

In decision block 116, controller 34 checks to see if an empty defect entry is available in the defect list. If there is, the process flows to block 117, where a new defect entry is created in the defect list. Then the process flows to decision block 118. If an empty entry is not available, the process flows directly from block 116 to block 118.

At decision block 118, defect detection system 14 checks to see if any further pixels are available for processing. If not, the process terminates. If more pixels are available, the process flows to block 120, where the next pixel is retrieved and the flow continues with decision block 108.

The process flows from decision block 110 to block 122 if the pixel's location is in the defect list. At block 122, controller 34 checks to see if the maximum count of dark pixels has been reached for that pixel location. If not, the pixel count for that location is incremented at block 124, a dark pixel is emitted at block 126, and the process flows to block 118. Otherwise, if the maximum count has been reached, the process flows to block 128, where a defect pixel is emitted, and the process flows to block 118.

A defect pixel is emitted as the logical result of controller 34 placing that pixel's location at its output as one of defect locations 42. Defect locations 42 are output either as they occur or in a block to the process of apparatus accepting the output of this process.

If at block 108, the pixel color is found to be a light pixel, the process flows to block 130, where a light pixel is emitted. The process then flows to block 132.

At block 132, controller 34 checks if the position of the light pixel is in the defect list. If it is not, the process flows to block 118, but if it is, the process flows to block 134 and then to block 118. At block 134, the entry for that light pixel's location is deleted from the defect list.

In summary, the flowchart shown in FIG. 4 illustrate the process by which pixels of an input document are processed into an output document where the input document contains light and dark pixels, and the output document contains light and dark pixels, as well as pixels which are marked as defective. Although the defective pixels might be separable into defective dark and defective light pixels, generally image restoration system 16 is able to operate to correct a defective pixel without knowing its color. Of course, if only continuously black pixels are flagged as defects, defect pixels will always have the same color, namely black. As FIG. 4 shows, the input document is also used to effect how later documents are processed.

FIG. 5 is an actual display of an accumulation of documents scanned using a defective scanning mechanism. Each pixel in FIG. 5 has a gray level corresponding to the number of documents in the accumulation which have a black pixel in that location. Most of the documents contained an artifact of a scratch on the scanner platen, which is clearly distinguishable from the text of the documents as a dark, diagonal line. Of course, the contrast represented in the defect list is even greater than that shown in FIG. 5, because, while the black pixels in FIG. 5 would have corresponding entries in the defect list with high counts, many of the gray pixels in FIG. 5 would have been deleted from the defect list, when the documents failed to have black pixels on consecutive documents.

FIG. 6 shows how scanned lists such as might be used for the defect list are typically updated. FIG. 6 shows an even scanned FIFO (FIFO: First In, First Out Register), an odd scanned FIFO 202, and a multiplexer 204. The wiring in FIG. 6 is illustrated for an even scan. As the name suggests, the role of the FIFO's switch each scan. During an even

scan, list elements are read from even FIFO 200 and written to odd FIFO 202, and during the following odd scan, list elements are read from odd FIFO 202 and written to even FIFO 200. The list elements are read out of one FIFO in order and written to the other FIFO, also in order, with some records deleted from the stream and new records interspersed in the stream by multiplexer 204. The effect of this is that the list is always in order and packed into one or the other FIFO. This effect could be had with only a single FIFO, but only if the space freed by deleted entries was completely necessary and sufficient for new records. Since this is generally never possible, two FIFO's are used.

FIG. 7 shows a more efficient use of memory to accomplish the same effect, namely to add, delete, and pack entries in a list in a single pass through the list. FIG. 7 shows memory array 250, such as might be maintained within buffer 40 (see FIG. 2). This memory array 250 is pointed to by three pointers stored in cursor registers 38. These three pointers are a read pointer (RP), a start of new list pointer (NEWSOP), and a right pointer (WP). The elements of array 250 between RP and NEWSOP and the new list is between NEWSOP and WP. The pointers are such that array 250 is a circular array, i.e., when a pointer is incremented past the bottom of array 250 it points to the first entry at the top of array 250.

During a scan, list entries are read from the array location pointed to by RP and an entry is written to the location pointed to by WP, then both RP and WP are incremented. Adding a new entry is effected by writing an entry to the entry pointed to by WP, but not incrementing RP. Deleting an entry is effected by incrementing RP without writing an entry to the entry pointed to by WP and not incrementing WP. In this way, the list always remains packed. When RP becomes equal to NEWSOP, that is a signal (sent to controller 34 in defect detection system 14) that the end of the list has been reached. At that point, no entries are read from the old data since there are no remaining entries to be read, and entries are only added, until the end of a scanned image is reached. This allows for a packed list to be maintained in half the space as that shown in FIG. 6.

FIG. 3 and its accompanying description presented the concept of a gray scale map for defect detection. The method described in FIG. 4 achieves this using very little memory, but only detects dark pixels (or more generally, the defects which are the same color as the less common pixel color). Using a frame buffer to accumulate the abnormal runs of both black and white pixels, both colors of defects can be detected. Of course, storage of a frame buffer requires considerable memory. For example, even personal computers today have screen resolutions on the order of 1,000 rows by 1,000 columns, or a million pixels. Thus, for each page to be scanned, another million pixels worth of data needs to be accumulated. This results in an enormous number of pixels when even a small number of pages is scanned. Because many bits are typically required for each pixel, the amount of memory required in such a system can become enormous quickly.

In a frame buffer embodiment of this invention, the need for such an enormous frame buffer is avoided by performing the accumulation of document frames in stages and compression of the frame buffer. Because of the way the frame buffers are accumulated, large compression ratios are easily achieved. This is illustrated by FIGS. 8 and 9.

FIG. 8 shows the process of accumulating document frames for detecting dark errors, where dark pixels are expected to be a minority of the pixels in an frame. As each