Algorithms for Suture Placement and Seed Placement: Planar Geometric Optimization for Surgery and Agriculture

Varun Kamat

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2023-196
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2023/EECS-2023-196.html

July 28, 2023

Acknowledgement

[abridged, see text for full]
I am very grateful to Prof. Goldberg and the lab for the research experience I've had - I have learnt so much in the last two and a half years.
Thank you to Prof. Friedland for being my second reader, and for providing insightful and helpful feedback!
A big thanks to all of my collaborators!
I’m very grateful to my parents for giving me 100% of their attention and love throughout my life. Thanks to my extended family.
Although I can’t list everyone, many friends and mentors have been mentioned in the Dedication section, thank you to everyone!
Well, I only have so much control over what will happen next, but I hope that my experiences will continue to be full of joy, meaning, connection, beauty, intrigue, and authenticity! Cheers, everyone!

Algorithms for Suture Placement and Seed Placement: Planar Geometric Optimization for Surgery and Agriculture

by

Varun Kamat

A thesis submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Ken Goldberg, Chair
Professor Gerald Friedland, Co-chair

August 2023

Algorithms for Suture Placement and Seed Placement: Planar Geometric Optimization for Surgery and Agriculture

by Varun Kamat

Research Project

Submitted to the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, in partial satisfaction of the requirements for the degree of **Master of Science, Plan II.**

Approval for the Report and Comprehensive Examination:

Committee:

Professor Ken Goldberg
Research Advisor

28 July 2023
(Date)

* * * * * * *

* * * * * * *

Professor Gerald Friedland
Second Reader

July 28th, 2023
(Date)

Algorithms for Suture Placement and Seed Placement:
Planar Geometric Optimization for Surgery and Agriculture

Abstract

Algorithms for Suture Placement and Seed Placement:
Planar Geometric Optimization for Surgery and Agriculture

by

Varun Kamat

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Ken Goldberg, Chair

Professor Gerald Friedland, Co-chair

Computational Geometry and Geometric Optimization aims to arrange multiple objects in a confined space to achieve some objective. In this thesis, I explore two such problems: Surgical Suture Placement and Seed Placement for Polyculture Gardens.

Suture placement is crucial for patient outcomes, and surgeons typically rely on rules of thumb and experience to choose needle entry and exit points for suture placement. I present SP2DEEF: Suture Planning 2D Equalizing Elliptical Forces, an algorithm that computes entry and exit points to optimize suture forces, improving wound closure and minimizing scarring. SP2DEEF takes as input the wound curve along with scaling information as input, and generates a full, optimized suture plan that can be fine-tuned by the surgeon. Experiments suggest that our suturing algorithm outperforms a naive baseline, and physical phantom experiments suggest that it performs comparably or superior to an expert surgeon. My team and I are currently in the process of developing a publicly-available website which will make the SP2DEEF pipeline fully accessible, the link will be provided shortly at https://github.com/BerkeleyAutomation/SP2DEEF.

The spatial arrangement of plants in a garden or small farm is especially important for polyculture agriculture to reduce water and pesticide use. We present PolyPoD, a new algorithm for seed placement. Given a polygonal planting area boundary (convex or non-convex) and number and types of seeds, PolyPoD generates the variable radius poisson disk distribution, yielding viable seed placements. Results suggest that PolyPoD outperforms our previous seed placement algorithm with the goal of avoiding over-competition for resources, underutilization of space, and plant segmentation difficulties. The PolyPoD platform will also offer free access on a website for polyculture farmers worldwide. Source code, documentation, and the website link are available at https://github.com/BerkeleyAutomation/PolyPoD.

To Jacob, Zevan, Alexander, Lane, Padmini, Esha, Bradley, Vivek, Sajal, Akhil, Prem, Nitin, Kristen, Vihan, Krish, Pascal, Parnika, Kruti, Stef, Margaret, Yuqi, Jose, Meg, Hillary, Willow, Christine, Jason, Derek, Tyler, Viraaj, Neha, Rucha, Rhea, Liya, Deepti, Natasha, Gurtej, Brian, Isadora, Taylor, Vivian, Carolyn, Kyle, Alison, Rebecca, Tom, Susan, Tina, Kurt, Wendell, Milan, my teachers, the AUTOLab community, my parents, and my extended family. I'm lucky to have so many wonderful people in my life!

Contents

List of Figures

List of Tables

Acknowledgments

Joining Professor Goldberg's lab gave me something to do in a pretty dark and empty time during the Spring of 2021. I had just recovered from my fourth orthopedic surgery, and I was still in the middle of COVID-19 quarantine. It was a step back into Computer Science after almost two years of being away from the field, and I am grateful that I got to begin my second wave of college on such a high note. I worked full-time for the next six months in the lab, completed my senior year at Berkeley, and signed up to do the Fifth Year MS, publishing two papers and culminating with this MS Thesis.

I am very grateful to Prof. Goldberg and the lab for giving me the opportunity to do research here - I have learnt so much from Professor Goldberg and everyone in the lab in the last two and a half years. Ultimately, I think that all of the thinking, writing, discussions, emails, meetings, experiments, coding, and more that I did here will stay with me for a long time, and I'll continue to draw value from my time here when I'm trying to solve other problems and discover things later in my professional career and personal life.

Thank you to Prof. Friedland for being my second reader, and for providing insightful and helpful feedback on this thesis!

A big thanks to all of my collaborators - research in AUTOLab is certainly a group effort and I am lucky to have been able to work with great teams!

I'm very grateful to my parents for giving me 100% of their attention and love throughout my life. They have really shaped me into the person I am today, and have given me the foundation and values that guide me though everything. I also want thank everyone in my extended family for being, well, my family! Much love to you all.

My life has been marked by great company all around. Although I can't list everyone, many of them have been mentioned in the Dedication section, because I really value all of the connections I've been able to make so far. It's a big part of what makes life worth it for me, and I hope to continue to stay connected, living, laughing, loving, supporting, and solving problems together with people I care about and relate to.

Well, I only have so much control over what will happen next, but I hope that my experiences will continue to be full of joy, meaning, connection, beauty, intrigue, and authenticity! Cheers, everyone!

Abstract

Algorithms for Suture Placement and Seed Placement:
Planar Geometric Opitimization for Surgery and Agriculture

by

Varun Kamat

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Ken Goldberg, Chair

Professor Gerald Friedland, Co-chair

Computational Geometry and Geometric Optimization aims to arrange multiple objects in a confined space to achieve some objective. In this thesis, I explore two such problems: Surgical Suture Placement and Seed Placement for Polyculture Gardens.

Suture placement is crucial for patient outcomes, and surgeons typically rely on rules of thumb and experience to choose needle entry and exit points for suture placement. I present SP2DEEF: Suture Planning 2D Equalizing Elliptical Forces, an algorithm that computes entry and exit points to optimize suture forces, improving wound closure and minimizing scarring. SP2DEEF takes as input the wound curve along with scaling information as input, and generates a full, optimized suture plan that can be fine-tuned by the surgeon. Experiments suggest that our suturing algorithm outperforms a naive baseline, and physical phantom experiments suggest that it performs comparably or superior to an expert surgeon. My team and I are currently in the process of developing a publicly-available website which will make the SP2DEEF pipeline fully accessible, the link will be provided shortly at https://github.com/BerkeleyAutomation/SP2DEEF.

The spatial arrangement of plants in a garden or small farm is especially important for polyculture agriculture to reduce water and pesticide use. We present PolyPoD, a new algorithm for seed placement. Given a polygonal planting area boundary (convex or non-convex) and number and types of seeds, PolyPoD generates the variable radius poisson disk distribution, yielding viable seed placements. Results suggest that PolyPoD outperforms our previous seed placement algorithm with the goal of avoiding over-competition for resources, underutilization of space, and plant segmentation difficulties. The PolyPoD platform will also offer free access on a website for polyculture farmers worldwide. Source code, documentation, and the website link are available at https://github.com/BerkeleyAutomation/PolyPoD.

Chapter 1

Introduction

Many problems in Computational Geometry [6] involve placing multiple objects in a finite space, with some objective. Such problems include Packing Problems and Art Gallery Problems. In Packing Problems, the goal is to densely pack objects together. The most common variants deal with packing circles or squares into as small of a circle, square or rectangle as possible [47]. In the Art Gallery Problem, there is an art gallery with walls, and the goal is to place as few guards as possible in the gallery such that all areas of the gallery are visible [46].

In this thesis, I explore two such problems: planning the layout of a garden, and placing sutures on a wound. In the garden planning problem, the goal is to place plants in a garden to satisfy multiple objectives: full utilization of the garden space, promoting polyculture farming, and satisfying the user's objectives. The full utilization aspect of this problem essentially boils down to a version of the packing problem where instead of fixing the number of objects to pack and attempting to minimize the size of the bin, we fix the size and shape of the bin and try to maximize the number of plants / area to be filled in the bin. However, this case is not very well-studied, especially in the case of arbitrary polygonal bin shapes as well as different sized objects. In any case, most optimization frameworks would be intractable in this setting due to highly non-convex constraints that ensure that different shapes do not overlap, as well as many discrete elements. Other algorithmic techniques that generate solutions are a more feasible route than closed-form optimization to finding solutions.

For the suture placement problem, the goal is to place sutures on a wound satisfying three primary objectives: achieving even, ideal closure force along the wound, minimizing shear force along the wound, and avoiding injury or damage to the tissue. I believe that this problem is related to the Art Gallery Problem: both have objects to be placed (sutures vs guards) that aim to exert influence (closure force vs clear line of sight) in a geometric environment (wound vs gallery) with restrictions dictated by features of the environment (wound curvature vs walls). The curves of the wound act very much like the walls of the art gallery, because a curve limits the influence of sutures on either side of it. I suspect that if there were a version of the Art Gallery Problem where each guard's influence decreases over space, similar to how suture force operates, and walls can have varying opacity based on the location of the guards, there may be some reduction of the Suture Placement Problem to the Art Gallery problem, although I have not worked through the details of such a reduction yet. Unlike the Garden Planning problem, however, the suture placement problem is much simpler: it operates on a single curve, and if we fix the number of sutures, an optimization boils down to a number of relatively smooth, relatively convex set of constraints, although the problem is not perfectly convex. Therefore, this problem is well-suited to closed form optimization.

Chapter 2

Automating 2D Suture Placement

2.1 Introduction

The suture placement work was done in conjunction with Viraj Ramakrishnan, Yashish Mohnot, Harshika Jalan, Julia Isaac, Vincent Schorp, Yahav Avigal, Aviv Adler, Dr. Danyal M Fer, and Prof. Ken Goldberg. In this project, I played a leading role, organizing meetings and coordinating tasks and roles. I ideated most of this project by myself, including the closed-form optimization framework with variables as points along the wound curve, and the math for all of the distance calculations. I also came up with the idea for a distance/orientation based force model by myself, with inspiration from the Diamond Model, and ideated the Elliptical Force model in conjunction with Aviv. I setup starter code for a all of the optimization framework, and implemented all of the Elliptical Force / Shear Force code by myself. I played a large role in the code for the other elements of the optimization as well, building starter code, helping with debugging, developing some parts of the optimization, as well as integrating different pieces of code together in our repository. Viraj, Harshika, Yashish, and Julia contributed to development on the optimization framework and graphic interface. Harshika and Viraj developed the distance calculations; Viraj, Yashish, Julia, and Harshika worked on optimization; Viraj and Julia worked on the Graphic Interface; Yashish and Harshika develped and ran code for the virtual experiments. Viraj and Harshika are in the process of developing the website interface. Vincent helped with ideation for the form of the suturing constraints as well as ideation for example wounds, and other areas of the project. Yahav played a co-leadership role, helping to divide tasks and set priorities for the project. Aviv contributed to ideation for the dimensions and equations for the Ellipse in our model. Dr. Danyal Fer was our medical advisor; he provided medical guidance and perspective on the usability of our algorithm, served as our expert surgeon in our physical experiments, and provided the final evaluation for our physical experiments. Prof. Goldberg oversaw the project and was our faculty advisor.

Suturing is the process of sewing a wound or laceration closed to allow it to heal naturally. It is extremely common in surgery and trauma care and involves long sequences of precise, repetitive movements – something that is often burdensome to humans.

This work focuses on one aspect of automating suturing: the sub-task of *suture planning*, which is to find an appropriate sequence of needle insertion and extraction points. Having a high quality placement of suture is crucial for healing, as having sutures that are too close or tight may lead to ischemia (under-supply of oxygen to the tissue), while having sutures too far apart may lead to insufficient closing force on the wound to ensure the edges stay together [12]. Furthermore, sutures may exert shear forces along the wound, which cause more pronounced scarring and lead to cosmetically unappealing results [42] [27]. Thus, it is ideal to plan sutures by directly optimizing the forces they are expected to generate. Humans cannot directly estimate closure and shear forces as they suture, and typically rely on rules of thumb, intuition, and experience to guide suture placement.

This project makes five contributions:

1. 1) A novel objective, optimizing Elliptical Force on the wound, based on an extension

of the well-known "diamond force model" [9, 10] to nonlinear 2D wound shapes.

2. A novel formulation of planar, non-linear suture planning as a constrained optimization problem.

3. An analysis of the SP2DEEF algorithm's output on wounds of varying degrees and curvature.

4. Experiments comparing the suture placements from SP2DEEF to those chosen by a human surgeon.

5. An interactive interface that allows surgeons to upload an image, trace the wound curve, view proposed sutures and edit suture placement.

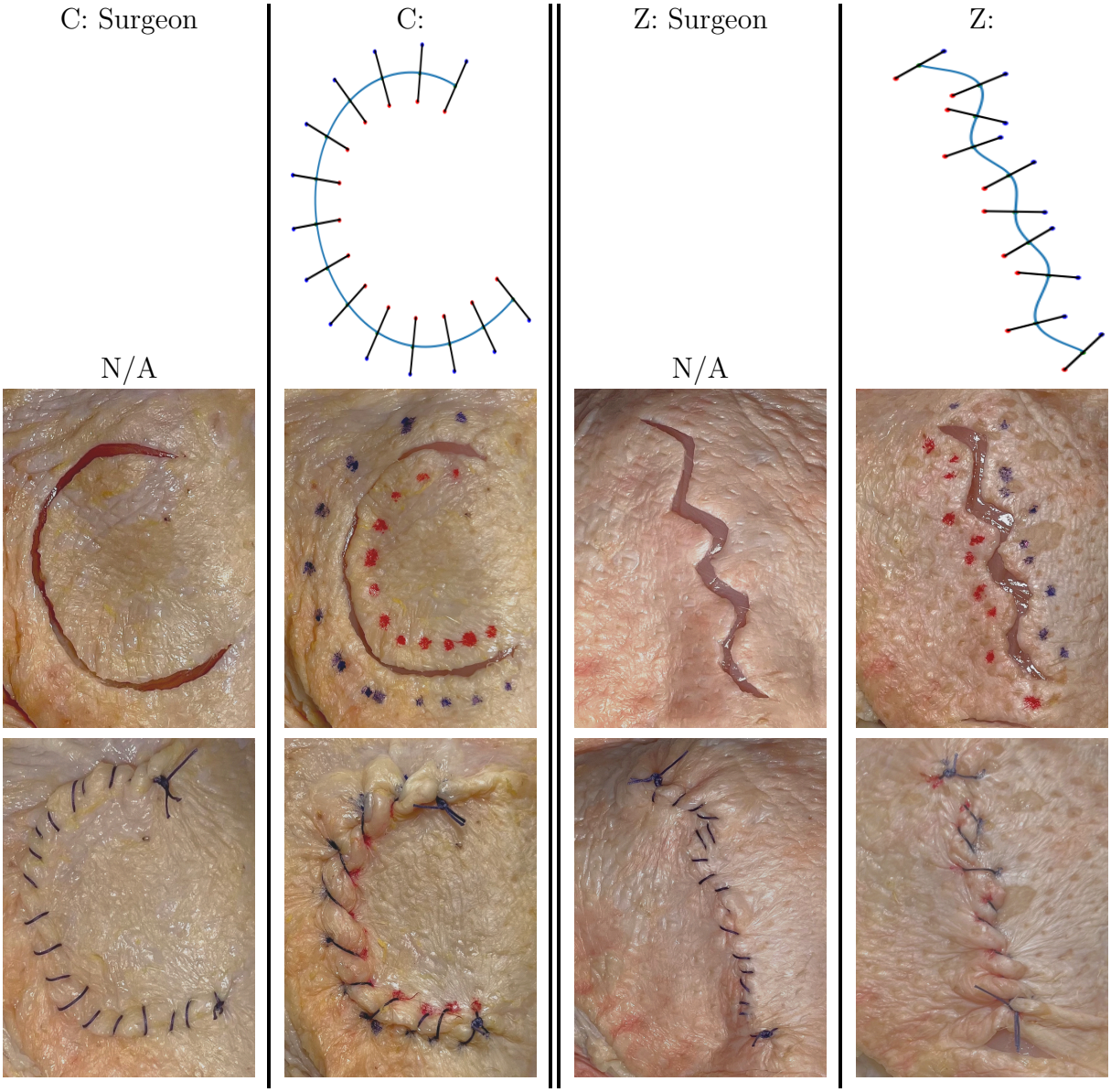

Figure 2.1: Physical Experiments with chicken skin. Top row: 's outputted placements; Middle row: Surgeon's initial state and physical ink markings of 's placements; Bottom row: after suturing. Left half: C-shaped wound; Right half: Z-shaped wound. First and third columns: Surgeon's placement; second and fourth columns: 's placement. 's placements were evaluated as equal or better than the surgeon's placement by the surgeon.

2.2 Related Work

Automated and robot-assisted suturing has seen considerable study over the last decade, with particular focus on two sub-tasks: *2D suture planning*, which considers where to place sutures (the focus of this work), and *3D needle path planning*, which considers how to guide the needle to best accomplish a desired suture. Automating either of these sub-tasks can provide valuable assistance to a human surgeon, while having both may potentially lead to a fully automated suturing system.

Needle Path Planning

Much of the work on autonomous suturing focuses on needle path planning in the vertical plane (orthogonal to the wound line), as pushing a needle through deformable tissue represents a major challenge for sensing and control. In general, needle path planning focuses on a single suture at a time.

Nageotte et al. [30] proposed a kinematic analysis and geometric modeling of the problem of the stitching task in laparoscopic surgery. The work particularly uncovers the degree of uncertainty which the surgeon has with regard to where the tip of the needle is; that is, the exit point may not be exactly where desired.

Schulman et al. [38] applied the *transfer trajectory algorithm* to take trajectories from human demonstrations and adapt them to new environment geometry for suture needle path planning. Another approach developed by Sen et al. [39], was one of sequential convex optimization. As with the other papers, the optimization was over the execution of a single suture. Their approach considers multiple sutures, but the suture locations are chosen by linear interpolation of the start and end points, and each suture is treated independently thereafter. This method does not capture the constraints that the curvature of a wound might place on suture placement. Later, Shademan et al. [40] studied suturing on intestinal tissue. Various constraints for good suturing are discussed in this work: for instance, that sutures should be perpendicular to the wound, and the gaps between the sutures must be small enough to avoid leakage, but not too small as to prevent bloodflow. However, they are primarily used as assumptions that hold for a set of planned sutures that have been decided beforehand.

Other papers consider additional aspects of needle path planning: for example, Pedram et al. [31] presented an algorithm that takes in the desired suture entry and exit points on a wound as input and computes the needle shape, diameter and path so that the execution of the suture satisfies recommended suturing guidelines. These guidelines included: a minimization of tissue trauma, orthogonal sutures, positioning the needle to allow for successful grasps, suture symmetry and being able to enter and exit at the points defined by the surgeon. The optimization weights were selected based on the recommendation of surgeons and fine-tuned during simulations. Jackson et al. [21] proposed a Kalman filter to model the internal deformation force generated by a needle as it is driven through tissue. Similarly, Pedram et al. [32] described a needle stitch path planning algorithm which deals with optimal

motion of the needle inside the tissue, with the goal of entering the tissue perpendicularly; reaching specific suture depth; and minimizing tissue trauma.

The Suture Planning Problem

There is prior work in planning the location of suture entry and exit points in the horizontal plane, for specific wounds, and in certain controlled settings. In particular, robotic minimally-invasive surgery (RMIS) must conform to the robot's kinematic constraints and the potentially very tight space in which the operation takes place.

Saeidi et al. [37] describe a planning algorithm for autonomous suturing using a segmented point cloud and demonstrated its application with the Smart Tissue Autonomous Robot (STAR) system. However, their technique assumed a straight-line wound (as can be expected in surgery, where the wound is the result of a surgical incision). Similarly, Thananjeyan et al. [44] studied suture planning for a circular wound.

The primary constraints in both works were limitations to the robot's movement ability, either from kinematics or from a sharply bounded workspace, with the objective being to ensure evenly-spaced sutures with the gaps between them being as close as possible to an ideal distance.

However, when dealing with more complex wound shapes, sutures may interact with each other in more complex ways, and thus planning and evaluating an effective set of sutures may require a more detailed model of how the sutures hold the wound together.

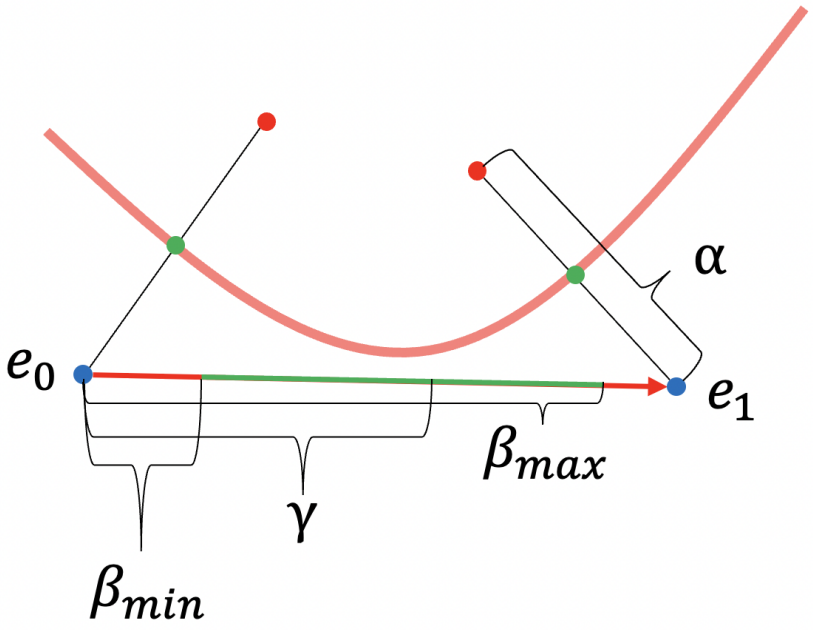

Figure 2.2: Wound centerline shown in red, with two entry points (red) and two exit points (blue). Consider the distance between extraction points e_0, e_1. If the distance is between β_{min} and β_{max}, i.e. the green zone, it meets the constraints. Hence e_0 and e_1 violate constraints.

2.3 Problem Statement

Given an image with points along the wound selected by the surgeon, the suture planning problem consists of choosing the number and placement of sutures to best close the wound. We denote the number of sutures as n, and, for most of what follows, we treat it as fixed and consider the optimization problem which represents the task of finding the best placement of n sutures; we discuss how n is selected in Section 2.4. The parameters of this problem come in two types: the first type denote physical constraints or objectives. For example, the minimum allowable distance between any two skin puncture points or the ideal distance between the insertion and extraction points of a single suture (which we refer to as the *suture width*, α) We represent such constraints with Greek letters. The second kind of parameter are scalar weights which set the relative importance of different components of the objective function, and are represented by $c^\star$ where the superscript $\star$ denotes the corresponding component.

Let α be the *suture widths*, that is, the distance between the insertion and extraction points of a single suture. These two points are placed at equal distance apart from the wound such that their midpoint lies on the wound curve. Additionally, no two insertion and/or extraction points should be closer than a certain minimum distance, denoted $\beta_{\min}$, and no two consecutive insertion and/or extraction points should be further than a certain maximum distance, denoted $\beta_{\max}$. We also define the *suture distance* which is the straight line distance between the midpoints of two stitches. Guidance for surgeons suggests that an ideal suturing distance is 5mm [15]. Let γ be this target distance between sutures, and let ℓ be the length of the wound.

Let L^d be the Mean Squared Error (MSE) defined as average difference between computed suture distance and the target suture distance γ. Let $L^{\text{var\_center}}$ be the variance of computed suture distances. Let $L^{\text{var\_ins\_ext}}$ be the variance of the distances between consecutive suture insertion and extraction points. In this term, we sum up the variance of the distance between insertion points and the variance of the distance between exraction points. Let L^f be the MSE between the closure force at each point along the wound curve and an ideal value. Similarly, Let L^{shr} be the MSE between the shear force at each point of the wound and an ideal value. Note that these target values are defined directly from the surgeon's input to the system.

Assume the suture planning is constrained to a given suture width α. Further assume that sutures are constrained to be orthogonal to the wound curve; therefore it is sufficient to specify the number n of sutures and at which points $0 \leq s_1 \leq \cdots \leq s_n \leq t_{\max}$ along the wound they are placed, to describe a complete suture plan of needle insertion and extraction points. Here, s_i serve as our decision variables (the center of suture i being $w(s_i)$). Additional we impose hard constraints that concern minimum and maximum distances allowed between consecutive insertion and extraction points. We denote these as constraints $A^{\min}, A^{\max}$. Finally, we enforce that sutures should not 'cross.' That is, if we draw a line from corresponding insertion point to extraction point for each suture, it should be the case that none of these lines are crossing. This constraint is denoted as A^c.

Each of the optimization terms $L^{\cdot}$ is weighted with a factor $c^{\cdot}$. The objective is then to

find the sequence of s_i for all $i \in \{1, ..., n\}$ satisfying:

$$\begin{aligned} \min \quad & c^d L^d + \\ & c^{\text{var\_center}} L^{\text{var\_center}} + \\ & c^{\text{var\_ins\_ext}} L^{\text{var\_ins\_ext}} + \\ & c^f L^f + c^{\text{shr}} L^{\text{shr}} \\ \text{s.t.} \quad & \\ & A^{\min}, A^{\max}, A^c \\ & 0 \leq s_1 \leq \cdots \leq s_n \leq t_{\max} \end{aligned} \tag{2.1}$$

Note that the number of sutures n is fixed in this problem; to minimize the loss over all possible suture plans, the problem needs to be solved with various different values of n, which we choose heuristically; see Section 2.4 for details.

The objective function consists of three geometric cost terms (L^d, $L^{\text{var\_center}}$ and $L^{\text{var\_ins}_{\text{e}}\text{xt}}$) and two force model-based cost terms, both measuring the mean squared error. Its solution represents a sequence of suture midpoints which should be close to evenly spaced while also explicitly ensuring good closure forces over the wound. We will set the weights to emphasize L^f (closure forces), with the other components there to provide numerical stability and refinement.

See sections 2.4, 2.4 and 2.4 for a complete mathematical description of the objective function and constraints.

2.4 Methodology

The SP2DEEF algorithm is divided into three distinct phases:

A. *Input*, in which the system queries the surgeon for points along the wound curve, desired suture width, and scaling information;

B. *Optimization with elliptical force model*, in which the system plans a set of sutures to minimize an objective function under constraints, utilizing an explicit model to estimate forces applied by the sutures to the wound;

C. *Adjustment*, in which the surgeon can optionally adjust the suture plan computed by the system.

Input

The interface first collects surgeon input, via a clicking interface, as depicted in Fig. 2.6. The surgeon selects two points on the image and inputs the measured distance between those points as well as the desired suture width.

The program scales the image based on the calibration points provided. The surgeon then clicks a sequence of points on the image tracing the shape of the wound. We then use the scipy interpolation function `scipy.interpolate.splprep` to generate a smooth B-spline that approximates a curve that goes through all points specified. We then generate an initial placement of sutures to 'warm start' the optimization. To that end, the algorithm computes the initial number of sutures by dividing the length of the wound curve by the ideal suture distance γ. It then places the corresponding number of equally spaced suture midpoints along the wound curve. The insertion and extraction points are determined by the perpendicularity constraint between the sutures and the curve as well as by the surgeon-specified suture width.

Optimization

SP2DEEF solves the placement of sutures on a wound as a constrained optimization problem. In order for the healing process to occur, the two sides of the wound must be brought together. Ideally, we would use a detailed model to predict skin deformation after suture placement. However, the mechanical behavior of skin is complex and influenced by the location on the body of the wound and the patient's height, weight, and age, amongst other factors, [12] making full modeling impractical. Instead of this, surgeons usually employ heuristics to plan where to place sutures. These heuristics state that sutures should be orthogonal to the wound, or that they should be spaced evenly and as close as possible to some ideal distance apart. However, relying exclusively on such rules can lead to solutions which leave parts of the wound insufficiently closed, because of insufficient closure force from sutures in the region. Thus, it is desirable to combine surgical heuristics with a model of the forces imparted

by the sutures to ensure that the entire wound is held sufficiently closed by the sutures. This 'hybrid' optimization problem leverages surgeons' experience while also ensuring that each point along the wound receives acceptable closure forces.

We assume that the wound lies on a 2D plane and place the sutures using only a 2D image of the wound as seen by an overhead camera; we assume that the wound has been positioned and rotated to minimize the distortion incurred by perspective effects. The wound is thus represented by a parametric spline computed by interpolating over points clicked by the surgeon on the image, which we denote as a function $w(t) = (x(t), y(t))$ where $w : [0, t_{\max}] \to^2$.

Since this optimization problem is in general nonconvex, we solve it with the SLSQP algorithm [22], which performs constrained optimization using linear approximation.

To find the best number n of sutures, we execute a bi-linear search from an initial naive estimation $\hat{n}$, as for a typical wound there is only a small range of 'reasonable' values of n. We use $\hat{n} = \lfloor \ell/\gamma \rfloor$, as this is the number of sutures resulting from naively spacing the sutures at distance γ along the length-ℓ wound, and is therefore likely to be close to the best number of sutures. We estimate ℓ via linear approximation and compute $\hat{n}$; then we solve the optimization problem (2.1) for all integers n such that $0.5 * \hat{n} \leq n \leq 1.4 * \hat{n}$ and return the suture plan with the lowest loss.

For evaluating the hyperparameter settings we use the same loss as our training loss.

Suture regularity constraints and objectives

The suture width α is given by the surgeon in the input phase of the process and the number of sutures n is selected by the algorithm. The other parameters $\beta_{\min}, \beta_{\max}, \gamma$, are assumed to be constant. For our experiments, we set these values based on consulting an expert in the field of surgery[1] These parameters are depicted in Fig. 2.2. We also include several key conditions for suture placement:

(i) All sutures should cross the wound at their midpoint and be orthogonal to the wound at that point.

(ii) Sutures should have width (distance between insertion and extraction points) of α.

The hyperparameters $\alpha, \beta_{\min}, \beta_{\max}, \gamma, \epsilon$ are set via... Since in general these conditions cannot be perfectly satisfied, some conditions are hard-coded as constraints and others are encoded via penalty terms in the objective function. Conditions (i) (orthogonality of sutures to the wound) and (ii) (suture width) mean that if the suture crosses the wound at some $w(s)$, the insertion and extraction points are uniquely determined (up to swapping); therefore, as discussed above, our decision variables are the points $0 \leq s_1 \leq \cdots \leq s_n \leq t_{\max}$ along the wound at which we want the sutures to cross, with suture i crossing at $w(s_i)$. Since the

[1] we consulted our co-author Danyal Fer, MD, Department of Surgery, University of California San Francisco East Bay.

wound w is represented as a B-spline, it has a well-defined derivative $w'(t) = (x'(t), y'(t))$ at each t. Then, interpreting $w(t), w'(t)$ as vectors in ^{2}, the insertion and extraction points are

$$a^0(s_i), a^1(s_i) = w(s_i) \pm \frac{\alpha}{2} \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} \frac{w'(s_i)}{w'(s_i)} \tag{2.2}$$

Here a^0 corresponds to the insertion point and a^1 corresponds to the extraction point and both are vectors in ^{2}. The side that corresponds to insertion and the side which corresponds to extraction was chosen arbitrarily and can be swapped if desired.

Conditions (iii) and (iv) are coded as the constraints labeled A^{crs} and $A^{\text{min}}, A^{\text{max}}$ respectively. A^{crs} states that for any $i \neq j$, the line segments $(a^0(s_i), a^1(s_i))$ and $(a^0(s_j), a^1(s_j))$ cannot cross; A^{min} states that no two different insertion and/or extraction points can be within β_{min} of each other; and A^{max} states that for any $i \in [0, n]$ and $z(\cdot) \in \{a^0(\cdot), a^1(\cdot), w(\cdot)\}$

$$z(s_{i+1}) - z(s_i) \leq \beta_{\text{max}} \tag{2.3}$$

(no consecutive suture insertion points, extraction points, or midpoints can be more than β_{max} apart).

Condition (v) states that the distance between the sutures should be as close as possible to the ideal suture distance γ while condition (vi) ensures that the sum of variances of the insertion, center and extraction points is low. To measure how well a set of sutures satisfies conditions (v) and (vi), we use mean squared error and variance. To encode the constraint concerning the endpoints of the wound, we add 'phantom' sutures at $s_0 = 0$ and $s_{n+1} = t_{\text{max}}$, giving:

$$L^{\text{idl}}(s_1, \ldots, s_n) = \sum_{z(\cdot)} \frac{1}{n+1} \sum_{i=0}^{n} (z(s_{i+1}) - z(s_i) - \gamma)^2 \tag{2.4}$$

$$L^{\text{var}}(s_1, \ldots, s_n) = \sum_{z(\cdot)} \left(\{z(s_{i+1}) - z(s_i)\}_{i=0}^{n} \right) \tag{2.5}$$

where $\sum_{z(\cdot)}$ indicates summing the function inside three times, with the points $z(s_i)$ being $a^0(s_i)$, $a^1(s_i)$ and $w(s_i)$, i.e. we take the average difference squared from ideal and the variance using the insertion points, the extraction points, and the midpoints and add them up. Conditions (i), (ii), (iii) and (iv) are treated as constraints while (v) and (vi) are encoded as penalty terms and in the objective function and optimized over.

Generalizing the Diamond Force Model

We supplement the suture regularity constraints and objectives with a model that aims to quickly estimate the forces applied by the sutures to the wound, inspired by the diamond force model [9, 10] on a linear wound, in which the force imparted on the wound by a suture has a particular intensity at the crossing point and drops off linearly (to a minimum of 0)

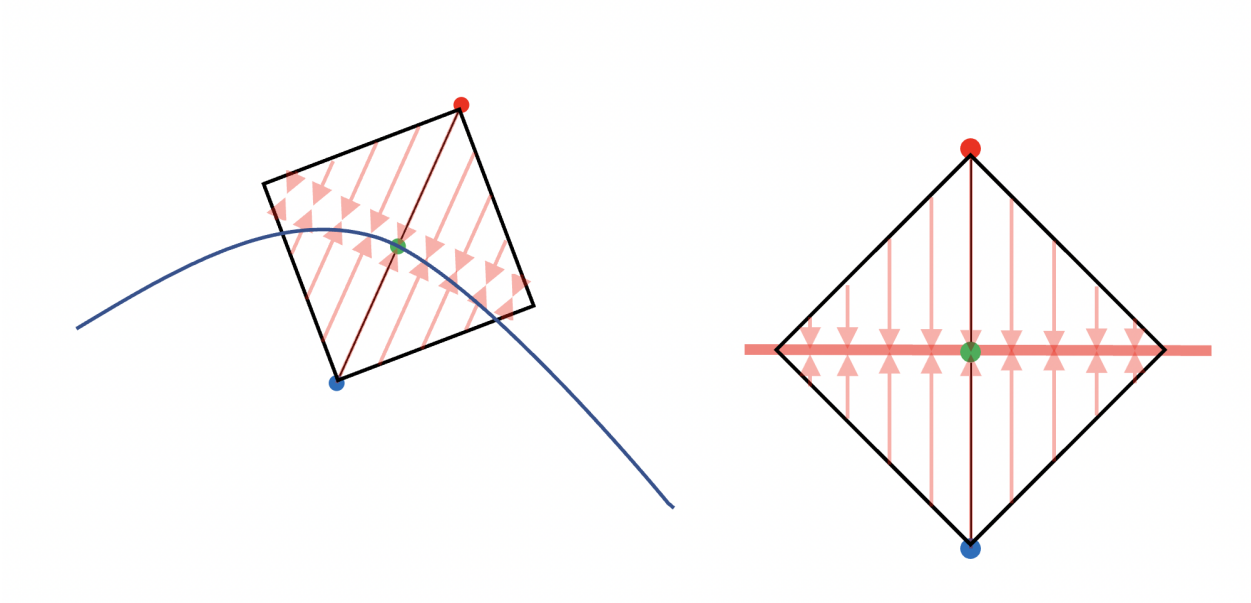

Figure 2.3: **Diamond Force Model.**. The diamond model does not generalize well to non-linear wounds. As shown to the left, the curve pulls away from the diamond line, and thus it is non-obvious how to calculate distances.

according to distance from the crossing point. An interesting feature of this model is that placing each suture at the point where the force from the previous suture drops to 0 yields a constant force across the wound. However, this model only considers linear wounds and does not generalize well when applied to curved wounds, as depicted in Fig. 2.3. We extend this to curved wounds by modifying it so that the forces imparted from an insertion or extraction point on the skin are parallel to the suture and decrease linearly (to a minimum of 0) based on an elliptical norm aligned with the suture. We choose an elliptical norm to represent the fact that the force of pushing on an elastic medium is transferred more strongly to points in line with the force than those to the side. We normalize the magnitude of the forces in our model by letting 1 unit of force be the ideal amount applied to close any given point on the wound, which is the amount applied by a suture to its midpoint. Since by symmetry the insertion and extraction points of a wound exert the same amount of force on the center, in opposite directions (since the suture center is the midpoint of the insertion and extraction points), this means that each insertion or extraction point exerts 0.5 units of force on its respective center. Thus, given a parameter $0 < \varepsilon \leq 1$ denoting the ratio of the shorter axis to the longer axis of the ellipse which defines the norm (to be discussed later), in our model the insertion point $a^0(s_i)$ has a suture-aligned distance from a point $z \in \mathbb{R}^2$ of

$$d^0(z, s_i) = \sqrt{d^0_{\parallel}(z, s_i)^2 + (d^0_{\perp}(z, s_i)/\varepsilon)^2} \tag{2.6}$$

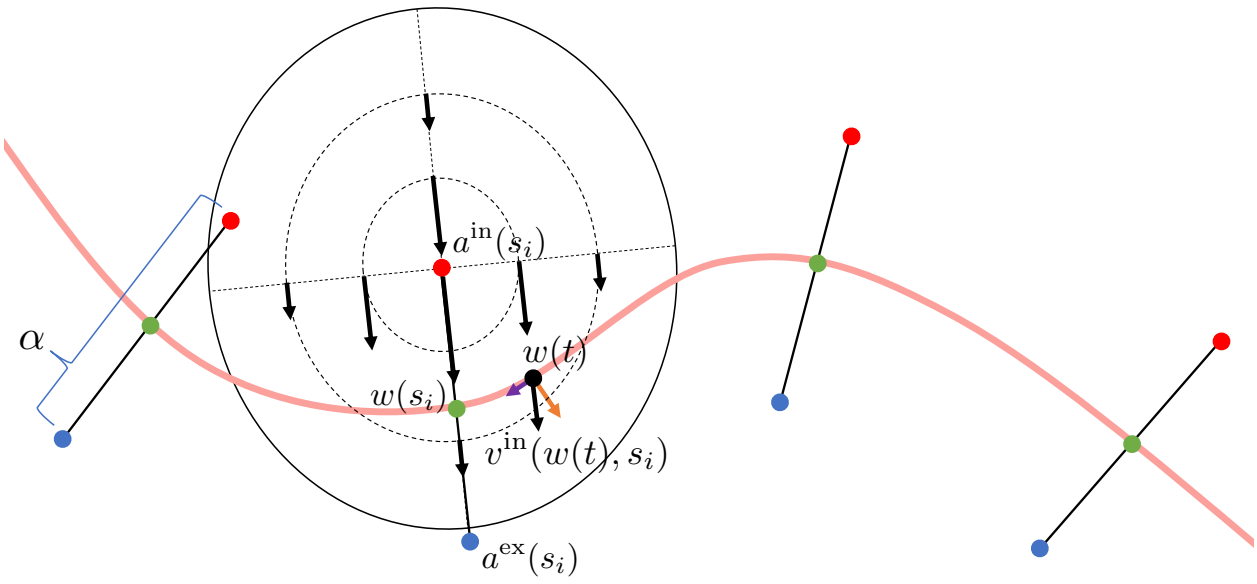

Figure 2.4: Elliptical Force Model. Sutures (width α) with force model around insertion point $a^0(s_i)$ depicted as an ellipse showing the region of nonzero force imparted from $a^0(s_i)$, with forces decreasing linearly from the center, with isocontours of force being ellipses. Purple and orange arrows show shear and closure forces generated at the wound point $w(t)$ by $a^0(s_i)$.

where $d^0_{\parallel}(z, s_i)$ is the distance between $a^0(s_i)$ and z on the axis parallel to $a^1(s_i) - a^0(s_i)$ and $d^0_{\perp}(z, s_i)$ is the distance between $a^0(s_i)$ and z on the axis perpendicular to $a^1(s_i) - a^0(s_i)$; we define the distance $d^1(z, s_i)$ from the extraction point $a^1(s_i)$ analogously. Then, the force exerted on z from $a^0(s_i)$ is the vector

$$v^0(z, s_i) = \frac{1}{2} \frac{\max(\eta - d^0(z, s_i), 0)}{\eta - \alpha/2} \frac{a^1(s_i) - a^0(s_i)}{\alpha} \tag{2.7}$$

where $\eta = \sqrt{(\alpha/2)^2 + (\gamma/\varepsilon)^2}$. The force $v^1(z, s_i)$ imparted by an extraction point $a^1(s_i)$ on z is defined analogously. The suture force model is shown in Fig. 2.4. Note that the force vector is parallel to $a^1(s_i) - a^0(s_i)$ and (since $\alpha = a^1(s_i) - a^0(s_i)$ by definition) has the following properties: (a) $v^0(w(s_i), s_i) = 1/2$ for any i (insertion points, and by symmetry extraction points, exert a force of magnitude $1/2$ on the suture center); and (b) on a linear wound with sutures placed an ideal γ distance apart, each suture center lies on the boundary of the regions in which the previous and next suture exert forces of positive magnitude. We then set $\varepsilon = 0.77$ so that ideally-spaced sutures on a linear wound exert forces which are as constant as possible along the length of the wound. These properties extend the diamond force model to non-linear wounds.

Closure and shear force objectives

Given the model described in Section 2.4 and Fig. 2.4 of how sutures exert force on skin, what does this imply about the quality of the sutures? While the insertion and extraction points are placed so that the suture (and thus the forces it exerts) are orthogonal to the wound at the crossing point, if the wound is not linear then at other points on the wound the exerted force may have both a *closure force* component orthogonal to the wound and a *shear force* component parallel to the wound. The total force exerted by insertion points (which push on one side of the wound) and the extraction points (which push on the other side of the wound) on $w(t)$, are denoted respectively as

$$F^0(t) = \sum_{i=1}^{n} v^0(w(t), s_i) \text{ and } F^1(t) = \sum_{i=1}^{n} v^1(w(t), s_i) \tag{2.8}$$

The total closure force at point $w(t)$ on the wound is:

$$f^{\mathrm{c}}(t) = (F^0(t) - F^1(t))^{\frac{a^1(s_i)-a^0(s_i)}{\alpha}} \tag{2.9}$$

(note the $1/\alpha$ term to normalize the suture width). This value is normalized so that the ideal value is 1 at every $w(t)$.

To get the total shear force at a given point t on the wound, we take the components of the forces parallel to the wound:

$$f^{\mathrm{s}}(t) = (F^0(t) - F^1(t))\begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}^{\frac{a^1(s_i)-a^0(s_i)}{\alpha}} \tag{2.10}$$

Since shear forces do not hold the wound together but may cause misalignment, the 'ideal' magnitude of the shear force is 0 at every t.

Force closure objective

We take m points $w(t_1), \ldots, w(t_m)$ and use the model above to estimate the closure and shear forces at these points. The average closure force penalty is then

$$L_{\mathrm{c}} = \frac{1}{m}\sum_{j=1}^{m}(1 - f^{\mathrm{c}}(t_j))^2 \tag{2.11}$$

(penalizing deviation from the ideal value of 1) and the average shear force penalty is

$$L_{\mathrm{s}} = \frac{1}{m}\sum_{j=1}^{m} f^{\mathrm{s}}(t_j)^2 \tag{2.12}$$

(penalizing deviation from the ideal value of 0). As with the regularity objective function, these components of the objective function are given weights c_{c} and c_{s}, respectively.

Parameter settings

We take α (suture width) as input from the user while we chose the values of $\beta_{\min}$, $\beta_{\max}$ (min and max distances between sutures) and γ (ideal distance between sutures) by consulting an expert in the field of surgery.

We developed the constraints using surgical suturing guidelines, setting $\beta_{\min} = 2.5$mm between two sutures as consulted by an expert surgeon[2] and $\beta_{\max} = 10$mm between consecutive sutures so as to not leave parts of the wound unclosed.

Adjustment

Finally, we provide a GUI for the surgeon to interact with. They are presented with the suture points, superimposed over the wound. The surgeon is able to drag the points to the desired place. See the top right of Fig. 2.6 for a view of the Adjustment Visualizer; the surgeon manually clicks and drags points in this window to adjust the placement.

[2]We consulted our co-author Danyal Fer, MD, Department of Surgery, University of California San Francisco East Bay.

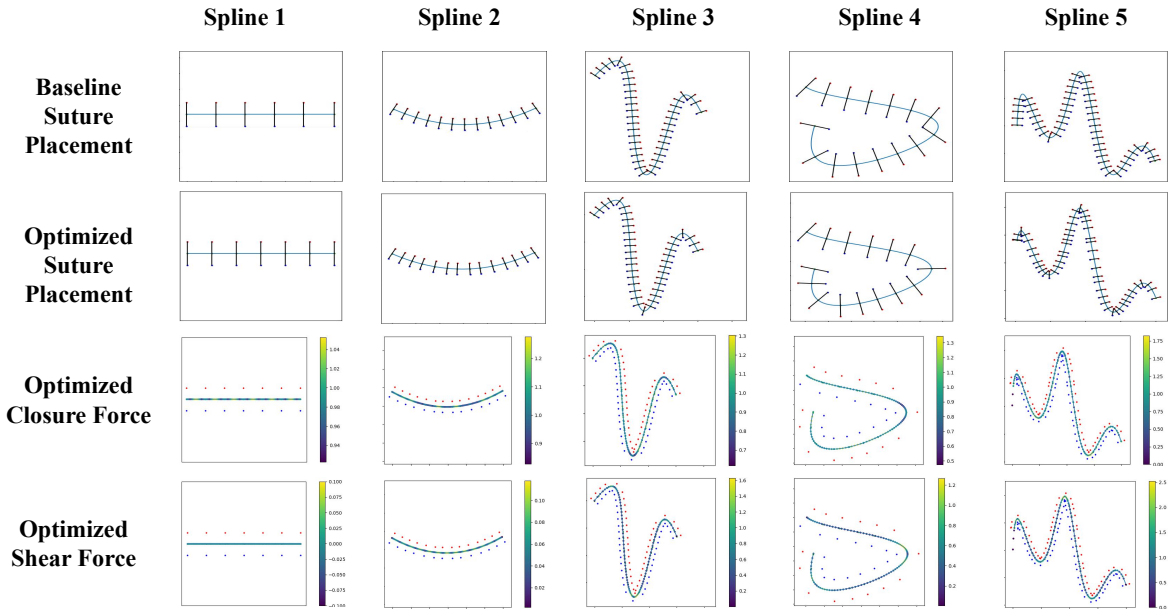

Figure 2.5: Results analyzing the suture placements for 5 synthetic splines. Top two rows present the difference in the sutures placed using the baseline, as compared to the optimization algorithm. The bottom 2 rows present the shear forces and closure forces created by the optimization suture placement

2.5 Experiments

Synthetic Splines

Our loss function contains weights on all 5 loss terms, which we manually tuned to the values: $c^d = 1$; $c^{var_center} = 12$; $c^{var_ins_ext} = 6$; $c^f = 15$; $c^s = 5$.

Baseline Algorithm: We evaluate a baseline algorithm, which evenly places sutures along the curve spaced at the ideal distance, on the splines. This can be implemented by starting at one end of the wound, travelling the ideal distance along the wound, placing sutures at increments of the ideal distance until the end of the wound.

We tested both SP2DEEF and the baseline on 5 splines generated manually. To evaluate the robustness of the algorithms we created simple splines and splines with a higher degree and curvature. The suture placement points generated by the algorithms are depicted in Fig. 2.5. Our results show that the suture placements generated by SP2DEEF are relatively consistent, robust and well spaced, especially for the simpler splines. SP2DEEF also deals well with sharp curves (depicted in Spine 3) as it places a higher number of sutures close to

Table 2.1: Suture Placement Results

Loss Type	Algorithm	Spline 1	Spline 2	Spline 3	Spline 4	Spline 5
Variance - center points	Baseline	0.00	**0.02**	2.06	2.79	**0.77**
	SP2DEEF	0.00	0.62	**1.51**	**1.24**	24.92
Variance - insertion / extraction points	Baseline	0.00	**0.05**	**28.12**	18.50	**47.66**
	SP2DEEF	0.00	1.25	28.40	**13.65**	87.12
Ideal distance loss	Baseline	8.48	**5.34**	106.50	86.68	**185.58**
	SP2DEEF	**4.50**	9.01	**64.88**	**25.95**	527.79
Closure Force loss	Baseline	3431.70	9410.64	9828.90	7776.41	8283.97
	SP2DEEF	**0.61**	**5.68**	**0.01**	**0.00**	**9.54**
Shear Force loss	Baseline	0.00	54.86	**29.00**	104.66	**17.52**
	SP2DEEF	0.00	**0.55**	39.13	**32.00**	113.43
Total loss (weighted)	Baseline	51483.98	141439.78	147878.44	117400.61	124827.93
	SP2DEEF	**13.65**	**111.9**	**449.2**	**282.73**	**2059.8**
Number of Sutures	Baseline	6	13	40	15	51
	SP2DEEF	7	14	39	15	53

the point of curvature.

SP2DEEF tended to choose a different number of sutures, when compared with the baseline. An optimal placement over an optimal number of sutures results in a lower closure force, while other terms tend to remain the same: the closure force that each suture exerts on the wound decreases sharply as the curve turns away from being perpendicular to the wound. To maintain sufficient closure force, it is necessary to increase the number of sutures. The baseline algorithm was not able to provide consistent closure force, as it is able to vary neither the number of sutures nor the placement to account for curvature.

While it reported high closure force losses, the baseline did well according to the variance and ideal distance metric, since it was initialized to be evenly spaced. The variances for the baseline aren't always 0, because we used a greedy algorithm which approximately placed sutures evenly. Furthermore, the insert and extract points may have additional variance on curved segments of the wound even if the centers are approximately evenly spaced.

One major improvement SP2DEEF saw was that it always avoided sutures crossing over and heavily penalized cases in which the sutures were placed too close to each other, in contrast to the baseline algorithm, which did both of these.

Overall, SP2DEEF outperforms the baseline; we believe that closure force is the most important metric as the main purpose of sutures is to provide closure forces to close a wound. Our algorithm finds a placement almost as well spaced as the baseline, with similar shear forces, while making sure that all the normalized closure forces are very close to the ideal. Although it does not outperform the baseline in the variance (and the shear sometimes), it makes up for this by returning a placement that optimizes the closure force.

Physical Experiments on Chicken Skin

We tested SP2DEEF in a physical experiment using chicken (thigh) skin. To find the most realistic wounds possible, we had a surgeon cut two wounds into the tissue with a scalpel[2]. These two wounds, as can be seen in Fig. 2.1, were chosen by the surgeon for their relevance to clinical wounds seen in practice, as well as their difficulty. As a baseline, we had a surgeon suture the wound as they saw fit. We then fed an image of the wound without sutures into our pipeline, clicked points to trace the wound shape, and recorded the optimized placement of the sutures, the output of our optimizer. We manually marked this placement on the phantom with ink, and had the surgeon implement the autonomous placement by suturing on top of the ink markings with USP Size 2-0 thread and a suture needle type GS-22 as depicted in Fig. 2.1.

Our placement was able to achieve wound closure on the phantom: we consulted an expert surgeon[2] who evaluated the sutures as equal or better than a human surgeon. In the test case, having SP2DEEF's placement marked on the skin helped the surgeon place sutures with consistent spacing throughout the whole wound. In contrast, in the baseline case, with no visual guides, the Surgeon stitches developed uneven spacing between suture locations as the wound was deformed during the suturing process.

The optimization process takes time on a scale of 10-30 seconds for a low number of sutures (6-10), and up to a few minutes for a large number of sutures (30-50). The runtime appears to scale closer which is costly for an operating room environment. This is a result that leaves room for improvement, as every minute spent in the operating room is very costly in terms of machine use and medical personnel time. The operation is not very memory-intensive, and does not scale exponentially with the size of the problem.

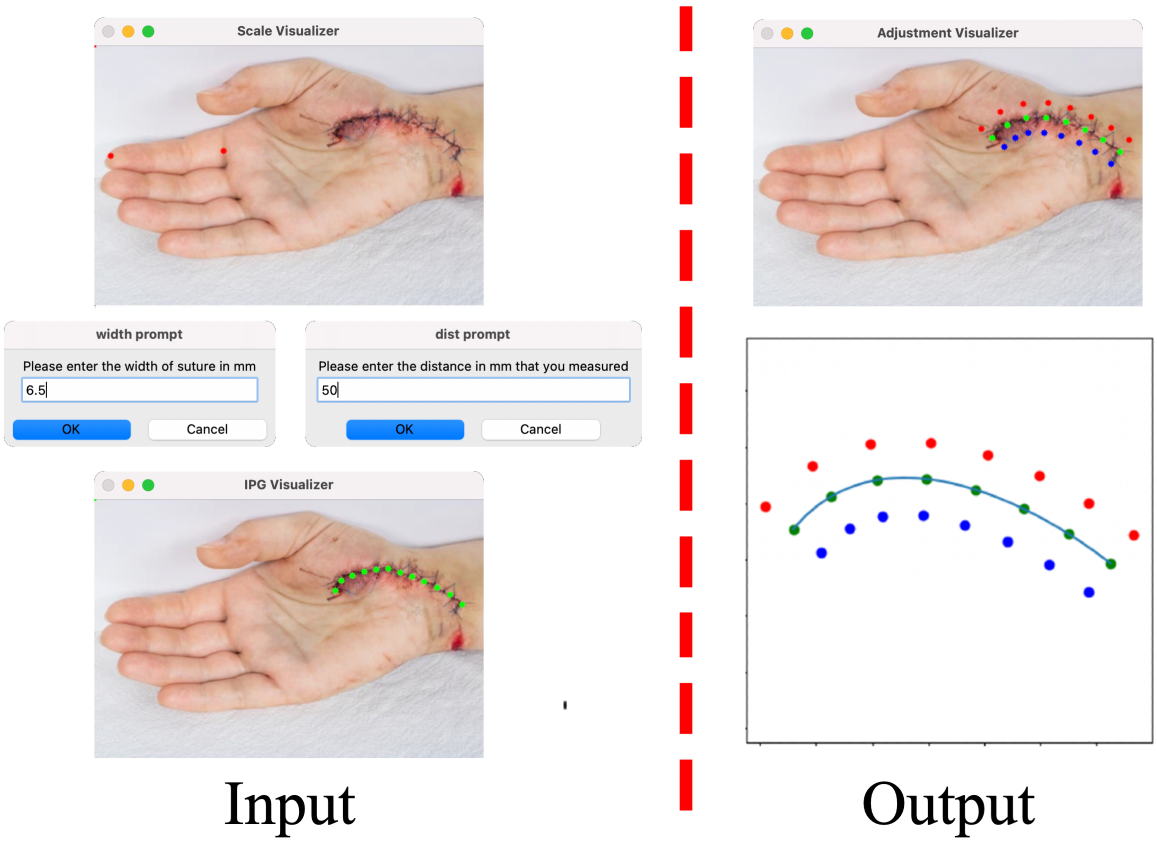

Figure 2.6: **The full autonomous pipeline for optimizing surgical suture placement.**. Input: Surgeon selects two points on the image (finger length) and inputs the measured distance between those points as well as the desired suture width. The surgeon is further asked to click points along the wound. Output: The wound fitted as a Bézier curve (green) with the result of the suture optimization (red and blue points). The suture plan is overlaid on the original wound image.

2.6 Limitations

In our formulation, the wound is represented as a spline, and therefore the representation cannot have sharp corners (or branches). It would not be difficult to make a special user interface feature and a curve fitting formulation such that we can accurately model such curves. For the optimization process, based on guidance from our co-author Danyal Fer, MD., the optimal way to suture a corner is to place the suture at the corner, such that it bisects the angle created by the corner. This is an intuitive solution. Based on this advice, assuming sutures on either side of the corner don't interact with each other, the problem of suturing the wound becomes subdivided into two independent problems of suturing each half of the wound, and each can be independently solved. If sutures from opposite sides of the corner do interact / intersect, an alternating optimization process may be utilized, or a joint optimization with special constraints for sutures that have the potential to intersect near the corner.

No information about the wound depth or width is incorporated. The wound width sometimes varies in practice, and there are even cases where one flap of skin is longer than the other, especially in cases where parts of the skin have to be debrided. Our representation of the wound as a line is inaccurate in this case, and a potential solution is to model both the left and right edge of the wound with a spline. Then, we can plan sutures that have two spokes: one jutting out from the right half, and one from the left. They will be connected together into a single, straight line segment when the skin is pulled together. New methods will have to be devised to optimize in this new framework.

SP2DEEF assumes a fixed suture width as opposed to being able to modify it for particular sutures. In general, the width of the suture should be relatively constant, so it is not a wildly inaccurate assumption. However, for tight curves, is is sometimes optimal to decrease the width of the sutures so that there is more space between successive insertion or extraction points; there may also be cases where it is necessary to do so to avoid suture crossings.

The explicit model of suture forces is only a rough approximation of how skin behaves under tension. In suturing, we are attempting to replace the inherent skin tension lost due to the breaking of the skin, and add a slight amount more such that the two wound flaps pull each other together. The inherent skin tensions varies from location to location on our skin as well as from individual to individual. So, even if the closure force remains constant, the amount of closure force that we need varies according to the individual and location of the wound on the skin. In addition, the closure force may not be constant due to skin curvature, varying thickness, etc. The skin is also multi-layered, so the epidermis, dermis, and subcutaneous layers have very different material properties which we do not account for.

No objective, external measure of suture placement quality was used to evaluate the algorithm. We are not sure of the existence of such a widely-accepted, scientifically reputable measure.

The sutures may be re-planned during the operation due to the wound changing shape when partially sutured. We do not have any re-planning process in our framework, but we

can modify it to do so in future work.

The optimization process takes time on a scale of 10-30 seconds for a low number of sutures (6-10), and up to a few minutes for a large number of sutures (30-50). The complexity of the problem scales somewhat non-linearly with the number of sutures, it appears to be more than linear but less than exponential. This is not surprising for a quadratic program, which is NP-hard in worst case, but our case may not be the worst in practice. Placing 30 or more sutures poses a large combinatorial set of choices to choose from, and a general-purpose optimization process uninformed about common sense practice for suturing, i.e. the SLSQP SciPy solver we use, may proceed slowly as it will not have "intelligent" non-gradient-based heuristics to deal with corners and curves of the wound, and may involve some level of brute-forcing many of the possible suture placements and using the gradient to adjust. A few minutes is costly for an operating room environment.

2.7 Conclusions

Prior work on the suture planning problem focused primarily on achieving evenly placed sutures on known wound shapes under kinematic or workspace constraints. This suggests the possibility of combining such constraints with the wound-focused constraints and objective developed in this work, for instance by modifying the objective function to include a consistency measure and also account for the difficulty of executing the planned sutures.

Having the surgeon efficiently identify the wound avoids the uncertainty and instability that comes with automated wound-detection from images, and having the surgeon be able to provide adjustments to internal parameters such as wound width on a case-by-case basis allows for a higher probability of successful placements in various medical environments which may be beyond the scope of current automation to recognize and adapt to. The final adjustment period, where the surgeon can optionally adjust the autonomous placement, provides the groundwork for future work into a time-efficient interactive human-machine collaborative process for the placement of sutures.

The surgeon evaluated SP2DEEF's placement as equal or better than their own, and also remarked that the physical ink markings on the skin decreased their mental load when suturing and helped them place sutures more evenly than they might have been able to do otherwise. Thus, SP2DEEF's autonomous placement provides the groundwork not only for fully autonomous suturing systems, but also for real-time assistance of manual suturing, by marking suture plans onto real skin with ink or, more efficiently, with light in real time.

SP2DEEF operates on 2-dimensional wounds. However, real wounds occur in 3 dimensions, so future work may include a generalization of SP2DEEF to 3 dimensions, where the wound shape and sutures are represented in real space. This will likely require the usage of a depth image of the scene, and an effective, noise-resistant surface interpolation method to determine the position of the skin and the wound at various points.

The algorithm can take a few minutes to run for large wounds. Perhaps the speed could be improved by the use of deep learning warm-starting, or other medically-aware intelligent warm-start heuristics.

The process currently runs in Python locally, but we are also in the process of creating a website for this whole pipeline so that the tool is accessible from anywhere in the world.

Chapter 3

PolyPoD: An Algorithm for Polyculture Seed Placement

3.1 Introduction

The PolyPoD project was done in conjunction with Shrey Aeron, Anrui Gu, Harshika Jalan, Simeon Adebola, and Professor Ken Goldberg. I conducted all of the ideation and development for the PolyPoD algorithm by myself. Simeon helped with ideation about which features to include in the final website version, and target audience for the algorithm. Shrey led development of the website and contributed to both back-end and front-end elements. Harshika and Anrui contributed to back-end and some elements of the front-end for the website. Prof. Goldberg oversaw the project and was our faculty advisor.

Polyculture farming has a number of advantages over monoculture farming, including reduced pests, better resource (light, water and nutrient) efficiency, soil preservation, companionship, aesthetics and higher overall yield [36, 8, 17, 23]. Polyculture farming is more sustainable [24] and similar to how plants grow in nature [8], but requires more planning and tending than monoculture. A key question is deciding how to locate the seeds given that some plant types grow much larger than others and need more space?

In this work we present a polyculture seed placement algorithm: PolyPoD, that allows users to draw a polygonal boundary space to have a planting arrangement that includes all the plants they want and uses all the space efficiently (Fig. 3.1). Seed placement plays a significant role not only in garden aesthetics but also in plant-plant interaction and symbiosis, which are key for polyculture planting [4].

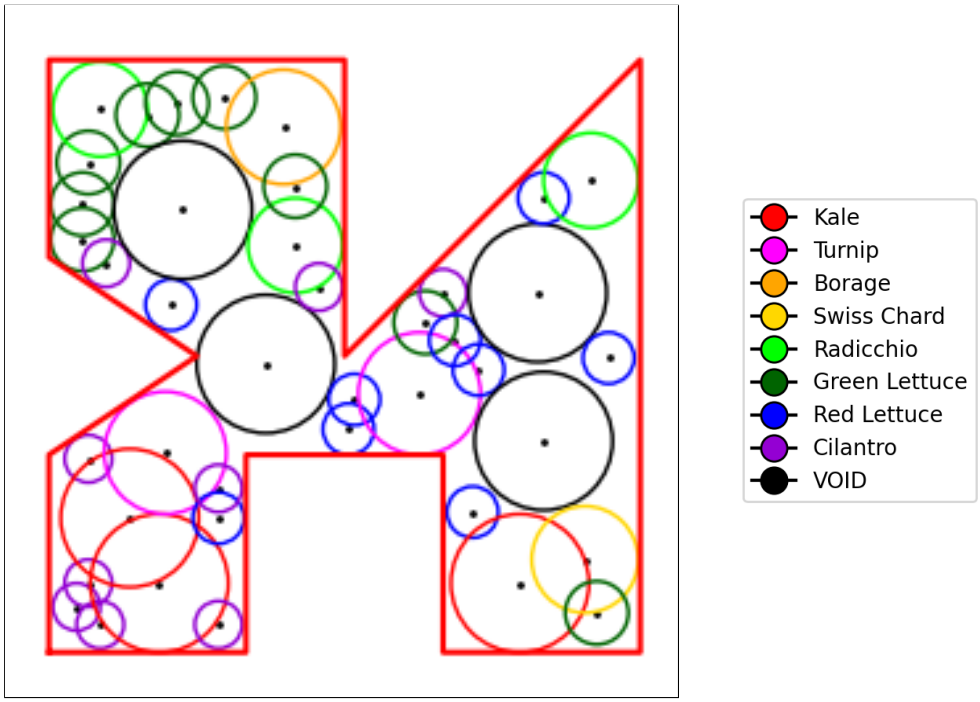

Figure 3.1: **Seed Placement.** The plants are colored by plant type; voids–desired spaces without any seeds–are colored black. shrinks the bounding polygon by the plant radius to determine the appropriate planting area. Note that the planting area example here is non convex.

3.2 Related Work

Precision Agriculture

Precision Agriculture makes use of sensors and analysis tools to reduce the use of water and pesticides, make better decisions, and improve crop yield. Trogo et al. [45] combines Decision Support System for Agrotechnology Transfer (DSSAT) with Automated Weather Station (AWS) sensors so that farmers can consider climate factors during the planting season. The interface allows them to run DSSAT by sending an SMS inquiry and identify the best dates to plant and harvest crops. Other prior work developed an autonomous agricultural robot that waters the plants based on soil moisture and pH level and captures pictures to identify diseases in plant leaves in real time [20]. Dong et al. [11] uses high resolution remote sensing images and machine learning methods to map the spatial distribution of soil nutrients in plain and mountainous areas.

Plant Companionship Scoring

It is critical to encourage the symbiotic relationship between a carefully selected group of plants, which requires managing complex information about the plant species as well as their relationship with the environment in terms of factors like nutrient utilization and chemical interactions [35]. Avigal et.al [3] determines companionship scores by empirical observation, and places the seeds of plants with a positive symbiotic relationship close to each other and vice versa for plants with a negative relationship. PolyPoD uses the AlphaGarden companionship metric by clustering plants with positive companionship, enforcing them with utility functions and expanding it to account for more objectives.

Poisson Disk Sampling

Poisson disk sampling is a geometric algorithm for generating tightly-packed points that are all some minimum distance from one another (Fig. 3.2). In [29], the authors enhance Poisson-disk sampling with three generalizations. [7] proposes a new algorithm that takes O(N) time to generate N Poisson disk samples where sample candidates are drawn only from a region near existing samples, and rejection sampling is used to find permitted regions. While this algorithm is fast [13], it does not allow variable radii. Mitchell et al. [29] is an example of an algorithm that allows variable radii.

Seed Placement

Seed placement has a significant impact on the ability of crops to grow consistently and their yield potential. Several optimization algorithms have been explored in prior works. The seed placement algorithm described in prior work [3] has undesirable properties such as clustering plants close together, planting near borders, and allowing for overlap between

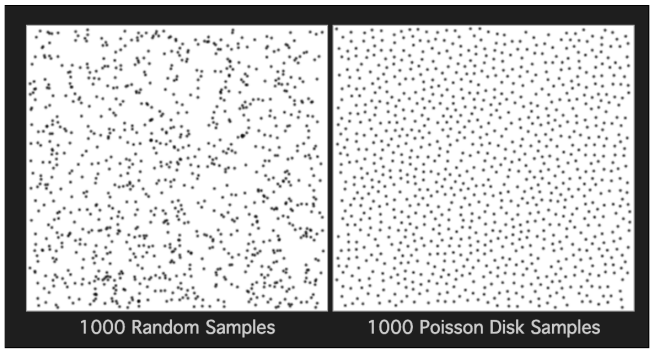

Figure 3.2: **Samples from Random vs Poisson Distribution.** Sampling from the random distribution causes clumping, whereas samples from the Poisson distribution are more evenly spaced [26].

plants of the same type. This leads to over-competition, under-utilization of garden space, and plant segmentation difficulties.

The invasive weed algorithm proposed by Mehrabian and Lucas [28] uses the observation that weeds can colonize a territory unless their growth is carefully controlled. In addition, in the invasive weed algorithm, if the maximum number of plants in the colony has been exceeded, the population size is reduced by eliminating the least fit plants so that best plants are not lost between iterations. Premaratne et al. propose the paddy field algorithm [33] which operates by dispersing seeds in the given space. The fittest plants give rise to the greatest number of seeds and out of these, depending on the number of neighbors of the plant, only a small subset become viable due to pollination. GrowVeg [25] is a seed placement planning website that draws on surveys of companionship research to form a large database of companionship planting information, which yields companionship recommendations to users. However, they mostly support straight line rows and do not offer much automation.

Seed placement has also been formalized in a simulation context. Gou et al. [18] proposes a simulation environment supporting the analysis of various seed placements but restricts to the strip intercropping setting. Simulation methods are transferrable to real-world agricultural policy: [5] minimizes the Mean Absolute Error (MAE) between simulated and real world individual plants.

AlphaGarden

The project [4, 3, 1, 5, 34, 2] seeks to develop an automated system for polyculture farming using sensors, customized pruning tools, and computer vision and deep learning based policies for irrigation and pruning. In addition to the system itself, practitioners in the field can also take advantage of relevant software and subcomponents of AlphaGarden. [34] and [2] used the PolyPoD algorithm as part of the pipeline.

3.3 Problem Statement

Given a polygonal boundary shape with n plants of K types with up to n_k of each type, including a type called 'void' which must be placed with no overlap allowed, to mitigate overcrowding.

Each plant type has a Growing Radius R_k, which represents the plants' average radius and an Inhibition Radius r_k which represents the radius within which overlap from other plants' growing circles is not allowed. We define an overlap factor $o_k \in [0, 1]$, which controls what proportion of plant radius can overlap with another plant. o_k defines a relationship between R_k and r_k:

$$r_k = o_k * R_k$$

We want to place all plants in the garden space such that no circle of radius R_k around any plant center intersects with a circle of radius r_j centered at any other plant center, i.e. there is no overlap between the Inhibition radius of one plant and the Growing radius of any other plants.

S is number of points in the grid: a standard grid is placed within the garden; the user can select the grid cell size. n is number of plants, as defined above. Let G_{grow} be the Growing Radius Proximity Grid which contains the distance to the nearest growing radius, and $G_{inhibit}$ be the Inhibition Radius Proximity grid which contains the distance to the nearest inhibition radius. One can access the entry corresponding to point p via $G_{grow}(p)$ and $G_{inhibit}(p)$.

3.4 PolyPoD Algorithm

We present Polyculture Poisson Disk (PolyPoD), a greedy seed placement algorithm.

Enforcing the Variable Radius Poisson Disk Distribution; Algorithm Overview

PolyPoD discretizes the garden space into a grid: one unit on the grid represents $1cm^2$ of space. PolyPoD first places voids (see 3.4) and then places all plants in the garden in descending order of average radius. Once a plant is placed, it does not move. PolyPoD is a greedy algorithm as it places plants optimally based on the current state, and then moves on to the next plant without tracing, which is suboptimal.

PolyPoD uses the Variable Radius Poisson Disk distribution to ensure each plant of type k is at least r_k away from other plants. We develop a novel algorithmic technique that guarantees satisfaction of the Variable Radius Poisson Disk constraints, as well as a data structure called the Proximity Grid which reduces the runtime from $O(SN^3)$ to $O(SN)$.

Two data structures are used: the Inhibition Radius Proximity Grid and the Growing Radius Proximity Grid. These are the same shape as the garden; each grid has one entry corresponding to each discrete point in the garden. The grids are initialized to have infinite value in each cell. The algorithm starts with a list of all points in the garden l_{legal}. The current plant type to be planted is k. To enforce the Variable Radius Poisson Disk constraints, PolyPoD iterates over all points in l_{legal} and filters out points p which do not satisfy:

$$G_{grow}(p) > ((1 - o_k) * r_k)$$
$$G_{inhibit}(p) > r_k$$

The algorithm also filters out points that do not meet other constraints, described in 3.4.

After filtering out points which do not meet the Variable Radius Poisson Disk constraints as well as the constraints in section 3.4, l_{legal} consists of all legal planting locations in the garden. By default, to choose a location to plant in, the algorithm chooses a point uniformly at random from l_{legal}. However, upon specification, the algorithm can also use a user-defined utility function to select locations according to some non-uniform function of the locations.

Once the algorithm chooses a position p to place the plant of type k, PolyPoD adds (p, k) to a list $x_{planted}$ of plant locations in the garden. PolyPoD updates the value of each grid cell in the Growing Radius Proximity Grid such that each cell contains the distance to the nearest Growing radius (measured by the distance from the cell point to the closest point on the other plant's radius). It also updates the value of each grid cell in the Inhibition Radius Proximity Grid such that each cell contains the distance to the nearest Inhibition radius (Fig. 3.3). This is achieved by: for each point s in the garden, setting the values:

$$d_{grow} = max(dist(s, p) - r_k, 0)$$
$$d_{inhibit} = max(dist(s, p) - (1 - o_k) * r_k, 0)$$

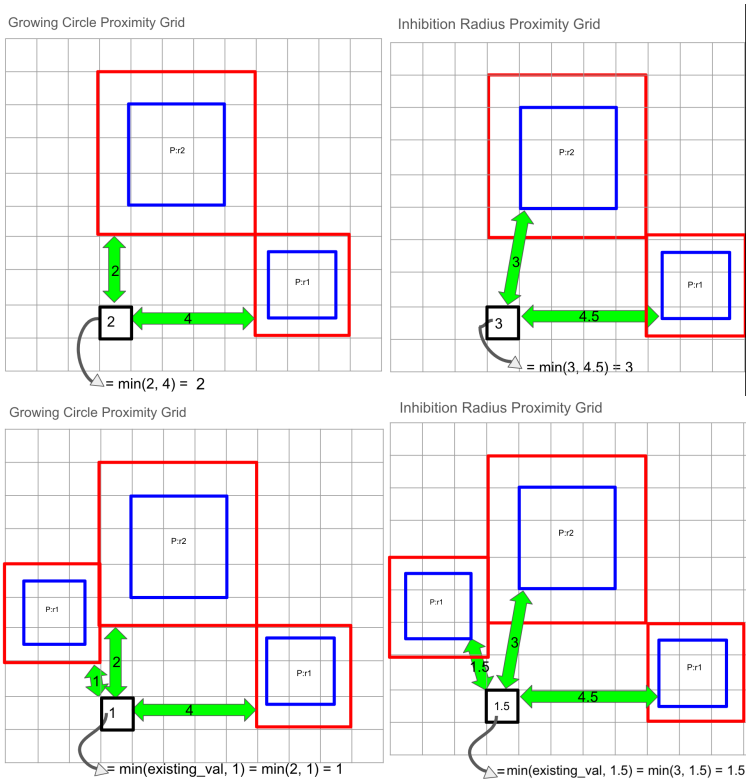

Figure 3.3: **Proximity Grids Before and After Placement.** Top: two plants are already in the garden. Bottom: a third plant has been planted. Growing Radius (red) and Inhibition Radius (blue) Proximity Grids are shown, as are the distances to the nearest Growing Radius (left) and Inhibition Radius (right).

where
d_{grow} is distance from s to growing radius of plant k
$d_{inhibit}$ is distance from s to inhibition radius of plant k.
Then,
If $d_{grow} < G_{grow}(m)$, then set $G_{grow}(m) = d_{grow}$.
If $d_{inhibit} < G_{inhibit}(m)$, then set $G_{inhibit}(m) = d_{inhibit}$

If one uses a naive approach that keeps track of only $x_{planted}$, a list of locations planted in the garden, then when filtering out points from l_{legal}, one has to check all locations in $x_{planted}$ to see if a constraint is violated, and the runtime grows by a factor of S^2. Using a proximity grid replaces this expensive operation with a constant-time reference to the grids. For all N plants planted, the proximity grid version of the algorithm has to perform a constant-time check over all S locations in the garden, yielding a runtime of $O(SN)$. For all N plants planted, the naive list-based version of the algorithm must perform a $O(N^2)$ check through $x_{planted}$ over all S locations in the garden, thus its runtime is $O(SN^3)$. However, PolyPoD

runs in $O(SN)$.

Additional Constraints

PolyPoD ensures that an additional set of constraints are met as well. It ensures that no plant radius R_k overlaps with the voids, and that the voids are sufficiently spaced apart from each other and from borders. It is possible to have different overlap (o) values for pairs of plant types, and PolyPoD uses this option to set the overlap of the voids with any other plants to zero, regardless of the overlap values of other plants.

PolyPoD can also optionally ensure that no plant overlaps with a plant of the same type. This can be achieved by setting the o value of one plant with another plant of the same type to zero. Preventing overlap of similar types of plants can greatly improve automated plant segmentation, because segmentation algorithms can easily confuse one type of plant with another if plants of the same type overlap [34].

PolyPoD also ensures that no plant radius R_k crosses any border. The border polygon b can be of any shape. Thus, PolyPoD uses the geometry library Shapely [16] to shrink b by the plant radius R_k to produce b_{shrink}. Then, we can plant the center of the plant anywhere in b_{shrink} and ensure that the plant's radius does not cross b.

Utility Function

As a special case, to select which location to plant from l_{legal}, PolyPoD can, upon specification, use a user-defined utility function $u(p, k, M, G_{grow}, G_{inhibit}, x_{planted})$ which takes a proposed location p for a plant of type k in a garden whose state is described by G_{grow}, $G_{inhibit}$, and $x_{planted}$, and some relation between plants described by the matrix M. Then, instead of selecting a location uniformly at random from l_{legal}, PolyPoD selects location p with probability proportional to $u(p, ...)$. While one can design many different utility functions that achieve different planting effects, in this paper we focus on a class of utility functions that cluster plants together. They enable companionship planting and can cluster plants of the same type or cluster plants in pairs (Fig. 3.4).

To enable companionship planting, M is assigned to the companionship matrix C, where C_{ij} describes the companionship relation between plants i and j as described in [3]. $C_{ij} = 1$ if plants i and j have a positive companionship relation, 0 if they have a neutral companionship relation, and -1 if they have a negative companionship relation. The utility function is then, for planting a plant of type i at location p, with $dist(a, b)$ being the Euclidean distance function:

$$\left(\sum_{(x,j)\in x_{planted}} \frac{C_{ij}}{dist(x,p)^2}\right)^h$$

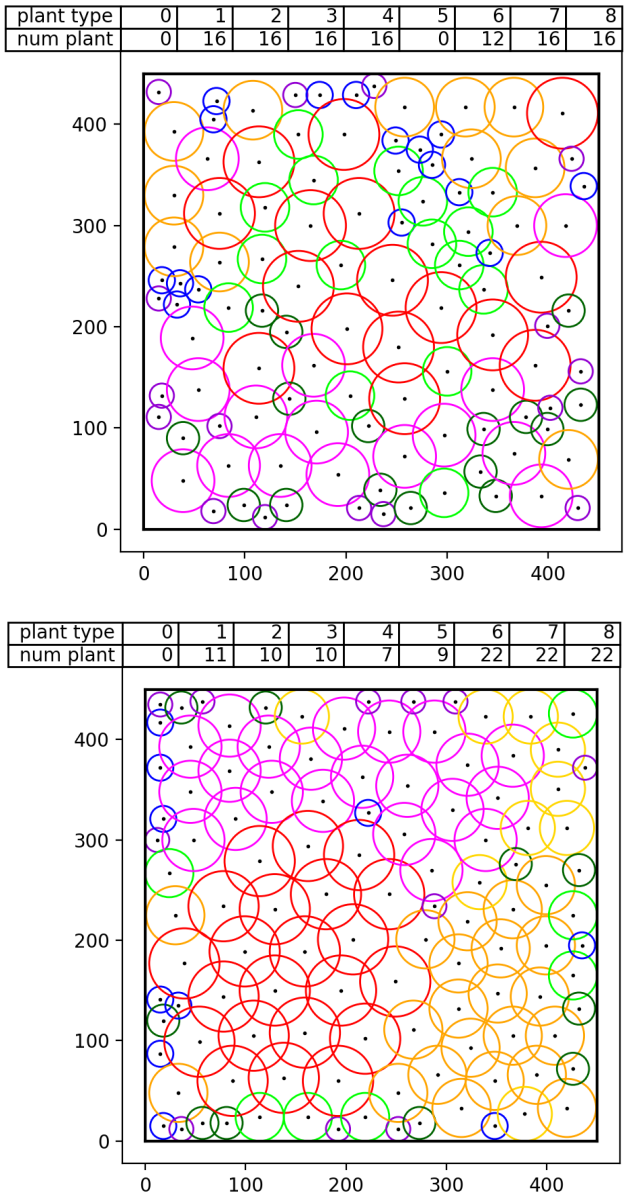

Figure 3.4: **Clustering Strategies.** can cluster according to plant companionship scores. Top: paired clusters of plants. Bottom: same-plant clustering ensures the same type of plants are close together.

The exponent h can be adjusted to vary how drastically companionship affects the probability distribution: lower values yield a distribution closer to uniform at random, and higher values yield distributions that are skewed towards companionship-favorable locations.

3.5 Utility Function, Clustering, and Companionship Planting

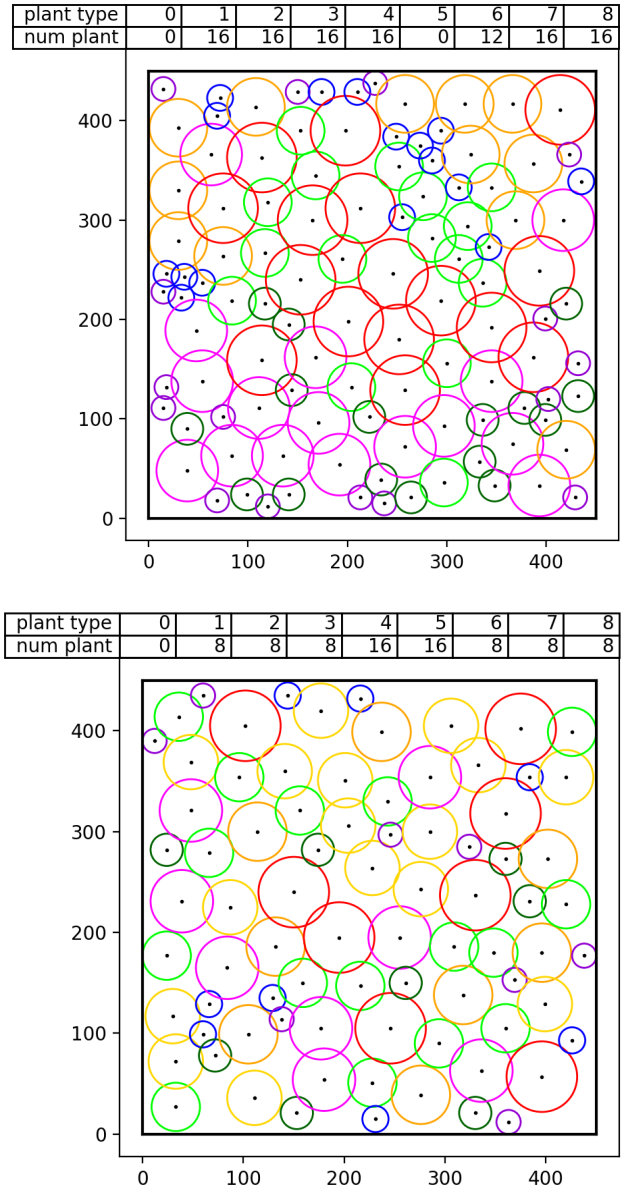

Figure 3.5: **Varying Densities.** Top: High Density Distribution with medium overlap and an uneven 3-dominant distribution. Bottom: Low Density Distribution with low overlap and an uneven 2-dominant distribution.

3.6 Garden Layout Features

PolyPoD supports a few garden layout features that provide customizability. Some polyculture farmers care about the aesthetics of their garden and may have other constraints that require placement adjustments, such as increasing the density of plants if the farmer is

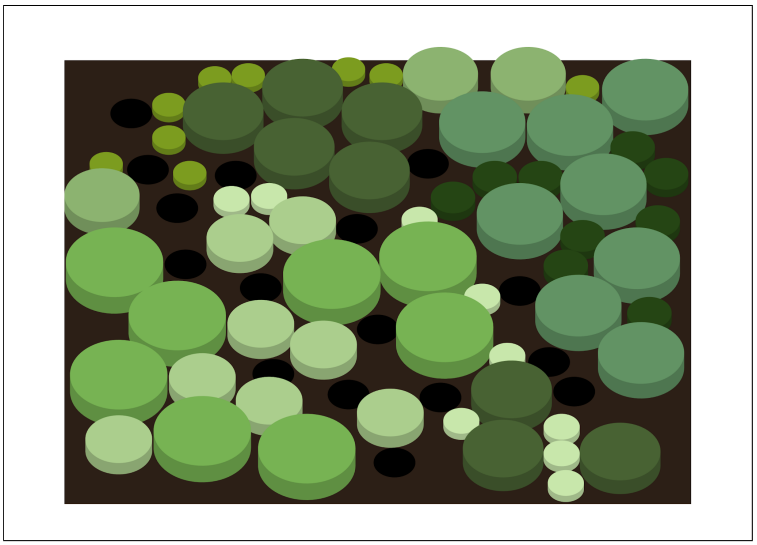

Figure 3.6: **Placement of Voids.** The sizes and number of voids (black) can vary according to user preferences. Garden represented in 3-dimensional form with plants as cylinders.

space-constrained relative to their garden yield requirement.

The flexibility of the utility function defined in IV-C enables the clustering of plants of the same type together by inputting $M = I$ into u, where I is the identity matrix. Another way to cluster plants that further supports companionship planting and provides visual appeal is the paired-plant clustering strategy, which clusters pairs of plants together. This is achieved by defining a matrix P where $P_{ij} = 1$ if plants i and j are paired, and inputting $M = P + I$ into u.

One can adjust the density of the garden by scaling the total number of each plant by a constant $d \in [0, 1]$: 0 represents an empty garden and 1 represents a full garden. See Fig. 3.5 for two seed placement examples of varying densities.

In a full garden, it is often desirable to have some empty space that mitigates overcrowding, creates visually aesthetic gaps, and provides useful features such as walkways. Fig. 3.6 shows a garden with numerous small circular voids. The voids serve to define sections of the garden devoid of plants, and the algorithm assumes that plants are present outside of these voids. The algorithm also supports the inverse framework, where the user defines spaces of the garden containing plants called Bounded Clusters, and the algorithm assumes that plants are not present outside of this space. This enables customized results like English-style gardens (Fig. 3.7).

One can also make gardens with 2-way or 4-way symmetry as a visual feature (Fig. 3.8)

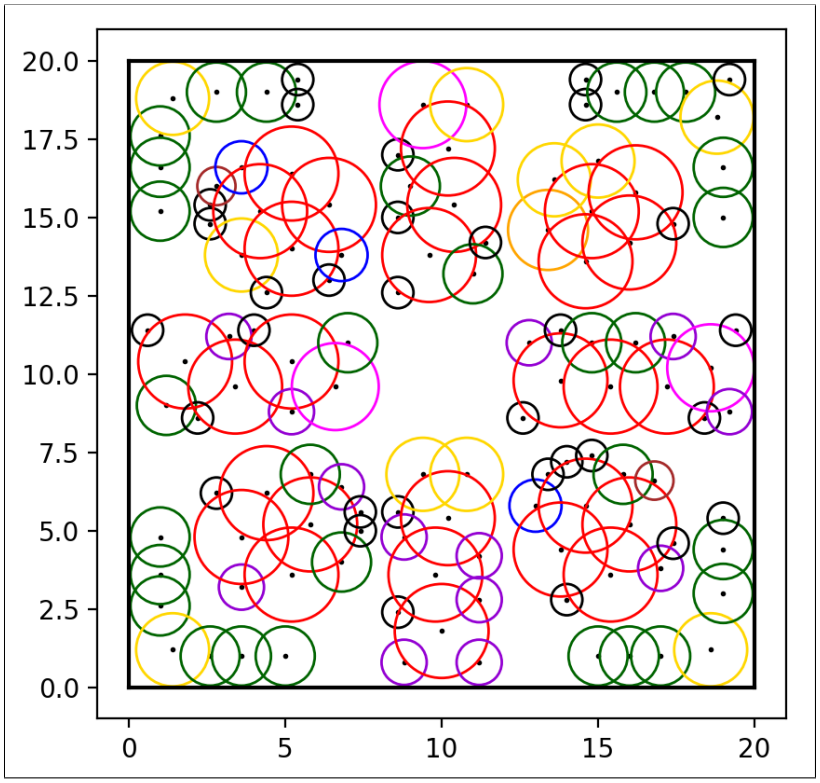

Figure 3.7: **Bounded Clusters.** An English-style Garden generated by

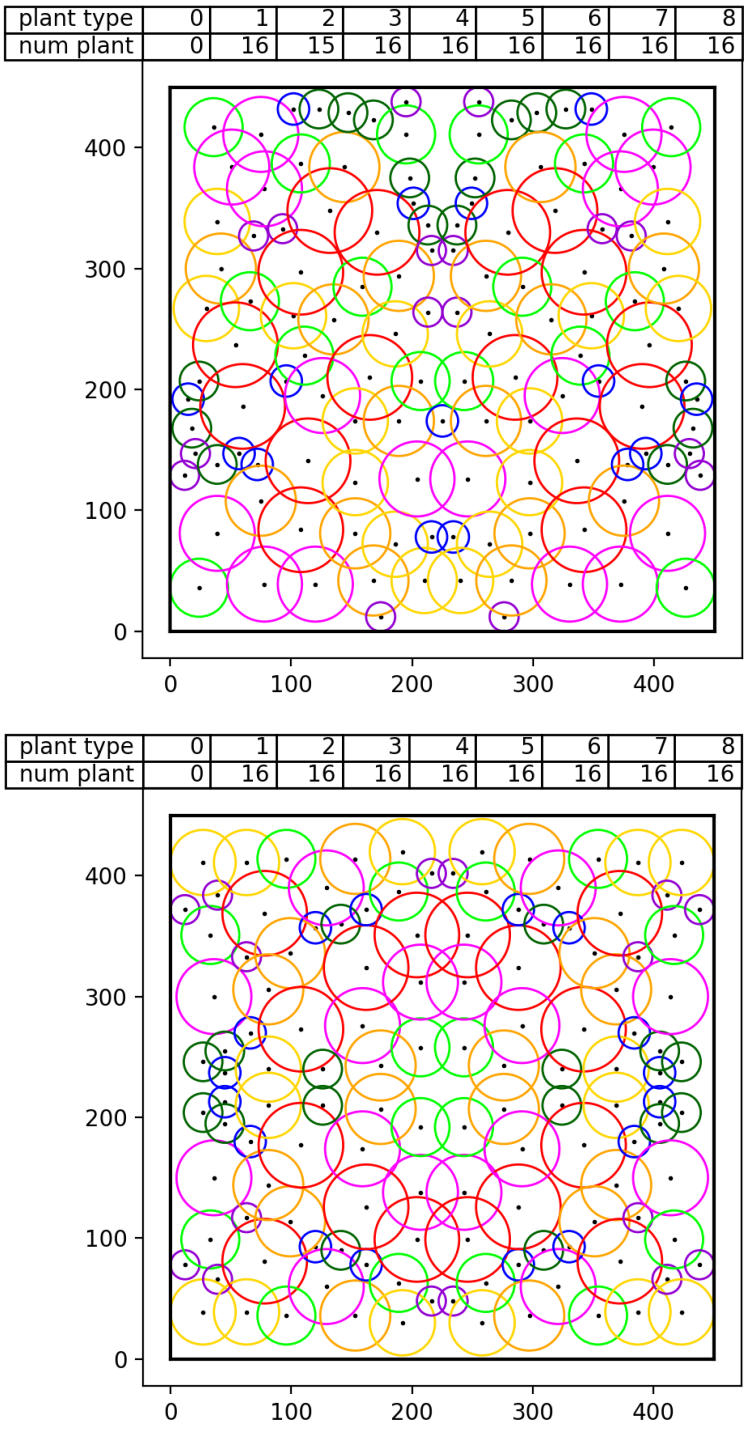

Figure 3.8: **Different Symmetries.** Top: 2-way symmetry in the vertical direction. Bottom: 4-way symmetry.

3.7 Results

Take an example case of $n = 16$ plants, $K = 9$ types (1 type = voids with 2 instances), $n_k = 2$, to be planted in a $1.5m \times 1.5m$ garden space. Running PolyPoD, Each seed placement takes about 30 seconds on a 2018 MacBook Pro. If PolyPoD cannot place all plants in the garden without violating constraints, it returns a garden with fewer plants than specified.

Note that the algorithm works with non-convex polygons as well; see Fig. 3.1 for an example.

In real-world physical experiments, a placement from PolyPoD was planted as a part of related work on the Alphagarden project [34] and [2]. Fig. 3.9 shows the seed placement arrangements and plants used in [34].

We have also made a website where users can input their garden specifications through a UI and receive a seed placement. Please visit our code repository (https://github.com/BerkeleyAutomation/PolyPoD) for the link, and see Fig. 3.10 for an image of the website. url breakurl [breaklinks]hyperref [block=ragged, maxbibnames=99]biblatex

Figure 3.9: **Physical Experiments.** Top: Seed placement generated by mirrored. Bottom: The physical garden reflecting the two placements above, mirrored across the centerline.

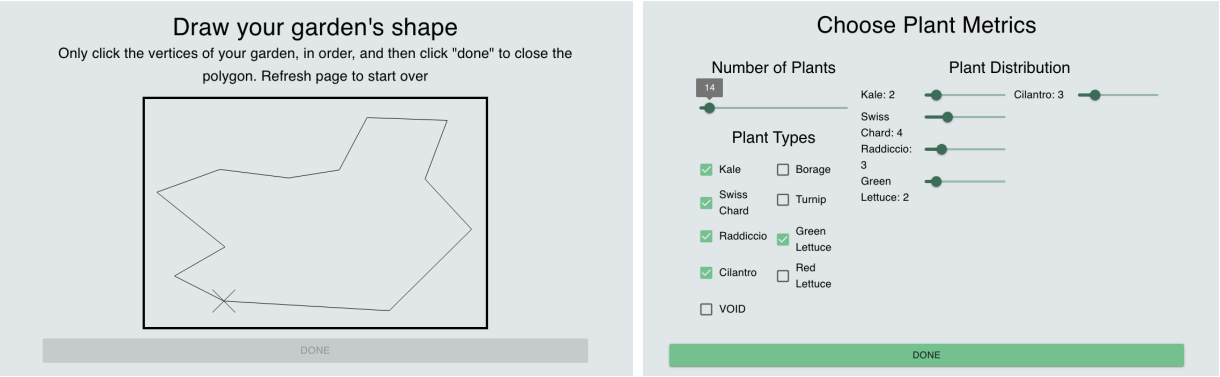

Figure 3.10: **The PolyPoD website.** Users can draw the shape of their garden, and input seed specifications via a UI. The website link can be found at https://github.com/BerkeleyAutomation/PolyPoD.

3.8 Limitations

PolyPoD currently does not contain any optimization process at all: plants are successively placed in one of any random locations that do not violate the Poisson Disk requirements. There could be some optimization process that optimizes plants further, if the objective is to give plants as much space as possible: for example, in Fig. 3.11, the blue plant in the bottom right corner could be moved into the corner to give it more space.

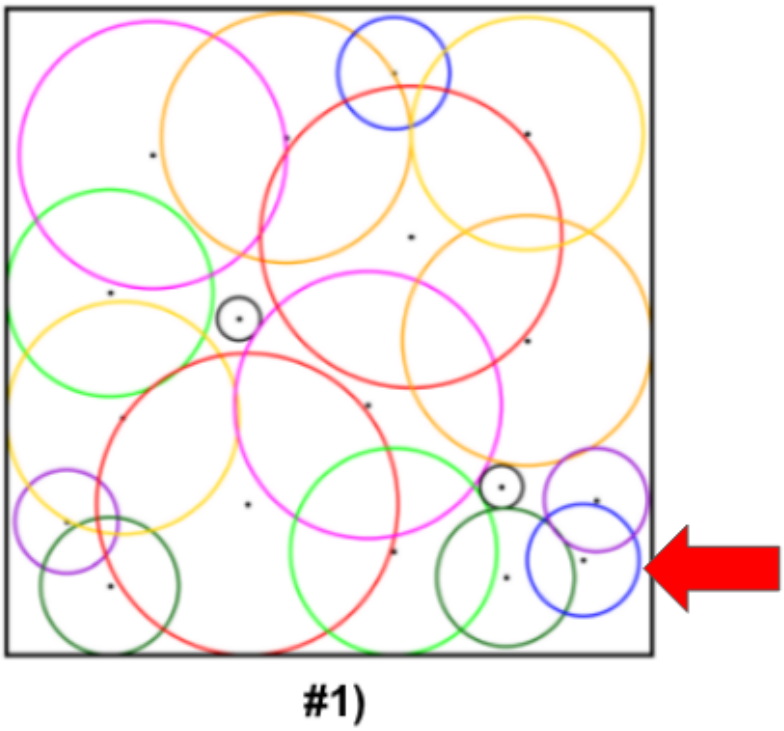

Figure 3.11: Limitation on Seed Placement: The indicated blue plant could be moved to the corner to give it more space.

PolyPoD in its final form is setup to handle polygonal garden shapes, and does not support curved garden shapes. However, mathematically this is not a difficult problem to solve, and I have made draft versions of PolyPoD that have worked with curved garden shapes: the method used an upper and lower boundary function for the garden shape, and the user could define the function to be anything. However, this functionality was removed in the final version because non-polygonal garden shapes were not thought to be common enough and it may add too much complexity to our user interface.

PolyPoD does not return results instantaneously: the algorithm takes a few minutes to run. It would be nice if users were able to see gardens in seconds displayed by their

specification: that would allow them to adjust parameters and interact with the garden design process in more or less real time. This may be necessary if we want users to be able to fine tune parameters such as density: with so many of them, fine-tuning in the current framework may take a hour. It would also be nice to quickly generate multiple garden candidates for a specification, and give the user multiple options to choose from. One of the main bottlenecks is the process that checks whether plants are in the border or not: we may be able to speed this process up by using a more efficient process than our current one, as removing this bound-checking process speeds the algorithm up by a factor of 10.

Ultimately, the growing radius of a plant is only an average and an estimation: plant size varies significantly from case to case, and plants also may not take up a perfectly circular area. They also tend to grow unevenly in one direction, often towards the light, but may exhibit other random, unpredictable growth patterns. Thus, there are limitations to how effective any seed placement algorithm can be guaranteed to be, and it may be beneficial to have a rolling planting process over time that plants an extra plant if one fails to germinate or is much smaller than expected. This process can also be used to replace plants that die, etc.

3.9 Conclusions

PolyPoD is an algorithm for polyculture seed placement that improves on a previous seed-planting algorithm. Myself and the team have built a website where users can input their garden shape and the specifications they want for their garden, and receive a seed placement by PolyPoD.

Another potential approach is to formulate the seed placement problem as an optimization problem where each seed is assigned a value at the user's discretion and the goal is to place seeds of maximum total value while satisfying the spacing conditions. This can be formulated as a nonconvex mixed-integer constrained quadratic program (MIQCP), which is generally NP-hard but in practice can sometimes achieve viable results with solvers or heuristics.

As the goal is usually to place plants relatively uniformly over the garden, we can make heuristics that consider multiple plants: for the biggest plants, if there are four plants to be placed they can be placed all at once evenly spaced from each other and the walls, either in a line or in a square, whichever one leads to more space in between the walls. However, extensive research may be required to design effective heuristics for variable-sized plants, and test each one thoroughly over a range of possible garden shapes. The problem may become intractable very quickly as we have to account for different plant sizes, numbers, garden shapes, densities, etc. Ultimately, by generating multiple placements with the same settings and analyzing patterns that are correlated to success between successive runs of the algorithm, we may be able to quantify which planting locations are optimal based on the settings inputted by the user.

Perhaps we could use some of the tools from Packing Problem literature to find solutions to gardens, because optimal to moderately-optimal solutions to maximal-density packing of N unit circles in a square exist for all $N < 10,000$, see Specht, [43]. However, the variable-radius case has not gotten as much attention, and literature on arbitrary garden (bin) shapes may not exist.

Additionally, there are other ways to stochastically adjust placements to improve them - one idea involves treating the plant circles as balls and moving them around in space, having them elastically collide with each other. This type of simulation is applied in "Hard Sphere" molecular simulations [14, 41]. While attempting to plant a plant with radius r, if there are currently no holes bigger than radius r for the plant to fit, we could do a "Hard Sphere" simulation, frequently check the simulation for holes that open wider than r and place the new plant in that hole, give it a random direction, and continue the simulation. After all plants have been planted, we could also continue the simulation and periodically check for garden attributes such as companionship values, perhaps recording the best placement so far and stopping when some amount of time has passed, or when we are confident that we have come across a relatively good placement. Another idea is to "magnetize" certain balls to be attracted to each other; if companionable plants are attracted to each other and tend to move towards each other, they will naturally cluster together after enough simulation time has passed.

Chapter 4

Conclusions

Ultimately, both of these projects are similar in that one has to consider how the different objects, sutures and plants, interact with each other and their environment, the wound and garden. Interestingly, the objects both reduce to points, as the orientation of the plant is not relevant here. The garden problem is relatively intractable and thus I develop an algorithm that generates solutions by iteratively placing plants and ensuring that plants abide by the Variable Radius Poisson Disk distribution to give each plant enough space. The suture placement problem is tractable, and thus I develop a closed-form optimization framework with hard-coded constraints and a hand-designed Closure Force and Shear Force model.

Both problems could be formulated as a quadratic optimization problem involving the placement of certain point variables in space, which is a relatively similar formulation.

The suturing problem is solved relatively well with an analytic optimization solution, so bringing in Machine Learning to the actual optimization process may work well, but also has the possibility to reduce robustness and add complexity, and could add computation and training time in a very safety-critical application. For the garden problem, I think that machine learning may be of aid here to offer some level of optimization, but I think that many of the most efficient ways to pack shapes into containers are not obvious - see Specht [43] - it is a challenging problem for machine learning to come up with more efficient solutions to these as any patterns that govern how the problem scales with the number of circles are very hard to spot. As our case has variable radii and arbitrarily polygonal garden shapes as well, learning to generalize across all of these variables may prove challenging for a machine learning system. However, given impressive advances in deep learning, I would not consider either of the optimization processes to be beyond the scope of deep learning.

Nonetheless, even if it is not involved with the optimization process itself, deep learning has significant potential to speed up both the Suture Placement and Seed Placement optimization by providing intelligent warm starts. Previous research has shown that warm-starting quadratic programs with the aid of machine learning can yield very good optimization speed improvements, see DJ-GOMP by Ichnowski et al. [19]. Both the Suture Placement and the Seed Placement algorithms are fast enough (worst case on the scale of a few minutes) to generate plenty of training examples for a machine learning system to train on, and then a deep network could provide good "warm starts" for future optimizations.

Simulation is an element that can improve both algorithms as well. For PolyPoD, stochastic methods such as a "Hard Sphere" simulation [14, 41] are likely to work well as they naturally represent the spatial constraints of the plants - see the last paragraph of Section 3.9 for an outline of such a system. For suturing, simulation of the skin and how the Closure Force dissipates through it, likely using FEM techniques, may help to build a more accurate model than our analytic one.

Both of these problems have been framed as planar geometric problems, but both in fact operate in 3 dimensions. For gardens, the third dimension is actually important as taller plants can block sunlight / provide shade for other plants: some small plants will prefer this, and some will prefer direct sunlight. Additionally, sturdy, tall plants such as corn can be used as scaffolding for less sturdy plants to latch onto, such as grapes or green beans. This could be modeled in 3D. For the suture placement, the distribution of forces in 3 dimensions is not

obvious, as the orientation of the skin can change along two axes, and the wound can curve along the third. A new analytical model can be built to accommodate all of these orientation changes, perhaps penalizing the skin orientation changes less that wound orientation changes relative to the skin surface. Additionally, simulating the skin with sutures in 3 dimensions may provide the most accurate results, although this may prove challenging due to the complexities of running FEM in 3 dimensions, as well as the thin nature of the string; in 2 dimensions, it would be easier to input point forces to the simulation.

Bibliography

[1] Simeon Adebola and Ken Goldberg. "Assisting Polyculture Farming in Africa". In: *2021 IEEE AFRICON*. 2021 IEEE AFRICON. ISSN: 2153-0033. Sept. 2021, pp. 1–2. DOI: 10.1109/AFRICON51333.2021.9570957.

[2] Simeon Adebola, Rishi Parikh, Mark Presten, Satvik Sharma, Shrey Aeron, Ananth Rao, Sandeep Mukherjee, Tomson Qu, Christina Wistrom, Eugen Solowjow, and Ken Goldberg. "Can Machines Garden? Systematically Comparing the AlphaGarden vs. Professional Horticulturalists". In: *2023 IEEE International Conference on Robotics and Automation (ICRA)*. 2023 IEEE International Conference on Robotics and Automation (ICRA).

[3] Yahav Avigal, Anna Deza, William Wong, Sebastian Oehme, Mark Presten, Mark Theis, Jackson Chui, Paul Shao, Huang Huang, Atsunobu Kotani, Satvik Sharma, Rishi Parikh, Michael Luo, Sandeep Mukherjee, Stefano Carpin, Joshua H. Viers, Stavros Vougioukas, and Ken Goldberg. "Learning Seed Placements and Automation Policies for Polyculture Farming with Companion Plants". In: *2021 IEEE International Conference on Robotics and Automation (ICRA)*. 2021, pp. 902–908. DOI: 10.1109/ICRA48506.2021.9561431.

[4] Yahav Avigal, Jensen Gao, William Wong, Kevin Li, Grady Pierroz, Fang Shuo Deng, Mark Theis, Mark Presten, and Ken Goldberg. "Simulating Polyculture Farming to Tune Automation Policies for Plant Diversity and Precision Irrigation". In: *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*. IEEE. 2020, pp. 238–245.

[5] Yahav Avigal, William Wong, Mark Presten, Mark Theis, Shrey Aeron, Anna Deza, Satvik Sharma, Rishi Parikh, Sebastian Oehme, Stefano Carpin, Joshua H. Viers, Stavros Vougioukas, and Ken Goldberg. "Simulating Polyculture Farming to Learn Automation Policies for Plant Diversity and Precision Irrigation". In: *IEEE Transactions on Automation Science and Engineering* (2022), pp. 1–13. DOI: 10.1109/TASE.2021.3138995.

[6] Mark Berg, Otfried Cheong, Marc Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. 3rd ed. Springer Berlin, Heidelberg, Oct. 2010. DOI: https://doi.org/10.1007/978-3-540-77974-2.

[7] Robert Bridson. "Fast Poisson disk sampling in arbitrary dimensions".
In: *ACM SIGGRAPH 2007 sketches*. SIGGRAPH07: Special Interest Group on Computer Graphics and Interactive Techniques Conference.
San Diego California: ACM, Aug. 5, 2007, p. 22. ISBN: 978-1-4503-4726-6.
DOI: 10.1145/1278780.1278807.
URL: https://dl.acm.org/doi/10.1145/1278780.1278807 (visited on 02/27/2023).

[8] Timothy E Crews, Wim Carton, and Lennart Olsson.
"Is the future of agriculture perennial? Imperatives and opportunities to reinvent agriculture by shifting from annual monocultures to perennial polycultures".
In: *Global Sustainability* 1 (2018).

[9] Jean Gaston Descoux, Walley J. Temple, Shirley A. Huchcroft, Cyril B. Frank, and Nigel G. Shrive.
"Linea Alba Closure: Determination of Ideal Distance Between Sutures".
In: *Journal of Investigative Surgery* 6.2 (1993). PMID: 8512892, pp. 201–209.
DOI: 10.3109/08941939309141609.
eprint: https://doi.org/10.3109/08941939309141609.
URL: https://doi.org/10.3109/08941939309141609.

[10] Uday Devgan. *Basic Principles of Ophthalmic Suturing*.
https://cataractcoach.com/2018/07/29/basic-principles-of-ophthalmic-suturing/.
Accessed: 2023-03-12.

[11] Wen Dong, Yingwei Sun, and Jiancheng Luo.
"Fine mapping of key soil nutrient content using high resolution remote sensing image to support precision agriculture in Northwest China". In: *2019 8th International Conference on Agro-Geoinformatics (Agro-Geoinformatics)*. 2019, pp. 1–5. DOI: 10.1109/Agro-Geoinformatics.2019.8820504.

[12] David L. Dunn, ed. *Ethicon Wound Closure Manual*. Ethicon, Incorporated, 1994.
URL: https://books.google.com/books?id=BzG0PwAACAAJ.

[13] Nicholas Dwork, Corey A. Baron, Ethan M. I. Johnson, Daniel O'Connor, John M. Pauly, and Peder E. Z. Larson. "Fast variable density Poisson-disc sample generation with directional variation for compressed sensing in MRI".
In: *Magnetic Resonance Imaging* 77 (). 2021-04-01, pp. 186–193. ISSN: 0730-725X.
DOI: 10.1016/j.mri.2020.11.012. URL:
https://www.sciencedirect.com/science/article/pii/S0730725X20306470
(visited on 03/18/2023).

[14] Jeppe C Dyre. "Simple liquids' quasiuniversality and the hard-sphere paradigm".
In: *Journal of Physics: Condensed Matter* 28.32 (June 2016), p. 323001.
DOI: 10.1088/0953-8984/28/32/323001.
URL: https://dx.doi.org/10.1088/0953-8984/28/32/323001.

[15] Oxford Medical Education. *Suturing techniques*. Apr. 2016. URL: https://oxfordmedicaleducation.com/clinical-skills/procedures/suturing-techniques/.

[16] Sean Gillies et al. *Shapely: manipulation and analysis of geometric objects*. toblerity.org, 2007–. URL: https://github.com/Toblerity/Shapely.

[17] S Gliessman, M Altieri, et al. "Polyculture cropping has advantages". In: *California Agriculture* 36.7 (1982), pp. 14–16.

[18] Fang Gou, Martin K van Ittersum, and Wopke van der Werf. "Simulating potential growth in a relay-strip intercropping system: model description, calibration and testing". In: *Field Crops Research* 200 (2017), pp. 122–142.

[19] Jeffrey Ichnowski, Yahav Avigal, Vishal Satish, and Ken Goldberg. "Deep learning can accelerate grasp-optimized motion planning". In: *Science Robotics* 5.48 (2020), eabd7710. DOI: 10.1126/scirobotics.abd7710. eprint: https://www.science.org/doi/pdf/10.1126/scirobotics.abd7710. URL: https://www.science.org/doi/abs/10.1126/scirobotics.abd7710.

[20] Asif Islam, Prem Saha, Masud Rana, Md. Mim Adnan, and Bishwajit Banik Pathik. "Smart Gardening Assistance System with the Capability of Detecting Leaf Disease in MATLAB". In: *2019 IEEE Pune Section International Conference (PuneCon)*. 2019, pp. 1–6. DOI: 10.1109/PuneCon46936.2019.9105677.

[21] Russell C. Jackson, Viraj Desai, Jean P. Castillo, and M. Cenk Çavuşoğlu. "Needle-Tissue Interaction Force State Estimation for Robotic Surgical Suturing". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Daejeon, South Korea: IEEE Press, 2016, pp. 3659–3664. DOI: 10.1109/IROS.2016.7759539. URL: https://doi.org/10.1109/IROS.2016.7759539.

[22] Dieter Kraft. "A software package for sequential quadratic programming". In: *Forschungsbericht- Deutsche Forschungs- und Versuchsanstalt fur Luft- und Raumfahrt* (1988).

[23] Matt Liebman. "Polyculture cropping systems". In: *Agroecology*. CRC Press, 2018, pp. 205–218.

[24] Raymond C Loehr, Edward F Neuhauser, and Michael R Malecki. "Factors affecting the vermistabilization process: Temperature, moisture content and polyculture". In: *Water Research* 19.10 (1985), pp. 1311–1317.

[25] Growing Interactive Ltd. *GrowVeg*. URL: https://www.growveg.com/garden-planner-intro.aspx.

[26] Hemalatha M. *Implementation of Poisson Disc Sampling in Javascript*. Medium. 2020-02-17. URL: https://medium.com/@hemalatha.psna/implementation-of-poisson-disc-sampling-in-javascript-17665e406ce1 (visited on 03/18/2023).

[27] Nick Marsidi, Sofieke A.M. Vermeulen, Tim Horeman, and Roel E. Genders. "Measuring Forces in Suture Techniques for Wound Closure". In: *Journal of Surgical Research* 255 (2020), pp. 135–143. ISSN: 0022-4804. DOI: https://doi.org/10.1016/j.jss.2020.05.033. URL: https://www.sciencedirect.com/science/article/pii/S0022480420303061.

[28] A. R. Mehrabian and C. Lucas. "A novel numerical optimization algorithm inspired from weed colonization". In: *Ecological Informatics* 1.4 (). 2006, pp. 355–366. ISSN: 1574-9541. DOI: https://doi.org/10.1016/j.ecoinf.2006.07.003. URL: https://www.sciencedirect.com/science/article/pii/S1574954106000665.

[29] Scott Mitchell, Alexander Rand, Mohamed Ebeida, and Chandrajit Bajaj. "Variable Radii Poisson-Disk Sampling". In: *Proceedings of the 24th Canadian Conference on Computational Geometry, CCCG 2012* 13 (Jan. 2012), pp. 241–258. DOI: 10.1023/A:1020568125418.

[30] Florent Nageotte, Philippe Zanne, Christophe Doignon, and Michel de Mathelin. "Stitching Planning in Laparoscopic Surgery: Towards Robot-assisted Suturing". In: *The International Journal of Robotics Research* 28.10 (2009), pp. 1303–1321. DOI: 10.1177/0278364909101786. eprint: https://doi.org/10.1177/0278364909101786. URL: https://doi.org/10.1177/0278364909101786.

[31] Sahba Aghajani Pedram, Peter Ferguson, Ji Ma, Erik Dutson, and Jacob Rosen. "Autonomous suturing via surgical robot: An algorithm for optimal selection of needle diameter, shape, and path". In: *2017 IEEE International Conference on Robotics and Automation (ICRA)* (2017), pp. 2391–2398. DOI: 10.1109/ICRA.2017.7989278.

[32] Sahba Aghajani Pedram, Changyeob Shin, Peter Walker Ferguson, Ji Ma, Erik P. Dutson, and Jacob Rosen. "Autonomous suturing framework and quantification using a cable-driven surgical robot". In: *IEEE Transactions on Robotics* 37 (2 Apr. 2021), pp. 404–417. ISSN: 19410468. DOI: 10.1109/TRO.2020.3031236.

[33] Upeka Premaratne, Jagath Samarabandu, and Tarlochan Sidhu. "A New Biologically Inspired Optimization Algorithm". In: *ICIIS 2009 - 4th International Conference on Industrial and Information Systems 2009, Conference Proceedings*. Dec. 2009. DOI: 10.1109/ICIINFS.2009.5429852.

[34] Mark Presten, Rishi Parikh, Shrey Aeron, Sandeep Mukherjee, Simeon Adebola, Satvik Sharma, Mark Theis, Walter Teitelbaum, and Ken Goldberg. "Automated Pruning of Polyculture Plants". In: *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*. 2022 IEEE 18th International Conference on Automation Science and Engineering (CASE). ISSN: 2161-8089. Aug. 2022, pp. 242–249. DOI: 10.1109/CASE49997.2022.9926632.

[35] Weizheng Ren, Liangliang Hu, Jian Zhang, Cuiping Sun, Jianjun Tang, Yongge Yuan, and Xin Chen. "Can positive interactions between cultivated species help to sustain modern agriculture?"
In: *Frontiers in Ecology and the Environment* 12.9 (2014), pp. 507–514.
DOI: 10.1890/130162. eprint:
https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.1890/130162.
URL: https://esajournals.onlinelibrary.wiley.com/doi/abs/10.1890/130162.

[36] Stephen J Risch. "Intercropping as cultural pest control: prospects and limitations".
In: *Environmental Management* 7.1 (1983), pp. 9–14.

[37] H. Saeidi, H. N.D. Le, J. D. Opfermann, S. Leonard, A. Kim, M. H. Hsieh, J. U. Kang, and A. Krieger. "Autonomous laparoscopic robotic suturing with a novel actuated suturing tool and 3d endoscope". In: *Proceedings - IEEE International Conference on Robotics and Automation* 2019-May (2019), pp. 1541–1547.
ISSN: 10504729. DOI: 10.1109/ICRA.2019.8794306.

[38] John Schulman, Ankush Gupta, Sibi Venkatesan, Mallory Tayson-Frederick, and Pieter Abbeel. "A case study of trajectory transfer through non-rigid registration for a simplified suturing scenario". In: *IEEE International Conference on Intelligent Robots and Systems* (2013), pp. 4111–4117. ISSN: 21530858.
DOI: 10.1109/IROS.2013.6696945.

[39] Siddarth Sen, Animesh Garg, David V. Gealy, Stephen McKinley, Yiming Jen, and Ken Goldberg. "Automating multi-throw multilateral surgical suturing with a mechanical needle guide and sequential convex optimization".
In: *Proceedings - IEEE International Conference on Robotics and Automation* 2016-June (2016), pp. 4178–4185. ISSN: 10504729.
DOI: 10.1109/ICRA.2016.7487611.

[40] Azad Shademan, Ryan S Decker, Justin D Opfermann, Simon Leonard, Axel Krieger, and Peter C W Kim. "Supervised autonomous robotic soft tissue surgery". en.
In: *Sci Transl Med* 8.337 (May 2016), 337ra64.

[41] Frank Smallenburg. "Efficient event-driven simulations of hard spheres".
In: *The European Physical Journal E* 45.3 (2022), p. 22.
DOI: 10.1140/epje/s10189-022-00180-8.
URL: https://doi.org/10.1140/epje/s10189-022-00180-8.

[42] Daegu Son and Aram Harijan. *Overview of surgical scar prevention and management*. 2014. DOI: 10.3346/jkms.2014.29.6.751.

[43] Eckard Specht. *The best known packings of equal circles in a square*. 2010.
URL: http://hydra.nat.uni-magdeburg.de/packing/csq/csq.html (visited on 06/28/2023).

[44] Brijen Thananjeyan, Ajay Tanwani, Jessica Ji, Danyal Fer, Vatsal Patel, Sanjay Krishnan, and Ken Goldberg. "Optimizing Robot-Assisted Surgery Suture Plans to Avoid Joint Limits and Singularities". In: *2019 International Symposium on Medical Robotics, ISMR 2019* (2019). DOI: 10.1109/ISMR.2019.8710194.

[45] Rhia Trogo, Jed Barry Ebardaloza, Delfin Jay Sabido, Gerry Bagtasa, Edgardo Tongson, and Orlando Balderama. "SMS-based Smarter Agriculture decision support system for yellow corn farmers in Isabela". In: *2015 IEEE Canada International Humanitarian Technology Conference (IHTC2015)*. 2015, pp. 1–4. DOI: 10.1109/IHTC.2015.7238049.

[46] Wikipedia. *Art gallery problem*. 2023. URL: https://en.wikipedia.org/wiki/Art_gallery_problem (visited on 06/29/2023).

[47] Wikipedia. *Packing problems*. 2023. URL: https://en.wikipedia.org/wiki/Packing_problems (visited on 06/29/2023).