An Adaptive Algorithm for Spatial Greyscale

ROBERT W. FLOYD AND LOUIS STEINBERG

ABSTRACT—This paper describes Error Diffusion, a greyscale algorithm for bi-level devices that provides high resolution, both spatial and dynamic, and less distracting texture than competing methods.

We assume that each picture element (pel) has a desired brightness between 0 and 1. The error diffusion algorithm scans the pels row by row. If the desired brightness for a pel is less than .5, we make the pel dark, otherwise light. Unless the desired brightness was exactly 0 or 1, we have introduced some error into the picture, that is, the picture is too light or too dark. To correct for this error, we modify the desired brightnesses of several neighboring (and yet to be scanned) pels.

We will show the result of applying error diffusion both to test patterns and to real scenes. Error diffusion provides higher spatial resolution at low contrast than current alternatives, at the cost of more memory and processing.

I. INTRODUCTION

It would be useful to be able to display continuous tone pictures on bi-level devices. This generally requires a spatial greyscale, that is, making more of the picture elements bright in bright areas of the picture, and fewer bright in dark areas of the picture. The problem is to design an algorithm to select which picture elements (pels) are to be light and which dark. This paper describes an algorithm that provides both high spatial and dynamic resolution, and freedom from unwanted textures such as Moire patterns. It provides higher spatial resolution at low contrasts than ordered dither[2] does, but it requires more memory and more processing than ordered dither.

II. CONSTRAINTS

We assume the constraints of a device like an AC plasma panel or a dot matrix printer, which has a fixed array of picture elements (pels), each of which must be in one of two states, bright or dark. We assume that time-division techniques are not available, as is the case with most such devices. Finally, we assume that the picture we desire to display is available with at least the spatial resolution of the device we wish to display it on. This rules out the straightforward technique of assigning a block of pels to each point of the picture and, for instance, lighting every fourth point of the block for a brightness of one fourth, since such a technique leads to an undesirable loss in resolution.

For simplicity of discussion, we will assume that for each pel we have a desired brightness normalized to a number between 0 (darkest) and 1 (brightest).

III. ERROR DIFFUSION

Our algorithm, error diffusion, is an adaptive algorithm, one in which the processing of one pel is dependent on the result of processing other pels.

The algorithm can be explained by the following line of thought: Consider a single pel. It can have an actual brightness of either 0 (dark) or 1 (bright), although the desired brightness at that point can be anywhere in between. Thus, a pel can introduce an error between +1 and -1 to the picture. For instance, if the desired brightness is $\frac{1}{4}$, then we can leave the pel dark and introduce an error of $-\frac{1}{4}$ (the picture is too dark by $\frac{1}{4}$ of a pel), or make the pel bright and introduce an error of $+\frac{3}{4}$ (the picture is too bright by $\frac{3}{4}$ of a pel). However, the errors of neighboring pels may cancel each other out, so that the total brightness of the several pels taken together is close to the total desired brightness. Thus, if we have decided that the pel with desired brightness $\frac{1}{4}$ is to be dark, then we have an error of $-\frac{1}{4}$, and would like neighbors of that pel to have an error of $+\frac{1}{4}$, to cancel out. Suppose we decide to have all of that error canceled by one of the neighbors. Then, we want that neighbor to be $\frac{1}{4}$ brighter than the original desired brightness. That is, we have modified the desired brightness of the neighbor. Now we repeat the process with the neighbor. If its modified desired brightness is above $\frac{1}{2}$ then light the neighbor, otherwise make it dark, and in either case modify the desired brightness of a neighbor of the neighbor to cancel the error introduced. The process can be thought of as diffusing the error from one pel among the neighboring pels, thus suggesting the name for the algorithm.

The algorithm actually modifies the desired brightness of several neighbors in order to have an effect in both the x and the y directions and to reduce the amount of texture introduced by the algorithm. The algorithm scans through the picture line by line. For each pel, it decides whether to make it dark or light, then for each neighbor that has not yet been decided, it modifies the desired brightness to compensate for a part of the error. At one point, the algorithm might be in the state shown in Figure 1.

```
       .
       .
       .
   ... X X X X X X X X X X X
       X X X X X X X X X X X
       X X X X P A O O O O O
       O O K D C B O O O O O
       O O O O O O O O O O O ...

       A: 7/16 of the error
       B: 1/16 of the error
       C: 5/16 of the error
       D: 3/16 of the error
```

Fig. 1 Parameters to the algorithm

X's are pels that have already been determined to be dark or light. P is the pel we are processing, and A, B, C, and D are its neighbors that have not yet been determined. The O's and K are the other pels that have not yet been determined. The error from P is apportioned to pels A - D as shown in the figure. These parameters were found mostly by trial and error, guided by the desire to have a region of desired density .5 come out as a checkerboard pattern. Other parameters are possible, but they produce more texture in the output picture. Four neighbors were chosen because that seems to be the smallest number of neighbors that gives good results. (There may, however, be some smaller set of neighbors that works, as we have not investigated this extensively.)

This method is similar to one described by Schroeder[3] for generating halftone pictures on a microfilm recorder that allowed a few brightness levels. The major difference is that Schroeder's method computes the error at a pel by comparing the actual brightness at the pel with the original desired brightness, rather than the modified desired brightness. In regions of desired brightness close to 0 (very dark regions), the error from any one pel's neighbors is too small to change the pel from dark to light, so all are dark. With error diffusion, the error is computed from the modified desired brightness, which includes the effect of error propagated into the pel from previous pels, and so the error builds up, eventually causing a pel to be light. A similar effect occurs in very bright regions, where Schroeder's method makes all pels bright, while error diffusion makes some dark. Because of these effects, Schroeder's method requires that a large number of neighbors be effected by a given pel, since the more neighbors the darker it can make a region without making it all dark.

It should perhaps also be noted that error diffusion has an obvious extension to devices that allow several brightness levels at each pel instead of only two. Simply choose the available brightness level that is the closest to the modified desired brightness, and compute the error accordingly. The rest of the method is the same.

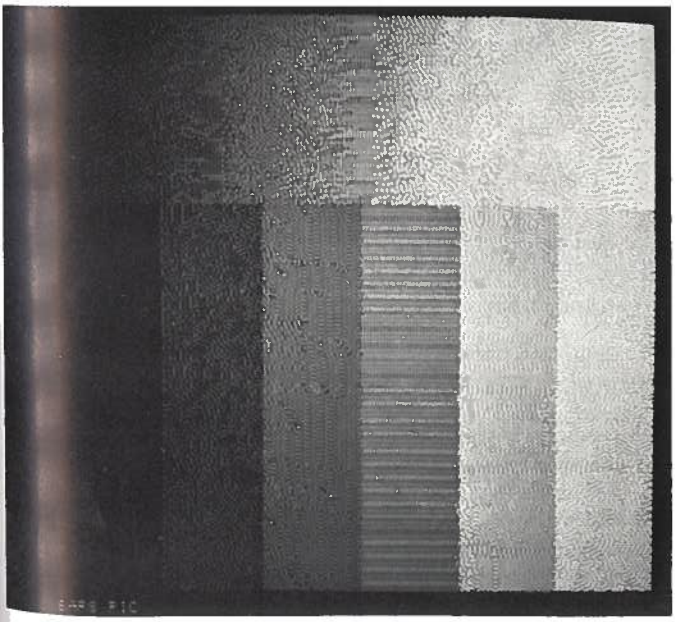

Fig. 2 Test pattern, 360 x 400 pels

IV. RESULTS

Figure 2 is the result of applying error diffusion to a test pattern. The pattern consists of bars with desired brightness of 0/6, 1/6, . . . , 5/6, and a band of constant slope from desired brightness of 0 to desired brightness of 1. Note that the result has some unwanted texture, especially where the texture changes around desired brightnesses that are multiples of 1/4 and 1/3. This seems to be caused by the fact that at these desired brightnesses, there are regular, stable patterns. When the desired brightness changes sufficiently, the pattern suddenly breaks up, leading to a change in texture. Figure 3 is based on calculated brightnesses for a sphere on a background of brightness 1/2. The algorithm scanned the lines of this picture from right to left, starting at the bottom. Because the algorithm is adaptive, the way the background is represented is affected by the sphere if the algorithm has scanned through the sphere when it reaches the background. These problems, the visibility of texture shift and the "history" effect, are the major problems with our algorithm. However, as can be seen from Figures 4 and 5, they have little effect on real images. These were produced by processing TV camera images that had been digitized to 5 and 6 bits per point (Figures 4 and 5 respectively). In Figure 5 each point in the data was first expanded to four points to increase the size of the image.)

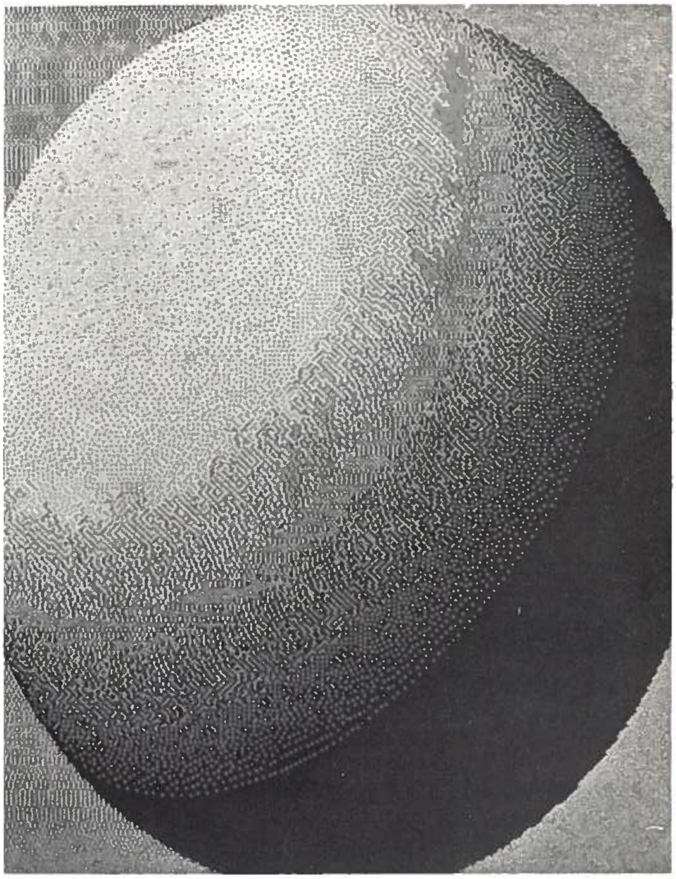

Fig. 3 Calculated sphere, 360 x 400 pels

In real scenes the texture shift problem shows up in what we call the "grain boundary" problem, in analogy of crystal grain boundaries. Consider some molten metal as it crystalizes. Crystals start to grow at many points. When two crystals grow into each other, they cannot merge, because their patterns of atoms (their crystal structures) have different orientations. The region where two crystals meet is called a grain boundary. When the error diffusion method is working on a region of fairly uniform desired brightness it can be thought of as a crystal growing process, where each successive line has bright pels laid down in a pattern determined by the influence of those of the previous line. The pattern can start out different in different parts of the region, and where the different patterns meet, the change in texture is often quite visible. This is most apparent in Figure 5, at the top, both just above the central post of the distributor and at the extreme right.

Fig. 4 TV image, 288 x 216 pels

Fig. 5 A Distributor, TV image 540 x 410 pels

Another problem resulting from the history effect is what we call the "knight's move" problem. As the algorithm advances across a region of low desired brightness, the error introduced by not lighting pels gradually builds up, until a pel is lit and the error reduced. Unfortunately, because error is diffused, no matter what the state of things was at the boundary of the region, as the algorithm advances, pels along a line at some orientation tend to have the same modified desired brightness. When a pel is lit, its near neighbors are made less likely to be lit, but as Figure 1 shows there is neither direct nor indirect feedback to the pel a knight's move away (marked K). Thus, it too tends to be lit, and the algorithm tends to give lines of lit points at this knight's move orientation. This is visible in Figure 5 in the darker areas.

V. CONCLUSION

Despite these problems, error diffusion gives better results than other available methods. In particular, it is better than ordered dither[2] in that it has better resolution in areas of low contrast, and has less distracting texture. Figure 6 is the same picture as Figure 4, processed with ordered

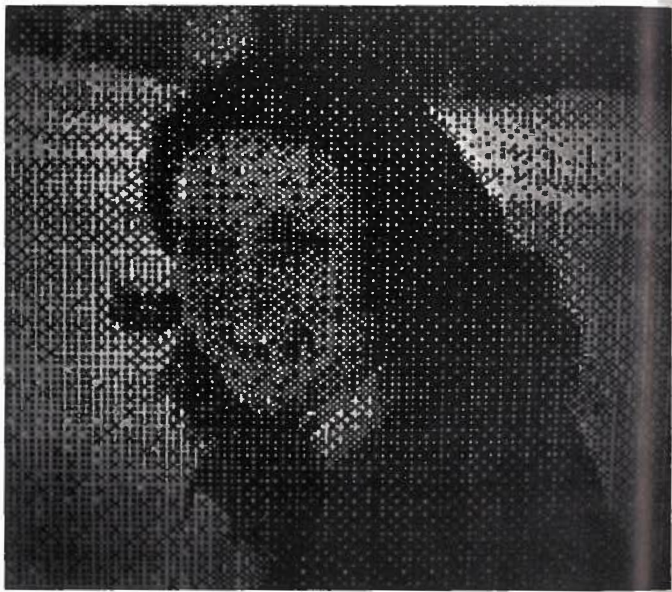

Fig. 6 Ordered Dither, using 8x8 matrix, on same data as Figure 4.

dither method. (Note especially the eyes and mouth as areas of low contrast.) On the other hand, error diffusion does take more memory and processing than ordered dither. (See [4] for a further comparison of error diffusion with ordered dither and other methods.)

REFERENCES

[1] This research was supported in part by NSF grant DCR72-03663 A03.

[2] C.N. Judice, J.F. Jarvis, and W.H. Ninke, "Using Ordered Dither to Display Continuous Tone Pictures on an AC Plasma Panel," Proc. Society for Information Display, 4th Quarter, 1974.

[3] M.R. Schroeder, IEEE Spectrum 6, 1969, 66-78.

[4] J.F. Jarvis, C.N. Judice, and W.H. Ninke, "A Survey of Techniques for the Display of Continuous Tone Pictures on Bi-level Displays," Computer Graphics and Image Processing, 5, 13-40 (1976).