Sorbotic Videomation: An Active Color Driven Sound Module

Matthew Bowman, Matthew Aldrich
Yale University

Abstract

A video to sound synthesizer has been designed and realized in Cycling 74's MAX environment. By using discrete modules within MAX the group successfully filters note data from moving video and gives an audio representation of it.

1 Introduction

This paper briefly describes the methods that were employed in making this intelligent synthesizer. Implemented in Cycling 74's MAX processing environment, the sound module is controlled by color driven events. By mapping the picture to an array and applying a color filter on the image, basic notes, given particular visual events, can be realized.

2 MAX Implementation

The synthesizer is comprised of various modules that allow for the input and processing of real time data. For instance, the midi keyboard can toggle and play any specific sequence of video. In addition, the video sequence can be left to cycle on its own.

2.1 Synthesizer Input

The main user interface allows for control of video by loading and unloading quicktime movies. This simple module can be found in the play/stop region of the attached schematic. It allows for quicktime video usage.

As mentioned previously, the synthesizer allows the user to control what frame, or how long to play a specific frame in the video sequence by pressing keys. In the schematic attached to this write-up, this information is generated by the notein module. By measuring the velocity and or key, specific frame counts in the video can be jumped to.

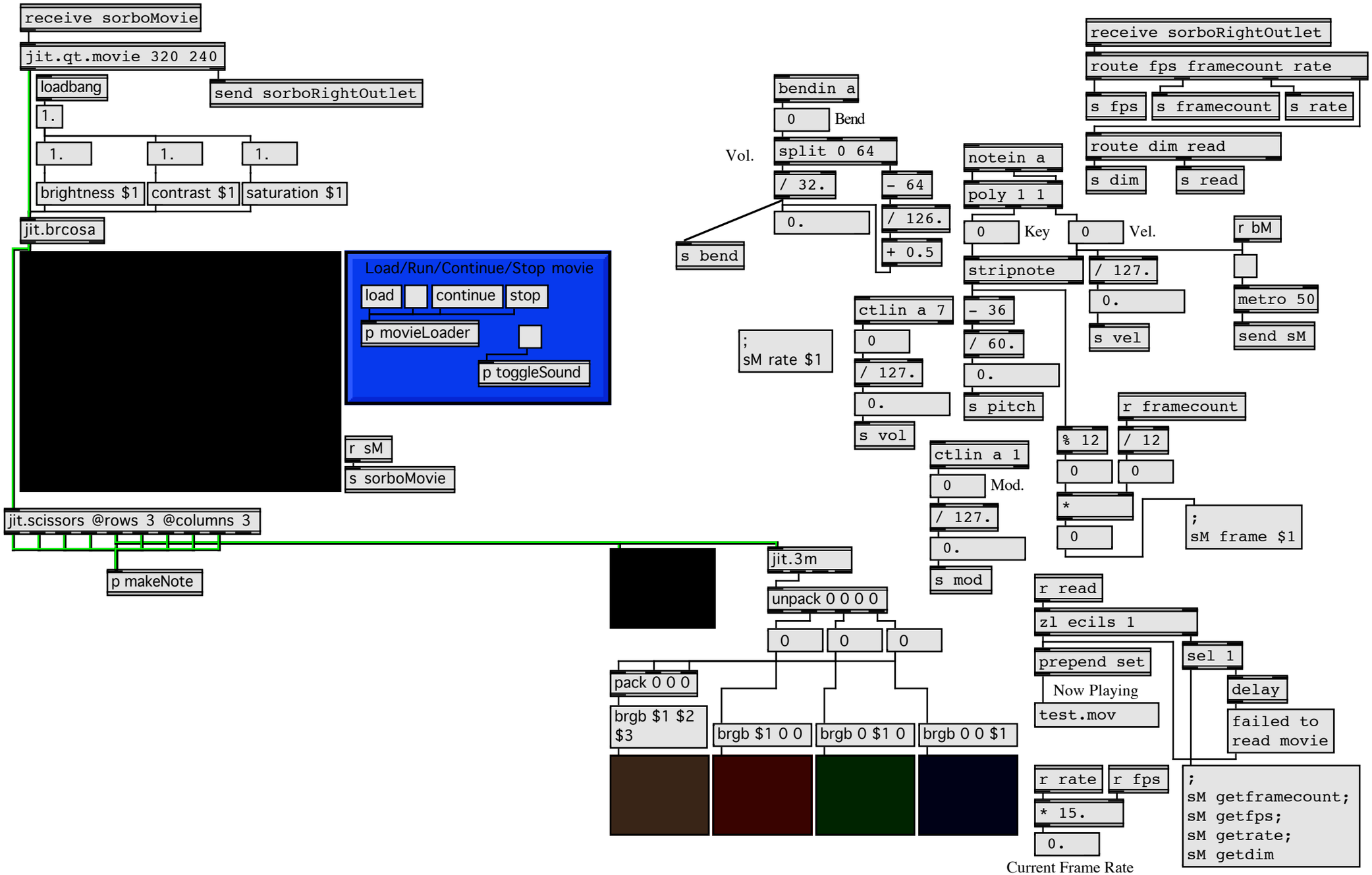
2.2 Synthesizer Output

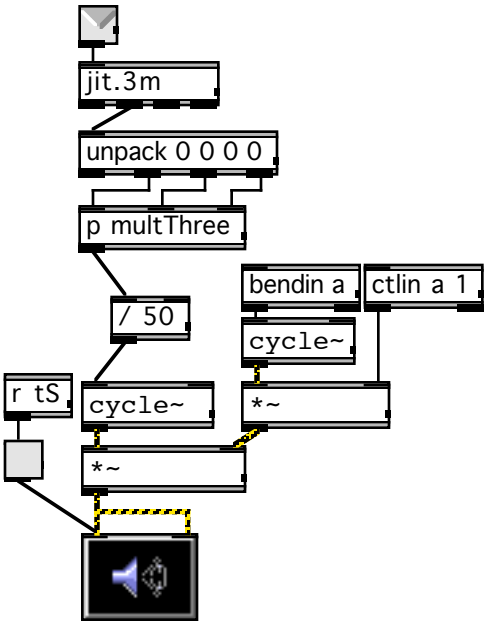
This project seeks to take video and have those colors present in the video control some sound events. How is this done? First and most obvious, the colors must be filtered from the original image. This is easily implemented in MAX with RGB filter. The next issue is how to address multiple notes. Technically, if you filter the image, you could have 3 different notes, however this is not very interesting. Hence, the group invented the SORBOTIC Video Engine. The design is simple, map an n x n array to the video sequence. In this case, to illustrate the idea, a 3 x 3 matrix was used. Each individual element in that matrix is RGB filtered. In order to get a note from these 3 values, an average of all these colors is taken. The best way to correlate the colors to a note is still an open question. From this stage, the output is fed into a simple sound module to illustrate the effects of the synthesizer. In addition, the frequency of the FM modulator and volume can be controlled from the midi keyboard to aid in illustrative examples.

3 Conclusion

In its current state, the synthesizer averages each element of an arbitrary matrix of video and outputs that data to a synthesizer in which an audio representation of the visual material is given. For example, a ball moving across the screen will give a different sequence of tones compared to a face of someone talking. However, the data generated by our method is not musically pleasing, a future implementation could focus on better blending or filtering of the music and then applying that data in an algorithmic manner to generate meaning notes.

The group would like to thank Dr. Suttor for advising them on this project as well as Dan Sobo - bang means do it!


receive sorboMovie
jit.qt.movie 320 240
loadbang
send sorboRightOutlet
1.
1.
1.
1.
brightness $1
contrast $1
saturation $1
jit.brcosa
Load/Run/Continue/Stop movie
load
continue
stop
p movieLoader
p toggleSound
r sM
s sorboMovie
jit.scissors @rows 3 @columns 3
p makeNote
jit.3m
unpack 0 0 0 0
0
0
0
pack 0 0 0
brgb $1 $2 $3
brgb $1 0 0
brgb 0 $1 0
brgb 0 0 $1
bendin a
0
Bend
Vol.
split 0 64
/ 32.
- 64
0.
/ 126.
+ 0.5
s bend
notein a
poly 1 1
0
Key
0
Vel.
stripnote
- 36
/ 60.
0.
s pitch
ctlin a 7
0
/ 127.
0.
s vol
;
sM rate $1
ctlin a 1
0
Mod.
/ 127.
0.
s mod
receive sorboRightOutlet
route fps framecount rate
s fps
s framecount
s rate
route dim read
s dim
s read
/ 127.
0.
s vel
r bM
metro 50
send sM
r framecount
% 12
/ 12
0
0
*
0
;
sM frame $1
r read
zl ecils 1
prepend set
Now Playing
test.mov
sel 1
delay
failed to read movie
r rate
r fps
* 15.
0.
Current Frame Rate
;
sM getframecount;
sM getfps;
sM getrate;
sM getdim


jit.3m
unpack 0 0 0 0
p multThree
/ 50
bendin a
ctlin a 1
cycle~
r tS
cycle~
*~
*~