

# *PlayStation Movie Compression and Playback Issues*



# *PlayStation Movie Compression & Playback Issues*

- ▶ Improving Movie Quality
- ▶ Interleaving Data With Movies
- ▶ Using A Movie As A Texture

# *Improving Movie Quality*

- ▶ Changing the data before compression
  - Preprocessing & filtering
  - 24-bit versus 16-bit
  - Custom Quantization Tables
- ▶ Changing Compression Ratio
  - Frame rate
  - Audio space considerations

# *Improving Movie Quality*

## ▶ Preprocessing & Filtering

- Using a “blur” or “despeckle” filter will remove small bits of video noise.
  - Blur
  - Blur More
  - Despeckle
  - Gaussian Blur

# *Improving Movie Quality*

## ▶ Preprocessing & Filtering

- Especially helpful with captured live video sequences
- Removes noise which uses up valuable space in the compression process.
- More space available to compress remainder of image where there's important detail.
- Use trial and error to determine best results.

# *Improving Movie Quality*

- ▶ Preprocessing & Filtering
  - Video Editing Software
    - Adobe Premiere
    - Asymetrix Digital Video Producer
    - Corel Lumiere
    - Star Media Video Action NT
    - Ulead Media Studio Pro

# *Improving Movie Quality*

## ▶ 24-bit versus 16-bit

- No increase in compressed data size
- 8-bits each for Red, Green, & Blue instead of 5-bits
  - No semi-transparent pixels
    - 16-bit mode uses 5-bits each for red, green, & blue, plus 1 bit for indicating a semi-transparent pixel.

# *Improving Movie Quality*


- ▶ 24-bit versus 16-bit
  - Rendered movies can look much better
    - Smoother gradients
  - Captured live video shows less improvement



# Improving Movie Quality

## ▶ 24-bit versus 16-bit

- Beware of CD/MDEC DMA conflict

- Data can be corrupted if CD sector transfer is initiated while MDEC DMA is happening.
- Sample movie playback code shows fix.
  - \PSX\SAMPLE\CD\MOVIE\TUTO0.C
  - In the library routine **StCdInterrupt()** which is used as the callback routine by **CdRead2()**, if MDEC is currently processing data, it defers the CD sector transfer and sets a flag instead.
  - In MDEC callback routine **strCallback()**, if flag  set, transfer CD sector by calling **StCdInterrupt()** when done transferring MDEC data.

# *Improving Movie Quality*

- ▶ 24-bit versus 16-bit means increased bus usage
  - 24-bit mode requires 50% more data to be transferred from MDEC using **DecDCTOut()**.
    - For 320x240, additional 76800 bytes per frame
      - 2304000 bytes per second @ 30fps
  - 24-bit mode requires 50% more data to be transferred to VRAM using **LoadImage()**.
    - For 320x240, additional 76800 bytes per frame
      - 2304000 bytes per second @ 30fps

# *Improving Movie Quality*

## ▶ Custom Quantization Tables

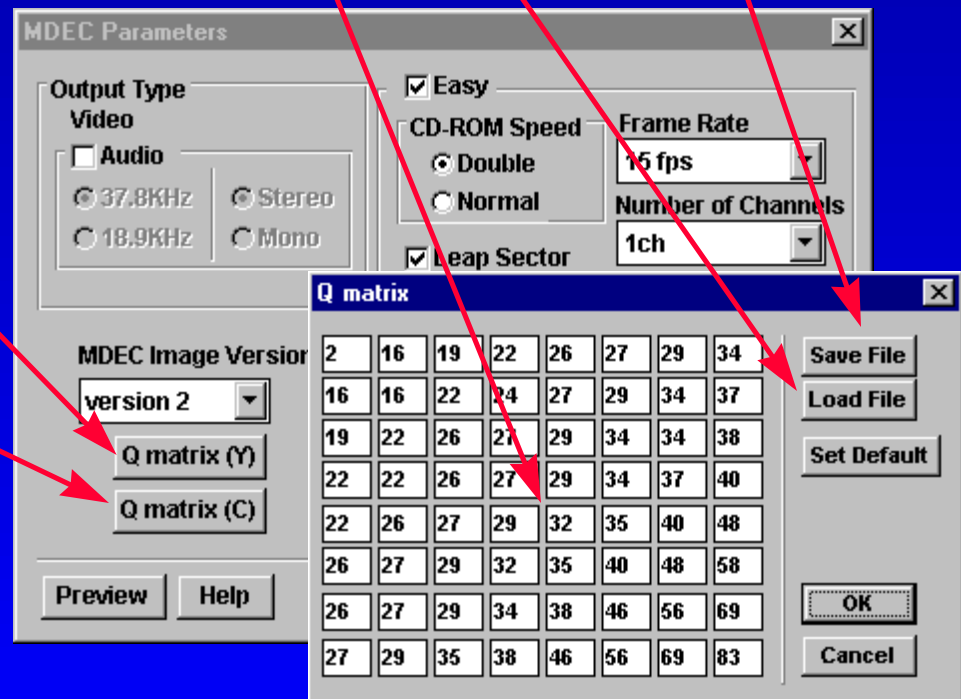
- Quantization Tables are normally optimized for mid-tones & highlight detail.
- Customized tables can improve shadow detail if that's more important.
- Gamma correction (contrast) and color tint can be adjusted at playback by adjusting custom table.

# Improving Movie Quality

## ▶ Custom Quantization Tables for Compression

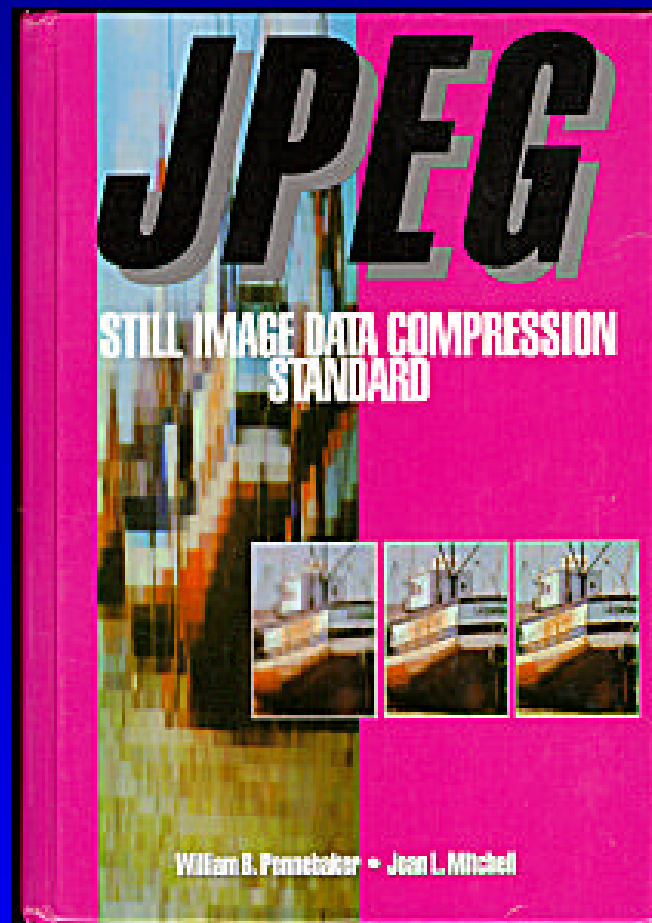
- Movie Converter allows you to edit, load, and save customized tables.

- Q matrix (Y) table controls brightness quantization
- Q matrix (C) table controls color quantization
- Only simple manual editing available.
- Finding right values done via trial and error.



# Improving Movie Quality

- ▶ Custom Quantization Tables
  - Why no tools to create quantization tables automatically?
    - Not a PlayStation specific problem... making customized quantization tables for JPEG-style compression on just about any platform is sort of a do-it-yourself project
    - See *JPEG Still Image Data Compression Standard* by Pennebaker & Mitchell for detailed technical information.



# *Improving Movie Quality*

- ▶ Decompressing Using Custom Quantization Tables
  - **DecDCTGetEnv()** copies current quantization tables from MDEC.
    - Runtime adjustments for gamma correction (contrast) and color tint may be done.
  - **DecDCTPutEnv()** copies customized quantization tables to MDEC.

# *Improving Movie Quality*

- ▶ Decompressing Using Custom Quantization Tables
  - **DecDCTReset(1)** halts decompression but keeps any customized tables that have been copied over.
  - Sample code demonstrates how to do this.
    - `\PSX\SAMPLE\PRESS\TUTO\TUTO7.C`

# Improving Movie Quality

## ▶ Adjust Your Movie's Frame Rate

- Consider playing movies at 20 fps instead of 30.
  - Works better for rendered movies than for live-action video.
    - Captured live video can appear jerky due to missing frames.
    - Movies should be rendered at 20 fps instead of 30 fps.
- Some movies may even work OK at 15 fps

Audio Format	Sectors Per Second (double-speed)	Video Sectors Per Frame		
		30fps	20fps	15fps
37.8KHz Stereo	37 audio, 113 video	3.76	5.65	7.53
37.8KHz Mono	18 audio, 132 video	4.4	6.6	8.8
18.9KHz Stereo	18 audio, 132 video	4.4	6.6	8.8
18.9KHz Mono	9 audio, 141 video	4.7	7.05	9.4



# *Improving Movie Quality*

## ▶ Audio Considerations

- Using lower-quality audio means less compression of video data is required, so you get better video quality
  - Mono -vs- Stereo
  - 18.9 kHz -vs- 37.8 kHz
  - Use MIDI instead of XA audio

# Improving Movie Quality

## ▶ Audio Considerations

- Mono -vs- Stereo
- 18.9 kHz -vs- 37.8 kHz

CD speed	Audio Format	Interleave Cycle
300 kps	37.8KHz Stereo	1 of every 8 sectors
300 kps	37.8KHz Mono	1 of every 16 sectors
300 kps	18.9KHz Stereo	1 of every 16 sectors
300 kps	18.9KHz Mono	1 of every 32 sectors
150 kps	37.8KHz Stereo	1 of every 4 sectors
150 kps	37.8KHz Mono	1 of every 8 sectors
150 kps	18.9KHz Stereo	1 of every 8 sectors
150 kps	18.9KHz Mono	1 of every 16 sectors

# Improving Movie Quality

## ▶ Audio Considerations

- Mono -vs- Stereo
- 18.9 kHz -vs- 37.8 kHz

CD speed	Audio Format	Sectors Per Second
300 kps	37.8KHz Stereo	37 audio, 113 video
300 kps	37.8KHz Mono	18 audio, 132 video
300 kps	18.9KHz Stereo	18 audio, 132 video
300 kps	18.9KHz Mono	9 audio, 141 video
150 kps	37.8KHz Stereo	37 audio, 38 video
150 kps	37.8KHz Mono	18 audio, 57 video
150 kps	18.9KHz Stereo	18 audio, 57 video
150 kps	18.9KHz Mono	9 audio, 65 video

# *Interleaving Data With Movies*

- ▶ Methods Of Interleaving
- ▶ How Much Space Is Available
- ▶ Using MOVCONV and MOVPACK
- ▶ Custom Tools For Interleaving

# *Interleaving Data With Movies*

## ▶ Methods of Interleaving

- Creating & Recovering Unused Space

- The number of sectors used to store each frame of a movie usually varies back and forth.
  - You can't really have 4.3 sectors per frame, so you have some frames which are 4 sectors and others which are 5 sectors.

# *Interleaving Data With Movies*

## ▶ Methods of Interleaving

- Allocating Space for User-defined data
  - Tell MOVCONV to use 6 sectors per frame instead of telling it to create a movie that runs at 20 fps.
    - Movie will have 120 video frames per second instead of 132.
    - 18 sectors will be used for 18.9 kHz stereo XA audio
    - 12 sectors per second available for program-specific data.
    - Total 150 sectors per second (double-speed)
- **Downside is this process requires customized tool to interleave user data and movie data.**

# *Interleaving Data With Movies*

- ▶ How Much Space Is Available
  - Depends on method of interleaving
  - Depends on how much video quality you're willing to sacrifice
  - Depends on audio requirements

# Interleaving Data With Movies

Audio Format	Audio Sectors Per Second	Non-Audio Sectors Per Second	Sectors Per Video Frame	Movie Frames Per Second	Video Frames Per Second	Leftover Frames Per Second
<b>No Audio</b>	<b>0</b>	<b>150</b>	<b>5</b>	<b>30</b>	<b>150</b>	<b>0</b>
<b>18.9 kHz Mono</b>	<b>9</b>	<b>141</b>	<b>4</b>	<b>30</b>	<b>120</b>	<b>21</b>
<b>18.9 kHz Stereo</b>	<b>18</b>	<b>132</b>	<b>4</b>	<b>30</b>	<b>120</b>	<b>12</b>
<b>37.8 kHz Mono</b>	<b>18</b>	<b>132</b>	<b>4</b>	<b>30</b>	<b>120</b>	<b>12</b>
<b>37.8 kHz Stereo</b>	<b>37</b>	<b>113</b>	<b>3</b>	<b>30</b>	<b>90</b>	<b>23</b>
<b>-----</b>						
<b>No Audio</b>	<b>0</b>	<b>150</b>	<b>7</b>	<b>20</b>	<b>140</b>	<b>10</b>
<b>18.9 kHz Mono</b>	<b>9</b>	<b>141</b>	<b>7</b>	<b>20</b>	<b>140</b>	<b>1</b>
<b>18.9 kHz Stereo</b>	<b>18</b>	<b>132</b>	<b>6</b>	<b>20</b>	<b>120</b>	<b>12</b>
<b>37.8 kHz Mono</b>	<b>18</b>	<b>132</b>	<b>6</b>	<b>20</b>	<b>120</b>	<b>12</b>
<b>37.8 kHz Stereo</b>	<b>37</b>	<b>113</b>	<b>5</b>	<b>20</b>	<b>100</b>	<b>13</b>
<b>-----</b>						
<b>No Audio</b>	<b>0</b>	<b>150</b>	<b>10</b>	<b>15</b>	<b>150</b>	<b>0</b>
<b>18.9 kHz Mono</b>	<b>9</b>	<b>141</b>	<b>9</b>	<b>15</b>	<b>135</b>	<b>6</b>
<b>18.9 kHz Stereo</b>	<b>18</b>	<b>132</b>	<b>8</b>	<b>15</b>	<b>120</b>	<b>12</b>
<b>37.8 kHz Mono</b>	<b>18</b>	<b>132</b>	<b>8</b>	<b>15</b>	<b>120</b>	<b>12</b>
<b>37.8 kHz Stereo</b>	<b>37</b>	<b>113</b>	<b>7</b>	<b>15</b>	<b>105</b>	<b>8</b>



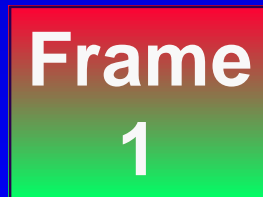
# *Using A Movie As A Texture*

- ▶ Data Layout
- ▶ Frame Rates
- ▶ Data Padding & System Bandwidth

# *Using A Movie As A Texture*

## ▶ Data Layout — Option 1

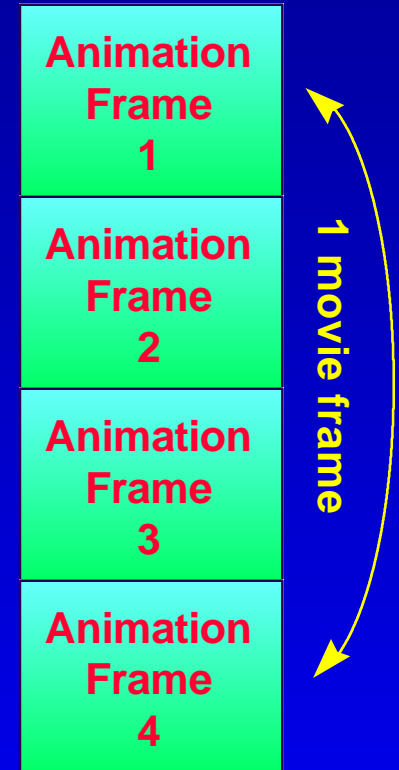
- 1 frame of the movie contains 1 frame of the texture animation
  - Timing is critical
    - Problems if CD errors occur
    - Synchronization problems if game frame rate changes
  - Takes less memory



# Using A Movie As A Texture

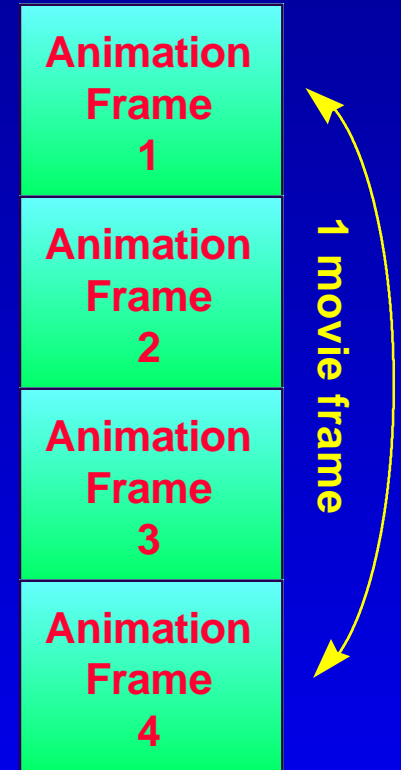
## ▶ Data Layout —Option 2

- 1 frame of the movie contains several frames of the texture animation
- Timing is less critical
  - Can recover from CD errors
- Uses more memory, but data can be left compressed until needed.



# *Using A Movie As A Texture*

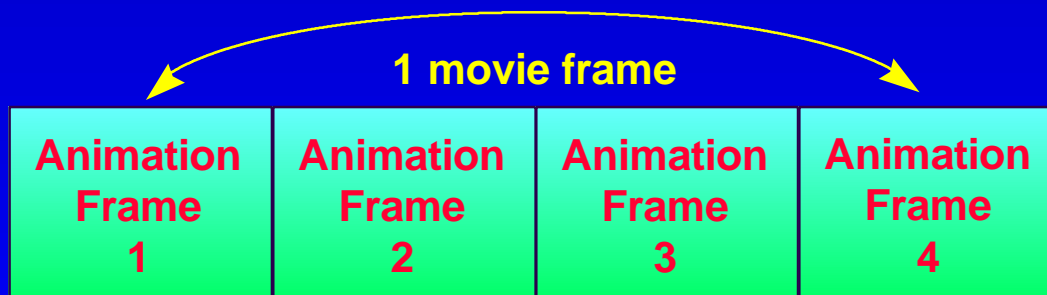
- ▶ Data Layout —Option 2
  - If texture is 16-pixels wide or less, then stack frames together as a vertical strip for most efficient decompression & memory usage.
  - Send data for each frame to MDEC for decompression as needed



# *Using A Movie As A Texture*

## ▶ Data Layout —Option 2

- If texture is wider than 16-pixels, then places frames side by side as a horizontal strip for most efficient decompression & memory usage.
- Send data for each frame to MDEC for decompression as needed



# *Using A Movie As A Texture*

## ▶ Frame Rates

- Frame rate of the movie being played does not have to match texture animation frame rate if each movie frame contains multiple animation frames.
- Must be able to skip frames if game's frame rate is not constant.
  - Use standard ring buffer mechanism for movie playback & decoding
  - Alter the *Transfer Movie Frame To VRAM* portion of the code as required.

# *Using A Movie As A Texture*

## ▶ Data Padding & System Bandwidth

- If texture data size does not use full bandwidth of CD, consider these ideas.
  - Use single-speed CDROM playback instead of double-speed.
  - Interleave XA audio & texture data together.

# *Using A Movie As A Texture*

## ▶ Data Padding & System Bandwidth

- If texture data size does not use full bandwidth of CD, consider these ideas.
  - Use Multi-channel XA audio data
    - Interleave with XA audio needed by other portions of the game which do not perform movie-texture streaming.
      - When you're playing that XA audio data, just ignore the texture movie data.
    - Interleave with dummy (silent) XA audio data



# *Using A Movie As A Texture*

## ▶ Data Padding & System Bandwidth

- If you can keep the CDROM moving, you can use less memory to store the texture movie.
- If you have the memory to spare, read ahead, pause CD, continue reading when ring buffer reaches low water mark.
  - Can't do this if texture movie is interleaved with XA audio

*The End*