# Customized 3D and 4D Design for Machine Knitting

Liu, Z.

**DOI**
10.4233/uuid:de97d58f-5d0e-47a6-b649-827062216fae

**Publication date**
2023

**Document Version**
Final published version

**Citation (APA)**
Liu, Z. (2023). *Customized 3D and 4D Design for Machine Knitting*. [Dissertation (TU Delft), Delft University of Technology]. https://doi.org/10.4233/uuid:de97d58f-5d0e-47a6-b649-827062216fae

Customized 3D and 4D Design for Machine Knitting Zishun Liu

ISBN 978-94-6384-423-9

# Customized 3D and 4D Design for Machine Knitting

Zishun Liu

# CUSTOMIZED 3D AND 4D DESIGN FOR MACHINE KNITTING

## Dissertation

for the purpose of obtaining the degree of doctor
at Delft University of Technology
by the authority of the Rector Magnificus prof. dr. ir. T.H.J.J. van der Hagen
Chair of the Board for Doctorates
to be defended publicly on
Monday, 20 March 2023 at 12:30 o'clock

by

**Zishun LIU**

Master of Science,
University of Science and Technology of China, Hefei, China.
born in Liaocheng, China.

This dissertation has been approved by the promotors.

Composition of the doctoral committee:

| | |
|---|---|
| Rector Magnificus | chairperson |
| Prof. dr. C.C.L. Wang | Delft University of Technology, promotor |
| Prof. dr. ir. J.M.P. Geraedts | Delft University of Technology, promotor |
| Dr. ir. E.L. Doubrovski | Delft University of Technology, copromotor |
| | |
| *Independent members:* | |
| Prof. dr. ir. K.M.B. Jansen | Delft University of Technology |
| Dr. K. Hildebrandt | Delft University of Technology |
| Prof. dr. Y.-K. Lai | Cardiff University, U.K. |
| Prof. C. Rizzi | University of Bergamo, Italy |
| Prof. dr. P. Vink | Delft University of Technology (reserve member) |


This research was partially supported by the China Scholarship Council.

*Keywords:* knitting; computational design; computational fabrication; 3D garment; 4D garment
*Printing:* Ridderprint, www.ridderprint.nl
*Cover:* Photo of a knitted book cover designed by the author.



ISBN 978-94-6384-423-9

An electronic version of this dissertation is available at
http://repository.tudelft.nl/.

Dedicated to my grandparents.

# CONTENTS

# SUMMARY

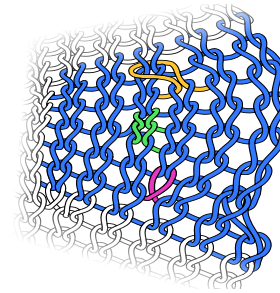
Garments, one of the human basic needs, were customized and handmade before the Industrial Revolution. After the realization of mass production, the cost of a piece of clothing became lower, but some disadvantages arose. Garments were no longer made to measure and overproduction caused environmental problems. The new developments in digital garment design and digital customization target addressing these limitations.

Garments are mainly made from two types of fabrics, namely woven and knitted fabric. Woven fabric, which is constructed with straight threads, is made into garments by cutting and sewing. Knitting warps a single yarn as loops and the interlaced loops form a fabric. Knitting a garment has several advantages over the cutting-and-sewing method. Instead of fabricating sheets that are later cut into patterns, knitting can directly manufacture clothing with significantly less waste, which resembles the advantage of additive manufacturing over subtractive methods. Multiple materials can be knitted in varied patterns by machines automatically. The 3D shape of knits can also be highly customized by machines with less human labor compared to woven garments. These advantages make knitting more suitable for digital customization compared to cutting-and-sewing.

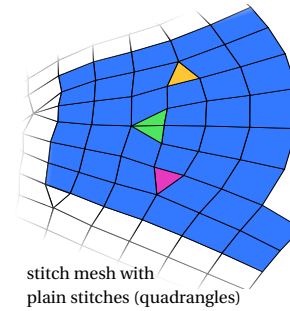
The computational design of knitting attracted increased attention in recent years. In this dissertation, we consider the customized design and fabrication of 3D and 4D garments as knitwears. The 3D knitwear fits the target human body, and the 4D knitwear also considers comfort during body movement. The main research question (**RQ**) is: *How to design customized 3D and 4D knitwear and generate instructions for a digital knitting machine?*

The main research question is answered by studying three sub-questions (RQ1–3), the first one of which is (**RQ1**) How to design a knitwear and its machine knitting instructions for a given 3D shape?

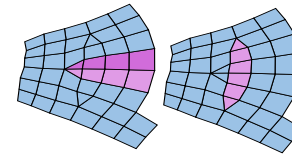
We start with a flattening-based design, which is closely related to the method used by garment designers. Designers generate 3D (non-planar) garments from planar fabrics by sewing some folded/cut-out parts, which are referred to as darts, to tailor the garment to the wearer's shape. The placement of darts has a big influence on the 3D shape of garments. Thus we pose the guiding question *"Where to put darts?"* to direct the development of this approach. We add darts by analyzing the distortion of flattening, i.e., compression or stretching, and considering the knittability constraints. The distortion is reduced after adding more darts. We convert the flattened shape to valid knitting instructions, which are arranged as a grid of stitches. Rows and columns in the grid intersecting with the darts are named short-rows and short-columns respectively. They are knitted with short-row and short-column (stitch transfer) techniques to close the darts in the knitting process. However, the accuracy and usability of this approach are limited. We were encouraged to develop more in-depth approaches to address accuracy and usability.



a 3D knitting structure



stitch mesh with
plain stitches (quadrangles)
shaping stitches (triangles)



short-rows short-column

Towards a more automated approach compared to the flattening-based design, we developed the direct 3D design approaches by stitch mesh generation, to further study **RQ1**. The stitch mesh represents the knitting structure as a polyhedral object on which each polygonal facet corresponds to a knitting stitch. It approximates the target shape using plain stitches and shaping stitches. The shaping stitches include the beginning and ending stitches of short-rows and short-columns. Shaping stitches are crucial for the fabric's curvature while pure plain stitches only generate planar fabrics. Then the guiding question is *Where to put shaping stitches?* The shaping stitches are corresponding to the 3D knitting techniques, i.e., short-row and short-column knitting. We generate knittable stitch meshes with the help of governing fields. When using a geodesic distance field as the governing field, only short-rows are generated. When using a Laplacian-based time field, the stitch mesh can consist of both short-rows and short-columns of stitches. Short-row knitting is simple but may generate some seams that need to be sewn afterwards. Physical fabrication for a short-column of stitches is based on a technique that transfers the loops of stitches between needle-beds at the rows of its beginning and ending stitches (called transfer stitches). This transfer technique, however, can often lead to manufacturing problems such as miss-transfer, yarn stretching, yarn abrasion, etc. Therefore, the number of transfer stitches needs to be minimized for reliable knitting. We tackle this challenge by formulating it as a field optimization problem with boundary conditions as variables. We argue that a field with optimized divergence (more parallel column curves) will become a governing field with fewer transfer stitches. We developed a modified stitch mesh generation algorithm that is coupled with governing fields more tightly, and Bayesian optimization is employed to implement the computation framework. We propose a graph-based algorithm to place shaping stitches on columns and generate knittable stitch meshes that can accurately capture the 3D shape of a garment. The performance of our approach for generating reliable knitting plans has been demonstrated in physically knitted examples.

Next, we present a new computational pipeline for designing and fabricating 4D garments as knitwear that considers comfort during body movement, which answers the second research question (**RQ2**): How to design a motion-aware 4D knitwear and generate its machine instructions? Compared to 3D garments tailored for body shape, the additional dimension in 4D garments is the time-variant effect of garments during motion. This is achieved by careful control of elasticity distribution to reduce uncomfortable pressure and unwanted sliding caused by body motion. Basically, flexible structures

at body joints will reduce pressure, and firm structures at limbs will reduce sliding. The guiding question is *Where to put elastic structures?* We exploit the ability to knit patterns in different elastic levels by single-jersey jacquard with two yarns with different elasticities. We design the distribution of elasticity for a garment by physics-based computation, the optimized elasticity of the garment is then converted into instructions for a digital knitting machine. A tiling algorithm assigns jacquard patterns on the stitch mesh to realize the designed distribution of elasticity. The effectiveness of our approach is verified by simulation results and physical specimens fabricated by knitting machines.

Now we have prepared the knitwear design methods for the intended 3D shapes and 4D motion sequences. The third research question (**RQ3**) is How to capture an individual's body models in different poses for 3D/4D knitwear customization? Traditionally, people measure body dimensions for garment customization. However, complex body shapes and motions can not be accurately captured by a few measurements. We study three types of digital human body modeling methods using different sensors.

- A photogrammetry approach using an expensive multi-camera system.
- A simple setup of a pair of depth sensors (Kinect).
- An artificial intelligence (AI) based approach using a single RGB image.

We can extract the human body information for knitting, using one of these approaches. 4D motion can be obtained by registering a sequence of 3D postures. At last, we apply the developed knitwear design methods to the captured human bodies.

In this dissertation, we researched computational knitwear design methods. We considered not only 3D fitting but also comfort during motion (4D). Our research can be applied in garment production (especially mass customization) or other knitting applications. Garment designers and other industrial designers can use the proposed methods to generate knitting instructions for free-form 3D surfaces. Our 4D design method helps designers place elastic or other varied knitting structures while keeping the intended 3D shape. This dissertation presents new perspectives on computational approaches to existing manufacturing techniques. It also provides enough details to further develop such design systems to be applied in practice.

# SAMENVATTING

Kleding, een van de basisbehoeften van de mens, werd vóór de industriële revolutie met de hand en op maat gemaakt. Na de totstandkoming van massaproductie werden de kosten van een kledingstuk lager, maar ontstonden er ook nadelen. Kleding op werd niet meer op maat gemaakt en er ondstond overproductie met grote gevolgen voor het milieu. Nieuwe ontwikkelingen in digitaal ontworpen en digitaal geproduceerd maatwerk van kleding richten zich op het aanpakken van deze beperkingen.

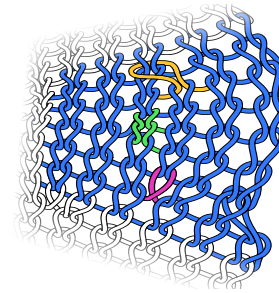
Kledingstukken worden voornamelijk gemaakt van twee soorten stoffen, namelijk geweven en gebreide stof. Geweven stof, die is opgebouwd uit rechte draden wordt door knippen en naaien tot kledingstukken gemaakt. Bij breien wordt een enkel garen vervormd tot lussen die in elkaar verstrengelt een stof vormen. Breien heeft verschillende voordelen ten opzichte van de knip- en naaimethode van geweven stof. In plaats van het vervaardigen van vlakke stoffen die vervolgens in patronen worden geknipt, biedt breien de mogelijkheid om kleding te vervaardigen terwijl aanzienlijk minder afval wordt geproduceerd. Dit lijkt op de voordelen van additieve productie ten opzichte van subtractieve methoden. Meerdere materialen kunnen door machines automatisch in verschillende patronen gebreid worden. De 3D geometrie van de breisels kan ook op maat worden gemaakt met minder handarbeid vergeleken met gewoven kledingstukken. Deze voordelen maken breien geschikter voor digitaal maatwerk vergeleken met de knip- en naaimethode.

De afgelopen jaren heeft computationeel ontwerpen van breiwerk steeds meer aandacht gekregen. In dit proefschrift behandelen we maatwerk voor ontwerp en fabricage van 3D- en 4D-breiwerk. 3D-breigoed is op maat gemaakt op de lichaamsvorm van de gebruiker, terwijl 4D-breigoed ook rekening houdt met comfort van de gebruiker tijdens beweging. De centrale onderzoeksvraag (**RQ**, *research question*) is: *Hoe kunnen we op maat gemaakt 3D- en 4D-breigoed ontwerpen en machineinstructies genereren voor een digitale breimachine?*

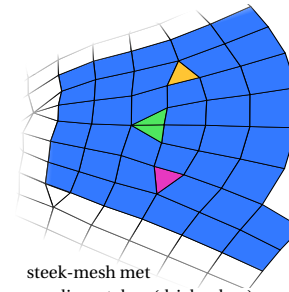
De hoofdvraag van het onderzoek wordt beantwoord door drie deelvragen (RQ1–3) te bestuderen, waarvan de eerste is (**RQ1**) Hoe kunnen we een breiwerk en de machinebreiinstructies voor een bepaalde 3D-vorm ontwerpen?

We beginnen met op afvlakking gebaseerd ontwerp, dat nauw verwant is aan de methode die door kledingontwerpers wordt gebruikt. Ontwerpers genereren niet-vlakke 3D-kledingstukken van vlakke stoffen door gevouwen of uitgeknipte delen aan elkaar te naaien, die figuurnaden worden genoemd. De plaatsing van figuurnaden heeft een grote invloed op de 3D-vorm van de kleding. Dus stellen we de begeleidende vraag *"Waar moeten we figuurnaden plaatsen?"* om de ontwikkeling van deze aanpak te sturen. We voegen figuurnaden toe door het analyseren van de verstoring van afvlakking, dat wil zeggen, compressie of uitrekken, en rekening houdend met de beperkingen van het breiproces. De verstoring wordt verminderd na het toevoegen van meer figuurnaden. We zetten de afgeplatte vorm om in geldige brei-instructies, die gerangschikt worden als een

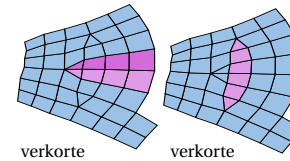
raster van steken. Rijen en kolommen in het raster die met de figuurnaad kruisen worden respectievelijk verkorte rijen en verkorte kolommen genoemd. Ze worden gebreid met een verkorte rij- en verkorte kolomtechniek (steekoverdracht) om de figuurnaden te sluiten tijdens het breiproces. De nauwkeurigheid en bruikbaarheid van deze aanpak zijn echter beperkt. We waren aangemoedigd om de volgende aanpakken te ontwikkelen om de nauwkeurigheid en bruikbaarheid te addresseren.



3D-breistructuur



steek-mesh met
reguliere steken (driehoeken)
vormgevende steken (vierhoeken)



verkorte rijen

verkorte kolommen

Voor een meer geautomatiseerde aanpak in vergelijking met het op afvlakking gebaseerde ontwerpen, hebben we directe 3D-ontwerpbenaderingen ontwikkeld die gebruik maken van een steek-mesh **RQ1**. De steek-mesh vertegenwoordigt de breistructuur als een veelvlakkig object waarop elk veelhoekig facet overeenkomt met een breisteek. Deze mesh benadert de doelvorm met reguliere steken en vormgevende steken. De vormgevende steken omvatten de eerste en laatste steken van verkorte rijen en kolommen. Vormgevende steken zijn cruciaal voor de kromming van de stof, terwijl reguliere steken alleen altijd vlakke stoffen genereren. Dan is de begeleidende vraag *Waar moeten we vormgevende steken plaatsen?* De vormgevende steken komen overeen met de 3D-breitechnieken, dat wil zeggen, breien van verkorte rijen en kolommen. We genereren breibare steek-meshes met behulp van referentievelden. Bij het gebruik van een geodetisch afstandsveld als referentieveld worden alleen verkorte rijen gegenereerd. Bij gebruik van een Laplaciaans tijdveld kan het steek-mesh bestaan uit zowel verkorte rijen als verkorte kolommen van steken. Breien met korte rijen is eenvoudig, maar kan enkele naden genereren die achteraf moeten worden genaaid. Fysieke fabricage voor een verkorte kolom van steken is gebaseerd op een techniek die de lussen van steken overbrengt tussen naaldbedden aan de rijen van eerste en laatste steken (overdrachtssteken genoemd). Deze overdrachtstechniek leidt echter regelmatig tot fabricageproblemen zoals het verkeerd overbrengen, het uitrekken van garen, schuren van garen, enz. Daarom moet voor de betrouwbaarheid van het breiden het aantal overdrachtssteken geminimaliseerd worden. We pakken deze uitdaging aan door het te formuleren als een veldoptimalisatieprobleem met randvoorwaarden als variabelen. We beargumenteren dat een veld met geoptimaliseerde divergentie (meer parallelle kolom curven) een referentieveld met minder overdrachtssteken zal worden. We hebben een aangepast algoritme ontwikkeld voor het genereren van steek-mesh dat beter is gekoppeld aan de referentievelden, en Bayesiaanse optimalisatie wordt gebruikt om het berekeningskader te implementeren. We stellen een op grafen gebaseerd algoritme voor om vormgevende steken op kolommen te plaatsen en breibare steek-meshes te genereren

die de 3D-vorm van een kledingstuk nauwkeurig kunnen vastleggen. De prestaties van onze aanpak voor het genereren van betrouwbare breiplannen zijn aangetoond in fysiek gebreide voorbeelden.

Vervolgens presenteren we een nieuwe computationele workflow voor het ontwerpen en vervaardigen van 4D-kledingstukken als breigoed dat rekening houdt met comfort tijdens lichaamsbeweging. Dit beantwoord de tweede onderzoeksvraag (**RQ2**): Hoe kunnen we 4D breigoed ontwerpen dat rekening houdt met lichaamsbeweging en de bijbehorende maachineinstructies genereren? Vergeleken met 3D-kledingstukken die zijn afgestemd op de lichaamsvorm, is de extra dimensie in 4D-kledingstukken het tijdvariante effect van kledingstukken tijdens het bewegen. Dit wordt bereikt door zorgvuldige controle van de elasticiteitsverdeling om ongemakkelijke druk en ongewenst schuiven veroorzaakt door lichaamsbeweging te verminderen. Kortom, flexibele structuren bij lichaamsgewrichten zullen de druk verminderen, en stijvere structuren bij ledematen zullen glijden verminderen. De begeleidende vraag is *Waar moeten we elastische structuren plaatsen?* Wij maken gebruik van de mogelijkheid om patronen te breien met verschillende elastischiteiten door middel van single-jersey jacquard met twee garens. Wij ontwerpen de verdeling van elasticiteit voor een kledingstuk door op fysica-gebaseerde berekeningen, de geoptimaliseerde elasticiteit van het kledingstuk wordt vervolgens omgezet in instructies voor een digitale breimachine. Een algoritme voor betegeling wijst jacquardpatronen aan op de steek-mesh om de ontworpen verdeling van elasticiteit te realiseren. De effectiviteit van onze aanpak is geverifieerd door simulatieresultaten en fysieke proefstukken die door breimachines zijn vervaardigd.

Nu hebben we de breigoed ontwerpmethoden voor de beoogde 3D-vormen en 4D-bewegingssequenties voorbereid. De derde onderzoeksvraag is (**RQ3**) Hoe kunnen we de lichaamsmodellen van individuen in verschillende poses vastleggen voor het personaliseren van 3D/4D breigoed? Traditioneel meten mensen lichaamsafmetingen voor het aanpassen van kleding. Complexe lichaamsvormen en bewegingen kunnen echter niet nauwkeurig worden vastgelegd met enkele metingen. Er worden drie soorten methoden voor het modelleren van het menselijk lichaam bestudeerd.

- Een fotogrammetrie-aanpak met behulp van een duur multi-camera systeem.
- Een eenvoudige opstelling van twee diepte-sensoren (Kinect 2).
- Een op kunstmatige intelligentie (KI, of artificiële intelligentie, AI) gebaseerde aanpak met behulp van een enkel RGB-beeld.

We kunnen de menselijke lichaamsinformatie voor breien, met behulp van een van deze aanpakken extraheren. 4D-beweging kan worden verkregen door een reeks 3D-houdingen te registreren. Tenslotte passen we de ontwikkelde breisels toe op de vastgelegde menselijke lichamen.

In dit proefschrift hebben we computationele methodes ontwikkeld voor het ontwerpen van breisels. We hebben niet alleen gekeken naar 3D-fitting maar ook naar comfort tijdens het bewegen (4D). Ons onderzoek kan worden toegepast in de productie van kleding (vooral massamaatwerk) of andere breitoepassingen. Kledingontwerpers en andere industriële ontwerpers kunnen de voorgestelde methoden gebruiken om breiinstructies te genereren voor 3D-oppervlakken met complexe en organische vormen. Onze

4D-ontwerpmethode helpt ontwerpers om elastische of andere gevarieerde breistructuren te plaatsen met behoud van de beoogde 3D-vorm. Dit proefschrift presenteert nieuwe perspectieven op computationele aanpakken voor bestaande maaktechnieken. Het biedt ook voldoende details om zulke ontwerpsystemen verder te ontwikkelen om toegepast te worden in de praktijk.

# GLOSSARY

**3D** Three dimensional; non-planar.

**3D garment** A garment that fits the 3D shape of the wearer's body.

**4D garment** Four-dimensional garment. It is a fitting 3D garment that also considers the wearer's comfort during body motion. The additional dimension is the time-variant effect of garments during motion.

**Course** A row of knitted stitches.

**Dart** A fold sewn into the fabric to take in ease and provide shape to a garment.

**Polygon mesh** (or mesh) A discrete representation of surfaces. It is a polyhedral object composed of polygonal facets.

**Polyline** A discrete representation of curves. It is a connected series of line segments.

**Shaping stitches** Special stitches that contribute to the 3D shaping of knitted fabrics. They are the ends of short-rows or short-columns that locate in the fabric interior.

**Short-column knitting** A knitting technique to produce 3D (non-planar) fabrics. In a short-column, at least one of its ends locates in the fabric interior.

**Short-row knitting** A knitting technique to produce 3D fabrics. In a short-row, at least one of its ends locates in the fabric interior.

**Stitch** The repeated element in knitting.

**Stitch mesh** A representation of knitting structures. It is a polygon mesh on which each facet represents a stitch.

**Wale** A column of knitted stitches.

# 0

# INTRODUCTION

## 0.1. RESEARCH PROBLEM AND MOTIVATION

Humans have a long history of knitting yarns into fabrics. Today, this technology is used for the mass production of garments using computerized knitting machines. Customized manufacturing has several advantages compared to mass production. It responds to customer needs [93] and creates value for customers. It reduces overproduction and has the potential to be more sustainable. Moreover, in the context of the recent worldwide pandemic of coronavirus disease 2019 (COVID-19), customized manufacturing and on-demand manufacturing can help companies respond rapidly to local demands and reduce reliance on global sources [183].

Knitting is one of the textile-making techniques, among weaving, crocheting, braiding, knotting, tufting, and non-weaving [81]. Garments are usually made from woven fabrics and knitted fabrics because they are more affordable for large-scale automation. The knitting method has several advantages over garment fabrication which uses woven fabrics. For 3D shaping, woven garments require cutting and sewing to construct *darts*. In this process, the cutting wastes materials and the sewing is labor-intensive. For multi-material garments, such as the 4D garments using materials with different elasticities, woven garments also require sewing patches from different materials. In both cases, knitting is much simpler. It minimizes material waste and maximizes the automation level in fabricating complex shapes with one or more types of materials, resembling additive manufacturing techniques [43], while cutting-and-sewing belongs to subtractive manufacturing. Moreover, the materials of knitting have the potential to be more sustainable than woven fabric [187, 60, 199, 198]. In summary, knitting requires much less human intervention and is more environment-friendly. Besides knits and woven fabrics, garments can also be made of non-woven fabrics, for example, in medical applications. The non-woven fabrics are usually planar, which are made into garments using the cutting-and-sewing process as woven fabrics, thus sharing the same limitations.

For customized garment design, an important question is to design for the target shape, such as the customer's body shape [201]. Garments that fit an individual's 3D

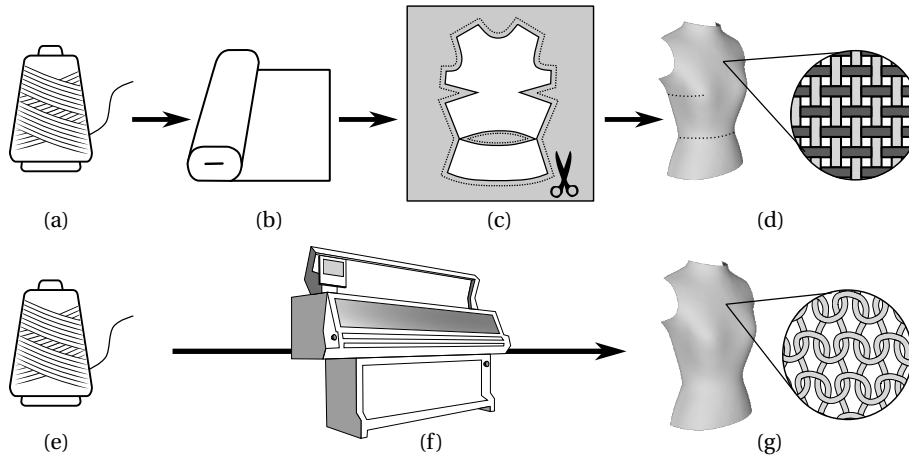



Figure 0.1: Comparison between garment making by weaving (top row, (a–d)) or knitting (bottom row, (e–g)). Woven fabrics (b) constructed with threads (a) are cut (c) and sewn into garments (d). The gray regions in (c) are wasted. Using a computerized knitting machine (f), yarns are knitted into knitwears more directly, with less material waste and less human intervention. Structures of weaving and weft knitting are shown in (d) and (g) respectively.

body shape are referred to as *3D garments*. It should be body-tight and provide uniform stress. Furthermore, garments that not only fit an individual's body but also fit the body motion are studied, we refer to as *4D garments* by taking the motion (i.e., the time-varied 3D body pose) as 4D information. 4D garments consider the user's comfort during motion by designing the distribution of knitting structures with different elasticity levels.

Nowadays, most knitting machines used in industry are computer-controlled and could allow on-demand manufacturing of custom knitwear. However, some essential steps in the workflow are still lacking. The research question (**RQ**) is *how to design customized 3D and 4D garments and generate the machine instructions?* The advantages of 3D/4D garments and machine knitting encourage us to develop the design tools for this question. This research is also aligned with Industry 4.0 [94], which requires digital modeling of the physical fabrication (cyber-physical systems), adaptation to human needs (but not adapting human needs to manufacturing systems), and corporate social responsibility (sustainability and resource-efficiency).

This dissertation aims to develop 3D and 4D knitting design methods for garment designers and other industrial designers. The methods can be used to accurately and robustly knit other free-from 3D surfaces besides garments. The 4D elastic structure placement method allows placing other varied knitting structures without distorting the intended shape. The proposed computational design methods will enrich the designers' toolbox to realize their idea and provide some new perspectives beyond trial-and-error. Henry Ford, the chief developer of the assembly line technique of mass production, once said "Any customer can have a car painted any colour that he wants so long as it is black" [50]. With the customized 3D and 4D garment design techniques for machine knitting, the envisioned future will be: *The customers can have garments with any style they want, as long as it is knitted.*

## 0.2. RELATED WORK

Researchers have been studying knitting from different perspectives. Materials scientists study the properties of the fibers and yarns for knitting as well as the effect of knit structures [5] and finishing treatments [152]. Mechatronics engineers develop knitting machines for efficient manufacturing [120]. They also develop wearable smart devices using knitting. Knitwear designers create knitted fabric from the view of fine arts and fashion. Spencer, 2001 [177] is a comprehensive textbook covering different aspects of knitting technology. Au, 2011 [5] and Maity et al., 2022 [118] discussed more recent advancements in knitting technology and its applications. There are also research works on the business model of on-demand knitting or mass customization [93].

This dissertation develops computational design tools, which belong to the domain of computer-aided design (CAD) and computer graphics (CG). This work mostly relies on the geometric computing tools developed by the CAD and CG communities. Besides geometry, physical simulation tools are also used (Chapter 4). Meanwhile, we consider manufacturing constraints throughout the research. Related works from different aspects are reviewed below.

### DIGITAL 3D KNITTING

Igarashi et al., 2008 [65] and Igarashi et al., 2008 [66] are two of the earliest computational knitting design methods for an input 3D model, which generate hand knitting instructions. The thesis of Underwood, 2009 [192] comprehensively studied the machine knitting techniques for 3D shaping, especially the implementations on Shima Seiki's WholeGarment® machines. The name of *stitch mesh* was first proposed by Yuksel et al., 2012 [221] to build yarn-level models of complex knitted garments. Narayanan et al., 2018 [134] presented the first computational approach that can transform 3D meshes into instructions for a computer-controlled knitting machine. The interactive method presented in Wu et al., 2019 [215] can generate stitch meshes that are ensured to be knittable by hand. Some of the representative works are reviewed here. More details will be discussed in Chapter 3.

### INTERACTIVE DESIGN SYSTEMS

Some interactive design systems were also developed for digital knitting, including a general visual programming interface for creating 3D objects with complex surface finishes (Narayanan et al., 2019 [135]), a system allowing the composition of knitted shapes and stitch patterns (Kaspar et al., 2019 [82]), a method to design periodic yarn-level knitting patterns (Leaf et al., 2018 [95]), and an interface that provides coupled knitting code and topological visualization of the knitting design (Yu and McCann, 2020 [220]). Kaspar et al., 2021 [84] developed a design tool inspired by traditional garment making based on cutting and sewing.

Knitwear design tools need to support multi-structure (like ribs) and multi-color knitting so that they can be effectively used by designers. Some design systems [80, 61, 129] enable knitting varied textures by varying stitch structures or yarns with different colors. In the knitting industry, Shima Seiki and Stoll, the two largest knitting machine manufacturers, have their own software suites [165, 179] that provide rich capabilities of knitting design.

### PHYSICAL SIMULATION OF KNITTED FABRICS

Given a knits design, it is straightforward to generate the interlocking relation (i.e., the topological relation) between its stitches. However, predicting the realistic geometry of the warped yarn requires physical simulation. Such methods are described in [123, 77, 221]. Among them, [77] accurately predicts the curling phenomenon on the edges of single-jersey knitting structures. Besides the local behavior of stitches, physical simulation methods [137, 178] also study the knits as fabrics and garments.

## 0.3. KNOWLEDGE GAP

Existing works proposed different approaches to knitting 3D fabrics automatically using digital knitting machines. However, there are still several major challenges to realizing accurate, reliable 3D knitting and motion-aware 4D knitting, despite the growing research in related areas. These issues defined the gap to be filled.

First, the knitting result may not reproduce the intended 3D shape. The stitches generated by existing design methods are usually deformed: in general, it is difficult to design a knitting pattern that lies on the target shape and keeps every stitch identical to its rest shape, while complying with the knittability constraints at the same time. There is some distortion between the intended shape and the design's rest shape (without external force applied), which should be reduced to generate perfect-fit 3D garments.

Second, there is a need to study the reliability of generated machine knitting instructions in the design process. Stitch transfer, a type of knitting machine operation, is not stable in some cases. The yarn stretching and abrasion harm the knitting process and the fabric quality. However, this operation is necessary for short-column shaping for 3D fabrics. Therefore, reliability should be studied when stitch transfer is heavily involved.

Third, garment stress and sliding affect the wearer's comfort during body motion. It has not been studied to consider this time-varying issue for knitwear design in the existing literature. This raised the question of how to design knitwear with locally varying elasticity to control garment stress and sliding, i.e., 4D garments.

These questions are tackled in this dissertation.

## 0.4. DISSERTATION ORGANIZATION

The following chapters of this dissertation are organized as below. Chapter 1 presents the background knowledge of knitting and the data representations for computational knitting design. Chapters 2 and 3 present the customized 3D knitwear design methods, which answer the first research (sub-)question (**RQ1**): how to design a knitwear and its machine knitting instructions for a given 3D shape. Chapter 4 presents the 4D knitwear customization method, which answers the second research question (**RQ2**): how to design a motion-aware 4D knitwear and generate its machine instructions. Chapter 5 uses body modeling methods to capture the body shapes of individuals and applies the developed knitwear design methods accordingly, which answers the last research question (**RQ3**): how to capture an individual's body models in different poses for 3D/4D knitwear customization. Chapters 2–5, as a whole, answer the main research question (**RQ**): *how to design customized 3D and 4D garments and generate the machine instructions?* At last, Chapter 6 summarizes the research findings of this dissertation. The relation between

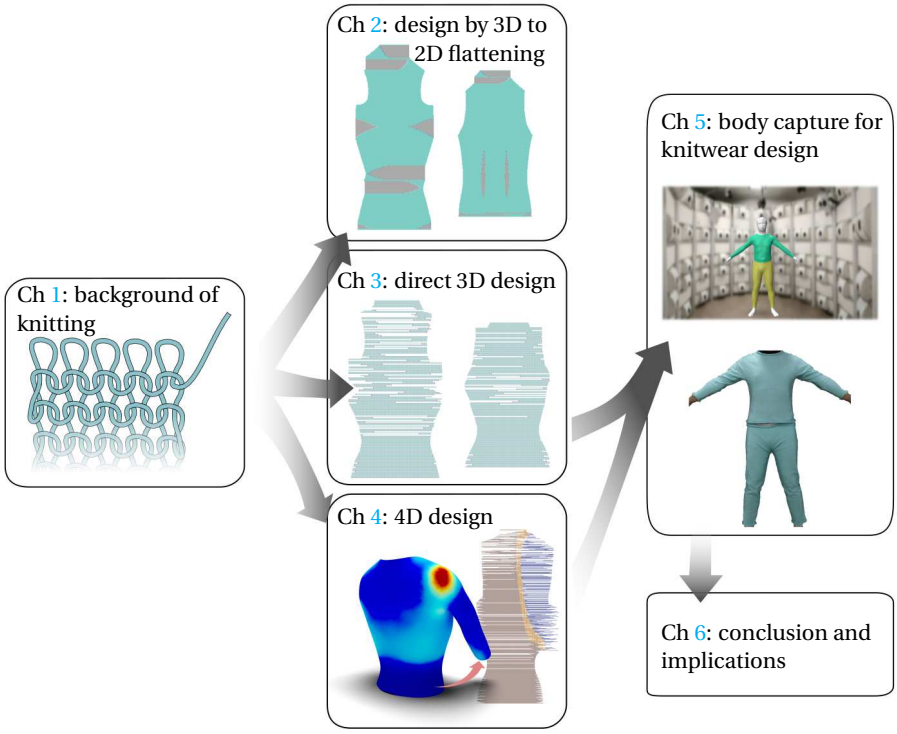



Figure 0.2: Overview of chapters 1–6. The background of knitting presented in Chapter 1 is fundamental to developing the computational design methods in the following chapters. Chapters 2–4 present different knitting design methods consecutively. Chapter 5 captures human body models for individuals and customizes knitwears using the design methods developed in previous chapters. Chapter 6 is the conclusion and implications.

these chapters is visualized in Fig. 0.2. Human body modeling is indispensable for accurate garment customization. However, its method is distant from the knitting design methods, thus it is presented after knitting design.

The structural engineer and architect Robert Le Ricolais said "The art of structure is how and where to put holes" [122]. The question "where to put holes" describes the structure design problem concisely, which is a good guiding question. The topics covered in Chapters 2, 3, & 4 are established as guiding questions in this style, supplementing the main research question (RQ) and sub-questions (RQ1–3).

- Chapter 2 presents a flattening-based 3D knitwear design method. Given the target 3D surface, we flatten it onto 2D to generate the knitting instruction. The key method to reduce the distortion of flattening is adding darts, like what garment designers do. Thus the guiding question of this chapter is *where to put darts*.
- Chapter 3 presents direct stitch mesh generation methods for given 3D surface. On stitch meshes, special stitches that provide intrinsic curvature to the mesh are named *shaping stitches*. The proposed methods place shaping stitches on the

mesh while considering the distortion, knittability, and reliability. Thus this chapter is about *where to put shaping stitches*.

- Chapter 4 presents the 4D knitwear design method that considers the garment's stress and sliding during body motion. We knit jacquard patterns with different elasticity levels on the garment. The soft pattern reduces stress and the firm pattern reduces sliding. Thus this chapter is about *where to put elastic patterns*.

# 1

# BACKGROUND OF KNITTING
## STARTING FROM 1D YARNS

In this chapter, let us start with the most basic knitting concepts and then come to the 3D shaping of knitting. I present several representations of knitting structures and the conversion methods between them. The principles of manufacturing to implement the knitting structures are described afterward. Visualization is important for conveying knitting structures. I put it in the appendix section 1.A of this chapter.

This chapter provides the preliminary of computational knitting. Most parts are either existing knowledge or their direct applications. The main contribution of this chapter is presenting a practical way of representing 3D knitting structures. Related contents can be found in the theses of Wu [212], Narayanan [133], and Kaspar [81]. Here I mainly consider knitting a sheet with one yarn piece, not knitting tubular shapes.

## 1.1. FUNDAMENTALS OF KNITTING

Human has a long history of producing fabrics from yarns, using different manufacturing methods, such as weaving, warp knitting, and weft knitting. A typical weft knitting structure is shown in Fig. 1.1. A single yarn is warped to form *loops*. A row of loops is named a *course*. Courses of loops are interlocked together to form a fabric. Loops interlocked consecutively in the same column form a *wale*. Interlocked loops are named *stitches*. These are the basic concepts of knitting a 2D fabric surface with a 1D yarn.

The knitting process is performed bottom-up in a zigzag manner. Repeating this process will knit a rectangular fabric. We may change the width of each course to generate curved boundaries of the fabric. However, the fabric is still flat, i.e., geometrically developable.

Several paragraphs of this chapter have been published in [109]: **Z. Liu**, X. Han, Y. Zhang, X. Chen, Y.-K. Lai, E.L. Doubrovski, E. Whiting, and C.C.L. Wang. *Knitting 4D garments with elasticity controlled for body motion*, ACM Trans. Graph., 40(4), 2021. doi: 10.1145/3450626.3459868. They have been enriched compared to the original published text.

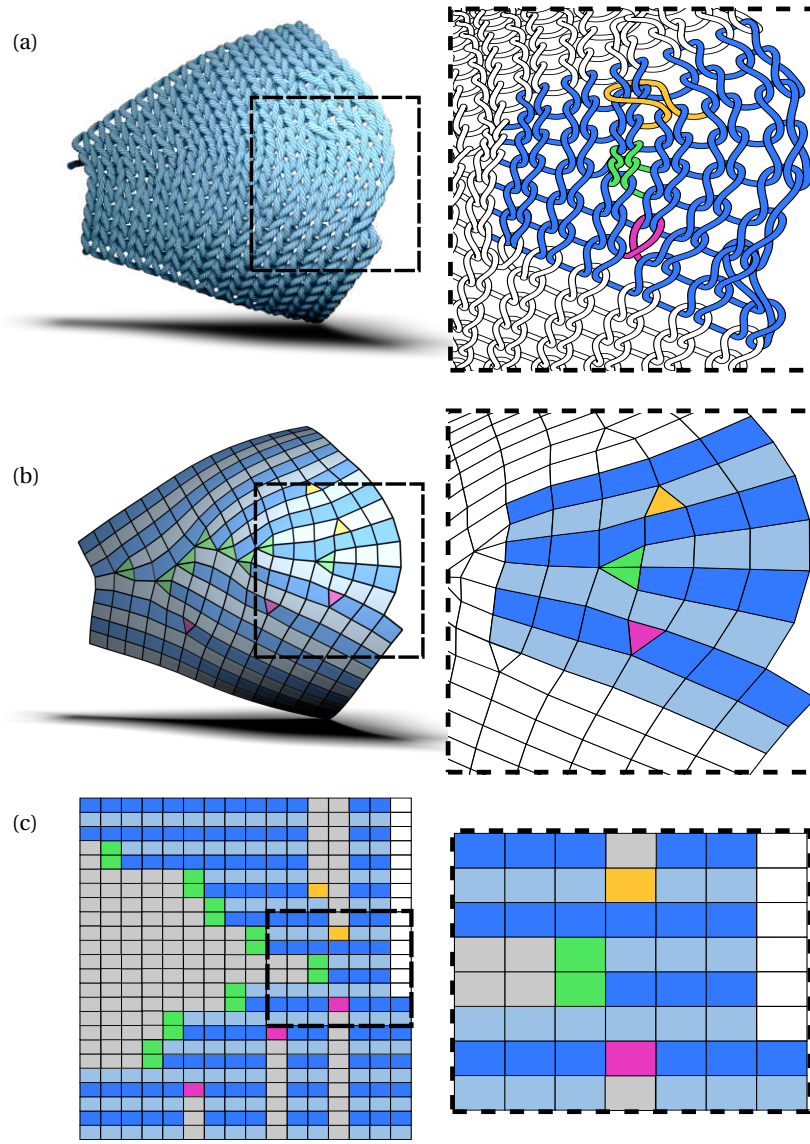



Figure 1.1: Three representations are used to illustrate the knitted stitches: (a) a yarn-level model which depicts the stitches faithfully, (b) a stitch mesh as an abstract representation at the fabric level where each triangle / quadrangle corresponds to a stitch in the yarn-level model, and (c) the knitting map as a 2D grid which actually provides the knitting instructions – rows with brighter blue cells are knitted from left to right and darker blue rows are the opposite. Gray cells correspond to collapsed regions in short-rows or short-columns.

3D shaping is crucial for designing knitwears that fit the target human bodies. There are two existing knitting techniques for 3D shaping. The first one is knitting short-rows. As shown in Fig. 1.1, some courses are knitted partially without reaching the left and/or right boundary of the fabric. These courses are named as *short-rows* [58]. The second one is increasing/decreasing by starting a new wale or ending a wale in the fabric interior. In this dissertation, these inserted or ended wales are named as *short-columns*. The same as short-rows, 'short' here means partial, not about the length. With these two techniques, 3D knitting is enabled.



## 1.2. REPRESENTATIONS

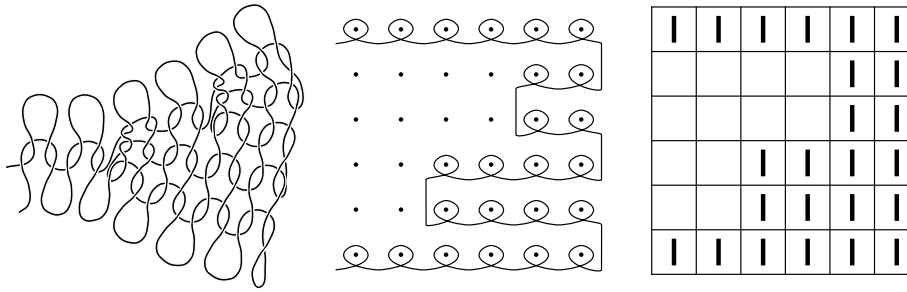
Three types of representations are used here to describe knitting structures. They are yarn models, stitch meshes, and knitting maps, as shown in Fig. 1.1. Each representation has its advantages and disadvantages.

- Yarn models depict the stitches by curves faithfully. They have realistic appearances but are difficult to design directly – users need to manipulate the 3D positions of vertices to realize desired interlocking relations of curves.
- A stitch mesh is an abstract representation on the fabric level where each triangle/quadrangle corresponds to a stitch. It is suitable for computational design but requires specific 3D modeling software interfaces to edit.
- A knitting map is a 2D grid that provides the knitting instructions (or knitting code, which can be used to instruct the knitting process). It is difficult to predict the knitted result for common users but easy to edit with widely used spreadsheet applications.

The concept of stitch mesh for knitting is proposed in [221]. It has been used in most research works on computational knitting [214, 145, 135, 215, 104, 216] and generalized to represent crochet [55], another fabrication technique that creates fabric surfaces from yarn by interlacing loops.

Besides stitch mesh, the other two representations, the yarn model and knitting map, have been widely used in the knitting industry. ISO 23606:2009 [71] sets two types of methods as the standards for representing weft-knitted fabrics. The first type is the *yarn path representation*. Line drawings of loop structures, like the yarn models here (Fig. 1.1(a) and Fig. 1.2(a)), belong to this type. This type also includes yarn paths drawn on rows of dots (Fig. 1.2(b)), in which the dots represent needles on needle beds. This method can represent both the structure of the knitted fabric and the working process of machine knitting, as demonstrated in Underwood, 2009 [192]. The second type is the *two-dimensional representation* (Fig. 1.2(c)), representing patterns on 2D grids, each grid cell for a stitch type. The knitting map used by us is similar to this type. This type is suitable for representing varied patterns, thus widely used in the knitter community [53].

Apart from these three types, there are some other representations. The *stitch map* proposed by Briar [21] is another representation for hand knitters. It gets rid of grids and aligns stitch symbols on the curved 2D domain, which is different from traditional knitting charts contained in grids. This method is similar to a flattened stitch mesh,



(a) (b) (c)

Figure 1.2: Other knitting structure representations: loop structure (a), yarn path (b), and chart (c).

illustrating the relaxed shape of a knitted flat fabric. TopoKnit [79] is a process-oriented representation for modeling the topology of yarns in weft-knitted textiles.

### 1.2.1. DATA STRUCTURES

Here are the data structures that describe the three knitting representations. This is preliminary for the upcoming rigorous discussion and computation. In this dissertation, the first knitting row goes from left to right.

**A yarn model** is a one-dimensional manifold curve segment. It is accompanied by a direction to indicate the knitting direction. A discrete model, polyline, is used to organize it digitally.

**A stitch mesh** is a manifold polygon mesh model. Each face belongs to one of the following face/stitch types:

- Quadrilateral (quad): for a regular stitch (or plain stitch).
- Left triangle: for the left end stitch of a short-row.
- Right triangle: for the right end stitch of a short-row.
- Bottom triangle: for the bottom stitch of a short-column, the position of an inside increasing stitch.
- Top triangle: for the top stitch of a short-column, the position of an inside decreasing stitch.

Each face $f$ is accompanied by a direction, indicating the face $f$ is knitted from left to right or the opposite. Then it is easy to find the next stitch of $f$ taking advantage of the half-edge data structure [19, 18]. We can also traverse all of $f$'s neighboring faces to determine if they belong to the same row as $f$ or the same column. Among all five stitch types, those four triangular ones are named *shaping stitches* collectively, without which the fabric can only be planar. They are corresponding to the turnings of short-rows (left and right triangles), splittings of short-columns (bottom triangles), and mergings of short-columns (top triangles)

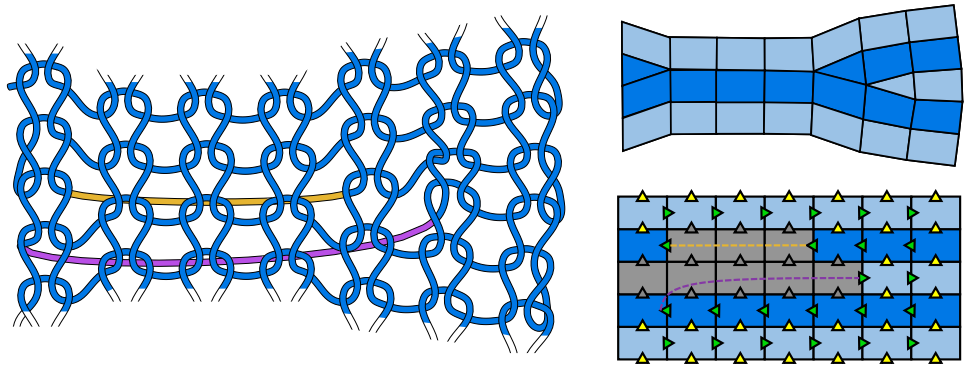


Figure 1.3: Long floats will be formed when discontinuity occurs in the same course (shown in yellow) and between courses (shown in purple). Both can lead to unstable knitting with incorrect stitches formed.

**A knitting map** is a rectangular grid, of which the row/column indices start from the bottom-left corner. Each of its cells belongs to one of the stitch types given above or one of the following two additional types:

- Collapsed cell: representing collapsed positions on the knitted fabric.
- Empty cell: representing cells outside of the knitted fabric.

Each non-collapsed/non-empty cell on a knitting map corresponds to an element on the stitch mesh, which means the knitting map is a flattened stitch mesh defined on a grid.

### 1.2.2. KNITTABILITY CONSTRAINTS

A knitting design represented by one of the data structures above may not be knittable, or can not be knitted robustly. A stitch mesh should satisfy the following constraints to represent a weft knitting fabric sheet knitted with a single yarn.

- Topological Constraint: none of the rows/columns can form cycles or helixes.
- Continuity Constraint: all stitches should be connected with 'one-stroke' following the knitting direction defined on each stitch. Specifically, we have
  - In-row continuity: all stitches in the same row should be neighboring connected.
  - Between-row continuity: the last stitch of the current row should be neighboring the starting stitch of the next row.

  Fig. 1.3 illustrates the situation if the stitch mesh is not 'continuous'.
- Shaping Constraint: the shaping stitches (triangles) should comply with the following additional constraints:
  - No boundary apexes. Each shaping stitch has a specific vertex, named apex, representing the very end of that short-row or short-column. The apexes

can not appear on the boundary. Otherwise, the shaping stitch should be replaced with a regular stitch since it reaches the boundary.

– Successive shaping stitches are usually not stable. If two bottom triangles are neighboring in the same row, it means there are two neighboring increasing stitches, which are difficult to be knitted due to yarn stretching (see Section 1.4.2). The case of two neighboring top triangles may cause three or more loops held by the same needle, which is not stable. Short-row ends (left/right triangles) always appear in pairs. However, four neighboring short-row end triangles in the same column will cause knitting difficulty when reducing the small holes at the short-row ends with tuck stitches (see Fig. 4.3 of [192]).

These constraints are defined on stitch meshes. To check if a knitting map is valid, we may check the knittability of the stitch mesh converted from it, using the conversion method that will be presented next. These constraints are concluded empirically. An axiomatic system will make the knittability easier to understand, but it is not developed in this dissertation.

## 1.3. CONVERSION BETWEEN REPRESENTATIONS

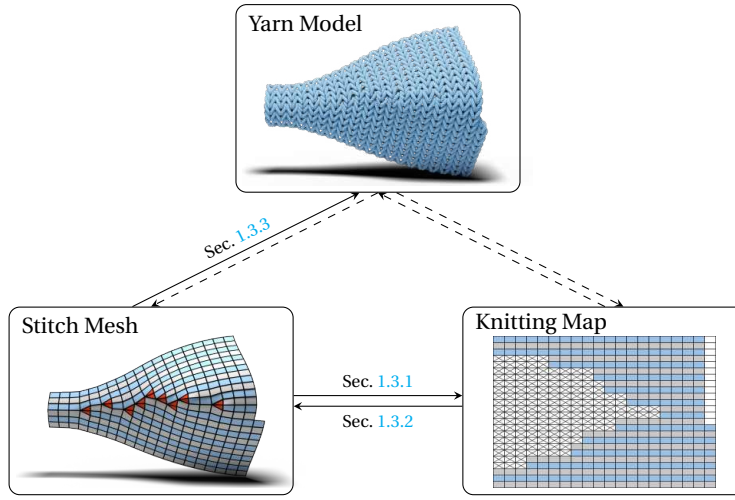



Figure 1.4: We are going to develop methods for the conversion between three types of knitting representations.

Now we have seen three types of knitting representations. Due to their advantages and disadvantages, we may use different representations in different scenarios. The conversion between them is necessary when switching from one scenario to another. For example, after drawing a 2D knitting map, the knitwear designer wants to see the 3D shape of the design. Then the conversion from knitting map to stitch mesh is required. The 3D model with yarn-level details will be desired if the designer wants to present the design to the customers. In another case, computational design methods generate stitch meshes directly (Chapter 3). We need to convert the stitch meshes as knitting maps to

execute them on knitting machines. The conversion methods are summarized in Fig. 1.4. Details will be presented in this section.

### 1.3.1. FROM STITCH MESH TO KNITTING MAP

Each element on the stitch mesh, either quadrangular or triangular, has at most four neighbors. It is straightforward to determine the relation of their row and column indices between neighboring face elements. After collecting the order relation between them, we use topological sorting to arrange the rows/columns. Then the knitting map is generated. The outline of this procedure is given in Algorithm 1.1.

**Algorithm 1.1** Conversion from stitch mesh to knitting map

```
1:  procedure mesh_to_map(S)
2:      G_r ← construct_in_row_graph()
3:      m ← find_connected_components(G_r)
4:      D_r ← construct_between_row_graph()
5:      topological_sorting(D_r)
6:      G_c ← construct_in_column_graph()
7:      n ← find_connected_components(G_c)
8:      D_c ← construct_between_column_graph()
9:      topological_sorting(D_c)
10:     M_{m×n} ← generate_knitting_map()
11:     fill_non_stitch_cells(M)
12:     return M
13: end procedure
```

Firstly, we construct an undirected graph $G_r$ to represent the in-row relation. Each face in the stitch mesh $\mathcal{S}$ is represented as a node in $G_r$. If two neighboring faces belong to the same row, these two nodes are connected by an edge. This step is finished after traversing all face-face connections of $\mathcal{S}$.

After that, we find the connected components of $G_r$. For each face $f_i$, the component containing it is determined, denoted as $R(f_i)$. The number of connected components $m$ is the number of rows in the target knitting map.

Then we find the order relation between rows, by constructing a directed graph $D_r$, taking $m$ components obtained in the last step as nodes. For each pair of neighboring faces $f_i, f_j$ belonging to the same column, if $f_i$ is knitted before $f_j$, then the row $R(f_i)$ is knitted before $R(f_j)$. In $D_r$, a directed edges is added from $R(f_i)$ to $R(f_j)$.

Now we conduct topological sorting on $D_r$. The resulting sorting $S(\cdot)$ maps a node in $D_r$ to an index, such that if there is an edge from $R(f_i)$ to $R(f_j)$, then $S(R(f_i)) < S(R(f_j))$. In this case, $S(R(f_i)) + 1 = S(R(f_j))$ is not always true, since short-rows may be inserted between $R(f_i))$ and $R(f_j)$.

Similar processes are applied to the in-column relation between faces until getting the topological order of $n$ columns (lines 6–9 in Algorithm 1.1). The only difference is that the topological order of rows is unique, but not for the columns. Since the fabric is knitted continuously by a yarn, the order between rows can be obtained by tracing the knitting sequence of the stitch mesh faces. The traced order is the unique solution

of the topological sort of $D_r$. But for the column, if two disconnected short-columns are inserted between two neighboring 'long'-columns, then the order of that two short-columns is ambiguous, resulting in different sorting solutions.

After getting the sorted row/column index of each face, we create a $m \times n$ grid and place the type of each face on the corresponding cell. Finally, we check all of the face neighboring relations on the mesh again. If two neighboring faces on $\mathcal{S}$ are not neighboring on the knitting map, then all cells between them are assigned as collapsing.



### 1.3.2. FROM KNITTING MAP TO STITCH MESH

Given a valid knitting map (Fig. 1.5(a)), the order of stitches is definite. We construct the stitch mesh topology by adding faces in the knitting sequence. The construction is a coarse simulation of the knitting process. Then a relaxed shape is computed to give a stitch mesh with plausible geometry.

When adding a new stitch face, we should maintain the manifoldness of the stitch mesh. For each vertex on the new face, we determine if it should be merged with an existing vertex on the stitch mesh.

After adding all faces, we get the 2D initial shape, lying in the $xy$-plane (Fig. 1.5(b)). The topology is correct, but the geometry (vertex locations) is not plausible. We deform the 2D initial shape to 3D with proper element shapes. We denote all vertices of the 3D stitch mesh as $\mathbf{V}$, and the vertices of the $i$-th face $f_i$ as $\{\mathbf{v}_j\}_{j\in f_i}$, $i = 1,\dots,m$. The face $\{\mathbf{v}_j\}_{j\in f_i}$ has an objective shape $\{\mathbf{v}'_j\}_{j\in f_i}$, which is determined by its stitch type. The transformation between $\{\mathbf{v}'_j\}_{j\in F_i}$ and $\{\mathbf{v}_j\}_{j\in f_i}$ should be as-rigid-as-possible (ARAP, [2, 64, 175]). Therefore the objective function is

$$\min_{\mathbf{V}} \sum_{i=1}^{m} w_i \sum_{j\in f_i} \left\| \mathbf{R}_i \mathbf{v}'_j + \mathbf{t}_i - \mathbf{v}_j \right\|_2^2, \tag{1.1}$$

in which $(\mathbf{R}_i, \mathbf{t}_i)$ is the best-fitting rigid transformation ($\mathbf{R}_i$ for rotation and $\mathbf{t}_i$ for translation) from $\{\mathbf{v}'_j\}_{j\in f_i}$ to $\{\mathbf{v}_j\}_{j\in f_i}$ that minimizes the inner sum, and $w_i$ is the area of $\{\mathbf{v}'_j\}_{j\in f_i}$ for weighting.

The solution is not unique because the fabric we are modeling is physically flexible. Eq. (1.1) is solved with Shape-Up [20, 40]. Shape-Up solves optimizations independently for each coordinate. If the $z$-coordinates of all vertices are zeros, then the result still lies in the $xy$-plane, as shown in Fig. 1.5(c). Therefore, the initial shape is warmed up with a $z$-direction displacement (Fig. 1.5(d)). The final result obtained with warming-up is shown in Fig. 1.5(e). The warming-up displacement is not unique.

### 1.3.3. FROM STITCH MESH TO YARN MODEL

There are five types of stitches. I designed the polyline yarn model of each basic stitch type manually, named canonical yarn models. A canonical yarn model of the corresponding type is deformed to fit a stitch mesh facet. All deformed yarn models are finally connected to form the final yarn model of the whole fabric.

In this dissertation, I generate the yarn model of a 3D stitch facet by deforming the canonical yarn models. The deformation problem can be formulated as an interpolation. Interpolation, which is ubiquitous in computational applications, can be achieved

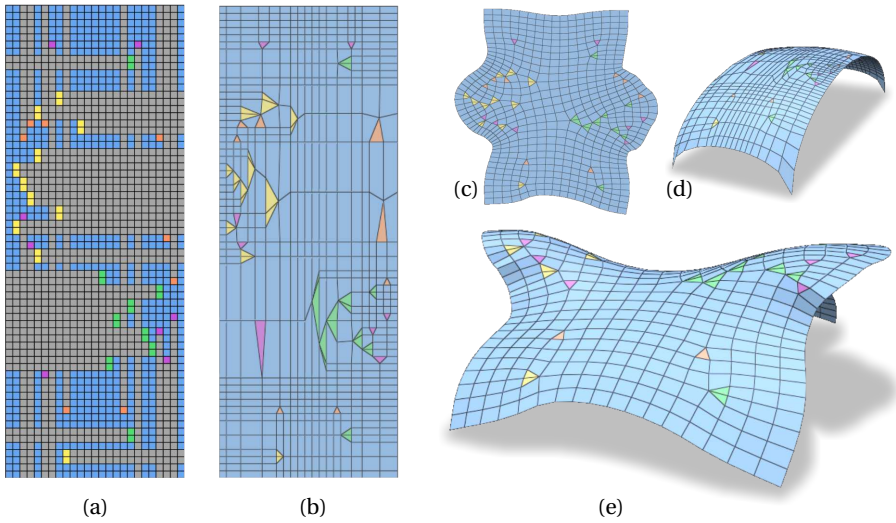



Figure 1.5: Convert a knitting map (a) to a stitch mesh (e). Firstly, convert the map (a) to a topologically correct mesh (b). Then, use the Shape-Up method [20] to predict the relaxed 3D shape. If Shape-Up starts from (b) directly, the 2D result (c) is produced, due to Shape-Up's axis-independent manner. Therefore, a $z$-direction displacement should be applied (d). At last, the 3D saddle surface (e) is obtained.

by (generalized) barycentric coordinates [225], or by solving differential equations [31]. Solving differential equations also produces a type of generalized barycentric coordinates, i.e., Harmonic coordinates. This reveals the connection between these two approaches of interpolation. The method in Yuksel et al., 2012 [221] uses mean value coordinates [49], which belong to generalized barycentric coordinates. The idea used here is also similar to deformation transfer [181].

Let us take the regular (quadrangular) stitch as an example (Fig. 1.6). The standard stitch quadrangle is defined as the unit square with vertices $\mathbf{s}_i = (0,0,0)$, $(0,1,0)$, $(1,1,0)$, $(1,0,0)$. Its canonical yarn model is a polyline composed of vertices $\mathbf{p}_j,\ j = 1,2,\ldots,k$. For a quadrangular facet $\{\mathbf{t}_i,\ i = 1,2,3,4\}$ on a stitch mesh, $\mathbf{p}_j$ is deformed to $T(\mathbf{p}_j)$ to fit $\{\mathbf{t}_i\}$. For $\mathbf{p}_j = (x_j, y_j, z_j)$, it is deformed to

$$T(\mathbf{p}_j) = \sum_{i=1}^{4} w_i \mathbf{t}_i + z_j \mathbf{n}, \tag{1.2}$$

in which $w_i$ is the bilinear interpolation weight related to $x_j, y_j$; $\mathbf{n}$ is the facet normal of $\{\mathbf{t}_i\}$. This transformation has the interpolation property that $T(\mathbf{s}_i) = \mathbf{t}_i$. From the viewpoint of deformation transfer, the deformation between $\mathbf{s}_i$ and $\mathbf{t}_i$ is transferred to $\mathbf{p}_j$. For other triangular stitch types, I use triangle barycentric coordinates for the interpolation and keep the other parts the same.

Besides the five basic stitch types, there are some special boundary stitches to prevent unraveling. They are casting on, binding off, outside increasing, and outside decreasing. The above method can also deform them to appropriate positions.

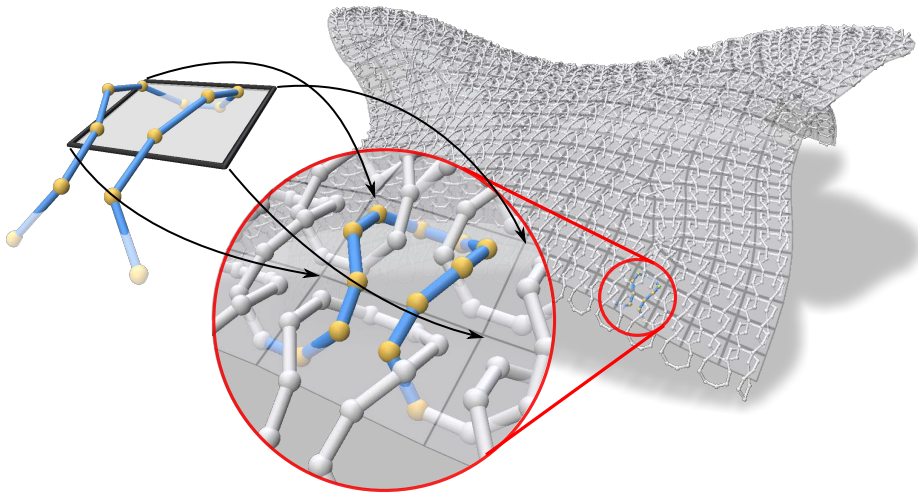


Figure 1.6: Converting stitch mesh to yarn model. A canonical (left) is deformed to fit a facet on the stitch mesh (right). The knitting map of this fabric is shown in Fig. 1.5.

### 1.3.4. FINAL REMARKS

Now we finished the conversion indicated by solid arrows in Fig. 1.4. In the applications, we can generate either a knitting map or a stitch mesh. Then the other representations are available.

In this dissertation, yarn models are not converted to other representations. The work presented by Kaspar et al., 2019 [83] is related, which synthesizes machine instructions from images of knitted fabrics with neural networks. This method can be adapted when it is necessary to convert yarn models to knitting maps by rendering yarn models into images and treating knitting maps as substitutes for machine instructions. We only have a limited number of stitch types, which are distinguishable when rendered from proper views. Then the conversion from yarn models to stitch meshes is implied. Thereafter, the directed graph of three nodes in Fig. 1.4 is *strongly connected*.

## 1.4. KNITTING 3D FABRICS

A central factor of knitwear customization is to knit the intended 3D shape. Short-row and short-column knitting are used in this dissertation for 3D shaping. Here are the advantages and disadvantages of them, from the perspective of manufacturing.

3D Shaping methods for knitting can be classified into three categories [177]:

1. varying the number of stitches in each row,
2. changing the structure of stitches,
3. altering the size of stitches.

Techniques in the second and the third categories are in general hard to provide precise shape control, thus not studied in this dissertation. There are two different ways to

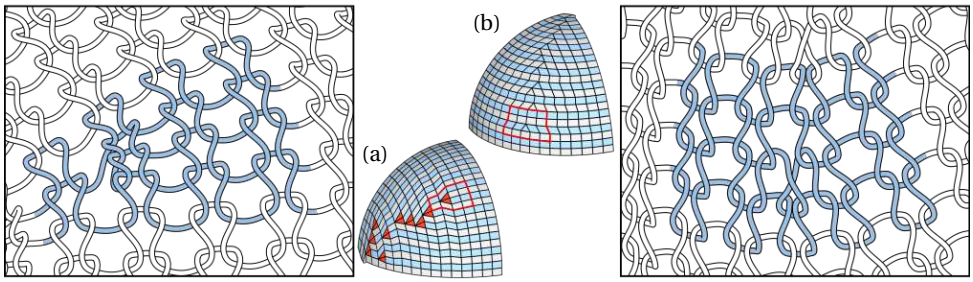



Figure 1.7: Two different strategies to knit the same 3D shape: (a) short-row shaping and (b) short-column shaping, where increasing / decreasing stitches between courses takes more operations on a machine with two needle-beds. For this example, knitting by only using short-row shaping can improve the knitting speed by 39.5% on the same machine.

change the number of stitches in different rows: short-row and short-column knitting, which will be discussed in this section.

### 1.4.1. SHORT-ROW KNITTING

Short-row shaping is also known as partial knitting, suspended stitches, held stitches, and fléchage [192]. The short-row shaping technique is as simple as suspending knitting the current row in the fabric interior and turning the carriage direction to knit the next row. The process is illustrated in Fig. 1.2(b), from which one can see why it is partial, suspended, and held. It can be realized on a single needle-bed of a knitting machine. Due to its simplicity, it is faster than short-column shaping (Fig. 1.7).

### 1.4.2. SHORT-COLUMN KNITTING AND THE STABILITY

In this dissertation, the term short-column knitting refers to knitting with stitch transfer. Stitch transfers can be classified into boundary transfer and interior transfer, according to the position of the transferred stitch. Boundary stitch transfer is often used to widen (or narrow down) the knitted fabric by introducing / terminating columns on the left / right boundary of the fabric. In contrast, interior stitch transfer contributes to the fabric curvature. In [70], the effect of boundary transfer is described as in-plane shaping and interior transfer as out-of-plane shaping.

To realize interior stitch transfer automatically on a knitting machine, the machine needs to have at least two needle-beds (named front- and back-beds as illustrated in Fig. 1.8). To increase one stitch in the interior of a row, all stitches at the left (or right) of the location to insert a new stitch are transferred to the back-bed. Then, the back-bed racks one pitch (i.e., the distance between neighboring needles on a needle-bed). After racking, the stitches with positions shifted by one pitch are transferred back to the front-bed. At the end of these operations, an empty needle is left in the current row, which is allowed to insert a new stitch in the next row – the starting stitch of a new short column. This process is illustrated in Fig. 1.8 by using the knitting notations as that of Fig. 1.2(b) and [192].

As presented in the process of stitch transfer (see Fig. 1.8(c)), steps 1-5 are for gener-

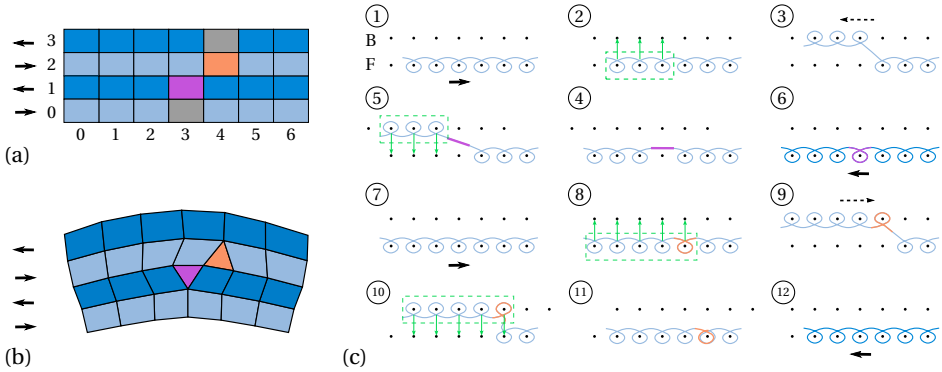



Figure 1.8: The illustration of interior stitch transfer for increasing / decreasing shaping process on a knitting machine with two needle-beds (B & F), where (a) the knitting map and (b) the corresponding stitch mesh are present to show locations of the starting stitch (purple) and the ending stitch (orange). Solid black arrows indicate the knitting directions of different rows (i.e., movements of the carriage), and the dashed black arrows are for the racking directions (i.e., movements of the back-bed). The processes of stitch transfer for increase / decrease are respectively shown in Steps 1-6 and Steps 7-12 of (c).

ating an empty needle on row 0 so that a new stitch (the purple one) can be inserted in row 1 as shown in Step 6. Steps 7-11 are employed to explain the stitch transfer between front- and back-beds on row 2 to merge the loop of orange stitch to its right neighboring stitch – i.e., two stitches are looped on the same needle after stitch transfer. As a result, the number of stitches can be reduced by one on row 3 (see also Step 12 of Fig. 1.8(c)).

Note that although the racking direction illustrated is toward the left for the starting stitch and toward the right for the ending stitch, it can also be racking toward an inverse direction. This is why there are two increasing operations (increase left / increase right) and two decreasing operations (decrease left / decrease right) described in [135]. A knitting planning algorithm can choose one according to different scenarios. A simple heuristic is to choose according to the location of stitches – i.e., the left or right portion of the needle-bed.

According to the process of stitch transfer conducted on a knitting machine as discussed above, we can find that there are some limitations of stitch transfer as summarized below:

- Stretching: When stitch transfer is applied to generate space for inserting a new stitch, both the yarn and the needles are stretched. When many new stitches are inserted at the same location, large stretching forces are generated and the needles have the risk to be broken. Therefore, it is common to only allow a small number of new stitches to be inserted at one location.

- Miss-transfer: When large stretching forces are presented during the stitch transfer, the loop of a stitch winded on the hook of a needle may be stretched away. As a result, an existing stitch can disappear during stitch transfer – i.e., miss-transfer occurs.

- Abrasion: When loops of stitches are repeatedly transferred between needles on

front- and back-beds, abrasion on yarns may make them broken. This problem can become very serious when multiple new stitches are inserted or merged on the same row.

- Speed: In general, the short-column shaping involves much more operations than the short-row shaping technique. It is much more time-consuming as shown in Fig. 1.7.



In the literature, [84] shows the problem of miss-transfer; [102] studies the efficiency of stitch transfer.

### 1.4.3. FABRICATION

In this dissertation, we mainly use a Shima Seiki knitting machine [164] for fabrication. Our design can also be performed on other machines or by hand.

Shima Seiki provides KnitPaint as its major design interface, as a part of its design system SDS-ONE APEX [165]. We use KnitPaint to convert our knitting instruction to Shima Seiki's machine code, then transfer the machine code to the knitting machine for fabrication. KnitPaint is a bitmap-based programming environment. Its knitting instruction representation is very close to the knitting map. Our knitting map can be converted with an element-wise substitution, with the help of *packages*. However, KnitPaint is not publicly available. Some information can be in academic publications like [192, 83].

As we have seen, multiple machine operations should be performed to knit a course with increasing or decreasing. Shima Seiki developed the knitting instruction representation method with packages. Using such a method, each row in the compressed map represents a course on the fabric, but its exact operations are encoded in another bitmap with multiple rows, named a package. The packages are similar to function-like macros in the C programming language, which will be expanded before executing. They simplified the knitting instruction generation process. Packages used in 3D sheet knitting are collected in Fig. 1.9.

Besides Shima Seiki, Stoll is another leading manufacturer of knitting machines. Stoll uses its own design software M1plus. Some of its information can be found in [194] (including its appendix). There are also some inexpensive automatic knitting machines, like [153, 86]. In the academic community, some machine-independent knitting machine programming methods are proposed [121, 26]. The easy-to-acquire hardware and software help the community with the research of knitting.

Besides the computerized machines mentioned above, we can also use hand knitting or manual machines. Hand knitting (Fig. 1.10(a)) has the best flexibility in knitting fancy structures [58], but it is slow, inaccurate, and has low scalability.

We can also execute the machine knitting instructions by hand on a manual machine [23, 168]. A Brother KH868, which has a single needle bed, requires extra hand tools to perform stitch transfers. The manual machine is more efficient than hand knitting.

## 1.A. VISUALIZATION OF KNITTING STRUCTURES

Visualization helps to understand the result of computational design. The rendering of knitwear and other types of textiles are surveyed in [27]. Different methods of knitwear

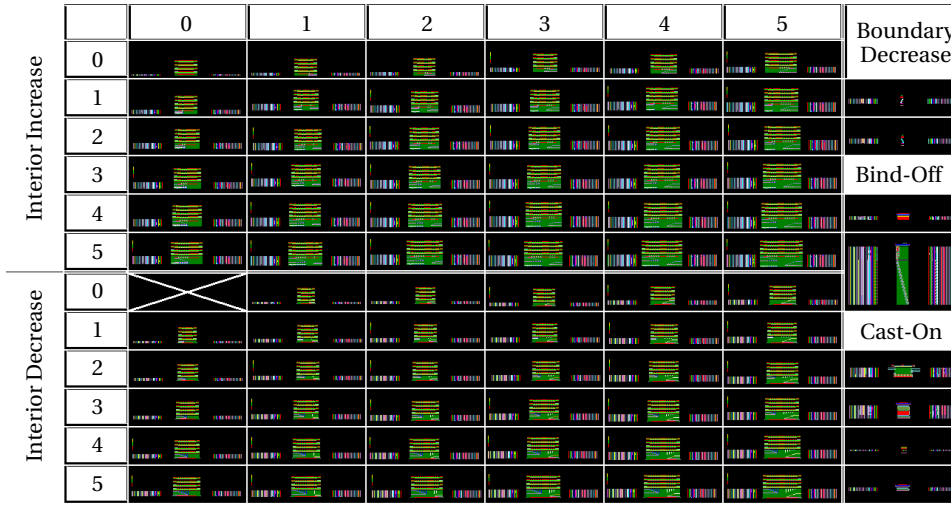
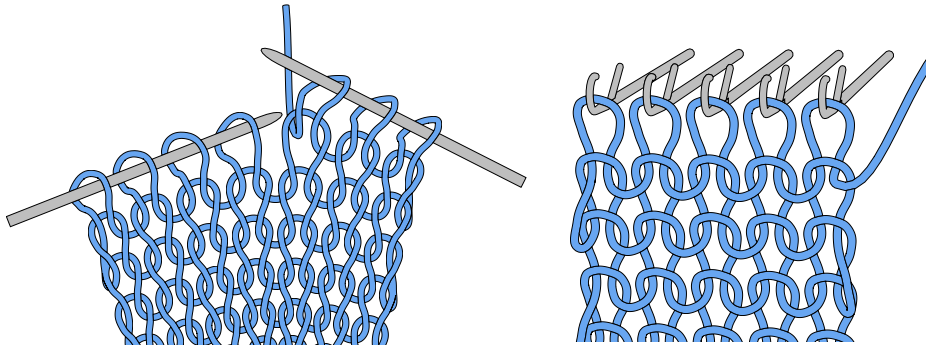



Figure 1.9: A collection of Shima Seiki packages used for 3D sheet knitting. The packages for interior increasing are collected in a $6 \times 6$ matrix, in which the $(i, j)$ cell contains the package for increasing $i$ stitches on the left and increasing $j$ stitches on the right. The interior decreasing matrix is the same. Packages for boundary decreasing, binding-off, and casting-on are presented on the right.



(a) (b)

Figure 1.10: Illustrations of knitting by hand (a) and manual machine (b). Please note that the manual machine exposes the back side of the fabric outwards.

rendering have been proposed in the last two decades [218, 30, 228, 229]. Wu and Yuksel [213] proposed a real-time rendering method with details of fiber, ply, and yarn.

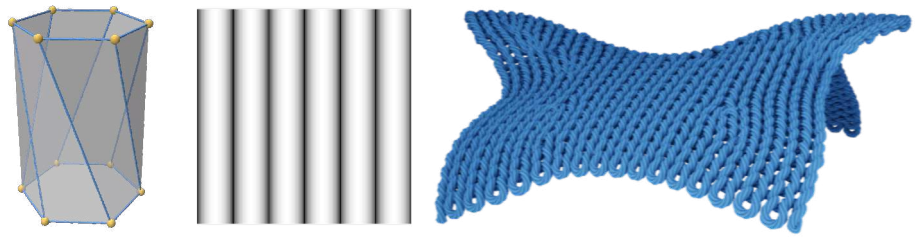
The yarn and stitch mesh rendering methods presented here mainly depend on texture mappings. Compared to other advanced techniques, these methods can be easily deployed on a wide range of platforms, including WebGL™ for web browsers and Unity® for virtual reality devices (e.g., HTC Vive). A vector graphics converting method is also presented here to illustrate yarn models in the style of technical drawings.

### 1.A.1. YARN MODEL RENDERING

This method converts the polyline yarn model to a smooth tube mesh, then uses a displacement map to realize ply-level details.

Firstly, a twisted $n$-gonal prism (Fig. 1.11(a)) and its texture image with $n$ repeated stripes (Fig. 1.11(b)) are constructed. The texture image is designed to be the displacement map. Thus each stripe is a side view height map of a cylinder. We use $n = 6$ to model a yarn with six plies.

Then the polyline model is converted to a Bézier curve and the textured prism is placed repeatedly along the curve. The repeated prism forms a tube mesh. Its displacement map provides the ply-level details (Fig. 1.11(c)).



(a) (b) (c)

Figure 1.11: Render yarn model with a displacement map. The twisted hexagonal prism (a) with displacement map (b) is placed repeatedly along the yarn model of the knitting design. The rendering result (c) shows ply-level details enabled by the displacement map. The knitting map and stitch mesh of this fabric is shown in Fig. 1.5.

The final result is rendered by Blender [14]. It is possible to further improve the realism by neural style transfer [113].

### 1.A.2. CONVERTING YARN MODELS TO VECTOR GRAPHICS

Vector graphics (Figs. 1.1(b–c)) has multiple advantages over raster images. They appear clean and exact at any size, thus suitable for scientific visualization. There are different methods of converting 3D meshes to 2D vector graphics. One type of method is tracing image feature curves [67] from rendered raster images (including shaded/shadeless images and depth/normal maps). Such methods suffer from ambiguity and inaccuracy. Another type of method is to extract geometric feature curves from the mesh directly [10, 45, 46]. The method adopted here belongs to this type.

Given a camera and a collection of 3D meshes (e.g., the rod meshes generated from yarn models) that are oriented, triangulated, manifold, intersection-free, and colored by connected components, the objective vector graphics should include boundary curves, occluding contours [10] observed from the given camera, and polygonal regions filled with correct colors.

Boundary curves of a manifold mesh are well-defined and easy to be extracted. An occluding contour is the projection of a curve on the mesh surface that separates locally visible and invisible regions, i.e., front- and back-facing regions. Thus they can be extracted by checking the normal of each face on the mesh. Part of a contour curve, either a boundary curve or an occluding contour, might be occluded by the surface. Because

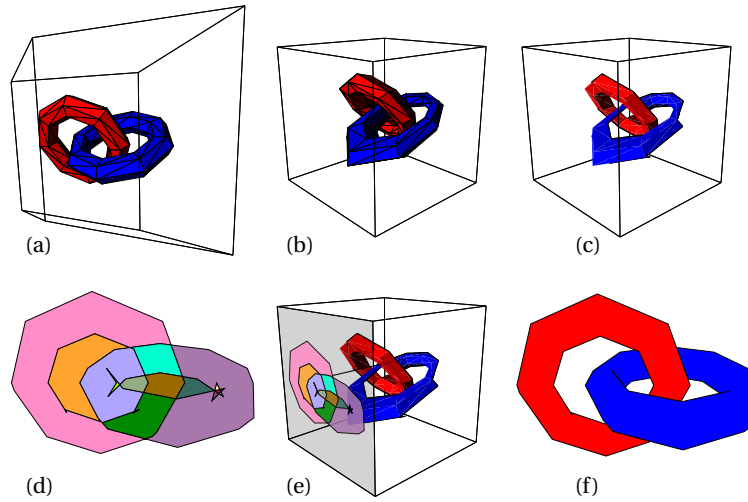



Figure 1.12: Converting a 3D scene to vector graphics. Given the scene in a view frustum (a), it is converted to the NDC space (b) and then the contour curves are detected (c). The projection of all contour curves forms a 2D arrangement (d). The invisible edges are removed by ray tests (e) and the final vector graphics is obtained (f).

both ends of an occluded curve segment can only be intersecting points between contour curves, we can get the curve segments from the *arrangement* [211] of all curves. Then the visibility of each curve segment is checked by ray tests. A region to be colored must be enclosed by visible contour curves. Hence the final color is determined by evaluating the *arrangement* faces.

**Algorithm 1.2** Converting 3D mesh to 2D vector graphics

1: **procedure** `mesh_to_vector_graphics`($\mathcal{M}, \mathcal{C}$)
2: $\mathcal{M}' \leftarrow$ `NDC_transformation`($\mathcal{M}, \mathcal{C}$)
3: $\mathcal{M}_f \leftarrow$ `back_face_removal`($\mathcal{M}'$)
4: $\mathcal{B} \leftarrow$ `trace_boundary`($\mathcal{M}_f$)
5: $\mathcal{A} \leftarrow$ `compute_2D_arrangement`($\mathcal{B}$)
6: $\mathcal{A}' \leftarrow$ `check_visibility`($\mathcal{A}, \mathcal{M}_f$)
7: `coloring`($\mathcal{A}', \mathcal{M}_f$)
8: **end procedure**

Steps of this procedure are given in Algorithm 1.2 and demonstrated in Fig. 1.12. First, the whole scene $\mathcal{M}$ is transformed into the camera $\mathcal{C}$'s normalized device coordinates (NDC) space (Fig. 1.12(a–b)), which is also known as the canonical view volume [119]. Then all projections are orthographic, thus easy to compute. Next, all back-facing faces are removed from the scene by checking the $z$-coordinate of face normal vectors in the NDC space (Fig. 1.12(c)).

Now we have a collection of front-facing mesh pieces $\mathcal{M}_f$. Their boundary curves

are exactly the contour curves of the original scene, including both original boundaries and occluding contours. The boundaries are traced and projected to the $xy$-plane. After computing the 2D arrangement [211] of the projection (Fig. 1.12(d)), the image space is segmented into non-overlapping regions (arrangement faces). The region boundaries (arrangement edges) are composed of two types of points, i.e., projections of contour curve points and intersections of contour curve projections.

The visibility of each arrangement edge is determined by ray tests (Fig. 1.12(e)). A 2D edge should be included in the final vector graphics if it is projected from a visible 3D curve segment on $\mathcal{M}_f$. I used the ray test implementation in trimesh [37]. If an invisible edge is removed, arrangement faces on its two sides are merged, forming a new polygonal region. Finally, all polygonal regions are colored by ray tests again (Fig. 1.12(f)). A triangle is retrieved using ray tests to get the color. The color of a region is unique since it is projected from a single connected component of $\mathcal{M}$.

The input mesh $\mathcal{M}$ is required to be intersection-free. Actually, it is already sufficient if mesh $\mathcal{M}_f$, the mesh after back face removal, is intersection-free. Intersections in the canonical yarn models are avoided in the design stage. Generally, intersections can be resolved using methods like [32].

### 1.A.3. STITCH MESH RENDERING

The stitch mesh can be rendered with yarn-level details, even if the yarn model is not constructed. Firstly, the depth maps of the canonical yarn models are rendered from an orthographic camera. Then, each facet of the stitch mesh is textured with the corresponding yarn model depth map (e.g., Fig. 1.13) for displacement mapping. In my experiments, the polygon number of a yarn model is 500 times as large as the stitch mesh. Rendering the stitch mesh saves time and computer memory.

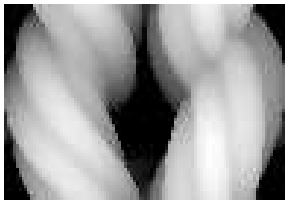


Figure 1.13: Displacement map of a regular facet on the stitch mesh.

# 2

# 3D TO 2D KNITTING DESIGN BY FLATTENING
## WHERE TO PUT DARTS?

## 2.1. INTRODUCTION

A knitting map is a flattened form of the stitch mesh, which inspires us to design knitting maps by flattening the 3D target shape to 2D. Since the 3D surfaces are usually non-developable, they are cut to reduce the distortion of flattening. The cuttings are referred to as *darts* in sewing garment design, which also applies to knitting. To design body-fitting garments using the flattening-based method, one of the most important problems is *where to put darts* since darts are crucial for 3D shaping.

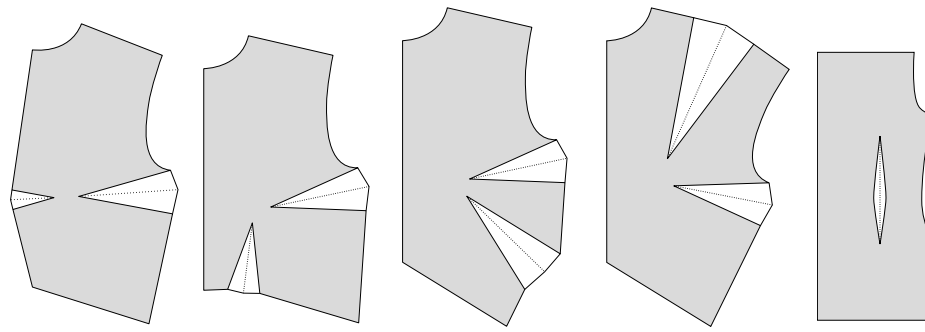



Figure 2.1: Sewing patterns with darts: front bodice (a–d) and back bodice (e). All patterns are adapted from https://freesewing.org, originally licensed under CC BY 4.0 (https://creativecommons.org/licenses/by/4.0/).

Darts are commonly used in garment design. Some example sewing patterns with darts are presented in Fig. 2.1. This dissertation starts by generating knitting maps from the darts placing approach, which is directly related to what designers and tailors use. The proposed computational design method analyzes the distortion of flattening and gives users hints to place darts. This chapter tackles the first research question (**RQ1**): how to design a knitwear and its machine knitting instructions for a given 3D shape. This method, which is preliminary to further research on more automatic methods, requires user interaction to place the darts.



The flattening is closely related to the parameterization of surface meshes. There have been numerous works on minimizing the parameterization distortion when the cutting seams are fixed. Some parameterization methods, e.g., [99], optimize both the flattening distortion and the seams. However, the flattened results, named 2D patterns, usually have irregular boundaries, making them difficult to be knitted. Some works, e.g., [227], consider manufacture constraints by controlling the seam lengths after flattening so that the seams can be sewn together without miss matching. Among the parameterization methods, cone flattenings [173] are automatically seamless. Some other approaches [125, 39, 161, 146, 68] approximate curved geometries with piece-wise developable surfaces, which can also be used for fabricating 3D shapes with planar materials. Our method will consider the knittability constraints when placing darts, which is not studied in existing works.

In this chapter, we first analyze the types of darts (Section 2.2). Then we flatten the 3D shapes to 2D (Section 2.3.1) with the knittability constraints considered. Next, we generate knitting maps from the 2D patterns (Section 2.3.2). Further discussions and conclusions are presented at last (Section 2.4).

## 2.2. TYPES OF DARTS

Darts can be categorized by shape and knitting method, respectively.

### 2.2.1. CLASSIFYING BY SHAPE

As we have seen in Fig. 2.1, the shapes of darts may be triangle-like (a–d) or diamond-like (e). They are named *one-end darts* and *two-end darts*, respectively. These two types classified by dart shape are related to the Gaussian curvature of the 3D target and the distortion of flattening.

On a dome-like shape, the Gaussian curvature is positive. When the shape is flattened, the center is compressed, and the boundary is stretched. The material will shrink to reduce the stretching if we cut it on the boundary. Therefore one-end darts are added to the boundary (the top row of Fig. 2.2). Similarly, the Gaussian curvature on a saddle shape is negative. When it is flattened, the boundary is compressed, and the center is stretched. We need cutting in the center to reduce stretching. Therefore two-end darts are added in the center (the bottom row of Fig. 2.2). On a developable surface with *zero* Gaussian curvature everywhere, darts are not needed. In Figs. 2.2(b–d), the distortion of each local flattening map is decomposed as scaling (compression or stretching) in a pair of orthogonal directions. The details are given in Section 2.A.1.

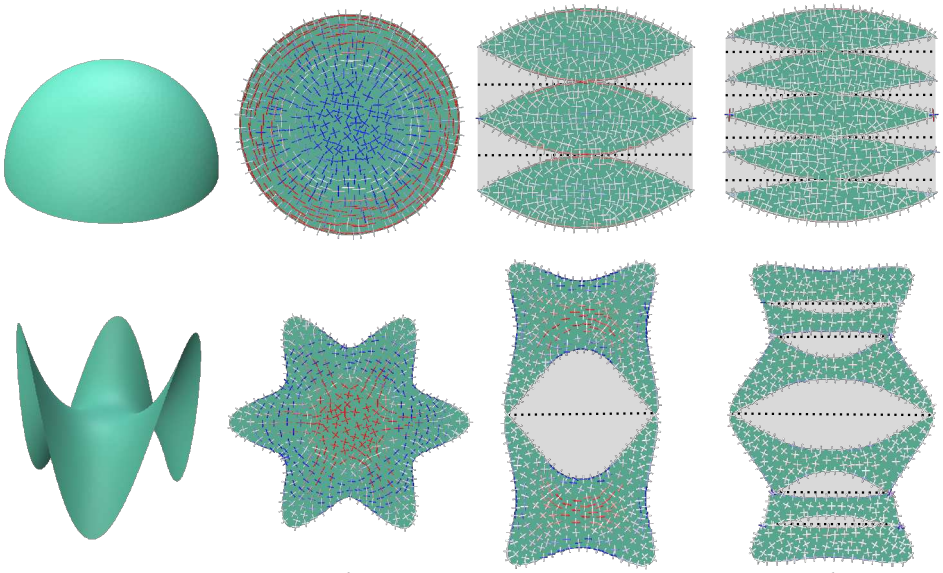



Figure 2.2: There are two types of darts according to their shape: one-end darts (the hemisphere on the top row) and two-end darts (the monkey saddle surface on the bottom row). Flattening the 3D shapes (a) to 2D (b), the distortion is visualized with color maps (blue for compression and red for stretching). We add one-end darts if the center is compressed (top) and two-end darts if the center is stretched (bottom). Adding more darts further reduces the distortion (c–d).



### 2.2.2. CLASSIFYING BY KNITTING METHOD

This dissertation considers two types of knitting strategies for 3D shaping: short-row and short-column. They correspond to two types of darts, *horizontal darts* and *vertical darts*, respectively.

We rasterize the 2D patterns as knitting maps on grids. The rasterized map should satisfy the knittability constraints (Section 1.2.2). We reformulate the constraints to assist the dart design. If a horizontal dart is knittable with short-rows, the constraints on it are roughly:

1. Alignment Constraint: two notches have the same $x$-coordinate. Then two dart legs will be knitted together without misalignment. The lengths of two dart legs may be different.
2. Angle Constraint: the angle between two dart legs should be less than a limit. This constraint avoids successive short-row ends in the same column (see Shaping Constraint in Section 1.2.2). However,

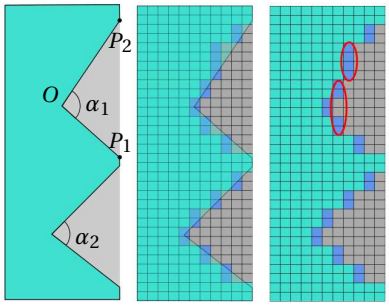



Figure 2.3: Constraints on horizontal darts. In the upper dart, two notches $P_1$ and $P_2$ have the same $x$-coordinate. The angle $\alpha_1$ between two dart legs $OP_1$, $OP_2$ is so large that there are neighboring short-row ends (marked red) in the knitting map. The lower dart is knittable without neighboring short-row ends.

the angle is difficult to evaluate since the dart legs are not straight in general. A large angle indicates adding more darts. The exact knittability still needs to be verified after generating the knitting map.

These constraints are illustrated in Fig. 2.3. If a vertical dart is knittable with short-columns, then:

1. Alignment Constraint: two notches have the same $y$-coordinate for boundary alignment.
2. Angle Constraint: the dart leg angle here is also constrained, to avoid neighboring increasing / decreasing in the same row.



Two-end darts should also comply with similar constraints. Again, it should be noted the exact knittability should be examined on the final knitting map.

## 2.3. KNITTING DESIGN BASED ON FLATTENING

The flattening-based knitting design method consists of two steps. The first step is to flatten a 3D mesh to 2D with some darts added. The next step is to generate a valid knitting map from the flattened 2D pattern.

### 2.3.1. FLATTENING 3D SURFACE WITH KNITTABLE DARTS

The flattening method should minimize the distortion of the flattened mesh, measured relative to the original 3D mesh. Thus we enforce the deformation of each triangular face to be as-rigid-as-possible (ARAP, [2, 64, 175]). Given a 3D mesh with $m$ faces, the optimization objective is

$$\min_{\mathbf{V}} \sum_{i=1}^{m} w_i \sum_{j \in f_i} \left\| \mathbf{R}_i \mathbf{v}'_j + \mathbf{t}_i - \mathbf{v}_j \right\|_2^2, \tag{2.1}$$

in which $\{\mathbf{v}'_j\}_{j \in f_i}$ is the collection of points on the $i$-th original 3D face $f_i$, $\{\mathbf{v}_j\}_{j \in f_i}$ is the flattened 2D face, $(\mathbf{R}_i, \mathbf{t}_i)$ is the best-fitting rigid transformation from $\{\mathbf{v}'_j\}_{j \in f_i}$ to $\{\mathbf{v}_j\}_{j \in f_i}$ (see Section 2.A.2), and $w_i$ is the face area for weighting.

We use Shape-Up [20, 40] to optimize it iteratively. In each iteration, we first estimate the best-fitting $(\mathbf{R}_i, \mathbf{t}_i)$ to project $\{\mathbf{v}'_j\}_{j \in f_i}$ to $\{\mathbf{v}_j\}_{j \in f_i}$ for each face independently; then we solve a linear system reconciling all projected vertex positions in a least-squares sense. If all vertices on the initial shape have zero $z$-coordinates, then the flattened shape is optimized within the $xy$-plane in the subsequent iterations, since Shape-Up solves optimizations independently for each coordinate (as discussed in Section 1.3.2). The Shape-Up iterations start from an initial flattening obtained by the *barycentric mapping* [19].

After flattening, we visualize the distortion as directions and scales of compression or stretching (see Section 2.A.1). Different types of darts are added to regions with large distortion. The shapes with darts will be flattened again. This design process is performed in iterations.

The dart notches should be aligned (see the alignment constraints in the last section) to make the dart knittable. In the flattening process, we set two notches of a dart as the

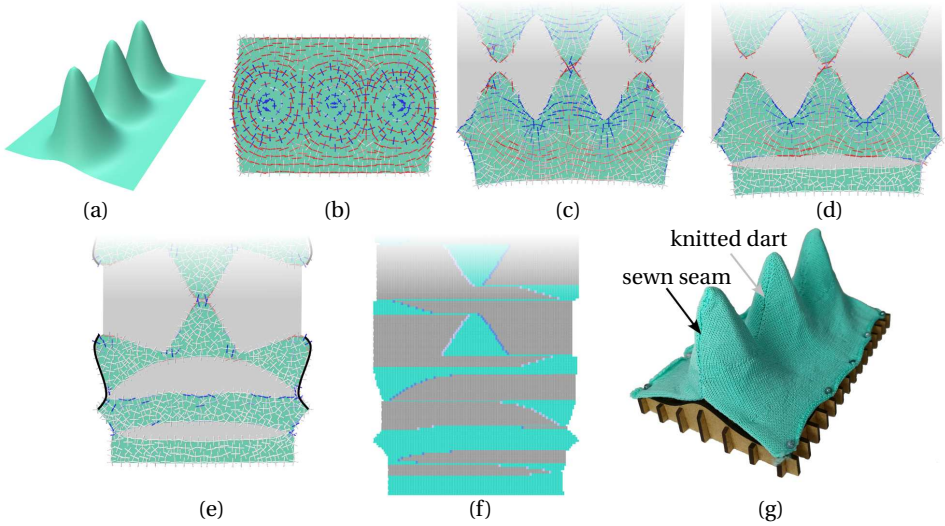



Figure 2.4: The triple-peak mesh (a) can be flattened as the 2D pattern (b). After adding one, three, and five darts, the flattening distortion is reduced in sequence, as shown in (c–e). The dart angles in (e), marked black, exceed the knitting constraint of horizontal darts, which are treated as boundaries during knitting and sewn afterward. The flattened pattern (e) is converted to the knitting map (f). The knitted result (g) is presented on a laser-cut support shape. The sewing seams are indicated. Only half of the patterns are shown in (c–f) for compactness.

same vertex. Then the coordinates of the two notches are the same after flattening. We translate the 2D patterns in $y$-direction to resolve the overlapping of one-end horizontal darts. The $x$-coordinates of notches will be kept the same. Vertical darts are flattened in the same way.

The flattening method is demonstrated with the triple-peak model shown in Fig. 2.4. If we flatten the original shape Fig. 2.4(a) without any darts, the peaks are extremely compressed (b). After cutting a horizontal dart across all peaks, the shape is flattened as (c). The upper part of the flattened 2D pattern is the same as the lower part, thus it is faded out for compact representation. More darts are added in highly distorted regions in (d) and (e). The knitting map and the knitted result are shown in (f) and (g) respectively.

### 2.3.2. KNITTING INSTRUCTIONS FROM 2D PATTERNS

The flattened 2D patterns are rasterized to knitting maps as the knitting instructions. The continuity constraints should be satisfied in the results. Inspired by [134, 104], we adjust the rasterized 2D patterns to realize the continuity of knitting.

Given the stitch size $s_w \times s_h$, we first rasterize the 2D pattern as a bitmap with the pixel size $s_w \times 2s_h$. Next, we split each row $i$ is split into two rows $2i$ and $2i+1$. Then the right-end of row $2i$ is aligned with the start of the next row, i.e., the right-end of row $2i+1$. We extend some rows to make sure continuity at the left ends. Around two-end darts (see the waist region on the top row of Fig. 2.5), each row contains two short-rows. We process all right-side short-rows first and then jump to the left-side short-rows.

To stabilize the casting-on and binding-off, we add long rows at the bottom and the


Figure 2.5: Flattening a mannequin surface to generate the knitting maps. The mannequin mesh is first split into the front bodice and back bodice (a). Each patch is flattened once (b) and again after adding darts in distorted regions (c). The 2D patterns (c) are rasterized as bitmaps in (d) and finally converted to knittable knitting maps (e) that comply with the continuity constraints. The flattened front bodice patterns are shown on the top row. The bottom row is the back.

top of the knitting maps when necessary. If the knitting map starts with a row shorter than the following ones (see bottom of the front bodice in Fig. 2.5(d)), there will be some outside widening when knitting. The fabrication might be unstable because of lacking pull-down force when knitting the widened columns. Therefore, we add a row at the bottom and produce two one-end darts (Fig. 2.5(e)). The knitting process is more stable after casting on such a long row. Knitting with branches will also be complex and unstable, like the top of both bodices in Fig. 2.5(d). We add long rows on top of them so that they can be knitted with a single yarn and the binding-off is more stable. The bottom of the back bodice in Fig. 2.5(d) is processed similarly.

Using the conversion methods presented in Chapter 1, we can convert a 2D knitting map to a 3D stitch mesh and align it with the target 3D shape. The distortion of stitch mesh can be evaluated on each face, as shown in Fig. 2.6(d), using the method presented in Section 2.A.2.

## 2.4. DISCUSSIONS AND CONCLUSIONS

This chapter presents a knitting map generation method based on flattening the 3D target shapes into 2D patterns. We add darts by analyzing the flattening distortion (compression or stretching) and considering the knittability constraints. The distortion is reduced after adding more darts. We convert the flattened shapes to valid knitting instructions. Both 3D knitting techniques, i.e., short-row and short-column knitting, are

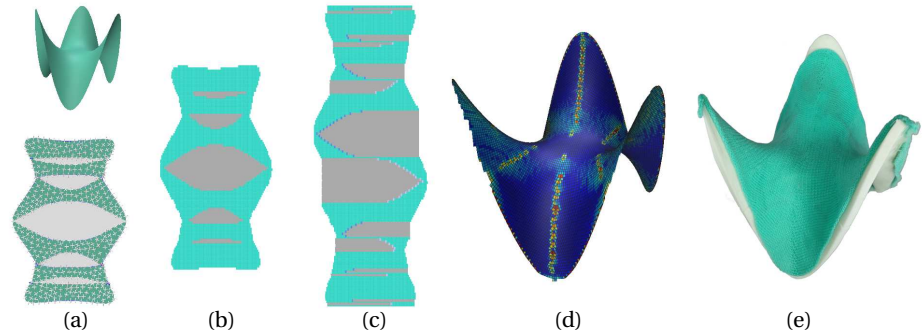



Figure 2.6: The saddle surface in Fig. 2.2 is flattened with darts (a) and converted to a bitmap (b). A knittable map (c) is obtained by adjusting the continuity in (b). After converting it to stitch mesh and aligning it with the target shape, the distortion can be evaluated (d). The knitted result is (e).

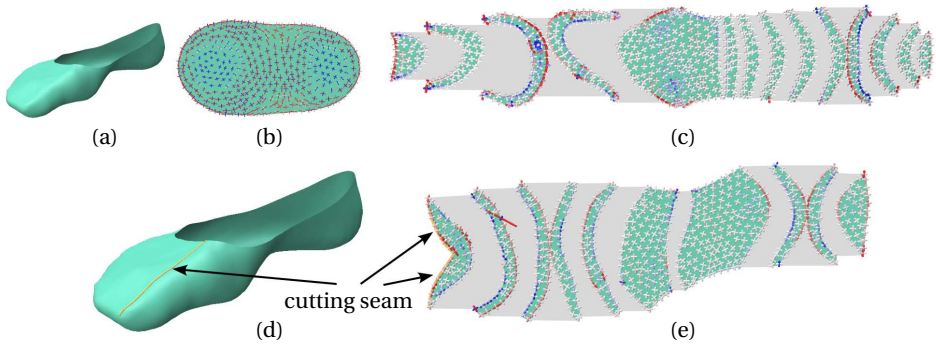



Figure 2.7: After flattening the sock mesh (a) to 2D, we find the toe and heel regions are compressed (b). The distortion is still large even if adding many darts (c). If the mesh is cut (d), the flattening distortion will be reduced (e).

employed to knit closed darts directly.

This approach has several failure cases. Using the flattening-based method, the knitting designs of triple-peak (Fig. 2.4) and sock (Fig. 2.7) need sewing after knitting. After flattening, the triple-peak has large darts that can not be knitted. The flattening distortion of the sock needs to be further reduced by cutting a seam, which will be sewn afterward. In a more challenging case, the skullcap (Fig. 2.8), there are some flipped triangles (fold-overs) caused by the alignment constraint for knittable darts. More complex darts should be added to overcome these limitations.

There are several directions to extend the current flattening-based method for knitting map generation. Now we can add multiple non-intersecting darts to the shape. It is interesting to design intersecting darts. We generate knitting maps by rasterizing 2D patterns, implying that the 2D patterns are traversed with grid-aligned paths. Flexible knitting directions, i.e., spatial-varying direction fields, should also be possible, like [84]. Then we may knit darts in general orientations other than horizontal or vertical. In ad-

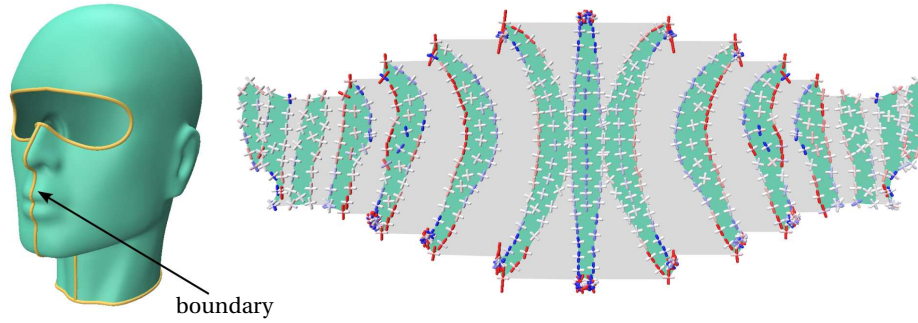



Figure 2.8: Given the skullcap model with disk-like topology, its flattening distortion is large even if adding many darts of the same orientation.

dition, the current design approach requires the manual placement of darts. A more automatic method is valuable. Anyway, we will see that all of these extensions are realized with the direct 3D design approach by stitch mesh generation, which will be presented in the next chapter.

## 2.A. DISTORTION ANALYSIS USING SVD

In this chapter, we analyzed the distortion of 2D flattening and the distortion of stitch meshes. Both used singular value decomposition (SVD) but in different ways.

### 2.A.1. DISTORTION OF FLATTENING

We add darts in regions with large flattening distortion. The distortion is obtained by SVD of the linear mapping between each pair of 3D-2D triangles, the same as [172].

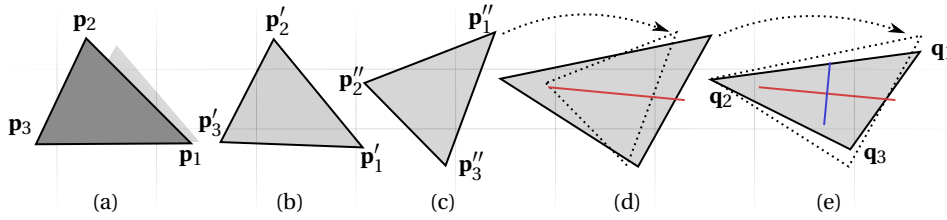



Figure 2.9: The SVD of a local mapping in mesh flattening. The 3D triangle $\{\mathbf{p}_i\}$ (a) is rotated to the $xy$-plane rigidly by $\mathbf{R}$ (b). Then we rotate it in the plane by $\mathbf{U}\mathbf{V}^\top$ (c). At last, it is scaled in a pair of orthogonal directions by $\sigma_1, \sigma_2$ (d–e). The scaling directions are two columns of $\mathbf{U}$, illustrated by line segments in (d–e), with the lengths representing scaling factors.

Given a 3D triangle mesh $\mathcal{M}$, we pick one of its faces and denote its vertices as $\mathbf{p}_i$, $i = 1,2,3$. After flattening $\mathcal{M}$ onto the $xy$-plane, $\mathbf{p}_i$ are mapped to $\mathbf{q}_i$ respectively. This local mapping can be described as a linear transformation, as illustrated in Fig. 2.9. Firstly, there is a rotation matrix $\mathbf{R}$ that maps the 3D triangle $\{\mathbf{p}_i\}$ to the $xy$-plane rigidly (Fig. 2.9 (a–b)). We have the 2D points $\mathbf{p}'_i = \mathbf{R}\mathbf{p}_i$, removing its last dimension. Then there is a $2 \times 2$

matrix $\mathbf{T}$ such that

$$\mathbf{T}\left(\mathbf{p}_2' - \mathbf{p}_1',\ \mathbf{p}_3' - \mathbf{p}_1'\right) = \left(\mathbf{q}_2 - \mathbf{q}_1,\ \mathbf{q}_3 - \mathbf{q}_1\right). \tag{2.2}$$

Taking the SVD of $\mathbf{T}$, $\mathbf{T} = \mathbf{U}\Sigma\mathbf{V}^\top$, we have two orthogonal matrices $\mathbf{U}, \mathbf{V}$ and a diagonal matrix $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2)$. Now we see the distortion comes from the scaling in a pair of orthogonal directions with the factors $\sigma_1, \sigma_2$ (Fig. 2.9 (d–e)). All the other transformations are rigid rotations without any distortion (Fig. 2.9 (a–c)).

Since we are analyzing the shape distortion, the translations are ignored in this process. We may choose different 3D-to-2D rotation matrices $\mathbf{R}$ and different linear combinations of edge vectors to compute the linear map $\mathbf{T}$. It does not change the results of $|\sigma_1|, |\sigma_2|$. The singular values $\sigma_1, \sigma_2$ are used to define various distortion metrics for mesh parameterization [172].



### 2.A.2. DISTORTION OF STITCH MESHES

Each stitch type has an ideal shape represented by a polygon $\mathbf{p}_i, i = 1, \ldots, n$, defined by the calibrated stitch size. After projecting the stitch mesh onto the target shape, an ideal stitch $\{\mathbf{p}_i\}$ is distorted to $\{\mathbf{q}_i\}$. Before evaluating the distortion, we need to align $\{\mathbf{p}_i\}$ to $\{\mathbf{q}_i\}$ rigidly.

According to [176], there is a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ such that

$$(\mathbf{R}, \mathbf{t}) = \underset{\mathbf{R}\in SO(3),\ \mathbf{t}\in\mathbb{R}^3}{\arg\min} \sum_{i=1}^{n} \left\| (\mathbf{R}\mathbf{p}_i + \mathbf{t}) - \mathbf{q}_i \right\|^2. \tag{2.3}$$

The centers of two triangles are $\bar{\mathbf{p}} = \sum_{i=1}^n \mathbf{p}_i / n$, $\bar{\mathbf{q}} = \sum_{i=1}^n \mathbf{q}_i / n$. Then we have the $3 \times 3$ matrix $\mathbf{S} = \mathbf{P}\mathbf{Q}^\top$, in which $\mathbf{P}_{3\times n}$ takes $(\mathbf{p}_i - \bar{\mathbf{p}})$ as columns and the similar is $\mathbf{Q}$. Using the SVD $\mathbf{S} = \mathbf{U}\Sigma\mathbf{V}^\top$, we have the solution of Eq. (2.3)

$$\mathbf{R} = \mathbf{V}\mathbf{U}^\top \quad \text{and} \quad \mathbf{t} = \bar{\mathbf{q}} - \mathbf{R}\bar{\mathbf{p}}. \tag{2.4}$$

The rigid transformation defined by $(\mathbf{R}, \mathbf{t})$ aligns $\{\mathbf{p}_i\}$ to $\{\mathbf{q}_i\}$ optimally in the least squares sense.

This method of computing best-fitting rigid transformation is widely used for mesh deformation (known as as-rigid-as-possible, ARAP [175, 20]) and point cloud registration [191]. In this dissertation, we have used it in the conversion from knitting map to stitch mesh (Section 1.3.2) and the 3D shape flattening (Section 2.3.1). Point cloud registration will be used for 3D scanning (Section 2.C.1).

In this section, both methods for distortion computing used SVD. It should be noted that their objectives are different. The estimated rotation matrices are also generally different.

## 2.B. ALGORITHMS OF MESH SPLITTING

In this chapter, input meshes are cut along curves to generate darts. Such a curve is a polyline path connecting two vertices on the mesh. Points on the polyline path are not necessarily vertices of the mesh but usually the points dividing some edges. We first split the mesh along the polyline, then cut the mesh along the polyline constructed by mesh

vertices. The splitting method is presented here and implemented in MeshUtility [106]. The cutting method is available in libigl [73], which is not discussed here.

In this scenario, we cut the meshes with geodesic paths. Each face is cut by a geodesic path at most once. Otherwise, the path segment on this face is not straight, not even the shortest. Consequently, the mesh can be split using Algorithm. 2.1.

**Algorithm 2.1** Simple mesh splitting

**Input** Mesh $\mathcal{M}$ and the polyline path. Vertex $i$ on the path divides edge $(e_{i0}, e_{i1})$ in ratio $r_i$.

1: **procedure** `mesh_splitting_simple`($\mathcal{M}$, $(e_{i0}, e_{i1}, r_i)$ for $i = 0, 1, \ldots, n-1$)
2: $L \leftarrow$ empty list ▷ Indices of the path after splitting.
3: **for** $i = 0, 1, \ldots, n-1$ **do**
4: **if** $e_{i0} = e_{i1}$ **then**
5: Append $e_{i0}$ into $L$. ▷ The path passes a mesh vertex.
6: **else**
7: Add vertex $q \leftarrow (1 - r_i)e_{i0} + r_i e_{i1}$.
8: Split edge $(e_{i0}, e_{i1})$ with $q$.
9: Append the index of $q$ into $L$.
10: **end if**
11: **end for**
12: **return** $\mathcal{M}$, $L$ ▷ Resultant mesh and path vertex index list.
13: **end procedure**

This simple method works for the at-most-once splitting thanks to the following reasons. Firstly, it keeps the mesh triangular, although it changes the mesh topology by edge splitting, i.e., removing two faces and adding four faces (line 8). Secondly, it adds new vertices but does not change the indices of existing vertices. Thus the vertex indices defined in the input $(e_{i0}, e_{i1})$ are always valid.

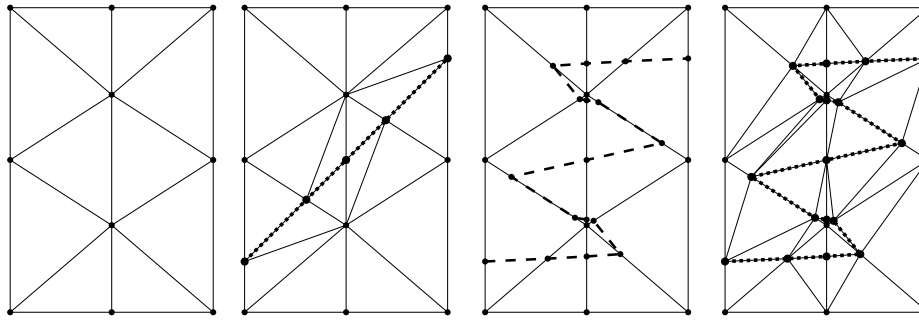



Figure 2.10: Split the mesh (a) with given paths. If each face is split at most once, like the dotted path in (b), we can split faces directly using Algorithm 2.1. If a face is split multiple times, like the dashed red path in (c), Algorithm 2.2 becomes necessary. We split edges first and generate a polygon mesh (c). Then we split polygon faces using the path segments and triangulate the remaining mesh at last (d).

In general, a face may be split by a path multiple times. We need the complete mesh splitting method, Algorithm 2.2. In this case, we add path segments explicitly (line 8) to make sure all path segments appear as mesh edges in the result. To do so, we keep polygons after edge splitting, without triangulating them immediately, as shown in Fig. 2.10. In the final triangulation step, degenerated triangles should be avoided.

**Algorithm 2.2** Complete mesh splitting

**Input** Mesh $\mathcal{M}$ and the polyline path. Vertex $i$ on the path divides edge $(e_{i0}, e_{i1})$ in ratio $r_i$.

1: **procedure** `mesh_splitting_complete`($\mathcal{M}$, $(e_{i0}, e_{i1}, r_i)$ for $i = 0, 1, \ldots, n-1$)
2: **for** each edge $e$ of $\mathcal{M}$ **do**
3: Collect all path vertices $q_i$ on $e$. ▷ For simplicity, sort $q_i$ before splitting.
4: Split $e$ with all $q_i$. ▷ But do not split any face.
5: **end for**
6: $L \leftarrow$ indices of path vertices on the mesh.
7: **for** each segment $(q_i, q_j)$ of the splitting path **do**
8: Add $(q_i, q_j)$ as an edge of $\mathcal{M}$ by splitting the face containing both vertices.
9: **end for**
10: Triangulate $\mathcal{M}$.
11: **return** $\mathcal{M}$, $L$ ▷ Resultant mesh and path vertex index list.
12: **end procedure**

## 2.C. NOTES ON 3D SHAPE ACQUISITION AND FABRICATION

The target shapes used in this chapter are obtained from various sources. Before designing the knitting plans, the target shapes are converted to triangular meshes as our algorithms require. After knitting, we use 3D shapes to support the flexible fabrics. In these steps, several geometry processing techniques and fabrication practices are applied, which are presented in this section.

### 2.C.1. RIGID 3D OBJECT SCANNING

The 3D model of the mannequin in Fig. 2.5 is scanned by a structured light system. This system provides frames of point clouds. We registered the frames together manually using the iterative closest point (ICP) method implemented in [34]. SVD is used in ICP to obtain rigid transformations between frames. Pairwise registration needs to be globally optimized (Fig. 2.11) to improve the alignment. We used [231]'s implementation of Choi et al., 2015 [33] for this. Finally, the complete oriented point cloud is converted to a mesh model with Poisson surface reconstruction [85].

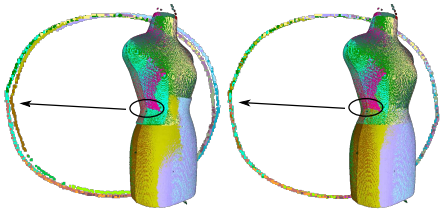


Figure 2.11: Pairwise registration and global alignment. Slices are shown in the zoom-in view.

### 2.C.2. SURFACE OF EXPLICIT FUNCTIONS

The triple-peak model (Fig. 2.4) and the monkey saddle (Fig. 2.2) are generated from explicit functions. Their processing procedures are similar.

**The triple-peak model** is the surface of $z = f(x, y),\ x \in [-1.5, 1.5],\ y \in [-1, 1]$, in which

$$f(\mathbf{x}) = w \sum_{i=0,1,2} \exp\left(-\frac{\|\mathbf{x} - \mathbf{c}_i\|^2}{\sigma^2}\right). \tag{2.5}$$

The parameter values are: the coordinates of $\mathbf{c}_i$s $(0,0)$, $(1,0)$, and $(-1,0)$; $w = 1.2$; and $\sigma^2 = 0.1$.

Given the explicit function of the surface, we generate the mesh model using the incremental isotropic remeshing framework (Algorithm 2.3). Details of this algorithm, including the boundary/feature preserving rules, are described in [19], and implemented in [106]. In the projection step (line 7), we keep the $x, y$-coordinates of each vertex and set the $z$-coordinate as the function value $f(x, y)$. For general shapes without explicit mathematical expression, the AABB (axis-aligned bounding box) tree in libigl [73] can be used for the projection.

**Algorithm 2.3** Incremental isotropic remeshing

```
1: procedure remesh(M, l, n)
2:     for i ← 0, 1, ..., n − 1 do
3:         split_long_edges(l)
4:         collapse_short_edges(l)
5:         equalize_valences()
6:         tangential_relaxation()
7:         project_to_surface()
8:     end for
9: end procedure
```

We scale the mesh uniformly so that the $x$-axis length is 15cm. Such a mesh is ready to be used for knitting design.

We fabricate a support shape, i.e., a solid object to support the knitted fabric. Then a solid mesh should be prepared. We add a rectangle as the base. The original shape and the base are connected with a graph-based tessellation method, a simplified version of Fuchs et al. [51].

The boundaries of the original shape and the base are two polyline loops. We denote them as $\Lambda_A = (a_i)_{i=1,\dots,m}$ and $\Lambda_B = (b_i)_{i=1,\dots,n}$ respectively. In the loop $\Lambda_A$, $a_i$ and $a_{i+m}$ are the same point; $a_i$ and $a_{i+1}$ are connected with a segment. Loop $\Lambda_B$ is the same. The directions of $\Lambda_A$ and $\Lambda_B$ should be the same. The objective is to connect $\Lambda_A$ and $\Lambda_B$, i.e., to fill the region between them. Firstly, we compute the pairwise distance matrix $D_{m\times n} = (\|a_i - b_j\|)_{i,j}$ and find its smallest value $D_{i_0,j_0}$. Then we treat two loops as non-loop curves, i.e., $\Lambda'_A = (a_i)_{i=i_0,\dots,m+i_0}$ and $\Lambda'_B = (b_i)_{i=j_0,\dots,n+j_0}$. Filling the region between $\Lambda'_A$ and $\Lambda'_B$ is tessellating triangles between them. Two types of triangles can be used:

- *A-advance*: triangle $a_i a_{i+1} b_j$, in which $i_0 \le i \le m + i_0 - 1$, $j_0 \le j \le n + j_0$.

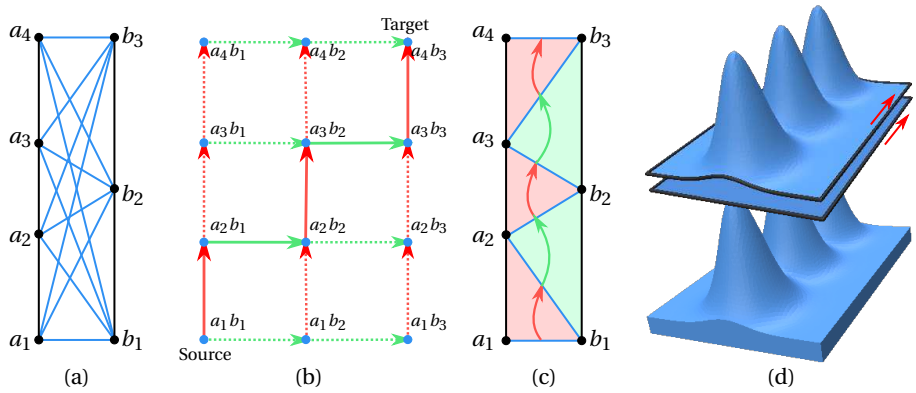



Figure 2.12: Graph-based tessellation to fill the region between two curves $(a_i)$ and $(b_i)$. Candidate bridges are drawn in blue in (a). The directed graph $\Gamma$ (b) includes the bridges as nodes and two types of edges: A-advance (red) and B-advance (green). The shortest path in $\Gamma$ corresponds to the tessellation result in (c). This method is applied to connect two components at the top of (d). The directions of the two boundary curves are aligned as the arrows indicate. The bottom of (d) is the watertight mesh after tessellating the gap.

- *B-advance*: triangle $a_i b_j b_{j+1}$, in which $i_0 \le i \le m + i_0$, $j_0 \le j \le n + j_0 - 1$.

Now we have numerous ways to arrange the candidate triangles. We formulate the problem of finding an optimal tessellation as a shortest-path problem on a directed graph [51]. For each bridge $a_i b_j$, we treat it as a node on a directed graph $\Gamma$. An *A-advance* triangle is a directed graph edge from $a_i b_j$ to $a_{i+1} b_j$ in $\Gamma$. A *B-advance* triangle is an edge from $a_i b_j$ to $a_i b_{j+1}$. Finally, each tessellation is corresponding to a path on $\Gamma$ from node $a_{i_0} b_{j_0}$ to node $a_{i_0+m} b_{j_0+n}$. We add weights on each edge of $\Gamma$. Each A/B-advance edge creates a triangle by adding a new bridge, $a_{i+1} b_j$ or $a_i b_{j+1}$, next to its predecessors. The edge's weight is defined as the length of the new bridge, as we have computed in the matrix $D_{m \times n}$. We find the shortest path on the weighted graph $\Gamma$ as the optimal tessellation, i.e., the one with the least total edge length. The process is illustrated in Fig. 2.12. For simplification, we can remove some long bridges exceeding some length tolerance $\tau$, if not corrupt the connectivity of $\Gamma$.

The shortest path problem here can be seen as a discrete optimization task that can be solved using *dynamic programming*, of which the idea is to divide the problem into simpler sub-problems in a recursive manner. Given the stripe region to tessellate, from $a_{i_0} b_{j_0}$ to $a_{i_1} b_{j_1}$, we denote the minimal cost as $C(i_0 j_0, i_1 j_1)$. Then

$$C(i_0 j_0, i_1 j_1) = \begin{cases} \| a_{i_1} b_{j_1} \|, & i_0 + 1 = i_1, j_0 = j_1; \\ \| a_{i_1} b_{j_1} \|, & i_0 = i_1, j_0 + 1 = j_1; \\ \min\limits_{i_0 \le i_2 \le i_1,\ j_0 \le j_2 \le j_1} C(i_0 j_0, i_2 j_2) + C(i_2 j_2, i_1 j_1), & \text{otherwise.} \end{cases} \tag{2.6}$$

The tessellation from $a_0 b_0$ to $a_m b_n$ with minimal cost can be found by solving the sub-problems. Dijkstra's algorithm for the shortest path problem is a special case of dynamic programming.

**The monkey saddle model** is part of the surface

$$z = x^3 - 3xy^2, \tag{2.7}$$

with $x^2 + y^2 \leq 1$. We convert the explicit function to a surface mesh by remeshing and scale it to 10cm×10cm×10cm.

For fabrication, the open surface is converted to a solid by thickening. We create an opposite surface by inverting the face orientation of the original surface. All vertices of the opposite surface are moved by $t$ in their normal directions, in which $t$ is the target thickness. At last, the original surface and the opposite surface are connected with the graph-based tessellation method presented before. For this simple case, this method works well without leading to self-intersections. For general meshes, the thickening method proposed by Wang and Chen [202] is more stable.



### 2.C.3. ALPHA SHAPE OF THE FOOT MODEL

The foot mesh in Fig. 2.7 is extracted from the base model of MakeHuman [188]. As we are not knitting socks with separate toes, the foot mesh is converted to the sock mesh by computing its alpha shape ($\alpha$-shape), a generalization of the convex hull but envelopes the input tightly, as shown in Fig. 2.13. I used [36] for it. The effect of $\alpha$-shape is similar to *morphological closing*, i.e., dilation followed by erosion, the 2D version of which is common in image processing. It can be achieved by mesh offsetting.

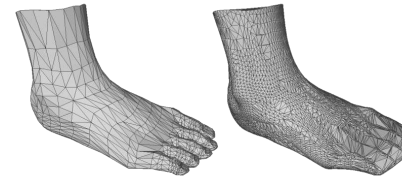


Figure 2.13: Foot model with toes and sock-like $\alpha$-shape.

### 2.C.4. SUPPORT SHAPE FABRICATION

The support objects are fabricated to provide rigid support for the knitted flexible fabrics. Two fabrication methods, fused filament fabrication (FFF) 3D printing and laser cutting, are used in our project. We used Raise3D's ideaMaker to perform slicing for 3D printing and Autodesk®'s Slicer for Fusion 360 [6] to generate laser cutting plans. The accuracy of 3D printing is much higher than that of the coarse laser cutting results. Nevertheless, laser cutting takes much less time and material costs.

# 3

# DIRECT 3D KNITTING DESIGN BY STITCH MESH GENERATION
## WHERE TO PUT SHAPING STITCHES?

## 3.1. INTRODUCTION

In the last chapter, I presented the 3D knitwear design method based on flattening, by placing knittable darts. This approach relies on user interactions and has difficulty generating complex darts (intersecting darts). In this chapter, I will present two 3D design methods that generate stitch meshes directly: one uses only short-row shaping, and the other uses both short-row and short-column shaping. These two direct design methods are more automatic and produce complex 'darts' to fit the target shapes. This chapter, together with the previous one, answers the research question (**RQ1**): how to design a knitwear and its machine knitting instructions for a given 3D shape.

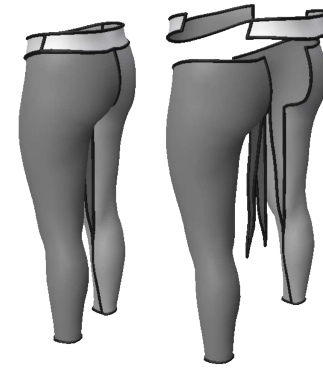


Figure 3.1: Illustration of a typical design of leggings. This is a 3D design composed of at least five fabric pieces. Right is an exploded view.

In the garment industry (sewing), designers have gone beyond darts for 3D shaping. Multiple fabric panels are used to approximate the non-developable 3D surface of the human body. This method has been applied not only in custom tailoring [206] but also in mass production (Fig. 3.1). Using knitting, we can also produce more accurate 3D shapes compared to the dart-based

Sections 3.1 and 3.2 of this chapter have been published in: **Z. Liu**, X. Han, Y. Zhang, X. Chen, Y.-K. Lai, E.L. Doubrovski, E. Whiting, and C.C.L. Wang. *Knitting 4D garments with elasticity controlled for body motion*, ACM Trans. Graph., 40(4), 2021. doi: 10.1145/3450626.3459868. A few small corrections and/or clarifications have been made to the original published text.

approach. There will be less material waste from cutting and less post-processing of sewing.

There are several existing works on direct knitting design for an input 3D surface. Igarashi et al., 2008 [65] is one of the earliest works, which only uses the short-column shape technique. Narayanan et al., 2018 [134] is the first automatic system to convert a 3D model into machine knitting instructions, which includes algorithms of field-guided remeshing, stitch tracing, and machine needle scheduling. Liu et al., 2020 [104] proposed a knitting design method for topologically disk-like surfaces with only short-rows, in which the knittability is achieved by face-shifting operations like [134]. Wu et al., 2019 [215] have proposed algorithms to convert a stitch mesh [221] into a knittable one. Their recent work Wu et al., 2022 [216] can convert a target surface into a topological disk by introducing a set of cuts, and valid knitting instructions are generated thereafter. However, none of these approaches exploit the shape distortion on knitted clothes caused by different numbers of stitches between rows. Details will be discussed in this chapter.



Stitch mesh generation is related to quadrangulation [17] since the stitch meshes used for knitting are quad-dominant. Wu et al., 2018 [214] adopt a quadrangulation method for stitch meshing. The results are applicable for rendering and animation, but they are not knittable. The quad meshing method in Lai et al., 2010 [92] can convert a mesh to a Celtic knot model, another type of textile structure. Manufacturing constraints are not considered in this work. As we have seen in Section 1.2.2, the knittable stitch mesh is a special subset of quad-dominant mesh complying with multiple constraints. Therefore, specially designed methods are required to generate *knittable* stitch meshes with low geometric distortion.

In this chapter, I will first present a short-row-only shaping approach. Then we will find its limitations. Then I will present an approach using both short-row and short-column shaping.

## 3.2. DISTORTION-CONTROLLED SHORT-ROW KNITTING

As we have seen in Section 1.4.2, the short-column shaping by stitch transfer has some limitations of being time-consuming and unstable. We will start with a short-row-only approach for its simplicity. The knittability constraints (Section 1.2.2) especially the continuity constraints are considered in the algorithm design.

Given the per stitch constant height $s_h$ and width $s_w$, we need to ideally generate a stitch mesh on the given 3D surface $\mathcal{M}$ using quadrangles of size $s_w \times s_h$ and triangles as the end of short-rows. To achieve this goal, we first generate wale curves with equal distance $s_w$, which are sampled into segments with length $s_h$. The final stitch meshes are constructed by connecting neighboring wale curves with optimized quadrangles / triangles.

### 3.2.1. GENERATION OF WALE CURVES

Given an input 3D surface $\mathcal{M}$ that is homeomorphic to a 2D disk, the geodesic distance field $F(\cdot)$ is first computed from the user-defined source (point or curve) on the input surface (see the left of Fig. 3.2(a)). Wale curves are generated from $F(\cdot)$'s isocurves that are evenly distributed with distance $s_w$.

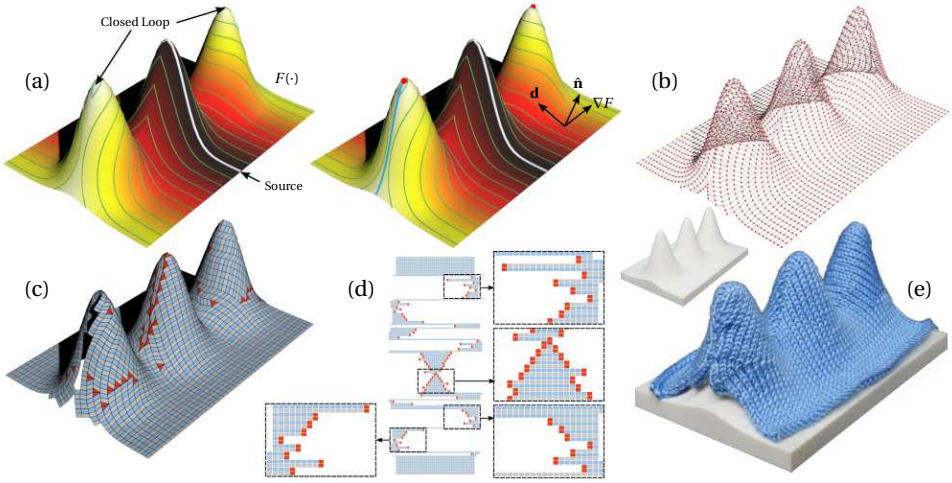



Figure 3.2: Steps for generating knittable stitch meshes on a given surface: (a–b) Isocurves of a geodesic distance-field $F(\cdot)$ are used as wale curves with segments having nearly equal length. The input mesh will be cut if there are closed wale curves. The resultant wale curves are sorted according to their values of $F(\cdot)$. (c) Optimal quadrangles/triangles are constructed between neighboring wale curves. (d) Courses are traced out and sorted on a stitch mesh to generate the final knitting map. Complex 'darts' are observed on the map. (e) The physical knitting result can be generated by following the knitting map, where the stitch size is 3.5mm × 2.8mm in this example.

A valid wale cannot be a closed curve, which has been presented as the Topological Constraints in Section 1.2.2. Therefore, if the source is a point, it should lie on the boundary of $\mathcal{M}$. If the source is a curve, it should be a curve cutting $\mathcal{M}$ into two pieces or a curve completely laying on $\mathcal{M}$'s boundary. In our project of garment design, pieces of an input garment usually have reflectional symmetry. The axial curve of symmetry that goes through the input mesh is adopted as the geodesic source. However, isocurves can still form closed loops in some extreme cases (see Fig. 3.2(a) for an example), which should be split. Inside a closed isocurve, there is a point $\mathbf{p}$ containing the local maximum of $F(\cdot)$. We can split the closed loop by computing the shortest path from $\mathbf{p}$ to the surface's boundary and cutting the surface along the path (see Fig. 3.2(a) for such a cut). This process is repeated until no closed loop is found on wales.

All wale curves generated in this way can be easily sorted by their geodesic distances, where the sign of distance can be assigned when the source curve separates an input surface into two regions – i.e., one side positive while the other side as negative. The direction of a wale curve can be determined as $\hat{\mathbf{n}} \times \nabla F$ with $\hat{\mathbf{n}}$ being the surface normal (see $\mathbf{d}$ shown in Fig. 3.2(a)). Vertices of a stitch mesh are generated on wale curves by sampling them into segments with length $s_h$. In the following step of mesh generation, quads and triangles are always constructed in pairs to ensure knittability (see details in Section 3.2.2). Therefore, an even number of segments is required for each wale. To achieve this, a wale curve's two ends are slightly extended when odd segments are obtained.

### 3.2.2. CONSTRUCTING MESHES BETWEEN WALE CURVES

Given two neighboring wale curves, there are numerous ways to connect the sampled points on them to form a stitch mesh. However, not all are knittable. We propose a graph-based algorithm to determine an optimized mesh that is knittable (Fig. 3.2(b)).

By using the short-row shaping technique, a knitting machine produces fabric pieces course-by-course from bottom to top in a zigzag manner while meeting both the in-course continuity and the continuity between courses. Without loss of generality, we assume the yarn carrier goes from left to right on courses with even indices (e.g., gray courses in Fig. 3.2) and from right to left on odd courses (e.g., blue courses in Fig. 3.2). Then, we impose the constraints of short-row knitting by defining where and how quads and triangles are presented in a stitch mesh.

A triangle defines the interior end (either head or tail) of a short-row, which is only located in the interior of a stitch mesh. After imposing the constraint of between-course continuity, triangles always appear in pairs and are one of two types:

- *Left-End Pair*: the lower triangle serves as the tail of a right-to-left knitting course while the upper one is the head of a left-to-right knitting course.
- *Right-end pair*: the lower triangle in the pair is the tail of a left-to-right knitting course and the upper one is the head of a right-to-left course.

Operators are defined below to ensure that triangles are only present in one of these two configurations. Similarly, quads are also constructed in pairs in our algorithm to ensure the knitting directions and the continuity of courses.

We employ a graph-based method to generate the mesh between neighboring wale curves, inspired by [51, 203] (Please refer to Section 2.C.1 for this approach in a simple scenario). Constructing a wale mesh is equivalent to finding the set of bridge edges (or simply called *bridges*) that connect vertices between two wale curves. Given two wale curves $A$ and $B$ that are sampled into $2n_a$ and $2n_b$ segments, our algorithm computes the optimal and knittable bridges connecting their vertices $\{a_i\}$ ($i = 0, 1, \ldots, 2n_a$) and $\{b_j\}$ ($j = 0, 1, \ldots, 2n_b$). At initialization (see Fig. 3.3(a)), we define candidate *anchor* bridges between wale vertices with even indices, where the bridge length is less than $\lambda$. A large threshold $\lambda$ will increase the computational complexity. We chose $\lambda = s_w + 2s_h$ in our implementation. Our graph-based method represents the anchor bridges as nodes on a graph $\Gamma$ (see Fig. 3.3(c)). We then generate a knittable stitch mesh by finding an optimal path that connects a selected set of nodes via operators.

Four different operators are defined to insert *intermediate* bridges between the anchors:

- *AB-advance*: Two quadrangles are formed between nodes $a_{2i}b_{2j}$ and $a_{2i+2}b_{2j+2}$ by adding two new bridges $a_{2i+1}b_{2j+1}$ and $a_{2i+2}b_{2j+2}$. This operator is illustrated as $E_{AB}$ in Fig. 3.3(b).
- *A-advance*: Two triangles are constructed between $a_{2i}b_{2j}$ and $a_{2i+2}b_{2j}$ by adding a right-end pair (see $E_A$ in Fig. 3.3(b)).
- *B-advance*: The purpose of this operator is to add two triangles as a left-end pair. However, as a left-end pair of triangles can only start from an odd course, we

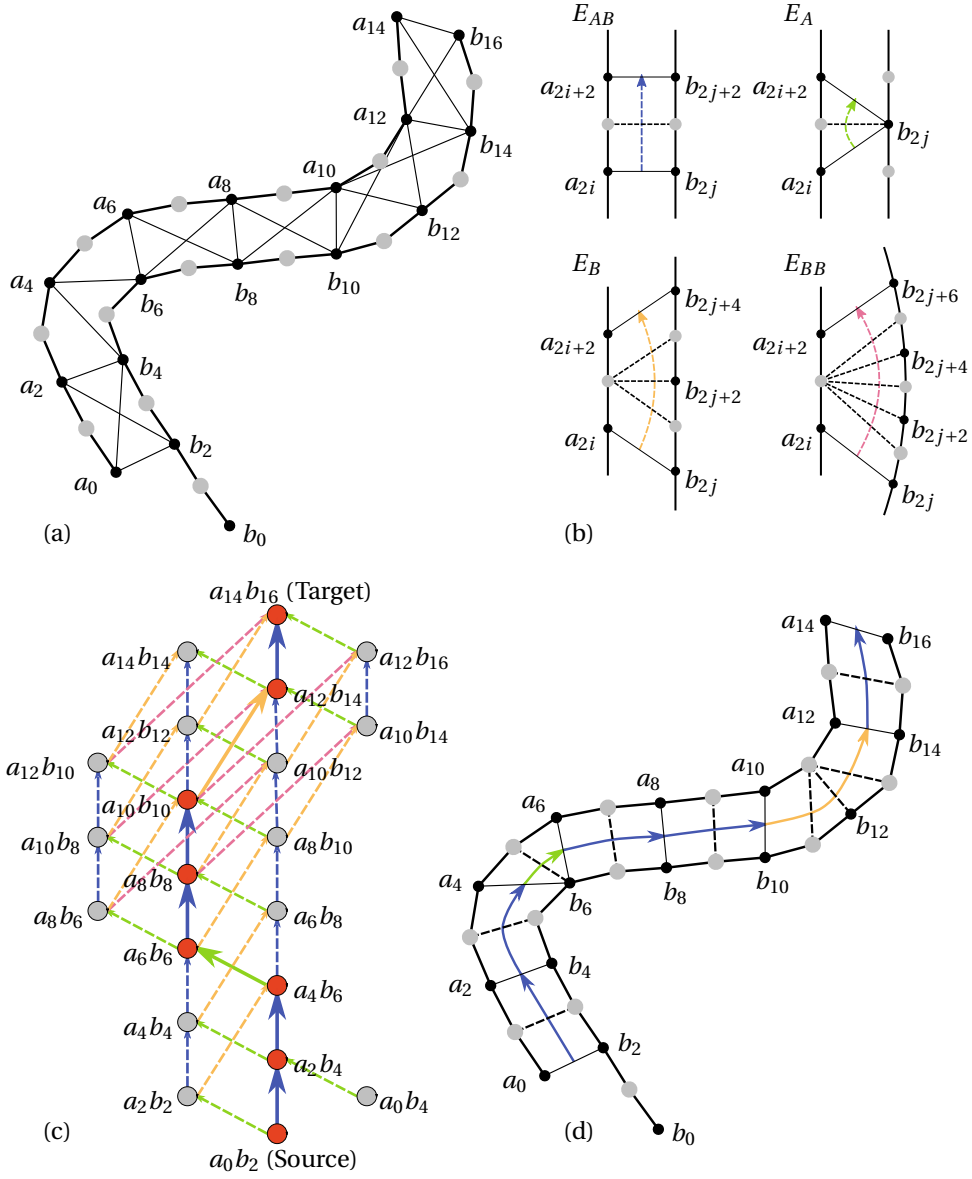



Figure 3.3: Graph-based generation of knittable stitch meshes between wale curves. (a) Points are sampled on two wale curves (even indices shown in black, odd indices in gray). Candidate anchor bridges connecting the wale curves are illustrated by line segments. (b) Four operators are introduced in our algorithm to form the mesh. These operators insert intermediate bridges (dashed lines) to construct pairs of triangles and quads between consecutive anchor bridges (solid lines). (c) The graph $\Gamma$ shows all possible configurations to tessellate the two wale curves. Anchor bridges are represented as nodes of $\Gamma$ and operators are denoted by edges of $\Gamma$ (different colors denote different types). The problem of finding an optimized tessellation between two wale curves is converted to finding the shortest path on $\Gamma$ (i.e., the bold arrows). (d) The resultant mesh tessellation corresponding to the shortest path on $\Gamma$.

add one quad, two triangles, and the other quad altogether between $a_{2i}b_{2j}$ to $a_{2i+2}b_{2j+4}$ (see $E_B$ shown in Fig. 3.3(b)).

- *BB-advance*: While right-end pairs can be consecutively presented in the stitch mesh by repeatedly applying the *A-advance* operator, consecutive left-end pairs cannot be realized by the *B-advance* operator. We define a new operator by adding one quad, four triangles, and the other quad between $a_{2i}b_{2j}$ to $a_{2i+2}b_{2j+6}$ (see $E_{BB}$ shown in Fig. 3.3(b)). Consecutive short-row ends are sometimes necessary to make the graph connected, though the knitting may form small holes.

Each operator defines a directed edge in $\Gamma$ with the weight defined as the total length of newly added bridges. If the length of any bridge introduced by an operator is longer than $\lambda$, we exclude its corresponding edge from $\Gamma$.

Given a source node and a target node, an optimal mesh can be generated by computing the shortest path between them on the graph $\Gamma$ (see Figs. 3.3(c) and (d)). Dijkstra's algorithm is employed. By using the bridge's length as weight (i.e., the metric to evaluate the quality of mesh), quads and triangles with less distortion can be generated (ref. [51]). When the wales are located near the boundary of an input surface, the lengths of two wale curves can have a significant difference. We then determine the source and the target nodes by the following method. Two vertices $a_{2i}$ and $b_{2j}$ are considered a bijective pair if $b_{2j}$ is $a_{2i}$'s closest neighbor on $B$ and vice versa. The first and the last bijective pairs are selected as the source and the target nodes in our approach, which results in stitch meshes with quads of good quality even on highly curved surfaces (e.g., the result shown in Fig. 3.2(c)). The graph-based algorithm introduced here can always generate a knittable stitch mesh after applying it to all the strips between wale curves, which is guaranteed by the four carefully defined operators. There may be cases where no path exists on $\Gamma$ connecting the source and target nodes when using a very conservative threshold $\lambda$. We can incrementally enlarge $\lambda$ by 10% until a path can be found.

As discussed in Section. 2.C.2, the shortest path problem can be seen as a dynamic programming task, which is more general.

### 3.2.3. DISTORTION CONTROL BY APEX DIFFUSION

Besides the shape of elements, we observe another source of distortion on stitch meshes – that is the location of triangles. As discussed in Chapter 1, each triangle in physical knitting is in fact formed by collapsing the corresponding stitch at the end of each course. If there are many triangles placed together, the distortion errors generated by this collapse are accumulated and can lead to significant distortion in the global shape of the knitted patch. For example, in the top row of Fig. 3.4, the global distortion of the hemisphere can be clearly observed from the results of the FEA simulation and also the physical fabrication.

To solve this problem, we introduce a *diffusion term* for the weight of an edge in the graph $\Gamma$. This term will penalize the triangle close to each other. We use the distance between apexes to measure the distance between triangles. The ideal shape of each triangle is an isosceles triangle defined by three edge lengths $s_w$, $s_w$, and $s_h$. The apex is the vertex where two edges of equal length meet. For an operator introducing triangles, the weight on its corresponding edge in the graph is defined as $L + \mu \exp(-D^2)$ to penalize

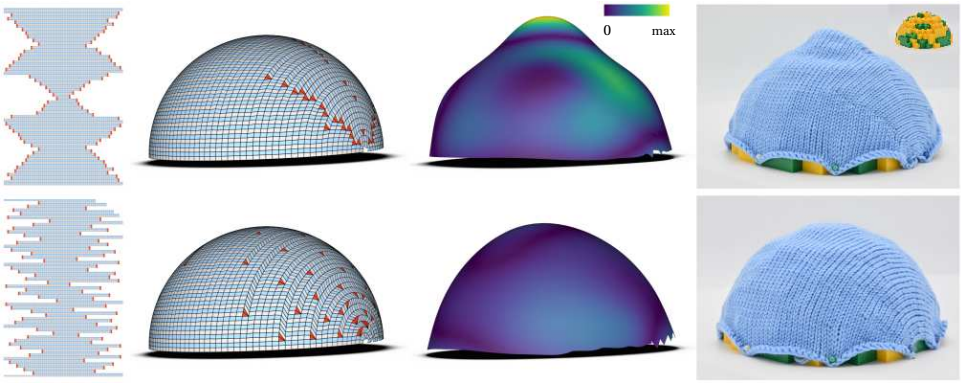



Figure 3.4: Knitted hemisphere. (Left-to-right) Knitting map, stitch mesh (before relaxation), 3D shape of knitting result simulated by FEA (i.e., relaxed), and the physical result of knitting. (Top row) The knitted hemisphere is distorted when the triangles (red elements) are crowded together. The color map visualizes the shape approximation error between the simulation result and the input 3D surface in terms of the hemisphere's radius $r$ (max = $0.15r$). (Bottom row) After applying the diffusion term onto the triangle apexes, the global shape error is significantly reduced. Note that the geometry of support for physical results is shown in the top-right corner. Stitch size calibrated in physical knitting is 3.2mm × 2.2mm. The fabrics curl at their boundaries due to the nature of the knitting structure [77].



large $L$ and small $D$ with $L$ being the total length of the newly added bridges and $D$ being the distance between the newly added apex and the apexes already existing while tessellating previous wales. A larger $\mu$ will make the apexes more diffused, but adding longer bridges and more irregular stitch elements before relaxation. We selected $\mu = 100s_w$ to reflect the influence of stitch size $s_w$. With the help of this diffusion term, scattered distribution of apexes will be generated when constructing meshes between wale curves. The distortion errors caused by triangles are therefore diffused into different regions. As a result, the 3D shape of input model can be better preserved on the knitting result (see the bottom row of Fig. 3.4).

### POST-PROCESSING OF STITCH MESH

After generating the wale curves by a geodesic distance field and constructing the stitch mesh between wale curves by a graph-based algorithm, a knittable stitch mesh can be obtained. To make it perfect at the boundary, two more operators are introduced for post-processing the stitch mesh.

- The first one is to correct pairs of triangles with their apex vertices located at the boundary of a patch (as illustrated in Fig. 3.5(a)). During physical knitting, regular stitches will be formed in these regions. To make the stitch mesh consistent with the knitting result, we replace triangles in these cases by quads.

- The second one is to correct the discontinuity between courses at the boundary of a patch (as illustrated in Fig. 3.5(b)). If the end of a boundary course is not aligned with the beginning of the next course, we add a few support stitches as accessory into the next course to avoid the long float.

Both operators are only applied to the boundary, which will not influence the interior global shape of a knitted patch.

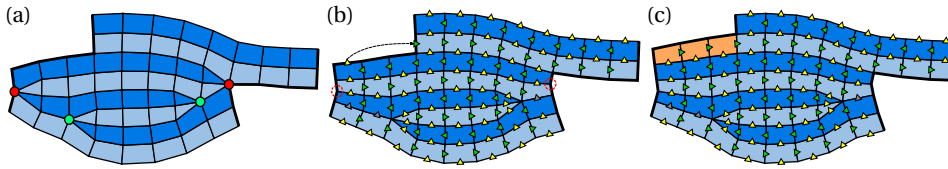



Figure 3.5: Two operators of boundary correction: (a) A pair of triangles is replaced by quads if their apex vertex is located on the boundary (denoted by red dots and circles in dash lines), and the triangles will remain when their apex vertex is in the interior (denoted by green dots). (b) The discontinuity between courses is corrected by adding support stitches as quads (displayed in orange) in the course next to a boundary course.



### 3.2.4. KNITTING MAP GENERATION

The knitting map can be extracted from a stitch mesh assigning each quad / triangle with its corresponding wale and course indices. The wale index of an element can be easily obtained from the indices of wale curves, which have been sorted. Now the problem is how to assign the course index for each element on a stitch mesh.

Considering elements adjacent to the same wale curve, two elements belong to the same course if and only if they share an edge on the wale curve. We determine elements in the same course by a flooding algorithm. After that, the course indices are determined by 1) converting neighboring courses into connected nodes on a graph and 2) applying a topological sorting algorithm on the graph to determine the order of courses in knitting.

#### TOPOLOGICAL SORTING OF COURSES

A topological sorting algorithm is employed in our implementation to determine the order of courses in knitting process by considering the between-rows continuity.

Two elements belong to the same course if and only if they share an edge on the wale curve. First, we determine elements in the same course by a flooding algorithm.

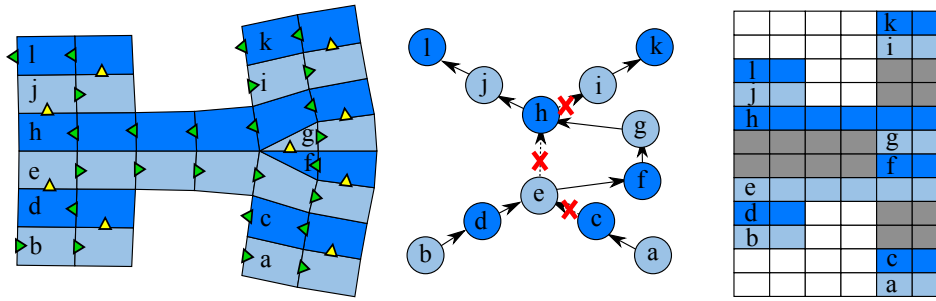



Figure 3.6: Given a mesh with courses determined by flooding (left), each course is represented as a node on a directed graph (middle). The order of courses can be determined by topological sorting. After sorting, the stitch mesh can be converted into a knitting map (right).

After that, graph-based *topological sorting* is used to find the order of courses. A *directed acyclic graph* (DAG) can be constructed by defining each course as a node on the graph (see Fig. 3.6 for an example). Directed edges are defined between two courses only when they have any neighboring elements, being the same as the stitch mesh to knitting map conversion in Section 1.3.1. The edge direction is the same as the wale curve's direction. If the fabric can be knitted with a single yarn, the topological order is unique. In some cases like Fig. 3.6, we sort the courses greedily. When a node has multiple predecessors or successors, we connect it to the nearest one, i.e., connect $e$ with $d$ and connect $h$ to $j$. The edge $c \rightarrow e$ is removed because $d$ is closer to $e$. The edge $h \rightarrow i$ is removed similarly. The edge $e \rightarrow h$ is excluded from the result because of $f$ and $g$.

Note that, two post-processing operators introduced in Fig. 3.5 are applied to the boundary of a stitch mesh after the DAG-based course sorting. The second operator (in Figs. 3.5(b-c)) fixes the discontinuity only when

1. there is no other course connected to the second course in the graph, and
2. the first course has not been connected to any other course.

For example, it cannot be applied to connect $h$ and $i$ in Fig. 3.6 unless the courses $j$ and $l$ do not exist (i.e., prevented by the first condition mentioned here). Similarly, the operator cannot be applied to connect $c$ to $e$ according to the second condition. In short, we never form any tree in the graph.

### 3.2.5. RESULTS AND DISCUSSION

We tested the performance of the short-row based 3D shaping technique proposed here. The source code is publicly available [1]. When generating knitting maps for production, a common industrial approach is to flatten 3D surfaces into 2D panels and then generate grids of stitches on the 2D panels. To flatten regions with high curvature, darts are added manually during the flattening step (see Fig. 3.7(a)). Different from clothes made by sewing, darts on knitwear will automatically be knitted together. However, the surface flattening process still introduces shape distortion errors. The result obtained from our short-row based 3D shaping technique shows significant improvement in fit – see Fig. 3.7(b), with the most pronounced difference in the waist region. When the central front/back lines are selected as the source to compute the geodesic distance-field $F(\cdot)$, the stitches generated by our approach are nearly symmetric.

As demonstrated in Fig. 3.4, we control the distortion of knitting a 3D surface by diffusing the apexes on the stitch mesh. We also verified our distortion control method on real garments like the skull cap shown in Fig. 3.8. The knitting map generated by short-row approach without apex diffusion produces many unwanted bumps (top row of Fig. 3.8), which can be completely eliminated by using the apex diffusion (bottom row of Fig. 3.8). A few pre-processing steps are applied to generate the stitch mesh and the knitting map for the skull cap. First, the topology of the original design with eye hole is not homeomorphic to a disk. It is unwrapped by adding a seam from the nose to neck. In both cases, the intersection curve between the model and its symmetrical plane is used as the geodesic source. Closed-loops are found on the isocurves generated from

[1] https://github.com/zishun/KnittingShortRows2021

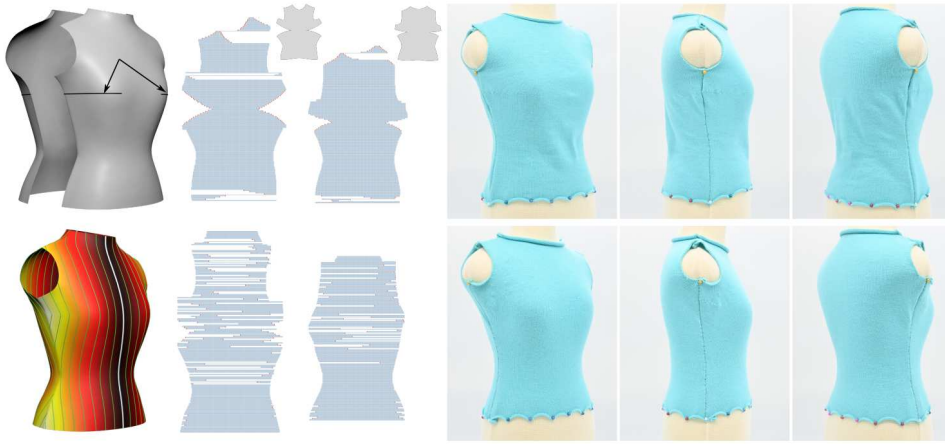


Figure 3.7: Comparison of knitting maps generated from the flattened 2D panels with manually added darts (a) vs. from the 3D surface with the help of geodesic distance-field (b). The same set of pinholes is used for the two tests. The physical knitting results show the advantage of our approach with the improved fit of the final garment. Note that the 2D panel results are generated by ARAP flattening without applying the constraints on darts presented in Chapter 2.

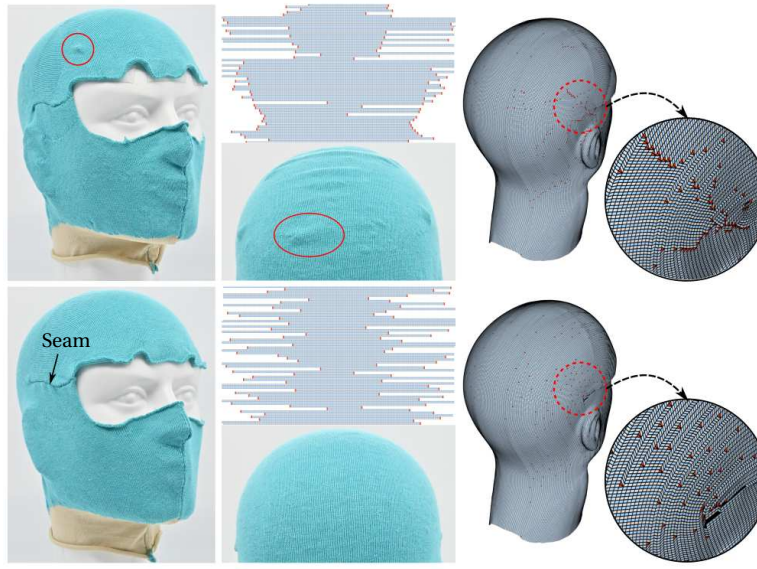



Figure 3.8: Knitted skull cap. (Top) Without apex diffusion: bumps form when the shape approximation errors at triangles are accumulated together (see red circles). (Bottom) With apex diffusion: the bumps can be completely eliminated. Accessory seams are added from the top of the ear to canthus to open the closed-loops formed on isocurves of the geodesic distance-fields.

this geodesic source; therefore, the method presented in Section 3.2.1 is applied to open closed-loops by adding new seams (see the ones from ear to canthus). Moreover, to make it easier to take on / off, a highly elastic yarn is employed to knit the neck part.

In extreme cases of complex shapes (e.g., the triple-peak in Fig. 3.2), topology with closed loops may be formed by the isocurves of the geodesic distance-field (although this was not found in our garment examples). Additional seams will need to be introduced, which may not be acceptable in specific applications. Models must also have disk topology, potentially requiring conversion using additional seams [224]. The triple-peak is originally designed to demonstrate the necessity of cutting closed loops. However, we can avoid the cutting (i.e., avoid the local maxima in the distance field) by rotating the source curve by 90 degrees. If the target shape is a $3 \times 3$-peak, the cutting will be necessary and the method presented in this section still works.

Besides closed isocurves, the isocurves may also have sharp corners (around the central peak of the triple-peak in Fig. 3.2), which are also not suitable for knitting. Because the knitted wales can hardly produce sharp corners. Both the closed isocurves and the un-smoothness of isocurves are caused by cut locus [42] of geodesic distance fields. A cut locus is a point on the surface that has multiple geodesic paths connected to the source of the geodesic distance field.

In the thesis of Narayanan [133], a sampling-based method is presented to generate short-row-only knitting maps, which are named transfer-free patterns. The implementation of that approach is more complex than ours.

There can be several interesting extensions on the hemisphere example (Fig. 3.4). We may knit the map twice to get a sphere. We can also knit a series of hemispheres with the radius gradually reduced to get a solid ball after rolling the small end into the core.

### 3.2.6. CONCLUSION

In this section, we propose a method to generate machine knitting code by only using the short-row shaping strategy. The method is more automatic and accurate compared to the flattening-based method. The continuity is achieved by a graph-based tessellation method. The distortion is controlled by a diffusion term.

Compared to short-column shaping, the short-row-only shaping is more efficient (Section 1.4.2). Moreover, the barrier of short-row shaping is much lower while the increase/decrease shaping strategy requires knitting machines equipped with more needle beds and more complicated mechanical structures. In the next section, we will present a knitting method for complicated surfaces by using increase/decrease shaping together with short-row shaping.

## 3.3. RELIABLE SHORT-COLUMN KNITTING BY OPTIMIZATION

### 3.3.1. INTRODUCTION

Short-row and short-column shaping are two important techniques for 3D knitting. Only using the short-row shaping technique for knitting instruction generation has some limitations, which can be explained by using the short-row shaping methods presented in the previous section. Isocurves on a geodesic distance field are extracted with a constant step distance to generate curves with a constant width as the boundaries of wales.

These curves are later subdivided into line segments with similar lengths, and the segments on two neighboring column curves are tessellated into knittable stitch meshes. The first problem of this method is the possibility of generating close-loops that cannot lead to valid wales in knitting. Additional seams need to be added, which will need manual operations to be sewed back – i.e., contradicting the principle of automatic machine knitting. Secondly, the smoothness of isocurves generated by this method is not guaranteed (i.e., sharp turns are formed). Unfortunately, sharp turns cannot be formed by wales on the physically knitted fabrics. A knitting plan with sharp turns in wales will result in unwanted distortion between the input shape and the physical knitting result. Both cases are shown in Fig. 3.2.

Modern knitting machines support both short-row and short-column shaping techniques to overcome these difficulties. Specifically, when highly distorted isocurves are generated, short-columns can be inserted to 'straighten' the isocurves (i.e., the wales of stitches). The physical realization of short-columns on the digital knitting machine is based on a technique that transfers the loops of stitches between needle-beds at the courses of its beginning and ending stitches (called *transfer stitches*). This transfer technique however can often lead to manufacturing problems such as miss-transfer, yarn stretching, yarn abrasion, etc. Another minor disadvantage of stitch transfer is that a knitting process with more stitch transfer operations is more time-consuming than those processes with only short-row shaping (see Section 1.4.2). Details of 3D shaping by using transfer stitches are given in Section 1.4.2.

To obtain machine instructions that result in reliable knitting, industrial practice is to test different knitting strategies by trial-and-error until finding one with a small number of transfer stitches. The problem to be solved in this work is how to effectively generate a stitch mesh with a minimized number of transfer stitches in an automatic way. We propose a field-based method to tackle this challenge by 1) minimizing the divergence-based metric on a governing field and 2) generating stitch meshes tightly aligned with the governing field.

### RELATED WORK

**Computational Knitting** Although computational knitting has been developed rapidly in recent years, the reliability of machine knitting caused by short-column shaping has not been considered and discussed in prior research. A field optimization-based approach is developed in our work to tackle this problem.

**Scalar / Vector Fields on Mesh** Scalar fields and vector fields are widely used in geometric computing. The governing field employed in our work is scalar. We optimize it via the divergence of its normalized cogradient field, which is a vector field. Vaxman et al., 2017 [197] and de Goes et al., 2016 [38] reviewed the theories, methods, and applications of vector field processing on meshes. Vector fields have been widely used to guide quad-mesh generation. Stitch meshes generated for digital knitting are in general quad-triangle mixed meshes with additional manufacturing constraints. Our field-governed stitch mesh generation is a variant of [134] but more tightly aligned with the isocurves of the field.

There are some other field generation methods related to our work. Stripe patterns

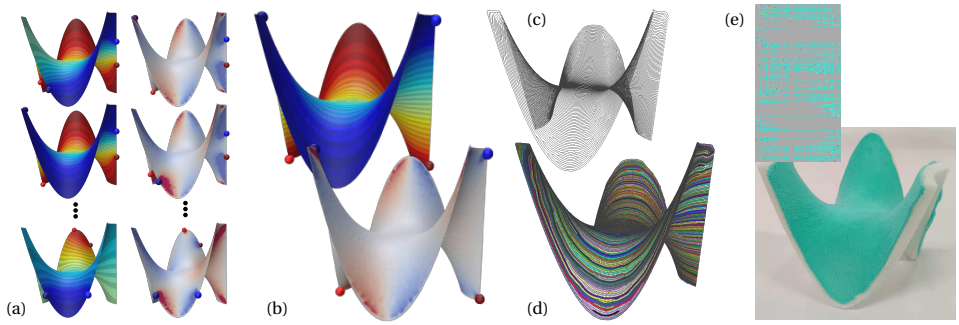



Figure 3.9: Overview of our stitch mesh generation method. (a) We optimize the Dirichlet boundary conditions on the boundary of a given surface $\mathcal{M}$ to determine an optimized scalar field, where the color maps provide the field-values $f(\mathbf{x})$ ($\forall \mathbf{x} \in \mathcal{M}$) and blue / red dots define the endpoints of boundary regions with Dirichlet conditions – i.e., $f(\mathbf{x}) = 0$ between blue dots and $f(\mathbf{x}) = 1$ between red dots. (b) The 'best' field is determined by minimizing the divergence-based metric $I(f)$, distributed values of which are given as maps next to the scalar field. (c) Wale curves are extracted by an algorithm as a variant of the course extraction algorithm presented in [134]. (d) Stitch mesh can be generated by knittability-ensured tessellation between wale curves (Section 3.2.2). (e) The final knitting instruction can be obtained from the stitch mesh and is represented as a bitmap to fabricate the physical result.



research conducted in [89] can generate short-column like patterns on meshes, which however is not knittable in general. Sharp et al. [162] computed parallel transport to generate the 'most parallel' vector fields. Vector fields with controlled smoothness and singularity are generated in [88]. All of these approaches need initial fields or predefined boundary conditions to be given, thus not fully automatic. In this work, we optimize a governing field without any predefined initial or boundary conditions. Moreover, the field optimized by our method can be used as the input of the other methods when other specific properties are preferred.

Several methods have been suggested in [134] for giving the boundary conditions to compute a governing field. For closed surfaces without boundaries, they suggested using local extrema of the mesh curvature as starting and ending points to effectively capture mesh features. However, when knitting a 3D surface with disk-like topology, the starting and ending courses/wales must be located on the boundary. They also suggested using the Fiedler vector [97], a low-order eigenvector of the mesh Laplacian, as the guidance field. However, such a field has local extrema in the interior region in some cases – leading to closed wale curves that are invalid for knitting.

**Optimization** Numerical optimizations are widely used in geometric modeling problems. Most methods rely on derivatives of the objective function. However, analytical derivatives are difficult to be obtained in some problems. The differentiation is also difficult to be approximated when the evaluation of the objective is very expensive. Black-box optimization or derivative-free optimization is useful in these situations.

Genetic algorithms, as a derivative-free method, have been used for shape registration [167] and structure generation [193]. Simulated annealing is used for 3D shape correspondence [63] and furniture arrangement design [219]. We employ Bayesian opti-

mization [47] in this work because of its good performance in hyper-parameter optimization for machine learning models that have been proved recently. Our problem involves a very expensive black-box function and also has variables with relatively low dimensions. These properties fit well with Bayesian optimization.

### OUR METHOD

We propose a field-based method to tackle the problem of reducing the number of transfer stitches for reliable knitting. An overview of our method is presented in Fig. 3.9. Akin to [134], a scalar field $f(\mathbf{x})$ is computed on a given mesh surface $\mathcal{M}$ in disk-like topology and the isocurves $f(\mathbf{x}) = \text{const}$ ($\forall \mathbf{x} \in \mathcal{M}$) are employed to generate the basic structures of a stitch mesh for $\mathcal{M}$. The field $f(\mathbf{x})$ is determined by solving a Laplacian equation on $\mathcal{M}$ with Dirichlet boundary conditions imposed on $\partial\mathcal{M}$. The structure generated by our algorithm is dual to [134], which uses isocurves to generate courses. Differently, we use isocurves as guidance to form wales.



As illustrated in Fig. 3.9(a), different governing fields can be generated when imposing different Dirichlet boundary conditions as regions $R_s \subset \partial\mathcal{M}$ with $f(\mathbf{x}) = 0$ ($\forall \mathbf{x} \in R_s$) and $R_e \subset \partial\mathcal{M}$ with $f(\mathbf{x}) = 1$ ($\forall \mathbf{x} \in R_e$). In Fig. 3.9(a), the boundary between two blue dots (as corner of $\partial\mathcal{M}$) is for $R_s$ and the boundary between two red dots is for $R_e$. We propose to optimize the field $f$ by adjusting the position of these four corner points – i.e., different scalar fields can be obtained as shown in Fig. 3.9(a).

It is not straightforward to directly predict the number of transfer stitches by giving the governing field $f(\mathbf{x})$ and the stitch size. Evaluating the number of transfer stitches by generating all wale curves is time-consuming when including it as a step in the iteration of optimization. Instead, we propose a new divergence-based metric defined on $f(\mathbf{x})$ (Section 3.3.2), the value of which is nearly monotonic to the number of transfer stitches. Bayesian optimization, a derivative-free method, is employed to find the 'best' field that minimizes the number of transfer stitches (details are provided in Section 3.3.5). Computational results demonstrate that this method works well in practice. An optimization result is given in Fig. 3.9(b) together with the wale curves and the corresponding stitch mesh.

Our method for generating wale curves and knittable stitch mesh is an extension of the graph-based tessellation method presented in Section 3.2.2, where isocurves from geodesic fields are employed and no wale curves for short-columns will be generated. Differently, the extension presented in this section can work for a wide range of scalar fields – the requirements are discussed in Section 3.3.3). Wale curves for short-columns are generated automatically by our algorithm (Section 3.3.4). Study shows that stitch meshes generated by our method can align with the isocurve structure of governing fields better than [134] and no shifting operation is needed. See Figs. 3.9(b) and (c) for the stitch mesh and its corresponding knitting map generated by our method.

In summary, we make the following technical contributions:

- A field optimization-based framework to generate stitch meshes with a minimized number of transfer stitches;
- A new divergence-based measurement as an objective function of the optimization framework for knitting code generation;

- An enhanced algorithm for extracting wale curves so that they can better align with the isocurves of a governing field.

Field optimization has not been considered in the literature on digital knitting yet. The pipeline proposed in this section has been tested by a variety of examples presented in Section 3.3.6. Physical knitting is conducted on both a high-end digital knitting machine and a household manual machine to verify the performance of our approach.

This section is organized as follows. We first introduce the divergence-based metric for field optimization in Section 3.3.2. After discussing the method to compute a valid governing field $f(\mathbf{x})$ (Section 3.3.3), we present the algorithm for extracting wale curves that are closely aligned with its isocurves (Section 3.3.4).

### 3.3.2. METRIC FOR FIELD EVALUATION

Given $\mathcal{M}$ as a triangular mesh surface in disk-like topology, we can generate a scalar field $f(\mathbf{x})$ on $\mathcal{M}$ without any local extrema to serve as the governing field for stitch mesh generation. A metric $I(f)$ is defined below to work as an approximate measurement for the number of transfer stitches to be generated – i.e., according to the analysis of knitting reliability (see also Section 1.4.2), the smaller the better.

For $f(\mathbf{x})$, we normalize its gradient $\mathbf{g} = \nabla f$ into a unit vector field denoted by $\hat{\mathbf{g}}$. Here the operator $\hat{\mathbf{v}}$ is defined for any non-vanishing vector field on $\mathcal{M}$. We define the rotation operator $\mathbf{J} : \mathbf{v} \mapsto \mathbf{n} \times \mathbf{v}$ with $\mathbf{n}$ being the surface normal at the location where the vector $\mathbf{v}$ is located. A rotated gradient field $\mathbf{Jg}$ is also referred to as the cogradient in [197].

$\mathbf{J}\hat{\mathbf{g}}$ as the normalized $\mathbf{Jg}$ defines the tangential direction of isocurves on a field. Given a non-boundary point $\mathbf{x}$ on $\mathcal{M}$, we can extract an isocurve of $f$ passing $\mathbf{x}$, a small neighborhood of $\mathbf{x}$, and sample two points $\mathbf{x}_+, \mathbf{x}_-$ in the neighborhood but located on two sides of the isocurve satisfying $f(\mathbf{x}_+) > f(\mathbf{x}) > f(\mathbf{x}_-)$.

1. If both $\mathbf{J}\hat{\mathbf{g}}$ defined on $\mathbf{x}_+$ and $\mathbf{x}_-$ are pointing to the isocurve, it means that the distance between neighboring isocurves tends to become smaller. This indicates the merging of wale curves as the end of a short-column.
2. If $\mathbf{J}\hat{\mathbf{g}}$ on $\mathbf{x}_+$ and $\mathbf{x}_-$ are pointing away from the isocurve, the distance change between neighboring isocurves has a trend to be larger. An additional column needs to be inserted here – the beginning of a short-column.

Both of these two cases relate to the divergence of the vector field $\mathbf{J}\hat{\mathbf{g}}$, where the first case gives a negative divergence and the latter case has a positive divergence. The relationship between $f(\mathbf{x})$, its isocurves, and the divergence of its normalized cogradient field $\mathbf{J}\hat{\mathbf{g}}$ has been illustrated in Fig. 3.10.

We propose to use the following metric as an approximate measurement of the number of transfer stitches that appeared on the stitch mesh generated under the guidance of $f(\mathbf{x})$.

$$I(f) = \int_{\mathcal{M}} \left| \operatorname{div} \mathbf{J}\hat{\mathbf{g}} \right| \mathrm{d}A \tag{3.1}$$

For vector fields defined on a local manifold region $X$, we can have the following formula derived from the Stokes theorem (ref. [197]):

$$\int_X \operatorname{div} \mathbf{v} \, \mathrm{d}A = \int_{\partial X} \langle \mathbf{v}, \mathbf{N} \rangle \, \mathrm{d}s \tag{3.2}$$

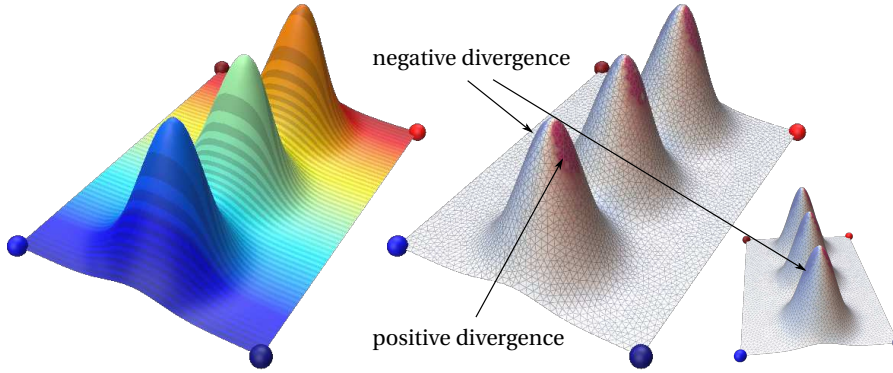



Figure 3.10: The illustration to explain the relationship between a scalar field $f(\mathbf{x})$, its isocurves (left), and the divergence of its cogradient field $\mathbf{J}\hat{\mathbf{g}}$ (right).

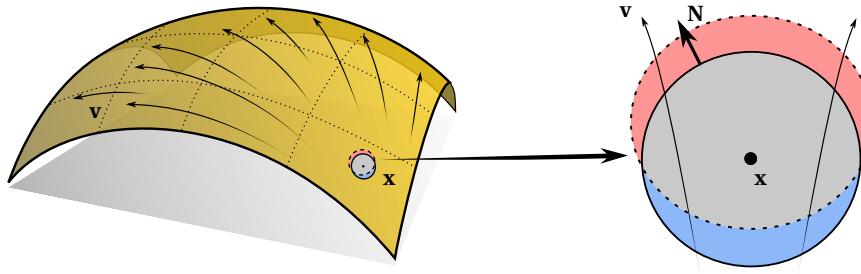



Figure 3.11: An infinitesimal region bounded by the solid circle (gray & blue) around a point $\mathbf{x}$ on the manifold is deformed by the vector field $\mathbf{v}$. Its image is the gray & red region (bounded by the dashed circle) after an infinitesimal time step. The right-hand side of Eq. (3.2) is the area difference (red minus blue), which is directly related to the divergence.

where $\mathbf{N}$ is the outwards normal vector defined on the boundary of the region $X$. The value of div $\mathbf{v}$ in fact measures the rate of area change under the flow generated by $\mathbf{v}$ (ref. [186]), which is also illustrated by Fig. 3.11. Positive divergence indicates the increase of area, the region is widened so that new columns should be inserted. Negative divergence indicates the decrease of area, the region is narrowed where some columns should be terminated.

For the metric defined in Eq. (3.1), the divergence on the normalized vector field $\mathbf{J}\hat{\mathbf{g}}$ instead of $\mathbf{J}\mathbf{g}$ is evaluated. This is because the transfer stitches only happen in the regions with area change caused by the direction change of cogradient field. In other words, the area change caused by the magnitudes of vectors should be eliminated. To penalize both the increasing (i.e., div $\mathbf{J}\hat{\mathbf{g}} > 0$) and decreasing (i.e., div $\mathbf{J}\hat{\mathbf{g}} < 0$) of short-columns, a field with nearly zero divergence is demanded. As the absolute value is used in Eq. (3.1), we cannot directly evaluate $I(f)$ by using the right of Eq. (3.2) to compute integral on the

boundary.

**On Geodesic distance field** For vector fields defined on manifolds, the curl is closely related to the divergences (ref. [197]) as

$$\operatorname{curl} \mathbf{v} = -\operatorname{div} \mathbf{J}\mathbf{v}. \tag{3.3}$$

It reveals the connections between this approach and the previous methods based on the geodesic distance field (e.g., Section 3.2 and [104]). If $f$ is a geodesic distance field without local extrema, that gives $\|\nabla f\| = 1$ and therefore $\hat{\mathbf{g}} = \mathbf{g}$. This leads to

$$\operatorname{div} \mathbf{J}\hat{\mathbf{g}} = \operatorname{div} \mathbf{J}\mathbf{g} = -\operatorname{curl} \mathbf{g} = -\operatorname{curl} \nabla f = 0 \tag{3.4}$$

where value zero is obtained as a gradient field is in general curl-free. By Eqs.(3.1) and (3.4), we know that $I(f) \equiv 0$ on a geodesic distance field. This reflects that no short-column is generated when using a geodesic distance field as a guide to generate wale curves.

**Evaluation in discrete form** A discrete form of the divergence evaluation on an input mesh $\mathcal{M}$ is necessary for numerical computation. Here we use the face-based representation of a vector field [38] so that the gradient of a field inside a triangle $\triangle ijk$ is a constant vector $\mathbf{g}_{ijk}$ defined by the scalar-field value defined on its three nodes. We can also obtain $\mathbf{v}_{ijk} = \mathbf{J}\hat{\mathbf{g}}_{ijk}$ in each triangular face.

Using Eq. (3.2) and the 1-ring face neighbors of a non-boundary vertex $i$ as the region $X$, we can have

$$[\operatorname{div} \mathbf{v}]_i = \frac{\sum_{\triangle ijk \in N(i)} \langle \mathbf{v}_{ijk}, \mathbf{J}\mathbf{e}_{kj} \rangle}{\sum_{\triangle ijk \in N(i)} A_{ijk}} \tag{3.5}$$

with $N(i)$ defining the faces adjacent to the vertex $i$. $\mathbf{J}\mathbf{e}_{kj} = \mathbf{n}_{ijk} \times \mathbf{e}_{kj}$ is an outwards vector defined as $\mathbf{N}$ for $\partial X$ in Eq. (3.2), and $A_{ijk}$ is the area of the triangle $\triangle ijk$ adjacent to the vertex $i$. With this discretization, the metric in Eq. (3.1) can be evaluated by

$$\bar{I}(f) = \sum_{i \in \mathcal{M} \setminus \partial\mathcal{M}} \bar{A}_i \left| [\operatorname{div} \mathbf{v}]_i \right|, \tag{3.6}$$

where $\bar{A}_i$ defines the Voronoi area of the vertex $i$.

### 3.3.3. HARMONIC FIELD AS GUIDANCE

The automatic machine knitting method presented in [134] computes a Harmonic field on a given model $\mathcal{M}$ as guidance to generate knitting courses. The boundary condition is specified by user interaction in their method. In short, given the Dirichlet boundary condition as $R_s$ and $R_e$ specified on $\partial\mathcal{M}$, the Laplacian equation below is solved.

$$\begin{aligned} \Delta f &= 0 \\ \text{s.t. } f(\mathbf{x}) &= 0 \quad (\forall \mathbf{x} \in R_s) \\ f(\mathbf{x}) &= 1 \quad (\forall \mathbf{x} \in R_e) \end{aligned} \tag{3.7}$$

The result is a harmonic field defined on $\mathcal{M}$. Harmonic functions on a topologically disk-like mesh have the following advantages.

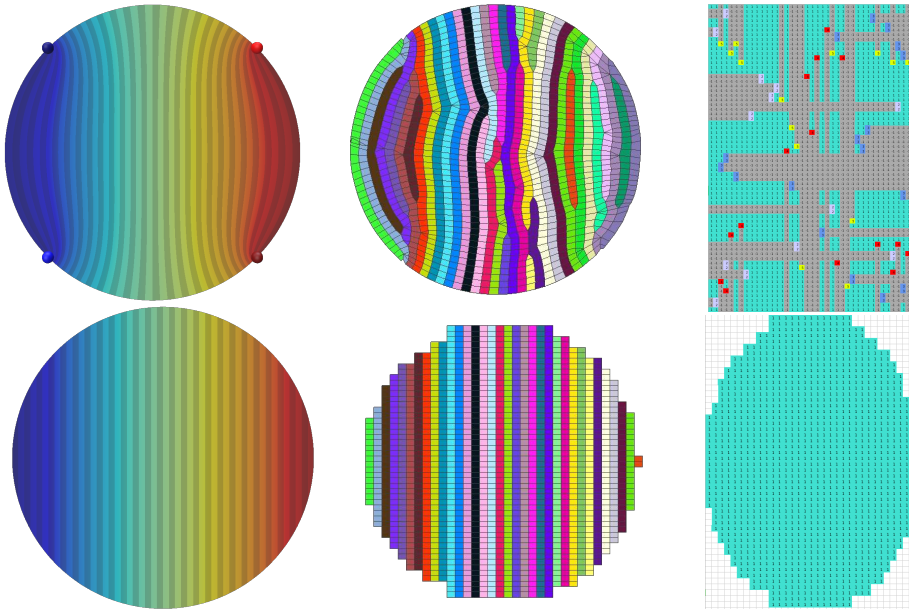


Figure 3.12: Comparison of wale curves generated from the boundary-aware field (top) vs boundary-unaware field (bottom). The given shape is a flat disk.

- Firstly, there is no interior local extrema on the resultant field $f(\mathbf{x})$ as Laplacian at a local extremum cannot be zero. This ensures that no close loop (such as the one shown in Fig. 3.5) will be formed by the isocurves of $f(\mathbf{x})$.

- Secondly, harmonic functions are boundary aware – i.e., the isocurves of $f(\mathbf{x})$ are either coincident with or orthogonal to the boundary $\partial\mathcal{M}$ (see Fig. 3.12 for an illustration). When this boundary-aware property is not preserved, the stitch mesh generated from the governing field may contain wales starting from the left / right boundary but not from the starting course of knitting. The outside widening operation [192] is needed to knit such wales, which could fail if insufficient dragging force is applied to the already knitted stitches.

Different types of scalar fields, other than harmonic functions, can also be employed as guidance for stitch mesh generation as long as there is no interior local extrema. The divergence-based metric defined in Eq. (3.1) can also be applied to find a 'best' boundary condition for these fields.

### 3.3.4. KNITTABLE STITCH MESH GENERATION

Wale curves are first generated by following the isocurves of an optimized scalar field $f(\mathbf{x})$. Note that the distance between wale curves is determined by the required width of stitches. After dividing the wale curves into line segments according to the required height of stitches, the knittable stitch meshes are constructed between wale curves by the graph-based tessellation (Section 3.2.2).

**Algorithm 3.1** Wale curve extraction

**procedure** `WaleCurveExtraction`($f : \mathbf{x} \mapsto [0,1], s_w \times s_h$)
  $V_a \leftarrow \{\mathbf{x} : f(\mathbf{x}) = 0\}$ ▷ active vertices
  $C_a \leftarrow$ the curve formed by $V_a$. ▷ active curve
  **while** $C_a \neq \emptyset$ **do**
    $\tau \leftarrow \max_{\mathbf{x} \in V_a} f(\mathbf{x})$ ▷ isovalue of the reference isocurve
    $d(\cdot) \leftarrow$ geodesic distance field sourced from the active curve $C_a$.
    Cut the mesh along the isocurve on the geodesic distance of isovalue $d(\cdot) = s_w$.
    Remove the region with $d(\cdot) < s_w$. Add new boundary vertices into the active vertex set $V_a$.
    **Call** Alg. 3.2: $C_a \leftarrow$ UPDATEACTIVECURVE($V_a, \tau$)
  **end while**
**end procedure**

**Algorithm 3.2** Update active curve

**procedure** `UpdateActiveCurve`($V_a, \tau$)
  $C_a \leftarrow$ the curve formed by $V_a$.
  **if** $\tau < \min_{\mathbf{x} \in V_a} f(\mathbf{x})$ **then**
    **return** $C_a$ ▷ the active curve is in front of the reference isocurve
  **end if**
  **if** found $s \subset C_a$ and SEGMENTACCEPTABLE($s$)=**True then**
    **return** $C_a \leftarrow s$ ▷ found a segment as a short curve
  **else**
    **return** $C_a$ ▷ cannot find an acceptable segment
  **end if**
**end procedure**
**procedure** `SegmentAcceptable`($s$)
  **return if** ($\tau > \max_{\mathbf{x} \in s} f(\mathbf{x})$ **and length**($s$) $> 2 s_h$)
**end procedure**

**Wale Curve Extraction** The wale curve extraction algorithm presented here is based on an extension of the algorithm introduced in [134], where the course curves (actually cycles) are generated with additional short-row courses by following the isocurves of a governing field. Differently, we adopt it here to generate wale curves so that short-columns are formed.

The main idea is as follows. Starting from the minimal value boundary of a governing field $f(\mathbf{x})$, the wale curves are extracted one after another, and the neighboring wale curves should have a nearly fixed distance according to the required width of stitches. The reference isocurve of a wale curve is defined by the maximal isovalue among all points on its previous wale curve. The region between a wale curve between its reference isocurve needs to be filled by short wale curves (see Fig. 3.13 for illustration). Pseudo-code of this algorithm can be found in Algorithm 3.1 and Algorithm 3.2.

However, such a method, which is basically the same as [134], may fail if the region

**Algorithm 3.3** Update the reference $\tau$

**procedure** `UpdateReference`($V_a, \tau$)
  $\mathbf{x}^* \leftarrow \arg\min_{\mathbf{x}\in C_a} f(\mathbf{x})$
  $d(\cdot) \leftarrow$ geodesic distance field sourced from $\mathbf{x}^*$
  $S_\tau \leftarrow \tau$-isocurve on $f$
  **if** $\max_{\mathbf{x}\in S_\tau} d(\mathbf{x}) > s_w$ **then** ▷ A large region should be filled with short wales. The reference $\tau$ should be smaller.
    $S_w \leftarrow s_w$-isocurve on $d(\cdot)$
    **return** $\tau \leftarrow \max_{\mathbf{x}\in S_w} f(\mathbf{x})$
  **end if**
  **return** $\tau$ ▷ Do not need to update.
**end procedure**



between a wale curve and its reference isocurve radically becomes very big (as illustrated in the top row of Fig. 3.13). We improve this algorithm by developing an additional step to determine a finer reference value for the wale curve generation, which is realized by a method presented in Algorithm 3.3. When calling Algorithm 3.3 between Step 8 and 9 in Algorithm 3.2 before calling Algorithm 3.2, the resultant wale curves can align with the isocurves of a governing field more tightly. The difference for the algorithms without vs. with this refinement step of reference values (i.e., Algorithm 3.3) can be found by the progressive results shown in Fig. 3.13.

**Wale Mesh Generation** After extracting the wale curves, we uniformly sample vertices on them using the stitch height. An even number of segments are sampled on each wale curve for the sake of simplicity. Then wale meshes are constructed using the graph-based method presented in Section 3.2.2. The difference is that the starting and ending elements of short-columns need to be considered specially.

Lastly, the following post-processing steps are applied to the stitch mesh.

- Triangles are corresponding to the ending stitch of a short row. When their apex vertices are located at the boundary of a patch, they need to convert into quadrangles to reflect the real stitch that can be formed at the boundary.
- The graph-based tessellation method can ensure the continuity between courses at the interior region and the right boundary of stitch mesh. However, the discontinuity between courses can still happen at the left boundary of its resultant mesh, which needs to be fixed by adding new stitches.

Both cases have been illustrated in Fig. 3.5.

### 3.3.5. OPTIMIZATION

To generate a knitting stitch mesh with minimized number of transfer stitches on an input model $\mathcal{M}$, we compute an optimized harmonic field $f(\mathbf{x})$ by changing the Dirichlet boundary conditions on $\partial\mathcal{M}$ (defined in Eq. (3.7)). The objective function of this optimization is $\tilde{I}(f)$ (see Eq. (3.6)), which is a discrete form of the metric defined in Eq. (3.1).

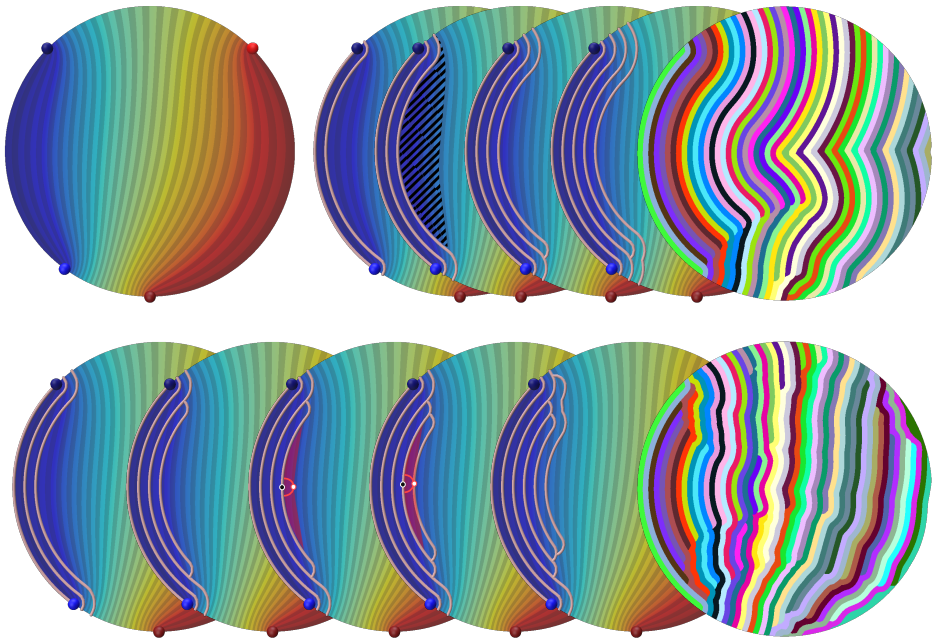


Figure 3.13: Wale curve extraction results generated by an algorithm akin to [134] (top row) and its improvement after adding Algorithm 3.3 to refine the isovalue of reference curve (bottom row). Progressive results are given here to show the major difference before vs. after this improvement – i.e., how to fill the gap between a wale curve and its reference isocurve, which is illustrated as the hatched region in the top row. Obviously, the result obtained in the bottom row is better aligned with the isocurves of the governing field (top-left). Note that the distance between the wale curves shown here is larger than a normal stitch width for giving a clearer visualization.



### VARIABLES FOR OPTIMIZATION

We first introduce the parameterization of variables for optimization. The domain $\mathcal{M}$ is a topological disk with one boundary loop. We parameterize the boundary loop into the $[0,1)$ interval anticlockwise. As discussed before, the Dirichlet boundary condition to be applied in this work are defined in two regions $R_s, R_e \subset \partial\mathcal{M}$ and $R_s \cap R_e = \emptyset$ as:

$$f(\mathbf{x}) = 0 \ (\forall \mathbf{x} \in R_s) \text{ and } f(\mathbf{x}) = 1 \ (\forall \mathbf{x} \in R_e).$$

The location of these two regions can be parameterized as a 4D vector $\mathbf{b} = (b_0, b_1, b_2, b_3)$ defined on the boundary of $\partial\mathcal{M}$ (see also the illustration in Fig. 3.14). $R_s$ and $R_e$ are represented by the intervals $[b_0, b_1]$ and $[b_2, b_3]$ respectively.

To ensure valid regions are defined, the variables $b_i$ should meet the following constraints

$$0 \le b_0 \le b_1 < b_2 \le b_3 < 1 + b_0 < 2 \tag{3.8}$$

Note that when $b_i > 1$, it represents the same point as $(b_i - 1)$ by winding. These constraints define the feasible space $\Omega$ for searching optimal fields.

In the discrete settings, we choose $b_i$s by using vertices on the boundary of the input mesh surface. The values of $\{b_i\}$s satisfying the constraints given in Eq. (3.8) may result

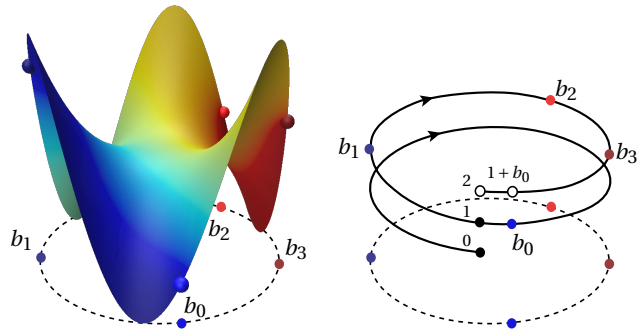



Figure 3.14: Corner points of the boundary conditions are used as variables for optimization, where the locations of corner points are parameterized on $\partial\mathcal{M}$.



in 2, 3, or 4 distinct points, since $b_0 = b_1$ and $b_2 = b_3$ are allowed. Degeneration happens when all distinct points fall in the same triangle. This leads to singularity in numerical computation, to resolve which we need to move one point out of this triangle.

### CONSTRAINED OPTIMIZATION

The objective function $\bar{I}(f_{\mathbf{b}})$ is difficult to be differentiated with respect to the variables $\mathbf{b}$, where $f_{\mathbf{b}}$ is the harmonic field determined by the boundary condition $\mathbf{b}$. A numerical scheme based on Bayesian optimization (BO), a derivative-free method, is employed by incorporating the constraints presented in Eq. (3.8). Generally, BO fits a surrogate model using samples of the objective, and an acquisition function is used to suggest the next location of samples. In this work, we use the trust region BO (i.e., TuRBO [47]), which can model local regions of the objective more accurately compared to the schemes that fit a global surrogate model on the whole search space. The surrogate models employed in TuRBO are Gaussian process models. The acquisition function returns the point that approximately minimizes the surrogate model in each step.

The feasible search space is constrained by Eq. (3.8). In the initialization step, we sample all points in the feasible space. During the iteration of optimization, the acquisition function may suggest evaluating the value of the objective function at a point $\mathbf{b}$ outside the feasible space $\Omega$. Then we evaluate the objective at a point $\mathbf{b}'$, which is

$$\mathbf{b}' = \underset{\beta \in \Omega}{\arg\min} \|\beta - \mathbf{b}\|^2. \tag{3.9}$$

The constraints presented in Eq. (3.8) are 6 independent linear constraints. Finding a feasible $\mathbf{b}'$ is a standard quadratic programming problem with linear constraints, solving which is much faster than determining the harmonic field (or evaluating the divergence) on the whole domain $\mathcal{M}$. If a point $\mathbf{b}$ is outside $\Omega$, we solve Eq. (3.9) to obtain $\mathbf{b}'$ and obtained a feasible sample point pair $(\mathbf{b}', \bar{I}(f_{\mathbf{b}'}))$. The pair $(\mathbf{b}, \bar{I}(f_{\mathbf{b}'}) + \exp(\|\mathbf{b}' - \mathbf{b}\|^2))$ will be employed to penalize the infeasible region. Both pairs are fed into the optimizer for the later iteration.

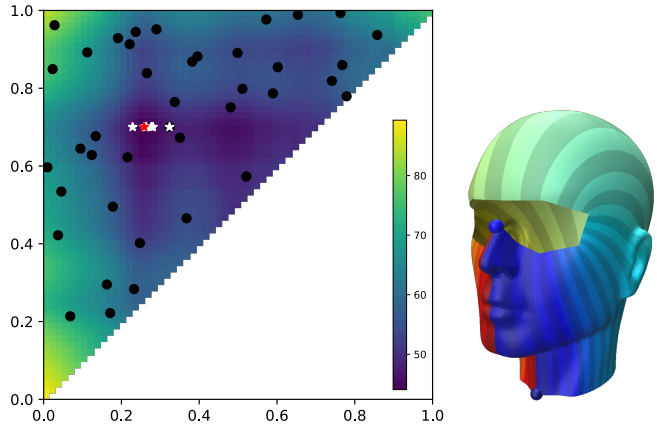



Figure 3.15: Optimization process by imposing the requirement of symmetry – i.e., only 2 degree-of-freedoms are allowed for boundary condition **b** now. Black dots are initial samples. Stars are found by the optimizer in the later steps. The red star indicates the finally determined point, which is corresponding to the optimized governing field shown on the right. Note that the model is cut from nose to neck for giving a disk-like topology.



We apply this numerical scheme of optimization to all examples presented in this section and can effectively obtain the optimized results (see Fig. 3.9 for the result obtained on a freeform surface). More examples can be found in Section 3.3.6.

#### EXTENSIONS

Many target shapes of knitting are axially symmetric. The knitting map aligned with the governing field is also preferred to be symmetric to produce symmetric knitwear. We can impose this requirement of symmetry on the boundary condition by reducing the variable **b** from 4D to 2D. The same BO-based scheme can still be applied to compute the optimized governing field, and the computation becomes faster as the optimization is conducted in a lower dimensional space now. The progress of optimization and the result of imposing this symmetry requirement on the skull cap model are shown in Fig. 3.15. Note that only samples in the upper-triangle of the 2D domain are feasible by ensuring the constraints presented in Eq. (3.8).

Because the harmonic fields are boundary aware, optimized Dirichlet regions are usually found between sharp corners on the boundary $\partial\mathcal{M}$. It is possible to search the optimized value of **b** among the vertices with sharp angles. Then the optimization space is reduced and the results can be obtained with shorter computing time. Nevertheless, we did not use this strategy in our tests in order to keep the generality of our method for cases without significant sharp angles.

### 3.3.6. RESULTS AND DISCUSSION

#### IMPLEMENTATION DETAILS AND HARDWARE

The computation method proposed in this section has been implemented by Python together with C++. Since we usually only need to compute geodesic distance around a wale curve in a small band region, a maximal propagation distance is imposed for the consid-

eration of efficiency. For Bayesian optimization, we used the source code provided by the authors of TuRBO [47]. The computation of all examples presented in this section can be completed within 1-10 minutes, which is much shorter than the physical knitting time which could be up to 1.5 hours. Scalar field visualization in this section is based on Polyscope [163].

We have tested our computational pipeline to design and fabricate a variety of examples with different 3D shapes. Two knitting machines are used to fabricate the examples given in this section to show the generality of our method. One One is Shima Seiki Mach 2XS153 [164] – a fully computerized knitting machine, which is an industrial-level machine having four needle beds with 15 needles per inch. Note that only two beds out of that four are used throughout our knitting process as models considered in this section all have the disk-like topology. The stitch size knitted by the Shima Seiki machine is 1.4 mm × 1.3 mm. The saddle model in Fig. 3.9, the skull cap in Fig. 3.19, and the sock in Fig. 3.20 are all knitted by the Shima Seiki machine. The knitting maps generated by our method can also be used to supervise the knitting operations on a Brother KH868 household machine [23] in a semi-automatic manner. The stitch size knitted by the Brother machine is 3.5 mm × 2.8 mm. The triple-peak shown in Fig. 3.21 is knitted by the Brother machine.



#### DIVERGENCE-BASED METRIC

A hemisphere example is employed to demonstrate the correlation between our metric defined by field divergence and the actual number of required transfer stitches. In this experiment, we sampled 100 different boundary conditions randomly, and run through the whole pipeline by each boundary condition to generate stitch meshes for counting the number of transfer stitches. The results are shown as the scatter points plotted in Fig. 3.16, which clearly show an approximately linear relationship. This result indicates that the divergence-based metric defined in our work is a good estimation of the required transfer stitches.

#### ROBUSTNESS TO RESOLUTION

In our approach, the harmonic field is computed by solving a Laplacian equation, which is less sensitive to the mesh resolution. Using cotangent discretization of the Laplace-Beltrami operator, the differential equation is discretized as a linear system. The discretization of the divergence operator is also less sensitive to the mesh resolution. We evaluate the robustness of the whole pipeline for computing the divergence-based metric by giving boundary conditions. As shown in Fig. 3.17, the skull cap model is meshed into 7 different levels of details. Ten random boundary conditions are given to evaluate the values of divergence-based metrics on these 7 resolutions, and the result for each boundary condition is plotted as a curve of the chart with the triangle number as the horizontal axis in the logarithmic scale. Ten curves are generated, from which we can conclude that the divergence-based metric is robust to the mesh resolution. As a consequence, we can perform the optimization on a low-resolution proxy of the mesh for better efficiency.

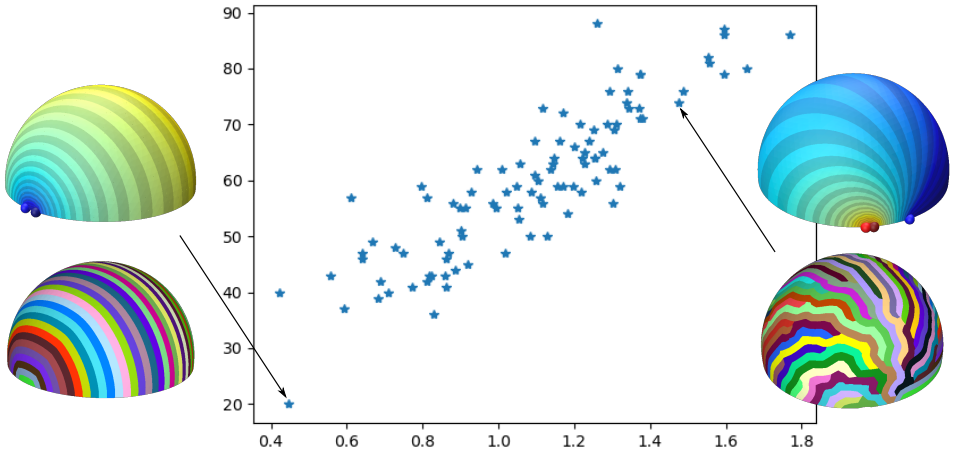



Figure 3.16: The chart on a hemisphere model shows the relationship between the divergence-based metric (the horizontal axis) and the number of required transfer stitches (the vertical axis).



### MULTIPLE LOCAL MINIMA

The objective function of our optimization is non-convex. The optimizer may find multiple local minima. We usually pick the best one among them. However, some of the others are also valuable as the candidates to inspire more design of different knitting plans (see Fig. 3.18 for an example).

### SEAM REMOVAL

When only using the short-row shaping to knit the skull cap model, the seam cuts still need to be added even after cutting along the seam from nose to neck to make it homeomorphic to a disk (see the left of Fig. 3.19). After using both short-row and short-column shaping techniques, the skull cap can be knitted without additional seams as the right of Fig. 3.19. Similar conclusions can be made from examples such as the sock (Fig. 3.20) and the triple-peak models (Fig. 3.21). Wale curves in closed loops can be effectively avoided when stitch transfer is applied.

### DISTORTION REDUCTION

Although stitch meshes generated by geometric computing can give a good approximation of the target shape, the knitted fabrics are still distorted [221]. This is caused by the fact that the stitches as elements in a stitch mesh are often not identical to their ideal shapes, i.e., rectangles or isosceles triangles. In this study, we compute the rest shape of a designed stitch mesh by using the geometry processing method presented in [20]. We evaluate the rotation-invariant distortion of each polygonal element on the stitch mesh. Colormap of the per-element distortion is shown in Fig. 3.21. As wale curves with sharp angles can be avoided after using the short-column shaping, the distortion is effectively reduced.

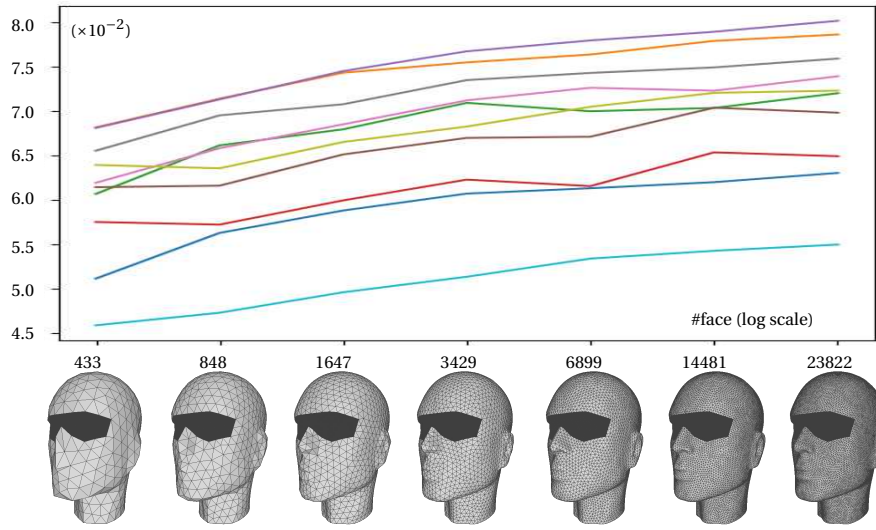



Figure 3.17: Study of the metric's robustness to the mesh resolution. The horizontal axis is a sequence of meshes in different resolutions. The vertical axis gives the value of metric defined in Eq. (3.1) normalized by the total area. The robustness is tested on 10 randomly generated boundary conditions, each of which is plotted as a curve.

### DISCUSSION AND FUTURE WORK

Thanks to the generality of our optimization framework, it is possible to extend this work in several directions. For example, the $L_1$ norm is currently employed in our objective function. It will be interesting to study how the effectiveness of optimization can be changed by using other general $L_p$ norms.

It is challenging to find the exact constraint for the maximally allowed number of transfer stitches. First of all, it is influenced by many factors, including the machine configurations, the elasticity of yarn, the structures/patterns to be knitted, the tension of the machine (the carriage and the pull-down system), and the original width of the fabric, etc. Secondly, conducting a trial experiment reaching the limit of a knitting machine may lead to mechanical damage (e.g., breaking the needles). One possible future research is to study the manufacturing constraints caused by stitch transfer systematically.

## 3.3.7. CONCLUSION

To generate reliable 3D digital knitting maps by using both short-row and short-column shaping, we present a field optimization method to minimize the number of transfer stitches that are considered the major sources of manufacturing problems such as miss-transfer, yarn stretching, yarn abrasion, etc. Locations of Dirichlet boundary conditions are formulated as the variables for optimization. It is impractical to directly include the module of stitch mesh generation in the loop of optimization. Alternatively, we define a new metric based on the divergence of cogradient field, which can indirectly measure the number of transfer stitches. Moreover, we also develop a modified stitch mesh gen-

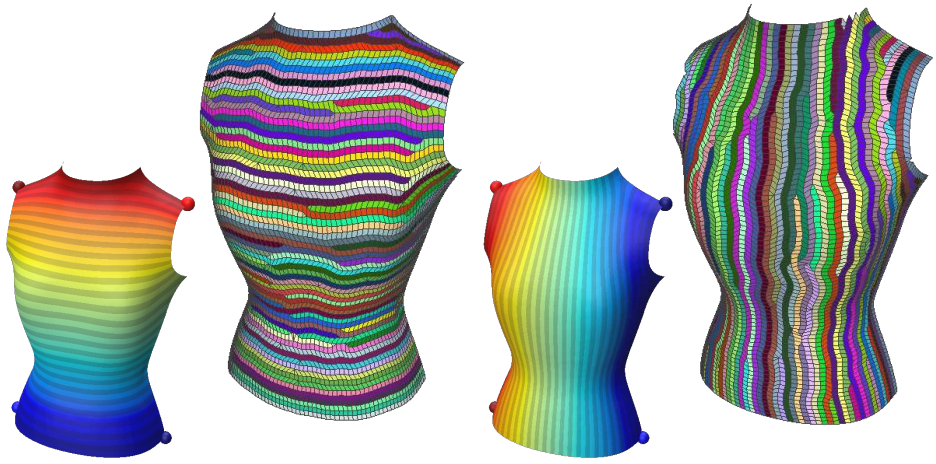


Figure 3.18: The optimizer TuRBO is able to find multiple local minima of the non-convex objective. Besides the one gives minimal value for the objective function (the left one), some other local minima also provide meaningful design – see the right one, which is a plan commonly employed in industry.



eration algorithm that can generate wale curves more tightly aligned with the isocurves of a governing field.

The results obtained in experimental tests are very encouraging. Our computation framework requires very little manual operations and therefore can significantly improve the level of automation in digital knitting. By being able to automatically search a 'best' governing field, this method can effectively reduce the distortion on the knitted 3D fabrics.

### 3.3.8. EXTENSIONS OF THIS WORK

I have extended the reliable 3D knitting design method from sheet knitting (e.g., disk-like topology) to tube knitting, concurrent with the writing of this dissertation. The main extensions are summarized below.

For tube knitting, courses of the stitch mesh are aligned with the isocurve directions of the governing field, being the same as AutoKnit [134] but different from the methods in Sections 3.2 & 3.3. Therefore, the divergence measurement is applied on the normalized gradient vector field $\hat{\mathbf{g}}$, instead of the cogradient field $\mathbf{J}\hat{\mathbf{g}}$. For the normalized gradient field, the measurement is still an effective estimation of the transfer stitch number, verified by both theory and numerical experiments. A special case is a divergence-free field corresponding to transfer-free knitting. If the normalized gradient of a governing field $f$ totally equals the cogradient of a smooth geodesic distance field, then the induced knitting map should be transfer-free (for the same reason as Eq. (3.4)). The same conclusion can be drawn from the relation between the geodesic curvature [42] of isocurves and the divergence of normalized gradient (see [41]). If the target shape is a surface of revolution, then there is a transfer-free governing field, in which the isocurves are identical

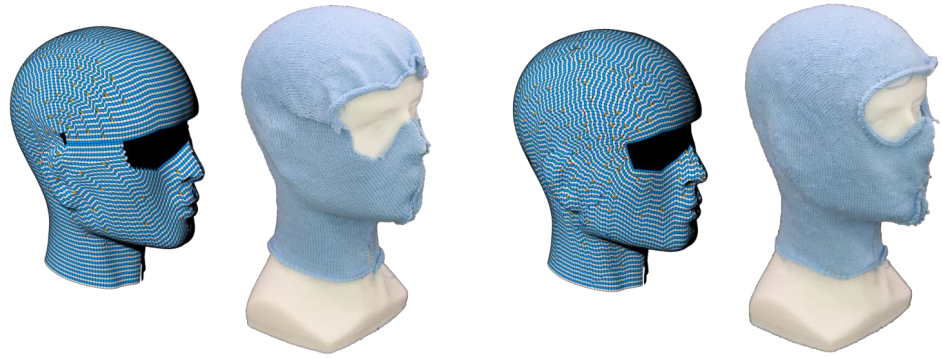


Figure 3.19: The skull cap model knitted by using (left) only short-row shaping and (right) both short-row and short-column shaping techniques. Both the stitch meshes and the knitted results are presented here. It can be observed that the additional seams at the corners of the eyes are not necessary anymore when the short-column shaping is allowed to use.



Figure 3.20: The sock example can be seamlessly fabricated by digital knitting using the stitch mesh (left) generated by our field-based optimization approach. The knitting result (right) perfectly fits the input shape of the feet which is 3D printed. The stitch mesh circled by dash-lines is a result generated by only using short-row shaping – therefore an additional seam is needed.

to the meridians (rotations of the generating curve). Surfaces isometric to a surface of revolution also have transfer-free governing fields.

By analyzing the field divergence, some intuitive choices of the initial governing field can be found. One may find the Fiedler vector of the mesh's Laplacian to approximate the shape's 'diameter direction'. Another option is to find the field orthogonal to the 'diameter direction', imitating the geodesic distance field on a surface of revolution. The initial fields are optimized by modifying their boundary conditions. The initial boundary conditions are usually defined on distinct vertices. They are extended as several curve segments to reduce the divergence-based metric. Users may design the governing field manually, during which the divergence-based metric can indicate unreliable regions with high divergence values.

Once a scalar governing field is optimized, a stitch mesh aligned with the field is generated. Firstly, the whole shape is decomposed into several simple components along field isocurves according to the singularities of the field. Each component is either homeomorphic to a disk or a tube. Then isocurve segments of the scalar field are extracted uniformly according to the gradient norm of the field (not according to the field value). In regions with smaller gradient norms, the field value varies slowly and the isocurves

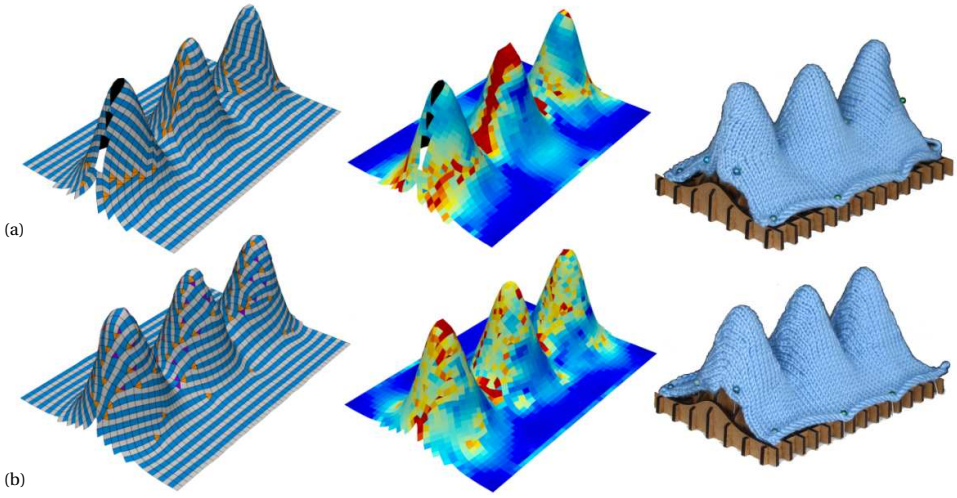



Figure 3.21: The result of triple-peak example by using (a) only short-row shaping and (b) both short-row and short-column shaping techniques. Colormaps are presented to illustrate the per-element distortion. The knitting results are also given.

are extracted more densely. After that, regions between neighboring isocurve segments are tessellated with quad-dominant meshes. Columns, including short-columns, are formed by tracing elements on these meshes. At last, the column curves are resampled uniformly. Regions between neighboring column curves are tessellated considering the knittability constraints. Shape components homeomorphic to disks are tessellated with the same graph-based method as this section. Those components homeomorphic to tubes are tessellated with a greedy graph search method. Experiments show that both cases are processed successfully.

This extension refined the work of this section. It considered the problem of governing field design as embedding the shape into a compact set in $\mathbb{R}^1$. The embedding's objective is defined as a divergence-based metric, which plays a similar role as the metrics defined for 2D embeddings, i.e., planar parameterization [172]. Lots of work have been conducted on 2D embeddings while 1D embedding still needs further research.

## 3.4. CONCLUSION

In this chapter, we presented two stitch mesh generation methods: one uses only short-row shaping and the other uses both short-row and short-column shaping. By using the short-column technique, the distortion is reduced and some sewing seams are avoided, though it is slower when short-columns are involved.

Compared to the flattening-based method in the previous chapter, these two direct methods are more automatic and thus easier to use. Users may pick boundary conditions to generate the governing field or just rely on the automatically optimized boundary conditions. The graph-based stitch mesh generation method guaranteed knittability. The complicated continuity adjustment steps, like that of the flattening-based method,

are largely reduced.

Compared to the other methods in the literature, our method is the first to control distortion using a diffusion term to make shaping stitches scattered instead of gathered. Our short-column knitting method is also the first to consider knitting stability by optimizing the knitting direction governed by a field. Transfer stitches, the main causes of knitting instability, are reduced after optimization. The proposed method can also be generalized from sheet knitting to tubular knitting, as discussed in Section 3.3.8.

## 3.A. ALGORITHM DETAILS

In Section 3.2, geodesic distance fields are computed on triangular meshes with polyline sources. In Sections 3.2 and 3.3, isocurves are extracted on scalar fields defined on triangular meshes. Some details are presented here. The source code of both algorithms is available in MeshUtility [106].



### 3.A.1. POLYLINE-SOURCED GEODESIC DISTANCE

A polyline-sourced geodesic distance field computation method is proposed in [16] by extending the vertex-sourced MMP (Mitchell, Mount, and Papadimitriou) method [126] of the exact shortest path algorithm on triangular meshes. We use a subdivision-based method that is easy to realize by using Danil Kirsanov's implementation of [126].

Given a triangular mesh with an edge collection $\mathcal{C}$ on it, we approximate the geodesic distance field sourced from $\mathcal{C}$. We subdivide the edges in $\mathcal{C}$ to simulate the geodesic distance from the interior points of the edges. It is clear to see the approximation error is bounded by the source edge length after subdivision, using the triangle inequality. Each edge can be subdivided into more segments to reduce the error. Another solution is to compute multiple distance fields while moving the subdivision points along the edges. For each vertex on the original mesh, the minimum value on all fields is the desired result. If two edges of a triangle belong to $\mathcal{C}$, we first split the face by adding a vertex before splitting the edges, like the bottom-right triangle in Fig. 3.22.

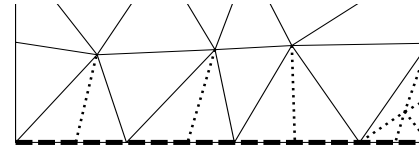


Figure 3.22: To approximate the geodesic distance on a triangular mesh (solid lines) from a polyline source (dashed), we subdivide the mesh by adding splitting edges (dotted).

Fast-marching is an approximate method supporting polyline-sourced computation by nature. A recent survey paper Crane et al. [35] reviews different types of geodesic distance computation methods comprehensively.

### 3.A.2. ISOCURVE EXTRACTION

Given a scalar field defined on a surface, an $s$-isocurve is the curve composed of all points with field value $s$ on the surface. It is also referred to as isoline, contour, or level set. Without the loss of generality, we extract the 0-isocurve of a piece-wise linear scalar field $f$ defined on triangular mesh $\mathcal{M}$. The extracted isocurve should be a 1D manifold whenever

it is possible, to benefit the downstream applications, such as splitting the mesh along the extracted isocurve (Section 2.B). Our algorithm consists of the following rounds for robustness:

1. Vertex round: checking each vertex $\mathbf{v}$ if $f(\mathbf{v}) = 0$. We use a user-defined tolerance for this checking. All vertices on 0-isocurve are indexed.
2. Edge round: checking each edge if it intersects 0-isocurve in its interior. All intersected edges are indexed. Cases covered in the last round are skipped.
3. Face round: extracting the 0-isocurve on each face, by connecting the points indexed in the last two rounds.

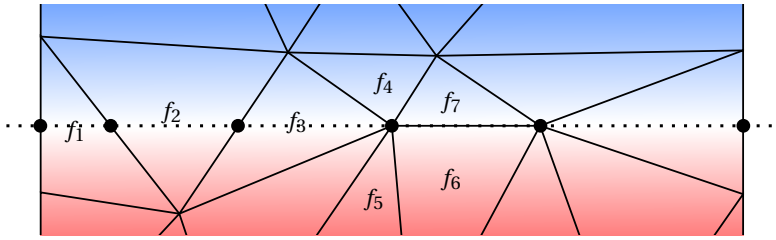



Figure 3.23: Possible situations of triangles intersecting an isocurve (the dashed line). Among three vertices of a triangle, there may be zero ($f_{1,2}$), one ($f_{3,4,5}$), or two ($f_{6,7}$) vertices located on the isocurve.

In the last round, the situation of each triangle $\mathbf{v}_i$ $(i = 0,1,2)$ can be denoted as a triplet $T = (t_0, t_1, t_2)$ , in which $t_i \in \{+,-,0\}$, representing $f(\mathbf{v}_i) >, <$, or $= 0$ respectively. All possible situations of a triangle are classified according to the number of its vertices on 0-isocurve. Examples of each type are shown in Fig. 3.23. Now we treat $T$ as order-independent for simplicity.

- No vertex on isocurve.
  - $T = (-,-,+)$ or $(-,+,+)$: extract a segment on this face ($f_1$ and $f_2$ in Fig. 3.23).
  - $T = (-,-,-)$ or $(+,+,+)$: empty.
- One vertex on isocurve.
  - $T = (0,-,+)$ extract a segment ($f_3$ in Fig. 3.23).
  - $T = (0,-,-)$ or $(0,+,+)$, empty ($f_4$ and $f_5$ in Fig. 3.23).
- Two vertices on isocurve.
  - $T = (0,0,-)$ or $(0,0,+)$, extract the edge connecting two on-curve vertices ($f_6$ and $f_7$ in Fig. 3.23).
- Three vertices on isocurve: $T = (0,0,0)$. All three edges of the triangle are isocurve segments. The solution of this special case should be adjusted based on the end goal of the isocurve.

All segments are connected in the last. Since we have indexed all isocurve vertices on mesh vertices or mesh edges, duplicate vertices will not be generated. The connected polyline may not be manifold if it passes a saddle point of the scalar field.

# 4

# 4D KNITWEAR DESIGN FOR BODY MOTION

## WHERE TO PUT ELASTIC STRUCTURES?

### 4.1. INTRODUCTION

Designing clothing that considers the movement of the body is an age-old problem in the garment industry. Computer graphics research has advanced many aspects of garment design, enabling perfect-fit 3D garments for varied body shapes (e.g., [24, 127]). Physics-based optimizations have also been employed to support both aesthetic design and comfort. Recently, the apparel industry has introduced innovations to vary elasticity in knitwear allowing customization for body motion, which is referred to as *4D garment design*. However, existing methods mainly rely on experience and intuition. Our work presented in this chapter enables the computational design and fabrication of 4D garments as knitwear. It tackles the research question (**RQ2**): how to design a motion-aware 4D knitwear and generate its machine instructions. In industry, the quality of a garment can be evaluated as the perfect-fit between the cloth and body. According to interviews with garment experts, relative sliding is the most direct measurement when garments under motion are considered. As explained in Fig. 4.1, the stress in the garment and the sliding of the garment over the body need to be controlled to achieve desired comfort and perfect-fit. On the one hand, large stresses in the garment caused by firm materials can lead to uncomfortable stretch and compression. On the other hand, when using soft materials, sliding of the garment over the body can lead to unwanted wrinkles. We introduce a computational pipeline to control stress and sliding on clothes by distributed

This chapter has been published in: **Z. Liu**, X. Han, Y. Zhang, X. Chen, Y.-K. Lai, E.L. Doubrovski, E. Whiting, and C.C.L. Wang. *Knitting 4D garments with elasticity controlled for body motion*, ACM Trans. Graph., 40(4), 2021. doi: 10.1145/3450626.3459868. A few small corrections and/or clarifications have been made to the original published text.

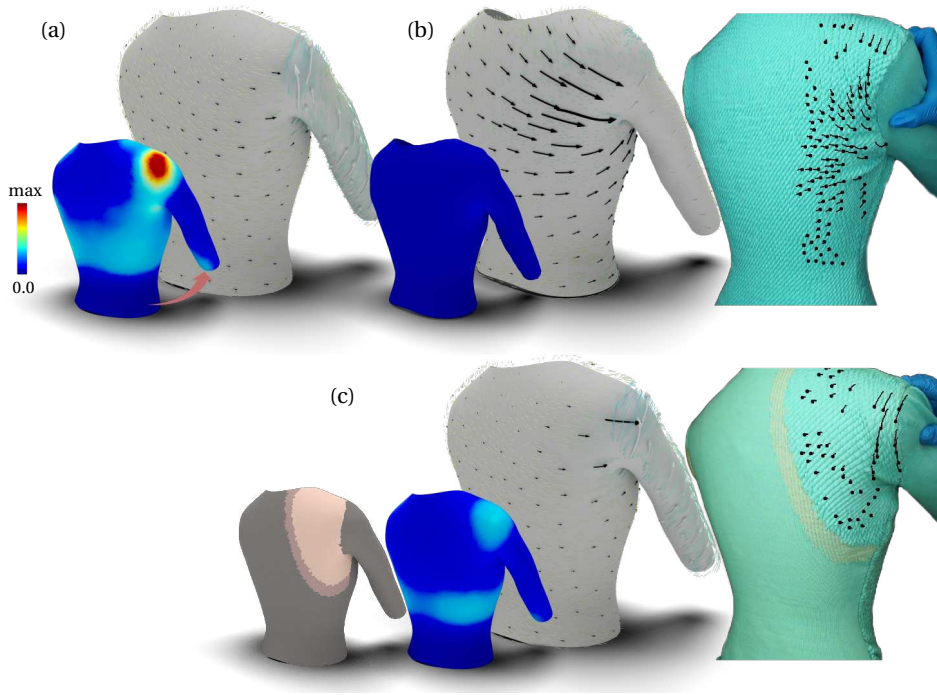



Figure 4.1: During the body motion of swinging arms, a perfect-fit 3D garment can have: (a) large stress when using firm materials – leading to uncomfortable pressure or (b) large sliding when using soft materials – resulting in unwanted wrinkles. Both factors are considered in an integrated way on a knitwear with optimized distribution of elasticity as a 4D garment (c) that minimizes the stress and controls the maximal sliding during body motion. Stresses and displacements are visualized as color maps and black arrows respectively, where the maximal stress is 40.75kPa. Our work enables a computational framework for designing 4D garments and automatically fabricating them on digital knitting machines. A knitwear as 4D garment is physically fabricated by knitting different 'percentages' of firm and soft yarns in different regions. We make the regions of different elasticity visible by using firm yarns in light-blue and soft yarns in white. Sliding trajectories on physical specimens are evaluated by a vision-based method and displayed as black curves.

elasticity for improving the comfort in motion, with a focus on the challenging case of tight fitting garments.

Compared to conventional 3D garment production techniques (sewing), digitally knitted clothes show the following advantages:

- The capability to generate garments with numerous 'darts', which are automatically 'sewed' together on knitted clothes and therefore fabricate garments with complex 3D freeform surfaces (Chapter 3).

- The capability to use mixed yarns in different regions that can locally control the level of elasticity to absorb deformations on human bodies in motion (see Fig. 4.1).

In our approach, we utilize these two capabilities to produce 4D garments.

### 4.1.1. OUR METHOD

To enable the design and fabrication of knitwear with controlled elasticity distribution, we first precisely fabricate the designed 3D shape by digital knitting, and then realize the elasticity variation in different regions by *single-jersey jacquard* (or jacquard for short. It is also referred as fair isle) with two yarns. These two components are strongly coupled. Specifically, without a fabrication method to perfectly realize the designed 3D shape, the garment will deform when worn on the body. In our work, we aim for the default pose to have uniform stresses (achieved by 3D shaping); then the elasticity assignment is only concerned with stresses and deformations that arise from motion. The stitch meshes generated by our algorithm use only the short-row knitting strategy to form 3D geometry, which can be executed on a knitting machine more efficiently. A tiling algorithm is employed to assign jacquard patterns on the stitch mesh to realize the designed distribution of elasticity.

In summary, we present the following technical contributions in this chapter:

- A method to generate machine knitting code for 3D garments with locally varying levels of elasticity, using different jacquard patterns with two yarns (one soft and one firm).
- An iterative algorithm to assign different levels of elasticity in different regions of a garment so that the deformation under body motion can be optimized. We consider reduced stress and controlled sliding to achieve 4D garment design.



To our knowledge, this is the first approach to simultaneously optimizing the comfort (reducing stress) and the perfect-fit (avoiding wrinkles caused by large sliding) under motion by tuning the elasticity of knitwear that can be automatically fabricated by digital knitting machines.

### 4.1.2. RELATED WORK

#### CLOTH SIMULATION AND GARMENT DESIGN

After the pioneering work of simulating 3D clothes as deformable objects [200], many computational methods have been developed in the computer graphics community to simulate the physical behavior of clothes on human bodies in motion (e.g., [8, 52]). The method of [11] can parse patterns made by professional designers and automatically generate virtual fitting results. With the help of these simulation and forward design techniques, inverse design of garments can be generated by optimizing the planar panels to achieve the desired effect of clothes fitting [100, 9, 190]. In another aspect, the so-called design transfer function [24, 124, 205] can be realized by generating the same fitting results when applied to human bodies in different 3D shapes and poses [4]. Limited by the means of fabrication, these methods often compute planar panels as the outcome. When sewn back into 3D garments, the elasticity is not tailor-made. In contrast, our approach enables programmable elasticity on 3D knitwear which can be digitally fabricated.

In order to simulate the physical behavior of knitted fabrics more precisely, computational models have been developed to simulate knitted cloth at the yarn level [137, 77]. The concept of the stitch mesh is introduced by [221] to provide a canvas-like abstraction

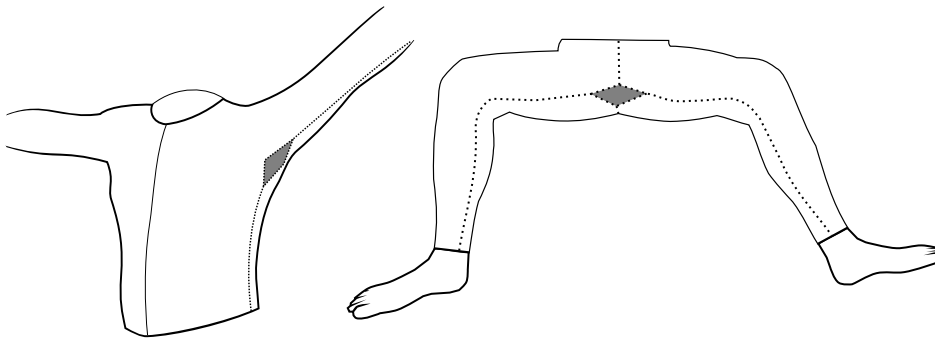


Figure 4.2: Sleeve gusset and crotch gusset (gray regions) are inserted into a seam to add breadth or reduce stress from tight-fitting clothing.



of the yarn model although it is not guaranteed to be knittable. With the help of these two techniques, [95] developed an interactive tool for designing yarn-level patterns for both knitted and woven cloth. Although these computational tools at the yarn-level are available in the literature, we, however, simulate garments with distributed elasticity at the level of triangular meshes for the sake of computational efficiency, where the orthotropic material properties are calibrated experimentally (ref. [207]) and assigned to each triangular element. The cloth simulator, ARCSim [132], embedded with the function of adaptive anisotropic remeshing [131, 130] can capture physical behavior in more detail while using a relatively coarse mesh as input. In our algorithm of inverse design, we modify the simulator to enable the assignment of different elasticity on different triangular elements.

Skintight garments applied to human bodies with tight-fitting clothing have been widely used in casual fashion, sportswear and medical treatment applications [127, 91, 204]. The work of [91] has considered aesthetic factors to develop an evolution process for inverse design of panel layout on 3D human bodies, which can be fabricated from planar panels generated by flattening algorithms (e.g., [206]). Physics-based computation is introduced in [204] to generate optimized 2D panels according to prescribed pressure distribution on a 3D body shape. Recently, [127] proposed a physics-based method to compute 2D panels of skintight garments by incorporating body deformation in multiple poses, pressure distribution, and seam traction into design objectives. Differently, our method does not have restrictions of 2D panels, enabling more accurate capture of the 3D shape. Moreover, we optimize the distribution of fabric elasticity based on physical simulation of garments during body motion, and propose a computational pipeline to enable automatic fabrication on digital knitting machines.

In the garment industry, designers insert a triangular or rhomboidal piece of fabric into a seam to add breadth or reduce stress from tight-fitting clothing. The inserted piece is named as *gusset*. A sleeve gusset and a crotch gusset are shown in Fig. 4.2. The gusset is folded and unfolded during body motion, which provides more flexibility. This is a simple realization of 4D garments. If the inserted gusset is never folded under motion, it only contributes to the 3D shaping, like the design in Fig. 3.1. Then it is not a 4D garment

for motion. Gussets are the places we add materials, which is dual to darts, where we remove materials.

#### COMPUTATIONAL FABRICATION OF KNITWEAR

Related work of computational design of knitting have been discussed in the previous chapters. Among these related works, the interface developed in [82] allows users to specify both the knitting primitives and the stitch patterns. Different stitch patterns are designed in a way like image editing. However, the difference in elasticity observed on different knitting patterns has not been adopted for functional usage. No prior research has been performed to enable the programmable tailor-control of elasticity on a knitwear. Note that 4D garments mentioned in this work are different from the soft actuated garments [1] and self-folding textiles [87], where the deformation is generated on clothes by knitted actuators and residual strains, respectively.

#### INVERSE DESIGN OF DEFORMABLE OBJECTS

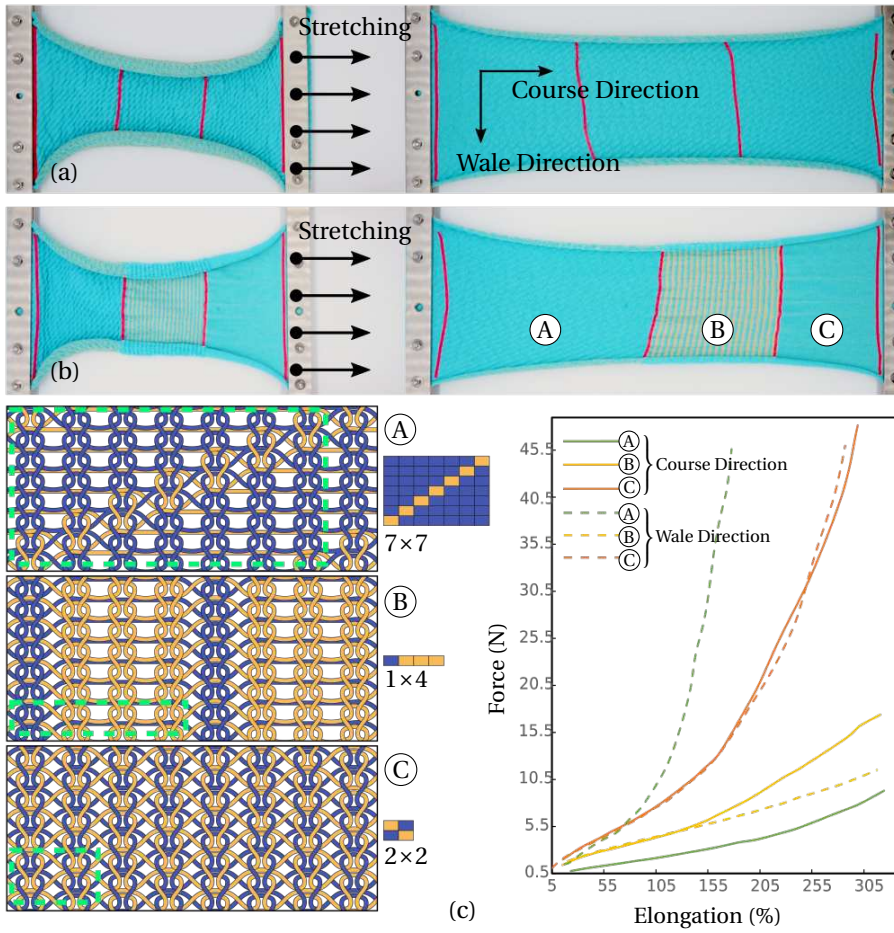
Designing distributed elasticity on garments is related to a wider range of inverse design applications that consider structural strength and deformation behavior of deformable objects. An inverse design approach was proposed in [116] for garments made from non-flat rest shape. Combinations of different materials and therefore tailor-made elasticity were adopted as design variables in [12, 169, 217] to achieve a desired deformation behavior. Besides multiple materials, structural strength [142, 29], cellular microstructures [160, 138] and composite silicone [222] have also been inversely designed for desired deformation in prior research. Because of complex physical phenomena that are difficult to simulate accurately, the properties of physical models are often captured by a data-driven methodology in these approaches (e.g., [12, 160, 138, 226, 222]). The same strategy is employed in our approach to capture the elasticity of different jacquard patterns by knitting two yarns – one soft and the other firm.

Elasticity on woven fabrics has been employed in applications of computational fabrication. Small structures [56] and layouts of inextensible polymer [143] are 3D-printed on stretched fabrics to form desired shapes after releasing stretch. Planar panels of fabric containers inflated by pressurized air [170] and pressures of injected viscous fluid are inversely designed using orthotropic material properties of fabrics. The same material model is employed in our physics-based inverse design, where the material parameters are measured from physically fabricated jacquard patterns.

## 4.2. ELASTICITY BY SINGLE-JERSEY JACQUARD

We adopt the jacquard technique to generate distributed elasticity for the same reason that we use short-row 3D shaping: efficiency and feasibility on low-cost machines (i.e., even for those with only one needle-bed). Jacquard is a technique to knit two or more yarns together when moving the carrier horizontally. Specifically, two yarns with different elasticity levels (soft and firm) are employed in our approach. The soft and firm yarns are led at different heights by the carrier, and the needle hooks are shifted to their corresponding height to determine which yarn is used to form the loop.

When altering the arrangement of the two yarns, different elasticities can be formed by changing the periodic pattern. See Fig. 4.3 for three example patterns that result in




| Stitch Size (Unit: mm) | Ⓐ | Ⓑ | Ⓒ |
|---|---|---|---|
| Width × Height | 0.125 × 0.0780 | 0.0984 × 0.0680 | 0.0812 × 0.105 |

Figure 4.3: Different from stretching a fabric knitted by using a single jacquard pattern (a), different elongations can be observed when using three different jacquard patterns (b). Yellow denotes soft yarns and blue is used for firm yarns. Three different jacquard patterns represented by yarn-level models and their corresponding strain-force curves (in horizontal and vertical tensile tests) are given in (c). When designing different jacquard patterns, floats are controlled within the lengths allowed by reliable machine knitting. The average stitch sizes of different jacquard patterns are also given.

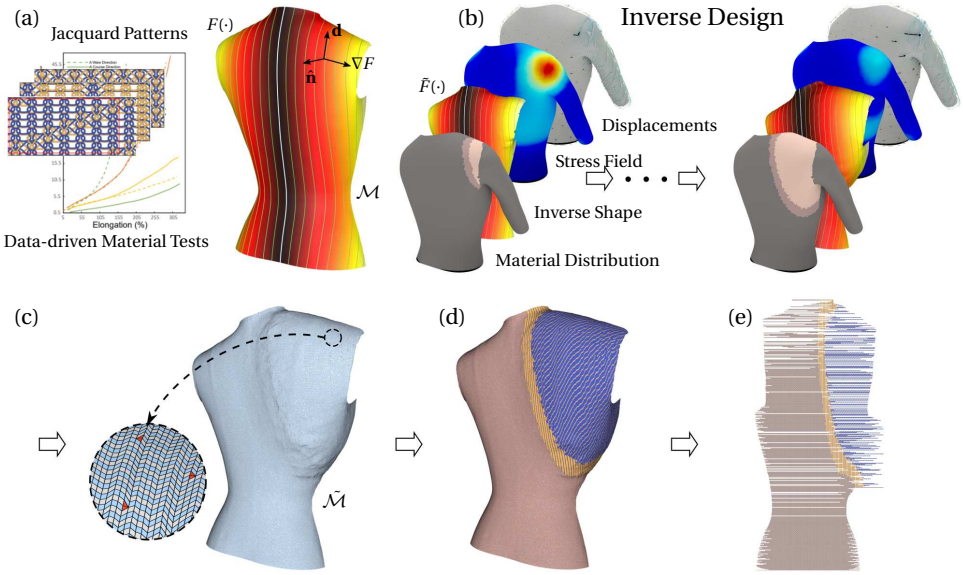



Figure 4.4: Our computational design and fabrication pipeline to produce 4D knitwear with elasticity controlled for body motion: (a) data-driven material tests of jacquard patterns and the geodesic distance-field $F(\cdot)$ on a garment $\mathcal{M}$ for assigning orientations for knitting stitches and orthotropic material simulation, (b) progressive updating of the soft / firm material distribution, (c) knittable stitch mesh generated on the inverse geometry, (d) jacquard patterns assigned on the stitch mesh and (e) the resultant knitting map. Note that, after computing the inverse geometry $\tilde{\mathcal{M}}$ of the garment to compensate for the variation of shrinkage ratios, the orientation of each triangle in orthotropic material simulation is re-evaluated on an updated geodesic distance-field $\tilde{F}(\cdot)$. Collapsed cells are not marked as gray on the knitting map. So are the other knitting maps in this chapter.

different strain-force curves. In single-jersey jacquard fabrics, the same knitting map can still be used to govern the machine operation. A remaining problem is how to assign the jacquard information to a knitting map. We present a tiling algorithm in Section 4.3.4 for this purpose.

When starting to knit a new course, loops in the previous course are taken off the hooks of the needles. Although the distances between neighboring needles are constant, loops formed by different yarns will shrink into different sizes. The elasticity control on a knitted garment needs to take this variation of shrinkage into consideration. We compute the inverse geometry of a 3D surface for incorporating this factor (see Section 4.3.3).

The three jacquard patterns in Fig. 4.3 are selected out of 13 patterns. More patterns with controlled float lengths can be designed to realize more elasticities. However, their strain-force curves may be intersecting. More importantly, if more patterns are used in the knitwear design, the region of each pattern on the garment will be smaller. Since we measure the homogenized physical properties of the jacquard structures, the structure areas should be significantly larger than a repeat unit of the patterns. Therefore, only three patterns with distinct elasticities are used here.

## 4.3. DESIGN AND FABRICATION OF 4D KNITWEAR

Given a 3D garment $\mathcal{M}$, the shape of which is designed around the human body $\mathcal{H}$ in a static pose, we evaluate the performance of $\mathcal{M}$ under a prescribed body motion of $\mathcal{H}$ for comfort on a physical simulator. Specifically, for comfort evaluation the maximal stress $\sigma_{\max}$ is measured on $\mathcal{M}$ among all elements at all time steps during the motion. The maximal sliding $d_{\max}$ between $\mathcal{M}$ and $\mathcal{H}$ in a user-specified region $\Omega$ is also detected during the whole motion sequence. The computational design and fabrication problem to be solved here is 1) to find a distribution $\mathcal{P}$ of elasticity that minimizes $\sigma_{\max}$ while controlling the allowed maximal sliding distance as $d_{\max} \leq \bar{d}$, and 2) to determine a knitting map $\mathcal{K}$ that can realize $\mathcal{P}$ by jacquard patterns with two yarns. $\bar{d}$ is a user-defined threshold.

We tackle the problem by first computing the distribution of elasticity $\mathcal{P}^e$ at the element level according to predefined jacquard patterns and then generating the knitting map $\mathcal{K}$ that can realize $\mathcal{P}^e$ by these jacquard patterns. The pipeline is illustrated in Fig. 4.4.



### 4.3.1. INVERSE DESIGN OF DISTRIBUTED ELASTICITY

The physical properties of elasticity that can be realized by jacquard patterns are determined by a data-driven method similar to [207]. Our flat sheets knitted from uniform jacquard patterns are simulated effectively by an orthotropic fabric model with physical properties measured as in [207]. The first step of our computational pipeline determines the orientation of knitting structures for each piece of a garment $\mathcal{M}$. A base line that indicates the knitting direction is defined by the user. The base line can be selected from the feature curves of the garment (e.g., the central line of a front / back piece) or specified according to a desired direction of knit stitches. A geodesic distance-field $F(\cdot)$ of the base line is computed to determine the orientation of stitches by assigning 1) the weft direction as the gradient of $F(\cdot)$, $\nabla F$, and 2) the wale direction as $\hat{\mathbf{n}} \times \nabla F$ with $\hat{\mathbf{n}}$ being the surface normal. With these assigned weft / warp directions (see the illustration in Fig. 4.4(b)), the material orientation in the orthotropic model can be determined and used in the physical simulation below.

The elasticity distribution is determined on $\mathcal{M}$ at the triangular element level by a greedy algorithm.

1. First, two simulations are conducted with the soft and the firm materials to obtain their corresponding maximal stress and sliding as $(\sigma^s_{\max}, d^s_{\max})$ and $(\sigma^f_{\max}, d^f_{\max})$ respectively.
2. We then initialize with a uniform distribution of firm material by assigning the material level as 0. Here we adopt different material levels as: 0-firm, 1-medium, 2-soft.
3. Regions with large stresses are progressively assigned into soft and medium by using the following steps (see Fig. 4.4(b)).
    - Regions of the designed domain with stress value larger than 90% of the current maximal stress are assigned as soft using material level 2.

- The filtering step is applied to perform 5 iterations of Laplacian smoothing on the material levels as floating-point values on all triangular elements.
- Quantization is conducted by rounding the material level from floating-point value back to integer as {0, 1, 2}, to generate a new material distribution in discrete levels.

4. After updating the material distribution, the simulation is run again to obtain the updated $\sigma_{\max}$ and $d_{\max}$.
5. Go back to (3) or stop the iteration when $d_{\max} > \bar{d}$.

### 4.3.2. ENABLING MACHINE KNITTING

Given a distribution of elasticity $\mathcal{P}^e$ assigned on $\mathcal{M}$, we compute the knitting map $\mathcal{K}$ in four steps to realize this 4D garment.

1. **Inverse geometry computation.** To compensate for the variation of shrinkage ratios among different jacquard patterns, an inverse geometry of $\mathcal{M}$ is computed as $\tilde{\mathcal{M}}$ (see Fig. 4.4(c)). Computing stitch meshes on the inverse shape $\tilde{\mathcal{M}}$ will then incorporate the pattern-specific shrinkage ratio on the rest shape $\mathcal{M}$ (Section 4.3.3). As the geodesic distance-field on $\tilde{\mathcal{M}}$ is changed to $\tilde{F}(\cdot)$, this step is also included in the routine of inverse design to re-assign the material orientation of each triangle.

2. **Knittable stitch meshes** are computed on the inverse geometry $\tilde{\mathcal{M}}$ by using the geodesic distance-field $\tilde{F}(\cdot)$ (see Fig. 4.4(c)). Quadrangles / triangles of a stitch mesh (short-row-only) are constructed between $\tilde{F}$'s isocurves by optimizing the shape of facets and repulsing the location of triangles, which is formulated as finding a shortest path on a graph. The algorithm is presented in Section 3.2.

3. **Jacquard patterns** are assigned to the knittable stitch meshes to realize the distributed elasticity (see Fig. 4.4(d)). A flooding algorithm is employed to determine the local row / column indices of tiling. An algorithm based on *minimal spanning tree* (MST) is developed to minimize the discontinuity in Section 4.3.4.

4. **The knitting map** that can be executed on a machine is generated from a stitch mesh (see Fig. 4.4(e)). The location of each stitch (quadrangle or triangle) on a knitting map can be determined by assigning it with the sorted indices of course and wale (see Chapters 1 & 3).

The knittable stitch mesh and knitting map generation method have been presented in previous chapters. Next, we present how jacquard patterns can be realized on the stitch meshes. A method to compensate for the variation of stitch sizes is first introduced. After that, we present a tiling algorithm that minimizes the discontinuity while assigning jacquard patterns.

### 4.3.3. STITCH-SIZE COMPENSATION

As analyzed in Section 4.2, stitch size is not uniform on jacquard fabric. The variation of stitch sizes is not negligible if two jacquard patterns with different elasticity are knitted on the same fabric (as illustrated in Fig. 4.5). Similar to the strategy of homogenization

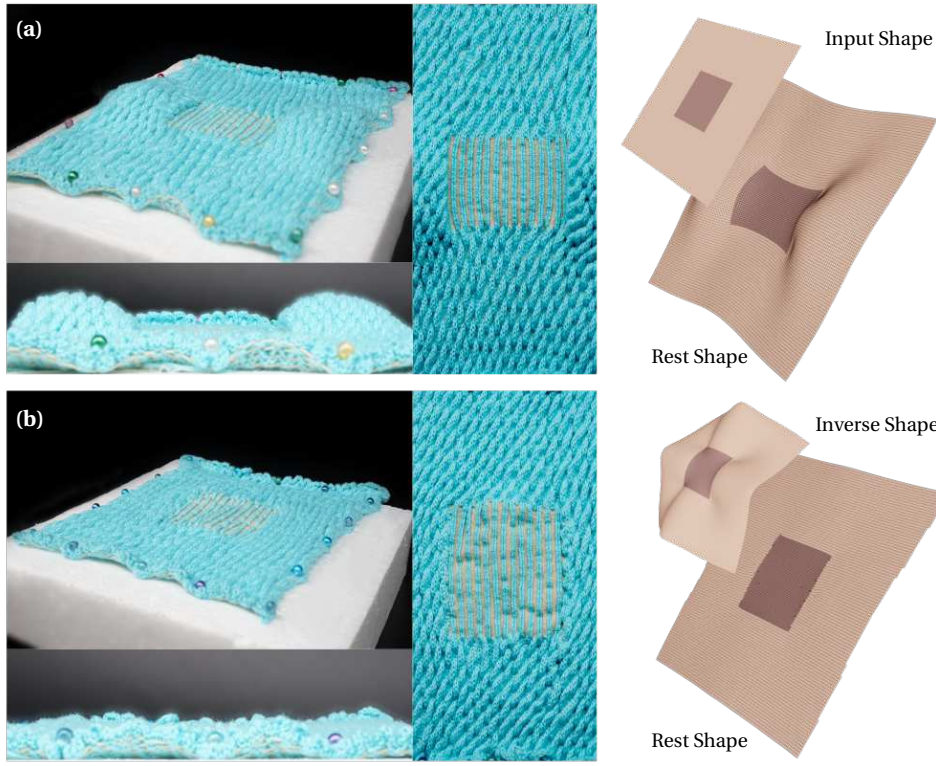



Figure 4.5: Caused by the different average stitch sizes presented on the knitted regions with different elasticity, unwanted wrinkles can be observed on both the physical knitting result and the rest shape obtained from simulation (a). The problem can be solved by computing an inverse shape to generate the stitch mesh and the knitting map – see (b) for the improved physical and simulation results.



applied for evaluating the elasticity on different jacquard patterns, we measure their average stitch sizes on the physical specimens. The size variation is compensated on the input triangular mesh $\mathcal{M}$ by computing an inversely deformed shape $\tilde{\mathcal{M}}$. The basic idea is that a region should be enlarged when it is assigned with a pattern having stitch-size smaller than the standard size $s_w \times s_h$. Inversely, the surface area should be reduced if the assigned pattern's stitch-size is larger than the standard size. In our implementation, the standard size is obtained by measuring the average of stitches on a knitted fabric using completely firm material. Note that, the stitch size variation of jacquard patterns is orthotropic, i.e., it could be larger than the standard size along one direction but smaller along the other direction.

By assigning the required elasticity (i.e., the corresponding jacquard pattern) to each triangle $f \in \mathcal{M}$, $f$'s shape on the inverse geometry $\tilde{\mathcal{M}}$ can be determined as $\tilde{f}$. When the average stitch-size of jacquard pattern assigned to $f$ is $\tilde{s}_w \times \tilde{s}_h$, the scaling ratio $s_w / \tilde{s}_w$ is applied to obtain $\tilde{f}$. A global shape of $\tilde{\mathcal{M}}$ can be obtained by blending these scaled triangles and following the Shape-Up iteration routine [20]. The source curve of geodesic

distance-field is obtained on $\tilde{\mathcal{M}}$. Then, an updated geodesic distance-field $\tilde{F}(\cdot)$ can be computed on $\tilde{\mathcal{M}}$. The isocurves are extracted as wale curves using the constant width $s_w$. We sample an even number of segments on each curve. The segment length on $\tilde{f} \in \tilde{\mathcal{M}}$ is $\tilde{s}_h s_w / \tilde{s}_w$, so that the stitch size on $\tilde{f}$ is $(\tilde{s}_w s_w / \tilde{s}_w) \times (\tilde{s}_h s_w / \tilde{s}_w)$. That is to say the stitch size on $f \in \mathcal{M}$ is $\tilde{s}_w \times \tilde{s}_h$, the same as the design target, if we undo the scaling.

There is another method that does not require inverse shape computing for stitch-size compensation. We can use different propagation speeds to get the distance field $F$ on $\mathcal{M}$. The propagation speed $\|\nabla F\|$ is constantly 1 when computing the geodesic distance field. If $\|\nabla F\| = s_w / \tilde{s}_w$ on face $f$, we can still extract isocurves using width $s_w$. Then the geometric wale width on $f$ is $\tilde{s}_w$. The distance field can be computed by fast-marching using the implementation in [106].

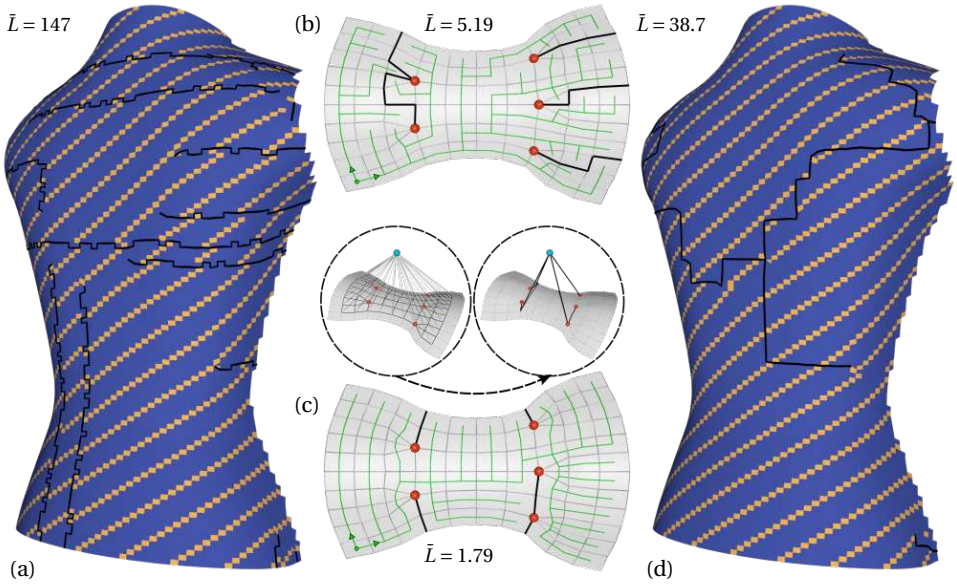



Figure 4.6: Tiling the designed jacquard patterns onto a stitch mesh. Tiling by direct flooding ((a) & (b)), where the order of flooding has been illustrated as green paths shown on the patch in (b). The bold black curves illustrate the boundaries of neighboring stitches with incompatible jacquard patterns – discontinuity. To reduce the length of incompatible curves, we compute them by constructing a *minimal spanning tree* (MST) to connect all apexes (shown as red points in (b) & (c)) and use all boundary nodes as a common virtual node (see the blue point shown inside the circles of (c)). As a result, the discontinuity can be significantly reduced (see the tiling result shown in (d)). The jacquard pattern shown in Fig. 4.3 is employed in (a) and (d) to demonstrate the discontinuity. The total length of incompatible boundary curves $\bar{L}$ is also given for every model.

### 4.3.4. TILING OF JACQUARD PATTERNS

The designed jacquard patterns are tiled on the stitch mesh to realize the designed distribution of elasticity in physical fabrication. First, the stitch mesh is segmented into different regions according to the assigned patterns. After that, each region with the same jacquard pattern is tiled independently.

To tile a jacquard pattern with dimension $m \times n$ onto a region of a stitch mesh is to de-

termine a pattern coordinate $(\bar{i}, \bar{j})$ for each stitch $F_{i,j}$ in the region, where $(i, j)$ denotes the course and the wale indices for a stitch. A proper tiling of a local neighborhood on the stitch mesh should be identical with neighborhood on the jacquard pattern. For two stitches $F_{i,j}$ and $F_{p,q}$ that are edge-neighbor to each other, the difference between their course-wale indices should be consistent with their pattern coordinates (by considering the winding). Specifically, the following conditions should be satisfied:

$$p - i \equiv \bar{p} - \bar{i} \pmod{m}, \qquad q - j \equiv \bar{q} - \bar{j} \pmod{n}. \tag{4.1}$$

When these are not satisfied, the pattern coordinates assigned to $F_{i,j}$ and $F_{p,q}$ are considered as *incompatible*. All the boundaries of incompatible neighbors (see the bold black curves shown in Fig. 4.6) form a seam of tiling, where the discontinuity of patterns will lead to errors in elasticity and thus needs to be minimized. Source of the tiling incompatibility is the irregular vertices at the apexes, where the number of adjacent elements is not four. Moreover, the incompatibility at one apex will also propagate to 1) the boundary of a patch or 2) the other apexes (see the black curves shown in Figs. 4.6(a) and (b). Therefore, it cannot be entirely eliminated when apexes are present in a stitch mesh.

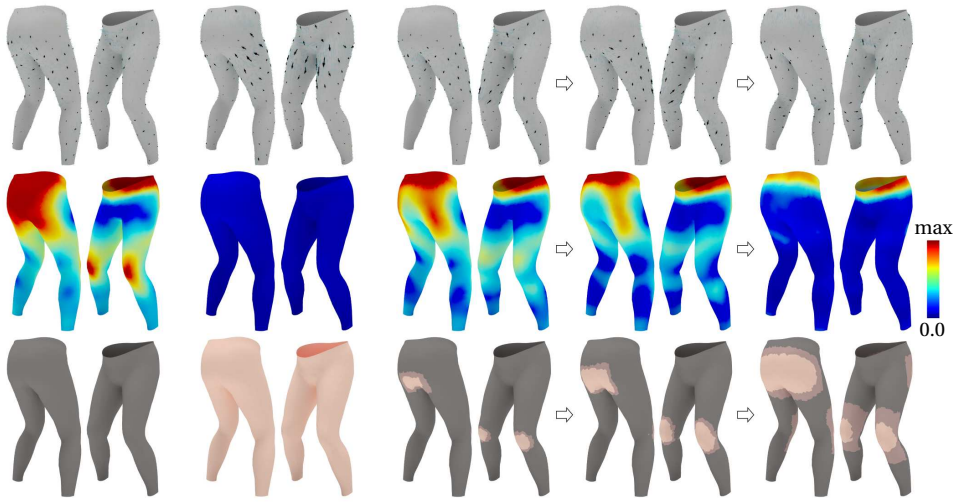
A flooding algorithm can be employed to assign the pattern coordinate of each stitch progressively by incorporating the above condition of compatibility. The order of flooding can be either *breadth-first search* (BFS) or any other orders that ensure to visit every stitch once. However, randomly determined order (and also BFS) can generate long seams (see Fig. 4.6(a) for an example).

Study shows that the incompatibility is caused by apexes and propagated along some seams into other apexes or the surface boundary. To reduce the incompatibility on a stitch mesh, we introduce an algorithm based on *minimal spanning tree* (MST) to minimize the length of seams. First of all, a graph $\Upsilon$ is constructed by considering every apex as a node. All boundary vertices of the region to tile are considered as a common virtual node in $\Upsilon$. An edge between two nodes on the graph $\Upsilon$ is constructed by computing the shortest path between them that travels along the edges of the stitch mesh. The length of this path is employed as the weight of the graph edge. The MST computed on $\Upsilon$ gives a tree of connected paths with minimal total length. When applying the flooding algorithm while avoiding crossing the paths determined by the MST tree, the total length of seams is minimized (see Figs. 4.6(c) and (d)). In fact, Steiner trees will produce the shortest seams by definition. The MST here is an approximation of the Steiner tree.

## 4.4. RESULTS AND DISCUSSION

### 4.4.1. HARDWARE AND IMPLEMENTATION DETAILS

We have tested our computational pipeline to design and fabricate a variety of examples with controlled elasticity. All examples presented in this section are knitted on a fully computerized Shima Seiki SVR093SPSV machine, which is a widely used industrial level machine having two needle beds with 14 needles per inch. Two different yarns – 40S cotton yarn (firm) and 75D rubber yarn (soft) are employed to realize different elasticity by jacquard patterns on the Shima Seiki machine. We generated the jacquard patterns manually considering the maximally allowed float length – that is less than 12 stitches on




(a) Firm material (b) Soft material (c) Progressive updates to soft region

Figure 4.7: Designing leggings with elasticity controlled for squat motion: (a) result by using firm material – with large stresses but small displacements, (b) the result by only using soft material – with small stresses but large displacements, and (c) progressive results by enlarging the soft region to reduce stresses while keeping small displacements. Stresses are displayed as a color map in all results (max = 3.9464kPa), and the displacements are visualized as a vector field in which the magnitudes are rendered as arrows with different lengths. $\bar{d}$ = 2.2cm is employed as the terminal threshold of inverse design.



the Shima Seiki machine we used. We tested 13 patterns and selected the A-B-C patterns in Fig. 4.3 to give maximal, medium and minimal elongation in the course direction. When all stitches are knitted by 40S cotton yarn, we obtain the standard stitch size as $s_w$ = 1.25mm and $s_h$ = 1.00mm. The average stitch sizes of different jacquard patterns can be found in Fig. 4.3.

When designing the distribution of elasticity, the cloth simulator ARCSim [132] that supports anisotropic materials, is employed to evaluate stresses and displacements during the whole sequence of body motion. We modify their code to permit assignment of different material properties at the triangle level, which enables the simulation of clothes with distributed elasticity. Accurate friction is difficult to model since aside from contact forces, it is also affected by skin roughness, local curvature, etc., and may vary in different regions for the same person. For example, elbows may produce higher friction than upper arms. We used the Coulomb friction model [22] (provided within ARCSim) to approximate the friction between cloth and human body. Specifically, we adopted a ball draping experiment to calibrate friction, thickness, density, bending and stretching. The friction coefficient remains unchanged during simulation. The homogenized material properties of different jacquard patterns can be obtained by the data-driven method presented in [207].

Motion data of mannequins, i.e., the long-sleeve example in Fig. 4.1, is captured by using markers for registration [110]. The legging example in Fig. 4.7 is extracted from the deformable human body in SMPL [111].

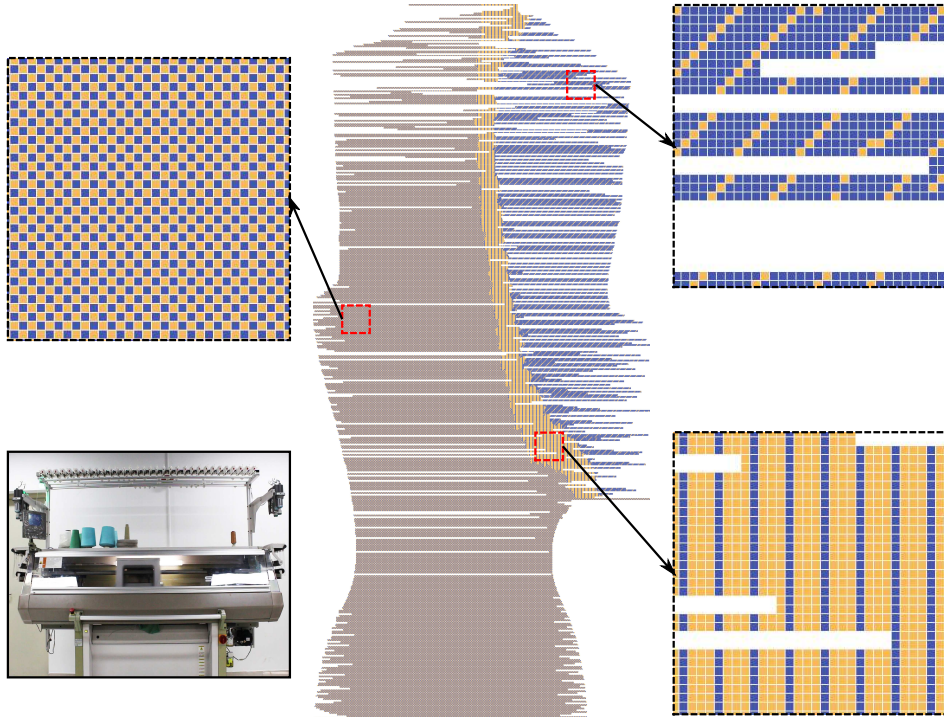


Figure 4.8: The knitting map for the back piece of the 4D garment shown in Fig. 4.1. The zoomed-in views show the tiled jacquard patterns in different regions. The knitting machine used in physical fabrication is also shown.

### 4.4.2. KNITWEAR AS 4D GARMENT

We present results for our proposed method on controlled distribution of elasticity to produce 4D garments. We first apply the inverse design algorithm presented in Section 4.3.1 on an example of leggings. As shown in Fig. 4.7, when fabric with a single material is applied, the motion of a half squat either 1) results in large stresses in the hip and knee regions when using firm material (see Fig. 4.7(a)) – causing discomfort in these regions; or 2) generates large sliding in the thigh region when applying uniformly soft material (see Fig. 4.7(b)) – thus forming unwanted wrinkles by friction. These simulation results are very close to our daily experience. Starting from the region with large stress, our inverse design algorithm progressively enlarges the area of soft material while controlling the maximally allowed sliding (displacements) during the body motion. The progressive results of our algorithm are shown in Fig. 4.7(c).

Similar progressive results to generate 4D garments are shown for a long sleeve garment (see Figs. 4.1 and 4.4). The deformation caused by lifting the right arm is propagated from the arm to a large area on the back when single soft material is used for the whole garment. After using different elasticity in different regions, the deformation decays within a smaller region. Using soft material in the shoulder region also helps to reduce the stress and therefore makes the garment more comfortable. The knitting map


Figure 4.9: One-shoulder knitted top with prescribed material distribution to achieve a desired shape. Firm material on the shoulder produces a stiff style, while soft material on the torso ensures comfort. From the simulation, it can be found that the style cannot be achieved using a uniformly soft material.



with tiled jacquard patterns for this garment is given in Fig. 4.8.

Fig. 4.9 shows a one-shoulder top design. The distribution of elasticity is specified by a designer, who adopts firm material in the shoulder region for styling purposes. Our pipeline provides a useful tool for designers to enable their ideas, achieving both the desired shape and the desired comfort. See the simulation and physical fabrication results in Fig. 4.9. Knitting maps for all 4D examples can be found in Fig. 4.10.

### 4.4.3. DISCUSSION

The main limitation of our approach is the simplification of conducting physical simulation at the triangular mesh level instead of the yarn level (i.e., homogenization is applied). As a consequence, we cannot optimize the distribution of elasticity at the yarn level. The distribution is also updated by heuristically assigning soft material at regions with large stresses instead of sensitivity analysis, which relies on the fast and accurate computation of physical models. Moreover, the orientation of orthotropic materials used in the simulation is assigned according to the geodesic distance-field $F(\cdot)$. After assigning different elasticities and computing the inverse geometry, the geodesic distance field will be changed. Therefore, the orientations of triangles in the material space can also change. In short, the fabric orientation in simulation and the varied inverse geometry caused by different distributions of elasticity are coupled. This is the other reason why we can only use the simulator to 'validate' a distribution of elasticity (the greedy method in Section 4.3.1) rather than conduct an optimization based on sensitivity analysis. It will be interesting to explore the technique of homogenized yarn-level simulation

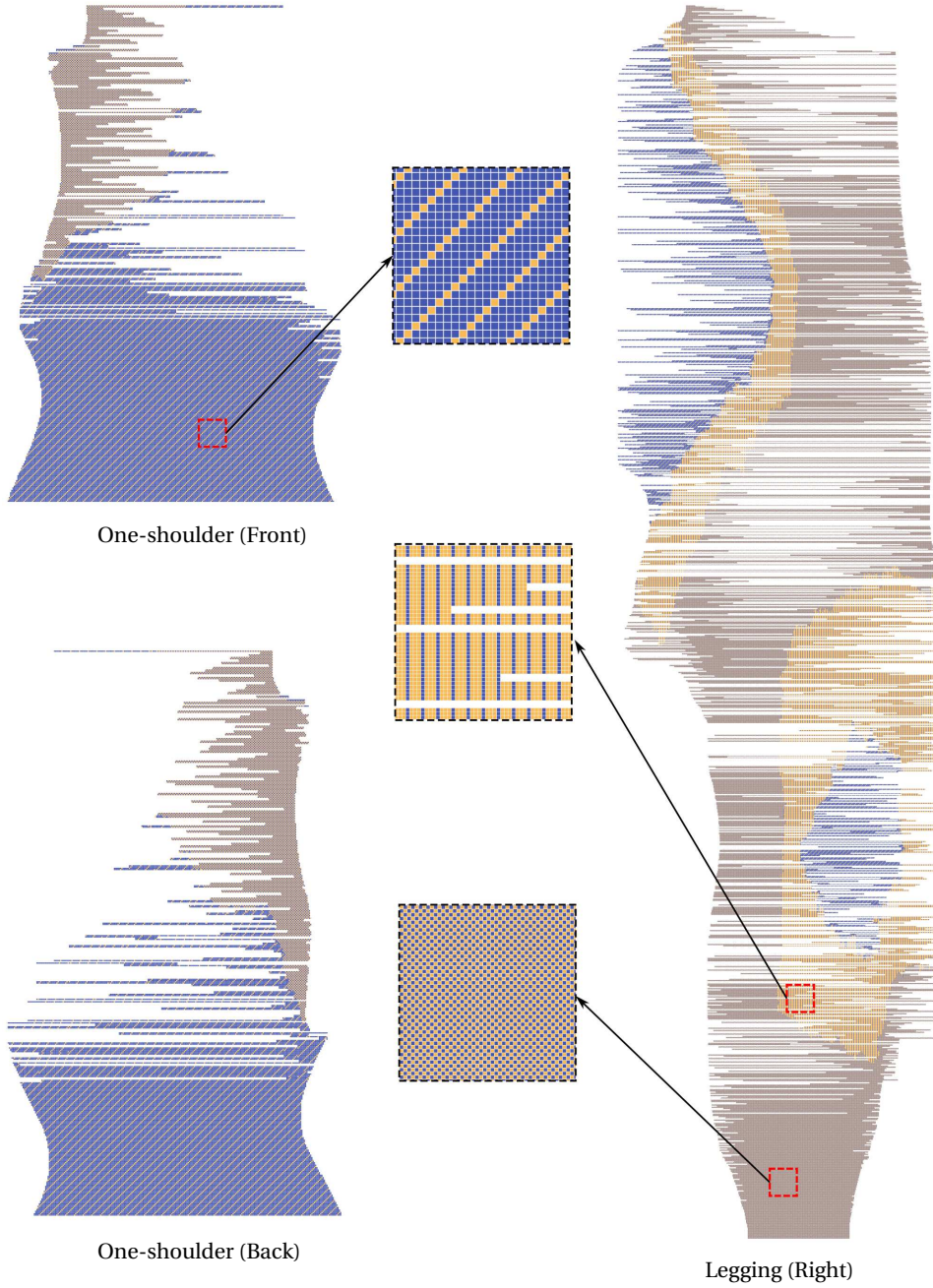



Figure 4.10: Knitting maps of 4D examples: leggings and one-shoulder top. For the symmetric pair of leggings, only the right part is given here.

[178] in future work.

When controlling the shape approximation error of a 3D knitted fabric, we introduced a heuristic to generate scattered apexes. Possible future work is to incorporate physical simulation in the loop of stitch mesh generation so that the locations of apexes can be optimized to further reduce the shape approximation error. Moreover, a simple friction model was employed in our simulation cycle. We plan to explore whether more precise prediction of sliding is possible when sophisticated friction models such as ARGUS [98] are used.

Constrained by float length, the maximal tile size used in our framework is $12 \times 12$. Within this range, the tile sizes are allowed to change freely in our pipeline as the variation is very small compared to the dimension of a full-size garment. Strong deformation is not observed at the discontinuous boundary of tiles. However, it has significant influence on the appearance and increases the chance of generating floats longer than the maximally allowed length.

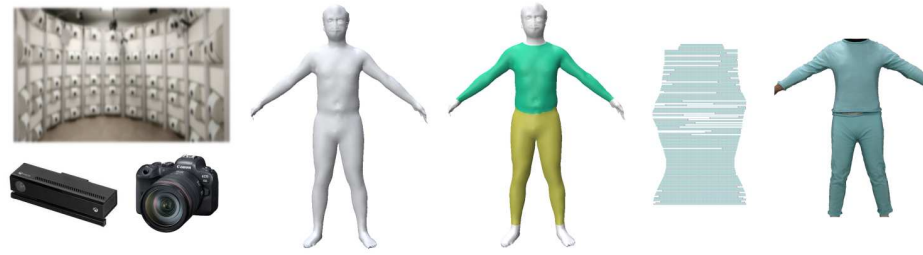
## 4.5. CONCLUSION

We present a computational pipeline to enable the design and fabrication of 4D garments as knitwear with elasticity controlled for body motion. A distribution of elasticity that can be realized by jacquard patterns using two yarns is computed by a physics-driven method. The optimized elasticity on the garment is converted into a knitting map to be executed on digital knitting machines. We have developed new algorithms for this computational pipeline, including distortion-controlled 3D knitting by a short-row shaping technique, tiling to generate jacquard patterns with higher continuity, and shape compensation for incompatible stitch sizes. We verified the performance of our approach on a variety of examples in both simulation and physical experiments. The results are encouraging: both the large stresses that cause pressure and the large sliding that can lead to discomfort are significantly reduced on the 4D garments produced by our pipeline.

# 5

# CUSTOMIZED KNITWEAR DESIGN FOR INDIVIDUALS

## 5.1. INTRODUCTION



(a) (b) (c) (d) (e)

Figure 5.1: The pipeline of scanning to knitting. Using different sensors (a), 3D body shapes and poses (b) can be captured. After analyzing the body model (c), knitting maps (d) can be generated and knitwears (e) are fabricated.

Capturing the customer's body shape is crucial for garment customization. Traditionally, tailors take the measurements by hand. Nowadays, different sensors, e.g., cameras and 3D scanners, can be used. In this chapter, I first present three types of body-capturing methods for individuals. Then knitwear designs are generated using the methods developed in the previous chapters. This chapter tackles the last research question (**RQ3**): how to capture an individual's body models in different poses for 3D/4D knitwear customization.

The human body capturing methods are categorized according to the input data:

The knitted sock in this chapter has been published in: **Z. Liu**, X. Han, Y. Zhang, X. Chen, Y.-K. Lai, E.L. Doubrovski, E. Whiting, and C.C.L. Wang. *Knitting 4D garments with elasticity controlled for body motion*, ACM Trans. Graph., 40(4), 2021. doi: 10.1145/3450626.3459868.

- Multiview RGB images captured by a multiview system.
- A pair of depth images captured by two depth sensors, e.g., Kinects.
- A single RGB image captured by common cameras, including smartphones.

The multiview system is expensive but gives the highest accuracy among these three methods. 3D reconstruction from a single RGB image is an ambiguous problem. We predict the 3D shape with machine learning techniques. The accuracy is the lowest. The price and accuracy of the Kinect-based method are in the middle of the other two approaches.

Relevant human body modeling methods using these sensors have been studied in existing works. The multiview system is easy to use. However, the scanned human body needs to be analyzed for garment making or other downstream applications. In Section 5.2, a template-fitting method is presented to segment the regions of interest from the scanned mesh, which is simple and easy to implement. In Section 5.3, a dual-Kinect calibration method is presented, which only relies on a planar board without any markers. The single-image based reconstruction method in Section 5.4 estimates hair and garment details that other works, like [78, 196], fail to.

Given the target body shape, we can design both body-tight garments and loose garments (Section 5.5). Body-tight garments can be extracted from the 3D body model directly. Loose garments are designed by style transferring given a template garment design and the corresponding reference body model.



## 5.2. BODY MODELING BY A MULTIVIEW SYSTEM

A multiview system (also referred to as a scanner, Fig. 5.2) captures RGB photos from different views covering the human body. The 3D human body can be reconstructed from these photos using the photogrammetry method. The workflow is briefly described below.

1. Photo capturing from different views.
2. Image feature extraction and matching. Feature points are extracted from photos and described as feature vectors (e.g., using [107]). Corresponding points from different views are matched taking advantage of the feature vectors.
3. Sparse reconstruction. If a feature point is observed in multiple photos, then it is possible to recover its 3D coordinates. Even if the camera's intrinsic parameters are unknown, a 3D structure, that differs from the truth up to a transformation, can be reconstructed [59]. A sparse 3D point cloud and camera parameters are obtained in this step. This step is referred to as structure-from-motion (SfM).
4. Dense reconstruction. A dense 3D point cloud can be generated using the photos and estimated camera parameters. This step is named multi-view stereo (MVS).
5. Surface reconstruction. A surface mesh is reconstructed from the dense point cloud.

Understanding the pipeline and photogrammetry principles benefits the scanning. For example, all cameras in the scanner should be synchronized to capture the human

Figure 5.2: Inside of botscan PRO S. More than 70 digital single-lens reflex (DSLR) cameras capture the scene (e.g., a human) inside the scanner from different viewpoints. This is a stitched panorama photo with distortions.

body at the same time. Otherwise, the structure-from-motion step will be inaccurate if the body moves between the captures of different cameras. A physical point can be reconstructed if it is observed from multiple cameras. Therefore the cameras should cover different viewpoints while overlapping their neighboring cameras. A physical point can be reconstructed if its projections are detected as feature points in multiple images. Thus regions with uniform colors or glossy material are difficult to reconstruct. The scanning quality will be improved if the person to be scanned wears diffuse material clothes. More details are available in the tutorial [144]. Numerous works have been done in this area, which will not be discussed here.

The term photogrammetry is used to describe the reconstruction workflow, which implies 3D measurements from photographs. It also covers other techniques like photometric stereo. In this section, we mainly used SfM and MVS.

We used the botscan PRO S[1] system, as shown in Fig. 5.2. Its cameras are well-designed for view coverage and overlapping. It is equipped with four DLP projectors to project feature patterns, which can be used when scanning 3D objects or human bodies with uniform colors. The scanner has fiducial markers (TRIP [112]) as ground control points for metric reconstruction [59]. Photos are exported to commercial photogrammetry software RealityCapture[2] for reconstruction. In our experiments, the open-source software COLMAP [158, 159] is also practicable.

### 5.2.1. TEMPLATE FITTING

The multiview system reconstructs the human body as a surface mesh with textures. Semantic information, like the positions of the neck, shoulder, and waist, is necessary for garment design. I use template fitting for this purpose. Given an articulated human body

[1] https://www.botspot.de/
[2] https://www.capturingreality.com/

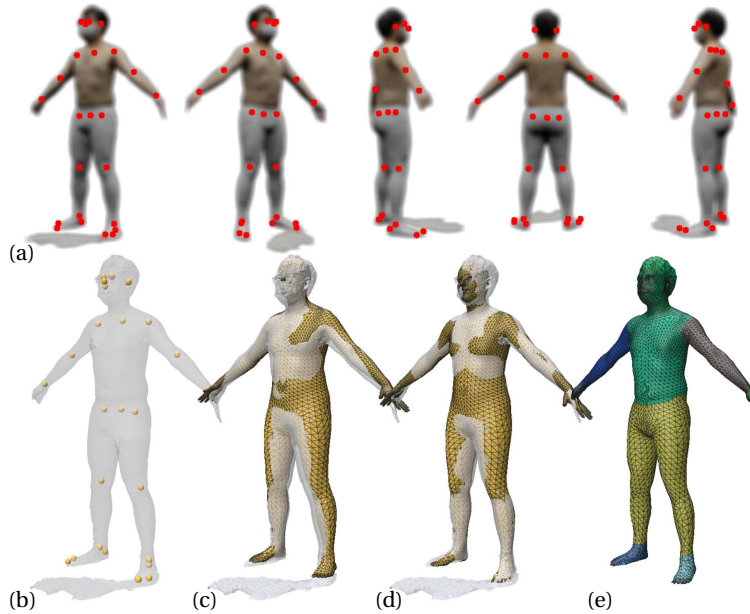



Figure 5.3: The template fitting process. 2D keypoints are detected on the rendered views (a) of the scanned 3D mesh. We reconstruct 3D keypoints (dark yellow balls in (b)) from the 2D views. An initial template model (an instance of SMPL, wireframe mesh in (c)) is estimated using the 3D keypoints. After the gradient-free optimization, the SMPL model (d) is better aligned with the scan (see the left arm and the right chest). The optimization iterations take around one minute. Finally, we project all SMPL vertices to the scanned mesh (e). Predefined information, such as the segmentation in (e), can be projected to the scanned mesh. Images in (a) are blurred to protect privacy. The scanned mesh is semi-transparent in (b–d).



template (like SMPL [111]) with pre-defined body keypoints, we can fit the template onto the scanned mesh to get the body keypoints on the scan.

SMPL (Skinned Multi-Person Linear model [111]) is a parametric human body template with a fixed mesh topology. Given a set of parameters $\Theta$ including body shape parameters, joint rotations, and the global translation, SMPL generates a mesh $\mathcal{S}(\Theta)$ representing a corresponding human body shape. For each human body keypoint (like the belly button, eyes, and joints), the indices of relevant vertices are fixed since the mesh topology is fixed. The keypoint information helps both template fitting and garment design.

Given a scanned mesh $\mathcal{M}$ of a minimally-clothed body, the template fitting objective is to find the optimal SMPL parameter $\Theta$ such that

$$\Theta = \arg\min_{\Theta} \sum_{\mathbf{p}_i \in \mathcal{S}(\Theta)} w_i \, \mathrm{d}^2\left(\mathbf{p}_i, \mathcal{M}\right), \tag{5.1}$$

in which $\mathrm{d}(\mathbf{p}, \mathcal{M})$ is the nearest distance from point $\mathbf{p}$ to mesh $\mathcal{M}$. For this non-convex optimization, a gradient-based method is proposed in BodyNet [196]. We use the (1+1) evolutionary algorithm [44] implemented in [147] to optimize the problem in a gradient-free manner, which is simple yet effective.

In the objective function Eq. (5.1), we use weights $w_i$ to adjust the importance of the vertices on the SMPL model. They are the Voronoi area of each vertex on the default shape of SMPL. In our optimization, the distances in the head, hands, and feet regions are not important. The weights of those vertices are set as zeros. The indices of those vertices are directly retrieved using the predefined body-part information.

The template-to-scan distance is used in Eq. (5.1) to measure the fitting error. We may also use the inverse direction, i.e., the scan-to-template distance, or both directions. The advantage of template-to-scan distance is that some spatial data structures, like an AABB tree, can be constructed from the static scan to accelerate the distance query in optimization iterations. The scan mesh can also be simplified (or decimated) for acceleration.

The initial correspondence between the scan and template is crucial in the optimization. Wrong correspondence means a template point $\mathbf{p}$ is projected to a distant point on scan $\mathcal{M}$, e.g., an arm point projected to the torso. To obtain a good initial template (SMPL parameter), we estimate 3D body keypoints from 2D body keypoints, thanks to the robust off-the-shelf 2D body keypoint detection tools [25]. Given the textured mesh $\mathcal{M}$ of a standing person, we render it as images (Fig. 5.3) from multiple views sampled around $\mathcal{M}$. On each image, we detect 2D body keypoints using OpenPose [25]. For a 3D keypoint, its 2D projection on each view constrains it on a 3D line based on the view's projection matrix. Multiple view projections formulate the sum of squared point-to-line distances as a least-squares problem, from which the 3D keypoint locations are estimated. Details of this part are given in Section 5.A. We estimate the initial SMPL parameter from the solved 3D keypoints, using VPoser (the variational human body pose prior) proposed in [141]. In this step, a conversion from SMPL-X [141] to SMPL is involved.



## 5.3. BODY MODELING BY A DUAL KINECT SETUP

Some depth cameras (or range sensors), such as Microsoft Kinect and Intel RealSense, have been developed for gesture-based user interfaces, for example, video games. They capture the depth information through structured light or time of flight. Researchers use Kinect as a 3D scanner since it has a millimeter-level accuracy and the price is even lower than a DSLR camera.

The output of each Kinect is a depth image, on which each pixel is corresponding to a 3D point. Thus we treat the depth image as a point cloud. The coordinates of the points are defined in the Kinect's camera space, i.e., a coordinate frame rigged with the camera. Same to the multiview RGB image-based method, it is better to use multiple Kinects at the same time, instead of moving a Kinect around the person being scanned. Scanning static 3D objects other than the human body is easier. The rigid point cloud registration methods (Section 2.C.1) can be used to align them together.

I propose to use the dual Kinect setup as shown in Fig. 5.4. There are two Kinects $\mathsf{K}_1$ and $\mathsf{K}_2$ facing each other, with a distance of around 2m. The person being scanned stands in the middle and faces $\mathsf{K}_1$. Then $\mathsf{K}_1$ scans the front side and $\mathsf{K}_2$ scans the back side. Two Kinects are triggered at the same time, producing two frames of point clouds. If the relative position between two Kinects is calibrated, two frames of point clouds can be merged together using the calibrated transformation matrix. The calibration result works for all scans thereafter if the setup is not moved.

Figure 5.4: Setup of the dual Kinect capturing system. A board is scanned for calibration. The blue/red point clouds are scanned by two Kinect separately. (3D models of the Kinect and the tripod are downloaded from GrabCAD.com, uploaded by Tommi Sintonen and Casey Rayback respectively.)

### 5.3.1. CALIBRATION

Two Kinects are facing each other. The point clouds captured by them have little overlap. Thus ICP type registration methods can not be used. We use a board for the calibration. Each Kinect captures one side of the board. Two planes captured by two Kinects are parallel. When we rotate the board several times, the transformation between two Kinects is estimated. The board should be flat with a uniform thickness. Both sides of it should be easy to capture by Kinects, i.e., they can not be dark, shiny, or transparent.

Usually, boards with chessboard patterns or ArUco markers are used for calibration. In [90], a polyhedron with markers is used. However, these patterns are captured by the RGB camera on Kinect, of which the location is different from its depth sensor. Our calibration process only relies on the depth sensor, although it takes time to rotate the board.

The calibration process is to find a transformation matrix that converts the points in the camera space of $\mathsf{K}_2$ to $\mathsf{K}_1$. First, we measure the thickness of the board $c$. Then, we hold the board in the middle of two Kinects and capture it. This step is repeated for $k \geq 3$ times with the board rotated. In a valid new scan, the board can not be parallel with any of the previous poses. Next, we detect the planes of the board in the point clouds.

Now we have $k$ pairs of planes: $\mathbf{n}_{1,i} \cdot (\mathbf{x} - \mathbf{p}_{1,i}) = 0$ in $\mathsf{K}_1$ and $\mathbf{n}_{2,i} \cdot (\mathbf{x} - \mathbf{p}_{2,i}) = 0$ in $\mathsf{K}_2$, for $i = 1, \ldots, k$. Plane normals $\mathbf{n}_{j,i}$ are unit vectors pointing outward the planes. Considering the virtual middle plane of the board, it is $\mathbf{n}_{j,i} \cdot (\mathbf{x} - (\mathbf{p}_{j,i} - (c/2)\mathbf{n}_{j,i})) = 0$. Denoting $\hat{\mathbf{p}}_{j,i} = \mathbf{p}_{j,i} - (c/2)\mathbf{n}_{j,i}$, we have that $\mathbf{n}_{2,i} \cdot (\mathbf{x} - \hat{\mathbf{p}}_{2,i}) = 0$ and $\mathbf{n}_{2,i} \cdot (\mathbf{x} - \hat{\mathbf{p}}_{2,i}) = 0$ are actually the same mid-plane defined in two camera spaces of $\mathsf{K}_1$ and $\mathsf{K}_2$. There exists a rigid transformation $(\mathbf{R}|\mathbf{t})$ that converts planes in space $\mathsf{K}_2$ to $\mathsf{K}_1$. The rotation block $\mathbf{R}$ maps plane normals from $\mathsf{K}_2$ to $\mathsf{K}_1$, i.e.,

$$\mathbf{R}\left(\mathbf{n}_{2,1}\ \mathbf{n}_{2,2}\ \cdots\ \mathbf{n}_{2,k}\right) = -\left(\mathbf{n}_{1,1}\ \mathbf{n}_{1,2}\ \cdots\ \mathbf{n}_{1,k}\right). \tag{5.2}$$

A linear system is obtained by transposing both sides of the equation. When $k > 3$, the linear system is over-determinant, which can be solved in the least-squares sense. The

matrix obtained from the solution may not be orthogonal. We use SVD again to ensure $\mathbf{R}$ is a valid rotation matrix. Next, we solve for the translation block $\mathbf{t}$. We have known that the points $\hat{\mathbf{p}}_{2,i}$ in $\mathsf{K}_2$ can be mapped to $\mathbf{R}\hat{\mathbf{p}}_{2,i}+\mathbf{t}$ in $\mathsf{K}_1$. The mapped point is on the mid-plane, i.e.,

$$\mathbf{n}_{1,i}\cdot(\mathbf{R}\hat{\mathbf{p}}_{2,i}+\mathbf{t}-\hat{\mathbf{p}}_{1,i})=0. \tag{5.3}$$

This is another linear system by taking all $i=1,\ldots,k$, from which the only variable $\mathbf{t}$ is solved.

Now we finished the computation for calibration. Two Kinects cover most regions on the body, which is sufficient for garment design.

### 5.3.2. TEMPLATE FITTING

The point cloud captured by the dual Kinect setup is fitted with a template (SMPL [111]) for garment design. The basic idea of the fitting is the same as that of the last section. We optimize SMPL parameters to minimize

$$\Theta=\arg\min_{\Theta}\sum_{\mathbf{p}_i\in\mathcal{C}} w_i\,\mathrm{d}^2\left(\mathbf{p}_i,\,\mathcal{S}(\Theta)\right), \tag{5.4}$$

The weights can be $w_i=1$ constantly after a uniform sampling. A downsampling of point cloud $\mathcal{C}$ also accelerates the optimization. For some point $\mathbf{p}\in\mathcal{S}(\Theta)$, the point-to-cloud distance ($\mathbf{p}$ to $\mathcal{C}$, like that in Eq. (5.1)) might be large if $\mathbf{p}$'s corresponding point is not captured in $\mathcal{C}$. Therefore the inverse direction, cloud-to-template distance, is used in the objective function. The gradient-free optimizer still works in this task. 3D keypoints can still be used to estimate an initial template. At last, the fitted SMPL template can be used for garment design, being the same as the previous section.

## 5.4. BODY MODELING FROM A SINGLE IMAGE BY DUAL DEPTH REPRESENTATIONS

### 5.4.1. INTRODUCTION

In this work, we propose a machine learning based method to predict the 3D shape of human bodies in general poses from a single RGB image, which can greatly simplify the means to create 3D models for knitwear design and other downstream applications. For example, the predicted 3D mesh model can be used to generate human bodies in different shapes and poses after rigging with a parametric model, as shown in Fig. 5.5. This enables a more user-friendly interface for the application of virtual clothes try-on – i.e., generating appearance images of a user's body shape in target clothes with customized poses.

Recently, several methods have been proposed to tackle the problem of reconstructing 3D human bodies from single images. Some works, such as Bogo et al. [15], Kanazawa et al. [78], applied template parametric models, like SMPL [111], to predict a 3D body shape and also the pose parameters from an input image. With a strong prior of body shape, these approaches can often produce reasonable results but have difficulty in capturing details like hair and cloth. To overcome this difficulty, Varol et al. [196] use a voxel-representation to predict the 3D shape of human body through *Deep Neural Networks* (DNNs), which is more flexible to capture the geometric details. However, the

(a) (b) (c) (d) (e)

Figure 5.5: By using a single RGB image (a) as input, our approach is able to predict a complete 3D human body (b) together with textures (c). The predicted 3D model can be further integrated with the parametric human model – SMPL [111] to easily change the shape (d) and the pose (e) of a human body in applications such as virtual try-on and reenactment.



voxel-representation used in their training pipeline has a lot of redundancy – i.e., only boundary voxels are contributing to the shape description. As a result, memory consumption is very high. This hinders the management of datasets with high resolution in DNNs. Some other works, like [196], represent the human body with voxels, which is more flexible to model details. However, voxel grid representations are redundant. Only voxels at the shape boundaries are informative, whose distribution is sparse. Differently, depth images that are more memory efficient than voxels are employed in [232] to represent 3D human bodies. However, a depth image only partially represents the 3D shape of a human body (e.g., front-view), which does not provide enough information for the aforementioned applications.

In this work, we propose to represent the 3D shape of a human body by a pair of depth images – named as *Dual-Depth Representation* (DDR), which contains two channels resembling two frames scanned from two opposite views (see Fig. 5.6 for an illustration). The 3D geometry of a human body is almost completely described by DDR (Section 5.4.3), except for those occluded regions or the regions tangential to the viewing directions. As already proved in [103], the surface of a human body can be successfully reconstructed from two scans when proper viewing directions (e.g., front and back) are selected.

Generating the DDR of a 3D human body from a single image can be considered an image-to-image translation problem when considering DDR as two additional channels of an input image viewing the human body along the same direction. Benefiting from DDR's compatibility with a 2D image, we can formulate the problem of DDR generation in the framework of a conditional *Generative Adversarial Network* (cGAN) – pix2pix [72]. However, it is observed that directly predicting DDR from RGB by a single cGAN may miss an essential part of a body if there is severe self-occlusion. Therefore, we enrich our framework by feeding it with the image-model pairs together with 2D keypoints of skeletons that are pre-estimated from 2D images (ref. [25]). When the input image is

taken along a direction different from the front-view, the missed region on a predicted DDR could be relatively large. A DNN is trained to repair the missed region by viewing the DDR along other viewing directions. Details of DDR prediction and missed region repairing can be found in Section 5.4.4. As a result, very accurate predictions of 3D human bodies can be generated from input 2D images – we observe an average error of 3.2cm per pixel per channel on our test set.

### 5.4.2. RELATED WORK

**3D shape representation for deep learning:** In recent years, a lot of research has been carried out in the area of deep learning for 3D models. Different from 2D images, it is not trivial to find a representation for 3D shapes that is not only suitable for neural network architecture but also able to encode geometry details. The most straightforward approach is to represent its occupancy on regular 3D grids as a 3D tensor (i.e., voxels [196, 108]). Voxel-based representations are memory expensive. As non-zero elements in voxel grids are usually sparse, several methods are proposed to compress the volumetric representation. Octree based approach [149, 185, 209] is one of the compression strategies, which fits into deep learning architectures with improved data structures for fast access. Another strategy is proposed in [151], which uses 6 orthogonal depth maps from 6 sides of the voxel grid to reduce memory consumption.

Besides voxels, point cloud [48, 75] and mesh [208] have also been used for deep learning. These methods are more flexible to represent geometry details, but the operations are also more expensive because the structures of these geometric representations are not as good as regular grids. Recent effort has been made to develop graph-based convolutional networks that can be directly computed on a triangular mesh [208]. Furthermore, it has also been proposed to represent 3D shapes as 2D images captured from multi-views [180, 101], which is suitable for shape recognition but not for the applications of reconstruction as the 3D information, like correspondence between views, is discarded.

**Human body reconstruction from 2D images:** For reconstructing the 3D human body from a single RGB image, although the approaches for general shapes (like [184, 48, 185, 208]) can be used, the state-of-the-art results are usually achieved with a method specifically developed for the articulated human bodies. A lot of methods have been developed by using human body templates. SCAPE [4] is one of the human body templates which represent human shape and pose with a set of parameters. Several reconstruction methods, such as [7, 54], are developed based on SCAPE. In [111], a more accurate and simple template SMPL is proposed. Many recently proposed methods [15, 78, 140] are based on fitting body shape and pose parameters of SMPL. These methods usually can produce reasonable results by taking advantage of the template and a prior of parameters to avoid self-intersection and impossible joint angles. Differently, [196] proposed to reconstruct a volumetric human model together with 2D/3D keypoints estimation within a multi-task framework.

Researchers also tried to reconstruct 3D human bodies from multi-view images. A multi-view reconstruction method is proposed in [3], where the SMPL template with offset is used. The template with offset method is more flexible compared to the pure

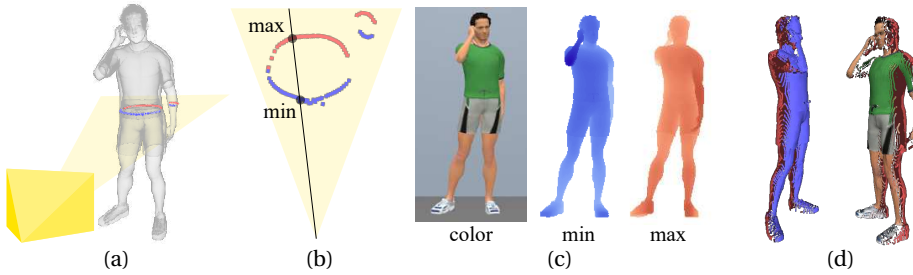



Figure 5.6: Dual-depth representation of a 3D human model: When a 3D model is projected onto the image (a), a set of points on the 3D shape is projected at the same pixel (b). Points that have the minimum/maximum depth are stored in the min/max channels of the dual-depth image respectively – see (c). The dual-depth representation of a human model can be easily converted into a point cloud as shown in (d), which imitates the two pieces of point clouds scanned from opposite directions of a human body – i.e., front and back views.

template, but still not able to generate geometric shapes that are far different from the template, such as long open hair or skirts. In another multi-view approach, [189], a volumetric representation is used.

**Conditional image synthesis:** Recently, conditional generative adversarial network (or cGAN) has been demonstrated to synthesize high fidelity images [72, 233] under the conditions of target image style and/or content. A general cGAN framework pix2pix is proposed in [72] for image-to-image translation when paired training data is available. Many problems of image processing and computer vision can fit into this framework, such as denoising, inpainting and super-resolution. Conditional GAN has also been used for view generation, especially clothed human view synthesis in [117, 166].

In this work, we adopt the pix2pix framework [72] for predicting the 3D shape of a human body based on a newly proposed dual-depth representation. Moreover, cGAN can be trained without paired supervision [233] , and can generate images with a high resolution up to $1024 \times 2048$. [210]. Our framework of 3D body shape prediction from a single RGB image is compatible with these features due to the properties provided by DDR.

### 5.4.3. DUAL-DEPTH REPRESENTATION

A depth image $\mathcal{I}$ can be considered as the projection of a 3D model $\mathcal{M}$ onto the image plane along the viewing direction. Each pixel of the image plane stores the *minimum* distance between the pixel and its corresponding 3D points on the projected model. In other words, a depth image only records the visible points from the viewpoint, which results in a partial 3D representation of the 3D model.

In this work, we proposed to represent the 3D shape with a dual-depth image $\mathcal{D}^{\mathbf{d}}$ according to a viewing point $\mathbf{d}$, which has two channels $\mathcal{D}^{\mathbf{d}} = (\mathcal{I}_{\text{min}}, \mathcal{I}_{\text{max}})$ storing both the *minimum* distance (denoted by $\mathcal{I}_{\text{min}}$) and the *maximum* distance (denoted by $\mathcal{I}_{\text{max}}$) along each projection line on a pixel (see Fig. 5.6). $\mathcal{I}_{\text{min}}$ covers all the points that are visible at the viewing point $\mathbf{d}$. $\mathcal{I}_{\text{max}}$ stores the points that are hidden 'behind' $\mathcal{M}$ but ideally will be visible by viewing $\mathcal{M}$ from its 'back'. With these min-max depth-channels,

the 3D geometry of $\mathcal{M}$ can be better captured except 1) the regions that are occluded from both the 'front' and the 'back' views and 2) the surfaces that are tangential to the viewing direction.

For the purpose of predicting the 3D shape of a human body from a single RGB image input, the dual-depth representation shows the following properties for supporting the computation in the framework of cGAN:

- **Completeness:** First, it provides a more complete 3D representation compared to a conventional depth image (i.e., with only $\mathcal{I}_{\text{min}}$). In our experiments, DDRs cover around 70% of surface areas, which is twice as much as the conventional depth image.
- **Compactness:** Second, DDR is more compact than the voxel representations (either binary voxel-sets or signed distance fields). The compactness makes both the memory effectiveness and the computational complexity of the 3D prediction problem similar to the other 2D image-to-image translation task. Therefore, it is possible to achieve $256 \times 256$ or higher resolutions on a normal hardware setup (e.g., $512 \times 512$ on a single graphics card with 11GB memory).
- **Compatibility:** Third, DDR is well aligned with the input RGB image. The pixel-wise correspondence, disregarding the blurry boundaries, helps to capture some details that template-based representations can not handle. Moreover, this pixel-wise structure of DDR is very compatible with the operators used in GAN frameworks.



While compactness and compatibility can be concluded from analysis, we conduct experiments to further verify the completeness of DDR below.

To study the capability for the geometric representation of DDR, a ratio of area coverage is used for the quantitative evaluation. Given a watertight polygon mesh $\mathcal{M}$ and a viewing point $\mathbf{d}$, a DDR can be obtained and denoted by $\mathcal{D}^{\mathbf{d}}(\mathcal{M})$. We define the area coverage ratio $r(\mathcal{D}^{\mathbf{d}}(\mathcal{M}))$ as the area of regions covered by $\mathcal{D}^{\mathbf{d}}(\mathcal{M})$ to the total visible surface area of $\mathcal{M}$ from any viewpoints outside $\mathcal{M}$. The cumulative distribution of $r(\mathcal{D}^{\mathbf{d}}(\mathcal{M}))$ for all $\mathcal{D}^{\mathbf{d}}(\mathcal{M})$ in our dataset $\mathcal{S}$ (details will be explained in Section 5.4.5) is plotted as the upper curve in the chart shown in Fig. 5.7, which provides statistical information about the percentage (horizontal axis) of a representation's coverage ratio above a given value (as a point on the curve). Specifically, for a point $(\bar{p}, \bar{r})$ on the curve of DDR, it defines that

$$\bar{p} = \frac{\#\{\mathcal{D}^{\mathbf{d}}(\mathcal{M}) \in \mathcal{S},\, r(\mathcal{D}^{\mathbf{d}}(\mathcal{M})) \geq \bar{r}\}}{\#\{\mathcal{D}^{\mathbf{d}}(\mathcal{M}) \in \mathcal{S}\}} \tag{5.5}$$

with $\#\{\cdots\}$ indicating the number of elements in a set. Three example DDRs at different coverage ratios are selected and visualized as point clouds in Fig. 5.7. The DDR at the left has a coverage ratio of 0.85, on which the missed parts are only caused by undersampling in the surface regions tangential to the viewing direction. The coverage ratio of the DDR in the middle is 0.67, which can still express the overall 3D geometry. In our test dataset $\mathcal{S}$, 80% of DDRs covers areas more than this case. The worst case of DDR in $\mathcal{S}$ is shown at the right of the figure with a ratio of 0.50. In this extreme case, almost the full area of the model's left arm is tangential to the viewing direction. In addition, the

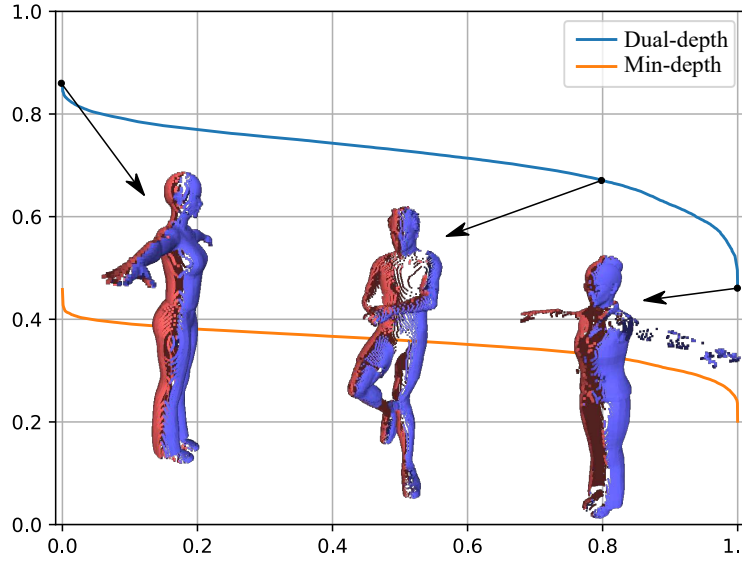



Figure 5.7: Area coverage distribution of the dual-depth $\mathcal{D}^{\mathbf{d}}(\mathcal{M})$ vs the minimal-depth image $\mathcal{I}^{\mathbf{d}}_{\text{min}}(\mathcal{M})$ in our dataset $\mathcal{S}$. For the point on a curve corresponding to a representation, given the point's vertical coordinate as the coverage ratio $r$, the point's horizontal coordinate indicates the percentage of models in the test dataset satisfies this coverage ratio.



inside parts of both legs are missed due to occlusion. For models in such orientations, keypoints indicating the skeleton of a human model are needed to successfully predict the 3D shape of a human body from a single RGB input image. And the geometry of missed regions can be recovered with the help of a view-based refinement network, the detail of which can be found in Section 5.4.4.

As a comparison, the distribution of coverage ratio by using only the min-depth images is also plotted in Fig. 5.7, where the point on the curve can be obtained by replacing $\mathcal{D}^{\mathbf{d}}(\mathcal{M})$ in Eq. (5.5) by $\mathcal{I}^{\mathbf{d}}_{\text{min}}(\mathcal{M})$. In summary, we can see that DDR makes a trade-off between completeness and compactness – i.e., offering a new representation that is suitable for single image based human body modeling.

### 5.4.4. FRAMEWORK FOR 3D PREDICTION

We propose a framework, as illustrated in Fig. 5.8, for predicting the 3D shape of a human body from a single RGB image. The framework is mainly based on a conditional GAN for DDR prediction. As an option, for a predicted DDR with a large area missed, a view-based refinement module based on a deep neural network is conducted to complete the occluded and tangential surface regions.

As a pre-processing step, 2D keypoints of a human model are estimated from the input RGB image with resolution: $w \times h$. All the 2D keypoints on both the training and the test images are generated by the off-the-shelf estimator, OpenPose [25]. The 2D coordinates of 18 keypoints are generated for each image, including 13 body joints and 5 face landmarks.

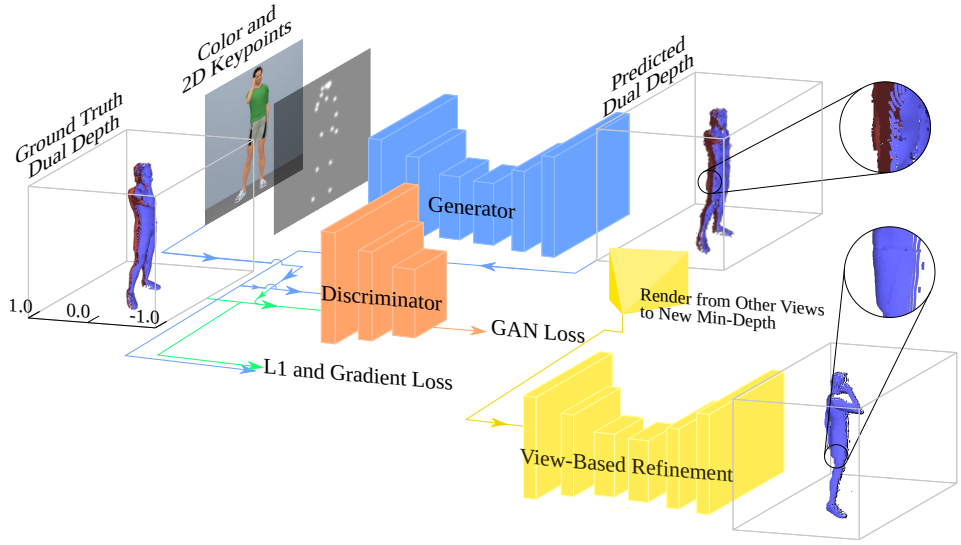



Figure 5.8: Framework for 3D prediction of human bodies: Given pairs of RGB images with 2D keypoints and DDR of ground-truth 3D models, a conditional generative adversarial network is trained. As an option, a view-based refinement network can be used to further improve the predicted DDR from other viewpoints. Note that, all models are scaled into the range of depth value at $[-1,+1]$, and the centers of pelvises are placed at the plane with zero depth value.



### DDR PREDICTION

We now present the cGAN framework for predicting DDR of a human body from a single RGB image and the 2D keypoints computed from the RGB image in the pre-processing step. For each pair of 2D-3D image correspondence, we have a 3-channel input RGB image, denoted as $x_1$, an 18-channel of 2D keypoints, denoted as $x_2$, and a 2-channel dual-depth image, denoted as $y$. Note that, each 2D keypoint is stored as a $w \times h$ heatmap, i.e., a small-deviation Gaussian kernel located at the 2D keypoint. For illustration, all 18 channels of 2D keypoint heatmaps are composited as a gray-scale image and shown in the top-left of Fig. 5.8. Now we need to train a mapping from this 21-channel image $x$ to a 2-channel image $y$, in which $x$ is concatenated as $(x_1, x_2)$.

Fitting the mapping from $x$ to $y$ is naturally an image-to-image translation problem. We adopt the cGAN framework pix2pix [72] to solve it. In this cGAN, a generator $G : (x, z) \mapsto y$ produces DDR output from the input information together with the random noise $z$. A discriminator $D$ is trained to detect the generator's 'fakes'. Adversarially, the generator $G$ is trained to predict DDRs that cannot be distinguished from the ground-truth DDRs by the discriminator $D$. That is to say, $G$ tries to minimize the loss function below while $D$ tries to maximize it,

$$\begin{aligned}\mathcal{L}_{\text{cGAN}}(G,D) = \mathbb{E}_{x_1,y}[\log D(x_1,y)] + \\ \mathbb{E}_{x,z}[\log(1 - D(x_1, G(x,z))].\end{aligned} \tag{5.6}$$

In addition to the above GAN loss, an $L_1$ loss below is used to encourage the output

$G(x,z)$ to be close to the ground-truth $y$,

$$\mathcal{L}_{\mathrm{L1}}(G) = \mathbb{E}_{x,y,z}[\|y - G(x,z)\|_1]. \tag{5.7}$$

Different from the output RGB images in other applications (such as photo synthesis in [72]), the output images as DDR in our framework have very clear geometric meanings to present a solid model, the surface of which must be smooth and regular. To impose this constraint on the outcome of our cGAN, another $L_1$ term defined in the gradient domain below is added to encourage more smooth results.

$$\mathcal{L}_{\nabla}(G) = \mathbb{E}_{x,y,z}[\left\|(\nabla_h + \nabla_v)\left(y - G(x,z)\right)\right\|_1], \tag{5.8}$$

in which operators $\nabla_h$ and $\nabla_v$ are gradients along the horizontal and vertical directions respectively.

After incorporating all these terms, the final objective of our cGAN is formulated as

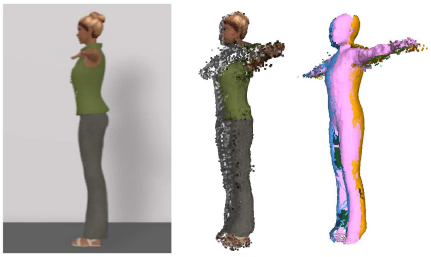
$$\begin{aligned} G^* = \arg\min_G \max_D \mathcal{L}_{cGAN}(G,D) + \\ \lambda_1 \mathcal{L}_{L1}(G) + \lambda_2 \mathcal{L}_{\nabla}(G). \end{aligned} \tag{5.9}$$

Note that, the same as [72], the noise $z$ appears in the form of dropout in generator $G$ at both training and testing time. The values of $\lambda_1$ and $\lambda_2$ are picked empirically. We use $\lambda_1 = 0.03$ and $\lambda_2 = 0.06$ in this framework for DDR prediction.

In our dataset, foreground pixels of DDR, i.e., pixels representing the human body, are normalized into the range of $[-1,+1]$ and the pelvis center depths of human models are placed at the plane with zero depth value. At background pixels, the values are filled with extrema $\pm 1$ as a result of the $z$-buffer algorithm [119]. This leads to a jump between the foreground and background depths at the boundary of foreground regions. As a result, the performance of DDR prediction will be mainly influenced by the foreground-background boundary, instead of the interested region (i.e., foreground), because of its large variance in depth value. To solve this problem, a background centering compensation is used, which is similar to the commonly used data pre-processing method of zero centering [96]. We assign the depth values of all background pixels (no matter in $\mathcal{I}_{\min}$ or $\mathcal{I}_{\max}$) to be *zero*. After this, the network in our prediction framework is easier to be trained, and the foreground and background pixels are easy to be distinguished.

### VIEW-BASED REFINEMENT

After predicting the DDR from a single RGB image, the overall 3D shape of a human body is well captured when the input image is taken along a proper viewing direction (e.g., the front view as shown in Figs. 5.6 (c) & (d). However, the occluded and the tangential regions are missed in general. To complete these missed regions, we propose a view-based refinement step to predict the missed regions from a DDR predicted by the aforementioned cGAN. The view-based refinement is realized on a general encoder-decoder DNN, which is fed by the rotated 3D point cloud from a predicted DDR. The output is a repaired point cloud with the missed regions repaired from the input point cloud.



(a) (b) (c)

Figure 5.9: View-based refinement of the lowest coverage case. Input RGB image (a), predicted dual-depth representation (b), refinement from four other views (c).

Given a predicted DDR $\mathcal{D}^{\mathbf{d}}$, we first extract the foreground pixels of $\mathcal{D}^{\mathbf{d}}$ by removing all other pixels with their depth value between ±1cm. The foreground pixels are then converted to 3D points. Next, the point cloud is rotated around the vertical axis with an angle $\phi$, and the point cloud is rendered as a new min-depth image $\mathcal{I}_{rot}$. Lastly, $\mathcal{I}_{rot}$ is fed into the refinement network. Then, the repaired min-depth image $\hat{\mathcal{I}}_{rot}$ can be obtained as the output of the network. Generally, the view-based refinement network serves as a virtual scanner that 'acquires' 3D depth image of a model from a new viewpoint. The training of this network is based on paired data generated from the dataset of 3D models (detailed in Section 5.4.5).

In our experiments, four new views that are ±45° and ±135° from the current viewing directions are used to refine the predicted models in DDR. The refined depth images are converted into 3D points and fused into a single point cloud, as shown in Fig. 5.9. Poisson surface reconstruction [85] is employed to generate the mesh model for a predicted 3D human body.

### 5.4.5. EXPERIMENTS

#### DATASET

Our dataset $\mathcal{S}$ used for training the DDR prediction network and the view-based refinement network is generated by using textured 3D human models. Part of the models is from the Pose-Varying Human Model (PVHM) dataset, which is a collection of synthetic human models released in [232]. PVHM contains 22 models with various appearances. Each of them is deformed into up to 1,200 different poses, resulting in 10,200 textured human models represented by polygon meshes. As models in PVHM are synthesized 3D models, which in general lack realism when being rendered into 2D RGB images. To enrich our training dataset, we collected 688 additional human models from Trimble 3D Warehouse[3] – examples of which are presented in Fig. 5.10. Other widely used human body datasets either have limited appearance types (e.g., Human3.6M [69]) or lack geometry details of hair and cloth (e.g., SURREAL [195]). We split all 3D models randomly with a proportion of 70% for training ($\mathcal{S}_T$), 10% for validation ($\mathcal{S}_V$) and 20% for evaluation ($\mathcal{S}_E$).

Human models in $\mathcal{S}_T$ are rendered to obtain ground-truth 2D (RGB) and 3D (DDR) pairs for training. All models are posed to standing on the $x$-$y$ plane with upright orientation placed along the $z$-axis. The virtual camera for rendering is set at 6 meters away, facing the center of the model horizontally. The focal length is fixed at 800 pixels. All 2D RGB images are rendered in the resolution of $264 \times 284$ to be prepared for random cropping in training. Each model in PVHM is rotated randomly 20 times along the $z$-axis, while each model from 3D Warehouse is rotated 100 times for each. Camera settings

[3] https://3dwarehouse.sketchup.com/

Figure 5.10: Example data collected from Trimble 3D Warehouse.



are the same for both the color and the DDR rendering. Different background colors, floor textures, and light settings are used in color rendering to obtain more realistic and diverse RGB images.

The dataset $\mathcal{S}$ is also used to train the view-based refinement network. Training of the refinement network is conducted after the DDR prediction network is ready to use after cGAN training. Input-output pairs for training the refinement network are prepared as described in Section 5.4.4.

RGB images from the in-shop clothes retrieval benchmark of DeepFashion [105] are used in our tests to evaluate the effectiveness of our method on real photos. All the images have a resolution of 256 × 256 pixels. Example results of DDR prediction are shown in Fig. 5.11. It demonstrates that our approach can successfully reconstruct the 3D shape of a human body from single images with varying viewpoint, pose, cloth, hair, and background.

### IMPLEMENTATION DETAILS

Our networks are implemented with PyTorch [139]. The training and inference are performed on a PC with a single GTX 1080 Ti GPU. When training the DDR prediction network as a cGAN, we use the random cropping from 264 × 284 to 256 × 256 together with the random horizontal flipping for the data augmentation. The batch size was set as 4, and the network was trained for 4 epochs. Each epoch takes around 160 minutes. And we set the learning rate at $10^{-4}$ for the first two epochs and $10^{-5}$ for the last two epochs. For the view-based refinement network, the batch size and the number of epochs are the

Figure 5.11: Example dual-depth images as the results of prediction. It is easy to find that dressed 3D human bodies in a variety of body shapes and poses can be successfully 'reconstructed' from input RGB images.

same as above. Similar learning rates are used in the epochs, but the training takes much shorter time than cGAN (i.e., around 50 minutes per epoch).

Our network architectures are akin to [72]. The DDR prediction network is a conditional generative adversarial network (cGAN) consisting of a generator and a discriminator.

The generator has an encoder-decoder architecture based on U-Net [150], which is given by

- encoder: `C64-C128-C256-C512-C512-C512-C512-C512`,
- decoder: `CD512-CD1024-CD1024-C1024-C1024-C512-C256-C128`.

In this generator, the input has 21 channels, and the output has two channels. Sizes of input and output are both $256 \times 256$. Each encoder layer downsamples by a factor of 2 while each decoder layer upsamples by the same factor of 2. A `C`$k$ layer is a convolutional layer with $k$ filters, followed by batch normalization (BatchNorm), except as otherwise noted. All convolutions have kernel size 4, stride 2, and padding size 1. A `CD`$k$ layer is a `C`$k$ layer with a dropout rate 0.5 after BatchNorm. Activations are applied between the convolutional layers. All activations in the encoder are leaky ReLUs, with a negative slope of 0.2, while ReLUs in the decoder are not leaky. Activation in the last layer of the decoder is a Tanh function. BatchNorm is not applied to the first `C64` layer of the encoder.

The encoder-decoder has a U-Net architecture, in which each layer $i$ in the encoder is concatenated to layer $n - i$ in the decoder, where $n = 16$ is the total number of layers.

Discriminator of the cGAN is applied on $70 \times 70$ patches, of which the architecture is given by: `C64-C128-C256-C512`. Notations for the layers are the same as that for the generator. A Sigmoid function is used after the last layer to predict a one-dimensional value between 0 and 1 for the input patch. The other activations are leaky ReLUs with a negative slope of 0.2. BatchNorm is not applied to the first `C64` layer.

### VIEW-BASED REFINEMENT NETWORK

The view-based refinement network has only an encoder-decoder architecture, which is the same as the generator for DDR prediction, except that its input has 19 channels, and the output has 1 channel.

### LOSS FUNCTION

We use the dataset $\mathcal{S}_E$ to evaluate the functionality (i.e., loss function) of our DDR prediction network. Three metrics are used for the evaluation: the average error of an image (EI), the average error per point (i.e., per pixel per channel, EP), and the average error of gradient image (EG). Specifically, we have

$$\text{EI} = \frac{1}{N}\sum_{i=1}^{N}\left\|\mathcal{D}_i^{\mathbf{d}} - \mathcal{D}_i^{\mathbf{d}*}\right\|_1, \tag{5.10}$$

$$\text{EP} = \frac{1}{N}\sum_{i=1}^{N}\frac{1}{2M_i}\left\|\mathcal{D}_i^{\mathbf{d}} - \mathcal{D}_i^{\mathbf{d}*}\right\|_1, \tag{5.11}$$

$$\text{EG} = \frac{1}{N}\sum_{i=1}^{N}\left\|(\nabla_h + \nabla_v)\left(\mathcal{D}_i^{\mathbf{d}} - \mathcal{D}_i^{\mathbf{d}*}\right)\right\|_1, \tag{5.12}$$

where $N$ is the number of DDRs in the test set, and $M_i$ is the number of foreground pixels in the $i$-th DDR. All metrics are evaluated as $L_1$ errors on foreground pixels only.

The evaluation results of the loss function in our DDR prediction network are shown in Table 5.1. Different from the pix2pix implementation, the loss function used in our network has an additional $L_1$ term of gradients. It has been experimented to remove this term (w/o gradient term), i.e., $\lambda_2 = 0$ in Eq. (5.9). It can be found that the EI error was slightly reduced when the gradient term was removed. However, the error in the gradient domain increases by 16% at the same time. That means the gradient term can significantly improve the output gradient while almost achieving the same $L_1$ loss. Similar results were also observed to achieve better results after adding the gradient term in [136].

| | EI | EP (cm) | EG |
|---|---|---|---|
| w/o gradient term | 278.770 | 3.061 | 206.016 |
| w/o bg. centering | 310.443 | 3.412 | 263.467 |
| full loss | 287.818 | 3.167 | 182.074 |

Table 5.1: Test error of DDR prediction networks trained with variants of the loss function on the testing set $\mathcal{S}_E$.

The effectiveness of background centering is also verified by comparing it with the result trained on the same data without the process of background centering (w/o bg. centering) but keeping everything else the same as when using the full loss function. When background centering is dropped, the $L_1$ error (EI) increases by 10%. At the same time, the other two errors (EP and EG) also increase significantly. Without the preprocess of background centering, the network was trained to fit the variance coming from the jump between foreground and background.

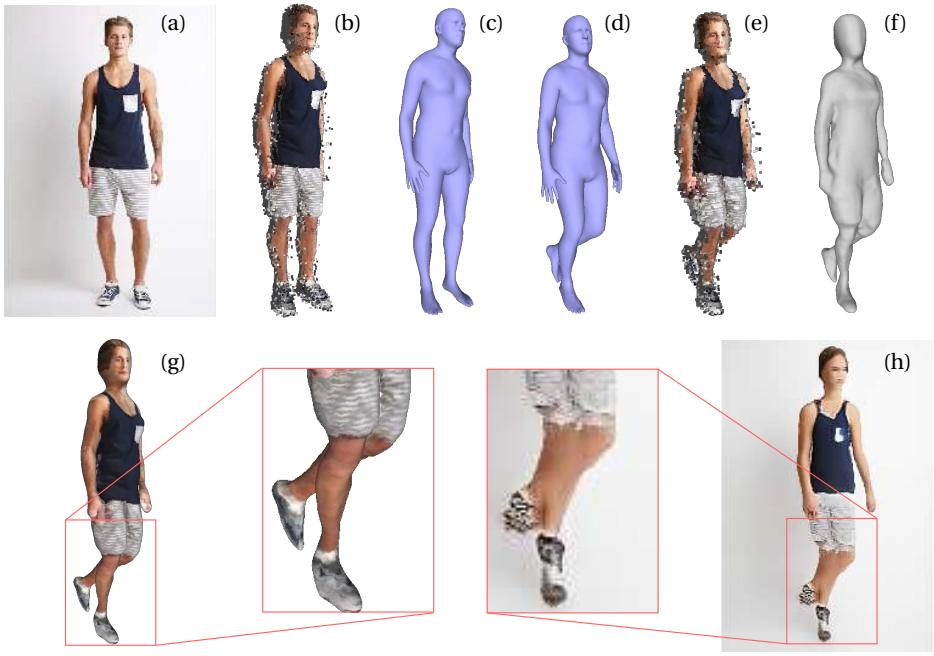



Figure 5.12: Step-wise result of DDR-based reconstruction and reenactment (a – f). The second row shows our result (g) compared to the result of [166] (h).



### ADVANTAGE OF 3D APPROACH

The downstream applications (such as reenactment) can be benefited by the 3D reconstructed human model. As shown in the first row of Fig. 5.12, we first fit the reconstructed point cloud (b) to the deformable parametric model SMPL [111] by using the extended SMPLify [196] method. In this step, the reconstructed point cloud is rigged with the SMPL model (c). When a new set of SMPL parameters is given (d), the rigged point cloud will be deformed to the target pose and shape (e). After that, the triangular mesh of a human model can be reconstructed for the deformed point cloud (f).

For the application of reenactment (i.e., person image generation), our 3D approach is compared to a 2D approach [166] as shown in the second row of Fig. 5.12. Benefiting from the reconstructed 3D structure, our method can easily handle out-of-plane rotation and occlusion between body parts. However, [166] has the advantage of synthesizing unseen regions. It will be an interesting future work to combine the 2D approach with our reconstruction method.

### GEOMETRY DETAIL AND ALIGNMENT

Single image based human body reconstruction results of our DDR-based approach are compared with HMR (Human Mesh Recovery) [78] and BodyNet [196] (see Fig. 5.13). Thanks to the flexibility of DDR and the newly collected dataset, our approach can successfully reconstruct hair and loose cloth while the other two methods cannot. The HMR

(a) (b) (c) (d)

Figure 5.13: With an input image shown in (a), the results from HMR [78] (b), BodyNet [196] (c), and our DDR approach (d) are compared.



method recovers a SMPL mesh, which is more suitable for deformation manipulation but fails to reconstruct the details. Second, DDR has a pixel-wise correspondence with the input RGB image by nature, which results in a reconstruction well-aligned with the image (although some error is introduced in the surface reconstruction step). The BodyNet approach is not aligned as well as ours because its voxel representation does not ensure such correspondence, as shown in the bare regions in the second example of Fig. 5.13.

### QUANTITATIVE EVALUATION

We measure the errors of point-to-surface distance and compare with the results of the multi-view approach [3] evaluated on BUFF dataset [223] (listed in Table 5.2). The point-to-surface error is plotted in Fig. 5.14. Errors at the extremities of the body are larger than in the other regions because of the ambiguity in estimating the articulated pose from a single view. It can also be observed that the back depth map does not show a significantly larger error than the front depth map.

We trained the view synthesis model [184] with the source code released by the authors[4]. The model was trained on our dataset for 400,000 iterations using the default setting in the source code. Given a single RGB image rendered with a body mesh from the BUFF dataset [223], depth maps from 0° and 180° views are synthesized with the trained model. We evaluated this result and ours with point-to-surface error. The color-coded error is plotted in Fig. 5.15. The view synthesis model recovers the overall shape of

[4] https://github.com/lmb-freiburg/mv3d

| | Subject ID | Multi-view [3] | Single-view (ours) |
|---|---|---|---|
| t-shirt, long pants | 00005 | $0.51 \pm 0.54$ | $1.22 \pm 0.98$ |
| | 00032 | $0.48 \pm 0.53$ | $1.42 \pm 1.01$ |
| | 00096 | $0.56 \pm 0.65$ | $1.55 \pm 1.25$ |
| | 00114 | $0.42 \pm 0.51$ | $1.63 \pm 1.37$ |
| | 03223 | $0.49 \pm 0.48$ | $1.03 \pm 0.86$ |
| soccer outfit | 00005 | $0.54 \pm 0.67$ | $1.52 \pm 1.33$ |
| | 00032 | $0.80 \pm 0.86$ | $1.36 \pm 1.09$ |
| | 00114 | $0.50 \pm 0.58$ | $1.41 \pm 1.17$ |
| | 03223 | $0.55 \pm 0.57$ | $1.33 \pm 1.11$ |

Table 5.2: Quantitative evaluation on BUFF dataset [223] with ground truth 3D mesh surface. We report the average point-to-surface distances in centimeters for every subject.

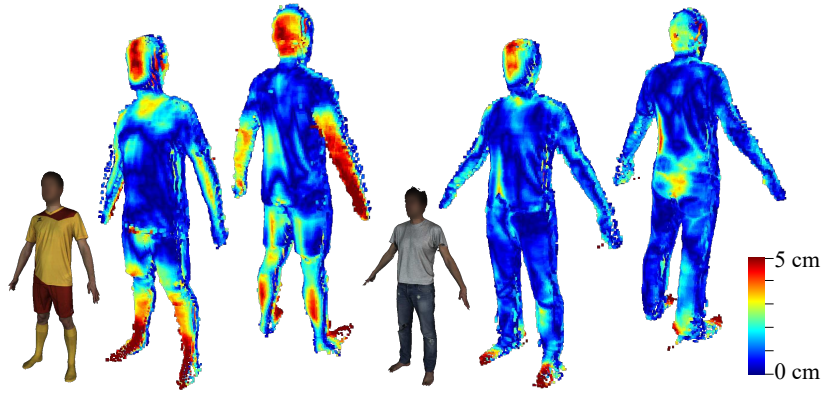



Figure 5.14: Color-map for illustrating the error distribution of our results comparing to the ground truth of BUFF dataset [223]. Each color-map is shown from two different views.

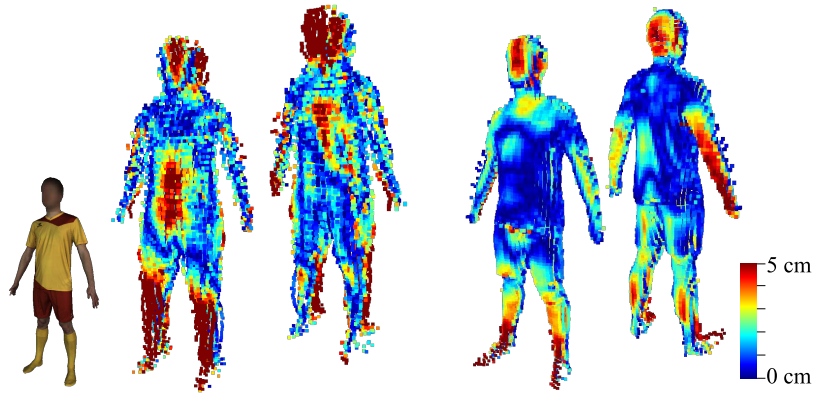



Figure 5.15: Comparison with the view synthesis approach [184].

a human body. The two predicted views are also compatible with each other. However, it can be observed that our result is more smooth compared to theirs. The reason is, firstly, a view synthesis network needs to solve three problems: front view depth estimation, view rotation, and novel view synthesis. In contrast, correspondences between two opposite views are straightforward with dual-depth representation. Our model only needs to learn front-view depth estimation and pixel-wise corresponding back-view synthesis, which are relatively easier to be solved. Secondly, background centering helped to reduce the data variation and improved estimation accuracy. Lastly, the view synthesis network was implemented to accept $128 \times 128$ input images, while ours takes $256 \times 256$ images, which also increases the difference.

### 5.4.6. CONCLUSION AND FUTURE WORK

In this work, a dual-depth representation (DDR) is proposed to enable the reconstruction of a 3D human body from a single RGB image. DDR is flexible to represent detailed geometry, compact to reach a high resolution, and compatible to fit into novel image generation frameworks. Also, it is advisable for the single view reconstruction task as it has pixel-wise correspondence with the input RGB image. On our testing dataset, an average error of 3.2cm (with reference to 176cm as the body height) was achieved. The limitation of DDR is that it can not cover occluded or tangential regions very well. Our experiments show that those regions can be repaired by a view-based refinement network. After mesh reconstruction, the resultant 3D human models are ready for applications such as virtual cloth try-on and augmented reality [148].



One limitation of our approach is its dependency on the pre-processing step for generating 2D keypoints. Recently developed end-to-end frameworks such as [196] can train different modules including 2D keypoint estimation and 3D shape prediction as a whole. Multiple modules are then tuned to collaborate with each other. This is a valuable direction to further investigate for improving our work.

There are several potential directions to further improve our framework. First, it is possible to train the 3D estimation network in an end-to-end manner to further improve the performance of prediction and make the system more user-friendly. Furthermore, we can integrate our geometry prediction approach with the techniques of texture completion (e.g., [117, 166]) to generate fully textured human models from single RGB images. It is also worth investigating the DDR-based 3D shape prediction method for general objects.

Artificial intelligence (AI) and machine learning are developing rapidly in recent years. A lot of papers have been published after the finish of this section's work in March 2019. Two more recent works, PiFU [154] and PiFUHD [155], generate the fully textured human body mesh from single images using an implicit neural representation. The THuman dataset presented in [230] contains approximately 7000 real-world human models, which is much more valuable than the dataset we collected in this work. Our dataset can be augmented with the deformation synthesis method proposed in [157].

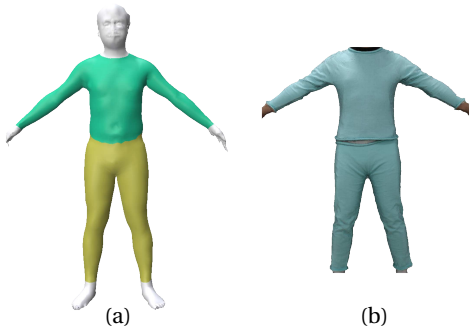
## 5.5. KNITWEAR FOR INDIVIDUALS

In this section, we will apply the 3D and 4D knitwear design methods to scanned human bodies.

### 5.5.1. 3D KNITWEAR

Given a designed knitwear template and a body shape, existing works can customize a 3D knitwear accordingly [182, 76]. Here we present our approach that generates body-tight garments directly from the scanned mesh or generates loose garments (like a dress) by style transfer.

A body-tight 3D garment can be designed by applying our 3D stitch mesh generation methods to the mesh extracted from the scanned 3D body model. After template fitting (Section 5.2.1), regions of the garments are segmented directly (Fig. 5.16(a)). The knitted garments are presented in Fig. 5.16(b).



(a) (b)

Figure 5.16: The scanned body mesh is fitted with the SMPL template, on which segmentation is pre-defined as the garment regions (a). The body-tight shirt and leggings are knitted for these regions (b).

Loose garments are designed by style transferring from a garment design rigged with a template body model (SMPL) to the deformed body template fitted by the scanned body model. Given the garment mesh $\mathcal{M}$ design for a template SMPL body model $\mathcal{S}(\Theta)$ and a deformed body template $\mathcal{S}(\Theta')$ fitted by the scan, we will generate the garment design $\mathcal{M}'$ that fits $\mathcal{S}(\Theta')$. We rig each vertex $\mathbf{v}$ on $\mathcal{M}$ with a triangular face $\mathbf{f}$ on $\mathcal{S}(\Theta)$ and write $\mathbf{v}$ as

$$\mathbf{v} = \mathbf{f}_0 + w_1(\mathbf{f}_1 - \mathbf{f}_0) + w_2(\mathbf{f}_2 - \mathbf{f}_0) + w_3\mathbf{n}, \tag{5.13}$$

in which $\mathbf{f}_i$s are three vertices of $\mathbf{f}$ and $\mathbf{n}$ is the face normal. This decomposition is unique since $\mathbf{f}_1 - \mathbf{f}_0, \mathbf{f}_2 - \mathbf{f}_0$, and $\mathbf{n}$ are linearly independent if $\mathbf{f}$ is not a degenerated face. On the deformed body templated $\mathcal{S}(\Theta')$, which is almost identical to the scan, we have the corresponding face $\mathbf{f}'$, which defines the deformed $\mathbf{v}$ using the same coefficients $w_i$. In practice, we rig vertex $\mathbf{v}$ with multiple faces (the 1-ring faces of $\mathbf{v}$'s closest vertex on $\mathcal{S}(\Theta)$) to make the deformed $\mathcal{M}'$ smoother. This process is referred to as *style transfer* since it regards the garment $\mathcal{M}$ as a stylized body shape $\mathcal{S}(\Theta)$ and transfers the style to $\mathcal{S}(\Theta')$ as a customized garment design $\mathcal{M}'$. The transfer relies on the vertex-wise correspondence between $\mathcal{S}(\Theta)$ and $\mathcal{S}(\Theta')$.

The style transfer method is the same as the yarn model generation method (the deformation transfer method [181] in Section 1.3.3) in essence, which deforms the canonical yarn model rigged with a regular stitch to a deformed stitch. This method can also be used for the training data augmentation for the AI-based human body reconstruction (Section 5.4) by treating the 3D body meshes as garments and deforming them by the fitted body templates.

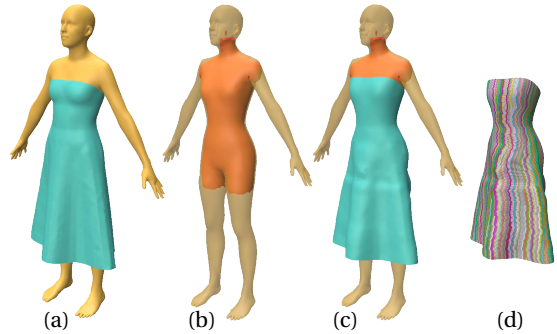



Figure 5.17: Given a dress designed for a template body model (a), we transfer the design to a scanned half-body mannequin (red in (b)) using its fitted template (semi-transparent in (b)). The transferred design (c) fits the target shape of the mannequin while some artifacts appear because of the legs' small deformation between the body templates in (a) and (b). The stitch mesh is colored by its columns (d).

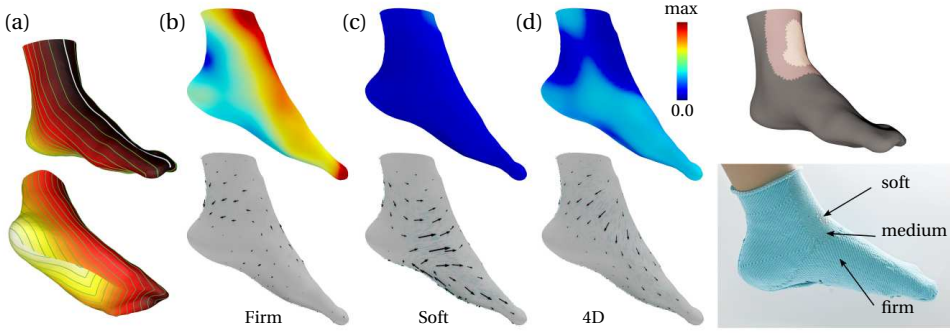



Figure 5.18: Sock with distributed elasticity: (a) geodesic distance field for generating the stitch mesh, (b) simulation of 3D knitting with firm material, (c) simulation of 3D knitting with soft material, (d) resulting 4D design with simulation and physical experiments. Both the stresses and the displacements have been significantly reduced. The maximal stress in this example is 0.6602kPa. In this example, $\bar{d} = 0.8$cm is employed as the terminal threshold of inverse design.

### 5.5.2. 4D KNITWEAR

The 4D knitwear design method proposed in Chapter 4 is applied to the scanned human body.

Fig. 5.18 shows a 4D sock design for a scanned foot. The distributed soft, medium, and firm elasticity is achieved by jacquard patterns. When comparing with the 3D designs using a single elasticity, both the stresses and the amounts of sliding (in terms of displacements) have been significantly reduced. Note that, although the sock shape is homeomorphic to a disk, we still add a cutting line at the back to make it easier to knit. In industrial practice, socks are usually knitted by courses in loops. The tubular knitting design method in Section 3.3.8 can be used to generate such a 4D garment without any cutting lines. The stitch-size compensation and jacquard pattern tiling methods in Chapter 4 should be applied.

## 5.6. CONCLUSION

In this chapter, we reconstruct human body shapes and customize knitwears for reconstructed individuals.

Three human body modeling methods are presented, using different equipment. The multiview system reconstructs the human body using the photogrammetry method, which takes a long time but provides better accuracy compared to the other methods. The dual Kinect system captures two point clouds from two opposite views (front and back of the human body). The calibration method guarantees the alignment between two point clouds. After calibration, the frame rate of capturing and the accuracy are determined by the depth cameras, i.e., the Kinects. The dual depth representation (DDR) helps reconstruct the human body from a single RGB image using AI-based methods. This is a fast and rough estimation, which is suitable for interactive applications (like virtual try-on) but not accurate enough for garment customization.

At last, the 3D and 4D garment design methods developed in the previous chapters are applied to the reconstructed human bodies. By template fitting, we can segment the human body into regions corresponding to garments. We can not only design body-tight garments extracted directly from the scanning, but also transfer an existing design of loose garments (like a dress) to the target human body.

## 5.A. 3D KEYPOINT ESTIMATION FROM MULTIVIEWS



3D body keypoint (especially joint) locations play an important role in the initial estimation of template fitting. Here is the detailed method to estimate 3D keypoints from its multiview projections.

We render the 3D human body from $n$ sampled views. 2D coordinates of keypoints are detected by OpenPose [25] on the views. Each view constrains a keypoint on a 3D line. The 3D locations of keypoints are estimated are minimizing the sum of squared point-to-line distances. This method requires each keypoint to be observed in at least two views out of all $n$ views.

Given a view's projection matrix $(\mathbf{R}|\mathbf{t})$ and a 2D keypoint $(u, v)$ detected on it, a 3D keypoint $\mathbf{x}^*$ should project to $(u, v)$. The projection error can be minimized by bundle adjustment. Differently, we can consider the 3D line on which $\mathbf{x}^*$ should be located. The 3D line is given by $\mathbf{d} \times (\mathbf{x} - \mathbf{a}) = \mathbf{0}$, in which $\mathbf{R}\mathbf{a} = -\mathbf{t}$ and $\mathbf{R}\mathbf{d} = (u, v, 1)^\top$. The line direction $\mathbf{d}$ is normalized for simplicity. Then the point-to-line distance is given by $\|\mathbf{d} \times (\mathbf{x} - \mathbf{a})\|$.

For all $n$ views, we need to find the minimizer

$$\mathbf{x}^* = \arg\min_{\mathbf{x}} \sum_{i=1}^{n} \left\| \mathbf{d}_i \times (\mathbf{x} - \mathbf{a}_i) \right\|^2. \tag{5.14}$$

There are two methods to solve this problem. One is to expand Eq. (5.14) with $\|\mathbf{p} \times \mathbf{q}\|^2 = \|\mathbf{p}\|^2 \|\mathbf{q}\|^2 - (\mathbf{p} \cdot \mathbf{q})^2$ and find the objective's stationary point by setting the gradient as zero. The result is given by a $3 \times 3$ linear system, as shown in the online post [5]. Another method is to solve Eq. (5.14) as a least-squares problem by writing each cross product as the

[5] https://stackoverflow.com/a/48201730

product of a skew-symmetric matrix and a vector, i.e.,

$$\mathbf{d}_i \times \mathbf{x} = [\mathbf{d}_i]_\times \mathbf{x} = \begin{pmatrix} 0 & -\mathbf{d}_{iz} & \mathbf{d}_{iy} \\ \mathbf{d}_{iz} & 0 & -\mathbf{d}_{ix} \\ -\mathbf{d}_{iy} & \mathbf{d}_{ix} & 0 \end{pmatrix} \begin{pmatrix} \mathbf{x}_x \\ \mathbf{x}_y \\ \mathbf{x}_z \end{pmatrix}. \tag{5.15}$$

The least-squares problem is to minimize $\|\mathbf{M}\mathbf{x} - \mathbf{b}\|$, in which

$$\mathbf{M} = \begin{pmatrix} [\mathbf{d}_1]_\times \\ [\mathbf{d}_2]_\times \\ \vdots \\ [\mathbf{d}_n]_\times \end{pmatrix}_{3n\times 3} \quad \text{and} \quad \mathbf{b} = \begin{pmatrix} \mathbf{a}_1 \times \mathbf{d}_1 \\ \mathbf{a}_2 \times \mathbf{d}_2 \\ \vdots \\ \mathbf{a}_n \times \mathbf{d}_n \end{pmatrix}_{3n\times 1}. \tag{5.16}$$

The solution is given by solving $\mathbf{M}^\top \mathbf{M}\mathbf{x} = \mathbf{M}^\top \mathbf{b}$, which is exactly the same linear system as the first method.

# 6

# CONCLUSION

In the final chapter, the contributions of this dissertation are summarized, including what has been done explicitly and how it may help the research community and society. At last, some possible future research directions are proposed.

## 6.1. ANSWERS TO THE RESEARCH QUESTIONS

This dissertation tries to answer the main research question (**RQ**) "How to design customized 3D and 4D knitwear and generate machine instructions?" This main research question is divided into three research questions and answered by corresponding approaches.

The first research question (**RQ1**) is how to design a knitwear and its machine knitting instructions for a given 3D shape. Since the body shape of every person is different, we need to customize knitting patterns for different 3D shapes. Therefore, knitting design methods for general 3D shapes are studied.

A flattening-based method is first proposed (Chapter 2). The target 3D surface is flattened onto the 2D plane to generate the knitting instructions. Darts are cut to reduce the distortion since the target 3D surfaces are usually non-developable. Therefore the guiding question *"where to put darts"* is studied. Darts are opened in regions where the flattening distortion is large. After cutting darts on the 3D mesh, it is flattened again while considering the manufacturing constraints, to generate valid knitting instructions. This approach demonstrates knitting's advantage in closing darts without sewing.

The flattening-based approach relies on user interactions and has difficulty generating complex darts for perfect fitting. To overcome the limitations, direct stitch mesh generation methods are proposed (Chapter 3). On stitch meshes, special stitches that provide intrinsic curvature to the mesh are named *shaping stitches*. Thus a guiding question is *"where to put shaping stitches"*. The proposed methods place shaping stitches on the mesh while considering knittability, distortion, and reliability.

The second research question (**RQ2**) is how to design a motion-aware 4D knitwear

and generate its machine instructions. The method is presented in Chapter 4, which considers the garment's stress and sliding during body motion. We knit jacquard patterns with different elasticity levels on the garment. Basically, the soft pattern reduces stress and the firm pattern reduces sliding. Physical simulations are involved to answer the guiding question *"where to put elastic patterns"*. This approach takes knitting's advantage in combining different structures automatically without cutting pieces from different fabrics and sewing them together.

The third research question (**RQ3**) is how to capture an individual's body models in different poses for 3D/4D knitwear customization. Three methods are presented using different equipment: (1) using the photogrammetry method to reconstruct 3D body shape with a multiview system, (2) reconstructing from point clouds captured by a calibrated dual Kinect setup, and (3) predicting the body shape from a single photo using artificial neural networks. A sequence of 3D body shapes can be registered for 4D design. After obtaining the body shape or sequence, the 3D and 4D knitwear design approach developed in the previous chapters can be used, as shown in Chapter 5.

## 6.2. IMPLICATIONS

This dissertation does not only answer research questions in this area but also has implications for other domains including other research areas, the related industry, and society.

### 6.2.1. IMPLICATIONS FOR KNITTING DESIGN



The presented methods provided new design tools for knitting design. Using these methods, knitwear designers can design perfect-fit garments like leggings and also design body-tight garments with elasticity controlled for body motion. More work will be attracted to realize the 3D and 4D knitwears in the industrial context. The knitwears designed by our methods are more complex compared to common garments, which will decrease the stability and efficiency of manufacturing. Especially, the knitting machine parameters need to be tuned carefully when using rubber yarns for 4D garments, to prevent damage to yarns or needles. Also, the rubber yarns increased the weight of garments, which is not favored by the wearers. A main concern of garment manufacturers is profit. More studies are necessary to improve the design such that more customers would like to pay for it. These factors can be studied in the knitting industry.

Besides garments, our methods can also design other objects like toys and even visualization media (or data physicalization) [171, 128].

This research encourages further development of knitting design software packages. The software should consider supporting stitch mesh for computation and visualization. Several advantages of stitch mesh have been verified. It can show the 3D structure but the 2D knitting map can not. Compared to yarn models, the complex interlocking relation of yarns is categorized as several neighboring relations between stitch types. Then the knittability checking is simplified as checking the compatibility between polygonal facets. The design process is also simplified. It also supports fast simulation of the knitting process and the final result at a coarse level. The 3D and 4D design approaches

can also be provided in the existing design software. The current research-oriented implementations are publicly available online, as a reference for further development. It still requires a lot of software engineering work to convert these prototypes into software packages that are friendly to designers or common users.

### 6.2.2. IMPLICATIONS FOR OTHER MANUFACTURING TECHNIQUES

This dissertation contributes directly to the computer-aided manufacturing (CAM) of knitting. The developed methods may also inspire other digital fabrication like 3D printing. Knitting and 3D printing can be compared since they create objects by adding materials (yarns or filaments), instead of subtracting. In general, the manufacturing process defines the structure, which defines the product's properties (Chapter 2 of [43]). The similarities between manufacturing methods induce the similarities in structures and properties. Some similarities between knitting and 3D printing are listed below.

- Both methods should comply with (semi-)continuity constraints. Breaking the continuity constraints may harm the knitting process, while continuous 3D printing paths benefit the printing.
- The starting procedures of both processes should be carefully treated, i.e., the casting-on of knitting and the initial layers of 3D printing.
- Both methods require extra material to provide the necessary structural support. Knitting waste yarn provides pull-down force for outside widening. Supporting structures in 3D printing help to fabricate overhangs.
- The manufacturing orientation changes the process and the properties of products.
- Elastic structures can be used in both methods ([160] and our 4D knitwear).



These similarities inspire us to generalize the computational design methods from one manufacturing technique to another.

### 6.2.3. IMPLICATIONS FOR OTHER RESEARCH DISCIPLINES

The 3D and 4D knitting design methods presented in this dissertation rely heavily on geometric computation techniques developed in CAD and CG areas. The knitting methods prompt more research in these areas. Especially, several algorithms are used repeatedly. They are fundamental and versatile.

- The remeshing method is used for 3D shape generation (Sec. 2.C.2) and mesh quality improvement.
- The graph-based tessellation method is used for wale stitch mesh generation (both with or without short-columns) and solid shape generation (filling holes or gaps on meshes, Sec. 2.C).
- The Shape-Up deformation method is used for relaxed shape simulation of knitting structures (Sec. 1.3.2), stitch-size compensation (Sec. 4.3.3) in 4D design, and flattening-based design (Sec. 2.3.1).

- The deformation transfer technique is used for yarn model generation (Sec. 1.3.3), style transfer in garment design (Sec. 5.5), and training data augmentation for AI-based body modeling (Sec. 5.4).

Besides geometric computing, physical simulation is also used in this work. It plays an important role in the 4D design approach. The new applications of physical simulation provide more motivation for developing more accurate and efficient simulation tools.

### 6.2.4. IMPLICATIONS FOR SOCIETY

The methods presented in this dissertation can be used in the knitting industry. In the age of industry 4.0, there are increased demands for short development periods, individualization on demand, flexibility, decentralization, and resource efficiency [94]. Our methods for customized design provide more tools to realize these improvements with the help of computer-controlled knitting machines. The stitch mesh in our works can be used to model knitting processes, which is potentially a digital twin of the knitting machine. The yarn models in our works can also be used in the metaverse as a type of visual media.

Our works generalized the existing garment design methods used by designers. Designers fold darts (Fig. 2.1) to produce 3D shapes from planar fabrics. We convert darts to knitting instructions. Complex 'panels' have been used in normal (not luxury) garments for better fitting (Fig. 3.1). We realize better 3D fitting by direct stitch mesh generation. Designers consider body motion by adding gussets (Fig. 4.2) in highly stretched regions, like the region under the arms or between the inner thighs. We generalize to 4D garments by knitting structures of different levels of elasticity in different regions, which is more flexible to control and more automatic to manufacture. These direct generalizations minimized the gaps between the garment industry and academic research.

The knitting design methods developed in this dissertation help fulfill people's need for knitwear customization. The growth of people's needs usually increases the pressures on the planet's natural resources and ecological systems. The knitting technology has the potential to be more sustainable since the yarns may be more recyclable [187, 60].

## 6.3. LIMITATIONS AND FUTURE WORK

The current work has several limitations. There are some interesting directions for further research.

The 3D knitting design aims for perfect fitting, thus the distortion should be minimized. We reduced the distortion in the 3D design of knitting, which uses a diffusion term to generate scattered short-row ends heuristically. A user-specified parameter controls the diffusion. A more systematic method will be valuable, for example, involving relaxed shape simulation in optimization iterations for the stitch mesh design.

The knitting reliability is considered by minimizing the number of transfer stitches. Yarn-level simulations will help us better understand this problem, such as the stretching of yarn when loops are transferred multiple times. Our stitch mesh already encodes the knitting process. Thus the process-aware yarn models have been ready to use. A future research direction is to apply yarn-level physical simulation to them.

Our 4D design approach considered the garment sliding during body motion. Besides movement-related comfort, there are also other factors of garment comfort and ergonomics, including moisture, thermal comfort, and sensorial (related to textile-skin contact) comfort [174, 13]. We may study the role of geometry and material on these comfort factors.

In the current 4D design framework, the elastic patterns are assigned incrementally. Typical structure or meta-material design works use sensitivity analysis, which relies on fast and accurate physical simulations. We may optimize the elastic pattern distribution in this direction, which is more generic.

Knitting has been applied to fabricate different wearable devices [62, 1, 156, 115, 114, 74]. It will be interesting to employ our 3D and 4D design methods in those applications. For example, the distortion of 3D shaping should influence the performance of the knitted sensors. Quantitative evaluations of the distortion's effect will justify the value of our work on distortion control in 3D knitting.

Material choice has a significant influence on manufacturing. Knitting is no exception. In this research, we used yarns made of cotton, nylon, or rubber. Other materials may enrich the applications of knitting. For example, conductive yarns can be knitted as wearable sensors. Natural yarns (like jute) can produce eco-friendly fabrics. Knitting stiff yarns are not as simple as flexible yarns. One should be careful with the damage to yarns or machine needles. The knitted preforms can be reinforced by inlaid carbon fibers, using a modified mechanized knitting machine [70].

Our design methods can be generalized to other textile manufacturing techniques. The current focus is weft knitting. It will be interesting to study the method's application on warp knitting. 3D shapes knitted with our methods can also be reinforced with inlay, as just mentioned, the results of which are named biaxial weft knitted fabrics [70] or co-weaving-knitting fabrics [28]. In the literature, the 3D textile sometimes refers to a solid 3D shape composed of multi-layers [70], instead of the single-layer 3D surface in this work. We can study combining our approach with these methods.

New research opportunities will evolve along with the advance of knitting machines designed for either large companies or individuals. There will be plentiful applications of knitting due to the improved accessibility of knitting machines [153, 86], regarding the explosive growth of 3D printing thanks to the expansion of 3D printers in the last decade.

The experience of technicians from the knitting industry will also inspire new research. In the communications with technicians, I learned that there are different designs for sleeves and shoulders (e.g., raglan sleeve and set-in sleeve) with different appearances. Meanwhile, their knitting stability and efficiency are also different. It is worth studying these practical experiences with quantitative research methods. The research will also improve technicians' understanding and effectiveness of their work.

# ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my promotors Prof. dr. Charlie C.L. Wang, Prof. dr. ir. Jo M.P. Geraedts, and my copromotor Dr. ir. Eugeni L. Doubrovski. I could not have undertaken this journey without your supervision. Special thanks to Dr. Doubrovski for translating the dissertation's summary and the propositions into Dutch.

I am also grateful to all the committee members for their work on my dissertation and defense.

During my PhD study, Prof. Yu-Kun Lai, Prof. YAM Yeung, Dr. Jun Wu, and Prof. Emily Whiting gave me a lot of help. Words cannot express my gratitude to them.

I would like to acknowledge the financial support from the China Scholarship Council and the Centre for Perceptual and Interactive Intelligence (CPII).

Thanks should go to all of the collaborators on the knitting project. Yuchen helped me to clarify numerous technical issues. She dedicated a lot to this project, including operating the knitting machines, sewing the knitted pieces, and preparing the figures, especially rendering the yarn models. Xingjian played an important and irreplaceable role in the physical simulation part of the work. Also, thanks to Singa for rendering the simulation results, preparing a video, and a few other works. I would like to extend my sincere thanks to the models, who would like to remain anonymous, for their contribution to the body scanning and knitwear try-on.

Many thanks to Kelvin for working together with me on knitting. Thanks to Mr. LO Kai Hung for training me in the operation of the Shima Seiki's knitting machine and software. He is a good teacher who allows me to learn from my mistakes.

I would like to acknowledge Prof. Conny Bakker, Dr. Holly McQuillan, and Dr. Jeremy Faludi for their sharing on sustainability assessment.

At Delft, I met many nice friends, who made the days not so lonely. I would like to thank Jian Fang, Tiantian Du, Yanze Yang, Haiyang Huang, and Jun's family. I am thankful to Rob, Wolf, Yabin, Tianyun, Tim, Richardo, and Chengkai. Thank you Bahar, Annemarijn, Bruna, Federico, Roya, and Thomas. I miss our paintings on the whiteboards in our office. I would also like to acknowledge our lab/department supporting staff for their wholehearted work. Kasper, Tom and Tom, Joy, and Nina, thanks for your work on our HeartBits project. Thanks to Rik for coordinating our course.

Many thanks to my friends at Cardiff University: Pengfei Guo, Hongjin Lyu, Zhongyu Jiang, and Qian Xie. Special thanks to Roberto and Yipeng for their knowledge sharing.

I would like to express my appreciation to Dr. Zhan Song, Yuping Ye, Lijun Shu. Xining Cui, and Ang Cai for their help with the scanning. It was a joyful experience at SIAT.

Thanks to my colleagues at CPII: Lip, Andy, Issac, and Wenting. I learned a lot from you guys. Also thanks to the administration team for their support.

Xiao, thank you for the sunshine you bring into my life. Thank you for understanding me so well. You have never tried to alter me, but you helped me make myself a better person. Your love is so powerful and supported me to rebuild my life. Thanks to my

family for your constant love, support, and all you have sacrificed for me. I am deeply indebted to all of you.

Last but not least, I would like to express my deepest appreciation to all the people who shared their software or data. The publicly available software (especially source code) and data fill the gap and lower the barrier to research. Their scholarly contribution should not be overlooked [57].

# BIBLIOGRAPHY


[1] Lea Albaugh, Scott Hudson, and Lining Yao. Digital fabrication of soft actuated objects by machine knitting. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, CHI'19, New York, NY, USA, 2019. ACM. ISBN 9781450359702. doi:10.1145/3290605.3300414.

[2] Marc Alexa, Daniel Cohen-Or, and David Levin. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 157–164, USA, 2000. ACM Press/Addison-Wesley Publishing Co. ISBN 1581132085. doi:10.1145/344779.344859.

[3] Thiemo Alldieck, Marcus Magnor, Weipeng Xu, Christian Theobalt, and Gerard Pons-Moll. Video based reconstruction of 3D people models. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8387–8397, 2018. doi:10.1109/CVPR.2018.00875.

[4] Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. SCAPE: Shape completion and animation of people. *ACM Trans. Graph.*, 24(3):408–416, July 2005. ISSN 0730-0301. doi:10.1145/1073204.1073207.

[5] Kin-Fan Au, editor. *Advances in Knitting Technology*. Woodhead Publishing Series in Textiles. Woodhead Publishing, 2011. ISBN 978-1-84569-372-5. doi:10.1533/9780857090621.

[6] Autodesk Inc. Slicer for Fusion 360. `https://knowledge.autodesk.com/support/fusion-360/troubleshooting/caas/downloads/content/slicer-for-fusion-360.html`, 2020. [Online; accessed 1-Aug.-2021].

[7] Alexandru O. Balan, Leonid Sigal, Michael J. Black, James E. Davis, and Horst W. Haussecker. Detailed human shape and pose from images. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. doi:10.1109/CVPR.2007.383340.

[8] David Baraff and Andrew Witkin. Large steps in cloth simulation. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 43–54, New York, NY, USA, 1998. ACM. ISBN 0897919998. doi:10.1145/280814.280821.

[9] Aric Bartle, Alla Sheffer, Vladimir G. Kim, Danny M. Kaufman, Nicholas Vining, and Floraine Berthouzoz. Physics-driven pattern adjustment for direct 3D garment editing. *ACM Trans. Graph.*, 35(4), July 2016. ISSN 0730-0301. doi:10.1145/2897824.2925896.

[10] Pierre Bénard and Aaron Hertzmann. Line drawings from 3D models: A tutorial. *Foundations and Trends in Computer Graphics and Vision*, 11(1-2):1–159, 2019. ISSN 1572-2740. doi:10.1561/0600000075.

[11] Floraine Berthouzoz, Akash Garg, Danny M. Kaufman, Eitan Grinspun, and Maneesh Agrawala. Parsing sewing patterns into 3D garments. *ACM Trans. Graph.*, 32(4), July 2013. ISSN 0730-0301. doi:10.1145/2461912.2461975.

[12] Bernd Bickel, Moritz Bächer, Miguel A. Otaduy, Hyunho Richard Lee, Hanspeter Pfister, Markus Gross, and Wojciech Matusik. Design and fabrication of materials with desired deformation behavior. *ACM Trans. Graph.*, 29(4), July 2010. ISSN 0730-0301. doi:10.1145/1778765.1778800.

[13] Kim B. Blair. Materials and design for sports apparel. In Aleksandar Subic, editor, *Materials in Sports Equipment*, volume 2, pages 60–86. Woodhead Publishing, 2007. ISBN 978-1-84569-131-8. doi:10.1533/9781845693664.1.60.

[14] Blender Online Community. *Blender – a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2021. URL https://www.blender.org.

[15] Federica Bogo, Angjoo Kanazawa, Christoph Lassner, Peter Gehler, Javier Romero, and Michael J. Black. Keep it SMPL: Automatic estimation of 3D human pose and shape from a single image. In *Computer Vision – ECCV 2016*, pages 561–578, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46454-1. doi:10.1007/978-3-319-46454-1_34.

[16] David Bommes and Leif Kobbelt. Accurate computation of geodesic distance fields for polygonal curves on triangle meshes. In *Proceedings of the Vision, Modeling, and Visualization Conference 2007*, VMV 2007, pages 151–160. Aka GmbH, 2007. ISBN 978-3-89838-085-0.

[17] David Bommes, Bruno Lévy, Nico Pietroni, Enrico Puppo, Claudio Silva, Marco Tarini, and Denis Zorin. Quad-mesh generation and processing: A survey. *Computer Graphics Forum*, 2013. ISSN 1467-8659. doi:10.1111/cgf.12014.

[18] Mario Botsch, Simon Steinberg, Stephan Bischoff, and Leif Kobbelt. OpenMesh – a generic and efficient polygon mesh data structure, 2002. 1st OpenSG Symposium.

[19] Mario Botsch, Leif Kobbelt, Mark Pauly, Pierre Alliez, and Bruno Lévy. *Polygon Mesh Processing*. A K Peters, Ltd., September 2010. ISBN 978-1-56881-426-1.

[20] Sofien Bouaziz, Mario Deuss, Yuliy Schwartzburg, Thibaut Weise, and Mark Pauly. Shape-Up: Shaping discrete geometry with projections. *Computer Graphics Forum*, 31(5):1657–1667, 2012. doi:10.1111/j.1467-8659.2012.03171.x.

[21] JC Briar. Stitch Maps. https://stitch-maps.com/, 2012. [Online; accessed 1-Jan.-2022].

[22] Robert Bridson, Ronald Fedkiw, and John Anderson. Robust treatment of collisions, contact and friction for cloth animation. *ACM Trans. Graph.*, 21(3):594–603, July 2002. ISSN 0730-0301. doi:10.1145/566654.566623.

[23] Brother Industries, Ltd. *How to Use Your Knitting Machine (KH868)*. Japan, 1990.

[24] Remi Brouet, Alla Sheffer, Laurence Boissieux, and Marie-Paule Cani. Design preserving garment transfer. *ACM Trans. Graph.*, 31(4), July 2012. ISSN 0730-0301. doi:10.1145/2185520.2185532.

[25] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2D pose estimation using part affinity fields. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1302–1310, 2017. doi:10.1109/CVPR.2017.143.

[26] Carnegie Mellon Textiles Lab. The "knitout" (.k) file format v0.5.4. `https://textiles-lab.github.io/knitout/`, 2020. [Online; accessed 1-Apr.-2022].

[27] Carlos Castillo, Jorge López-Moreno, and Carlos Aliaga. Recent advances in fabric appearance reproduction. *Computers & Graphics*, 84:103–121, 2019. ISSN 0097-8493. doi:10.1016/j.cag.2019.07.007.

[28] Hsin-Chuan Chen, Kuei-Chi Lee, and Jia-Horng Lin. Electromagnetic and electrostatic shielding properties of co-weaving-knitting fabrics reinforced composites. *Composites Part A: Applied Science and Manufacturing*, 35(11):1249–1256, 2004. ISSN 1359-835X. doi:10.1016/j.compositesa.2004.04.006.

[29] Xiang Chen, Changxi Zheng, Weiwei Xu, and Kun Zhou. An asymptotic numerical method for inverse elastic shape design. *ACM Trans. Graph.*, 33(4), July 2014. ISSN 0730-0301. doi:10.1145/2601097.2601189.

[30] Yanyun Chen, Stephen Lin, Hua Zhong, Ying-Qing Xu, Baining Guo, and Heung-Yeung Shum. Realistic rendering and animation of knitwear. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):43–55, 2003. doi:10.1109/TVCG.2003.1175096.

[31] Dan Cheng and **Zishun Liu**. Spatio-temporal prediction of missing temperature with stochastic Poisson equations. *Extremes*, 24(1):163–175, 2021. ISSN 1572-915X. doi:10.1007/s10687-020-00397-w.

[32] Gianmarco Cherchi, Marco Livesu, Riccardo Scateni, and Marco Attene. Fast and robust mesh arrangements using floating-point arithmetic. *ACM Trans. Graph.*, 39 (6), November 2020. ISSN 0730-0301. doi:10.1145/3414685.3417818.

[33] Sungjoon Choi, Qian-Yi Zhou, and Vladlen Koltun. Robust reconstruction of indoor scenes. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5556–5565, 2015. doi:10.1109/CVPR.2015.7299195.

[34] Paolo Cignoni, Marco Callieri, Massimiliano Corsini, Matteo Dellepiane, Fabio Ganovelli, and Guido Ranzuglia. MeshLab: an open-source mesh processing tool. In *Eurographics Italian Chapter Conference*. The Eurographics Association, 2008. ISBN 978-3-905673-68-5. doi:10.2312/LocalChapterEvents/ItalChap/ItalianChapConf2008/129-136.

[35] Keenan Crane, Marco Livesu, Enrico Puppo, and Yipeng Qin. A survey of algorithms for geodesic paths and distances. *arXiv*, 2020. doi:10.48550/arXiv.2007.10430.

[36] Tran Kai Frank Da, Sébastien Loriot, and Mariette Yvinec. 3D alpha shapes. In *CGAL User and Reference Manual*. CGAL Editorial Board, 5.3 edition, 2021. URL https://doc.cgal.org/5.3/Manual/packages.html#PkgAlphaShapes3.

[37] Dawson-Haggerty et al. trimesh. https://trimsh.org/, 2019. [v3.2.0].

[38] Fernando de Goes, Mathieu Desbrun, and Yiying Tong. Vector field processing on triangle meshes. In *ACM SIGGRAPH 2016 Courses*, SIGGRAPH '16, New York, NY, USA, 2016. Association for Computing Machinery. ISBN 9781450342896. doi:10.1145/2897826.2927303.

[39] Philippe Decaudin, Dan Julius, Jamie Wither, Laurence Boissieux, Alla Sheffer, and Marie-Paule Cani. Virtual garments: A fully geometric approach for clothing design. *Computer Graphics Forum*, 25(3):625–634, 2006. doi:10.1111/j.1467-8659.2006.00982.x.

[40] Mario Deuss, Anders Holden Deleuran, Sofien Bouaziz, Bailin Deng, Daniel Piker, and Mark Pauly. ShapeOp – a robust and extensible geometric modelling paradigm. In *Modelling Behaviour: Design Modelling Symposium 2015*, pages 505–515. Springer International Publishing, Cham, 2015. ISBN 978-3-319-24208-8. doi:10.1007/978-3-319-24208-8_42.

[41] Manfredo P. do Carmo. *Riemannian Geometry*. Birkhäuser, Boston, MA, 1 edition, 1992. ISBN 978-0-8176-3490-2.

[42] Manfredo P. do Carmo. *Differential Geometry of Curves and Surfaces: Revised & Updated Second Edition*. Dover Publications, Inc., Mineola, NY, 2016. ISBN 978-0-486-80699-0.

[43] Eugeni L. Doubrovski. *Design Methodology for Additive Manufacturing: Supporting Designers in the Exploitation of Additive Manufacturing Affordances*. PhD thesis, Delft University of Technology, 2016. URL https://doi.org/10.4233/uuid:d4214bb0-5bfd-43fe-af42-01247762b661.

[44] Stefan Droste, Thomas Jansen, and Ingo Wegener. On the analysis of the (1+1) evolutionary algorithm. *Theoretical Computer Science*, 276(1):51–81, 2002. ISSN 0304-3975. doi:10.1016/S0304-3975(01)00182-7.

[45] Elmar Eisemann, Holger Winnemöller, John C. Hart, and David Salesin. Stylized vector art from 3D models with region support. *Computer Graphics Forum*, 27(4): 1199–1207, 2008. doi:10.1111/j.1467-8659.2008.01258.x.

[46] Elmar Eisemann, Sylvain Paris, and Frédo Durand. A visibility algorithm for converting 3D meshes into editable 2D vector graphics. *ACM Trans. Graph.*, 28(3), July 2009. ISSN 0730-0301. doi:10.1145/1531326.1531389.

[47] David Eriksson, Michael Pearce, Jacob R Gardner, Ryan Turner, and Matthias Poloczek. Scalable global optimization via local Bayesian optimization. In *Advances in Neural Information Processing Systems*, volume 32, Red Hook, NY, USA, 2019. Curran Associates Inc.

[48] Haoqiang Fan, Hao Su, and Leonidas Guibas. A point set generation network for 3D object reconstruction from a single image. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2463–2471, 2017. doi:10.1109/CVPR.2017.264.

[49] Michael S. Floater. Mean value coordinates. *Computer Aided Geometric Design*, 20 (1):19–27, 2003. ISSN 0167-8396. doi:10.1016/S0167-8396(03)00002-5.

[50] Henry Ford and Samuel Crowther. *My Life and Work*. Garden City Publishing Company, 1922. ISBN 1514186160.

[51] Henry Fuchs, Zvi Meir Kedem, and Samuel Parker Uselton. Optimal surface reconstruction from planar contours. *Commun. ACM*, 20(10):693–702, October 1977. ISSN 0001-0782. doi:10.1145/359842.359846.

[52] Rony Goldenthal, David Harmon, Raanan Fattal, Michel Bercovier, and Eitan Grinspun. Efficient simulation of inextensible cloth. *ACM Trans. Graph.*, 26(3):49–es, July 2007. ISSN 0730-0301. doi:10.1145/1276377.1276438.

[53] Leslie Gonzalez. Reading and understanding charts. *Cast On*, August–October: 64–68, 2011.

[54] Peng Guan, Alexander Weiss, Alexandru O. Bālan, and Michael J. Black. Estimating human shape and pose from a single image. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1381–1388, 2009. doi:10.1109/ICCV.2009.5459300.

[55] Runbo Guo, Jenny Lin, Vidya Narayanan, and James McCann. Representing crochet with stitch meshes. In *Symposium on Computational Fabrication*, SCF '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450381703. doi:10.1145/3424630.3425409.

[56] Ruslan Guseinov, Eder Miguel, and Bernd Bickel. CurveUps: Shaping objects from flat plates with tension-actuated curvature. *ACM Trans. Graph.*, 36(4), July 2017. ISSN 0730-0301. doi:10.1145/3072959.3073709.

[57] Lou Hafer and Arthur E. Kirkpatrick. Assessing open source software as a scholarly contribution. *Commun. ACM*, 52(12):126–129, December 2009. ISSN 0001-0782. doi:10.1145/1610252.1610285.

[58] Vikki Haffenden and Frederica Patmore. *The Knitting Book*. Dorling Kindersley Limited, 2019. ISBN 978-1-4564-8240-2.

[59] Richard I. Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2nd edition, 2004. ISBN 0521540518.

[60] HKRITA. The garment-to-garment (G2G) recycle system. https://www.hkrita.com/en/garment2garment, 2018. [Online; accessed 1-Apr.-2022].

[61] Megan Hofmann, Lea Albaugh, Ticha Sethapakadi, Jessica Hodgins, Scott E. Hudson, James McCann, and Jennifer Mankoff. 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*, UIST '19, pages 5–16, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450368162. doi:10.1145/3332165.3347886.

[62] Megan Hofmann, Jennifer Mankoff, and Scott E. Hudson. 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*, UIST '20, pages 1234–1247, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450375146. doi:10.1145/3379337.3415590.

[63] Benjamin Holzschuh, Zorah Lähner, and Daniel Cremers. Simulated annealing for 3D shape correspondence. In *2020 International Conference on 3D Vision (3DV)*, pages 252–260, 2020. doi:10.1109/3DV50981.2020.00035.

[64] Takeo Igarashi, Tomer Moscovich, and John F. Hughes. As-rigid-as-possible shape manipulation. *ACM Trans. Graph.*, 24(3):1134–1141, July 2005. ISSN 0730-0301. doi:10.1145/1073204.1073323.

[65] Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. Knitting a 3D model. *Computer Graphics Forum*, 27(7):1737–1743, October 2008. doi:10.1111/j.1467-8659.2008.01318.x.

[66] Yuki Igarashi, Takeo Igarashi, and Hiromasa Suzuki. Knitty: 3D modeling of knitted animals with a production assistant interface. In *Eurographics 2008 - Short Papers*. The Eurographics Association, 2008. doi:10.2312/egs.20081011.

[67] Inkscape Project. Inkscape: Open source scalable vector graphics editor. https://inkscape.org/, 2020. [v0.92.5].

[68] Alexandra Ion, Michael Rabinovich, Philipp Herholz, and Olga Sorkine-Hornung. Shape approximation by developable wrapping. *ACM Trans. Graph.*, 39(6), November 2020. ISSN 0730-0301. doi:10.1145/3414685.3417835.

[69] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1325–1339, 2014. doi:10.1109/TPAMI.2013.248.

[70] Natalie R. Ishmael. *Manufacturing and mechanical characterisation of 3D biaxial weft knitted preforms for composites*. PhD thesis, The University of Manchester, 2009.

[71] ISO. Textiles – Knitted fabrics – Representation and pattern design. Standard, International Organization for Standardization, Geneva, Switzerland, March 2009.

[72] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5967–5976, 2017. doi:10.1109/CVPR.2017.632.

[73] Alec Jacobson, Daniele Panozzo, et al. libigl: A simple C++ geometry processing library. https://libigl.github.io/, 2018.

[74] Kaspar M.B. Jansen. Performance evaluation of knitted and stitched textile strain sensors. *Sensors*, 20(24), 2020. ISSN 1424-8220. doi:10.3390/s20247236.

[75] Li Jiang, Shaoshuai Shi, Xiaojuan Qi, and Jiaya Jia. GAL: Geometric adversarial loss for single-view 3D-object reconstruction. In *Computer Vision – ECCV 2018*, pages 820–834, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01237-3. doi:10.1007/978-3-030-01237-3_49.

[76] Benjamin Jones, Yuxuan Mei, Haisen Zhao, Taylor Gotfrid, Jennifer Mankoff, and Adriana Schulz. Computational design of knit templates. *ACM Trans. Graph.*, 41 (2), December 2021. ISSN 0730-0301. doi:10.1145/3488006.

[77] Jonathan M. Kaldor, Doug L. James, and Steve Marschner. Simulating knitted cloth at the yarn level. *ACM Trans. Graph.*, 27(3):1–9, August 2008. ISSN 0730-0301. doi:10.1145/1360612.1360664.

[78] Angjoo Kanazawa, Michael J. Black, David W. Jacobs, and Jitendra Malik. End-to-end recovery of human shape and pose. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7122–7131, 2018. doi:10.1109/CVPR.2018.00744.

[79] Levi Kapllani, Chelsea Amanatides, Genevieve Dion, Vadim Shapiro, and David E. Breen. TopoKnit: A process-oriented representation for modeling the topology of yarns in weft-knitted textiles. *Graphical Models*, page 101114, 2021. ISSN 1524-0703. doi:10.1016/j.gmod.2021.101114.

[80] Ayelet Karmon, Yoav Sterman, Tom Shaked, Eyal Sheffer, and Shoval Nir. KNITIT: A computational tool for design, simulation, and fabrication of multiple structured knits. In *Proceedings of the 2nd ACM Symposium on Computational Fabrication*,

SCF '18, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450358545. doi:10.1145/3213512.3213516.

[81] Alexandre Kaspar. *Garment Design Workflows for On-Demand Machine Knitting*. PhD thesis, Massachusetts Institute of Technology, 2022.

[82] Alexandre Kaspar, Liane Makatura, and Wojciech Matusik. Knitting skeletons: A computer-aided design tool for shaping and patterning of knitted garments. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, UIST '19, pages 53–65, New York, NY, USA, 2019. ACM. ISBN 9781450368162. doi:10.1145/3332165.3347879.

[83] Alexandre Kaspar, Tae-Hyun Oh, Liane Makatura, Petr Kellnhofer, and Wojciech Matusik. Neural inverse knitting: From images to manufacturing instructions. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3272–3281, Long Beach, California, USA, 09–15 Jun 2019. PMLR. URL http://proceedings.mlr.press/v97/kaspar19a.html.

[84] Alexandre Kaspar, Kui Wu, Yiyue Luo, Liane Makatura, and Wojciech Matusik. Knit sketching: From cut & sew patterns to machine-knit garments. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi:10.1145/3450626.3459752.

[85] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 61–70, Goslar, DEU, 2006. Eurographics Association. ISBN 3905673363. doi:10.2312/SGP/SGP06/061-070.

[86] Kniterate. Kniterate: The digital knitting machine. https://www.kniterate.com, 2015. [Online; accessed 1-Apr.-2022].

[87] Chelsea E. Knittel, Diana S. Nicholas, Reva M. Street, Caroline L. Schauer, and Genevieve Dion. Self-folding textiles through manipulation of knit stitch architecture. *Fibers*, 3(4):575–587, 2015. ISSN 2079-6439. doi:10.3390/fib3040575.

[88] Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. Globally optimal direction fields. *ACM Trans. Graph.*, 32(4), July 2013. ISSN 0730-0301. doi:10.1145/2461912.2462005.

[89] Felix Knöppel, Keenan Crane, Ulrich Pinkall, and Peter Schröder. Stripe patterns on surfaces. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi:10.1145/2767000.

[90] Tsz-Ho Kwok, Kwok-Yun Yeung, and Charlie C. L. Wang. Volumetric template fitting for human body reconstruction from incomplete data. *Journal of Manufacturing Systems*, 33(4):678–689, 2014. ISSN 0278-6125. doi:10.1016/j.jmsy.2014.05.009.

[91] Tsz-Ho Kwok, Yan-Qiu Zhang, Charlie C. L. Wang, Yong-Jin Liu, and Kai Tang. Styling evolution for tight-fitting garments. *IEEE Transactions on Visualization and Computer Graphics*, 22(5):1580–1591, 2016. doi:10.1109/TVCG.2015.2446472.

[92] Yu-Kun Lai, Miao Jin, Xuexiang Xie, Ying He, Jonathan Palacios, Eugene Zhang, Shi-Min Hu, and Xianfeng Gu. Metric-driven RoSy field design and remeshing. *IEEE Transactions on Visualization and Computer Graphics*, 16(1):95–108, 2010. doi:10.1109/TVCG.2009.59.

[93] Jonas Larsson, Joel Peterson, and Heikki Mattila. The knit on demand supply chain. *Autex Research Journal*, 12(3):67–75, 2012. doi:10.2478/v10304-012-0013-9.

[94] Heiner Lasi, Peter Fettke, Hans-Georg Kemper, Thomas Feld, and Michael Hoffmann. Industry 4.0. *Business & Information Systems Engineering*, 6(4):239–242, August 2014. ISSN 1867-0202. doi:10.1007/s12599-014-0334-4.

[95] Jonathan Leaf, Rundong Wu, Eston Schweickart, Doug L. James, and Steve Marschner. Interactive design of periodic yarn-level cloth patterns. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi:10.1145/3272127.3275105.

[96] Yann A. LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient BackProp. In *Neural Networks: Tricks of the Trade*. Springer, 2nd edition, 2012. ISBN 978-3-642-35289-8. doi:10.1007/978-3-642-35289-8_3.

[97] Bruno Lévy and Hao (Richard) Zhang. Spectral mesh processing. In *ACM SIGGRAPH 2010 Courses*, SIGGRAPH '10, New York, NY, USA, 2010. Association for Computing Machinery. ISBN 9781450303958. doi:10.1145/1837101.1837109.

[98] Jie Li, Gilles Daviet, Rahul Narain, Florence Bertails-Descoubes, Matthew Overby, George E. Brown, and Laurence Boissieux. An implicit frictional contact solver for adaptive cloth simulation. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi:10.1145/3197517.3201308.

[99] Minchen Li, Danny M. Kaufman, Vladimir G. Kim, Justin Solomon, and Alla Sheffer. OptCuts: Joint optimization of surface cuts and parameterization. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi:10.1145/3272127.3275042.

[100] Minchen Li, Alla Sheffer, Eitan Grinspun, and Nicholas Vining. FoldSketch: Enriching garments with physically reproducible folds. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi:10.1145/3197517.3201310.

[101] Wenhui Li, Anan Liu, Weizhi Nie, Dan Song, Yuqian Li, Weijie Wang, Shu Xiang, Heyu Zhou, Ngoc-Minh Bui, Yunchi Cen, Zenian Chen, Huy-Hoang Chung-Nguyen, Gia-Han Diep, Trong-Le Do, Eugeni L. Doubrovski, Anh-Duc Duong, Jo M. P. Geraedts, Haobin Guo, Trung-Hieu Hoang, Yichen Li, Xing Liu, **Zishun Liu**, Duc-Tuan Luu, Yunsheng Ma, Vinh-Tiep Nguyen, Jie Nie, Tongwei Ren, Mai-Khiem Tran, Son-Thanh Tran-Nguyen, Minh-Triet Tran, The-Anh Vu-Le, Charlie C. L. Wang, Shijie Wang, Gangshan Wu, Caifei Yang, Meng Yuan, Hao Zhai, Ao Zhang, Fan Zhang, and Sicheng Zhao. SHREC 2019 - monocular image based 3D model retrieval. In *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2019. ISBN 978-3-03868-077-2. doi:10.2312/3dor.20191068.

[102] Jenny Lin, Vidya Narayanan, and James McCann. Efficient transfer planning for flat knitting. In *Proceedings of the 2nd ACM Symposium on Computational Fabrication*, SCF '18, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450358545. doi:10.1145/3213512.3213515.

[103] Shengjun Liu and Charlie C. L. Wang. Orienting unorganized points for surface reconstruction. *Computers & Graphics*, 34(3):209–218, 2010. ISSN 0097-8493. doi:10.1016/j.cag.2010.03.003. Shape Modelling International (SMI) Conference 2010.

[104] Yige Liu, Li Li, and Philip F. Yuan. A computational approach for knitting 3D composites preforms. In *Proceedings of the 2019 DigitalFUTURES*, pages 232–246, Singapore, 2020. Springer Singapore. ISBN 978-981-13-8153-9. doi:10.1007/978-981-13-8153-9_21.

[105] Ziwei Liu, Ping Luo, Shi Qiu, Xiaogang Wang, and Xiaoou Tang. DeepFashion: Powering robust clothes recognition and retrieval with rich annotations. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1096–1104, 2016. doi:10.1109/CVPR.2016.124.

[106] **Zishun Liu**. MeshUtility: A collection of python utilities for mesh processing. `https://github.com/zishun/MeshUtility`, 2021. [Online; accessed 1-Oct.-2021].

[107] **Zishun Liu**, Zhenxi Li, Juyong Zhang, and Ligang Liu. Euclidean and Hamming embedding for image patch description with convolutional networks. In *2016 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1145–1151, Los Alamitos, CA, USA, July 2016. IEEE Computer Society. doi:10.1109/CVPRW.2016.146.

[108] **Zishun Liu**, Juyong Zhang, and Ligang Liu. Upright orientation of 3D shapes with convolutional networks. *Graphical Models*, 85:22–29, 2016. ISSN 1524-0703. doi:10.1016/j.gmod.2016.03.001. SI\: CVM 2016 selected Papers.

[109] **Zishun Liu**, Xingjian Han, Yuchen Zhang, Xiangjia Chen, Yu-Kun Lai, Eugeni L. Doubrovski, Emily Whiting, and Charlie C. L. Wang. Knitting 4D garments with elasticity controlled for body motion. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi:10.1145/3450626.3459868.

[110] Matthew Loper, Naureen Mahmood, and Michael J. Black. MoSh: Motion and shape capture from sparse markers. *ACM Trans. Graph.*, 33(6), November 2014. ISSN 0730-0301. doi:10.1145/2661229.2661273.

[111] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graph.*, 34(6), October 2015. ISSN 0730-0301. doi:10.1145/2816795.2818013.

[112] Diego Lopez de Ipinã, Paulo R. S. Mendonça, and Andy Hopper. TRIP: A low-cost vision-based location system for ubiquitous computing. *Personal and Ubiquitous Computing*, 6:206–219, 2002. ISSN 1617-4909. doi:10.1007/s007790200020.

[113] Fujun Luan, Sylvain Paris, Eli Shechtman, and Kavita Bala. Deep photo style transfer. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6997–7005, 2017. doi:10.1109/CVPR.2017.740.

[114] Yiyue Luo, Yunzhu Li, Pratyusha Sharma, Wan Shou, Kui Wu, Michael Foshey, Beichen Li, Tomás Palacios, Antonio Torralba, and Wojciech Matusik. Learning human-environment interactions using conformal tactile textiles. *Nature Electronics*, 4(3):193–201, March 2021. ISSN 2520-1131. doi:10.1038/s41928-021-00558-0.

[115] Yiyue Luo, Kui Wu, Tomás Palacios, and Wojciech Matusik. KnitUI: Fabricating interactive and sensing textiles with machine knitting. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA, 2021. Association for Computing Machinery. ISBN 9781450380966. doi:10.1145/3411764.3445780.

[116] Mickaël Ly, Romain Casati, Florence Bertails-Descoubes, Mélina Skouras, and Laurence Boissieux. Inverse elastic shell design with contact and friction. *ACM Trans. Graph.*, 37(6), December 2018. ISSN 0730-0301. doi:10.1145/3272127.3275036.

[117] Liqian Ma, Xu Jia, Qianru Sun, Bernt Schiele, Tinne Tuytelaars, and Luc Van Gool. Pose guided person image generation. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[118] Subhankar Maity, Sohel Rana, Pintu Pandit, and Kunal Singha, editors. *Advanced Knitting Technology*. The Textile Institute Book Series. Woodhead Publishing, 2022. ISBN 978-0-323-85534-1. doi:10.1016/C2020-0-02301-7.

[119] Steve Marschner and Peter Shirley. *Fundamentals of Computer Graphics*. CRC Press, 4th edition, 2016. ISBN 978-1-4822-2941-7.

[120] Carmine Mazza and Paola Zonda. *Reference Book of Textile Technologies - Knitting*. Association of Italian Textile Machinery Manufacturers (ACIMIT), Milan, Italy, 2nd edition, 2002.

[121] James McCann, Lea Albaugh, Vidya Narayanan, April Grow, Wojciech Matusik, Jennifer Mankoff, and Jessica Hodgins. A compiler for 3D machine knitting. *ACM Trans. Graph.*, 35(4), July 2016. ISSN 0730-0301. doi:10.1145/2897824.2925940.

[122] Peter McCleary. Robert Le Ricolais, ou l'art de la fuite en avant (French) [or the art to escape forward]. *Le Carré Bleu (French) [The Blue Square]*, 2:40–45, 1994. ISSN 0008-68-78.

[123] Michael Meißner and Bernhard Eberhardt. The art of knitted fabrics, realistic & physically based modelling of knitted patterns. *Computer Graphics Forum*, 17(3): 355–362, 1998. doi:10.1111/1467-8659.00282.

[124] Yuwei Meng, Charlie C. L. Wang, and Xiaogang Jin. Flexible shape control for automatic resizing of apparel products. *Computer-Aided Design*, 44(1):68–76, January 2012. ISSN 0010-4485. doi:10.1016/j.cad.2010.11.008.

[125] Jun Mitani and Hiromasa Suzuki. Making papercraft toys from meshes using strip-based approximate unfolding. *ACM Trans. Graph.*, 23(3):259–263, August 2004. ISSN 0730-0301. doi:10.1145/1015706.1015711.

[126] Joseph S. B. Mitchell, David M. Mount, and Christos H. Papadimitriou. The discrete geodesic problem. *SIAM Journal on Computing*, 16(4):647–668, 1987. doi:10.1137/0216045.

[127] Juan Montes, Bernhard Thomaszewski, Sudhir Mudur, and Tiberiu Popa. Computational design of skintight clothing. *ACM Trans. Graph.*, 39(4), July 2020. ISSN 0730-0301. doi:10.1145/3386569.3392477.

[128] Annika Muehlbradt, Gregory Whiting, Shaun Kane, and Laura Devendorf. Knitting access: Exploring stateful textiles with people with disabilities. In *Designing Interactive Systems Conference*, DIS '22, pages 1058–1070, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450393584. doi:10.1145/3532106.3533551.

[129] Georges Nader, Yu Han Quek, Pei Zhi Chia, Oliver Weeger, and Sai-Kit Yeung. KnitKit: A flexible system for machine knitting of customizable textiles. *ACM Trans. Graph.*, 40(4), July 2021. ISSN 0730-0301. doi:10.1145/3450626.3459790.

[130] Rahul Narain, Armin Samii, and James F. O'Brien. Adaptive anisotropic remeshing for cloth simulation. *ACM Trans. Graph.*, 31(6), November 2012. ISSN 0730-0301. doi:10.1145/2366145.2366171.

[131] Rahul Narain, Tobias Pfaff, and James F. O'Brien. Folding and crumpling adaptive sheets. *ACM Trans. Graph.*, 32(4), July 2013. ISSN 0730-0301. doi:10.1145/2461912.2462010.

[132] Rahul Narain, Armin Samii, Tobias Pfaff, and James F. O'Brien. ARCSim: Adaptive refining and coarsening simulator. `http://graphics.berkeley.edu/resources/ARCSim/`, 2013. [Online; accessed 16-Mar.-2020].

[133] Vidya Narayanan. *Foundations for 3D Machine Knitting*. PhD thesis, Carnegie Mellon University, 2021.

[134] Vidya Narayanan, Lea Albaugh, Jessica Hodgins, Stelian Coros, and James McCann. Automatic machine knitting of 3D meshes. *ACM Trans. Graph.*, 37(3), August 2018. ISSN 0730-0301. doi:10.1145/3186265.

[135] Vidya Narayanan, Kui Wu, Cem Yuksel, and James McCann. Visual knitting machine programming. *ACM Trans. Graph.*, 38(4), July 2019. ISSN 0730-0301. doi:10.1145/3306346.3322995.

[136] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2270–2279, 2017. doi:10.1109/CVPR.2017.244.

[137] Olivier Nocent, Jean-Michel Nourrit, and Yannick Remion. Towards mechanical level of detail for knitwear simulation. In *The 9th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, pages 252–259, 2001. URL https://otik.uk.zcu.cz/handle/11025/11277.

[138] Julian Panetta, Qingnan Zhou, Luigi Malomo, Nico Pietroni, Paolo Cignoni, and Denis Zorin. Elastic textures for additive fabrication. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi:10.1145/2766937.

[139] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[140] Georgios Pavlakos, Luyang Zhu, Xiaowei Zhou, and Kostas Daniilidis. Learning to estimate 3D human pose and shape from a single color image. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 459–468, 2018. doi:10.1109/CVPR.2018.00055.

[141] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed A. Osman, Dimitrios Tzionas, and Michael J. Black. Expressive body capture: 3d hands, face, and body from a single image. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10967–10977, 2019. doi:10.1109/CVPR.2019.01123.

[142] Jesús Pérez, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, José A. Canabal, Robert Sumner, and Miguel A. Otaduy. Design and fabrication of flexible rod meshes. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi:10.1145/2766998.

[143] Jesús Pérez, Miguel A. Otaduy, and Bernhard Thomaszewski. Computational design and automated fabrication of Kirchhoff-Plateau surfaces. *ACM Trans. Graph.*, 36(4), July 2017. ISSN 0730-0301. doi:10.1145/3072959.3073695.

[144] Marc Pollefeys. Visual 3D modeling from images, 2002. URL https://people.inf.ethz.ch/pomarc/pubs/PollefeysTutorial.pdf. [tutorial notes].

[145] Mariana Popescu, Matthias Rippmann, Tom Van Mele, and Philippe Block. Automated generation of knit patterns for non-developable surfaces. In *Humanizing Digital Reality: Design Modelling Symposium Paris 2017*, pages 271–284, Singapore, 2018. Springer Singapore. ISBN 978-981-10-6611-5. doi:10.1007/978-981-10-6611-5_24.

[146] Michael Rabinovich, Tim Hoffmann, and Olga Sorkine-Hornung. Discrete geodesic nets for modeling developable surfaces. *ACM Trans. Graph.*, 37(2), February 2018. ISSN 0730-0301. doi:10.1145/3180494.

[147] Jérémy Rapin and Olivier Teytaud. Nevergrad - A gradient-free optimization platform. https://github.com/FacebookResearch/Nevergrad, 2018.

[148] Konstantinos Rematas, Ira Kemelmacher-Shlizerman, Brian Curless, and Steve Seitz. Soccer on your tabletop. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4738–4747, 2018. doi:10.1109/CVPR.2018.00498.

[149] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. OctNet: Learning deep 3D representations at high resolutions. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6620–6629, 2017. doi:10.1109/CVPR.2017.701.

[150] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing. ISBN 978-3-319-24574-4. doi:10.1007/978-3-319-24574-4_28.

[151] Stefan Roth and Stephan R. Richter. Matryoshka networks: Predicting 3D geometry via nested shape layers. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1936–1944, 2018. doi:10.1109/CVPR.2018.00207.

[152] Asim Kumar Roy Choudhury. *Principles of Textile Finishing*. Woodhead Publishing Series in Textiles. Woodhead Publishing, 2017. ISBN 978-0-08-100646-7. doi:10.1016/C2014-0-04207-4.

[153] Gerard Rubio. OpenKnit: open source digital knitting. http://openknit.org/, 2014. [Online; accessed 1-Jan.-2022].

[154] Shunsuke Saito, Zeng Huang, Ryota Natsume, Shigeo Morishima, Hao Li, and Angjoo Kanazawa. PIFu: Pixel-aligned implicit function for high-resolution clothed human digitization. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2304–2314, 2019. doi:10.1109/ICCV.2019.00239.

[155] Shunsuke Saito, Tomas Simon, Jason Saragih, and Hanbyul Joo. PIFuHD: Multi-level pixel-aligned implicit function for high-resolution 3D human digitization. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 81–90, 2020. doi:10.1109/CVPR42600.2020.00016.

[156] Vanessa Sanchez, Conor J. Walsh, and Robert J. Wood. Textile technology for soft robotic and autonomous garments. *Advanced Functional Materials*, 31(6): 2008278, 2021. doi:10.1002/adfm.202008278.

[157] Josua Sassen, Klaus Hildebrandt, and Martin Rumpf. Nonlinear deformation synthesis via sparse principal geodesic analysis. *Computer Graphics Forum*, 39(5): 119–132, 2020. doi:10.1111/cgf.14073.

[158] Johannes L. Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4104–4113, 2016. doi:10.1109/CVPR.2016.445.

[159] Johannes L. Schönberger, Enliang Zheng, Jan-Michael Frahm, and Marc Pollefeys. Pixelwise view selection for unstructured multi-view stereo. In *Computer Vision – ECCV 2016*, pages 501–518, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46487-9. doi:10.1007/978-3-319-46487-9_31.

[160] Christian Schumacher, Bernd Bickel, Jan Rys, Steve Marschner, Chiara Daraio, and Markus Gross. Microstructures to control elasticity in 3D printing. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi:10.1145/2766926.

[161] Nicholas Sharp and Keenan Crane. Variational surface cutting. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi:10.1145/3197517.3201356.

[162] Nicholas Sharp, Yousuf Soliman, and Keenan Crane. The vector heat method. *ACM Trans. Graph.*, 38(3), June 2019. ISSN 0730-0301. doi:10.1145/3243651.

[163] Nicholas Sharp et al. Polyscope, 2019. www.polyscope.run.

[164] Shima Seiki Mfg., Ltd. Mach 2XS153. https://www.shimaseiki.com/product/knit/mach2xs/, 2019. [Online; accessed 1-Jan.-2022].

[165] Shima Seiki Mfg., Ltd. SHIMA SEIKI Releases New Design System. https://www.shimaseiki.com/news/press/apex4.html/, 2019. [Online; accessed 1-Jan.-2022].

[166] Aliaksandr Siarohin, Enver Sangineto, Stéphane Lathuilière, and Nicu Sebe. Deformable GANs for pose-based human image generation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3408–3416, 2018. doi:10.1109/CVPR.2018.00359.

[167] Luciano Silva, Olga R. P. Bellon, and Kim L. Boyer. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5): 762–776, 2005. doi:10.1109/TPAMI.2005.108.

[168] Silver Knitting Institute. *Bible for Machine Knitting: Guide to Knitting Techniques*. Japan, 1977.

[169] Mélina Skouras, Bernhard Thomaszewski, Stelian Coros, Bernd Bickel, and Markus Gross. Computational design of actuated deformable characters. *ACM Trans. Graph.*, 32(4), July 2013. ISSN 0730-0301. doi:10.1145/2461912.2461979.

[170] Mélina Skouras, Bernhard Thomaszewski, Peter Kaufmann, Akash Garg, Bernd Bickel, Eitan Grinspun, and Markus Gross. Designing inflatable structures. *ACM Trans. Graph.*, 33(4), July 2014. ISSN 0730-0301. doi:10.1145/2601097.2601166.

[171] Noeska Smit. Data knitualization: An exploration of knitting as a visualization medium. In *Proceedings of alt.VIS*, 2021. doi:10.17605/OSF.IO/C2UGS.

[172] Jason Smith and Scott Schaefer. Bijective parameterization with free boundaries. *ACM Trans. Graph.*, 34(4), July 2015. ISSN 0730-0301. doi:10.1145/2766947.

[173] Yousuf Soliman, Dejan Slepčev, and Keenan Crane. Optimal cone singularities for conformal flattening. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi:10.1145/3197517.3201367.

[174] Guowen Song, editor. *Improving Comfort in Clothing*. Woodhead Publishing Series in Textiles. Woodhead Publishing, 2011. ISBN 978-1-84569-539-2. doi:10.1533/9780857090645.

[175] Olga Sorkine and Marc Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Fifth Eurographics Symposium on Geometry Processing*, SGP '07, pages 109–116, Goslar, DEU, 2007. Eurographics Association. ISBN 9783905673463. doi:10.2312/SGP/SGP07/109-116.

[176] Olga Sorkine-Hornung and Michael Rabinovich. Least-squares rigid motion using SVD, 2017. URL https://igl.ethz.ch/projects/ARAP/svd_rot.pdf.

[177] David J. Spencer. *Knitting Technology: A Comprehensive Handbook and Practical Guide*. Woodhead Publishing, Cambridge, England, 3rd edition, 2001. ISBN 978-1-85573-333-6. doi:10.1533/9781855737556.

[178] Georg Sperl, Rahul Narain, and Chris Wojtan. Homogenized yarn-level cloth. *ACM Trans. Graph.*, 39(4), July 2020. ISSN 0730-0301. doi:10.1145/3386569.3392412.

[179] Stoll. Pattern software M1PLUS®. https://www.stoll.com/fileadmin/user_upload/pdfs/Brochures_english/M1plus_15_gb.pdf. [Online; accessed 1-Jan.-2022].

[180] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3D shape recognition. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 945–953, 2015. doi:10.1109/ICCV.2015.114.

[181] Robert W. Sumner and Jovan Popović. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, August 2004. ISSN 0730-0301. doi:10.1145/1015706.1015736.

[182] Dominik Surc, Dominik Michel, Alexander Mirosnicenko, Alexander Artschwager, and Uwe Roeder. Scan to knit - from body scan directly to the knitting machine. In *Proc. of 3DBODY.TECH 2020 - 11th Int. Conf. and Exh. on 3D Body Scanning and Processing Technologies*, 2020. doi:10.15221/20.30.

[183] Md. Sarower Tareq, Tanzilur Rahman, Mokarram Hossain, and Peter Dorrington. Additive manufacturing and the COVID-19 challenges: An in-depth study. *Journal of Manufacturing Systems*, 60:787–798, 2021. ISSN 0278-6125. doi:10.1016/j.jmsy.2020.12.021.

[184] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Multi-view 3D models from single images with a convolutional network. In *Computer Vision – ECCV 2016*, pages 322–337, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46478-7. doi:10.1007/978-3-319-46478-7_20.

[185] Maxim Tatarchenko, Alexey Dosovitskiy, and Thomas Brox. Octree generating networks: Efficient convolutional architectures for high-resolution 3D outputs. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2107–2115, 2017. doi:10.1109/ICCV.2017.230.

[186] Michael E. Taylor. Differential geometry, 2018. URL https://mtaylor.web.unc.edu/wp-content/uploads/sites/16915/2018/04/DGEOM.pdf. [course notes].

[187] Textile Today. Spinners can take price advantage using recycled yarn. https://www.textiletoday.com.bd/spinners-can-take-price-advantage-using-recycled-yarn/, 2020. [Online; accessed 1-Apr.-2022].

[188] The MakeHuman team. MakeHuman: Open source tool for making 3D characters. http://www.makehumancommunity.org/, 2020. [v1.2.0].

[189] Matthew Trumble, Andrew Gilbert, Adrian Hilton, and John Collomosse. Deep autoencoder for combined human pose estimation and body model upscaling. In *Computer Vision – ECCV 2018*, pages 800–816, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01249-6. doi:10.1007/978-3-030-01249-6_48.

[190] Nobuyuki Umetani, Danny M. Kaufman, Takeo Igarashi, and Eitan Grinspun. Sensitive couture for interactive garment modeling and editing. In *ACM SIGGRAPH 2011 Papers*, SIGGRAPH '11, New York, NY, USA, 2011. ACM. ISBN 9781450309431. doi:10.1145/1964921.1964985.

[191] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991. doi:10.1109/34.88573.

[192] Jenny Underwood. *The design of 3D shape knitted preforms*. PhD thesis, RMIT University, 2009. URL https://researchrepository.rmit.edu.au/esploro/outputs/9921861597201341.

[193] Benjamin Vaissier, Jean-Philippe Pernot, Laurent Chougrani, and Philippe Véron. Genetic-algorithm based framework for lattice support structure optimization in additive manufacturing. *Computer-Aided Design*, 110:11–23, 2019. ISSN 0010-4485. doi:10.1016/j.cad.2018.12.007.

[194] Daan van der Valk. Knitted smart textile sensors: Integrating technlogy into garments by using knitting. Master's thesis, Delft University of Technology, 2020. URL http://resolver.tudelft.nl/uuid:df5c1ddc-0296-4122-845c-3c28506c147d.

[195] Gül Varol, Javier Romero, Xavier Martin, Naureen Mahmood, Michael J. Black, Ivan Laptev, and Cordelia Schmid. Learning from synthetic humans. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4627–4635, 2017. doi:10.1109/CVPR.2017.492.

[196] Gül Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. BodyNet: Volumetric inference of 3D human body shapes. In *Computer Vision – ECCV 2018*, pages 20–38, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01234-2. doi:10.1007/978-3-030-01234-2_2.

[197] Amir Vaxman, Marcel Campen, Olga Diamanti, David Bommes, Klaus Hildebrandt, Mirela Ben-Chen, and Daniele Panozzo. Directional field synthesis, design, and processing. In *ACM SIGGRAPH 2017 Courses*, SIGGRAPH '17, New York, NY, USA, 2017. Association for Computing Machinery. ISBN 9781450350143. doi:10.1145/3084873.3084921.

[198] Natascha M. van der Velden. *Making Fashion Sustainable: The Role of Designers*. PhD thesis, Delft University of Technology, 2016. URL https://doi.org/10.4233/uuid:8c66ca0a-605e-4f22-a4f1-f59b7e9ac874.

[199] Natascha M. van der Velden, Martin K. Patel, and Joost G. Vogtländer. LCA benchmarking study on textiles made of cotton, polyester, nylon, acryl, or elastane. *The International Journal of Life Cycle Assessment*, 19:331–356, 2014. ISSN 1614-7502. doi:10.1007/s11367-013-0626-9.

[200] Pascal Volino, Martin Courchesne, and Nadia Magnenat Thalmann. Versatile and efficient techniques for simulating cloth and other deformable objects. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '95, pages 137–144, New York, NY, USA, 1995. ACM. ISBN 0897917014. doi:10.1145/218380.218432.

[201] Charlie C. L. Wang. *Geometric Modeling and Reasoning of Human-Centered Freeform Products*. Springer-Verlag London, 2013. ISBN 978-1-4471-4360-4. doi:10.1007/978-1-4471-4360-4.

[202] Charlie C. L. Wang and Yong Chen. Thickening freeform surfaces for solid fabrication. *Rapid Prototyping Journal*, 19(6):395–406, January 2013. ISSN 1355-2546. doi:10.1108/RPJ-02-2012-0013.

[203] Charlie C. L. Wang and Kai Tang. Optimal boundary triangulations of an interpolating ruled surface. *Journal of Computing and Information Science in Engineering*, 5(4):291–301, February 2005. ISSN 1530-9827. doi:10.1115/1.2052850.

[204] Charlie C. L. Wang and Kai Tang. Pattern computation for compression garment by a physical/geometric approach. *Computer-Aided Design*, 42(2):78–86, February 2010. ISSN 0010-4485. doi:10.1016/j.cad.2009.02.018.

[205] Charlie C. L. Wang, Kin-Chuen Hui, and Kai-Man Tong. Volume parameterization for design automation of customized free-form products.

*IEEE Transactions on Automation Science and Engineering*, 4(1):11–21, 2007. doi:10.1109/TASE.2006.872112.

[206] Charlie C. L. Wang, Yunbo Zhang, and Hoi Sheung. From designing products to fabricating them from planar materials. *IEEE Computer Graphics and Applications*, 30(06):74–85, November 2010. ISSN 1558-1756. doi:10.1109/MCG.2009.155.

[207] Huamin Wang, James F. O'Brien, and Ravi Ramamoorthi. Data-driven elastic models for cloth: Modeling and measurement. *ACM Trans. Graph.*, 30(4), July 2011. ISSN 0730-0301. doi:10.1145/2010324.1964966.

[208] Nanyang Wang, Yinda Zhang, Zhuwen Li, Yanwei Fu, Wei Liu, and Yu-Gang Jiang. Pixel2Mesh: Generating 3D mesh models from single RGB images. In *Computer Vision – ECCV 2018*, pages 55–71, Cham, 2018. Springer International Publishing. ISBN 978-3-030-01252-6. doi:10.1007/978-3-030-01252-6_4.

[209] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: Octree-based convolutional neural networks for 3D shape analysis. *ACM Trans. Graph.*, 36(4), July 2017. ISSN 0730-0301. doi:10.1145/3072959.3073608.

[210] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional GANs. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8798–8807, 2018. doi:10.1109/CVPR.2018.00917.

[211] Ron Wein, Eric Berberich, Efi Fogel, Dan Halperin, Michael Hemmer, Oren Salzman, and Baruch Zukerman. 2D arrangements. In *CGAL User and Reference Manual*. CGAL Editorial Board, 5.3 edition, 2021. URL `https://doc.cgal.org/5.3/Manual/packages.html#PkgArrangementOnSurface2`.

[212] Kui Wu. *Modeling, Rendering, and Simulating Knits*. PhD thesis, The University of Utah, 2019.

[213] Kui Wu and Cem Yuksel. Real-time cloth rendering with fiber-level detail. *IEEE Transactions on Visualization and Computer Graphics*, 25(2):1297–1308, 2019. doi:10.1109/TVCG.2017.2731949.

[214] Kui Wu, Xifeng Gao, Zachary Ferguson, Daniele Panozzo, and Cem Yuksel. Stitch meshing. *ACM Trans. Graph.*, 37(4), July 2018. ISSN 0730-0301. doi:10.1145/3197517.3201360.

[215] Kui Wu, Hannah Swan, and Cem Yuksel. Knittable stitch meshes. *ACM Trans. Graph.*, 38(1), January 2019. ISSN 0730-0301. doi:10.1145/3292481.

[216] Kui Wu, Marco Tarini, Cem Yuksel, James McCann, and Xifeng Gao. Wearable 3D machine knitting: Automatic generation of shaped knit sheets to cover real-world objects. *IEEE Transactions on Visualization and Computer Graphics*, 28(9):3180–3192, 2022. doi:10.1109/TVCG.2021.3056101.

[217] Hongyi Xu, Yijing Li, Yong Chen, and Jernej Barbič. Interactive material design using model reduction. *ACM Trans. Graph.*, 34(2), March 2015. ISSN 0730-0301. doi:10.1145/2699648.

[218] Ying-Qing Xu, Yanyun Chen, Stephen Lin, Hua Zhong, Enhua Wu, Baining Guo, and Heung-Yeung Shum. Photorealistic rendering of knitwear using the lumislice. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '01, pages 391–398, New York, NY, USA, 2001. Association for Computing Machinery. ISBN 158113374X. doi:10.1145/383259.383303.

[219] Lap-Fai Yu, Sai-Kit Yeung, Chi-Keung Tang, Demetri Terzopoulos, Tony F. Chan, and Stanley J. Osher. Make it home: Automatic optimization of furniture arrangement. *ACM Trans. Graph.*, 30(4), July 2011. ISSN 0730-0301. doi:10.1145/2010324.1964981.

[220] Tianhong Catherine Yu and James McCann. Coupling programs and visualization for machine knitting. In *Symposium on Computational Fabrication*, SCF '20, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450381703. doi:10.1145/3424630.3425410.

[221] Cem Yuksel, Jonathan M. Kaldor, Doug L. James, and Steve Marschner. Stitch meshes for modeling knitted clothing with yarn-level detail. *ACM Trans. Graph.*, 31(4), July 2012. ISSN 0730-0301. doi:10.1145/2185520.2185533.

[222] Jonas Zehnder, Espen Knoop, Moritz Bächer, and Bernhard Thomaszewski. Metasilicone: Design and fabrication of composite silicone with desired mechanical properties. *ACM Trans. Graph.*, 36(6), November 2017. ISSN 0730-0301. doi:10.1145/3130800.3130881.

[223] Chao Zhang, Sergi Pujades, Michael Black, and Gerard Pons-Moll. Detailed, accurate, human shape estimation from clothed 3D scan sequences. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5484–5493, 2017. doi:10.1109/CVPR.2017.582.

[224] Eugene Zhang, Konstantin Mischaikow, and Greg Turk. Feature-based surface parameterization and texture mapping. *ACM Trans. Graph.*, 24(1):1–27, January 2005. ISSN 0730-0301. doi:10.1145/1037957.1037958.

[225] Juyong Zhang, Bailin Deng, **Zishun Liu**, Giuseppe Patanè, Sofien Bouaziz, Kai Hormann, and Ligang Liu. Local barycentric coordinates. *ACM Trans. Graph.*, 33(6), November 2014. ISSN 0730-0301. doi:10.1145/2661229.2661255.

[226] Xiaoting Zhang, Xinyi Le, Zihao Wu, Emily Whiting, and Charlie C. L. Wang. Data-driven bending elasticity design by shell thickness. *Computer Graphics Forum*, 35 (5):157–166, 2016. doi:10.1111/cgf.12972.

[227] Yunbo Zhang and Charlie C. L. Wang. WireWarping++: Robust and flexible surface flattening with length control. *IEEE Transactions on Automation Science and Engineering*, 8(1):205–215, 2011. doi:10.1109/TASE.2010.2051665.

[228] Shuang Zhao, Wenzel Jakob, Steve Marschner, and Kavita Bala. Building volumetric appearance models of fabric using micro CT imaging. *ACM Trans. Graph.*, 30 (4), July 2011. ISSN 0730-0301. doi:10.1145/2010324.1964939.

[229] Shuang Zhao, Fujun Luan, and Kavita Bala. Fitting procedural yarn models for realistic cloth rendering. *ACM Trans. Graph.*, 35(4), July 2016. ISSN 0730-0301. doi:10.1145/2897824.2925932.

[230] Zerong Zheng, Tao Yu, Yixuan Wei, Qionghai Dai, and Yebin Liu. DeepHuman: 3D human reconstruction from a single image. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7738–7748, 2019. doi:10.1109/ICCV.2019.00783.

[231] Qian-Yi Zhou, Jaesik Park, and Vladlen Koltun. Open3D: A modern library for 3D data processing. *arXiv*, 2018. doi:10.48550/arXiv.1801.09847.

[232] Hao Zhu, Hao Su, Peng Wang, Xun Cao, and Ruigang Yang. View extrapolation of human body from a single image. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4450–4459, 2018. doi:10.1109/CVPR.2018.00468.

[233] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017. doi:10.1109/ICCV.2017.244.

# CURRICULUM VITÆ

## Zishun LIU

27-12-1991 Born in Liaocheng, Shandong, China.

## EDUCATION

2016–2022 PhD student
Delft University of Technology
Delft, the Netherlands
*Dissertation:* Customized 3D and 4D Design for Machine Knitting
*Promotors:* Prof. dr. Charlie C.L. Wang
Prof. dr. ir. Jo M.P. Geraedts
*Copromotor:* Dr. ir. Eugeni L. Doubrovski

2013–2016 Master's student in Computational Mathematics
University of Science and Technology of China
Hefei, Anhui, China

2009–2013 Undergraduate student in Computational Mathematics
University of Science and Technology of China
Hefei, Anhui, China

## PROFESSIONAL EXPERIENCE

2020–2022 Research Assistant
Centre for Perceptual and Interactive Intelligence (CPII)
Hong Kong, China

# PUBLICATION

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