Extended View Toolkit

Peter Venus
Alberstrasse 19
Graz, Austria, 8010
mail@petervenus.de

Marian Weger
Krenngasse 45
Graz, Austria, 8010
mail@marianweger.com

Cyrille Henry
France
ch@chnry.net

Winfried Ritsch
Institute for Electronic Music and Acoustics
Graz, Austria, 8010
ritsch@iem.at

Abstract

Extended View Toolkit is a set of PD/GEM abstractions for combining multiple video or image sources into a panoramic image or video as well as for projection setups with multiple projectors or projection environments with challenging geometric forms.

It features a set of abstractions that are able to combine multiple related image-sources (like video input, video playback) into a consistent panoramic image.

The toolkit also contains abstractions to create multi-screen, multi-projector environments to enable immersive representation of the created panoramic material.

Keywords

panoramic, video, stitching, projection, mapping

1 Introduction

The abstractions for the Extended View Toolkit have evolved around the idea to create a low-cost, DIY panoramic-video camera to create content for a media-installation featuring a screen that was wrapped around the audience by 270 degrees [1], which was then further developed and expanded during a second installation.

The idea of creating panoramic images is nothing new, even the creation of video with a wide viewing angle has a history going back to the seventies. But using just common, cheap consumer electronics and open-source software, in this case webcams and Puredata, for the task of creating moving images in panoramic view seemed a good challenge and also a good starting point for others to experiment with the possibilities of this extended field of vision.

The primary goal for the toolkit was to provide a basic set of tools to fulfill the task of combining multiple image-sources into one continuous image and a second set of tools, that enables the user to create immersive projections.

The whole toolkit is structured in a modular way to make it easily adoptable for users using different setups of cameras or projection environments, rather then having a complex patch that is hard to understand in its functionality and inner structure.

It can be divided into 3 groups of abstractions, where one group contains a set of tools to process various input-sources, a second set that that provides the necessary functionality to organize the projection of content and a third set, that helps with different tasks in the organization of the content to be projected.

Furthermore, all the abstractions feature a message-system that allows complete control of every parameter via OSC and also a preset storage system.

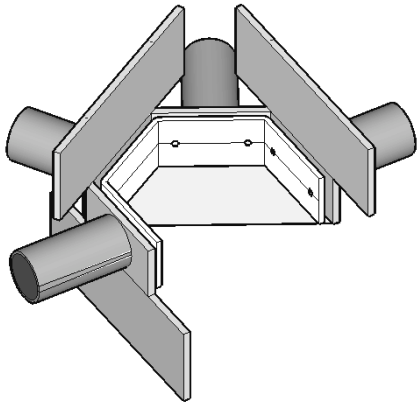
2 Input processing

In this section we want to describe the mechanisms to stitch[1] multiple image-sources into one continuous image. We will give a short introduction on the problems that occur when stitching images and describe the solutions we are using in the toolkit.

2.1 Image stitching

There are a couple of ways to create images with a wide viewing angle. The most common ones are built in digital cameras and mobile phones nowadays already and involve taking a series of pictures while turning the camera, that later get analyzed and combined with the help of complex algorithms. The results of this are quite impressive, but do not allow shooting video. Another common solution would involve a spherical mirror or a fisheye-lense, both of which are suitable for shooting video, but involve a manageable but heavy distortion in the resulting image.

1 The process to combine multiple related image sources into one continuous (panoramic) image is known as “stitching”



Picture 1: camera setup

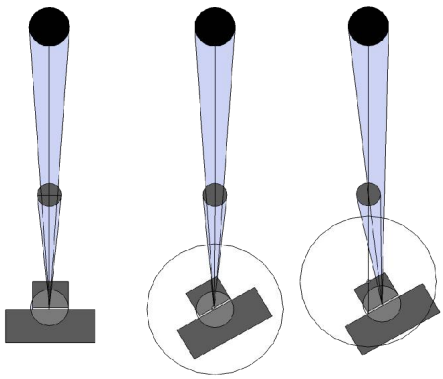
We chose to build an array of cameras, arranged on a circle and aligned horizontally, because in theory we would end up with a higher resolution and the single image sources are not so distorted.

2.1.1 Problems

When panoramic image material is created from rotating cameras or cameras aligned on a circle, some problems have to be taken into account. Two of those are parallax errors and to align the material to form continuous horicontal lines.

2.1.2 Parallax error

Parallax describes the relative change in the position of an observed object introduced through the shift of the observer. If a camera is rotated around its optical center, images can be stitched together quite easy, since they have similar projective properties [2]. Picture 2 tries to illustrate the problem of camera rotation around the camera centre (in the middle) and off-centre rotation (on the right). It can be seen, that the two objects are still in line when the camera is rotated around its centre.



Picture 2: camera rotation

The parallax effect is most obvious for objects close to the observer, but has far less impact on objects farther away.

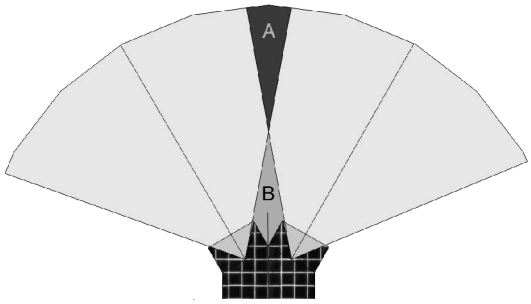
This problem is of great importance for panoramic video and image creation, since we rely on sources that overlap in a portion of its visual information, where the images can be stitched together.

Now, if a couple of cameras are setup on a circle horizontally, it is nearly impossible to align the rotational axis on the camera centre due to their construction size.

To reduce the impact of the parallax on the final image material and keep the construction, a little trick has to be applied:

If one reduces the overlap between the single camera modules to a minimum, the resulting image-sources would still share a portion of information with each other, marked with “A” in picture 3, exactly this portion, where ideally the parallax has much less effect: the information in the distance.

Objects that are closer and in between the optic field of two camera modules (B in picture 3) simply are not shown at all, much like a blind spot of a car side mirror.



Picture 3: camera modules; overlap in between

This leads towards a tradeoff: the smaller the overlap is, the better the stitching works, but the farther the observed scenery has to be away to record continuous movements along the whole visual field of the camera array.

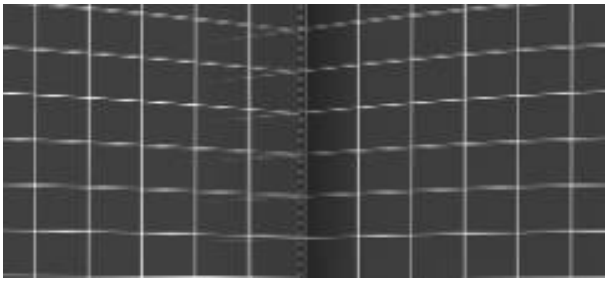
Through experimentation we found out, that, with an angle of 60 degrees between the camera modules[2], a minimum distance of 2 to 3 meters for the observed scenery shows acceptable results, given that the radius of the circle, the modules are arranged on is as small as possible.

2.1.2 Lens distortion

The second aspect that introduces problems when combining the image or video-sources together, is the fact, that most wide angled lenses cause radial distortion, which becomes visible especially towards the edges of the captured image in the form of

2 Sony ps3eye webcams, 75° opening angle

originally straight lines appearing curved. If this problem is not solved, combining images together to form a panorama would result in blur in the area, where the images overlap.



Picture 4: blurring in overlap area of 2 images (detail)

Most of the time, one has to deal with two relatively simple forms of radial distortion:

Picture 5: lens distortion *Picture 6: fixed distortion*

One, where straight lines are bend away from the image center (barrel distortion), and one where they are bend towards it (pincushion distortion) [3].

The solution implemented in the toolkit to solve this problem, is way simpler. The shader assumes, that the image-material is taken with a perfect lens, therefore being completely planar with no distortion. Since the cameras are aligned on a circle, the resulting image should be projected on a cylinder. The shader parameter adjusts the cylinder distance for every image to adjust the continuity between the images. This solution, although not being perfectly correct, helps solving the problem introduced through the lens-distortion as well as allowing continuous horizontal alignment of the image-material.

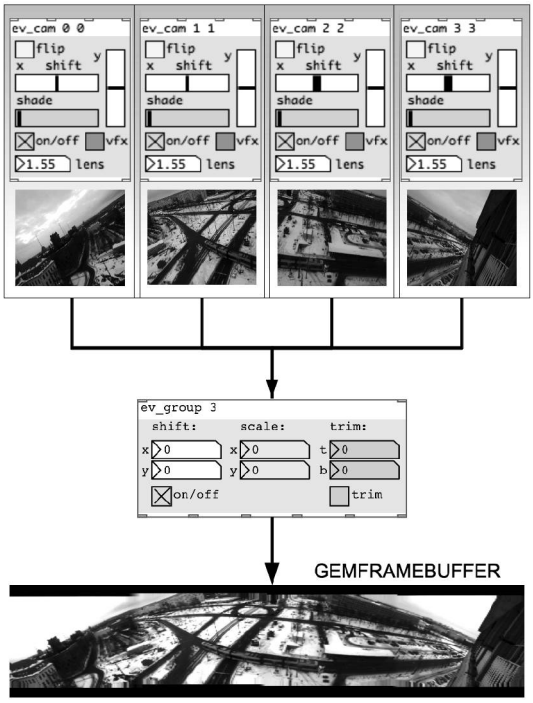
2.2 Stitching images using the toolkit

The Extended View Toolkit contains three different input-abstractions, of which all have the same functionality and just differ in their accepted input-source. There is one for opening images, one for loading video-files and one for processing a live-input from a camera.

The idea is, that for every source, e.g. camera, a new abstraction is loaded, so that in the case of four camera-sources, four camera-input-abstractions are created, which then allow to stitch the four sources together.

The stitching process has to be done manually, but is quite straight forward. Every input-abstraction has control parameters for horizontal and vertical alignment of nearby image-sources, distance and softedge blending.

Once the videos or images are aligned horizontally and vertically, the distance is set right, the softedge can be applied to blend between the adjoined images. The softedge (shader-based alpha blending) just applies on the left side of the video or image source.



Picture 7: image stitching process

The group abstraction in picture 7 allows the stitched video or image to be further treated as one unit, enabling the user to scale, resize and shift the whole without having to re-adjust stitching parameters. It also enables masking of the overhanging images from top and bottom, to get one cohesive rectangular panorama image or video.

3 Output processing

For enabling an immersive experience to the observer, panoramic video is proposed to be thrown on a screen that is surrounding the observer.

The following part outlines a basic approach for creating immersive projection environments as well as video mappings on geometric objects.

3.1 Problems

For completely covering a screen that encircles the observer, analogue to the extended view camera, a multi-projector-system is chosen.

For lower costs and easier construction, the immersive screen does not need to be integral, but can be made of multiple, individual screens. The resulting construction frames make rear projection impossible, as they would induce shadows on screen. To get as little shadows as possible from the

observers, the projectors can only be placed in the center, above their heads.

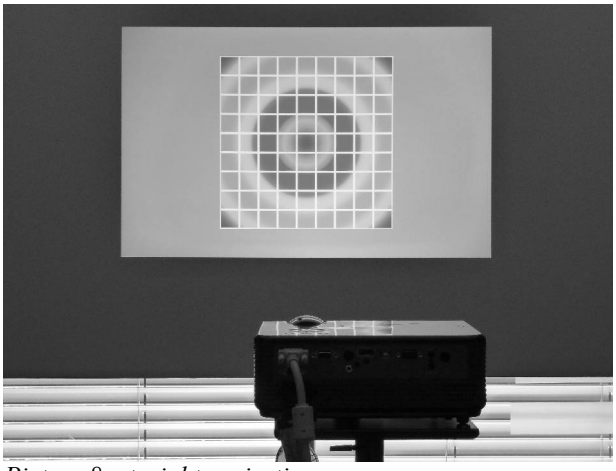
Now, another problem occurs: It is not always possible to mount the projectors on their ideal positions. And since the toolkit aims at low-budget solutions, we cannot rely on trapeze adjustment, lens-shift and zooming capabilities of the equipment, or on even, integral and flat walls to project on.

The outcome is, that all these features must be implemented in software.

3.2 Video projection with GEM

3.2.1 Straight projection

The most basic projection system consists of one projector, throwing a picture perfectly straight on screen.



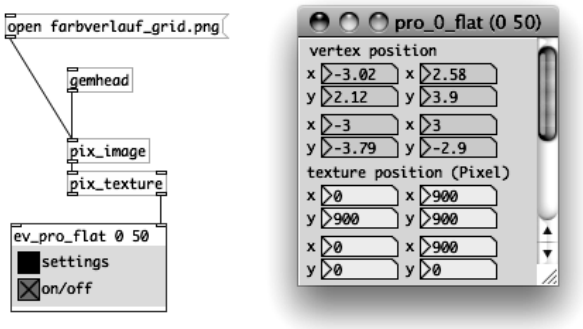
Picture 8: straight projection

3.2.2 Angular projection

By not perfectly positioned projectors, the resulting image on screen gets geometrically distorted.

Picture 9: angular projection

To compensate these deformations, the image must be distorted in the opposite direction, before it is thrown on screen. For enabling this, the projection module is being introduced.

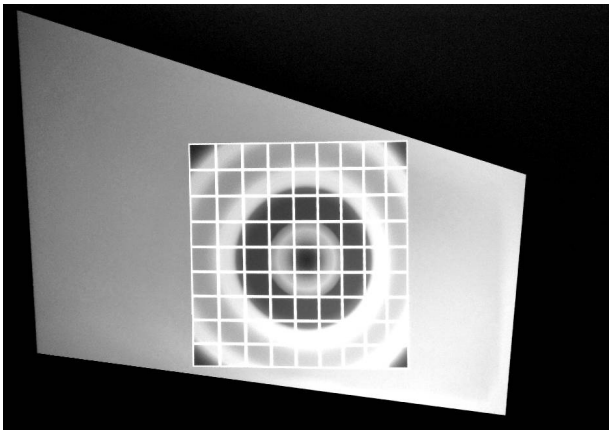


Picture 10: patch with one projection module

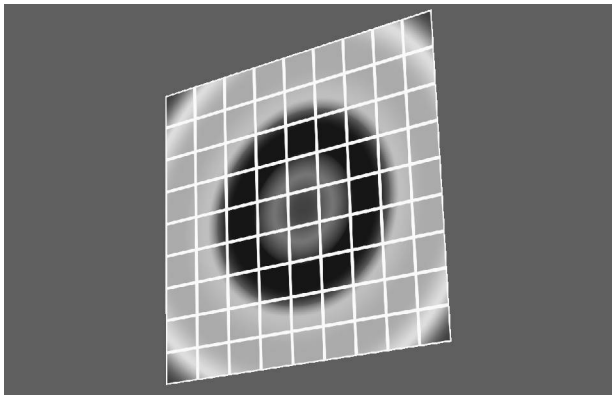
A projection module basically consists of one [mesh_square] object, onto which a texture is drawn via [pix_texture]. An openGL vertex shader gets the freely relocatable vertex coordinates, and correspondingly distorts the projection plane inside the GEM window (gemwin). The same shader also processes the texture coordinates to draw only the selected quadrilateral part of the texture.

In this example, we want to project the whole 900x900 pixels texture, so the four movable texture points lie in its corners.

While the vertex coordinates are specified in GEM units, the texture coordinates need to be given in pixels. The points need to be moved by visual judgement until the projection appears in the right way on screen.



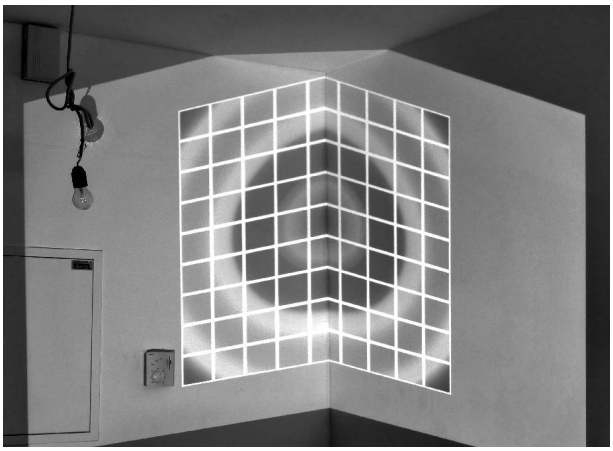
Picture 11: corrected image on screen



Picture 12: resulting gemwin content

3.2.3 Multiple screens

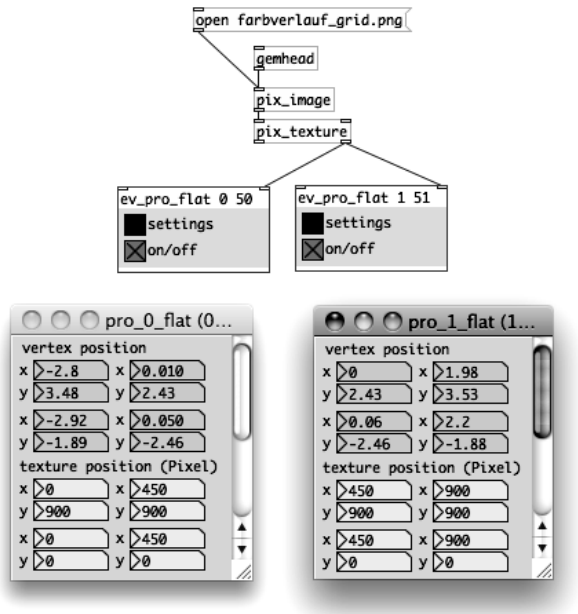
Edges in the screen cut the projection surface into multiple parts.



Picture 13: projection into a corner, without correction

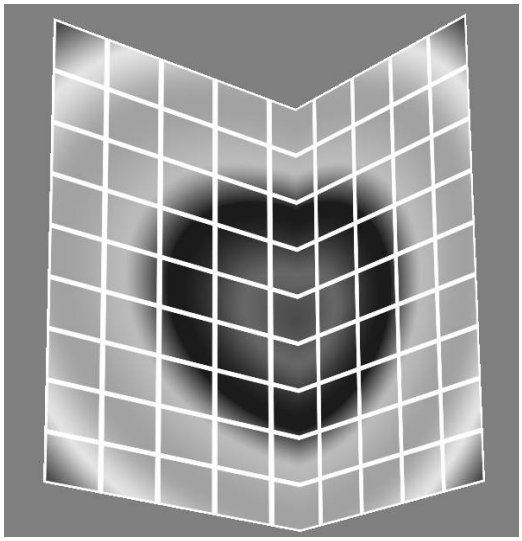
Therefore, in this case, a continuous image, parallel to the room edges, must be created out of primitives, which have to be treated individually.

These primitives (= projection modules) can share the same texture from which each one cuts out its desired part.



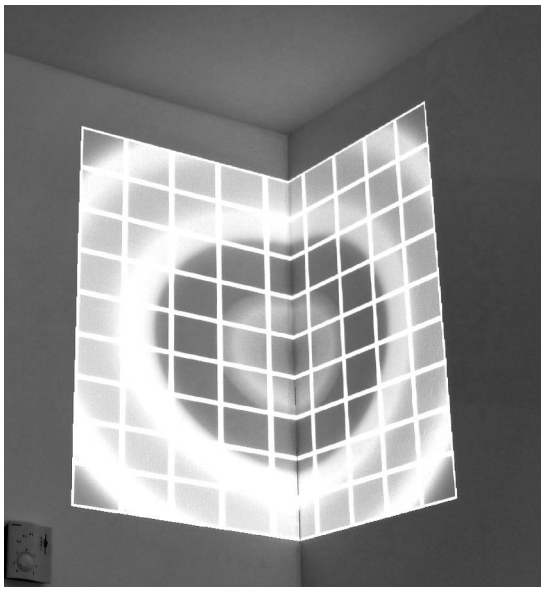
Picture 14: patch with geometric correction

Here, the texture is cut vertically into two equal parts, where module 1 gets the left section with horizontal pixel positions x = 0 to x = 450, and module 2 gets the other side of the image, starting at x = 450.



Picture 15: gemwin contents

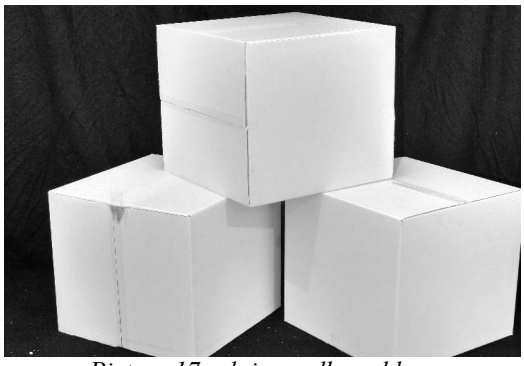
To show only the created projection planes on the wall, the gemwin background must be blackened.



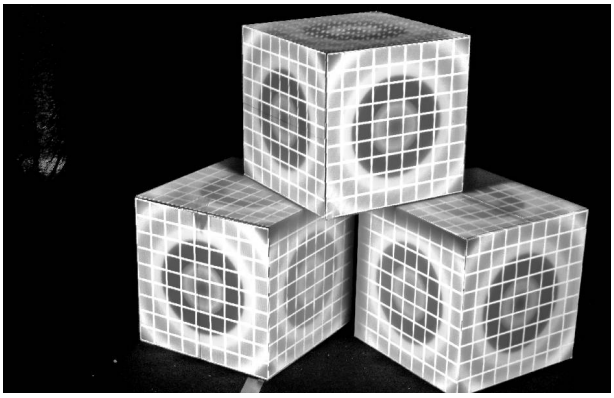
Picture 16: resulting image on the wall

3.2.4 Mapping on polygonal objects

With the same approach, it is also possible, to map video material on more complex polygonal objects, like cubic cardboard boxes:

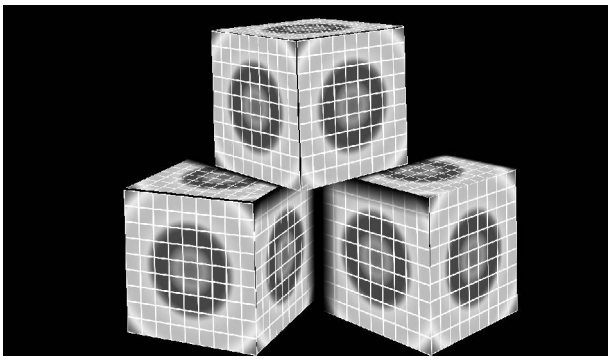


Picture 17: plain cardboard boxes

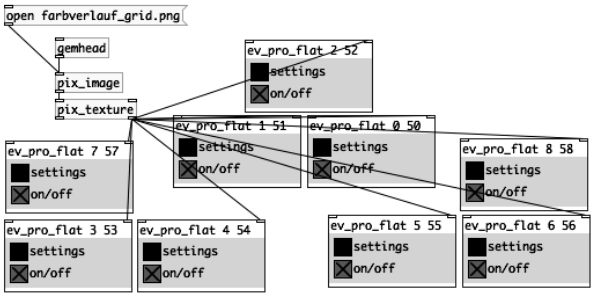


Picture 18: video mapping with one projector

As the photograph was taken from nearby the projector position, the gemwin looks very similar.



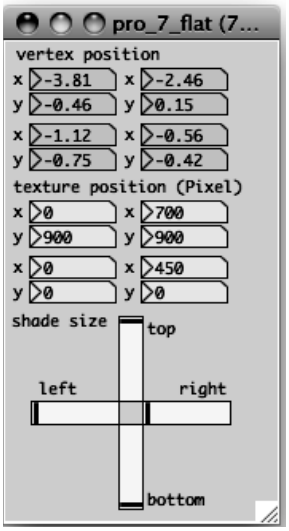
Picture 19: gemwin content



Picture 20: patch with 9 projection modules

In this example, with one projector, it is not possible to project behind the top cube, because its shadows evoke blind spots on the lower cubes.

Projection module 7 (top side of the lower left cube) can only cover a trapezoid part of the full square, as it is concealed by the box on the top. To eliminate geometric distortion, its texture is also set as only the corresponding trapezoid part of the full square texture:



Picture 21: settings

3.2.5 Soft-edging & overlap

One trick to hide the sharp edges, caused by the shadows, is to create fades on the affected sides. This feature, which is done by a fragment shader, becomes very handy, especially when it comes to multiple projectors:

The screws of projector mounts tend to loosen after a while. So, if the installation needs to be stable for a longer time, it should be immune to tiny movements of the projectors. The solution is a crossfading overlap between the projectors on the cost of image sharpness in these border areas. Obviously, not only the projection plane, but also its texture needs to overlap.

3.2.6 Multiple Projectors

We notice, that for every projector, the geometrical distortion of the image must be treated individually. And if one projector covers multiple surfaces, they must be treated individually as well.

As GEM allows just one instance, and can only display its video output on the desktop, a multi-projector-system requires an extended desktop set up by graphics driver and operating system.

To distribute the video to the single projectors, one big gemwin can then be spanned over the whole extended desktop, which is portioned hard / pixel-by-pixel onto the projectors. The projection planes can now be allocated onto the individual projectors by shifting them inside the gemwin.

3.2.7 Curved screens

So far, we only covered flat projection planes. To get rid of curved walls, the four corner-points of the vertex are expanded by five additional center-points in between. The 2x2 point matrix therefore becomes a 3x3 matrix whose extra points can be used to bend the sides of the quadrilateral or add convexity.

3.2.8 Realtime Systems

If the texture is generated in realtime, the video material needs to be rendered into a gemframebuffer, which then acts as a texture for the projection modules. This way, also realtime-stitched panoramic video or 3D-renderings can be distributed to the projection modules. For complex setups with different textures for the projection planes, it is also possible to create multiple framebuffers.

3.3 Immersive Media Environments

Picture 22: immersive extended view installation

By combining the discussed projection techniques, the problems in projection have been solved satisfactory, and we are now able to create the encircling panoramic video projection environments, the toolkit originally aimed at.

4 Conclusion

This paper refers to version 0.2 of the toolkit, but should be still valid for future releases.

The development of the Extended View Toolkit is being constantly continued. By now, it covers only the basic issues of image stitching and projection mapping. What it really needs at this point, are projects, that give feedback and suggestions for further improvements and extensions.

More information on the project can be obtained at the official homepage [4], where it is provided as a free download.

6 Acknowledgements

This project was made possible through support by

COMEDIA - Cooperation and Mediation in Digital Arts, and

IEM - Institute for Electronic Music and Acoustics / Kunstuniversität Graz.

Our thanks go to Peter Innerhofer, who created a gstreamer and python based streaming solution, suitable for panoramic video [5]. It has been developed in close collaboration with the Extended View Toolkit.

Additional thanks go to Johannes Zmölnig for constant support.

References

[1] Medienkunstlabor Graz: a camera for mklAVVE; www.medienkunstlabor.at

[2] R.Hartley & A. Zisserman: Multiple View Geometry; Chapter 8.4; pp 202; Cambridge University Press

[3] R. Szeliski: Computer Vision- Algorithms and Applications, Chapter 2.1.6, pages 52- 53; Springer Verlag

[4] Extended View Toolkit – official homepage http://extendedview.mur.at

[5] Peter Innerhofer: Streaming Solution with Gstreamer;
https://github.com/peonic/streaming