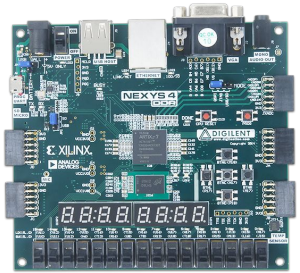
JAW Dropping Visual Effects via Audio Spectral Analysis

Julian, Aaron, Wings

Project Overview

- Audio based visual effect generator
- Basic
  - Proof of concept with simple audio and image inputs and with simple color transformation
- Expected
  - Desired visual effects to more complex inputs and more types of transformations
- Stretch
  - Visual effects to live inputs



Basic Goal Implementation

- Input audio from short recording or direct stream from cell phone/laptop
- FFT and filters applied to create appropriate number of bins based on number of contours detected on input image
- Input image characteristics
  - Simple
  - Closed contour
  - High contrast
  - Polygon
  - Stored in memory
- Edge detection completed in MatLab with each edge being outputted as a “sprite”
  - Each contour/sprite will map to a frequency bin
- One of the following transformed will be applied to the contours as a function of the frequency
  - Color change
  - Entire image movement across the screen
  - Individual edge pulsation in response to different frequencies

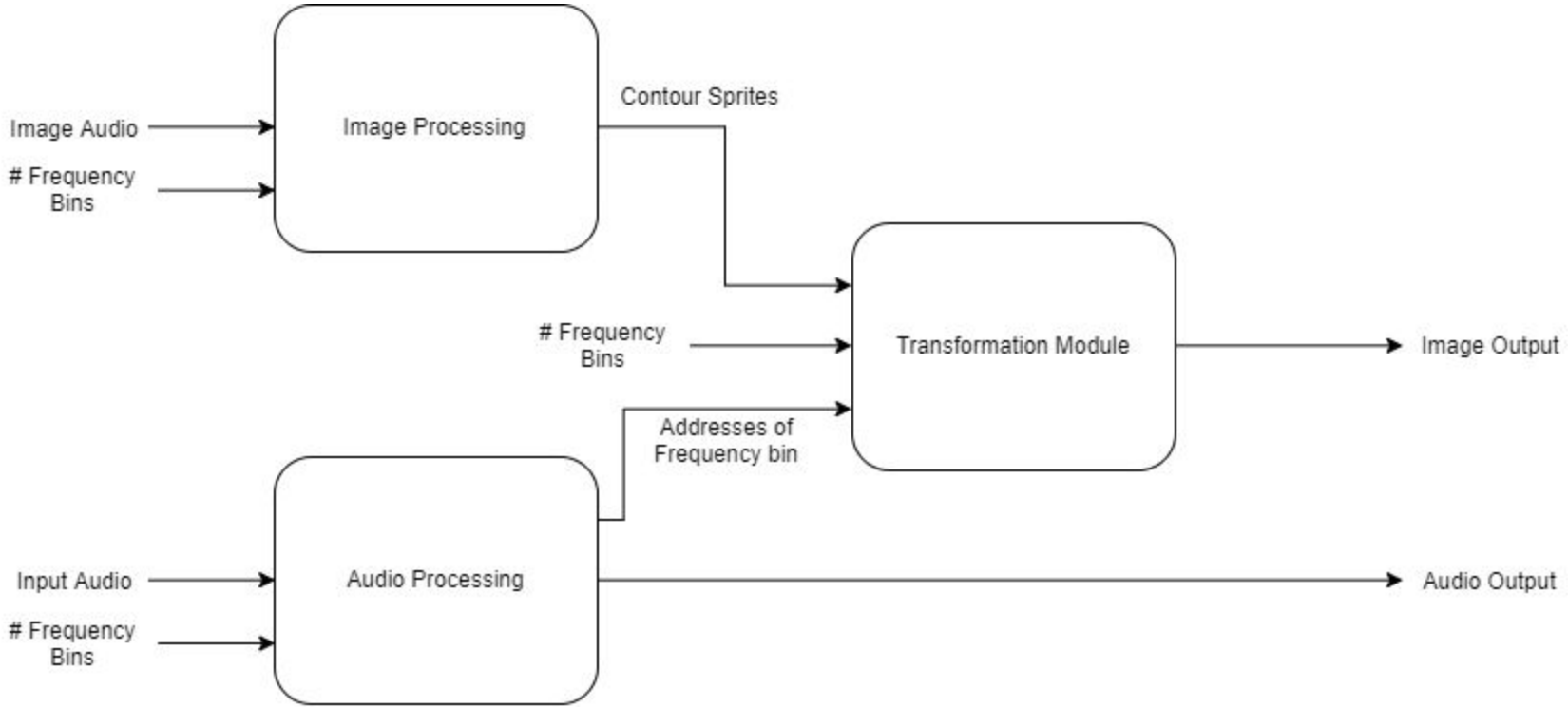
Expected Goal Upgrades

- Image input from camera
- Edge detection implemented in Verilog
- More complex contours than simple polygons
  - Requires normal vector estimation
- Better filtering
- Higher resolution
  - Approximate curves as lines, thus the smaller the portion of the curve approximated by each line, the higher the resolution
- Multiple, selectable transformations

Stretch Goal

- More transformations
  - Efficient computations
- Live audio sampled via a microphone
  - Appropriate sample rates, hardware quality, and filtering
- Live video from a stream of frames taken by the camera
  - Latency issues

High Level Block Diagram



Audio Processing Module

- Read in audio data
  - Live or recorded samples
  - Stored songs in SD card
- Generate FIR filter coefficients in MatLab
- Create multiple filters to establish bins as well as create interesting effects
- Apply FFT to filter outputs with bins defined either by number of possible contours or defining the number of possible contours via the bin selection

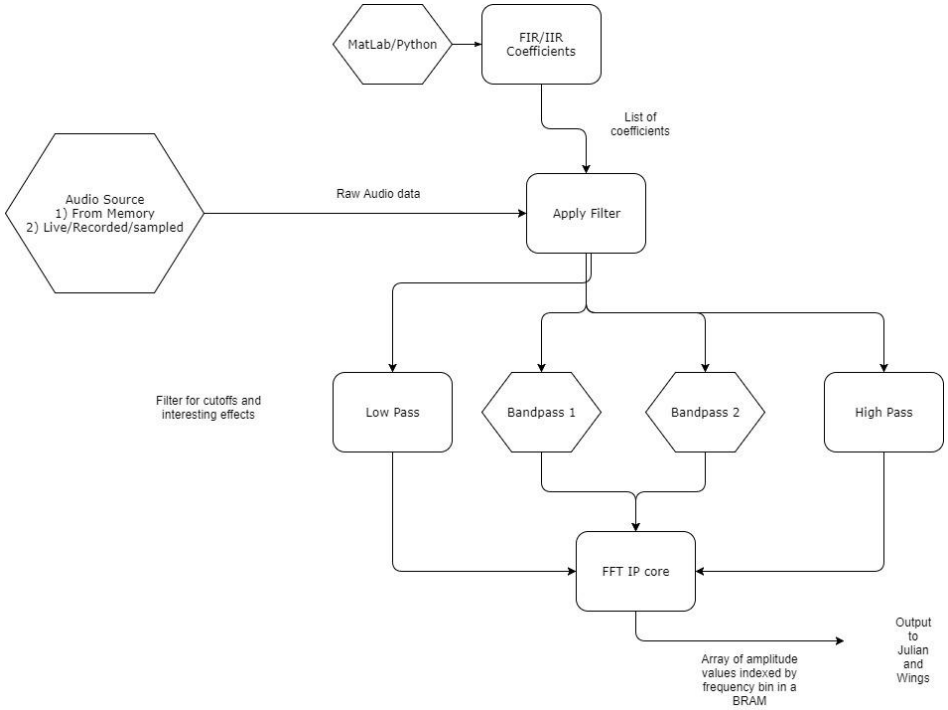
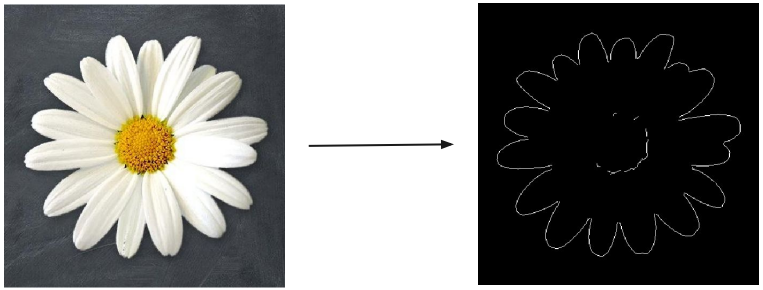
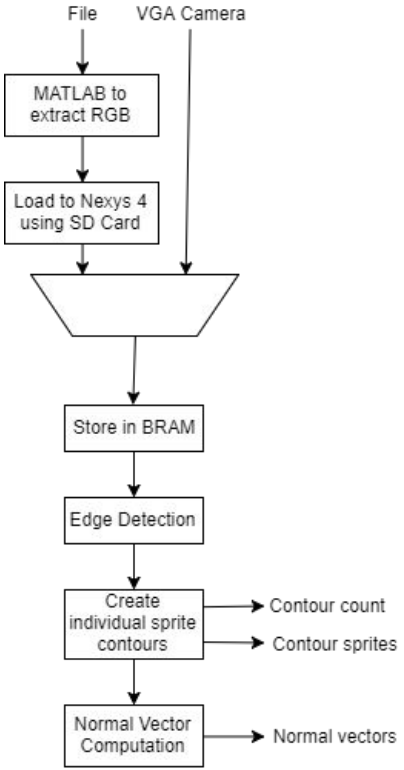


Image Processing Module

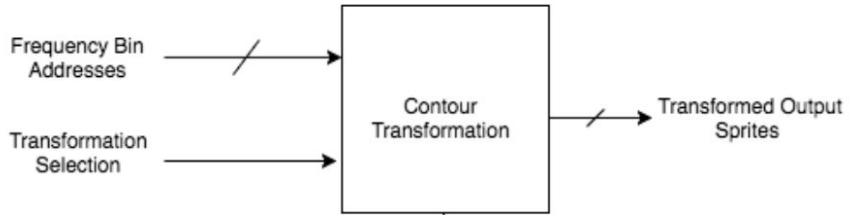
- Input image either from file or VGA camera
- Edge detection
  - RGB → Black and White
  - Gaussian Smoothing
  - Sobel Algorithm
- Separation of edges to individual sprites
  - Normal vector for each sprite





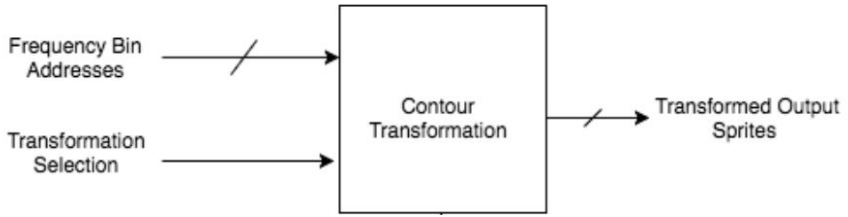
Transformation Module: Inputs

- Input Contour Sprites and Normal Vector
- Transformation selected by user
  - Transformation 1 - Color Gradient
  - Transformation 2 - Amplitude Mapping
  - Transformation 3 - Amplitude Mapping with effects



Transformation Module: Outputs

- Image Sprites, with (x,y) array
- Module modifies x,y array according to transformation
- Modified image sprites output to VGA module (hcount, vcount)



Interfaces

- FFT Output: data addresses and bin count
  - Transformation mappings
  - Contour Definition
- Edge detection/contour definition: sprites per contour, normal vectors, original image
  - Transformation mappings
  - Contour count determines frequency bin division in basic implementation
- Transformation mappings: modulated image
  - XVGA module

	Nov 5	Nov 12	Nov 19	Nov 26	Dec 3
Interface with SD Card					
Pictures from camera + SD Card					
Create Sprite Contours					
Write updatable sprite modules					
IP Cores: Clock Wizard and FFT					
Skeleton Contour Module; Implement Color Transformation					
Edge Detection					
Audio filters					
Normal Vectors					
Integrate Color Transformation					
Work on complex transformations					
Debugging + Stretch Goals					