MASTER THESIS RESEARCH PROPOSAL

Finding dense and well-shaped convex clusters in 2-dimensional point sets

Student:
Gianmarco Picarella
g.picarella@students.uu.nl

Supervisors:
Marc van Kreveld
Frank Staals
Sjoerd de Vries

Department of Information and Computing Sciences
March 6, 2025

1 Introduction

1.1 Problem definition

Given a point set P in d-dimensional space, a discrete convex polygon (DCP) in P is a convex polygon with vertices $V \subseteq P$. The convex hull of a point set is always a DCP. Interestingly, many real-world problems arising in different disciplines (e.g., medicine and the oil industry) can be easily modeled as optimization problems, where a DCP maximizing (or minimizing) a certain objective function must be found. These problems may require taking into consideration additional constraints, usually defined as scalar values limiting specific properties of the polygon (e.g. the maximum allowed DCP area is bounded by $a_{\max}$). The geometric knapsack problem is a generalization of these problems, which has been extensively studied for a variety of different objective functions and constraints [1].

We now introduce the notion of diameter. Let P be a set of 2-dimensional points. The diameter d of P is the maximum Euclidean distance between two points $p, q \in P$. The diameter of any convex polygon is always equivalent to its major axis length.

In this master's thesis project, we focus on a variation of the geometric knapsack problem that, to the best of our knowledge, has not been studied in existing research. The problem is defined as follows.

Problem 1 *Given a set of 2-dimensional points P find a subset $S \subseteq P$ with $|S| > 2$ such that the convex hull of S satisfies the following properties:*

1. *The diameter of S does not exceed d_{max}.*
2. *The area defined by the convex hull of S does not exceed a_{max}.*
3. *S has maximum cardinality.*

The formulation of problem 1 comes from the generalization of a real-world need in the field of oncology, specifically in breast cancer prognosis.

1.2 Motivation

Breast cancer is among the most prevalent cancer types in the world and the primary cause of mortality due to cancer in women. According to the

World Health Organization (WHO), breast cancer was responsible for 670.000 deaths worldwide in 2022, and it is expected to reach approximately 1.5 million deaths by 2050 [2].

Histological grading, which is considered one of the most powerful prognostic tools for early stage invasive breast cancer, involves assessing a tumour by examining how abnormal the cancer cells and tissue appear under a microscope and estimating the likely growth and spread rate of the cancer cells. Specific tumour features are observed, and an overall grade is assigned using a standardized system like the Nottingham grading system [3].

The mitotic count (MC), which has been shown to provide the strongest prognostic value among many tumour features [4], is a density measurement usually expressed as the number of mitoses per mm^2. This value is generally determined by a two-step procedure. First, the pathologist selects the patient slide containing a region which visually shows the highest tumour proliferation. Then, within this region, the pathologist locates a sub-region of area $2mm^2$ containing the highest number of mitotic cells. The mitotic count is the number of mitotic cells in the sub-region located in the previous step.

Several studies [5, 6] showed that both the selection of areas and the identification of mitotic cells are prone to variability between observers and even within the same observer over time, resulting in limited accuracy and reproducibility hampering the clinical utility of MC. Consequently, there is a growing need for computer-assisted methods to support pathologists in achieving more consistent and accurate results.

In the context of the Mitosis Domain Generalisation Challenge 2021 (MIDOG21) [7], a research group from the University Medical Center Utrecht (UMC Utrecht) developed a deep learning model [8] capable of detecting mitoses with high accuracy in Whole-Slide Images (WSIs; digitalized histological slides) produced with four different scanners. The model was trained on the MIDOG21 dataset [9], a collection of human breast cancer tissue samples with mitotic figures annotated by an expert pathologist. A tissue sample (WSI) from the dataset is shown in Figure 1a. To improve visualization, the WSI has been cropped to a square format.

The model [8] was then expanded by an automatic area selector method which computes the convex region (modeled as a DCP) with the highest mitotic proliferation and its MC. Figure 1 illustrates the process for computing the MC for a single WSI in MIDOG21. Pathologists can specify the maximum extension and area for a DCP to be considered valid. Problem 1 offers

a practical and complete framework for modelling these requirements.

The current area selection method employs either an exact or heuristic algorithm based on the number of mitotic cells to process. While the exact algorithm provides optimal solutions, its asymptotic time complexity is exponential in the number of mitotic cells considered. On the other hand, the heuristic algorithm operates in polynomial time, offering significantly faster computations. However, it provides no guarantees regarding the accuracy of its solutions. This is not ideal, as achieving predictable quality in results is an essential requirement for all medical software. In practice, because the point sets are usually too big for the exact algorithm (i.e. above 25 points), the heuristic algorithm is the most used algorithm.

Our work tries to address these limitations to enhance the current capabilities of the area selection method in terms of speed and accuracy by implementing and benchmarking new exact (and possibly ϵ-approximation) algorithms for Problem 1.

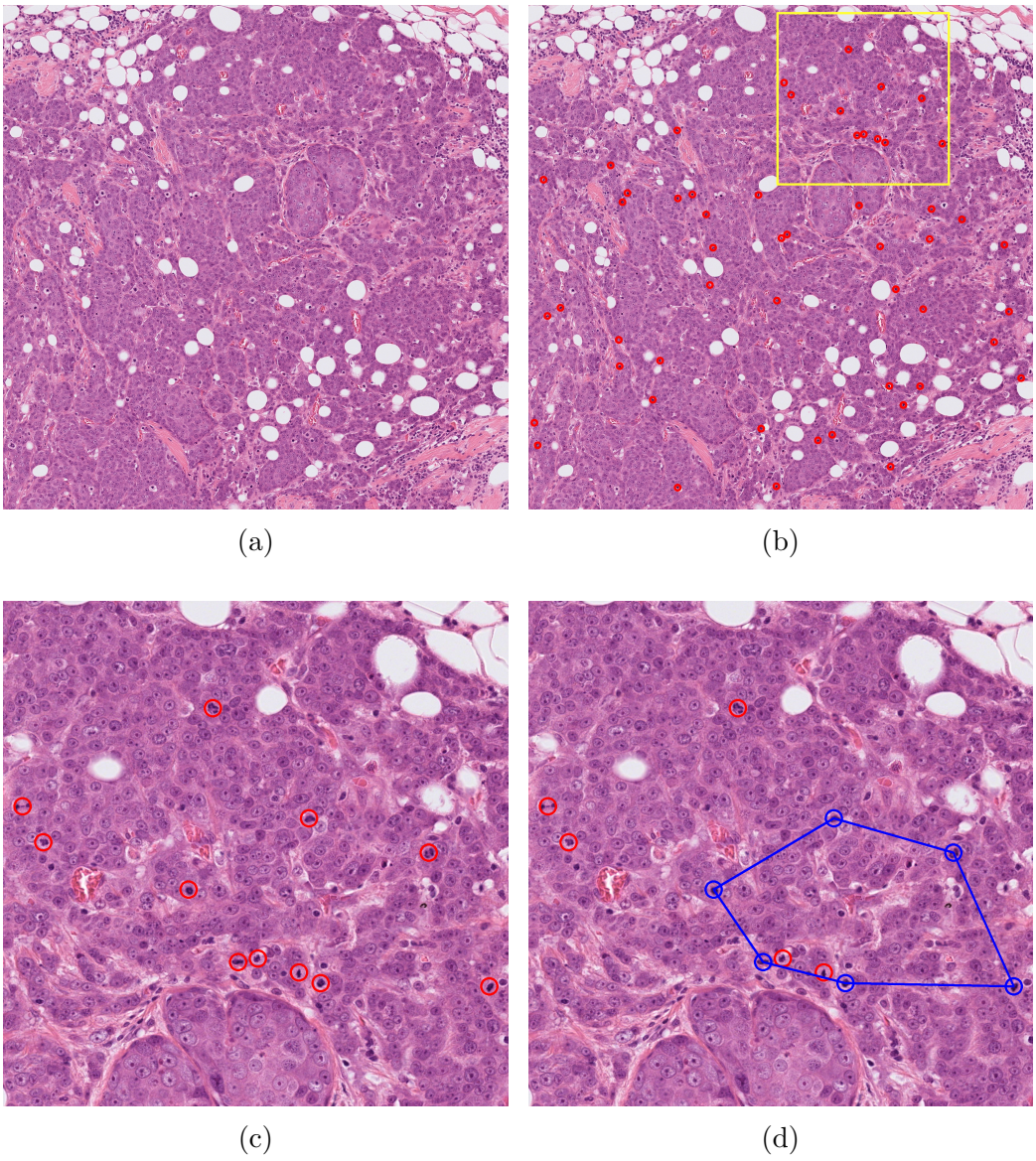
2 Related work

2.1 Geometric knapsack problem

The knapsack problem asks to fill a given knapsack of limited capacity with items from a given collection such that the total value of the items is maximized. The decision problem form of the knapsack problem (i.e. can a total value of at least v be achieved without exceeding the capacity c) is NP-complete. Therefore, no polynomial time algorithm exists that is both exact and fast for all input cases.

The geometric knapsack problem is a reinterpretation of the classic knapsack problem within a geometric framework. In this setting, we are interested in finding a maximum-value subset $S \subseteq P$ obeying the knapsack capacity, where P is a set of geometric objects (e.g. points, polygons, etc.) and the knapsack capacity is a measurable property of a geometric object generated by S. We are interested in problems for which the capacity is the area of the convex hull of S and the value of each point is 1, thus $V(S) = |S|$, where $V(\cdot)$ returns the sum of the point values in S.

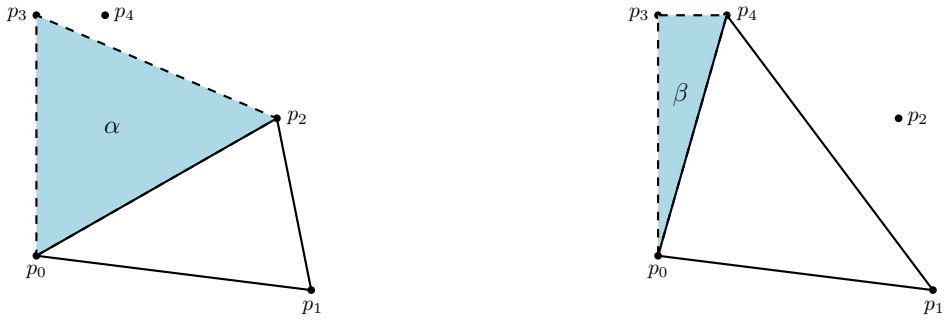
Arkin et al. [10] presented a complete analysis of several variations of the geometric knapsack problem considering limited area or perimeter and unlimited area and perimeter while having point values $v_i \in \mathbb{Z}$ or $v_i \in \mathbb{R}_+$.



(a) (b)

(c) (d)

Figure 1: (a) Input breast cancer tissue. (b) Overview on low magnification of the deep learning model output. Every detected mitotic figure is marked with a red dot. The detected hotspot area is marked in yellow. (c) High magnification of the hotspot area shown in (b). (d) Identical area as in (c) in which the automatic area selector located the highest proliferation (HP) area. The mitotic figures enclosing the HP area are marked in blue.



(a) (b)

Figure 2: (a) Adding p_3 to triangle $p_0p_1p_2$ has cost α. (b) Adding p_3 to triangle $p_0p_1p_4$ has cost β. Clearly, $\alpha > \beta$.

They prove that both problems are NP-hard and derive several polynomial time algorithms running in $O(kn^3)$ and using $O(kn^2)$ space, which are adaptable to produce solutions satisfying our requirements 2 and 3 for Problem 1. Eppstein et al. [11] provide a generalization of these algorithms for finding convex k-gons (i.e. a DCP defined by k vertices) minimizing or maximizing monotone decomposable functions (MDF). Similar procedures are also derived by Fisher [12].

The challenge arising in the geometric version of the knapsack problem is that the cost of adding a new item to the knapsack dynamically changes based on the previously chosen items. As an example, consider Figure 2. Based on the convex hull determined by the previously chosen points, adding the new point p_3 to the knapsack has a different area (or perimeter) cost.

2.2 Convex k-gons minimizing a decomposable weight function

The work from Eppstein et al. [11] presented an $O(kn^3)$ time and $O(kn^2)$ space dynamic programming algorithm for finding minimum area convex k-gons. More importantly, their solution can find convex k-gons minimizing any monotone decomposable weight function (MDF) defined on any polygon $\mathcal{C}$.

Definition 1 *A weight function W is decomposable iff for any polygon $\mathcal{C} = \langle p_1, ..., p_m \rangle$ and any index $2 < i < m$*

$$W(\mathcal{C}) = \diamond(W(\langle p_1, ..., p_i \rangle), W(\langle p_1, p_i, p_{i+1}, ..., p_m \rangle), p_1, p_i)$$

where $\diamond$ takes constant time to compute. W is called monotone decomposable iff $\diamond$ is monotone in its first (and, hence, in its second) argument.

When W is decomposable we can cut the polygon $\mathcal{C}$ into two subpolygons $\mathcal{C}_1$ and $\mathcal{C}_2$ along the segment p_1p_i and derive $W(\mathcal{C})$ from $W(\mathcal{C}_1)$, $W(\mathcal{C}_2)$ and some information about the cut segment. Some examples of MDF include the area, perimeter, sum of internal angles and the number of points enclosed by $\mathcal{C}$.

We now provide a brief explanation of how the algorithm works. Given a point set P with n points and $k \leq n$, the algorithm creates a four-dimensional array $AR[n, n, n, k]$ storing at each location $AR[i, j, l, m]$ the area of the smallest convex m-gon $\mathcal{C}$ defined as follows: the point p_i is the bottom-most vertex of $\mathcal{C}$, p_j is the next vertex in counterclockwise order and all points of $\mathcal{C}$ are located on the same side of the line $\overline{p_jp_l}$ as p_i. The minimum area k-gon is the minimum value among the n^3 entries in $AR[\cdot, \cdot, \cdot, k]$. Therefore, this algorithm aims to fill AR up to $m = k$.

In the initialization phase, all values in $AR[\cdot, \cdot, \cdot, 2] = 0$ because the area of a convex 2-gon is 0. Let us fix our attention to one point p_i in P. We define P_i as the points above the horizontal line passing through p_i sorted in a clockwise order around p_i. For each point $p_j \in P_i$ we consider all the candidate points $p_l \in P_i \setminus \{p_j\}$ sorted by the slope of the lines $\overline{p_ip_l}$ in clockwise order. Treating the points p_j and p_l in this ordering implies that the minimum area m-gon stored in $AR[i, j, l, m]$ is either the same value obtained with the predecessor of p_l (pred(l) from now on) or it involves p_l as a new point defining the m-gon. Figure 3 presents these two cases. When p_l lies on the right side of $\overline{p_ip_j}$ then it is discarded, thus $AR[i, j, l, m] = AR[i, j, \text{pred}(l), m]$. If p_l lies on the left side of $\overline{p_ip_j}$, then p_l may be a new vertex of the minimum area m-gon. The new area is the sum of the area of triangle $\triangle p_ip_jp_l$ and the minimum area $(m-1)$-gon in $AR[i, l, j, m-1]$. Therefore, $AR[i, j, l, m]$ is set to the minimum of this value and $AR[i, j, \text{pred}(l), m]$. Repeating this procedure for all possible p_j and p_l up to $m = k$ requires $O(kn^2)$ work and provides the minimum area m-gon defined by the set of points above p_i (including p_i as the bottom-most vertex of the polygon). Finding the minimum area m-gon

(a) (b)

Figure 3: (a) p_l lies on the right side of $\overline{p_i p_j}$. (b) p_l lies on the left side of $\overline{p_i p_j}$.

for P requires repeating the above procedure to all $p_i \in P$. Therefore, the total work needed by this algorithm is $O(kn^3)$.

Eppstein et al.'s work provides an efficient way for computing solutions satisfying requirements 2 and 3 for Problem 1. Unfortunately, considering the additional constraint on the diameter of the k-gon is impossible because the diameter of a convex polygon is not an MDF.

Other related works are the one from Fisher [12], which proposed an $O(n^3 \log n)$ time algorithm for finding maximum area convex polygons defined by points labelled as "positive" and without containing any point labelled as "negative". If the set of "negative" points is empty, then the result is the convex polygon having maximum area. Unfortunately, the same limitation regarding the diameter also applies here.

Several improvements on top of these results have been presented in later work by Eppstein in [13] and more recently by Keikha in [14]. Eppstein proposed an algorithm finding the minimum area convex k-gon in $O(n^2 \log n + 2^{6k} n^2)$, offering an asymptotic improvement when $k < \log n/6$. This algorithm is limited to area minimization problems and seems to be not extendable to other MDFs.

Keikha [14] improves Eppstein et al.'s algorithm [11] to $O(n^3 \log k)$ for point sets in convex position by using a divide-and-conquer technique which merges two convex polygons of size 2^t at each iteration, for $t = 0, ..., \log k$.

The work from Mitchell et al. [15] uses similar algorithms to that of

Eppstein et al. [11] for counting convex k-gons, all convex polygons, all empty convex polygons and all convex polygons containing a given point. Thanks to the numerous visualizations, we found this work very useful for better understanding Eppstein et al.'s work.

2.3 Automatic area selection method

2.3.1 Overview

Stathonikos et al. [8] introduce an automatic area selection method used to compute the mitotic count (MC) from a set of 2-dimensional points representing the location of the miotic figures detected in a whole-slide image (WSI).

This method is composed of two algorithms: an exact and a heuristic algorithm. Given an input set P, the main goal is to find the set $S \subseteq P$ having maximum cardinality and convex hull area and diameter bounded by $a_{\max}$ and $d_{\max}$.

As a brief overview, the method works as follows. The input set P is partitioned using a sliding window technique on the 2-dimensional plane. Then, each resulting subset is processed independently by either using the exact or the heuristic algorithm. The optimal solution is guaranteed to be a subset contained in one of the sets generated by the partition. A detailed analysis of the method is provided in the following sections. In Section 2.3.2, we describe the technique used to partition the input set into (potentially) smaller subsets that can be processed independently. In Section 2.3.3, we describe the exact algorithm. Finally, in Section 2.3.4, we describe the heuristic algorithm.

2.3.2 Input partitioning

The input set P is partitioned using a sliding window technique, which generates subsets of P such that one subset contains the optimal result. Figure 4 presents a visual example of how such windowing works.

The window has a fixed side length l and a step size s and is allowed to slide either horizontally or vertically. Despite the lack of an explicit constraint on the maximum diameter, having a properly set side length and step size effectively limits the maximum diameter of the resulting region. As an example, consider using $l = 2d_{\max}$ and $s \leq 0.5l$.

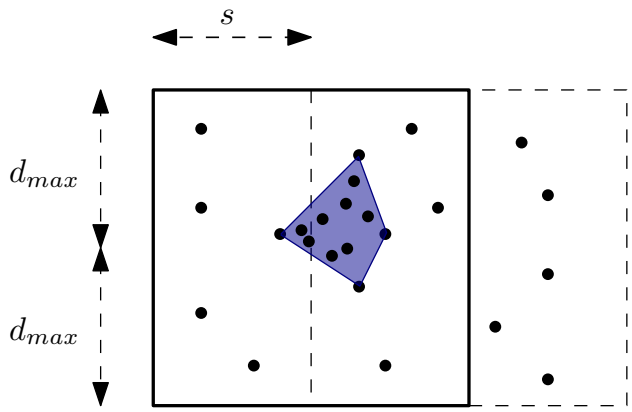


Figure 4: The sliding window with side $l = 2d_{\max}$ and $s = d_{\max}$. The optimal convex region is marked in blue and overalps two consecutive windows.

These settings guarantee that all convex regions with diameter bounded by $\frac{d_{\max}\sqrt{2}}{2}$ are always processed. Additionally, there may be some potential regions having diameter up to $d_{\max}\sqrt{2}$. These regions are easily filtered out by checking the diameter of the shape.

The step size is limited to that range because the optimal solution could be defined by points contained in two consecutive windows. When s is within that range, the optimal solution with diameter bounded by $d_{\max}$ is always guaranteed to be found in the partition of P.

This partitioning technique provides a way to reduce (for many point set distributions) the overall amount of work needed to find the optimal solution while still preserving output optimality. For each window, either the exact or heuristic algorithm is used to process the enclosed points. If the number of points is smaller than 26, then the exact algorithm is used; otherwise, the heuristic algorithm is used. The small threshold is due to the exponential time complexity of the exact algorithm, which heavily limits its usage to relatively small point sets.

2.3.3 Exact algorithm

Given a subset of points $S \subseteq P$ generated with the window sliding technique, the exact algorithm uses a brute-force approach, selecting out of all the possible subsets in 2^S the one generating a convex hull satisfying the area requirement and having maximum cardinality.

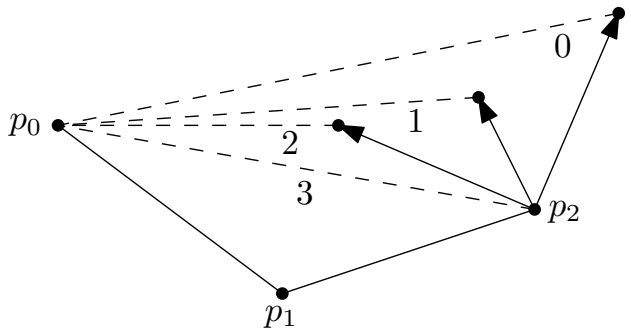


Figure 5: Convex paths generated by the exact algorithm from point p_2 and sorted counterclockwise around p_2. The dashed edges close the polygons.

This is achieved by recursively constructing all the convex paths originating from any starting point $s_i \in S$, in the order of the sorted angles these paths make with respect to the x-axis.

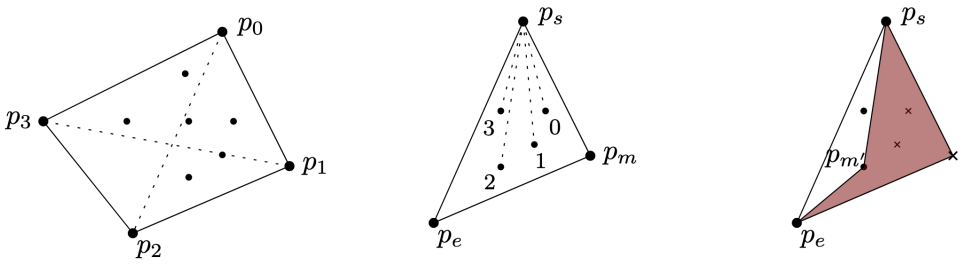
The algorithm uses a recursive procedure, which constructs all the convex paths originating from any point in $p_0 \in S$, in the order of the sorted angles these paths make with respect to the x-axis. At every recursive step i, the algorithm sequentially chooses the rightmost point p_i that has not been explored yet and that keeps the path convex (see Figure 5). All the remaining points lying on the left side of $\overline{p_i p_0}$ are discarded for the next recursive step; in fact, they are either already visited points or interior points that cannot generate any new convex path from p_i. The recursion ends when the area of the convex path exceeds $a_{\max}$ or there are no points left to explore. Because all the possible convex polygons in S are explored, the optimal solution is always guaranteed to be found.

If S is a convex set, then the algorithm will process 2^n different convex polygons. Even for non-convex sets, the average number of convex polygons is nearly exponential. For this reason, using this algorithm even for small sets of 25 points is very inefficient.

2.3.4 Heuristic area selection

Given a subset of points $S \subseteq P$ generated with the window sliding technique, the heuristic algorithm computes the convex hull of S and then uses a greedy approach to shrink the convex hull until its area is smaller or equal to $a_{\max}$.

The greedy step iterates through all the triples of consecutive points along



(a) (b) (c)

Figure 6: (a) The initial convex hull. The dotted lines represent the third side of triangles generated by all consecutive triples of points along the hull. (b) The triangle $\triangle p_0 p_1 p_2$ with interior points sorted clockwise around p_s. (c) The point p'_m maximizes the area erased over points removed ratio.

the current convex hull. Each triple is defined by the points p_s, p_m, p_e, namely the starting, middle and end points. For any triple, the goal is to find a point q enclosed by the triangle $\triangle p_s p_m p_e$, which generates a new triangle $\triangle p_s q p_e$ maximizing the amount of erased area over the number of points removed. Figure 6 shows the greedy step for one triangle. The optimal shrink found across all triples is the one applied to the current convex hull. This step is repeated until the convex hull area falls below $a_{\max}$. This procedure preserves convexity and can be performed efficiently.

The worst-case time complexity for the heuristic algorithm is $O(n^2 \log n)$ and $O(n)$ space. In fact, each greedy step has time complexity $O(n \log n)$: for all triangles defined by a triple of consecutive points, all the points in the triangle are sorted clockwise with respect to the vertex p_s (i.e. the vertex with the lowest index in the triple), with total work $O(n \log n)$. The triangle maximizing the area erased over points removed is then found in $O(n)$. This step can be repeated at most n times, thus leading to the time complexity $O(n^2 \log n)$.

Because the greedy step makes every time a shortsighted choice, optimality is hardly reachable, and the output quality greatly varies based on S.

2.4 Statistical clusters analysis

Statistical clustering methods, such as DBSCAN [16], rely on clustering points based solely on pairwise distances, without considering the constraints in terms of area and diameter of the convex hull formed by the clustered points. Additionally, these techniques usually require careful parameter selection which may lead to significant differences in the outcomes even for the same input set.

2.5 Diameter of 2-dimensional point sets

Shamos [17] proposed an efficient $O(n)$ technique for computing the diameter of a convex polygon by considering all the parallel antipodal pairs of points along the hull. The diameter is determined by the length of the segment connecting the pair of furthest antipodal points.

3 Research questions

In this master project, we aim to investigate how to efficiently implement new algorithms for Problem 1 and how their running time relate to the point sets and parameters used.

The first key aspect to consider is the choice of the measure for quantifying the extension of a discrete convex polygon (DCP). Although Problem 1 defines the extension of a DCP by its diameter, selecting an appropriate measure required careful thinking and consideration of both theoretical and practical implications. For this reason, we decided to motivate our choice with a dedicated research question. In general, we are interested in measures providing compact representations that are also feasible to implement and are not expensive to compute. We phrase the research question precisely:

RQ1: *What is the most suitable measure for quantifying the compactness of a DCP shape that offers a compact representation and is feasible for algorithmic implementation?*

With the term compact representation, we refer to the ability to describe a DCP extension with the minimum amount of information possible. In terms of algorithmic feasibility, we are looking for measures that are practically fast to compute and possibly straightforward to implement. The aspect

ratio was initially the measure used by pathologists describing the maximum ratio of extension for DCPs. Specifically, they indicated an aspect ratio $5:1$. Therefore, it seemed logical to work with aspect ratios in our algorithm as well. In practice, we soon realised the aspect ratio is not a proper measure for our needs.

The next step is about designing the actual algorithm for Problem 1. Given the experimental nature of this master project, prof. van Kreveld and Staals will take care of the theoretical work needed to design a proper algorithmic solution for the problem at hand, while the student will focus on its actual implementation and benchmarking. At the time of writing, although the theoretical work is still not finalized, an initial draft of the algorithm has already been provided to the student for review and to aid in developing an effective plan of action.

As presented in Section 2.3, the algorithms used in the current area selection method tackle a simplified version of Problem 1, where the diameter of the input set is a priori constrained using a partitioning technique. In this context, Eppstein et al.'s algorithm [11] is an ideal replacement for the current exact algorithm, as it significantly reduces time complexity (from exponential to polynomial) while still providing exact results. For this reason, in addition to our new algorithm, we plan to implement the algorithm by Eppstein et al..

The following research question provides an exact definition of the task:

RQ2: *How can we implement our new algorithm and Eppstein et al.'s algorithm [11] in the most computationally efficient way, and what optimizations can be applied to achieve good performance in practice?*

In the final part of this project, we plan to run several experiments benchmarking the implementations for our new algorithm and Eppstein et al.'s algorithm. These experiments aim to analyse the dependence of the algorithms' running time on the point set size and dispersion. Additionally, we will examine how the density and mitotic count (MC) of the computed region by Eppstein et al.'s algorithm compare to the diameter and MC of the optimal region found by the new algorithm.

We measure the point set dispersion using the average nearest neighbour distance (ANN) and its standard deviation, which is computed as $\text{ANN} = \frac{1}{n}\sum_{i=1}^{n} d_i$ where d_i is the distance from the nearest neighbour of p_i.

We perform these experiments using synthetic point sets of increasing size

generated in a square. For each size, we generate point sets with increasing ANN using a Gaussian distribution with increasing σ and a uniform distribution. The maximum area and diameter used by both algorithms (except for Eppstein et al.'s algorithm using only the area) will be fixed to a constant value for all experiments. We cannot use the point sets from MIDOG21, as their varying sizes and ANN values prevent us from setting up a controlled experiment that effectively addresses our objectives.

We also plan to run experiments with both algorithms for samples from MIDOG21. We will consider a subset of samples from MIDOG21 having visually different patterns and sizes.

We phrase the research question precisely:

RQ3: *How do the running times of our new algorithm and Eppstein et al.'s algorithm depend on the point set size and ANN, and how does the mitotic count (MC) computed by Eppstein et al.'s algorithm compare to the optimal MC generated by the new algorithm?*

- ***RQ3.1:*** *How do the running times of our new algorithm and Eppstein et al.'s algorithm depend on the point set size?*

- ***RQ3.2:*** *How do the running times of our new algorithm and Eppstein et al.'s algorithm depend on the point set ANN?*

- ***RQ3.3:*** *How do the diameter and mitotic count (MC) of the region found by Eppstein et al.'s algorithm compare to the optimal diameter and MC computed by the new algorithm?*

Due to the strict timeline of this master's project, we have chosen to focus on the three primary research questions presented. Whether an ϵ-approximation algorithm is feasible for Problem 1 remains uncertain. However, if prof. van Kreveld and Staals are able to develop such an algorithm and time permits, we would consider addressing an additional research question, which we present here for completeness:

RQ4: *How can we implement the ϵ-approximation algorithm in the most computationally efficient way, and what optimizations can be applied to achieve*

good performance in practice?

Performing additional experiments to measure the performance of the ϵ-approximation algorithm would require too much time. For this reason, we expect to leave these experiments for future work.

We have not considered a comparison between the new algorithm and the current area selection method for a variety of reasons. The current exact algorithm has exponential time complexity and can process sets of up to 25 points while our algorithm is expected to have polynomial time complexity and be able to process several hundreds of points. Comparing their output quality would be unnecessary, as both algorithms produce the same exact result. The heuristic algorithm is instead expected to be always faster than the new algorithm. A comparison between the heuristic algorithm output quality and the actual exact solution has been already carried out and presented in [8].

4 Methodology

4.0.1 What is the most suitable measure for quantifying the extension of a DCP that offers a compact representation and is feasible for algorithmic implementation?

We will analyse three extension measures applicable to convex polygons: aspect ratio, width and diameter. Based on Problem 1 requirements, we will compare them in terms of compactness, conveyed information and algorithmic complexity.

4.0.2 How can we implement our new algorithm and Eppstein et al.'s algorithm [11] in the most computationally efficient way, and what practical optimizations can be applied to achieve optimal performance?

We plan to develop both algorithms using a Test Driven Development (TDD) approach. We will use a low-level programming language (C) to achieve an efficient implementation for both algorithms. We will use the integrated environment in Visual Studio for development and debugging and several external libraries for visualization (RayLib), unit testing (Unity Test) and benchmarking (ubench.h). A version control system will be used to keep

track of the changes and the overall evolution of the codebase, which will be hosted on GitHub. The components of both algorithms will be unit tested using synthetically generated data.

In the final report, we will provide a link to our public implementations and a thorough description of our implementation choices.

This same methodology applies to the optional RQ 4.

4.0.3 How do the running times of our new algorithm and Eppstein et al.'s algorithm depend on the point set size and ANN, and how does the mitotic count (MC) computed by Eppstein et al.'s algorithm compare to the optimal MC generated by the new algorithm?

The sub-research questions RQ 3.1 and RQ 3.2 will be answered with several experiments using synthetic data generated in a 20x20mm square. For RQ 3.1, we plan to have an input size ranging between $250 \leq n \leq 500$ with a step of 25, so in total, 11 different values. The points will be generated using a uniform distribution.

For RQ 3.2, we plan to fix the input size to a value $250 \leq n \leq 500$ and generate different point sets with decreasing ANN. The first point set is a grid of equally spaced points, while the remaining point sets are generated using a gaussian distribution centred at $\mu = 10mm$ with decreasing standard deviation between $1 \leq \sigma \leq 10$ and step of 1, so in total 10 different values.

For both experiments, we expect to disable CPU scaling and Hyperthreading and repeat the test at least 10 times. This will prevent severe fluctuations in the measured runtime (and thus lower runtime variance) for both algorithms. Both the maximum allowed area and diameter are fixed for all experiments to $a_{\max} = 2mm^2$ and $d_{\max} = 4mm$.

For RQ 3.3, we plan to measure the difference in diameter and MC between the new algorithm and Eppstein et al.'s algorithm. We will perform this measurements during the experiments for RQ 3.1 and RQ 3.2.

Finally, we will run the same experiments using samples from MIDOG21 having visually different patterns and sizes. We plan to provide a table in the appendix of the final report listing for each point set tested the following information: the name of the sample, a picture showing the points distribution, the ANN of the point set (including the standard deviation) and the running time and MC of both algorithms.

The results for all three research questions will be presented with multiple plots and thoroughly discussed in the final report.

In the final report, we will provide a link to our public implementations and a thorough analysis of our implementation choices.

5 Time Plan

The student will stick to this time plan as much as possible. The unallocated time may be used for unforeseen reasons or delays.

Task	**Deadline**
Setup development environment	20/11/2024
Study the new algorithm	27/11/2024
Implement the new algorithm and Eppstein et al.'s algorithm [11]	10/01/2025
Setup and run experiments	15/01/2025
Analyze and plot the experimental data produced in the previous step	23/01/2025
Analyze different extension measures for convex polygons (width, aspect ratio and diameter)	28/01/2025
Prepare the final report	28/02/2025
Prepare the presentation slides	12/03/2025
Improve the final report and presentation slides	24/03/2025
Final deadline	30/04/2025

Table 1: Expected timeline

References

[1] E. M. Arkin, S. Khuller, and J. S. Mitchell, "Geometric knapsack problems," *Algorithmica*, vol. 10, no. 5, p. 399–427, Nov. 1993. [Online]. Available: https://doi.org/10.1007/BF01769706

[2] Y. Xu, M. Gong, Y. Wang, Y. Yang, S. Liu, and Q. Zeng, "Global trends and forecasts of breast cancer incidence and deaths," *Scientific Data*, vol. 10, no. 1, p. 334, May 2023. [Online]. Available: https://doi.org/10.1038/s41597-023-02253-5

[3] E. A. Rakha, M. E. El-Sayed, A. H. Lee, C. W. Elston, M. J. Grainge, Z. Hodi, R. W. Blamey, and I. O. Ellis, "Prognostic significance of nottingham histologic grade in invasive breast carcinoma," *Journal of Clinical Oncology*, vol. 26, no. 19, pp. 3153–3158, 2008, pMID: 18490649. [Online]. Available: https://ascopubs.org/doi/abs/10.1200/JCO.2007.15.5986

[4] J. M. Chang, A. E. McCullough, A. C. Dueck, H. E. Kosiorek, I. T. Ocal, T. K. Lidner, R. J. Gray, N. Wasif, D. W. Northfelt, K. S. Anderson, and B. A. Pockaj, "Back to basics: Traditional nottingham grade mitotic counts alone are significant in predicting survival in invasive breast carcinoma," *Annals of Surgical Oncology*, vol. 22, pp. 509–515, 2015. [Online]. Available: https://api.semanticscholar.org/CorpusID:22671536

[5] M. Veta, P. van Diest, M. Jiwa, S. Al-Janabi, and J. Pluim, "Mitosis counting in breast cancer: object-level interobserver agreement and comparison to an automatic method," *PLoS ONE*, vol. 11, no. 8, pp. 1–13, Aug. 2016.

[6] M. Bonert and A. J. Tate, "Mitotic counts in breast cancer should be standardized with a uniform sample area," *BioMedical Engineering OnLine*, vol. 16, 2017. [Online]. Available: https://api.semanticscholar.org/CorpusID:255863374

[7] M. Aubreville, N. Stathonikos, C. A. Bertram, R. Klopfleisch, N. ter Hoeve, F. Ciompi, F. Wilm, C. Marzahl, T. A. Donovan, A. Maier, J. Breen, N. Ravikumar, Y. Chung, J. Park, R. Nateghi, F. Pourakpour, R. H. Fick, S. Ben Hadj, M. Jahanifar, A. Shephard,

J. Dexl, T. Wittenberg, S. Kondo, M. W. Lafarge, V. H. Koelzer, J. Liang, Y. Wang, X. Long, J. Liu, S. Razavi, A. Khademi, S. Yang, X. Wang, R. Erber, A. Klang, K. Lipnik, P. Bolfa, M. J. Dark, G. Wasinger, M. Veta, and K. Breininger, "Mitosis domain generalization in histopathology images — the midog challenge," *Medical Image Analysis*, vol. 84, p. 102699, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841522003279

[8] N. Stathonikos, M. Aubreville, S. de Vries, F. Wilm, C. Bertram, M. Veta, and P. van Diest, "Breast cancer survival prediction using an automated mitosis detection pipeline," *Journal of Pathology: Clinical Research*, vol. 10, no. 6, Nov. 2024, © 2024 The Author(s). The Journal of Pathology: Clinical Research published by The Pathological Society of Great Britain and Ireland and John Wiley Sons Ltd.

[9] M. Aubreville, C. A. Bertram, N. Stathonikos, M. Veta, T. Donovan, N. ter Hoeve, F. Ciompi, C. Marzahl, F. Wilm, K. Breininger, A. Maier, and R. Klopfleisch, "MItosis DOmain Generalization Challenge (MICCAI- MIDOG 2021) Training Data," Mar. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.4643381

[10] E. M. Arkin, S. Khuller, and J. S. Mitchell, "Geometric knapsack problems," *Algorithmica*, vol. 10, no. 5, p. 399–427, Nov. 1993. [Online]. Available: https://doi.org/10.1007/BF01769706

[11] D. Eppstein, M. Overmars, G. Rote, and G. Woeginger, "Finding minimum areak-gons," *Discrete & Computational Geometry*, vol. 7, no. 1, pp. 45–58, Jan 1992. [Online]. Available: https://doi.org/10.1007/BF02187823

[12] P. Fischer, "Sequential and parallel algorithms for finding a maximum convex polygon," *Comput. Geom. Theory Appl.*, vol. 7, no. 3, p. 187–200, Feb. 1997. [Online]. Available: https://doi.org/10.1016/0925-7721(95)00035-6

[13] D. Eppstein, "New algorithms for minimum area k-gons," in *Proceedings of the Third Annual ACM-SIAM Symposium on Discrete Algorithms*, ser. SODA '92. USA: Society for Industrial and Applied Mathematics, 1992, p. 83–88.

[14] V. Keikha, "On optimal w-gons in convex polygons," *CoRR*, vol. abs/2103.01660, 2021. [Online]. Available: https://arxiv.org/abs/2103.01660

[15] J. S. Mitchell, G. Rote, G. Sundaram, and G. Woeginger, "Counting convex polygons in planar point sets," *Information Processing Letters*, vol. 56, no. 1, pp. 45–49, 1995. [Online]. Available: https://www.sciencedirect.com/science/article/pii/0020019095001305

[16] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, ser. KDD'96. AAAI Press, 1996, p. 226–231.

[17] M. I. Shamos, *Computational geometry.* Ph.D. dissertation, Yale University, 1978.