Doctoral Dissertation

Bundling Problems in Geometric Optimization

Dongwoo Park (박 동 우)

Department of Computer Science and Engineering

Pohang University of Science and Technology

2017

기하 형체들을 효과적으로 배치하는 알고리즘

Bundling Problems in Geometric Optimization

Bundling Problems in Geometric Optimization

by

Dongwoo Park
Department of Computer Science and Engineering
Pohang University of Science and Technology

A dissertation submitted to the faculty of the Pohang University of Science and Technology in partial fulfillment of the requirements for the degree of Doctor of philosophy in the Computer Science and Engineering

Pohang, Korea
12. 20. 2016
Approved by
Hee-Kap Ahn
Academic advisor

Bundling Problems in Geometric Optimization

Dongwoo Park

The undersigned have examined this dissertation and hereby certify that it is worthy of acceptance for a doctoral degree from POSTECH

12. 20. 2016

Committee Chair	Hee-Kap Ahn	(Seal)
Member	Sungwoo Park	(Seal)
Member	Kyungmin Bae	(Seal)
Member	Chan-Su Shin	(Seal)
Member	Antoine Vigneron	(Seal)

DCSE 박 동 우. Dongwoo Park
20110723 Bundling Problems in Geometric Optimization,
기하 형체들을 효과적으로 배치하는 알고리즘
Department of Computer Science and Engineering , 2017,
84p, Advisor : Hee-Kap Ahn. Text in English.

ABSTRACT

Arranging objects is an important and fundamental problem in many fields of industry such as packaging and logistics. Bundling is one frequently used method to arrange objects. Given k geometric objects in $\mathbb{R}^d$, the bundling problem asks to find transformations of the given objects such that the size of the convex hull of the objects is minimized while they remain disjoint in their interiors. In this thesis, we study three variants of bundling problem according to the following criteria: 1) the number, 2) the dimension, 3) the complexity of the given objects. We propose efficient algorithms for these bundling problems.

Contents

I. Introduction

People have been trying to arrange objects efficiently for a long time. From an economical perspective, efficient packing can reduce the amount of space required to store them, or increase the number of objects that can be stored in a given volume. For example, in logistics, one way to increase profits is to minimize the required warehouse space, or the size of packaged products.

This question has received significant attention in a number of disciplines. An early example is the problem of how to efficiently stack cannonballs on the decks of ships; this problem was posed by Johannes Kepler in the 17th century. This problem can be generalized as follows. Given and congruent spheres fill as much of the available volume as possible while they remain disjoint in their interiors. Kepler guessed that the face-centered cubic arrangement (Figure 1.1) has greater density than any other arrangements; this is known as Kepler's conjecture [23]. Kepler's conjecture was not proven until 1998, after the development of computational geometry [17].

Packing problems have been studied in computational geometry. The packing problem is to find a smallest region, called a container, of predetermined shape that packs the input objects under translations or rigid motions. In most cases, the containers are of simple convex shapes such as rectangles and circles or squares, and input objects are polygons or polytopes in two-dimensional or three-dimensional space. Sugihara et al. [33] considered a related problem of minimizing the disk packing of a set of disks with applications to minimizing the sizes of holes through which sets of electric

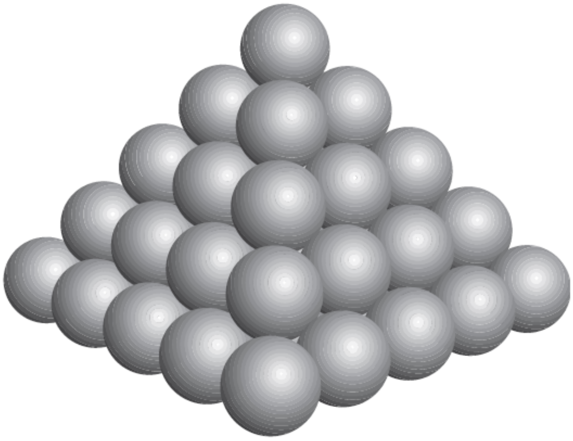


Figure 1.1: The face-centered cubic arrangement

wires are to pass. The authors proposed an $O(n^4)$-time heuristic method that makes use of the Voronoi diagram of circles, where n is the number of disks. Milenkovic studied the packing of a set of polygons into another polygon container with applications in the apparel industry [27]. He gave an $O(n^{k-1} \log n)$ time algorithm for packing k convex n-gons under translations into a minimum area axis-parallel rectangle container. Later, Alt and Hurtado [7] presented a near-linear time algorithm that packs two convex polygons into a minimum area or perimeter rectangle. Recently, Egeblad et al. [13] presented an efficient method to pack polytopes into another polytope container under translation in arbitrary dimensions. For arbitrary numbers of input polygons, the packing problem is shown to be NP-hard even if the polygons are rectangles [11]; this proof was derived by reducing the partition problem [15] to this problem.

The *bundling problem* is closely related to the packing problem. In the bundling problem, the shape of the container is not predetermined. The details and definition of the bundling problem will be introduced in Section 1.1. In this thesis, I introduce the bundling problems for various types of input objects.

1.1 Bundling Problems

The lexical meaning of "bundle" is a collection of things, or a quantity of material, tied or wrapped up together. As the name implies, we do not predetermine the shape of a container that holds the items. Imagine that you are wrapping office supplies in a cloth. When the wrapping task has been completed, the shape of volume enclosed by the cloth is determined by the placement of things inside it. To find the most efficient bundling, we first need to design the structure of the bundling problem that describes this situation.

The Structure of Bundling Problem Given a set of non-overlapping geometric objects, find transformations of input objects such that the size of the convex hull of the transformed objects is minimized.

Based on the above structure of the bundling problem, we define various variants of the bundling problem by adding restrictions to elements, as below.

- The type of input objects
- Transformations
- Measures for the size of convex hull

By using the above structure and restrictions of the bundling problem, we review previous results on the bundling problem.

Given two convex polygons P_0 and P_1, Lee and Woo [25] presented a $O(n)$-time algorithm to find a translation τ_1 of P_1 that minimizes the area of the convex hull of $P_0 \cup \tau_1 P_1$ where n is the total number of vertices. Their algorithm can be easily extended to the case in which the perimeter of the convex hull is minimized.

Some results allow rigid motions of two convex polygons. Tang et al. [34] gave an $O(n^3)$-time algorithm to find a rigid motion that minimizes the area of the convex hull of two convex polygons, Ahn and Cheong [4,5] presented a near-linear time approximation algorithm for finding a rigid motion that minimizes either the perimeter or the area of the convex hull of two convex polygons.

1.1.1 Thesis Outline

The bundling problem has been studied less than the packing problem. In the plane, the only results consider bundling of two convex polygons; no results has been presented for the bundling problem in three-dimensional space, and there is no result for the bundling problem in three dimensional space. In this thesis, I introduce how to extend previous bundling problems according to the number of input polygons and the dimensionality ($\geq$ 3) of the space, for simple polygons.

I study the problem of bundling more than two convex polygons in the plane. In Chapter II, I introduce the problem of bundling three convex polygons as follows.

Problem 1 (bundling three convex polygons) Given three convex polygons P_0, P_1 and P_2 having n vertices in total in the plane, find two translations τ_1, τ_2 of P_1, P_2 such that P_0 and the translated copies $P_1 + \tau_1$ and $P_2 + \tau_2$ do not overlap and the area or the perimeter of the convex hull of $(P_0 \cup (P_1 + \tau_1) \cup (P_2 + \tau_2))$ is minimized.

I study bundling convex polytopes in three or higher dimensions. In Chapter III, I introduce the problem of bundling two convex polytopes as follows

Problem 2 (bundling two convex polytopes) Given two convex d-polytopes P and Q having n vertices in total in $\mathbb{R}^d$ for $d \geq 3$, find a translation τ of Q such that P and the translated copy $Q + \tau$ do not overlap and the surface area or the volume of the convex hull of $(P \cup (Q + \tau))$ is minimized.

I also study bundling objects more complex than convex polygons. In Chapter IV, I introduce the problems for bundling simple objects as follows.

Problem 3-1 (Bundling two simple polygons under translations) Given two simple polygons P and Q having n vertices in total in the plane, find a translation τ of Q such that P and the translated copy $Q + \tau$ do not overlap and the perimeter or the area of the convex hull of $(P \cup (Q + \tau))$ is minimized.

Problem 3-2 (Bundling two simple polygons under rigid motions) Given two simple polygons P and Q having n vertices in total in the plane, find a rigid motion ρ of Q such that P and ρQ do not overlap and the perimeter or the area of the convex hull of $(P \cup \tau Q)$ is minimized.

Problem 3-3 (Bundling two polyhedra under translations) Given two polyhedra P and Q having n vertices in total in three-dimensional space, find a translation τ of Q such that P and the translated copy $Q + \tau$ do not overlap and the perimeter or the area of the convex hull of $(P \cup (Q + \tau))$ is minimized.

1.1.2 Related Works

Minkowski Sums Given two sets $P \subset \mathbb{R}^d$ and $Q \subset \mathbb{R}^d$, for any d, the Minkowski sum of P and Q, denoted by $P \oplus Q$, is defined as the set

$$P \oplus Q := \{p + q \mid p \in P, q \in Q\}.$$

The Minkowski sum is used in a wide range of application, including robot motion planning [8, 12, 32], computer-aided design and manufacturing (CAD/CAM) [14, 28], and bundling problems [2, 10, 29]

Consider for example the bundling problem for two convex polygons under translations. In this application, without loss of generality, we assume that one polygon P is stationary and another polygon Q is allowed to translate and the origin o lies in Q. The Minkowski sum $S = P \oplus -Q$ is the set of all vectors t such that $Q + t$ makes it intersect P, where $Q + t$ is Q translated by t and $-Q$ is the reflection of Q with respect to o. In the bundling problem, we only consider the complement S^{C} of S, called a *configuration space*, hence two polygons are not allowed to overlap each other. Minkowski sums play a important role in the design and the analysis of bundling algorithms.

The complexity of Minkowski sums Consider two polygons P and Q in the plane. If both P and Q are convex polygons, then $P \oplus Q$ is a convex polygon with at most $m + n$ vertices, where m and n are vertices of P and Q respectively. $P \oplus Q$ can be computed in $O(m + n)$ time [12]. If P is convex and Q is non-convex, then $P \oplus Q$ has $O(mn)$ vertices [22], and it can be computed in $O(mn \log(m + n))$ time [12, 26]. If both P and Q are non-convex polygons, then $P \oplus Q$ has $O(m^2n^2)$ vertices, and it is computed in $O(m^2n^2 \log(m + n))$ time by decomposition approaches [12, 21]. These bounds of the complexity of $P \oplus Q$ are tight in the worst case [12]. Given two convex d-polytopes P and Q in the d-dimensional space, the complexity of the Minkowski sum $P \oplus Q$ has the upper bound $O((m+n)^{\lfloor \frac{d+1}{2} \rfloor})$ [19, 20, 31], and it can be computed in $O((m+n)\log(m+n) + (m+n)^{\lfloor \frac{d+1}{2} \rfloor})$ time [9].

Geometric Shape Matching Given two geometric objects, geometric shape matching is a typical method to decide how much they resemble each other [6, 36]. We first select a measure that reflects the similarity between two given objects. Next, we transform given objects by translation, rotation, and scaling. Then we compute similarity based on the measure.

We can measure the quality of similarity by minimizing the convex hull. This measure is usually used in matching between convex objects. Given two convex objects P and Q in d-dimensional space, we find a translation vector t such that the volume of the convex hull of $(P \cup (Q+t))$ is minimized. The shape matching by this measure is equivalent to bundling problems except that the input objects are allowed overlap.

Shape matching for two convex polygons P and Q with n vertices in total in the plane has been studied by Ahn and Cheong [4]. They proposed a $O(n \log n)$-time algorithm for finding an optimal translation t minimizing the volume of the convex hull of $(P \cup (Q+t))$. They also proposed a $O(\epsilon^{-1/2} \log n + \epsilon^{-3/2} \log \varepsilon^{-1/2})$-time algorithm for finding a rigid motion that gives a $(1+\epsilon)$-approximation to the optimal. The shape matching problem based on this measure is extended to higher dimensions. Given two convex polyhedra P and Q in d-dimensional space for a fixed $d \geq 3$, the shape matching is studied by Ahn et al [3]. They gave an expected $O(n^{d+1-\frac{3}{d}}(\log n)^{d+1})$-time algorithm for finding an optimal translation vector t of Q minimizing the volume of convex hull of $(P \cup (Q + t))$.

II. Bundling Three Convex Polygons

The problem of bundling convex polygons in the plane under translation is to find a translation for each of them such that the translated polygons are contained in a smallest possible convex region while their interiors are disjoint. This problem can be modeled as follows: given a set $\mathcal{P} = \{P_0, \ldots, P_{k-1}\}$ of k convex polygons in the plane with n vertices in total, find k translations $\tau_0, \ldots, \tau_{k-1}$ of $P_0, \ldots, P_{k-1}$ such that the translated copies $\tau_i P_i$'s, for $0 \leq i \leq k-1$, do not overlap each other and the area or the perimeter of the convex hull of $\bigcup_{i=0}^{k-1} \tau_i P_i$ is minimized.

In this chapter, we study the bundling problem in the plane when $k = 3$. Without loss of generality, we assume that P_0 is stationary. We show that the translation space of P_1 and P_2 can be decomposed into $O(n^2)$ cells in each of which the combinatorial structure of the convex hull remains the same. Moreover, we show that the description of the objective function for each cell can be fully described using constant space and the function description for a neighboring cell can be updated in constant time by coherence. This was done by a careful analysis on all event configurations at which the combinatorial structure of the convex hull changes. We then present an $O(n^2)$-time algorithm using $O(n^2)$ space that returns an optimal pair of translations.

2.1 Preliminaries

Let $P_0, \ldots, P_{k-1}$ be k convex polygons in $\mathbb{R}^2$ with n vertices in total. For a vector $\tau \in \mathbb{R}^{2k}$, we write $\tau = (\tau_0, \ldots, \tau_{k-1})$, where $\tau_i \in \mathbb{R}^2$. The *translate* of P_i by τ_i, denoted by $\tau_i P_i$, is $\{a + \tau_i \mid a \in P_i\}$. We let $U(\tau) = \bigcup_{i=0}^{k-1} \tau_i P_i$ and let $\mathsf{conv}(\tau) := \mathsf{conv}(U(\tau))$. The perimeter and the area of $\mathsf{conv}(\tau)$ are denoted by $|\mathsf{conv}(\tau)|$ and $\|\mathsf{conv}(\tau)\|$, respectively.

Ahn and Cheong [5] studied the area and perimeter functions and observed the following.

Lemma 1 (Ahn and Cheong [5]). *The function $f : \mathbb{R}^{2k} \to \mathbb{R}$ with $f(\tau) = |\mathsf{conv}(\tau)|$ is convex for any $k \geq 2$. The function $g : \mathbb{R}^{2k} \to \mathbb{R}$ with $g(\tau) = \|\mathsf{conv}(\tau)\|$ is convex and piecewise linear for $k = 2$, but this is not necessarily the case for $k > 2$.*

The bundling problem can be viewed as an optimization problem of minimizing $|\mathsf{conv}(\tau)|$ or $\|\mathsf{conv}(\tau)\|$ over $\tau \in \mathbb{R}^{2k}$ subject to $\tau_i P_i \cap \tau_j P_j = \emptyset$ for all $0 \leq i < j \leq k-1$. One can reduce the search space by a simple observation.

Lemma 2. *For the bundling problem with respect to either area or perimeter, there is an optimal translation vector $\tau^* \in \mathbb{R}^{2k}$ such that the union $U(\tau^*)$ is connected.*

Proof. If $U(\tau^*)$ is connected, we are done. Suppose that $U(\tau^*)$ consists of more than one connected component. Since any two connected components of $U(\tau^*)$ are disjoint in their closures, their convex hulls may overlap but do not cross.

If there is one component C such that $\mathsf{conv}(C) = \mathsf{conv}(\tau^*)$, then all the other components are contained in $\mathsf{conv}(C)$. We can always translate one of them to touch C and become connected to C keeping it contained in $\mathsf{conv}(C)$. This translation process makes no change to $\mathsf{conv}(\tau^*)$ but decreases the number of connected components by one. We repeat this process and finally have a single connected component.

Otherwise, there are at least two connected components whose convex hulls appear on the boundary of $\mathsf{conv}(\tau^*)$. Let L be the set of all such connected components of $U(\tau^*)$. Since the convex hulls of any two components of L do not cross, there is at least one connected component $C \in L$ whose convex hull appears on the boundary of $\mathsf{conv}(\tau^*)$ only once. Let e and e' be the two edges of $\mathsf{conv}(\tau^*)$ between C and the other components $\mathsf{conv}(L \setminus \{C\})$. Then one can translate C in the direction parallel to one of e and e' to have less area and perimeter, which contradicts to the optimality of τ^*. □

We can thus concentrate only on the cases where the k polygons are connected. We shall call $\tau \in \mathbb{R}^{2k}$ a *configuration* if $U(\tau)$ is connected. A configuration τ is *feasible* if and only if the interiors of the translates are disjoint under τ. Thus, our goal is to find an optimal feasible configuration with respect to area or perimeter.

Let $\mathcal{K}$ be the set of configurations for given k polygons $P_0, \ldots, P_{k-1}$. Each configuration $\tau \in \mathcal{K}$ is associated with several properties describing the structure of the convex hull $\mathsf{conv}(\tau)$. If $\tau_i P_i$ and $\tau_j P_j$ are in contact, then a vertex v of P_i lies on an edge e of P_j under τ, or vice versa. We call the pair (v, e) a *contact* induced by τ. Let $C(\tau)$ be the set of contacts induced by a configuration $\tau \in \mathcal{K}$. Note that Lemma 2 implies that $|C(\tau)| \geq k - 1$.

The convex hull $\mathsf{conv}(\tau)$ is bounded by a closed polygonal curve consisting of some edges of the translated polygons $\tau_i P_i$'s and line segments connecting two vertices of the $\tau_i P_i$'s. We call such a segment of the latter type a *bridge*. More specifically, there is a bridge (u, v) connecting a vertex u of a polygon and a vertex v of another polygon if these polygons appear consecutively along the boundary of the convex hull. Note that a bridge (u, v) is a degenerate edge of length 0 if u and v are in contact. Let $H(\tau)$ be the set of pairs of vertices, each of which makes a bridge induced by $\tau \in \mathcal{K}$. Two configurations $\tau, \tau' \in \mathcal{K}$ are said to have the same combinatorial structure if both $C(\tau) = C(\tau')$ and $H(\tau) = H(\tau')$ hold.

In the following sections, we will show that the configuration space can be decomposed into a number of cells in which the configurations have the same combinatorial structure, so that the area or the perimeter function is described and minimized. For convenience of elaboration, we make a general position assumption on input polygons in the sense that no two polygon edges are parallel. If there are two or more parallel edges, we make an ordering of end vertices of them such that the end vertices of an edge of P_i appear ahead of those of an edge of P_j in the ordering if $i < j$. Two end vertices of an edge are ordered counter-clockwise. Our algorithm handles the end vertices of parallel edges following this ordering when they are in a tie.

2.2 The Configuration Space for Three Polygons

In this section, we study the configuration space of three convex polygons P_0, P_1 and P_2 under translation. We first investigate the configuration space by introducing a parametrization of configurations. Then we define events and curves in the configuration space from which the combinatorial structure of the convex hull or the motion of the input polygons changes, and analyze their complexity.

2.2.1 Parametrization of configurations for two or three polygons

As a warm-up exercise, consider the case of $k = 2$ where two convex polygons P_0 and P_1 are given. By Lemma 2, any configuration $\tau \in \mathcal{K}$ requires P_1 to touch P_0. Imagine that P_0 is stationary and P_1 translates around P_0 in the counter-clockwise direction, keeping them touching each other, until P_1 then reaches back to the initial position. The set $\mathcal{K}$ of configurations thus forms a space homeomorphic to a unit circle. This motion of P_1 around P_0 is piecewise affine, and the total distance that P_1 travels is exactly $|P_0| + |P_1|$. Therefore, letting $L := |P_0| + |P_1|$, the interval $[0, L)$ fully describes the configuration space $\mathcal{K}$: For any $\lambda \in [0, L)$, let $\tau(\lambda)$ be the configuration whose corresponding translated copy of P_1 is a snapshot at a moment when P_1 travels a distance of exactly λ around P_0 from its initial position.

We now turn to the case of $k = 3$, where three convex polygons P_0, P_1, and P_2 are given as input. Lemma 2 implies that in any configuration $\tau \in \mathcal{K}$, at least one of the three polygons touches the other two, simultaneously. Without loss of generality, we assume that both P_1 and P_2 translate around P_0 in the counter-clockwise direction keeping touching P_0, while P_0 remains stationary. Let $\mathcal{K}_0 \subset \mathcal{K}$ be the space of configurations in which $\tau_0 = (0, 0)$ and P_0 touches both of P_1 and P_2. As discussed above for $k = 2$, the distance that each of P_1 and P_2 travels around P_0 is exactly L_1 and L_2, respectively, where $L_1 := |P_0| + |P_1|$ and $L_2 := |P_0| + |P_2|$. See Figure 2.1. Then, any pair $(\lambda_1, \lambda_2) \in [0, L_1) \times [0, L_2)$ corresponds to a configuration $\tau(\lambda_1, \lambda_2)$ when P_1 and P_2 travel around P_0 by distance exactly λ_1 and λ_2, respectively, from their initial positions.

Notice that the definition of configurations do not prevent P_1 and P_2 from overlapping each other; rather, the translates of P_1 and P_2 around P_0 are independent, and are determined independently by two different parameters λ_1 and λ_2, respectively. We denote by $P_1(\lambda_1)$ and $P_2(\lambda_2)$ the translated copy of P_1 and P_2, respectively, corresponding to the parameters λ_1 and λ_2, respectively. By abuse of notation, we shall call a pair (λ_1, λ_2) a configuration in $\mathcal{K}_0$ and regard $\mathcal{K}_0$ to be $[0, L_1) \times [0, L_2)$.

2.2.2 Events and event curves

Recall that any configuration $\tau \in \mathcal{K}_0$ is associated with the set $C(\tau)$ of contacts and the set $H(\tau)$ of bridges of the corresponding convex hull $\mathsf{conv}(\tau)$. These two com-

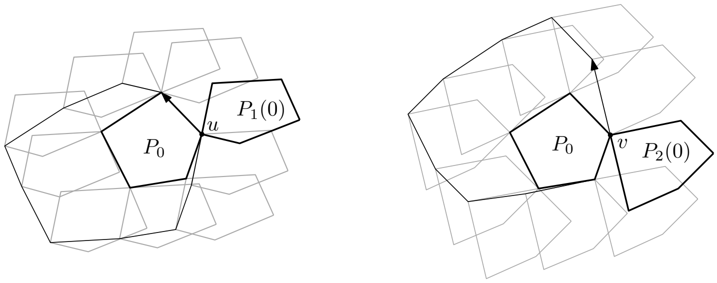
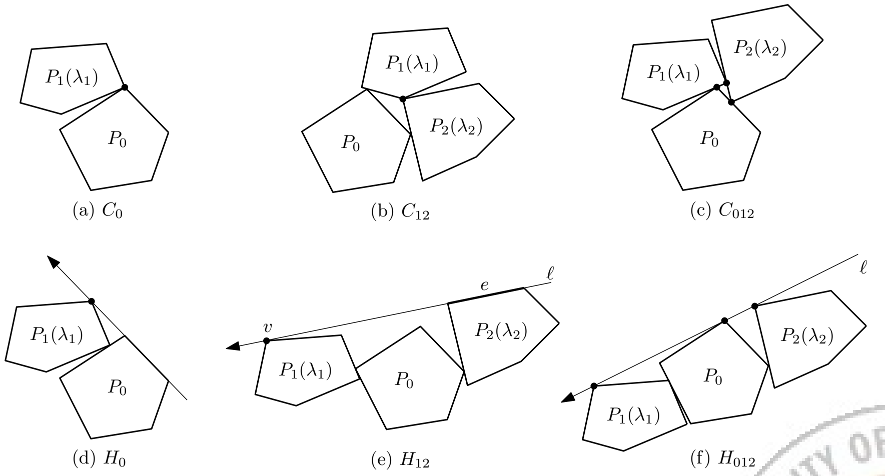


Figure 2.1: Sliding P_1 and P_2 around P_0: we parameterize the configuration space $\mathcal{K}_0$ by a pair of parameters (λ_1, λ_2) for $\lambda_1 \in [0, L_1)$ and $\lambda_2 \in [0, L_2)$.

binatorial associates determine the structure of the convex container and the motion of the polygons, thus being helpful in describing the objective function on the configuration space as will be shown in next sections. One natural approach would decompose the configuration space, $\mathcal{K}_0$ into cells in each of which $C(\tau)$ and $H(\tau)$ remain the same for all configurations τ in the cell.

We call a configuration $\tau = (\lambda_1, \lambda_2) \in \mathcal{K}_0$ an *event* if it is one of the following cases:



(a) C_0 (b) C_{12} (c) C_{012}

(d) H_0 (e) H_{12} (f) H_{012}

Figure 2.2: Corresponding translates of the three polygons at events of all six types.

C_0 **event** A vertex of $P_1(\lambda_1)$ or $P_2(\lambda_2)$ reaches a vertex of P_0; that is, a vertex-vertex contact occurs between P_0 and one of the others. See Figure 2.2(a).

C_{12} **event** $P_1(\lambda_1)$ and $P_2(\lambda_2)$ touch each other at a vertex of $P_1(\lambda_1)$ and a vertex of $P_2(\lambda_2)$; that is, a vertex-vertex contact occurs between P_1 and P_2. See Figure 2.2(b).

C_{012} **event** $P_1(\lambda_1)$ and $P_2(\lambda_2)$ touch each other; that is, the three polygons are pairwise touching and it holds that $|C(\tau)| = 3$. See Figure 2.2(c).

H_0 **event** $P_i(\lambda_i)$, for $i = 1$ or 2, is tangent to the supporting line of an edge of P_0 from the side containing P_0, or vice versa. See Figure 2.2(d).

H_{12} **event** $P_1(\lambda_1)$ is tangent to the supporting line of an edge of $P_2(\lambda_2)$ from the side containing $P_2(\lambda_2)$, or vice versa. See Figure 2.2(e).

H_{012} **event** The three polygons P_0, $P_1(\lambda_1)$, and $P_2(\lambda_2)$ have a common tangent line ℓ and the three lie in the same side of ℓ. See Figure 2.2(f).

Remark that $\mathcal{K}_0$ includes configurations whose corresponding translates of P_1 and P_2 may overlap each other; the set of C_{012} events subdivides $\mathcal{K}_0$ into two subspaces such that every configuration in one subspace causes a nonempty overlap of P_1 and and P_2, and every configuration in the other subspace does not cause such overlap. Note, however, that all the changes of $C(\tau)$ and $H(\tau)$ can be captured by a series of the events, when τ continuously moves inside $\mathcal{K}_0$ while it avoids overlap between P_1 and P_2. In particular, although some portions of H_{12} events indeed imply an overlap between P_1 and P_2, it suffices to track all the changes of $H(\tau)$ by H_0, H_{12}, and H_{012} events if τ continuously moves without any overlap. On the other hand, events of type C_{12} and C_{012} by definition help τ avoid such overlap. Note that a C_{12} event is a C_{012} event by definition. In our further discussions, any C_{12} event will be considered as a special case of C_{012} events, and will play a role as a "breakpoint" when we trace other types of events.

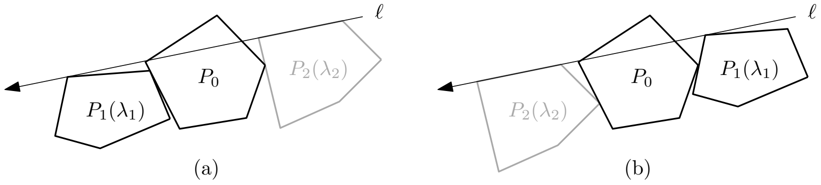


Figure 2.3: Illustration of decomposing the set of H_{12} events. There are two different types of translations for P_1 around P_0 for a directed line ℓ tangent to P_1; (a) P_1 is ahead of P_0, or (b) P_0 is ahead of P_1 along ℓ. The analogue also holds for P_2.

We claim that the set of all events forms a set of curves in the configuration space $\mathcal{K}_0 = [0, L_1) \times [0, L_2)$, and thus decomposes the space into cells. To see this more precisely, we partition the set of all events into subsets according to each type as follows:

Partition of C_0 events Any C_0 event corresponds to a vertex-vertex contact, involving a pair (v, v') of vertices, exactly one of which belongs to P_0. We denote by $\gamma_{vv'}^{C_0} = \gamma_{v'v}^{C_0}$ the set of all C_0 events with the involved pair (v, v'). Hence, the set of all C_0 events are partitioned into subsets $\gamma_{vv'}^{C_0}$ where v is a vertex of P_0 and v' is a vertex of P_1 or P_2.

Partition of C_{012} events For any C_{012} event $\tau = (\lambda_1, \lambda_2)$, $P_1(\lambda_1)$ and $P_2(\lambda_2)$ touch each other. There are two cases: either $P_1(\lambda_1)$ is ahead of $P_2(\lambda_2)$ as depicted in Figure 2.2(c), or $P_2(\lambda_2)$ is ahead of $P_1(\lambda_1)$. In this case, we say that$P_1(\lambda_1)$ is *ahead* of $P_2(\lambda_2)$ if $P_2(\lambda_2 + \epsilon)$ overlaps $P_1(\lambda_1)$ for arbitrarily small $\epsilon > 0$. We denote the set of C_{012} events corresponding to the former by $\gamma_1^{C_{012}}$ and the set of C_{012} events corresponding to the latter by $\gamma_2^{C_{012}}$. Hence, we partition the set of all C_{012} events into $\gamma_1^{C_{012}}$ and $\gamma_2^{C_{012}}$. Note that every C_{12} event coincides with a C_{012} event by definition, and thus all the C_{12} events are included in $\gamma_1^{C_{012}} \cup \gamma_2^{C_{012}}$.

Partition of H_0 events Any H_0 event τ corresponds to a collinearity of an edge e and a vertex v, one of which belongs to P_0. Let ℓ be the supporting line of e and assume that ℓ is directed so that the two polygons that each of v and e belongs to lie on its left side. There are two cases: the vertex v is ahead of e or behind e, along the directed line ℓ. We denote the set of all H_0 events corresponding to the former by $\gamma_{ve}^{H_0}$ and the set of all H_0 events corresponding to the latter by $\gamma_{ev}^{H_0}$. Thus, the set of all H_0 events are partitioned into subsets $\gamma_{ve}^{H_0}$ and $\gamma_{ev}^{H_0}$ for any vertex v and any edge e of the input polygons where one of v and e belongs to P_0.

Partition of H_{12} events Any H_{12} event $\tau = (\lambda_1, \lambda_2)$ corresponds to a collinearity of an edge e and a vertex v, each of which belongs mutually to P_1 or P_2. Let ℓ be the supporting line of e and assume that ℓ is directed so that both $P_1(\lambda_1)$ and $P_2(\lambda_2)$ lie on its left side. Note that ℓ translates as e (and the polygon containing e) translates. Observe that there are two different types of translations for P_1 around P_0 such that ℓ keeps being tangent to the translate of P_1; P_1 is ahead of P_0 along the directed line ℓ (Figure 2.3(a)) or vice versa (Figure 2.3(b)). The analogue also holds for P_2. Thus, $\tau = (\lambda_1, \lambda_2)$ falls into one of the four cases. We denote by $\gamma_{ve,11}^{H_{12}}$ the set of H_{12} events (λ_1, λ_2) defined by (v, e) such that $P_1(\lambda_1)$ is ahead of P_0 and $P_2(\lambda_2)$ is also ahead of P_0. Similarly, define the other three $\gamma_{ve,12}^{H_{12}}$, $\gamma_{ve,21}^{H_{12}}$, and $\gamma_{ve,22}^{H_{12}}$. Figure 2.2(e) shows a H_{12} event in the set $\gamma_{ve,12}^{H_{12}}$. Thus, the set of all H_{12} events are partitioned into subsets $\gamma_{ve,11}^{H_{12}}$, $\gamma_{ve,12}^{H_{12}}$, $\gamma_{ve,21}^{H_{12}}$, and $\gamma_{ve,22}^{H_{12}}$ for any vertex v of P_1 and any edge e of P_2, or vice versa.

Partition of H_{012} events Any H_{012} event $\tau = (\lambda_1, \lambda_2)$ is associated with a line ℓ commonly tangent to the three polygons P_0, $P_1(\lambda_1)$, and $P_2(\lambda_2)$. We assume that ℓ is always directed so that P_0, together with the other two polygons, lies on its left side. We have again four cases as we have for H_{12} events; either $P_1(\lambda_1)$ (or $P_2(\lambda_2)$) is ahead of P_0 along ℓ or is behind P_0. We divide the set of H_{012} events into four subsets as we did for H_{12} events and denote them by $\gamma_{11}^{H_{012}}$, $\gamma_{12}^{H_{012}}$, $\gamma_{21}^{H_{012}}$, and $\gamma_{22}^{H_{012}}$, respectively. Figure 2.2(f) shows a H_{012} event in the set $\gamma_{12}^{\mathrm{H}_{012}}$.

So far we discussed how to partition the events of each type into subsets. All these subsets together form a partition of all the events. We let Γ be the partition of all events into nonempty subsets defined as above. We show in the following that every $\gamma \in \Gamma$ forms a monotone curve in $\mathcal{K}_0$ with several nice behaviors.

Lemma 3. *Each set $\gamma \in \Gamma$ forms a curve in $\mathcal{K}_0$ that is monotone in both axes, and consists of at most three connected components. In addition, γ has following properties according to its type.*

- *If γ is a subset of C_0 or H_0 events, then γ is a line parallel to the λ_1-axis or the λ_2-axis.*
- *If γ is a subset of C_{012} events, then γ is non-decreasing and piecewise linear, each of whose breakpoints coincides with a C_0 or C_{12} event.*
- *If γ is a subset of H_{12} events, then γ is piecewise linear, each of whose breakpoints coincides with a C_0 event.*
- *If γ is a subset of H_{012} events, then γ is non-decreasing and piecewise hyperbolic, each of whose breakpoints coincides with a C_0 or H_0 event.*

Proof. We consider each type separately.

- *(C_0 events) When γ is a subset of C_0 events.*

  Without loss of generality, assume that $\gamma = \gamma_{vv'}^{C_0}$ where v belongs to P_0 and v' belongs to P_1. By definition, such a C_0 event implies a contact between v and

v'. This fixes the position of P_1, and thus the value of λ_1. On the other hand, P_2 is free to slide around P_0. We thus have $\gamma = \{(c, \lambda_2) \mid \lambda_2 \in [0, L_2)\}$ for some constant $c \in [0, L_1)$. Therefore, γ is parallel to the λ_2-axis. Analogously, if v' belongs to P_2, then γ is parallel to the λ_1-axis.

- *(H_0 events) When γ is a subset of H_0 events.*

  This case is similar to C_0 curves; all the events in γ has a fixed λ_1- or λ_2-coordinate if P_1 or P_2 is involved in the event, respectively. Thus, γ is parallel to the λ_1-axis or to the λ_2-axis.

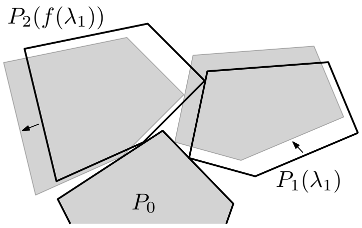


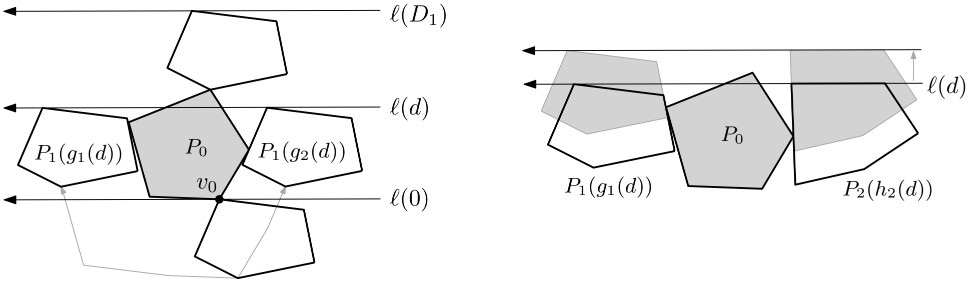
Figure 2.4: Illustration to the proof of Lemma 3 for C_{012} events.

- *(C_{012} events) When γ is a subset of C_{012} events.*

  Thus, $\gamma = \gamma_1^{C_{012}}$ or $\gamma_2^{C_{012}}$. Assume that $\gamma = \gamma_1^{C_{012}}$. The other case can be shown analogously. Recall that γ consists of all configurations $(\lambda_1, \lambda_2) \in \mathcal{K}_0$ such that the three polygons P_0, $P_1(\lambda_1)$, and $P_2(\lambda_2)$ are pairwise touching, and $P_2(\lambda_2)$ is ahead of $P_1(\lambda_1)$. Let us fix λ_1 to be a value in $[0, L_1)$. Then, we have two cases: (1) there exists a unique $\lambda_2 \in [0, L_2)$ such that the three polygons P_0, $P_1(\lambda_1)$, and $P_2(\lambda_2)$ are pairwise touching and $P_2(\lambda_2)$ is ahead of $P_1(\lambda_1)$, or (2) there are more than one such values of λ_2.

  In the former case, we now walk along γ by increasing λ_1 from 0 to L_1, and observe the behavior of the three polygons corresponding to the current point

(λ_1, λ_2) on γ. Since $P_2(\lambda_2)$ should touch $P_1(\lambda_1)$ and be ahead of $P_1(\lambda_1)$, it also has to slide around P_0 in the counter-clockwise direction, as λ_1 increases. The latter case in fact can happen when both P_0 and $P_1(\lambda_1)$ touch a common edge of $P_2(\lambda_2)$, so P_2 is able to slide while keeping in contact to both. This implies that γ is monotone because we can switch the role of λ_1 and λ_2 by symmetry, while γ may contain a vertical line segment in $\mathcal{K}_0$. These two cases imply that γ is continuous as λ_1 increases, except at the limits of $\mathcal{K}_0$ (that is, when $\lambda_1 = 0$ or L_1, or $\lambda_2 = 0$ or L_2), and γ has a non-negative slope at every point, so non-decreasing. In addition, one can easily see that the motion of $P_1(\lambda_1)$ and $P_2(\lambda_2)$ is linear while the contact $C(\tau)$ remains the same.



(a) (b)

Figure 2.5: Illustration to the proof of Lemma 3 for H_{12} curves.

- *(H_{12} events) When γ is a subset of H_{12} events.*

  In this case, we have $\gamma = \gamma^{H_{12}}_{ve,ij}$ for some $1 \leq i, j \leq 2$. We consider all the four subsets for fixed v and e, simultaneously. Without loss of generality, assume that e is horizontal and the polygon that e belongs to lies below e in $\mathbb{R}^2$. We also assume that the bottommost vertex v_0 of P_0 lies on the line $y = 0$. Define $\ell(d)$ for $d \geq 0$ to be the line $\{y = d\}$, directed to the left.

Let t_0 be the value such that the topmost vertex of $P_1(t_0)$ touches v_0 of P_0. Let t_1 be the value such that $P_1(t_1)$ is the highest translate of P_1 that touches P_0, and let D_1 be such that $\ell(D_1)$ passes through the topmost vertex of $P_1(t_1)$. Observe then that for any $0 < d < D_1$, there are two translates of P_1 each of which touches P_0 and $\ell(d)$, and P_1 lies to the left of $\ell(d)$. See Figure 2.5(a). We denote these two translates by $P_1(g_1(d))$ and $P_1(g_2(d))$; we have $g_1(0) = g_2(0) = t_0$ and $g_1(D_1) = g_2(D_1) = t_1$, and $P_1(g_1(d))$ is to the left of $P_1(g_2(d))$ for any $0 < d < D_1$ in $\mathbb{R}^2$. Analogously, we define D_2 for P_2, and also define $h_1(d)$ and $h_2(d)$ for P_2 that are analogous to $g_1(d)$ and $g_2(d)$, respectively. Then, observe that $(g_i(d), h_j(d)) \in \gamma^{H_{12}}_{ve,ij}$ for any $0 \leq d \leq \min\{D_1, D_2\}$.

By adjusting the coordinate system of $\mathcal{K}_0$, we assume that $g_1(0) = g_2(0) = 0$ and $h_1(0) = h_2(0) = 0$. Then, both g_1 and h_1 are decreasing from L_1 and L_2, respectively, while both g_2 and h_2 are increasing from 0 because $P_1(g_1(d))$ and $P_2(h_1(d))$ slide around P_0 in the clockwise direction while $P_1(g_2(d))$ and $P_2(h_2(d))$ slide in the counter-clockwise direction as d increases. This also implies the continuity of g_i and h_j for any $1 \leq i, j \leq 2$. Putting it all together, we conclude that $\gamma^{H_{12}}_{ve,ij}$ is the graph of a partial function of λ_1 that is either increasing or decreasing; it is increasing if $i = j$, or decreasing otherwise. For example, Figure 2.5(b) illustrates the case of $(i, j) = (1, 2)$, where one observes that $\gamma^{H_{12}}_{ve,12}$ is the graph of a decreasing function. Hence, in the general case where $g_1(0) = g_2(0) \neq 0$ and $h_1(0) = h_2(0) \neq 0$, each $\gamma^{H_{12}}_{ve,ij}$ consists of at most three connected components whose endpoints lie at the limit of $\mathcal{K}_0$.

Finally, we show that γ is piecewise linear with breakpoints at C_0 events. For the purpose, consider the grid G on $\mathcal{K}_0$ generated by all the curves of C_0 events and pick any grid cell σ intersected by γ. By definition, any configuration $\tau \in \gamma \cap \sigma$ has a fixed contact set $C(\tau)$. Since γ is a curve as observed above, $\gamma \cap \sigma$ is also a curve and can be expressed by the set of $(g_i(d), h_j(d))$ over some sub-interval I of $[0, \min\{D_1, D_2\}]$. Since $C(g_i(d), h_j(d))$ is constant over $d \in I$, $g_i(d)$ and $h_j(d)$ are linear functions on d. (See Figure 2.5(b).) This implies that $\gamma \cap \sigma$ is a line segment. Therefore, γ is a monotone and piecewise curve whose breakpoints coincide with C_0 events.

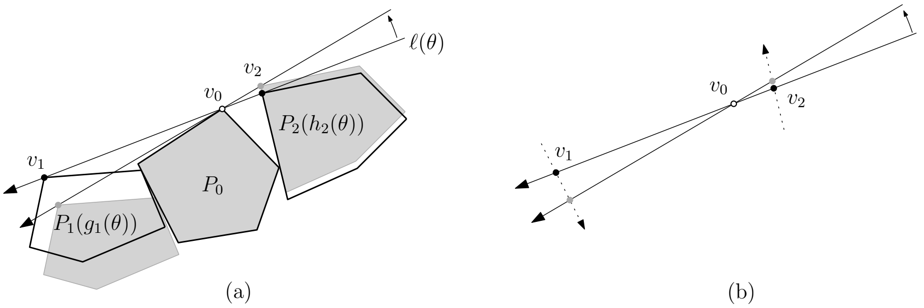


Figure 2.6: Illustration to the proof of Lemma 3 for H_{012} curves.

- *(H_{012} events) When γ is a subset of H_{012} events.*

  That is, $\gamma = \gamma_{ij}^{H_{012}}$ for some $1 \leq i, j \leq 2$. For any H_{012} event (λ_1, λ_2), there is a directed line that is commonly tangent to the three polygons, as discussed above. For any $\theta \in [0, 2\pi)$, consider the directed line $\ell(\theta)$ oriented in θ and tangent to P_0 such that P_0 lies on the left side of $\ell(\theta)$. Then, there are exactly two different values of $\lambda_1 \in [0, L1)$ such that $P_1(\lambda_1)$ is tangent to $\ell(\theta)$ from the left of $\ell(\theta)$; either $P_1(\lambda_1)$ is ahead of P_0 along $\ell(\theta)$ or not. We let $g_1(\theta)$ and $g_2(\theta)$ be the values of λ_1 for the former and the latter cases, respectively. Analogously, we define $h_1(\theta)$ and $h_2(\theta)$ for $P_2(\lambda_2)$. We then observe that $(g_i(\theta), h_j(\theta)) \in \mathcal{K}_0$ is an H_{012} event for any $\theta \in [0, 2\pi)$ and any $1 \leq i, j \leq 2$. Moreover, we have

$(g_i(\theta), h_j(\theta)) \in \gamma_{ij}^{H_{012}}$.

Observe that g_i and h_j increase continuously as θ increases from 0, unless $g_i(\theta) = 0$ or $g_i(\theta) = L_1$, or $h_j(\theta) = 0$ or $h_j(\theta) = L_2$. The inverse functions of g_i and h_j are also well defined: there is a unique $\theta \in [0, 2\pi)$ such that $g_i(\theta) = \lambda_1$ for any fixed $\lambda_1 \in [0, L_1)$. This implies that each set $\gamma_{ij}^{H_{012}}$ is monotone in both axes, and is the graph of an increasing function f_{ij} from $[0, L_1)$ to $[0, L_2)$. Figure 2.6(a) illustrates the case of $(i, j) = (1, 2)$. In order to see that f_{ij} is piecewise hyperbolic, consider the grid G on $\mathcal{K}_0$ generated by all C_0 and H_0 curves. Recall that any curve of type C_0 or H_0 is an axis-parallel line in $\mathcal{K}_0$. In any cell σ of G that is intersected by γ, we have a constant contact set C and bridges. Since γ is monotone, $\gamma \cap \sigma$ is connected and thus it gives us an open interval $I = (\theta_1, \theta_2)$ such that $\gamma \cap \sigma = \{(g_i(\theta), h_j(\theta)) \mid \theta_1 < \theta < \theta_2\}$. For any $\theta \in I$, observe that $\ell(\theta)$ always passes through three fixed vertices v_0, v_1 and v_2 such that v_a belongs to P_a for $a = 0, 1, 2$. (See Figure 2.6(a) for an illustration.) Also, $\ell(\theta)$ rotates at v_0 as $\theta \in I$ increases. Each of v_1 and v_2 translates in a fixed direction as θ increases in the interval I. Observe that the direction of v_1 and v_2 is determined by the contacts C. This constrains the motion of P_1 and P_2 to be all linear along γ inside σ. We analyse f_{ij} using basic trigonometry as follows. Let $\lambda_2 = f_{ij}(\lambda_1)$. In Figure 2.6, let d_1 and d_2 be the Euclidean distance from v_0 to the lines supporting the motion of v_1 and v_2 locally. Then, the pair (λ_1, λ_2) can be parameterized as $(d_1 \tan(\theta + \alpha_1) + l_1, d_2 \tan(\theta + \alpha_2) + l_2)$, where c_1, c_2, l_1, l_2 are constants and θ represents the orientation of the line $\ell(\theta)$. Eliminating the parameter θ yields

$$\lambda_2 = d_2 \tan\left(\tan^{-1}\left(\frac{\lambda_1 - l_1}{d_1}\right) + \alpha_2 - \alpha_1\right) + l_2.$$

This equation is simplified as follows by applying the "addition formula for tangent": $\tan(u+v) = (\tan u + \tan v)/(1 - \tan u \tan v)$.

$$\left(\lambda_1 - l_1 - \frac{d_1}{\tan(\alpha_2 - \alpha_1)}\right)\left(\lambda_2 - l_2 + \frac{d_2}{\tan(\alpha_2 - \alpha_1)}\right) = d_1 d_2 \cdot \frac{\tan^2(\alpha_2 - \alpha_1) + 1}{\tan^2(\alpha_2 - \alpha_1)}.$$

This represents an equation of a hyperbolic curve of the form $\lambda_2 = f_{ij}(\lambda_1) = 1/(c_1 + c_2\lambda_1) + c_3$ for some constants c_1, c_2, c_3, and f_{ij} is increasing unless γ reaches the boundary of $\mathcal{K}_0$ in σ. This shows the final case of the lemma.

This completes the proof of the lemma. $\square$

We call each $\gamma \in \Gamma$ an *event curve* of type C_0, C_{012}, H_0, H_{12}, or H_{012} according to its type. As shown in the proof of Lemma 3, each event curve $\gamma \in \Gamma$ consists of one, two, or three connected components unless it is axis-parallel. Note that the endpoints of γ occur only when $\lambda_1 \in \{0, L_1\}$ or $\lambda_2 \in \{0, L_2\}$, except the endpoints of H_{12} event curves that lie on a C_{012} event curve. This discontinuity is because the configuration space $\mathcal{K}_0$ is indeed periodic; if we extend $P_1(\lambda_1)$ and $P_2(\lambda_2)$ for $\lambda_1 > L_1$ and $\lambda_2 > L_2$, then we have $P_1(\lambda_1 + L_1) = P_1(\lambda_1)$ and $P_2(\lambda_2 + L_2) = P_2(\lambda_2)$, then γ becomes connected.

2.2.3 Complexity of event curves

We now discuss the complexity of event curves and of their arrangement $\mathcal{A}(\Gamma)$.

Lemma 4. *The family* Γ *consists of* $O(n)$ *event curves and the number of* C_{12} *events is bounded by* $O(n)$*. Also, each event curve in* Γ *consists of either* $O(n)$ *line segments or* $O(n)$ *hyperbolic segments.*

Proof. We take event curves of each type into account. Note that there are exactly two C_{012} event curves and four H_{012} event curves as defined above. Let n_0, n_1, and n_2 be the number of vertices of P_0, P_1, and P_2, respectively.

A C_0 event curve $\gamma^{C_0}_{vv'}$ is associated with a pair of vertices (v, v'), where v is a vertex of P_0 and v' is a vertex of P_1 or P_2. The number of such pairs (v, v') with $\gamma^{C_0}_{vv'} \in \Gamma$ is exactly $n_0 + n_1$ if v' is a vertex of P_1 since each vertex can appear and disapper in the contact between P_0 and P_1 exactly once while P_1 slides around P_0 keeping in contact with P_0. The same analysis can also be seen in the proof of Lemma 4 of Lee and Woo [25]. Analogously, the number of such pairs (v, v') that v' belongs to P_2 is exactly $n_0 + n_2$. Thus, there are exactly $2n_0 + n_1 + n_2$ C_0 event curves in Γ.

For any edge e of P_0, there are exactly two translates of P_1 such that P_1 is tangent to the supporting line of e, and thus a vertex of P_1 lies on the line. Equivalently, for any edge e' of P_1, there are exactly two translates of P_1 such that P_0 is tangent to the supporting line of e'. This implies that the number of H_0 event curves defined by P_0 and P_1 is exactly $2n_0 + 2n_1$. Analogously, the number of H_0 event curves defined by P_0 and P_2 is exactly $2n_0 + 2n_2$. On the other hand, the number of H_{12} event curves is exactly $4n_1 + 4n_2$ since there are exactly four event curves for each edge of P_1 and P_2, as discussed in the proof of Lemma 3. Therefore, the number of H_0 and H_{12} event curves is exactly $4n_0 + 6n_1 + 6n_2$.

For the number of C_{12} events, recall that each C_{12} event event lies on a C_{012} curve. Walking along a C_{012} event curve γ, we have a continuous motion of P_1 and P_2 where the three polygons are pairwise touching, and P_1 and P_2 slide around P_0 in the counterclockwise direction. During this walk, observe that P_1 indeed slides around P_2. Hence, the number of C_{12} events along a C_{012} event curve is exactly $n_1 + n_2$. Since we have two C_{012} event curves, the number of C_{12} events in total is $2n_1 + 2n_2$.

To show the second statement, we bound the number of breakpoints on each event curve of type H_{12}, H_{012}, and C_{012}. Recall that any C_0 or H_0 event curve is axis-parallel by Lemma 3. Consider an H_{12} event curve $\gamma \in \Gamma$. Let m be the number of C_0 events on γ; that is, the number of C_0 event curves intersected by γ. The proof of Lemma 3 implies that γ consists of m line segments. Again by Lemma 3, γ is monotone and therefore $m = O(n)$ since there are $O(n)$ C_0 event curves that are axis-parallel. Similarly, any C_{012} event curve consists of $O(n)$ line segments and any H_{012} event curve consists of $O(n)$ hyperbolic segments. □

We now consider the arrangement $\mathcal{A}(\Gamma)$ of the event curves in Γ.

Lemma 5. *The complexity of the arrangement $\mathcal{A}(\Gamma)$ is $O(n^3)$, and each of its edges is either a line segment or a hyperbolic arc. More specifically, the number of crossings between any two event curves in Γ is $O(n)$.*

Proof. We first show that the number of crossings between any two event curves in Γ is bounded by $O(n)$. By Lemma 3, any C_0 or H_0 event curve is axis-parallel and any $\gamma \in \Gamma$ is monotone in both axes. Thus, any C_0 or H_0 event curve intersects any other event curve at most once.

Consider two event curves $\gamma_1, \gamma_2 \in \Gamma$ of type C_{012}, H_{12}, or H_{012}. By Lemmas 3 and 4, each γ_i is monotone and has $O(n)$ breakpoints. And γ_i is either linear or hyperbolic on any interval of $[0, L_1)$ between two consecutive breakpoints of γ_i. This implies that there are at most two values of λ_1 in each of the intervals such that $f_1(\lambda_1) = f_2(\lambda_1)$.

Therefore, there are at most $O(n)$ crossings between γ_1 and γ_2. Since there are only $O(n)$ event curves of type C_{012}, H_{12}, or H_{012} by Lemma 4, we have at most $O(n^3)$ crossings. Hence, the combinatorial complexity of the arrangement $\mathcal{A}(\Gamma)$ is bounded by $O(n^3)$. □

Recall that the two C_{012} event curves divide $\mathcal{K}_0$ into two regions, one consisting of all feasible configurations and the other of all infeasible configurations in $\mathcal{K}_0$. We denote by $\mathcal{F} \subset \mathcal{K}_0$ the former region. Since we want to find an optimal feasible configuration, we are interested in the feasible region $\mathcal{F}$ and how it is decomposed.

In the following, we show that the complexity of $\mathcal{A}(\Gamma)$ restricted to $\mathcal{F}$ is $O(n^2)$. By Lemma 3, any C_0 or H_0 event curve crosses any other event curve at most once. Since there are only two event curves of type C_{012} and four event curves of type H_{012}, the number of combinations (γ_1, γ_2) of any two event curves of type C_{012}, H_{12}, or H_{012} but not both of type H_{12} is $O(n)$, which implies that the total number of crossings from such combinations is at most $O(n^2)$ by the proof of Lemma 5. The following lemma shows that the total number crossings from the combinations of any two event curves both of type H_{12} is $O(n^2)$.

Lemma 6. *Any two H_{12} event curves cross at most twice in $\mathcal{F}$. Therefore, the arrangement $\mathcal{A}(\Gamma)$ consists of $O(n^2)$ vertices and edges in $\mathcal{F}$.*

Proof. Consider two distinct H_{12} event curves $\gamma_1, \gamma_2 \in \Gamma$. If one of the two is non-increasing and the other non-decreasing, then they cross at most once and we are done.

Without loss of generality, assume that both γ_1 and γ_2 are non-decreasing such that γ_1 is defined by a collinearity of a vertex v of P_1 and an edge e of P_2 and γ_2 is by a vertex v' and an edge e' whichever of P_1 and P_2 they belong to. Suppose that γ_1 crosses γ_2 at (λ_1, λ_2) in $\mathcal{F}$. Let ℓ be the line supporting e of $P_2(\lambda_2)$ and ℓ' be the line supporting

e' at this configuration. Then, we have $v \in \ell$ and $v' \in \ell'$. Also, let d be the distance between v and the closer endpoint of e along ℓ. We then observe that for any crossing in $\gamma_1 \cap \gamma_2$ the distance between v and the closer endpoint of e must be exactly d. This can be seen by simple geometry: Imagine that P_1 moves along ℓ towards e of P_2 from infinity, and see the distance between the line supporting e' and the vertex v'. There is at most one instance where e' and v' are aligned, and the distance between v and the closer endpoint of e is exactly d at the moment.

Now, consider the location of P_1 and P_2 as above. Since $(\lambda_1, \lambda_2) \in \mathcal{F}$, they do not overlap each other. We then have at most two possible positions of P_0 that touches both P_1 and P_2. This means that there are at most two such coordinates (λ_1, λ_2), and thus two H_{12} event curves can cross at most twice in $\mathcal{F}$. Since there are $O(n)$ H_{12} event curves in Γ, this suffices to show that the number of crossings in $\mathcal{F}$ among all H_{12} event curves is $O(n^2)$. $\square$

Figure 2.7 shows the arrangement $\mathcal{A}(\Gamma)$ of the event curves for the three input polygons depicted in Figure 2.1. For our purpose, it suffices to decompose the feasible region $\mathcal{F}$. The next lemma shows that the contacts $C(\tau)$ and the bridges $H(\tau)$ stay constant in each cell.

Lemma 7. *The arrangement $\mathcal{A}(\Gamma)$ of the event curves decomposes the feasible region $\mathcal{F} \subset \mathcal{K}_0$ into cells σ such that both $C(\tau)$ and $H(\tau)$ remain constant over all $\tau \in \sigma$.*

Proof. Fix any cell σ of $\mathcal{A}(\Gamma)$. Suppose to the contrary that $(C(\tau_1), H(\tau_1)) \neq (C(\tau_2), H(\tau_2))$ for some $\tau_1, \tau_2 \in \sigma$. Consider any path $\pi \subset \sigma$ between τ_1 and τ_2. The path $\pi \subset \sigma$ describes a continuous motion of two polygons P_1 and P_2. If $C(\tau_1) \neq C(\tau_2)$, then π contains an event of type C_0, C_{12}, or C_{012} by continuity. Similarly, if $H(\tau_1) \neq H(\tau_2)$, then π contains an event of type H_0, H_{12}, or H_{012} again by

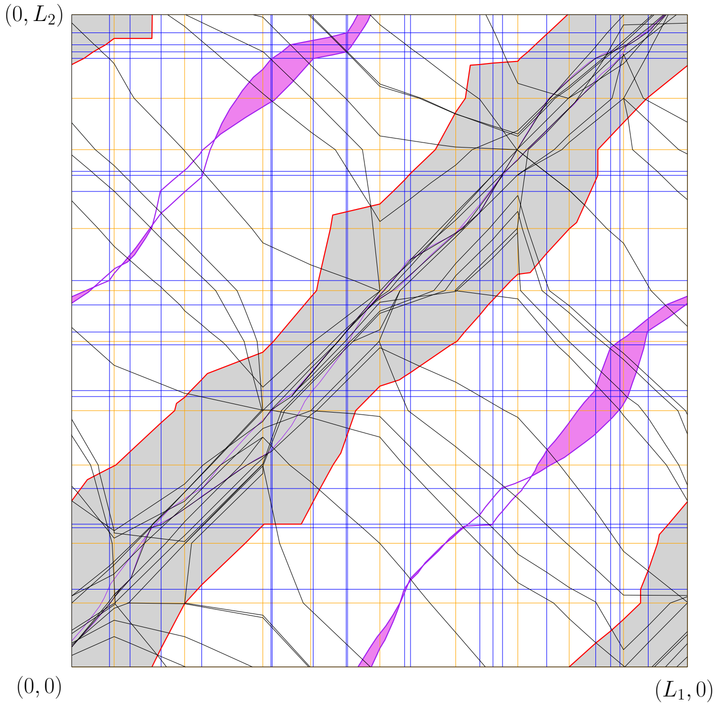


Figure 2.7: The arrangement $\mathcal{A}(\Gamma)$ of the event curves in the configuration space $\mathcal{K}_0$: The placements of $P_1(0)$ and $P_2(0)$ correspond to the left and right figures of Figure 2.1, respectively. Event curves of type C_0 (blue), H_0 (orange), H_{12} (black), C_{012} (red), and H_{012} (purple). Any configuration in the gray region is infeasible, so the feasible region $\mathcal{F}$ is the complement of the gray region. For any configuration τ in the purple region enclosed by H_{012} event curves, we have $|H(\tau)| = 4$.

continuity. However, σ contains no event of any type by definition of $\mathcal{A}(\Gamma)$, a contradiction. $\square$

2.3 Algorithms

Recall that all configurations in $\mathcal{K}_0$ assume P_0 to keep contact with both P_1 and P_2. Alternating the role of P_0 by P_1 or P_2, we achieve a complete description of the configuration space $\mathcal{K}$. Letting $\mathcal{K}_1$ and $\mathcal{K}_2$ be the analogous configuration space for P_1 and P_2, respectively, we have $\mathcal{K} = \mathcal{K}_0 \cup \mathcal{K}_1 \cup \mathcal{K}_2$.

In this section, we present an algorithm that computes an optimal feasible configuration that minimizes the area or the perimeter of the convex hull of the three convex polygons under translation. The arrangement $\mathcal{A}(\Gamma)$ of the event curves is indeed sufficient to deal with the area or perimeter function in each feasible cell.

Lemma 8. *Let σ be any cell of $\mathcal{A}(\Gamma)$ with $\sigma \subset \mathcal{F}$. The area function is hyperbolic paraboloidal on σ if $|H(\tau)| = 3$ for $\tau \in \sigma$, or linear otherwise; the perimeter function is unimodal on σ and on any edge incident to σ, and is of $O(1)$ descriptive complexity.*

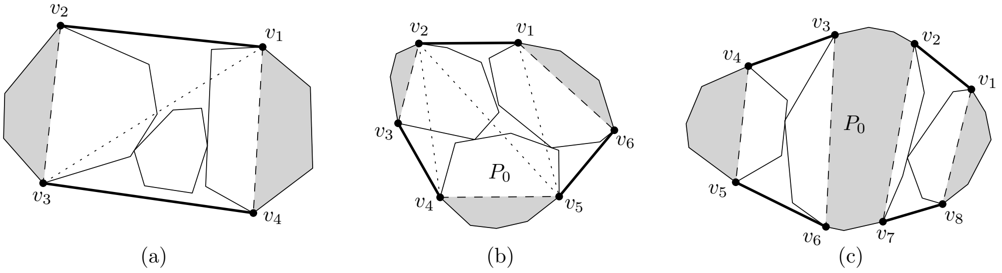
Proof. We analyze the area or perimeter function restricted in σ. By Lemma 7, the contacts $C = C(\tau)$ and the bridges $H = H(\tau)$ stay constant over $\tau \in \sigma$.

Each bridge in H is an edge of $\mathsf{conv}(\tau)$ between two vertices of the translated polygons. Let $(v_1, v_2), \ldots, (v_{2|H|-1}, v_{2|H|})$ denote the bridges in H, and x_i and y_i be the x- and y-coordinates of v_i in $\mathbb{R}^2$ under the translation by any $(\lambda_1, \lambda_2) \in \sigma$. Since the translations of $P_1(\lambda_1)$ and $P_2(\lambda_2)$ are linear inside σ, we can view both of x_i and y_i as functions linear on either λ_1 or λ_2.

We first consider the area function restricted in σ. Let f_σ be the function mapping $\tau \in \sigma$ to the area $\|\mathsf{conv}(\tau)\|$ of the convex hull of the translates by τ. We handle each case according the cardinality of H.

Note that $2 \leq |H| \leq 4$ since $\sigma \subset \mathcal{F}$. The lower bound is easy since $|H| = 1$ implies a vertex-vertex contact, that is, an event, which is impossible in a cell σ of $\mathcal{A}(\Gamma)$.

For the upper bound, observe that P_1 or P_2 cannot appear twice along the boundary of $\mathsf{conv}(\tau)$; if P_1 is the case, then there are four bridges connecting P_1 to P_0 or P_2 and P_2 cannot touch P_0, a contradiction. Since P_0 can appear at most twice along the boundary of $\mathsf{conv}(\tau)$, this implies that $|H| \leq 4$.



(a) (b) (c)

Figure 2.8: Illustration to the proof of Lemma 8 according to the number of bridges on a cell σ: (a) $|H| = 2$, (b) $|H| = 3$, (c) $|H| = 4$. The thick segments depict the bridges on the boundary of the corresponding convex hull and the gray regions are invariable components over all $\tau = (\lambda_1, \lambda_2) \in \sigma$.

Case of $|H| = 2$. In this case, $\mathsf{conv}(\tau)$ contains one of the three translates in its interior by any $\tau \in \sigma$. This implies that only two of them appear on the boundary of $\mathsf{conv}(\tau)$ and they are connected by two bridges (v_1, v_2) and (v_3, v_4). We assume that v_1 and v_4 belong to one polygon and v_2 and v_3 to the other, as shown in Figure 2.8(a). Observe then that $x_4 - x_1$ and $y_4 - y_1$ are constant for any $(\lambda_1, \lambda_2) \in \sigma$ since v_1 and v_4 belong to the same polygon. Analogously, both $x_3 - x_2$ and $y_3 - y_2$ are constant. We let a_1, b_1, a_2, b_2 be those constants such that $a_1 = x_4 - x_1$, $b_1 = y_4 - y_1$, $a_2 = x_3 - x_2$, and $b_2 = y_3 - y_2$.

The convex hull $\mathsf{conv}(\tau)$ for $\tau \in \sigma$ consists of two invariable components and a quadrilateral $Q(\tau)$ formed by v_1, v_2, v_3, v_4. Thus, $f_\sigma(\tau)$ is described as the area of the quadrilateral $Q(\tau)$ plus a constant. We now focus on the area of the quadrilateral $Q(\tau)$. Since $Q(\tau)$ is the union of two triangles $\triangle v_1v_2v_3$ and $\triangle v_1v_3v_4$, its area $\|Q(\tau)\|$ is explicitly formulated as follows:

$$\begin{aligned}
\|Q(\tau)\| &= \frac{1}{2}\begin{vmatrix} x_2-x_1 & y_2-y_1 \\ x_3-x_1 & y_3-y_1 \end{vmatrix} + \frac{1}{2}\begin{vmatrix} x_3-x_1 & y_3-y_1 \\ x_4-x_1 & y_4-y_1 \end{vmatrix} \\
&= \frac{1}{2}\begin{vmatrix} x_2-x_1 & y_2-y_1 \\ x_2-x_1+a_2 & y_2-y_1+b_2 \end{vmatrix} \\
&\quad +\frac{1}{2}\begin{vmatrix} x_2-x_1+a_2 & y_2-y_1+b_2 \\ a_1 & b_1 \end{vmatrix} \\
&= \frac{1}{2}|b_2(x_2-x_1)-a_2(y_2-y_1)| \\
&\quad +\frac{1}{2}|b_1(x_2-x_1-a_2)-a_1(y_2-y_1-b_2)|,
\end{aligned}$$

where $|\cdot|$ denotes the determinant of a square matrix or the absolute value of a real number by abuse of its usage. Thus, the area function f_σ is shown to be linear to $\tau = (\lambda_1, \lambda_2) \in \sigma$ since all x_i and y_i are linear in either λ_1 or λ_2.

Case of $|H| = 3$. In this case, the convex hull $\mathsf{conv}(\tau)$ for any $\tau \in \sigma$ consists of three components that do not change within τ and the hexagon formed by the six vertices involved in H. Let $v_1, \ldots, v_6$ be the vertices involved in H indexed in the counter-clockwise order along the boundary of $\mathsf{conv}(\tau)$ in such a way that v_4 and v_5 belong to P_0 as shown in Figure 2.8(b). Thus, f_σ is represented by the area of the hexagon plus a constant. The hexagon is further divided into four triangles $\triangle v_2v_3v_4$, $\triangle v_1v_5v_6$, $\triangle v_2v_4v_5$, and $\triangle v_1v_2v_5$. Observe that the area of all triangles but $\triangle v_1v_2v_5$ is linear in σ since v_2 and v_3 belong to a common polygon, v_1 and v_6 belong to another common polygon, and v_4 and v_5 are stationary. The area of

$\triangle v_1v_2v_5$ is represented as follows:

$$\|\triangle v_1v_2v_5\| \;=\; \frac{1}{2}\begin{vmatrix} x_1 - x_5 & y_1 - y_5 \\ x_2 - x_5 & y_2 - y_5 \end{vmatrix}.$$

Note that x_5 and y_5 are constant since v_5 is stationary. By substituting this for f_σ we obtain that $f_\sigma(\lambda_1, \lambda_2)$ is of the form $c_1\lambda_1\lambda_2 + c_2\lambda_1 + c_3\lambda_2 + c_4$ for some constants $c_1, \ldots, c_4$, which is reformulated into $c'_1(\lambda_1 + c'_2)(\lambda_2 + c'_3) + c'_4$. This equation indeed describes a hyperbolic paraboloid.

Case of $|H| = 4$. Let $v_1, \ldots, v_8$ be the vertices involved in H indexed in the counterclockwise order along the boundary of $\mathsf{conv}(\tau)$ in such a way that v_2, v_3, v_6, and v_7 belong to P_0, as shown in Figure 2.8(c). Note that P_0 must have four vertices in H in this case. Then, the convex hull $\mathsf{conv}(\tau)$ for $\tau \in \sigma$ consists of three invariable components and two quadrilaterals $Q_1(\tau)$ and $Q_2(\tau)$, formed by v_1, v_2, v_7, v_8 and by v_3, v_4, v_5, v_6, respectively. Thus, the function f_σ can be represented by $\|Q_1(\tau)\| + \|Q_2(\tau)\|$ plus a constant. Without loss of generality, we assume that v_1 and v_8 belong to P_1 while v_4 and v_5 belong to P_2. We then observe that $\|Q_i(\tau)\|$ is dependent only on λ_i for $i = 1, 2$ and, moreover, is linear in λ_i. Hence, we conclude that f_σ is linear on σ in this case.

Summarizing the above argument, we obtain the lemma about the area function.

We now turn to the perimeter function. Considering each case of $|H|$ as done above for the area function, the perimeter $|\mathsf{conv}(\tau)|$ of the convex hull $\mathsf{conv}(\tau)$ for $\tau = (\lambda_1, \lambda_2) \in \sigma$ is described as follows:

$$|\mathsf{conv}(\tau)| = c + \sum_{j=1}^{|H|} \sqrt{(x_{2j} - x_{2j-1})^2 + (y_{2j} - y_{2j-1})^2},$$

where $c > 0$ denotes a constant in $\mathbb{R}$ and x_i and y_i are the x- and y-coordinates of v_i as defined above. Since $|H| \leq 4$ and both x_i and y_i are linear in either λ_1 or λ_2, this simply shows that the perimeter function is of constant complexity. In addition, the convexity, and thus the unimodality of the perimeter function directly follows from Lemma 1: Any cell σ of the arrangement $\mathcal{A}(\Gamma)$ with $\sigma \subset \mathcal{F}$ corresponds to a set of translation vectors $\tau \in \mathbb{R}^4$ that lie in a common 2-dimensional affine subspace of $\mathbb{R}^4$ since the configurations $\tau = (\lambda_1, \lambda_2) \in \sigma$ translate each of P_1 and P_2 along a line in $\mathbb{R}^2$. By the same argument, restricted on any edge e of $\mathcal{A}(\Gamma)$ with $e \subset \mathcal{F}$, the convexity of the perimeter function can be shown if e is a line segment, since a line segment lying in the closure of a cell of $\mathcal{A}(\Gamma)$ corresponds to a line segment in $\mathbb{R}^4$.

What is left is the case where e is a hyperbolic arc. In this case, e is of type H_{012} and is described by an equation of the form $\lambda_2 = h(\lambda_1) = 1/(c_1 + c_2\lambda_1) + c_3$ for some constants c_1, c_2, and c_3 by Lemma 3 and its proof. As discussed above, the x and y-coordinates x_i and y_i of v_i are described as linear functions in λ_1 and λ_2. So, substituting λ_2 by $h(\lambda_1)$ in the above equation for $|\mathsf{conv}(\tau)|$ yields a function $g(\lambda_1)$ that maps λ_1 to the perimeter of the corresponding convex hull of the three polygons along the edge e. A careful analysis of the function g shows that its derivative has at most one real zero. We thus conclude that g has at most one local minimum except two limits of λ_1 corresponding to the endpoints of e. This implies the unimodality of the perimeter function along the hyperbolic edge e. $\square$

Our algorithm consists of two phases. First, it computes and stores the arrangement $\mathcal{A}(\Gamma)$ in the feasible region $\mathcal{F}$ only, and second it traverses each cell of the arrangement and computes an optimal translation for the area or the perimeter function restricted to the cell.

By Lemma 8, the second phase is relatively easy once the cells and the edges of $\mathcal{A}(\Gamma)$ lying in $\mathcal{F}$ are fully specified. At this phase, we visit every cell in $\mathcal{F}$ by crossing over an incident edge and thus moving to a neighboring cell. Then, by coherence, the description of the objective function restricted in the next cell can be obtained in constant time. Lemma 8 guarantees that the area or the perimeter function can be minimized in $O(1)$ time in a cell or on each of its bounding edges. Hence, the total time complexity of the second phase is bounded by $O(n^2)$ time by Lemma 6.

The arrangement can be easily computed in $O(n^2 \log n)$ time by a typical plane-sweep algorithm. In the following, we focus on improving the time bound to $O(n^2)$ for the task.

2.3.1 Computing the arrangement $\mathcal{A}(\Gamma)$ in $\mathcal{F}$

In order to compute the arrangement $\mathcal{A}(\Gamma)$, we first compute all the event curves in Γ with full description, and then identify all the intersections among them that lie in $\mathcal{F}$. Recall that $\mathcal{F}$ is bounded by two C_{012} event curves. Therefore, $\mathcal{F}$ can be constructed once we compute the two C_{012} event curves.

Preprocessing. As a preprocessing, we take any two polygons P_i and P_j for $0 \leq i < j \leq 2$ and move P_j around P_i keeping a contact to P_i in the counter-clockwise direction. During this motion, we gather all occurrences of vertex-vertex contact in order and store them into a sorted list $\mathcal{C}_{ij}$ with the corresponding pairs of vertices. In addition, we maintain two external common tangents of P_i and P_j and gather all occurrences at which one of the two tangents supports an edge of P_i or P_j. We also store them into a sorted list $\mathcal{H}_{ij}$ with the corresponding pairs of vertex and edge. Let us make each of $\mathcal{C}_{ij}$

and $\mathcal{H}_{ij}$ to be a circular list for later use. This preprocessing can be handled in $O(n)$ time as done in [25]. Observe that each member of $\mathcal{C}_{01}$ and $\mathcal{C}_{02}$ describes a C_0 event curve in $\mathcal{K}_0$, and each of $\mathcal{H}_{01}$ and $\mathcal{H}_{02}$ describes an H_0 event curve. We thus find all C_0 and H_0 event curves by traversing these lists.

Computing the event curves. Let G be the grid on $\mathcal{K}_0$ induced by all the C_0 and H_0 event curves. The other event curves can be obtained by tracing each across the grid cells of G. Consider the four H_{012} event curves. By Lemma 3, each H_{012} event curve γ appears to be a hyperbolic segment in each grid cell σ intersected by itself, and the equation of each segment in σ can be described with help of the lists $\mathcal{C}_{ij}$ and $\mathcal{H}_{ij}$. We first locate its starting point at $\lambda_1 = 0$ in $O(n)$ time from the lists $\mathcal{C}_{ij}$ and $\mathcal{H}_{ij}$, and then we trace γ cell by cell. As we walk along γ and move to the neighboring grid cell σ', we immediately tell the change of the contacts or the bridges from the lists $\mathcal{C}_{ij}$ and $\mathcal{H}_{ij}$ so that the equation of γ in σ' can be updated in $O(1)$ time. Hence, tracing γ spends time proportional to the number of grid cells of G that are intersected by γ. Lemma 4 tells us that the number of such grid cells, and thus the cost of tracing an H_{012} event curve is $O(n)$.

The other event curves of different types can be traced in the same fashion, taking $O(n)$ time for each. While tracing a C_{012} event curve, we can also specify all C_{12} events: this can be done by looking up the list $\mathcal{C}_{12}$ with a pointer that indicates the current contact between P_1 and P_2. Tracing an H_{12} event curve needs to look up the list $\mathcal{H}_{12}$; in fact, only the members of $\mathcal{H}_{12}$ can determine an H_{12} event curve. We hence can compute all the event curves in Γ with their full description in $O(n^2)$ time.

Specifying all necessary crossings. We then compute the arrangement $\mathcal{A}(\Gamma)$ in $\mathcal{F}$ by specifying all necessary crossings among the event curves in Γ.

Note that for any two event curves $\gamma_1, \gamma_2 \in \Gamma$, all the crossings between them can be computed in $O(n)$ time by Lemmas 3 and 5. For all pairs (γ_1, γ_2) of event curves such that γ_1 is of type C_{012} or H_{012} and γ_2 is of type C_{012}, H_{12}, or H_{012}, we are thus able to specify all the crossings between γ_1 and γ_2 in $O(n^2)$ time, since the number of such pairs (γ_1, γ_2) is $O(n)$. What remains is to specify the crossings among the H_{12} event curves.

For the last task, we take only feasible portions of every event curve into account. Let $\Gamma_{\mathcal{F}} := \{\gamma \cap \mathcal{F} \mid \gamma \in \Gamma\}$. Computing $\Gamma_{\mathcal{F}}$ can be done by cutting each $\gamma \in \Gamma$ by the C_{012} event curves and discarding its infeasible portions. Each feasible portion γ' of an event curve γ inherits its type from γ. Fortunately, this cutting does not increase the number of event curves much, especially, H_{12} event curves.

Lemma 9. *The number of H_{12} event curves in $\Gamma_{\mathcal{F}}$ is $O(n)$.*

Proof. We indeed show that any H_{12} event curve in Γ crosses a C_{012} event curve at most twice, which directly implies the lemma.

Let $\gamma \in \Gamma$ be an H_{12} event curve defined by (v, e). For any configuration $(\lambda_1, \lambda_2) \in \gamma$, $P_1(\lambda_1)$ and $P_2(\lambda_2)$ have a common external tangent that supports e. If it is also a C_{012} event, in addition $P_1(\lambda_1)$ and $P_2(\lambda_2)$ must touch each other. In such a scene, P_0 must touch both $P_1(\lambda_1)$ and $P_2(\lambda_2)$; there are at most two such possibilities, implying at most two coordinates $(\lambda_1, \lambda_2) \in \gamma$. □

Lemma 6 implies that the number of crossings in $\mathcal{F}$ between a fixed H_{12} event curve γ and all the other H_{12} event curves is $O(n)$. In the following, we show that all such crossings can be specified in $O(n)$ time. By Lemma 9, it suffices to conclude the total $O(n^2)$ time.

For the purpose, we need some more observations. Let $\gamma \in \Gamma_{\mathcal{F}}$ be an H_{12} event curve defined by a pair $(v, e) \in H_{12}$ of vertex v and edge e of P_1 and P_2. For any configuration $(\lambda_1, \lambda_2) \in \gamma$, $P_1(\lambda_1)$ and $P_2(\lambda_2)$ have a common external tangent that supports e. Let g be a partial function defined on $[0, L_1)$ whose graph corresponds to γ. Define $d_\gamma(\lambda_1)$ to be the distance between v and the endpoint of e that is the closer to v in the corresponding translates $P_1(\lambda_1)$ and $P_2(g(\lambda_1))$. Observe that d_γ is linear in a grid cell σ of G since the translations of P_1 and P_2 are linear in σ along γ.

On the other hand, we consider the other external common tangent $\ell(\lambda_1)$ of $P_1(\lambda_1)$ and $P_2(g(\lambda_1))$. When an edge e' of P_1 or P_2 lies on $\ell(\lambda_1)$, we have a crossing between γ and another H_{12} event curve γ' defined by (v', e') for the vertex v' lying on $\ell(\lambda_1)$; we let $\delta_{v'e'} := d_\gamma(\lambda_1)$ at such a value of λ_1. By a geometric observation, at such a crossing, $P_1(\lambda_1)$ and $P_2(g(\lambda_1))$ have two external common tangents, one supporting e and the other supporting e'; this fixes a unique value of $d_\gamma(\lambda_1)$ to be $\delta_{v'e'}$. This implies that for any λ_1, γ' crosses γ at $(\lambda_1, g(\lambda_1))$ if and only if $d_\gamma(\lambda_1) = \delta_{v'e'}$. See Figure 2.9 for an illustration.

We thus perform the following procedure. We compute the value $\delta_{v'e'}$ for all $(v', e') \in H_{12}$ and store them into a sorted array Δ with corresponding label (v', e'). Then, we traverse the cells intersected by γ in order along γ to find all occurrences such that $d_\gamma(\lambda_1) = \delta$ for some $\delta \in \Delta$. This completely specifies all crossings between γ and all other H_{12} event curves in $\Gamma_{\mathcal{F}}$. To compute Δ, initially make P_1 and P_2 touch each other, keeping a common tangent going through v and e, and consider the other common tangent line ℓ. If we move P_2 in the direction parallel to e and away from P_1,

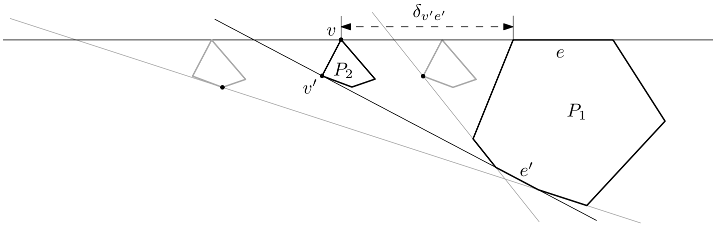


Figure 2.9: Illustration to the list Δ.

then the tangent line ℓ will rotate monotonously in one direction. This implies that the order of $\delta_{v'e'}$ follows from the order of (v', e') in the list H_{12}. Thus, we can compute Δ in $O(n)$ time.

Let $\delta_1, \ldots, \delta_m$ be the members of Δ in the order. Once we compute Δ, we walk along γ by increasing λ_1 to find all λ_1 such that $d_\gamma(\lambda_1) = \delta$ holds for some $\delta \in \Delta$. Since d_γ is linear in each cell σ of G intersected by γ, the task is not difficult if we maintain a variable a such that $\delta_a \leq d_\gamma(\lambda_1) < \delta_{a+1}$ for the current value of λ_1. Hence, we can find all crossings on $\gamma \cap \sigma$ with the other H_{12} event curves in time $O(1 + c)$, where c is the number of the reported crossings in σ. If we sum up this over all grid cells intersected by γ, we obtain $O(n)$ time bound.

Putting it all together, we can specify all intersections among curves in $\Gamma_{\mathcal{F}}$ in $O(n^2)$ time, which are the vertices of $\mathcal{A}(\Gamma_{\mathcal{F}})$. We then cut each event curve $\gamma \in \Gamma_{\mathcal{F}}$ by the crossings on γ to obtain the edges of $\mathcal{A}(\Gamma_{\mathcal{F}})$. Since we now know all the information about the vertices and edges of the planar arrangement $\mathcal{A}(\Gamma_{\mathcal{F}})$, all of its cells can be specified in time proportional to the number of edges: one possible way would incrementally insert the edges one by one into the plane, keeping the information of current cells. Since each edge insertion causes the change of exactly one of the current cells, this takes $O(1)$ time per edge insertion with a standard data structure representing the current status, see for an example [12, Chapter 8.3]. As a result, we can build the arrangement $\mathcal{A}(\Gamma_{\mathcal{F}})$ in $O(n^2)$ time.

We finally conclude our main result.

Theorem 1. *Given three convex polygons P_0, P_1, and P_2 having a total of n vertices, one can find in $O(n^2)$ time using $O(n^2)$ space an optimal pair (τ_1, τ_2) of translation vectors such that the interiors of P_0, $\tau_1 P_1$ and $\tau_2 P_2$ are disjoint, and the area $\|\mathsf{conv}(P_0 \cup \tau_1 P_1 \cup \tau_2 P_2)\|$ or the perimeter $|\mathsf{conv}(P_0 \cup \tau_1 P_1 \cup \tau_2 P_2)|$ is minimized.*

2.4 Concluding Remarks

We study the bundling problem for three convex polygons in the plane and present an efficient algorithm for the problem with quadratic running time and space. Our contributions can be summarized as follows:

- We study the configuration space for three convex polygons. The configuration is described as parametrization of the traveling distance between two convex polygons in Section 2.2.1. We show that the configuration space can be decomposed into $\Theta(n^2)$ cells in each of which the combinatorial structure of the convex hull remains the same.

- We show that the description of the objective function for each cell can be fully described using constant space and the function description for a neighboring cell can be obtained in constant time by using coherence. By using the description of the objective function, we develop an efficient algorithm for this problem in $O(n^2)$ time, which is faster than typical plane-sweep algorithm.

- Our approach in this chapter can naturally be extended to the cases of $k > 3$

convex polygons. It would be interesting to investigate their combinatorial and algorithmic complexity for small k.

III. Bundling Two Convex Polytopes

In this chapter, we study the problem of *bundling* them under translations given two convex d-polytopes P and Q in a d-dimensional space for some constant $d \geq 3$. More precisely, the problem asks to find a translation vector $t \in \mathbb{R}^d$ of Q that minimizes the volume or the surface area of the convex hull of $P \cup Q_t$ under the restriction that their interiors remain disjoint, where $Q_t = \{q + t \mid q \in Q\}$.

We give an $O(n^3)$-time algorithm for $d = 3$ to find a translation vector t^* that attains the minimum volume or surface area of the convex hull of $P \cup Q_{t^*}$, where n denotes the total number of vertices of both polytopes P and Q. Our algorithm constructs an arrangement in our translation space and evaluates the volume or surface area function on each cell of the arrangement. Our approach extends to any fixed dimension $d > 3$, yielding a first exact algorithm with running time $O(n^{d+\lfloor \frac{d}{2} \rfloor (d-3)})$.

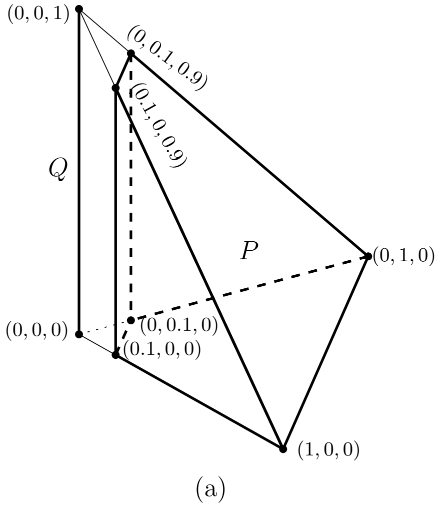
3.1 Preliminaries

For any subset $A \subseteq \mathbb{R}^d$, let $\mathrm{bd}(A)$ be the boundary of A and $\mathsf{conv}(A)$ the convex hull of A. We denote by $|A|$ and $\|A\|$ the surface area and the volume of A, respectively, when both are well defined for A.

Let P and Q be convex d-polytopes in $\mathbb{R}^d$ and n denote the number of vertices of P and Q in total. Without loss of generality, we assume that P is stationary and only Q can be translated by vectors $t \in \mathbb{R}^d$. We denote by Q_t the translate of Q by $t \in \mathbb{R}^d$, that is, $Q_t = \{q + t \mid q \in Q\}$.

Let $\mathrm{vol}(t) := \|\mathsf{conv}(P \cup Q_t)\|$ and $\mathrm{surf}(t) := |\mathsf{conv}(P \cup Q_t)|$. Once t is fixed and the description of $\mathsf{conv}(P \cup Q_t)$ is identified, we can evaluate $\mathrm{vol}(t)$ and $\mathrm{surf}(t)$ in time linear in the complexity of $\mathsf{conv}(P \cup Q_t)$.

Ahn et al. [3] showed that the function $\mathrm{vol}(t)$ is convex on the whole domain $\mathbb{R}^d$. The convexity of the function $\mathrm{surf}(t)$ was proved by Ahn and Cheong [5] for the 2-dimensional case only, but their argument can easily be extended to higher dimensions by using Cauchy's surface area formula for a compact convex subset (see Theorem 5.5.2 in [24]).



(a) (b)

Figure 3.1: Two polytopes P and Q that are separated in their optimal placement with respect to both (a) volume and (b) surface area.

For our problem where no overlap between the two polytopes is allowed, one might conjecture that there should be an optimal solution such that the two polytopes are in contact with each other. Much to our surprise, this is not always the case. Figure 3.1 illustrates an example of two polytopes P and Q such that their translates must be *separated* at their optimal placement with respect to both of the volume

$\mathrm{vol}(t)$ and the surface area $\mathrm{surf}(t)$. The construction starts with a tetrahedron $T = \mathsf{conv}(\{(0,0,0),(1,0,0),(0,1,0),(0,0,1)\})$ in $\mathbb{R}^3$ with the (x,y,z)-coordinate system. Let P be the polytope obtained by intersecting T with the halfspace $\{x+y \geq 0.1\}$, and let Q be the line segment between two points $(0,0,0)$ and $(0,0,1)$.

Then, this original placement of P and Q minimizes the volume function $\mathrm{vol}(t)$, that is, $\mathrm{vol}(t)$ attains its minimum at $t = (0,0,0)$. Observe that the corresponding convex container is $T = \mathsf{conv}(P \cup Q)$ as illustrated in Figure 3.1(a). One can check that the volume $\mathrm{vol}(t)$ increases if Q translates in any direction from its original position. The convexity of $\mathrm{vol}(t)$ implies that this placement is indeed the unique minimum of $\mathrm{vol}(t)$. Clearly, P and Q are separated in this optimal placement. Further, the minimum surface area of the convex hull of P and Q_t occurs at $t \approx (0.041, 0.041, -0.035)$, as illustrated in Figure 3.1(b). In this placement, P and Q_t are separated as well. Note that this construction of P and Q can be extended to dimensions higher than 3.

As discussed above, the objective functions $\mathrm{vol}(t)$ and $\mathrm{surf}(t)$ are convex in $t \in \mathbb{R}^d$. Thus, if t^* is an optimal solution for our problem without overlap, then either P and Q_{t^*} are separated or P and Q_{t^*} are in contact. In the former case, which is also the case of the construction in Figure 3.1, t^* minimizes $\mathrm{vol}(t)$ or $\mathrm{surf}(t)$ over the whole domain $\mathbb{R}^d$, so any algorithm minimizing $\mathrm{vol}(t)$ or $\mathrm{surf}(t)$ when overlap is allowed can handle this case, see for example [3]. While it is not mentioned in [3], their algorithm works for minimizing the surface area function $\mathrm{surf}(t)$.

In this paper, therefore, we focus on the problem where the two polytopes P and Q_t are required to be in contact with each other. That is, we want to minimize the volume or the surface area of the convex hull under the restriction that the two polytopes are in contact.

Representing the configuration space Without loss of generality, we assume that Q contains the origin. Let r be a point of Q that corresponds to the origin. We call it *the reference point* of Q. Any translation of Q is then specified by a location of the reference point. Imagine that we slide Q along the boundary of P over all possible translations t such that P and Q_t are in contact. Then, the trajectory of r form the boundary of the *Minkowski difference* of P and Q, denoted by $P \oplus (-Q)$, where $\oplus$ denotes the Minkowski sum and $-Q$ denotes the point reflection of Q with respect to the origin. This fact is already well known in motion planning [12].

Lemma 10. *The set of translations $t \in \mathbb{R}^d$ such that P and Q_t are in contact forms the boundary of $P \oplus (-Q)$.*

In our problem, we restrict the two polytopes P and Q to be in contact, and thus the set of all such translations determines the space of all configurations. Lemma 10 suggests that the *configuration space* $\mathcal{K}$ should be defined as the boundary of $P \oplus (-Q)$.

Since P and Q are convex, computing the configuration space $\mathcal{K} = \mathrm{bd}(P \oplus (-Q))$ for P and Q, and consequently specifying all the faces of $\mathcal{K}$ can be done efficiently by a lifting technique, called the *Cayley trick*. This concept starts by introducing the *weighted Minkowski sum* $(1 - \lambda)P_1 \oplus \lambda P_2$ of two convex d-polytopes P_1 and P_2 for $0 \leq \lambda \leq 1$. The Cayley trick then lifts P_1 and P_2 into a space of one dimension higher with a $(d + 1)$-st coordinate x_{d+1} as follows: P_1 is embedded in the hyperplane $\{x_{d+1} = 0\}$ and P_2 in $\{x_{d+1} = 1\}$. To obtain the weighted Minkowski sum of P_1 and P_2 for any $0 \leq \lambda \leq 1$, one computes the convex hull $\mathsf{conv}(P_1 \cup P_2)$ in $\mathbb{R}^{d+1}$ and slices it through the hyperplane $\{x_{d+1} = \lambda\}$. Observe that the Minkowski sum $P_1 \oplus P_2$ is just a scaled copy of the slice at $\lambda = \frac{1}{2}$. We refer to Huber et al. [18] for more details regarding the Cayley trick.

Note that the convex hull of P_1 and P_2 in $\mathbb{R}^{d+1}$ coincides with the convex hull of the vertices of P_1 and P_2. Since the complexity of $P_1 \oplus P_2$ does not exceed that of the convex hull $\mathsf{conv}(P_1 \cup P_2)$, we have the upper bound $O((n_1 + n_2)^{\lfloor \frac{d+1}{2} \rfloor})$ on the complexity of the Minkowski sum $P_1 \oplus P_2$ of two convex d-polytopes [31], where n_1 and n_2 denote the number of vertices of P_1 and P_2, respectively. Computing $P_1 \oplus P_2$ can be done in $O((n_1+n_2)\log(n_1+n_2)+(n_1+n_2)^{\lfloor \frac{d+1}{2} \rfloor})$ time [9] for any fixed $d \geq 2$. Using this in our configuration space $\mathcal{K}$ yields the following.

Lemma 11. *Let P and Q be convex d-polytopes with n vertices in total for any fixed $d \geq 2$. The configuration space $\mathcal{K} = \mathrm{bd}(P \oplus (-Q))$ for P and Q has $O(n^{\lfloor \frac{d+1}{2} \rfloor})$ combinatorial complexity and can be computed in $O(n \log n + n^{\lfloor \frac{d+1}{2} \rfloor})$ time.*

In the following sections, we introduce a decomposition of the configuration space $\mathcal{K}$ and describe a complete algorithm, mainly for dimension $d = 3$. This will lead to a direct extension to higher dimension for $d > 3$.

3.2 Subdividing the Configuration Space

In this section, we assume $d = 3$. For any translation $t \in \mathcal{K}$, P and Q_t are in contact. More precisely, a vertex, edge, or facet f of P touches a vertex, edge, or facet g of Q_t for $t \in \mathcal{K}$, while the interiors of P and Q_t are disjoint. We call the pair (f, g) the *contact pair* at translation $t \in \mathcal{K}$, denoted by $C(t)$. Our approach is to subdivide the configuration space $\mathcal{K}$ into cells so that the contact pair and the convex hull structure of the polytopes do not change within each cell. We then obtain an expression for the volume or surface area function, $\mathrm{vol}(t)$ or $\mathrm{surf}(t)$, in each cell, and compute its minimum.

By Lemmas 10 and 11, the configuration space $\mathcal{K} = \mathrm{bd}(P \oplus (-Q))$ describes all possible translation vectors and can be constructed in $O(n^2)$ time for $d = 3$. In the following, we further investigate the structure of the configuration space $\mathcal{K}$ to understand the correspondence between each of its faces and the corresponding contact pair.

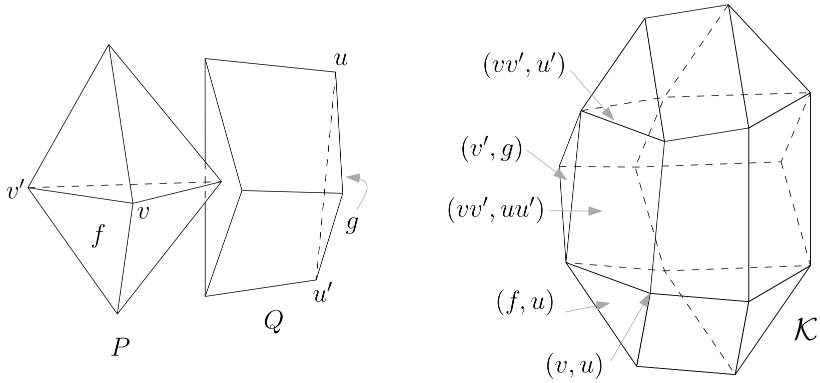


Figure 3.2: Contact pairs between P and Q, and the configuration space $\mathcal{K}$. Each of vertex-facet pairs, (f, u) and (v', g), defines a facet, an edge-edge pair (vv', uu') defines a facet, a vertex-edge pair (vv', u') defines an edge, and a vertex-vertex pair (v, u) defines a vertex in the configuration space $\mathcal{K}$.

Imagine that Q is translated around P in all possible ways, staying in contact with each other. This motion is piecewise linear: For any face a of P and face b of Q, let $\sigma_{a,b} \subset \mathcal{K}$ denote the set of translations $t \in \mathcal{K}$ such that $C(t) = (a, b)$. In the following, we discuss only the case where $\sigma_{a,b} \neq \emptyset$.

(1) When a is a facet and b is a vertex, $\sigma_{a,b}$ forms a polygon, which is in fact a translate of a. See (f, u) in Figure 3.2. When a is a vertex and b is a facet, then $\sigma_{a,b}$ forms a polygon which is a translate of the point reflection of b. See (v', g) in Figure 3.2. More importantly, observe that $\sigma_{a,b} = a \oplus (-b)$ forms a facet (or a 2-face) of $\mathcal{K}$.

(2) When both a and b are edges, the subset $\sigma_{a,b}$ forms a parallelogram $a \oplus (-b)$ that is a facet of $\mathcal{K}$. See (vv', uu') in Figure 3.2.

(3) When a is a vertex and b is an edge, $\sigma_{a,b}$ forms a line segment that is a translate of $-b$ by translation vector a. When a is an edge and b is a vertex, $\sigma_{a,b}$ forms a line segment that is a translate of a. See (vv', u') in Figure 3.2. In this case, $\sigma_{a,b}$ forms an edge of $\mathcal{K}$.

(4) When both a and b are vertices, $\sigma_{a,b}$ is a point $a - b$, which is a vertex of $\mathcal{K}$. See (v, u) in Figure 3.2.

These observations are summarized as follows.

Lemma 12. *Each face (of any dimension) of the configuration space $\mathcal{K}$ corresponds to the set of translations t with the same contact pair $C(t)$.*

Hull event planes and horizons In addition, we have to handle changes in the combinatorial structure of the convex hull $\mathsf{conv}(P \cup Q_t)$ while t continuously varies over $\mathcal{K}$. A change in the structure of the convex hull occurs when a vertex of P and Q either sticks out $\mathsf{conv}(P \cup Q_t)$ from inside or sinks into $\mathsf{conv}(P \cup Q_t)$ from its boundary. In either case, such a change corresponds to the following degenerate situation: Q_t touches the supporting plane of a facet f of P in the same side where P lies. For any facet f of P, consider the set Π_f of all such degenerate translation vectors $t \in \mathbb{R}^3$. Since a unique vertex of Q_t must lie on the supporting plane of f for all $t \in \Pi_f$, this set Π_f forms a plane in the space $\mathbb{R}^3$. We then define $h_f := \Pi_f \cap \mathcal{K}$. We call Π_f the *hull event (hyper)plane* and h_f the *hull event horizon* for facet f. Each $t \in h_f$ is called a *hull event*. The same holds for any facet of Q.

Lemma 13. *For any facet f of P or Q, the hull event horizon h_f forms a closed polygonal curve in $\mathcal{K}$ consisting of $O(n^2)$ line segments.*

Proof. By definition, Π_f is a plane and $h_f = \Pi_f \cap \mathcal{K}$. Thus, h_f is an intersection between a plane and $\mathcal{K}$. As observed in Lemmas 10 and 11, $\mathcal{K}$ is a convex polytope of complexity $O(n^2)$. Hence the lemma follows. □

Now, we consider the subdivision $\mathcal{A}$ of $\mathcal{K}$ induced by h_f for all facets f of P and Q. Observe that for each cell σ of $\mathcal{A}$, the structure of the convex hull $\mathsf{conv}(P \cup Q_t)$ for all $t \in \sigma$ does not change, as for such a change we would need to cross at least one hull event horizon. Since all the hull event horizons are polygonal on $\mathcal{K}$, $\mathcal{A}$ refines the faces of $\mathcal{K}$. We thus regard $\mathcal{A}$ as another convex polytope with parallel facets and edges. Together with Lemma 12, we conclude the following.

Lemma 14. *Let σ be a face of $\mathcal{A}$. Then, both the contact pair $C(t)$ and the structure of the convex hull $\mathsf{conv}(P \cup Q_t)$ stay constant over all $t \in \sigma$.*

We now bound the complexity of $\mathcal{A}$ with help of the following observation.

Lemma 15. *For any two distinct facets f and g of P or Q, the hull event horizons h_f and h_g cross at most twice.*

Proof. By definition, $h_f \cap h_g = \Pi_f \cap \Pi_g \cap \mathcal{K}$. Thus, the intersection of two hull event horizons is the intersection of $\mathcal{K}$ and a line. Since $\mathcal{K}$ is a convex polytope, $h_f \cap h_g$ consists of at most two points. □

Since there are $O(n)$ facets of P and Q in total, Lemmas 13 and 15 imply an immediate upper bound $O(n^3)$ on the complexity of $\mathcal{A}$.

Lemma 16. *The polytope $\mathcal{A}$ consists of $O(n^3)$ faces (vertices, edges, and facets).*

This bound $O(n^3)$ might seem easy and improvable, but it is shown to be tight in the worst case.

3.2.1 Tight lower bound construction for $\mathcal{A}$

Figure 3.3 illustrates an instance of two polytopes which make $\Omega(n)$ closed polygonal curves, each consisting of $\Omega(n^2)$ line segments. Let us describe how to construct two polytopes P and Q more precisely. Figure 3.3(a) illustrates Q viewed at approximately 7 times magnification. It looks like an "axe" whose head is the segment uu' and whose blade is the polygonal chain marked by thick segments in the figure. The polytope P is illustrated in Figure 3.3(b), which can be described as the convex hull of a folding fan with rotating center (pivot) at c and the zigzag edges (thick segments) along its tip. Then we could see that every blade edge constitutes an edge-edge contact pair with each zigzag edge as the blade chain is turning dully. Figure 3.3(c) shows the configuration space $\mathcal{K}$ for P and Q, which has $\Omega(n^2)$ parallelogram facets corresponding to those edge-edge contact pairs.

Note now that all front facets incident to c have almost the same slope, and all back facets incident to c have almost the same slope as well. Consider the hull event horizon h_f for a front facet f incident to c. Imagine the motion of Q_t (in the original scale) as t moves along h_f. Then during this motion, the vertex u'' of Q should lie on the supporting plane of f, and each zigzag edge of P sweeps over all the blade edges of Q, resulting in $\Omega(n^2)$ crossings with parallelogram facets of $\mathcal{K}$. See the blue curves in Figure 3.3(d). Similarly, for any other front and back facet f', the motion of Q_t along $t \in h_{f'}$ results in $\Omega(n^2)$ crossings over the parallelogram facets of $\mathcal{K}$. Therefore, the subdivision $\mathcal{A}$ of $\mathcal{K}$ has complexity $\Omega(n^3)$.

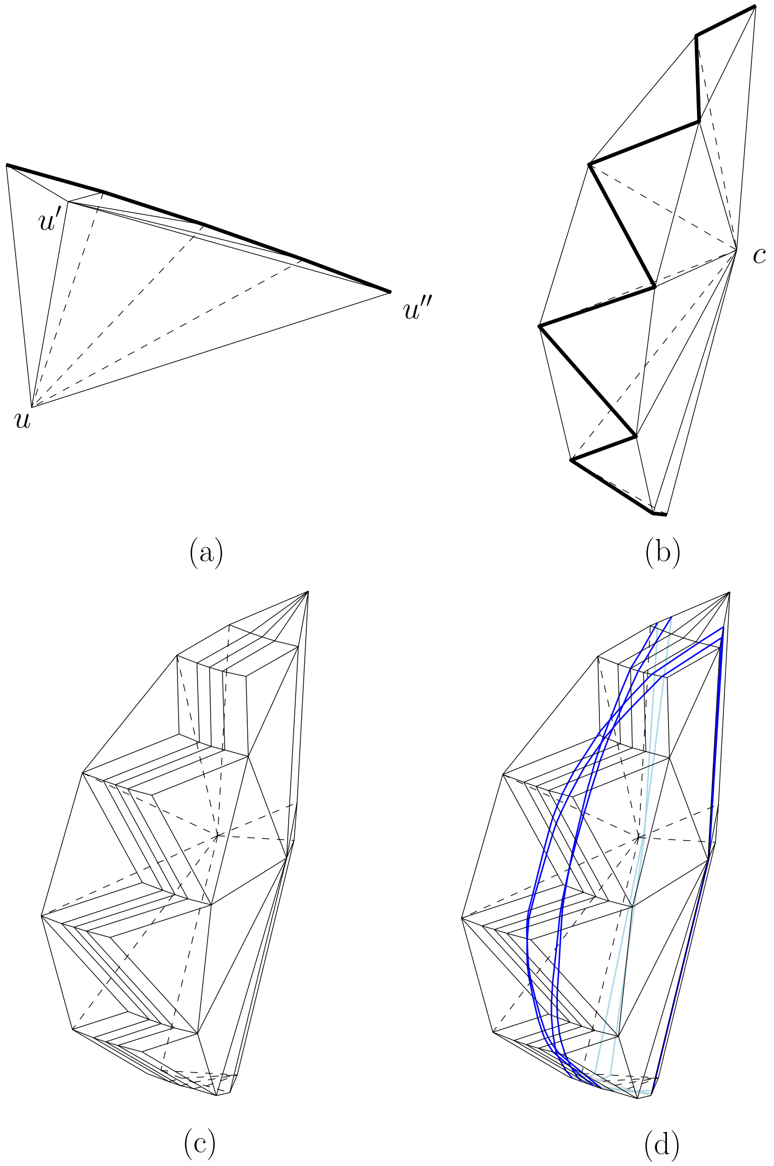


Figure 3.3: A construction of two polytopes P and Q such that each hull event horizon crosses $\Omega(n^2)$ facets of $\mathcal{K}$. (a) Polytope Q (at 7 times magnification). (b) Polytope P. (c) $P \oplus (-Q)$ whose boundary is $\mathcal{K}$. (d) Four hull event horizons (blue) are drawn on $\mathcal{K}$. Each of them crosses $\Omega(n^2)$ facets of $\mathcal{K}$.

3.3 Algorithm

In this section, we describe our algorithm for the case of dimension $d = 3$. Given two convex 3-polytopes P and Q with n vertices in total, our algorithm runs through three stages:

(i) Compute the configuration space $\mathcal{K}$.

(ii) Compute the subdivision $\mathcal{A}$ of the faces of $\mathcal{K}$.

(iii) For each face σ of $\mathcal{A}$, minimize the volume $\mathrm{vol}(t)$ or surface area $\mathrm{surf}(t)$ over $t \in \sigma$.

This basically performs an optimization process over the whole configuration space $\mathcal{K}$. Thus, the correctness of our algorithm follows directly. In the following, we describe each stage in more details.

Stage (i) can be done by computing the Minkowski sum $P \oplus (-Q)$, which takes $O(n^2)$ time as described in Lemma 11. Recall that $\mathcal{K}$ consists of $O(n^2)$ faces.

In Stage (ii), we repeatedly insert every hull event horizon h_f into $\mathcal{K}$; that is, we cut those faces of $\mathcal{K}$ crossed by h_f and produce new faces. Let $\mathcal{A}_i$ be the resulting subdivision after the i-th insertion of an event hull horizon, so $\mathcal{K} = \mathcal{A}_0$ and $\mathcal{A} = \mathcal{A}_m$, where $m = O(n)$ denotes the number of facets of P and Q. At the i-th insertion, let h_f be the horizon to be inserted. We then compute the corresponding hull event plane Π_f and merge it with $\mathcal{A}_{i-1}$ by tracing h_f and specifying those faces of $\mathcal{A}_{i-1}$ crossed by h_f. This process can be done in time proportional to the number of faces of $\mathcal{A}_{i-1}$ crossed by h_f, which is bounded by $O(n^2 + i)$ according to Lemmas 13 and 15. Summing this bound over all $i = 1, \ldots, m$ results in $O(mn^2 + m^2) = O(n^3)$.

Stage (iii) performs an actual optimization process for each face σ of $\mathcal{A}$. By

Lemma 14, we know that restricting our objective function to each face σ of $\mathcal{A}$ guarantees no change in the contact pair $C(t)$ and the structure of the convex hull over $t \in \sigma$. This means that every vertex of $\mathsf{conv}(P \cup Q_t)$ can be represented by a linear function of t, and $\mathsf{conv}(P \cup Q_t)$ can be triangulated into the same family of tetrahedra in the following way: (1) Triangulate each facet of $\mathsf{conv}(P \cup Q_t)$ if it is not a triangle, and (2) triangulate the interior of $\mathsf{conv}(P \cup Q_t)$ by choosing a point c in the interior of P and connecting c to all the vertices of $\mathsf{conv}(P \cup Q_t)$ with edges.

Let $\mathcal{T}_\sigma$ be the set of those triangles on $\mathrm{bd}(\mathsf{conv}(P \cup Q_t))$ obtained in step (1). Also, for each triangle $\triangle \in \mathcal{T}_\sigma$, let $\triangle^+$ be the tetrahedron with base $\triangle$ and apex c. Since P is assumed to be stationary, c is fixed and the vertices of each triangle $\triangle \in \mathcal{T}_\sigma$ are linear functions of t on σ. We hence write $\triangle(t)$ and $\triangle^+(t)$ as functions of $t \in \sigma$ to denote the geometric triangle and tetrahedron for any fixed $t \in \sigma$. Observe that $\mathrm{vol}(t) = \sum_{\triangle \in \mathcal{T}_\sigma} \|\triangle^+(t)\|$ and $\mathrm{surf}(t) = \sum_{\triangle \in \mathcal{T}_\sigma} |\triangle(t)|$. The volume of a tetrahedron is represented by a cubic polynomial in the coordinates of its vertices, and the area of a triangle by a quadratic polynomial. That is, in a face σ of $\mathcal{A}$, the volume and surface area functions are represented by polynomials of degree three or two. Hence, they can be minimized in $O(1)$ time after having its explicit formula in $O(\mathrm{card}(\mathcal{T}_\sigma)) = O(n)$ time, where $\mathrm{card}(\mathcal{T}_\sigma)$ is the cardinality of $\mathcal{T}_\sigma$. Hence, $O(n)$ time is sufficient for each face of $\mathcal{A}$ to minimize $\mathrm{vol}(t)$ or $\mathrm{surf}(t)$. This implies an $O(n^4)$-time algorithm as $\mathcal{A}$ consists of $O(n^3)$ faces.

Below, we will show that we can do this task in $O(1)$ average time for each face σ of $\mathcal{A}$ by exploiting coherence between adjacent facets.

3.3.1 Exploiting coherence

Let σ and σ' be two adjacent facets of $\mathcal{A}$, sharing an edge e. Assume that we have just processed σ and we are about to process σ'. We maintain $\mathcal{T}_\sigma$ and all formulas representing $|\triangle(t)|$ and $\|\triangle^+(t)\|$ for each $\triangle \in \mathcal{T}_\sigma$ and their sums (which are $\mathrm{surf}(t)$ and $\mathrm{vol}(t)$). In order to efficiently process the next facet σ', we need to update these invariants. We have two cases here: the edge e is either a portion of an edge of $\mathcal{K}$ or a portion of a hull event horizon h_f for some facet f of P or Q.

For the former case, we have $\mathcal{T}_{\sigma'} = \mathcal{T}_\sigma$, but the coordinates of the vertices of $\mathsf{conv}(P \cup Q_t)$ should be changed, since the contact pair $C(t)$ changes by Lemma 12. This causes changes in all formulas for $|\triangle(t)|$ and $\|\triangle^+(t)\|$ for $\triangle \in \mathcal{T}_{\sigma'}$. Thus, in this case, we spend $O(n)$ time because $\mathcal{T}_\sigma$ consists of $O(n)$ triangles.

For the latter case, where e is a portion of h_f for some facet f of P or Q, σ and σ' belong to a common facet of $\mathcal{K}$. Thus, the contact pair $C(t)$ does not change over $\sigma \cup \sigma'$, while the triangulations $\mathcal{T}_\sigma$ and $\mathcal{T}_{\sigma'}$ differ. Note that for $\triangle \in \mathcal{T}_\sigma \cap \mathcal{T}_{\sigma'}$, the formulas for $|\triangle(t)|$ and $\|\triangle^+(t)\|$ remain the same over $t \in \sigma \cup \sigma'$. Thus, in this case, we are interested in those triangles $\triangle$, which are in the symmetric difference between $\mathcal{T}_\sigma$ and $\mathcal{T}_{\sigma'}$, denoted by $\mathcal{T}_e$. Since $e \subset h_f$, for any $t \in e$, P and Q_t form a degenerate configuration such that a vertex u of P or Q lies on the supporting plane of f. As t moves into σ' or into σ, the triangles on f disappear and the triangles determined by each edge incident to f and vertex u appear. This implies that the number of triangles in the symmetric difference $\mathcal{T}_e$ does not exceed twice the number of edges incident to facet f. In order to maintain our invariants, we are done by specifying all appearing and disappearing triangles $\triangle \in \mathcal{T}_e$ and then updating the formulas for the volume or surface area. This can be done in $O(N_f)$ time, where N_f denotes the number of edges incident to f.

To conclude our main result, we need the following lemma.

Lemma 17. *The total number of triangles in $\mathcal{T}_e$ over all edges e of $\mathcal{A}$ that are portions of some hull event horizon is bounded by $O(n^2 \cdot \sum_f N_f) = O(n^3)$.*

Proof. For each facet f of P and Q, the corresponding hull event horizon h_f consists of $O(n^2)$ edges of $\mathcal{A}$. Let E_f be the set of edges of $\mathcal{A}$ that are portions of h_f. Then, we have $\sum_{e \in E_f} \mathrm{card}(\mathcal{T}_e) = O(n^2 \cdot N_f)$, where $\mathrm{card}(\mathcal{T}_e)$ is the cardinality of $\mathcal{T}_e$. This holds for any facet f of P and Q. Therefore, the total time for the updates is bounded by $\sum_f \sum_{e \in E_f} \mathrm{card}(\mathcal{T}_e) = O(n^2 \cdot \sum_f N_f)$, which is at most $O(n^3)$ as the number of facets of 3-polytopes P and Q is $O(n)$. □

We are now ready to describe stage (iii) of our algorithm. We traverse all facets of $\mathcal{A}$ from an arbitrary initial facet σ_0. For the first time, we compute $\mathsf{conv}(P \cup Q_t)$ for some $t \in \sigma_0$ and all the invariants from scratch in $O(n^2)$ time. We then minimize our objective function $\mathrm{vol}(t)$ or $\mathrm{surf}(t)$ over $t \in \sigma_0$. As we move on to the next facet σ' from the current facet σ, we update our invariants as described above, according to the type of the edge e between σ and σ', and minimize the objective function. We repeat this procedure until we traverse all the facets of $\mathcal{A}$.

By a standard traverse, such as the depth first search, we do not cross the same edge more than twice. This implies that the total cost of crossing edges that come from hull event horizons is not more than $O(n^3)$ by Lemma 17. Moreover, if we take a little smarter traverse order, then we can bound the number of crossed edges that are portions of edges of $\mathcal{K}$, by $O(n^2)$. Since each edge crossing of this type costs $O(n)$ time, we finally bound the total cost of updates by $O(n^3)$ time.

We finally conclude the following theorem.

Theorem 2. *Given two convex* 3*-polytopes* P *and* Q *with* n *vertices in total, a minimum convex container bundling* P *and* Q *under translations without overlap can be computed in* $O(n^3)$ *time with respect to volume or surface area.*

3.4 Extension to Higher Dimensions

Our approach to dimension $d = 3$ immediately extends to any fixed dimension higher than three. In this section, we let $d \geq 2$ be any fixed number, and P and Q be two convex d-polytopes with n vertices in total. It is easy to check that Lemma 12 holds for any $d > 3$. As for $d = 3$, the hull event hyperplane Π_f for each facet f of P or Q is defined in an analogous way and the intersection $\mathcal{K} \cap h_f$ defines the hull event horizon h_f. The subdivision $\mathcal{A}$ of $\mathcal{K}$ induced by all the hull event horizons possesses the property of Lemma 14.

One important task is to bound the complexity of the subdivision $\mathcal{A}$.

Lemma 18. *For any fixed* $d \geq 2$*, the complexity of the subdivision* $\mathcal{A}$ *is* $O(n^{\lfloor \frac{d}{2} \rfloor (d-3)+d})$.

Proof. The configuration space $\mathcal{K}$ for dimension d is the boundary of $P \oplus (-Q)$ by Lemma 10. It consists of $O(n^{\lfloor \frac{d+1}{2} \rfloor})$ faces. Further, P and Q have at most $O(n^{\lfloor \frac{d}{2} \rfloor})$ facets (faces of dimension $d-1$). Thus, we have $O(n^{\lfloor \frac{d}{2} \rfloor})$ many hull event horizons.

In order to bound the complexity of the subdivision $\mathcal{A}$, we count the new faces created by the hull event horizons on $\mathcal{K}$. Each of these new faces is an intersection between a face of $\mathcal{K}$ and one or more hull event horizons. For $1 \leq k \leq d-1$, let F_k be the number of those new faces that are intersections of a face of $\mathcal{K}$ and k hull event horizons. Then, we claim that

$$F_k = \begin{cases} O(n^{\lfloor \frac{d+1}{2} \rfloor + k \lfloor \frac{d}{2} \rfloor}), & 1 \leq k \leq d-2 \\ O(n^{(d-1)\lfloor \frac{d}{2} \rfloor}), & k = d-1 \end{cases}.$$

Recall that a hull event horizon is the intersection of a hull event hyperplane and $\mathcal{K}$. That is, F_k counts the new faces of $\mathcal{A}$ that are intersections of a face of $\mathcal{K}$ and k hyperplanes. If $k = d-1$, then the intersection of $k = d-1$ hyperplanes is a 1-flat, which is a line. Since the intersection of a line and the boundary of a convex d-polytope consists of at most two points, we have

$$F_{d-1} = \binom{O(n^{\lfloor \frac{d}{2} \rfloor})}{d-1} = O(n^{(d-1)\lfloor \frac{d}{2} \rfloor}).$$

For $k < d-1$, the intersection of k hyperplanes is a $(d-k)$-flat, and it crosses at most $O(n^{\lfloor \frac{d+1}{2} \rfloor})$ faces of $\mathcal{K}$. This implies that, for any $1 \leq k \leq d-2$,

$$\begin{aligned} F_k &= \binom{O(n^{\lfloor \frac{d}{2} \rfloor})}{k} \cdot O(n^{\lfloor \frac{d+1}{2} \rfloor}) \\ &= O(n^{\lfloor \frac{d+1}{2} \rfloor + k \lfloor \frac{d}{2} \rfloor}), \end{aligned}$$

as claimed.

The complexity of $\mathcal{A}$ is not more than $\sum_{1 \leq k \leq d-1} F_k$, which is bounded by

$$O(n^{\lfloor \frac{d+1}{2} \rfloor + (d-2)\lfloor \frac{d}{2} \rfloor}) = O(n^{\lfloor \frac{d}{2} \rfloor (d-3) + d}).$$

$\square$

Note that the bound for $d = 2$ or 3 in Lemma 18 matches the previously known upper bounds: Lee and Woo [25] for $d = 2$ and the last sections of this paper for $d = 3$.

Our algorithm for $d = 3$ also extends to any fixed dimension $d > 3$. Stage (i) can be done in $O(n^{\lfloor \frac{d+1}{2} \rfloor})$ time, resulting in the configuration space $\mathcal{K}$ of complexity $O(n^{\lfloor \frac{d+1}{2} \rfloor})$ by Lemmas 10 and 11.

For stage (ii), there are $O(n^{\lfloor \frac{d}{2} \rfloor})$ facets of d-polytopes P and Q, and thus the same number of hull event horizons on $\mathcal{K}$. As done for $d = 3$, we compute the subdivision $\mathcal{A}$ of $\mathcal{K}$ by adding the hull event horizons one by one. This can be done in time $O(n^{\lfloor \frac{d}{2} \rfloor (d-3)+d})$ by Lemma 18.

Stage (iii) performs optimization over each facet σ of $\mathcal{A}$ based on the triangulation $\mathcal{T}_\sigma$. In this case, the triangulation $\mathcal{T}_\sigma$ subdivides the boundary of $\mathsf{conv}(P \cup Q_t)$ into $(d-1)$-simplices $\triangle$ (i.e., simplices of dimension $d-1$). For each $\triangle \in \mathcal{T}_\sigma$, we augment one more interior point $c \in P$ to obtain $\triangle^+$ as the d-simplex and thus to triangulate the interior of $\mathsf{conv}(P \cup Q_t)$. Note that the number of $(d-1)$-simplices in $\mathcal{T}_\sigma$ is at most $O(n^{\lfloor \frac{d}{2} \rfloor})$. The d-dimensional volume of a d-simplex is represented by a polynomial of degree d in the coordinates of its vertices, and so is the volume function $\mathrm{vol}(t)$, while the surface area function $\mathrm{surf}(t)$ is represented by a polynomial of degree $d-1$ since it is the sum of $(d-1)$-dimensional volumes of all $\triangle \in \mathcal{T}_\sigma$. By exploiting the coherence among the facets of $\mathcal{A}$, as done for $d = 3$, we can complete stage (iii) in time $O(n^{\lfloor \frac{d}{2} \rfloor (d-3)+d})$.

We conclude the following.

Theorem 3. *For any fixed $d \geq 2$ and two convex d-polytopes P and Q with n vertices in total, a minimum convex container bundling P and Q under translations without overlap can be computed in $O(n^{\lfloor \frac{d}{2} \rfloor (d-3)+d})$ time with respect to volume or surface area.*

3.5 Concluding Remarks

In this chapter, we present the first exact algorithm for the problem bundling two convex polytopes in three dimensional space. Our approach can be extended for the bundling problem for any fixed dimension $d > 3$. Our contributions can be summarized as follows:

- We show that two convex polytopes can be separated in an optimal placement as illustrated in Figure 3.1. Lemma 2 shows convex polygons in the plane can be placed in contact in any optimal placement. It is easy to fall into the trap of extending this lemma in higher dimensional space intuitively. This observation reveals the fact that Lemma 2 cannot be extended in three or higher dimensional space.

- Our algorithm answers an optimal translation for $d = 3$ in (n^3) time. We also show that the algorithm is running in in $\Omega(n^3)$ in the worst case. The algorithm is extended to higher dimensional space and answers an optimal solution in $O(n^{\lfloor \frac{d}{2} \rfloor (d-3)+d})$ time.

IV. Beyond Convexity

In this chapter, given two simple polygons P and Q in the plane, we study the bundling problem of finding a placement φQ of Q such that $P \cup \varphi Q$ is contained in a smallest possible convex region while P and φQ are disjoint in their interiors. We consider minimizing either the area or the perimeter of the convex hull of $P \cup \varphi Q$ under either translations or rigid motions.

4.1 Preliminaries

Throughout this chapter, we use $V(R)$ and $E(R)$ to denote the set of vertices and the set of edges of a polygon R, respectively. In three dimensional space, we use $F(R)$ to denote the set of faces of a polyhedron R. For a compact set S in the plane (or three dimensional space), we use $\mathsf{conv}(S)$ to denote the convex hull of S, and $\|\mathsf{conv}(S)\|$ to denote the area (volume) of $\mathsf{conv}(S)$. Note that we mainly focus on minimizing the area (volume) of the convex hull in the following. The same algorithm works for minimizing the perimeter (surface area) since the perimeter (surface area) function is also piecewise linear under translations, and piecewise trigonometric under rigid motions.

4.2 Minimizing the Convex Hull Under Translations

Let P and Q denote two simple polygons in the plane with n and m $(n \geq m)$ vertices, respectively. Let $Q + \tau$ denote the translated copy of Q by $\tau \in \mathbb{R}^2$, that is, $Q + \tau = \{p + \tau \mid q \in Q\}$. We explain how to find an optimal translation τ^* such that the area $\|\mathsf{conv}(P \cup (Q + \tau^*))\|$ is minimized while P and $Q + \tau^*$ have disjoint in their interiors. It is known that the area function is convex if the polygons are allowed to intersect [5]. We will show how to find an optimal translation using this convexity.

For two simple polygons P and Q, we call an edge of $\mathsf{conv}(P \cup Q)$ a *bridge* if it has an endpoint at a vertex of P and an endpoint at a vertex of Q.

Lemma 19. *Any two disjoint simple polygons in the plane have zero or two bridges in their convex hull.*

Proof. The vertices of $\mathsf{conv}(P \cup Q)$ are from $V(P)$ and $V(Q)$ and each bridge has one endpoint at a vertex of $V(P)$ and the other at a vertex of $V(Q)$. Therefore, there are even number (including 0) of bridges on $\mathsf{conv}(P \cup Q)$.

Assume to the contrary that there are more than two bridges on the convex hull. Let p_i, q_i for $1 \leq i \leq 4$ be four bridges in $\mathsf{conv}(P \cup Q)$ for $p_i \in V(P)$ and $q_i \in V(Q)$. Since P is connected, $\mathsf{conv}(P \cup Q) \setminus P$ consists of at most four connected components. One connected component has at most two q_i's for which $p_i q_i$ is a bridge. Since Q is connected, at least one boundary chain of Q connecting two q_i's crosses P. This implies that the polygons overlap in their interiors. See the Figure 4.1. □

For a pair of a vertex $v \in V(P)$ and an edge $e \in E(Q)$ (or a vertex $v \in V(Q)$ and an edge $e \in E(P)$), let $T_{\mathsf{ve}}(v, e)$ denote the set of all translations τ of P that align v on e. We call such a pair a *vertex-edge pair*. For a pair of a vertex $v \in V(P)$ and an edge $h \in E(\mathsf{conv}(Q)) \setminus E(Q)$ (or a vertex $v \in V(Q)$ and an edge $h \in E(\mathsf{conv}(P)) \setminus E(P)$), let $T_{\mathsf{vh}}(v, h)$ denote the set of all translations τ of P that align v on h. We call such a pair a *vertex-hull-edge pair*. Note that $T_{\mathsf{ve}}(v, e)$ (and $T_{\mathsf{vh}}(v, h)$) is connected and contained in a line in the translation space $\mathbb{R}^2$. If $T_{\mathsf{ve}}(v, e)$ (and $T_{\mathsf{vh}}(v, h)$) is not degenerate, the

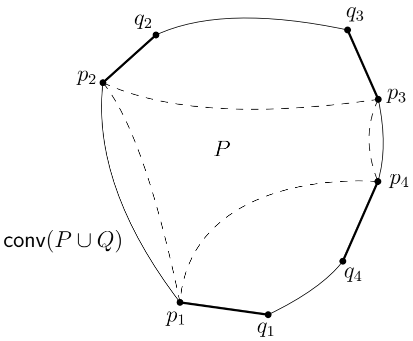


Figure 4.1: Four bridges p_1q_1, p_2q_2 and p_3q_3 and p_4p_4 on the boundary of $\mathsf{conv}(P \cup Q)$. It is impossible to connect the four q_i's in a closed curve in $\mathsf{conv}(P \cup Q)$ without crossing P.

line is uniquely defined. Let $\ell(v, e)$ denote the supporting line of $T_{\mathsf{ve}}(v, e)$. Then $\ell(v, e)$ intersects other supporting lines determined by vertex-edge pairs and vertex-hull-edge pairs. We call the intersection of two supporting lines *a double-contact*.

Lemma 20. *An optimal translation τ^* lies in $T_{\mathsf{ve}}(v, e)$ of a vertex-edge pair (v, e).*

Proof. Assume that there is an optimal translation τ such that P and $Q + \tau$ are disjoint. If there is no bridge in $\mathsf{conv}(P \cup (Q + \tau))$, then either $Q + \tau$ is contained in a connected component of $\mathsf{conv}(Q) \setminus Q$ or Q is contained in a connected component of $\mathsf{conv}(P) \setminus P$. In both cases, we translate $Q + \tau$ along a direction parallel to the convex-hull edge of the component without increasing the area until the two polygons become in contact with a vertex-edge pair.

If there are two bridges in $\mathsf{conv}(P \cup (Q + \tau))$, then we can always translate P in a direction parallel to one of the bridges such that the area of the convex hull decreases, which contradicts to the optimality of τ. □

By using the two lemmas above, we give a characterization of an optimal placement.

Lemma 21. *There is an optimal placement τ^* that lies at an intersection of two supporting lines in the translation space $\mathbb{R}^2$. Moreover, τ^* lies in $T_{\mathsf{ve}}(v, e)$ of a vertex-edge pair (v, e).*

Proof. Consider two disjoint simple polygons P and Q that are in contact at a vertex v of P and an edge e of Q. By Lemma 19, there are zero or two bridges, say $p_l q_l$ and $p_r q_r$ ($p_l, p_r \in V(P)$ and $q_l, q_r \in V(Q)$), in the convex hull of P and Q. Imagine that Q is translated by τ while P and Q are in contact with the vertex-edge pair (v, e). If there is no bridge, the area function $f(\tau) = \|\mathsf{conv}(P \cup (Q + \tau))\|$ is constant. Otherwise, the function is determined by the quadrilateral $p_l p_r q_r q_l$ because the area of each component of $\mathsf{conv}(P \cup (Q + \tau)) \setminus p_l p_r q_r q_l$ remains the same during the translation as long as the combinatorial structure of $\mathsf{conv}(P \cup (Q + \tau))$ remains the same. Note that the area of quadrilateral $p_l p_r q_r q_l$ changes linearly as τ changes. So the area function is non-increasing in one of the two directions parallel to $\ell(v, e)$. Therefore the area is minimized at the intersection of two supporting lines one of which is determined by a vertex-edge pair. □

Based on the lemma above, our algorithm searches, for each supporting line determined by a vertex-edge pair, the intersections with other supporting lines, and finds an intersection realizing the minimum area as follows. For each vertex-edge pair (v, e), we compute the intersections on $\ell(v, e)$ with other supporting lines and sort them along $\ell(v, e)$ in $O(nm \log n)$ time. Then we process the translations τ, each of which is at the intersection of $\ell(v, e)$ and other supporting line, one by one in order and compute

$\|P \cup (Q + \tau)\|$ if P and $Q + \tau$ are disjoint in their interiors.

4.2.1 Disjointness test

We check whether the two polygons are disjoint or not by counting the number c of connected components in the overlap of the polygons for translations of Q along $\ell(v, e)$ for a vertex-edge pair (v, e). We consider the following four types of events that occur during the translation of Q along $\ell(v, e)$. For ease of description, we enumerate the event types with respect to vertices of P in the following. There are another four event types with respect to vertices of Q, and they can be defined analogously.

- Type (a): A convex vertex v of P enters into Q and a new connected component appears in the overlap locally around v. See Figure 4.2(a).

- Type (b): A convex vertex v of P leaves Q and a connected component of the overlap disappears locally around v. See Figure 4.2(b).

- Type (c): A reflex vertex v of P leaves Q and a connected component is subdivided into two connected components in the overlap around v. See Figure 4.2(c).

- Type (d): A reflex vertex v of P enters into Q and two distinct connected components in the overlap are merged into one around v. See Figure 4.2(d).

The counter c is initialized to 0 in the beginning. During the translation along the supporting line determined by a vertex-edge pair, it increases by 1 for each event of type (a) or (c), and decreases by 1 for each event of type (b) or (d). Since the number of connected components in the overlap of P and $Q + \tau$ changes only for events of the four types above, P and $Q + \tau$ are disjoint if and only if $c = 0$.

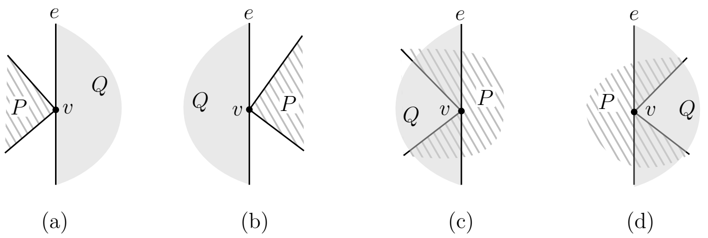
For a vertex-edge pair $T_{\mathsf{ve}}(v, e)$ for any v and e, it may contain a double-contact that aligns the two polygons to be disjoint in their interiors, which we call *a feasible double-contact*. We apply the concept of the convolution of the two polygons [16, 30] to find all feasible double-contacts as follows. A *state* x is a pair of a position $\dot{x}$ and a direction $\vec{x}$. A *move* is a set of states with constant direction and position varying along a line segment parallel to the direction. A *turn* is a set of states with constant position and direction varying along an arc of the circle of directions. A *polygonal trip* is a continuous sequence of moves and turns. A polygonal trip is closed if it starts and ends at the same state. A *polygonal tracing* is a collection of closed polygonal trips.

Given a simple polygon P, we denote by $\hat{P}$ the counter-clockwise tracing of the boundary of P. Then the convolution of $\hat{P}$ and $\hat{Q}$, denoted by $\hat{P} * \hat{Q}$, is defined as follows. If p and q are states in $\hat{P}$ and $\hat{Q}$, respectively, having the same direction $\vec{p} = \vec{q}$, the state $c = (\dot{p} + \dot{q}, \vec{p})$ is a state of the convolution.

Every vertex-edge pair containing a feasible double-contact is covered by the convolution. Therefore, it is sufficient to test the disjointness of the two polygons on the convolution by following lemma.

Lemma 22. *The convolution of two polygons contains all vertex-edge pairs containing a feasible double-contact.*

Proof. The *winding number* of a point x with respect to $\hat{P} * -\hat{Q}$ is the number of connected components in $P \cap (-Q + x)$ [16]. Thus, the Minkowski sum $P \oplus -Q$ is the region of $\hat{P} * -\hat{Q}$ such that the winding number of a point $x \in \hat{P} * -\hat{Q}$ is non-zero. By the definition of a vertex-edge pair, a vertex-edge pair containing no feasible double-contact is a region with non-zero winding number. From this, we know that the convolution $\hat{P} * -\hat{Q}$ covers every vertex-edge pair containing a feasible double-contact. □



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

Figure 4.2: Four types of events with respect to vertices of P that occur during the translation of P to the right.

4.2.2 Evaluation of the area

To evaluate $\|\mathsf{conv}(P \cup (Q+\tau))\|$, we obtain a canonical triangulation T_τ of $\|P \cup \mathsf{conv}((Q+\tau))\|$ as follows. Choose a point c in the interior of Q and connecting c to all the vertices of $\mathsf{conv}(P \cup (Q+\tau))$ by line segments. Then the area of convex hull $\|\mathsf{conv}(P \cup (Q+\tau))\|$ is the sum of the area of $\triangle \in T_\tau$. To evaluate the area, we maintain an area formula for each triangle of T_τ and their sum.

Let τ_1 and τ_2 be two translations corresponding to two adjacent double-contacts on a supporting line of a vertex-edge pair. Assume that we have just processed τ_1 and we are about to process τ_2 and we have the set of area formula for the triangles in T_{τ_1} and the sum of the formula. The translation of Q from $Q+\tau_1$ to $Q+\tau_2$ may cause a change to the convex hull: either an edge is split into two edges or two adjacent edges merge into one. This makes a constant number of changes in T_{τ_1}: a triangle disappears and a new triangle appears. (See Figure 4.3). We compute T_{τ_2} of triangles with their formula and the sum of the formula from T_{τ_1} in $O(1)$ time using $O(n)$ space.

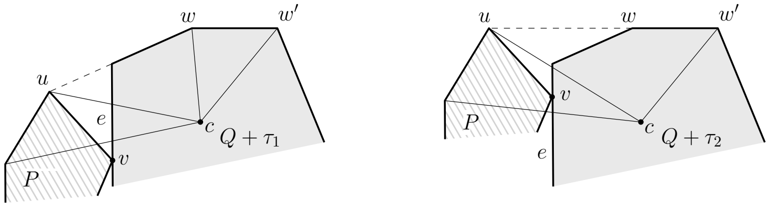


Figure 4.3: When polygon $Q + \tau_1$ moves to $Q + \tau_2$, two triangles, cuw and cww', disappear and new triangle cuw' appears.

As mentioned above, the edges of the convolution of $\hat{P}$ and $\hat{Q}$ cover all vertex-edge pairs containing a feasible double-contact. We can compute the convolution in $O(k^2 \log n)$ time using $O(k)$ space, where $k = O(nm)$ is the size of convolution. [30].

Theorem 4. *Let P and Q be simple polygons with n and m vertices with $n \geq m$, respectively. We can compute a translation $\tau \in \mathbb{R}^2$ that minimizes $\|\textsf{conv}(P \cup (Q+\tau))\|$ satisfying $\textsf{int}(P) \cap \textsf{int}(Q+\tau) = \emptyset$ in $O(n^2 m^2 \log n)$ time using $O(nm)$ space.*

4.2.3 Using linear space

Now we show how to find an optimal translation using only linear space, at the expense of slightly increased time complexity. The key idea is to compute the double-contacts on a supporting line determined by a vertex-edge pair one by one along the line and evaluate the area of the convex hull, instead of computing all the intersections at once.

Finding the next intersection. Again, let $\ell(v, e)$ denote the supporting line determined by a vertex-edge pair (v, e). We compute the double-contacts on $\ell(v, e)$ one by one as follows. We maintain a min-heap H consisting of $O(n)$ double-contacts. Initially, it contains, for each vertex u of P, the first intersection of $\ell(v, e)$ among the supporting lines of (u, e') pairs for all $e' \in E(Q)$. It also contains, for each edge f

of P, the first intersection of $\ell(v, e)$ among the supporting lines of (v', f) pairs for all $v' \in V(Q)$.

Our algorithm gets the first intersection τ (determined by, say (u, f)) from H, determines whether P and $Q + \tau$ are disjoint in their interiors, and computes the area of $\mathsf{conv}(P \cup (Q + \tau))$ if they are disjoint in their interiors. Then it frees the space used for processing the intersection and inserts the next intersection (with respect to u or to f) to H. The intersection can be computed in $O(m)$ time by selection algorithm. Then we check whether a double-contact is in convolution or not by disjointness test when we insert it to the heap, which takes $O(\log n)$ time. Thus, we can search an optimal double-contact in convolution in $O(k^2(m + \log n))$ time.

Theorem 5. *Let P and Q be simple polygons with n and m vertices with $n \geq m$, respectively. We can compute a translation $\tau \in \mathbb{R}^2$ that minimizes $\|\mathsf{conv}(P \cup (Q + \tau))\|$ satisfying $\mathsf{int}(P) \cap \mathsf{int}(Q + \tau)) = \emptyset$ in $O(n^2 m^2 (m + \log n))$ time using $O(n)$ space.*

4.3 Minimizing the Convex Hull Under Rigid Motions

In this section, we allow reorienting Q and show how to find an optimal rigid motion $\rho \in \mathbb{R}^2 \times [0, 2\pi)$ which minimizes $\|\mathsf{conv}(P \cup \rho Q)\|$.

For two vertex-edge pairs (v, e) and (v', e') of $v, v' \in V(P) \cup V(Q)$ and $e, e' \in E(P) \cup E(Q)$, let $t(ve, v'e')$ denote the set of all rigid motions ρ of Q that align v on e and v' on e' simultaneously. Similarly, for a vertex-edge pairs (v, e) and a vertex-hull-edge pair (v', h) of $v, v' \in V(P) \cup V(Q)$, $e \in E(P) \cup E(Q)$, and $h \in E(\mathsf{conv}(P)) \cup E(\mathsf{conv}(Q))$, let $t(ve, v'h)$ denote the set of all rigid motions ρ of Q that align v on e and v' on h simultaneously. Note that $t(ve, v'e')$ and $t(ve, v'h)$ might be empty. Let $\ell(ve, v'e')$ denote the supporting curve defined by a nonempty set $t(ve, v'e')$

in the motion space. See Fig. 4.4. The supporting curve can be represented by a trigonometric function as follows. The supporting curve $\ell(ve, v'e')$ determined by $T_{\textsf{ve}}(v, e)$ and $T_{\textsf{ve}}(v', e')$ can be represented by one of two types of trigonometric functions depending on whether v and v' are from the same polygon or not. Let v_l and v_r be the endpoints of edge e. Similarly, let v'_l and v'_r be the endpoints of edge e'. Let $t = \frac{|v_l v|}{|v_l v_r|}$, where $|uu'|$ is the distance between two points u and u'. Assume that e is parallel to x-axis for the sake of convenience.

Consider the case that v and v' are from the same polygon. We use $x(u)$ and $y(u)$ to denote the x- and y-coordinate of a point u. Then v' has x-coordinate $t + |vv'|\cos(\theta)$ and y-coordinate $|vv'|\sin(\theta)$, where θ is the angle between vv_r and vv'. We can derive $kx(v'_l) + (1-k)x(v'_r) = t + |vv'|\cos(\theta)$ and $ky(v'_l) + (1-k)y(v'_r) = |vv'|\sin(\theta)$, where k is a real, $0 \le k \le 1$. Then, we derive the following formula by dispelling the parameter k,

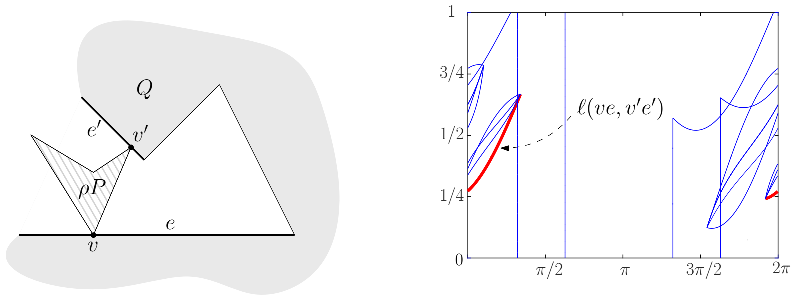
$$t = x(v_r) + \frac{x(v_l) - x(v_r)}{y(v_l) - y(v_r)}(|vv'|\sin(\theta + \theta_v)) - |vv'|\cos(\theta + \theta_v).$$

Next, consider the case that v and v' are from different polygons. Let θ_l and θ_r be the angle between vv_r and vv'_l and the angle between vv_r and vv'_r, respectively. Then v'_l has x-coordinate $t + |vv'_l|\cos(\theta + \theta_l)$ and y-coordinate $|vv'_l|\sin(\theta + \theta_l)$, and v'_r has x-coordinate $t + |vv'_r|\cos(\theta + \theta_r)$ and y-coordinate $|vv'_r|\sin(\theta + \theta_r)$. Then, we derive the following formula similarly to the above case,

$$t = x(v) - \frac{|vv'_l|\cos(\theta + \theta_l) - |vv'_r|\cos(\theta + \theta_r)}{|vv'_l|\sin(\theta + \theta_l) - |vv'_r|\sin(\theta + \theta_r)}(y(v) - |vv'_r|\sin(\theta + \theta_r)).$$

The supporting curve $\ell(ve, v'h)$ corresponding to $t(ve, v'h')$ is also defined analogously.

Our algorithm works similarly to the one in Section 4.2. We compute, for each vertex-edge pair (v, e), the supporting curves with other vertex-edge pairs or vertex-hull-edge pairs. Let C_{ve} denote the set of the supporting curves determined by (v, e). Next, we search, for each supporting curve in C_{ve}, the intersections with other supporting curves in C_{ve}. We do this for each vertex-pair (v, e). Then we evaluate a local optimal on each supporting curve. Note that the area function is also trigonometric and an optimal rigid motion may lie on any place of the supporting curve. (Under translations, we evaluate only the intersections of two supporting lines.) The intersections of two supporting curves may cause a change to the combinatorial structure of convex hull or to the disjointness of two polygons. Thus we update the formula of the area of convex hull and the disjointness at intersections of two supporting lines if necessary.



(a)

(b)

Figure 4.4: (a) A vertex v lies on an edge e and a vertex v' lies on an edge e' simultaneously. (b) Two vertex-edge pairs (v, e) and $(v'e')$ determine the supporting curve $\ell(ve, v'e')$.

The number of vertex-edge pairs is $O(nm)$. For a vertex-edge pair (v, e), the number of supporting curves in C_{ve} is bounded by the number of other vertex-edge pairs or vertex-hull-edge pairs, which is $O(nm)$. Any two supporting curves cross each other at most $O(1)$ times since each curve is represented by a trigonometric function, whose domain is a set $\{x \mid 0 \leq x < 2\pi\}$. Thus, we conclude the following theorem.

Theorem 6. *Let P and Q be simple polygons with n and m vertices with $n \geq m$, re-*

spectively. We can compute a rigid motion ρ that minimizes $\|\mathsf{conv}(P \cup \rho Q)\|$ satisfying $\textsf{\textit{int}}(P) \cap \textsf{\textit{int}}(\rho Q) = \emptyset$ in $O(n^3 m^3 \log n)$ time using $O(nm)$ space.

By using an approach similar to the one in Section 4.2.3, we can find a rigid motion that minimizes the area of the convex hull using only linear space, at the expense of slightly increased time complexity.

Theorem 7. *Let P and Q be simple polygons with n and m vertices with $n \geq m$, respectively. We can compute a rigid motion ρ that minimizes $\|\mathsf{conv}(P \cup \rho Q)\|$ satisfying $\textsf{\textit{int}}(P) \cap \textsf{\textit{int}}(\rho Q) = \emptyset$ in $O(n^3 m^3 (m + \log n))$ time using $O(n)$ space.*

4.4 Extension to Three Dimensional Space

One may think that Lemma 21 extends to three dimensional space naturally. However, this does not work even for two convex polyhedra in 3-dimension. Ahn et al. [1] showed two convex polytopes whose unique optimal translation separates them apart. Thus, for an optimal translation τ^*, P and $Q + \tau^*$ are either apart or in contact. In the case that P and $Q + \tau^*$ are apart, we find the optimal translation by any algorithm for minimizing the volume of their convex hull [3].

Therefore, we focus on the problem where the two polyhedra are supposed to be in contact with each other. There are three types of contact pairs – vertex-facet pairs (v, f), vertex-hull-facet pairs (v, h), and edge-edge pairs (e, e) – each of which defines a set of translations in the 3-dimensional translation space, denoted by $T_{\mathsf{vf}}(v, f)$, $T_{\mathsf{vh}}(v, h)$, $T_{\mathsf{ee}}(e, e')$, respectively. Here, a hull facet is either a facet of $F(\mathsf{conv}(P)) \setminus F(P)$ or a facet of $F(\mathsf{conv}(Q)) \setminus F(Q)$, where $F(R)$ denotes the set of facets of a polyhedron R. Clearly, a nonempty set $T_{\mathsf{vf}}(v, f)$ (or $T_{\mathsf{vh}}(v, h)$) forms a polygon in the space $\mathbb{R}^3$, which is a translate of $f - v$ (or $h - v$) if $v \in V(P)$, or a translate of $-f + v$ (or

$-h + v$) otherwise. A nonempty set $T_{\mathsf{ee}}(e, e')$ forms a parallelogram $e \oplus (-e')$ in the 3-dimensional translation space.

Lemma 23. *If there is an optimal translation that aligns two polytopes to be in contact, it always lies at the intersection of three supporting planes of translation polygons determined by contact pairs. Moreover, one of the three supporting planes must be determined by a vertex-facet or edge-edge pair.*

Proof. It suffices to prove that the volume function $f(\tau) = \|\mathsf{conv}(P \cup (Q + \tau))\|$ is piecewise linear along an arbitrary line. Assume that Q moves along a line parallel to the x-axis, that is, for $t \in \mathbb{R}$, $\tau = Q + (t, 0, 0)$. We will show that the function $\omega(t) = \|\mathsf{conv}(P \cup (Q + (t, 0, 0)))\|$ is piecewise linear. Let Q_k be the intersection of a polyhedron Q and a plane $z = k$. Then ω can be expressed as the sum of $\mathsf{conv}(P \cup (Q + \tau))_k$ over all $k \in \mathbb{R}$. Each of the intersections can be interpreted by the area function of the convex hull of two convex polygons which is piecewise linear [5]. The sum of piecewise linear functions is also piecewise linear, and therefore, ω is piecewise linear. □

Every two supporting planes meet along a line unless they are parallel. Thus, we consider each intersection line of two supporting planes and compute the intersections of the line with other supporting planes. Let $H(v, f)$ denote the supporting plane of $T_{\mathsf{vf}}(v, f)$.

We check whether the two polytopes are disjoint or not similarly to *disjointness test* in Section 4.2.

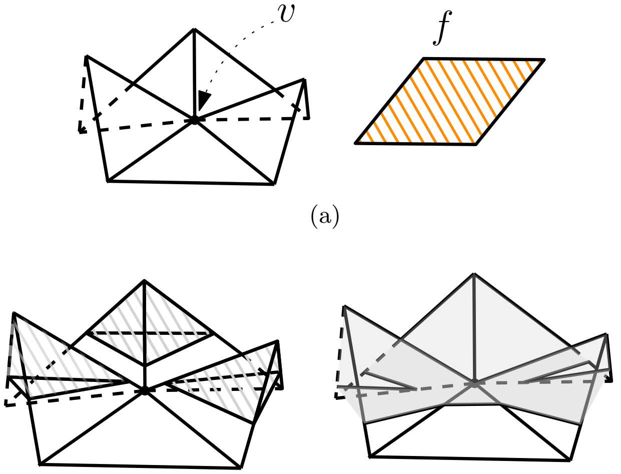
4.4.1 Disjointness test

We maintain the counter c which denotes the number of the edges intersecting the other polyhedron during the translation along the intersection line of two supporting planes. We update the counter c depending on the types of supporting planes we encounter during the translation. No supporting plane $H(v, h)$ of a hull facet h contributes to the counter since h is not part of a polyhedron. Each supporting plane determined by an edge-edge contact may increase or decrease the counter c by 4. Each supporting plane determined by a vertex-facet contact (v, f) may increase or decrease the counter c up to n: some edges incident to v stop intersecting the other polyhedron while some edges start intersecting the other polyhedron. See Figure 4.5. In addition, we check whether one polyhedron is fully contained in the other polyhedron or not by checking whether a vertex v of one polyhedron is contained in the other polyhedron and $c = 0$. We can update whether v is contained in the other polyhedron or not during translation along the intersection line of two supporting planes in $O(1)$ time.

The number of connected components that appear or disappear at $H(v, f)$ is bounded by the degree of v, denoted by $d(v)$. Thus, we can evaluate the disjointness in $O(d(v))$ time per each supporting plane of a vertex-facet pair (v, f), and the total time complexity for the disjointness test along an intersection line is bounded by $O(nm)$, which is the total number of vertex-facet pairs we encounter during translation along the intersection line.

4.4.2 Evaluation of the volume

The volume of $\mathsf{conv}(P \cup (Q + \tau))$ can be evaluated by subdividing it into a set of tetrahedra as follows. First, triangulate each facet of $\mathsf{conv}(P \cup (Q + \tau))$. Let T_τ be



(a)

(b)

Figure 4.5: (a) A vertex v and a face f make the supporting plane $H(v, f)$. (b) Three connected components are merged into one connected component at $H(v, f)$.

the set of those triangles on the boundary $\partial\mathsf{conv}(P \cup (Q + \tau))$. Next, choose a point c in the interior of Q and connect c to every vertices of $\mathsf{conv}(P \cup (Q + \tau))$ with edges. For each triangle $\triangle \in T_\tau$, let $\triangle^+$ be the tetrahedron with base $\triangle$ and apex c. Thus, the volume of $\mathsf{conv}(P \cup (Q + \tau))$ is the sum of $||\triangle^+||$ for $\triangle \in \partial\mathsf{conv}(P \cup (Q + \tau))$. Hence, the volume can be computed in $O(\text{card}(T_\tau)) = O(n)$ time, where $\text{card}(T_\tau)$ is the cardinality of T_τ. We show that evaluating the volume for each intersection is computed in $O(1)$ time by amortized analysis exploiting coherence as follows.

4.4.3 Exploiting coherence

Let τ and τ' be translations corresponding to two adjacent intersections on an intersection line ℓ of two supporting planes. Assume that we have just processed τ and we are about to process τ'. We maintain T_τ, a set of formulas each of which represents the area function $||\triangle^+||$ for a $\triangle \in T_\tau$, and sum of these functions which is the volume formula of $\|\mathsf{conv}(P \cup (Q + \tau))\|$.

Assume that τ' is the intersection of ℓ with a supporting plane determined by a vertex-facet contact (v, f). The translation τ' from τ causes a change to the convex hull of the two polyhedra. We update $T_{\tau'}$ from T_τ and their formulas of $\triangle^+ \in T_{\tau'}$ accordingly as follows. A vertex v of $P + \tau'$ lies on the supporting plane of f so that the triangles defined on f disappear (and their tetrahedra) and the triangles defined by the edges connecting the boundary vertices of f and v appear at τ'. This implies that the number of triangles that disappear or appear at the supporting plane does not exceed the number of boundary vertices of facet f. We update $T_{\tau'}$ by removing all triangles that disappear and adding all triangles that appear and then update the formulas for the volumes. This can be done in $O(N_f)$ time, where N_f denotes the number of boundary vertices of f. The sum of N_f for $f \in F(P) \cup F(Q)$ is bounded by $2 \times (|V(P)| \cdot |F(Q)| + |V(Q)| \cdot |F(P)|)$. Thus, we can evaluate the volume in $O(1)$ time per each intersection.

The number of supporting plane is $O(nm)$. Thus, the number of the intersection lines determined by two supporting planes is $O(n^2m^2)$. Each intersection line intersects $O(nm)$ supporting planes. Thus, we conclude following theorems.

Theorem 8. *Let P and Q be polyhedra with n and m vertices with $n \geq m$, respectively. We can compute a placement $\tau \in \mathbb{R}^3$ that minimizes $\|\mathsf{conv}(P \cup (Q + \tau))\|$ satisfying $\mathsf{int}(P) \cap \mathsf{int}(Q + \tau) = \emptyset$ in $O(n^3m^3 \log n)$ time using $O(nm)$ space.*

Theorem 9. *Let P and Q be polyhedra with n and m vertices with $n \geq m$, respectively. We can compute a placement $\tau \in \mathbb{R}^3$ that minimizes $\|\mathsf{conv}(P \cup (Q + \tau))\|$ satisfying $\mathsf{int}(P) \cap \mathsf{int}(\varphi Q) = \emptyset$ in $O(n^3m^3(m + \log n))$ time using $O(n)$ space.*

4.5 Concluding Remarks

In this chapter, we present algorithms for this problem using much less space than the complexity of the Minkowski sum. When the orientation of Q is fixed, we find an optimal translation of Q in $O(n^2m^2 \log n)$ time using $O(nm)$ space, where n and m ($n \geq m$) denote the number of edges of P and Q, respectively. When we allow reorienting Q, we find an optimal rigid motion of Q in $O(n^3m^3 \log n)$ time using $O(nm)$ space. In both cases, we find an optimal placement of Q using only $O(n)$ space at the expense of slightly increased running time. For three dimensional space and two polyhedra, we find an optimal translation in $O(n^3m^3 \log n)$ time using $O(nm)$ space. We summarize our results in Table 4.1.

Algorithm	Time	Space
Under translations in 2D	$O(n^2m^2 \log n)$	$O(nm)$
	$O(n^2m^2(m + \log n))$	$O(n)$
Under rigid motions in 2D	$O(n^3m^3 \log n)$	$O(nm)$
	$O(n^3m^3(m + \log n))$	$O(n)$
Under translations in 3D	$O(n^3m^3 \log n)$	$O(nm)$
	$O(n^3m^3(m + \log n))$	$O(n)$

Table 4.1: Time and space complexities of our algorithms.

V. Conclusions and Future Works

This thesis introduced algorithms for bundling problem handling various geometric objects: convex polygons, convex d-polytopes and simple polygons.

In Chapter II, I introduced the problem bundling three convex polygons in the plain, and then presented an efficient algorithm for the problem. I proposed an efficient way to represent placements of three convex polygons. I defined events and their corresponding curves in the configuration space from which the combinatorial structure of the convex hull or the motion of the input polygons changes. We also analyzed the complexity of the configuration space. I proposed a $O(n^2)$-time algorithm for this problem in which the objective function for each cell and its description can be updated in constant time and space by using coherence.

In Chapter III, I introduced the problem bundling two convex d-polytopes for any fixed dimension $d \geq 3$ and presented an efficient algorithm for this problem. I observed two convex polytopes can be separated in an optimal placement. I proposed a $O(n^3)$-time algorithm for three dimensional space when two polytopes are in contact. I also showed that our approach to dimension $d = 3$ extends to any fixed dimension higher than three and it answered an optimal solution in $O(n^{\lfloor \frac{d}{2} \rfloor (d-3)+d})$ time.

In Chapter IV, I introduced the problems bundling two simple polygons or two simple polytopes in two or three dimensional space and presented first algorithms for these problems. I considered not only rigid motions of polygons but also translations. I also proposed algorithms for this problems using much less space than the complexity of the Minkowski sum.

I introduced three bundling approaches according to the type of input geometric objects. However so far I have considered a small number of input objects. The problem of how to bunle a number of geometric objects is shown to be NP-hard [11]. One reasonable approach to this problem is to design approximation schemes [35] that improves the efficiency solutions to bundling problems.

요약문

물체들을 효율적으로 배치하는 문제는 다양한 산업에서 뿐만 아니라 일상 생활에서도 매우 중요한 질문 중 하나이다. 특히 물류의 배송 및 보관에 있어서 공간을 효율적으로 사용하는 것은 이윤을 창출하기 위한 매우 중요한 요소로 작용하기 때문에 이러한 문제는 활발하게 연구되고 있다. 계산기하 분야의 발전과 함께 기하 물체 및 이들 간의 상호 관계에 대한 기하학적인 성질이 연구 되고 있으며, 이를 이용하여 물체 배치 문제에 대한 다양한 해결이 제시되고 있다.

본 학위 논문에서는 기하 물체를 효율적으로 배치하는 방법 중 하나인 번들링(Bundling) 문제에 대하여 다룬다. 이전에 주로 연구 되었던 패킹(Packing) 문제에서는 기하 물체를 담는 용기(Container)의 형태 또한 주어지고 이 용기의 크기를 최소화하도록 한다. 이와 달리 번들링 문제에서는 용기의 형태가 미리 정의되어 있지 않다. 최적의 물체 배치에 따라 용기의 형태가 결정되기 때문에 보다 효율적으로 용기를 선택 할 수 있다는 장점이 있다.

번들링 문제는 주어진 기하 물체를 포함 하는 가장 작은 부피의 볼록 헐(Convex hull)을 구하는 문제이다. 우리는 기하 물체의 1) 개수, 2) 허용되는 운동(Transformation), 3) 종류라는 세 가지 관점을 통해 번들링 문제에 접근한다. 기하 물체들이 가지는 다양한 기하학적 성질을 규명하고 이를 바탕으로 효율적인 알고리즘을 제시한다.

References

[1] H. Ahn, S. W. Bae, O. Cheong, D. Park, and C. Shin. Minimum convex container of two convex polytopes under translations. In *Proc. 26th Canadian Conf. Comput. Geom. (CCCG 2014)*, 2014.

[2] H.-K. Ahn, J. Abardia, S. W. Bae, O. Cheong, S. Dann, D. Park, and C.-S. Shin. The minimum convex container of two convex polytopes under translations. Submitted manuscript.

[3] H.-K. Ahn, P. Brass, and C.-S. Shin. Maximum overlap and minimum convex hull of two convex polyhedra under translations. *Computational Geometry*, 40(2):171–177, 2008.

[4] H.-K. Ahn and O. Cheong. Stacking and bundling two convex polygons. In *Proc. 16th Internat. Sympos. Algo. Comput. (ISAAC 2005)*, volume 3827 of *LNCS*, pages 40–49, 2005.

[5] H.-K. Ahn and O. Cheong. Aligning two convex figures to minimize area or perimeter. *Algorithmica*, 62:464–479, 2012.

[6] H. Alt and L. J. Guibas. Discrete geometric shapes: Matching, interpolation, and approximation. *Handbook of computational geometry*, 1:121–153, 1999.

[7] H. Alt and F. Hurtado. Packing convex polygons into rectangular boxes. In *Proc. 3rd Japanese Conf. Discrete Comput. Geom. (JCDCG 2000)*, volume 2098 of *LNCS*, pages 67–80, 2001.

[8] J. Barraquand and J.-C. Latombe. Robot motion planning: A distributed representation approach. *The International Journal of Robotics Research*, 10(6):628–649, 1991.

[9] B. Chazelle. An optimal convex hull algorithm in any fixed dimension. *Discrete & Computational Geometry*, 10:377–409, 1993.

[10] J. Choi, D. Park, and H.-K. Ahn. Bundling two simple polygons to minimize their convex hull. To appear in Proc. The 11th International Conference and Workshops on Algorithms and Computation (WALCOM 2017), 2017.

[11] K. Daniels and V. Milenkovic. Multiple translational containment, part i: An approximation algorithm. *Algorithmica*, 19:148–182, 1997.

[12] M. de Berg, O. Cheong, M. van Kreveld, and M. Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, 3rd edition, 2008.

[13] J. Egeblad, B. Nielsen, and M. Brazil. Translational packing of arbitrary polytopes. *Comput. Geom.: Theory and Appl.*, 42:269–288, 2009.

[14] G. Elber and M.-S. Kim. Offsets, sweeps, and minkowski sums. *Computer-Aided Design*, 31(3):163, 1999.

[15] M. Garey and D. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W.H. Freeman and Co., San Francisco, CA, 1979.

[16] L. Guibas, L. Ramshaw, and J. Stolfi. A kinetic framework for computational geometry. In *Proceedings of the 24th Annual Symposium on Foundations of Computer Science (FOCS 1983)*, pages 100–111. IEEE, 1983.

[17] T. C. Hales. The kepler conjecture. *arXiv preprint math.MG/9811078*, 1998.

[18] B. Huber, J. Rambau, and F. Santos. The Cayley trick, lifting subdivisions and the Bohne–Dress theorem on zonotopal tilings. *J. Eur. Math. Soc.*, 2:179–198, 2000.

[19] M. I. Karavelas, C. Konaxis, and E. Tzanaki. The maximum number of faces of the minkowski sum of three convex polytopes. In *Proceedings of the twenty-ninth annual symposium on Computational geometry*, pages 187–196. ACM, 2013.

[20] M. I. Karavelas and E. Tzanaki. The maximum number of faces of the minkowski sum of two convex polytopes. In *Proceedings of the twenty-third annual ACM-SIAM Symposium on Discrete Algorithms*, pages 11–28. SIAM, 2012.

[21] A. Kaul, M. A. O'Connor, and V. Srinivasan. Computing minkowski sums of regular polygons. In *Proceedings of the 3rd Canadian Conference on Computational Geometry (CCCG 1991)*, pages 74–77, 1991.

[22] K. Kedem, R. Livne, J. Pach, and M. Sharir. On the union of jordan regions and collision-free translational motion amidst polygonal obstacles. *Discrete & Computational Geometry*, 1(1):59–71, 1986.

[23] J. Kepler. *Vom sechseckigen Schnee*, volume 273 of *Ostwalds Klassiker der Exakten Wissenschaften*. Akademische Verlagsgesellschaft Geest & Portig K.-G., Leipzig, 1987. Strena seu de Nive sexangula, Translated from the Latin and with an introduction and notes by Dorothea Goetz.

[24] D. A. Klain and G.-C. Rota. *Introduction to Geometric Probability*. Cambridge University Press, 1997.

[25] H. Lee and T. Woo. Determining in linear time the minimum area convex hull of two polygons. *IIE Trans.*, 20:338–345, 1988.

[26] D. Leven and M. Sharir. Planning a purely translational motion for a convex object in two-dimensional space using generalized voronoi diagrams. *Discrete & Computational Geometry*, 2(1):9–31, 1987.

[27] V. Milenkovic. Translational polygon containment and minimum enclosure using linear programming based restriction. In *Proc. 28th Annu. ACM Sympos. Theory Comput. (STOC 1996)*, pages 109–118, 1996.

[28] A. Paoluzzi, V. Pascucci, M. Vicentino, C. Baldazzi, and S. Portuesi. *Geometric programming for computer aided design*. J. Wiley, 2003.

[29] D. Park, S. W. Bae, H. Alt, and H.-K. Ahn. Bundling three convex polygons to minimize area or perimeter. *Computational Geometry*, 51:1–14, 2016.

[30] G. Ramkumar. An algorithm to compute the minkowski sum outer-face of two simple polygons. In *Proceedings of the twelfth annual symposium on Computational geometry*, pages 234–241. ACM, 1996.

[31] R. Seidel. The upper bound theorem for polytopes: an easy proof of its asymptotic version. *Comput. Geom.: Theory and Appl.*, 5:115–116, 1995.

[32] M. Sharir. Algorithmic motion planning in robotics. *Computer*, 22(3):9–19, 1989.

[33] K. Sugihara, M. Sawai, H. Sano, D.-S. Kim, and D. Kim. Disk packing for the estimation of the size of a wire bundle. *Japan J. Industrial and Applied Math.*, 21:259–278, 2004.

[34] K. Tang, C. Wang, and D. Chen. Minimum area convex packing of two convex polygons. *Internat. J. Comput. Geom. Appl.*, 16:41–74, 2006.

[35] V. V. Vazirani. *Approximation algorithms.* Springer Science & Business Media, 2013.

[36] R. C. Veltkamp. Shape matching: similarity measures and algorithms. In *Shape Modeling and Applications, SMI 2001 International Conference on.*, pages 188–197. IEEE, 2001.

Acknowledgements

먼저 부족한 저를 6년간 포기하지 않고 지도해주신 안희갑 교수님께 감사의 말씀을 드리고 싶습니다. 교수님으로부터 연구하는 모습뿐 아니라, 항상 배려심있고 긍정적인 에너지를 주시는 인간적인 모습에서 많은 것을 배운 것 같습니다. 멀리서 저의 학위 심사에 와주신 신찬수 교수님, Antoine 교수님, 그리고 배상원 교수님 감사 드립니다. 바쁘신 와중에 저의 학위 심사에 참석해 주신 박성우 교수님, 배경민 교수님께도 감사 드립니다. 제가 힘들어 할 때마다 저의 얘기를 들어준 완빈이형, 상섭이형에게 감사의 말을 전하고 싶습니다. 듬직한 형들 덕분에 저도 무사히 졸업할 수 있게 된 것 같습니다. 그리고 많은 프로젝트를 함께하면서, 맡은 일을 책임감 있게 진행하는 윤호형의 모습에서 많은 것을 배웠습니다. 항상 해맑은 웃음을 가진 효실누나, 길지 않은 시간 같이 있었지만 오랜 연구실 식구 같았던 혜선이, 부족한 나를 잘 따라 준 후배 상덕이, 연구실 활력소 은진이, 랩장한다고 고생한 민규, 연구실 구석구석 꼼꼼히 챙기는 성환이, 연애만큼 연구도 열정적인 진영이, 그리고 이제 연구실을 이끌어갈 샛별들 민철, 준수, 종민 다들 고맙다.

제가 대학원 간다고 했을 때, 저의 의견을 지지해준 아버지, 어머니 항상 감사드리고 존경합니다. 제가 표현이 서툴러 사랑한다는 말을 지금껏 전하지 못하였는데 이 자리를 빌려 말하고 싶습니다. 아버지, 어머니 진심으로 사랑합니다. 그리고 만날 때마다 하나라도 더 챙겨주려는 누나, 매형 감사합니다. 항상 웃음으로 날 맞이해주는 우리 귀여운 조카 예준이, 예성이 건강하고 훌륭하게 자라다오. 그리고 마지막으로 항상 배려하는 모습으로 나에게 진정한 사랑이 뭔지 일깨워준 여자친구 진영아 사랑한다. 앞으로도 알콩달콩 행복하게 지내자.