Error-Bounds on Curvature Estimation

Sven Utcke

Universität Hamburg, Fachbereich Informatik, Arbeitsbereich Kognitive Systeme, Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
`utcke@informatik.uni-hamburg.de`
`http://kogs-www.informatik.uni-hamburg.de/~utcke`

Zusammenfassung. Die Berechnung der Krümmung einer digitalen Kontur an jedem Punkt der Kontur ist eine Häufig benötigte Operation in der Bildverarbeitung, sei es die Berechnung differentieller Invarianten oder Curvature Scale Space. Es ist jedoch seit langem bekannt, dass die Berechnung von Krümmung anfällig gegenüber Rauschen auf der Kontur ist. Wir zeigen, wie und in welchem Maße Rauschen den relativen Fehler der berechneten Kontur beeinflusst. Ein interessantes Resultat ist, dass ganz entgegen der Intuition die Berechnung von Krümmung für nur gering gekrümmte Kontur-Stücke aufgrund der beschränkten Bildgröße unmöglich ist, während es für stärker gekrümmte Konturstücke möglich ist, zu akzeptablen Lösungen zu kommen.

Error-Bounds on Curvature Estimation

Sven Utcke

Universität Hamburg, Fachbereich Informatik, Arbeitsbereich Kognitive Systeme,
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
`utcke@informatik.uni-hamburg.de`
`http://kogs-www.informatik.uni-hamburg.de/~utcke`

Abstract. Estimation of a digital curve's curvature at any given point is needed for many tasks in computer vision, be it differential invariants or curvature scale space. However, curvature estimation is known to be very susceptible to noise on the contour. We shall show how noise on the contour affects the relative accuracy of the curvature computation. One interesting result is that, contrary to intuition, the calculation of the curvature for low-curvature regions is in fact impossible for common image-sizes, while reasonable results may be obtained for higher-curvature regions.

1 Introduction

In the early years of computer vision, the need to calculate a digital curve's curvature used to arise quite frequently. However, this did turn out to be a rather harder task than one originally appreciated (see [1] who compares several different approaches), and today the knowledge about the difficulty of the task has become so ingrained in the computer vision community that curvature is hardly used outside the scale space community, where much of the work is reduced to locating the zero crossings of curvature [2].

We believe, however, that only very little attention has so far been given to the question *why* the calculation of curvature is hard. Worring's approach [1] is only relevant for edgels on a pixel-grid; Kovalevsky [3] recently extended this to edgels with sub-pixel accuracy, using his own edge-finder. However, both authors basically follow the assumption that curvature for low-curvature regions can be calculated more accurately than for high-curvature regions, simply because for the former smoothing will be less invasive than for the latter, and as a consequence concentrate on high-curvature regions with a radius of curvature below 40 pxl.

This paper strives to demonstrate that for real images the curvature of low-curvature regions is in fact just as difficult (if not impossible) to compute. This is due to the limited extend of outlines in real images, and we shall see that neither very low nor very high curvature can be estimated reliably under real life conditions.

The remainder of this paper is organised as follows: Section 2 gives the analytic derivation of the expected relative error in the curvature-radius estimated

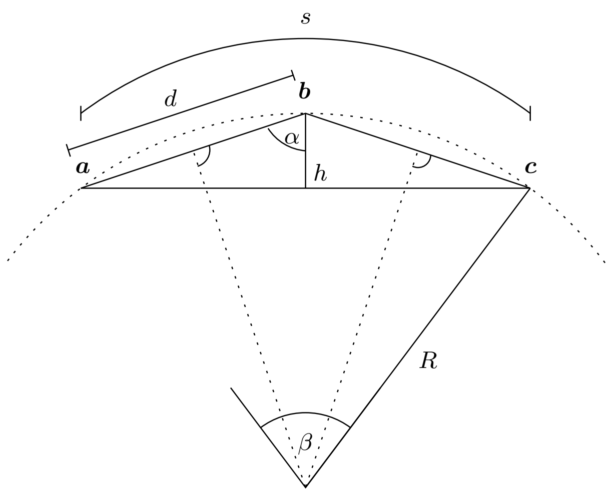

Fig. 1. Simple model for the calculation of curvature

for a very simple case, this serves to give a feeling for the mathematics involved. Section 3 gives results for a more involved but more realistic case, but without the analytic derivation. Based on these, Sec. 4 gives numeric examples of what can be expected from locally operating approaches for the calculation of the curvature, and Sec. 5 summarises our findings.

2 Analytic Derivation of a Simple Case

We will first analyse a very simple case. Given 3 points $\{\boldsymbol{a}, \boldsymbol{b}, \boldsymbol{c}\}$ on a circle, we can calculate the circle's midpoint as the intersection of the two perpendicular bisectors of the 2 lines connecting point $\boldsymbol{a}$ with $\boldsymbol{b}$ and $\boldsymbol{b}$ with $\boldsymbol{c}$; the circle's radius is then straightforwardly given as the distance from the circle's midpoint to any of the three points on the circle. Figure 1 gives an overview of the construction.

The main source of error using this setup is the uncertainty in the position of the three points. For a digital contour given as edgels found by some implementation of an edge finder like the Canny algorithm [4], only the edgel's uncertainty perpendicular to the curve is of interest, while the exact location of the edgel along the curve is of no significance.

In our very simple first model we will analyse the relative accuracy of the radius as a function of the arc-length s between the edgels used for its calculation (compare Fig. 1). We will assume that the position of the middle edgel varies with standard deviation σ along the vertical-direction, and that the two other edgels vary with standard deviation σ orthogonal to the line-segments $\boldsymbol{ab}$ and $\boldsymbol{bc}$ (so that d is constant — this only introduces a small error, which will later be ignored anyway). In addition, we will introduce a further simplification by assuming symmetry between the two sides of the construction. This allows us to fix the position of one of the points and instead assume that the height h of the triangle varies with

$$\sigma_h = \sqrt{1 + \sin^2(\alpha)}\,\sigma \tag{1}$$

(projecting the edgels' uncertainty onto the vertical axis).

The entire model is of course an oversimplification: we only use 3 out of N points, the end-edgels' covariance should be taken in radial direction (i. e. orthogonal to the contour), we assume that the distributions of the two endpoints (edgels) are not independent of each other (which they are). We also assume that curvature is constant over the entire curve-section under consideration (which most likely it isn't). However, we do feel that the above model is well suited to give an initial idea of the behaviour of curvature calculation.

The theorem on intersecting lines gives us the radius as

$$R = \frac{d^2}{2h}\,. \tag{2}$$

Its derivative with respect to h is

$$\frac{\partial R}{\partial h} = -\frac{d^2}{2h^2}\,, \tag{3}$$

and we can use this to calculate the standard deviation of the radius using simple linear error propagation as

$$\sigma_R = \sigma_h \frac{d^2}{2h^2}\,. \tag{4}$$

As stated above, we are interested in σ_R as a function of the arc-length s between the two outer edgels; it is

$$\alpha = \frac{\pi}{2} - \frac{\beta}{4}\,, \tag{5}$$

$$\beta = \frac{s}{R} \tag{6}$$

$$h = d\cos(\alpha) \tag{7}$$

and substituting (1) as well as (5), (6), and (7) in (4) leads to

$$\sigma_R = \frac{\sigma_h}{2}\frac{\sqrt{1+\sin^2(\alpha)}}{\cos^2(\alpha)} = \frac{\sigma}{2}\frac{\sqrt{1+\cos^2\left(\frac{s}{4R}\right)}}{\sin^2\left(\frac{s}{4R}\right)}\,. \tag{8}$$

In order for a result R to be usable we would require its relative error to remain sufficiently small, i. e.

$$\frac{\sigma_R}{R} \overset{!}{<} \rho \ll 1\,. \tag{9}$$

Note that, as we only analyse the *relative* error ρ it does of course not make any difference whether we calculate the radius R or the curvature $\kappa = 1/R$.

We will now calculate the minimum arc-length s necessary to satisfy (9), and from there, assuming a known average distance from edgel to edgel (set to 1 pxl in the following for simplicity), the number of pixels necessary to achieve a desired relative accuracy $\rho = \sigma_R/R$. As

$$0 \leq \cos^2\left(\frac{s}{4R}\right) \leq 1 \tag{10}$$

we get from (8)

$$4R \arcsin\left(\sqrt{\frac{\sigma}{2R\rho}}\right) \leq s \leq 4R \arcsin\left(\sqrt{\frac{\sigma}{\sqrt{2}R\rho}}\right) \tag{11}$$

and for $\sigma \ll R\rho$, which is almost always the case

$$\frac{4}{\sqrt{2}}\sqrt{\frac{\sigma}{\rho}R} \leq s \leq \frac{4}{2^{1/4}}\sqrt{\frac{\sigma}{\rho}R}\,. \tag{12}$$

Note that the left and right side of (12) differ only by a factor of $2^{1/4} \approx 1.19$; in both cases the minimum required arc-length s for a required relative accuracy $\rho = \sigma_R/R$ is proportional to the same term and approximately

$$s \approx 3\left(\frac{\sigma}{\rho}R\right)^{0.5}. \tag{13}$$

The approach outlined above is actually rather wasteful. An obvious improvement would be to use all N edgels instead of just three, and the next section will indeed have a closer look at an approach which does just that.

3 Using N Edgels

When looking for a description of a curve-segment, instead of trying to calculate this description directly from the edgels it is not uncommon to fit an algebraic curve to the edgels and use the parameters of the curve with the smallest error of fit to calculate the description. These curves are often polynomials in x and y, and in our particular application, where we are trying to find the radius of curvature of a given curve, it is a natural choice to fit part of a circle to the edgels[1].

Different methods exist to calculate the error of fit between a curve and N edgels. Probably the most intuitive method is to minimise the sum of orthographic, squared distances, which for a circle with midpoint (x_0, y_0) and radius R would be

$$C = \frac{1}{N}\sum_{i=1}^{N}\left(\sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} - R\right)^2. \tag{14}$$

[1] This is especially the case as we assume a curve with constant curvature — in general more complicated models will yield better results, although one should be wary of too complex models [5].

However, in general no closed form solution exists for (14), and it has therefore become customary to use the so called algebraic distance

$$C = \frac{1}{N} \sum_{i=1}^{N} \left((x_i - x_0)^2 + (y_i - y_0)^2 - R^2 \right)^2 , \tag{15}$$

for which a closed form solution exists [6]. It is worth noting that minimising either (14) or (15) in their above form will introduce an additional bias into the solution, which we will ignore in the following.

It is now interesting to see how many edgels N, or alternatively how long a segment s, will be needed to achieve a given relative error $\rho = \sigma_R / R$ when calculating a circle which minimises one of (14) or (15). The tool to do so is the implicit function theorem, which allow us to calculate the Jacobian $\boldsymbol{J_g}$ of the unknown function $\boldsymbol{c} = \boldsymbol{g}(\boldsymbol{e})$ with $\boldsymbol{c} = (x_0, y_0, R)^T$ and $\boldsymbol{e} = (x_1, \ldots, x_N, y_1, \ldots, y_N)^T$ whose solution minimises either (14) or (15) (in the latter case $\boldsymbol{g}$ could actually be calculated explicitly), it is

$$\boldsymbol{J_g} = - \left(\frac{\partial^2 C}{\partial \boldsymbol{c}^2} \right)^{-1} \left(\frac{\partial^2 C}{\partial \boldsymbol{c} \partial \boldsymbol{e}} \right)^T \Bigg|_{\boldsymbol{c}_0} \tag{16}$$

provided that the Hessian $\frac{\partial^2 C}{\partial \boldsymbol{c}^2}$ is indeed invertible at the point of the solution $\boldsymbol{c}_0$. If we assume that all edgels $(x_i, y_i)^T$ share the same covariance $\sigma \boldsymbol{I}_2$, where $\boldsymbol{I}_2$ is the 2×2 identity matrix — this is a reasonable model for variance σ perpendicular to the curve — we can calculate the covariance matrix of the vector $\boldsymbol{c} = (x_0, y_0, R)^T$ as

$$\sigma \boldsymbol{J_g} \boldsymbol{J_g}^T = \sigma \left(\frac{\partial^2 C}{\partial \boldsymbol{c}^2} \right)^{-1} \left(\frac{\partial^2 C}{\partial \boldsymbol{c} \partial \boldsymbol{e}} \right)^T \left(\frac{\partial^2 C}{\partial \boldsymbol{c} \partial \boldsymbol{e}} \right) \left(\frac{\partial^2 C}{\partial \boldsymbol{c}^2} \right)^{-1^T} \Bigg|_{\boldsymbol{c}_0} . \tag{17}$$

Evaluating (17) analytically, a purely mechanical process, leads to rather lengthy (although for (15) still quite manageable) equations outside the scope of this paper. A numerical evaluation is, however, perfectly possible, and we can then fit a function to the numerical results; using the squared orthographic distance (14) the output of these calculations is well approximated by a function

$$N \approx 2 \left(\frac{\sigma}{\rho} R \right)^{0.4} , \tag{18}$$

while for the algebraic distance (15) a good approximation is

$$N \approx 3.7 \left(\frac{\sigma}{\rho} R \right)^{0.4} . \tag{19}$$

This is very similar in structure to (13), but with an exponent of 0.4 rather than 0.5. We will see in Sec. 4 what this means for practical applications.

4 Numeric Examples

In Sec. 2 and 3 we have calculated the minimum number of edgels required in order to compute a radius of curvature R with given relative accuracy σ_R/R, compare (13), (18), and (19). It is now instructive to see what actual values for s or N are required when dealing with real images. For this it is important to realise that for any $N > 3$ and a curve with non-constant curvature, what we really calculate is not the curvature at the particular point under consideration, but some sort of average curvature. Assuming noise-free data, and as long as the real curvature in the interval under consideration is a monotonic function of the arc-length, we can at least guarantee that the curvature thus calculated would be correct for some point inside the interval — although almost certainly not for the point under consideration. This is called localisation uncertainty in [3]. However, as soon as the curvature over arc-length ceases to be a monotonic function, arbitrary results are possible even in the case of noise-free data.

From the above it is immediately clear that the region of N points from which curvature is calculated should not cross extrema of curvature (whose position we only know after the fact); to keep the localisation error small it is also desirable to keep N small enough so that curvature inside this interval doesn't vary too much. As a rule of thumb most authors would limit the arc under consideration to somewhere between $\pi/4 \leq \beta \leq \pi/2$. This automatically defines a lower limit on the curvature-radius which can be calculated, and this lower limit seems to be what most authors are most interested in [1,3].

But there is also an upper limit on N dictated by the fact that the entire object, usually containing several extrema of curvature, has to fit into a usually PAL-sized ($768\,\text{pxl} \times 576\,\text{pxl}$) image, limiting the maximum usable section to well below 100 pxl— [3] cites values around 4 pxl–20 pxl. This upper limit has in our opinion been mostly ignored, and it is the aim of this paper to demonstrate that very low curvature is at least as difficult to calculate as very high curvature.

For a numerical evaluation it is important to know with what subpixel-precision we can expect to locate edges in real images. As cited in [7] a basic implementation of the Canny algorithm [4] will typically produce edgels with a standard derivation perpendicular to the edge from somewhere about $\sigma \approx 0.1\,\text{pxl}$ to $\sigma \approx 0.3\,\text{pxl}$ depending on the actual CCD (3-chip RGB versus BW versus Bayer RGB) and lens used as well as the contrast between regions. As most affordable colour cameras use only one CCD with a Bayer-filter and often rather cheap lenses, $\sigma = 0.3\,\text{pxl}$ is usually all we can hope for. Together with NTSC or, at best, PAL sized images as they are still predominant in computer vision, we are therefore hampered both by low image quality and small images.

Figure 2 shows how many edgels $N(R)$ are needed to compute the curvature (radius) with a relative accuracy of both $\rho = \sigma_R/R = 3\,\%$ and $\rho = 33\,\%$, using (18). It is worth remembering that the latter corresponds to a worst-case error of about $3\rho \approx 100\,\%$, which means that not even the sign of curvature can be calculated reliably! But even accepting such an error we see from Fig. 2 that only for radii between $5\,\text{pxl} \leq R \leq 400\,\text{pxl}$ can we calculate the curvature if we limit the arc to $\beta < \pi/4$ and $N < 20\,\text{pxl}$. The much more realistic (though still

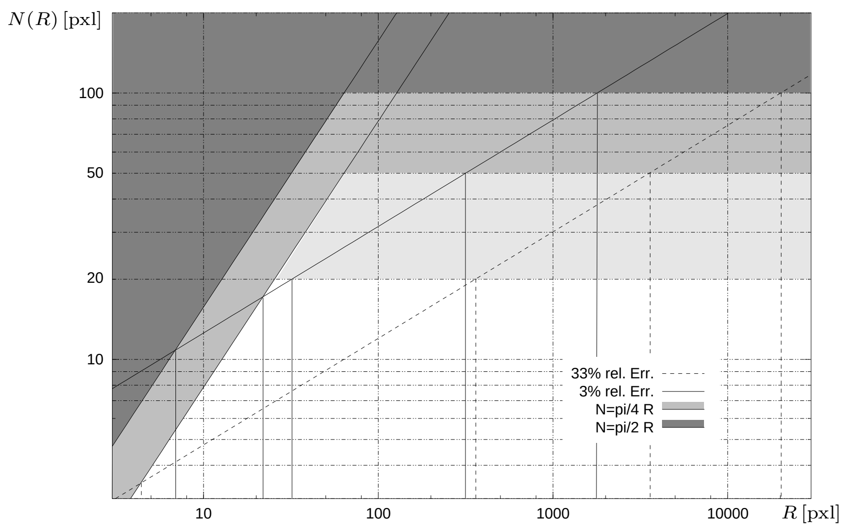

Fig. 2. Number of edgels $N(R, \rho, \sigma = 0.3\,\text{pxl})$ needed to calculate the radius of curvature R with a relative error of less than $\rho = 33\,\%$ (*dashed line*) or $\rho = 3\,\%$ (*solid line*). Inside the grayed out regions R can not even be determined with this accuracy if we assume a maximum arc $\beta = \pi/4$ $(\pi/2)$ and a maximum number of edgels $N = 20\,\text{pxl}$ $(50\,\text{pxl},\ 100\,\text{pxl})$

big) relative error of $\rho = 3\,\%$ can, with the same assumptions, only be reached for $20\,\text{pxl} \leq R \leq 30\,\text{pxl}$, which is essentially useless. But even allowing for arcs up to $\beta \leq \pi/2$ and $N \leq 100\,\text{pxl}$, we can still only calculate radii between approx. $8\,\text{pxl} \leq R \leq 2000\,\text{pxl}$. And it is worth remembering that in an image with PAL resolution of $768\,\text{pxl} \times 576\,\text{pxl}$ an arc of $N = 100\,\text{pxl}$ will almost certainly contain an extremum of curvature even for only moderately complex objects, rendering the result thus computed essentially worthless.

We can actually increase the location accuracy (i. e. the likelihood that the curvature estimated belongs to a point near the one for which we estimated it) by fitting a sufficiently complicated model to the curve; even just fitting a general conic cross-section rather than a circle would improve localisation. It should however be clear that the curves in Fig. 2 already constitute a best-case estimate — curvature calculated from real, non-circular data will always be less correct unless an *exact* model of the curve is known.

So what can be done to increase the relative accuracy of the radius of curvature calculated, at least to such an extent that radii of the correct order of magnitude can be calculated? Equations (13), (18), and (19) show us the way: we can either decrease σ, or increase the resolution of the image, thus increasing the number of points N on any given curve-segment, which would increase the maximum radius of curvature.

Decreasing σ, i. e. increasing the accuracy with which edgels can be located, unfortunately only offers limited room for improvements. Canny's algorithm is already quite good at this, and even algorithms which were specifically tuned to improved location accuracy[8] promise little better than $\sigma = 0.1\,\text{pxl}$. This basically leaves higher resolution as the only way out, and thankfully modern digital cameras offer just that. Using a 6 Mega-pixel camera (i. e. increasing the resolution by a factor of approx. 3.7) means that only about $3.7^{-0.6} \approx 45\,\%$ of the same curve still needs to be visible when compared to PAL, as the number of edgels on any given piece of curve increases, while at the same time their variance decreases.

Finally, please note that since we used linear error propagation throughout this paper, we can rely on the exact numerical values only for sufficiently small g, while for larger values only qualitative statements are possible.

5 Conclusion

In this paper, we have shown that using locally operating methods the curvature of both high-curvature as well as low-curvature regions can not be calculated reliably from standard PAL-sized images even under optimal conditions and extending the meaning of “locally operating” to rather large regions; only for a possibly quite small range of medium-small curvatures is it at all possible to calculate a meaningful result. Increasing the resolution of the image by a factor u (linearly) will increase the maximum radius of curvature which can be calculated reliably (by $u^{0.6}$), but this does not eliminate the underlying problem.

As did [1,3], this paper only considered local algorithms for the calculation of curvature. It is possible that more global methods (gray-level-based, spline-fit, or scale-space based methods) could provide additional accuracy. However, this has not been analysed.

References

1. Worring, M., Smeulders, A.W.M.: Digital curvature estimation. Comput Vision, Graphics & Image Processing: Image Understand **58** (1993) 366–382
2. Mokhtarian, F., Mackworth, A.K.: A theory of multiscale, curvature-based shape representation for planar curves. IEEE Trans Pattern Anal Mach Intell **14** (1992) 789–805
3. Kovalevsky, V.: Curvature in digital 2d images. International Journal of Pattern Recognition and Artificial Intelligence **15** (2001) 1183–1200
4. Canny, J.F.: A computational approach to edge detection. IEEE Trans Pattern Anal Mach Intell **8** (1986) 679–698
5. Kanatani, K.: Geometric information criterion for model selection. Int J Comput Vision **26** (1998) 171–189
6. Pratt, V.: Direct least-squares fitting of algebraic surfaces. Computer Graphics **21** (1987) 145–152
7. Utcke, S.: Grouping based on projective geometry constraints and uncertainty. In: Proc Int Conf Comput Vision, Bombay, IEEE CS, Narosa Publishing House, New Delhi (1998) 739–746
8. Overington, I.: Computer Vision: A unified, biologically-inspired approach. Elsevier, Amsterdam (1992)