DEPARTMENT OF ELECTRICAL ENGINEERING

TECHNION - ISRAEL INSTITUTE OF TECHNOLOGY

Internship Task IAESTE 2012

Improve Numerical Scheme for the Ricci Flow

Name:	Veikko Ruth
Address:	Wehrstraße 2 99734 Nordhausen Germany
Course of studies:	Biomedical engineering
Matriculation:	2006
Supervisor:	Eli Appleboim

March 23, 2013

Abstract

Based on the paper [ASZ12] this documentation describes the work on the former Ricci flow algorithm which was provided by the *Vision and Image Sciences Laboratory* of the *Technion- Israel Institute of Technology*.

A discrete version of the Ricci flow, applicable to images, can be used for denoising, single-image-based enhancement and super-resolution. It is the only flow which effects the metric of the image more than the image itself. The flow preserves image structure better than other state-of-the-art image enhancement schemes. It is based on the combinatorial Ricci curvature defined by Forman. (cp. [ASZ12])

Contents

1 Introduction

If the image is considered as a surface embedded in some Euclidean space it is possible to apply curvature flows. The flows can be represented in the following form:

$$\frac{\delta I}{\delta t} = \mathscr{O}(I), \tag{1.1}$$

$\mathscr{O}$ is the operator sensitive to curvature in the image I. The curvature which is used here is the Ricci curvature. It represents an average of sectional curvatures. Also it behaves as the Laplacian of the metric (see [Ber03]).

The Ricci flow is defined as (see [Ham82], [Ham86]):

$$\frac{\delta \mathscr{G}}{\delta t} = -Ric(I), \tag{1.2}$$

where $\mathscr{G}$ is the image metric and Ric denotes the Ricci curvature. By observation of equation 1.2 it can be recognised that the mainly involved value for the Ricci flow is the metric of the image. For dimension $n = 2$, which is mostly used in image processing, the Ricci curvature equals twice the Gauss curvature.

To be able to use the Ricci curvature on discrete images Forman describes in [For03] a combinatorial analogue of it. The operator is introduced in the context of cell complexes. For image processing this cell complex can be seen as the mesh grid which contains the pixel values of a single image. The exact adoption to image processing is shown in [SAGZ09].

The following work contains a short outline of the numerical Ricci flow implementation and shows detailed where the problems of this scheme are located. Also it is shown how some of the problems were already solved.

2 Presentation of the Numerical Scheme

2.1 Overview

In the former source code the Ricci flow is used to perform super-resolution. At first one of six example images passes through an normalising process which sets the image values in between zero and one. The image is saved as the variable I. Afterwards a combination of downsamping and upsampling is used to reduce the image information and simulate a poor version of the image which is saved as I_p. Now some specialised discrete version of the Ricci flow is used to enhance the poor image. It is divided into the following computations.

Calculating Weights

The weight function should reinforce the Ricci flow at the edges of the image. To achieve this the gradients in x- and y- direction are computed. Afterwards the absolute values of both gradients are merged in a matrix I_{norm}. In a final step it is modified as part of the equation 2.1 and the matrix W results.

$$W = 0,1 + \left(10 \circ \frac{1}{I_{\text{norm}} + 10}\right) \tag{2.1}$$

Out of this matrix a second matrix W_{reverse} is computed. Therefore every element of W is subtracted from a matrix of the same size filled with ones. Then the values of the result are normalised to the range zero to five. After this step W_{reverse} is successfully computed.

Ricci Flow

The Ricci flow itself is performed in an iterative process which is shown in figure 2.1. g represents the metric of the image and Ric the associated Ricci curvature. In this discrete case t stands for the different iteration steps. Following the diagram the new

image $I(t+1)$ should result from the Ricci curvature applied on the metric of the original image $I(t)$.

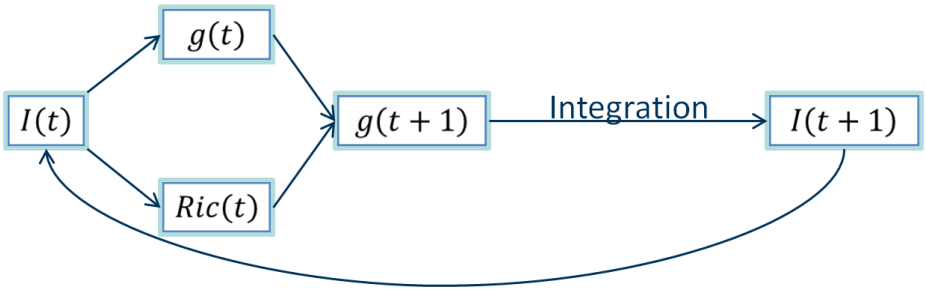

Figure 2.1: Ricci flow diagram [ASZ12]

To perform the whole process the following steps are realized. In the beginning the gradients of I_p in x- and y- direction are computed. They are saved as I_x and I_y. Afterwards the sign functions of the gradients are saved as $I_{x,\text{sign}}$ and $I_{y,\text{sign}}$. Now the metric in x- and y- direction can be calculated with the equations 2.2.

$$g_x = \sqrt{1 + I_x^2} \qquad g_y = \sqrt{1 + I_y^2} \tag{2.2}$$

The Ricci curvature is computed as defined in [ASZ12] from the original image I. It is possible to choose in between combinatorial weighting and geometric weights. The latter used an additional parameter β which influences the sensitivity of curvature to the image gradient (for additional information see [SZ98]). Attention should be paid to the fact that the curvature is only computed in vertical and horizontal direction. The results are saved as two matrices $ricci_{\text{ver}}$ and $ricci_{\text{hor}}$.

The following step, shown in equation 2.3 and 2.4, performs the manipulation of the metric including the Ricci curvature and the associated weights.

$$g_{x,\text{new}} = g_x + W_{\text{reverse}} \circ ricci_{\text{hor}} - W \circ ricci_{\text{hor}} \tag{2.3}$$

$$g_{y,\text{new}} = g_y + W_{\text{reverse}} \circ ricci_{\text{ver}} - W \circ ricci_{\text{ver}} \tag{2.4}$$

To achieve an image out of the manipulated image metric the following two steps are necessary. At first x- and y- gradients are recalculated out of the metrics. This can be done by a simple transposition of the equations 2.5 and an additional multiplication with the corresponding sign function as shown in equation 2.5.

$$I_{x,\text{new}} = \sqrt{\left|g_{x,\text{new}}^2 - 1\right|} \circ I_{x,\text{sign}} \quad \text{and} \quad I_{y,\text{new}} = \sqrt{\left|g_{y,\text{new}}^2 - 1\right|} \circ I_{y,\text{sign}} \tag{2.5}$$

The sign function is fundamentally important because the metric space is non-negative and the sign information was lost during the transformation to the metrics.

The second step is the integration which creates the new image. For this problem the method of Poisson solver (see [ARC06]) is used. To avoid artefacts only the difference in between the original gradient and the new gradient is integrated. The result is a deviation image I_d which is added to the original image.

Thus one iteration step is finished and the newly iterated image is the origin for the next loop cycle.

2.2 Detailed Examination

The major aim of this work is to prove the correct implementation of the Ricci flow and the identification of hidden mistakes and discrepancies. To perform this the effect of every mentioned calculation regarding denoising, single-image-based enhancement and super-resolution is checked.

Additional Weights

With regard to an universal Ricci flow implementation the additional weights W and W_{reverse} are not suitable. They cause a specialised Ricci flow with bigger effect at edges.

Image Metric

The calculation of the metric out of the gradient as shown in equation 2.2 is no big issue. The fact that the metric in diagonal directions is unattended might cause future problems.

Ricci Curvature

The computation of the Ricci curvature was checked in detail. It is calculated exactly as defined in [ASZ12]. Similar to the image metric the curvature component in diagonal direction is missing.

Manipulation of the Metrics

As mentioned the manipulation caused by the Ricci flow is proceeded in the equations 2.3 and 2.4. Differently than expected regarding to equation 1.2 the functions contain two curvature components with different sign function. They represent a forward-

and reverse- flow component. W_{reverse} increases the reverse component rapidly. Thus the resulting flow is not the typical Ricci flow which describes transients, rather it performs an edge sharpening process which can be used for super-resolution.

Recalculation of the Gradients

The recalculation of the gradients out of the manipulated metrics seems simple. The problem is that the used sign functions are not manipulated. In case that the sign should change, caused by the influence of the curvature, the new gradient points in the wrong direction. Also if the sign function was zero and the flow generates a non zero value in the new metric it will turn into zero again when the equation 2.5 is used.

Integration of the Gradients

To verify the correct functionality of the Poison solver method the whole gradients instead of gradient differences, as mentioned in section 2.1, are integrated. If the described iteration process is performed a few times with those changes the result is quite inaccurate. In figure 2.2 the original image is shown on the left and the result after five iterations on the right. An obvious shift in x- and y- direction is visible.

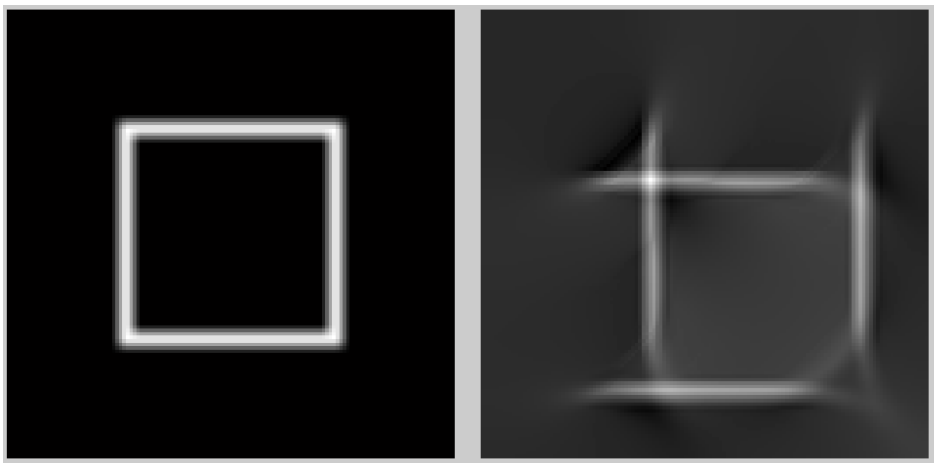

Figure 2.2: *left:* original image *right:* result after five iterations (artefacts caused by the Poison solver method are clearly visible)

3 Suggested Improvements

The following suggestions are based on the aim to achieve an an universal valid basic Ricci flow implementation. If necessary this basic version can be changed afterwards to make it more suitable for special fields of activity.

Additional Weights and Image Metric

In an universal flow version is no need for additional weights and a reverse flow component so these parts were removed. The equations to manipulate the metrics change in

$$g_{x,\text{new}} = g_x - ricci_{\text{hor}} \quad \text{and} \quad g_{y,\text{new}} = g_y - ricci_{\text{ver}}. \tag{3.1}$$

Recalculation of the Gradients

The problem while recalculating the gradients out of the metrics is that the sign function was not changed. It needs to be adapted to the changed image metric. There are two fundamental problems which need to be considered:

1. When and how should a zero sign change?

2. When should a plus sign swap to minus and vice versa?

With respect to the fact that the flow describes transients a zero sign should turn into the sign of an immediate neighbour. If for example a positive charge expands on a surface it turns the zero charged neighbouring areas positive as well.

The second problem must be considered when a new metric value leaves the range defined as values greater or equal to one. According to equation 2.2 this behaviour should not be possible. If it occurs the sign should change to the opposite and the metric must by corrected. The concerning metric value must be changed in a manner that the distance to one stays equal. At the position where the value drops below one it must be replaced by the complement which is greater than one.

Both suggestions are implemented in an initial solution which still causes errors, especially the change from negative to positive sign. Out of this reason the negative

to positive exchange is not used in the current source code. In future work those implementations need to be double checked. Also it is possible that the missing diagonal components in metric and Ricci flow create unexpected behaviour of the metric which leads to the fact that the adapted sign function is not corresponding with the manipulated metric.

Integration of the Gradients

As shown before the Poison solver method generates shift artefacts if used in a loop process several times in a row. This probably didn't stand out when the code was developed because it was used for a singular integration. In a detailed observation of the function a simple incrementation error was detected which shifts the resulting image one pixel in x- and y- direction compared to the original. This error was weakened by using just the gradient differences for integration. In figure 3.1 the origin picture is shown on the left. In the middle the result of the old code after five iterations is shown (still with integration of gradient differences). On the right the result after correcting the incrementation error with using the whole gradients for integration is shown (also after five iterations). Both results are processed with the old metric manipulation for edge sharpening.

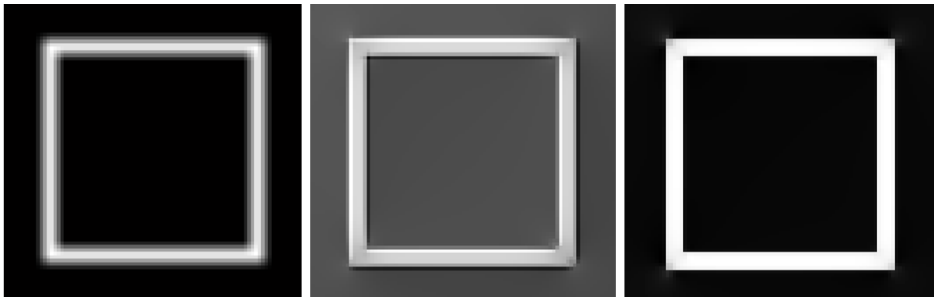

Figure 3.1: *left:* original image *middle:* result after five iterations (old code) *right:* result after five iterations (new code)

An extraordinary improvement in respect to the image contrast is visible. In detailed observation the corners of the right image are slightly blurred which is probably caused by the missing diagonal components in metric and Ricci flow.

Preservation of the Gradients and Metrics

An explicit examination of the iteration structure shown in figure 2.1 with the knowledge of an incorrect integration process suggests new problems. The error caused by

the integration will add up with every iteration step. An easy solution to reduce this error partly is the preservation of the new gradient which is directly connected to the new metric $g(t+1)$. Instead of computing gradient and metric from the possibly defective integrated image $I(t+1)$ they are taken form the previous iteration step. The first initial gradient and the following metric is calculated from the origin image. The resulting new diagram is shown in figure 3.2.

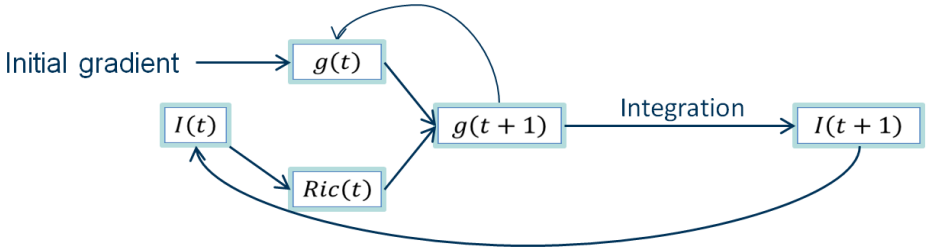

Figure 3.2: Ricci flow diagram with gradient preservation

Now the error caused by the integration is only existing in the Ricci flow component. But this component is quite small compared with the size of the metric one and its effect is equally little.

This extension of the implementation should suppress the Poison solver caused problems as far as possible. But it might cause new problems, because the calculated curvature does not fit exactly to the corresponding metric. This should be checked detailed in future work. The implementation of diagonal metric and curvature might help here as well, because it fills a gap of information which is probably missing for an accurate integration.

Absolute Curvature

The first approaches concerning the problems with the sign function used a absolute version of the curvature. The non negative curvature values caused that the metric values never left the range equal or greater than one. The results do not represent an exact version of the Ricci flow, but this modification has also worth mentioning properties. It reduces areas with small curvature and preserves edges to the disadvantage of the image contrast. An example is shown in figure 3.3. This approach was not pursued but it might be a starting point for future ideas.

Figure 3.3: *left:* original image *right:* image after flow with absolute curvature

Additional Features

During the examination process of the old implementation several features were added to check the correct computation of the Ricci flow. In the beginning of the new code the user has the opportunity to choose in between five different evaluation methods and new test images to check the algorithm. The new test images include a chirp image, an e-function image and a image of known patterns. The five evaluation methods are:

1. *Video of the flow:* Every new frame represents the image after one iteration.
2. *Video of the flow:* Every new frame represents a 2D plot from a horizontal slice located at the middle of the image. The axis of ordinate represents the pixelvalues.
3. *2D Plot of slices:* Horizontal slices of the original image and the gradient, metric, curvature etc. after the flow process
4. *Result and means:* Image after flow and mean of curvature and metric over iteration steps
5. *Plot of surfaces:* 3D plot of the image, metric and curvature after the flow. The third dimension represents the pixelvalues.

4 Conclusion and Future Prospects

Summarized in this work the provided numerical scheme for the Ricci flow was analysed in detail and all the weak points were laid open. For most of the problems a solution was found or a detailed solution proposal was given. More precisely the integration process (Poison solver method) was debugged, an initial implementation to adjust the sign function for gradient recalculation was presented and a gradient preservation to suppress other integration errors was embedded.

In figure 4.1 the results of the current implementation are visible. On one hand the desired transient is visible and on the other hand this transient is clearly non-uniform. A further improved version of the sign correction might solve this problem.

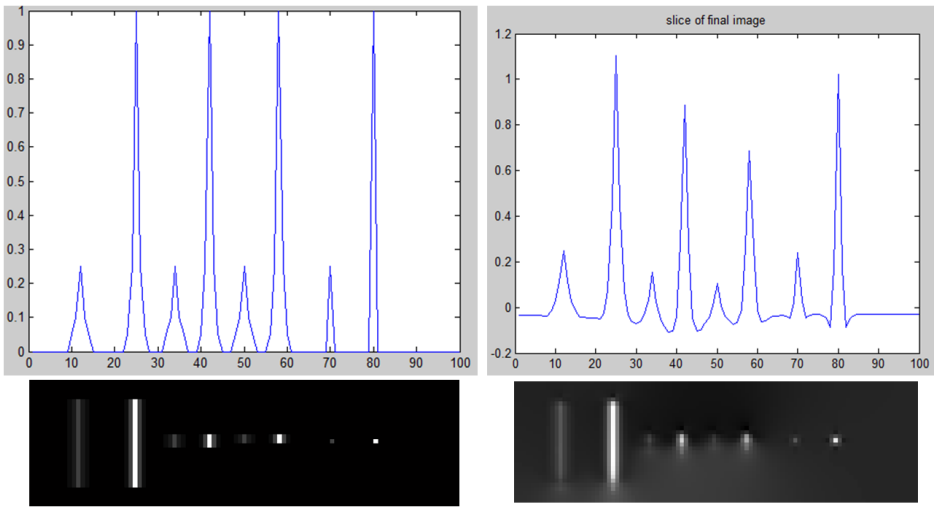

Figure 4.1: *left:* original image with horizontal middle slice (top) *right:* image after one iteration with horizontal middle slice (top)

Also for most of the test images and initial values (numbers of iteration, β) the Ricci values drop to zero if the iteration steps tend to infinity. But in some cases the Ricci values are rising constantly which is an undesirable behaviour. Also the metric values are increasing with the number of iterations instead of becoming constant. Both problems might be caused by the gradient preservation and the non fitting metric-curvature combination. A connection to a wrong sign function is also possible.

Besides all mentioned error reasons the missing diagonal values in metric and curvature may cause all the problems. So if all suggested improvements are implemented and double checked the next step is to realize an algorithm which can compute the missing variables.

All in all this work is a perfect base for future improvements and provides evaluation methods which make a quick and precise verification of changes very easy.

List of Figures

Bibliography

[ARC06] Agrawal, A. ; Raskar, R. ; Chellappa, R.: What is the Range of Surface Reconstructions from a Gradient Field? In: *ECCV*, 2006

[ASZ12] Appleboim, Eli ; Saucan, Emil ; Zeevi, Yehoshua Y.: Ricci curvature and flow for image denoising and super-resolution. In: *Signal Processing Conference (EUSIPCO), 2012 Proceedings of the 20th European*, 2012

[Ber03] Berger, M. A.: *Panoramic View of Riemannian Geometry.* Berlin : Springer-Verlag, 2003

[For03] Forman, R.: Bochner's Method for Cell Complexes and Combinatorial Ricci Curvature. In: *Discrete and Computational Geometry* 29(3) (2003), S. 323–374

[Ham82] Hamilton, R.: Three-manifolds with positive Ricci curvature. In: *J. Diff. Geom.* 17 (1982), S. 255–306

[Ham86] Hamilton, R.: The Ricci Flow on Surfaces. In: *A.M.S. Contemp. Math.* 71 (1986)

[SAGZ09] Saucan, E. ; Appleboim, E. ; G., Wolansky ; Zeevi, Y. Y.: Combinatorial Ricci Curvature and Laplacians for Image Processing. In: *Proc. of CISP'09* Bd. 2, 2009, S. 992–997

[SZ98] Sochen, N. ; Zeevi, Y. Y.: Representation of Colored Images by Manifolds Embedded in Higher-Dimensional Non-Euclidean Space. In: *IEEE. Proc. of ICIP98*, 1998