# Peeling Digital Potatoes

Loïc Crombez Guilherme D. da Fonseca
Yan Gérard

Université Clermont Auvergne and LIMOS, Clermont-Ferrand, France

**Abstract**

The potato-peeling problem (also known as convex skull) is a fundamental computational geometry problem that consist in finding the largest convex shape inside a given polygon. The fastest algorithm to date runs in $O(n^8)$ time for a polygon with $n$ vertices that may have holes. In this paper, we consider a digital version of the problem. A set $K \subset \mathbb{Z}^2$ is *digital convex* if $\text{conv}(K) \cap \mathbb{Z}^2 = K$, where $\text{conv}(K)$ denotes the convex hull of $K$. Given a set $S$ of $n$ lattice points, we present polynomial time algorithms for the problems of finding the largest digital convex subset $K$ of $S$ (*digital potato-peeling problem*) and the largest union of two digital convex subsets of $S$. The two algorithms take roughly $O(n^3)$ and $O(n^9)$ time, respectively. We also show that those algorithms provide an approximation to the continuous versions.

## 1 Introduction

The *potato-peeling problem* [24] (also known as *convex skull* [35]) consists of finding the convex polygon of maximum area that is contained inside a given polygon (possibly with holes) with $n$ vertices. The fastest exact algorithm known takes $O(n^7)$ time without holes and $O(n^8)$ if there are holes [12]. The problem is arguably the simplest geometric problem for which the fastest exact algorithm known is a polynomial of high degree and this high complexity motivated the study of approximation algorithms [11, 26]. Multiple variations of the problem have been considered, including triangle-mesh [1] and orthogonal [19, 36] versions. In this paper, we consider a digital geometry version of the problem.

Digital geometry is the field of mathematics that studies the geometry of points with integer coordinates, also known as *lattice points* [28]. Different definitions of convexity in $\mathbb{Z}^2$ have been investigated, such as digital line, triangle, line [27], HV (for Horizontal and Vertical [4]), and Q (for Quadrant [17]) convexities. These definitions guarantee that a digital convex set is connected (in terms of the induced grid subgraph), which simplifies several algorithmic problems.

Throughout this paper, however, we use the main and original definition of digital convexity from the geometry of numbers [25]. A set of lattice points $K \subset \mathbb{Z}^d$ is *digital convex* if $\text{conv}(K) \cap \mathbb{Z}^d = K$, where $\text{conv}(K)$ denotes the convex hull of $K$. This definition does not guarantee connectivity of the grid subgraph, but provides several other important

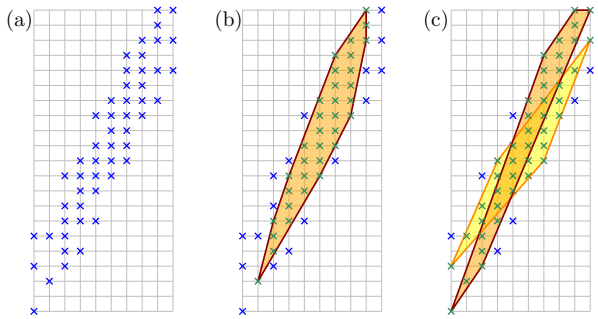


Figure 1: (a) Input lattice set $S$. (b) Largest digital convex subset of $S$ (Problem 1). (c) Largest union of two digital convex subsets of $S$ (Problem 2).

mathematical properties, such as being preserved under certain affine transformations. The authors recently showed how to efficiently test digital convexity in the plane [15]. A natural question is to determine the largest digital convex subset.

The *digital potato-peeling problem* is defined as follows and is illustrated in Figure 1(a,b).

**Problem 1** (Digital potato-peeling)**.** *Given a set $S \subset \mathbb{Z}^2$ of $n$ lattice points described by their coordinates, determine the largest set $K \subseteq S$ that is digital convex (i.e., $\text{conv}(K) \cap \mathbb{Z}^2 = K$), where largest refers to the area of $\text{conv}(K)$.*

Our algorithms can easily be modified to maximize the number of points in $K$ instead of the area of $\text{conv}(K)$. Compared to the continuous version, the digital geometry setting allows us to explicitly represent the whole set of input points, instead of limiting ourselves to polygonal shapes with polygonal holes. Note that the input of the continuous and digital problems is intrinsically different, hence we cannot compare the complexity of the two problems. Related continuous problems have been studied, such as the maximum volume of an empty convex body amidst $n$ points [18], or the *optimal island problem* [6, 22], in which we are given two sets $S_p, S_n \subset \mathbb{R}^2$, and the goal is to determine that largest subset $K \subseteq S_p$ such that $\text{conv}(K) \cap S_n = \emptyset$.

Heuristics for the digital potato-peeling problem have been presented in [10, 13], but no exact algorithm was known. We solve this open problem by providing the first polynomial-time exact algorithm.

We also solve the question of covering the largest area with two digital convex subsets. The problem is defined as follows and is illustrated in Figure 1(a,c).

**Problem 2** (Digital 2-potato peeling)**.** *Given a set $S \subset \mathbb{Z}^2$ of $n$ lattice points described by their coordinates, determine the largest set $K = K_1 \cup K_2 \subseteq S$ such that $K_1$ and $K_2$ are both digital convex, where largest refers to the area of $\text{conv}(K_1) \cup \text{conv}(K_2)$.*

A related continuous problem consists of completely covering a polygon by a small number of convex polygons inside of it. O'Rourke showed that covering a polygon with the minimum number of convex polygons is decidable [29, 30], but the problem has been shown to be NP-hard with or without holes [16, 31]. Shermer [34] presents a linear time algorithm for the case of two convex polygons and Belleville [8] provides a linear time algorithm for three. We are not aware of any previous results on finding a fixed (non-unit) number of convex polygons inside a given polygon and maximizing the area covered.

### Our results

We present polynomial time algorithms to solve each of these two problems. In Section 2, we show how to solve the digital potato-peeling problem in $O(n^3 + n^2 \log r)$ time, where $r$ is the diameter of the input $S$. We adapt an algorithm designed to solve the optimal island problem [6, 22]. This algorithm builds the convex polygon $\text{conv}(K)$ through its triangulation. We use Pick's theorem [32] to test digital convexity for each triangle and the $O(\log r)$ factor in the running time comes from the gcd computation required to apply Pick's theorem. The algorithm makes use of the following two properties: (i) it is possible to triangulate $K$ using only triangles that share a common bottom-most vertex $v$ and (ii) if the polygons lying on both sides of one such triangle (including the triangle itself) are convex, then the whole polygon is convex.

These two properties are no longer valid for Problem 2, in which the solution $\text{conv}(K_1) \cup \text{conv}(K_2)$ is the union of two convex polygons. Also, since convex shapes are not pseudo-disks (the boundaries may cross an arbitrarily large number of times), separating the input with a constant number of lines is not an option. Instead of property (i), our approach uses the fact that the union of two (intersecting) convex polygons can be triangulated with triangles that share a common vertex $\rho$ (that may not be a vertex of either convex polygon). Since $\rho$ may not have integer coordinates, we can no longer use Pick's theorem, and resort to the formulas from Beck and Robins [7] or the algorithm from Barvinok [5] to count the lattice points inside each triangle in $O(\text{polylog}\, r)$ time.

Furthermore, to circumvent the fact that the solution no longer obeys property (ii), we use a directed acyclic graph (DAG) that encapsulates the orientation of the edges of both convex polygons. For those reasons, the running time of our algorithm for Problem 2 increases to $O(n^9 + n^6 \,\text{polylog}\, r)$. The corresponding algorithm is described in Section 3.

In Section 4, we show that a solution to the digital version of the problems provides an approximation to the continuous versions, establishing a formal connection between the continuous and digital versions.

Reducing the complexity of our algorithms or extending the result to higher numbers of convex polygons remain intriguing open questions, which are discussed in Section 5. Throughout, we assume the RAM model of computation, in which elementary operations on the input coordinates take constant time.

## 2 Digital Potato Peeling

In this section, we present an algorithm to solve the digital potato-peeling problem in $O(n^3 + n^2 \log r)$ time, where $n$ is the number of input points and $r$ is the diameter of the point set.

Fischer [22] and Bautista et al. [6] showed how to solve the following related problem in $O(n^3)$ time, where $n$ is the total number of points.

**Problem 3** (Optimal Island)**.** *Given two sets $S_p, S_n \subset \mathbb{R}^2$, determine the largest subset $K \subseteq S_p$ such that* $\text{conv}(K) \cap S_n = \emptyset$.

The potato peeling problem 1 for an input $S \subset \mathbb{Z}^2$ is the optimal island problem with $S_p = S$ and $S_n = \mathbb{Z}^2 \setminus S_p$. Restricting the problem to the bounding box of $S_p$, makes $S_n$

finite as $|S_n| = O(r^2)$. The resulting $O(r^6)$ complexity being very large relative to $r$, we do not use this direct approach. Nevertheless, the algorithm provides some key insights.

The algorithm consists of two phases. First, a list $\mathcal{T}$ of all *valid* triangles is computed. A triangle $\triangle$ is said to be valid if its vertices are a subset of $S_p$ and if $\triangle \cap S_n = \emptyset$. Second, using $\mathcal{T}$ and the fact that every convex polygon has a fan triangulation in which all the triangles share a common bottom vertex, the solution is computed by appending valid triangles using dynamic programming. In order to adapt this algorithm to solve the digital potato peeling, it suffices to compute the list of valid triangles $\mathcal{T}$.

### 2.1 Valid Triangles

For any triangle whose vertices are lattice points $\triangle$, and any digital set $S$: $|\triangle \cap S| = |\triangle \cap \mathbb{Z}^2|$ implies that $\triangle$ is valid. As in [6], we use the following result of Eppstein et al. [20] to compute $|\triangle \cap S|$.

**Theorem 1.** *Let $S$ be a set of $n$ points in the plane. The set $S$ can be preprocessed in $O(n^2)$ time and space in order to, for any query triangle $\triangle$ with vertices in $S$, compute the number of points $|\triangle \cap S|$ in constant time.*

In order to compute $|\triangle \cap \mathbb{Z}^2|$, first, for all pairs of points $p_1, p_2 \in S$, we compute the number of lattice points lying on the edge $p_1p_2$ using a gcd computation. This takes $O(n^2 \log r)$ time, where $r$ is the diameter of $S$. Now, using Pick's formula [32] which requires to compute both $area(\triangle)$ and the number of lattice points lying on the edges of $\triangle$, we determine in $O(1)$ time the validity of a triangle. Since there are $O(n^3)$ triangles with vertices in $S$, the list $\mathcal{T}$ of all valid triangles is computed in $O(n^3 + n^2 \log r)$ time. Using $\mathcal{T}$, the algorithm of Bautista et al. [6] determines the largest convex polygon formed by triangles in $\mathcal{T}$ in $O(n^3)$ time. Hence, we have the following theorem.

**Theorem 2.** *There exists an algorithm to solve Problem 1 (digital potato peeling) in $O(n^3 + n^2 \log r)$ time, where $n$ is the number of input points and $r$ is the diameter of the input.*

## 3 Digital 2-Potato Peeling

In this section, we show how to find two digital convex sets $K_1, K_2$, maximizing the area of $\text{conv}(K_1) \cup \text{conv}(K_2)$. We note that the solution described in this section can easily be adapted to solve the optimal 2-islands problem:

**Problem 4** (Optimal 2-Islands)**.** *Given two sets $S_p, S_n \subset \mathbb{R}^2$, determine the largest union of subsets $K_1 \cup K_2$ such that $K_1 \cup K_2 \subseteq S_p$, $\text{conv}(K_1) \cap S_n = \emptyset$ and $\text{conv}(K_2) \cap S_n = \emptyset$.*

Consider a solution of the digital 2-potato peeling problem. Either the two convex hulls intersect or they do not (Figure 2). We treat those two cases separately and the solution to Problem 2 is the largest among both. Hence, we consider the two following variations of the 2-potato-peeling problem.

**Problem 5** (Disjoint 2-potato peeling)**.** *Given a set $S \subset \mathbb{Z}^2$ of $n$ lattice points given by their coordinates, determine the largest two digital convex sets $K_1 \cup K_2 \subseteq S$ such that $\text{conv}(K_1) \cap \text{conv}(K_2) = \emptyset$.*

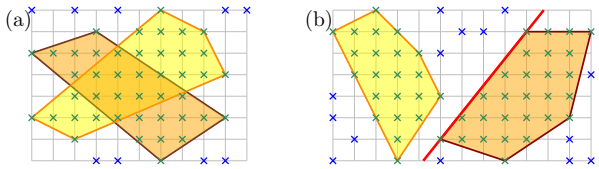


Figure 2: (a) The two optimal sets intersect. (b) The two optimal sets are disjoint and there is a supporting separating line.

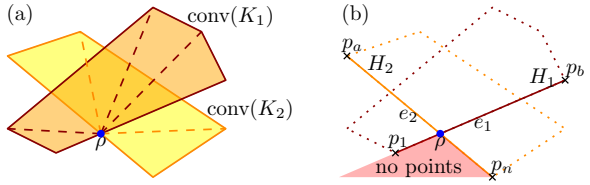


Figure 3: (a) A fan triangulation of two intersecting convex polygons from a point $\rho$. (b) Definitions used to solve Problem 7.

**Problem 6** (Intersecting 2-potato peeling)**.** *Given a set $S \subset \mathbb{Z}^2$ of $n$ lattice points given by their coordinates, determine the largest union of two digital convex sets $K_1 \cup K_2 \subseteq S$ such that* $\mathrm{conv}(K_1) \cap \mathrm{conv}(K_2) \neq \emptyset$*. In this case, largest means the maximum area of* $\mathrm{conv}(K_1) \cup \mathrm{conv}(K_2)$*.*

## 3.1 Disjoint Convex Polygons

Any two disjoint convex shapes can be separated by a straight line. Moreover two convex polygons can be separated by a supporting line of an edge of one of the convex polygons (Figure 2(b)).

For each ordered pair of distinct points $p_1, p_2 \in S$, we define two subsets $S_1, S_2$. The set $S_1$ contains the points on the line $p_1, p_2$ or to the left of it (according to the direction $p_2 - p_1$). The set $S_2$ contains the remaining points.

For each pair of sets $S_1, S_2$, we independently solve Problem 1 for each of $S_1$ and $S_2$. Since there are $O(n^2)$ pairs and each pair takes $O(n^3 + n^2 \log r)$ time, we solve Problem 5 in $O(n^5 + n^4 \log r)$ time.

## 3.2 Intersecting Convex Polygons

The more interesting case is when the two convex polygons intersect (Problem 6). Note that it is possible to triangulate the union of two convex polygons that share a common boundary point $\rho$ using a fan triangulation around $\rho$ (Figure 3). Hence we consider the following rooted version of the problem.

**Problem 7** (Rooted 2-potato peeling)**.** *Given a set $S \subset \mathbb{Z}^2$ of $n$ lattice points represented by their coordinates and two edges $e_1, e_2 \in S^2$ that cross at a point $\rho$, determine the largest union of two digital convex sets $K_1, K_2 \subseteq S$ such that $e_1$ is an edge of* $\mathrm{conv}(K_1)$ *and $e_2$ is an edge of* $\mathrm{conv}(K_2)$*.*

Let $\rho$ be the intersection point of $e_1, e_2$. The strategy of the algorithm to solve Problem 7 is to encode the problem into a DAG $(V, E)$ whose longest directed path corresponds to the desired solution. To avoid confusion, we use the terms *node* and *arc* for the DAG and keep the terms *vertex* and *edge* for the polygons. It is well known that the longest directed path in a DAG $(V, E)$ can be calculated in $O(|V| + |E|)$ time [33].

Let $\mathcal{T}$ be the set of valid triangles with two vertices from $S$ and $\rho$ as the remaining vertex. The nodes $V = \mathcal{T}^2 \cup \{v_0\}$ are ordered pairs of valid triangles and a starting node $v_0$. The number of nodes is $|V| = O(n^4)$. Before we define the arcs, we give an intuitive idea of our objective.

Each node $(\triangle_1, \triangle_2) \in V$ is such that $\triangle_1$ (resp. $\triangle_2$) is used to build the fan triangulation of $\text{conv}(K_1)$ (resp. $\text{conv}(K_2)$). The arcs will be defined in a way that, at each step as we walk through a path of the DAG, we add one triangle to either $\text{conv}(K_1)$ or to $\text{conv}(K_2)$. The arcs enforce the convexity of both $\text{conv}(K_1)$ and $\text{conv}(K_2)$. Furthermore, we enforce that we always append a triangle to the triangulation that is the least advanced of the two (in clockwise order), unless we have already reached the last triangle of $\text{conv}(K_1)$. This last condition is important to allow us to define the arc lengths in a way that corresponds to the area of the union of the two convex polygons. Figure 4 illustrates the result of following a path on the DAG.

The edge $e_1$ (respectively, $e_2$) from the problem input defines two halfplanes, one on each side. Let $H_1$ (resp. $H_2$) be the halfplane that contains $K_1$ (resp. $K_2$). We have not yet determined $K_1$ or $K_2$, but all four possibilities of halfplanes may be tried independently. From now on, we only consider the $O(n)$ points of $S$ lying in the region $H_1 \cup H_2$. Let $p_1, \ldots, p_n$ be the points of $S$ sorted clockwise around $\rho$, breaking ties arbitrarily. The edge $e_1$ (resp. $e_2$) has $p_1$ (resp. $p_n$) as a vertex. We define the indices $a < b$ such that $e_1 = (p_1, p_b), e_2 = (p_a, p_n)$ (Figure 3).

We are now ready to define the set $E$ of arcs of the DAG. There are three types of arcs. The *type-0* arcs start from the initial node $v_0$ to $(\triangle_1, \triangle_2)$ if $p_1$ is a vertex of $\triangle_1$ and $p_a$ a vertex of $\triangle_2$. These two triangles of vertices $\rho, p_1, p_j$ with $j > 1$ and $\rho, p_a, p_v$ with $v > a$ are respectively bounded by the edges $e_1$ and $e_2$. They initialize the triangulations of our two polygons $\text{conv}(K_1)$ and $\text{conv}(K_2)$. There are $O(n^2)$ type-0 arcs.

A *type-1* arc corresponds to advancing the triangulation of $\text{conv}(K_1)$, while a *type-2* arc corresponds to advancing the triangulation of $\text{conv}(K_2)$. There are $O(n)$ type-1, 2 arcs coming out of each node. A *type-1* arc goes from $(\triangle_1, \triangle_2)$ to $(\triangle_3, \triangle_2)$ if:

- the quadrilateral $\triangle_1 \cup \triangle_3$ is convex,
- $\triangle_1$ has vertices $\rho, p_i, p_j$ with $i < j < b$,
- $\triangle_2$ has vertices $\rho, p_u, p_v$ with $a \leq u < v$,
- $\triangle_3$ has vertices $\rho, p_j, p_k$ with $j < k \leq b$,
- and $j \leq v$.

Similarly, there is a type-2 arc from $(\triangle_1, \triangle_2)$ to $(\triangle_1, \triangle_4)$ if:

- the quadrilateral $\triangle_2 \cup \triangle_4$ is convex,
- $\triangle_1$ has vertices $\rho, p_i, p_j$ with $i < j \leq b$,

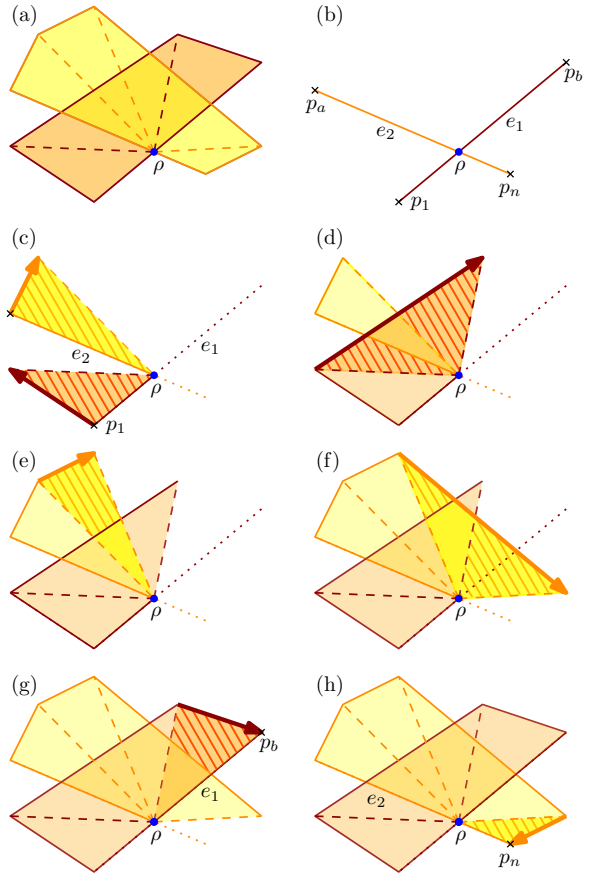


Figure 4: Steps of the algorithm from Section 3.2. Figure (a) represents the solution, while Figures (b) to (h) represent the triangulation obtained at each node of a path. The newly covered area that is assigned as the length of the corresponding arc is marked. In (b), we have the initial pair of edges $e_1, e_2$ which corresponds to the starting vertex $v_0$. After following a type-0 arc, a first pair of triangles with vertices $p_1$ and $p_a$ is obtained in (c). The triangle $\triangle_1$ is brown and triangle $\triangle_2$ yellow. From (c) to (d), we follow a type-1 arc. The triangle $\triangle_1$ (less advanced than triangle $\triangle_2$) advances. From (d) to (e), we follow a type-2 arc, since triangle $\triangle_2$ is less advanced. From (e) to (f) we have again a type-2 arc, and from (f) to (g) we have a type-1 arc. In (g), the triangle $\triangle_1$ has reached the final node $p_b$ and cannot advance anymore. We have only type-2 arcs to follow until $\triangle_2$ reaches $p_n$, at a node in $V_1$.

- $\triangle_2$ has vertices $\rho, p_u, p_v$ with $a \leq u < v$,
- $\triangle_4$ has vertices $\rho, p_v, p_w$ with $v < w$,
- and either $v \leq j$ or $j = b$.

The length of each arc corresponds to the area of the new region covered by appending a new triangle by following the arc. Therefore, the length of a type-0 arc from $v_0$ to $(\triangle_1, \triangle_2)$ is the area of $\triangle_1 \cup \triangle_2$. The length of a type-1 arc from $(\triangle_1, \triangle_2)$ to $(\triangle_3, \triangle_2)$ is defined as the area of $\triangle_3 \setminus \triangle_2$. Similarly, the length of a type-2 arc from $(\triangle_1, \triangle_2)$ to $(\triangle_1, \triangle_4)$ is defined as the area of $\triangle_4 \setminus \triangle_1$.

We define a set of *end* nodes $V_1$ as follows. A node $(\triangle_1, \triangle_2)$ is an end node if $p_b$ is a vertex of $\triangle_1$ and $p_n$ is a vertex of $\triangle_2$. The construction of the DAG allows us to prove the following lemma.

**Lemma 3.** *There is a bijection between the directed paths of the DAG $(V, E)$ (starting from $v_0$ and ending in $V_1$) and the digital convex sets $K_1, K_2 \subset S$ such that $e_1$ is an edge of $\operatorname{conv}(K_1)$ and $e_2$ is an edge of $\operatorname{conv}(K_2)$. Furthermore, the length of each path is equal to the corresponding area of $\operatorname{conv}(K_1) \cup \operatorname{conv}(K_2)$. (We assume that $K_1$ (resp. $K_2$) lie above the supporting line of $e_1$ (resp. $e_2$).)*

*Proof.* First we show that the existence of two digital convex sets $K_1, K_2 \subset S$ as in the lemma statement implies the existence of a directed path in the DAG as in the lemma statement. Let $K_1$ (resp. $K_2$) be two convex sets lying above the supporting line of $e_1$ (resp. $e_2$). Both $\operatorname{conv}(K_1)$ and $\operatorname{conv}(K_2)$ contain $\rho$ as a boundary point and hence can be triangulated from $\rho$. It is easy to see that there is a path corresponding to this triangulation. Next, we show that the converse also holds.

The definition of the arcs is such that advancing through one of them adds a triangle to one of the two polygons while preserving convexity, which ensures that all paths correspond to convex polygons. Furthermore, the starting node ensures that the two convex polygons respectively start from $p_1$ and $p_a$, while the set of ending nodes ensure that the two convex polygons respectively end at $p_b$ and $p_n$. Hence all paths from $v_0$ to $V_1$ correspond to two convex polygons that fit the lemma statement, one from edge $e_1 = p_1, p_b$ and one from edge $e_2 = p_a, p_n$. The validity test on each triangle ensures that the paths describes digital convex sets.

The definition of the arcs enforces that we only move forward the least advanced triangle, that is the triangle that has the minimum maximum index among its vertices. The only exception is when $\operatorname{conv}(K_1)$ is completed, that is the triangle with vertex $p_b$ has been added to its triangulation. This ensures that the new area covered by a type-$1, 2$ arc is simply the set theoretic difference of two triangles (instead of a triangle and an arbitrary convex object). As the length of the arcs is defined as the area of the difference of the two triangles, the total length of the path is equal to the area of the union of the two convex polygons. Hence each path from $v_0$ to $V_1$ describe two digital convex sets $K_1, K_2 \in S$ such that $e_1$ is an edge of $\operatorname{conv}(K_1)$ and $e_2$ is an edge of $\operatorname{conv}(K_2)$, and the length of each path is equal to the corresponding area of $\operatorname{conv}(K_1) \cup \operatorname{conv}(K_2)$. □

**Theorem 4.** *There exists an algorithm to solve Problem 2 (digital 2-potato peeling) in $O(n^9 + n^6 \operatorname{polylog} r)$ time, where $n$ is the number of input points and $r$ is the diameter of the input.*

*Proof.* As explained in Section 3.1, solving the disjoint case (Problem 5) takes $O(n^5 + n^4 \log r)$ time. Next, we show how to solve the rooted intersecting case (Problem 7) in $O(n^5 + n^2 \operatorname{polylog} r)$ time, proving the theorem.

Assume without loss of generality that $K_1, K_2$ are respectively above the supporting lines of $e_1, e_2$ (all four possibilities may be tried independently).

Our algorithm starts by computing the DAG $(V, E)$ with $O(n^4)$ nodes, each representing a pair of triangles. Since each node has at most $O(n)$ incoming arcs, the number of arcs is $O(n^5)$. Hence the longest path can be found in $O(n^5)$ time.

To build the set of nodes $V$, we need to test the validity of $O(n^2)$ triangles. Since $\rho$ may not be a lattice point, Pick's theorem [32] cannot be used. Still, $\rho$ is a rational point with denominators bounded by $O(r^2)$. Hence, we can use either the formulas from Beck and Robins [7] or the algorithm from Barvinok [5] to calculate the number of lattice points $|T \cap \mathbb{Z}^2|$ inside each triangle $T$ in $O(\operatorname{polylog} r)$ time. As in Section 2, we compute $|T \cap S|$ using a triangle range counting query, which takes $O(\log n)$ time after preprocessing $S$ in $O(n^2)$ time [14]. The triangle is valid if and only if $|T \cap \mathbb{Z}^2| = |T \cap S|$. The two steps to test the validity of a triangle take $O(\operatorname{polylog} r)$ and $O(\log n)$ time. Since the diameter $r$ of $n$ lattice points is $\Omega(\sqrt{n})$, the dominating term is $O(\operatorname{polylog} r)$. Hence, we test the validity of each triangle in $O(\operatorname{polylog} r)$ time, which gives a total time of $O(n^2 \operatorname{polylog} r)$ to build the list of valid triangles required to build $V$.

Consequently, we solve Problem 7 in $O(n^5 + n^2 \operatorname{polylog} r)$ time. To obtain a solution to Problem 2, we note that there are $O(n^2)$ candidates for the edge $e_1$, as well as for the edge $e_2$. Testing all $O(n^4)$ possible edges $e_1, e_2$, we achieve the claimed running time of $O(n^9 + n^6 \operatorname{polylog} r)$ time. □

# 4 From Digital to Continuous

In this section, we show that the exact algorithms for the digital potato-peeling problem and the digital 2-potato-peeling problem can be used to compute an approximation of the respective continuous problems with an arbitrarily small approximation error. For simplicity, we focus on the potato-peeling problem, but the 2-potato-peeling case is analogous. We note that the reduction presented here does not lead to efficient approximation algorithms and is presented only to formally connect the continuous and digital versions of the problem.

**Problem 8** (Continuous potato-peeling)**.** *Given a polygon $P$ (that may have holes) of $n$ vertices, determine the largest convex polygon $K \subseteq P$, where largest refers to the area of $K$.*

We start with some definitions. Let $K_C$ be the polygon of the optimal solution to the continuous problem above and $A_C$ be the area of $K_C$. Given an approximation parameter $\varepsilon > 0$, we show how to obtain a set of lattice points $S \subseteq P$ such that the area $A_D$ of the convex hull of the solution $K_D$ of Problem 1 with input $S$ satisfies $|A_C - A_D| = O(r\varepsilon)$. In this section, we use lattice points that are not integers, but points with coordinates that are multiples of $\varepsilon$. Let $\Lambda_\varepsilon$ denote the set of all points with coordinates that are multiple of $\varepsilon$. Of course, a uniform scaling maps $\Lambda_\varepsilon$ to the integer lattice used in the remainder of the paper, and hence the integer lattice algorithms also apply to $\Lambda_\varepsilon$.

For a polygon $P$, the *erosion* of $P$, denoted $P^-$ is the subset of $P$ formed by points within $L_\infty$ distance at least $2\varepsilon$ of all points outside $P$ (Figure 5(a)). Let $A_C^-$ be the area of the optimal solution to the continuous potato-peeling problem with input $P^-$.

We only give here the main directions of the proof. A more detailed version of the proof can be found in the appendix 5. First, by bounding the number of lattice cells that a convex curve of a given length can cross, we bound by $O(r\varepsilon)$ the area difference between any convex polygon and the convex hull of the lattice points inside it. Then, we use an erosion of $2\varepsilon$ in order to smooth the input and avoid difficulties related to comb like input polygons. We bound the area difference by $O(r\varepsilon)$ between the solution of problem 1 for any polygon and the solution for the erosion of this polygon. Finally, despite the digital solution being potentially outside the input polygon, it can be shown that the area lying outside the input polygon is bounded by $O(r\varepsilon)$ which gives us the following theorem:

**Theorem 5.** *Let $A_C$ be the area of the solution $K_C$ of Problem 8 with input polygon $P$ of diameter $r$. Let $\varepsilon > 0$ be a parameter and $S = \Lambda_\varepsilon \cap P^-$, where $P^-$ is the erosion of $P$ by $2\varepsilon$ and $\Lambda_\varepsilon$ is the lattice of size $\varepsilon$. The area $A_D$ of the convex hull of the solution $K_D$ of Problem 1 with input $S$ satisfies $|A_C - A_D| = O(r\varepsilon)$.*

The polygon $\mathrm{conv}(K_D)$ in the previous theorem may partially extend outside $P$. Nevertheless, the solution $K_D$ of Problem 1 can be used to obtain a convex polygon $K \subseteq P$ which has an area $A$ satisfying $|A_C - A_D| = O(r\varepsilon)$.

# 5 Conclusion and Open Problems

The (continuous) potato peeling problem is a very peculiar problem in computational geometry. The fastest algorithms known have running times that are polynomials of substantially high degree. Also, we are not aware of any algorithms (or difficulty results) for the natural extensions to higher dimensions (even 3d) or to a fixed number of convex bodies.

In this paper, we focused on a digital version of the problem. Many problems in the intersection of digital, convex, and computational geometry remain open. Our study falls in the following framework of problems, all of which receive as input a set of $n$ lattice points $S \subset \mathbb{Z}^d$ for constant $d$ and are based on a fixed parameter $k \geq 1$.

1. Is $S$ the union of at most $k$ digital convex sets?
2. What is the smallest superset of $S$ that is the union of at most $k$ digital convex sets?
3. What is the largest subset of $S$ that is the union of at most $k$ digital convex sets?

In [15], the authors considered the first problem for $k = 1$, presenting polynomial time solutions (which may still leave room for major improvements for $d > 3$). We are not aware of any previous solutions for $k > 1$. In contrast, the continuous version of the problem is well studied. The case of $k = 1$ can be solved easily by a convex hull computation or by linear programming. Polynomial algorithms are known for $d = 2$ and $k \leq 3$ [8, 34], as well as for $d = 3$ and $k \leq 2$ [9]. The problem is already NP-complete

for $d = k = 3$ [9]. Hence, the continuous version remains open only for $d = 2$ and fixed $k > 3$.

It is easy to obtain polynomial time algorithms for the second problem when $k = 1$, since the solution consists of all points in the convex hull of $S$. The continuous version for $d = k = 2$ can be solved in $O(n^4 \log n)$ time [3]. Also, the orthogonal version of the problem is well studied (see for example [21]). We know of no results for the digital version.

In this paper, we considered the digital version of the third problem for $d = 2$ and $k = 1, 2$, presenting algorithms with respective running times of $O(n^3 + n^2 \log r)$ and $O(n^9 + n^6 \operatorname{polylog} r)$, where $r$ is the diameter of $S$. Since the first problem trivially reduces to the third problem, we also solved the first problem for $k = d = 2$ in $O(n^9 + n^6 \operatorname{polylog} r)$ time. It is surprising that we are not aware of any faster algorithm for the first problem in this particular case.

The third problem for $d > 2$ or $k > 2$ remains open. The DAG approach that we used for $d = 2$ is unlikely to generalize to higher dimensions, since there is no longer a single order by which to transverse the boundary of a convex polytope. Surprisingly, even the continuous version seems to be unresolved for $d > 2$ or $k \geq 2$.

# References


[1] B. Aronov, M. Van Kreveld, M. Löffler, and R. I. Silveira. Peeling meshed potatoes. *Algorithmica*, 60(2):349–367, 2011.

[2] C. Audet, P. Hansen, and F. Messine. Extremal problems for convex polygons. *Journal of Global Optimization*, 38(2):163–179, 2007.

[3] S. W. Bae, H.-G. Cho, W. Evans, N. Saeedi, and C.-S. Shin. Covering points with convex sets of minimum size. *Theoretical Computer Science*, 718:14–23, 2018.

[4] E. Barcucci, A. D. Lungo, M. Nivat, and R. Pinzani. Reconstructing convex polyominoes from horizontal and vertical projections. *Theoretical Computer Science*, 155(2):321–347, 1996.

[5] A. I. Barvinok. A polynomial time algorithm for counting integral points in polyhedra when the dimension is fixed. *Mathematics of Operations Research*, 19(4):769–779, 1994.

[6] C. Bautista-Santiago, J. M. Díaz-Báñez, D. Lara, P. Pérez-Lantero, J. Urrutia, and I. Ventura. Computing optimal islands. *Operations Research Letters*, 39(4):246–251, 2011.

[7] M. Beck and S. Robins. Explicit and efficient formulas for the lattice point count in rational polygons using Dedekind-Rademacher sums. *Discrete & Computational Geometry*, 27(4):443–459, Jan 2002.

[8] P. Belleville. On restricted boundary covers and convex three-covers. In *5th Canadian Conference on Computational Geometry (CCCG)*, pages 467–472, 1993.

[9] P. Belleville. Convex covers in higher dimensions. In *7th Canadian Conference on Computional Geometry (CCCG)*, pages 145–150, 1995.

[10] G. Borgefors and R. Strand. An approximation of the maximal inscribed convex set of a digital object. In *13th International Conference on Image Analysis and Processing (ICIAP)*, pages 438–445, 2005.

[11] S. Cabello, J. Cibulka, J. Kyncl, M. Saumell, and P. Valtr. Peeling potatoes near-optimally in near-linear time. *SIAM Journal on Computing*, 46(5):1574–1602, 2017.

[12] J. S. Chang and C. K. Yap. A polynomial solution for the potato-peeling problem. *Discrete & Computational Geometry*, 1(2):155–182, 1986.

[13] J.-M. Chassery and D. Coeurjolly. Optimal shape and inclusion: open problems. In *Mathematical Morphology: 40 Years On, International Symposium on Mathematical Morphology*, Computational Imaging and Vision. Springer Verlag, 2005.

[14] B. Chazelle, M. Sharir, and E. Welzl. Quasi-optimal upper bounds for simplex range searching and new zone theorems. *Algorithmica*, 8(1-6):407–429, 1992.

[15] L. Crombez, G. D. da Fonseca, and Y. Gérard. Efficient algorithms to test digital convexity. In *21st International Conference on Discrete Geometry for Computer Imagery (DGCI)*, 2019.

[16] J. Culberson and R. A. Reckhow. Covering polygons is hard. *Journal of Algorithms*, pages 17:2–44, 1994.

[17] A. Daurat. Salient points of q-convex sets. *International Journal of Pattern Recognition and Artificial Intelligence*, 15(7):1023–1030, 2001.

[18] A. Dumitrescu, S. Har-Peled, and C. D. Tóth. Minimum convex partitions and maximum empty polytopes. In F. V. Fomin and P. Kaski, editors, *Algorithm Theory – SWAT 2012*, pages 213–224, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[19] M. Dutt, A. Biswas, P. Bhowmick, and B. B. Bhattacharya. On finding an orthogonal convex skull of a digital object. *International Journal of Imaging Systems and Technology*, 21(1):14–27, 2011.

[20] D. Eppstein, M. Overmars, G. Rote, and G. Woeginger. Finding minimum area k-gons. *Discrete & Computational Geometry*, 7(1):45–58, 1992.

[21] C. Evrendilek, B. Genç, and B. Hnich. Covering points with minimum/maximum area orthogonally convex polygons. *Computational Geometry*, 54:32–44, 2016.

[22] P. Fischer. Sequential and parallel algorithms for finding a maximum convex polygon. *Computational Geometry*, 7:187–200, 02 1997.

[23] Y. Gérard, A. Vacavant, and J. Favreau. Tight bounds in the quadtree complexity theorem and the maximal number of pixels crossed by a curve of given length. *Theoretical Computer Science*, 624:41–55, 2016.

[24] J. E. Goodman. On the largest convex polygon contained in a non-convex n-gon, or how to peel a potato. *Geometriae Dedicata*, 11(1):99–106, 1981.

[25] P. M. Gruber. Geometry of numbers. In *Handbook of Convex Geometry, Part B*, pages 739–763. Elsevier, 1993.

[26] O. Hall-Holt, M. J. Katz, P. Kumar, J. S. Mitchell, and A. Sityon. Finding large sticks and potatoes in polygons. In *17th annual ACM-SIAM symposium on Discrete algorithm (SODA)*, pages 474–483, 2006.

[27] C. E. Kim and A. Rosenfeld. Digital straight lines and convexity of digital regions. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4(2):149–153, 1982.

[28] R. Klette and A. Rosenfeld. *Digital geometry: Geometric methods for digital picture analysis*. Elsevier, 2004.

[29] J. O'Rourke. The complexity of computing minimum convex covers for polygons. In *20th Allerton Conference on Communication, Control, and Computing*, pages 75–84, 1982.

[30] J. O'Rourke. The decidability of covering by convex polygons. Technical Report JHU-EECS 82-4, Johns Hopking University, 1982.

[31] J. O'Rourke. Some NP-hard polygon decomposition problems. *IEEE Transactions on Information Theory, IT-30*, pages 181–190, 1983.

[32] G. Pick. Geometrisches zur zahlenlehre. *Sitzungsberichte des Deutschen Naturwissenschaftlich-Medicinischen Vereines für Böhmen "Lotos" in Prag.*, v.47-48 1899-1900, 1899.

[33] R. Sedgewick and K. Wayne. *Algorithms*. Addison-Wesley Professional, 2011.

[34] T. C. Shermer. On recognizing unions of two convex polygons and related problems. *Pattern Recognition Letters , 14(9)*, pages 737–745, 1993.

[35] T. C. Woo. The convex skull problem. Technical report, Department of Industrial and Operations Engineering, University of Michigan, 1986.

[36] D. Wood and C. K. Yap. The orthogonal convex skull problem. *Discrete & Computational Geometry*, 3(4):349–365, 1988.

# Appendix

# From Digital to Continuous

The *width* of $P$ is the minimum distance between two parallel lines $\ell_1, \ell_2$ such that $P$ is between $\ell_1$ and $\ell_2$.

The following lemma that bounds the area difference between a convex polygon and the convex hull of its intersection with a lattice set will be useful to our proof.

**Lemma 6.** *Let $C$ be a convex polygon of diameter $r$. The convex hull $H = \mathrm{conv}(C \cap \Lambda_\varepsilon)$ satisfies*

$$\mathrm{area}(C) \leq \mathrm{area}(H) + 6\sqrt{2}\pi r\varepsilon + 16\varepsilon^2.$$

*Proof.* The lattice $\Lambda_\varepsilon$ induces a grid with vertex set $\Lambda_\varepsilon$ and square cells of side length $\varepsilon$. Let $X^-$ be the set of grid cells that are completely contained in $C$ and $X^\partial$ be the set of cells that are partially contained in $C$. All cells in $X^\partial$ intersect the boundary $\partial C$ of $C$.

Since the perimeter of a convex shape is at most $\pi$ times its diameter [2], the perimeter of $\partial C$ is at most $\pi r$. Since a curve of perimeter $p$ intersects at most $3p/\varepsilon\sqrt{2}+4$ grid cells of side length $\varepsilon$ [23], we have $|X^\partial| \leq 3\pi r/\varepsilon\sqrt{2} + 4$.

All cells in $X^-$ are contained in $H$ and $C$ is covered by $X^- \cup X^\partial$. Therefore, the area of $C \setminus H$ is at most the area in $X^\partial$, which is

$$\varepsilon^2|X^\partial| \leq 4\varepsilon^2 \cdot \left(\frac{3}{\varepsilon\sqrt{2}}\pi r + 4\right) = 6\sqrt{2}\pi r\varepsilon + 16\varepsilon^2,$$

proving the lemma. □

The following lemma bounds the area difference between the optimal solutions of the continuous potato peeling problem with inputs $P$ and $P^-$.

**Lemma 7.** *Let $P$ be a polygon of diameter $r$ and $P^-$ be the erosion of $P$. Let $C$ (resp. $C'$) denote the largest convex polygon inside $P$ (resp. $P^-$). We have the following inequality:*

$$\mathrm{area}(C) \leq \mathrm{area}(C') + 2\sqrt{2}\pi r\varepsilon.$$

*Proof.* The erosion $C^-$ of $C$ is a convex polygon that lies inside $P^-$. Hence the area of $C'$ is at least as large as the area of $C^-$.

As $C$ is a convex polygon of diameter at most $r$, the perimeter of $C$ is at most $\pi r$. As every eroded points from $C$ in order to obtain $C^-$ are inside $C$ and at a maximum distance of $2\sqrt{2}\varepsilon$ of the boundary of $C$, they are all included inside a set of rectangles that lie inside $C$ with the edges of $C$ as sides and width $2\sqrt{2}\varepsilon$. Hence, the area difference between $C$ and its erosion is at most $2\sqrt{2}\varepsilon\pi r$, which proves the lemma. □

The digital solution may have portions that lie outside the input polygon $P$ of the continuous version. However, this portion cannot be too big, as shown in the following lemma.

**Lemma 8.** *Let $P$ be a polygon of diameter $r$ and $P^-$ be the erosion of $P$. Let $S = P^- \cap \Lambda_\varepsilon$, and $K_D$ be the largest digital convex subset of $S$. The following inequality holds:*

$$\mathrm{area}(\mathrm{conv}(K_D) \setminus P) \leq 2r\varepsilon.$$

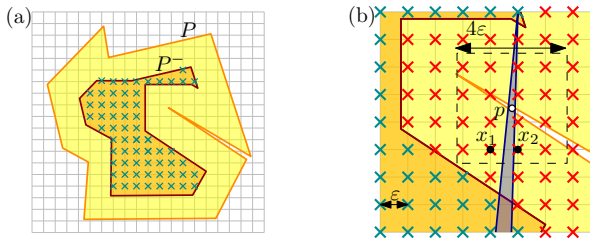


Figure 5: (a) A polygon $P$, its erosion $P^-$, and the set $\Lambda_\varepsilon \cap P^-$. (b) To include the point $p$ that is outside $P$, $\mathrm{conv}(K_D)$ has to go between the lattice points within $L_\infty$ distance $2\varepsilon$ of $p$.

*Proof.* Let $p$ be a point in $\mathrm{conv}(K_D) \setminus P$. As $S$ is included inside $P^-$, all the lattice points within $L_\infty$ distance $\varepsilon$ of $p$ are not in $S$ (see Figure 5). All 16 lattice points at a $L_\infty$ distance less than $2\varepsilon$ of $p$ are not in $S$. Hence, in order to include $p$, $\mathrm{conv}(K_D)$ has to lie between two vertically (or horizontally) consecutive lattice points $x_1$ and $x_2$, which are separated by distance $\varepsilon$. Furthermore $p$ is at a horizontal (or vertical) distance strictly greater than $\varepsilon$ from $x_1$ and $x_2$. The widest angle the incoming and outgoing edges of $C$ can form is hence $2\arctan(1/2)$, effectively forming a turning angle of at least $\pi - 2\arctan(1/2)$. As the sum of turning angles inside a convex polygon is equal to $2\pi$ and can never decrease, and as $\pi - 2\arctan(1/2) > 2\pi/3$ such a turning angle can only happen twice. Also, as in order to include any point $p$ outside of $P$, $\mathrm{conv}(K_D)$ has to go in between $x_1$ and $x_2$, the width of this (possible non-contiguous region) including $p$ is at most $\varepsilon$ and the diameter at most $r$, hence, the area is bounded by $r\varepsilon$. Therefore, there can be no more than two such regions in $\mathrm{conv}(K_D)$ (even though each of them can enter and leave $P$ multiple times), which proves the lemma. □

Using lemma 7, it follows that $A_C - A_C^- \leq 2\sqrt{2}\pi r\varepsilon$. Lemma 6 gives us that $A_C^- - A_D \leq 6\sqrt{2}\pi r\varepsilon + 16\varepsilon^2$. Lemma 8 gives us that $A_D - 2r\varepsilon \leq A_C$. Hence,

$$A_C - 8\sqrt{2}\pi r\varepsilon - 16\varepsilon^2 \leq A_D \leq A_C + 2r\varepsilon,$$

proving the following theorem.

**Theorem 9.** *Let $A_C$ be the area of the solution $K_C$ of Problem 8 with input polygon $P$ of diameter $r$. Let $\varepsilon > 0$ be a parameter and $S = \Lambda_\varepsilon \cap P^-$, where $P^-$ is the erosion of $P$ by $2\varepsilon$ and $\Lambda_\varepsilon$ is the lattice of size $\varepsilon$. The area $A_D$ of the convex hull of the solution $K_D$ of Problem 1 with input $S$ satisfies $|A_C - A_D| = O(r\varepsilon)$.*

The polygon $\mathrm{conv}(K_D)$ in the previous theorem may partially extend outside $P$. Nevertheless, the solution $K_D$ of Problem 1 can be used to obtain a convex polygon $K \subseteq P$ which has an area $A$ satisfying $|A_C - A_D| = O(r\varepsilon)$.

The same proof strategy can be applied to obtain an approximation to the continuous version of the 2-potato-peeling problem using the digital version of the problem.