Nonlinear Diffusion

These notes summarize the way I present this material, for my benefit. But everything in here is said in more detail, and better, in Weickert's paper.

1 Introduction: Motivation for non-standard diffusion

First, let's begin with some definitions and examples of why we want to use other kinds of diffusion beyond Gaussian smoothing.

We have considered the case of a noisy step edge, and noticed that Gaussian smoothing smooths across the edge. Gaussian smoothing makes sense away from the edge, but we might want to do something different around the edge. If smoothing is not the same everywhere, we can call this *non-homogenous*. We want to do this so smoothing is influenced by local image properties.

We might want smoothing to depend on the smoothed image, not the original image. That is, if smoothing eliminates some unimportant structure, that structure should not continue to affect the smoothing. When the smoothing depends on the current, diffused image, though, we get a nonlinear equation for the smoothing, which we call non-linear diffusion.

Finally, we may also want the direction of the flux to depend on image properties. If the intensities are linear (form a ramp) we want the flux to be in the direction in which the ramp is going down, which is the direction of the gradient. This is essentially a 1D problem, with the same direction we had in 1D. If the flux is always in the direction of the gradient, we call the diffusion isotropic. However, near a step edge, we might want the flux to be parallel to the edge, which is perpendicular to the gradient. If the direction of the flux varies in the image, we call it *anisotropic*.

2 Diffusion in 2D

We discussed homogenous, isotropic diffusion in 1D because it is essentially the same in 1D and 2D. Anisotropic diffusion only makes sense in 2D, though, so we need to move to a 2D discussion of diffusion. We will use notation consistent with Weickert's article, so:

$f(x)$ is the density at time 0 (ie the image). $u(x,t)$ is the density at position x and time t.

In 1D homogenous, isotropic diffusion, the equation for flux is:

$$j(x,t) = -D\frac{\partial u}{\partial x}.$$

In 2D, the equation for flux becomes:

$$j = -D\nabla u$$

Here, the flux, j, will be a vector, indicating that the flux has a magnitude and direction. If D is a scalar, then the flux is in the direction of the gradient of u, and we have isotropic diffusion. If D is a constant scalar, then the flow is also homogenous, and equivalent to Gaussian smoothing. Note that this is exactly analogous to the 1D case. However, D can also be a tensor (2x2 matrix) which maps the gradient to a different direction and magnitude. When D is a spatially varying tensor that depends on u, then the direction of the flux can depend on local image properties (as well as its magnitude, of course).

In 1D we have the equation:

$$\frac{\partial u}{\partial t} = -\frac{\partial j}{\partial x}$$

This is essentially expressing the conservation of mass. It says that varying flux deposits excess mass. In 2D, we have the equation:

$$\frac{\partial u}{\partial t} = -\mathrm{div} j.$$

where div is the divergence. That is,

$$\mathrm{div} j = \frac{\partial j}{\partial x} + \frac{\partial j}{\partial y}$$

This says that the change in concentration is proportional to the divergence of the gradient field. Divergence is just measuring the way the gradient is piling up at a point. For example, if the gradient is constant, the divergence is 0.

Finally, in 1D we had the diffusion equation:

$$\frac{\partial u}{\partial t} = D\frac{\partial^2 u}{\partial x^2}$$

In 2D the diffusion equation becomes:

$$\frac{\partial u}{\partial t} = \mathrm{div}(D\nabla u)$$

3 Non-linear diffusion - Perona-Malik diffusion

If we stick with isotropic diffusion, we cannot regulate the direction of the diffusion (so we actually could consider this in 1D) we only regulate the amount. It makes sense to have less diffusion where the image is changing fast, that is, near edge-like things. We can do this with:

$$\frac{\partial u}{\partial t} = \mathrm{div}(g(x)\nabla u)$$

Here we write D as $g(x)$ to indicate that it is a scalar function of image position. That is, $g(x)$ is a spatially varying diffusion. If $g(x)$ only depends on the initial image, then it is effectively constant and so we get a linear PDE. A natural choice would be a function that decreases with $\|\nabla f\|$. One example in use is:

$$g(\|\nabla f\|^2) = \frac{1}{\sqrt{1 + \frac{\|\nabla f\|^2}{\lambda^2}}}$$

for some constant $\lambda > 0$. Notice this decreases from 1 to 0 as $\|\nabla f\|$ grows. When $\|\nabla f\| = \lambda$, $g = \frac{1}{2}$.

As mentioned above, this produces artifacts, because even after a small structure disappears, it continues to influence the image. It also seems unpleasing, because if two points have the same $\|\nabla f\|$ they will always experience the same amount of smoothing, even if after some smoothing one has a derivative of 0, and the other still has a high derivative. This might happen with a ramp edge with noise wiggles added.

So instead, we make g depend on the current, partially smoothed image, not the initial image. This whole approach was introduced by Perona-Malik, who suggested:

$$g(\|\nabla u\|^2) = \frac{1}{\sqrt{1 + \frac{\|\nabla u\|^2}{\lambda^2}}}$$

This means that as we smooth, we modulate the amount of smoothing at each location by the current magnitude of the gradient.

4 Instability and artifacts

This works well in practice. Unfortunately, when we analyze Perona-Malik, we find that it is an ill-posed PDE. That means the solutions may not be unique, and they can be unstable (that is, change a lot with small changes in the initial conditions). The way to fix this is with regularization, which means, some kind of smoothing. This is why it works pretty well in practice; discretization acts as a kind of regularization, because it is like approximating the function with a piecewise constant function, which is smoother. This can produce weird results, though, such as worse artifacts when we discretize more. A more principled way to fix this is with

$$g(\|\nabla u_\sigma\|^2) = \frac{1}{\sqrt{1 + \frac{\|\nabla u_\sigma\|^2}{\lambda^2}}}$$

where u_σ denotes the density smoothed with a Gaussian. That is, we could write: $u_\sigma = K_\sigma * u$.

5 Anisotropic Diffusion

As mentioned before, we may want to diffuse in different directions in different parts of the image. When there is an edge, we could diffuse orthogonal to it. So we want to construct the diffusion tensor, D, so it smooths more in the direction where the image is coherent (ie, there is not so much variation in intensity) and less in direction where it is not coherent. We can do this in the following way:

First, we denote the eigenvectors of D to be v_1 and v_2. We construct D so that v_1 is parallel to ∇u_σ (and of course, v_2 is perpendicular to it). We denote the corresponding eigenvalues of D to be $\lambda_1 = g(\|\nabla u_\sigma\|^2)$ and λ_2, with $\lambda_2 = 1$. We apply this D in the standard diffusion equation:

$$\frac{\partial u}{\partial t} = \operatorname{div}(D\nabla u)$$

Let's unpack this equation a little bit. First of all, notice that if the direction of ∇u_σ is the same as the direction of ∇u, then this diffusion is the same as Perona-Malik diffusion. In general, ∇u

has components in v_1 and v_2 which get scaled by λ_1 and λ_2. If ∇u only has a component in the direction of v_1, this gets scaled by λ_1, but the direction of $D\nabla u$ is the same as the direction of ∇u. The key thing about this method is that sometimes the flux is *not* in the direction of ∇u. As an example, suppose ∇u_σ has a very large magnitude, and a direction different from ∇u. Then λ_1 will be near 0 and the component of ∇_u that is in the direction of ∇u_σ will be wiped out by D, but the orthogonal component of ∇_u will be preserved, and there will be flux in that direction.

So this method relies on the image gradient having different directions at different scales. The idea is that at a larger scale, ∇u_σ is determined by the structure in the scene. At small scale, ∇u reflects noise that needs to be smoothed, which can be quite different.

To summarize some cases:

- If the intensity variation is the same in all directions, the diffusion tensor will have two arbitrary eigenvectors with equal eigenvalues. This means D reduces to a scalar, and diffusion is isotropic.
- If the direction of the gradient is also the direction of greatest variation, then this is one D's eigenvectors. That means that the smoothing will still occur in the direction of the gradient.
- The gradient is computed on a smaller scale than the intensity variation. If the two are unrelated, so the gradient direction is at a 45 degree angle to the direction of greatest variation, then: If the variation in intensity is almost the same in both directions, smoothing occurs in direction of gradient; as coherence increases, smoothing increases in coherence direction.

Why is it done this way? The direction of flux is never perpendicular to the direction of gradient. In fact it always must have a component in the opposite of the direction of gradient, and no component in the direction of gradient. These properties follow from the fact that the diffusion tensor is positive definite, which is needed to ensure good properties of the diffusion, such as having a unique solution.

We can see the importance of making D positive definite with a simple example. Suppose D had an eigenvector in the direction of the gradient, but this had a negative eigenvalue. This would allow us to construct a diffusion that is the inverse of ordinary diffusion, and that undoes it effects. So, with ordinary diffusion, we can take very different images and smooth them until they are almost constant, and all look nearly the same. If we applied the inverse process, we could start with nearly identical, smooth images, and produce images that look radically different. This means that inverting diffusion is a very unstable process, and can't be computed effectively.

6 The Beltrami View

Sochen et al. give another way to formulate anisotropic diffusion, which seems very intuitive. Recall that we can think of ordinary diffusion as the process of replacing each intensity with a weighted average of neighboring intensities, where each neighboring intensity is weighted by the Gaussian of its distance to the pixel. In the Beltrami framework, we do exactly the same thing, except we think of an image as a 2D surface in 3D. Each point, (x, y), with intensity $u(x, y)$ is mapped to the 3D point $(x, y, Cu(x, y))$. C is a constant, that indicates how to weight intensity vs. spatial coordinates. Then, we take a weighted average, using the geodesic distance in this space, rather than the Euclidean distance in (x, y) space.

Suppose, for example, we have two nearby points on opposite sides of a step edge. The geodesic distance between them will be very big, because it depends on the change in intensity as well as the change in (x,y) coordinates. This means that a point near an edge is updated by the average of points that are mostly on the same side of the edge.

This produces anisotropy in a very natural way. In the direction where there is little variation in intensity, more pixels are nearby in terms of geodesic distance, so more pixels are used in the average.

Notice that the parameter C controls the degree of anisotropy. If C is small, this reduces to Gaussian smoothing. As C grows, the effect of intensity variations is magnified. In the extreme case, as C goes to infinity, only intensity variations matter. Consider the 1D case. Any point on a curve that is not an extrema has a neighborhood of larger intensity values on one side, and an identical but negative neighborhood on the other side. So the only points affected by this smoothing are the extrema, which move to become less extreme. This evolves piecewise constant regions, where the amount of variation between extrema is constantly reduced. This is related to total variation diffusion, because it reduces the total variation of the curve.

7 Non-linear Diffusion and Segmentation

Note that non-linear diffusion tends to produce a kind of segmentation. We get regions that are piecewise constant, or nearly so, in the course of the diffusion.