Automated system for panoramic capture

By David Ness

BACVA3

Abstract

This essay is an attempt to automate, or at least semi automate the process of capturing a panoramic image. Firstly this report how and why we would want to capture such images, and then an investigation into what can be done to speed up and automate the process, using mostly standard equipment, but always cheap. The project proposed involves the use of stepper motors to control a camera around it axes via an intuitive computer program.

Table of contents

Introduction

The 2d photograph has always played a part in the creation of realistic 3d environments in computer generated environments. The computer has led us to use and take pictures in many different ways for different purposes. One method which provides significant advantages over others for use in computer graphics is the panorama.

Aims of this report

This essay will firstly discuss what panoramas are, how they are captured, and what purpose they serve in computer graphics, and what advantages and disadvantages they provide over other methods in the given field. This paper then goes on to describe how the process of panoramic image capture can be improved, and partly automated via a computer program and motor control, to provide a cheap, and efficient system for high resolution, wide angle images, that can be used to create immersive computer generated environments.

1.1 What are panoramas

Panoramas are attempt to represent extremely wide angle of view. Capturing these wide angles has a few different methods, including using cameras with multiple lenses, cameras with lenses moving in an arc around the film plane(“Swing Lens” or “Slit Scan”). More recently, software has allowed regular cameras, with normal focal length lenses to capture images around a centre axis and stitch these images into one, (“Segmented”). It is this method that this paper will concentrate on.

Fig. 1

Figure 2.

Figure 1 shows 5 of 50 photos in total of a “segmented” panorama. The photos were shot with a 28mm lens, and a 30% overlap.

Figure 2 shows how seamless the stitched images can appear. Created with Realviz’s Stitcher application. The full set of images, and a 360 degree panorama can be found in appendix B.

1.2 Method for panoramic capture

1.2.1 Finding the nodal point

The “nodal point” of a lens is the point inside a lens where light rays cross before being focused onto the CCD / film plane. When taking the pictures for a panorama, it is desirable to rotate the camera around the centre axis of the the nodal point of the camera lens. Not doing this may result in problems due to parallax during the stitching process. Parallax is the apparent shifting of foreground objects relative to background objects when you change your point of view position.

While it is not essential to accurately position the camera for each image, it does make things a easier if the lens is rotated as close as possible too its nodal point.

1.2.2 Lens focal lengths

The most common type of lens is around 18mm – 35mm, this requires that around 50 images are needed to make a fully spherical panorama. An alternative to this is a fish eye lens, which has a large field of view, usually around 120 – 180 degrees using a fish eye lowers the number of pictures required too around four. Although this has the advantage of both having fewer pictures needed, and can even be used hand held, fish eyes lens are expensive, and have few other uses. Using fish eyes reduces the possible output resolution to something nearer a normal picture.

1.2.3 Use of tripod, and tripod heads

A tripod is generally required to make the point each photo is taken from consistent and steady. An average tripod head is unsuitable for rotating a camera around its nodal point. For this reason special tripod heads have been developed. These heads allow various cameras to be position around its nodal point.

1.2.4 Camera settings

The setting on the camera should remain consistent throughout the image shooting. For this reason the “manual mode” of a camera should be used. Its highly unlikely that one exposure will suit the entire panorama, with some areas being brighter and some being in shadow. An average exposure should be taken and the values programmed into the camera. The focus of the camera should not change, and for this reason a small aperture setting should be used to capture the largest depth of field possible.

1.2.5 Shooting images

One of the key advantages of panoramas is that they can use any type of lens. Of course the shooting requirements change for each focal length, which means we need some way of knowing how many shots we need for the focal length of the lens.

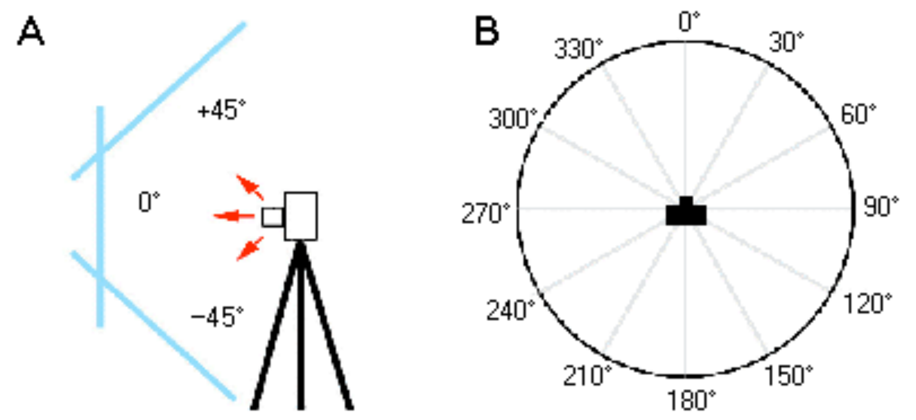


Figure 3 showing the camera setup from the side (A) and the top (B) for a panorama. Taken from the Realviz sticther documentation.

Information on this is usually provided for average focal lengths (18mm, 25mm, 28mm, 35mm for example) in the documentation of the software being used to stitch. Users have also created tools for calculating this, and we can of course calculate this ourselves (this is discussed further in section 4.5).

> *To capture a full 360° × 180° view of the scene images need to be captured in rows. That is, in addition to capturing images in a 360° circle as you normally would to make a panorama, you also need to capture rows of images with the camera tilted up and down.*
>
> *For example, a panorama might consist of three rows of images. Each row of images consists of 12 shots. The images in each row are captured at 30° increments:*
>
> *The top row is captured with the camera tilted up at 45° pitch. The middle row is captured with the camera levelled at 0° pitch. The bottom row is captured with the camera tilted down at –45° pitch.*

Once we have all the images required, software special designed to merge the images into one can be used. We refer to this process as “stitching”.

> *Most stitching an overlap of about 30% between adjacent images. The number of images needed to achieve a 30% overlap will depend on the Field of View of the lens being used. (Stitcher Documentation)*

1.3 Why panoramas?

1.3.1 Image-based Rendering

Image Based Rendering (IBR) is the attempt to represent two-dimensional images, (i.e. photographs that have been taken in the real world) on three-dimensional geometry, sometimes also referred to as two point five dimensional or even matte painting. IBR encompasses many different methods, the most simplest of which is texture mapping.

A variation of texture mapping widely used in IBR is projection mapping, where images are mapped to geometry from a single viewpoint. Once the textures have been projected onto geometry, small camera moves can be archived without noticeable distortion of the textures.

There are many advantages to image based rendering as opposed to texturing environment from scratch. One of the major advantages is that creating a realistic environment in 3d can become very computationally expensive as more and more realism is sought. Algorithms that attempt to simulate physically realistic effects such as global illumination and ambient occlusion are complex, and despite advances in processing power. By using IBR, the realism of the scene is implicit in the images and does not need to be calculated. Even extremely large images can be rendered with relative speed. This factor alone has made IBR an important asset in creating feature film, where output resolutions are high.

Panoramas are particularly well suited to image-based rendering, due to being able to capture whole environments and therefore the chances of coming away from locations with the needed images to completely reconstruct the scene is increased because nothing is out of the frame. Panoramas, because they are made up of many images can have extremely large resolutions, which may be desired in areas such as special effects for film.

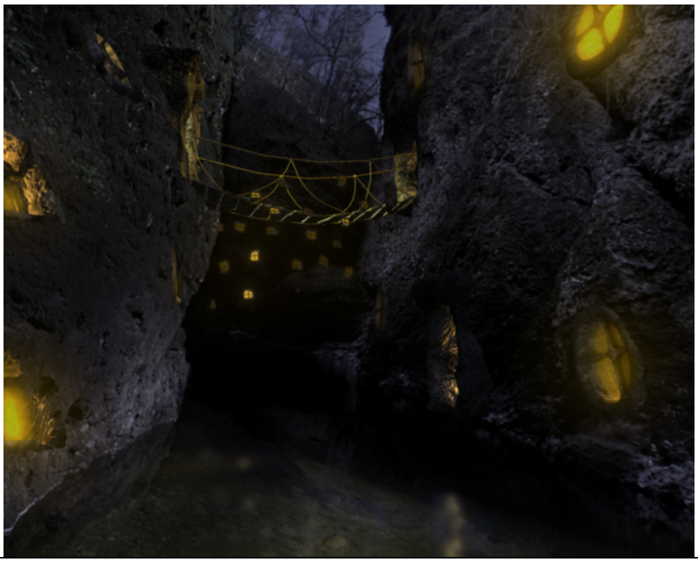


Figure 4 showing my specialist project, where a panorama was used as a background, which allowed for a camera rotation. Parts of the panorama were also projected onto the geometry to create integration with the original photos and synthetic elements. A video of the project can be found in Appendix B.

They can be used in scenes where the camera rotates around its own axes, and modifications can be made to the field of view to create a “zooming” effect, but translation isn't possible without noticeable incorrect perspective. A way around this is to texture projection, so that the panorama can be mapped onto geometry and thus, small camera moves can be archived.

1.3.2 Image-based Modelling

Image-base modelling (IBM) is the process of creating three-dimensional models from one or more two-dimensional images. ([1] Debevec 1997). Image-based modelling is usually used in conjunction with image-based rendering. The two combined, can realistic, accurate and computationally fast representations of an environment.

Image-based modelling has many advantages over normal modelling. Firstly, the process of conventional modelling is labour intensive, both in terms of research and execution. For example, supposing an interior environment was being modelled, either a large amount of measurements would need to be documented, or architectural plans would need to be acquired.

Secondly, accuracy of the model can be hard to archive.

Many photos are usually required for image-based modelling which makes panoramas are well suited to this task. Panoramas also have many other advantages over taking singles images. These advantages relate to the calibration of the photographs to the CG environment. Since the point at which the photos were taken is at the centre of the world, the camera doesn't have to matched to the photo as would normally be required. Secondly, sometimes the focal length available on some lenses may not be wide enough of cover the area required, for instance photographing a building in a street. For image based modelling it would be useful to capture the whole building, but suppose the street does not allow the photographer to move far enough away from the building to fit it all in view? Panoramas capture the whole environment so this is not an issue. Examples of image based modelling can be found in the apendix A, figs. 11.1, 11.2 ,11.3.

1.3.3 Image-based Lighting

Both image-based modelling and image-based rendering are intuitive ways of getting real world scenes into three-dimensional packages. However, what happens when we wish to add synthetic objects into these environments?

“This difficult task requires that the object be lit consistently with the surfaces in their vicinity, and that the interplay of light between the objects and their surroundings be properly simulated. Specifically, the objects should cast shadows, appear in reflections, and refract, focus and emit light just as real objects would” ([2] Debevec 1998)

In his 1998 paper, Paul Debevec proposed a method to which is accurately able to light objects that interact with the environment, casting appropriate shadows, are properly reflective, can reflect and focus light and exhibit appropriate diffuse interreflection *(Debevec 1998).*

The method proposed made use of his previous work, ([1] Debevec, 1997), using HDR (a description of HDRI is provided in 3.1) of images to correctly capture radiance information over a full 360 by 180 degree range. This was archive by photographing a highly reflective sphere, referred to as a “light probe”. This information is then used to drive a global illumination simulation.

Figure 5, showing a mirrored ball can be used as a light probe. The probe acts as an extream fish eye, covering 360.

The quality of information produced by a light probe is dependant on the reflective properties of the probe. Any spherical object can be used as a probe, some examples could be at its most basic, a Christmas bobble [3] to industrial steel bearings which Paul Debevec has suggested has having the best reflective qualities.

The problem is that a Christmas bobble while cheap, have poor reflective qualities, while a steel bearing weighs a lot.

Capturing a panorama of the environment can archive the same effect as a light probe, however as the number of exposures required for a HDRI multiplied by the number of images required to cover the 360 range, a panoramic HDRI quickly becomes impractical.

1.4 In summary

The last three sections have described how panoramas can be put to good use over these different areas to recreate environments. These three different areas can all be archived by different means, but by using panoramas, the can all be done with one process. In summary, panoramas have the following general advantages:

- Ease of calibration of CG camera to environment.
- A large field of view makes sure subject can be seen in one image so there is no mismatching seams.
- Relationships between objects can easily be seen.
- Once the images have been stitched into the final image, the original images can be discarded, making file management easier.
- Output resolution can be very high
- When used as backplates, manipulation / editing can easily be archive, where complex compositing and tracking would be required with film / video.

1.5 A combined solution

Panoramic image capture was a large part of my specialist project in my second year. Some of the failures of this project should be explained in ordered to provide a background for the motivation for the system which will be proposed in section 4.1.

The project was considered a “set extension” project. The aim was to use a real word environment and add in computer generated elements to create an interesting atmosphere and aesthetic appeal. A camera move was desired, but tracking was to time consuming for the scope of the project. After some research the best alternative appeared to be a panorama.

Panoramic representations only allow for rotations and modifications to the field of view at a single viewing position, (i.e. translation cannot be easily archived), however it is surprising how effective this method can be. (McMillan and Gortler ,1999).

At the time, only a compact digital camera was owned, with the widest focal length being 28 mm. At this focal length it required 50 images to cover a fully spherical panorama.

Shooting of the panorama took place in a small gorge in Scotland, during a spring morning. Lighting conditions inside the gorge were at two extremes, in some places the sunlight was casting bright, “hot spots” onto the rocks, while other parts were in deep shadow. One exposure setting was not enough to cover the whole exposure range. The result was that the information in the sky and highlights was completely lost.

A HDRI panorama was desirable here and could be archive by taking multiple exposures at each stage of the panorama, however fifty images were required for a normal panorama, and at least 5 exposures would be required at each stage, meaning at least 250 images were needed, and this would have been imposable to capture by hand.

Although the method used to capture the images was not perfect, the use of panoramas themselves proved incredibly useful. Geometry to represent the gorge was easily constructed from the images, and the camera move was convincing, and easily executed. The panoramic image itself was easy to edit in Photoshop and despite the large output resolution, rendering was still relatively fast. (See apendix A figs 9 and 10)

The overall success of this project has lead me to investigate how panoramic image capture could be automated in an effort to speed up the process and thus allow for HDRI panoramas with an average digital camera.

The purpose in doing this is to archive an integrated solution that can be used in both image-based rendering, modelling and lighting to recreate realistic environments.

3.1 High Dynamic Range Images

Both traditional photography and early computer graphics suffered from the problem of being able to represent a wide range of light intensity's.
In photography, when taking a picture, the camera, or photographer must assess the scene and determine how much light must be let into the camera. This is called exposure. The camera operator may choose to expose for the most interesting part of the scene, or simply an average that best exposes all of the shot. When too much light is let into the camera over exposure occurs which leads to the “burning” of highlights. When under exposure occurs, information where the scene is most dark, is lost. As Paul Debevec explains in his 1998 paper on HDRI,

> *“Accurately recording light in a scene is difficult because of the*

> *high dynamic range that scenes typically exhibit; this wide range of brightness is the result of light sources being relatively concentrated. As a result, the intensity of a source is often two to six orders of magnitude larger than the intensity of the non-emissive parts of an environment." [2]*

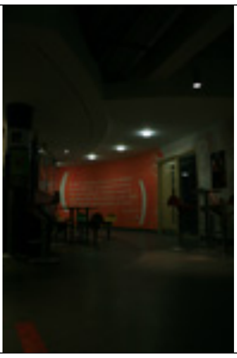
When information is lost due to this, we referred to it as clipping, which happens when the values are greater or less than the limits of the format / device, and the information is simply lost.

High Dynamic Range Imaging (HDRI) is an attempt to store a wider range of exposure than traditional imaging techniques would allow. Originally conceived for the use in computer graphics by Greg Ward when he was developing a physically based rendering system. He observed that it was unacceptable to "throw away" information calculated by the engine when writing the final result to a normal image format. He decided to write his own image format to store the wide range calculated by the rendering engine, and the first HDRI format (Radiance RGBE) was conceived.
Paul Debevec later presented a technique [1] for combining a series of multiple exposures taken with a normal camera to create a HDR image (this method is explained further in section 3.1. This has now become a widely used in both computer graphics, the film industry and photography.
By using this method the problems mentioned previously to do with photography exposure became less significant with the introduction of HDRI.

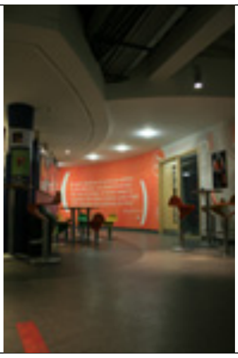
As previously mentioned, HDRI photographs can be archived by using a series of photographs at different exposure settings. Due to this a tripod is required to prevent the camera from moving in between exposures (although some software now allows for automatic correction for mismatching photos). A set of events to capture a HDR image may look something like this:

- A correct / average exposure for the scene is sought. A picture with these settings is taken.
- The camera settings are then modified to take both images that underexposed and over exposed. In Paul Debevecs original paper on the subject he notes that only two images are required in theory to make a HDR image, but in practice at least 3 are needed. Sometimes 5 or 7 or more maybe used.
- Under / over exposing is archived by changing the shutter speed, while keeping the aperture, and ISO speed the same.
- Increments of +/- two stops are usually used.

-4 EV



-2 EV



Evaluated exp.

+2 EV

+4EV

Unfortunately, most picture formats (JPEG, BMP etc) store 8 bit numbers which means that they can store values for each pixel between 0 and 255. Because HDR has such a wider range of values, these existing formats are not suitable for storing HDR information. Most HDR formats use 32 bit floating point values.
Some HDR image formats:

- OpenEXR (.exr) – an open source format, developed by Industrial Light & Magic for use in computer imaging applications. Source code was released in 2003.

- Radience 32 bit (.hdr) – The original HDR format developed by Greg Ward. Since it was the first, and for a long time the only HDR format, it is widely supported by many packages.

3.2 HDRI Panoramas

Panoramic HDRI is normally archived by taking various exposures at each stage of the panorama and then fusing these images together into a single HDR image (bracketing). These HDR images are then stitched in the conventional way (although few packages support this feature).

An automatic bracket is usually a standard feature on digital cameras. However it normally only used to take three pictures, one at the evaluated exposure, and one slightly overexposed, and one slightly underexposed. Usually no more than + / - two evaluated stops are allowed.

While this is a nice feature, and may even be of use in HDR photos with a low range of exposure is present. However a HDR usually requires at least 5 or 7 exposures(-6 EV, -4EV, -3EV, Evaluated exposure, +2EV, +4EV, +6EV respectively).

Some high end digital SLRs are now have features to take as many as 9 exposures in one bracket (Canon 1D MkIII), however these features are only present in high end digital cameras, usually costing around £4,000, and therefore escape the scope of this project.

3.3 What can be done to speed up the process with a conventional camera

Its quite common for digital camera manufactures to provide software for connecting Canon cameras to computers for what is referred to as “remote capture”, whereby the parameters of the camera can be controlled through a piece of software. In this way a bracketing feature similar to that found in high end digital cameras can be emulated from a computer program.

Using this software can increase the speed with which panoramas can be captured. One reason being that using a computer is far less fiddly, especially when the camera is mounted on a panoramic tripod head, which often means the screen or controls are not easily assessable. Another is that two people can effectively take the panorama, one rotating the tripod head and the other operating values.

This process still requires a lot of repetition, and it was felt that the process could be automated in an attempt to both speed up the process and make it less tedious.

After some research it was found that some digital camera manufacturers provide a software developers kit (SDKs) for controlling specific cameras form custom programs.

Canon, Nikon, and Olympus all offer SDKs, while Sony does not.

3.4 The professional approach

Since panoramic photography is wildly used in the effects industry there are high end solutions available.

Slit Scan

One such solution is the Panoscan Mk3. The Panoscan camera uses a tri-linear CCD array in much the same way as a slit film rotational camera might operate. The camera assembles an image by capturing a single line of pixels at a time while rotating through a 400 degree arc, with resolution capability up to 9,000 by 65,000 pixels.

The Panoscan MK-3 camera is also capable of capturing extremely wide exposure range. While most film cameras can capture approximately 6 EV of dynamic range, the Panoscan sensor can capture up to 12 EV. This extended range captures all of the highlight and shadow detail that would be missed on film or regular digital exposures. Furthermore, the digital image from the Panoscan can be combined into HDR images.

4.1 Project proposal

Inspired by high end panoramic systems that use motors to rotate a camera around its axes, and the ability to use a library for controlling the camera from a custom program, I propose a system for the control of a standard digital camera, which semi-automates, and thus simplifying the process of capturing HDR images, panoramas, and HDR panoramas, with mostly standard equipment and at a reasonable price.

The project will consist of two main parts. One is a mechanical device for rotating a camera round its center axes, utilizing stepper motors, and the other is a piece of software that controls the camera attributes and releasing along with the rotating of the camera. The aim is to produce an interface that is intuitive and helpful and user friendly.

This will be archived by creating a program with the following features:

- A graphical user interface that allows for seamless control of camera parameters.
- Features that facilitate the capture of HDR images
- Features the facilitate panoramic low and high density panoramas
- A built in “panoramic calculator”

Ideally the program would also combine the low dynamic range images into high dynamic range images. However the amount of time and knowledge required for this would exceed the scope of this project.

4.2 Canon SDK

The proposed project requires that the camera be controlled via a computer, preferably through a Universal Serial Bus (USB). Many manufactures offer such functionality through a C-type library. Canon, Nikon and Olympus both supply “Software Developers Kits (SDKs)”, although Sony do not.

A Canon 400D was used throughout development of this project, and therefore the Canon SDK was used. Canon has two types of SDK, one for their high “EOS” range type cameras referred to as the “EDSDK”, and one for their consumer “Powershot” models, referred to as “PS-Rec SDK”.[12]

The Canon 400d is an EOS model, and therefore the EDSDK was used. The EDSDK is described as being;
"[an] interface for accessing different camera models and their data. Using EDSDK allows users to implement Canon EOS digital camera features in software". [11]
This is to say although only the 400d has been used to develop this application, since the EDSDK is a common interface for most EOS cameras, all cameras compatible with the EDSDK should work with the developed application. A list of these cameras can be found in appendix ?

Acquiring the EDSDK requires membership to Canons "Digital Image Developers Programme". Membership is assessed, and not everyone can acquire the SDK. Under the license agreement, the SDK cannot be redistributed, and therefore anyone wishing to used the program produced for this project would have to apply to Canon for the necessary DLL files.

4.3 Graphical User Interface with Microsoft Foundation Classes

The EDSDK documentation states that the library is available for both Windows and Macintosh platforms. However, until very recently only the windows version was available from the European developers program.

This meant that the development platform was restricted to Windows. This played a major part in the choice of windowing system. Due to the nature of the project, an interface system that lent itself to a utility type application, as well as being able to display an openGL was sought after.

There were two windowing systems that were considered, Microsoft Foundation Classes, and wxWidgets. Microsoft Foundation Classes (MFCs). MFCs come as part of Visual Studio (the development environment) and are C++ based classes that wrap round parts of the Microsoft API. Because of this native windows style controls are easily created. They come as part of Visual Studio and are well integrated with plenty of documentation.

wxWidgets while being available for many platforms, and supports openGL rendering, is not a standard part of the Visual Studio package.

Since the project was restricted to Windows, MFCs were chosen as the windowing interface.

4.4 Stepper motors

The rotation of the camera would ideally be very accurate however not is not a necessity, since stitching software is designed too cope with imperfectly space and aligned images. It was still however preferable to have accuracy. Normal Direct Current (DC) which are the most common type of motors are inaccurate, since no knowledge of the position can be known or found out easily.

One type of motor that is used where accuracy is important is the stepper motor.

Stepper motors operate much differently from normal DC motors, which rotate when voltage is applied to their terminals. Stepper motors, on the other hand, effectively have multiple "toothed" electromagnets arranged around a central gear-shaped piece of iron.

The electromagnets are energized by an external control circuit, such as a controller. To make the motor shaft turn, first one electromagnet is given power, which makes the gear's teeth magnetically attracted to the electromagnet's teeth. When the gear's teeth are thus aligned to the first electromagnet, they are slightly offset from the next electromagnet. So when the next electromagnet is turned on and the first is turned off, the gear rotates slightly to align with the next one, and from there the process is repeated. Each of those slight rotations is called a "step." In that way, the motor can be turned a precise angle .[6]

The stepper motor is therefore particularity well suited to the application of precisely rotating a camera around its axis. As stated, control of stepper motors is usually archived through micro- controllers, and is therefore more complex than DC motors. Simple, pre-made control boards are readily available however.

Such a control was bought from “Pc in Control” [9] allows control of the board from a Dynamic Link Library file.

4.5 Panoramic calculator

While information on focal lengths and the amount of images that a required are part of the documentation of the stitching software, the user is restricted by the focal lengths described by this documentation. Many web sites have “panoramic calculators”, where the user can input arbitrary overlap and focal lengths and the output tells the user how many photographs are needed and at what angles.

As one of the aims of this project is to provide an intuitive way of capturing panoramas, one of the features implemented in the application allows for the user to input arbitrary overlap and focal length values and preview the result in an openGL window. The system then controls the camera, rotating and capturing at the values arrived at by the panoramic calculator.

Calculating the amount of images needed is based on the angle of view of the lens. The angle of view can be calculated from a) the focal length and b) the size of the film back / CCD sensor and some basic trigonometry.

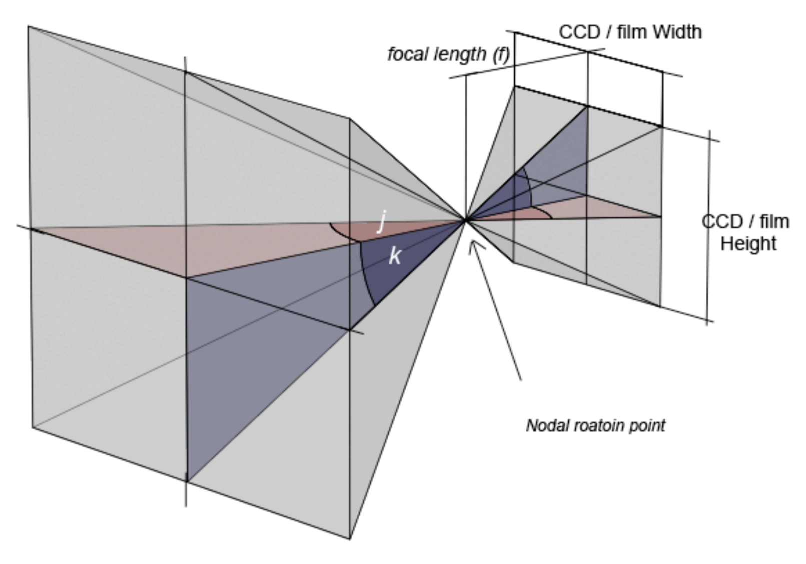


Figure 7 The angles j and k can be found using trigonometry. j = Arctan(f / 2d), where d is the width of the sensor. K can be computed in a simlar way. This is not always true however, as in case of macro photography, this however is not a problem since panoramas do not usually deal with macro photography.

5.1 Development of final product

While the learning curve for using Microsoft Foundation Classes was somewhat steep, it was certainly worth it. By using native Windows controls, both programming and the end result are easy to use. The only major disadvantage of MFCs is that they inherit some of the problems with the Windows API.

Most of the controls avaliable from MFCs follow a standard format, this format is however, sometimes broken for no reason, and without documentation.

Most of the solutions to these kinds of problems were found from various webs sites and forums.

The Canon SDK once setup was straight forward to use. Again the documentation was the weak point. Some of the sample code provided had spelling errors and typographical errors. One feature provided by the EDSDK is the ability to download images from the camera to the PC as they are taken. To do this both an “event handler”, which essentially routes events from the camera to te appropriate part of the program, and an “object hander”, which deals with downloading files, are required.

Unfortunatly neither of these features we implemented in the program, this was partly down to time constraints and partly down to the documentation again. This unfortunately means that the images taken stay on the camera and have to be downloaded with the standard Canon software.

The main feature not implemented was the communication between the stepper motors and the camera. This again was mostly down to lack of time, as well as lacking some hardware to properly support the camera.

While only a single row panorama was implemented at this stage, it would be reasonable to assume that from this basis it wouldn’t be to difficult implement multi row panoramas.

5.2 Further / Future Development

Although the project is lacking some of the important features set out in this project, I believe the foundation is there, and the program still achieves the goal of being able to assist with single HDR images.

Another interesting feature would be to the program automatically assemble to the low range images in an appropriate format such as .hdr or .exr.

5.3 In Conclusion

While this project was fundamentally about a subject I was familiar with, a lot of the elements that were required were not so farmiliar. These different elements worked with a varying amounts of success. Ultimately, while having at least a start on the features proposed, the program is somewhat incomplete. It is therefore unforeseen whether the purpose of this project is usuefull. I believe that it is and will continue to develop the program to help me capture panoramas.

Bibliography

[1] Paul E. Debevec and Jitendra Malik. “Recovering High Dynamic Range Radiance Maps from Photographs.” SIGGRAPH 97, August 1997.

[2] Paul E. Debevec. “Rendering Synthetic Objects into Real Scenes: Bridging Traditional and Image-Based Graphics with Global Illumination and High Dynamic Range Photography”. SIGGRAPH 98, July 1998.

[3] “HDRI The Cheap And Nasty Way” by Andrew Whitehurst
URL : http://www.andrew-whitehurst.net/hdri_tut.html

[4] Paul E. Debevec, Camillo J. Taylor, and Jitendra Malik. “Modeling and Rendering Architecture from Photographs.” In SIGGRAPH '96, August 1996.

[5] L. McMillan and Gortler, “Image-based rendering: A new interface between computer vision and computer graphics”, 1999.

[6] Wikipedia article on stepper motors URL: http://en.wikipedia.org/wiki/Stepper_motors

[7] Panoramic Calculators on the web:
http://www.cambridgeincolour.com/tutorials/image-projections.htm
http://www.worldserver.com/turk/quicktimevr/panores.html

[8] http://www.panoscan.com/

[9] http://www.pc-control.co.uk/stepperbee_info.htm

[10] Gnomon Workshop DVD by Greg Downing http://www.thegnomonworkshop .com/dvds/gdo03.html

[11] Canon EDSDK Documentation, avalible with the EDSDK package from [12]

[12] http://www.didp.canon-europa.com/

Apendix B is a list of suplimentry files found in the “supFiles” folder accompnaying this report

[13] Folder “cobhamPanImages” – unstitched images used in fig 1.

[14] Large version (TIFF) of the panorama used in fig 2.

[15] Specialist Project final video, .mv2.

Apendix A

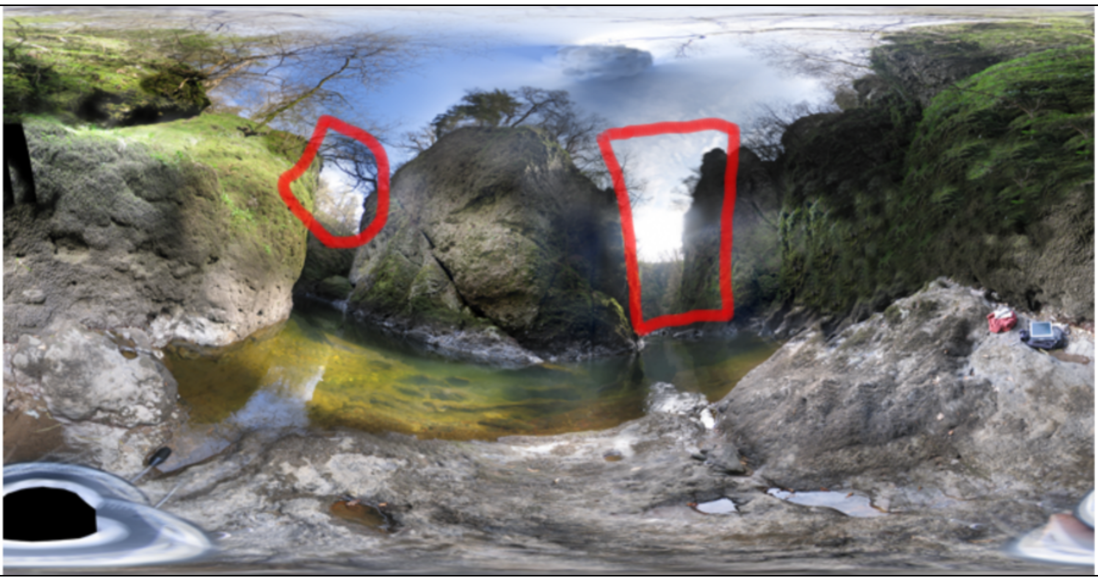


Figure 9 showing the original panorama. Highlighted in red are areas where the light intensity was to great to capture. Information in these areas had to be edited in photoshop. Some areas in show are lost as well.

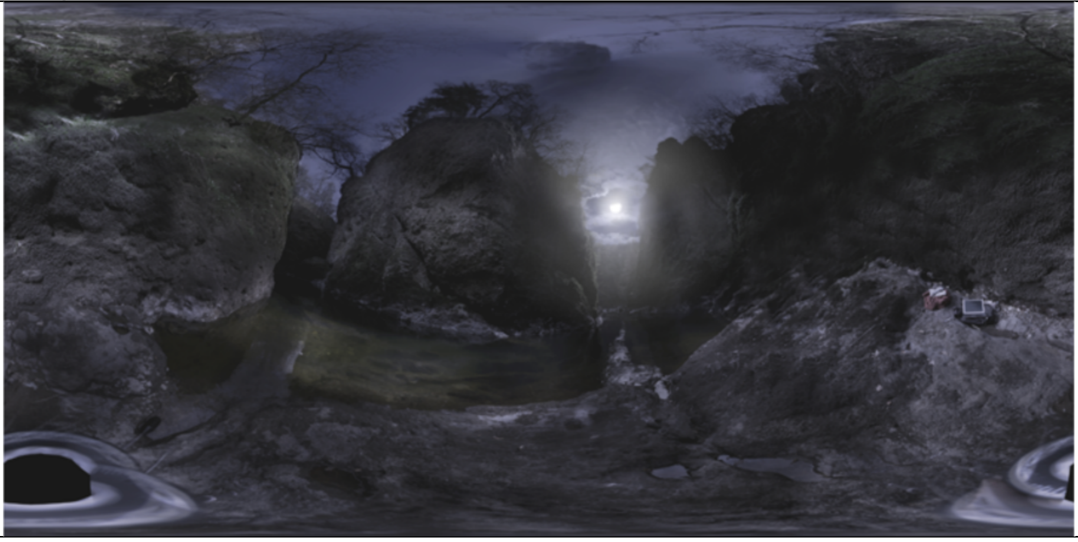


Figure 10 showing the day to night conversion that was performed on the panrama for use as a backplate. The process was relativley straight as most of the work was done in photoshop, and the image was treaded as an other.

Figures 11.1,11.2, 11.3 showing the the modeling of a Mosque from a Panorama, and the final textured model.

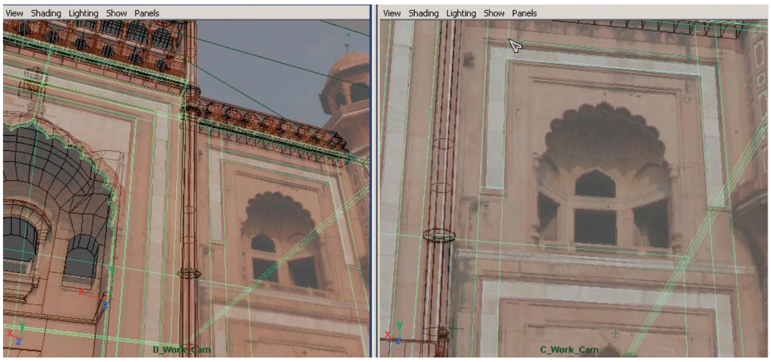


Figure 11.1

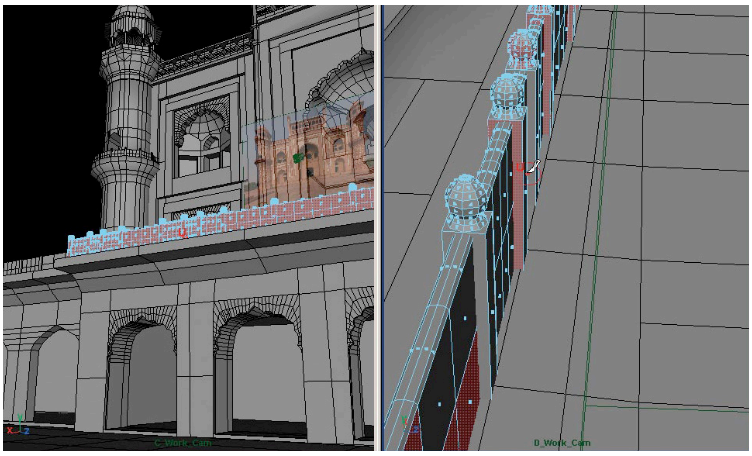


Figure 11.2

Figure 11.3

[]EDSDK v 2.1 compatiable cameras:
EOS 5D, EOS 40D, EOS 30D, EOS 20D, EOS-1Ds Mark II, EOS-1D Mark II, EOS-1D Mark II N, EOS-1D Mark III, EOS 400D, EOS 350D)