A vectorial algorith for tracing discrete straight lines in N-dimensional generalized grids

Luis Ibanez, Chafiaa Hamitouche, Christian Roux

To cite this version:

Luis Ibanez, Chafiaa Hamitouche, Christian Roux. A vectorial algorith for tracing discrete straight lines in N-dimensional generalized grids. IEEE Transactions on Visualization and Computer Graphics, 2001, 7 (2), pp.97-108. 10.1109/2945.928163 . hal-02928142

HAL Id: hal-02928142
https://hal.science/hal-02928142

Submitted on 29 Jun 2022

A Vectorial Algorithm for Tracing Discrete Straight Lines in N-Dimensional Generalized Grids

Luis Ibanez, Chafiaa Hamitouche, and Christian Roux

Abstract—This paper presents an algorithm to trace discrete straight lines in regular grids of any dimension. Most known line tracing algorithms have been developed in $\mathbb{Z}^2$ and $\mathbb{Z}^3$ orthogonal grids. The contribution of this paper is the definition of a method to trace lines in nonorthogonal grids in any dimension. This method is not restricted to being used with a specific grid connectivity as other widespread methods are. Good performance can be achieved because only additions are used during line tracing.

Index Terms—Discrete line tracing, digital topology, discrete geometry.

1 INTRODUCTION

THE work presented in this paper was motivated by the need to perform visualization in 3D medical image data stored in a noncubic grid. Specifically, our interest is centered on the use of discrete ray casting in Body Centered Cubic (BCC) and Face Centered Cubic (FCC) grids [17]. The search for an algorithm well-suited to these nonorthogonal grids have led us to construct a general approach for tracing lines in regular grids in any dimension. Our particular approach of representing the grid as a discrete vector space provides a completely general frame in which topological problems can be analyzed. Recent interest in using higher dimensional and nonorthogonal grids has emerged, especially in medical applications [6], [7], [8], [17]. Some of these grids provide important advantages for increasing performance in signal processing tasks [12], [15] and have geometric symmetries which lead to better topological properties [11], [13], [14], [18], [16].

In Section 2, the familiar case of tracing lines in square grids using Bresenham's well-known algorithm is presented. The problem is expressed in an original vector space approach. Section 4 shows how it can be applied to the cubic grid in 3D. A generalization of the procedure to regular grids of any dimension is presented in Section 6. Finally, discussion and perspectives are presented in Section 7.

2 LINE TRACING IN THE SQUARE GRID

2.1 Bresenham's Algorithm

The problem of tracing a line between points A and B in the classical square grid is illustrated in Fig. 1. Shaded squares around points A and B represent the concept of *pixels* used in image processing and computer graphics, filled and hollow circles representing *grid points* as used in discrete topology. The objective of line tracing is to select the set of grid points, or pixels, leading to the best path, approximating the continuous Euclidean line AB. Bresenham's algorithm solves this problem by the following procedure [1]:

```
1. Determine the dominant direction of the line:
   $X$ axis if the slope is less than $45°$, $Y$ axis otherwise.
2. Take one step along the dominant direction.
3. Test if a step should be taken in the secondary direction:
   if affirmative: take it.
4. If not yet in point B, go to (2).
```

Since a discrete line is considered a connected sequence of grid points, the notion of "*connectedness*" should be defined. The most usual definitions of connectedness in the square grid are illustrated in Fig. 2. Four-connectivity defines that the neighbors of a grid point are only the nearest points in the same horizontal and in the same vertical coordinates. Eight-connectivity includes the points in diagonals as neighbors, too.

We present here the procedure for tracing a 4-connected line. The only valid directions of movement to pass from one line point to another are the horizontal and vertical ones. The criterion to decide if a step should be taken in a secondary direction is derived from the analytical equation of the line. Fig. 3 shows the geometrical relations involved in line tracing. Let (x_{a1}, x_{a2}) be the coordinates of point A,

---

- *L. Ibáñez is with the Division of Neurosurgery, CB #7060, University of North Carolina at Chapel Hill, Chapel Hill, NC 27599-7060. E-mail: ibanez@cs.unc.edu.*
- *C. Hamitouche and C. Roux are with the Departement ITI, ENST-Bretagne, Technopole de Brest-Iroise, BP 832, 29285 Brest, France, CEDEX. E-mail: {chafiaa.hamitouche, christian.roux}@enst-bretagne.fr.*

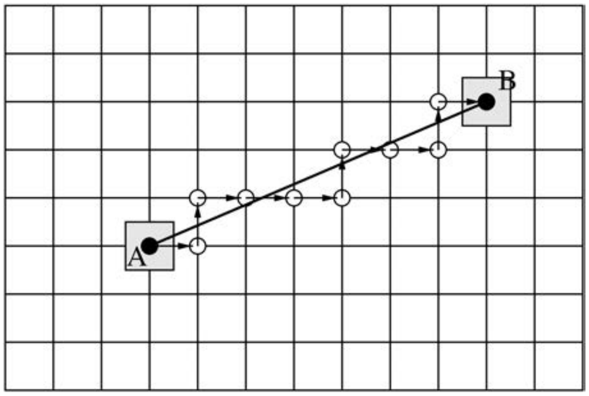

Fig. 1. Tracing a line on the square grid with Bresenham's algorithm. Shaded squares represent pixels; circles represent grid points.

(x_{b1}, x_{b2}) those of point B, and (x_1, x_2) the coordinates of an arbitrary point P. Vector $\mathbf{V_t} = ((x_{b1} - x_{a1}), (x_{b2} - x_{a2}))$ is parallel to the line AB, while vector $\mathbf{V_p} = ((x_{b2} - x_{a2}), -(x_{b1} - x_{a1}))$ is orthogonal to it. The vector $\mathbf{V_{PA}} = ((x_1 - x_{a1}), (x_2 - x_{a2}))$ gives the position of point P relative to point A. The Euclidean distance d from point P to the line AB can be measured by the scalar product shown in (1).

$$d(P, AB) = \frac{\mathbf{V_{PA}} \cdot \mathbf{V_p}}{\|\mathbf{V_p}\|}. \tag{1}$$

This equation can be expressed in terms of coordinates as

$$\begin{aligned} d(P, AB) \cdot \|\mathbf{V_p}\| = {} & (x_1 - x_{a1})(x_{b2} - x_{a2}) \\ & - (x_2 - x_{a2})(x_{b1} - x_{a1}). \end{aligned} \tag{2}$$

Note that all terms on the right side of (2) are integers. The value of $d(P, AB)$ can be interpreted as the error involved in using P as one of the points representing the line AB. The vector norm $\|\mathbf{V_p}\|$ and the coordinates of points A and B are constant. The equation of the continuous Euclidean line AB is obtained when d is identically null. That is, the Euclidean line AB is defined as the set of points P such that $d(P, AB) = 0$, hence,

$$x_2 = \frac{(x_{b2} - x_{a2})}{(x_{b1} - x_{a1})}(x_1 - x_{a1}) + x_{a2}. \tag{3}$$

To avoid noninteger values in the computation, the product $d(P, AB) \cdot \|\mathbf{V_p}\|$ is used instead of the true distance $d(P, AB)$ when deciding whether a point belongs to the discrete line or not. This constant factor $\|\mathbf{V_p}\|$ does not change the nature of the problem since we have to look for grid points P minimizing $d(P, AB)$ in any case. We call $d(P, AB)$ the *Euclidean* distance and $D(P, AB)$ the *arithmetical* distance from point P to the Euclidean line AB, the latter being defined by

$$D(P, AB) = d(P, AB) \cdot \|\mathbf{V_p}\|. \tag{4}$$

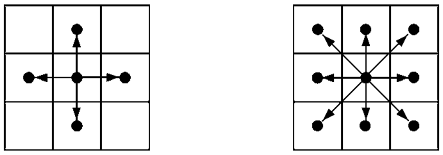

Fig. 2. Connectivity on the square grid. Left: 4-connectivity, one grid point has four neighbors. Right: 8-connectivity, one grid point has eight neighbors. Filled circles represent grid points, squares correspond to pixels.

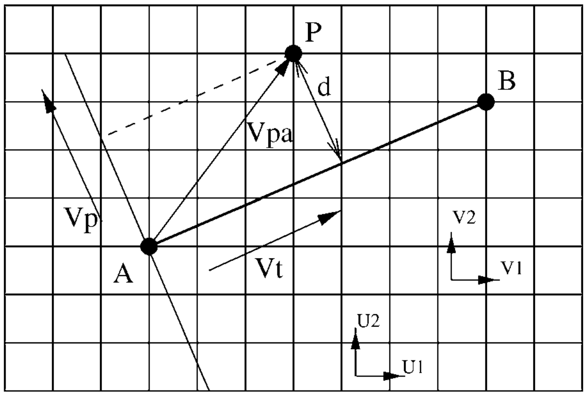

Fig. 3. Vectorial relations of the line tracing problem.

Bresenham's algorithm exploits the fact that (x_1, x_2) are integers. A change of ±1 in the x_1 coordinate produces a change of $\pm(x_{b2} - x_{a2})$ in $D(P, AB)$, while a change of ±1 in the x_2 coordinate produces a change of $\mp(x_{b1} - x_{a1})$ in $D(P, AB)$, as can be deduced from (2) and (4). The value of $D(P, AB)$ is used as a criterion to decide whether a step should be taken in the secondary direction or not during the line tracing process. Bresenham's algorithm computes $D(P, AB)$ in an incremental way, updating its value at each step. For example, to trace a line with a slope value below 45° and 4-connectivity, the dominant direction is the horizontal and the secondary direction is the vertical one. The pseudocode for tracing this line is therefore

```
1. Initialize current point $P$ at position $A$:
   $x_1 = x_{a1}$, $x_2 = x_{a2}$
2. Initialize control variable: $D = 0$.
3. Take a step in dominant direction $X$:
   add 1 to $x_1$, add $2(x_{b2} - x_{a2})$ to $D$.
4. Test if a step should be taken in the
   secondary direction $Y$:
   if $D > [(x_{b1} - x_{a1}) - (x_{b2} - x_{a2})]$ then
   add 1 to $x_2$, add $-2(x_{b1} - x_{a1})$ to $D$
5. If point $P$ is different from point $B$,
   go to Step 3, else end.
```

Where variable D in the pseudocode stands for $2D(P, AB)$ and is named *control variable* because it determines which action must be taken.

2.2 Vectorial Approach to Bresenham's Algorithm

The algorithm proposed in this paper is motivated by the need to trace lines in non orthogonal grids. The first thing that should be done to manage this kind of grid is to use a vector basis to represent them. This section addresses the formalization of Bresenham's algorithm in vectorial terms.

2.2.1 Vector Space Concepts

The simplicity of the square grid makes it unnecessary to use a vector space description, but its presence is implicit anyway. When a point P is said to have coordinates (x_1, x_2), this means that the vector $\mathbf{P}$ relating point P to the origin is: $\mathbf{P} = x_1\mathbf{V}_1 + x_2\mathbf{V}_2$, where vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ are the unitary vectors in the horizontal and vertical directions, respectively, as shown in Fig. 3.

Vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ both belong to the $\mathbb{R}^2$ space, while x_1 and x_2 coordinates both belong to the Z space. For this reason, it is usual to consider only coordinates (x_1, x_2) and place the problem in the $\mathbb{Z}^2$ space. Nevertheless, this simplification cannot be done when dealing with nonorthogonal grids. Vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ are said to be the vector basis of the grid. It is useful to define another vector basis composed of vectors $\mathbf{U}_1$ and $\mathbf{U}_2$ such that

$$\mathbf{V}_1 \cdot \mathbf{U}_1 = 1, \quad \mathbf{V}_1 \cdot \mathbf{U}_2 = 0, \quad \mathbf{V}_2 \cdot \mathbf{U}_1 = 0, \quad \mathbf{V}_2 \cdot \mathbf{U}_2 = 1. \tag{5}$$

The vector basis $\{\mathbf{U}_1, \mathbf{U}_2\}$ is said to be the *reciprocal* of $\{\mathbf{V}_1, \mathbf{V}_2\}$. Every vector $\mathbf{V}$ in a grid can be expressed as a linear combination of the grid basis vectors as: $\mathbf{V} = x_1\mathbf{V}_1 + x_2\mathbf{V}_2$, where $x_1, x_2 \in \mathbb{Z}$. If a grid vector $\mathbf{V}$ is expressed in terms of the grid basis, any vector orthogonal to $\mathbf{V}$ must be expressed in terms of the reciprocal vector basis. That is trivial in the square grid because the grid basis and its reciprocal are the same, but it is not so simple in nonorthogonal grids.

2.2.2 Connectivity Expressed in Terms of Vectors

The grid connectivity is completely specified by the set of vectors describing all the possible relative positions of two neighboring points. For example, 4-connectivity in the square grid is defined by the vector set T:

$$T = \{(1,0), (-1,0), (0,1), (0,-1)\}. \tag{6}$$

The relation *"be neighbor of"* is symmetric, hence, if a vector $\mathbf{V}$ is in the set T, vector $-\mathbf{V}$ should be too. The set T of vectors corresponding to 8-connectivity shown in Fig. 2 is

$$\begin{aligned} T = \{&(1,0), (-1,0), (0,1), (0,-1), (1,1), \\ &(-1,1), (1,-1), (-1,-1)\}. \end{aligned} \tag{7}$$

One of the first things to do when tracing lines is to choose the type of connectivity. The concepts of principal and secondary directions used in Section 2.1 correspond simply to the pair of vectors defining which movements are possible from one grid point to a neighbor. Tracing a 4-connected line implies using only vectors $\mathbf{V}_1 = (1,0)$ and $\mathbf{V}_2 = (0,1)$ as valid directions.

2.2.3 Quadrant Selection

As is usually done, we consider only one quadrant because the others can be obtained by symmetry [1], [3], [5], [14], [22]. The selection of the quadrant in which a line represented by a vector $\mathbf{V}_t$ should be traced corresponds in vectorial terms to the choice of two vectors out of all vectors in the set T. The right quadrant is selected by taking the pair of vectors from T in which vector $\mathbf{V}_t$ can be expressed with components $x_1 \geq 0$ and $x_2 \geq 0$, x_1, x_2 being as small as possible. This choice will give us the two vectors which are best aligned with the Euclidean line AB among all vectors in T. It should be noted that the pair $\{\mathbf{V}_1, \mathbf{V}_2\}$ must form a vector basis in order to guarantee that a vector $\mathbf{V}_t$ can be expressed with integer components (x_1, x_2). Further simplification can be done to analyze only the case $(x_1 \geq x_2)$ because the other case is obtained by symmetry. The task of quadrant selection is trivial in the square grid, but it becomes complex in nonorthogonal and higher dimensional grids [11], [13], [14], and the vectorial approach presented here is general enough to face this complexity.

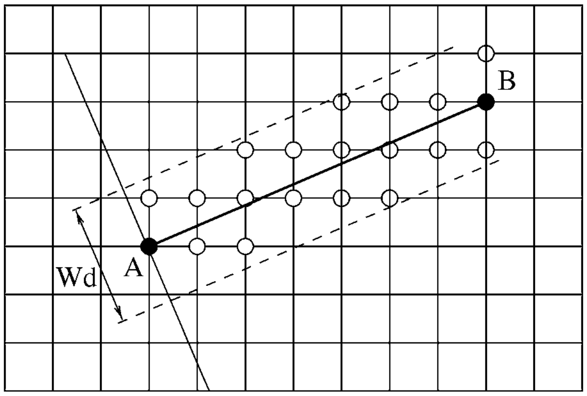

Fig. 4. Digital line defined by Reveillès.

2.2.4 Vector Component Paradigm

When tracing a line, we want to obtain a connected set of points. Hence, we are restricted to taking steps following either $\mathbf{V}_1$ or $\mathbf{V}_2$. The Vector $\mathbf{V}_t$ relating end points A and B can be expressed in terms of a vector basis $\{\mathbf{V}_1, \mathbf{V}_2\}$ as: $\mathbf{V}_t = x_1\mathbf{V}_1 + x_2\mathbf{V}_2$. From the components (x_1, x_2), we know that to go from point A to point B, we have to take x_1 steps in direction $\mathbf{V}_1$ and x_2 steps in direction $\mathbf{V}_2$. The problem is now reduced to establish in which *order* these steps should be taken. In order to better approximate the Euclidean line AB, as defined in (3), we would like to put the points of the discrete line as near as possible to the line AB.

2.3 Definitions of Discrete Lines

2.3.1 Digital Geometry Definition

A fairly general definition of the 2D digital line has been proposed by Reveillès [20] and was further extended to digital planes by Andrès [21]. Their approach considers a band of width W_d around the Euclidean line AB. Grid points lying inside this band belong to the digital line, as is shown in Fig. 4, and these points satisfy the following equation:

$$0 \leq Mx_1 + Nx_2 < w, \tag{8}$$

where M, N, and w are integers. The term w in (8) is called the *arithmetic* width and can be related to the Euclidean width W_d of the discrete line shown in Fig. 4 by $w = W_d\sqrt{M^2 + N^2}$. A relation between line width w and connectivity has been established as follows [20]:

- $w = |M| + |N|$ produces 4-connected lines which are called *standard* lines,
- $w = max(|M|, |N|)$ produces 8-connected lines, called *naive* lines.

2.3.2 Vector Space-Based Definition

In our method, the concept of a discrete straight line is built from the notion of connectivity as defined by the set T of vectors in Section 2.2.2. These vectors describe the relative positions between two grid points that make it possible to consider them as neighbors. A discrete curve between grid points A and B is defined here as a sequence S of grid points beginning in A and ending in B such that every point has exactly two neighbors in S except for the end points A and B which have only one neighbor in S. A similar definition can be found in [25], but is restricted to square and cubic grids. A discrete straight line L is a particular case of a discrete curve S. The notion of *straightness* [27] leads us to look for a discrete curve S such that the distances from its points to the Euclidean line AB are as small as possible. Our procedure is such that, for any point of the sequence, the next point is choosen as the closest point to the Euclidean line among the points that can be reached by movements defined by the basis vectors of the quadrant.

2.3.3 Procedure for Tracing the Line

The first point P of the discrete curve is placed at point A. The distance $D(P, AB)$ is zero in this case. The decision about whether to take the next step in direction $\mathbf{V}_1$ or $\mathbf{V}_2$ is taken depending on the outcome of the test:

$$2 \cdot D(P, AB) < (x_{b1} - x_{a1}) - (x_{b2} - x_{a2}). \tag{9}$$

If the test's outcome is true, the next step is taken along the dominant direction $\mathbf{V}_1$; otherwise, direction $\mathbf{V}_2$ is preferred. A step taken in direction $\mathbf{V}_1$ produces an increase of $(x_{b2} - x_{a2})$ in $D(P, AB)$, while a step taken in direction $\mathbf{V}_2$ produces an increase of $-(x_{b1} - x_{a1})$ in $D(P, AB)$. The value of $2 \cdot D(P, AB)$ is defined as the *control variable* of the algorithm. Subsequent steps are taken by iteratively applying the test of (9). The procedure ends when the curve passes by point B, which can be guaranteed because $D(B, AB) = 0$ and the method advance selecting the points with minimum $|D(B, AB)|$.

3 HEXAGONAL GRID CASE

3.1 Introduction

The hexagonal grid is a good case for introducing the line tracing problem in nonorthogonal grids. This grid can be defined by vectors $\mathbf{V}_1 = (1, 0)$ and $\mathbf{V}_2 = (\frac{1}{2}, \frac{\sqrt{3}}{2})$ which form a vector basis. Vector $\mathbf{P}$ represents the position of a grid point P with respect to the origin. It can be expressed as a linear combination of vectors $\mathbf{V}_1$ and $\mathbf{V}_2$, defined by $\mathbf{P} = x_1\mathbf{V}_1 + x_2\mathbf{V}_2$. The integer numbers (x_1, x_2) are said to be the components of vector $\mathbf{P}$ in the vector basis $\{\mathbf{V}_1, \mathbf{V}_2\}$. The basic elements of the line tracing problem in the hexagonal grid are shown in Fig. 5. In terms of vectors we have: The position of point A relative to the origin is $\mathbf{A} = x_{a1}\mathbf{V}_1 + x_{a2}\mathbf{V}_2$, the position of point B relative to the origin is $\mathbf{B} = x_{b1}\mathbf{V}_1 + x_{b2}\mathbf{V}_2$, and the position of point B relative to point A is $\mathbf{V_t} = (x_{b1} - x_{a1})\mathbf{V}_1 + (x_{b2} - x_{a2})\mathbf{V}_2$. Vector $\mathbf{V_t}$ is parallel to the Euclidean line AB.

Differences with respect to square grid begin to appear when trying to find the vector $\mathbf{V_p}$ normal to vector $\mathbf{V_t}$. In nonorthogonal grids, a normal vector cannot necessarily be expressed (with integer coefficients) in terms of the grid vector basis. It can, however, be expressed in terms of the reciprocal grid vector basis $\{\mathbf{U}_1, \mathbf{U}_2\}$. These vectors can be obtained by placing basis vectors $\{\mathbf{V}_1, \mathbf{V}_2\}$ as columns of a matrix M_v:

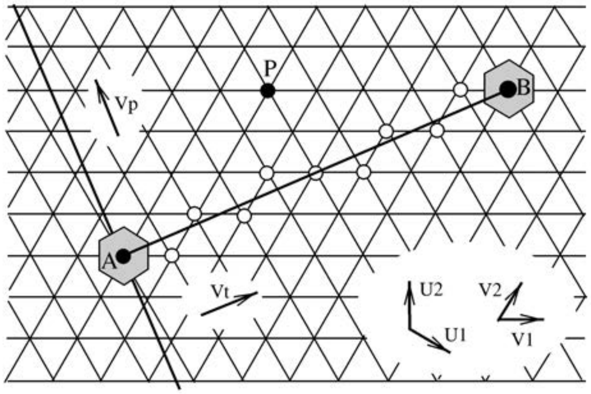

Fig. 5. Tracing a digital line on the hexagonal grid.

$$M_v = [\,\mathbf{V}_1 \quad \mathbf{V}_2\,] = \begin{bmatrix} 1 & \frac{1}{2} \\ 0 & \frac{\sqrt{3}}{2} \end{bmatrix}. \tag{10}$$

Let the matrix M_u be defined by

$$M_u = M_v \left(M_v^T M_v\right)^{-1}. \tag{11}$$

The columns of matrix M_u are the vectors $\{\mathbf{U}_1, \mathbf{U}_2\}$. This last equation can be used even if the number of vectors is inferior to the dimension of the space, that is, if the $\mathbf{V_i}$ vectors span a subspace, for example, two vectors in a three-dimensional space. A particular case happens when the number of vectors is equal to the space dimension, in this case, M_v is a square matrix and the expression can be simplified to

$$M_u = \left(M_v^T\right)^{-1}. \tag{12}$$

In the particular case of the hexagonal grid, the result is

$$M_u = [\,\mathbf{U}_1 \quad \mathbf{U}_2\,] = \begin{bmatrix} 1 & 0 \\ \frac{-\sqrt{3}}{3} & \frac{2\sqrt{3}}{3} \end{bmatrix}, \tag{13}$$

which means that $\mathbf{U}_1 = (1, \frac{-\sqrt{3}}{3})$ and $\mathbf{U}_2 = (0, \frac{2\sqrt{3}}{3})$.

The vector $\mathbf{V_p}$ orthogonal to $\mathbf{V_t}$ can be obtained by commuting the components of vector $\mathbf{V_t}$ expressed in the vector basis $\{\mathbf{V}_1, \mathbf{V}_2\}$ and changing the sign of one of them. This gives us: $\mathbf{V_p} = -(x_{b2} - x_{a2})\mathbf{U}_1 + (x_{b1} - x_{a1})\mathbf{U}_2$. This result is crucial for the line tracing process because vector $\mathbf{V_p}$ is used in (1) to compute the distance $d(P, AB)$. In the particular case of the example shown in Fig. 5, we have $\mathbf{V_t} = 6\mathbf{V}_1 + 4\mathbf{V}_2$ and $\mathbf{V_p} = -4\mathbf{U}_1 + 6\mathbf{U}_2$. Taking into account (5), it is easy to verify that $\mathbf{V_t} \cdot \mathbf{V_p} = 0$.

3.2 Line Tracing

3.2.1 Selecting the Grid Connectivity

In the hexagonal grid, it is natural to choose a 6-connectivity, as shown in Fig. 6. This leads to defining a set T of vectors as

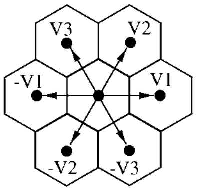

Fig. 6. Connectivity on the hexagonal grid. Filled circles represent grid points, hexagons represent pixels. Vectors $\mathbf{V}_1$, $\mathbf{V}_2$, $\mathbf{V}_3$, and their negatives relate each grid point to its neighbors.

$$T = \{\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3, -\mathbf{V}_1, -\mathbf{V}_2, -\mathbf{V}_3\}, \tag{14}$$

with vector $\mathbf{V}_1$ and $\mathbf{V}_2$ as defined in Section 3.1 and vector $\mathbf{V}_3 = (-\frac{1}{2}, \frac{\sqrt{3}}{2})$.

3.2.2 Determination of the Vector Basis

Once the end points A and B of the line are defined, we look for the vector basis that is best adapted to describe the vector $\mathbf{V_t}$ connecting points A and B. As stated in Section 2.2.3, we select from set T the pair of vectors forming a vector basis such that the components of $\mathbf{V_t}$ satisfy $(x_1 \geq 0)$, $(x_2 \geq 0)$; (x_1, x_2) being as small as possible. This selection of basis is equivalent to the choice of the quadrant in the square grid.

The reason for including the above condition for (x_1, x_2) can be easily seen from Fig. 5 and Fig. 6. For example, $\mathbf{V_t}$ of Fig. 5 can be expressed in one of the following ways, among others:

- in terms of vector basis $\{\mathbf{V}_2, -\mathbf{V}_3\}$ with components $(10, 6)$,
- in terms of vector basis $\{\mathbf{V}_1, \mathbf{V}_3\}$ with components $(10, 4)$,
- in terms of vector basis $\{\mathbf{V}_1, \mathbf{V}_2\}$ with components $(6, 4)$.

The latter case simply represents the notion of taking the two vectors in T aligned in the best way with the Euclidean line AB.

3.2.3 Computation of the Orthogonal Vector

To find the vector $\mathbf{V_p}$ orthogonal to $\mathbf{V_t}$ we need to first build the reciprocal vector basis $\{\mathbf{U}_1, \mathbf{U}_2\}$. This can be done by the procedure detailed in Section 3.1. This procedure does not necessarily imply an overhead in computing time because it can be performed before tracing any specific line. That is, the reciprocal vector basis can be precomputed for all possible combinations of vector basis $\{\mathbf{V}_1, \mathbf{V}_2\}$. In the hexagonal grid case, the possible combinations are only six.

Once the vector basis is known, the vector $\mathbf{V_p}$ can be built by exchanging the components of vector $\mathbf{V_t}$ and changing the sign of one of them. If we have two vectors $\mathbf{X}$ and $\mathbf{Y}$, the first expressed in terms of the grid vector basis as $\mathbf{X} = x_1\mathbf{V}_1 + x_2\mathbf{V}_2$ and the second in terms of the reciprocal vector basis as $\mathbf{Y} = y_1\mathbf{U}_1 + y_2\mathbf{U}_2$, then their scalar product, taking into account (5), is $\mathbf{X} \cdot \mathbf{Y} = x_1 y_1 + x_2 y_2$. Hence, an easy way to set $\mathbf{Y}$ orthogonal to $\mathbf{X}$ is to choose $y_1 = x_2$ and $y_2 = -x_1$.

3.2.4 Definition of the Control Variable

During the line tracing process, the distance from grid points P to the Euclidean line AB is computed using (1). To avoid noninteger numbers, it can be transformed to compute $2D(P, AB)$ instead of $d(P, AB)$ following (2).

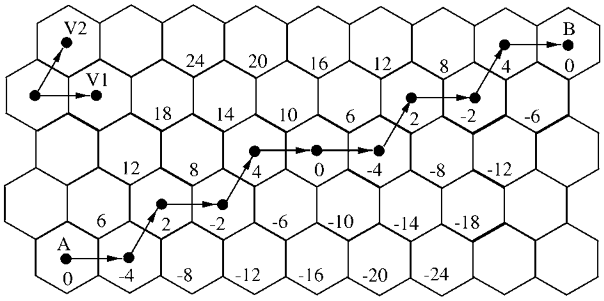

Fig. 7. Values of control variable in each hexagonal grid point (pixel) to trace a line from point A to point B.

Bresenham's algorithm, presented in Section 2.1, can be used without modification to trace lines in the hexagonal grid, provided that its terms are interpreted in the context presented in this section.

The value of distance $D(P, AB)$ for most hexagonal pixels in the particular case of $\mathbf{V_t} = 6\mathbf{V}_1 + 4\mathbf{V}_2$ is shown in Fig. 7. The process can be described a strategy for going from pixel A to pixel B, taking steps in either direction $\mathbf{V}_1$ or $\mathbf{V}_2$, and choosing, at each time, to pass by the pixel with the lower absolute value $|D(P, AB)|$.

From this description it is easy to see that an ambiguity arises if the two possible cases have the same absolute value, which happens when the components (x_1, x_2) of $\mathbf{V_t}$ are equal. This is the trivial case of tracing a 45° slope line in the square grid using 4-connectivity. The question is whether to move horizontally first or vertically first. The answer is simply: *You can choose*. But, the choice should be the same each time the ambiguity arises again.

3.2.5 The Extension of Reveillès's Definition to the Hexagonal Grid

The definition of the digital line proposed by Reveillès in [20] can be easily extended to nonorthogonal grids. Equation (8) can be used with the difference that the terms x_1 and x_2 should be interpreted as the components of the vector position $\mathbf{P}$ of point P expressed in the grid vector basis $\{\mathbf{V}_1, \mathbf{V}_2\}$. The terms M and N should be interpreted geometrically as the components of the vector $\mathbf{V_p}$ orthogonal to the Euclidean line. This vector can be expressed in the *reciprocal* vector basis $\{\mathbf{U}_1, \mathbf{U}_2\}$ as $\mathbf{V_p} = M\mathbf{U}_1 + N\mathbf{U}_2$.

Note that Reveillès definition produces digital lines which lie at one side of the Euclidean line instead of being centered over it. This can be corrected by subtracting $w/2$ from (8) and multiplying by a factor 2 to avoid fractions:

$$-w \leq 2Mx_1 + 2Nx_2 < w. \tag{15}$$

In the hexagonal grid, it can be proven that the choice of $w = |M| + |N|$ produces 6-connected lines. For the particular case shown in Fig. 7, we have $M = -4$, $N = 6$, and $w = 10$, hence, Reveillès's standard line is composed of hexagonal pixels such that $-\frac{w}{2} \leq D(P/AB) < \frac{w}{2}$, which coincides with the line we traced because we have used 6-connectivity.

4 3D CASE

The cubic grid or $\mathbb{Z}^3$ grid is the most commonly studied 3D case for line tracing. It has been addressed intensively by research involved in volume visualization and digital geometry [2], [4], [10], [22], [25], [28], [33], [31]. A general approach to rasterize parametric curves, surfaces, and volumes in the $\mathbb{Z}^3$ grid is presented in [28]. Some methods used in the cubic grid have been generalized to the $\mathbb{Z}^N$ space. An *N*-dimensional Bresenham's algorithm is presented in [32]. A definition of discrete line in $\mathbb{Z}^N$ is presented in [26] and a definition based on combinatorics is presented in [29].

4.1 3D Digital Line Definitions Proposed

Kim [25] has defined the digital line in the cubic grid as a sequence of grid points whose projections on the coordinate planes are 2D 8-connected digital lines. An algorithm is proposed to recognize a digital line but not to trace it. Andres et al. have proposed a definition of the digital line using the concept of supercover [23], the digital line being composed of all the grid points whose Voronoi regions are intersected by the Euclidean line. This approach has the disadvantage of producing digital lines in which one grid point can have more than two neighbors in the line, especially in nonorthogonal grids.

Figueiredo and Reveillès have defined the 3D digital line as the intersection of two nonparallel digital planes [22]. This definition has recently been used for the implementation of a discrete ray casting algorithm [30]. The projection of a point on the plane orthogonal to the line is used to guide the line tracing process. Their algorithm is restricted to the $\mathbb{Z}^3$ grid and 26-connectivity. Digital planes have also been defined by Andrès [21] as a generalization of the Reveillès 2D digital line [20].

A discrete straight line is defined here as the sequence of grid points such that the first and the last point of the sequence are the endpoints of the Euclidean line and, for each point in the sequence, the next point is the one that is closest to the Euclidean line among the three points that can be reached by the possible movements defined in the vector basis of the selected octant.

4.2 The Vectorial Approach

4.2.1 Introduction

The position of point A with respect to the origin is represented by vector $\mathbf{A} = x_{a1}\mathbf{V}_1 + x_{a2}\mathbf{V}_2 + x_{a3}\mathbf{V}_3$. The position of point B with respect to the origin is represented by vector $\mathbf{B} = x_{b1}\mathbf{V}_1 + x_{b2}\mathbf{V}_2 + x_{b3}\mathbf{V}_3$. The position of point B with respect to point A is the vector $\mathbf{V_t} = (x_{a1} - x_{b1})\mathbf{V}_1 + (x_{a2} - x_{b2})\mathbf{V}_2 + (x_{a3} - x_{b3})\mathbf{V}_3$.

4.2.2 Defining Connectivity

To define connectivity, we have to specify the set T of vectors connecting the positions of grid points considered as neighbors. Note that this is a *choice* which must be taken prior to the line tracing process.

4.2.3 Selecting the Octant

The process analog to choosing the quadrant in the square grid in 2D is the selection of the octant in the cubic grid or $\mathbb{Z}^3$ grid and in a general 3D grid case is the choice of a cell spanned by three vectors. This corresponds to the choice of three vectors out of the set T such that they form a vector basis. The selection of vector basis depends on the line to be traced. The right basis is the one for which the components (x_1, x_2, x_3) of vector $\mathbf{V_t}$ satisfy $(x_1 \geq 0)$, $(x_2 \geq 0)$, $(x_3 \geq 0)$, and are as small as possible. This criterion guarantees to get the combination of vectors that are aligned in the best way with the line. The direction associated with the maximum value among $\{x_1, x_2, x_3\}$ is called the *dominant* direction, the others are called *secondary* directions.

4.2.4 Determination of the Reciprocal Grid

Once the vector basis is chosen, the reciprocal vector basis should be computed. Let the vector basis be composed of vectors $\{\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3\}$. The procedure is analog to the one developed for 2D in Section 3.1. The 3×3 matrix M_v is formed by placing vectors $\{\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3\}$ as columns. The Matrix M_u is computed using (11). The columns of matrix M_u are the vectors $\{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3\}$ forming the reciprocal basis [13]. They verify the basic relation between a reciprocal basis and the grid vector basis:

$$\mathbf{V_i} \cdot \mathbf{U_j} = \begin{cases} 1 & if \quad i = j \\ 0 & if \quad i \neq j \end{cases} \; \forall i, j \in [1, N], \tag{16}$$

where N is the dimension of the grid, in our current case $N = 3$.

4.2.5 Computing the Distance to the Line

To compute the distance from a grid point P to the Euclidean line AB, we now need not only one vector $\mathbf{V_p}$ orthogonal to $\mathbf{V_t}$, as in the 2D cases treated in Section 2 and Section 3, but two such vectors. Let $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$ be two nonparallel vectors, both orthogonal to vector $\mathbf{V_t}$. These two vectors, not necessarily orthogonal between them, define a plane in the space $\mathbb{R}^3$. A Euclidean line AB is a one-dimensional object. When it is placed in an N-dimensional space $\mathbb{R}^N$, the distance from a point P to the line should be calculated in an $(N-1)$-dimensional space $\mathbb{R}^{(N-1)}$. This space is called here the *subspace* orthogonal to the Euclidean line AB. In the particular case of 3D space ($N = 3$), the orthogonal subspace is simply the plane normal to the line. The vectors $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$ describe this subspace in the sense that they can be used as its vector basis. The position of any point in the orthogonal subspace can be expressed as a linear combination of $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$. It should be noted that, even though $\mathbf{U_{p1}}, \mathbf{U_{p2}} \in \mathbb{R}^3$, the orthogonal subspace that they represent is a plane.

Computing vectors to describe the orthogonal subspace. This section presents how the vectors $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$ can be computed in terms of the reciprocal basis vectors $\{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3\}$. The method presented here is not unique, but it has the advantage of producing two vectors well-suited for describing the subsequent line tracing process. The line to be traced is defined by vector $\mathbf{V_t} = x_1\mathbf{V}_1 + x_2\mathbf{V}_2 + x_3\mathbf{V}_3$, with $x_1 = (x_{b1} - x_{a1})$, $x_2 = (x_{b2} - x_{a2})$, and $x_3 = (x_{b3} - x_{a3})$, as presented in Section 4.2.1. For the sake of simplicity, we address here the particular case of $x_1 \leq x_2 \leq x_3$ in which $\mathbf{V}_3$ is the dominant direction; the other cases can be solved by symmetry [14], [22]. We apply a process analog to that used

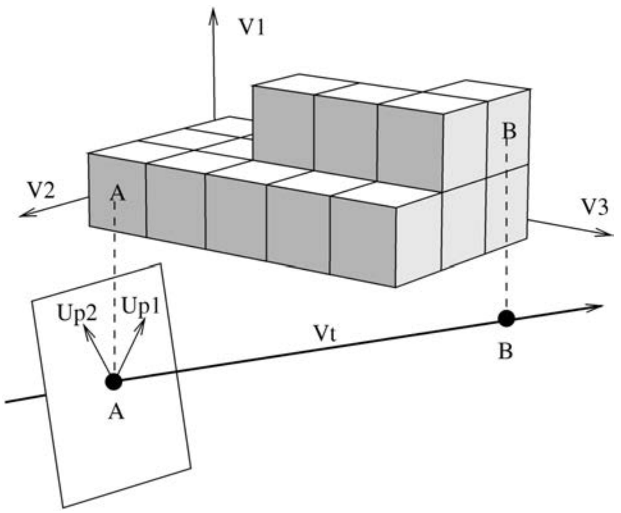

Fig. 8. The line tracing problem in the cubic grid. A line is traced between points A and B.

in 2D, see Section 3.2.3. Two components of vector $\mathbf{V_t}$ are exchanged, the sign of one of them is commuted, and both are used as components in the reciprocal vector basis. The remaining components are set to zero. Hence, to construct $\mathbf{U_{p1}}$, we take x_1 and x_3 to form $\mathbf{U_{p1}} = x_3\mathbf{U}_1 + 0\mathbf{U}_2 - x_1\mathbf{U}_3$. To construct $\mathbf{U_{p2}}$, we take x_2 and x_3 to form $\mathbf{U_{p2}} = 0\mathbf{U}_1 + x_3\mathbf{U}_2 - x_2\mathbf{U}_3$. It can be easily verified from (16) that $\mathbf{U_{p1}} \cdot \mathbf{V_t} = 0$, $\mathbf{U_{p2}} \cdot \mathbf{V_t} = 0$, $\mathbf{U_{p1}} \cdot \mathbf{V}_2 = 0$, and $\mathbf{U_{p2}} \cdot \mathbf{V}_1 = 0$. A schematic representation of vectors $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$ is shown in Fig. 8, where the plane represents the subspace orthogonal to $\mathbf{V_t}$.

In matrix notation, the vectors $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$ can be arranged as the columns of matrix M_{Up}, related to M_u by

$$M_{Up} = M_u \, M_{sp}, \tag{17}$$

where M_{sp} is called *subspace projection matrix* and is defined by

$$M_{sp} = \begin{bmatrix} x_3 & 0 \\ 0 & x_3 \\ -x_1 & -x_2 \end{bmatrix}. \tag{18}$$

In the same way that vector basis $\{\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3\}$ has a reciprocal basis $\{\mathbf{U}_1, \mathbf{U}_2, \mathbf{U}_3\}$, vector basis $\{\mathbf{U_{p1}}, \mathbf{U_{p2}}\}$ of the orthogonal subspace has a reciprocal basis $\{\mathbf{W_{p1}}, \mathbf{W_{p2}}\}$. They satisfy

$$\mathbf{W_{p\mathit{i}}} \cdot \mathbf{U_{p\mathit{j}}} = \begin{cases} 1 & if \quad i = j \\ 0 & if \quad i \neq j \end{cases} \forall i, j \in [1, N-1] \tag{19}$$

and can be constructed from vectors $\{\mathbf{U_{p1}}, \mathbf{U_{p2}}\}$ using the procedure described in Section 3.1, particularly (11).

Projection of a vector on the orthogonal subspace. Let $\mathbf{M} = m_1\mathbf{V}_1 + m_2\mathbf{V}_2 + m_3\mathbf{V}_3$ be a grid vector. Let vector $\mathbf{M_p}$ be the projection of $\mathbf{M}$ on the orthogonal subspace. It can be expressed as $\mathbf{M_p} = m_{p1}\mathbf{W_{p1}} + m_{p2}\mathbf{W_{p2}}$, where the components m_{p1} and m_{p2} are integers. These components can be computed from the scalar product of vector $\mathbf{M}$ with the basis vectors of the orthogonal subspace $\mathbf{U_{p1}}$ and $\mathbf{U_{p2}}$:

$$\begin{aligned} m_{p1} &= \mathbf{M} \cdot \mathbf{U_{p1}} = m_1 x_3 - m_3 x_1 \\ m_{p2} &= \mathbf{M} \cdot \mathbf{U_{p2}} = m_2 x_3 - m_3 x_2. \end{aligned} \tag{20}$$

It is interesting to note that the projection of $\mathbf{V}_1$ on the orthogonal subspace is $(x_3\mathbf{W_{p1}})$, the projection of $\mathbf{V}_2$ is $(x_3\mathbf{W_{p2}})$, and the projection of $\mathbf{V}_3$ is $(-x_1\mathbf{W_{p1}} - x_2\mathbf{W_{p2}})$. That is, each one of the secondary directions of movements is projected parallel to one of the reciprocal vectors of the orthogonal subspace. This property allows us to easily follow the line tracing process using the projection of the discrete line points on the orthogonal subspace. In matrix notation, the components after the projection can be expressed as:

$$\begin{bmatrix} m_{p1} \\ m_{p1} \end{bmatrix} = \begin{bmatrix} x_3 & 0 & -x_1 \\ 0 & x_3 & -x_2 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}. \tag{21}$$

Using the definition of M_{sp} from (18), this is equivalent to

$$\begin{bmatrix} m_{p1} \\ m_{p1} \end{bmatrix} = \left(M_{sp}\right)^T \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}. \tag{22}$$

Euclidean distance to the line. Let vector $\mathbf{M}$ represent the position of a point with respect to point A, then $\mathbf{M} = (\mathbf{P} - \mathbf{A})$. The magnitude of $\mathbf{M_p}$ is the Euclidean distance from the point P to the line AB. This magnitude can be computed from

$$\left\|\mathbf{M_p}\right\|^2 = \mathbf{M_p} \cdot \mathbf{M_p} = [\, m_{p1} \quad m_{p2} \,] \Big(M_{Wp}^T M_{Wp} \Big) \begin{bmatrix} m_{p1} \\ m_{p2} \end{bmatrix}, \tag{23}$$

where M_{Wp} is the matrix formed by using the vectors $\mathbf{W_{pi}}$ as columns

$$M_{Wp} = [\, \mathbf{W_{p1}} \quad \mathbf{W_{p2}} \,], \tag{24}$$

in the same way as in Section 3.1 for (10). Then, the matrix product is

$$M_{Wp}^T M_{Wp} = \begin{bmatrix} \mathbf{W_{p1}} \cdot \mathbf{W_{p1}} & \mathbf{W_{p1}} \cdot \mathbf{W_{p2}} \\ \mathbf{W_{p2}} \cdot \mathbf{W_{p1}} & \mathbf{W_{p2}} \cdot \mathbf{W_{p2}} \end{bmatrix}. \tag{25}$$

The relationship between a vector basis and its reciprocal implies that

$$M_{Wp}^T M_{Wp} = \Big(M_{Up}^T M_{Up} \Big)^{-1}, \tag{26}$$

where

$$M_{Up}^T M_{Up} = \begin{bmatrix} \mathbf{U_{p1}} \cdot \mathbf{U_{p1}} & \mathbf{U_{p1}} \cdot \mathbf{U_{p2}} \\ \mathbf{U_{p2}} \cdot \mathbf{U_{p1}} & \mathbf{U_{p2}} \cdot \mathbf{U_{p2}} \end{bmatrix}. \tag{27}$$

Using (17), we can find that

$$M_{Up}^T M_{Up} = M_{sp}^T \left(M_u^T M_u \right) M_{sp} \tag{28}$$

and, using (11), this becomes

$$M_{Up}^T M_{Up} = M_{sp}^T \left(M_v^T M_v \right)^{-1} M_{sp}. \tag{29}$$

Hence, we can rewrite (23) as

$$\left\|\mathbf{M_p}\right\|^2 = [\, m_{p1} \quad m_{p2} \,] \Big(M_{sp}^T \left(M_v^T M_v \right)^{-1} M_{sp} \Big)^{-1} \begin{bmatrix} m_{p1} \\ m_{p2} \end{bmatrix} \tag{30}$$

and, from (22), we arrive finally at

$$\left\|\mathbf{M_p}\right\|^2 = [\,m_1 \quad m_2 \quad m_3\,] M_{sp}\left(M_{sp}^T\left(M_v^T M_v\right)^{-1} M_{sp}\right)^{-1} M_{sp}^T \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}. \tag{31}$$

Let's note that the inverse cannot be distributed inside the parentheses because the matrix M_{sp} is not square. This expression is equivalent to the quadratic form

$$\left\|\mathbf{M_p}\right\|^2 = \sum_{i,j} m_i m_j \Theta_{ij}, \tag{32}$$

where the symmetric matrix Θ is

$$\Theta = M_{sp}\left(M_{sp}^T\left(M_v^T M_v\right)^{-1} M_{sp}\right)^{-1} M_{sp}^T. \tag{33}$$

M_v is completely defined by the grid vector basis and the octant in which the line is going to be traced. M_{sp} depends only on the components of vector $\mathbf{V_t}$. All the possible M_v matrices can be precomputed once the grid is defined, one for every possible octant. Matrix M_{sp} is computed once the endpoints of the line are defined. In this way, when we start the line tracing process, the elements of matrix Θ are known constants.

Once we are tracing a line, the choice of the direction on which the next step should be taken is made by evaluating the effect of every possible step on the value $\left\|\mathbf{M_p}\right\|^2$. The direction of movement that results in the minimum value for $\left\|\mathbf{M_p}\right\|^2$ will be chosen.

Given that the matrix Θ is constant for a line AB, it is possible to compute the value of $\left\|\mathbf{M_p}\right\|^2$ incrementally. Let's assume that we know the value of $\left\|\mathbf{M_p}\right\|^2$ for a certain point P whose components are $\{m_i\}$ with respect to point A. If one of these components, say m_k, is incremented by one, the value $\left\|\mathbf{M'_p}\right\|^2$ of the new square distance is given by

$$\begin{aligned}\left\|\mathbf{M'_p}\right\|^2 &= \sum_{i,j\neq k} m_i m_j \Theta_{ij} + 2(m_k+1) \\ &\quad \sum_{j\neq k} m_j \Theta_{kj} + (m_k+1)^2 \Theta_{kk} \\ &= \sum_{i,j} m_i m_j \Theta_{ij} + 2\sum_j m_j \Theta_{kj} + \Theta_{kk} \\ &= \left\|\mathbf{M_p}\right\|^2 + 2\sum_j m_j \Theta_{kj} + \Theta_{kk}.\end{aligned}$$

Let's define the vector $\mathbf{R_p}$ as having components $2\sum_j m_j \Theta_{kj}$ on the vector basis $\{\mathbf{U_{Pk}}\}$

$$\mathbf{R_P} = \sum_k \left(2\sum_j m_j \Theta_{kj}\right) \mathbf{U_{Pk}} \tag{34}$$

and define M_R the $1 \times n$ matrix composed with its components

$$M_R = \begin{bmatrix} 2\sum_j m_j \Theta_{1j} \\ 2\sum_j m_j \Theta_{2j} \\ \cdots \\ 2\sum_j m_j \Theta_{nj} \end{bmatrix}, \tag{35}$$

the scalar product $\mathbf{R_p} \cdot \mathbf{M_p}$ is equal to $\left\|\mathbf{M_p}\right\|^2$. In some formalizations, the vector $\mathbf{M_p}$ will be called *covariant* and the vector $\mathbf{R_p}$ will be called *contravariant*. To update the value of $\left\|\mathbf{M_p}\right\|^2$, we have to increment it by the *k*th component of M_R plus the term Θ_{kk}. The column matrix M_R has to be updated too by adding to it the double of the *k*th column from Θ in order to be ready for the next iteration

$$M'_R = \begin{bmatrix} 2\sum_j m_j \Theta_{1j} \\ 2\sum_j m_j \Theta_{2j} \\ \cdots \\ 2\sum_j m_j \Theta_{nj} \end{bmatrix} + 2\begin{bmatrix} \Theta_{1k} \\ \Theta_{2k} \\ \cdots \\ \Theta_{nk} \end{bmatrix}. \tag{36}$$

In this way, by keeping the $n \times n$ matrix Θ and the $1 \times n$ matrix M_R as auxiliary data, it is possible to compute the square distance from one grid point P to the Euclidean line AB with only $(n+1)$ additions.

4.3 Line Tracing

To trace the line we must take the sequence S of connected points which minimizes the distance $D(S, AB)$ as stated in Section 2.3.2. The distance from a point P to the Euclidean line AB is now computed by projecting the point P onto the subspace orthogonal to the line. This subspace is defined by the vector basis $\{\mathbf{U_{p1}}, \mathbf{U_{p2}}\}$. Since we are considering the case $x_1 \leq x_2 \leq x_3$, it is clear that the dominant direction of the line is that of $\mathbf{V}_3$. The other two vectors $\mathbf{V}_1$ and $\mathbf{V}_2$ are secondary directions. Knowing the octant and the endpoints of the line, we can precompute the matrix Θ.

Let's consider that we start with the point P coincident with point A. Vector $\mathbf{M} = \mathbf{P} - \mathbf{A}$ has components $\{m_1, m_2, m_3\}$ all null. The projection of $\mathbf{M}$ on the orthogonal subspace is $\mathbf{M_p}$. Its magnitude $\left\|\mathbf{M_p}\right\|$ is the distance from point P to the line AB. Our strategy is to always advance to the next point in the grid that results in the minimum value of $\left\|\mathbf{M'_p}\right\|$. As the value $\left\|\mathbf{M'_p}\right\|^2$ is computed incrementally from the value $\left\|\mathbf{M_p}\right\|^2$, we can concentrate our test into a search for the minimum possible increment.

The query for the best next step is then equivalent to the query for the k that minimizes the expression $2\sum_j m_j \Theta_{kj} + \Theta_{kk}$. In the implementation, the values of the first term are stored in the column matrix M_R, which is initially null. The values of the second term are the diagonal of matrix Θ. The test implies to add the matrix M_R and the diagonal of Θ and select the *k*th element for which this sum is minimum. Then,

$$k = \arg_i \min\left\{2\sum_j m_j \Theta_{ij} + \Theta_{ii}\right\}. \tag{37}$$

In practice, a column matrix M_T can be computed as the sum

$$M_T = M_R + diagonal(\Theta). \tag{38}$$

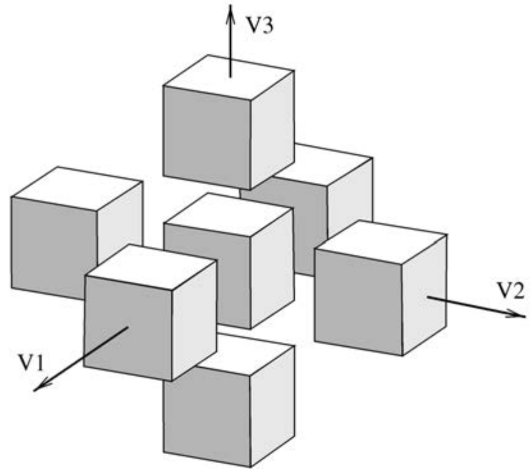

Fig. 9. Six-connectivity in the cubic grid. The voxel in the center has six neighbors.

Once k is identified, a step is taken in that direction by incrementing the component m_k by one. Then the matrix M_R is updated by adding the kth column of Θ to it. The actual value of $\left\|\mathbf{M}_\mathbf{p}\right\|^2$ doesn't need to be computed; it is sufficient to chose the minimal increment at each time.

This criterion acts locally and, for this reason, it cannot guarantee that the discrete line traced will be the global optimal line. It produces, however, lines composed by points that try to *adhere* to the Euclidean line AB in a local best effort.

4.4 The Algorithm to Trace 3D Discrete Lines

It is now possible to modify Bresenham's algorithm of Section 2.1 and apply it to 3D lines. The resulting algorithm has the following steps:

```
1. Initialize current point $P$ in position $A$:
   $x_{p1} = x_{a1}$, $x_{p2} = x_{a2}$, $x_{p3} = x_{a3}$.
2. Compute the matrix $\Theta$.
3. Initialize column matrix $M_R$ to zero.
4. Compute the column matrix $M_T$ as the sum of
   $M_R$ and the diagonal of $\Theta$.
5. Select $k$ as the index of the minimum element in $M_T$.
6. Take a step in direction $k$, that is, increment $m_k = m_k + 1$.
7. Update matrix $M_R$ by making
   $M_R = M_R + 2 \cdot column_k(\Theta)$.
8. If point $P$ is different from point $B$,
   go to Step 4, else end.
```

This algorithm continuously selects the next point for the discrete line as the one closest to the Euclidean line among all the possible movements.

5 THE CUBIC GRID CASE

In this section, we present an example of how this algorithm works for tracing a line in a cubic grid. We first express in vectorial terms the basic elements needed for tracing a line from point A to point B in a cubic grid. These elements are shown in Fig. 8. The cubic grid can be defined by a vector basis composed of vectors $\mathbf{H}_1 = (1, 0, 0)$, $\mathbf{H}_2 = (0, 1, 0)$, $\mathbf{H}_3 = (0, 0, 1)$.

5.1 Defining Connectivity

Here, we select 6-connectivity in 3D as shown in Fig. 9. In this connectivity, two grid points A and B are considered neighbors if their coordinates satisfy $|x_{b1} - x_{a1}| + |x_{b2} - x_{a2}| + |x_{b3} - x_{a3}| \leq 1$. The T set defining 6-connectivity is $T = \{\mathbf{H}_1, \mathbf{H}_2, \mathbf{H}_3, -\mathbf{H}_1, -\mathbf{H}_2, -\mathbf{H}_3\}$.

Let's consider the case of a line from $\mathbf{A} = (0, 0, 0)$ to $\mathbf{B} = (2, 3, 5)$. This line lies in the first octant, so the vector basis is directly the set

$$\{\mathbf{V}_1 = \mathbf{H}_1, \mathbf{V}_2 = \mathbf{H}_2, \mathbf{V}_3 = \mathbf{M}_3\}.$$

Vector $\mathbf{M}$ is equal to $2\mathbf{V}_1 + 3\mathbf{V}_2 + 5\mathbf{V}_3$, that is, the components are $x_1 = 2$, $x_2 = 3$, and $x_3 = 5$.

5.2 Projection on the Orthogonal Subspace

In the cubic grid, the computation for the reciprocal vector basis leads to the trivial result: $\mathbf{U}_1 = \mathbf{V}_1$, $\mathbf{U}_2 = \mathbf{V}_2$, and $\mathbf{U}_3 = \mathbf{V}_3$. Using vectors $\mathbf{V}_\mathrm{i}$, we can form the matrix M_v, which turns out to be the Identity matrix. The computation of matrix Θ from (33) is reduced to

$$\Theta = M_{sp}\left(M_{sp}^T M_{sp}\right)^{-1} M_{sp}^T.$$

For our particular numerical example, M_{sp} is

$$M_{sp} = \begin{bmatrix} x_3 & 0 \\ 0 & x_3 \\ -x_1 & -x_2 \end{bmatrix} = \begin{bmatrix} 5 & 0 \\ 0 & 5 \\ -2 & -3 \end{bmatrix}$$

and matrix Θ turns out to be

$$\Theta = \frac{1}{38}\begin{bmatrix} 34 & -6 & -10 \\ -6 & 29 & -15 \\ -10 & -15 & 13 \end{bmatrix}.$$

It is easy to verify that

$$[x_1 \quad x_2 \quad x_3]\Theta\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = [2 \quad 3 \quad 5]\Theta\begin{bmatrix} 2 \\ 3 \\ 5 \end{bmatrix} = 0,$$

which geometrically means that the projection of point B on the orthogonal subspace is superimposed onto the projection of point A.

5.3 Running the Algorithm

Let's now run the algorithm for this particular example. The following tables show the numerical values of the main elements involved in the line tracing process. For the purpose of the algorithm, we can ignore the fraction 1/38 in matrix Θ because we are only concerned with comparing values.

Table 1 shows the evolution of column matrix M_R. Its elements are initialized to zero. Table 2 shows the evolution of column matrix M_T, which is, at each stage, the value of M_R plus the diagonal of Θ. The index k of the minimum element of M_T is selected as the direction for taking the next step. Matrix M_T is updated by adding the column of Θ corresponding to the selected index. Table 3 shows the evolution of point P coordinates; this is the final output of

TABLE 1
Evolution of Column Matrix M_R

M_R	0	1	2	3	4	5	6	7	8	9
1	0	-20	-32	36	16	-4	-16	-36	32	20
2	0	-30	28	16	-14	-44	14	-16	-28	30
3	0	26	-4	-24	2	28	-2	24	4	-26
k	3	2	1	3	3	2	3	1	2	3

The three components are arranged vertically, their consecutive values are arranged from left to right. The last row contains the index that is selected as step direction at each iteration. At each iteration, the matrix is updated by adding the double of the kth column of Θ.

the algorithm, the sequence of points that compose the discrete line.

6 GENERAL CASE

The generalization of the line tracing procedure is presented here for any regular grid of dimension N.

6.1 Definition of the Grid

A regular grid of dimension N is completely characterized by a set of N linearly independent vectors $\{\mathbf{H_i}\}, i \in [1, N]$ and each vector $\mathbf{H_i} \in \mathbb{R}^N$. A grid point P is defined by its position vector $\mathbf{P} = \sum_{i=1}^{N} x_{pi}\mathbf{H_i}$, where coefficients x_{pi} are said to be the *components* of $\mathbf{P}$ in the vector basis $\{\mathbf{H_{i=1,N}}\}$ and satisfy $x_{pi} \in \mathbb{Z}$, $\forall i \in [1, N]$.

6.2 Selecting Connectivity

Connectivity is a *choice* to be taken prior to the line tracing process. It is defined by specifying the set T composed of all the vectors relating neighbors in the grid. This choice is not trivial at all, the topological properties of the traced line depending completely on the vectors in set T. It can be *recommended* to use connectivities defined by the grid's Voronoi regions: Two grid points are neighbors if their N-dimensional Voronoi regions share a common $(N-1)$ dimensional face [8], [10]. However, this is not a requirement of our algorithm. Of course, vectors in T are linear combinations of the vectors $\{\mathbf{H_i}\}$, $i \in [1, N]$.

To trace a line from a grid point A at position $\mathbf{A} = \sum_{i=1}^{N} x_{ai}\mathbf{H_i}$ to a grid point B at position $\mathbf{B} = \sum_{i=1}^{N} x_{bi}\mathbf{H_i}$, we have to find the sequence of grid point related by vectors in T going from point A to point B. The grid vector representing the line to be traced is $\mathbf{V_t} = \mathbf{B} - \mathbf{A}$.

TABLE 2
Evolution of Column Matrix M_T

M_T	0	1	2	3	4	5	6	7	8	9	10
1	34	14	2	70	50	30	18	-2	66	54	34
2	29	-1	57	45	15	-15	43	13	1	59	29
3	13	39	9	-11	15	41	11	37	17	-13	13
k	3	2	1	3	3	2	3	1	2	3	

The three components are arranged vertically, their consecutive values are arranged from left to right. At each step, it contains the sum of M_R *and the diagonal of matrix* Θ. *The last row contains the index that is selected as step direction at each iteration.*

TABLE 3
Evolution of Point *P*'s Position as Represented by Vector $\mathbb{M}$

M	0	1	2	3	4	5	6	7	8	9
1	0	0	1	1	1	1	1	2	2	2
2	0	1	1	1	1	2	2	2	3	3
3	1	1	1	2	3	3	4	4	4	5
k	3	2	1	3	3	2	3	1	2	3

The last row contains the index that is selected as step direction at each iteration.

6.3 Selecting the Optimal Vector Basis

The optimal vector basis depends on the discrete line to be traced. This problem is the generalization of the quadrant selection problem in the square grid or the octant selection problem in the cubic grid. The rule is to take a set of N vectors $\{\mathbf{V_i}\}$, $i \in [1, N]$, from the set T such that they form a vector basis for the grid. The vector $\mathbf{V_t}$ representing the line could be expressed in terms of these vectors with coefficients $x_i \geq 0, \forall i \in [1, N]$, as small as possible. Once the set of vectors $\{\mathbf{V_i}\}$ is determined, we can construct matrix M_v by using the vectors as columns.

6.4 Defining the Orthogonal Subspace

Once the vector basis $\{\mathbf{V_i}\}$, $i \in [1, N]$, is selected, the reciprocal vector basis can be obtained by forming the $N \times N$ matrix M_v with the N vectors $\mathbf{V_i}$ placed as columns. The matrix M_u is computed following (11). The columns of matrix M_u are the vectors $\{\mathbf{U_i}\}$, $i \in [1, N]$, of the reciprocal vector basis.

It should be noted that the procedure presented in Section 4.2.5 is not possible if vector $\mathbf{V_t}$ has null components. The solution in this case is to reduce the dimension N of space to obtain only nonnull components in $\mathbf{V_t}$. Then, we reconsider the problem in a lower dimensional grid. The simplest example for this case is to trace a 3D line which lies in the plane XY, which is really a 2D process.

The Euclidean line AB is a 1D object. The orthogonal subspace to the line is an $(N-1)$-D one. In order to define this orthogonal subspace, we need $(N-1)$ linearly independent vectors $\{\mathbf{V_{pi}}\}$, $i \in [1, N-1]$, orthogonal to the vector $\mathbf{V_t}$.

We renumber the vector basis $\mathbf{V_i}$ in order to obtain $\mathbf{V_t}$ components such that $x_i \leq x_j, \forall i < j$. In this way, the dominant direction is the one indicated by vector $\mathbf{V}_N$, the remaining $\mathbf{V_i}$, $i = 1, \ldots, (N-1)$, vectors indicate secondary directions. Vectors $\{\mathbf{U_{pi}}\}$, $i \in [1, N-1]$, can be defined by

$$\mathbf{U_{pi}} = x_N\mathbf{U_i} - x_i\mathbf{U_N}.$$

We can then find the generalized subspace projection matrix M_{sp} as the $(N-1) \times N$ matrix:

$$M_{sp} = \begin{bmatrix} x_N & 0 & \cdots & 0 \\ 0 & x_N & \cdots & 0 \\ 0 & 0 & \ddots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ -x_1 & -x_2 & \cdots & -x_{N-1} \end{bmatrix}.$$

The resulting vector basis $\{\mathbf{U_{pi}}\}$, $i \in [1, N-1]$ of the orthogonal subspace has a reciprocal vector basis $\{\mathbf{W_{pi}}\}$, $i \in [1, N-1]$. In the line tracing process, it is not really necessary to compute $\mathbf{W_{pi}}$ vectors and they are introduced here only to express the projections of grid vectors on the orthogonal subspace.

6.5 Projection of Vectors on the Orthogonal Subspace

Let $\mathbf{M}$ be the grid vector defined by $\mathbf{M} = \sum_{i=1}^{N} m_i \mathbf{V_i}$. Let $\mathbf{M_p}$ be the projection of $\mathbf{M}$ on the orthogonal subspace. It can be expressed in terms of the reciprocal vector basis of the orthogonal subspace as $\mathbf{M_p} = \sum_{i=1}^{N-1} m_{pi} \mathbf{W_{pi}}$. The coefficients m_{pi} are computed by $m_{pi} = \mathbf{M} \cdot \mathbf{U_{pi}}$, $i \in [1, N-1]$. It is easy to verify that the projection of vectors $\mathbf{V_i}$, $i \in [2, N]$, corresponding to secondary directions, is $x_N \mathbf{W_{pi}}$, while the projection of the dominant direction vector $\mathbf{V}_1$ is $-\sum_{i=1}^{N-1} x_{(i+1)} \mathbf{W_{pi}}$.

In practice, it is enough for us to compute directly the matrix Θ using (33), for which we only need the matrices Mv and M_{sp}.

6.6 The Generalized Algorithm

We present here the generalized algorithm that can be applied to any regular grid of any dimension.

```
1. Initialize current point $P$ at position $A$:
   $x_{pi} = x_{ai}$, $\forall i \in [1, N]$.
2. Compute the matrix $\Theta$.
3. Initialize column matrix $M_R$ to zero.
4. Compute the column matrix $M_T$ as the
   sum of $M_R$ and the diagonal of $\Theta$.
5. Select $k$ as the index of the minimum element in $M_T$th column o.
6. Take a step in direction $k$, that is, increment $m_k = m_k + 1$.
7. Update matrix $M_R$ by making
   $M_R = M_R + 2 \cdot column_k(\Theta)$.
8. If point $P$ is different from point $B$,
   go to Step 4, else end.
```

Hence, this algorithm traces the N dimensional line, using only sums.

A C++ implementation of this algorithm is freely available from http://www.cs.unc.edu/~ibanez/Papers/LineTracing/Code.

7 CONCLUSION

A general description of the problem of tracing discrete straight lines on grids of different dimensions has been presented in this paper. A particular emphasis has been done on the treatment of nonorthogonal grids.

A vectorial approach has been developed as a formalization of the line tracing procedure. The fact that grids are well-represented in terms of vector basis leads to a natural treatment of the topological problems embedded on grids.

The basic criterion of our algorithm for tracing the discrete approximation of the Euclidean line AB is to find at each step the point that is closest to the Euclidean line AB among all the possible points allowed by the connectivity of the grid.

An efficient matricial process has been stablished for computing the distance from grid points to the Euclidean line. It allows us to reuse computations in a recursive framework. Only floating point sums are required for finding the next point of the line at each step. This fact stands as the main reason for the high performance of the algorithm.

The present algorithm can be considered as a generalization of Bresenham's to higher dimensional and nonorthogonal grids.

ACKNOWLEDGMENTS

The authors acknowledge the complete and detailed comments made by the reviewers. They largely contributed to improving the content and presentation of this paper.

REFERENCES

[1] J.E. Bresenham, "Algorithm for Computer Control of a Digital Plotter," *IBM Systems J.,* vol. 4, pp. 25-30, 1965.

[2] A.E. Kaufman, "Volume Synthesis," *Proc. Sixth Int'l Workshop Discrete Geometry for Computer Imagery (DGCI 96),* pp. 87-100, Nov. 1996.

[3] A.S. Glassner, *Graphics Gems.* Academic Press, 1990.

[4] G. Sakas, M. Grimm, and A. Savapoulos, "Optimized Maximum Intensity Projection (MIP)," *Proc. Eurographics Workshop,* pp. 51-63, June 1995.

[5] J. Foley, A. van Dam, S. Feiner, and J. Hughes, *Computer Graphics: Principles and Practice.* New York: Addison Wesley, 1990.

[6] S. Matej and R.M. Lewit, "Efficient 3D Grids for Image Reconstruction Using Spherically-Symmetric Volume Elements," *IEEE Trans. Nuclear Science,* vol. 42, no. 4, pp. 1361-1370, Aug. 1995.

[7] S. Matej, G.T. Herman, and A. Vardi, "Binary Tomography on the Hexagonal Grid Using Gibbs Priors," *Int'l J. Imaging Systems Technology,* vol. 9, pp. 126-131, 1998.

[8] G.T. Herman, "Finitary 1-Simply Connected Digital Spaces," *Graphical Models and Image Processing,* vol. 60, no. 1, pp. 46-56, Jan. 1998.

[9] G.T. Herman and E. Zhao, "Jordan Surfaces in Simply Connected Digital Spaces," *J. Math. Imaging and Vision,* vol. 6, pp. 121-138, 1996.

[10] G.T. Herman, "Biomedical Applications of Digital Geometry," *Proc. Third IEEE-EMBS Int'l Summer School,* June 1998.

[11] J. Serra, *Image Analysis and Mathematical Morphology.* Academic Press, 1982.

[12] R. Ulichney, *Digital Halftoning.* Cambridge, Mass.: MIT Press, 1987.

[13] J.H. Conway and N.J.A. Sloane, *Sphere Packing Lattice and Groups,* second ed. Springer-Verlag, 1993.

[14] G.E. Martin, *Transformation Geometry, An Introduction to Symmetry.* Berlin: Springer-Verlag, 1982.

[15] D.E. Dudgeon and R.M. Mersereau, *Multidimensional Digital Signal Processing.* Englewood Cliffs, N.J.: Prentice-Hall, 1984.

[16] L. Ibáñez, C. Hamitouche, and C. Roux, "Determination of Discrete Sampling Grids with Optimal Topological and Spectral Properties," *Proc. Sixth Int'l Workshop Discrete Geometry for Computer Imagery (DGCI 96),* pp. 181-192, 1996.

[17] L. Ibáñez, C. Hamitouche, and C. Roux, "Ray-Tracing and 3-D Object Representation in the BCC and FCC Grids," *Proc. Seventh Int'l Workshop Discrete Geometry for Computer Imagery (DGCI 97),* pp. 235-241, Dec. 1997 (available from: http://www-iti.enst-bretagne.fr/~ibanez/papers/dgci97).

[18] H.S.M. Coxeter, *Introduction to Geometry,* second ed. John Wiley & Sons, 1969.

[19] D. Nogly and M. Schladt, "Digital Topology on Graphs," *Computer Vision and Image Understanding,* vol. 63, no. 2, pp. 394-396, Mar. 1996.

[20] J-P. Reveillès, "Géométrie discrete, calculs en nombres entiers et algorithmique," thèse de doctorat d'etat, Département d'Informatique, Université Louis Pasteur, Strasbourg, 1991.

[21] E. Andrè, "Cercles discrets et rotations discretes," thèse de doctorat, Université Luis Pasteur, Strasbourg, 1992.

[22] O. Figueiredo and J.-P. Reveillès, "A Contribution to 3D Digital Lines," *Proc. Fifth Int'l Workshop Discrete Geometry for Computer Imagery (DGCI 95),* pp. 187-189, Sept. 1995.

[23] E. Andres, P. Nehlig, and J. Françon, "Supercover of Straight Lines, Planes and Triangles," *Proc. Seventh Int'l Workshop Discrete Geometry for Computer Imagery (DGCI 97),* pp. 243-253, 1997.

[24] J. Françon, J.-M. Schramm, and M. Tajine, "Recognizing Arithmetic Straight Lines and Planes," *Proc. Sixth Int'l Workshop Discrete Geometry for Computer Imagery (DGC I96),* pp. 141-150, 1996.

[25] C.E. Kim, "Three-Dimensional Digital Line Segments," *IEEE Trans. Pattern Analysis and Machine Intelligence,* vol. 5, no. 2, 1983.

[26] I. Stojmenovic and R. Tosic, "Digitization Schemes and the Recognition of Digital Straight Lines, Hyperplanes, and Flats in Arbitrary Dimensions," *Contemporary Math.,* vol. 119, pp. 197-212, 1991.

[27] A. Rosenfeld, "Digital Straight Line Segments," *IEEE Trans. Computers,* vol. 23, pp. 1264-1269, 1974.

[28] A. Kaufman, "An Algorithm for 3D Scan Conversion of Parametric Curves, Surfaces and Volumes," *Computer Graphics,* vol. 21, no. 4, pp. 171-179, 1987.

[29] K. Voss, *Discrete Images, Objects, and Functions in* Z^n. Springer-Verlag, 1993.

[30] E. Andrès and J.-P. Reveillès, "Discrete Ray-Casting," *Proc. Eighth Int'l Conf. Discrete Geometry for Computer Imagery (DGCI '99),* pp. 435-446, Mar. 1999.

[31] J. Amantides and A. Woo, "A Fast Voxel Traversal Algorithm for Ray Tracing," *Proc. EUROGRAPHICS '87,* pp. 3-10, 1987.

[32] M. Slater, "Tracing a Ray through Uniformly Subdivided *n*-Dimensional Space," *The Visual Computer,* vol. 9, pp. 39-46, 1992.

[33] P. Nehlig and C. Montani, "A Discrete Template Based Plane Casting Algorithm for Volume Viewing," *Proc. Fifth Int'l Workshop Discrete Geometry for Computer Imagery (DGCI 96),* pp. 71-79, Sept. 1995.