360-Degree Video Playback Demo

PlayStation®Vita Programmer Tool Runtime Library Release 3.50

Table of Contents

About This Document

This document explains techniques used in the 360-Degree Video Playback Demo sample code. The sample code is located in the PlayStation®Vita SDK package at:

```
sample_code/audio_video/demo_360degree_video
```

Typographic Conventions

This document uses the following typographic conventions.

Hyperlinks

Hyperlinks (underlined and in blue) are available to help you to navigate around the document. To return to where you clicked a hyperlink, select **View** > **Toolbars** > **More Tools** from the Adobe® Reader® main menu, and then enable the **Previous View** and **Next View** buttons.

Hints

A GUI shortcut or other useful tip for gaining maximum use from the software is presented as a Hint surrounded by a box. For example:

Hint: This hint provides a shortcut or tip.

Notes

Additional advice or related information is presented as a Note surrounded by a box. For example:

Note: This note provides additional information.

Text

- Names of keyboard functions or keys are formatted in a bold serif font. For example, **Ctrl**, **Delete**, **F9**.
- File names, source code, and command-line text are formatted in a fixed-width font. For example:

```
m_targetBufferData[idx]);      // pointer to the surface data
```

Related Documentation

Any updates or amendments to this guide are in the release notes that accompany the release package.

Document serial number: 000004892117

Document serial number: 000004892117

1 Demo Overvew

The sample demonstrates panoramic video and image display using touch and motion controls. You can pan or rotate the view horizontally using either the back touch panel or motion controls.

The video and the still image each has a 360-degree horizontal field of view. The sample renders the panoramic video texture by unwrapping the texture coordinates and mapping the texture to appropriate geometry.

Figure 1 Screenshot

The data was recorded using a Sony Bloggie™ Touch camera with 360 lens attached. The same camera and lens combination can capture both video and still images, and the sample can display both. You can produce similar panoramic video and still data using other types of cameras and lens attachments. Generally, the type of device used is a catadioptric system for panoramic images.

For more information:

- Sony Bloggie™ Touch

  http://presscentre.sony.eu/content/detail.aspx?ReleaseID=6122&NewsAreaId=2

- Catadioptric cameras:

  http://en.wikipedia.org/wiki/Panoramic_photography - Catadioptric_cameras

Menu Settings and Controls

Menu options:

- **show raw data**: Select to display the raw and intermediate data; otherwise, no data is displayed.
- **show still image**: Select to display the still JPG image instead of the video.

Square button: Toggle between touch and motion controls.

Back touch panel: Drag horizontally to pan the video or image.

- The sample uses libSystemGesture to process the touch panel data using the drag gesture type. The drag velocity is used to determine the change in angle of the view.

Motion sensor: Set the orientation of the video or image. Hold the controller in front of you with the screen upright and facing you. The controller orientation about the downward/gravity axis sets the orientation.

- The sample uses the motion library to retrieve the sensor state, then uses the orientation matrix to set the view or projection matrix.

Video Playback and Still-Image Loading

The source data output from the Sony Bloggie™ Touch camera is either a high-definition MPEG-4 video or a JPEG image.

The video playback functionality is encapsulated in the `AvPlayer` class. The MPEG-4 video format is streamed into a buffer using the Video Player (sceavplayer) library and the internal API use of the class is based on the sample code for the basic movie player, `api_libavplayer/simple_mp4_player`. See the SDK documentation chapter "Audio/Video: Video Player Library Overview and Reference" for details.

The still image is loaded and a libgxm texture is set up in the `JpegTexture` class. This uses the JPEG Decoder (scejpegdec) library to decode the very large JPEG image file. The sample code also downsamples the image while decoding, because the original image is larger than the maximum hardware decode size. This results in a decompressed RGBA texture data buffer. See the SDK documentation chapter "Audio/Video: JPEG Decoder Library Overview and Reference" for details.

In both cases, after the raw data is in a suitable buffer, it can be used as a libgxm texture in rendering.

Method of Unwrapping and Rendering the Image

Figure 2 shows an example of the raw data output by the Sony Bloggie™ Touch camera. The 360 mirror/lens attachment captures the panoramic image in a ring-shaped area. Figure 3 shows the same data after it has been unwrapped into a standard texture coordinate range. After the texture is unwrapped, it can easily be used as a regular texture and mapped onto any geometry for display.

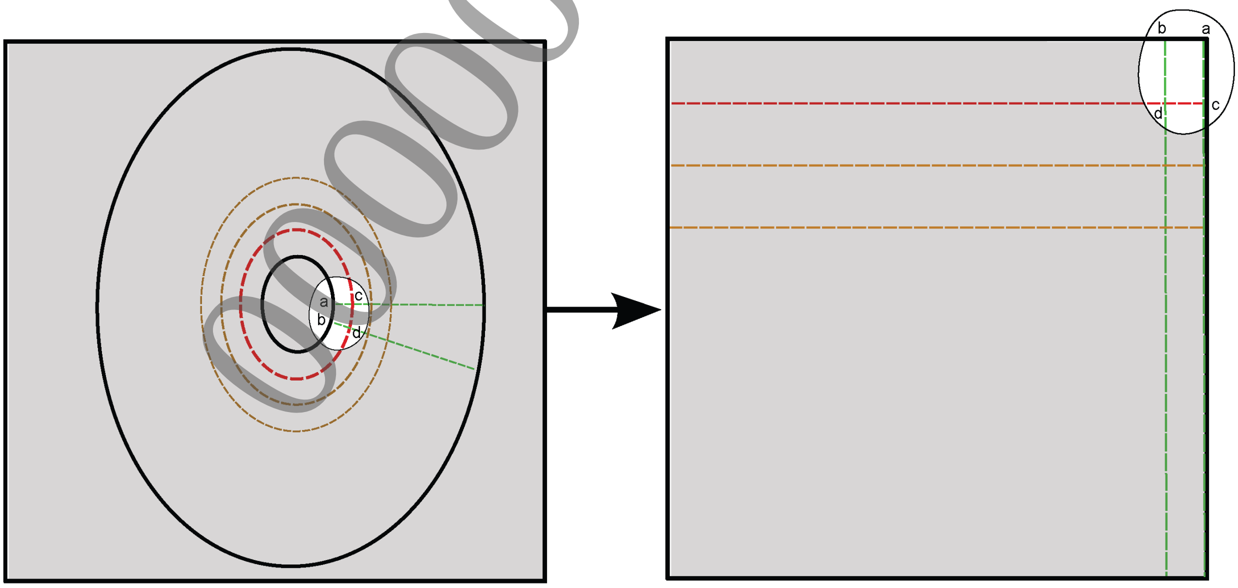
Figure 2 Raw Image

Figure 3 Unwrapped Texture

Texture Unwrap Calculation

The ring-shaped raw data can be transformed to give a rectangular panoramic image that contains the full 360-degree scene horizontally. In the source code this is done by assuming perfect optics, so each path traced by a series of equally spaced ellipses becomes the new texture coordinates. Figure 4 shows an example of mapping the first few ellipses to their corresponding arrangement in the unwrapped texture. The highlighted area with texture coordinates *a,b,c,d* represents the texture data for the first quad in the generated geometry, before and after the unwrapping.

Figure 4 Diagram of Unwrapping Texture Coordinates



The parametric formula for an ellipse is

```
X(t) = a.cos(t)
Y(t) = b.sin(t)
```

where `a` and `b` are the major and minor radii and `t` is an angle between 0 and 2*pi*. The ellipse is centered at the origin and its major axis is along the X-axis.

The calculation in the sample code is based on this formula but also uses the positioning and dimensions of the ellipses in the raw data. These settings are contained in the `BloggieData` data structure and are specified for each input file. The parameters are used to shift or scale the paths so that the final texture coordinates fit exactly in the 0 to 1 *uv* range.

In practice, the optical system is not perfect. For example, there may be additional distortion due to a nonideal lens or mirror shape, and these could possibly be specific to each camera and lens combination.

Texture Rendered on Geometry

The unwrapped texture can be mapped onto any shape of geometry but, to achieve a realistic view of the original scene, a flat or cylindrical object works best. The flat mapping best preserves the distant straight lines in scenes with buildings; however, the interiors of rooms are less distorted when projected on the inside of a cylinder.

The flat mapping is done by drawing one or two planes each with the unwrapped texture. The view angle is used to determine the position of the first plane and whether a second plane is needed because the edge of the first plane is visible.

2 Video and Still Data

Data File Dependent Parameters

To change the movie or still image displayed, edit the definitions in the source file `demo_360degree_video.cpp`:

```
#define FILENAME_VIDEO "app0:audio_video/movie/bloggie360short.mp4"
#define FILENAME_STILL "app0:audio_video/movie/oxfordCircus360.jpg"
static BloggieData videoParams = {1280,720,669.,350.,88,333,0.,3.14,0.,0.};
static BloggieData stillParams = {3840,2160,1965.,1078.,263.,975.,0.,1.,0.,0.};
```

The `BloggieData` data structure is defined in `demo_360degree_video.h`. The data parameter values may need to be adjusted for different data files to get the best result because different lens and camera combinations can have slightly different offsets.

You can check the effect of the current data parameters by viewing the raw and unwrapped data using the **show raw data** menu option. If the unwrapped image has areas visible from outside the desired donut region then the data parameters need some adjustment.

The exact settings can be found by checking the original data file containing the ring-shaped image. The example data was measured in image editing software, with the width, height, center, and radii in pixel units. However, only the proportions are used, so you can use data from screen captures or other units. The north setting is in radians, with 0 radians at the three o'clock position in the raw data and increasing in the clockwise direction.

Video and Still Image Quality

Although the resolution of the raw data is high, the unwrapping process necessarily results in a change in the final displayed aspect ratio and resolution:

- The still image is captured at higher resolution than the video, so the final display of the image is also higher resolution than the video.
- The theoretical resolution achievable with the video data depends on the data-file-dependent parameters mentioned earlier.
- For both still and video, if the difference between the outer and inner radius of the source data is N pixels, then the vertical resolution in the unwrapped image is also N. The horizontal resolution is variable, increasing from top to bottom in the unwrapped image; it is also related to the data parameters because the radii determine the number of pixels crossed by each ellipse path. This image is mapped onto an object with an aspect ratio calculated by assuming a 360:50 degree field of view covered by the entire texture.

Because the source data is initially in JPEG or MP4 format, these resolution effects are sometimes mitigated, although compression artifacts can become more obvious, and there may be resampling during decoding.

The decoded data is also sampled due to the process of generating the unwrapped texture coordinates in geometry vertices, and the texture is bilinear filtered during rendering. For the sample code, the sampling interval at this point was chosen manually by decreasing the interval until there was no longer a noticeable difference in output.

Generally, for any image capture device, there are differences in the quality of raw data caused by conditions in the original environment. Normally, for example, a well-lit outdoor scene will be less blurred than an indoor scene. Bright lights in the scene can cause reflections and other lens effects.

Additionally, there are movement and orientation effects that can degrade the raw data and affect the final result. A common issue with all panoramic images occurs when footage is taken while the camera is

Document serial number: 000004892117

not horizontal; when the image is corrected for the scene elevation, there is a loss in the usable vertical field of view.

In the sample code it is assumed that the raw data was captured on a horizontal plane so there is no loss in the vertical field of view or correction due to elevation. If you examine the still image closely, you can see that this is not really the case, so some distortion in the unwrapped texture is probably due to the camera elevation.

Video-Editing Software and Settings

The .mp4 video from the Sony Bloggie™ Touch 360 camera can be used directly so there is no need to encode the data again unless you want to edit the movie. To edit a panoramic video, you must encode the source data as an HD video file using video editing software such as:

- Freemake video converter

  The Sony Bloggie™ video in this sample was edited using this software and data was saved as PlayStation®3 HD 720p MP4 format.
- Sony Vegas™ Pro

Follow the instructions for making regular video format in the SDK documentation chapter “Audio/Video: PlayStation®Vita Movie Data Creating Guidelines”, but choose a HD frame size to encode.

Document serial number: 000004892117

Document serial number: 000004892117