Chapter 7

Four Algorithms to Smooth a Shape

In this short but important chapter, we discuss algorithms whose aim it is to smooth shapes. Shape must be understood as a rough data which can be extracted from an image, either a subset of the plane, or the curve surrounding it. Shape smoothing is directed at the elimination of spurious, often noisy, details. The smoothed shape can then be reduced to a compact and robust code for recognition. The choice of the right smoothing will make us busy throughout the book. A good part of the solution stems from the four algorithms we describe and their progress towards more robustness, more invariance and more locality. What we mean by such qualities will be progressively formalized. We will discuss two algorithms which directly smooth *sets*, and two which smooth Jordan curves. One of the aims of the book is actually to prove that both approaches, different though they are, eventually yield the *very same process*, namely a curvature motion.

7.1 Dynamic shape

In 1986, Koenderink and van Doorn defined a *shape* in $\mathbb{R}^N$ to be a closed subset X of $\mathbb{R}^N$ [78]. They then proposed to smooth the shape by applying the heat equation $\partial u/\partial t - \Delta u = 0$ directly to $\mathbf{1}_X$, the characteristic function of X. Of course, the solution $G_t * \mathbf{1}_X$ is not a characteristic function. The authors defined the evolved shape at scale t to be

$$X_t = \{\mathbf{x} \mid u(t, \mathbf{x}) \geq 1/2\}.$$

The value $1/2$ is chosen so the following simple requirement is satisfied: Suppose that X is the half-plane $X = \{(x, y) \mid (x, y) \in \mathbb{R}^2, x \geq 0\}$. The requirement is that this half plane doesn't move,

$$X = X_t = \{(x, y) \mid G_t * \mathbf{1}_X(x, y) \geq \lambda\},$$

and this is true only if $\lambda = 1/2$. There are at least two problems with dynamic shape evolution for image analysis. The first concerns nonlocal interactions, as illustrated in Figure 7.1. Here we have two disks that are near one another.

Figure 7.1: Nonlocal interactions in the dynamic shape method. Left to right: Two close disks interact as the scale increases. This creates a new, qualitatively different, shape. The change of topology, at the scale where the two disks merge into one shape, also entails the appearance of a singularity (a cusp) on the shape(s) boundaries.

The evolution of the union of both disks, considered as a single shape, is quite different from the evolution of the disks separately. A related problem, also illustrated in Figure 7.1, is the creation of singularities. Note how a singularity in orientation and the curvature of the boundary of the shape develops at the point where the two disks touch. Figure 7.2 further illustrates the problems associated with the dynamic shape method.

7.2 Curve evolution using the heat equation

We consider shapes in $\mathbb{R}^2$ whose boundaries can be represented by a finite number of simple closed rectifiable Jordan curves. Thus, each curve we consider can be represented by a continuous mapping $f : [0,1] \to \mathbb{R}^2$ such that f is one-to-one on $(0,1)$ and $f(0) = f(1)$, and each curve has a finite length. We also assume that these curves do not intersect each other. We will focus on smoothing one of these Jordan curves, which we call C_0. We assume that C_0 is parameterized by $s \in [0, L]$, where L is the length of the curve. Thus, C_0 is represented as $\mathbf{x}_0(s) = (x(s), y(s))$, where s is the length of the curve between $\mathbf{x}_0(0)$ and $\mathbf{x}_0(s)$.

At first glance, it might seem reasonable to smooth C_0 by smoothing the coordinate functions x and y separately. If this is done linearly, we have seen from Theorem 2.3 that the process is asymptotic to smoothing with the heat equation. Thus, one is led naturally to consider the vector heat equation

$$\frac{\partial \mathbf{x}}{\partial t}(t,s) = \frac{\partial^2 \mathbf{x}}{\partial s^2}(t,s) \tag{7.1}$$

with initial condition $\mathbf{x}(0,s) = \mathbf{x}_0(s)$. If $\mathbf{x}(t,s) = (x(t,s), y(t,s))$ is the solution of (7.1), then we know from Proposition 1.9 that

$$\inf_{s\in[0,L]} x_0(s) \le x(t,s) \le \sup_{s\in[0,L]} x_0(s),$$
$$\inf_{s\in[0,L]} y_0(s) \le y(t,s) \le \sup_{s\in[0,L]} y_0(s),$$

for $s \in [0, L]$ and $t \in [0, +\infty)$. Thus, the evolved curves C_t remain in the rectangle that held C_0. Also, we know from Proposition 2.5 that the coordinate functions $x(t,\cdot)$ and $y(t,\cdot)$ are C^∞ for $t > 0$. There are, however, at least two reasons that argue against smoothing curves this way:

Figure 7.2: Nonlocal behavior of shapes with the dynamic shape method. This image displays the smoothing of two irregular shapes by the dynamic shape method (Koenderink–van Doorn). Top left: initial image, made of two irregular shapes. From left to right, top to bottom: dynamic shape smoothing with increasing Gaussian variance. Notice how the shapes merge more and more. We do not have a separate analysis of each shape but rather a "joint analysis" of the two shapes. The way the shapes merge is of course sensitive to the initial distance between the shapes. Compare with Figure 7.4.

(1) When $t > 0$, s is no longer a length parameter for the evolved curve C_t.

(2) Although $x(t, \cdot)$ and $y(t, \cdot)$ are C^∞ for $t > 0$, this does not imply that the curves C_t have similar smoothness properties. In fact, it can be seen from Figure 7.3 that it is possible for an evolved curve to cross itself and it is possible for it to develop singularities.

How is this last mentioned phenomenon possible ? It turns out that one can parameterize a curve with corners or cusps with a very smooth parameterization: see Exercise 7.1.

In image processing, we say that a process that introduces new features, such as described in item (2) above, is not *causal*. [1]

7.3 Restoring locality and causality

Our main objective is to redefine the smoothing processes so they are local and do not create new singularities. This can be done by alternating a small-scale linear convolution with a natural renormalization process.

[1] This informal definition should not be confused with the use of "causality," as it is used, for example, when speaking about filters: A filter F is said to be causal, or realizable, if the equality of two signals s_0 and s_1 up to time t_0 implies that $Fs_0(t) = Fs_1(t)$ for the same period.

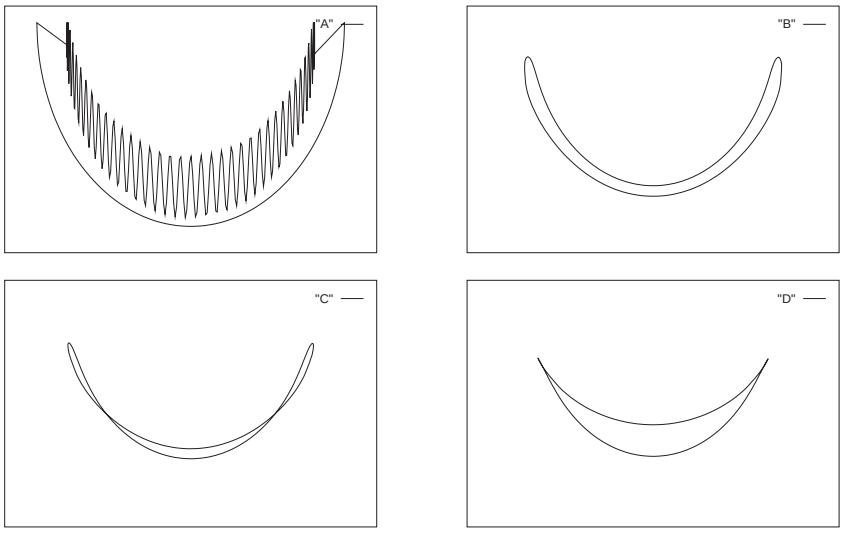

Figure 7.3: Curve evolution by the heat equation. The coordinates of the curves are parameterized by the arc length and then smoothed as real functions of the length using the heat equation. From A to D: the coordinates are smoothed with an increasing scale. Each coordinate function therefore is C^∞; the evolving curve can, however, develop self-crossings (as in C) or singularities (as in D).

7.3.1 Localizing the dynamic shape method

In the case of dynamic shape analysis, we define an alternate dynamic shape algorithm as follows:

Algorithm 7.1 (The Merriman–Bence–Osher algorithm).

(1) Convolve the characteristic function of the initial shape X_0 with G_h, where h is small.

(2) Define $X_1 = \{\mathbf{x} \mid G_h * \mathbf{1}_{X_0} \geq 1/2\}$.

(3) Set $X_0 = X_1$ and go back to (1).

This is an iterated dynamic shape algorithm. The dynamic shape method itself is an example of a *median filter*, which will be defined in Chapter **??**. The Merriman–Bence–Osher algorithm is thus an *iterated median filter* (see Figure 7.4). We will see in Chapters **??** and **??** that median filters have asymptotic properties that are similar to those expressed in Theorem 3.3. In the case of median filters, the associated partial differential equation will be a curvature motion equation (defined in Chapter **??**).

7.3.2 Renormalized heat equation for curves

In 1992, Mackworth and Mokhtarian noticed the loss of causality when the heat equation was applied to curves [93]. Their method to restore causality looks, at least formally, like the remedy given for the nonlocalization of the dynamic

Figure 7.4: The Merriman–Bence–Osher shape smoothing method is a localized and iterated version of the dynamic shape method. A convolution of the binary image with small-sized Gaussians is alternated with mid-level thresholding. It uses the same initial data (top, left) as in Figure 7.2. From left to right, top to bottom: smoothing with increasing scales. Notice that the shapes remain separate. In fact, their is no interaction between the evolving shapes. Each one evolves as if the other did not exist.

shape method. Instead of applying the heat equation for relatively long times (or, equivalently, convolving the curve $\mathbf{x}$ with the Gaussian G_t for large t), they use the following algorithm:

Algorithm 7.2 (Renormalized heat equation for curves).

(1) Convolve the initial curve $\mathbf{x}_0$, parameterized by its length parameter $s_0 \in [0, L_0]$, with the Gaussian G_h, where h is small.

(2) Let L_n denote the length of the curve $\mathbf{x}_n$ obtained after n iterations and let s_n denote its length parameter. For $n \geq 1$, write $\tilde{\mathbf{x}}_{n+1}(s_n) = G_h * \mathbf{x}_n(s_n)$. Then reparameterize $\tilde{\mathbf{x}}_{n+1}$ by its length parameter $s_{n+1} \in [0, L_{n+1}]$, and denote it by $\mathbf{x}_{n+1}$.

(3) Iterate.

This algorithm is illustrated in Figure 7.5. It should be compared with Figure 7.3.

Theorem 7.1. *Let $\mathbf{x}$ be a C^2 curve parameterized by its length parameter $s \in [0, L]$. Then for small h,*

$$G_h * \mathbf{x}(s) - \mathbf{x}(s) = h\frac{\partial^2 \mathbf{x}}{\partial s^2} + o(h). \tag{7.2}$$

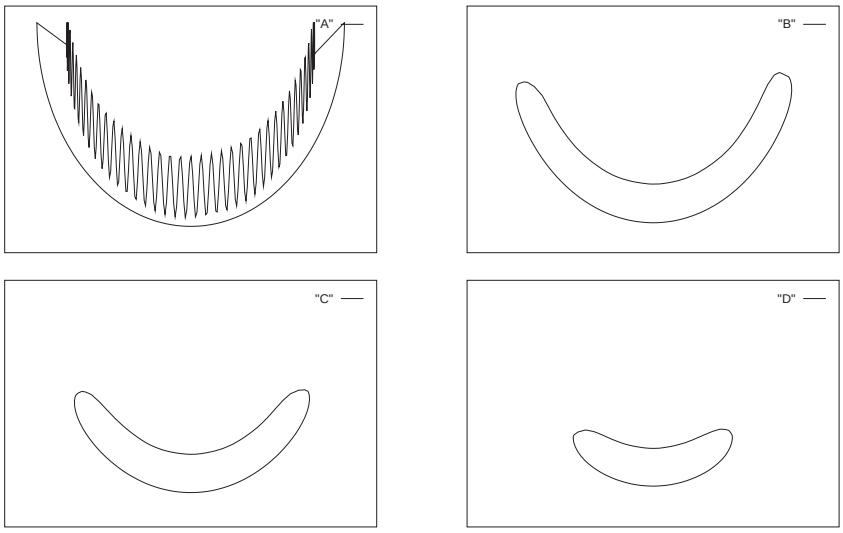

Figure 7.5: Curve evolution by the renormalized heat equation (Mackworth–Mokhtarian). After each smoothing step, the coordinates of the curve are reparameterized by the arc length of the smoothed curve. From A to D: the curve is smoothed with an increasing scale. Note that, in contrast with the linear heat equation (Figure 7.3), the evolving curve shows no singularities and does not cross itself.

This theorem is easily checked, see Exercise 7.2

In view of (7.2) and what we have seen regarding asymptotic limits in Theorem 3.3 and Exercise 3.5, it is reasonable to conjecture that, in the asymptotic limit, Algorithm 7.2 will yield the solution of following evolution equation:

$$\frac{\partial \mathbf{x}}{\partial t} = \frac{\partial^2 \mathbf{x}}{\partial s^2}, \tag{7.3}$$

where $\mathbf{x}_0 = \mathbf{x}(0, \cdot)$. It is important to note that (7.3) is *not* the heat equation (7.1). Indeed, from Algorithm 7.2 we see that s must denote the length parameter of the evolved curve $\mathbf{x}(t, \cdot)$ at time t. In fact $\partial^2 \mathbf{x}/\partial s^2$ has a geometric interpretation as a curvature vector. We will study this nonlinear curve evolution equation in Chapter **??**.

7.4 Exercises

Exercise 7.1. Construct a C^∞ mapping $f : [0, 1] \to \mathbb{R}^2$ such that the image of $[0, 1]$ is a square. This shows that a curve can have a C^∞ parameterization without being smooth. ■

Exercise 7.2. Prove Theorem 7.1. If $\mathbf{x}$ is a C^3 function of s, then the result follows directly from Theorem 3.2. The result holds, however, for a C^2 curve. ■

7.5 Comments and references

Dynamic shape, curve evolution, and restoring causality. Our account of the dynamic shape method is based on the well-known paper by Koenderink and van Doorn in which they introduced this notion [78]. The curve evolution by the heat equation is from the first 1986 version of curve analysis proposed by Mackworth and Mokhtarian [92]. See also the paper by Horn and Weldon [61]. There were model errors in the 1986 paper [92] that were corrected by the authors in their 1992 paper [93]. There, they also proposed the correct intrinsic equation. However, this 1992 paper contains several inexact statements about the properties of the intrinsic equation. The correct theorems and proofs can be found in a paper by Grayson written in 1987 [56]. The algorithm that restores causality and locality to the dynamic shape method was discovered by Merriman, Bence, and Osher, who devised this algorithm for a totally different reason: They were looking for a clever numerical implementation of the mean curvature equation [101].

Topological change under smoothing. We have included several figures that illustrate how essential topological properties of an image change when the image is smoothed with the Gaussian. Damon has made a complete analysis of the topological behavior of critical points of an image under Gaussian smoothing [34]. This analysis had been sketched in [159].

Chapter 8

Affine Invariant Image Comparison

If a physical object has a smooth or piecewise smooth boundary, its images obtained by cameras in varying positions undergo smooth apparent deformations. These deformations are locally well approximated by affine transforms of the image plane.

In consequence the solid object recognition problem has often been led back to the computation of affine invariant image local features. Such invariant features could be obtained by normalization methods, but no fully affine normalization method exists for the time being. As a matter of fact, the scale invariance, which actually means invariance to blur, is only dealt with by methods inspired from the scale space theory, like the SIFT method. By simulating zooms out, this method normalizes the four translation, rotation and scale (blur) parameters, out of the six parameters of an affine transform. Affine normalization methods like MSER or Hessian Affine normalize with respect to all six parameters of the affine transform, but this normalization is imperfect, not dealing rigorously with blur for MSER, or not starting with affine invariant scale space extrema for Hessian Affine.

The method proposed in this chapter, affine SIFT (A-SIFT), simulates all image views obtainable by varying the two camera parameters left over by the SIFT method. Then it normalizes the other four parameters by simply using the SIFT method itself. The two additional parameters are the angles (a longitude and a latitude) defining the camera axis orientation. Mathematical arguments will be given in Chapter 9 to prove that the resulting method is fully affine invariant, up to an arbitrary precision.

Against any prognosis, simulating all views depending on the two camera orientation parameters is feasible with no dramatic computational load. The method permits to reliably identify features that have undergone tilts of large magnitude, up to 30 and more, while state-of-the-art methods do not exceed tilts of 2.5 (SIFT) or 4.5 (MSER). This chapter puts in evidence the role of high *transition tilts*: while a tilt from a frontal to an oblique view exceeding 6 is rare, higher transition tilts are common as soon as two oblique views of an object are compared (see Fig. 8.1). Thus, a fully affine invariance is required for 3D scene analysis. This fact is substantiated by many experiments.

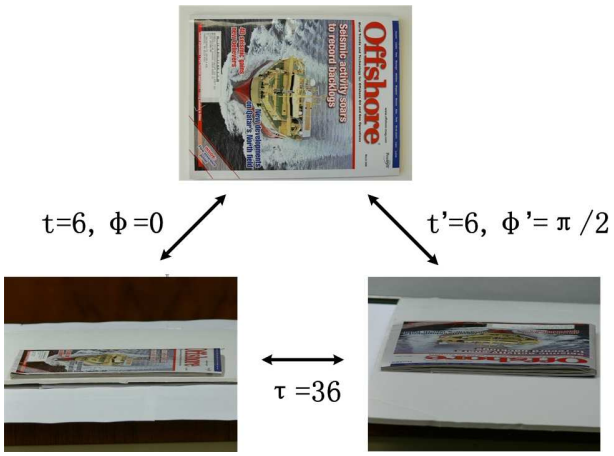

Figure 8.1: High transition tilts

Section 8.1 gives the main decomposition formula of affine maps used throughout the paper and its geometric interpretation in terms of cameras at infinity. Section 10.1 describes and discusses a method that attempts affine invariance by normalization: MSER. Section 8.2 describes the A-SIFT algorithm and discusses precursors. Section 8.3 presents and experiments the crucial notion of *transition tilt*.

8.1 The affine camera model

The general (solid) shape recognition problem starts with several photographs of a physical object, possibly taken with different cameras and view points. These digital images are the *query* images. Given other digital images, the *search* images, the question is whether some of them contain, or not, a view of the object taken in the query image. A solid object's view can deform from an image to another for two obvious reasons: First, because it underwent some physical deformation, and second, because the change of camera position induced an apparent deformation.

Image distortions arising from viewpoint changes can be locally modeled by affine planar transforms, provided the object's boundaries are piecewise smooth. In other terms, a perspective effect can be modeled by a combination of several different affine transforms in different image regions (see Fig. 8.3). Indeed, by first order Taylor formula, any planar smooth deformation $(x, y) \to (X, Y) = (F_1(x, y), F_2(x, y))$ can be locally approximated around each point $(x_0, y_0) \to (X_0, Y_0)$ by the affine map

$$\begin{pmatrix} X{-}X_0 \\ Y{-}Y_0 \end{pmatrix} = \begin{bmatrix} \frac{\partial F_1}{\partial x}(x_0, y_0) & \frac{\partial F_1}{\partial y}(x_0, y_0) \\ \frac{\partial F_2}{\partial x}(x_0, y_0) & \frac{\partial F_2}{\partial y}(x_0, y_0) \end{bmatrix} \begin{pmatrix} x{-}x_0 \\ y{-}y_0 \end{pmatrix} + O\begin{pmatrix} (x{-}x_0)^2 + (y{-}y_0)^2 \\ (x{-}x_0)^2 + (y{-}y_0)^2 \end{pmatrix}. \tag{8.1}$$

Thus, all object deformations and all camera motions are locally approximated by affine transforms. For example, in the case of a flat object, the

Figure 8.2: Geometric interpretation of the Taylor formula (8.1): Although the global deformation of each wall is strongly projective (a rectangle becomes a trapezoid), the it local deformation is affine: each tile on the pavement is almost a parallelogram. Indeed, projective maps are C^1 and therefore locally affine. The painting, due to Uccello, is one of the first Renaissance paintings with a correct geometric perspectives following the rules invented by Brunelleschi.

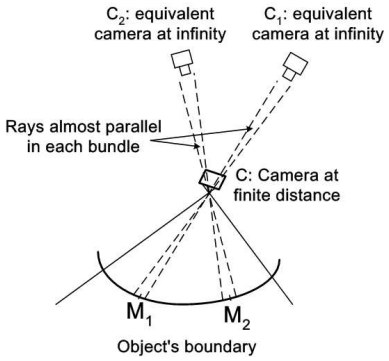

Figure 8.3: Another way to understand why the local object apparent deformations are affine. Local planar homographies are equivalent to multiple local cameras at infinity. Cameras at infinity generate affine deformations of planar objects. This is true even if the object under observation is curved, because it is then locally planar. Thus, the overall apparent deformation of the object is C^1, and Formula (8.1) applies.

deformation induced by a camera motion is a planar homographic transform, which is smooth and therefore locally tangent to affine transforms.

The converse statement is true: *any affine transform with positive determinant can be interpreted as the apparent deformation induced on a planar object by a camera motion, the camera being assumed far away from the object.* Thus, under the local smoothness assumption of the object's boundary, the (local) deformation model of an image $u(x, y)$ under a deformation of the object or under a camera motion is

$$u(x, y) \to u(ax + by + e, cx + dy + f),$$

where the mapping

$$\begin{pmatrix} x \\ y \end{pmatrix} \to \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} e \\ f \end{pmatrix}$$

is any affine transform of the plane with positive determinant. The above statements rely on the next crucial following decomposition formula.

Theorem 8.1. *Any linear planar map whose matrix A has strictly positive determinant has a unique decomposition*

$$A = H_\lambda R_1(\psi) T_t R_2(\phi) = \lambda \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{bmatrix} \tag{8.2}$$

where $\lambda > 0$, λt is the determinant of A, R_i are rotations, $\phi \in [0, \pi[$, and T_t is a tilt, namely a diagonal matrix with a first eigenvalue equal to $t \geq 1$ and the second one equal to 1.

Proof. Consider the real symmetric positive semi-definite matrix $A^t A$, where A^t denotes the transposed matrix of A. By classic spectral theory there is an orthogonal transform O such that $A^t A = ODO^t$ where D a diagonal matrix with ordered eigenvalues $\lambda_1 \geq \lambda_2$. Set $O_1 = AOD^{-\frac{1}{2}}$. Then

$$O_1 O_1^t = AOD^{-\frac{1}{2}} D^{-\frac{1}{2}} O^t A^t = AOD^{-1} O^t A^t = A(A^t A)^{-1} A^t = I.$$

Thus, there are orthogonal matrices O_1 and O such that

$$A = O_1 D^{\frac{1}{2}} O^t. \tag{8.3}$$

Since the determinant of A is positive, the product of the determinants of O and O_1 is positive. If both determinants are positive, then O and O_1 are rotations and we can write $A = R(\psi) D R(\phi)$. If ϕ is not in $[0, \pi[$, changing ϕ into $\phi - \pi$ and ψ into $\psi + \pi$ ensures that $\phi \in [0, \pi[$. If the determinants of O and O_1 are both negative, replacing O and O_1 respectively by $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} O$ and $\begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} O_1$ makes them into rotations without altering (8.3), and we can as above ensure $\phi \in [0, \pi[$ by adapting ϕ and ψ. The final decomposition is obtained by taking for λ the smaller eigenvalue of $D^{\frac{1}{2}}$. □

Exercise 8.1. The aim of the exercise is to show the uniqueness of the decomposition (8.2). Assume there are two decompositions $\lambda R_1 \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} R_2 = \lambda' R'_1 \begin{bmatrix} t' & 0 \\ 0 & 1 \end{bmatrix} R'_2$. Using the uniqueness of the eigenvalues of a matrix show first that $\lambda = \lambda'$, $t = t'$. You will obtain a relation of the form $R_1 D R_2 = D$ where D is diagonal and R_1 and R_2 are rotations. Deduce from this relation that $R_1 D^2 R_1^t = D^2$. Deduce from this last relation the form of R_1, conclude carefully. ■

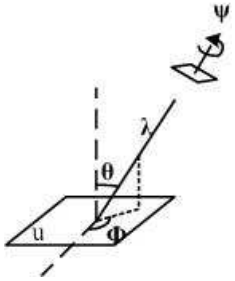

Figure 8.4: Geometric interpretation of the decomposition formula (8.2).

Exercise 8.2. Consider two cameras looking at a flat square piece of landscape which is assimilated to an infinite resolution image $u_0(x, y)$ (See Fig. 8.4). The first camera is very far above the landscape and looking down perpendicularly to the landscape.

(i) Assuming the first camera is pin-hole, show that the generated image is a square image $u_0(\mu R(\psi)(x, y))$. Consider the coordinate system $(O, \vec{i}, \vec{j}, \vec{k})$ such that $(\vec{i}, \vec{j})$ are the coordinate vectors in the square image u_0, parallel to the image sides, and O is the image center.

(ii) Assume a second pinhole camera has its optical axis pointing down to O. Assume its optical axis is supported by the unit vector with coordinates $(\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$. Assume again that this camera is very far from the square piece of landscape, so the light rays coming from the landscape to the camera are almost parallel. Thus the image formation on this second camera is assimilated to an orthogonal projection of the landscape u_0 onto a plane passing by the camera center C and orthogonal to the optical axis. Taking adequate coordinates on this coordinate plane, show that the generated image is $u_0\left(R(\psi_1)T_{t_1}R(\phi_1)\begin{pmatrix} x \\ y \end{pmatrix}\right)$ for some values of ψ_1, ϕ_1, t, that you will relate to ϕ, ψ, and θ.

■

Fig. 8.4 shows a camera motion interpretation of this affine decomposition: ϕ and $\theta = \arccos 1/t$ are the viewpoint angles and ψ parameterizes the camera spin. Thus, this figure illustrates the four main parameters in the affine image deformation caused by a camera motion, starting from a frontal view u. The camera is assumed to stay far away from the image. The camera can first move parallel to the object's plane: this motion induces a translation $\mathcal{T}$ that is not represented here. The camera can rotate around its optical axis (rotation parameter ψ). Its optical axis can take a θ angle with respect to the normal to the image plane u. This parameter is called *latitude*. The plane containing the normal and the new position of the optical axis makes an angle ϕ with a fixed

vertical plane. This angle is called *longitude*. Last but not least, the camera can move forward or backward. This is the zoom parameter λ. The motion of a frontal view $\lambda = 1$, $t = 1$, $\phi = \psi = 0$ to a slanted view corresponds to the image deformation $u(x, y) \to u(A(x, y))$ given by (8.2).

8.2 A-SIFT : combining simulation and normalization

The idea of combining simulation and normalization is the main successful ingredient of the SIFT method. This method normalizes rotations and translations, but simulates all zooms out of the query and of the search images. Because of the feature, it is the only fully scale invariant method.

A-SIFT simulates with enough accuracy *all* distortions caused by a variation of the direction of the optical axis of a camera (two parameters). Then it normalizes the other four by the SIFT method, or any other method that is rotation, translation, and scale invariant. More specifically, the method proceeds by the following steps. (See Fig. 8.5.)

A-SIFT algorithm

1. Each image is transformed by simulating all possible affine distortions caused by the change of orientation of the camera axis of camera from a frontal position. These distortions depend upon two parameters: the longitude ϕ and the latitude θ. The images undergo ϕ-rotations followed by tilts with parameter $t = |\frac{1}{\cos\theta}|$ (a tilt by t in the direction of x is the operation $u(x, y) \to u(tx, y)$). For digital images, the tilt is performed as t-subsampling, and therefore requires the previous application of an antialiasing filter in the direction of x, namely the convolution by a gaussian with standard deviation $c\sqrt{t^2 - 1}$. For good antialiasing, $c \simeq 0.8$, see Chapter 4.2.

2. These rotations and tilts are performed for a finite and small number of latitudes and longitudes, the sampling steps of these parameters ensuring that the simulated images keep close to any other possible view generated by other values of ϕ and θ.

3. All simulated images are compared by SIFT.

4. To be more specific, the latitudes θ are such that the associated tilts follow a geometric series 1, a, a^2, , ..., a^n, with $a > 1$. The choice $a = \sqrt{2}$ is a good compromise between accuracy and sparsity. The value n can go up to 6 or more, if the tilts are simulated on the query and the searched image, and up to 10 and more if the tilts are simulated on one image only. That way, transition tilts going up to 64 and more can be explored.

5. The longitudes ϕ are for each tilt an arithmetic series 0, b/t, ..., kb/t, where $b \simeq 72^\circ$ seems again a good compromise, and k is the last integer such that $kb/t < 180^\circ$.