Part III

Computing the Hausdorff distance between polyhedral surfaces in $\mathbb{R}^3$

Many real world applications like, e.g., molecular docking, CAD/CAM, terrain comparison for GIS, etc., give rise to geometric pattern matching problems in three-dimensional space. In most of these applications the patterns are modeled by triangulated simple (i.e., not self-intersecting) polyhedral surfaces; such surfaces can be seen as sets of triangles with the additional property that any two distinct triangles in the set do not intersect in their relative interiors.

Definition 6.1 (Δ-pattern). *A finite set P of triangles in $\mathbb{R}^3$ with the property, that any two distinct triangles in P do not intersect in their relative interiors will be called a* Δ-pattern. *The* size *of the Δ-pattern P is the number of triangles in the set P.*

In this part the similarity of Δ-patterns will be assessed with the aid of the Hausdorff distance δ_H (c.f., Definition 1.4 on page 7) and a generalization thereof, the so-called $L_{\mathbf{B}}$-Hausdorff distance which is defined with respect to a centrally symmetric convex body $\mathbf{B} \subseteq \mathbb{R}^3$.

Recall that such a body $\mathbf{B}$ defines a metric on $\mathbb{R}^3$ in the following way:

Definition 6.2 ($L_{\mathbf{B}}$-distance, $L_{\mathbf{B}}$-norm). *Let x and y be two points in $\mathbb{R}^3$. Then $\mathrm{d}_{L_{\mathbf{B}}}(x,y)$ denotes the* $L_{\mathbf{B}}$-distance *between x and y,*

$$\mathrm{d}_{L_{\mathbf{B}}}(x,y) := ||x-y||_{L_{\mathbf{B}}},$$

with $||z||_{L_{\mathbf{B}}}$ being the $L_{\mathbf{B}}$-norm *of $z \in \mathbb{R}^3$,*

$$||z||_{L_{\mathbf{B}}} := \mathsf{min}\{c \geq 0 \mid z \in c \cdot \mathbf{B}\}.$$

By modifying the Definition of δ_H (c.f., Definition 1.4) accordingly to use the $L_{\mathbf{B}}$-distance instead of the Euclidean distance to measure the distance between points in $\mathbb{R}^3$ we can define a distance measure that constitutes a generalization of the Hausdorff distance δ_H:

Definition 6.3 ($L_{\mathbf{B}}$-Hausdorff distance, one-sided $L_{\mathbf{B}}$-Hausdorff distance). *Let P and Q be compact sets in $\mathbb{R}^3$. Then $\delta_H^{\mathbf{B}}(P,Q)$ denotes the* $L_{\mathbf{B}}$-Hausdorff distance *between P and Q, defined as*

$$\delta_H^{\mathbf{B}}(P,Q) := \mathsf{max}\left(\tilde{\delta}_H^{\mathbf{B}}(P,Q), \tilde{\delta}_H^{\mathbf{B}}(Q,P)\right), \text{ with}$$
$$\tilde{\delta}_H^{\mathbf{B}}(P,Q) := \sup_{x\in P} \inf_{y \in Q} \mathrm{d}_{L_{\mathbf{B}}}(x,y), \text{ the one-sided } L_{\mathbf{B}}\text{-Hausdorff distance from } P \text{ to } Q.$$

Variants of the Hausdorff distance that are frequently considered for comparing geometric patterns are obtained by choosing $\mathbf{B} = \mathbf{B}_1 = \{(x,y,z) \in \mathbb{R}^3 \mid |x|+|y|+|z| \leq 1\}$, or $\mathbf{B} = \mathbf{B}_\infty = \{(x,y,z) \in \mathbb{R}^3 \mid \mathsf{max}(|x|,|y|,|z|) \leq 1\}$, the unit balls of the L_1 and L_∞ metric, respectively. For $\mathbf{B} = \mathbf{B}_2 = \{(x,y,z) \in \mathbb{R}^3 \mid x^2+y^2+z^2 \leq 1\}$ (the three-dimensional unit ball) the resulting distance is the Hausdorff distance δ_H.

In the following we give an algorithm that computes $\delta_H(P,Q)$ in $\mathcal{O}_\epsilon((mn)^{15/16+\epsilon}(m^{17/16}+n^{17/16}))$ randomized time for given Δ-patterns P and Q of size m and n, respectively (c.f., Theorem 6.19 on page 68). We also present an algorithm that decides whether the Hausdorff distance with respect to a polyhedral distance function between P and Q is at most δ in $\mathcal{O}_\epsilon((m+n)^{2+\epsilon} + mn(\sqrt{m^{1+\epsilon}} + \sqrt{n^{1+\epsilon}}))$ randomized time (c.f., Theorem 6.12 on page 66).

Some of the material has already been published in [11].

In the following we develop efficient algorithms for the measuring problem for Δ-patterns in $\mathbb{R}^3$ with respect to the Hausdorff distance.

Problem 6.4 (Δ-pattern δ_H-measure problem – optimization version).
Given *two Δ-patterns $P, Q \subseteq \mathbb{R}^3$.*
Compute *$\delta_H(P, Q)$.*

In the course of that we will first devise efficient methods to tackle the corresponding decision problem.

Problem 6.5 (Δ-pattern δ_H-measure problem – decision version).
Given *two Δ-patterns $P, Q \subseteq \mathbb{R}^3$, and some $\delta > 0$.*
Decide*, whether $\delta_H(P, Q) \leq \delta$.*

Godau [35] gives a randomized algorithm that computes $\delta_H(P, Q)$ in $\mathcal{O}_\epsilon(mn(n^{1+\epsilon} + m^{1+\epsilon}))$ expected time, where m and n denote the size of P and Q, respectively; this algorithm is cubic in the diagonal case, where $m = \Theta(n)$.

In section 6.3 below, we improve upon these results and give an algorithm that computes $\delta_H(P, Q)$ in $\mathcal{O}_\epsilon((mn)^{15/16+\epsilon}(m^{17/16} + n^{17/16}))$ randomized expected time (c.f., Theorem 6.19 on page 68); in the diagonal case this becomes $\mathcal{O}_\epsilon(n^{47/16+\epsilon}) \approx \mathcal{O}(n^{2,9375})$ and constitutes the first subcubic solution for Problem 6.4. For variants of the Hausdorff distance, which we get when using a polyhedral distance function instead of the Euclidean distance to measure the distance between points in $\mathbb{R}^3$, we improve our results in section 6.2 and obtain an algorithm that solves the decision problem in $\mathcal{O}_\epsilon((m+n)^{2+\epsilon} + mn(\sqrt{m^{1+\epsilon}} + \sqrt{n^{1+\epsilon}}))$ time (c.f., Theorem 6.12 on page 66); in the diagonal case, where $m = \Theta(n)$, this becomes $\mathcal{O}_\epsilon(n^{5/2+\epsilon}) \approx \mathcal{O}(n^{2,5})$. Again the algorithm of Godau was the most efficient procedure known thus far for that problem.

For the remainder of this chapter, P and Q are Δ-patterns of size m and n, respectively, and $\delta > 0$ is fixed. As before, P^0 will denote the set of vertices of P.

6.1 Outline of the method

The basic idea is best illustrated if we consider the problem in a somewhat simplified setting. To this end we make use of a different distance function, which we get when using a polyhedral distance function instead of the Euclidean distance to measure the distance between points in $\mathbb{R}^3$. For the remainder of this chapter, $\mathbf{B}$ will be a centrally symmetric convex body, $\mathbf{B}_P$ will be a centrally symmetric convex polyhedron and $\mathbf{B}_2 = \{(x, y, z) \in \mathbb{R}^3 \mid x^2 + y^2 + z^2 \leq 1\}$ will be the three-dimensional unit ball.

We have that $\tilde{\delta}_H^{\mathbf{B}}(P, Q) \leq \delta$ iff for each point of P there is a point of Q that is δ-close (wrt the $L_{\mathbf{B}}$-distance). Therefore it makes sense to look at the set of all points that are δ-close to Q:

Definition 6.6 ($L_{\mathbf{B}}$-δ-neighborhood, $L_{\mathbf{B}}$-δ-boundary). *Let Q be a compact set in $\mathbb{R}^3$. Then $\mathsf{nh}_\delta^{\mathbf{B}}(Q)$ denotes the $L_{\mathbf{B}}$-δ-*neighborhood *of Q, defined as*

$$\mathsf{nh}_\delta^{\mathbf{B}}(Q) := \{x \in \mathbb{R}^3 \mid \mathrm{d}_{L_{\mathbf{B}}}(x, Q) \leq \delta\} = Q \oplus (\delta \cdot \mathbf{B}), \text{ and}$$

$\mathsf{bd}_\delta^{\mathbf{B}}(Q)$ denotes the boundary of the $L_{\mathbf{B}}$-δ-neighborhood *of Q, i.e.,*

$$\mathsf{bd}_\delta^{\mathbf{B}}(Q) := \{x \in \mathbb{R}^3 \mid \mathrm{d}_{L_{\mathbf{B}}}(x, Q) = \delta\}.$$

Our results are based on the following simple observation that is subsumed in Lemma 6.7 below: The one-sided $L_{\mathbf{B}}$-Hausdorff distance from P to Q is at most δ iff all vertices of P are contained in the $L_{\mathbf{B}}$-δ-neighborhood of Q and none of the triangles in P intersects the boundary $\mathsf{bd}_\delta^{\mathbf{B}}(Q)$.

Lemma 6.7 (The connection between $\tilde{\delta}_H^{\mathbf{B}}$ and $\mathsf{bd}_\delta^{\mathbf{B}}$). *Let and P, Q be Δ-patterns in $\mathbb{R}^3$, and $\delta > 0$. Then*

$$\tilde{\delta}_H^{\mathbf{B}}(P, Q) < \delta \iff P \subset \mathsf{nh}_\delta^{\mathbf{B}}(Q) \iff P^0 \subset \mathsf{nh}_\delta^{\mathbf{B}}(Q) \text{ and } P \cap \mathsf{bd}_\delta^{\mathbf{B}}(Q) = \emptyset.$$

So we are left with the task of verifying whether $P^0 \subset \mathsf{nh}_\delta^{\mathbf{B}}(Q)$ ('inclusion property') and $P \cap \mathsf{bd}_\delta^{\mathbf{B}}(Q) = \emptyset$ ('intersection property') hold. The first property can be checked in $\mathcal{O}(mn)$ steps by computing the distance of each vertex in P^0 to each triangle in Q in $\mathcal{O}(1)$ time. Note that this method can also compute the triangles $\Delta \in P$ that contain a vertex outside of $\mathsf{nh}_\delta^{\mathbf{B}}(Q)$.

Lemma 6.8 (Checking the inclusion property). *For $\mathbf{B} \in \{\mathbf{B}_P, \mathbf{B}_2\}$ we can decide whether $P^0 \subset \mathsf{nh}_\delta^{\mathbf{B}}(Q)$ in $\mathcal{O}(mn)$ time.*

In the following two sections we will describe efficient algorithms to verify the second property when $\mathbf{B}$ is a convex polyhedron or a ball. The basic approach will be the same in both cases, but the details (and the runtime) will differ marginally.

6.2 The $L_{\mathbf{B}_P}$-Hausdorff distance of Δ-patterns

For a triangle $\Delta \in Q$, the set $\mathsf{nh}_\delta^{\mathbf{B}_P}(\Delta)$ is the convex hull of three copies of $\delta \cdot \mathbf{B}_P$ centered at the vertices of Δ. Its boundary is a convex polyhedron and, since $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ is contained in the union of these boundaries, it is a polyhedral set, i.e., it is the union of a finite set of trinagles that do not intersect in their relative interiors.

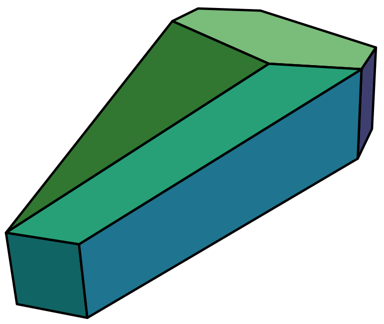

Figure 6.1: The $L_{\mathbf{B}_1}$-δ-neighborhood of a triangle in $\mathbb{R}^3$.

The complexity of $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ is the number of its vertices, edges, and 2-faces.

Theorem 6.9 (Computation of $\mathsf{bd}_\delta^{\mathbf{B}_P}$*, Aronov/Sharir, [19, 20]).* The boundary $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ has complexity $\mathcal{O}(n^2)$ and can be computed in $\mathcal{O}(n^2 \log^2 n)$ randomized expected time.

The algorithm of Aronov/Sharir computes a description of $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ where each 2-face is partitioned into triangles. In order to verify the intersection property we need a method to detect triangle-triangle intersections:

Theorem 6.10 (Triangle intersection queries for $\mathsf{bd}_\delta^{\mathbf{B}_P}$*, Pellegrini, [42]).* Let T be a set of k triangles in $\mathbb{R}^3$ with disjoint interiors. We can build a data structure of size $\mathcal{O}_\epsilon(k^{4+\epsilon})$ in $\mathcal{O}_\epsilon(k^{4+\epsilon})$ time, such that for any query triangle Δ we can decide in $\mathcal{O}_\epsilon(k^\epsilon)$ time if Δ intersects T.

Lemma 6.11 (Checking the intersection property – polyhedral case). *We can decide whether $P \cap \mathsf{bd}_\delta^{\mathbf{B}_P}(Q) = \emptyset$ in $\mathcal{O}_\epsilon(n^{2+\epsilon} + mn^{3/2+\epsilon})$ randomized expected time.*

Proof. In a first step we compute a description of $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ with the algorithm from Theorem 6.9. This can be done in $\mathcal{O}(n^2 \log^2 n)$ time and yields a set of $\mathcal{O}(n^2)$ triangles that partition the boundary of $\mathsf{nh}_\delta^{\mathbf{B}_P}(Q)$. Now we distinguish two cases:

$m^2 \leq n$: We run the algorithm of Theorem 6.10 to build a data structure of size $\mathcal{O}_\epsilon(m^{4+\epsilon})$ in $\mathcal{O}_\epsilon(m^{4+\epsilon})$ time that supports triangle intersection queries to P in $\mathcal{O}_\epsilon(m^\epsilon)$ time and then we query this data structure with all triangles in $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ to test for intersections in $\mathcal{O}_\epsilon(n^2 m^\epsilon)$ steps. The total time spent is

$$\mathcal{O}_\epsilon(m^{4+\epsilon} + n^2 m^\epsilon) = \mathcal{O}_\epsilon(n^{2+\epsilon} + n^2 m^\epsilon) = \mathcal{O}_\epsilon(n^{2+\epsilon}).$$

$n \leq m^2$: We partition P into $g = \lceil m/n^{1/2} \rceil$ groups of $k = n^{1/2} \leq m$ triangles each. For each group, we run the algorithm of Theorem 6.10 to build a data structure of size $\mathcal{O}_\epsilon(k^{4+\epsilon})$ in $\mathcal{O}_\epsilon(k^{4+\epsilon})$ time that supports triangle intersection queries in $\mathcal{O}_\epsilon(k^\epsilon)$ time and then we query this data structure with all triangles in $\mathsf{bd}_\delta^{\mathbf{B}_P}(Q)$ to test for intersections in $\mathcal{O}_\epsilon(n^2 k^\epsilon)$ steps. The total time spent is

$$\mathcal{O}_\epsilon(g(k^{4+\epsilon} + n^2 k^\epsilon)) = \mathcal{O}_\epsilon(g n^{2+\epsilon/2}) = \mathcal{O}_\epsilon(mn^{3/2+\epsilon}).$$

□

By applying Lemmas 6.8 and 6.11 twice we get the main result of this section:

Theorem 6.12 (Δ-pattern $\delta_H^{\mathbf{B}_P}$-measure problem – decision version). *We can decide whether $\delta_H^{\mathbf{B}_P}(P,Q) \le \delta$ in $\mathcal{O}_\epsilon((m+n)^{2+\epsilon} + mn(\sqrt{m^{1+\epsilon}} + \sqrt{n^{1+\epsilon}}))$ randomized expected time.*

6.3 The Hausdorff distance of Δ-patterns

In the Euclidean case we follow the same basic approach as in the previous section. However, since the geometric structure of the boundary $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ is more complicated, we have to resort to data structures that do not match the efficiency of the ones we used in the polyhedral case. Therefore the runtime of the algorithm will be slightly worse.

For a triangle Δ, the set $\mathsf{nh}_\delta^{\mathbf{B}_2}(\Delta)$ is the convex hull of three copies of a δ-ball centered at the vertices of Δ; it is the (non-disjoint) union of three balls of radius δ around the vertices of Δ, three cylinders of radius δ around the edges of Δ, and a triangular prism of height 2δ around Δ. The complexity of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ is the number of vertices, edges, and 2-faces

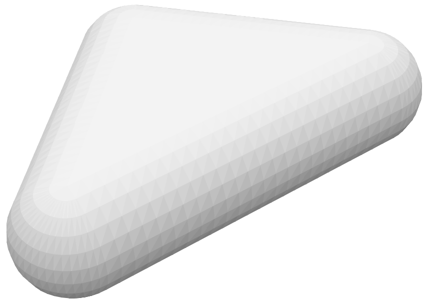

Figure 6.2: The $L_{\mathbf{B}_2}$-δ-neighborhood of a triangle in $\mathbb{R}^3$.

of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$. A 2-face of the boundary is a maximal connected closed subset of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$, contained in one spherical, cylindrical, or triangular portion of $\mathsf{bd}_\delta^{\mathbf{B}_2}(\Delta)$, for some $\Delta \in Q$. An edge of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ is a maximal connected subset of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ contained in two 2-faces and a vertex of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ is contained in three 2-faces.

Theorem 6.13 (Computation of $\mathsf{bd}_\delta^{\mathbf{B}_2}$, Agarwal/Sharir, [3, 4]). The boundary $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ has complexity $\mathcal{O}_\epsilon(n^{2+\epsilon})$ and can be computed in $\mathcal{O}_\epsilon(n^{2+\epsilon})$ randomized expected time.

The algorithm of Agarwal/Sharir computes a description of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ where each 2-face is partitioned into semialgebraic surface patches of constant description complexity. Each of these surface patches is contained in one spherical, cylindrical, or triangular portion of $\mathsf{bd}_\delta^{\mathbf{B}_2}(\Delta)$ for some $\Delta \in Q$ (the same Δ that contains the corresponding 2-face) and is bounded by at most four arcs. Each arc in turn is part of the intersection of the portion of the boundary that contains the patch with either a plane, or an δ-sphere, or an δ-cylinder. A polynomial expression defining a patch is formed by the conjunction of five atomic expressions of degree at most two: one polynomial equation describing the portion of $\mathsf{bd}_\delta^{\mathbf{B}_2}(\Delta)$ that contains the patch (i.e., a cylinder, a sphere, or a plane) and at most four polynomial inequalities defining the arcs (again these are equations describing a cylinder, a sphere, or a plane).

In order to verify the intersection property we need a method to detect intersections between the triangles in P and the surface patches of $\mathsf{bd}_\delta^{\mathbf{B}_2}$. We will apply a standard approach suggested in [27] and [25] and transform this problem to a semialgebraic point-location problem.

Lemma 6.14 (Triangle intersection queries for $\mathsf{bd}_\delta^{\mathbf{B}_2}$). *Let Ω be a set of k semialgebraic sets of constant description complexity in $\mathbb{R}^3$. We can build a data structure of size $\mathcal{O}_\epsilon(k^{16+\epsilon})$ in $\mathcal{O}_\epsilon(k^{16+\epsilon})$ randomized expected time, such that for any query triangle Δ we can decide in $\mathcal{O}_\epsilon(k^\epsilon)$ time if Δ intersects Ω.*

Proof. Let $\Delta(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}; \mathbf{x})$ be a polynomial expression that defines a triangle Δ depending on its three vertices $\mathbf{p_1}, \mathbf{p_2}$, and $\mathbf{p_3}$, i.e., $\Delta = \{\mathbf{x} \in \mathbb{R}^3 \mid \Delta(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}; \mathbf{x}) \text{ holds}\}$; we can form Δ as the conjunction of three linear inequalities and one linear equation. Let $\Gamma(\mathbf{x})$ be a polynomial expression that defines a set $\Gamma \in \Omega$, i.e., $\Gamma = \{\mathbf{x} \in \mathbb{R}^3 \mid \Gamma(\mathbf{x}) \text{ holds}\}$. For some fixed Γ, consider the set $C_\Gamma = \{(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}) \in \mathbb{R}^9 \mid (\exists \mathbf{x} : \Delta(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3}; \mathbf{x}) \wedge \Gamma(\mathbf{x})) \text{ holds}\}$. If we look at $\mathbb{R}^9$ as the configuration space of the set of all triangles in 3-space, then C_Γ is the set of (the parameters of) all triangles that intersect Γ. By quantifier elimination [30] we can find a polynomial expression $C_\Gamma(\mathbf{p_1}, \mathbf{p_2}, \mathbf{p_3})$ that defines C_Γ; therefore this set is semialgebraic, too.

Let $F = \{f_\Gamma^1, \dots, f_\Gamma^l \mid \Gamma \in \Omega\}$ denote the set of $\mathcal{O}(k)$ many polynomials that appear in the atomic polynomial expressions forming the expressions C_Γ. With the algorithm of Theorem 3.2 we can compute a point-location data structure of size $\mathcal{O}_\epsilon(k^{16+\epsilon})$ in $\mathcal{O}_\epsilon(k^{16+\epsilon})$ time for the arrangement of the varieties $f_\Gamma^i = 0$ defined by F. Since the signs of all polynomials in F and therefore the validity of each polynomial expression C_Γ is constant for each cell of the decomposition of $\mathbb{R}^9$ induced by these varieties, the claim follows. $\square$

Lemma 6.15 (Checking the intersection property – Euclidean case). *We can decide whether $P \cap \mathsf{bd}_\delta^{\mathbf{B}_2}(Q) = \emptyset$ in $\mathcal{O}_\epsilon(mn^\epsilon + n^{2+\epsilon}m^{15/16+\epsilon})$ randomized expected time.*

Proof. In a first step we compute a description of $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ with the algorithm from Theorem 6.13. This can be done in $\mathcal{O}_\epsilon(n^{2+\epsilon})$ time and yields a set of $\mathcal{O}_\epsilon(n^{2+\epsilon})$ semialgebraic surface patches of constant description complexity that partition the boundary of $\mathsf{nh}_\delta^{\mathbf{B}_2}(Q)$. Now we distinguish two cases:

$n^{32} \leq m$: We run the algorithm of Lemma 6.14 to build a data structure of size $\mathcal{O}_\epsilon(n^{32+\epsilon})$ in $\mathcal{O}_\epsilon(n^{32+\epsilon})$ time that supports triangle intersection queries to $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ in $\mathcal{O}_\epsilon(n^\epsilon)$ time and then we query this data structure with all triangles in P to test for intersections in $\mathcal{O}_\epsilon(mn^\epsilon)$ steps. The total time spent is

$$\mathcal{O}_\epsilon(n^{32+\epsilon} + mn^\epsilon) = \mathcal{O}_\epsilon(mn^\epsilon).$$

$m \le n^{32}$: We partition $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$ into $g = \lceil n^{2+\epsilon}/m^{1/16} \rceil$ groups of $k = m^{1/16} \le n^2$ surface patches each. For each group, we run the algorithm of Lemma 6.14 to build a data structure of size $\mathcal{O}_\epsilon(k^{16+\epsilon})$ in $\mathcal{O}_\epsilon(k^{16+\epsilon})$ time that supports triangle intersection queries in $\mathcal{O}_\epsilon(k^\epsilon)$ time, and then we query this data structure with all triangles in P to test for intersections in $\mathcal{O}_\epsilon(mk^\epsilon)$ steps. The total time spent is

$$\mathcal{O}_\epsilon(g(k^{16+\epsilon} + mk^\epsilon)) = \mathcal{O}_\epsilon(gm^{1+\epsilon/16}) = \mathcal{O}_\epsilon(n^{2+\epsilon}m^{15/16+\epsilon}).$$

Note that this algorithm can also compute the triangles $\Delta \in P$ that intersect $\mathsf{bd}_\delta^{\mathbf{B}_2}(Q)$. □

Putting Lemma 6.8 and Lemma 6.15 together, we obtain

Lemma 6.16 (Δ-pattern $\tilde{\delta}_H$-measure problem – decision version). *We can compute the set $X = \{\Delta \in P \mid \tilde{\delta}_H(\Delta, Q) > \delta\}$ in $\mathcal{O}_\epsilon(mn + n^{2+\epsilon}m^{15/16+\epsilon})$ randomized expected time.*

By applying Lemma 6.16 twice we get an algorithm for Problem 6.5:

Theorem 6.17 (Δ-pattern δ_H-measure problem – decision version). *We can decide whether $\delta_H(P, Q) \le \delta$ in $\mathcal{O}_\epsilon((mn)^{15/16+\epsilon}(m^{17/16} + n^{17/16}))$ randomized expected time.*

With the result described in the following Theorem and the well known Clarkson/Shor technique, c.f. [28], we can easily turn the algorithm for the decision problem into a randomized procedure that actually computes the minimal distance.

Theorem 6.18 (Computing $\tilde{\delta}_H$ of a triangle to a Δ-pattern, Godau, [35]). Let Δ be a triangle in $\mathbb{R}^3$. There is a randomized algorithm that computes $\tilde{\delta}_H(\Delta, Q)$ in $\mathcal{O}_\epsilon(n^{2+\epsilon})$ expected time.

Theorem 6.19 (Δ-pattern δ_H-measure problem – optimization version). *We can compute $\delta_H(P, Q)$ in $\mathcal{O}_\epsilon((mn)^{15/16+\epsilon}(m^{17/16} + n^{17/16}))$ randomized expected time.*

Proof. First we give a randomized algorithm to compute $\tilde{\delta}_H(P, Q)$. We follow a strategy similar to that proposed in [2]. Initially we set $\delta = 0$ and $X = P$. Then we repeat the following steps until X becomes empty:

Choose a random triangle $\Delta \in X$ and compute $\delta' = \tilde{\delta}_H(\Delta, Q)$ in $\mathcal{O}_\epsilon(n^{2+\epsilon})$ time with the algorithm from Theorem 6.18. Set δ to $\mathsf{max}(\delta, \delta')$. Now compute the set $X' = \{\Delta \in X \mid \tilde{\delta}_H(\Delta, Q) > \delta\}$ in $\mathcal{O}_\epsilon(mn + n^{2+\epsilon}m^{15/16+\epsilon})$ time with the algorithm from Lemma 6.16. Finally set X to X'.

Obviously the last value of δ will be $\tilde{\delta}_H(P, Q)$. As is shown in [28], the expected number of iterations is $\mathcal{O}(\log m)$ and therefore the expected time to compute $\tilde{\delta}_H(P, Q)$ with this algorithm is $\mathcal{O}_\epsilon((n^{2+\epsilon} + mn + n^{2+\epsilon}m^{15/16+\epsilon}) \log m)$. □