EFFICIENT RASTER-GRAPHING OF BIVARIATE FUNCTIONS
BY INCREMENTAL METHODS

Wolfgang G. Kramarczyk
Max-Planck-Institut für Ernährungsphysiologie
Rheinlanddamm 201
D-4600 Dortmund 1
Federal Republic of Germany

1. Introduction

The graph of a bivariate function \( z = f(x, y) \) is a surface in a three­
dimensional space. Suppose that we have ascribed discrete intensities
or pseudo-colors \( i = 1, 2, \ldots, n_{\text{colors}} \) to the graph in such a manner that

\[
i(x, y) = 1 + \left\lceil \frac{f(x, y) - b}{\Delta z} \right\rceil,
\]

where \( \lceil \cdot \rceil \) means the nearest integer function, \( \Delta z \) is a positive dis­
cretization step and \( b \) is a lower bound of \( f(x, y) \) on a rectangle with
opposite corners at \( (x_{\min}, y_{\min}) \) and \( (x_{\max}, y_{\max}) \). \( i = 0 \) is reserved for
the background color. Suppose further that we have sampled these
intensities on a grid \( (x_k, y_l) \), where

\[
x_k = x_{\min} + k \cdot \Delta x \quad , \quad k = 0, 1, \ldots, n_{\text{cols}} - 1,
\]

\[
y_l = y_{\min} + l \cdot \Delta y \quad , \quad l = 0, 1, \ldots, n_{\text{rows}} - 1,
\]

and stored them in a two-dimensional array. Such an array may be
considered to be a raster-image of the graph viewed from the top.
Let us call it the source image.

Usually, the graph's shape can be better visualised, when viewed
from a different direction than from the top. This paper deals with
the generation of projected raster-images, showing the graph from an
arbitrary direction specified by the user, without that parts, which
are hidden. Diffused illumination is assumed, so that the intensity
(or pseudo-color) of any graph element appears unaffected, when
observed from an other viewpoint. If a straight line connects two
points on a graph, the intensity varies linearly along that line.

We are concerned with four types of projected images:
- dot images (unconnected vertices),
- profile images (adjacent vertices connected in either x or y direction),
- crosshatched images (adjacent vertices connected in both directions),
- rendered images (crosshatched images rendered by linear interpolation of intensities).

The development of an efficient algorithm for these types of visual surface representation has been stimulated by the research work done on pattern formation in thin layers of chemically reactive solutions. All four types are useful in the two-dimensional spectrophotometry of such planar patterns, where images recorded by a computerized video equipment with a high spatial and temporal resolution are processed [1-2]. The source images encountered in these applications have hundreds of thousands pixels and the raster resolution of the final graphics output is 1280 x 1024 pixels x 8 bits per pixel.

To achieve reasonable processing times and to avoid propagation of rounding errors, our algorithm operates in the image plane and uses integer incremental methods wherever possible. No intersection points are computed. The floating-point processor is used only at the initialization stage.

The hidden line, resp. hidden surface problem involved in displaying projected graphs of bivariate functions is largely simplified by the geometric fact, that for any given relative position of the observer with respect to the graph, all facets $F_1, \ldots, F_n$ of the viewed graph can be enumerated in such a manner that $F_i$ cannot be obscured by $F_j$ if $i < j$, [3]. Moreover, one can explicitly indicate such obscured orders which are valid for any function. In raster graphics there are two strategies possible to make use of this fact.

According to the first strategy, which may be called the painter's approach, the facets are processed in reverse obscuration order, that means from rear to front. As each facet is processed, it is filled with uniform or varying pixel values. Thus the obscured portion of previously displayed facets are overwritten by the later ones. This approach is appropriate for any kind of projection and is easy to implement. The shortcoming is that even totally obscured facets, which do not appear on the resulting image, must be processed as completely as the partially or totally visible ones. This may be time-consuming. The task of filling facets, when displaying crosshatched, profile or dot surfaces is a burden for the host CPU, unless