MANAGING AND REPRESENTING IMAGE WORKFLOW IN PREPRESS APPLICATIONS

Dennis L. Venable[1]
Rob Buckley[1]
Toshiya Yamada[2]

Keywords: Images, Prepress, Structured Images, Image Processing

Abstract: The growing availability and accessibility of digital images, imaging software and electronic printers is leading to new and innovative prepress and printing applications. Examples are demand color printing and custom publishing. Rather than develop independent solutions on an application-by-application basis, technologies which simplify and integrate solutions for a broad spectrum of applications are needed. This paper will present a standard representation for describing color raster images in terms of data objects such as scanned images, text, graphics, and image processing operations. It will also show how this representation can be used in several prepress and print applications spanning the gamut of the create-edit-assemble-store-print work process.

Problem of Raster Representation

The publication process is an interaction between three domains:

Customer	- create content
Prepress	- prepare final form for printing
Press	- the press room.

[1] Xerox Corporation
[2] Fuji Xerox Corporation

Within these domains, many elements of the work process can be identified. In the Prepress domain we can define work process elements that are both skills-based (Create and Edit) and potentially-automated (Assemble, Store, Retrieve.) Of course, these are generic terms and are not intended to be complete.

The problem of interest is the representation of raster image data in the prepress work process. At each step of the work process, representations must be converted into a form suitable for the next step of the work flow. A simple job such as the interactive creation of a single-use image utilizes only a few elements of the work process prior to submission to the press room. Many off-the-shelf applications can be used for this job. However, each application uses its own raster image representation which tends not to be interoperable with other applications. Variable data imaging jobs, where multiple Create-Edit work processes are combined by Assembly, tend to use custom applications with unique raster representations which, again, tend not to be interoperable.

In this paper we present a standard representation for raster imagery that can be used to simplify and integrate solutions for a broad spectrum of prepress applications. The intent is not to develop a new final-form raster representation, but to develop a single, editable representation usable throughout the gamut of the prepress work process, encompassing both skills-based and automated work process elements.

Structured Images Approach

Historically, our research efforts in digital image processing have been directed at developing software technology to make prepress operations more capable and able to produce higher quality output. The Structured Image raster representation was developed to aid in applying these technologies in the prepress work process. A Structured Image (SI) represents a compound raster image as a collection of device-independent objects and the image processing operations associated with them. SI provides a common raster-oriented representation that can be used in a wide range of operations throughout the gamut of the prepress work process. The scope of a Structured Image is the representation of a compound raster image; not a page or a document.

Figure 1 is a graphical representation of a Structured Image. The horizontal arrow represents a *pasteboard*, the frame onto which objects are placed. The pasteboard is analogous to the graphic artist's layout board. The pasteboard defines the shape, or aspect ratio, of the output image raster.

The arrowhead at the right end of the pasteboard represents the rendering of an SI into a final form, typically a raster image. Vertical lines are called *component pasteboards* and represent the processing of component objects. The first component pasteboard is called the *Initializer* and is used to specify any colorization or background pattern for the pasteboard. Rectangles represent object readers, or *ReadObjects*, which are responsible for reading an object in its native format and converting (rendering) it into appropriate raster data. Circles represent image processing operations, or *IPOs*, which perform image processing on the rendered object data.

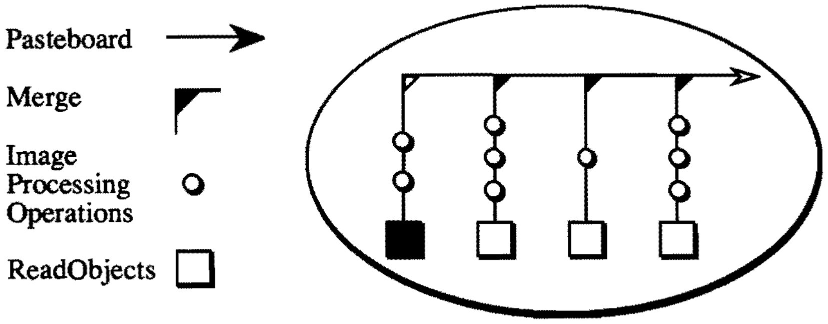


Figure 1. Structured Image Representation

The list of supported object types and image processing operations is not restricted; the current SI implementation uses a table-driven approach where supported objects and image processing operations are made available to the Structured Image rendering software by registering them in a table.

Structured Image Features

Size is not explicitly defined within a Structured Image; the desired output image size is specified as a parameter to the rendering algorithm. Thus, an SI can be rendered at any size or resolution and maintain the "same look."

To render a Structured Image into final form, the required size and resolution must be specified. Objects are converted to raster form using the appropriate ReadObject, image processing operations are applied, and the results merged onto the pasteboard at the location, in the order, and with the opacity specified in the SI.

SI objects may originate, potentially, from any data type that can be converted into a raster image. Examples of valid SI objects are ASCII strings, CGM graphics, TIFF images, PhotoCD images, and even other Structured Images (see Fig. 2.).

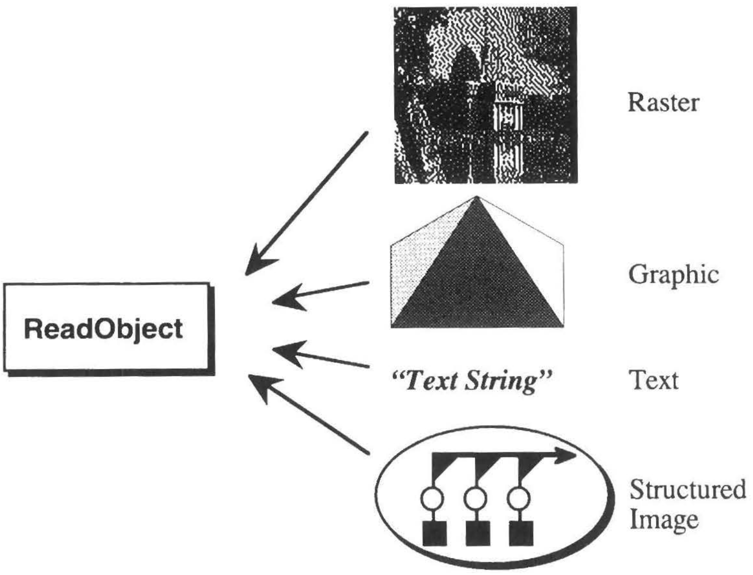


Figure 2. Sample SI Object types

A single SI object may be stored in multiple representations differing in format, resolution, dimension, and so on. Based on a merit function evaluated for each available representation, the SI rendering mechanism will choose the appropriate representation to render an output raster based on the desired dimensions, output device, etc. This allows, for example, low resolution imagery to be used for Edit work process element such as interactive layout, but automatically use high resolution imagery for Assembly and printing. The representations need not be of the same type, i.e., a single SI object may point to a CGM graphic and a low-resolution rasterized version of the graphic generated by a caching system. During the rendering of a Structured Image, the original data is never modified, thus objects maintain their heritage for selectability and modification.

In our current implementation, ReadObject renderers for raster and text objects have been developed. The raster renderer is capable of reading a many raster image formats including TIFF, GIF, TARGA, SUNRAS, PCX, IFF, BMP, and others. The text renderer utilizes outline font technology to render single font, single line text objects at arbitrary sizes.

Structured Images are stored in a Structured Image Definition (SID) file using a human-readable syntax called the Structured Image Definition Language (SIDL). The SID contains all image processing information, positions, etc., required to construct the output raster image from original object data in a device independent manner. Positions and sizes are described relative to the dimensions of the pasteboard. Pointers to object data are stored within the SID rather than actual object data. This allows the SID files to reference data objects stored in archival formats such as CD ROM without making unnecessary copies, and also allows SID files to be generally small. In the current implementation, objects can be specified as files on the network, entries in a database, or as placeholders to be bound, or specified, when an output image is to be created.

All image processing operations (IPOs) executed during the creation of a Structured Image are defined within the SID. Therefore, modifications to any step in the work process of preparing the final form image is as simple as modifying the appropriate part of the SID. Sophisticated tools can be used to aid in such editing, but simple text editors suffice.

Structured Images allow the definition of named non-rectangular regions called *selections*. Example selections are rectangles, polygons, compressed bitmaps, mathematical definitions, etc. Selections can be used as parameters to image processing operations to limit the region of processing, or as attributes associated with *ReadObjects* to automatically crop object data to appropriate areas.

An important characteristic of Structured Images is that of delayed binding. Both objects and image processing operations/parameters can be defined as placeholders within a Structured Image. At render time, appropriate object or image processing data can be dynamically bound to the SI to generate the output raster. An *Unbound* SI is one that contains placeholders for objects or image processing data. Unbound Structured Images are used to implement variable data imaging.

Examples

Structured Image technology was developed as a means for representing the prepress work process in a wide variety of prepress applications. In this section we describe several applications where Structured Images were used to describe and generate complex output raster images.

Interactive Layout

The simplest and most obvious example of using Structured Images is in the interactive creation of a compound raster image. Figure 3a is a screen dump of an interactive editor which reads and write Structured Images as its native file format. This editor supports many image processing operations such a scaling, rotation, masking, colorization, painting, framing, text annotation, etc., in a device and resolution independent manner. In this Figure, an image is partially laid out. Figure 3b is a printout of the final output raster image. This image was originally printed at 200 lines per inch on a dye sublimation printer at exactly the size requested for reproduction in this proceedings. Structured Images provided the format by which the construction of the image was defined. At any time, any operation that had previously been specified could have been modified without having to start over or invoke "Undoes" on other work.

Variable Data Imaging

Custom image assembly is an increasingly important application in prepress. SI provides the means to define a Job Application Template for customized imagery. SI also provides the framework by which the templates are filled with variable data. This is demonstrated in the two examples presented below.

The first example shows an application where an SI with unbound objects (Figure 4a) is used to automatically create and print an employee information chart. Notice that the primary SI points to several text and raster objects, and, additionally, a second Structured Image. The primary SI defines unbound objects for the organization name, manager, etc. The second SI defines a template for an image containing the employee's name and phone number centered at the bottom of his/her photograph. This SI is rendered by defining the requested output image size and the name of the desired organization. A database is automatically accessed to provide the specific information required to resolve the unbound objects, and the image is generated. Figures 4b and 4c are sample prints using this Structured Image printed at the size required for these proceedings.

The second example shows the database binding of image processing operations. In this example, the Structured Image shown in Figure 5a has fully bound data objects, but parameters to the image processing operations (the colorization parameters) are unbound. The Structured Image references three raster objects, all of which point to the same image of a 1984 Pontiac

Fiero. Each component pasteboard applies crop, scale, and colorize image processing operations to the car image. The color parameter in the colorization operation has been defined as unbound in the SI. To print this image, the SI rendering software is invoked with the required output size and three color names as parameters. A database is accessed to convert the color names into meaningful color values, and the image is generated. Figure 5b is a gray version of the resulting color print.

The unbound Structured Images shown in these examples were created using the interactive image editor mentioned in the first example. This editor has a mode which allows the user to identify objects to be unbound. Additional editing was required to define the colorization parameter as unbound.

Gang Scanning

We have shown how SI can be used as a common representation in two applications of the prepress work process. We will now demonstrate how SI can aid in improving productivity by taking advantage of the object-oriented nature of SI for data representation.

Gang scanning is a means of improving productivity by scanning many objects at once on a scanner rather than scanning objects one at a time. In this example, several photographs are placed and roughly aligned on the scanner platen and scanned at the desired resolution in a single pass. The scanner interface software will analyze a low resolution (captured automatically during the high resolution scan) of the scanned image to determine the location and orientation of each photograph.

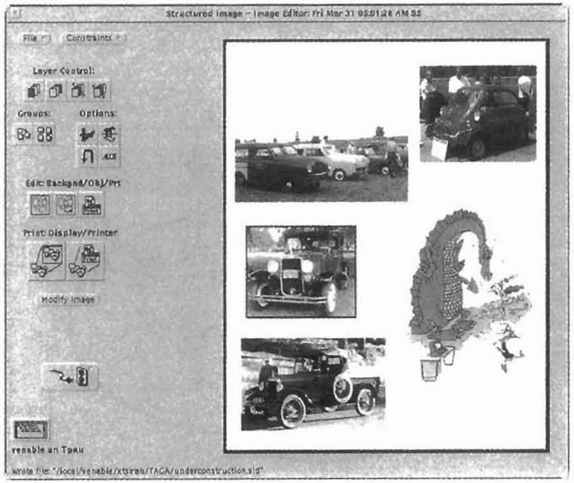
A Structured Image is then created where each photograph is represented as an SI object. All objects point to the original scanned image, but each object defines a unique SI selection to restrict the region of interest to the bounding box of the photograph in the original scan. The rotation required to correct any skew is specified as an image processing attribute to the object. Alternatively, rotation correction could have been specified as an image processing operation in the component pasteboard. With the selection and orientation attribute, the *ReadObject* renderers can automatically extract and straighten the appropriate photograph from the full scanned image. Each extracted photograph is then merged onto the pasteboard in a simple grid pattern. Figure 6a is an original scanned image; Figure 6b depicts the Structured Image automatically generated by the gang scanner interface; Figure 6c is the rendering of the auto-generated Structured Image.

Once the gang scanned photographs are represented by the object-oriented Structured Image, many other operations can be applied easily. For example, it is a simple matter to request each object to be rendered independently for storage in disk files, or as separate database entries. Figure 7a again shows the automatically generated Structured Image and Figure 7b shows each extracted image. As an additional example, a mechanism called *Template Filtering* exists where one SI is filtered through a template (unbound) Structured Image to produce a new output image. The filtering mechanism binds the objects defined in one SI with the unbound objects of the template to produce a fully bound SI. Figure 7c shows an example template SI and Figure 7d shows the result of the template filtering. This example shows the use of SI to combine the automated capture of raster information with variable data imaging.

Conclusions

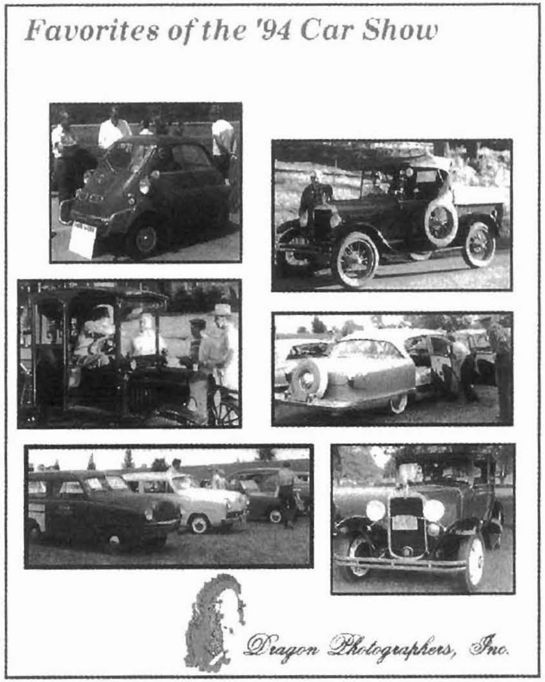
Structured Image technology is an object-oriented approach to the representation and manipulation of compound raster images. A Structured Image represents a raster image as a collection of device-independent objects and associated image processing operations. Structured Images can be created automatically or with an interactive editor, with the objects or operations either defined at the time of creation or later when the Structured Image is rendered for viewing or printing.

The examples discussed above have shown how the Structured Image raster description can be used as a common representation in several prepress applications. We have shown the example of using SI to describe the interactive creation and layout of raster imagery. We have shown several examples where SI was used to automate prepress applications such as variable data imaging and gang scanning. We are currently investigating additional prepress work process elements and applications in an effort to expand the Structured Image coverage of prepress applications. We invite the readers interest and comments on this topic.



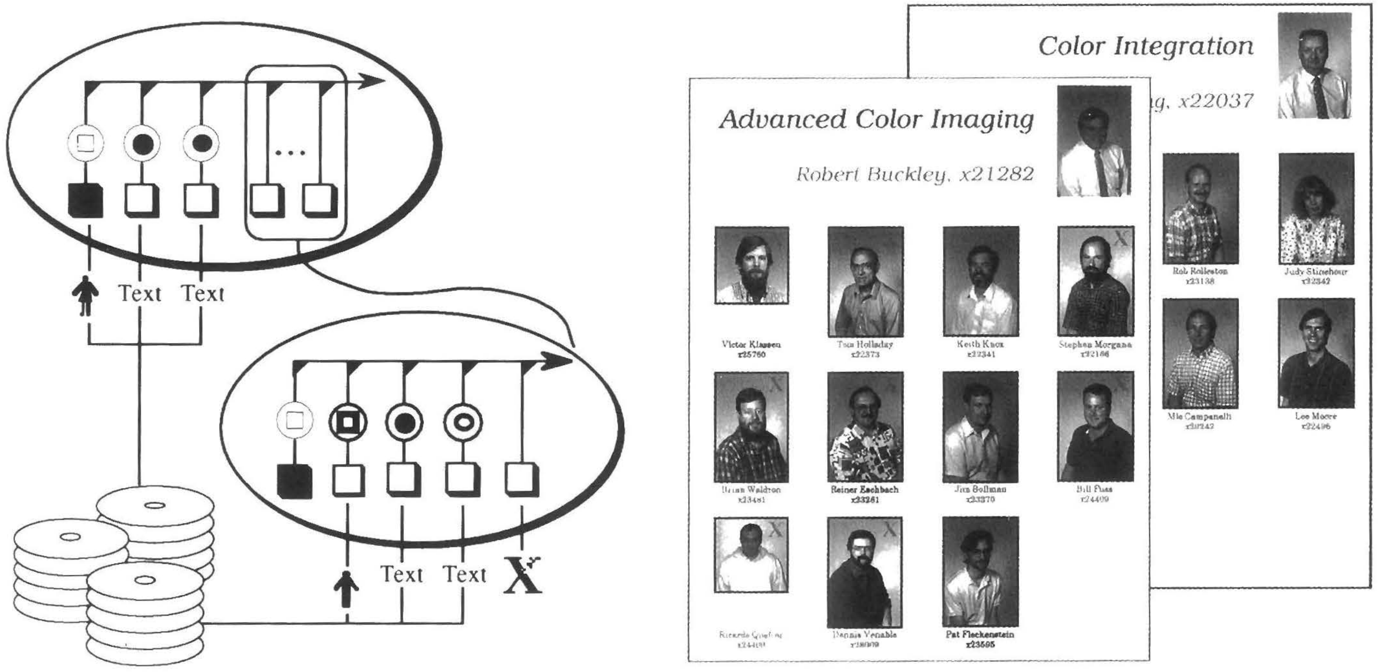
wrote file: "/local/venable/xtsiren/TACA/underconstruction.sid"

a)



b)

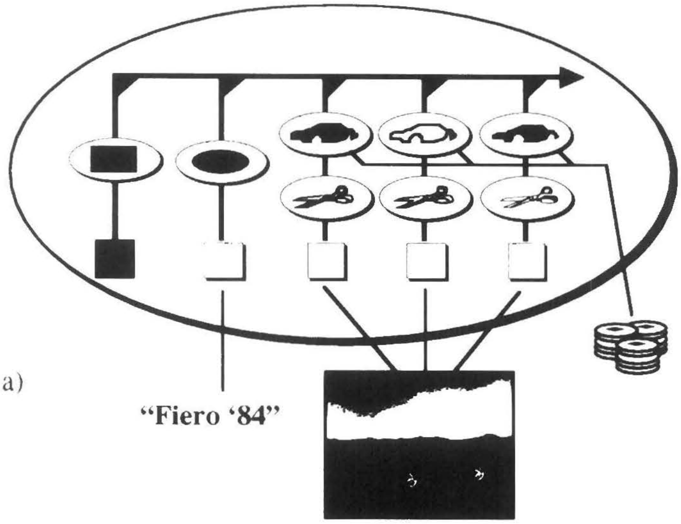
Figure 3. Structured Image-based Interactive Image Editing
a) SI interactive image editor
b) Output raster



a) b)

Figure 4. Variable Data Imaging: Variable Objects
a) Structured Image
b) Example renderings

a)



b)

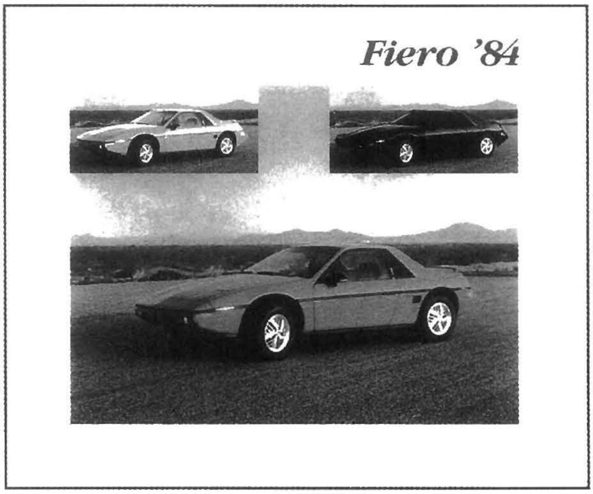


Figure 5. Variable Data Imaging: Variable Operations
a) Structured Image
b) Example renderings

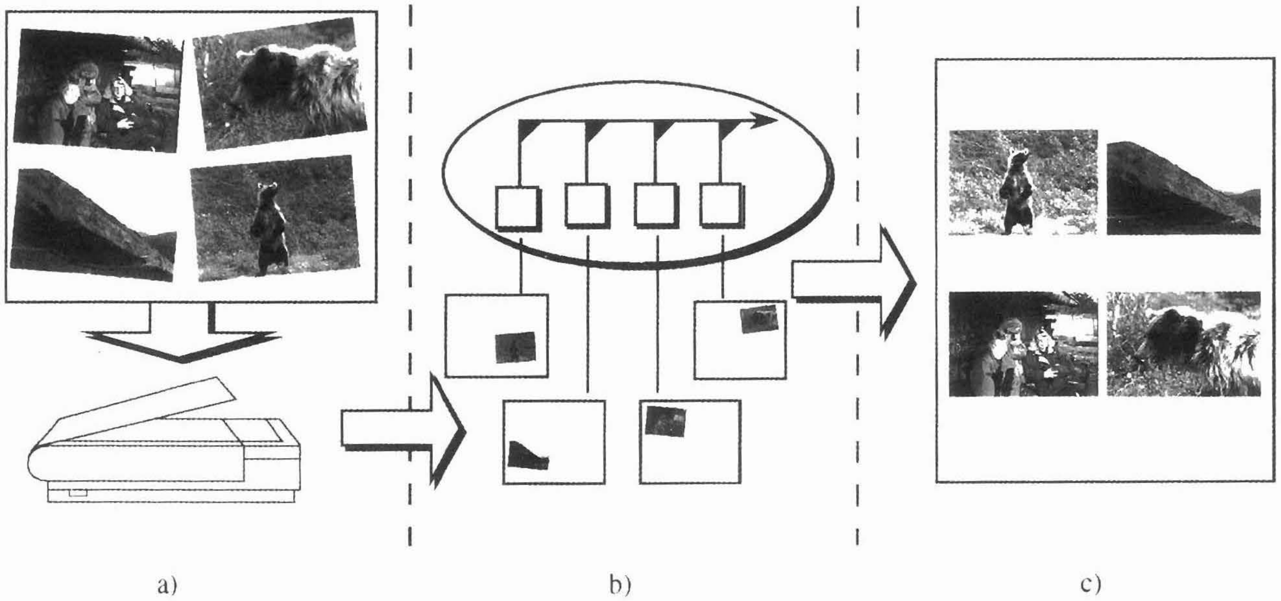


Figure 6. Gang Scanning
a) Original scanned image
b) Automatically generated Structured Image
c) Rendered Structured Image

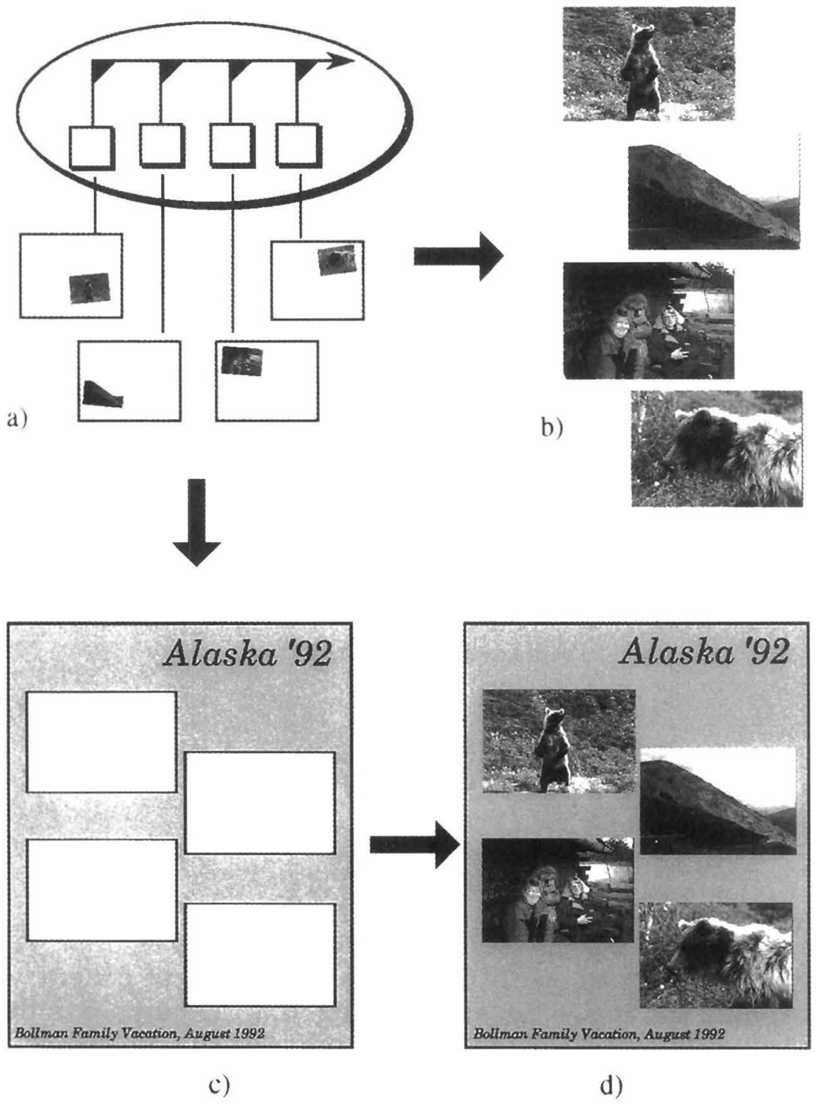

a)
b)
Alaska '92
Bollman Family Vacation, August 1992
Alaska '92
Bollman Family Vacation, August 1992
c)
d)

Figure 7. Gang Scanning
- a) Automatically generated Structured Image
- b) Extracted images
- c) SI template
- d) Result of template filtering