Curvature regularization for resolution-independent images

John MacCormick[1] and Andrew Fitzgibbon[2]

[1] Department of Computer Science, Dickinson College, USA
[2] Microsoft Research, Cambridge, UK

Abstract. A resolution-independent image models the true intensity function underlying a standard image of discrete pixels. Previous work on resolution-independent images demonstrated their efficacy, primarily by employing regularizers that penalize discontinuity. This paper extends the approach by permitting the *curvature* of resolution-independent images to be regularized. The main theoretical contribution is a generalization of the well-known elastica energy for regularizing curvature. Experiments demonstrate that (i) incorporating curvature improves the quality of resolution-independent images, and (ii) the resulting images compare favorably with another state-of-the-art curvature regularization technique.

Keywords: curvature; elastica; regularization

1 Introduction and related work

Viola *et al.* [19, 20] introduced the notion of a *resolution-independent* latent image to model the true intensity function underlying a standard image of discrete pixels. Figure 1 gives an example of the approach: the true intensity function is approximated by a piecewise linear function u, whose linear patches are defined on a triangle mesh. The crucial feature is that the mesh's vertices are positioned with arbitrary precision, which frees the model from any notion of discrete pixels. The vertex positions and patch intensities are determined by minimizing an energy that includes a regularizer term, which models the prior expectations of resolution-independent images in general.

Previous work on resolution-independent images employed a regularizer based primarily on the *discontinuities* in u. The main contribution in this paper is to extend the regularizer to incorporate the *curvature* of u. Starting from the well-known *elastica* energy [2], we derive explicit expressions for computing the elastica energy on the smooth and non-smooth regions of the image domain. The non-smooth region includes *steps* and *corners* (defined rigorously later), leading to separate step energy and corner energy terms in the energy functional. The paper also includes practical experiments demonstrating the benefits of the approach and a favorable comparison with another state-of-the-art curvature regularizer.

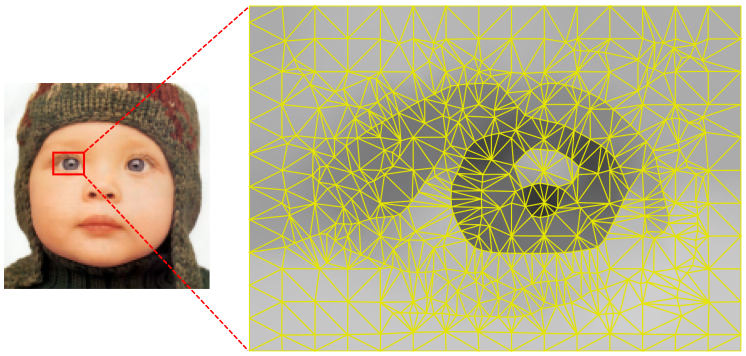


Fig. 1. In this paper, resolution-independent images are modeled as in Viola *et al.* [20], employing a piecewise linear intensity function defined on a triangle mesh whose vertices are positioned with arbitrary precision.

There is a considerable amount of related work on curvature regularization, including a long history of variational and level set methods (e.g. [9, 12, 16]), methods derived from the elastica energy (e.g. [2, 10]), and other approaches such as total curvature [4] and Gaussian derivatives [5]. The work of Schoenemann *et al.* [15] and Strandmark and Kahl [17] is most closely related to the present paper. These approaches regularize curvature based on a fixed [15] or adaptive [17] mesh, employing linear programming relaxations for optimization. However, the meshes are restricted to a fixed small set of edge angles, so lines not at those orientations must be jagged. In our work, all angles are equally treated (ignoring floating point issues). The curvature term contrasts with this paper in that it applies to binary images and to corners with exactly two prongs (as defined in Section 4.1); the approach here permits resolution-independent images with arbitrary intensities and multi-pronged corners. Hence, we believe the novel theoretical contribution of the paper is twofold: first, the well-known approach of regularizing curvature by minimizing an elastica energy is reformulated so that it can be applied explicitly to resolution-independent images (Sections 3 and 4); second, this reformulation leads to a corner energy that has not, to our knowledge, been studied previously (Section 4.1).

2 The set of resolution-independent images

At the core of our approach is the concept of a *resolution-independent image*. Formal mathematical definitions are given in our technical report [6]. Here, we rely primarily on intuition to convey the essential concepts. A resolution-independent image is produced by an idealized camera with infinite resolution, infinite color depth, infinite depth of field, and zero noise. The resulting image u is defined on a connected subset Ω of $\mathbb{R}^2$, with intensities in the continuous range $[0, 1]$. We assume the world consists of piecewise smoothly-varying objects, giving rise

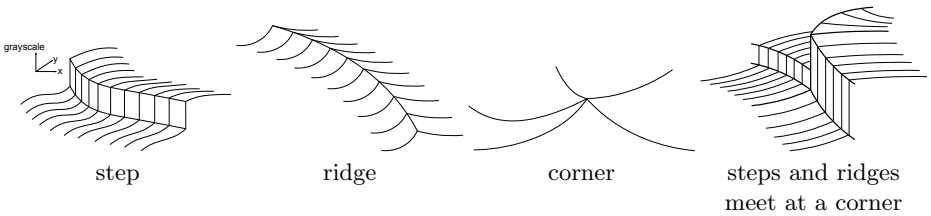


Fig. 2. Taxonomy of the jump set of a resolution-independent image. Each panel shows part of a 2D grayscale resolution-independent image, visualized as a surface. The image plane coincides with the horizontal x-y plane, and the grayscale intensity of the image is plotted on the vertical z axis, as indicated by the set of axes on the left.

to resolution-independent images that are also piecewise smooth. So Ω is partitioned into a *differentiable* region D (where u is continuously differentiable) and its complement J, termed the *jump set* (where u or its derivative is discontinuous). As shown in Figure 2, it is useful to further partition J into *steps*, *ridges*, and *corners*—so $\Omega = D \cup J_{\text{step}} \cup J_{\text{ridge}} \cup J_{\text{corner}}$. Some mild assumptions [6] guarantee that u is well-behaved near the jump set. In particular, $\lim_{x \to x_0} u(x)$ exists for any $x_0 \in J$, and is independent of the path used to approach x_0, provided the path remains in the differentiable region D.

3 Regularizers for resolution-independent images

We are interested in imposing a prior on resolution-independent images u. This will be done via a real-valued regularizer $E(u)$, with the usual interpretation of E as an energy functional, so that u-functions with low values of E have high prior probability.

The *elastica energy* is a commonly-used regularizer for curvature in computer vision applications. The one-dimensional version of this energy, derived from the physical energy required to bend a thin pliable rod into a given smooth shape, was considered as early as 1744 by Euler ([3]; see this paper's appendix for details). For a smooth curve Γ parameterized by arc length s, it is given by

$$E_{\text{1D-elastica}}(\Gamma) = \int_s (a + b\kappa(\Gamma, s)^p)\, ds. \tag{1}$$

Here, $a, b, p \geq 0$ are constants and $\kappa(\Gamma, s)$ is the (unsigned) curvature of Γ at s, as defined in elementary geometry. Physics (and Euler) say that $p = 2$, but other values may give good results in computer vision applications.

Of more direct interest here is the generalization of elastica energy to two dimensions, as proposed by Masnou and Morel [8], and employed by many others (e.g. [15]). This two-dimensional elastica energy is given by

$$E_{\text{elastica}}(u) = \int_{\mathbf{x} \in \Omega} (a + b\kappa_{\text{LL}}(\mathbf{x})^p) |\nabla u(\mathbf{x})|\, d\mathbf{x}. \tag{2}$$

Here $\kappa_{\mathrm{LL}}(\mathbf{x})$ is the (unsigned, 1D) curvature of the level line of u passing through $\mathbf{x} \in \Omega$. Chan *et al.* [2] provide a detailed and illuminating derivation of the 2D elastica energy (2) from the 1D pliable-rod definition (1). The basic idea is to integrate the 1D version over levels l; the extra weight of $|\nabla u(\mathbf{x})|$ in (2) then appears as the Jacobian when transforming from height and arc-length parameters (l, s) to image plane parameters $\mathbf{x} = (x, y)$.

As with the one-dimensional elastica energy, the two-dimensional energy (2) has an intuitive physical interpretation: it is the total amount of energy that would be expended to build the image out of thin, horizontal, pliable rods, assuming the energy of each individual rod is given by Equation (1) multiplied by the height spacing δl between rods. Note that for this physical analogy to be appropriate, the rods must be horizontal (so that they correspond to level sets), and they should be placed at equally-spaced heights separated by δl. As we will be repeatedly appealing to this physical interpretation of the elastica energy later, let us call it the *pliable rod analogy*. As our first practical example, the next subsection calculates the elastica energy of a step in the image.

3.1 Computation of the step contribution

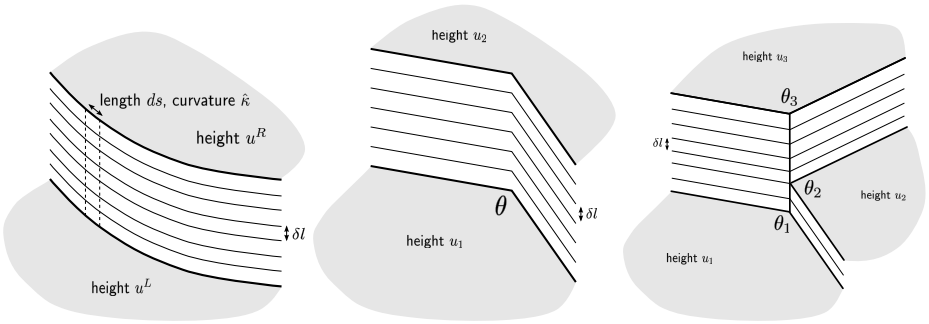
Consider a small portion ds of J_{step} shown in Figure 3(a), where the portion is small enough that we can approximate u^L and u^R as constant. To build this part of the image requires stacking horizontal rods directly on top of each other. Each individual rod has energy $\delta l(a + b\hat{\kappa}^p)\, ds$, by definition. The total height of the stack is just $|u^L - u^R|$, so the contribution of this stack is $(a + b\hat{\kappa}^p)|u^L - u^R|\, ds$. Integrating over all elements of the step set, this is equivalent to stating that the contribution of the entire step set to the elastica energy is

$$\int_{J_{\mathrm{step}}} (a + b\hat{\kappa}^p)|u^L - u^R|\, ds. \tag{3}$$

Obviously, the above argument is based on physical intuition rather than mathematical rigor, which may trouble some readers. In this particular case, it is relatively easy to give a more rigorous calculation, based on smoothing u with a small unit-volume kernel, applying the definition of elastica energy (2) that is valid for smooth u, then taking the limit as the width of the kernel tends to zero. However, we prefer the approach based on physical intuition because it is easier to understand, and our final goal does not require mathematical rigor. We need to construct an energy whose minimization results in pleasing resolution-independent images; constructing that energy via plausible physical reasoning is a perfectly acceptable approach. Hence, in the remainder of the paper, we will appeal to physical intuition whenever necessary without attempting to inject additional rigor.

4 Curvature-related extensions of the elastica energy

This section describes the main theoretical contributions of the paper. It first gives details of how to compute the contribution to the elastica energy due to



(a) step (b) 2-prong corner (c) 3-prong corner

Fig. 3. Computing the elastica energy of steps and corners.

corners (Sections 4.1). Section 4.2 then unifies the preceding calculations into a single generalized elastica energy. Finally, Section 4.3 describes the variant of the generalized elastica energy appropriate for the triangle meshes used in the present paper. To the best of our knowledge, all three subsections present primarily novel material.

4.1 Computation of the corner contribution

For the reasons given in Section 3.1, we use the pliable rod model to compute the energy of a sharp corner point, such as the one in the right-most panel of Figure 2. It can be shown [6] that every corner lies at the endpoint of two or more *prongs*: smooth curves in the jump set. The right-most panel of Figure 2 has five prongs, for example. The image intensity is, by definition, smooth between prongs. Some mild additional assumptions (essentially Lipschitz conditions, as described in the technical report [6]) guarantee that the intensity tends to sensible limits as we approach the prongs and the corner point itself. Thus, by considering a sufficiently small neighborhood of the corner, the intensity function can be approximated arbitrarily well by using prongs that are straight lines and an intensity that is constant between prongs.

Let us first examine the simplest possible case: a two-prong corner (see Figure 3(b)), whose prongs meet with interior angle θ, with constant intensity values u_1 and u_2 on the inside and outside of the corner respectively. How would we build this geometric shape using horizontal pliable rods? As shown in Figure 3(b), each rod must be bent through angle $|\pi - \theta|$, and the rods are stacked vertically (using our standard vertical spacing, δl).

Here we encounter an apparent difficulty: the construction requires sharp corners in the rods, but this leads to infinite energies if we insist on an elastica energy of the form (1). Two easy solutions to suggest themselves. One solution is to take $p = 1$. In this particular case, the energy remains finite (and is easily seen to be $b|\pi - \theta|$). The other solution is to adopt a more general physical model of our rods: simply declare that the rods are made of some material that

can be bent into a sharp corner using finite energy. For example, this energy could be $b|\pi - \theta|^p$ for some p, or the energy could also incorporate robustness by employing, say, $b\min(\tau, |\pi - \theta|^p)$ for some threshold τ. Later experiments use the non-robust version, which performs well for our applications.

Adopting this (literally) more flexible definition of a rod, the energy of each rod is $b|\pi - \theta|^p\,\delta l$, and the total height of the stack of rods is $|u_2 - u_1|$. Integrating over l, we obtain the total corner energy for a two-pronged corner as $b|\pi - \theta|^p|u_2 - u_1|$.

Now let us turn to the general case of a multi-pronged corner, with N prongs. (See Figure 3(c) for a 3-prong example.) For concreteness, label the prongs from 1 to N in an anti-clockwise direction. As before, we may assume the prongs are straight lines and the image takes on constant values $u_1, u_2, \ldots, u_N$ on the *wedges* between each consecutive pair of prongs. So u_1 is the value of the image on wedge 1 between prongs 1 and 2, and so on up to u_N, which is the value on wedge N between prong N and prong 1. The angle of wedge i is θ_i. In what follows, subscripts are computed modulo N. In particular, u_{N+1} means the same thing as u_1, and similarly for θ_{N+1}.

We can think of this simplified geometry as a circular pie cut into wedges, where each wedge happens to be of a different height and angle. We need to calculate the energy required to build this multi-level pie out of pliable rods, as in Figure 3(c). One simple approach uses recursion: find the lowest wedge, and build up the sides of the wedge to the height of the lowest adjacent wedge. At this point, the lowest wedge has effectively been removed from the structure, and the problem has been reduced to building a new pie that possesses one fewer wedge. The recursion can bottom out at two wedges, which is the two-prong case considered above. Alternatively, we can make our final formula (4) a little more elegant by bottoming out at one wedge, which is a degenerate "corner" of zero energy.

More formally, let i^* be the index of the lowest wedge, so $i^* = \arg\min_{i\in\{1,\ldots,N\}} u_i$. Let j^* be the index of the lowest wedge adjacent to i^*, so $j^* = \arg\min_{i\in\{i^*-1,i^*+1\}} u_i$. Let $C = (u_i, \theta_i)_{i=1}^N$ denote the N-prong corner, and $E_{\text{cnr}}(C)$ the desired elastica energy of this corner. Write $\hat{C}$ for the $(N-1)$-prong corner that results from filling in wedge i^* up to height u_{j^*}. Then compute $E_{\text{cnr}}(C)$ recursively according to

$$E_{\text{cnr}}(C) = \begin{cases} 0 & \text{if } N = 1, \\ E_{\text{cnr}}(\hat{C}) + b|\pi - \theta_{i^*}|^p|u_{i^*} - u_{j^*}| & \text{if } N > 1. \end{cases} \tag{4}$$

Later, we will consider a generalization of this formula in which $|\cdot|$ is replaced by a robust function $\rho_\tau(\cdot) \equiv \min(\tau, |\cdot|)$ and raised to a power α. We also make explicit the dependence on parameters b, p by writing

$$E_{\text{cnr}}(C; b, p, \alpha) = \begin{cases} 0 & \text{if } N = 1, \\ E_{\text{cnr}}(\hat{C}; b, p, \alpha, \tau) & \\ \quad + b|\pi - \theta_{i^*}|^p \rho_\tau(u_{i^*} - u_{j^*})^\alpha & \text{if } N > 1. \end{cases} \tag{5}$$

4.2 A generalized elastica energy for all resolution-independent images

The elastica energy (2) decomposes into integrals over four regions: differentiable (D), step (J_{step}), ridge (J_{ridge}), and corner (J_{corner}). Our technical report [6] gives details for J_{ridge}, and this term is omitted here and in the remainder of the paper, since our focus is on the corner energy. Hence, by substituting (3) and (5) into (2), (and making some further generalizations described shortly) we obtain a generalized elastica energy E_G (where "G" stands for "generalized"):

$$\begin{aligned} E_G(u) = \lambda_1 \int_{\mathbf{x}\in D} (a + b\kappa_{\text{LL}}(\mathbf{x})^{p_1})|\nabla u(\mathbf{x})|^{\alpha_1}\, d\mathbf{x} \\ + \lambda_2 \int_{J_{\text{step}}} (a + b\hat{\kappa}^{p_2})|u^L - u^R|^{\alpha_2}\, ds + \lambda_3 \sum_{\text{corners } C} E_{\text{cnr}}(C; b, p_3, \alpha_3) \end{aligned} \tag{6}$$

Here we have allowed an arbitrary exponent α_i for the gradient factor, an arbitrary coefficient λ_i, and an arbitrary curvature exponent p_i in each term. The generalization to arbitrary α_i, λ_i, p_i is not justified by any physical or theoretical reasoning. Rather, we appeal to the fact that we are seeking a regularizer that works well in practice. The generalization is justified if, by generalizing a physically realistic expression to one that is not physically realistic, we can obtain better performance when analyzing real images. As we shall soon see, numerous previous authors have taken exactly the same approach. But it should be noted that the natural (i.e. physically realistic, according to the pliable rod model) values for the α_i, λ_i are all 1, and for the p_i the natural value is 2.

Let us now examine how the generalized elastica energy (6) relates to previous work. By taking $\lambda_1 = \lambda_2 = \alpha_1 = \alpha_2 = 1$, and $\lambda_3 = b = 0$, we recover the total variation [21], up to a constant multiplier. By taking $\lambda_3 = b = 0$, we obtain an expression similar to the regularizer used by Viola *et al.* [20]—which, as previously noted, is the direct inspiration for the present work. Hence, the high-level claim that the present work adds a notion of curvature to Viola *et al.* can now be made more explicit: this paper incorporates the corner energy, by permitting $\lambda_3 \neq 0$ in (6).

4.3 Elastica energy for images on a triangle mesh

We are particularly interested in computing the generalized elastica energy E_G for images that are piecewise linear on a triangle mesh. These images have zero curvature on the interiors of all the triangles, so $\kappa_{\text{LL}} \equiv 0$ on D. Moreover, the mesh edges (which are all straight lines between triangle vertices) have zero curvature too, so $\hat{\kappa} \equiv 0$ on J_{step}. It is easy to see that this renders irrelevant the values of p_1, p_2, a, b in (6). These observations result in a simplified form of the

elastica energy for triangle meshes, E_T (where the "T" stands for "triangle"):

$$\begin{aligned} E_{\mathrm{T}}(u) = \lambda_1 \int_{\mathbf{x}\in D} |\nabla u(\mathbf{x})|^{\alpha_1}\, d\mathbf{x} \\ + \lambda_2 \int_{J_{\mathrm{step}}} |u^L - u^R|^{\alpha_2}\, ds + \lambda_3 \sum_{\text{corners } C} E_{\mathrm{cnr}}(C; b, p_3, \alpha_3, \tau) \end{aligned} \tag{7}$$

Previous work [19] has shown some benefits from taking $\alpha_1 = 2, p_i = 1$. The experiments in this paper also adopt these settings, and set all other constants (λ_i, α_i) to their physically realistic value (1.0), except where stated otherwise. The robustness parameter τ is set to 10% of the dynamic range in the input image.

5 Algorithmic details

The algorithm used here for computing resolution-independent images is modeled closely on Viola's work [19, 20], where the reader can find many additional details. First, we need a data term that expresses the affinity between u and some input image I. This input I is a standard, discrete set of grayscale pixel values denoted I_i. We assume pixel i of I was formed by blurring the true (continuous) intensity function with some kernel κ_i. This leads to a data term $\mathcal{D}(u, I)$ of the form

$$\mathcal{D}(u, I) = \lambda_0 \sum_i \|I_i - \int_\Omega \kappa_i(x) u(x)\, dx\|. \tag{8}$$

Here, λ_0 is the *data gain* expressing the relative importance of the data and regularization terms. Experiments in this paper take $\lambda_0 = 10$ (unless stated otherwise), a value that was determined by trial and error to yield reasonable performance on a variety of inputs. For the norm $\|\cdot\|$, we use the square of the standard Euclidean norm. Ideally, the kernel functions κ_i would be estimated from the point spread function of the camera used to capture I, but this lies outside the scope of the present paper. We take the pragmatic and simple choice of setting κ_i to be a 2D square box function, equal to 1 on the unit square centered at pixel i and zero elsewhere.

The computation of a resolution-independent image $\hat{u}$ is achieved by minimizing the total energy $\mathcal{E}(u, I)$, which combines the triangle mesh energy (7) and data term (8):

$$\hat{u} = \arg\min_u \mathcal{E}(u, I) = \arg\min_u (E_T(u) + \mathcal{D}(u, I)). \tag{9}$$

Recall that u is a piecewise linear triangle mesh, parameterized by: (i) the 2D locations of each vertex in the mesh (two real parameters per vertex); and (ii) the height and slope of each triangle in the mesh (three real parameters per triangle). The average density of the mesh is application-dependent. In experiments reported here, the typical distance between neighboring vertices is 1–3 pixels. Even on modest-sized images, this leads to tens of thousands of triangles and

vertices, and hundreds of thousands of parameters. For example, the 256×256 input of the segmentation result in Figure 7 leads to a mesh with over 42,000 triangles, 21,000 vertices and a resulting total of 170,000 parameters.

All experiments in this paper perform the minimization (9) over these parameters by first initializing the mesh to a reasonable estimate, then applying an off-the-shelf nonlinear optimizer. Specifically, the initialization is done by using a regular grid of vertices spaced 1.5 pixels apart, augmented by further vertices placed at subpixel locations identified as edgels by a Canny edge detector. The intensity values are initialized by assigning each triangle the constant intensity obtained by integrating I over the triangle.

Minimization is performed in Matlab via Schmidt's `minFunc`[3], using the LBFGS algorithm [11] with default options. Note that LBFGS is a quasi-Newton method, requiring the objective function's derivative but not its Hessian. The derivative of non-corner terms is taken from Viola [19]; the derivative of the corner energy (5), although tedious to implement and debug, requires only elementary geometry and calculus.

The experiments described here require hundreds or thousands of iterations to reach convergence (as defined by the default `minFunc` criteria). The approach is thus rather computationally expensive. The computational cost of the experiments reported here, all employing Matlab implementations on a 2012 desktop PC with an Intel Core2 Q9400 CPU, range from several CPU-core-minutes (for the results of Figure 5) to nearly 50 CPU-core-hours (for the results of Figure 7).

As previously stated, the above approach follows Viola in many respects. There are two important differences, however. First, we perform joint optimization over all parameters simultaneously. This contrasts with Viola's approach, which alternates between optimizations over the vertex location variables and the intensity height/slope variables, and also employs so-called *N/Z flip* moves, which make global changes to the mesh structure.

Second, we take a simpler approach to the problem of degenerate triangles—triangles that become excessively narrow slivers as the optimization proceeds. If the mesh contains one or more problematic slivers, we remove a vertex from each sliver, and re-triangulate the resulting hole using a constrained Delauney triangulation [13]. Each new triangle's intensities must then be initialized based on the nearest undisturbed triangle, and the entire minimization restarted. In principle, this could lead to extremely slow convergence. In practice, however, we find that running sliver-removal just once before beginning any minimization is typically sufficient.

Figure 6, discussed in more detail below, demonstrates that our *joint* minimization approach has similar performance to the more elaborate *alternating* approach of previous work. Moreover, the joint approach is simpler to implement and appears to encounter fewer problems with degenerate triangles. (Note that this discussion compares optimization approaches only. So in this experiment, both the joint and alternating approaches incorporated the corner energy, and therefore required the corner energy derivative also.)

[3] Mark Schmidt, `http://www.di.ens.fr/~mschmidt/Software/minFunc.html`, 2012

6 Results

6.1 Qualitative assessment of incorporating curvature

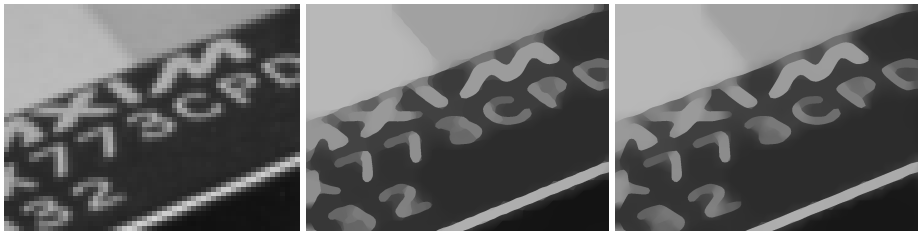
Figure 4 demonstrates the main qualitative result of this paper: incorporating curvature into the energy functional leads to modest improvements in the quality of resolution-independent images. Given the input (a), we run our experiments twice with identical settings—except for the parameter λ_3 of (7) which is switched from 0 (corresponding to "without curvature") to 1 (corresponding to "with curvature") between experimental runs. Panels (b) and (c) are renderings of the resolution-independent images produced by the two runs. To the human eye, panels (b) and (c) outputs appear extremely similar, with excellent reconstructions in some regions (e.g. the *M*, the two 7s, and the *C*) and imperfect ones in others (e.g. the *P* and the *3* have their interiors incorrectly filled).

But as shown in panels (d)–(g), which zoom in on some particular regions of interest, there are some subtle but important differences between the two outputs. (Note that these panels represent extreme super-resolution, showing regions that are 11×7 pixels in the input image.) Specifically, panels (d) and (e) show the letter *C* derived from the input, without and with curvature respectively. In both cases, the curved shape of the *C* has been recovered surprisingly well, albeit imperfectly. More importantly, the output computed with curvature shows some improvement over the without-curvature output: in panel (e), the outline of the *C* represents a smoother curve, and the grayscale values in the interior of the *C* are also smoother. Panels (f) and (g) show a portion of a straight specular edge. Again, the extreme super-resolution performs well in both cases, recovering boundaries that are nearly straight despite the blocky, staircase-like input. And we again see artifacts of the triangle mesh in both outputs: a few triangles with incorrectly-inferred grayscale values protrude from the main strip of high intensity. But the more important point is that panel (g), computed with curvature, produces a straighter boundary for the high-intensity strip, when compared with panel (f) (which was computed without curvature).

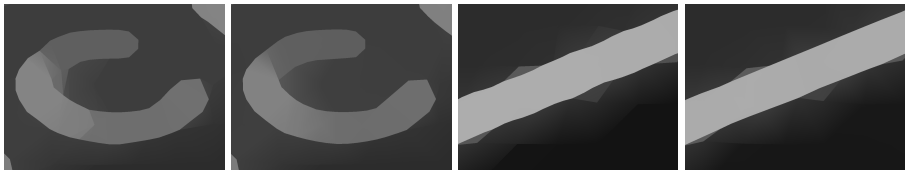
Although we have shown outputs for only one image here, the results are typical. It is reasonable to conclude that incorporating curvature produces modest improvements in the detailed structure of resolution-independent images.

6.2 Quantitative assessment of incorporating curvature

In this subsection, we confirm the previous qualitative results with a quantitative assessment based on peak signal-to-noise ratio (PSNR). The experiment involves the task of simultaneous denoising and upsampling, as shown in Figure 5. The ground truth image in panel (a) is a 64×64 detail of the well-known "peppers" image. Panel (b) is a blurred, noisy version of the ground truth. It was created by first averaging 4×4 blocks of (a), then adding Gaussian noise with standard deviation equal to 5% of the image's dynamic range. This results in a 16×16 image to be used as input to the algorithm for estimating resolution-independent images (Section 5). As with the previous experiment, outputs were produced



(a) 60×46 input image (b) output without curvature energy (c) output with curvature energy



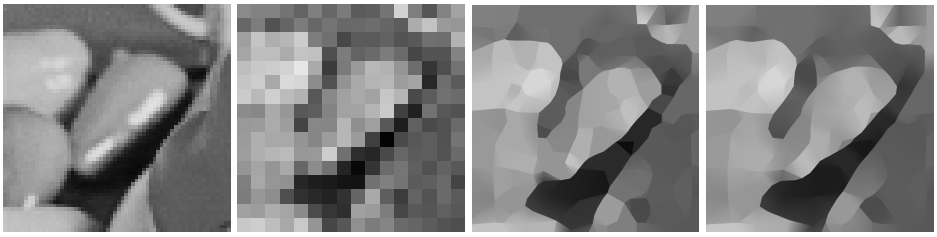
(d) detail without curvature energy (e) detail with curvature energy (f) detail without curvature energy (g) detail with curvature energy

Fig. 4. Comparison of resolution-independent images computed with and without curvature energy.

without and with curvature energy, shown in panels (c) and (d) respectively. A subjective assessment seems to confirm the previous experiment, since the with-curvature result appears to have smoother object boundaries, and smoother grayscale values within objects.

Because we have the ground truth for this experiment, we can also assess these outputs quantitatively, by computing their PSNR with respect to the ground truth. The left panel of Figure 6 shows the results. This graph also demonstrates the sensitivity of the computation to the data gain parameter, λ_0, in Equation (8). The data gain is varied on the horizontal axis, with the corresponding PSNRs for the computations with and without curvature shown on the vertical axis. A higher PSNR corresponds to a higher-quality reconstruction, so it is clear that the with-curvature results are superior to the without-curvature results for each value of the data gain. A further experiment demonstrated that these results are typical, on average. A 64×64 patch was selected uniformly at random from each of the first 40 images of the Berkeley Segmentation Dataset [7], and the above experiment was run with identical settings on all 40 patches. The mean improvement in PSNR after switching on curvature energy was 0.17 dB; further details are in the technical report [6].

As discussed at the end of Section 5, this paper employs a simple joint optimization approach, contrasting with the more elaborate alternating approach of previous work. The right panel of Figure 6 shows the computational expense of these two approaches for the experiment described above (i.e. simultaneously denoising and upsampling the "peppers" image). It is clear that the energy minimization proceeds at roughly the same rate for both approaches, but the



(a) ground truth image (64×64) (b) input image: noisy, subsampled, 16×16 version of (a) (c) output without curvature energy (d) output with curvature energy

Fig. 5. Simultaneous denoising and super-resolution.

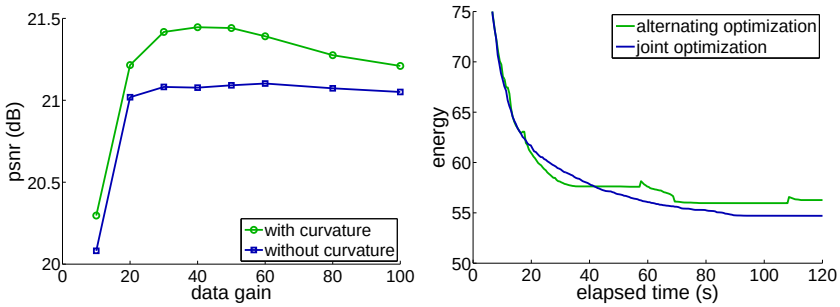


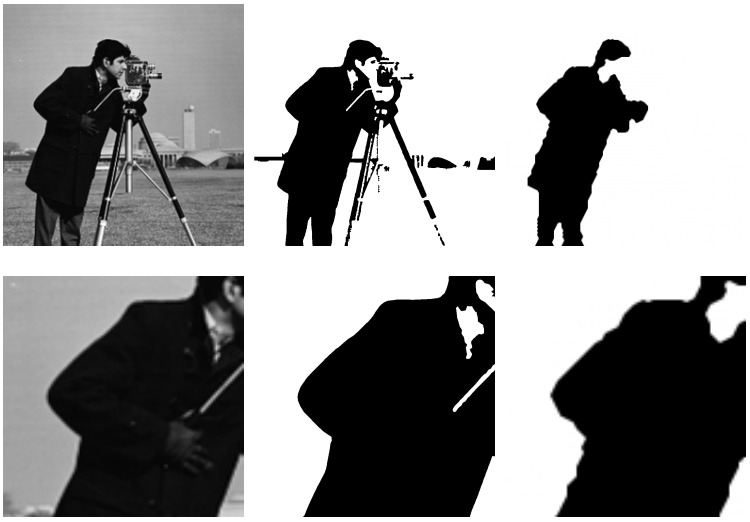
Fig. 6. Left: Estimating a resolution-independent image with curvature energy produces superior PSNR. Right: The simple joint optimization approach of this paper has similar computational expense to the more elaborate alternating approach of previous work.

alternating approach is less smooth since it encounters degenerate triangles more often. The resulting retriangulation can also lead to an increase in the energy value.

6.3 Comparison with alternative curvature regularization

A comprehensive comparison with other curvature regularization techniques is beyond the scope of this paper, since our main objective is to enhance the theory and practice of resolution-independent images. Here, we provide here a comparison with one recent state-of-the-art approach: the technique of Strandmark and Kahl [17]. The Strandmark-Kahl (SK) approach minimizes a particular choice of elastica energy over an adaptive mesh, but the precise form of the energy and the methodology for adapting the mesh are quite different to the present paper.

In addition, the SK approach is targeted at regularizing curvature in *binary* output images, which can therefore be regarded as foreground-background segmentations. So that our results can be compared directly with SK, we obtain a binary image by thresholding a rendering of the resolution-independent output.



input image (256×256) | binarized resolution-independent image | SK output [17]

Fig. 7. Resolution-independent segmentation.

More precisely, given an input image I, we run the minimization algorithm of Section 5 to obtain a resolution-independent image u. Each triangle in the mesh of u is assigned a new constant intensity of 1 or 0 according to whether the triangle's mean intensity is above a given threshold. The resulting u' is a binary, resolution-independent image, and can be rendered at any desired resolution for comparison with other algorithms.

Figure 7 shows the results of this comparison on the well-known "cameraman" image. The bottom row comprises details from the top row, and the final column shows the best SK output on this image. (Here, "best" simply means the most visually pleasing result appearing in SK [17]; the image was kindly provided by the first author of that paper.) Comparing the middle and right columns of Figure 7, we see that the approach of this paper has segmented many more thin, elongated regions than SK. But this difference is of little interest—without specifying a particular application and associated error metric, one cannot say whether it is preferable to include elongated regions in the foreground or not. Of much more interest is the qualitative nature of the cameraman's boundary. We see that this paper's approach yields boundaries that are considerably more smooth and visually appealing than the SK output. This is particularly noticeable on the coat, right arm, and legs.

7 Conclusion

The key contribution of the paper was the derivation of a novel corner energy (5), used to regularize curvature in resolution-independent images modeled by piece-

wise linear triangle meshes. Experiments showed qualitative and quantitative improvements in the accuracy of resolution-independent images inferred using the new curvature regularizer. The technique also compared favorably with a state-of-the-art approach for binary segmentation. The clearest opportunity for future work is to incorporate an energy term for the ridge set. It may also be possible to reduce the computational expense of the approach by employing different techniques for mesh generation (e.g. [1, 14, 18]).

Acknowledgments. The first author is grateful to Microsoft Research, UK (where this work was conducted) for support as a Visiting Researcher. We are indebted to Chris Francese for assistance with translating Euler's Latin. We also thank the two anonymous reviewers who provided useful comments.

References

1. Adams, M.D.: An improved content-adaptive mesh-generation method for image representation. In: Proc. ICIP (2010)
2. Chan, T.F., Kang, S.H., Shen, J.: Euler's elastica and curvature-based inpainting. SIAM Journal on Applied Mathematics 63(2), 564–592 (2002)
3. Euler, L.: Methodus inveniendi lineas curvas maximi minimive proprietate gaudentes. Bousquet (1744)
4. Goldluecke, B., Cremers, D.: Introducing total curvature for image processing. In: Proc. ICCV. pp. 1267–1274 (2011)
5. Koenderink, J.J., van Doorn, A.J.: Surface shape and curvature scales. Image and Vision Computing 10(8), 557–564 (1992)
6. MacCormick, J., Fitzgibbon, A.: Curvature regularization for resolution-independent images. Tech. rep., Dickinson College (2013)
7. Martin, D., Fowlkes, C., Tal, D., Malik, J.: A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In: Proc. ICCV. pp. 416–423 (2001)
8. Masnou, S., Morel, J.M.: Level lines based disocclusion. In: Proc. ICIP. pp. 259–263 (1998)
9. Mitiche, A., Ben Ayed, I.: Variational and Level Set Methods in Image Segmentation. Springer (2011)
10. Mumford, D.: Elastica and computer vision. In: Bajaj, C. (ed.) Algebraic Geometry and Its Applications, pp. 491–506. Springer (1994)
11. Nocedal, J., Wright., S.J.: Numerical Optimization. Springer Verlag (1999)
12. Osher, S., Fedkiw, R.: Level Set Methods and Dynamic Implicit Surfaces. Springer (2003)
13. Paul Chew, L.: Constrained delaunay triangulations. Algorithmica 4, 97–108 (1989)
14. Sarkis, M., Diepold, K.: Content adaptive mesh representation of images using binary space partitions. IEEE Trans. Image Processing 18(5), 1069–1079 (2009)
15. Schoenemann, T., Kahl, F., Cremers, D.: Curvature regularity for region-based image segmentation and inpainting: A linear programming relaxation. In: Proc. ICCV. pp. 17–23 (2009)
16. Sethian, J.A.: Level Set Methods and Fast Marching Methods. Cambridge University Press, 2nd edn. (1999)

17. Strandmark, P., Kahl, F.: Curvature regularization for curves and surfaces in a global optimization framework. In: Proc. EMMCVPR. pp. 205–218 (2011)
18. Vasilescu, M., Terzopoulos, D.: Adaptive meshes and shells: Irregular triangulation, discontinuities, and hierarchical subdivision. In: Proc. CVPR. pp. 829–832 (1992)
19. Viola, F.: Resolution-independent image models. Ph.D. thesis, University of Cambridge (2011)
20. Viola, F., Cipolla, R., Fitzgibbon, A.: A unifying resolution-independent formulation for early vision. In: Proc. CVPR (2012)
21. Ziemer, W.P.: Weakly Differentiable Functions: Sobolev Spaces and Functions of Bounded Variation. Springer (1989)

Appendix: Euler's definition of elastica

Numerous papers cite Euler's work on elastica, but it is surprisingly difficult to track down the relevant excerpt. It appears in Euler's 1744 publication [3], *Methodus inveniendi lineas curvas* ..., Appendix I ("Additamentum I"), paragraph 2 (p247):

> ... ut inter omnes curvus eiusdem longitudinis, qua non solum per puncta A & B transeant, sed etiam in his punctis a rectis positione datis tangantur, definiatur ea in qua sit valor huius expressionis $\int \frac{ds}{RR}$ minimus.

This can be translated as:

> ... that, of all curves of the same length, which not only pass through points A and B, but also are touched at these points by given tangents, it is defined by that in which the value of the expression $\int \frac{ds}{R^2}$ is the smallest.