6

In the Rigid Kingdom

> ...He had Cinderella sit down, and, putting the slipper to her foot, he found that it went on very easily, fitting her as if it had been made of wax.
>
> C. Perrault, *Cinderella*

Imagine a glamorous royal ball hosted by a young Prince in his palace. Among hundreds of elegantly dressed guests, a fair lady comes uninvited. The Prince, struck by her radiant beauty, falls in love from the first sight. But all of a sudden, as the tower clock bell sounds the first stroke of midnight, the mysterious guest slips from the Prince's arms and vanishes into the darkness without a word of goodbye, leaving as the only evidence of her visit a tiny glass slipper. The Prince swears to marry the girl whose petite foot fits into it. He commands all maids in his kingdom to measure the slipper, and after a long search finally finds a poor girl, whose foot fits perfectly. The Prince recognizes his fair guest, declares his love to her, and they marry and live happily ever after.

In this brief synopsis, the reader will certainly recognize the plot of Cinderella.[1] This fairy tale illustrates the problem of *surface similarity*. Speaking in our language, the Prince was looking for a *distance function* that given two surfaces (those of the slipper and the girl's foot) provides a quantitative measure of their similarity. For this discussion, we will assume that as well as the glass slipper, Cinderella's foot is *rigid* and cannot be bent, folded, or deformed in any way; one can only change its location and orientation in space. Formally, we say that our objects are subsets of $\mathbb{R}^3$ with the standard Euclidean metric, in which the isometry group contains only rigid transformations: translation and rotation (reflections are usually excluded because they are not physical transformations). Similarity of two surfaces in such a case is extrinsic and, up to a Euclidean isometry, can be thought of as a measure of their congruence.

In this chapter, we explore tools for comparison of extrinsic geometries in a way invariant to rigid transformations. We will start our discussion in a pursuit after a representation of two surfaces X and Y that is invariant to Euclidean isometries. Next, we will view the similarity problem through the prism of numerical optimization and see how it is related to another problem of *alignment* or *correspondence* of rigid surfaces.

A. Bronstein et al., *Numerical Geometry of Non-Rigid Shapes*,
Monographs in Computer Science, DOI 10.1007/978-0-387-73301-2_6

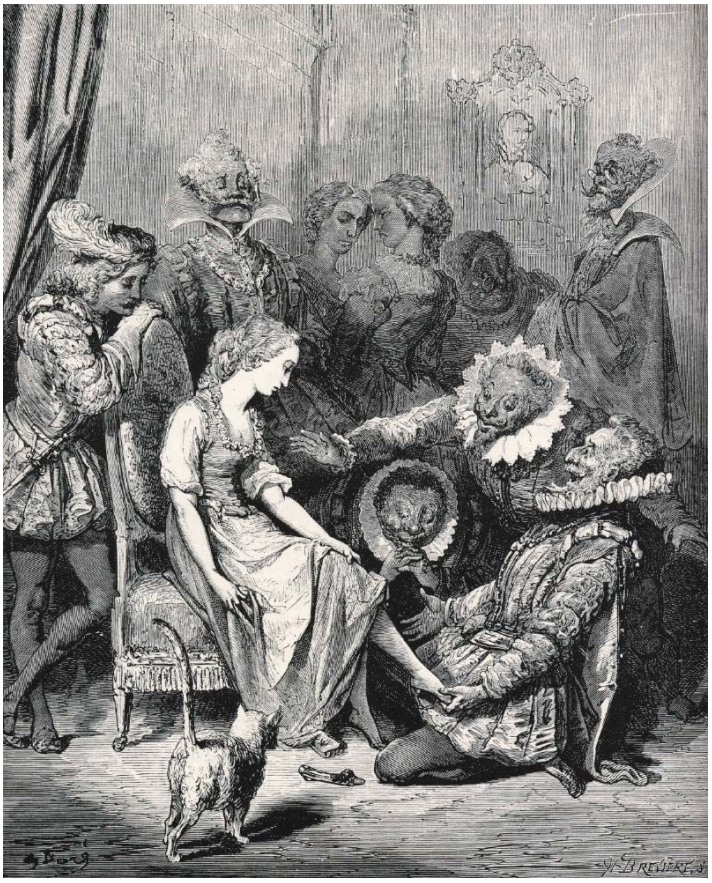

Figure 6.1. Cinderella trying on the slipper in Gustave Doré's engraving.

6.1 Moments of joy, moments of sorrow

Every rigid transformation in $\mathbb{R}^3$ can be described by six parameters: three rotation angles $\theta = (\theta^1, \theta^2, \theta^3)^{\mathrm{T}}$ about the x, y, and z axes, respectively, and three translation coordinates $t = (t^1, t^2, t^3)^{\mathrm{T}}$. Such a transformation repositions a vector x in $\mathbb{R}^3$ to

$$x' = Rx + t \;=\; R_1 R_2 R_3 x + t,$$

where

$$R_1 \;=\; \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta^1 & \sin\theta^1 \\ 0 & -\sin\theta^1 & \cos\theta^1 \end{pmatrix}, \; R_2 \;=\; \begin{pmatrix} \cos\theta^2 & 0 & \sin\theta^2 \\ 0 & 1 & 0 \\ -\sin\theta^2 & & \cos\theta^2 \end{pmatrix},$$

and

$$R_3 = \begin{pmatrix} \cos\theta^3 & \sin\theta^3 & 0 \\ -\sin\theta^3 & \cos\theta^3 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

are rotation matrices.[2]

A straightforward approach for getting rid of rigid isometries is to find a Euclidean transformation that brings a surface X to some "canonical" placement in $\mathbb{R}^3$. For example, if we could identify a landmark point s_0 on X, translating the surface by $t = -s_0$ would always bring that point to the origin, resolving the ambiguity in surface position. However, finding landmark points requires additional information about the surface, which is not always available.

Nevertheless, there exist several points that can be found for every three-dimensional surface. One of such points is the *extrinsic centroid* (the terms *center of mass* and *center of gravity* are often used as synonyms),

$$x_0 = \int_X x dx, \tag{6.1}$$

which is essentially the "average location" of X (note that unlike its intrinsic counterpart we have encountered in Chapter 3, the extrinsic centroid does not necessarily belong to X). Clearly, translating the surface in such a way that x_0 coincides with the origin resolves the translation ambiguity.

Next, we have to resolve the remaining three degrees of freedom due to rotation. This can be done by finding a direction in which the surface has maximum extent, and aligning it, say, with the the e_1 axis (Figure 6.2, left). Because a direction is described by a unit vector in $\mathbb{R}^3$, this step resolves only two of the three degrees of freedom. The remaining degree of freedom is due to the rotation ambiguity about the e_1 axis. However, we can apply the same idea again by rotating the surface such that the projection on the e_2e_3 plane, which can be illustrated as the footprint of the shadow cast by the surface (Figure 6.2, right), has the maximum extent in the direction of the e_2 axis.

Formally, the first direction we are looking for can be defined as the one that maximizes the *variance* of the projection of X onto it,

$$d_1 = \arg \max_{d_1 : \|d_1\|_2 = 1} \int_X (d_1^{\mathrm{T}} x)^2 dx,$$

where we assume that the surface has already been translated so that $x_0 = 0$. Observe that the integrand $(d_1^{\mathrm{T}} x)^2$ can be written as $d_1^{\mathrm{T}} x x^{\mathrm{T}} d_1$. Because d_1 does not participate in the integration, we can write

$$d_1 = \arg \max_{d_1 : \|d_1\|_2 = 1} d_1^{\mathrm{T}} \left(\int_X x x^{\mathrm{T}} dx \right) d_1 \;=\; \arg \max_{d_1 : \|d_1\|_2 = 1} d_1^{\mathrm{T}} \Sigma_X d_1.$$

Σ, is a 3×3 matrix, whose elements

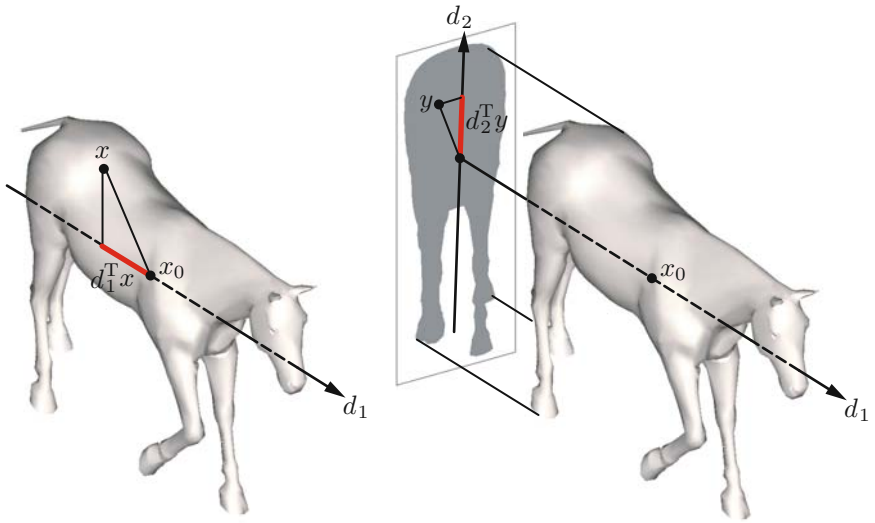

Figure 6.2. The first principal direction d_1 of the surface maximizes the variance of the projection of X onto it (left). Then, the surface is projected onto the plane orthogonal to d_1 (right, grayed) and the second principal direction d_2 is chosen as the maximum variance direction in that plane.

$$\sigma_{ij} = \int_X x^i x^j \, dx \tag{6.2}$$

are usually referred to as the *second-order geometric moments* of the surface,[3] and the direction d_1 maximizing the projection variance is called the first *principal direction*. Observe that the first principal direction, which has to maximize $d_1^{\mathrm{T}} \Sigma d_1$, is nothing but the first eigenvector of Σ corresponding with its maximum eigenvalue. In order to find the second principal direction, we have to project the surface onto the plane orthogonal to d_1 and find the vector d_2 in that plane, which maximizes the variance of the projection. Obviously, d_2 corresponds with the second largest eigenvector of Σ.

Because the matrix Σ is symmetric, it admits unitary diagonalization, that is, $\Sigma = U^{\mathrm{T}} \Lambda U$, where Λ is a diagonal matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \lambda_3$ of Σ along the diagonal, and U is a unitary matrix whose columns are the corresponding eigenvectors. We leave as an exercise (Problem 6.1) the proof of the fact that U^{T} is a rotation matrix aligning d_1 and d_2 with the e_1 and e_2 axes, respectively. Clearly, after such an alignment, the *main* second-order moments σ_{ii} coincide with λ_i, whereas the *mixed* second-order moments (that is, the off-diagonal elements σ_{12}, σ_{13} and σ_{23}) vanish.

Thus far, we have seen that the transformation $(R, t) = (U^{\mathrm{T}}, -U^{\mathrm{T}} x_0)$ resolves the ambiguity of rigid isometries and brings the surface into a "canonical" configuration in the Euclidean space (Figure 6.3). Our goal is now to

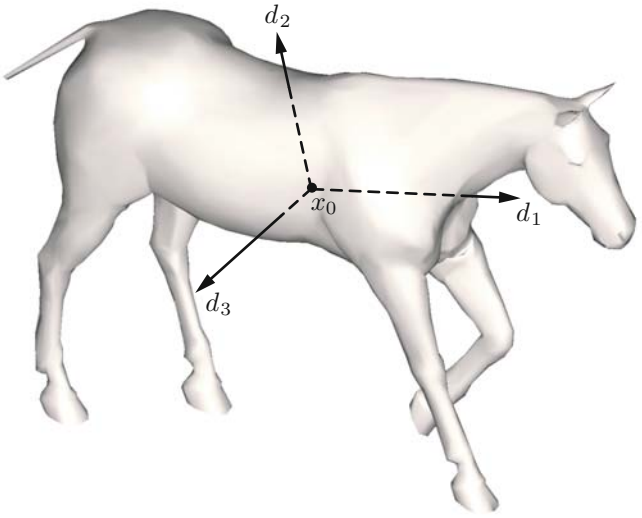

Figure 6.3. The two principal directions d_1, d_2 and a unit vector d_3 orthogonal to them define a natural coordinate system of the surface. Aligning these principal directions with the axes e_i of the standard Euclidean basis resolves the rotation ambiguity.

compare between two surfaces X and Y and quantify their similarity. We observe that the three eigenvalues λ_1, λ_2, and λ_3 of Σ provide some information about the surface extrinsic geometry. Indeed, a shape similar to a sphere is expected to have $\lambda_1 \approx \lambda_2 \approx \lambda_3$, whereas a more elongated surface should definitely have $\lambda_1 \gg \lambda_2$. In other words, the ratios $\lambda_2 : \lambda_1$ and $\lambda_3 : \lambda_1$ describe the shape *eccentricity*, and their magnitude express the shape *scale*.

We do not have to stop at the second-order moments and can define the $(p+q+r)$-th order geometric moment as

$$m_{pqr} = \int_X (x^1)^p (x^2)^q (x^3)^r dx. \tag{6.3}$$

Note that the center of gravity of the surface is a vector of its first-order moments, $x_0 = (m_{100}, m_{010}, m_{001})^{\mathrm{T}}$, whereas the elements of Σ correspond with $\sigma_{11} = m_{200}, \sigma_{22} = m_{020}, \sigma_{33} = m_{002}$ (diagonal elements), and $\sigma_{12} = m_{110}, \sigma_{13} = m_{101}, \sigma_{23} = m_{011}$ (off-diagonal elements). Higher-order moments depend on the surface position and orientation; they should be computed after performing the alignment step that eliminates the first-order and mixed second-order moments. The discretization of the integral in equation (6.3) is left as an exercise to the reader (Problem 6.3).

Intuitively, higher-order geometric moments provide us information about the surface: the more m_{pqr}'s we take, the better we can identify our object. It appears that if all moments of two surfaces coincide, the surfaces are identical. In order to understand this property, let us rewrite the (p, q, r) geometric moment of a surface as

$$m_{pqr}(f) = \int_{\mathbb{R}^3} \psi_{pqr}(x) f(x) dx \;=\; \langle \psi_{pqr}, f \rangle, \tag{6.4}$$

where $\psi_{pqr}(x) = (x^1)^p (x^2)^q (x^3)^r$, and $f : \mathbb{R}^3 \to \mathbb{R}$ is a superposition of characteristic functions, taking the value of "infinity" for $x \in X$ and zero elsewhere[4] in $\mathbb{R}^3$. Using these notations, we immediately notice that $\{m_{pqr}\}_{p,q,r=0}^{\infty}$ assume the role of the decomposition coefficients of f in the set of monomial functions $\{\psi_{pqr}\}_{p,q,r=0}^{\infty}$. Because $\{\psi_{pqr}\}$ span the space of all finite energy (more precisely, square integrable or L^2) functions on $\mathbb{R}^3$, the set of coefficients $\{m_{pqr}\}$ is unique for each surface. Indeed, if the functions f and g describing two surfaces X and Y, respectively, differ by some $h = f - g$ with non-zero energy (that is, $\int_{\mathbb{R}^3} h^2(x) dx > 0$), then there must exist some non-zero coefficients $m_{pqr}^h(h)$ such that $h = \sum m_{pqr}(h) \psi_{pqr}$. Consequently,

$$m_{pqr}(f) = \langle \psi_{pqr}, f \rangle \;=\; \langle \psi_{pqr}, g + h \rangle \;=\; m_{pqr}(g) + m_{pqr}(h) \;\neq\; m_{pqr}(f),$$

at least for some values of p, q, and r. This means that the set of all geometric moments constitutes a unique descriptor of a given surface, which is also invariant to rigid isometries if proper alignment is performed. This descriptor is also complete, meaning that, at least theoretically, one can recover[5] the surface from $\{m_{pqr}\}_{p,q,r=0}^{\infty}$.

Generally, all moments are needed to uniquely identify a surface. If we are given only a truncated set $\{m_{pqr}\}_{p,q,r=0}^{P}$ of moments up to the P-th order, there exist an infinitely large number of surfaces differing only in moments above the P-th order. However, it appears that this variety of surfaces becomes more and more similar to our surface as we increase P. In other words, even a finite set of high-order moments serves as a "fingerprint" or "signature" that identifies a sufficiently narrow class of surfaces. Ideally, we would like to be able to say that surfaces with bounded "frequencies" can be uniquely described by a finite set of moments.[6] Unfortunately, in the case of geometric moments, it is difficult to express the geometric properties of such surfaces. For this reason, geometric moments are not the best choice for measuring similarity of shapes. Other types of moments having a more clear "frequency" interpretation such as the spherical harmonics [188] or the Legendre moments [376] are usually preferred.

Using a finite set of moments, either geometric or other, we can quantify the similarity of two surfaces X and Y by applying some norm to the difference between their finite moment signatures $\{m_{pqr}(X)\}$ and $\{m_{pqr}(Y)\}$, for example,

$$d_{\mathrm{MOM}}(X,Y) = \sum_{p,q,r=0}^{P} (m_{pqr}(X) - m_{pqr}(Y))^2. \tag{6.5}$$

Said differently, d_{MOM} is a *distance function* that measures the dissimilarity between two surfaces (hereinafter, we use the term "distance" in a broad sense, not necessarily implying that d_{MOM} is a metric). Provided that X and Y are aligned prior to computing d_{MOM}, this distance function is invariant to rigid isometries. Surfaces having small distance between them are supposed to be nearly congruent (extrinsically similar), and conversely, nearly congruent surfaces result in a small d_{MOM}.

However, it is important to mention that the moment signature distance d_{MOM} has several flaws. First, recall that the continuous surfaces X and Y that we have been using freely are never available; all we have are samplings of the surfaces. It appears that the computation of moments is sensitive to the sampling, or more precisely, to sampling non-uniformity. Second, a relatively dense sampling is required in order to obtain reliable results. Third, computation of high-order geometric moments is sensitive to acquisition noise and inaccuracies due to the use of finite-precision arithmetics (see Problem 6.5). These shortcomings may limit the applicability of surface comparison methods based on moment signatures. Yet, a more serious disadvantage of d_{MOM} is that we cannot use it as a criterion of *partial similarity*.

Returning to our fairy tale example, imagine that the Prince imprudently drops the glass slipper, which breaks apart. Using moments signatures, he would never succeed in finding Cinderella, as a part of the slipper obviously has different moments than the does complete one. It is clear that the Prince needs a better distance function that still works even when the surfaces are given only partially. To his help comes a family of the so-called *iterative closest point* algorithms (*ICP* for short), first introduced by Chen and Medioni [99], and then independently by Besl and McKay [31].

6.2 Iterative closest point algorithms

The idea behind the iterative closest point algorithms is simple: given two surfaces, X and Y, find the rigid transformation (R,t), such that the transformed surface $Y' = RY + t$ is as "close" as possible to X. "Closeness" is expressed in terms of some *surface-to-surface distance* $d(RY + t, X)$. More precisely, ICP can be formulated as the minimization problem,

$$d_{\mathrm{ICP}}(X,Y) = \min_{R,t} d(RY + t, X). \tag{6.6}$$

The minimum surface-to-surface distance expresses the extrinsic similarity of X and Y. Because the minimum is searched over all Euclidean transformations, d_{ICP} is clearly invariant to rigid isometries. ICP was first proposed and

is currently used mainly for *registration* (alignment) of surfaces. In fact, the optimal rigid transformation (R^*, t^*) is the best alignment between Y and X.

Iterative closest point algorithms differ in the choice of the surface-to-surface distance $d(Y', X)$ and the numerical method for solving the minimization problem. One of the possible candidates for such a distance could be the Hausdorff distance

$$d_{\mathrm{H},\mathbb{R}^3}(Y', X) = \max\left\{\sup_{x\in X} d_{\mathbb{R}^3}(x, Y'), \sup_{y\in Y'} d_{\mathbb{R}^3}(y, X)\right\},$$

which we have already encountered in Chapter 3. However, the Hausdorff distance is rarely used in practice due to its sensitivity to outliers: difference in a single sample can make d_{H} arbitrarily large. Most commonly, $d(Y', X)$ is expressed as the sum of squared distances between all points on Y' to the surface X,

$$d(Y', X) = \sum_{y\in Y'} d^2(y, X). \tag{6.7}$$

Because Y is discrete, the sum is finite and can be thought of as an L_2 approximation of the Hausdorff distance. Note that in this formulation $d(Y', X)$ is not symmetric, yet this "unaesthetic" lack of symmetry allows ICP to handle partially missing data. Indeed, assume that Y' is congruent to a subset of the surface X. Because every point y on Y' also exists on X, we obtain $d(Y, X) = 0$ and, consequently, $d_{\mathrm{ICP}}(X, Y) = 0$. That is, we are able to tell that a part is similar to the whole. If we now take X to be congruent to a part of Y, no matter how we rotate and translate Y, there will always be points on it that have no corresponding points on X and thus $d_{\mathrm{ICP}}(X, Y)$ will not vanish. This means that the whole surface is not similar to its part, which in most cases satisfies our intuition.

The variety of choices of the surface-to-surface distance $d(Y', X)$ is now shifted to the choice of the squared *point-to-surface distance* $d^2(y, X)$. The simplest possibility is to find for every $y \in Y'$ the *closest point*[7] x^* on X and define $d^2(y, X)$ as the Euclidean distance to that point,

$$d^2(y, X) = \min_{x\in X} \|x - y\|_2^2 \;=\; \|x^* - y\|_2^2. \tag{6.8}$$

This *point-to-point distance* (Figure 6.4, left) was first proposed by Besl and McKay [31] and was probably the origin of the name "iterative closest point" that labeled this family of rigid registration algorithms. Finding the closest point on X for every y on Y establishes a correspondence between the two surfaces. Clearly, every y may have its own closest point, and theoretically, we have to go over all the points of X to find it for a given y.

Observe that the point-to-point distance treats X as a cloud of points. However, in reality X is a surface, and when a point gets sufficiently close to it, X can be approximated locally as a plane. Hence, if X is given as a triangulated mesh, we can choose $d^2(y, X)$ to be the *point-to-plane distance*

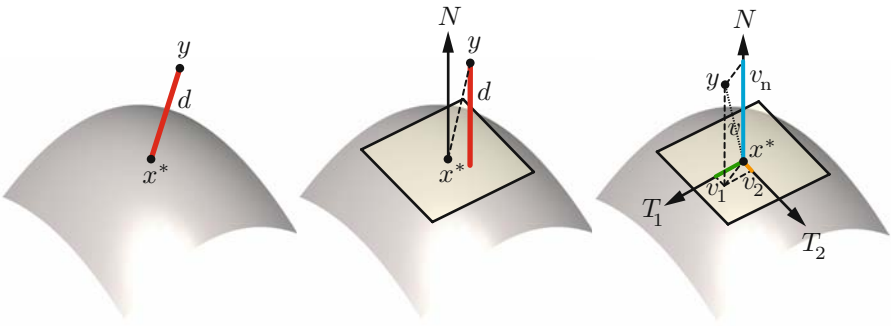

Figure 6.4. The point-to-surface distance approximated using the point-to-point (left), point-to-plane (center), and point-to-quadratic surface (right) distances. v and v_{n}, v_1, and v_2 denote the vector $y - x^*$ and its projections on N, T_1, and T_2, respectively.

$$d^2(y, X) = \min_{x \in X} \langle N(x), x - y \rangle^2, \tag{6.9}$$

where $N(x)$ denotes the unit normal vector to the surface X at the point x (Figure 6.4, center). However, now our situation is even worse than before, as the closest point x^* is no more restricted to be one of the samples of the surface X and can be therefore found anywhere on its triangular faces. Obviously, it is impractical to search for the exact closest point. A reasonable compromise is to approximate x^* by the closest sample of X as we did in the point-to-point distance. Such an approximate point-to-plane distance was used by Chen and Medioni [99].

The point-to-plane distance is based essentially on a local first-order approximation of the surface by a plane. We can refine this model by using a second-order approximation, which in addition to the normal vector N also requires the two principal curvatures κ_1, κ_2 and the corresponding principal directions T_1 and T_2 at every point.[8] Pottmann and Hofer [316] showed that the second-order Taylor approximant[9] of the squared point-to-surface distance can be expressed as

$$\begin{aligned} d^2(y, X) \approx \frac{d}{d - \rho_1} \langle T_1(x^*), y - x^* \rangle^2 + \frac{d}{d - \rho_2} \langle T_2(x^*), y - x^* \rangle^2 \\ + \langle N(x^*), y - x^* \rangle^2, \end{aligned} \tag{6.10}$$

where $\rho_i = 1/\kappa_i$ are the principal curvature radii at the point x^*, and d is the signed distance to the closest point, defined as $d = \|y - x^*\|_2$ when x^* is found at the same side of the surface pointed by the normal, and $d = -\|y - x^*\|_2$ when x^* is located at the other side (Figure 6.4, right).

Observe that for $d \ll \rho$, the first two terms vanish and $d(y, X)$ becomes the point-to-plane distance (6.9). At the other extremity, when $d \gg \rho$, one has

$$d^2(y, X) \approx \langle T_1, y - x^* \rangle^2 + \langle T_2, y - x^* \rangle^2 + \langle N, y - x^* \rangle^2 \;=\; \|x^* - y\|_2,$$

which is nothing but the point-to-point distance (6.8). Using wave terminology, the point-to-point distance is a second-order accurate "far field" approximation of the true point-to-surface distance, whereas the point-to-plane distance is second-order accurate in the "near field." This corresponds with our intuition: observed from a distance, X behaves like a point, whereas at short distances, the planar approximation is more accurate.

The Pottmann-Hofer distance (6.10) gives an accurate approximation to the point-to-surface distance for all ranges of d. Its only problem is that for some values of d, this approximation may become negative. To avoid this problem, Pottmann and Hofer proposed the following non-negative quadratic approximant

$$\begin{aligned} d^2(y, X) \approx \frac{d}{d + \rho_1} \langle T_1(x^*), y - x^* \rangle^2 + \frac{d}{d + \rho_2} \langle T_2(x^*), y - x^* \rangle^2 \\ + \langle N(x^*), y - x^* \rangle^2 . \end{aligned}$$

In general, it appears to be the best choice for the squared point-to-surface distance; its only disadvantage is the need to compute the principal curvatures and directions on the surface X. When X is contaminated by noise or sparsely sampled, this is not a trivial task.

6.3 Enter numerical optimization

Thus far, we have explored three types of functions measuring the squared distance between a point y and the surface X. Any of these distances can be employed in the ICP algorithm by plugging it into $d(RY + t, X)$ in (6.6). Our next goal is to find such a Euclidean transformation (R, t) that minimizes $d(RY + t, X)$. A straightforward way is to find the correspondence between Y and X, construct the objective function (whose terms $d^2(y', X)$ depend on the correspondence), and find the rigid isometry (R, t) that minimizes this objective function. Once we have the optimal rigid isometry, we apply it to the surface Y hoping that now it is aligned in the best way with X. However, we may discover that the transformation has changed the correspondence and the new objective can be further minimized by another rigid transformation. Therefore, we repeat the entire process again until the surface Y comes to a halt, that is, the optimal rigid transformation is close enough to the identity transformation. Formally, this leads to the iterative procedure shown in Algorithm 6.1.

This is essentially the way the first ICP algorithms worked. Step 3 can be performed using any unconditional minimization method. For the point-to-point distance, there even exists a closed-form solution for the optimal (R, t) [208].

input : Surfaces X and Y.
output : Optimal alignment (R, t), extrinsic similarity d_{ICP}.
initialization: $Y' = Y$.

1 **repeat**
2 Find the correspondence $x^*(y) = \arg\min_{x \in X} \|y - x\|_2^2$ for all $y \in Y$.
3 Minimize the error function

$$(R, t) = \arg\min_{R,t} \sum_{y \in Y'} d^2(Ry + t, X)$$

4 Transform $Y' \longleftarrow RY' + t$.
5 **until** *convergence*

Algorithm 6.1. Iterative closest point algorithm.

The above ICP algorithm is actually a heuristic approach, and little can be said about its convergence. The optimal rigid transformation (R, t) found in Step 3 minimizes the objective based on the correspondence found in Step 2. However, after the transformation is applied, the function $d(RY' + t, X)$ may be different from the one for which the transformation was found. Consequently, it is not guaranteed that this simple ICP algorithm will generate a monotonically decreasing sequence of objective function values and eventually converge.[10] On the other hand, we are already acquainted with various numerical optimization methods that guarantee convergence at least to a local minimum. An attempt to fill this apparent gap by putting the iterative closest point algorithms on this solid numerical ground seems to be imminent.

This was probably the motivation that guided Mitra *et al.* [277], who in 2004 made an important step toward this goal. The authors noted that the quadratic approximant to $d^2(y, X)$ can be written as

$$d^2(y, X) \approx y^{\mathrm{T}} Q(y) y + b(q)^{\mathrm{T}} y + c(y), \tag{6.11}$$

where $Q(y)$ is a 3×3 symmetric positive definite matrix, $b(y)$ is a 3×1 vector, and $c(y)$ is a scalar. Clearly, this function is valid only locally in the neighborhood of y, implying that Q, b, and c depend on y.

Example 6.1 (quadratic approximation of squared distances). In this example, we show how different squared distances can be brought into the form of (6.11). For the squared point-to-point distance, we can write

$$d^2(y, X) = \|y - x^*\|_2^2 \;=\; (y - x^*)^{\mathrm{T}}(y - x^*) \;=\; y^{\mathrm{T}} y - 2y^{\mathrm{T}} x^* + x^{*\mathrm{T}} x^*;$$

hence, $Q(y) = I$, $b(y) = -2x^*$, and $c(y) = x^{*\mathrm{T}} x^*$. For the squared point-to-plane distance,

$$\begin{aligned} d^2(y, X) = \langle N, y - x^* \rangle^2 &= (N^{\mathrm{T}} y - N^{\mathrm{T}} x^*)^2 \\ &= (N^{\mathrm{T}} y)^2 - 2N^{\mathrm{T}} y N^{\mathrm{T}} x^* + (N^{\mathrm{T}} x^*)^2 \\ &= y^{\mathrm{T}}(NN^{\mathrm{T}})y - 2(NN^{\mathrm{T}} x^*)^{\mathrm{T}} y + (N^{\mathrm{T}} x^*)^2; \end{aligned}$$

hence, $Q(y) = NN^{\mathrm{T}}$, $b(y) = -2NN^{\mathrm{T}} x^*$, and $c(y) = (N^{\mathrm{T}} x^*)^2$.

We can plug this quadratic form into the ICP objective function (6.7), obtaining

$$\begin{aligned} d(RY + t, X) &= \sum_{y' \in RY + t} d^2(y', X) = \sum_{y' \in RY + t} y'^{\mathrm{T}} Q(y') y + b(y')^{\mathrm{T}} y' + c(y') \\ &= \sum_{y \in Y} (Rq + t)^{\mathrm{T}} Q(Ry + t)(Ry + t) + b(Ry + t)^{\mathrm{T}} (Ry + t) + c(Ry + t), \end{aligned}$$

which should be minimized with respect to the rigid transformation (R, t).

This function is hard to minimize, as it involves $Q(Ry + t)$, $b(Ry + t)$ and $c(Ry + t)$, whose functional dependence on R and t might be complicated due to the possible changes in the correspondence between Y and X. However, assuming small motion (i.e., the rigid transformation is nearly the identity transformation, $RY + t \approx Y$), we can omit this dependence, writing

$$d(RY + t, X) \approx \sum_{y \in Y} (Ry + t)^{\mathrm{T}} Q(y)(Ry + t) + b(y)^{\mathrm{T}} (Ry + t) + c(y)$$

(the scalar $c(y)$ can be discarded, as it does not depend on R or t). The new objective appears much easier to minimize, as it is quadratic in R and t. Yet, if we use the elements of R as our optimization variables, we have to enforce the orthonormality of R in order to guarantee that it remains a rotation matrix. This makes optimization cumbersome.

An alternative is to use the three rotation angles $\theta = (\theta^1, \theta^2, \theta^3)$ as optimization variables. In this case, the objective function becomes nastier, due to the complicated dependence of R on θ, which involves trigonometric functions,

$$R = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\theta^1 & \sin\theta^1 \\ 0 & -\sin\theta^1 & \cos\theta^1 \end{pmatrix} \begin{pmatrix} \cos\theta^2 & 0 & \sin\theta^2 \\ 0 & 1 & 0 \\ -\sin\theta^2 & 0 & \cos\theta^2 \end{pmatrix} \begin{pmatrix} \cos\theta^3 & \sin\theta^3 & 0 \\ -\sin\theta^3 & \cos\theta^3 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

At this point, to our help comes the fact that the small motion assumption implies in particular a small rotation, $\theta \ll 1$. Hence, using the first-order Taylor approximations $\cos\theta \approx 1$ and $\sin\theta \approx \theta$, we can linearize the rotation matrix R as follows:

$$R \approx \begin{pmatrix} 1 & \theta^1 & -\theta^2 \\ -\theta^1 & 1 & \theta^3 \\ \theta^2 & -\theta^3 & 1 \end{pmatrix}. \tag{6.12}$$

Using the linearized R, our objective becomes quadratic with respect to the six rigid isometry parameters $\theta = (\theta^1, \theta^2, \theta^3)$ and $t = (t^1, t^2, t^3)$, and we can use the Newton method to minimize it.

However, in spite of our small motion assumption, in practice the Newton method may find a large transformation as the minimizer of $d(RY + t, X)$. Because our approximation to $d^2(y, X)$ is valid only locally, it may increase the objective function. In such cases, we should only make a small step in the direction of the transformation. In order to do it in a consistent way, let (R', t') be a small transformation that when applied sequentially η times coincides with the large transformation (R, t). Formally, this can be written as

$$\begin{aligned} Ry + t &= \underbrace{R'(\cdots(R'(R'y + t') + t')\cdots) + t'}_{\eta \text{ times}} \\ &= R'^{\eta} y + (R'^{\eta-1} + R'^{\eta-2} + \cdots + R' + I)t'. \end{aligned}$$

Demanding $R'^{\eta} y = Ry$, one has $R' = R^{1/\eta}$, corresponding with a rotation by θ/η. Multiplying the equation $t = (R'^{\eta-1} + R'^{\eta-2} + \cdots + R' + I)t'$ by $(R' - I)$ from the left, one obtains the "telescopic" matrix polynomial

$$\begin{aligned} (R' - I)t \;&=\; (R' - I)(R'^{\eta-1} + R'^{\eta-2} + \cdots + R' + I)t' \\ &= (R'^{\eta} + R'^{\eta-1} + \cdots + R' - R'^{\eta-1} - R'^{\eta-2} - \cdots - I)t' \;=\; (R'^{\eta} - I)t', \end{aligned}$$

from where $t' = (R - I)^{-1}(R' - I)t$. This simple relation can be extended to non-integer values of η as well. The step size η has to be chosen sufficiently small to guarantee a decrease of the objective function. This can be done using, for example, the Armijo rule, as was proposed by Mitra *et al.* This approach results in a significantly more stable ICP algorithm, exhibiting better convergence.

6.4 Rigid correspondence

Note that at each iteration of the ICP algorithm where Y is transformed, the correspondence between X and Y may change and has to be recomputed. Even in the elegant formulation proposed by Mitra *et al.*, this need is inevitable, as the parameters $Q(y), b(y)$, and $c(y)$ in the quadratic form (6.11) depend on y and have to be found again once Y is transformed. The simplest way to solve this problem is by computing the parameters *on demand*, i.e., for every point $y \in Y'$ at every iteration of the algorithm, we have to find the closest point,

$$x^*(y) = \arg\min_{x \in X} \|y - x\|_2^2.$$

This sounds like a potentially expensive algorithm. Indeed, if modern ICP algorithms were implemented this way, they would have been terribly slow.

Fortunately, there exist techniques for avoiding exhaustively searching over all points on the surface. Observe that X subdivides $\mathbb{R}^3$ into a collection of Voronoi cells

$$V(x) = \{y \in \mathbb{R}^3 : \|y - x\|_2 < \|y - x'\|_2 \ \ \forall x' \neq x\},$$

containing all points in $\mathbb{R}^3$ that are closer to x than to any other point on X. Finding the closest point in X given a query point y can be formulated as determining the Voronoi cell to which y belongs. This observation motivates the techniques that use efficient data structures for fast retrieval of the closest point, without exhaustively searching all points in X. It appears that even when X is given as a cloud of points, its Voronoi cells are convex polyhedra with generally complicated shapes, hardly computable efficiently. However, we can approximate the true Voronoi cells using some simpler shapes. One of such approaches is based on a hierarchical data structure called the *k-dimensional* (or *kd*) *tree* [25]. Each node of the kd tree corresponds with a partition of the space ($\mathbb{R}^3$ in our case) by a plane perpendicular to one of the axes. For example, the root node splits the space into two regions: $\{x^1 < 0\}$ and $\{x^1 \geq 0\}$. The first region is assigned to the left child, whereas the second region is assigned to the right child. Each child may introduce further splitting, e.g., $\{x^1 \geq 0\}$ is divided into $\{x^1 \geq 0\} \cap \{x^2 < 1\}$ and $\{x^1 \geq 0\} \cap \{x^2 \geq 1\}$, and so on. A leaf represents a (possibly unbounded) box-shaped region in $\mathbb{R}^3$. These boxes approximate the Voronoi cells of X. Using versions of the kd tree allows finding the approximate nearest neighbor of y in X with logarithmic complexity [11]. This significantly alleviates the computational burden of the iterative closest point algorithm.

However, if we use the quadratic form (6.11) as proposed by Mitra *et al.*, the need to recompute the correspondence at each iteration for every $y \in Y'$ still seems somewhat superfluous. Indeed, we never use the correspondence explicitly. All we need is to find the quadratic form parameters Q, b, and c for a given query point y. Because the squared distance function $d^2(y, X)$ is at least $\mathcal{C}^0$, these parameters vary smoothly and therefore, for a sufficiently small region around y, the terms Q, b, and c remain nearly constant. Once again, the idea of hierarchical space partitioning can be exploited here. Leopoldseder *et al.* [249] proposed an *octree*-like structure that recursively splits the space into eight octants, until the variance of the quadratic form parameters in the created box-shaped cell falls below a small threshold. Once the tree is precomputed for the surface X, it allows the retrieval of $Q(y)$, $b(y)$, and $c(y)$ with logarithmic complexity.

It is worthwhile noting that all surface-to-surface distances we have discussed were based on the knowledge of correspondence between the two surfaces. We may therefore say that finding the rigid correspondence is the principal ingredient of ICP. To emphasize this fact, Rusinkiewicz and Levoy even suggested the backronym *iterative corresponding point* for ICP as a replacement for the original *iterative closest point* [329]. In addition to being in the

core of ICP, correspondence between two objects is required in many other applications. We defer this discussion to Chapter 12, where the correspondence problem is explored in the more general non-rigid setting.

As a concluding remark, a few words ought to be said about the initialization of the ICP algorithm. Being a non-convex minimization problem, ICP may converge to a wrong local minimum if initialized incorrectly. While finding a good initialization is considered a largely open problem typically solved *ad hoc*, in the past few years guaranteed globally optimal initialization schemes were proposed. In [170], Gelfand *et al.* address this issue using a *branch and bound* global optimization algorithm to find the initial rigid correspondence. A variant of this approach is proposed in [251] by Li and Hartley.

Suggested reading

A good overview of shape similarity techniques can be found in Veltkamp's papers [387, 388, 373]. Moment-based shape descriptors are reviewed in [409, 320]. The reader is also referred to [150] for an interesting discussion on reconstructing a shape from its moments. The review paper [329] discusses efficient variants of the ICP algorithm and shows their convergence in different scenarios. Convergence is also discussed in [317]. An interesting paper by Ezra *et al.* [154] presents lower and upper bounds on the number of iterations in ICP algorithms. A tighter lower bound as well as a probabilistic upper bound are presented in [10]. In [318], Pottmann *et al.* introduce a "correspondence-less" approach to rigid surface registration based on their quadratic approximation to the squared point-to-surface distance previously discussed in this chapter. Another interesting "correspondence-less" approach is proposed by Charpiat *et al.* [98], where a smooth approximation to the Hausdorff distance is studied.

Software

A C++ implementation of ICP is available in the VTK and ITK toolboxes.

Problems

6.1. Show that the rotation matrix aligning the principal directions with the axes is the diagonalizing matrix of Σ.

6.2. Try to characterize the class of surfaces completely described by a *finite* set of their geometric moments $\{m_{pqr}\}_{p,q,r=0}^{N}$.

6.3.✓ Derive a consistent way to discretize the geometric moment integral.

6.4. Discuss the use of the three-dimensional Fourier harmonics as a replacement to the geometric moments. How can the translation invariance of Fourier harmonics be helpful?

6.5. ✓ In reality, finite-precision arithmetics are used to compute the moments. Assume that the coordinates of the surface points are represented with the absolute error of, say, $\epsilon = 10^{-8}$. What will be the relative error of m_{pqr}? How can this complicate the use of geometric moments?

6.6 (Research question). Suppose the surface is acquired each time from a different known viewing angle, with partial occlusions. Given the signature of the surface moments for each angle, what can be said about the moments of the entire surface? Can it be reconstructed from such partial observations?

6.7. Derive the distance in equation (6.10) and show that it is a second-order approximation to the true point-to-surface distance.

6.8. Prove that the squared point-to-surface distance is not $\mathcal{C}^2$ for query points located on the surface's medial axis.

6.9. Derive the quadratic form parameters Q, b, and c for the second-order point-to-surface distance (as was shown in Example 6.1). Compare them with the point-to-point and point-to-plane distances. What can be said about the convexity of the quadratic form?

6.10. ✓ Derive a closed-form solution for the optimal rigid isometry (R, t) minimizing the ICP objective function with the squared point-to-point distance.

Notes

[1]The earliest record of this popular fairy tale originated in China in the mid-ninth century. There, the fair Ye Xian had the smallest foot in the kingdom, a synonym of beauty in the Chinese culture. In the West, the most renowned version of Cinderella belongs, perhaps, to the pen of the French author Charles Perrault (1628–1703) [311].

[2]In order to include reflections, we can multiply x' by a diagonal matrix containing ± 1's along the diagonal.

[3]In statistics, Σ is called the *covariance matrix*, and the process of finding the variance-maximizing orthogonal directions is usually referred to as *principal component analysis* (PCA) or the *Karhunen-Loéve transform* (KLT). PCA allows one to construct a low-dimensional approximation of a multi-dimensional random process that captures its "most significant" part (in the L_2 sense). The invention of principal component analysis is usually attributed to the American statistician and economist Harold Hotelling [210], though similar ideas date back to Pearson [308].

[4]Such functions are called *Dirac's delta functions*. If the reader feels uncomfortable with such a formulation, he or she can think of the surface X as of a thin shell; in this case, f takes some constant value for x belonging to the shell, and 0

otherwise. The constant is selected in such way that $\int_X f(x)dx = 1$. For vanishing shell thickness, a delta function is obtained.

[5]The problem of reconstructing the surface from its moments is often called the *inverse moment* problem, though in the signal processing jargon, the term *synthesis* is more adequate [150]. Because the polynomial basis $\{\psi_{pqr}(x) = (x^1)^p(x^2)^q(x^3)^r\}$ is non-orthogonal, reconstruction of a surface from its geometric moments requires the so-called *biorthonormal* basis, which behaves badly in this case. For this reason, in applications where reconstruction is important, geometric moments are of little use. The preference is given to orthonormal bases that allow direct reconstruction of the surface according to $f = \sum m_{pqr}\psi_{pqr} = \sum\langle\psi_{pqr}, f\rangle\psi_{pqr}$. An example of such type of moments are the Legendre moments, which follow in spirit the Fourier transform [376].

[6]Readers familiar with the Nyquist-Shannon sampling theorem will find such a statement analogous to saying that a band-limited signal can be fully represented by its discrete sampling.

[7]The closest point is sometimes referred to as the *normal footpoint*, as the line segment connecting x^* and y is always perpendicular to the surface.

[8]Some surfaces may have *umbilical points*, where the principal curvature directions are not well-defined. In this case, we may take any two orthogonal vectors T_1, T_2 in the tangent plane.

[9]More precisely, the second-order approximant to d^2 does not always exist. At the points located on the surface's medial axis, d^2 is not $\mathcal{C}^2$. Such points have to be detected and excluded in order not to jeopardize the convergence of ICP algorithms based on quadratic surface approximation.

[10]Under certain conditions, Ezra *et al.* [154] show that ICP converges and present a bound on the number of iterations.