6. Differential structure of images

"*If I had more time, I would have written you a shorter letter*", Pascal (1623-1662)

6.1 The differential structure of images

In this chapter we will study the differential structure of discrete images in detail. This is the structure described by the local multi-scale derivatives of the image. We start with the development of a toolkit for the definitions of heightlines, local coordinate systems and independence of our choice of coordinates.

```
<< FrontEndVision`FEV`; Off[General::spell];
Show[Import["Spiral CT abdomen.jpg"], ImageSize -> 170];
```

Figure 6.1 An example of a need for segmentation: 3D rendering of a spiral CT acquisition of the abdomen of a patient with Leriche's syndrome (EuroRAD case #745, authors R. Brillo, A. Napoli, S. Vagnarelli, M. Vendola, M. Benedetti Valentini, 2000, www.eurorad.org).

We will use the tools of *differential geometry*, a field designed for the structural description of space and the lines, curves, surfaces etc. (a collection known as *manifolds)* that live there.

We develop strategies for the generation of formulas for the detection of particular *features*, that detect special, semantically circumscribed, local meaningful structures (or properties) in the image. Examples are edges, corners, T-junctions, monkey-saddles and many more. We develop operational detectors in *Mathematica* for all features described.

One can discriminate local and multi-local methods in image analysis. We specifically discuss here local methods, at a particular local neighborhood (pixel). In later chapters we look at multi-local methods, and enter the realm of how to connect local features, both by studying similarity in properties with neighboring pixels ('perceptual grouping'), relations over scale ('deep structure') and relations given by a particular *model*. We will discuss the use of the local features developed in this chapter into 'geometric reasoning'.

Why do we need to go in detail about local image derivatives? Combinations of derivatives into expressions give nice feature detectors in images. It is well known that $\sqrt{\left(\frac{\partial L}{\partial x}\right)^2 + \left(\frac{\partial L}{\partial y}\right)^2}$ is a good edge detector, and $\left(\frac{\partial L}{\partial y}\right)^2 \frac{\partial^2 L}{\partial x^2} - 2\,\frac{\partial L}{\partial x}\,\frac{\partial L}{\partial y}\,\frac{\partial^2 L}{\partial x\,\partial y} + \left(\frac{\partial L}{\partial x}\right)^2 \frac{\partial^2 L}{\partial y^2}$ is a good corner detector. But how do we come to such formula's? We can make an infinite number of such expressions. What constraints can/should we impose to come to a reasonably small set of *basis* descriptors? Is there such a *basis*? It turns out there is, and in this chapter we will derive a formal complete set of such descriptive elements.

A very important constraint in the development of tools for the description of image structure is to be independent of the choice of coordinates. We will discuss coordinate transformations, like translations, rotations, zooming, in order to find a way to detect features *invariant* to such coordinate transformations. In fact, we will discuss three 'languages' in which it is easy to develop a general strategy to come up with quite complex image structure detectors:

gauge coordinates, Cartesian tensors, and algebraic polynomial invariants. All these methods have firm roots in mathematics, specifically differential geometry, and form an ideal substrate for the true understanding of image structure.

We denote the function that describes our landscape (the image) with $L(x, y)$ throughout this book, where L is the physical property measured in the image. Examples of L are luminance, T1 or T2 relaxation time (for MRI images), linear X-ray absorption coefficient (for CT images), depth (for range images) etc. In fact, it can be any scalar value. The coordinates x, y are discrete in our case, and denote the locations of the pixel. If the image is 3-dimensional, e.g. a stack of images from an MRI or CT scanner, we write $L(x, y, z)$. A scale-space of images, observed at a range of scales σ is written as $L(x, y; \sigma)$. We write a semicolon as separator to highlight the fact that σ is *not* just another spatial variable. If images are a function of time as well, we write e.g. $L(x, y, z; t)$ where t is the time parameter. In chapter 17 we will develop scale-space theory for images sampled over time. In chapter 15 we study the extra dimension of color in images and derive differential features in color-space, and in chapter 13 we derive methods for the extraction of motion, a vectorial property with a magnitude and a direction. We firstly focus on static, spatial images.

6.2 Isophotes and flowlines

Lines in the image connecting points of equal intensity are called *isophotes*. They are the heightlines of the intensity landscape when we consider the intensity as 'height'. Isophotes in 2D images are curves, and in 3D surfaces, connecting points with equal luminance.

(Greek: isos ($\iota\sigma o\varsigma$) = equal, photos ($\phi o\tau o\varsigma$) = light): $L(x, y) = \text{constant}$ or $L(x, y, z) = \text{constant}$. This definition however is for a continuous function. But the scale-space paradigm solves this: in discrete images isophotes exist because these are *observed* images, and thus *continuous* (which means: infinitely differentiable, or C^∞). Lines of constant value in 2D are `Contours` in *Mathematica*, which can be plotted with `ContourPlot`. Figure 6.2 illustrates this for a blurred version of a 2D image.

```
im = Import["mr128.gif"][[1, 1]];
Block[{$DisplayFunction = Identity, dp, cp},
 dp = ListDensityPlot[gD[im, 0, 0, #]] & /@ {1, 2, 3};
 cp = ListContourPlot[gD[im, 0, 0, #],
     ContourStyle → List /@ Hue /@ (.1 Range[10])] & /@ {1, 2, 3};
 pa = MapThread[Show, {dp, cp}]]; Show[GraphicsArray[pa],
 ImageSize -> 400];
```

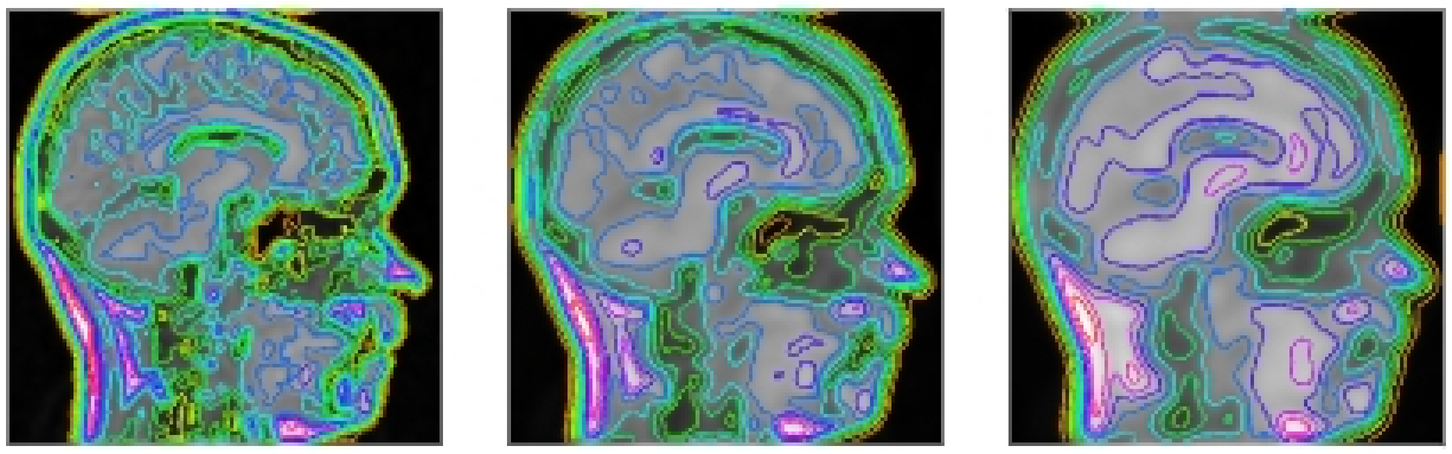

Figure 6.2 Isophotes of an image at various blurring scales: from left to right: $\sigma = 1$, $\sigma = 2$ and $\sigma = 3$ pixels. Image resolution 128^2. Ten isophotes are plotted in each image, equidistant over the available intensity range. Each is shown in a different color, superimposed over the grayvalues. Notice that the isophotes get more 'rounded' when we blur the image. When we consider the intensity distribution of a 2D image as a landscape, where the height is given by the intensity, isophotes are the heightlines.

Isophotes are important elements of an image. In principle, all isophotes together contain the same information as the image itself. The famous and often surprisingly good working segmentation method by thresholding and separating the image in pixels lying within or without the isophote at the threshold luminance is an example of an important application of isophotes. Isophotes have the following properties:

- isophotes are closed curves. Most (but not all, see below) isophotes in 2D images are a so-called Jordan curve: a non-self-intersecting planar curve topologically equivalent to a circle;
- isophotes can intersect themselves. These are the critical isophotes. These always go through a saddlepoint;
- isophotes do not intersect other isophotes;
- any planar curve is completely described by its *curvature*, and so are isophotes. We will define and derive the expression for isophote curvature in the next section.

- isophote shape is independent of grayscale transformations, such as changing the contrast or brightness of an image.

A special class of isophotes is formed by those isophotes that go through a *singularity* in the intensity landscape, thus through a minimum, maximum or saddle point. At these places the intensity landscape is horizontal, the local spatial derivatives are all zero. Only at saddle points isophotes intersect themselves, and just above and below this intersection its neighbor