RC 21145(94534)21APR97 Computer Science / Mathematics

# Research Report

## PanoramIX: Photorealistic Multimedia 3D Scenery

**William L. Luken, James S. Lipscomb, Keh-Shin Cheng, William Gaddy, Walter Heger, Fred Ring, Jai Menon**
IBM T. J. Watson Research Center
P. O. Box 218
Yorktown Heights, NY 10598

# PanoramIX: Photorealistic Multimedia 3D Scenery

*William L. Luken, James S. Lipscomb, Keh-Shin Cheng,*
*William Gaddy, Walter Heger, Fred Ring, and Jai Menon*

**Abstract**

Conventional Virtual Reality and 3D applications, including those based on VRML, allow continuous movement of the viewpoint but pay a heavy price in a poor performance, crude 3D models, low quality images, and the need for expensive hardware. PanoramIX takes the opposite position. The viewer is fixed to one spot with travel possible only in occasional giant steps, but look-around performance is much better, the image looks better, and no special hardware is desired.

**Introduction**

While not limited to network-based applications, PanoramIX provides an especially effective means of delivering interactive 3d scenes over the World Wide Web. In addition to a stand-alone application for browsing panoramic scenes, PanoramIX is also available as a Netscape plug-in for Windows 95, Windows NT, and Apple Mac PowerPC. This plug-in also works with Microsoft Internet Explorer. AIX users can configure the PanoramIX browser as a helper application to experience high-performance virtual reality on the World Wide Web.

Panoramic scenes can be captured from a variety of sources including computer generated images as well as real-life photographs. PanoramIX accepts standard image file format data, allowing highly realistic images to be created from any of a number of advanced 3d modeling and rendering tools. Digital cameras as well as film cameras with wide-angle lenses can be used to capture real-life scenes. here is no need for special fisheye lenses. Standard image processing tools can be used to prepare digitized photographs for use by PanoramIX. Also, an interactive authoring tool is available for Windows 95 and Windows NT.

Hotlinks embedded in the panoramic scenes allow the viewer to jump from one panorama to another, for example to different seats in a stadium.

PanoramIX can protect your intellectual property by making available views of your data (e.g. a CAD model) without making the data itself available.

**Previous Work**

PanoramIX [PANO] is one of several viewers for panoramic images.

The plenoptic function describes everything that is visible from a point in space. UNC's plenoptic rendering uses the disparity between two points of view mapped to cylinders to extract depth by analyzing the optical flow along epipolar lines and to calculate and display occlusion and perspective effects.

Light field rendering captures many views of a scene organized to allow display from arbitrary camera positions without depth information or feature matching both for panoramic and for orbit-mode views. Display is performed by extracting 2D slices from a 4D function.

The Lumigraph is similar to the light field. It combines a 4D function of captured images with approximate geometric information to interpolate camera positions.

A multi-perspective panorama is a large flat image along which a rectangular viewing window is dragged. The large image is drawn in such a way that the view through the rectangular viewing window appears to be a combination of 3-D rotation, scaling, and zoom. The viewing window must be small compared to the large flat image or else the viewer will notice that object edges that one expects to be straight are actually slightly curved in order to accomplish the illusion of complex camera motion.

Creating panorama from individual rectilinear photos requires registering them to each other accurately to avoid seams in the final panorama. A Jacobian of matrices relating pairs of images can be fed to a minimization routine to accomplish this.

Other commercial panoramic viewers include QuickTimeVR from Apple [CHEN95a, CHEN95b, APPL], RealVR Traveler from Live Picture [LIVE] (was named RealSpace VR), SurroundVideo from Black Diamond [BLAC], PhotoBubbles from IPIX/Intel [IPIX] (was named Omniview), Jutvision from Visdyn Corp. [VYSD], OLIVER from OLiVR Corp. [OLIV], Warp TV from Warp Ltd. [WARP], SmoothMove from Infinite Pictures [SMOO], dubVR from dub Media Inc [DUBM], Surround View from Market Information Services of America, Inc.[SURR].

PanoramIX has nearly the union of the features offered by these other programs plus a simple, open, nonproprietary data format (ASCII and JPEG), Java and VB Script maps for easy navigation through multi-room environments, particularly easy installation, multiple instances on a page (a challenge in 256-color mode), rectilinear images allowed as input, streaming video and audio in the panorama, shared views between users, and z-buffered depth. PanoramIX is lacking compared to these others only a publically-available pure Java version, the multi-resolution zoom of Live Picture, and the pre-scripted viewing motions of PhotoBubbles. PanoramIX is available from the IBM Internet Media Group website [PANO].

**PanoramIX Architecture**

The principal elements of the PanoramIX architecture are illustrated in Figure 1. The central feature of this design is the PanoramIX Application Programming Interface (API). This is a set of 29 function calls which support all of the services needed by PanoramIX applications. These services include operations such as reading a PanoramIX control file, setting a view direction, creating a projected image, and placing an image in a window.

The PanoramIX kernel is written in procedural C code with a single source library for Win32*/Intel*, MacOS*/PowerPC*, and AIX* platforms. This code is compiled into a dynamic link library for the Win32 platform and an object library for the AIX platform. Other platforms have also been supported in the past, but have not been maintained due to resource limitations.

At least two-thirds of the object code comprising the PanoramIX kernel is represented by image libraries for tiff, gif, and jpeg files. These image libraries are very stable and require very little development effort beyond the initial porting work. The remaining portion of the kernel is comprised of a file i/o component, a view direction module, a panoramic projection module, a

window operations module, hot-link support modules, and small modules for handling exceptions and bookkeeping. All performance critical functions are performed by the PanoramIX kernel.

In principal, a wide spectrum of PanoramIX applications may be created based on the PanoramIX API. A simple prototype for such applications is represented by a module called the PanoramIX Browser. This module is analogous to an image file viewer or media player for audio or video files. Given the name of a PanoramIX control file, this module reads the control data and all constituent image files required to form a panoramic scene. It then manages all of the user interface interactions to enable a user to look around, manipulate objects in the scene, and act on hot spots imbedded in a scene. Like the PanoramIX kernel, the PanoramIX Browser has been developed for AIX*, Win32*, and MacOS* platforms. Additional details are provided in the next section.

The PanoramIX Netscape* plug-in, PanoramIX ActiveX* control, and PanoramIX panorama editor represent three additional examples of applications based on the PanoramIX API. Each of these includes a shared source C code library based on the PanoramIX Browser. In addition, each has a set of unique modules written in C, C++, or Java*. For example, the Netscape* plug-in includes a block of C code which forms an interface between the browser-based shared code and Web browsers which support Netscape* plug-ins. This allows the PanoramIX functions to be exploited by either Netscape Navigator* or Microsoft Internet Explorer*. The PanoramIX Netscape plug-in and ActiveX control can also interact with javascript or vbscript modules contained in HTML web pages.

**Panoramic Rendering**

The process of presenting panoramic images of a 3d scene consists of two main phases: data collection and image projection. In the first phase, data representative of a 3d scene is collected and stored in a suitable data structure. In the current implementation of PanoramIX, the data collection phase is performed with respect to a fixed point in the 3d space of the scene. This is called the view point for the data structure.

In the second phase, the contents of this data structure are used to construct an image of the scene appropriate to a specified set of viewing conditions. This is called the image projection phase. The viewing conditions may consist of an azimuth angle, an elevation angle, and a field of view angle with respect to the view point. These conditions may also include a tilt angle representing the angle between the vertical axis of the image and the axis used to define the azimuth and elevation angles.

These two phases are very asymmetric because The first phase is performed once for the scene, but the second phase must be repeated many times as the viewing angles are varied. Consequently, the second phase is very performance-critical and must be implemented very efficiently. The first phase, however, is much less sensitive to performance.

There are a wide variety of possible choices for the data structure forming the link between the data collection phase and the image projection phase. For example, this structure may consist of the colors representing each direction defined by a uniform grid of azimuth angles and elevation values. In this case, the azimuth angles and elevation values are defined with respect to the axis of a cylinder centered on the view point. This is called the standard or cylindrical

projection mode. In this case, the scene is projected onto the surface of a cylinder in the data collection phase, and the resulting cylindrical data structure is used to create images of the scene in the image projection stage.

The use of a cylindrical data structure for the scene is very advantageous for the image projection stage, at least as long as the elevation angles for the projected images remain relatively small. In this case, the image projection phase may be implemented in two stages.

In the first stage, the values in the cylindrical data structure are projected onto a planar surface parallel to the axis of the cylinder. Each column of the resulting image corresponds to a column within the cylindrical data structure. This stage is sufficient to account for the horizontal parallax in the projected image.

In the second stage, the image created in the first stage is projected onto a second image perpendicular to the view direction. In this stage, each row of the second image corresponds to a row of data values in the first image. This second stage accounts for the vertical parallax resulting from non-zero view elevation angles. Although the treatment of vertical parallax is essential for creating accurate images for non-zero elevation angles, this can be neglected in exchange for improved performance, especially when the view elevation angle is within roughly 20 degrees of horizontal.

As an alternative to the cylindrical data structure, it is possible to consider a spherical data structure. However, it is not possible to define a uniformly distributed set of sample points on the surface of a sphere, and spherical coordinates, like cylindrical coordinates, require identification of a special "z-axis". That is, the x, y, and z axes cannot be treated equivalently even though they are all the same to a sphere.

Instead of using a spherical data structure, we have found it advantageous to use an octahedral data structure to represent a nearly uniform set of sample directions distributed over all possible directions. This data structure is composed of eight triangular data grids each representing one octant of a sphere. Within each triangular grid, the first row has one element, the second row has two elements, and so forth. That is, each triangular grid To has (n+1)*(n+2)/2 elements To(i,j) where i=0 to j, and j=0 to n. Within the first octant T0, the data value(s) associated with element T0(i,j) include the color representing direction (x0=i, y0=j-i, z0=n-j). For each of the other octants, element To(i,j,k) represents the direction (xo, yo, zo)=(+/-x0, +/-y0, +/-z0).

The resulting octahedral data structure forms the basis for the "spherorama" projection mode in PanoramIX. In this case, the color representing any direction vector (x,y,z) is determined by element (i,j) of the grid To for octant "o", where the octant is identified by the signs of x, y, and z, and the values of i and j are determined by the absolute values of x, y, and z as

$i = n*x/(x+y+z)$, and
$j = n*(x+y)/(x+y+z)$.

Although the spherorama projection mode cannot match the performance characteristics of the cylindrical projection mode, it allows all axes (x, y, and z) to be treated symmetrically. This makes it possible to look in any direction, including straight up and straight down. In addition, there is not additional penalty for allowing the view orientation to be tilted.

**Background Composition**

Like any multimedia authoring or content creation process, the creation of PanoramIX data sets involves several steps and requires suitable content development tools. Because PanoramIX is based entirely on standard image file formats (TIFF, GIF, and JPEG) and plain ASCII text files, it is possible to create fully functional PanoramIX data sets using any ASCII text editor and any of a wide choice of image processing utilities which are available for various platforms. Although this is possible, it is not necessarily easy, efficient, or convenient. Consequently, a versatile authoring tool has been developed to accelerate the creation of PanoramIX data sets.

In the case of PanoramIX data sets, the first step in the authoring process consists of composing the background. This may be followed by adding optional embellishments such as animated overlays and hot spots. These optional features are considered in the following section on advanced features. The background composition process comprises the first phase in the panoramic rendering process: collecting data representative of a 3d scene and placing this into a suitable data structure.

There are many ways to collect data representing a 3d scene. These may be divided into the following categories:

1. Computer generated images
2. Images obtained from standard film cameras with rectilinear lenses
3. Images obtained from standard digital cameras
4. Images obtained from professional panoramic cameras
5. Images based on nonlinear optics such as fisheye lenses and curved mirrors.

In the case of computer generated images, the author must first create or acquire a 3d digital model of the geometry of the scene. There are a wide variety of 3d modeling tools that can be used to create such 3d models. Once the model has been established, one must select the view point and render the scene several times to create a set of images based on the selected view point. This is most easily accomplished with a set of six images representing the sides of a cube centered on the view point. This set of images is created only once for a particular view point, and the time required to create these images has no effect on the performance of the subsequent panoramic image display operations. Consequently, it is best to use 3d rendering software such as Photorealistic Renderman or 3D Studio Max which deliver the highest degree of realism in the resulting images. It is also important to ensure that all six images are rendered with identical conditions. For example, one must not use any virtual light sources which move with the viewer or virtual camera is rotated.

In the case of a real scene, it is much easier to capture the scene with a real camera than it is to create a 3d model of the scene. In this case, however, one must deal with a number of limitations inherent with real cameras as opposed to a virtual camera. The first problem with a real camera is the limited field of view. In the case of a standard 35mm camera, this is determined by the focal length (l) of the lens. Assuming the lens is focused on infinity, the horizontal field of view is given by

$\theta_h = 2*\text{atan}(w/2l)$,

where w is the width of the image (36mm). Likewise, the vertical field of view is given by

$\theta_v = 2*\text{atan}(h/2l)$,

where h is the height of the image (24mm). Consequently, a 35mm camera with a standard 50mm lens captures a horizontal field of view less than 40 degrees and less than 30 degrees in the vertical direction. A set of 12 images captured with such a lens would be sufficient to cover 360 degrees in 30 degree intervals with adequate overlap between adjacent images, but the small vertical field of view is generally inadequate.

A wide-angle angle lens with a focal length of 28mm is capable of capturing images with a horizontal field of view of more than 60 degrees and a vertical field of view of more than 45 degrees. Such a lens may be used to capture a 360 degree scene with 8 images in 45 degree intervals. This is satisfactory as long as one does not require top and bottom images for a "spherorama" composition. In order to obtain sufficient overlap between the side images and a top or bottom image, it is necessary to use a lens with a focal length of 14mm or less. Such lenses are considered to be extreme wide-angle lenses.

Unlike a virtual camera, it is impossible to aim a real camera exactly in a specified direction, or to rotate a real camera by an exact angle about a precise axis. The errors resulting from imperfect aiming and rotation of a real camera can be minimized through the use of suitable camera mounting apparatus. Such equipment is called a "pan head". Even with a good pan head, the images captured with a real camera will suffer from errors caused by imprecise orientation of the camera. These errors include small rotation about the direction of view, inexact rotation of the camera about the vertical axis, and rotation of the camera about the horizontal axis defined by the direct product of the vertical axis and the view direction.

Additional errors may be introduced when the images are printed and scanned. For example, film processing laboratories usually crop away varying amounts of each image. Consequently, the center of the resulting print no longer coincides with the center of the original image. Imperfect orientation of a photograph in a scanner causes the scanned image to be rotated. Rotation of an image with a resolution of 640x480 by ¼ degree, for example, can cause a 3-pixel shift between the left and right edges of the image.

Another source of errors arises from exposure variations. These can arise when the photographs are captured, when they are printed, and when they are scanned.

The process of combining a set of scanned photographs into a seamless panorama is called "stitching". This includes projecting each image onto a cylinder, compensating for all of the errors introduced in the image capture process, and blending the overlapping portions of adjacent images. The PanoramIX authoring tool includes facilities for handling all of these operations. These include automatic alignment of adjacent images based on pattern matching, and automatic compensation for exposure variations.

The process of stitching a set of photographs into a seamless panorama is summarized in the following figures. The initial data consists of a set of six to twelve overlapping digital images. The first two images in such a set are shown in Figure 2. The first step in the stitching process

consists of projecting each of these images from a flat surface to the surface of a cylinder centered on the view point. The flat surface occupied by each raw image is tangent to the surface of this cylinder, and the cylinder has a radius (R) determined by 2*pi*R = Wpan, where Wpan is the number of columns in the resulting panoramic image. The center of each image is thus located at a distance R from the view point, while the distance from the view point to the left and right edges of each image is given by

$D(\theta_h) = R*\sec(\theta_h / 2)$,

which is greater than R by a factor of $\sec(\theta_h / 2)$. Consequently, the effect of projecting this image onto the cylinder is to reduce the height of the left and right edges by a factor of $R/D=\cos(\theta_h / 2)$. The intermediate columns are likewise reduced by a factor of $\cos(\alpha)$ where $\alpha$ is the angle between each column and the center of the image. The result is to warp each image into the forms shown in Figure 3.

After the images have been projected onto a cylinder, the overlapping portions of adjacent images can be can be merged and blended together to form a seamless composite, as shown in Figure 4. The optimum overlap can be determined by shifting the image on the right until the differences between overlapping pixel color values has been minimized. This shifting must include both horizontal and vertical motions to compensate for errors in the camera orientations.

This pair-wise matching of adjacent images is repeated until the last image has been blended with the previous image. At this point, the right portion of the last image should match the overlapping portion of the first image, but a perfect match will be prevented by errors in the image orientations and estimated fields of view for each image. Comparison of the overlapping portions of the first and last images can be used to determine corrections for these errors leading to a seamless panorama for all image pairs including the first and last image.

After the images have been stitched together to form a panoramic image, the resulting cylindrical image can be projected back onto a flat surface tangent to the cylinder at any azimuth. This makes it possible to determine accurate representations of what would be seen in any direction. A possible result is shown in Figure 5 which shows an image representing a view direction intermediate between the directions used to capture the two raw images shown in Figure 2.

Capturing a scene with a digital camera is similar to capturing a scene with a standard film camera except that one does not need to wait for the images to be processed, printed, and scanned. Instead, the digital images come straight out of the digital camera. In addition, one does not need to worry about errors and artifacts arising from the printing and scanning processes. On the other hand, it remains nearly impossible to obtain digital cameras with suitable wide-angle lenses, especially for the cameras costing less than US$1000. The usage of these cameras is currently increasing very rapidly, and the availability of suitable lenses can be expected to improve greatly within the next year.

A professional panoramic camera is a specialized device that captures a 360 degree image while rotating about a vertical axis. Such devices have been in use for over 100 years and there are professional photographers who specialize in the operation of these devices. The resulting images are captured on photographic film which must be processed, printed, and scanned to obtain a digital panoramic image. The resulting image has only one seam where the ends meet. The

PanoramIX authoring tool has special functions for aligning these ends and creating a seamless panoramic image. Professional panoramic cameras must not be confused with low-cost consumer market cameras which claim to support a "panoramic format". Such cameras have nothing to do with genuine panoramic photography.

It is also possible to capture a scene using a fish-eye lens (see PhotoBubble, IPIX). Two such images, each covering 180 degrees, can be sufficient to capture an entire scene. In addition, several devices have been developed which employ curved mirrors to capture an entire scene in one image (e.g., BeHere). The resulting image is processed, printed, and scanned to create a high resolution digital image. Appropriate software can then be used to convert the resulting raw data into a panoramic image similar to that obtained from a digital camera. The resulting panoramic image is perfectly suitable for use by PanoramIX. This method is especially useful for capturing scenes with many people, traffic, or other moving objects.

**Advanced Features**

Progressive Display : Modem web connection is too slow for download of only the finished panorama, which even JPEG compressed comes to a minimum of 50-100 KB. (30-60 seconds realistically on a 28.8 modem). That is a long time to look at nothing. Some panoramic viewers use progressive JPEG, but this produces a distracting wipe of increased resolution that crawls down the screen. PanoramIX downloads first a complete low-resolution JPEG, which the user can navigate while the high-resolution JPEG downloads. Following download there is a switchover to the second image. PanoramIX implements two switchover schemes. A low-memory-usage scheme freezes the first image for a few seconds while the first picture is destroyed and then the second one is formed. A high-memory usage scheme builds the data structure for the second image completely while the user continues to navigate the first image. We are evaluating which is the better tradeoff but tentatively favor the high-memory-usage scheme, because the memory penalty is small owing to the small size of the first low-resolution image.

Hotlinks : Double-clicking on certain hot areas of the panorama cause actions. Actions can be jumps to other views, commands to bring up text in adjacent web-browser frames, sound, etc. Flexibility here is enhanced by the simple ASCII control file, which the author can edit. PanoramIX uses exclusive-OR to shade the hot areas at the user's request as a preview of where things can happen. This gives the full outline of the area with low memory overhead, but performance was a challenge. To save storage the map of hot areas is stored at a lower resolution than the full panorama, which allows obvious opportunity to speed the inner loop that searches out the hot areas for preview.

Sprites [PATR] are 2D animated images that are overlaid onto the panoramic scene to provide motion or more detail. These can be manipulated by the mouse. First the panorama is drawn in the viewing window, then the sprite is overlaid in a separate drawing loop. Chroma-keying allows the sprite outline to that of the embedded object, not the enclosing rectangle. Chroma-key can also be applied to the panorama to put the sprite apparently behind the panorama.

Streaming Temporal Media: Bamba, an IBM streaming video technology [BAMB] can be embedded in the panorama using the same procedure as with sprites. For example chroma-keying the panorama can make the video appear to be behind the panorama [AIRC].

Direction-sensitive Bamba audio

External Authoring Interface: The Java Interface, also known as the External Authoring Interface (EAI), allows PanoramIX to communicate with any Java or Javascript objects on the web page interactively. The Interface implements an Observer-Observable model to allow other objects to listen on the PanoramIX events, such as viewing direction, field-of-view, or entering a new scene. It also allows other objets to manipulate PanoramIX, such as changing viewing direction, field-of-view or scene.

The Java Interface is available to both the PanoramIX plug-in and the PanoramIX Java Applet/standalone. In the case of plug-in, it is based on Netscape's LiveConnect technology. In the case of Java Applet/Standalone, it is based on Java Native Interface (JNI).

Collaborative Component: Two or more PanoramIX sessions may share viewpoints and other control attributes with one another via a PanoramIX server running somewhere. Via the same server, PanoramIX sessions may also collaborate with other PanoramIX-related applications such as a FloorMap applet.

The PanoramIX Collaborative Component is just another events observer of PanoramIX. At initialization, it establishes socket connection with the PanoramIX server. Then it listens on PanoramIX events and sends them to the server. In the meantime, it also listens for events from the server and calls the appropriate PanoramIX Java interface methods to handle them.

Java Applet: The two alternative approaches are: (a) Java wrapper that blits images and traps mouse events, retaining a core compute-intensive kernel in native code, and (b) pure Java.

Currently a JDK1.1-based PanoramIX Java Applet/Standalone with native kernel exists. With the exception of the performance-critical image rendering code, the applet/standalone is written in pure Java. The image buffer is allocated in Java and passed to the native code via JNI. After the image has been rendered and ready to be displayed, the native code send a message to the applet/standalone which then draws the image on the applet/frame window.

An effort to rewrite PanoramIX kernel in Java is under way. Although Java performance is a concern, PanoramIX image rendering code may particularly benefit from the Java just-in-time compilers and JDK's HotSpot Java Virtual Machine.

Cross Platform Implementation: PanoramIX is supported on several platforms: Windows 95/NT, Unix (AIX, Irix), Macintosh, and Java machines (Network Stations and such). Notable architectural modifications as well as development of one of the first-ever ActiveX controls for the Macintosh..

Implementing these on the Macintosh platform from a source code base shared with the Windows/Intel platform presents many challenges; among them differing memory management models (no protected memory model and a separate user-sized heap for each application) and user interface details. An additional complication is that in contrast to the Intel 80X86 and Pentium series, the PowerPC processors in typical use on the Macintosh favor floating-point calculations and are less-favorably oriented to memory-intensive loads and stores. This requires that the PanoramIX projection code be hand-optimized for each platform to minimize cache defeats and

tune performance, since refresh frame-rate is a critical user-perception issue. Furthermore, many Macintosh web platforms such as Netscape, Internet Explorer/ActiveX, and Java are not afforded the same testing and quality control as their Windows/Intel counterparts. Ensuring reliability on the Macintosh frequently involves coding around problems inherent in other released software. This was especially true developing PanoramIX support for interaction with scripting and control APIs, such as LiveConnect/JavaScript with the Netscape plugin, VBScript/JavaScript with the ActiveX control, and the Macintosh Java environment in general.

**Software Delivery**

Netscape Navigator requires a plug-in. One problem with most install executables is that they cannot find all versions of Netscape Navigator or Internet Explorer on a system, because the Windows Registry can have a reference to only one of each. We bypass the registry, searching the disks exhaustively, something that all install executables should do. This eliminates install failures.

Internet Explorer can simplify installation for the user by using an ActiveX control.

An OLE control is a dynamically linked library which exposes several standard interfaces to an external component, the OLE container. These interfaces allow the container to manipulate the OLE control during run-time and allow the updating of controls without having to recompile or otherwise change the OLE container. An ActiveX control is a simplified OLE control for Internet apps which is only required to support at least one interface (which handles the lifetime of the object) and has the ability to register itself on a user's platform. The ability to self-register implies that an ActiveX control can be installed on a user's computer by accessing the appropriate web page, provided that the user allows the download. The ActiveX control is compiled on several platforms and delivered by the server onto the user's platform based on HTML code which interrogates the browser to find out which one is running.

An ActiveX control was built for the purpose of allowing a user to simply access a home page and to have an component be downloaded on the platform without requiring the user to disrupt the running session of Internet Explorer and to allow two-way communication with the container. The PanoramIX ActiveX control was built from a Microsoft components library called BaseCtl which provides an application framework for ActiveX controls. Additionally, we added a new interface, which exposes methods to the external system which allow external manipulation of our object. In this way, we were able to allow programmatic control from Web pages through a scripting language, or any other future container which recognizes our interface. Typically, ActiveX containers are MS Internet Explorer and Lotus Notes.

The ActiveX object in the HTML page references an ".inf" file which in turn is keyed to the platform the user is running and which component to download. The inf file specifies a version number which is used to compare against the current version of the ActiveX control last downloaded on the user's platform. If the version number on the user's platform is less than the version number specified in the file, the latest version is download. In this way, one ".inf" file can be used to download either the WinTel version or the Macintosh version of the ActiveX control. Use of other keywords in the HTML page are used as indicators to which component to download, the PanoramIX ActiveX control or the Netscape Navigator plug-in.

## Applications

PanoramIX has a wide variety of applications, including virtual shopping malls, tourism, hotel reservations, mechanical CAD, and architectural design. Software developers can also create customized applications by using the PanoramIX Application Programming Interface (API) to exploit the PanoramIX kernel. For example, a network-based virtual reality game could control the properties of animated sprites or avatars in a panoramic scene based on the actions of multiple players.

Examples of PanoramIX on the web include the Air Canada [AIRC], Hilton Hotels [HILT], the Nagano Olympics [OLYM], and a view of Mars [MARS]. IBM's Internet-Media Group website also has some technology examples [EXAM].

## Conclusions

PanoramIX panoramic display program with an especially wide range of input and output options. The greatest interest from the marketplace seems to be in health-care facilities tours, travel and tourism, and merchandise display.


## Acknowledgments

We are grateful to Pierre Darmon for documentation, to Arnaud Debayeux for data preparation, and to Bengt-Olaf Schneider and Gabriel Taubin for management direction.


## Bibliography


[PANO] http://www.software.ibm.com/net.media

[PLEN95] McMillan, L. and Bishop, G. Plenoptic modeling, an image based rendering system. Proc. 1995 ACM SIGGRAPH Conf., Computer Graphics, v 29, #3 (Aug. 1995), pp. 39-46.

[LIGH96] Levoy, M. and Hanrahan, P. Light field rendering Proc. 1996 ACM SIGGRAPH Conf., Computer Graphics, v 30, #3 (Aug.. 1996), pp. 31-41.

[LUMI96] Gortler, S.J., Grzeszczuk, R., Szeliski, R., and Cohen, M.F. The Lumigraph. Proc. 1996 ACM SIGGRAPH Conf., Computer Graphics, v 30, #3 (Aug. 1996), pp. 43-54.

[MULT97] Wood, D.N., Finkelstein, A., Hughes, J.F., Thayer, C.E., and Salesin, D.H. Multiperspective panoramas for cel animation. Proc. 1997 ACM SIGGRAPH Conf., Computer Graphics, v 31, #3 (Aug. 1997), pp. 243-250.

[MOZA97] Szeliski, R. and Shum, Heung-Yeung. Creating full view panoramic image mosaics and environment maps. Proc. 1997 ACM SIGGRAPH Conf., Computer Graphics, v 31, #3 (Aug. 1997), pp. 251-258.

[CHEN95a] Chen, S.E. and Miller, G.S.P., Cylindrical to Planar Image Mapping Using Scanline Coherence, US Pat. No 5396583, March 7, 1995.

[CHEN95b] Chen, S.E. Quicktime VR -- an image-based approach to virtual environment navigation, Proc. 1995 ACM SIGGRAPH Conf., Computer Graphics, v 29, #3 (Aug. 1995), pp. 29-38.

[APPL] http://qtvr.quicktime.apple.com/

[LIVE] http://www.livepicture.com/

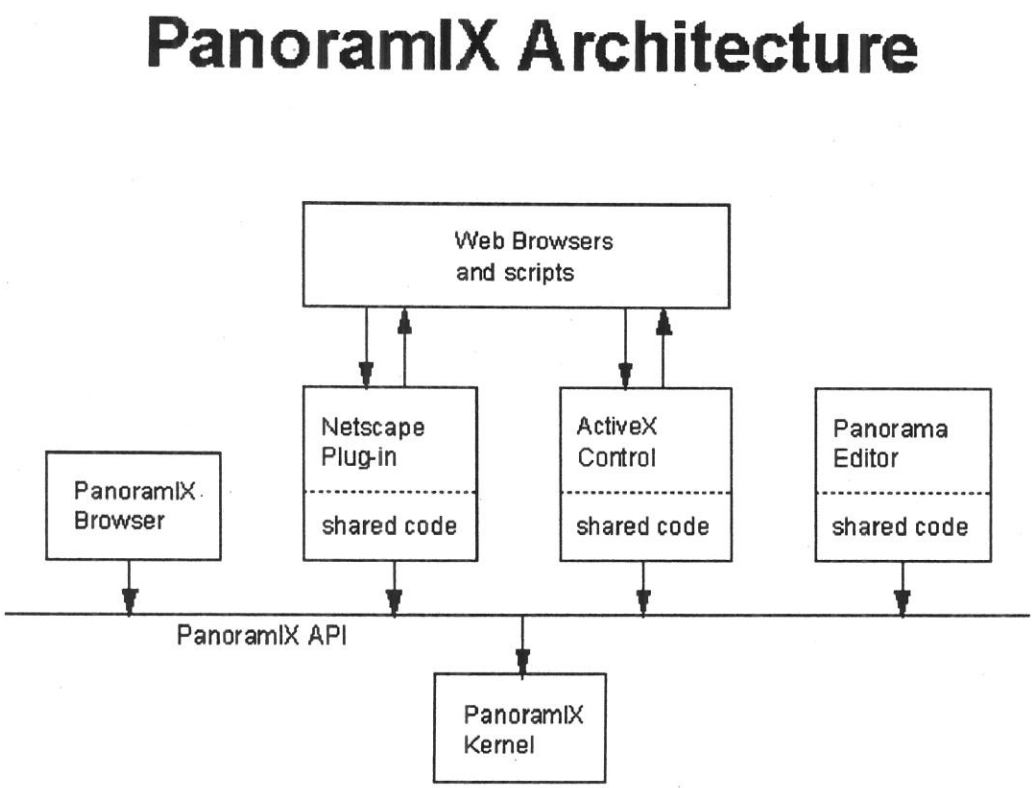
[BLAC] http://www.bdiamond.com /products/surround/default.htm

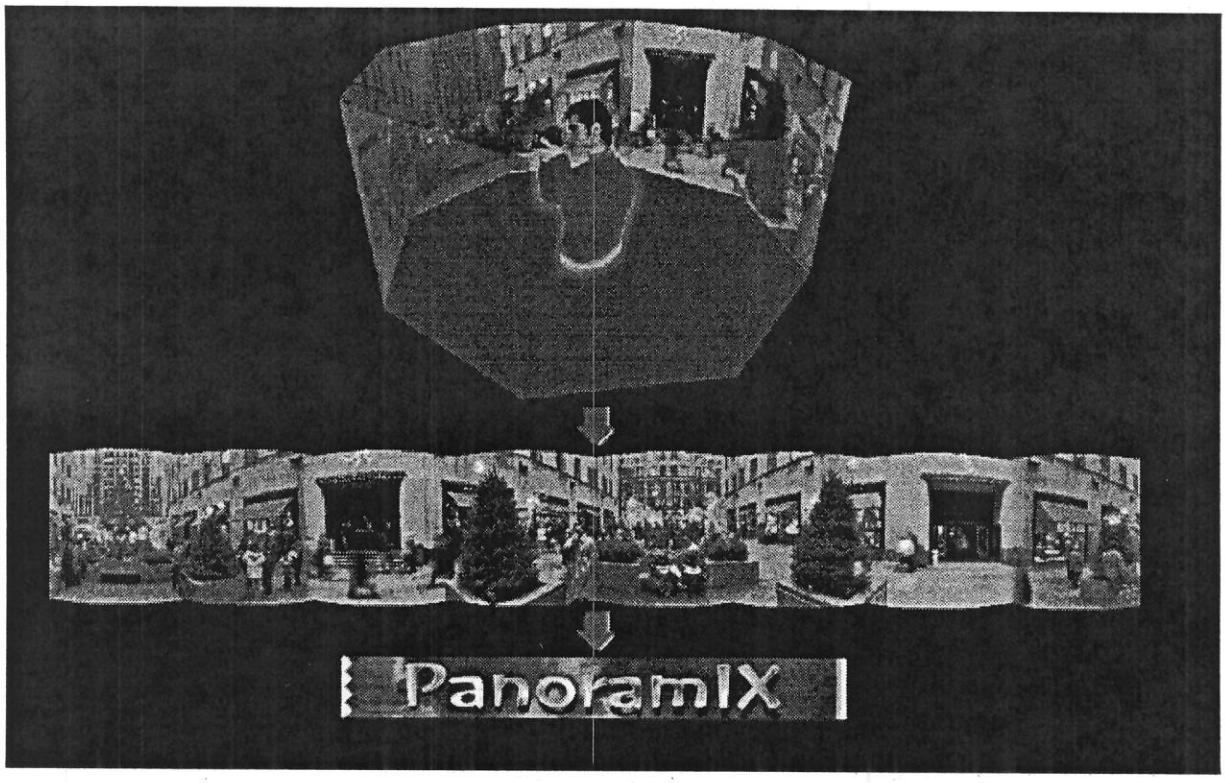
[IPIX] http://www.ipix.com/home/home.htm

[VYSD] http://www.visdyn.com/

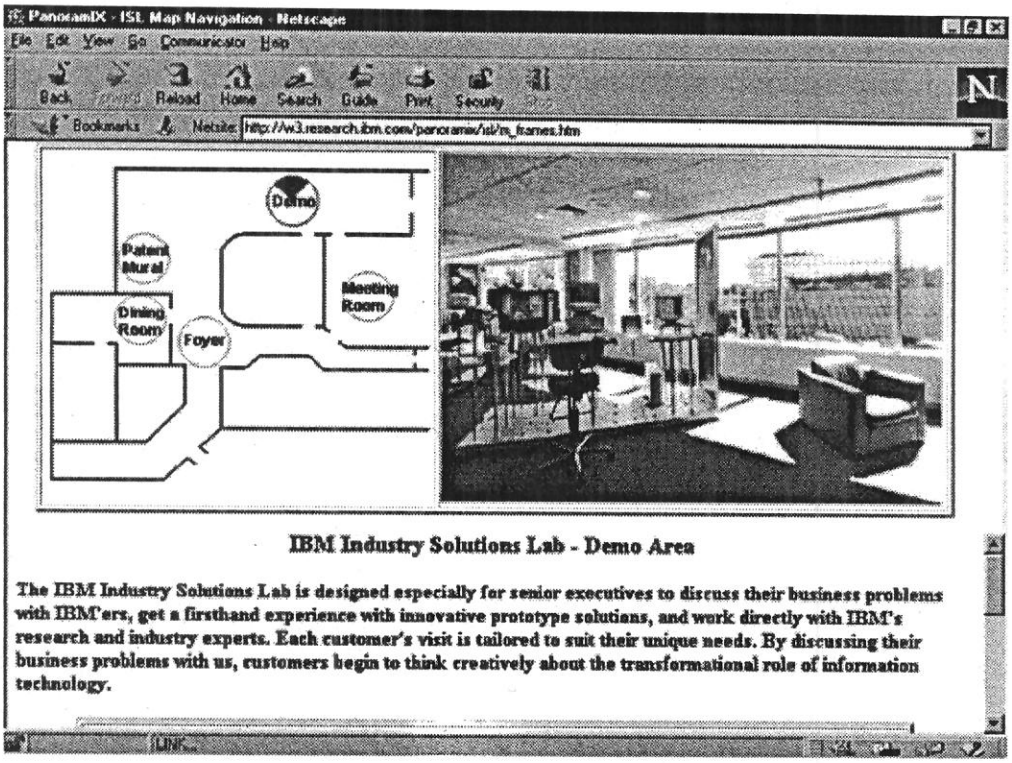
[OLIV] http://www.olivr.com/
[WARP] http://www.warp.com/
[SMOO] http://www.smoothmove.com/
[DUBM] http://www.dubmedia.com/
[SURR] http://www.misa.com/surround.htm
[PATR] http://adtech.internet.ibm.com/patrick/panoramixv
[BAMB] http://www.alphaworks.ibm.com/formula/bamba
[AIRC] http://www.aircanada.ca/about-us/virtual-worlds/
[HILT] http://www.hilton.com/panoramix/nyhilton/nyhilton.htm
[OLYM] http://www.nagano.olympic.org/goto/nagano/pan_e.shtml
[MARS] http://www.rs6000.ibm.com/resource/features/1997/mars/panoramix.html
[EXAM] http://www.software.ibm.com/net.media/panoramix/demos/index.html

**Figures**






**PanoramIX authoring from a sequence of images stitched into a panorama**




**PanoramIX browsing, showing Java map that gives view direction and field of view**