VIDEO SUMMARIZATION

Developer: Jiaqi Liu & Lisha Xu

Catalog

- Project Description and Result
- Classes and Functions:
  - *Read-in Files*
  - *Creating shots*
  - *Analyzing Video*
  - *Analyzing Audio*
  - *Feed Back Result*
- Problems and improvements
  - *Accuracy of shots division*
  - *Sound coherence*
  - *Appropriate weight and threshold*
  - *Audio Pause Function*

Project description

- This project concentrates on extracting the essence of a video. The programm divides video into shots for a better performance in graphic and audio analysis. Luminance and color level are calculated in each shots and later being checked with a threshold to find out the most significant shots.
- 9 files:
  - *videoDisplay.java*
  - *ImageDisplay.java*
  - *PlaySound.java*
  - *VideoAnalyzer.java*
  - *audioAnalyze.java*
  - *summarize.java*
  - *ComposeVideo.java*
  - *PlayWaveException.java*
  - *PlayWaveFile.java*

Display

- Compile
  - *javac ./../*.java*
- VideoDisplay.java (main display class)
  - *Input format:*

    java videoDisplay.java [RgbFolderName] [AudioFileName]

    eg. java videoDisplay.java .\concert\ .\concert.wav
  - *The program uses 2 threads to play video and audio at the same time*
- ImageDisplay.java
  - *Read rgb images and show in Jframe*
  - *Get the frame rate through dividing the number of frames by audio time length*
  - *Ultilize timerTask to flush the images*
  - *Calibrate frame number by the audio playback position*

VideoAnalyzer.java

- Create shots – color histogram
  - *R,G,B are divided into 4 levels. The whole color space is seperated into 64 parts.*
  - *Generate the color histogram of the whole frame by counting the pixel number in*
  - *each color area.*
  - *Calculate the difference of two adjacent frames' color histogram*
  - *Record shot interval positions where the differences are very large.*

- Image Analysis – average luminance
  - *Calculate the average luminance of each frame*
  - *Get the difference of two adjacent frames' luminance*
  - *Divide the number of differences over threshold by the length of the shot*
  - *Add the result to image weight list*

audioAnalyzer.java

- Audio analysis – relative audio strength
  - *Read the audio information of each shot to one buffer each time*
  - *Calculate average strength of this audio shot*
  - *Find the maximum and minimum strength of this audio shot*
  - *Shift the strength by adding the absolute value of minimum strength to eliminate*
  - *the negative value*
  - *Calculate the relative audio strength through dividing the average strength by maximum strength*

Compose the result

- summarize.java
  - *Combine the image weight and audio weight in appropriate rate*
  - *Select high combined weight shots for summarized video*

- ComposeVideo.java
  - *Save the highlight shots' frames into a new folder*
  - *Compose the highlight shots' audio into a new audio wav file.*
  - *Add shots between two highlight shots to ensure continuity*

Problems and improvements

- Accuracy of shots division
  - *Try to add the block luminance. But it's ineffective*
- Sentence coherence
  - *When different camera shots are used to describe one scene, sentences could be seperated if we just extract some shots.*
- Appropriate weight and threshold
  - *Attempt coefficients from the given example. Maybe not be suitable for all situations*
- Pause
  - *Because the design of audio buffer, the video will play a little more seconds after the pause button was clicked.*

Thanks for watching