overall waviness of knit fabrics (Figure 4). Instead, artists employ bands of color, introducing regular patterns within each band to make region boundaries distinct. These patterns effectively define color boundaries, enhancing the legibility of the image overall. Further, artists work with a small number of colors. Traditional patterns have achieved shading by alternating two yarn colors to provide a striped pattern as a "third color". Trying to render more than three logical colors using only two yarns is ineffective as the image becomes too busy. Finally, In addition to reducing the color palette, designers ensure that the image is segmented into simple, large, continuous regions of color. Luckily, these observations mean we can build upon a large corpus and long history of image processing work to transform our input images into simplified, quantized targets.

5.2 Multi-View Illusion Knitting Constraints

In addition to incorporating artists' hard-won insights, we establish formal constraints to model the design space for new types of illusions; Table 1 provides high-level comparisons. As noted earlier, "single-view" encodes an input image A from the sides and a simple striped pattern from the head-on. Our new "single-view + head-on" illusion embeds different images A and B from the head-on and side views. Most challenging, our novel "double-view" illusions constrain the design to produce different images depending on which side the fabric is viewed from. Unlike the single-view constraints in section 2, these new illusion constraints are often *unsatisfiable* for given input images A and B. Our goal is to apply artists' insights to help satisfy as many constraints as necessary to achieve effective illusion knits.

	Left side	Head-on	Right side
Single-view	Image A	Striped	approx. Image A
Single-view + head-on	Image A	Image B	approx. Image A
Double-view	Image A		Image B

Table 1. Description of expected views.

Using the same notation as in section 2, single-view + head-on illusion designs must set $\texttt{color}(P_{i,j}) = B_{i,j}$ and satisfy:

$$A_{i,j} \neq A_{i+1,j} \implies \texttt{stitch}(P_{i,j}) = \texttt{FLAT}$$
$$\texttt{stitch}(P_{i-1,j}) = \texttt{FLAT} \implies \texttt{color}(P_{i,j}) = A_{i,j}$$

This is much more constrained than single-view. For example, in the head-on image B, color c can only exist where c is also seen from the side. In practice, these constraints are difficult to satisfy without B using some form of striped pattern as its background; otherwise A cannot exploit occlusion in the side view.

For double-view illusions, the top view is unconstrained. Assuming row indices increase from left-to-right, a perfect double-view illusion will satisfy:

$$A_{i,j} \neq A_{i+1,j} \implies \texttt{stitch}(P_{i,j}) = \texttt{FLAT}$$
$$\texttt{stitch}(P_{i-1,j}) = \texttt{FLAT} \implies \texttt{color}(P_{i,j}) = A_{i,j}$$
$$B_{i,j} \neq B_{i-1,j} \implies \texttt{stitch}(P_{i,j}) = \texttt{FLAT}$$
$$\texttt{stitch}(P_{i+1,j}) = \texttt{FLAT} \implies \texttt{color}(P_{i,j}) = B_{i,j}$$

Again, many pairs of input images A and B will fail to satisfy these constraints, further motivating the need for flexible design tools to explore the illusion knitting space. For example, consider the input targets from the inset figure for a double-view illusion. This scenario presents an unsolvable pair of inputs: one image demands white where the other demands black.

5.3 An Image-Editing Illusion Design System

Inspired by artists' insights and novel illusion constraints, we propose a design system based on *tiles*: n by m sub-patterns that are assigned to and repeated over a region of the pattern. Tiles align with how artists use regular patterns within large regions of color. Tiles also provide control in allowing users to find alternatives, given the constraints for double-view. For example, to address the above example, a user could decide to use tiles of striped patterns instead of the black color, taking inspiration from traditional illusion knitting, where stripes serve as a valid "pattern color". Figure 5 illustrated how the tension is resolved via a 3 by 1 tile: white flat, black flat, and white bump. When this tile is viewed from the left, the white bump occludes the black flat, making the image appear white. When viewed form the right, we see the white bump **and** black flat, inducing a striped image.

Using tiles as a fundamental construct, we developed a workflow that enables users to reduce many pattern design tasks to familiar image editing tasks, e.g., quantizing input images into large, simply-colored regions. Combined with our differentiable renderer for previewing illusions without manufacturing them,this approach

Fig. 5. A knit version of our recolored input patterns. On the left, the black background and white square has become a striped background and white square, and on the right, the white background and black square has become a white background and striped square.

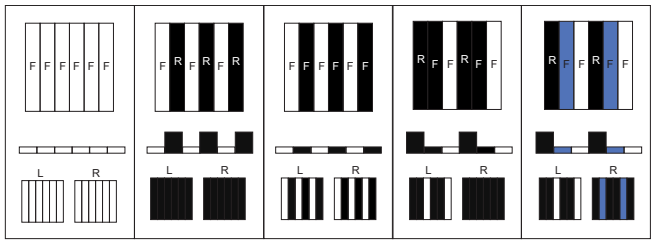


Fig. 6. New tiles types, assuming infinite tiling. Each tile is shown as a top-down design (top), with its height map (middle) and two side views (bottom).

greatly simplifies and accelerates iterating to achieve effective illusions via re-balancing, image editing, and simple color selection while simultaneously providing fine-grained control.

Specifically, our tool computes all color regions of the inputs, which for double-view emerge from the overlap of the inputs, and are the same as the input colors in single-view. Using our interface, users can then select and preview many candidate tiles for each color region. Guided by the illusion constraints, we design a series of new canonical tiles, which give rise to default color assignments, pictured in Figure 6. Each tile maps to an existing type of pattern color: solid or two-color alternating stripes, and can be used in designing single-view head-on and double-view illusions.

We also automatically generate suggested assignments from color region to pattern color for double-view. We filter all possible assignments on two properties: first, each pattern color in an overlap must share at least one yarn color with the other color in that overlap; it is impossible to make assignments if there are more overlaps than possible tiles. Second, we check that the recolored assignments maintain the lightness order of the original input.

6 FABRICATION

We focus this discussion on enabling fabrication on a V-bed knitting machine, which uses two rows of needles (called beds). When creating textures, the front bed is used for knit stitches and the back bed for purls. This machine has multiple yarn carriers for colorwork but these must coordinate correctly to ensure the knit object stays in one single piece. For more detailed background on how such machines work, see Supplemental Material Section 2.

Single-view knits posed no issue with one color per course, but with the introduction of tiles, we now have color changes within a row, which entails mixing colorwork and texture. Existing machine schedulers [Lin and McCann 2021; Lin et al. 2023, 2018; McCann et al. 2016; Narayanan et al. 2019] are not designed to support the mixed colorwork and texture in illusion patterns. There are many

existing techniques for colorwork: *Plating* works all yarns together, producing color patterns by switching which yarn sits in front; *Stranded* knitting passes un-used colors behind stitches, forming strands (aka. floats) along the back; *Intarsia* portions the knitted item into mono-color blocks, intertwining the yarns at the borders of the blocks; and *doubleknit jacquard*, arguably the most prevalent technique for machine-knit colorwork, uses all needles on both beds. We give a more detailed overview of these techniques in Supplemental Material Section 4.Importantly, these techniques do not typically integrate with textures in a systematic way.

We propose a novel colorwork strategy that supports the colorwork and textures found in illusion knits. Our strategy is based on four key properties of illusion patterns: (1) bumps must be clear so that the illusion is visible; (2) patterns often have many large regions or long runs of a single color, especially given how many of our presented tiles are striped; (3) there may be areas with dense color switching, for example when small overlap regions border differently-colored overlap regions; (4) purls may occur anywhere, and on the knitting machine, this means that the back bed cannot freely be used — it is needed to create purl stitches.

From (1), we decide against plating as a colorwork technique, as bulky knit stitches interfere with the illusion. Properties (2) and (3) indicate there is no good choice between stranded and intarsia, which are better suited for frequent and infrequent color changes, respectively. Ideally, we use stranded knitting in areas of fine detail, while keeping carriers divergent in large single-color areas. We therefore propose a best-of-both-worlds combination of stranded colorwork and intarsia. However, each of these techniques has specific challenges to address. In stranded, floats should not appear on the technical front of the knit item, and in intarsia, different yarns must be securely connected so the piece does not unravel. While solutions exist in contexts where each technique is typical, they may not apply when combining these techniques together and under the constraints imposed by (4). Thus, we introduce three compiler passes to address each of these concerns. The pseudocode for each is available in Supplemental Material Section 4.1.

6.1 Float avoidance

Floats appear when a carrier knits on one part of the bed, then moves to knit on another section of the bed. Because the yarn is not cut, it runs along the work loose. Carriers move in the space between the front and the back bed. Thus, floats appear on the front of the work when this movement passes in front of stitches on the back bed. To avoid floats, whenever a carrier moves freely (i.e. not while knitting), we check for purls it might cross in front of, and wrap the carrier movement with extra transfer instructions such that the stitches move to the front bed before the carrier passes, then back to the back bed after.

This operation incurs a cost of two extra carriage passes per free carrier movement if there are back-bed stitches to transfer. Therefore, checking that this operation is necessary saves time.

6.2 Yarn catching

Changing colors typically means using a new yarn. Without explicitly ensuring that two yarns are “caught”, switching colors causes gaps in the knit. These gaps are especially apparent when they span multiple rows in the same column. Imagine a degenerate example where a knit piece consists of two different-color rectangles side-by-side. Without catching, the two yarns never contact each other, and so the piece becomes two single-color rectangles once it is cast off.

In intarsia, two yarns are not easily caught: the carrier rail of the knitting machine is designed to avoid yarns tangling. There is a folk methodthat joins two pieces using the back bed. However, it may not be appropriate when the back bed is densely occupied, as is the case for illusion knits.

In our case, we use a plating stitch at the beginning of a color change to catch. Though we did not want to use plating everywhere, it has a minimal effect when used sporadically. Both yarns get used in the same stitch, and are pulled through one loop together, providing a point of connection, or durable yarn contact, between two sides. We control which color is more prominent by specifying it is to be laid down first.

6.3 Yarn path planning and anchoring

In our generated patterns, since carriers do not always move together, we must take care when scheduling Given a carriage direction, the knitting machine expects the carrier to originate from behind the carriage in that direction. Consider a scenario where the carrier is currently stopped the middle of the bed, and in the next course it is going to be used to knit from the left side to the right side of the bed. No matter which end of the course the carrier starts from, the carrier will be in front of the carriage, so no yarn is placed onto the needles and no stitch is formed.

To ensure that every stitch is knit, we propose two different strategies. The first is to split the next course where the yarn is anchored, first knitting in one direction from the anchor to one side, then turning around and knitting from the anchor to the other side.

However, there may be cases where this strategy is not possible. Imagine a plated stitch where the carriers involved are on separate sides of the carriage. There is no direction that the plated stitch can be knit such that both carriers will come from behind the carriage. In this scenario, we introduce an “anchoring stitch”. The secondary carrier being plated will first make an anchoring stitch (for example, a knit) to anchor it on the same side of the carriage as the primary carrier. Of course, this movement may incur more float-avoidant transfers.

7 EVALUATION

7.1 Automatic Generation of Single-view Illusions

The Mona Lisa and Cypress illusions in Figure 1 and Figure 8 (b) demonstrate our tool’s capability to automate the generation of single-view illusion patterns, significantly accelerate the process, and enable non-experts to create intricate knit designs. In these examples, the user only needs to input a grayscale image and potentially adjust the thresholds for a 3-color quantization.

Users may also want to perform some pre-processing on their input to mitigate issues with naive quantization. Figure 7a showcases a sequence of experiments with Munch’s The Scream, demonstrating the application of rich image editing capabilities within familiar

interfaces before importing into our tool. The initial image downloaded from Wikipedia is quantized within the UI (a), but lacks contrast and appears fuzzy around the face and hands. Applying a posterize filter in Illustrator improves the crispness of the image (b), although some sky details are lost. Manual editing or utilizing a diffusion model are alternative approaches for obtaining a clearer version (c and d, respectively). The knitted outputs of (b) and (d) are presented in Figure 8.

7.2 Expanding the Space of Knit Illusions

In addition to automating the design of known illusion types, we showcase our tool's potential to expand the range of achievable illusions in knitting. One notable advancement is the generation of illusions with multiple views, which was previously unexplored. Below we describe examples generated using our tool, all of which took five to twenty minutes of iterating on the input images, yarn choices, tile selection, and editing.

First, we show in Figure 9 an example of generating an image that can be viewed head-on but is occluded in a side view. This is possible because the design adheres to the constraints defined in subsection 5.2.

Furthermore, we demonstrate the creation of double-view illusions, where two two-color images are knitted using two yarn colors. Instead of enforcing a single color for the background and another for the foreground, which is infeasible based on the constraints we defined, we employ a striped pattern for the background. This approach yields sharp results in various illustrations, such as the bunny and teapot in our teaser image (Figure 1) and the ECG pulse and heart (Figure 10 (a)). We even tackle more challenging cases involving text, where one side reads "SIGGRAPH" and the other reads "ASIA 2023" (Figure 10 (b)).

Finally, we explore the use of multiple colors in our fabrication technique. In one example, we showcase the SIGGRAPH logo in red and blue on one side and an orange logo for SIGGRAPH ASIA 2023 on the other side (Figure 10 (c)). This example uses four yarn colors, with the background remaining fixed while the logo colors change on each side. To further push the boundaries of our method, we knit two complex Van Gogh paintings using a three-color quantization and three yarn colors. This requires a total of nine unique tiles. Although the results may not be extremely crisp, the different views are distinct and the images can be detected (Figure 10 (d)). This demonstrates the potential of our technique, which can be further enhanced with thinner yarn or larger knitted pieces for increased resolution.

7.3 Fabrication

We have implemented our knitting pattern to machine instruction pipeline, which parses a logical illusion design into a physical design, then into the KnitGraph data structure, then schedules and compiles it to the Knitout assembly language. To fabricate these patterns, we compile to Shima Seiki's proprietary KnitPaint format and knit it on a 7-gauge SWG091N2 machine. Our patterns ranged from about fifty minutes (ECG pulse and heart) to six hours (Van Gogh). The fabrication time for single-view patterns is determined by the size, and in double-view, determined by pattern complexity. In addition to the many knitting results we have illustrated above, we also show in Figure 7b what happens when each of the key features of our fabrication algorithm is not applied, demonstrating their importance.

8 LIMITATIONS AND FUTURE WORK

Our work presents the first tool that allows users to explore the space of illusion knitting where more than one image can be hidden, opening up several avenues for future work.

First, we only considered the limited space of illusions achieved through knits and purls on a knitting machine (Section 4). More complex stitch types and other fabrication techniques expand the possibilities for illusion knitting. For example, knitting a color change into a purl bump can partially achieve different colors on each side of the bump. New design tools and interfaces would be essential to better understand this more complex design space and to make design decisions that trade off different aspects, e.g., physical limitations of yarns and fabrication methods, and visual complexity.

Second, there are other colorwork techniques that could be combined with illusion knitting, and we did not fully explore the space. We also did not consider sliders when combining colorwork and textures, which could catch yarn for intarsia and for anchoring. It calls for more complex scheduling algorithms where the aesthetics of the knitted piece also becomes a consideration.

Our knitted pieces tended to have small areas of dropped stitches. We believe this is due to the many extra transfers incurred. Any time a transfer occurs, there is a chance loops will fall off the needle. Further investigation into reducing transfers would be very valuable, improving the percent of dropped stitches.

Finally, our renderer assumes that each pixel is seen at the same viewing angle. In real life, we view the closer part of the image at a greater angle, which means for double-view illusions, closer features in the opposite image may be visible. Future work could seek to augment the renderer, and further explore it to leverage, for example, our soft constraints, observations, and other perception-related objectives outside of MSE to achieve interesting effects.

In general, we believe that illusion knitting presents interesting challenges in computational knitting and can be a great case study for developing novel methods in pattern design, pattern scheduling, and user control. Our work serves as a first framework for working with more general illusion knitting and we hope to continue expanding its functionalities.

9 CONCLUSION

We develop a theoretical understanding of illusion knitting with the key insight that both the microgeometry that underpins illusions and the principles behind illusions can be expressed as constraints. Combined with practices from artists in the craft, we build a design tool for creating illusion knitting patterns that largely reduces illusion design to familiar image editing tasks. To fabricate these complex designs combining colorwork and texture, we introduce novel machine knitting scheduling techniques. We develop completely new kinds of illusion designs and establish a foundation for the community to further explore illusion knitting.

REFERENCES

Lea Albaugh, Scott Hudson, and Lining Yao. 2019. Digital Fabrication of Soft Actuated Objects by Machine Knitting. In *Extended Abstracts of the 2019 CHI Conference on Human Factors in Computing Systems*. ACM, Glasgow Scotland Uk, 1–4. https://doi.org/10.1145/3290607.3313270

Lea Albaugh, Scott E Hudson, and Lining Yao. 2023. Physically Situated Tools for Exploring a Grain Space in Computational Machine Knitting. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*. ACM, Hamburg Germany, 1–14. https://doi.org/10.1145/3544548.3581434

Lea Albaugh, James McCann, Scott E. Hudson, and Lining Yao. 2021. Engineering Multifunctional Spacer Fabrics Through Machine Knitting. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–12. https://doi.org/10.1145/3411764.3445564

Marc Alexa and Wojciech Matusik. 2011. Images from Self-Occlusion. In *Proceedings of the International Symposium on Computational Aesthetics in Graphics, Visualization, and Imaging* (Vancouver, British Columbia, Canada) *(CAe '11)*. Association for Computing Machinery, New York, NY, USA, 17–24. https://doi.org/10.1145/2030441.2030445

Marc Alexa and Wojciech Matusik. 2012. Irregular pit placement for dithering images by self-occlusion. *Computers & Graphics* 36, 6 (2012), 635–641.

Amit Bermano, Ilya Baran, Marc Alexa, and Wojciech Matusk. 2012. ShadowPix: Multiple Images from Self Shadowing. *Computer Graphics Forum* 31, 2pt3 (May 2012), 593–602. https://doi.org/10.1111/j.1467-8659.2012.03038.x

Teresa Harmon. 2011. *A Guide to Illusion Knitting: It's not magic, it's just fun!* CreateSpace Independent Publishing Platform, US.

Megan Hofmann, Lea Albaugh, Ticha Sethapakadi, Jessica Hodgins, Scott E. Hudson, James McCann, and Jennifer Mankoff. 2019. KnitPicking Textures: Programming and Modifying Complex Knitted Textures for Machine and Hand Knitting. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. ACM, New Orleans LA USA, 5–16. https://doi.org/10.1145/3332165.3347886

Megan Hofmann, Jennifer Mankoff, and Scott E. Hudson. 2020. KnitGIST: A Programming Synthesis Toolkit for Generating Functional Machine-Knitting Textures. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. ACM, Virtual Event USA, 1234–1247. https://doi.org/10.1145/3379337.3415590

Vivian Hoxbro. 2004. *Shadow Knitting*. Interweave, Loveland, Colo.

Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. 2008. Knitting a 3D Model. *Computer Graphics Forum* 27, 7 (Oct. 2008), 1737–1743. https://doi.org/10.1111/j.1467-8659.2008.01318.x

Benjamin Jones, Yuxuan Mei, Haisen Zhao, Taylor Gotfrid, Jennifer Mankoff, and Adriana Schulz. 2022. Computational Design of Knit Templates. *ACM Transactions on Graphics* 41, 2 (April 2022), 1–16. https://doi.org/10.1145/3488006

Alexandre Kaspar, Kui Wu, Yiyue Luo, Liane Makatura, and Wojciech Matusik. 2021. Knit sketching: from cut & sew patterns to machine-knit garments. *ACM Transactions on Graphics* 40, 4 (Aug. 2021), 1–15. https://doi.org/10.1145/3476576.3476614

Jenny Lin and James McCann. 2021. An Artin Braid Group Representation of Knitting Machine State with Applications to Validation and Optimization of Fabrication Plans. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, Xi'an, China, 1147–1153. https://doi.org/10.1109/ICRA48506.2021.9562113

Jenny Lin, Vidya Narayanan, Yuka Ikarashi, Jonathan Ragan-Kelley, Gilbert Bernstein, and James McCann. 2023. Semantics and Scheduling for Machine Knitting Compilers. *ACM Trans. Graph.* 42, 4 (Aug. 2023), 17. https://doi.org/10.1145/3592449 In Press.

Jenny Lin, Vidya Narayanan, and James McCann. 2018. Efficient transfer planning for flat knitting. In *Proceedings of the 2nd ACM Symposium on Computational Fabrication*. ACM, Cambridge Massachusetts, 1–7. https://doi.org/10.1145/3213512.3213515

Zishun Liu, Xingjian Han, Yuchen Zhang, Xiangjia Chen, Yu-Kun Lai, Eugeni L. Doubrovski, Emily Whiting, and Charlie C. L. Wang. 2021. Knitting 4D garments with elasticity controlled for body motion. *ACM Transactions on Graphics* 40, 4 (Aug. 2021), 1–16. https://doi.org/10.1145/3450626.3459868

Yiyue Luo, Kui Wu, TomÃ¡s Palacios, and Wojciech Matusik. 2021. KnitUI: Fabricating Interactive and Sensing Textiles with Machine Knitting. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. ACM, Yokohama Japan, 1–12. https://doi.org/10.1145/3411764.3445780

Yiyue Luo, Kui Wu, Andrew Spielberg, Michael Foshey, Daniela Rus, TomÃ¡s Palacios, and Wojciech Matusik. 2022. Digital Fabrication of Pneumatic Actuators with Integrated Sensing by Machine Knitting. In *CHI Conference on Human Factors in Computing Systems*. ACM, New Orleans LA USA, 1–13. https://doi.org/10.1145/3491102.3517577

James McCann, Lea Albaugh, Vidya Narayanan, April Grow, Wojciech Matusik, Jennifer Mankoff, and Jessica Hodgins. 2016. A compiler for 3D machine knitting. *ACM Transactions on Graphics* 35, 4 (July 2016), 1–11. https://doi.org/10.1145/2897824.2925940

Niloy J. Mitra and Mark Pauly (Eds.). 2009. Shadow art. *ACM Transactions on Graphics* 28, 5 (2009), 1–7. https://doi.org/10.1145/1661412.1618502

Georges Nader, Yu Han Quek, Pei Zhi Chia, Oliver Weeger, and Sai-Kit Yeung. 2021. KnitKit: a flexible system for machine knitting of customizable textiles. *ACM Transactions on Graphics* 40, 4 (Aug. 2021), 1–16. https://doi.org/10.1145/3450626.3459790

Sadako Nakamura, Ryo Tsugawa, Keiko Kitao, and et al. 1982. かくし絵ニット:やさしい棒針編のマジック*[Hidden Picture Knit : The Magic of Gentle Needle Knitting]*. Nihon Vogue, Tokyo.

Vidya Narayanan, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James Mccann. 2018. Automatic Machine Knitting of 3D Meshes. *ACM Transactions on Graphics* 37, 3 (June 2018), 1–15. https://doi.org/10.1145/3186265

Vidya Narayanan, Kui Wu, Cem Yuksel, and James McCann. 2019. Visual knitting machine programming. *ACM Transactions on Graphics* 38, 4 (Aug. 2019), 1–13. https://doi.org/10.1145/3306346.3322995

orochi. 2006. かくし絵ニット・パターン作成ツールVersion 1.1 [Hidden Picture Knit・Pattern Making Tool]. https://orochiknit.com/archive/kt001.html

Marios Papas, Wojciech Jarosz, Wenzel Jakob, Szymon Rusinkiewicz, Wojciech Matusik, and Tim Weyrich. 2011. Goal-based Caustics. *Computer Graphics Forum* 30, 2 (2011), 503–511. https://doi.org/10.1111/j.1467-8659.2011.01876.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2011.01876.x

Hao Peng, Lin Lu, Lin Liu, Andrei Sharf, and Baoquan Chen. 2019. Fabricating QR codes on 3D objects using self-shadows. *Computer-Aided Design* 114 (2019), 91–100.

Thiago Pereira, Carolina LA Paes Leme, Steve Marschner, and Szymon Rusinkiewicz. 2017. Printing anisotropic appearance with magnetic flakes. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 1–10.

Maxine Perroni-Scharf and Szymon Rusinkiewicz. 2023. Constructing Printable Surfaces with View-Dependent Appearance. In *Special Interest Group on Computer Graphics and Interactive Techniques Conference Conference Proceedings (SIGGRAPH '23 Conference Proceedings)*. ACM, New York, NY, USA, 10. In Press.

Mariana Popescu, Matthias Rippmann, Tom Van Mele, and Philippe Block. 2018. Automated Generation of Knit Patterns for Non-developable Surfaces. In *Humanizing Digital Reality: Design Modelling Symposium Paris 2017*, Klaas De Rycke, Christoph Gengnagel, Olivier Baverel, Jane Burry, Caitlin Mueller, Minh Man Nguyen, Philippe Rahm, and Mette Ramsgaard Thomsen (Eds.). Springer, Singapore, 271–284. https://doi.org/10.1007/978-981-10-6611-5_24

Christian Regg, Szymon Rusinkiewicz, Wojciech Matusik, and Markus Gross. 2010. Computational highlight holography. *ACM Transactions on Graphics (TOG)* 29, 6 (2010), 1–12.

Kaisei Sakurai, Yoshinori Dobashi, Kei Iwasaki, and Tomoyuki Nishita. 2018. Fabricating reflectors for displaying multiple images. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 1–10.

Yuliy Schwartzburg, Romain Testuz, Andrea Tagliasacchi, and Mark Pauly. 2014. High-contrast computational caustic design. *ACM Transactions on Graphics* 33, 4 (July 2014), 74:1–74:11. https://doi.org/10.1145/2601097.2601200

Xavier Snelgrove, Thiago Pereira, Wojciech Matusik, and Marc Alexa. 2013. Parallax Walls: Light fields from occlusion on height fields. *Computers & graphics* 37, 8 (2013), 974–982.

Debbie Stoller. 2004. *Stitch 'n Bitch: The Knitter's Handbook*. Workman Publishing Company, New York.

Tim Weyrich, Pieter Peers, Wojciech Matusik, and Szymon Rusinkiewicz. 2009. Fabricating microgeometry for custom surface reflectance. *ACM Transactions on Graphics (TOG)* 28, 3 (2009), 1–6.

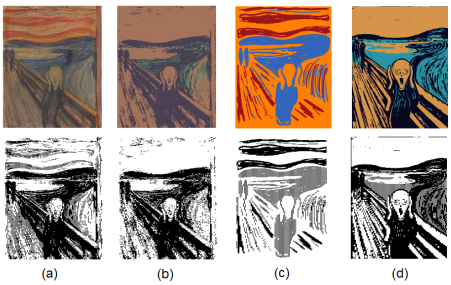
WoollyThoughts. 2023. Information. http://www.illusionknitting.woollythoughts.com/information.html

Kui Wu, Xifeng Gao, Zachary Ferguson, Daniele Panozzo, and Cem Yuksel. 2018. Stitch meshing. *ACM Transactions on Graphics* 37, 4 (Aug. 2018), 1–14. https://doi.org/10.1145/3197517.3201360

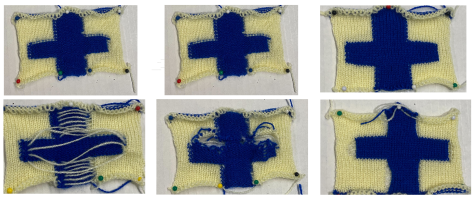
Kui Wu, Hannah Swan, and Cem Yuksel. 2019. Knittable Stitch Meshes. *ACM Transactions on Graphics* 38, 1 (Feb. 2019), 1–13. https://doi.org/10.1145/3292481

Cem Yuksel, Jonathan M. Kaldor, Doug L. James, and Steve Marschner. 2012. Stitch meshes for modeling knitted clothing with yarn-level detail. *ACM Transactions on Graphics* 31, 4 (Aug. 2012), 1–12. https://doi.org/10.1145/2185520.2185533

Jiani Zeng, Honghao Deng, Yunyi Zhu, Michael Wessely, Axel Kilian, and Stefanie Mueller. 2021. Lenticular Objects: 3D Printed Objects with Lenticular Lens Surfaces That Can Change Their Appearance Depending on the Viewpoint. In *The 34th Annual ACM Symposium on User Interface Software and Technology* (Virtual Event, USA) *(UIST '21)*. Association for Computing Machinery, New York, NY, USA, 1184–1196. https://doi.org/10.1145/3472749.3474815



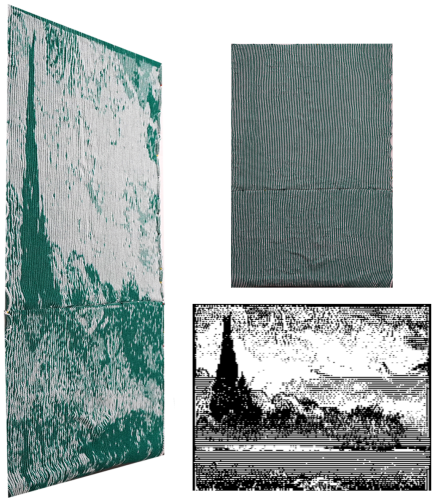
(a) Iterating on quantizations of the Scream. Above is the input image; below is the rendered generated pattern.



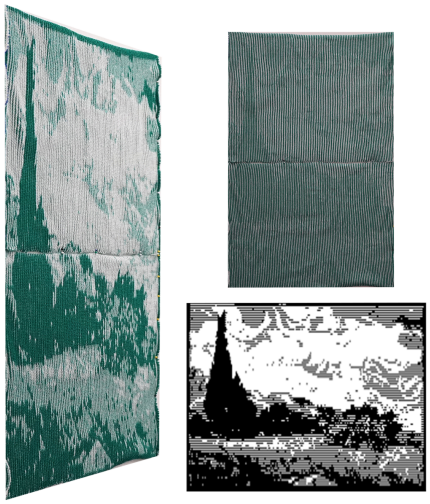
(b) Knitting with (top) and without (bottom) each contribution: float avoidance (left), path planning (middle), catching (right) creates floats, dropped stitches, and unconnected knits respectively.

(c) Scream (quantized by diffusion model)

(d) Scream (quantized by hand)



(a) Cypress (optimization generated)



(b) Cypress (tool generated)

Fig. 8. Single-view examples. When viewed head-on, each of these knit pieces look like striped images (top right), but when viewed from the side (left), they resemble and quantized inputs (bottom right).

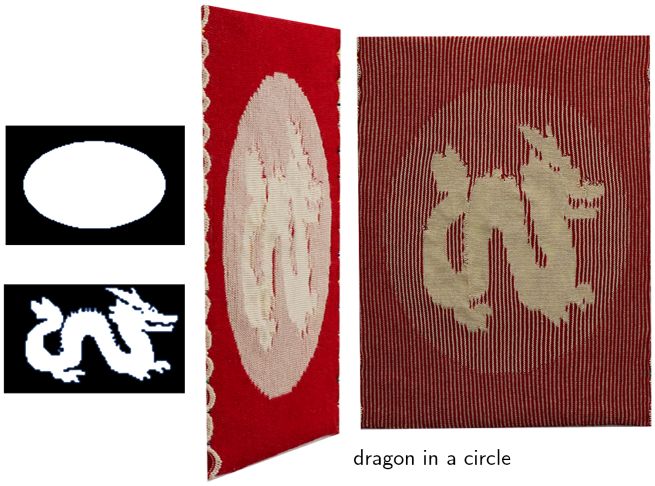


Fig. 9. A single-view head-on illusion and its inputs. Viewed from the front, the dragon is visible, but when viewed at an angle, the larger circle illusion engulfs it and only a circle is visible.

(a) ECG pulse and heart

(b) SIGGRAPH ASIA 2023

(c) SIGGRAPH and SIGGRAPH ASIA Logos

(d) Van Gogh's Sunflowers and Self-portrait

Fig. 10. Double-view examples, pictured with their inputs, and in (a) and (d), the rendered prediction of their patterns. In each of these examples, the image from the left and right viewing angle differs, as depicted.