Structures and Metrics For Image Storage & Interchange

Victor Callaghan
University of Essex,
Wivenhoe Park,
Colchester CO4 3SQ, England

Keith Prettyjohns
Photometrics Ltd,
3440 East Britannia Drive,
Tucson, AZ 85706, USA

Adrian Alvarez
Digital Equipment Corporation,
Louisiana No. 80,
Col. Napoles, Mexico

Abstract:

There are hundreds of different image file specifications in existence. A recent informal survey recorded almost 100 formats in use by USENET readers alone. Thus, an imaging practitioner is faced with a large and sometimes bewildering range of image file standards to chose from which, when coupled with the sparsity of studies in the area, makes acquiring a general overview of the field a difficult task. This paper will seek to address this problem by reviewing the overall topic of image formats, describing the most notable standards, proposing a set of related metrics and providing a source of further information.

1 Introduction

The format of image files is a fundamental issue in imaging systems. It determines how easy it is to process, move or exchange images between different users or systems. There are hundreds of different image file specifications originating from such sources as commercial companies, research groups and individuals. This variety of formats becomes a particular problem when images need to be shared between different systems and users. The reasons for such a large number of standards are probably attributable to the fact that, historically, the imaging industry's main customers have largely been highly trained scientists and engineers who were only too happy to write their own application specific code, using their own image file formats and data structures. The diverse nature of imaging applications (e.g. astronomy, military/government, medicine, biology, and publishing) often meant that code and file structures had little general applicability. The tendency to develop imaging standards independently of other disciplines was fuelled further by the natural inclination of these workers to publish their own specialist conferences and journals[1]. The advent of powerful graphic workstations has magnified the problem by spawning a new generation of imaging applications such as multi-media and desk-top publishing. Thus, for a variety of reasons, we have a profusion of imaging standards.[2] Currently, an attempt is being made by the International Standards Organization (ISO) to define a universally accepted standard. In the longer term, this holds the promise of reducing the complexity of designing imaging systems and software even though in the short term, the engineer is still faced with a bewildering choice of image file formats[3].

This paper will attempt to cast some light onto this rather poorly documented subject by presenting a list of the formats in use, describing the most common ones and offering some related metrics. In addition, current developments in the field will be discussed.

2 Image Storage & Interchange Systems

The study of image file formats is largely an investigation into how image data can be efficiently represented and combined with ancillary information that pertains to the image for the purposes of image storage, processing and communication.

2.1 Some Imaging Basics

This paragraph introduces the main imaging terms that will be used elsewhere in this paper.[4] In the context of image file standards, the term image can be interpreted as being a two dimensional matrix (x,y) whose elements equate to the light intensity at the corresponding spatial position of a real or abstract scene. Image capture or *digitization* is the process by which a continuous image is converted into a *discrete* image. Being discrete means that the image light intensity is sampled at a fixed number of locations and quantized to a fixed number of levels. The size of the image matrix (x,y) is commonly referred to as the *spatial resolution*. Each sample or display point is referred to as a *pixel* (picture element), and its maximum brightness value as the *contrast resolution*. A typical image will contain both *luminance* (tonality or light intensity) and *chrominance* (colour) data. An image with no chrominance information and only two intensity levels is called *bi-tonal* (or bi-level), whilst one with several intensity levels is known as *multi-tonal* (or grey-level). Images which include colour information are known as *polychromatic* (or multi-spectral) whilst those with no chrominance information are known as *monochromatic* (or black & white). Images require a lot of storage space (e.g. a 24bit 1k x 1k colour image requires 3 MBs). By coding images, it is possible to compress them in order to use considerably less storage. Hence, *compression* is a common function in image storage formats. Compression schemes which do not incur loss of data and make it possible for the original image to be reconstructed so as to contain the exact same pixel values are referred to as *loss-less*, whilst those that do not allow the reconstruction of the exact original image data values are termed *lossy*. Some images are generated by means other than sampling, such as those computed from physical sensory information in medical nuclear magnetic resonance or synthesized graphic images.

The principal objective of a good image file format is to provide a structure which maximizes the utility of image data across a set of applications.

2.2 Format Characteristics

Whilst a digital image is essentially just a two dimensional array, the abundance of differing file formats suggests that there are many ways to store and represent images. These differences often relate to such things as the size of the image accommodated, provisions for tonal or colour support, the use of compression algorithms and the number and type of additional fields catered for (e.g. image specifications, annotations, contextual information etc.).

A simple image file format might include the following features:

Header
 *A File Identifier
 * Image Specification

Image Data
 *Look-Up Table
 *Image Raster

Footer
 *File terminator

Figure 1 - Simple Image File Structure

The *header* is an area in which non-image data is usually placed. The *file identifier* is usually a string which identifies the file type (e.g. TIFF uses 42) and the format revision number. The file type identification is often referred to as a signature or magic number. The *image specification* contains information about the image which needs to be read before the image can be decoded and displayed. For example, it might contain data on the encoding system (e.g. compression), height, width and depth of the image. Auxiliary information can also be included in the header (e.g. date the image was created and method of creation etc.).

The *Image Data* section of the format is where the image is located. A *Look Up Table* (LUT) is simply a tabular mapping mechanism. In this scheme, raw image pixel data values form an index to a table, thereby enabling them to be translated to other values (e.g. RGB colours). The *Image Raster* area contains the image itself. It can be encoded in various forms (e.g. compression, byte ordering etc.) which need to be understood before the image can be recovered. Images are very expensive in terms of storage space, which explains why image compression is common.

The *Footer* is similar to the Header in that any non image data can be held in it. Normally, it contains only a flag, the *File Terminator*, to signal the end of the file. However, it is sometimes convenient to place extensions resulting from format revisions in this area (e.g. the 1989 revision of the TGA format uses the footer to hold new fields such as a signature and extension/developer area pointers).

In a rapidly advancing environment such as imaging, a format needs to have sufficient flexibility to be able to adapt to future changes and avoid premature obsolescence.

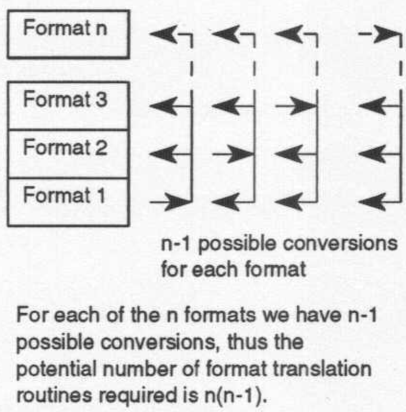


Figure 2 - Conversion Diagram

2.3 Interchange Issues

As described in the preceding section, many companies, institutions or universities have generated their own software and image file formats. These are usually tailored to their own computer facilities and applications, which can differ enormously. As a result, it is frequently necessary to translate image file formats whenever it is required to exchange imagery between institutions.[5] Clearly, a convenient way to effect the required translation is to build a software tool. Most of the time the receiver has to perform image conversion from the sender's format into his own. This requires appropriate conversion tools and may cause loss of information. It can be done (and often is) in an ad-hoc manner, writing in-house conversion software as the need arises. The disadvantages of this in-house approach is that it requires someone to acquire an intimate knowledge of the incoming format and expend time writing the necessary code, both of which are usually superfluous to the primary objectives of the work. Whilst, given suitably skilled personnel, this is clearly a possible solution, it is not necessarily the most efficient way to manage the overall translation problem. For instance, given that image translation is a task which is common to many institutions, it would make sense to avoid duplication and the consequent waste of effort by sharing conversion programs between institutions. Also, by using a structured method rather than an ad-hoc translation scheme, it is possible to reduce the number of translation routines required. For example, consider figure 2. This illustrates the format translation problem. As can be seen from this diagram, theoretically, n(n-1) different translation routines are required to convert image data directly among n formats. The introduction of a new format to the image interchange group requires the software development of additional (2n-1) converters.

It is possible to reduce the number of conversion routines required by using multiple conversion steps. However, this is usually not the best solution as the probability of finding a conversion path without information loss is massively reduced by the increase in conversion steps. Thus, using this method to avoid information loss requires a great deal of care and knowledge of the image formats concerned.

A popular variation of the above method is based on the use of a single interim format, an *Interchange Format*. Here, any arbitrary conversion from one format to another would require two steps: firstly the translation of the source format to the interchange format and secondly the translation of the interchange format to the target format. Hence the number of converters required for all possible format conversions may be reduced from n(n-1) in the direct scheme to 2n using the single interchange format system. To eliminate image information loss during the conversion process, the minimum requirement of an interchange format is its ability to support a loss-less form, to which the other formats can unpack. Where the encoding method and other auxiliary data need to be preserved, the multitude of formats and their rich variety of auxiliary field types can lead to a universal interchange format becoming more complex than other standards. Such a complexity could dissuade programmers from adopting it as their basic format. However, the complexity can be reduced by preserving only the basic image data or targeting a subset of imaging file formats. An example of a relatively simple interchange format is the PBMPLUS set, whilst a more complex and universal standard is the IPI standard currently being developed by ISO.

Clearly, the need for interchange formats to preserve universality whilst maximizing storage and communication efficiency creates overheads, which will always give specialized formats a clear simplicity and processing advantage within a particular application domain.

Thus, in summary, given the variety of imaging file formats in use today, interchange formats represent an efficient way of facilitating the exchange of digital images.

3 Brief Overview of Common Image File Standards

Within the space limitations of this paper, it is not possible to provide an in-depth description of all known file formats (footnote 1). Therefore, a subset based on popularity[3] or notability have been chosen for discussion. For instance, GIF, TIFF and Portable bitmaps are clearly popular standards, whilst PCX is noteworthy due to its association with IBM PCs. IPI, Photo-YCC and HDF are emerging standards. A source of information on obtaining file standards is provided at the end of this paper.

3.1 PBM

PBM is an acronym for Portable Bit-Map format (referred to as the "portable format" in the rest of this description). It was developed originally to allow bitmaps to be sent by mailers unable to handle pure binary. It is one of the most straightforward image file formats in widespread use. This simplicity has made it popular as a general image interchange format and it forms the basis of the popular PBM-Plus image file conversion toolkit. The standard is broken into 3 parts. There are separate formats for bi-tonal, grey-scale and colour images known respectively as PBM (portable bi-tonal map), PGM (portable grey-scale map) and PPM (portable pix map). All of these portable formats have a common structure consisting of a header followed by the image. The header contains a signature, the spatial resolution of the image (width & height) and, in the case of PGM and PPM formats, a maximum grey or colour component value. In portable format terms, the signature is referred to as a magic number and identifies the file type. Images can be held in raw binary or ASCII; this is also flagged in the signature. The principal difference between the Portable Format variants is the form the image data takes. The PBM variant represents an image by a stream of ASCII or Binary 1s and 0s, mirroring the raster image starting from the top left of the screen. For convenience, the stream is broken into lines of no more than 70 characters. PGM adopts the same principle and differs only in that the stream is composed of ASCII or binary grey values between 0 and the maximum depth set in the header. In the PPM variant, the stream is broken into sub-groups of 3 characters representing red, green and blue values of the relevant pixel, again within the range set in the header. In the binary mode, the maximum grey level or colour component depth is limited to a byte (i.e. 255).

3.2 SunRaster

As the name suggests, this format was introduced by Sun for use on their range of workstations. It is a fairly simple format, consisting of a header, an image and a colour map. The header consists of eight 32-bit integers which signal the width, height, depth, length and type of raster image contained in the file. In addition, the header commences with a unique "magic" number which identifies it as being a SunRaster file and ends with details on the type and length of the colour map. Both monochrome and RGB colour images are catered for. The main image raster formats supported are raw-bitmap, RLE, (X)RGB and (X)BGR types. The depth of the image can be set to be 1, 8 or 24 bits per pixel. There is a 32 bit version which includes an alpha channel. The format is based on that used and defined in the Sun PixRect graphics library.

3.3 GIF

GIF (Graphics Interchange Format) was developed by Compuserve, the world's largest on-line information service, to enable their users to exchange colour graphic files independently of the hardware platform they own.[2,4] Currently, there are two main versions of GIF, introduced in May 1987 and July 1989 respectively. In simple terms, a GIF file consists of a header followed by a set of multiple images with accompanying colour maps. As is common with such image files, the header commences with a unique character string (signature) to signal its file type and version. The header also contains a segment known as the screen descriptor which gives the overall parameters of GIF images, such as the overall image area needed (can be virtual or logical rather than physical screen space), the background screen colour and the image depth. Following the header are the images. These have three components: an image descriptor, colour map (optional) and the raster data itself. The image descriptor defines where the image will be positioned within the overall screen image and flags the presence of an associated local colour map. For image compression, GIF uses a variation of the LZW (Lempel-Ziv & Welch)[6] algorithm known as

(1) Readers interested in medical or biomedical applications may care to refer to publications by Dean[1] and NEMA.[12]

Variable-Length-Code LZW. Current versions support image depths of 1 to 8 bits. The most recent version of GIF (89a) uses two notional levels of abstraction, blocks and sub-blocks, to describe the GIF mode. A block is defined as being some set of statements prescribing the information needed to enable the graphics (or some sub-part) to be reproduced in their intended form. Thus, most of the header and image descriptors can be seen to be block type data. Raster data is contained in sub-blocks. In this model, a sequence of blocks and sub-blocks is referred to as a GIF Data Stream. GIF is a well defined and popular standard which, coupled with Compuserve's policy of providing the specification free of charge, should ensure it remains a leading image format for some time to come.

3.4 PCX

PCX (PC graphiX) is the image file format used by ZSoft Corporation's PC Paintbrush graphics application. Undoubtedly, the highly successful combination of this product with the ever prevalent IBM PC has contributed to its popularity. PCX has a rather simple and fairly rigid structure which has almost wholly been driven by IBM graphics display hardware requirements. A PCX file consists of a header, two colour maps (optional) and a raster image. The first two bytes of the header form a signature, identifying it as a PCX file and providing the version number. Other bytes in the header provide information such as the image resolution. There are no restrictions on the spatial resolution of the image although the contrast resolution (bits per pixel) is limited to one byte. Images can have up to four planes. For monochrome images only a single plane is required whilst to support colour, up to four planes can be utilized (red, green, blue, intensity). The image is usually encoded in an RLE dialect. The original colour map only provided for 16 colours and was wholly contained within the header area. Each map entry comprises 3 bytes (red, green, blue). In recent versions of PCX, ZSoft have added provision for a second "extended" colour map (referred to as an "extended colour palette" in PCX jargon) placed after the image and offering 256 entries or colours. PCX images are intended to fill the entire display screen. A variant of the PCX standard known as PCC (PC Clip art) provides a mechanism for images to be displayed in chosen parts of the display area.

3.5 TGA (Targa)

This image file standard was developed by Truevision Inc. in 1984 to support their widely used video graphics products. A revised specification (version 2.0) was introduced in 1989. It is claimed to be the first widely used true-colour format. The format consists of a fairly traditional header, colour map and image. An image may be variable in size and can be represented in monochrome, true colour (i.e. no colour map but direct storage of images in RGB values) and direct- or pseudo-colour (i.e. using colour map). Where compression is required, run-length encoding (RLE) is employed.[7] The use of a colour map is optional and only a minimal set of restrictions exist (e.g. variable length map, with each map entry being an integral number of bytes, usually 2, 3 or 4). The revision in 1989 introduced three slightly unusual segments which follow the image, namely the developer & extension areas and a footer. The original specification did not include a file signature in the header; this omission has been corrected in the revised format by including it in a footer segment. To determine the file type or version, it is necessary to read the last 26 bytes of the file. The developer and extension areas are mechanisms for the file to be customized and extended to provide a better match for the needs of a particular application in a way that avoids making the main specification too general and complex. The developer area is a variable size portion of the file's space which may be used to store information of any type. Truevision recommend it be used only for application specific data. The size and format of this area is left open to the developer but to enable the area to be segmented into various fields, a system of sub-sectors with IDs (TAGs) and a directory is employed. Truevision maintains a list of registered TAGs (and their contents) so that, where relevant, 3rd parties can read the information within this area. However, the nature of this area means most file readers would skip it as they would have no requirement, or be unable, to read this information. The extension area is a portion of the file that is used to store additional information about the file itself. Its fields (both function and size) are set by Truevision in response to internal development and external requests. Examples of fields in this area are: name of image creator, date image was saved, postage stamp image etc.

3.6 FITS

The FITS (Flexible Image Transport System) image file standard was proposed in 1981 to service the specialist needs of the astronomical community. In particular, it originated from the need to transfer large astronomical images between installations using 9-track, half-inch magnetic tape. Since its inception, it has been modified many times and is now defined in logical terms rather than relating to any particular storage medium. Currently, a draft of the formal codification of existing FITS practices is being considered for endorsement by the International

Astronomical Union. A simple FITS file would consist of a header and an image. In FITS jargon, these would be combined in a structure referred to as a HDU (Header and Data Unit). FITS images differ from other, more traditional images in that they are arrays which can have up to 999 dimensions and that the data can be represented in unsigned bytes, 16/32bit 2's complement or 32/64 bit floating point form. There are no limitations on the spatial dimensions of the image. The information usually held in traditional image headers is present in sub-areas of the header known as "card images" and contains data prefixed by keywords which convey such things as the numbers of bits per pixel etc. There is no explicit signature, although there are mandatory keywords in the header image cards, which can be used in an equivalent manner. The simple structure outlined above can be augmented by other structures known as random groups, conforming extensions and special records. These provide mechanisms to cater for other structures, which can be more complex than simple 2D-image arrays.

3.7 TIFF

TIFF (Tagged Image File Format) is said to be a superset of most image file formats. It was developed in 1985 to service the needs of scanning and desk-top publishing vendors.[2] Its aim was to help these markets by preventing the profusion of competing proprietary standards. To do this successfully, it needed to incorporate enough functionality to eliminate the need for alternative formats. This led to it sacrificing simplicity to powerful functionality and flexibility. Thus, the price TIFF pays for being a universal imaging format is complexity. Lindley[8] illustrates the relative complexity of TIFF by referring to two similar libraries he wrote for TIFF and PCX, which comprised 13,500 and 2,600 lines of C code respectively. This complexity is widely recognized and it is claimed by many that there is no single piece of software that utilizes all the functionality and attributes of TIFF. This complexity usually shows up in TIFF readers, although TIFF writers can be relatively simple. To manage this complexity, TIFF (revision 6.0) now segments TIFF features into sub-groups; TIFF Baseline and TIFF Extension features. TIFF Baseline refers to a minimum set of TIFF features that *all* general purpose readers should implement whilst TIFF extensions are those that will only be used by special applications. Thus, the complexity of writing TIFF software has been reduced by decreasing the average number of features needing to be supported. This concept builds onto the notion of classes used in TIFF 5.0. These were recommendations on the usage of a sub-set of TIFF options for specific applications which enabled some generality to be sacrificed in order to reduce the related software complexity. The structure of a TIFF file comprises a header, n File directories and n Images arranged in a hierarchical structure. The header contains a byte order flag, a signature to identify the file type and a pointer to the first Image File Directory. The first Image File Directory holds a pointer to the raster data (plus header data) and to the next Image Directory. This directory is linked, in turn, to the next Image Directory and so on, until all image directories are linked in a chain. Thus, multiple images may be stored within a single TIFF file using a system of links to provide a hierarchical system of headers and related images. Each image has its own Image File Directory (a sort of sub-image header). These directories contain entries, each providing particular information about the image related to the directory in question. Each such entry is identified by a unique number known as a tag. The meaning of each tag is clearly defined and has a variable associated with it which either directly provides the data concerned (if it comprises 4 or less bytes) or points to where the data is (providing a 2nd layer to the hierarchy). Images in a TIFF file are divided into groups of raster lines referred to as strips (each usually less than 8 Kbytes). Tags within an Image File Directory point to these strips. Using strips allows images to be handled in smaller sub-sections rather than as a whole. Up to 65,535 tag numbers are available. Tags are divided into those reserved for public (0-32,767) and private (32,768-65,535) use. To preserve widespread usability, applications for private tag usage are administered by Aldus. Thus, much of the flexibility and power of TIFF can be attributed to its use of tags and pointers. TIFF 6.0 defines 74 tags (as against 45 in version 5.0). Examples of the information provided by tags are image resolution, size & location of image strips, compression method and image creation data. TIFF images can be bi-tonal (1bit), grey-level (2-16bits) and colour (up to 24 bits). RGB colour maps are available (size = $3_{RGB} \times 2^{BitsPerPixel}$). Various image formats and compression schemes are supported, such as raw binary, CCITT (fax), CMY, CMYK, PackBits (Mac), RLE, JPEG and LZW.

3.8 EPS

An EPS (Encapsulated PostScript) file holds image information in the form of a program (a page description language) which instructs the display where and how to draw lines and fill surfaces so as to reconstruct the original image. Thus, it differs radically from the conventional ways of storing images which use bitmaps and arrays. Essentially, an EPS file is a standard postscript language file with an optional bitmap at the beginning to allow applications (e.g. word-processors) which are unable to understand PostScript commands, to include and display the image. An EPS file usually contains a header, image bitmap (optional) and the image description. The first line of the file is a signature which confirms the file type and revision number. Postscript programs are intended to be generated by other programs rather than humans. However, it uses a simple syntax based on a printable subset of ASCII (plus new-line) so programs can be constructed and read by humans. Postscript utilizes an interpretive stack machine model. Images are constructed by "inking" areas of the display, in any colour. There is a rich set of drawing instructions such as those for drawing lines, filling shapes and cropping areas etc. An EPS file should conform to the latest version of the Adobe Document Structuring Convention (see EPS references) and be well-behaved, returning the including environment to its original state upon completion. The variety of operators and rigid syntax can make the creation of related EPS software relatively complex.

4 Recent Developments in Image File Formats

The formats discussed above are well established, many having been revised numerous times. However, there are several more recent developments which, although too new to appear in the popularity rankings, have the potential to become major contenders. Worthy of special note is the proposed IS-12087 standard, which addresses both image formats and processing. It is currently at least 24 months away from full ratification and is enjoying the active support in its definition of many leading imaging companies and individuals. Another format, which is already receiving much attention, is Photo-CD/YCC from Kodak. Photo-CD products, manufactured in conjunction with Philips, are beginning to appear on the market. An interesting variation of the TIFF concept is to be found in the DSF format. Finally, data sharing is not only a problem confined to those involved in imaging, but potentially affects all computer users. HDF is a good attempt to solve this general problem. The following sections give a brief description of these developments.

4.1 Photo CD & YCC

Photo-CD is a system originally proposed by the Consumer Imaging Division of Eastman Kodak[9] to enable images captured on traditional photographic film to be digitally transferred to a compact disc (CD) and displayed on a television using a special Photo-CD player (which also plays standard audio CDs). To develop the CD technology, Kodak joined forces with N.V. Philips.[10] Photo-YCC is a device independent colour encoding method employed by the Photo-CD system. It was based on the CCIR 600-1 & 709 video standards, which enables it to minimize the display processing overheads. To reduce this overhead further, images are decomposed into a hierarchy of increasing resolution image components, ranging from 128 lines by 192 pixels through to 2048 lines by 3072 pixels (the aspect ratio of a 35mm frame is 3:2). The components which make up a single image are stored in a grouping on the CD known as an *Image Pac*. An Image Pac consists of a header (known as an IPA - Image Pac Attributes), the image components (in order of increasing resolution) and an extension field (IPE - Image Pac Extensions). The IPA contains data such as the image source whilst the IPE is currently undefined. In the Photo-YCC encoding scheme, colour pixels are represented by one 8-bit luma and two 8-bit chroma components. The standard defines a non-linear conversion from RGB to luma and chroma which is claimed to allow rapid, efficient quantization to video display metrics without sacrifice of gamut, colour fidelity or device independent performance. Image compression is achieved using a combination of quantization and Huffman encoding. A Photo-CD disc has a hierarchical file structure. At the top level is a directory called PHOTO_CD; it contains a sub-directory called IMAGES which in turn points to the image Pacs themselves. An Image Pac for a 35mm frame will typically need 3-6MBytes. A standard 120mm disc can hold about 100 images.

4.2 IPI

As was mentioned earlier in this paper, it is surprising that, despite the importance and widespread use of imaging systems, to date, there is no international standard relating to imagery (as distinct from graphics). The lack of such a standard for image files has surely been a significant factor in the creation of today's large and diverse population of image file formats. Fortunately, such an international standard is currently being prepared under the auspices of the ISO. It is known as the IPI (Image Processing and Interchange) standard[11] (footnote 2). Work started on it in 1990, after an ISO letter ballot, and is currently at committee draft stage, which means it is probably two years or more away from final ratification. The standard is slightly unusual in that it addresses both image transfer and processing. The standard is organized into three parts: generic architecture, programmers' imaging kernel system (PIKS) and the image interchange format (IIF). Operations to be addressed by the PIKS section include image enhancement, restoration, analysis, transport,

(2) Thanks to Dr. Adrian Clark for providing an advance copy of his Eurographics paper introducing IPI.

compression and basic classification and visualization. Operations specifically excluded from PIKS include computer graphics, sensor acquisition, device control, communications, multimedia, windowing systems and image understanding. Computer language dependency is to be excluded from the IPI standard, being delegated to bindings to be described in a separate standard. The IIF section of the standard addresses two areas: the first is the structure of image data itself, whilst the second concerns how such data can be passed back and forth to a PIKS compatible application. In simplified terms, it is intended that the IIF data format will contain a format descriptor, an optional header and a contents section. The format descriptor will hold information on the file type, a conformance profile and a version number. The contents section will contain two parts, the first describing the data structure whilst the second will be the data itself. At this point, the IPI proposal becomes somewhat complex and difficult to summarize as it provides for a huge variety of data structures. In highly simplified terms, the standard proposes three fundamental data types: basic, non-image and image. Image data can be one of two types: elementary images and compound images. Elementary image data types consist of such things as pixel arrays, whereas compound data refers to more complex structures intended to provide higher levels of structured access to large sets of related images. Compound images utilize structures such as arrays, lists, pointers, strings, tables and sets. Non-Image data types include colour tables, histograms, feature lists and image annotation. Basic data types include bits, characters and various number types of which the more complex structures, described above, are built. Data will be passed between IIF and PIKS implementations using a data stream. This stream and control mechanism is, in IPI jargon, collectively referred to as a gateway. It is expected that IIF will support most popular compression schemes, such as those used in fax groups 3 & 4, JBIG, JPEG and MPEG. IPI is widely considered to be significantly more complex than TIFF (a point which may become more meaningful when IPI-IIF is eventually ratified and released to the world!). Like the TIFF *classes* mechanism, IPI is proposing to reduce application software complexity by defining IPI subsets called *profiles* for certain applications. On completion, the standard will be assigned the number ISO-12087.

4.3 HDF

The name HDF (Hierarchical Data Format) disguises a powerful concept addressing the general computing problem of data interchange and manipulation. It comprises data structures, software tools and a well structured mechanism for future adaptation and growth. This latter aspect is perhaps its most significant advantage over many of the competing schemes described elsewhere in this paper. It allows it to evolve in both a horizontal and vertical direction (in an application sense) as technology and demand change. The standard was developed by the National Centre for Supercomputing Applications (NCSA) at the University of Illinois at Urbana-Champaign. Conceptually, the structure of a HDF file can be viewed as a header (inc. signature) followed by a number of data objects. Data objects are structures consisting of a data descriptor and a data element. The data descriptor contains a tag designating the data type, a unique reference number and pointers to the data itself. NCSA manage the assignment of tag definition. At the time of writing, only about 40 of the potential 65,536 tags have been defined by NCSA (although some have been allocated in block to other bodies). This flexible approach means that a wide variety of data types can be supported, ranging from simple text to complex numeric arrays. The use of pointers enables the data descriptors to be physically separate from the data itself. In fact, HDF group data descriptors use a linked list to form data descriptor blocks. The physical structure of a HDF file consists of a header, data descriptor block and data elements. The only constraint on the distribution of these components is that the first data descriptor block must follow the header, the other aspects of structure being maintained by the pointers described previously. The software support takes the form of high (application) level interfaces (e.g. image processing) and a low level interface (i.e. for basic HDF manipulation). The low level interface can be used to build and manage files of any sort, including custom designs. These interfaces may be called from either C or Fortran. Using such prefabricated software greatly speeds up application program development. The penalty incurred by entwining software support with the storage structure is a loss of portability. However, NCSA have attempted to address this by providing support for most scientific computers. HDF also defines minimum data object "sets" that can be used for specified applications such as imaging (c.f. TIFF classes & IPI profiles). 8 and 24 bit images are supported. Finally, HDF is freely available via ftp and is supplied with clear and comprehensive documentation.

4.4 DSF

A variation of the TIFF format known as DSF (Data Storage Format) has been proposed by Dr. Ming Xie of INRIA in France. In simplified terms, he has effectively replaced the image strips (i.e. the bottom of the TIFF hierarchy) with fields which he calls datamaps. A datamap is defined only as a set of bytes thereby pre-imposing only a minimal structure upon it. Also, unlike TIFF, the directories and datamaps in DSF are referenced by their names (character string) and can be dynamically created or deleted. This format was inspired by Dr. Xie's work on dynamic vision systems where he needed to deal with quantities such as camera parameters, image sequences, sets of contour chains, etc. He has implemented DSF as a C++ class.

5 Factors in Selecting a File Format

The process of selecting the most appropriate image file structure for a particular application involves gathering format information from a variety of sources and attempting to make a comparative analysis of it. For instance, in order to decide which format might be most appropriate, it may be necessary to gather and analyze multifarious information such as file structure, field specifications, relative popularity of file formats or even some measure of the difficulty in writing associated software. The main obstacles encountered are obtaining copies of format specifications, dealing with the variability of the style and technical content (some descriptions, such as for IPI and TIFF, are relatively complex), as well as obtaining any generally applicable yardsticks to assist in the task. The following section will attempt to give assistance by offering some empirical measures which might be brought to bear on these problems.

5.1 Format Popularity Ratings

Whoever investigates how images are held in a computer must surely conclude that the industry has spawned a numerous and diverse range of image file standards. Although no formal study of the popularity of Image File Formats is known to the authors, two recent *informal* surveys based on USENET produced some interesting results (see appendix 2). For instance, the most striking example of format variety is illustrated in Paul Raveling's survey[3] in 1991 which found that from a set of only 108 respondents, some 99 file formats were being used! A summary of these surveys is given in the following table. TIFF, GIFF and PBM dominate both surveys. The difference in VIFF (khoros) ratings might be explained by the inclusion of the comp.soft-sys.khoros news group in the CAP survey. Formats are ranked by frequency of reported use (figures in bracket indicate ranking).

FORMAT	Raveling 91	CAP 93
TIFF	30% (2)	48% (1)
GIF	52% (1)	33% (2)
VIFF	3% (23)	30% (3)
PBM series	30% (2)	27% (4)
Homebrew	23% (5)	24% (5)
Targa	9% (10)	21% (6)
SunRaster	26% (4)	18% (7)
SGI	8% (12)	15% (8)
EPS	14% (7)	9% (9)
FITS	8% (11)	6% (10)
PCX	6% (16)	6% (10)

Table 1 - USENET Format Popularity Rating

Clearly, one should be careful not to draw any firm conclusions from such small informal surveys. Nevertheless, they serve to illustrate the variety and profusion of image file formats being used by the imaging community. They would appear to support the widely held view that the situation regarding image file formats is a bit messy and, together with the large number of formats encountered, illustrates the requirement for image conversion tools. Many practitioners argue that an ideal solution would be the creation of an internationally defined image file format that could be used by the entire imaging industry. This format should be able to represent image information in a program, compiler, machine and device independent manner. This would eliminate the messy problems associated with porting images between computer systems and software packages. Unfortunately, although the ISO are trying to develop such an image file format, at the time of writing, no such standard exists. In the meantime, we are left with numerous "parochial" file formats, most of which have been designed by various bodies according to the needs of a given application domain, or according to the requirements of a certain user group. Frequently, because formats are designed for one particular use, they tend to be inflexible and preclude future expansion e.g. they allow only one particular image size.

5.2 A Structure Taxonomy

In other areas, classification schemes have been used with good effect to aid description and selection. The existence of such numerous and diverse image file formats has made it difficult to establish a universal taxonomy for image file formats. The following section will attempt to address this problem by proposing a classification scheme based on the image file structure.

This taxonomy categorizes file *structure* as being one of four main types dependent on the existence of two fundamental properties: *field hierarchy* and *semantic labels*. It was considered that these parameters had the most fundamental effect on the overall logical structure of a format. Fields are said to be hierarchical if they hold variable pointers to other fields (i.e. they may have sub-fields); otherwise they are said to be flat. A semantic label is a field prefix variable (e.g. a tag or keyword) that *explicitly* conveys the meaning of the field and its contents. The alternative is an unlabeled field where the meaning is *implicitly* conveyed via its position. Thus, in this taxonomy, fields are classified as belonging to one of the following types:

FI	(Flat Structure, Implicit Labels)
FE	(Flat Structure, Explicit Labels)
HI	(Hierarchical Structure, Implicit Labels)
HE	(Hierarchical Structure, Explicit Labels)

An example of semantic labels are TIFF Tags, whilst TIFF IFD fields can be considered as hierarchical structures. It is clear that combinations of field hierarchy and labels could give rise to very powerful and complex format structures.

The following diagram gives the taxonomy of the main formats studied in this paper.

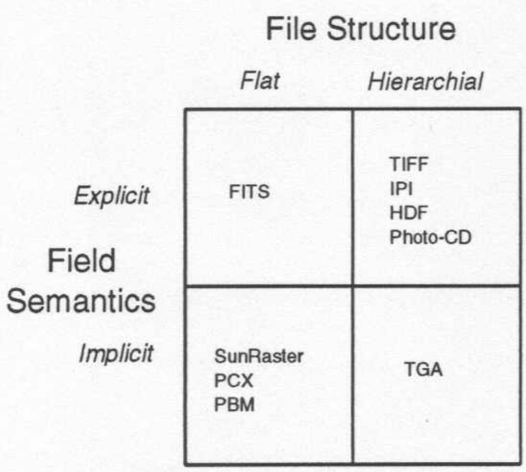
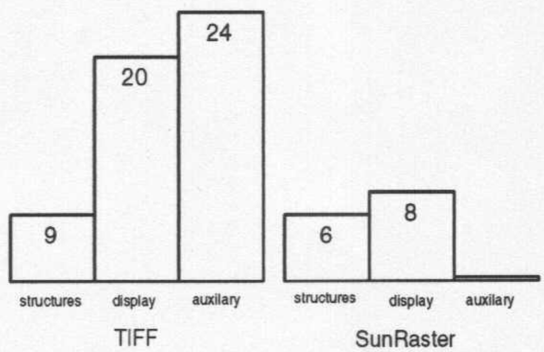


Figure 3 - Format Taxonomy Map

5. 3 Functionality Profiles

A functionality profile is a type of histogram, which maps out the distribution of functionality within a format. By selecting categories relevant to a given imaging application and counting the occurrence of such features within a particular file structure, it is possible to plot a functionality profile for an image format. Category domains do not have to be mutually exclusive or tied directly to explicit format fields (e.g. TIFF Tags). Comparing profiles with identical category templates for differing formats enables comparative assessments to be made. For example, by looking at a functionality profile, the quantity and distribution of functions might be used to get a quick impression as to the areas the format is biased towards or to get a feel for the overall complexity. Clearly, the usefulness of a functionality profile will lie in the appropriate selection of histogram categories (c.f. benchmarks). Whilst functionality profiles do not add to the available information, they package it in a more compact and visually digestible form.

To illustrate the concept, an example (see figure 4) using functionality profiles based on three simple categories, namely, *structures*, *display* and *auxiliary* is offered for the SunRaster and TIFF (version 5) formats. A rigorous definition of these categories is not given, since the figure and the informal explanation attached to it should be sufficient to exemplify the general principle.



Profile Key:
Structures: raster encoding methods, including compression (e.g. RGB, LWZ etc.).
Display: features associated with the display of the image (e.g. width, height etc.).
Auxiliary: all other data (e.g. date, author, obsolete features etc.).

Figure 4 - Functionality Profiles

When analyzing such profiles it must be remembered that they provide a comparison in terms of feature quantity only, leaving out the underlying semantic and functional complexity. Nevertheless, they can provide some interesting insights into formats. For instance, a superficial examination of these profiles immediately reveals that TIFF has many more fields to read than SunRaster. A closer inspection shows that much of this extra complexity is associated with fields not essential to the display of the images. It is hoped that, eventually, certain standard profiles (e.g. analogous to TIFF classes or IPI profiles) might be developed and made generally available for all popular formats.

5.4 A Programming Complexity Metric

As discussed previously, the imaging community is exposed to a large and diverse set of file standards. At the heart of each standard lies the image itself. However, each standard adopts its own unique strategy for encoding the image, deciding what additional information to include and choosing the most appropriate field or file structure. Given the relatively simple array-like structure of a basic image, the variety of formats must surely be a testament to human ingenuity. One result of this intellectual effort is that most formats wrap the image in a complex shroud of transformations e.g. compression, coded tags or commands etc. Before such an image can be displayed or processed, these encapsulating codings need to be stripped away to reveal the basic image. Thus, conceptually, the image could be regarded as being buried in a sea of coding procedures. This situation is diagrammatically depicted in the following figure.

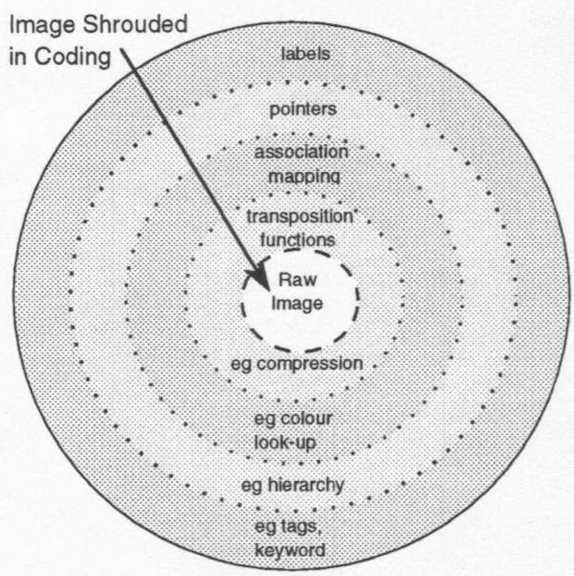


Figure 5 - Image Shroud Diagram

Therefore, in order to display or process an image, each format has a decoding overhead associated with it. This overhead manifests itself in areas such as machine processing, human comprehension and software implementation. For any given application, disadvantages such as format complexity need to be weighed up against advantages such as lower storage requirements or greater flexibility. The optimum balance will depend on the application. For instance, in the case of in-house applications, external portability may well be a lesser concern than programming effort. As a result, in-house applications frequently utilize their own dedicated format tailored to their particular applications. However, for commercial companies where universality is desirable, the opposite is often true. Deciding on the most appropriate format usually involves a lengthy and often tedious study of the image format specifications.

Application programmers take into account not only issues such as portability and performance but also programming complexity. The latter is notoriously difficult to judge; the following is offered as an empirically derived metric, which gives a very rough indication of the relative programming complexity associated with the differing formats.

Taking the view that, in order to use an image, the application software needs to perform a number of functional and associative transformations on the image file (e.g. decompress the image, recover colour table etc.), the proposed metric, which we will refer to as *the transformation index*, provides a figure loosely related to the number and complexity of such operations. It is based on an arithmetic summation of the main transformations required to display a picture, as shown on the *"image shroud diagram"*. It is derived from an examination of the image and the summing of the transformations as follows:

Transformation Index = T
(where $T = t_1 + t_2 + t_3$ etc.)

FEATURE	-------WEIGHT------	
	Present	absent
raw image [t_1]	1	0
transposition functions [t_2]	1	0
association mapping [t_3]	1	0
field pointers [t_4]	1	0
field labels [t_5]	1	0

The figure in brackets represents a weight (in this case all equal) attached to each of these features. The sum of the weights for any particular format becomes the *transformation index*. Thus, the index illustrated above is a single number between 1 and 5 (1 equating to the lowest level of transformation index, whilst 5 is the highest). Clearly, by varying the choice of category or corresponding weights, it would be possible to generate different transformation indexes.

For example, applying the *simple* complexity index defined above to TIFF & SUN RASTER formats we have:

FEATURE	TIFF	SUN-RASTER
Raw image [t_1]	1	1
Transposition functions [t_2]	1	1
Association mapping [t_3]	1	1
Field pointers [t_4]	1	0
Field labels [t_5]	1	0
Total weight	5	3

Table 2 - Transformation Index Calculations

Thus, the complexity index for TIFF and SUN-RASTER is 5 & 3 respectively, which is consistent with the fact that TIFF is a more complex format than SunRaster (in terms of coding operations required to display the image). The ranking of the other formats considered in this paper, based on this *simple* index, is given in the following table:

Ranking	Format	Transformation Index
1.	TIFF	5
1.	IPI	5
1.	Photo-CD	5
1.	HDF	5
5.	FITS	4
5.	Targa	4
7.	GIF	3
7.	PCX	3
7.	SunRaster	3
10.	PBM	2

Table 3 - Transformation Index Ranking

As will be evident from the description, this measure is roughly consistent with the apparent complexity of the formats. Where a feature is optional, the calculation makes the worst case assumption, usually that it is present and will need a transformation. For example, Sun-Raster can use a raw or RLE compressed format; the calculation assumed compression was being used. Thus, it is *a measure of the maximum depth that an image can be buried below the display or processing level*, in terms of required transformations.

6 Summary

Currently, there exists a large and diverse set of imaging file formats. They differ largely in respect of their field structure and label semantics. In general terms, these structures seek to support easy file interchange, minimization of storage as well as efficiency in local processing. If the number of image file formats is to be reduced, one capable of universal application needs to be developed. Clearly, devising a format to satisfy all these requirements whilst keeping its level of complexity low enough to ensure widespread usability is a most difficult task. Only time will tell how successful universal approaches such as the proposed ISO standard will be at unifying the image format field. In the meantime, imaging practitioners are faced with the problem of dealing with the existing and somewhat dirty topic of image file formats. This paper has introduced a number of metrics: popularity ratings, file taxonomy, format profiles and complexity. It is hoped that they will assist discussions and deliberations on image file formats.

The authors may be contacted at the addresses given on the front of this paper or via internet at: callv%essex.ac.uk

References

1. P. Dean, L. Mascio, D. Ow, D. Sudar, J. Mullikin *"Proposed Standard for Image Cytometry Data Files"*, Cytometry, 11, 561-569 (1990).
2. W. E. Carlson, *"A Survey of Computer Graphics Image Encoding and Storage Formats"*, Computer Graphics, 25(2), 67-75, (1991).
3. P. Raveling, *"USENET Image Format Popularity Ranking"*, Raveling@Unify.com (partly reproduced in appendix 2), (1991).
4. W.Gonzalez, *"Digital Image Processing"*, Addison-Wesley, (1987).
5. G. L. Graef, *"Graphics Formats"*, Byte, 14(9), 305-310, (1989).
6. T. A. Welch, *"A Technique for High Performance Data Compression"*, Computer, 17(6), 8-19, (1984).
7. S. K. Chang, *"Principles of Pictorial Information Systems Design"*, Prentice-Hall, (1989).
8. C. A. Lindley, *"Practical Image Processing in C"*, Wiley, (1991).
9. S. Stepnes, *"Behind the Scenes of the Photo-CD System"*, Kodak Pixels (Special DCI Edition), 1-2, (1991).
10. R. K.Jurgen, *"Digital Video"*, IEEE Spectrum, 29(3), 24-30, (1992).
11. A. Clark, *"An Introduction to Image Processing and Interchange Standard"*, in Eurographics (Tutorial programme T7), Cambridge England, 7-11th September 1992.
12. ACR-NEMA, *"Digital Imaging & Communications"*, Standards Publication No. 300-1985, National Electrical Manufacturers Association (NEMA), Washington DC, (1986).

Appendix 1 - Sources of Image Format Information

ACR-NEMA (Standard 300-1985)
The American College of Radiologists-National Electrical Manufacturers Association (ACR-NEMA) standard relates to medical imaging and is published by:

National Electrical Manufacturers Association
2101 L Street, N.W.,
Washington, DC 20037

DSF
For further details contact:
Dr. Ming XIE
INRIA Sophia-Antipolis
2004, Route des Lucioles
06902 SOPHIA-ANTIPOLIS, France
ming@sophia.inria.fr

EPS
Refer to the following Adobe publications:
a) Encapsulated PostScript Files Specification, Version 2.0
b) Document Structuring Conventions

FITS
More information can be obtained from:
NASA Science Data Systems Standards Office
Goddard Space Flight Center
Greenbelt MD 20771
USA
Internet: nsdsso@nssdca.gsfc.nasa.gov

GIF
A full specification is available from:
Compuserve Inc
Graphics Technology Department
5000 Arlington Centre Boulevard
Columbus, Ohio, 43220

HDF
Specifications and software are available by anonymous ftp from ftp.ncsa.uiuc.edu
(IP address 141.142.20.50) or by writing to:
NCSA -HDF
University of Illinois at Urbana-Champaign
605 E. Springfield Ave.
Champaign IL61820

ICDS
The ICDS (Image Cytometry Data Standard) is a multi-dimensional image format targeted at biomedical applications (footnote 3) and is available from:

Phillip N. Dean
Biomedical Sciences Division
Lawrence Livermore National Laboratory
PO Box 5507
L-452, Livermore, CA 94551

IPI
At the date of writing this is only at the committee draft stage. Information on national contact points can be obtained from:
International Standards Organization
Case Postale 56
Geneva 20
Switzerland

PBM
This format is used by the Portable Bitmap Manipulation package, PBMPlus written by Jef Poskanzer (jef@well.sf.ca.us). PBMPlus is available from most imagery archives (e.g. export.lcs.mit.edu:contrib/pbmplus*.tar.Z)

PCX
More information can be obtained from:
ZSoft Corporation
450 Franklin Road
Suite 100
Marietta, GA 30067
(404) 428-0008
Also, see p186 Lindley[8]

Photo-CD & YCC
Information is available from:
Eastman Kodak Company
Kodak Information Center
Dept. E, 343 State Street
Rochester, NY 14650-0811

SunRaster
On Sun workstations see: /usr/include/rasterfile.h and man rasterfile;
In Sun documentation see:
a) SunView System Programmers Guide
b) Pixrect Reference Manual

TIFF
The TIFF 6.0 specification (final draft released 3rd June 1992) is available from Aldus and Microsoft. They both provide forums on Compuserve and in addition, Aldus operate a TIFF Developer's desk on 206-628-6593 (USA), see address below. The specification is also available on-line in TIFF6.ps.Z at zamenhof.cs.rice.edu (directory pub/graphics.formats). Also, a TIFF library (public domain) written by Sam Leffler is available by anonymous ftp from: ucbvax.berkeley.edu (pub/tiff/v2.2.tar.Z). Aldus & Microsoft can be contacted at:

Aldus Corp (Developers Desk)
411 First Avenue South
Seattle, WA 98104-2871

Microsoft Corp
16011 NE 36th Way
Redmond WA98073-9717

TGA
A full specification may be obtained from:
Truevision Inc
7340 Shadeland Station
Indianapolis IN 46256-3925
or BBS: 317 - 577 - 8783 (TRUE)

On-line Archives
There are a large number of on-line archives in the world, which store much useful programming and imagery material. Access is usually by means of either a modem or WAN. There are far too many archives to include an exhaustive listing here. For those in the academic and research community with access to USENET, a useful source of information is a directory of Internet sites accepting anonymous FTP (and mail retrieval) posted by Tom Czarnik (ftp-list@netcom.com) in the following news groups: comp.misc, comp.sources.wanted, alt.sources.wanted, & news.answers. For example, with regard to image file formats, the two following US archives are worth looking at:

(3) Thanks to Jim Mullikin of Delft University for providing information on this format.

zamenhof.cs.rice.edu (in directory pub/graphics.formats) and ftp.ncsa.uiuc.edu (in directory /misc/file.formats/graphics). A most noteworthy imaging archive is PEIPA (Pilot European Image Processing Archive). This is a dedicated image processing archive, largely funded by the British Machine Vision Association (BMVA), which will, eventually, form the basis of a pan-European archive in association with DG 13 of the EC. The usual anonymous FTP access is provided (peipa.essex.ac.uk or numeric address 155.245.115.161). Further information is available from Dr. Adrian Clark (email: alien@essex.ac.uk). Also, worthy of special mention is a system known as Archie. This is a group of servers keeping a central index of files held at the various FTP sites. Some Archie sites are: archie.ans.net (North America), archie.sura.net (North America), archie.mcgill.ca (Canada), archie.funet.fi (Finland/ Mainland Europe), archie.au (Australia/New Zealand), archie.doc.ic.ac.uk (Great Britain/Ireland), archie.unl.edu (North America), cs.huji.ac.il (Israel). To receive a "help file" send email to: archie@site_name.

Appendix 2 - Usenet Surveys

These are the results of two informal polls of USENET readers pertaining to their usage of image file formats. Both lists have been truncated where reported use fell below 3.

(i) Raveling Survey

This survey was conducted by Paul Raveling in 1991[10] on comp.graphics news group. The results were posted to comp.graphics, alt.graphics.pixutils and comp.ai.vision (footnote 4).

There were 108 responses, mentioning some 99 different formats. They are ranked in descending order of citation frequency (alphabetic order within groups).

56 GIF
32 PBM/PGM/PPM(PBMPLUS)
32 TIFF
28 Sun rasterfile
25 Homebrew (footnote 5)
16 Utah RLE
15 Postscript (including Encapsulated Postscript)
13 XBM (X11R4 bitmap)
11 XWD
10 Targa (TGA)
9 FITS
9 SGI (Silicon Graphics format [s?])
7 IFF (Amiga)
7 PICT
7 PICT2
6 PCX (Windows)
5 BMP (Windows)
4 FBM
4 GEM (.IMG)
4 HDF
4 HIPS
3 ILBM (Amiga)
3 viff
3 MacPaint
3 MTV "PIC"

(ii) CAP Survey

The CAP (Callaghan, Alvarez & Prettyjohns) survey was conducted by the authors during February 1993 on comp.graphics, alt.graphics.pixutils and comp.soft-sys.khoros news groups. The additional news groups in this survey may account for some of the differences in results between the two polls (e.g the inclusion of the khoros news group will have undoubtedly affected the position of VIFF).

There were 66 responses, mentioning some 53 different formats. As with the Raveling survey most respondents cited more than one format in use. The ranking is in descending order of citation frequency (alphabetic order within groups).

32 TIFF
22 GIF
20 VIFF (Khoros)
18 PBM
16 Homebrew
14 Targa (TGA)
12 SunRaster
10 SGI
6 EPS (postscript)
4 FITS
4 PCX
3 HDF
3 IFF
3 PICT
3 UTAH RLE
3 XBM

Victor L. Callaghan currently holds an academic post at the University of Essex, England, where he teaches computer science. He gained his B.Eng in electronics and Ph.D in computer systems from the University of Sheffield in 1978 and 1982 respectively. Having begun his career working in the aerospace field, he spent a number of years both in industry and academia, putting his skills to use in a wide variety of projects. Most recently, Dr. Callaghan has been working on object-oriented image processing software and distributed imaging architectures.

Keith N. Prettyjohns is a Vice President of Photometrics Ltd., a manufacturer of high resolution cooled cameras. He has been working with CCDs and imaging technology for the past 15 years. Initially working on industrial imaging applications in the UK, he then took a faculty position at the University of Arizona, Optical Sciences Center. In 1984 he became a co-founder of Wyko Corporation, an optical test instrument company based in Tucson. He left Wyko and joined Photometrics in 1989. Dr. Prettyjohns has chaired two SPIE conferences on CCD technology

(4) Thanks to Ray Suorsa of NASA for the background information.
(5) "Homebrew" are in-house formats.

and applications. He has a Ph.D and B.Eng from the University of Sheffield, UK.

Adrian Alvarez received the degree of B.Sc in electromechanical engineering from the Universidad Nacional Autonama de Mexico in 1985. He has been employed by Digital Equipment de Mexico since 1988, winning an excellence award from them in 1989. As a result, he was sponsored by DEC to attend a masters degree course at the University of Essex. It was during these studies that he developed an interest in imaging. The University of Essex awarded him an M.Sc in computer science in 1991.