united nations
educational, scientific
and cultural
organization

international atomic
energy agency

the
abdus salam
international centre for theoretical physics

SMR1537/6

Summer School and Conference on Real Algebraic Geometry and its Applications

(4 - 22 August 2003)

Geometry of Images
(Geometry and Computer Vision)

Yosef Yomdin
Department of Mathematics
The Weizmann Institute of Science
Rehovot 76100
Israel

Y. Yomdin 22.7.03

GEOMETRY OF IMAGES
(Geometry and Computer Vision)

ICTP Lectures, August 2003

1. Lectures Program

Lecture 1

- Introduction to Image Processing and Compression ([2,3], [1], pp. 1-9)
- An overview of the Geometric Image representation (VIM) ([1], Section 2 below)

Lecture 2

- Singularities, Normal Forms, Flexible High Order Representation ([4], [5,6],
- [1], pp. 9-11)
- Singularities of Images; VIM structure and rendering ([1], pp. 11-18, Section 2 below)

Lecture 3

- High order detection of Ridges ([1], pp. 18-25, Section 3 below)
- High order detection of Edges. Detection of Color Profiles ([1], pp. 18-25, Sections 4, 5 below)
- Jet Algebra and Multi-order computations (Sections 3-5 below)

Lecture 4

- Coding of VIM data: general structure (Section 6 below)
- Coding of point-sets, of curves and of the Background color (Section 6 below)
- (Optional) A short computer demonstration of VIM

2.VIM structure and rendering considerations

1.1. Characteristic Lines and their Color Cross-Sections

Below we use as equivalent the names “VIM” and “Synthesized Texture”.

Characteristic lines (Lines) are the lines (either visible or virtual) on the image along which the image visual pattern is consistent.
’Lines’ are represented by their virtual “central lines” and “brightness cross-sections (Color Profile”, or CP). The central line captures in the most accurate way the geometric shape of the Line, while the CP describes the brightness (color) behavior in the orthogonal direction to the line.
The central line is given by a second or third order spline curve (preferably a second order).
CPs are given by a small number of model shape types, each characterized by a small number of parameters. CPs are stored at some predefined “CP control points” on the central line, and interpolated between these control points. The line sections defined between these control points are called Line Segments (LS).
Figure shows an LS of a Line with a CP interpolated along it, at point ‘t’, which is used to calculate point ‘u’ color.

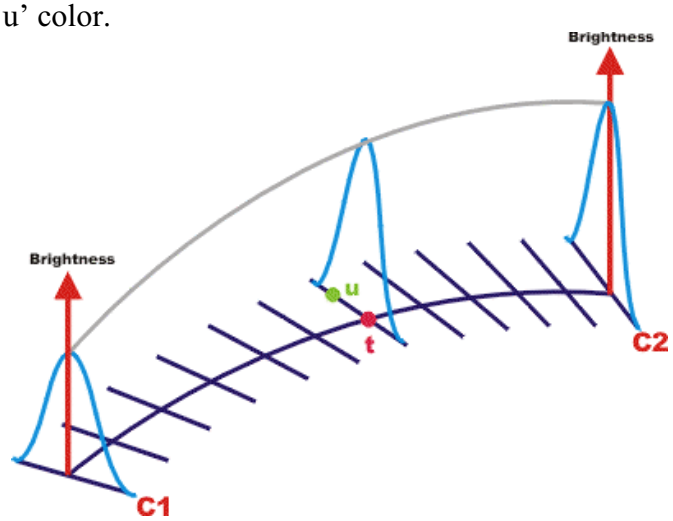


Figure 1 – Color interpolation along a curve with color profiles at end points.

1.1.1. Parameters of Lines' Geometry

All the geometric parameters below are described in the coordinate system (x,y) in the image plane, with the unit length taken to be the distance between two neighboring pixels.

The endpoints of the Line and their crossings are called “Terminal Points”, or TP. Other points on the lines are defined as Line Points (LP), and the in-between line sections are

the Line Segments (LS). If a Line forms a simple closed curve, one of its points is chosen to be a TP.
All the TPs and LPs on the Synthesized image are described (in a certain order) by their coordinates (x,y) in the above coordinate system.
Lines are represented by special second order splines, that we call P-curves.
A P- curve is a chain of convex arcs S_i, i = 1, 2, …, n, starting at the point $z_0 = (x_0, y_0)$ and ending at the point $z_n = (x_n, y_n)$. The arcs S_i and S_{i+1} have the common point $z_i = (x_i, y_i)$, called a vertex, which is an LP. Each Arc is an LS. For a closed curve the initial point z_0 and the end point z_n may coincide.
Being represented by an ordered list of consecutive points, each P-curve possesses a natural orientation.
Either parabolic or circular arcs S_i are used, in such a way that for each $[z_{i-1}, z_i]$ the arc S_i is completely characterized by the height h_i of its center over the segment $[z_{i-1}, z_i]$. This height is taken with the sign plus, if the arc is curved to the left side of the P-curve, and with the sign minus, if the arc is curved to the right.
In particular, if the parabolic arcs Si are chosen, they are taken to be symmetric with respect to the line passing through the center of the straight segment $[z_{i-1}, z_i]$ and orthogonal to it. Thus for $[z_{i-1}, z_i]$ given, the symmetric parabolic arc Si is completely characterized by the height h_i of its center over the segment $[z_{i-1}, z_i]$.
Consequently, a P-spline curve is given by the following parameters:
The coordinates (x_i, y_i) of the vertices z_i, i = 0, 1, …, n, and the heights h_i, i = 1, …, n, of the arcs $_{Si}$. It may be more convenient to keep the coordinates (x_0, y_0) of the starting point z_0, and the vector coordinates (v_i, w_i) of the segments $[z_{i-1}, z_i]$, i = 1, … , n, for the rest, $v_i = x_i - x_{i-1}$, $w_i = y_i - y_{i-1}$.
The arcs S_i of the P-curves, representing poly-links, are called below “links”.
Notice, that for each P-curve, representing Line, its starting point and its endpoint are known in advance, being the TPs.
To summarize, the central lines of the Lines of a Synthesized image are completely described by the following parameters:

1. The list of the TPs and LPs, with their coordinates. Notice that the information, given by TPs is redundant and can be reconstructed from the rest of the data. TPs do not appear in the Nodes.
2. The list of LSs, specified by their vertices. Since the choice of one of two possible orientations of the Line is important, each Line is given by an ordered list of its vertices zi (Line Point), starting at one of the endpoints of Line and continuing following the natural order of the vertices, together with the heights of the corresponding Line Segment. Alternatively, the vectors [zi-1, zi] and the heights can be specified.

Parameters of Lines' Brightness Cross-Sections *(*Color Profiles)

In the basic Color Profile (CP) default configuration, described below, the CPs control points coincide with the vertices (Line Points) of the P-curves, representing the Line. The type of the CP (edge or ridge) does not change along the Line. Each ridge (i.e. a Line with the ridge type CP) is marked either as a “separating” or as a “non-separating” one. Any edge is separating.
The basic Synthesized images representation uses two types of cross-sections: edge CP and ridge CP.

1.1. Color profiles (CP)

1.1.1. Edge CP

EdgeCP, as shown on Figure 2, is completely described by the following parameters:

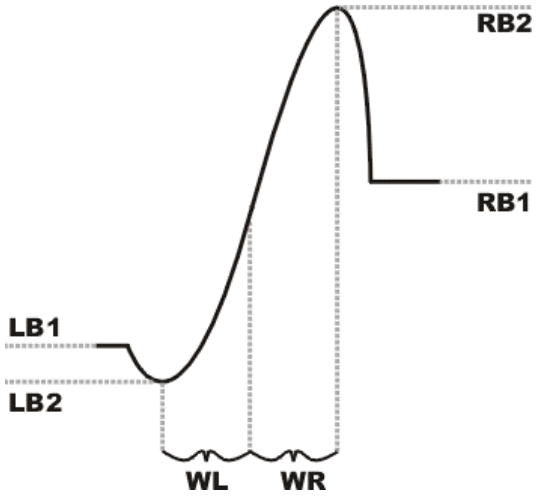


Figure 2 - Edge color profile.

- Left width WL
- Right width WR (default assumption is that WL = WR)
- Left brightness LB1
- Left brightness LB2
- Right brightness RB1
- Right brightness RB2

**The reason for appearance of the “bumps” on both sides of the edge CPis that in most of natural images, either scanned or taken by a video or a still digital camera, the actual edges cross-sections usually contain such bumps. They are caused by the physics of the light propagation, by specifics of scanning procedures and, on the other side, by the human visual perception.

However, for different kind of images, such as synthetic computer-generated images quite different shapes of the cross – sections might appear. This is true also for certain types of scanners, and especially, for other sensors, like infra-red ones. In more advanced profiles of Synthesized representation fairly general shape models of cross-sections can be used, leading to more types of Color Profiles.

The “margin” parameters LB1 and RB1, being important cross-section features, are stored together with other cross-section parameters. However, they have another natural visual interpretation as the background brightness along the edge on its different sides. It is this interpretation, that is used in the recommended reconstruction (decoding) algorithm: the margin cross-section parameters enter the background Area (in between the Lines) reconstruction Procedure.

In most of applications, the “height” of the cross-section bump can be taken to be a certain fixed fraction of the total height of the cross section. This fact is utilized in the “aggregation” level of the Synthesized representation, by encoding of the actual heights as the corrections to the default ones. In an ultimate implementation of this mode the

parameters LB2 and RB2 can be eliminated and replaced by their default values, computed through the rest of the parameters.**

1.1.2. Separating Ridge CP

Separating Ridge CP, as shown on Figure 3, is completely described by the following parameters:

- Left width WL
- Right width WR
- Left brightness LB1
- Left brightness LB2
- Central brightness CB
- Right brightness RB1
- Right brightness RB2

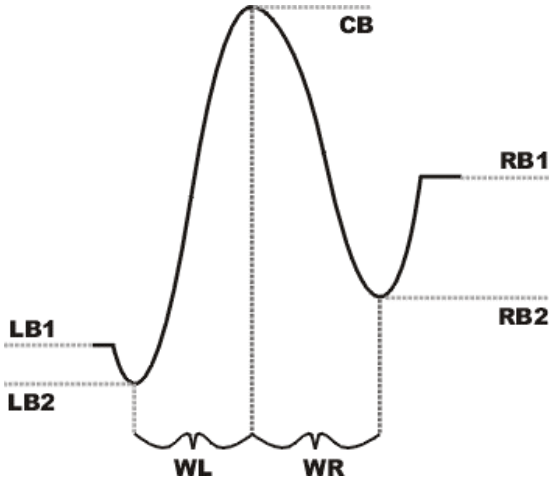


Figure 3 – Separating ridge color profile.

All the remarks above referring to edges, concerning the nature of the bumps and the redundancy of the side brightness parameters, as well as the interpretation of the margin brightness parameters LB1 and RB1 as the Background values, remain valid also for ridges.

1.1.3. Non-Separating Ridge CP

This type of CP has the same parameters as the separating one, besides the margin parameters LB1 and RB1, which are not defined; see Figure 4.

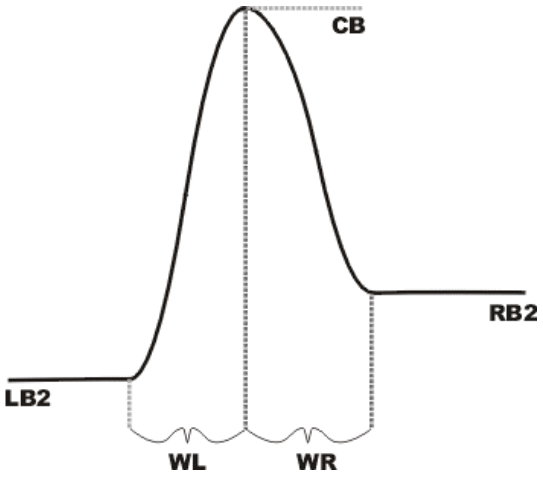


Figure 4 – Non-separating ridge color profile.

On a higher aggregation level various default configurations can be adopted, simplifying data representation. For example, the right and the left width of the ridge are usually roughly equal, so a single width parameter can be used. The right and left brightness values of a non-separating ridge can be identified with the background values at the corresponding points, etc.

If a color image is represented, all the brightness parameters are defined independently for each color component, while the same width parameters are kept. These color components can be R, G, B, CMYK, YUV or other formats. Below we use the word "brightness" to denote any component of these color formats.
As it was stated above, the Color Profiles (cross-sections) are specified at each LP of the Line.
To summarize, Color Profiles (cross-sections) of characteristic lines are specified by the following parameters:

- Type of a cross-section (edge, ridge or separating ridge) on each LP.
- Width and brightness parameters of the cross-sections, as specified above, at each of the vertices of the P-curves, representing LPs.

In case of a color image, the type and the width of the cross-sections are the same for each color component, while the brightness parameters are specified independently for each color.
Of course, on a higher aggregation level various known visual spatial-color correlations can be used to reduce the data size.
A closed Line can be marked as a boundary contour of the Synthesized image or an Object within it.

1.1.4. Patches

Patches capture fine scale details in the image. They are represented by Gaussian-shaped or parabolic-shaped mathematical models, blended with the background along their elliptic-form margins; see the Figure 5.

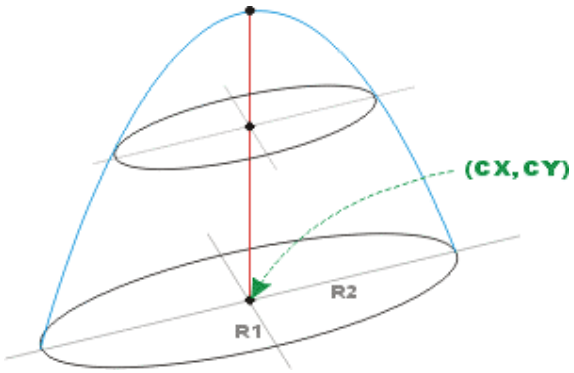


Figure 5 – Patches definition.

Each patch is specified by the following parameters:

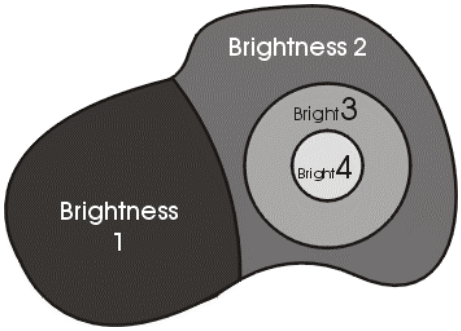
- The coordinates (Cx, Cy) of its center
- The sizes R1 and R2 a of the bigger and the smaller semi-axes of the base ellipse
- The direction a (angle with the x-axis) of the main semi-axis of the base ellipse
- The brightness value CB at the center
- The "margin" brightness value MB

In case of a color image the brightness values at the center of each patch are specified independently for each color separation.
On the aggregation level various default configurations can be adopted, simplifying data representation. For example, for patches smaller than a few pixels in size, the distinction between the sizes of the two semi-axes of the elliptic base is hardly visually appreciated. Consequently, a single size parameter can be used in this case, while the angle is omitted. In most situations the margin brightness value MB of the patch can be identified with the Background brightness at the patch center.

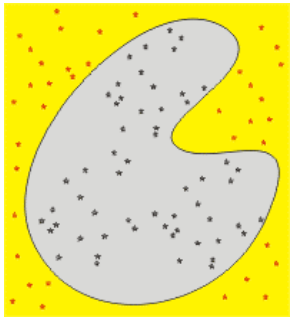
1.1.5. The Area Color AC ('Background')

(This term is used in Imaging in many different situations. Below we use it to denote the part of the Synthesized image, "complementary" to the Lines and patches. It can be called also the "slow scale image component". In the Synthesized Texture script the term "Area Color" is used as equivalent to the "background" in the present informal text).
Background is determined by the following elements:

1. All the separating Lines are excluded from the background domain.
2. Some of the image subparts, completely bounded by separating characteristic Lines or/and the image borders, are provided with their (single) global background brightness value GB, or a Color Gradient. See the figure below. (In fact, these subparts, called Synthesized Sub-Textures, play an important role on higher representation and animation levels. They are specified by Sub-Texture numbers, and the values GB are attached to the Sub-Textures).



3. A certain number of "background representing points" (AC – Area Color Points) is defined, each point carrying its brightness value. These values are further interpolated between the background representing points in such a way that the interpolation does not "cross" the separating Lines. See the figure below
4. The margin brightness values of the CPs of separating Lines are blended with the background along the margins of these separating lines.



The following parameters participate in the calculation of the background brightness values (as described in detail in the Procedure BB below):

- Geometric parameters of all the separating Lines.
- Margin brightness values (LB1 and RB1 above) of the CPs of all the separating Lines.
- List of the "background representing points" (ACs), each one given by its (x,y) coordinates and its brightness value.
- Single background global brightness values GB (or a Color Gradient) for some of the Sub-Textures. (In the parameters structure these values are associated with the Sub-Texture numbers, which, in turn, are associated with the Lines, bounding the Sub-Texture).

On the aggregation level various default configurations can be adopted, simplifying data representation. In particular, a regular grid of background representing points can be used, to eliminate the need to store coordinates of these points. (Notice, however, that in a general structure, allowing, in particular, for geometric transformations and animations of the images, all the geometric parameters must be represented in a geometrically invariant form, without reference to a fixed grid).

1.1.6. Crossings

Crossings of Lines are represented by crossings of their central lines and by blending of their brightness values near the crossings (Figure 6).

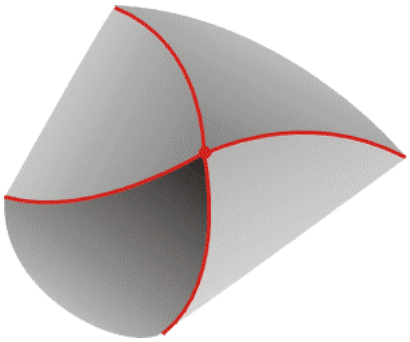


Figure 6 – Crossings.

In the data structure described above, crossings are represented by TPs (Terminal Points). At each TP, CPs are given for each of the Lines starting (or ending) at this TP. No compatibility conditions between these CPs are assumed in the basic Synthesized profile. **However, in most practical situations the side brightness (color) values on appropriate sides of the adjacent characteristic lines at their crossing are roughly equal. This fact is used on the higher aggregation level to reduce the data size (Figure 7)

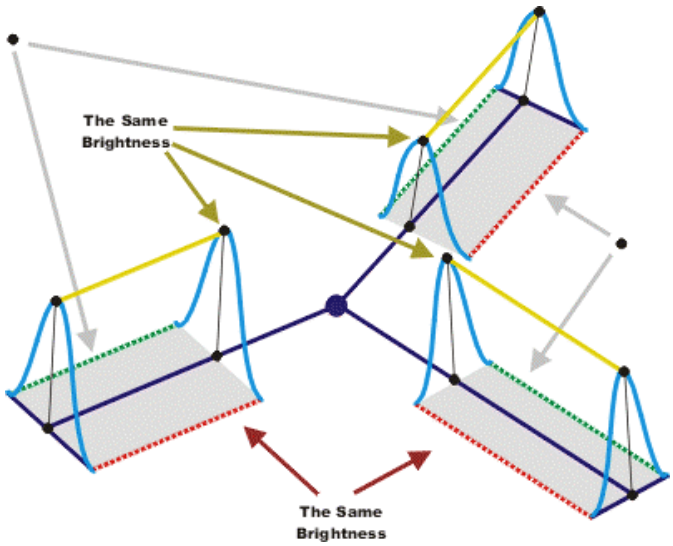


Figure 7 – Same brightness at crossing of multiple characteristic lines.

Splitting is used on the higher aggregation level to reduce the data size and to preserve the image visual continuity along the characteristic Lines. **

1.1.7. Ranges and Resolutions

1.1.7.1. The Range of Synthesized Parameters

A typical range of all the above parameters depends on the type of the images to be represented, on their resolution, on the specifications of the display and on visual conditions. Below we refer to the usual PC screen applications.
For Synthesized representation of high resolution images of the real world a typical range of parameters is as follows:

Length of a Line Segment	2 - 32 pixels
Height of a Line Segment	not larger than its length
Width of an edge	1 - 8 pixels
Width of a ridge	1.5 - 16 pixels
Density of the background representing points	between 1 per 4x4 pixels block to 1 per 32x32 pixels block
Color components	between 1 and 256 gray levels

For Synthesized representation of synthetic images (in particular, in creation of synthetic images using Synthesized authoring tools) the range of the Synthesized parameters is not strongly limited. Wide characteristic Lines, up to tens of pixels in width, can be used. Only a few of the background representing points (APs) may be needed (or no such points at all).

1.1.7.2. Default Resolution

The default resolution of all the geometric parameters is 1/16 of a pixel size.
The default resolution for each color component is 256 gray levels.
On higher levels of aggregation and quantization much lower resolutions can be allowed, according to a psycho-visual significance of each parameter. Also specifications of the display and expected visual conditions are taken into account.

2. SynthesizedTexture Rendering Algorithm

2.1. Synthesized Rendering Overview

The reconstruction algorithm starts with the input parameters, described above, and computes the brightness (color) value of the synthesized image at each of its pixels. It consists of the following principal parts:

- Computing brightness of Lines
- Computing brightness of Patches
- Computing brightness of the background
- Blending the computed brightness values into the final image

In the case of a color image these computations are performed independently for each color component.

2.1.1. Computing brightness of characteristic Lines (Procedure BL below)

In order to compute the brightness values of the cross-sections (the CPs) along the characteristic line, a coordinate system (u,t) is associated to its central line. Here u is the distance from the central line (with the sign) and t is the coordinate (along the line) of the projection on the central line.(Procedure DL). Using the coordinates (u,t) the brightness of the cross-sections (computed in the Procedure CS) is interpolated between the control points to a complete neighborhood of the characteristic line.

2.1.2. Computing brightness of patches (Procedure BP below)

The brightness of a patch is computed using a certain Gaussian – type brightness function, with the basis (in the image plane) – the ellipse, defined by the input patch parameters, and the height of the vertex equal to the input brightness parameter. A specific choice of the model Gaussian – type brightness function is influenced by the same considerations as the choice of the cross-sections model shapes. In particular, it can be taken to be a paraboloid with the elliptic basis and the vertex as specified by the input parameters.

2.1.3. Computing brightness of the background (Procedure BB below)

Background brightness values are computed by an interpolation between the brightness values of closed background components, the margin brightness values of the separating characteristic Lines and the brightness values of the background Area Points. The interpolation process is performed in such a way that the interpolation does not “cross” separating Lines.
In particular, this can be achieved by a “signal expansion algorithm”, in which the background representing points (and the margins of separating lines) transmit their brightness value to the neighboring pixels, which in turn transmit it to their neighbors etc. In this expansion the signal is transmitted only to the neighboring pixels, lying on the same side of each of the separating characteristic lines. Finally the background brightness value is computed at each pixel as a weighted average of the brightness values, received by this pixel in the process of signal expansion. The weights reflect the distances to corresponding background representing points.
The range of the signal expansion from each background representing point (AC) is limited to a certain constant, reflecting the density of the background representing points. Under a proper choice of this constant, the above algorithm is computationally very efficient, since only a few operations are performed per each pixel.

2.1.4. Blending the computed brightness values into the final image (Procedure MAIN below)

The final brightness values of a synthesized image are computed as the values of the characteristic lines, patches or the background at the interior pixels of the corresponding parts of the image. At the margin areas of the characteristic lines and of the patches the final brightness is computed by averaging their brightness values with the background brightness, with the weights compute in the Procedures WL and WP, respectively.

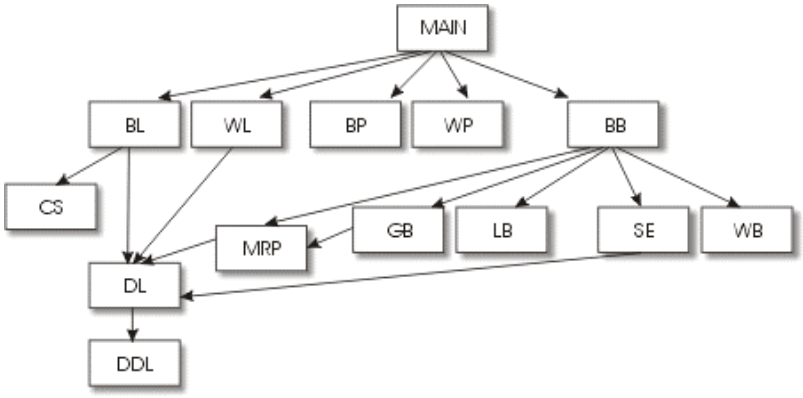
2.2. Detailed Description of the Reconstruction Algorithm

As it was described above, the reconstruction algorithm starts with the input parameters, as above, and computes the brightness (color) value of the synthesized image at each given point z in the image plane. It consists of the following principal parts:

- Computing brightness of characteristic lines (Procedure BL below)
- Computing brightness of patches (Procedure BP below)
- Computing brightness of the background (Procedure BB below)
- Blending the computed brightness values into the final image (The MAIN Procedure below)

In the case of a color image these computations are performed for each color component. In the description of each Procedure below, the names of the Procedures, called in the process of computations, are stressed by bold face.

2.2.1. Block Diagram of the Reconstruction Algorithm



2.2.1.1. Procedure MAIN: Computing the Final Brightness Values

For any point z in the image plane the final brightness B(z) of the Synthesized image at the point z is computed according to the following formula:

$$B(z) = \frac{WL(z)\cdot BL(z) + WB(z)\cdot BB(z) + \sum_s WP_s(z)\cdot BP_s(z)}{WL(z) + WB(z) + \sum_s WP_s(z)} \qquad (1)$$

Here $BB(z)$, $BL(z)$ and $BP_s(z)$ are the brightness functions of the background, of the characteristic lines and of the patches, computed in the Procedures **BB**, **BL** and **BP**, respectively, and the sum $\sum_s$ runs over all the patches P_s.

The weight functions $WL(z)$ and $WP_s(z)$ are computed in the Procedures **WL** and **WP**, respectively, and

$$WB(z) = 1 - max(WL(z), WP_s(z)).$$

Division by the sum of all weight functions in formula (1) guarantees that their sum is identically 1 and that formula (1) is a normalized averaging.

2.2.1.2.Procedure BL: Brightness of characteristic Lines

This procedure computes for any point z on the image the brightness BL(z), as defined by the cross-sections (CP) of the characteristic lines.
BL(z) needs to be computed only for those z which are "close enough" to at least one of the characteristic lines in the texture, as expressed by the weight function WL.
**The most intuitive and natural way to define the brightness of a characteristic line is to associate to it a coordinate system (uu, tt), with uu(z) the (signed) distance of the point z to the line along the normal direction, and tt(z) the length parameter along the curve of the orthogonal projection pp(z) of z onto the line.
Then the brightness cross-sections are computed according to the coordinate uu and interpolated along the line with respect to the coordinate tt.
The corresponding algorithm can be constructed. However, it provides some serious drawbacks:

1. Actual computing of the coordinates uu and tt is mathematically complicated task
2. Even for smooth curves without corners the normal direction is correctly defined only in a small neighborhood of the curve (of the size of a couple of pixels in realistic situations). Outside this neighborhood no natural mathematical solution exist for defining the normal, the projection etc.
3. For spline curves with corners between some of their links (which usually appear in realistic situations) the normal is not defined even locally. Once more, the situation can be corrected by using the mid of the corner angles, but the global difficulties of (2) remain and algorithms become rather complex.

Consequently, we show below an algorithm, which can be considered as an approximation to the “ideal one” above. Its main advantage is that the “coordinates” uu and tt (called below u and t) can be computed independently for each Line Segment (link) of all the collection of characteristic Lines. Moreover, the computation can be ultimately rendered as rather efficient (although the description below may look somewhat complicated).**

Below for any point z, u(z) is the “distance of z to characteristic lines”, S(z) is the closest link to z (with respect to the distance u) in the collection of characteristic lines, and t(z) is the parameter, measuring the projection of z onto S(z), rescaled to the segment [0, 1].
S(z), u(z) and t(z) are computed by the Procedure **DL**, described below.
The Procedure BL splits according to whether the link S(z) has a “free end” (i.e. an endpoint, not belonging to any other link) or not.

1. **S(z) does not have “free ends”.**

Let C_1 and C_2 denote the equations of the two cross-sections (normalized to the unit width, as described in the Procedure **CS** below) at the two endpoints of the link S(z). For u(z) > 0 let W_1 and W_2 denote the respective right widths RW_1 and RW_2 of the cross-

sections at these points. For u(z) < 0 let W_1 and W_2 denote the respective left widths LW_1 and LW_2 of the cross-sections at these points. Then in each case

$$BL(z) = t(z) \cdot C_1\left(\frac{u(z)}{W(z)}\right) + (1 - t(z)) \cdot C_2\left(\frac{u(z)}{W(z)}\right)$$

where W(z) is the interpolated width

$$W(z) = t(z) \cdot W_1 + (1 - t(z)) \cdot W_2$$

and the values $C_1\left(\frac{u(z)}{W(z)}\right)$ and $C_2\left(\frac{u(z)}{W(z)}\right)$ are computed by the procedure CS.

2. S(z) has a "free end".

If for this "free end" the parameter t is zero, the brightness BL(z) is computed as above for t(z) > 0. For $-DE < t(z) < 0$,

$$BL(z) = [1 + t(z)/DE] \cdot C_1\left(\frac{u(z)}{W(z)}\right) - [t(z)/DE] \cdot BM,$$

and for $t(z) < -DE$,

$$BL(z) = BM.$$

Here DE is a positive tuning parameter, defining the shape of the end of a characteristic line. BM is half of the sum of the brightness parameters LB2 and RB2 of the cross-section at the free end.
If for this "free end" the parameter t is one, t(z) is replaced by 1-t(z) in the above formula.

** The formula above provides one of possible choices of the shape of characteristic lines near their ends. It assumes that the cross-section brightness gradually descends to the "mid value" value BM inside the prescribed distance DE. Other shapes can be defined, by properly computing the width and the brightness in the neighborhood of the end point.**

2.2.1.3. Procedure CS: Color Profiles (Cross-Sections) of characteristic Lines

This procedure computes a brightness value of an edge or a ridge (unit width) cross-section CS(u) for any given cross-section "interior" brightness parameters, as described above, and for any value of u.
In the Procedure BL u is the distance u(z) to the line, normalized by the width W(z) of the line, so the width parameter W is taken into account inside the BL, and it does not appear below. Similarly, the margin brightness parameters LB1 and RB1 enter the computations in the Background brightness Procedure BB.

Edge Cross-Section.

Normalized edge cross-section NEC(u) is defined as follows (Figure 8):

$$NEC(u) = \begin{cases} u < -1, & 0 \\ u > 1, & 1 \\ -1 < u < 0, & 0.5(u+1)^2 \\ 0 < u < 1, & 1 - 0.5(u-1)^2 \end{cases}$$

Thus the recommended edge cross-section is composed of two symmetric parabolic segments.

For given brightness parameters LB2 and RB2, the value CS(u) is computed as

$$CS(u) = LB2 + (RB2 - LB2) \cdot NEC(u)$$

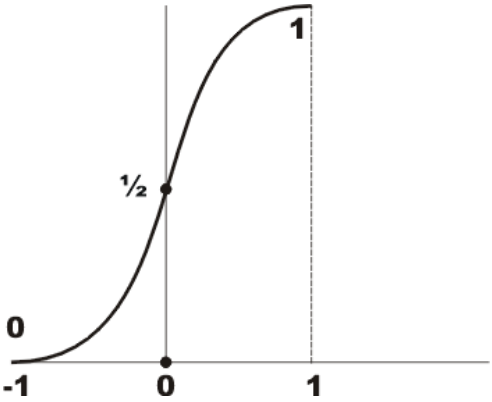


Figure 8 – Edge cross-section definition.

Ridge Cross-Section

As for edges, the width of the ridges is taken into account in the Procedure BL. Similarly, the margin brightness parameters LB1 and RB1 enter the computations in the Background brightness Procedure BB. Consequently the ridge cross-section computed in the current Procedure CS, is the same for separating and non-separating ridges, and is defined by the parameters LB2, CB and RB2, as follows (Figure 9):

$$CS(u) = LB2 + (CB - LB2) \cdot NEC(2u+1), \quad \text{for } u < 0, \text{ and}$$
$$CS(u) = RB2 + (CB - RB2) \cdot NEC(-2u+1), \quad \text{for } u > 0.$$

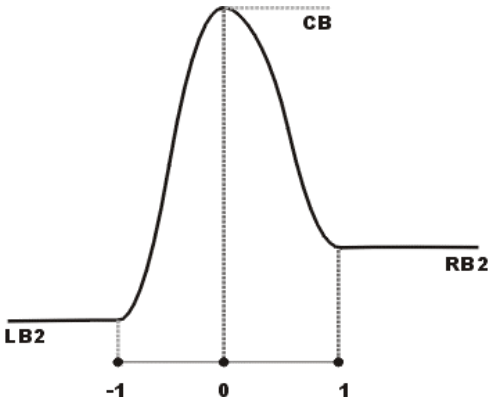


Figure 9 – Ridge cross-section definition.

** Thus the recommended ridge cross-section is composed of two edge cross-sections, properly aggregated.
In the process of the blending of these cross-sections with the background (which incorporates the margin brightness values LB1 and RB1) we get back essentially the same cross-section, as shown above (Fig. 10).**

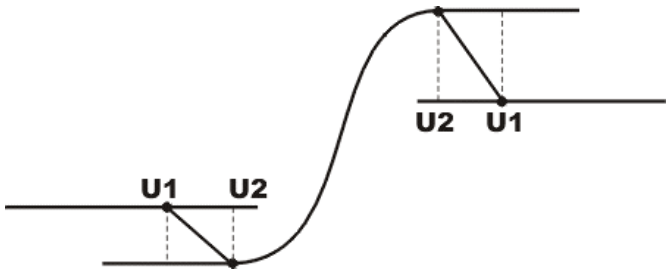


Figure 10 – Blending two cross-sections with the background.

2.2.1.4. Procedure WL: Weight Function of characteristic Lines

This block computes the weight function WL(z), which is used in a final blending of the characteristic lines with the background. The function WL(z) is equal to one in a certain neighborhood of the characteristic lines, and is zero outside of a certain larger neighborhood.

More accurately:

$$WL(z) = \begin{cases} 1 & |\boldsymbol{u(z)}| < UL_2 \cdot W(z) \\ \dfrac{W(z)\cdot UL_1 - |u(z)|}{W(z)\cdot (UL_1 - UL_2)} & UL_2 \cdot W(z) < |\boldsymbol{u(z)}| < UL_1 \cdot W(z) \\ 0 & |\boldsymbol{u(z)}| > UL_1 \cdot W(z) \end{cases}$$

The distance u(z) is computed in the Procedure **DL**.
UL_1 and UL_2 are tuning parameters; see the last section "Tuning Parameters".
Fig. 11 shows a typical cross-section and a general shape of the weight function WL(z).

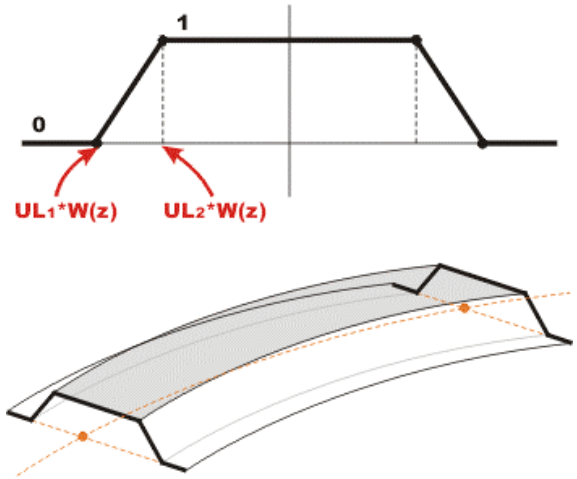


Figure 11 – Typical cross-section and general shape of WL(z)

2.2.1.5. Procedure DL: Distance to characteristic Lines

This Procedure computes for any point z in the texture:

1. The point p(z) on the characteristic lines which is nearest to z, i.e. the "projection" p(z) of z onto the set of characteristic lines.
2. The distance u(z) between z and p(z).
3. The link S(z) on which p(z) resides.
4. The proportion t(z) in which p(z) divides the link S(z).

Note u(z), p(z) and t(z) are NOT exactly the Euclidean distance, the corresponding mathematical projection and proportion respectively; however, in most cases they give a reasonable approximation for these mathematical entities.

These data are computed in the following steps:

1. For each link (Line Segment, LS) S_i in the texture, the corresponding $p_i(z)$, $u_i(z)$, $t_i(z)$ are computed in Procedure **DDL** (See the Synthesized-C figure below)
2. S(z) is defined as the link S_j, for which the minimum of the absolute values $|u_i(z)|$ is attained (See the figure Synthesized-D below)
3. u(z) is defined as the function $u_j(z)$ for the link S_j = S(z)
4. t(z) is defined as $t_j(z)$ for the above link S_j

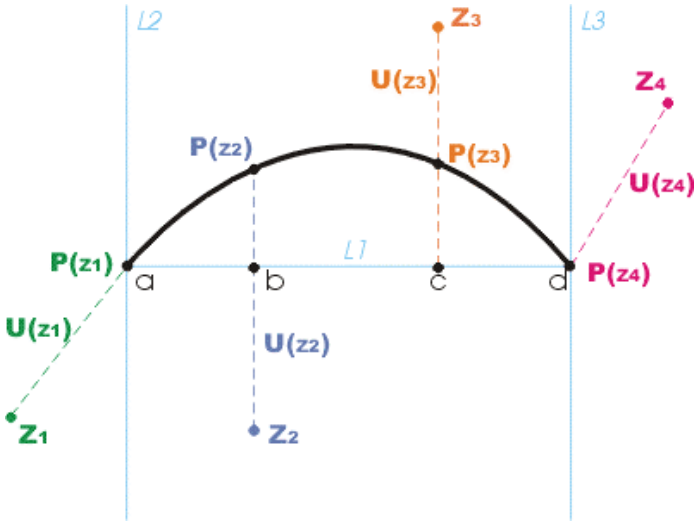


Figure - Synthesized-C

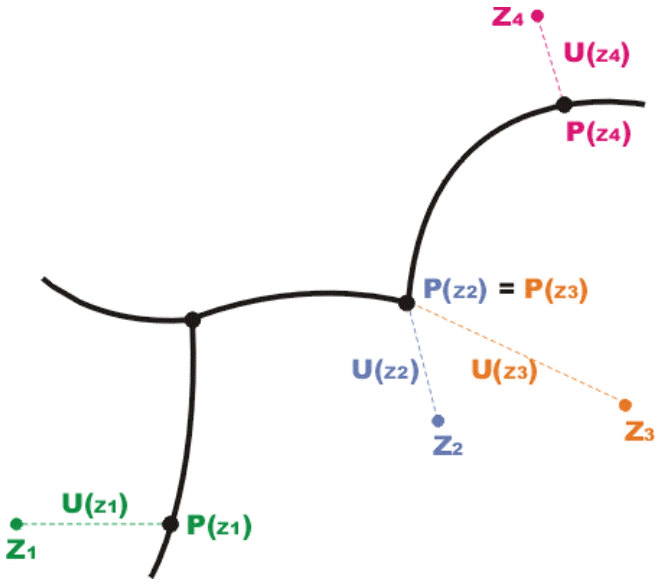


Figure - Synthesized-D

2.2.1.6. Procedure DDL: Distance to a Line Segment

This procedure computes for any point z its (signed) distance u(z) to a given link S (Line Segment, LS), the projection p(z) of the point z onto the link S and the parameter t(z). The Procedure is essentially represented on figure Synthesized-C above (which shows, in particular, equidistant lines for the points z1 and z4.

The straight oriented segment [a, d], joining the end points of the link S is constructed, with the orientation, induced from the orientation of the Line, containing S. l_1 is the

straight line, containing the segment [a, d]. l_2 and l_3 are the straight lines, orthogonal to l_1 and passing through a and d, respectively.

Now, for any z in the image plane, the function u(z) is constructed as follows:

For z between l_2 and l_3, the absolute value |u(z)| is the length of the segment, joining z and S and orthogonal to l_1 .

For z left to l_2 , |u(z)| is defined as follows:

$$|u(z)| = \left[d(z,l_1)^2 + Dd(z,l_2)^2\right]^{/2}.$$

Here d(z, l_1) and d(z, l_2) are the distances from z to $l_1$and

l_2 respectively. D is a tuning parameter, with a typical value D = 4.

For z right to l_3 , |u(z)| is defined as

$$|u(z)| = \left[d(z,l_1)^2 + Dd(z,l_3)^2\right]^{/2}.$$

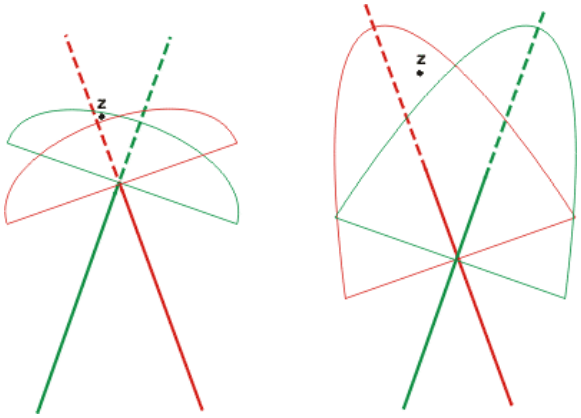
Let *ll* be an oriented line, formed by the interval of the line l_1 from infinity to a, then by the link S from a to d, and then by the interval of the line l_1 from d to infinity.

For z right to *ll* (with an orientation as above) the sign of u(z) is "+". For z left to *ll*, the sign of u(z) is "-".

For z between l_2 and l_3, the projection p(z) is the intersection point of S and of the segment, joining z and S and orthogonal to l_1 . For z left to l_2 , p(z) is a, and for z right to l_3 , p(z) is d.

For any z, t(z) is the proportion, in which the projection of z onto the line l_1 subdivides the segment [a, d]. For example, for the point z_2 and z_3 on Fig Synthesized-D above. $t(z_2)$ = (b-a)/(d-a), and $t(z_3)$ = (c-a)/(d-a), respectively. For z left to l_2 , t(z) < 0, and for z right to l_3 , t(z) > 1.

** The special form of the function u(z) above (for z outside the strip between l_2 and l_3) is motivated by the following reason: when computing in the Procedure BL the brightness of the line near a sharp corner, the form of the distance function u(z) determines which link will be taken as the closest to the points in the sector stressed on the Figure below. For the distance, computed as above, with the parameter D > 1, this choice is matched with the sign of u(z), as defined above. If we would have chosen D < 1, for z in the sector stressed on the Figure below, the choice of the nearest link, together with the proposed computation of the sign of u(z), would produce a color from the incorrect side of the line. See the figure below.**



2.2.1.7. Procedure BP: Brightness of Patches

Let Cx, Cy, R1, R2, a, CB and MB be the parameters of a certain patch P_s, as described above.
Let M be the linear transformation of the plane, transforming the basis ellipse of the patch to the unit circle. M is a product of the translation by (-Cx, -Cy), the rotation matrix to the angle –a, and the rescaling 1/R1 and 1/R2 times along the x and y axes, respectively. If we put for

$z = (x, y)$, $(x'(z), y'(z)) = M(x, y) = M(z)$,

then the equation of the basis ellipse of the patch is given by

$x'(z)^2 + y'(z)^2 = 1$.

The brightness function $BP_s(z)$ of the patch is then given by

$BP_s(z) = 0$ for $x'(z)^2 + y'(z)^2 > UP1^2$,

$BP_s(z) = MB$ for $1 < x'(z)^2 + y'(z)^2 < UP1^2$, and

$$BP_s(z) = MB + (CB - MB) \cdot (1 - x'(z)^2 - y'(z)^2), \text{ for } x'(z)^2 + y'(z)^2 < 1.$$

Here UP1 > 1 is a parameter. See the Figure below.

2.2.1.8. Procedure WP: Weight Function of Patches

The weight function $WP_s(z)$ for a patch P_s as above is defined by

$WP_s(z) = 0$ for $uu(z) > UP1$,

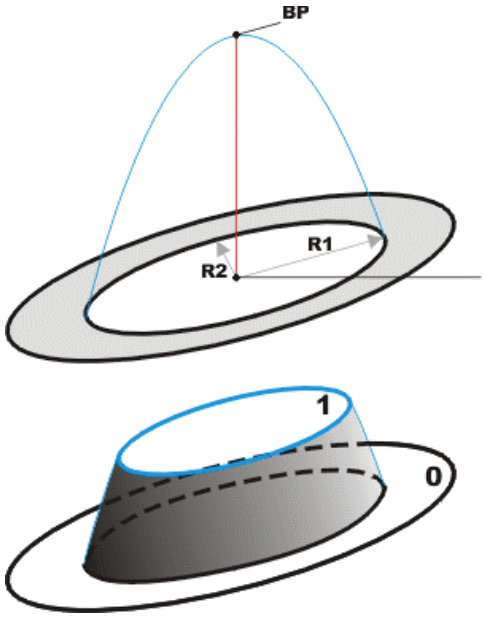
$WP_s(z) = 1$ for $uu(z) < UP2$, and

$WP_s(z) = (UP1 - uu(z))/(UP1 - UP2)$

for uu(z) between UP2 and UP1,

where uu(z) denotes the square root of $x'(z)^2 + y'(z)^2$.

Here UP2, 1 < UP2 < UP1, is another tuning parameter. See the figure below.



2.2.1.9. Procedure BB: Brightness of Background

This Procedure computes the brightness value of the background at any point z of the image. This value is obtained as a result of interpolation between the "global" background brightness values, the margin brightness values of the characteristic Lines and the brightness values at the background representing points (ACs – Area Color Points),. The main difficulty is that the interpolation is not allowed to cross the separating lines. To overcome this difficulty a special "distance" d between the points on the image is introduced, which is the length of the shortest pass, joining these points, and not crossing separating Lines, and which is computed in the Procedure **SE below. Then averaging weights are computed through the distance d.**

This block uses as an input a certain collection of the background representing points Z_i, (containing the input background representing points, as described above, and the margin representing points, produced by the block "**MRP**', described below). At each point Z_i the brightness value Bb_i is given.

The background brightness value BB(z) is finally produces by the block BB as follows: BB(z) is the weighted sum of the global brightness GB and of the Local brightness functions $Bb_i(z)$ over all the background representing points Z_i :

$$(2) \qquad BB(z) = (1/S_1(z))[WG(z)BG + \sum_i WR(d(z,Z_i))Bb_i(z)]$$

$$\text{Here } S_1(z) = WG(z) + \sum_i WR(d(z,Z_i)),$$

so the expression (2) is normalized to provide a true averaging of the corresponding partial values.

The global brightness value BG is provided by the Procedure **GB** below, or by any of the Gradient Nodes.

The computation of the Local brightness functions $Bb_i(z)$ is performed in the Procedure **LB** below.
The distance functions $d(z, Z_i)$ are computed in the Procedure **SE** below.
The computation of the weight functions $WR(d(z, Z_i))$ is performed in the Procedure **WB** below.
The weight GW(z) of the global background value GB is defined as
$GW(z) = 1 - \max_i WR(d(z, Z_i))$.

GW(z) is zero at any z, where at least one of the weights of the representing points is 1, and GW(z) is one at any z where all the weights of the background representing points vanish.

2.2.1.10. Procedure GB: Global Brightness of Background

This Procedure computes the global background value GB, which appears in the expression (2) in the Procedure BB.

By definition, if the point z is inside the background region of a Sub-Texture number r, for which the global value GBr is defined, GB is equal to this global value GBr. If the point z is inside the background region of a Sub-Texture, for which the global background value is not defined, GB is equal to the default global value DGB. If DGB is not defined, GB is equal to zero. Alternatively, Color Gradients can be used.

The current procedure consists in a signal expansion that transmits to each pixel its Sub-Texture number. We describe it shortly, since it essentially belongs to a higher data representation level.

First the procedure MRP is applied, which creates margin representing points, carrying the corresponding Sub-Texture numbers. These numbers are taken from the corresponding poly-links.

Second, the Signal Expansion Procedure is applied to the margin representing points, essentially as in the block SE, with the following difference: only the marking and the number of the Sub-Texture is transmitted between the pixels on each step of signal expansion.

As this procedure is completed, each pixel in the image memorizes the number of the Sub-Texture, to which it belongs.

2.2.1.11. Procedure LB: Local Brightness of the Background

Two types of the local brightness functions $Bb_i(z)$ are used. For the first type (zero order) $Bb_i(z)$ is identically equal to the input brightness value Bb_i at the point Z_i .
For the second type (first order) $Bb_i(z)$ is equal to $L_i(z)$, where $L_i(z)$ is the linear function, such that

$L_i(Z_i) = Bb_i$

and Li provides the best approximation of the input brightness values at the N nearest to Z_i background representing points. The choice of the type of the local brightness function is determined by the flag LBF: LBF is zero for the zero order and LBF is one for the first order of the functions $Bb_i(z)$. Here N is an integer valued tuning parameter.
** Typical value of N is 4 or 9: usually the background representing points form a regular or an almost regular grid, and the nearest neighbors are taken at each point Z_i to construct the linear function $L_i(z)$.**

2.2.1.12. Procedure WB: Weights for the Background

As implied by the form of the expression, the weights $WR(d(z, Z_i))$ depend only on the distance $d(z, Z_i)$ from the point z to the background representing point Zi. The model function of one variable WR is specified by three tuning parameters UB1 and UB2, UB1 > UB2 > 0, and BVS (Background Weight smoothness), 0 < BVS < 1, and is defined as follows:

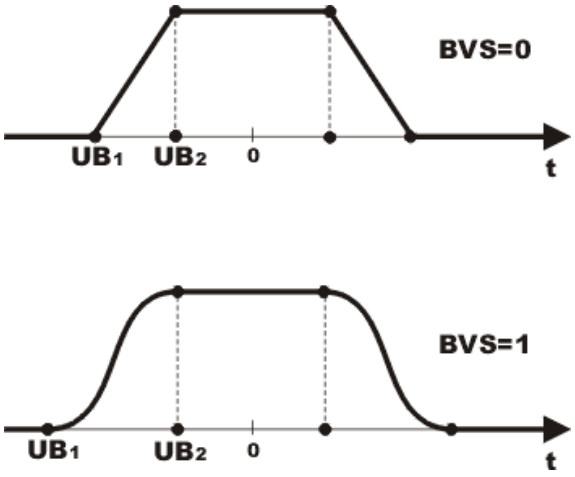
$WR(t) = 0$ for $|t| > UB1$,

$WR(t) = 1$ for $|t| < UB2$, and

$WR(t) = BVS(3v^2 - 2v^3) + (1 - BVS)v$, for $UB2 < |t| < UB1$,

where $v = (|t| - UB2)/(UB1 - UB2)$.

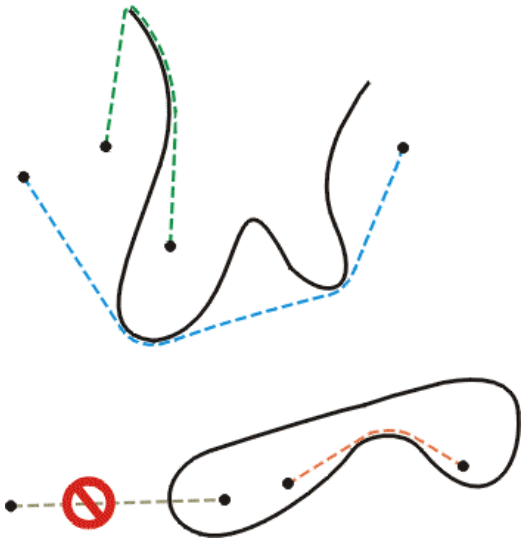
See the Figure below.



2.2.1.13. Procedure SE: Signal Expansion

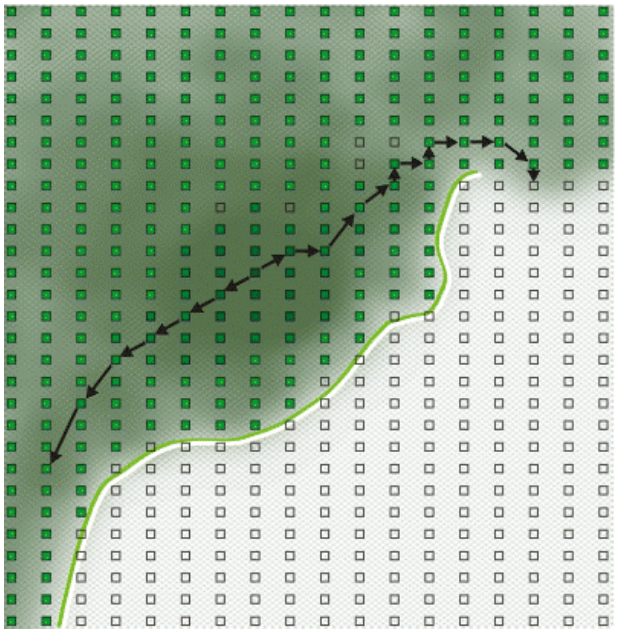
Let D denote the domain of the Synthesized image, with “cuts” along all the separating Lines PL_i. For any two points z_1, z_2 in D the distance $dd(z_1, z_2)$ is defined as the (Euclidean) length of the shortest path, joining z_1 and z_2 in D and avoiding all the cuts PL_i. See the figure below.
**It might be assumed that the influence of the color at z_1 to the color at z_2 decreases as the distance $dd(z_1, z_2)$ increases. However, a precise computation of the distance dd is a

rather complicated geometric problem. Consequently, we use instead of the distance $dd(z_1, z_2)$ its approximation $d(z_1, z_2)$, which is computed through a "signal expansion algorithm", as described below.**



The block SE computes the distance $d(z_1, z_2)$ for any two points z_1 and z_2 in the image plane. The algorithm is not symmetric with respect to z_1 and z_2: in fact, for a fixed point z_1, the distance $d(z_1, z_2)$ is first computed for any pixel z_2 of the image. Then an additional routine computes $d(z_1, z_2)$ for any given z2 (and not necessarily a pixel). Below the notion of a "neighboring pixel" is used. It is defined as follows: for z not a pixel, the four pixels at the corners of the pixel grid cell, containing z, are the neighbors of z. For z a pixel, its neighbors are all the pixels, whose coordinates in the pixel grid differ by at most one from the coordinates of z.
Below we assume that a certain data structure is organized, in which to any pixel p on the image plane a substructure is associated, allowing to mark this pixel with certain flags and to store some information, concerning this pixel, obtained in the process of computation. We do not specify here this data structure, using instead expressions like "the pixel p is marked", "the pixel p memorizes…" etc.
Now for z_1 and z_2 given, the distance $d(z_1, z_2)$ is computed in the following steps:
For any pixel p the distance u(p) to the *separating* Line PL_i is computed and stored at this pixel. The computation of u(p) is performed by the procedure **DL**, described above, applied only to separating poly-links PL_i .
Those pixels p, for which $u(p) < FU$, are marked as "forbidden" pixels. The forbidden pixels are excluded from all the rest of computations, and those pixels that are not forbidden, are called below "free" ones. Here FU is a tuning parameter.
Now the proper "signal expansion" starts. In the first step each of the free neighbor pixels of z_1 is marked, and this pixel memorizes its Euclidean distance from z_1 as the auxiliary distance dd, to be computed. Generally, in the k-th step, any free unmarked pixel p, at least one of whose free neighboring pixels was marked in the previous steps, is marked. This pixel memorizes as its auxiliary distance dd(p) from z_1, the minimum of dd at the neighboring free pixels plus the Euclidean distance of p to the neighboring pixel, at

which the minimum is attained. This process is continued the number of steps, equal to the maximal dimension of the image (in pixels). After it is completed, each free pixel p on the image plane memorizes its auxiliary distance dd(p) from z_1.
Now for any given point z_2 on the image plane, its distance $d(z_1, z_2)$ from z_1 is computed as maximum of D1 and D2, where D1 is the Euclidean distance of z_2 to z_1 , and D2 is the minimum over the free neighboring to z_2 pixels p, of dd(p) + the Euclidean distance of z_2 to p.
This completes the computation of the distance $d(z_1, z_2)$.
See the Figure below.
** The tuning parameter FU determines the size of a neighborhood of the separating Line, where all the pixels are marked as forbidden. Taking any value of FU, larger than 0.8, excludes a possibility of signal expansion crossing separating lines. Indeed, for any two neighboring pixels, which are on different sides of a separating line, at least one is closer to the line than 0.8 and hence is marked as forbidden. To provide stability of finite accuracy computations a bigger value of U may be taken. However, in this case signal expansion will not pass a “bottle-neck” between two separating lines, which are closer to one another than 2FU. Normally such regions will be covered by the cross-sections of these lines. However, a sub-pixel grid can be used to guarantee that signal expansion passes thin “bottle-necks”



Various implementation issues.

Computation of the distance $d(z_1, z_2)$ and its usage inside the background grid interpolation form one of the central parts of the Synthesized image reconstruction. Consequently, the efficiency of the implementation of these blocks is crucial for an overall efficiency of the reconstruction.

A straightforward implementation of the signal expansion algorithm, as described above, is not optimal. However, a number of rather natural and simple modifications bring the efficiency of the signal expansion to only few operations per pixel.

1. Multi-scale implementation. The image is subdivided into blocks in 2 – 3 scales (for example, blocks of 16x16, 8x8 and 4x4 pixels). First signal expansion is performed between the highest scale blocks (say, 16x16), exactly as described above. Forbidden are the blocks, crossed by separating characteristic lines. In the second stage the forbidden blocks are subdivided into 8x8 sub-blocks, and the expansion is performed for them. The new forbidden sub-blocks are subdivided into the 4x4 ones, and the expansion is repeated. In the last stage the expansion is completed on the pixels level.
2. For an application to the background grid interpolation, the distance $d(z_1, z_2)$ has to be computed only till it riches the threshold UB1 (since for larger distances the weight functions vanish). This fact allows one to restrict the number of steps in signal expansion to UB1 + 1.
3. Signal expansion and memorization of the distances at the free pixels can be implemented for all the background representing points at once (especially since the above distance restriction usually makes for any pixel only the information relevant, concerning a few neighboring background grid points).
4. In the process of signal expansion, all the mathematical data required in the interpolation block (like Euclidean distances and weight functions) can be computed incrementally in a very efficient way.**

2.2.1.13. Procedure MRP: Margin Representing Points

This Procedure constructs a grid of representing points on the margins of all the characteristic Lines together with the background brightness values (APs) at these points. Later the constructed margin points are used (together with the original background representing points) in the interpolation process in the block BB.
The margin representing points Mz_j are produced in the following steps:

1. On each Line, the points w_k are built with the distance UM1 from one another, starting with one of the ends (the distance is measured along the Line). The last constructed point on each Line may be closer to the end Terminal Point of this Line, than UM1.
2. At each w_k the line l_k orthogonal to the Line and intersecting it at w_k is drown. If w_k turns out to be a vertex of the Line with a nonzero angle between the adjacent Line Segments LS (links), or a crossing, l_k is taken to be the bissectrice of the corresponding angle.
3. On each line l_k two points (one point in the case of the bissectrice of the crossing joint angle) are chosen at the distance $UM2 \cdot W(w_k)$ from the intersection point w_k of l_k with the Line (from the crossing w_k, respectively). All the chosen points, in a certain chosen order, form the output margin representing points Mz_j.
4. At each margin representing point Mz_j constructed, as described above, the corresponding margin background brightness value Bb_j is computed by

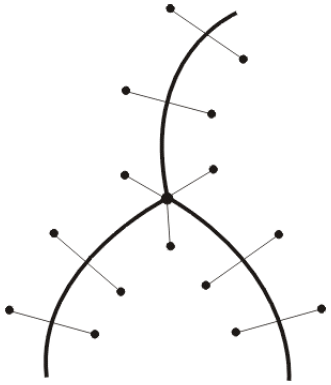
$$Bb_j = tA + (1-t)B,$$

where A and B are the margin values (LB1 or RB1, respectively) of the cross-sections at the ends of the Line Segment $S(Mz_j)$, nearest to the point Mz_j, and $t = t(Mz_j)$.

$S(Mz_j)$ and $t(Mz_j)$ are computed by the Procedure **DL.**

In the current Procedure UM1 and UM2 are tuning parameters (the first one absolute and the second relative to the width), satisfying UM1 < UB1, 1 < UM2 < UL1.

See the figure below.



The four figures below illustrate influence of different elements of the background for different values of tuning parameters.
Figure AA shows an extrapolation of the margin values of the characteristic Line.
Figure BB illustrates the influence of the background representing points (APs).
Figure CC shows representing points with a larger parameter UB1.
Figure DD shows a patch.

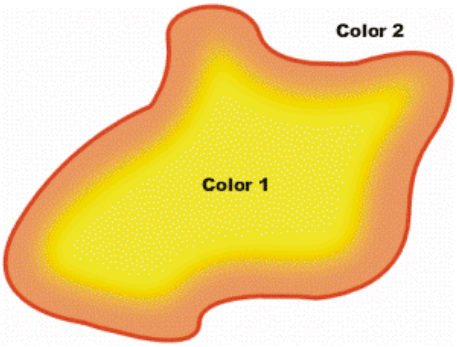


Figure Synthesized-AA

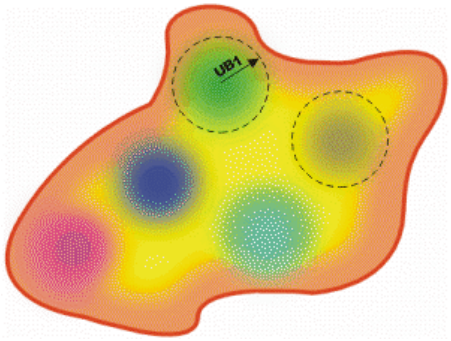


Figure Synthesized-BB

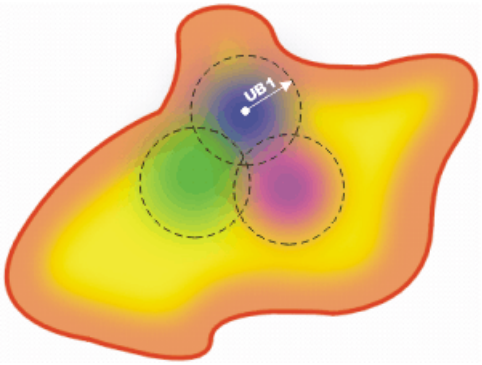


Figure Synthesized-CC

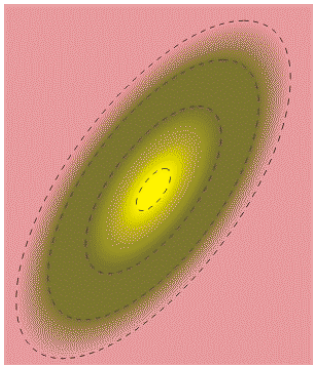


Figure Synthesized-DD

2.3. SynthesizedTexture Rendering - Tuning Parameters

All the tuning parameters, listed below, are not changed in the process of sending and playing of images and animations. However, their tuning for specific visual displays (and for specific classes of images) may improve the visual quality and the player efficiency.
UL1 – the distance from the characteristic Lines (as a proportion to the width of the line), within which the CP (cross-section) brightness BL(z) is computed. It coincides with the exterior size of the lines averaging region. Used in the Proceduress BL and WL.
UL2 – the interior size of the lines averaging region (as a proportion to the width of the line). Used in the Procedure WL
UP1 and UP2 are the corresponding parameters for patches (referring to the relative distance with respect to the patch size). Used in the Procedures BP and WP.
UB1 and UB2 are the (absolute) size parameters for the weight functions of the background representing points, APs. The parameter BVS characterize the smoothness of these weight functions. Used in the Procedure WB.
The flag LBF specifies the order (0 or 1) of the local brightness representation. Used in the Procedure LB.
The integer parameter N is the number of neighboring background representing points taken to define the local linear approximations of the brightness. Used in the Procedure LB.

FU is the distance of the separating lines at which pixels are marked as “forbidden”. Used in the Procedure SE.
UM1 and UM2 are the absolute and the relative to the width parameters in construction of margin representing point. Used in the Procedure MRP.
D > 1 is the parameter, defining the “asymmetry” of the distance to the Line Segment, computed in its side areas. Used in the Procedure DDL.
DE > 0 is a parameter, defining the length of the end area of characteristic Lines. Used in the Procedure BL

3. High Order Detection of Ridges

3.1. Visual specification of a ridge

Visually, ridges are specified as narrow strips, separated from the background by their (locally uniform) brightness or color. The central line of the ridge is the line, which represents in the most accurate way its geometric shape. It consists of the points, where the brightness in each orthogonal direction attains its extremal value.

Ridges present one of the most important examples of uniform characteristic lines: the visual pattern along the ridge qualitatively repeats itself. This pattern is represented by the cross-section, having the brightness extremum at the central line of the ridge.

Within this basic pattern, quite different shapes may appear as ridge cross-sections. Experiments show that human visual perception is highly sensitive to the ridge cross-section shape. To guarantee a visually faithful reconstruction of a ridge the geometric accuracy in the cross-sections identification has to be at least of order 0.1 pixel and the brightness approximation accuracy has to be at least of order of a few gray levels.

As far as geometric accuracy of ridge identification is concerned, it also has to be at least of order 0.1 pixel, to provide a visually faithful reconstruction of, for example, text patterns in a scale of several pixels. The same accuracy is typically required at highly curved regions of the ridge, its “corners” etc.

These very strict accuracy requirements make application of high order high accuracy methods inevitable in ridge detection, which assumes as one of its goals a visually faithful reconstruction of the image along the ridges. In particular, this fact justifies a necessity of a very detailed mathematical analysis of ridges, as they are expressed in properties of the brightness function.

3.2. Mathematical description of ridges through curvatures; equivalence of several definitions.

One of the difficulties with the mathematical treatment of ridges is that the definition of a ridge in visual terms, given above (visually, ridges are specified as narrow strips, separated from the background by their (locally uniform) brightness or color) can be translated into mathematical terms in several apparently not equivalent ways.

Consider the brightness $z = f(x,y)$ at each point (x,y) of the image as a continuous and smooth function, represented by the surface of its graph in the three-dimensional space. In this interpretation ridges appear as narrow elongated channels (for dark ridges) or clips (for bright ones) on the brightness surface. It seems to be intuitively clear that at the points of the ridge's central line, this surface is maximally curved in the direction, orthogonal to the ridge. In other words, the curvature of the brightness surface in the direction, orthogonal to the ridge's central line, is maximal among all the directions. In differential-geometric terms this means that one of the principal curvature directions of the surface (corresponding to the bigger main curvature) is orthogonal to the ridge, while the other principal curvature direction (corresponding to the smaller main curvature) is parallel to the ridge. This condition asserts that any ridge is a so-called "curvature line" on the brightness surface; it can be expressed as a certain first order differential equation. It is referred below as **Condition A.**

On the other hand, from the definition in visual terms it seems to be equally clear, that the brightness function, restricted to any line, orthogonal to the ridge, must have an extremum at the points of the ridge's central line. This implies that the derivative of the brightness function in this orthogonal direction is zero, or, equivalently, the gradient of the brightness function is parallel to the ridge. This means that any ridge is a so-called "gradient line" of the brightness function. Once more, this condition, referred below as **Condition C**, can be expressed as a certain first order differential equation.

It is also very important to have a mathematical definition of the ridge by an "algebraic" (and not differential) equation in terms of the brightness function f and its derivatives. Indeed, solution of differential equations requires global data processing, while "algebraic" equations can be solved locally. The last case is strongly preferable because of stability and efficiency reasons.

Such an "algebraic" condition can be produced as follows: we combine two of the "intuitive" observations above (that the orthogonal direction to the ridge is the bigger principal curvature direction, and that the derivative of the brightness function in this direction at the ridge point must be zero), but not insist that this principal curvature direction is strictly orthogonal to the ridge.

As a result we get the following:

Condition B.

At each point of the ridge's central line the derivative of the brightness function in the bigger principal curvature direction is zero.

This condition will be used below as the main working definition of the ridges. It can be written explicitly as a certain "algebraic" (and not differential!) equation, involving the brightness function f and its derivatives up to the second order. This equation is explicitly given in the Addendum 1.

It turns out that this condition B, strictly speaking, is **mathematically incompatible with the conditions A and C**: the bigger principal curvature directions in general are not orthogonal to the lines, defined by the equation of the condition B (and the gradient directions are not in general tangent to these lines). This incompatibility occurs as a result of several rather subtle mathematical effects: interaction between the first and the third order derivatives of the brightness function and the influence of the high curvature of the ridges themselves. In particular, for the brightness function a quadratic polynomial (and in a "second differential eigenvalues" setting, described below), ridges are straight lines, satisfying all the conditions A, B and C.

The following basic mathematical fact justifies our choice of the mathematical interpretation of visual ridge definition. It also explains, to a certain extent, why our intuitive appreciation of ridges translates into different mathematically incompatible conditions.

If the gradient of the brightness function is small along the ridge, then the curve, defined by the Condition B, satisfies approximately also the conditions A and C.

A rigorous proof of this fact is given separately. It is obtained via the "jet formulae", presented below. In particular, the analysis of the deviation can be performed via jet expression for the coefficient c1. In the same way certain geometric explanation of the above fact can be given, the role of the third derivative and of curved ridges with a "big" gradient can be clarified.

Using an "algebraic", and not a differential equation to define ridges is advantageous in several aspects.

First of all, solving differential equation of conditions A and C is basically unstable from the point of view of our main purpose – ridge detection: starting at a certain point of a ridge both curvature and gradient line can deviate far away from the ridge in a short time.

Secondly, the "algebraic" definition requires only local computations.

The conditions above have been formulated in terms of the differential-geometric curvature directions of the graph of the brightness function, considered as a surface in the three-dimensional space.

Similar conditions can be given in terms of the eigenvalues and eigendirections of the second differential of the brightness function f, without explicit referring to the differential geometry of its graph. As far as the ridges are concerned, our intuition seams to equally support each of the approaches.
Computationally the second one is much simpler.

It is not clear a priori that these two approaches are mathematically compatible, and if not, to what extent they may disagree in practically important situations. Consequently, mathematical comparison of the above conditions is important for a practical implementation: it allows one to use a "correct" translation of the intuitive notion of the ridge into mathematical terms, and to significantly simplify computations. This comparison and analysis, is given below. It produces, in particular, the following important fact:

Ridges, defined by the Condition C, are the same for "differential-geometric" curvature directions and for the eigendirections of the second differential of the brightness function f.

Consequently, all the specific computations below are performed within the second approach.

The above mathematical description was given under a simplifying assumption that the brightness function of the processed image was continuous and differentiable, and that all the required derivatives of this function were readily available. In practice the brightness function is given on a discrete grid of pixels, so to evaluate its derivatives a certain continuous mathematical approximation of this function must be produced. For most of real life images (except the synthetic ones) the brightness function is noisy. It is well known that numerical computation of high order derivatives is very sensitive to noise. This sensitivity must be taken into account in the processing.

There are many known methods of relatively stable computation of derivatives of discrete functions. These methods are based on various forms of local or global approximation by standard mathematical tools, like polynomials, trigonometric functions, wavelets etc. Each of these methods can be successfully used. In particular, any required derivative of the brightness function can be produced at each pixel of the image by a convolution of the brightness with an appropriate convolution mask. Usually a special case of convolution is used, based on a weighted polynomial approximation of the brightness function on square windows of different size. This method is described in detail below.

As a result, at each pixel of the image (and if necessary, at intermediate points) the approximate values of the derivatives of the brightness function are produced. However,

these derivatives values once more form a discrete set of data, and not the continuous functions. Consequently, the formulae and equations of the Section 3.3 below are not directly applicable. These formulae and equations have to be translated into a form, where their input will be the set of derivatives values at discrete points. Their output will represent (up to the prescribed order) the ridge elements, identified in the vicinity of these given points.

The formulae and expressions of the required type are known in Mathematical Analysis, more specifically, they are widely used in the so-called Jet Algebra computations, appearing in Singularity theory. In each specific problem these expressions have to be developed independently, but being developed, they can be applied to the whole class of similar situations. Being usually rather complicated algebraically, these expressions have to be specially arranged for efficient and robust numerical calculations. It is here that multi-order considerations become especially important.

Below all the "Jet formulae" necessary for the second order ridge elements detection are produced on the base of the global mathematical equations.

3.3. Differential-Geometric description of Ridges

3.3.1. Definitions and Mathematical Background

Definition 1 *Let* f *be a brightness function,* κ_1, κ_2 *– principal curvatures of the surface* $z = f(x, y)$ *and* $\underline{du} = (du_1, du_2)$ *the direction of maximal curvature* k_1. *The curve* L *is called a ridge, if the directional derivative* $D_{\underline{u}} f$ *vanishes at each point* $(x, y) \in L$ *:*

(1.1) $$D_{\underline{u}} f = f_x du_1 + f_y du_2 = 0$$

It is well known that the equation for principal directions can be written in the form

(1.2) $$(Fg - Gf)\lambda^2 + (Eg - Ge)\lambda + (Ef - Fe) = 0, \qquad \lambda = \frac{du_2}{du_1},$$

or

(1.3) $$\begin{vmatrix} du_2^{\,2} & -du_1 du_2 & du_1^{\,2} \\ E & F & G \\ e & f & g \end{vmatrix} = 0 \quad ,$$

where $I = Edu_1^{\,2} + 2Fdu_1du_2 + Gdu_2^{\,2}$ and $II = edu_1^{\,2} + 2fdu_1du_2 + gdu_2^{\,2}$ are the first and the second fundamental forms of the surface.

The equation (1.2) determines two directions $\frac{du_2}{du_1}$ and $\frac{dv_2}{dv_1}$, in which the normal curvature κ obtains its extreme values.

In the case of $z = f(x,y)$ we obtain the following expressions of I and II :

$$E = 1 + f_x^{\,2}\ , \qquad F = f_x f_y\ , \qquad G = 1 + f_y^{\,2}$$

$$e = \frac{f_{xx}}{\sqrt{1 + f_x^{\,2} + f_y^{\,2}}}, \qquad f = \frac{f_{xy}}{\sqrt{1 + f_x^{\,2} + f_y^{\,2}}}, \qquad g = \frac{f_{yy}}{\sqrt{1 + f_x^{\,2} + f_y^{\,2}}}.$$

Then the equation (1.2) becomes

$$\begin{aligned}(1.2')\qquad &\left[f_x f_y f_{yy} - (1 + f_y^{\,2}) f_{xy}\right]\lambda^2 + \left[(1 + f_x^{\,2}) f_{yy} - (1 + f_y^{\,2}) f_{xx}\right] \\ &\qquad + \left[(1 + f_x^{\,2}) f_{xy} - f_x f_y f_{xx}\right] = 0\end{aligned}$$

Proposition 1. *If the curve L is a ridge, then it necessarily satisfies the equation*

$$(1.4)\qquad f_{xy}(f_y^{\,2} - f_x^{\,2}) + f_x f_y \left(f_{xx} - f_{yy}\right) = 0$$

Remark 1. If f_y vanishes identically, then the equation (1.2') yields $\lambda = 0$. This means that one of the principal directions is $d\underline{u} = (1,0)$, and the condition (1.1) can be re-written in the form $f_x = 0$. Similarly, in the case $f_x = 0$ identically, the equation (1.1) becomes $f_y = 0$. In both cases the surface $z = f(x,y)$ is a cylinder, and in both cases the equation (1.4) vanishes identically. Hence the equation (1.4) can be efficient only under the assumption that the surface $z = f(x,y)$ has a more complicated form.

Example 1 Let $f(x,y) = \left(x^2 - y\right)^2$. Then $f_x = -4xy + 4x^3$, $f_{xx} = -4y + 12x^2$, $f_y = 2(y - x^2)$, $f_{yy} = 2$, $f_{xy} = -4x$. By substituting into (1.4), we get $2(y - x^2)^3 = 0$. It is clear geometrically that the line $L = \left\{y - x^2 = 0\right\}$ is the only ridge in this case.

Remark 2. It should be stressed that the equation (1.4) has been derived from the quadratic equation (1.2'), which includes both maximal and minimal curvature directions. Therefore equation (1.4) is also satisfied on curves, where $D_{\underline{v}} f(x,y)$ vanishes in the direction $\underline{v}$ of minimal curvature:

Example 2 Let $f(x,y) = \lambda_1 x^2 + \lambda_2 y^2, \quad \lambda_1 > \lambda_2$. The equation (1.4) provides $4\lambda_1\lambda_2 xy(2\lambda_1 - 2\lambda_2) = 0$. Hence the solutions are the lines $x = 0$ and $y = 0$, but only the line $x = 0$ satisfies ridge definition.

The fact the equation (1.4) can generate two families of lines, corresponding to maximal and minimal curvatures, while we are interested only in one of them, seems to be its deficiency. We easily get free of these difficulties owing to our central concept of multigrid and multiorder: We use the equation (1.4) only on the second step of high order jet construction, providing the first order jet detection is preliminary performed.

In what follows, we suppose that f_x and f_y do not vanish identically, hence the equation (1.4) can be used to determine the principal curvature directions.

We shall show now that lines defined by (1.4) have a few interesting geometric properties.

1. It is well known that two directions, tangent to the surface $z = f(x,y)$ and corresponding to the principal curvature directions $\underline{u}$, $\underline{v}$ are orthogonal (as directions in R^3). It turns out that along ridges the principal directions $d\underline{u} = (du_1, du_2)$ and $dv = (dv_1, dv_2)$ are also orthogonal as directions in R^2.

Indeed, for $\lambda_1 = \dfrac{du_2}{du_1}$ and $\lambda_2 = \dfrac{dv_2}{dv_1}$ we obtain from (1.2') and (1.4):

$$\lambda_1\lambda_2 = \frac{(1+f_x^{\,2})f_{xy} - f_x f_y f_{xx}}{f_x f_y f_{yy} - (1+f_y^{\,2})f_{xy}} = -1 .$$

2. We shall show now that ridge can be defined in three equivalent forms.

Let $H = \begin{vmatrix} f_{xx} & f_{xy} \\ f_{xy} & f_{yy} \end{vmatrix}$ be a Hessian of the brightness function f and let $\underline{\alpha} = (\alpha_1, \alpha_2)$ and $\underline{\beta} = (\beta_1, \beta_2)$ be its eigenvectors, corresponding to the eigenvalues γ_1, γ_2 , $|\gamma_1| > |\gamma_2|$.

Definition 1 - *as given above;*

Definition 2 . *Curve* L *is called a ridge, if the directional derivative* $D_\alpha f(x,y)$ *vanishes at each* $(x,y) \in L$ *:* $D_\alpha f(x,y) = 0$.

Definition 3. *Curve L is called a ridge, if along L the gradient vector field is orthogonal to the eigenvector field, corresponding to* γ_1.

The equivalence of definitions (1) – (3) follows from the following two propositions.

Proposition 2. *Curve L satisfies the equation (1.4) if and only if* $D_{\underline{\alpha}} f(x,y)$ *for each* $(x,y) \in L$, *where* α *is one of `eigenvectors of Hessian* H.

Proof. Write the characteristic equation of f : $\begin{vmatrix} f_{xx}-\gamma & f_{xy} \\ f_{xy} & f_{yy}-\gamma \end{vmatrix} = 0$, or

$$\gamma^2 - (f_{xx} + f_{yy})\gamma + (f_{xx}f_{yy} - f_{xy}{}^2) = 0 . \tag{1.5}$$

It can easily be seen that (1.4) can be written as

$$\left(\frac{f_{xx}f_y - f_{xy}f_x}{f_y}\right)^2 - (f_{xx} + f_{yy})\frac{f_{xx}f_y - f_{xy}f_x}{f_y} + (f_{xx}f_{yy} - f_{xy}{}^2) = 0 \tag{1.6}$$

It follows immediately from (1.5) and (1.6) that $\gamma_1 = \dfrac{f_{xx}f_y - f_{xy}f_x}{f_y}$ is one of eigenvalues of H. Hence the vector $\underline{w} = (-f_{xy}, f_{xx} - \gamma_1)$ is an eigenvector.

Now we can find the derivative $D_{\underline{w}} f$ at the point $(x,y) \in L$:

$D_{\underline{w}} f = f_x(-f_{xy}) + f_y\left(f_{xx} - \dfrac{f_{xx}f_y - f_{xy}f_x}{f_y}\right) = 0$.

Conversely, let L be a line such that $D_{\underline{\alpha}} f(x,y) = 0,\ (x,y) \in L$, for the eigenvalue γ and the corresponding eigenvector $\underline{\alpha} = (-f_{xy}, f_{xx} - \gamma)$.

Then $D_{\underline{\alpha}} f = 0$ yields $\gamma = \dfrac{f_{xx}f_y - f_{xy}f_x}{f_y}$, and substituting γ in the characteristic equation (1.5) we get the equation (1.6), which is equivalent to (1.4)

Proposition 3. *Let* α *be the eigenvector of* H. *Then* $D_{\underline{\alpha}} f(x,y) = 0$ *for each* $(x,y) \in L$ *if and only if the gradient vector field* $f_x \partial_x + f_y \partial_y$ *is also the eigenvector field along* L.

(The proof is evident).

The equivalence of Def. (1) and Def.2 is used efficiently in our algorithm. One can see in this equivalence the mathematical basis for our central concept of multigrid and multiorder.

3.3.2. **Local ridge element detection algorithm**

The algorithm consists of two main shapes. On the first one we perform the second order analysis on 3x3 pixels cell (or 5x5 cells with appropriate choice of weights) to detect possible ridge elements.

This stage of algorithm results in the list of linear ridge elements. Each element consists of a point (x_0, y_0), which is a center of ridge element and vector (u_1, u_2) which is presumably tangent to the ridge.

On the second stage, the high order approximating polynomials, constructed on 5x5 pixels cells with more uniformly distributed weights, are used to detect curvilinear (high outer) ridge elements.

In preferred embodiment we describe construction of the second order curvilinear ridge elements, based on third order approximation $P_3(x,y) = \sum_{i=0}^{3}\sum_{j=0}^{i} \mathrm{a}_{ij} x^j y^{i-j}$ of the brightness function. In the case of fourth order approximating polynomial the procedure is essentially the same.

This stage of algorithm results in the list of second order curvilinear ridge elements. Each element consists of a central point (x_0, y_0) which is the origin of local coordinate system, the direction (u_1, u_2) of the x-axis in this local system, which is the eigendirection of the second order form $\sum_{j=0}^{2} a_{2j} x^j y^{2-j}$, corresponding to biggest eigenvalue, and the coefficients c_0, c_1, c_2 of the second order jet $y = c_0 + c_1 x + c_2 x^2$ for the curvilinear ridge element in this local coordinate system.

3.3.3. Jet formulae for ridges

In the step of computation of the derivatives, at each pixel of the image (and if necessary, at intermediate points) the approximate values of the derivatives of the brightness function have been produced. However, these derivatives values once more form a discrete set of data, and not the continuous functions. Consequently, the formulae and equations, given in the Section 3.3.1 above, are not directly applicable. Indeed, a direct

substitution of the values of the brightness function and of its derivatives into any of these expressions would produce not more than an indication of whether the pixel considered belongs to the ridge, i.e. a zero order information. Instead we would like to get a second order approximation of the ridge central line. Thus the formulae and equations of the Section 3.3.1 have to be translated into a form, where their input will be the set of derivatives values at a given point, while their output will represent (up to the prescribed order) the ridge element, identified in the vicinity of this given point.

The formulae and expressions of the required type are known in Mathematical Analysis, more specifically, they are widely used in the so called Jet Algebra computations, appearing in Singularity theory. In each specific problem these expressions have to be developed independently, but being developed, they can be applied to the whole class of similar situations. Being usually rather complicated algebraically, these expressions have to be specially arranged for efficient and robust numerical applications. It is here that multi-order considerations become especially important.

Third order analysis. Identification of the second order ridge elements.

We look now for a second order local approximation to a solution of (1.4). Let $r(x_0, y_0, u_1, u_2)$ be the first order local ridge element , detected by the second order analysis. This means that the second order approximating polynomial, centered at (x_0, y_0) and diagonalized , has the form

$$P_2(x, y) = \widetilde{a}_{00} + \widetilde{a}_{10}x + \lambda_1 x^2 + \lambda_2 y^2,$$

where $\left|\widetilde{\lambda}_1\right| << \left|\widetilde{\lambda}_2\right|$ and (u_1, u_2) is the eigenvector of λ_1 .

Let $P_3(x, y)$ be the third order approximating polynomial on the 5×5 pixels cell and also centered at (x_0, y_0) , with diagonalized second order part:

$$P_3(x, y) = b_{00} + b_{10}x + b_{01}y + b_{20}x^2 + b_{02}y^2 + b_{30}x^3 + b_{21}x^2 y + b_{12}xy^2 + b_{03}y^3 .$$

The coefficients of P_3 are different from those of P_2 because of two reasons: pixels weights in the approximation have been changed and the first order approximation is now influenced by the third order terms presence. Nethertheless, these coefficients reflect the cylindrical form of the brightness function, hence, we can assume that $\left|b_{20}\right| << \left|b_{02}\right|$ and b_{01} is small enough for the following inequality to hold:

$$b_{02}{}^2 - 3b_{01}b_{03} \geq 0 .$$

Then the origin can be shifted to the point $(0, \delta_1)$, where δ_1 the zero of $b_{01} - 2b_{02}\delta + 3b_{03}\delta^2 = 0$, satisfying $\left|\delta_1\right| < \left|\delta_2\right|$. Then P_3 can be written as

(2 .1) $$P_3(x,y) = a_{00} + a_{10}x + a_{02}x^2 + a_{02}y^2 + \sum_{i=0}^{3} a_{i,3-i}x^i y^{3-i} \ .$$

We look for a ridge as a solution of the equation (1.4) with $f(x,y) = P_3(x,y)$. Substituting for f_x, f_y, f_{xx}, f_{xy}, f_{yy} the corresponding expression, we get the following equation:

(2.2)

$$(2a_{12}x + 2a_{12}y)\left[(a_{10} + 2a_{20}x + 3a_{30}x^2 + 2a_{12}xy + a_{12}y^2)^2 - (2a_{02}y + a_{21}x^2 + 2a_{12}xy + 3a_{03}y^2)^2\right] \times$$
$$+ (a_{10} + 2a_{20}x + 3a_{30}x^2 + 2a_{21}xy + a_{12}y^2)(2a_{02}y + a_{21}x^2 + 2a_{12}xy + 3a_{03}y^2) \times$$
$$(2a_{02} - 2a_{20} + 2a_{12}x - 2a_{21}y + 6a_{03}y - 6a_{30}x) = 0$$

Let $y = \sum_{i=0}^{\infty} c_i x^i$ be Taylor expansion for a solution. By substituting it in (2 .2) and by equating to zero the coefficient of x^m, we get the following conditions:

(a) for $m = 1$:

(2.3) $$2a_{12}c_0((a_{10} + a_{12}c_0^2)^2 - (2a_{02}c_0 + 3a_{03}c_0^2)^2) +$$
$$(a_{10} + a_{12}c_0^2)(2a_{02}c_0 + 3a_{03}c_0^2)(2a_{02} - 2a_{20} - 2a_{21}c_0 + 6a_{03}c_0) = 0$$

This equation is satisfied for $c_0 = 0$, so we can put $c_0 = 0$.

(b) for $m = 1$:

(2.4) $$2(a_{21} + a_{12}c_1){a_{10}}^2 + 4a_{10}a_{02}(a_{02} - a_{20})c_1 = 0$$

If $a_{10} \neq 0$, then we have

(2.5) $$c_1 = -\frac{a_{10}a_{21}}{a_{10}a_{12} - 2a_{02}a_{20} + 2{a_{02}}^2} \ .$$

The denominator in (2.5) doesn't vanish: since $|a_{02}| >> |a_{20}|$ and $|a_{02}| >> |a_{10}|$, the condition $a_{10}a_{12} - 2a_{02}a_{20} + 2{a_{02}}^2 = 0$ means that $|a_{12}| >> |a_{02}|$. The last condition is impossible, because $|a_{02}| >> |a_{20}|$ holds for the second order approximating polynomial, centered at the same point.

(c) for $m = 2$:

(2.6) $(2a_{21}+2a_{12}c_1)2a_{10}a_{20}+2a_{12}c_2a_{10}{}^2+a_{10}(2a_{02}c_2+a_{21}+2a_{12}c_1+3a_{03}c_1{}^2)(a_{02}-a_{20})+$
$2a_{10}a_{20}a_{02}c_1+4a_{20}a_{02}(a_{02}-a_{20})c_1=0$

If $a_{10}\neq 0$, then (2.6) provides

(2.7)

$$c_2=[(2a_{21}+2a_{12}c_1)2a_{10}a_{20}+2a_{10}a_{20}a_{02}c_1+a_{10}(a_{21}+2a_{12}c_1+3a_{03}c_1{}^2)(a_{02}-a_{20})+$$
$$4a_{20}a_{02}(a_{02}-a_{20})c_1]/(2a_{12}a_{10}{}^2+2a_{10}a_{02}),$$

where the denominator $2a_{12}a_{10}{}^2+2a_{10}a_{02}$ doesn't vanish for the same reason: $|a_{12}a_{20}|<<|a_{02}|$.

Hence, in the case $a_{10}\neq 0$ the second order jet $c_0+c_1x+c_2x^2$ for a solution of (2.2) is determined by (2.5) , (2.7).

(d) We can assume now that $a_{10}=0$. Then by equating to zero the coefficient of x^3 in (2.2) , we have:

(2.8)

$$(2a_{21}+2a_{12}c_1)(4a_{20}{}^2-4a_{02}{}^2c_1{}^2)+2a_{02}c_1(3a_{30}+2a_{21}c_1+a_{12}c_1{}^2)(2a_{02}-2a_{20})+$$

$$4a_{20}a_{02}c_1(2a_{12}-2a_{21}c_1+6a_{03}c_1-6a_{30})+$$
$$2a_{20}(2a_{02}c_2+a_{21}+2a_{12}c_1+3a_{03}c_1{}^2)(2a_{02}-2a_{20})=0$$

If $a_{20}\neq 0$, then $c_1=0$ by (2.6) and the equation (2.8) becomes

$$8a_{21}a_{20}{}^2+2a_{20}(2a_{02}c_2+a_{21})(2a_{02}-2a_{20})=0.$$

Therefore, in the case $a_{10}=0$, $a_{20}\neq 0$ we have

$$c_2=-\frac{a_{21}(a_{20}+a_{02})}{2a_{02}(a_{02}-a_{20})},$$

and the second order is determined also in this case.

We assume now that $a_{20}=0$. Then (2.8) becomes

$$(2a_{21}+2a_{12}c_1)(-4a_{02}^2c_1{}^2)+(3a_{30}+2a_{21}c_1+a_{12}c_1^2)4a_{02}c_1a_{02}=0,$$

which is equivalent to

(2.9) $$c_1(3a_{30} - a_{12}c_1^2) = 0 .$$

If $a_{30}^2 + a_{12}^2 \neq 0$, then $c_1 = 0$ is one of solutions, so we can put $c_1 = 0$.
If $a_{30}^2 + a_{12}^2 = 0$, then (2.9) vanishes identically.

(e) By equating to zero the coefficient of x^4, we have

(2.10) $$\begin{aligned} &-8a_{12}a_{02}^2c_1c_2 - (2a_{21} + 2a_{12}c_1)(4a_{02}a_{21}c_1 + 8a_{02}a_{12}c_1^2 + 8a_{02}^2c_1c_2 + 12a_{01}a_{03}c_1^3) + \\ &2a_{02}c_1(3a_{30} + 2a_{21}c_1 + a_{12}c_1^2)(2a_{12} - 2a_{21}c_1 + 6a_{03}c_1 - 6a_{30}) + 2a_{02}(3a_{30} + 2a_{21}c_1 + a_{12}c_1^2)\times \cdot \\ &(2a_{02}c_2 + a_{21} + 2a_{12}c_1 + 3a_{03}c_1^2) + 4a_{02}^2c_1(2a_{21}c_1 + 2a_{12}^2c_1c_2) = 0 \end{aligned}$$

If $a_{30}^2 + a_{12}^2 \neq 0$, then $c_1 = 0$ by (d) and it follows from (2.10) that

(2.11) $$c_2 = -\frac{a_{21}}{2a_{02}} .$$

If $a_{30}^2 + a_{12}^2 = 0$, then (2.10) yields

(2 .12) $$3a_{21}^2c_1 + 2a_{21}^2c_1^3 + 3a_{21}a_{03}c_1^3 = 0 .$$

If $a_{21} \neq 0$, then $c_1 = 0$ is one of solutions of (2 .12), so we can put $c_1 = 0$. If $a_{21} = 0$, then (2 .12) vanishes identically.

(f) We have now $a_{10} = a_{20} = a_{30} = a_{12} = 0$. By equating to zero the coefficient of x^5 in (2.2) , we have

(2.13)

$$\begin{aligned} &2a_{12}(4a_{21}^2c_1^2 - 8a_{02}^2c_1c_3 - 24a_{02}c_1^2c_2 - 4a_{02}^2c_2^2 - a_{21}^2 - 9a_{03}^2c_1^2 - 4a_{02}a_{21}c_2 - \\ &12a_{02}a_{03}c_2c_1^2 - 6a_{21}a_{03}c_1^2) + 4a_{21}a_{02}c_1(2a_{02}c_3 + 6a_{03}c_1c_2) + \\ &2a_{21}c_1(2a_{02}c_2 + a_{21} + 3a_{03}c_1^2)(-2a_{21}c_1 + 6a_{03}c_1) + \\ &4a_{21}a_{02}c_1^2(-2a_{21}c_2 + 6a_{03}c_2) + 4a_{02}a_{21}c_2(2a_{02}c_2 + a_{21} + 6a_{03}c_1^2) + \\ &4a_{02}a_{21}c_1c_2(-2a_{21}c_1 + 6a_{03}c_1) + 8a_{02}^2a_{21}c_1c_3 = 0 \end{aligned}$$

If $a_{21} \neq 0$, then $c_1 = 0$ by (e) and (2.13) yields $-4a_{21}^2a_{02}c_2 - 2a_{21}^3 = 0$. Hence

$$c_2 = -\frac{a_{21}}{2a_{02}},$$

and the second order jet is determined also in the case

$$a_{10} = a_{20} = a_{30} = a_{12} = 0 \ , \ a_{21} \neq 0 \ .$$

If $a_{21} = 0$, then the third order approximating polynomial

$$P_3(x, y) = a_{00} + a_{02} y_2 + a_{03} y^3$$

is a cylinder, and obviously the second order jet for a ridge element is just $y = 0$.

4. High Order detection of Edges

4.1. Visual and mathematical description of edges. Comparison with mathematical description of ridges.

Ridges and edges present one of the most important examples of uniform characteristic lines. As it was described in detail above, visually ridges are specified as narrow strips, separated from the background by their (locally uniform) brightness or color. Edge is specified as boundaries between two regions with relatively different gray level properties.
More formally, the definition of characteristic features can be made in terms of brightness function. Brightness function is understood as a function representing image intensity (in particular it may be the intensity of one of the standard color separations, R, G and B). We will call it also gray level of the image and denote by $f(x, y)$ its value at the point with the coordinates (x, y).

Consider the brightness z = f(x,y) at each point (x,y) of the image as a continuous and smooth function, represented by the surface of its graph in the three-dimensional space. In this interpretation ridges appear as narrow elongated channels (for dark ridges) or clips (for bright ones) on the brightness surface. It seems to be intuitively clear that at the points of the ridge's central line, this surface is maximally curved in the direction, orthogonal to the ridge. In other words, the curvature of the brightness surface in the direction, orthogonal to the ridge's central line, is maximal among all the directions. In the first part of the current Application two equivalent definitions of the ridge are discussed, and their equivalence is proved. In each one of them, the ridge's central line is characterized by extremal property of curvature along it, which can be classified as a "local" property of the brightness function. According to this "locality", ridge's central line satisfies functional equations written in terms of brightness function derivatives.
Edges, on the contrary, can't be characterized by "local" properties of brightness function. They appear as steep slopes of the surface $z = f(x, y)$, and in the (x, y)-plane they appear as strips possessing the following property: the gradient ∇f is big in the central zone of the strip and drops to zero near the boundary lines of the strip, while the brightness function is monotone in each transversal direction.

In particular, the central line of the edge can be defined only with respect to the edge's boundary lines (for example, as a "middle" line of the two boundaries). The points of this central line may be mathematically indistinguishable from the neighboring points: all the derivatives of the brightness function may be virtually the same inside a certain "edge strip", much wider, than the desired central line.

Consequently, there exist no equation for the edge's central line in terms of the values of the brightness function and its derivatives. Therefore, edges have inherent "semi-local" nature and can't be described in the same way as ridges.

On the other hand, it is intuitively clear that edges are well-defined lines on the image, and their central lines have an important visual meaning and can be identified with a high accuracy.

The method, given below, is based on a "semi-local" definition of edges. It starts not with an "ideal" brightness function, but with its polynomial approximations in certain prescribed scales. An important feature of the method is that it uses not only the domains where the polynomial approximation of the brightness function is accurate, but also the domains, where this polynomial approximation starts to deviate from the original. This stresses the semi-local character of the method: the approximating polynomials are analyzed and processed not only near the window center (where the approximation is the best) but on all the window.

Specifically, on the edges, the third order polynomial approximation of the brightness function (on the scale slightly larger than the width of the edge) is usually accurate till the edge boundary lines. Then it starts to deviate from the brightness function. In fact, such polynomial approximation stresses the boundary lines, making them ridges. This happens even in cases where the image itself does not have ridges at the edge boundary. The details are given below.

Exactly as in the case of ridges, all the formulae and equations, which are used in edge description, have to be translated into a form, where their input will be the set of derivatives values at discrete points (at a given point). Their output will represent (up to the prescribed order) the ridge elements, identified in the vicinity of this given point.

In the case of ridges, the required Jet Algebra formulae are provided in Section 3.3.3 above. The discribed edges detection algorithm is based on these formulae and on some additional Jet expressions, provided in the text of the method description below.

Local Edge Descriptors

At the local level of edge detection the following local edge structures can be consecutively identified by means of multi-scale and multi-order analyses (in the vicinity of given point) :

1. Bilateral "gradient element" (or zero-order Local Edge Descriptor).

It consists of two endpoints E_1, E_2 and brightness function values $f(E_1), f(E_2)$ at these points. Alternatively it can be described by the center *C* of the segment $[E_1, E_2]$, the width *h* and the unit direction vector (u_1, u_2) orthogonal to the segment $[E_1, E_2]$ The geometric interpretation of bilateral "gradient element" is the following: it represents a cylindrical surface ,which is parallel to the spatial vector $(u_1, u_2, 0)$ and has a cubic directrix (cross-section). The cross-section curve has extrema at points E_1, E_2 (See Fig.).

2. Unilateral "gradient element".

If the edge under detection is wide enough with respect to the size of the coarse scale cell, then the initial steps of the detection procedure can't result in the identification of bilateral "gradient element": only one extreme point of the cross-section can be caught within one cell. The received information is memorized in the form of unilateral "gradient element". It consists of one endpoint E, width *h*, the direction vector (u_1, u_2) and the brightness function value $f(E)$. The pair of adjacent unilateral "gradient element" can be unified into bilateral "gradient element" by means of special wide edge construction procedure. If unilateral "gradient element" doesn't have the adjacent counterpart, it is excluded from further consideration.

3. First order (linear) Local Edge Descriptor (linear LED).

It consists of two endpoints E_1, E_2, two unit direction vectors (u_{i1}, u_{i2}) , $i = 1,2$ (which don't have to be parallel), and brightness function values $f(E_1), f(E_2)$. The corresponding surface is not in general cylindrical, but it has a cubic-like cross-section along E_1E_2 with extrema at points E_1, E_2, and in the vicinity of the endpoint E_i, $i = 1,2$, it has a form of a linear ridge element, centered at E_i and directed along (u_{i1}, u_{i2}) .

4. Second order Local Edge Descriptor.

It consists of two endpoints E_i, $i = 1,2$, two unit direction vectors (u_{i1}, u_{i2}) , two second order jets $c_{i1}\xi + c_{2i}\xi^2$, and brightness function values $f(E_i)$. The corresponding surface is characterized by the following features:

- the cross-section along E_1E_2 has a cubic-like form with extrema at points E_1, E_2;
- in the vicinity of the endpoint E_i, $i = 1,2,$ it has a form of second order ridge element, centered at E_i and represented by the jet $c_{i1}\xi + c_{2i}\xi^2$ in a local coordinate system with ξ-axis in the direction of (u_{i1}, u_{i2}) ;
- brightness function is monotone along each transversal section.

Example

The brightness function $f(x, y)$ is given by

$$f(x, y) = \left(y - x^2\right)^3 - 3\left(y - x^2\right)$$

In this example the edge forms a strip, bounded by two ridges:

$y = x^2 + 1$ and $y = x^2 - 1$.

The brightness function, restricted to "vertical" lines $x = const$, attains its extrema exactly on these ridges.

The following typical visual patterns produce easily predictable detection results:

1. A regular edge. This is a single edge, separating two regions with different brightness function properties, with the width that doesn't exceed 3-4 pixels. This pattern produces a high value of the gradient on any 3x3 pixels window, centered properly, and on a coarse scale passes the third order test for LED: the third order polynomial, restricted to orthogonal section, has both minimum or maximum points inside the chosen cell.
   In this case the linear (or the first order) LED is generally identified with high accuracy of both geometric and brightness parameters. The curvilinear (or the second order) LED can be identified using fourth order procedure.

2. An imaginary edge. Assume that the ridge is well delineated and has a width of at least 2-3 pixels, and approximating polynomials of second and third order are constructed on 3x3 and 5x5 cells respectively, centered at the middle of the ridge slope. In this case LED will be identified, and the whole ridge's slope will be represented by a chain of LED. The edges of this kind are classified as

"imaginary " ones. They should be incorporated into other characteristic lines and eliminated from the final list of edges. On the other hand, imaginary edges are usually identified with high geometric and brightness accuracy, which makes them the dominant tool in the detection of geometrically adherent characteristic lines.

3. A "wide" edge. If the edge has a width of 4 pixels or more, the corresponding "gradient element" can't be caught by a single third order polynomial on 5x5 cell. Depending on edge's width and the location of the cell, there exist the following possibilities : (a) the orthogonal section doesn't have extremum points inside the cell; (b) the orthogonal section has only one extremum point inside the cell . In the first case the LED can't be identified and the corresponding pattern is represented by slow-scale (background) elements. In the second case the unilateral "gradient element" is identified. Two adjacent unilateral "gradient elements" will be unified in the regular bilateral LED, if the required adjacency conditions are satisfied. The unilateral "gradient elements", which don't have adjacent counterpart, are excluded from the further consideration.

5. High order detection of Color Cross-Sections

Generally the method proposed is a combination of a multi-scale and a multi-order approaches, as described above. In the specific implementation of the algorithm the fourth order approximation of the brightness function on a larger scale produces the fourth order approximation of the ridge's cross-section. The second order approximation of the brightness function on a finer scale produces the second order (parabolic) approximation of the ridge's cross-section. The zero order approximation of the cross-section is produces via the direct pixel values plotting. Finally, the output cross-section is obtained as a fourth order curve, fitting in the best way all these three sets of data.

Roughly the same procedure is used for edges. It includes also the detection of local geometric edge elements, described above, and the identification of their local Cross-Sections.

6. An Overview of the VIM Coding

6.1. Summary

The “raw” VIM Texture, as described in Section 2 above, can be realized in a form of a computer memory structure, or can be stored as a file (textual or binary). While transparent in its structure and convenient for processing, this file may be rather big in its size. Although any standard loss-less statistic compression, like Zip, usually reduces this size to a fraction of the original image, to get a compact VIM compression more sophisticated tools are required. VIM structure provides such tools, allowing one to utilize visual correlations between different spatially associated parameters, to eliminate significant redundancy in raw data and to take into account specifics of human visual perception.

This document provides an overview of general principles of VIM Coding and a general description of the encoding of each of the parameters in the VIM Texture.

6.2. General Principles of VIM Coding

Organization of storage and encoding of VIM Data reflects general principles of VIM data structure:

6.2.1. Simple Structure versus powerful Authoring Tools

The structure of VIM itself, in both its levels (raw and compressed) is kept simple and transparent. On the other side, the authoring tools are assumed to be sophisticated.

One of manifestations of this VIM feature is, that the most natural way to explain and to illustrate VIM representation is on cartoon-like images. All the structural aspects of can be authentically represented by fairly simple such examples.

But VIM Authoring Tools allow one to produce as well VIM representations of complicated high-resolution images of real world. The VIM structure remains the same also here. However, VIM elements become much more dense, their visual role and visual interaction between different elements become much more complicated.

Another manifestation of the above principle concerns VIM Coding. The structure of the aggregated and quantized data streams (which are statistically encoded in the final stage) remains simple and transparent. However, powerful Authoring Tools allow for an adaptation of the VIM parameters to the specifics of the Aggregation and Coding and in this way to a significant reduction of the data size.

6.2.2. Elimination of redundancies via proper Data Aggregation

Most of Data Streams in RVIM present strong non-uniformity in their statistical distribution, as well as strong inter-correlations. In the overall Coding organization one can either leave removing these redundancies to the final statistical loss-less encoding (Huffman Coding) or to eliminate them in earlier stage, by a proper Data Aggregation.

VIM structure provides a full control on all the geometric and the color features of the image, and thus a possibility for a clever Data Aggregation on all the levels. Experiments show, that this Aggregation provides a strongly better data compression, than a straightforward application of the Huffman Coding, as well as a much better utilization of the specifics of the Human visual perception.

Organization of virtually any of the Data Streams, described below, gives examples of Data Aggregation.

6.2.3. Specifics of a human visual perception in VIM Coding

6.2.3.1. Y I Q instead of RGB

In the main mode, VIM coding uses Y I Q color components, rather than standard RG B ones (a common feature with JPEG format and others). The quantization of the I and Q components is usually stronger than of the Y component.

6.2.3.2. Low color sensitivity for a small angular size

It is well known, that a human visual sensitivity to brightness of a visual pattern (and especially to its color) strictly decreases with the angular size of the pattern. The angular size of VIM Lines and especially Patches is usually rather small. Consequently, the Y and especially the I and the Q components of the Lines Color Profiles are quantized much stronger than the corresponding components of the Area Color. In one Coding mode the I and the Q components of the Color Profiles and of the Patches are not stored at all. In an advanced Coding mode the quantization thresholds for Y, I and Q components of the Color Profiles and of the Patches depend on their width (size).

6.2.3.3. Visual redundancy of Color Profiles

Our experiments have shown that a visual sensitivity to some elements of the Color Profile is rather low. In particular, this concerns the interior brightness parameters RB_2 and LB_2. (“Bump” parameters. See the document “VIM Texture: Technical Specifications”).However, the presence of the typical Profile shape, described by these parameters (margin “bumps”) is important for an overall image quality. Consequently, we compute a prediction for these parameters on the base of the others (and of the global image properties) and encode only the corrections to these predicted values. In one Coding mode the parameters RB_2 and LB_2 are not stored at all.

6.2.3.4. Different visual sensitivity to “near” and “non-near” geometry

It is well known that our visual sensitivity to the geometric shapes is much higher for “geometrically near” visual patterns than for isolated ones. Rather strong geometric distortions in a position of a line, passing far away from other patterns, will not be perceived at all, while even a small distortion of one of a couple of closely neighboring lines immediately “pops to the eyes”. This fact is taken into account in the VIM structure already in the explicit definition of the Crossings and Splittings of the Lines. The geometric parameters of the Terminal Points, representing Crossings and Splittings, are stored with a higher accuracy than that of the usual Line Points. In Advanced Coding mode the “Aggregated Crossing” and the “Aggregated Color Profile” are used, which captur the most common cases of VIM elements visual aggregation. Also in Lines quantization their mutual position can be taken into account.

6.3. Encoding of RVIM Parameters

The VIM Texture comprises various parameters with different mathematical meaning and visual significance. The following main groups of parameters are encoded separately:

6.3.1. “Centers”.

This includes encoding coordinates of the Lines Terminal Points and coordinates of the centers of Patches (and, in an “explicit” Coding mode, coordinated of the Area Color Points), together with the data specifying the type of the encoded point. The main aggregation tool here is the encoding of points with respect to a certain regular cell partition of the image plane. This eliminates redundancy related with an explicit memorizing of the order of the points and allows one to take into account expected points density.

6.3.2. “Terminal Points”.

At Terminal Points the “topological” structure of the system of the Lines is stored. This is achieved by storing the branching structure of these points and by associating the adjacent Lines to the corresponding Terminal Points. Also the accurate coordinates of the starting Line Points of the adjacent Lines are stored at Terminal Points. In an Advanced Coding mode, at a Terminal Point an accurate geometry of the corresponding Crossing of the Lines is stored, together with a color data, allowing for a compact representation of the Color Profiles of the adjacent Lines.

6.3.3. “Lines”.

Basically, the encoding of Line Geometry is quite straightforward. After quantizing the coordinates of the Line Points, the vector of the first Line Segment is stored, together with the offsets of the subsequent Line Segment Vectors from the preceding ones. However, aggregation with the Terminal Points is used, since the starting and the ending Line Points are already stored at the corresponding Terminal Points.

6.3.4. “Area color”.

In the regular Coding mode, the coordinates of the Area Color Points (AC’s) are not explicitly stored. Instead, their brightness (color) values are aggregated with respect to a certain regular cell partition of the image plane. This eliminates redundancy related with an explicit memorizing of the position of the Area Color Points (this precise position is usually visually insignificant) and allows one to take into account expected points density. A portion of the Area Color parameters is associated with Lines Color Profiles (margin color or brightness). These color values at the Line margins are stored together with other Color Profile parameters. Further aggregation of the Area Color data is achieved in the “Two - Scale Area Color Coding”, where, in particular, a redundancy is eliminated between the Area Color values at the AC’s and at the margins of Lines Profiles.

6.3.5. “Color Profiles”.

The parameters of the Color Profiles allow for a natural aggregation, taking into account their visual role and typical behavior. Thus, Profile “bumps” parameters, which normally reflect the image origin and behave in a coherent way at all the parts of the image, are represented as corrections to certain predicted values. The Central Color of non-separating Ridges (and of Patches) is naturally stored relative to the Area Color at the corresponding points.

In the next step Color Profiles are naturally aggregated along the Lines. Thus only the Profile at the starting Line Point and the subsequent differences are stored. To further eliminate data redundancy along the Lines, sub-sampling is applied to the Line Points, at which the Color Profiles are stored. Finally, Color Profiles of different Lines at their common Terminal Points (Interior Points, Crossing and Splittings) are naturally aggregated between them.

6.3.6. “Patches”.

The coordinates of the centers of Patches are encoded as the “Centers”, as described above. The rest of the geometric and the color parameters of the Patches are stored in a straightforward way. Some of attributes of the human visual perception are taken into account: as the size of the Patch decreases, its accurate shape (and color!) become visually insignificant, and the corresponding data is quantized with a coarser step, or is not stored at all.

6.3.7. “Depth”.

Depth data is stored in three main modes. In the “direct” mode the depth values are stored as an “additional color component”, thus appearing as the part of the “Area Color”, the “Color Profiles” and the “Patches”, exactly as the other color components. The only difference is that the “Depth Profile” of Lines is very simple, comprising only one value at the center. In the second mode, only analytic depth models are stored, one for each Sub-Texture. In the decoding process the depth at each relevant point of a Sub-Texture is computed through the stored model. In the third “mixed” mode the depth values are stored as corrections to the “predictions” of the models.

6.3.8. “Multi-layer” Area Color Coding.

In VIM Texture some Sub-Textures may occur “on-top” or “under” other Sub-Textures. In raw form, where the Area Color Points are stored together with their coordinates, depth & color values and the index of the Sub-Texture they belong to, no interpretation problems appear. However, in a procedure of the Area Color Coding, described above, where the brightness (color) and the depth values of the Area Color Points are aggregated with respect to a certain regular cell partition of the image plane, the cell partition is to be duplicated for each layer of Sub-Textures. Also a separate bounding rectangle is memorized for each Sub-Texture, to avoid storing of irrelevant cells.

6.4. Multi-Scale Coding

The Coding scheme, as described above, is augmented by application of a Multi-Scale approach. Essentially in each of the groups of the parameters it is possible first to encode data on the coarse scale, and then to represent the fine-scale data as corrections to the coarse-scale predictions. Multi-Scale approach is used in the encoding of the Lines geometry, Lines Color Profiles and of the Area Color and Depth. The basic VIM structure distinguishes right away fine scale details – patches and short ridges. These elements are naturally excluded from the coarse-scale data.

6.4.1. Multi-Scale Coding of Line Geometry.

On the coarse scale Lines are approximated with a smaller number of Line Segments and with a coarser quantization of the coordinates of the Line Points and Vectors and of the Line Segments Heights. On the fine scale the coordinates of the new Line Points are given in the coordinate system, associated with the coarse-scale curves, and hence appear as “corrections” to the coarse-scale data. The size of these corrections should not exceed the allowed error of the coarse-scale approximation.

6.4.2. Multi-Scale Coding of Area Color.

On the coarse scale a larger cell-size of the regular partition is chosen (usually, twice or four times the original cell-size). The Area Color data are aggregated with respect to the coarse partition, and the corresponding Area Color representation is formed. Later the fine-scale Area Color is represented as corrections to the coarse scale. Here also the encoding procedure can be built in such a way that the maximal possible size of the corrections is known a priori. Exactly the same procedure can be applied to the Depth values.

6.4.3. Multi-Scale Coding of Color Profiles.

In VIM Texture Color Profiles are stored at the Line Points (bounding the Line Segments). On the coarse scale Color Profiles are stored only at a sparser sub-sampling of the Line Points. The stored values are interpolated to the rest of the Line Points, thus providing a coarse-scale prediction of the Profiles. At the fine-scale Profiles are stored as corrections to these predictions.

6.5. VIM data streaming and error resiliency

The natural multi-scale structure of the compressed VIM data, as described above, is important in two central problems of data transmission: data streaming and data error resiliency. When streaming VIM data, the coarse VIM image is transmitted first, providing a reasonable quality approximation to the original image (the lines geometry is less accurate, the color values are somewhat “low-pass filtered” and certain fine scale details disappear). Then the fine-scale corrections and elements (Patches and short Ridges) are streamed, gradually enhancing the image visual quality.

As far as the error resilience is concerned, only the coarse-scale data (whose data size is usually a fraction of the total size) is to be carefully error-protected in the transmission process. Any error or even a total misinterpretation of the fine-scale data leads to only limited (and usually local) degradation of the image quality. Indeed, as it was stressed above, the maximal size of the corrections is known a priori. Consequently, any error in

their transmission cannot lead to a larger discrepancy of the image than this a priori bound.

References

Geometric Image Representation (VIM)

[1] M. Briskin, Y. Elichai, Y. Yomdin, How can Singularity Theory help in Image Processing, in *Pattern Formation in Biology, Vision and Dynamics*, M. Gromov, A. Carbone, Editors, World Scientific (2000), 392-423.

General Image Processing

[2] D. Rogers, R. Earnshaw, Editors. Techniques for Computer Graphics, Springer-Verlag, 1987.

General Image Compression and Coding

[3] J. Storer, Image and Text Compression, Kluver Academic Publishers, 1992.

Singularity Theory

[4] V.I. Arnol'd, S.M. Gusein-Zade, A.N. Varchenko, Singularities of differentiable maps, Volumes I and II, *Monographs in Mathematics 82 (83)*, Birkhauser Boston, Inc., Boston, MA, 1985 (1988).

Flexible High Order Discretization

[5] Y. Elichai, Y. Yomdin, Flexible high order discretization of geometric data for global motion planning, *Theoretical Computer Science A,* Vol. **157**, Elsevier Science B.V., (1996), 53-77.

[6] Z. Wiener, Y. Yomdin, From formal numerical solutions of elliptic PDE's to the true ones, *Mathematics of Computations* **69**, (2000), No. 229, 197-235.

In the sections of General Image Processing, General Image Compression and Coding, and general Singularity Theory there exists a huge literature. We give only one reference in each of these areas just to help the reader to start.