Use of images in commercial and office systems

by P. J. Somerville

This paper examines some of the simpler processing techniques that may usefully be performed on bi-level (two-tone) images by a competent commercial applications programmer if a few basic (albeit complex internally) tools are provided. Such processes include storage, indexing, changing resolution, rotating, trimming, and then display and printing. These processes can provide the facilities for an enterprise to incorporate images and image data into its office systems and into its main line-of-business data processing applications.

Images, or noncoded information (NCI), can be captured in various ways and with equipment of varying complexity. The captured NCI data may be organized in different ways and contain a range of detail. Examples of these categories include (1) capture—TV camera, light-sensitive diodes, and charge-coupled device (CCD) arrays, (2) representation—analogue and digital, (3) organization—canonical, and eight-bit parallel, (4) detail—halftone, gray-scale (continuous tone), and color, and (5) volume—raw data and compressed data.

Many of the currently used image processing techniques have originated in application areas where the considerable processing required has been justified for social reasons, such as in the processing of Landsat images for earth resource studies[1,2] or medical X-rays. Others have originated where the processing cost is relatively insignificant when compared to the capture cost, as in processing photographs of the rings of Saturn that were taken from a spacecraft. From this area much work has been done in developing techniques for processing noncoded information, or images, including image enhancement, continuous tone image handling, feature recognition, content decoding, and incorporation with other data types.

At the low end of the processing spectrum are the techniques associated with handling analogue and digital facsimile transmissions of bi-level (black/white) image data. These include the efficient compression of image data for effective use of the limited bandwidth available with facsimile transmission.

This paper will consider the middle area where the majority of applications for image processing may fall with the advent of cheap processing power and economical bulk storage. This area addresses the typical requirements of a commercial enterprise for image applications. In general terms these requirements include (1) the ability to create, capture, view, and print images, (2) communicating these images both as "paper" (facsimile) and "live" (videoconferencing),[3] (3) modifying images in content, shape, and size, (4) incorporating image data with other forms of information (text, data, voice), and (5) analyzing documents and encoding the text and data content.

The commercial applications that generate these requirements include

- Office systems—external correspondence and paperwork (that is really data but happens to be in image form) and nontext or nondata information (real image data)

- Line-of-business systems—application and claim forms for insurance, signature verification for banking, engineering specifications with drawings for manufacturing, and catalogs for distribution
- Printing and publishing systems

This paper documents a project that addressed the handling of medium-resolution, unenhanced images with commercially available IBM hardware (scanner, processors, displays, and printers). Problems and their solutions are discussed in terms of the IBM Scanmaster I, the System/370 architecture processors, the IBM 3270 family of display devices, and IBM printers including the 3800 Model 3, the 6670 Information Distributor, and the 4250.

The software environment utilized was the Customer Information Control System/Virtual Storage (CICS/VS) and Distributed Office Support System/Professional Support (DISOSS/PS) for capture, storage, and indexing of two-tone images and the Graphical Data Display Manager (GDDM) for the display and presentation of images.

The specific problems tackled and for which solutions were developed or found were

- Image capture, storage, and retrieval
- Decompression of images scanned and compressed by the Scanmaster I
- Understanding the compromises required to match medium-resolution images to low-resolution displays
- Providing a user interface for displaying images that is similar to the interface for displaying text
- Making images and segments of images available to application programmers as just another form of data
- Provision of basic software to commercial users to allow further development of commercial image-handling techniques

Other areas that were contemplated and discussed during the project included image annotation with another image and with data in image form, image creation, and image recognition.

Image capture, storage, and indexing

Images were captured for this work by using the Scanmaster I (8815).[4] The Scanmaster consists of three major elements: a scanner, a printer, and a control unit.

The scanner has a roller feed for the image document. An optical system focuses a single 8.5-inch (216-mm) scan line onto a 1728-bit CCD array at 203 bits to the inch (8 per mm). Scans are performed at the rate of 196 or 98 per inch (7.7 or 3.85 per mm) vertically.

The printer is a roll feed electrostatic stylus printer that prints at 203 pels per inch horizontally and 196 pels per inch vertically and thus matches the geometry of the scanner.

The control unit provides a number of functions. In addition to controlling the scanner and printer, it provides data compression and decompression and a Systems Network Architecture (SNA) advanced program-to-program communication (APPC) logical unit appearance.[5] It also converts textual data received across the communication link to a rastered prestige elite 12-pitch font to allow the printer component to print text. Some of the control functions are requested by the operator either by using an optically mark-sensed coversheet that precedes the document in the hopper or by using a numeric keypad and the switches on the control panel.

The Scanmaster hardware depends on an SNA APPC environment in a host CPU. This environment is provided by the Advanced Communications Function/Virtual Telecommunications Access Method (ACF/VTAM) Version 2 with the Advanced Communications Function/Network Control Program (ACF/NCP) in an IBM 3705 Communications Controller. The programming required to support the APPC is incorporated in CICS/VS Release 1.6. The application code required to interpret and to act upon the functions requested by the Scanmaster operator is included in DISOSS Version 3.[6] The operator may distribute the image to a given user and file the image document with a unique reference number.

When an image document is filed or distributed to another user who is defined to the DISOSS system and registered to use DISOSS/PS,[7] this user may retrieve or receive the document and then may add coded descriptive information to allow contextual rather than numeric filing. This information includes author of the document, recipients, data, subject, and a number of keywords or search terms. Codes to limit access by authorized users may also be added. The image document, once filed, may be retrieved by any user with the appropriate access codes, together with text documents with a retrieval request, including the original filing terms.

The application support with these programs is restricted to the filing, retrieval, and distribution of image documents with one or more full-size pages as originally scanned by Scanmaster. To permit manipulation of these pages and their display (rather than printing) and to allow a similar set of functions on subimages extracted from these pages, further processing is required. These processes are described in the following sections.

Image decompression

The whole page images that are stored in the host library are stored in their compressed format together with Document Content Architecture (DCA) image descriptors that describe their characteristics. When originally scanned by the scanner element of the Scanmaster, approximately half a million bytes of data per page are generated. Using hardware that implements a proprietary data compression algorithm, we obtain a compression factor of between 10 and 50, depending on the complexity of the image on the page. Thus, between 10 and 50 kilobytes of data may typically represent a page.

The required decompression routines had been written with the Series/1 as a target.

Although this compression has considerable advantages for the purposes of image transmission and storage, other image processing functions cannot be performed until decompression is performed adjacent to those functions. For example, to view the image on a display unit that does not have decompression hardware requires that a software decompression technique must be developed.

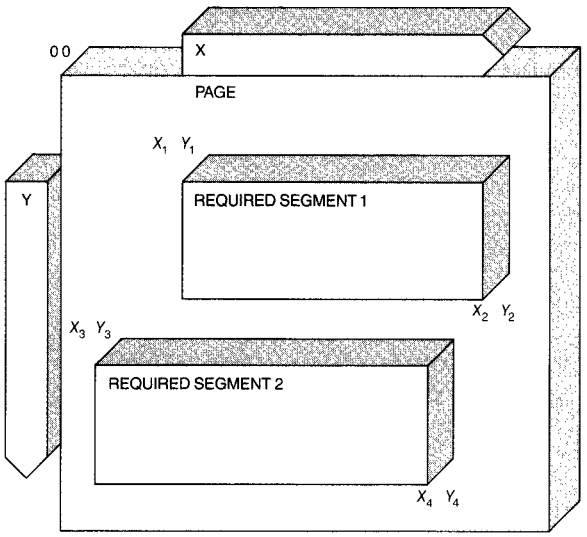
Various programmers had written System/370 code to perform this function, but the end result was that the decompression of a standard A4 page (and recreation of the four million bits involved) was taking from 10 million to 150 million instructions to achieve. For interactive work in a commercial environment, both the cost and the response time were unacceptable.

At the IBM Research Division in Yorktown Heights, New York, fast algorithms had been developed for the Series/1-based videoconferencing system[3] using a development tool based on the Virtual Machine Facility/Conversational Monitor System (VM/CMS) which would execute in a VM environment and could then be cross-compiled for a number of target machines. The required decompression routines had been written with the Series/1 as a target, but it proved possible to move these routines first to a VM/CMS environment for optimization for a System/370 instruction set and then to create object code that was operating-system-independent and that could be invoked from a PL/I program. After various iterations of optimizations achieved by experience gained on other target machines, decompression of an A4 page is now typically performed in one to three million System/370 instructions, depending on the complexity of the page. Since the remainder of the project depended on the viability of a fast decompression, the author is indebted to the Image Technologies Department in the Research Division for allowing unrestricted access to the code.

This speed is fast enough to allow interactive use of the function, and indeed, it is preferable, when main storage is a constraint, to read the compressed page from storage as required and then to decompress what is needed, rather than to write out a full decompressed page (half a megabyte) and then to read the decompressed data that is actually to be worked on.

When decompressed, the image is represented as one bit per picture element (pel) with 1728 bits (216 bytes) per horizontal scan line. The first bit is the top left pel, and the last bit is the rightmost bit of the last scan line of the page. This representation is known as the canonical form. For simple image manipulation, this form is probably the easiest for a "commercial" programmer to use. There is a rectangular array of bits that map directly to the rectangular image they represent. (A 1 bit is black, and a 0 bit is white.) For complex image manipulation, some representation other than canonical may prove useful (e.g., run length encoding[8]). Control information must be associated with this image to define such attributes as resolution and the horizontal and vertical dimensions as a minimum. Decompression of the data stream has been implemented as a two-step process. The first step generates the end-point coor-

Figure 1 Selecting segments from an image page



dinates of each color change (black/white), and the second generates the appropriate number of 1 or 0 bits for the output data stream. A characteristic of the encoding technique used is that decoding must start at the beginning of the image and then continue until a sufficient amount, in a vertical, down-the-page sense, of the image has been presented to the user application in a canonical form. However, if only, say, a segment (x_1 y_1, x_2 y_2) of the image is required (Figure 1), the most efficient way of obtaining this segment from the compressed form is as follows.

Decompress the first $y_1 - 1$ rows using the first step only (generate end points). Decompress rows y_1 through y_2 using the first step, and then use the second step (generate the output bits) merely for pels x_1 through x_2 in each row. At this point stop the process but maintain sufficient control information so that if the next requirement is to obtain another segment further on in the page (x_3 y_3, x_4 y_4), the first step of decompression may start at row $y_2 + 1$, thus saving the processing of rows 1 through y_2.

Because of the number of bits to be processed for an image, a considerable amount of processing is to be performed for decompression. Depending on the complexity of the document (the number of changes from black to white), the number of System/370 instructions required to decompress an A4 page is between one and three million. However, the potential storage savings and the savings in the operating system input/output processing realized by using the compressed rather than the decompressed form can largely offset the processing cost of decompression.

Compromises on resolution

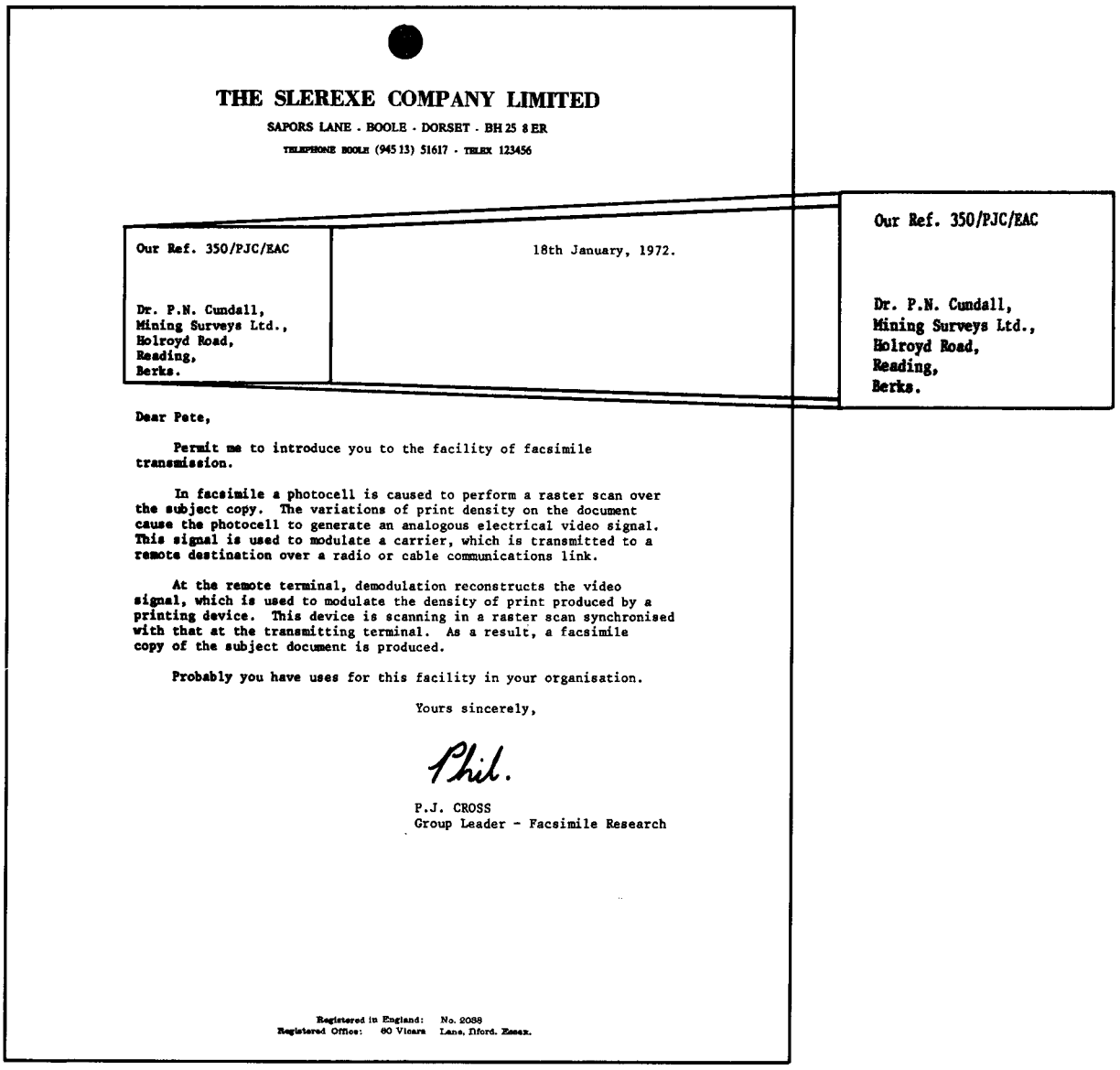
As already discussed, the resolution of the Scanmaster input data is 203 and 196 pels per inch in the horizontal and vertical directions, respectively. To within ±2 percent it has "square" pels. If this image is to be sent to a device such as a display unit that has less resolution (typically, a display device of the IBM 3270 Information Display System has around 70 pels per inch), two extreme options may be taken when attempting to display an image. These options are illustrated in Figures 2 and 3. The page shown in Figure 2 is one of the standard images used for evaluation of medium-resolution bi-level image capability. This illustration shows the pel dimensions of an A4 page scanned by Scanmaster at fine scan (203 and 196 pels per inch). It also shows the potential pel dimensions of an IBM 3279 Color Display Station.

The term used to denote the amount of detail present has been called "resolution."

With the original superimposed on the display, it is not difficult to see that only a small part of the original scanned data may be displayed on a 3279. Also, because the pels are bigger (and not square) on this device, the image that is displayed is magnified (and distorted) by a factor of approximately three in the horizontal direction and by four vertically.

The first of the extreme options is to display merely what will fit at the original level of detail (as in Figure 2), i.e., about one fourteenth of the area of the page. For most pages, the lack of context will mean that this is not really usable except for examining fine

Figure 2 Direct bit map—scanner to display; original on left



THE SLEREXE COMPANY LIMITED

SAPORS LANE · BOOLE · DORSET · BH25 8ER

TELEPHONE BOOLE (945 13) 51617 · TELEX 123456

Our Ref. 350/PJC/EAC

18th January, 1972.

Dr. P.N. Cundall,
Mining Surveys Ltd.,
Holroyd Road,
Reading,
Berks.

Our Ref. 350/PJC/EAC

Dr. P.N. Cundall,
Mining Surveys Ltd.,
Holroyd Road,
Reading,
Berks.

Dear Pete,

Permit me to introduce you to the facility of facsimile transmission.

In facsimile a photocell is caused to perform a raster scan over the subject copy. The variations of print density on the document cause the photocell to generate an analogous electrical video signal. This signal is used to modulate a carrier, which is transmitted to a remote destination over a radio or cable communications link.

At the remote terminal, demodulation reconstructs the video signal, which is used to modulate the density of print produced by a printing device. This device is scanning in a raster scan synchronised with that at the transmitting terminal. As a result, a facsimile copy of the subject document is produced.

Probably you have uses for this facility in your organisation.

Yours sincerely,

Phil.

P.J. CROSS
Group Leader - Facsimile Research

Registered in England: No. 2038
Registered Office: 60 Vicars Lane, Ilford, Essex.

detail. It will not, for example, allow the user to read the full width of a line of text. However, every bit of detail originally captured can be displayed a segment at a time. The second of the extreme options is to reduce the number of pels horizontally in the ratio of 1728:720 (12:5) and vertically in the ratio 2100:384 (6:1 approximately). This will allow a distorted (compressed vertically) representation of the original page with about one pel for each original 13 pels at around the original horizontal size. Depending on the algorithm used for selecting the 12 pels to be discarded, the end result may or may not be usable (readable). It depends also on the level of detail in the original. (See Figure 3.)

Between these two extremes is a range of options that allow for greater readability than the latter but with more area to be displayed than in the former.

Figure 3 Whole page reduced to fit on the display screen

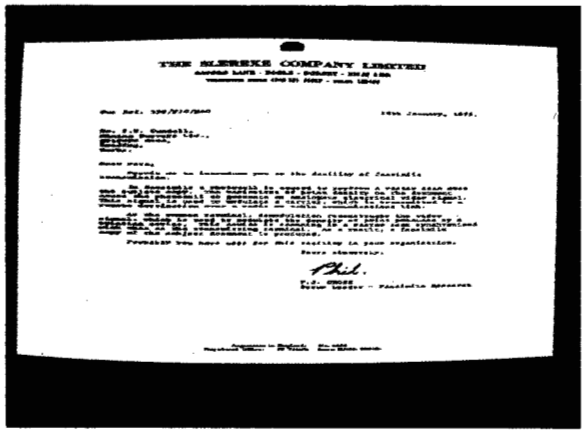


Figure 4 Page reduced by a linear factor of three

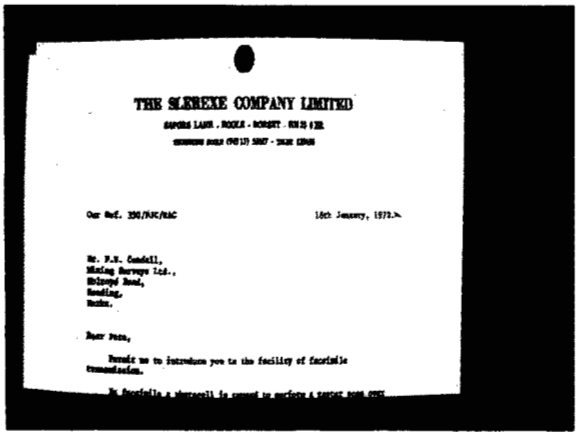


Figure 5 Page reduced by a linear factor of two

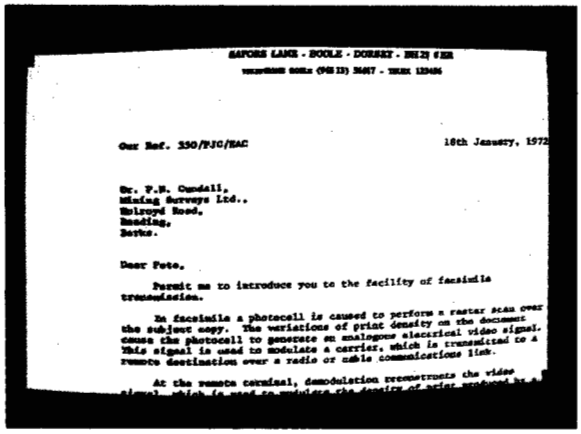
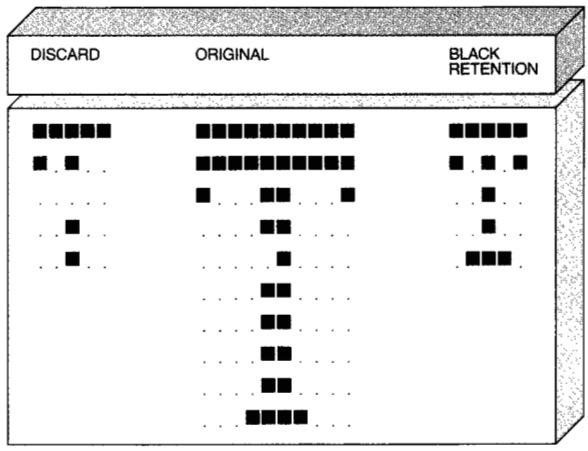


Figure 6 Effect of the two techniques on a 2:1 reduction



In terms of processing, the term used to denote the amount of detail present has been called "resolution." Original resolution refers to the scanned level of detail and reduced resolution to the other options. Resolution reduction may, with the appropriate algorithms, be performed by any arbitrary factor. The initial factors used, however, were integer only (three and two). This restriction permitted us to build an image-viewing prototype that allowed, with a factor of three, display of the full width of an A4 page in 1728/3 or 576 pels on 80 percent of the 3279 screen width and 55 percent of the page height. (See Figure 4.)

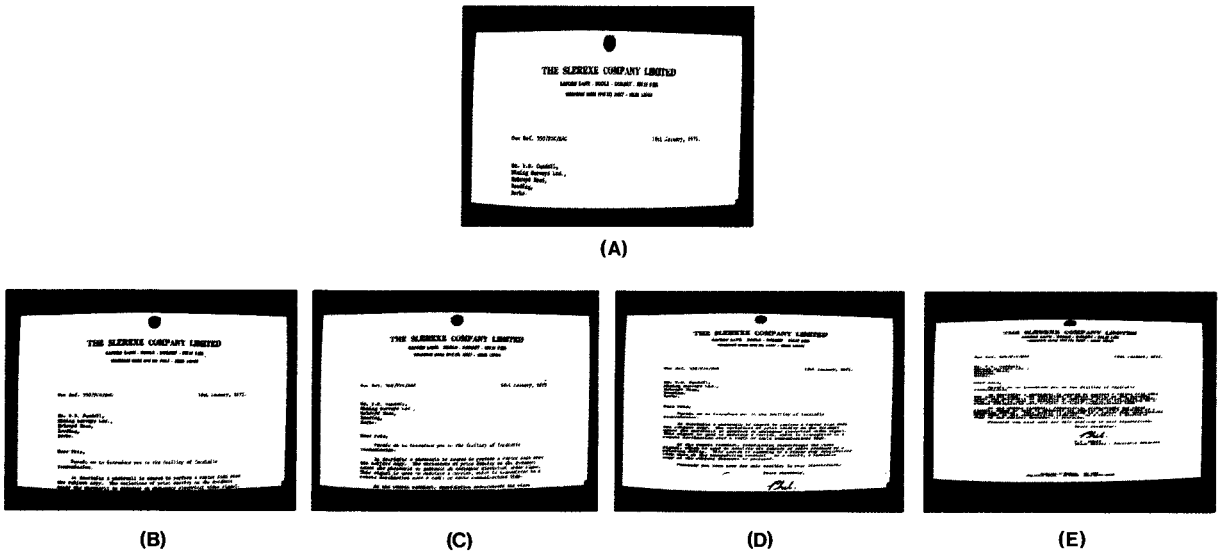
At a factor of two, some 83 percent of the page width [720/(1728/2)] and 36 percent of the page height was visible.

At a factor of two in each direction, nearly everything on a normal "office" document was fully legible. However, if the line length of the original was more than 7.2 inches (as with the document shown in Figure 5), it was not possible to read the content naturally. At a factor of three in each direction, clear originals could be read with difficulty.

Before discussing the next options that were tried, it will be useful to consider the strategy adopted for reducing resolution.

As stated in the introduction, the objective was to keep processing costs for the commercial environment to a minimum. There were no overriding social

Figure 7 Effect of various levels of vertical compression: (A) No compression, (B) Reduced to three rows out of four original rows, (C) Reduced to two rows out of three, (D) Reduced to one out of two, (E) Reduced to one out of three



(A) (B) (C) (D) (E)

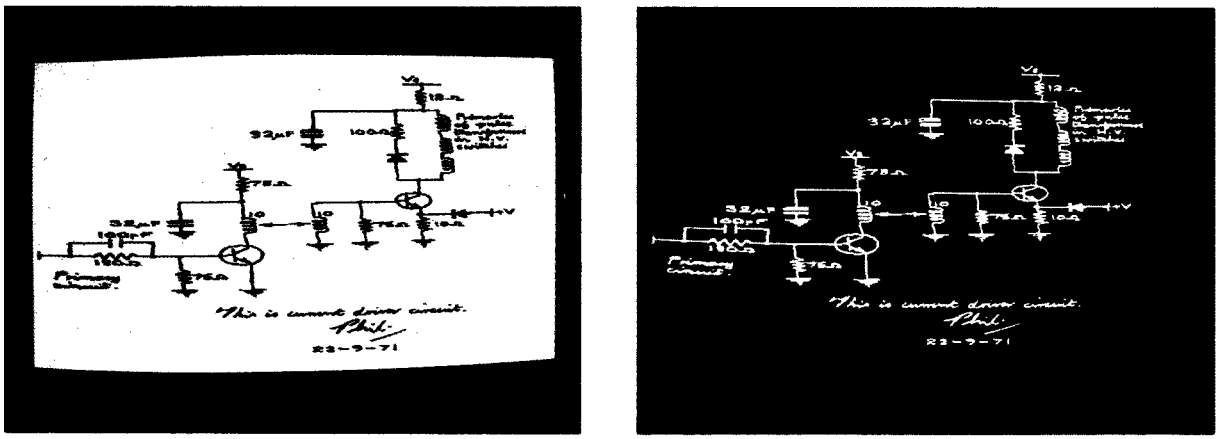
or scientific reasons for potential users to utilize the whole of their processing power on image-related tasks. It was decided at the outset that techniques that relied on looking at each bit and its neighbors individually were not possible because of the number of bits involved. Two techniques were tried that could be performed at the byte level at least. The first merely discarded the appropriate number of bits in each direction, and the second "ORed" the bits in each direction, which has the effect of retaining the black pels. This is illustrated in Figure 6.

In practical terms, discarding pels seems to give a "thin" image with bits missing, and for a fine typed font, where the character strokes are only one or two pels wide, the characters can actually disappear. And yet, the "ORing" tended to blacken and smudge characters such as "a" and "m" where the hole or the gap between strokes was less than three pels wide. Eventually the black pel retention method was adopted. The programming technique used was first of all to OR the appropriate number of scan lines together to reduce the vertical resolution by an integer factor and then to combine pels horizontally. The latter is done by using the translate (TR) instruction followed by the PACK instruction to perform the table lookup and pel recombination required in as few (albeit slow) instructions as possible for the greatest possible number of bits. Considered but not implemented were techniques to avoid processing where there was no need. For example, an all-white line may be reduced in resolution by a factor of two by the simple expedient of moving half of a white line to the output area in one move. The compromise to be made here is that the checks to determine whether a process should be performed must not take longer than the processes saved.

So far only integer changes in resolution have been discussed and with equal values in both directions. It can be seen from examination of the simple technique for the vertical factor that for a given horizontal factor it is trivial to have a variable vertical factor. Experimentation with this led to further developments. When displaying on an IBM 3279 display station that has a nonsquare pel, a truer representation of the original could be displayed by use of a factor of three vertically in conjunction with two horizontally and four vertically for three horizontally. This amount of additional vertical compression did not seem to lead to a significant loss of legibility but did allow up to 50 percent more page content to be displayed at one time. Increasing the factor a step at a time (thus increasing the vertical compression) until the whole page length was displayed gave an insight into the utility of using a display in this way and showed what of the content was legible.

The greatest mismatch, however, when using integer factors was that either only 80 percent of the page width was visible or only 80 percent of the available display width was used. This mismatch led to an-

Figure 8 The options of positive and negative display



other compromise to use a noninteger resolution change on a reduced amount of data. Since the extra reduction was only required when displaying full page width, a noninteger reduction, which takes more processing than integer reduction, on the already-reduced (by a linear factor of two) image means that only one quarter of the bits need to be processed in this way. An algorithm designed for the specific purpose of changing the resolution of images from 240 pels per inch to 200 pels per inch (to allow Scanmaster printing of images created for the 6670 and 3800 printers which both print at 240 pels per inch) happens also to convert an image with a width of 864 (1728/2) pels to a width of 720 pels (same as available on a 3279). This algorithm, applied after the integer reduction of two, then gave a total reduction of 2.4, which allowed the full page width to be viewed without the loss of legibility incurred by going to a factor of three. The compromise involved was to perform the 240–200 (6:5) *after* 2–1. All noninteger changes in resolution are best in terms of quality for continuous-tone (gray-scale) or high-resolution images. The quality would have been better if the processing had been performed 6:5, then 2:1, but the time taken would have been longer.

With the original detail available if needed and the full page width at reduced resolution also available, there is the basis for an image display system on the IBM 3270 family of displays. Figure 7 shows the effect of different vertical compression on an image already reduced in each direction by 12:5. The vertical compression is achieved by ORing the discarded line with the next-used line and then using the combined line rather than two separate ones.

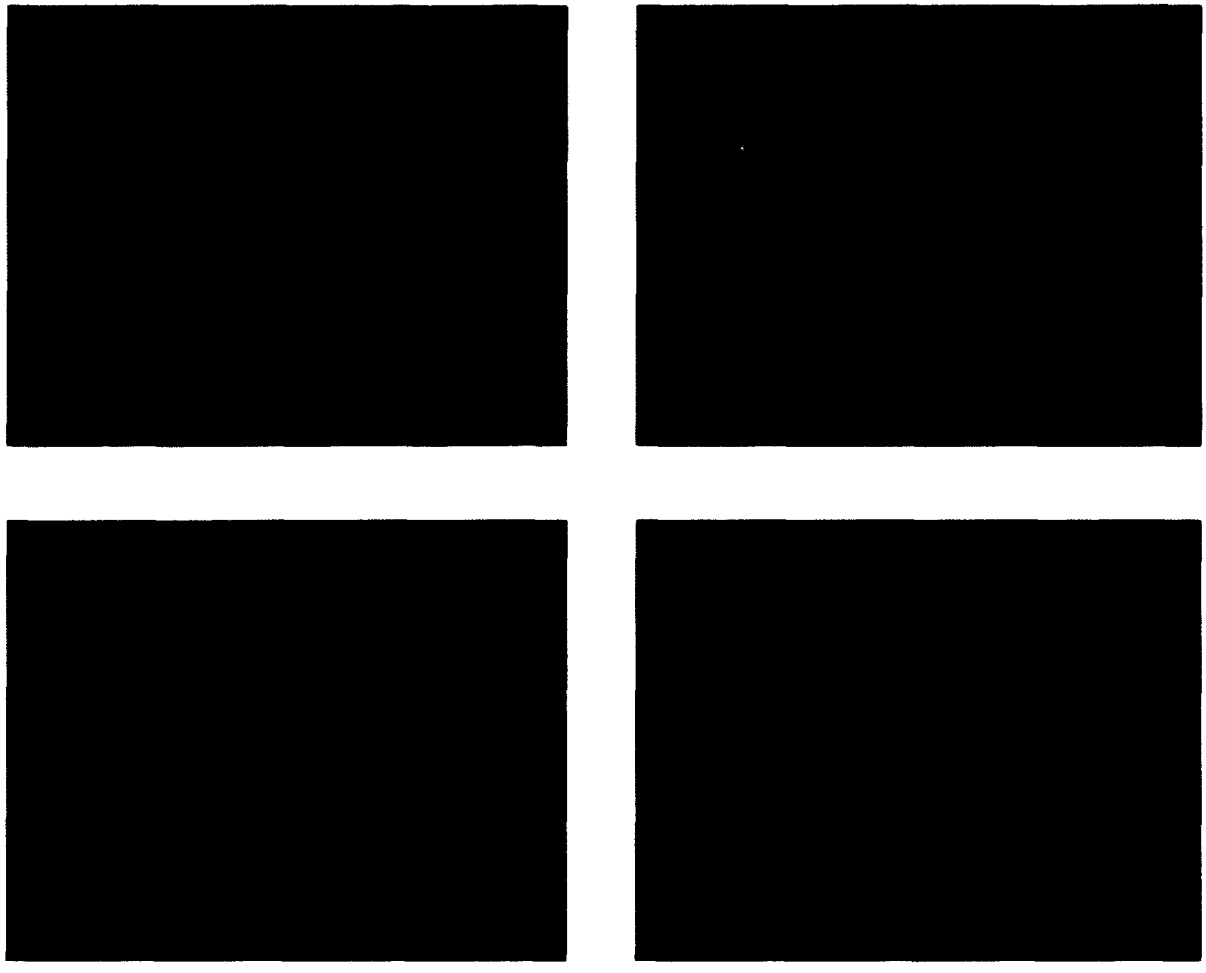
User interface for displaying images

The display devices of the 3270 family are terminals widely used for data processing and office applications. With the addition of the Program Symbol Set (PSS) features, they have been used for business graphics applications. They may also be used to incorporate images with any text and data applications that already exist. The software that is used to drive them is the Graphical Data Display Manager (GDDM) Program Product.[9]

GDDM provides an application program interface for images such that a user program may, in a GDDM call (GSIMAGE), pass a canonical string of image bits to GDDM together with information concerning the rectangular shape of the resulting image. GDDM then will generate the 3270 data stream that causes the 3270 display equipped with PSS to display that image, together with any other data that has been passed in the data stream. PSS provides a total of 1140 unique character cells (of typically either a 9-by-12 or 9-by-16-bit matrix, depending on the display station used) that may be displayed. On a display of 2560 characters (32 by 80), if an image causes GDDM to generate a unique character for each character display position, only about the top 45 percent of the display height could be used. In practice an image document contains enough white space so that the remaining 1139 characters are sufficient (all white pels for any area of the character cell size are counted as one unique character).

To create the PSS features and to send them to a device is one of the roles of GDDM, and this process-

Figure 9 Examples using the 3290 Information Panel for image display



ing requires a significant number of instructions. It also generates a large amount of data to be transmitted to the 3274 controller to which the display device is attached. This must be considered when building a user interface for image display.

With a monochrome display, images may be displayed with the "black bits" either lit or dark so that the effect on, for example, a green phosphor CRT is to have a printed page appear as either green image on dark background or dark image on green background. On a color display where white and black can be displayed, the image may be shown either as a photographic positive or as a negative (see Figure 8). Human factors and the type of images displayed may dictate which of the two the user prefers. Indeed, with a color display any suitable combination of two contrasting colors may be chosen.

During the project new IBM devices such as the 3290 Information Panel were becoming available. This device is a plasma display with a pel capacity of 960 by 750. It is also provided with a native image capability that does not require PSS to drive it. Even though some of the limitations of the older devices were removed, it was felt that any user interface should be the same for all types of devices. The advantages of the large panel would be gained by using a common interface to obtain more of an image at a higher resolution. Examples of images displayed on a 3290 panel are shown in the photographs in Figure 9, and the extra area available for

Figure 10 The character prompt line for image viewing

display is shown to advantage. Orange on a dark background is the normal way of presenting an image on this device.

When displaying traditional text documents (typically a half or one third of a page at a time), the user may, with a small amount of processing, scroll to whatever part of the page he wants in view. In the extreme, scrolling can be done one line at a time on a system with a subsecond response time for trivial transactions. Because of the amount of data involved in transmitting an image between a host processor and an unintelligent terminal, this level of scrolling is not achievable for image documents.

Consider some of the aspects of an image document that the user may have in his "electronic in-basket" or as part of a processing application:

a. Is it a random document about which he knows nothing?
b. Does he want to look at a predefined spot on a business form to see the signature in a great degree of detail?
c. Does he want to look at and then index the document (all the relevant information is contained in the top half of the page)?
d. Is the document in landscape format (more wide than high) and thus been scanned sideways?

For a user for whom the majority of image documents fall into a particular category, it should be possible to provide a view that is useful for his task the first time, and he should not need to do much scrolling or redisplaying because of the amount of data transmission involved. In the cases above, the first view might be

a. Full page display so that headings, logos, and signatures can be recognized and further action then taken.
b. Original resolution display of a segment with top left coordinates of x, y.
c. Full page width but no vertical compression, giving half-page view at good legibility.
d. Rotation of the image before display.

Having obtained his first view, a user with the random document (a) is the most likely to want to have a further display once he has got an impression of what it is he is looking at. (Think of what happens when you pull a letter out of a newly opened envelope. What makes you look at the top?—at the bottom?—or the second page? It is an impression, not a detailed look.)

With the whole page displayed and an impression of the content, the user needs a facility to get to any other view in one iteration of redisplay. Actions he may wish to request at once may include

- Change of resolution (see more or less detail)
- Change of vertical compression (in addition to vertical resolution; also see a greater or lesser height)
- Selection of an x, y coordinate to be the top left of the new display
- Scrolling up or down, left or right
- Rotation of a landscape page or one scanned upside down
- Selection of another page

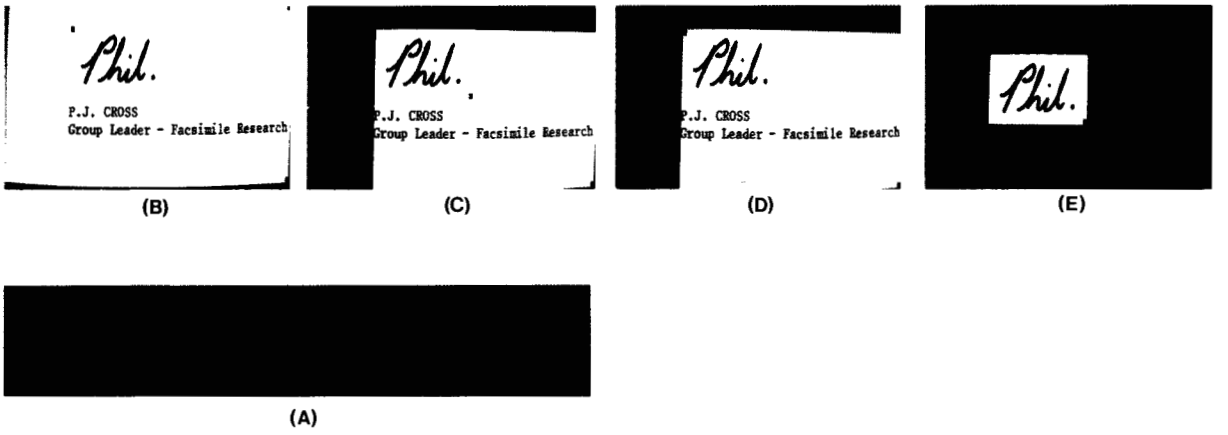
An objective was to limit the area on the screen used for character prompting of the user.

In the Image View Facility program offering,[10] the tangible outcome of the work behind this paper, an interface has been implemented that permits the user to handle this type of request without knowledge of an extensive set of commands.

Because of the relatively small size of display area available for image display, an objective was to limit as far as possible the area on the screen used for character prompting of the user. Two character lines at the bottom of the screen are used (see Figure 10).

The first line gives details of the image name and the page within a multipage document. It also shows the

Figure 11 The image clipping application: (A) Prompt lines for clipping, (B) Selection of top left corner of image, (C) Result of B, (D) Selection of bottom right corner, (E) Result of D



current settings of resolution, vertical compression, and rotation from the original scanned orientation. The second line contains entry fields for the user to request amounts of vertical and horizontal scrolling, selection of another page from the document, and changes in resolution, vertical compression, and rotation. To assist the user, program function keys may be used to change the contents of these entry fields. The initially displayed values in these fields are the same as those of the current values. The scrolling fields are set to zero. Repeatedly pressing, for example, the program function key associated with vertical compression will cause the field to cycle between the valid values for that field. Alternatively the user may key valid values into the fields he wishes to change. To get a degree of fine scrolling, the cursor may be moved into the image area and positioned at the point in the image that should become the top left reference point for subsequent actions (e.g., for scrolling or for display of a segment at original resolution).

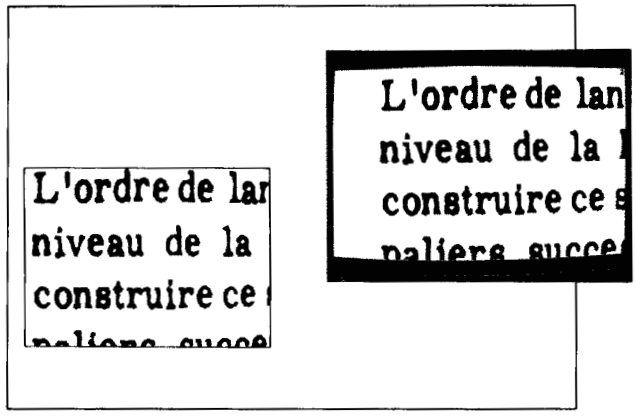
If the user does require further help information to make his selection, a character help function is provided through a program function key which, although it uses the entire screen area, does not destroy the image area in the PSS buffers. This function is managed by GDDM. On exit from the help function, the image may be quickly redisplayed without regenerating and retransmitting the PSS data stream. A similar approach is taken with display of error messages.

This approach gives the user considerable flexibility in viewing an image and in allowing him to go quickly to the part of the image he wants at the level of detail required for the application. He may set defaults for all changing display values so that the first view of each image is the one that he wants.

The user may wish to perform other character mode activities while viewing an image. An example is to enter coded information such as indexing terms that should be associated with that image. The technique used to achieve this, while still using only two lines from the display, is to allow the user to change the use of these two lines. One set, for example, provides the viewing manipulation; another set provides a data entry area and user-prompting information for entering the data. An example of yet another image application that was implemented in this way was that of image clipping. In this application, a rectangular section from the image is selected by the user at the display by the use of the character cursor. The block cursor is used to define two opposite corners of the rectangular area required. Program function keys are assigned to image corners, and the effect of the clipping may be seen interactively as each corner is selected. (See Figure 11.)

Because of the character cell size, this facility does not allow for clipping to a finer level than between six and nine pels horizontally and between nine and sixteen pels vertically, depending on the cursor characteristics of the terminal in use. It represents a cutting error of up to about one twelfth of an inch. Again, this is a compromise between simplicity of use and implementation and the function provided. Once the segment has been selected, the user may cause that segment to be extracted from the image

Figure 12 Use of advanced digital magnification techniques for increasing the pel count for printing at high resolution



and filed separately from the original image together with descriptors giving the size and starting x, y coordinates relative to the original.

Other applications that are suited to this approach are the routing of an image to another type of output device such as a printer (discussed in the next section) or to a more advanced image-processing application.

The objective with the user interface implemented was to provide as flexible a tool as possible to allow users to incorporate their own unique application functions.

Printing images

Printing an image requires an all-points-addressable bit printer. There are a number of examples of these in IBM's standard product range.

The IBM 3287 Color Printer that attaches to the standard 3270-family cluster controller (the 3274) is driven with program symbol sets. It is a low-resolution device that can be used as a convenience printer. It may be driven directly by the GDDM program product in a manner similar to a display device. Normally a GDDM user is interacting with a display, but the program in the System/370 host using the GDDM facilities may specify another output device and provide the hardware characteristics for that device. When this program requests GDDM to print out the page created (it may also contain text and graphics as well as image), GDDM creates the device-dependent data stream to drive the defined printer.

Another printer which may be used is the IBM 3800 Model 3 printing subsystem. It is a high-speed mainframe channel-attached laser printer with considerable image capability. It is driven by a complex data stream containing structured fields that define the data content. Image content is described in structured fields comparable to those that define the image obtained from the IBM Scanmaster. Information on segment positioning and resolution is provided in the data stream. Since the 3800-3 has a resolution of 240 pels per inch in each direction, an image captured on the Scanmaster at 200 pels per inch would be reduced in size by 240:200 on printing. A resolution change of 5:6 (the transform of which the previously described 6:5 transform is the reverse) on the image before passing it to the printer allows for a true reproduction of the original (±2 percent).

An image in its bi-level canonical form is as easy to process as data on 80-column punched cards.

For very high-quality output suitable as camera-ready copy for the printing industry, the IBM 4250 electro-erosion printer provides a complete all-points-addressable capability at 600 pels per inch. To drive this with images sourced at 200 pels means either a threefold size reduction on printing or some sophisticated processing to maintain quality and avoid the "laddering" that occurs when purely replicating the number of pels in each direction by three. The result of some early work of digital magnification of 200-pel input to 600-pel output is shown in Figure 12 together with a photographic magnification of the original for comparison.

As with the 3287 printer described above, GDDM is used to create output suitable for the device geome-

try. However, the printer is actually driven using the Composed Document Printing Facility (CDPF) program product.[11]

The IBM 6670 Model 2 Information Distributor as a system-attached printer has a limited "programmable" font capability at 240 pels per inch. Small image segments can be converted into font information (analogous to PSS for the 3270 family) and transmitted to the 6670 for printing.[12] In practice this is limited to a few square inches of image in a page of text, but this in general is sufficient for logos or signatures that must be included in a page of otherwise textual information.

Images as another form of data

So far the discussion has centered around the processes necessary to capture, index, file, retrieve, display, and print images. The unique image processes were decompression and resolution changes.

However, it is hoped that the reader now realizes that an image in its bi-level (one bit per pel) canonical form is as easy to process as data on 80-column punched cards. For Scanmaster images, each page conceptually generates, in the decompressed form, a box of 216-column binary punched cards. As with punched card data, it is easy to select cards and columns for listing or display and for selectively filing. The difference is that it is noncoded information. There is no easy way of adding it up or comparing it with other data. At present it is only possible to refer to it and to manipulate it. The manipulation can be simple or very complex.

Examples of simple manipulation are integer changes in resolution as described above and clipping (selection of a range of columns from a range of cards). With these two basic operations, a rudimentary "scissors and paste" application can be built.

Moving a segment onto a page will overwrite what is there; ORing a segment onto a page will merge the segment with what is there already. More advanced use of Boolean logic on the bits that comprise two images will give various effects that are used in the printing industry, which deals, in its end product, exclusively with images.

A more complex operation that may be performed on images is rotation. It does not present much of a problem for a multiple of 90-degree rotation. The only thing to be considered is the number of bits that have to be moved individually, which is difficult in byte-oriented hardware. A 90-degree clockwise turn is illustrated in Figure 13 for an 8-by-8-bit image.

Turning an image an arbitrary amount is a much greater problem. Algorithms exist to handle this situation, but it is not the type of code that a programmer puts together in an afternoon. As with many complex numerical processes, it is a tool that users would eventually expect to be available as a subroutine. Indeed, the simpler routines are already available with the Image View Facility.

As described earlier, the significant characteristic of canonical form image data is its volume. When the original image has been captured by the Scanmaster,

Figure 13 Rotation of an 8-by-8-bit image

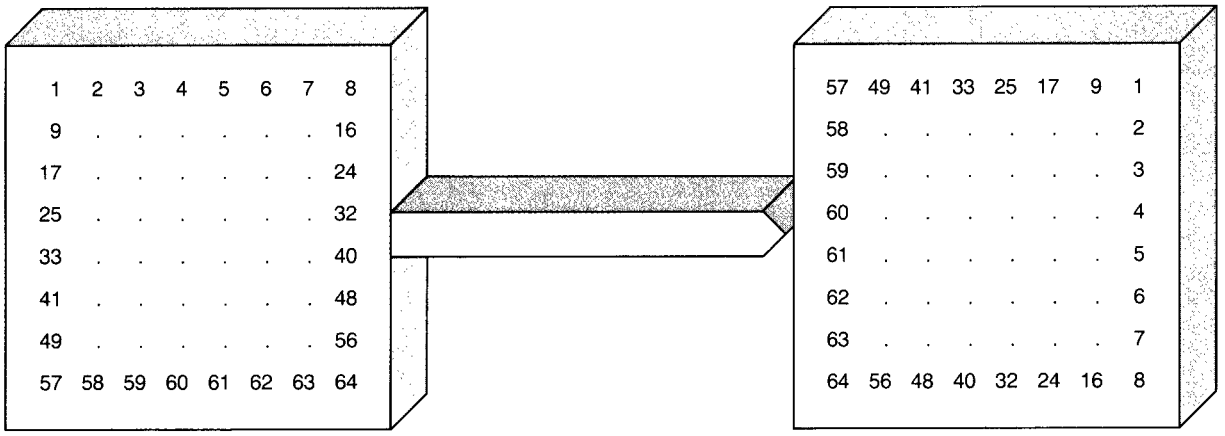
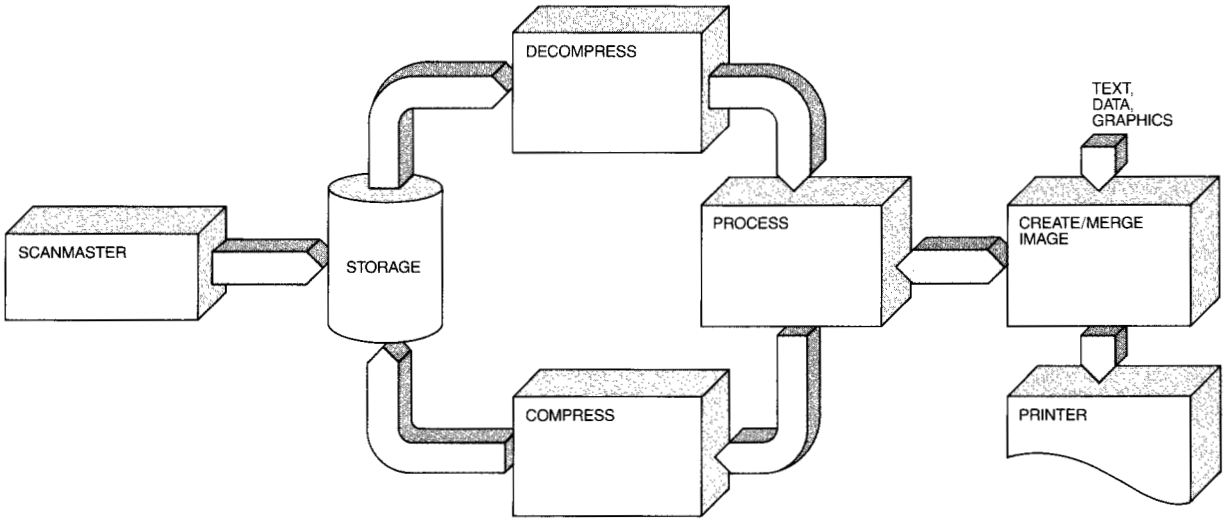


Figure 14 The possible flow of a simple image processing system, including capture, storage, and processing of images as discussed in this paper



the compressed form is available, and decompression is used when referring to an image. If that image is manipulated and actually changed using some process, a compression routine is required if disk storage requirements are not to be prohibitive or if that changed image is to be printed on a Scanmaster. Such an algorithm exists in System/370 code and is one of the basic kit of subroutines available to the image programmer.

To complete the picture shown in Figure 14, the means of creating image (pel) data from text, data, and graphics, such as provided with the GDDM and CDPF support of the 4250 printer mentioned earlier, gives a complete information merge to create a printed image page.

Before an organization takes up the task of creating and modifying images, consideration should be given to the enormous variety of data that is intrinsically image in its employees' day-to-day activities. Consider those applications and data that have not been put on line because somewhere they contain a greater or lesser element of noncoded or image content. A few examples will illustrate this:

- Pictures in a training manual
- Sketches of parts in a storekeeper's picking list
- Handwritten notes in the "electronic office"
- Signatures in a credit-checking application
- Proposal forms in an insurance claims application
- Location sketches for utility supplier customer files

A further step in the integration of image into a data processing system is concerned with the automatic interpretation of the content of the image. Possibly the simplest image that may be subjected to this process is typewritten data on a white sheet of paper. There are a number of techniques developed for optical character recognition that achieve a high degree of reliability.[13] For images with a mixture of text, graphics, and pure pictorial images, work is being done[14] to separate the different blocks on the page and to handle each according to its characteristics.

A research prototype in the related area of optical character recognition (OCR)[15] improved on classical OCR by utilizing the additional redundancy available through word (rather than character) matching. This prototype decodes that part of the scanned page that could be recognized as words in a particular font; words not recognized would be analyzed for recognizable characters, and finally, unrecognized text would be treated as image segments.

To allow a page to be created from different information types, it is essential to develop a data structure that can contain the different types. This struc-

ture is known as Document Content Architecture and has been described in a paper in a previous issue of this journal.[16]

Some basic tools

The basic kit of software subroutines required by a programmer writing an image application include

- Compression/decompression
- Resolution changing, both up and down, integer, and noninteger
- Rotation—90, 180, and 270 degrees, and general
- Trimming and erasing (clipping the edges and cutting a hole)
- Merging, overlaying, and underlaying

With these subroutines as tools and the ability to capture and store images as well as to create images from coded data, a user can start to address the enormous number of image-related applications that exist in every organization.

Conclusion

Image is the medium through which mankind sees the world. The division of information into data, text, image, and so on is predicated by the current limitations of information processing systems. The inclusion of image as a valid and useful information type that can be processed in a way similar to already established types will bring nearer the day when the processing systems we use do not require us to decide which type we are using.

Acknowledgment

The author has acted more as a system integrator than as an original thinker on this project and has brought together the original work of many researchers and developers from throughout the IBM Corporation. Some of this work is cited in the list of references. Others to whom special thanks are due include Joan Mitchell, Fred Mintzer, Karen Anderson, Jerry Goertzel, John Dawkins, Gary Wooding, Paul Freeman, Roger Way, Peter Hurrell, Peter Stucki, David Adler, David Miller, Martin Hibbert, Ed Todd, Chris Gage, J. Bruce Smith, and John Minshull. Many other professionals throughout the Corporation are to be thanked for their encouragement and advice.

Cited references

1. P. Franchi, J. Gonzalez, P. Mantey, C. Paoli, A. Parolo, and J. Simmons, "Design issues and architecture of HACIENDA, an experimental image processing system," *IBM Journal of Research and Development* **27,** No. 2, 116–126 (March 1983).
2. F. Orti, M. Rebollo, J. Jimenez, J. Lopez, and A. M. De-Aragon, "Geographic data from Landsat," *Perspectives in Computing* **1,** No. 1, 39–42 (February 1981).
3. D. Anastassiou, M. K. Brown, H. C. Jones, J. L. Mitchell, W. B. Pennebaker, and K. S. Pennington, "Series/1-based video-conferencing system," *IBM Systems Journal* **22,** Nos. 1/2, 97–110 (1983).
4. *IBM Scanmaster I (Machine Type 8815) Description Manual,* GA18-2094, IBM Corporation; available through IBM branch offices.
5. Known variously as LUC or LU6.2. It is described in *Systems Network Architecture: Format and Protocol Reference Manual,* SC30-3112, IBM Corporation; available through IBM branch offices.
6. *DISOSS/370 Version 3 General Information Manual,* GC30-3085, IBM Corporation; available through IBM branch offices.
7. *Distributed Office Support System/Professional Support (DISOSS/PS)* is a program (Program Number 5796-PRH) that provides 3270 display support for DISOSS/370. *User's Guide,* SH20-2695, IBM Corporation; available through IBM branch offices.
8. Y. Takao, *An Approach to Image Editing and Filing,* Tokyo Scientific Center Report, G318-1554, IBM Japan (November 1981).
9. *Graphical Data Display Manager (GDDM) General Information Manual,* GC33-0100, IBM Corporation; available through IBM branch offices.
10. *Image View Facility Program Description and Operations Manual,* SB19-5919 (Program Number 5785-ECX), IBM Corporation; available through IBM branch offices.
11. *Composed Document Printing Facility (CDPF) General Information Manual,* GC33-6133, IBM Corporation; available through IBM branch offices.
12. *IBM 6670 Font Editing System Program Description and Operations Manual,* SB11-5744 (Program Number 5785-FAW); *IBM 6670 Image Printing System Program Description and Operations Manual,* SB11-5979 (Program Number 5785-FAZ), IBM Corporation; available through IBM branch offices. These two programs assist the user in creating and using fonts on the 6670.
13. R. G. Casey and C. R. Jih, "A processor-based OCR system," *IBM Journal of Research and Development* **27,** No. 4, 386–399 (July 1983).
14. K. Y. Wong, R. G. Casey, and F. M. Wahl, "Document analysis system," *IBM Journal of Research and Development* **26,** No. 6, 647–656 (November 1982).
15. N. F. Brickman and W. S. Rosenbaum, "Word Autocorrelation Redundancy Match (WARM) technology," *IBM Journal of Research and Development* **26,** No. 6, 681–686 (November 1982).
16. M. R. DeSousa, "Electronic information interchange in an office environment," *IBM Systems Journal* **20,** No. 1, 4–22 (1981).

Reprint Order No. G321-5225.

Peter J. Somerville *IBM United Kingdom Limited, Rosanne House, Bridge Road, Welwyn Garden City, Hertfordshire AL8 6JH, England.* Mr. Somerville is a Senior Systems Engineer in the Eastern branch office of the ISAM Division. He joined IBM in 1965 after working as a production controller. He has been involved with a wide range of large and small systems hardware applicable across most of the industry sectors to which IBM markets its products. He is currently responsible for advising other professionals in his office on all aspects of office systems and end-user interfaces. Mr. Somerville has a master of arts degree in mechanical sciences from the University of Cambridge.