Supplementary materials

Detailed algorithm steps

Color clustering: Target color is determined using the hue, saturation and luminance (HSL) values of each pixel. For each $pixel$ we define $HueVal(pixel) \in \{0, \cdots, 360\}$, $SatVal(pixel) \in \{0, \cdots, 1\}$ and $LumVal(pixel) \in \{0, \cdots, 1\}$ to denote its hue, saturation and luminance values respectively.

Let $f_{color}(pixel) = \text{argmin}_{color \in \{Red, Green, Blue, Yellow\}} |HueVal(color) - HueVal(pixel)|$

The color of each $pixel$ will then be:

$$Color(pixel) = \begin{cases} Black, & LumVal(pixel) < DarkThr \\ White, & \begin{pmatrix} LumVal(pixel) > BriThr \mid \\ \begin{pmatrix} LumVal(pixel) > DarkThr \ \& \\ SatVal(pixel) < SatThr \end{pmatrix} \end{pmatrix} \\ f_{color}(pixel), & else \end{cases}$$

Where $DarkThr = 0.05$, $BriThr = 0.9$, and $SatThr = 0.13$.

The HSL space can be portrayed as a cylindrical space where the pixel hue value is the angle of rotation around the cylinder's base circle, thus making both 0 and 360 hue values red. This cylindrical representation demands an additional manipulation in the f_{color} function that does not appear in the equation for simplicity reasons. Sample color clustering results are shown in Fig. 3A.

Instrument-to-color mapping: Red is mapped to a Reggae organ, Green to a Rapman's Reed, Blue to Brass instruments, Yellow to string instruments, White to a Choir and Black to silence (Fig. 1A).

Instrument recording: A set of 24 wave files per instrument were recorded using a sampler (Native Instruments Kontakt 4; 44100Hz, 16-bit, 1-channel, 5 sec duration) representing all notes ranging from C2 up to G6 on a pentatonic scale (C, D, E, G, A), corresponding to the 24 vertical pixels accommodated by the EyeMusic. Recordings were then normalized (for each sample, $originalSample$, in the recording) according to the following:

Let $Samples_{file}$ denote all the samples between the 2nd and 4th seconds for each recorded .wav $file$. Let $GlobalMaximum$ denote the largest sample within all recorded .wav files. Then

$$normalizedSample = \frac{originalSample}{SD(Samples) \times GlobalMaximum}$$

Pixel attenuation: The volume of each pixel is attenuated using the luminance value of that pixel and is set per sample according to:

$$attenuatedSample = originalSample \times LumVal(pixel)$$

Resulting audio file construction: For each column, the algorithm combines all its non-silent rows in order to create the resulting audio representing that column. The number of samples constituting a column varies and depends on the configurable output audio file duration. Let $sample_i r$ be sample number i in row number r, then:

$$resultSample_i = \frac{(sample_i 1 + sample_i 2 + \cdots + sample_i N)}{\sqrt{N}}$$

Where i spans the number of samples in the pre-recorded normalized .wav files that constitute the relevant column and N is the number of non-silent rows.

Resulting audio file duration: The user can configure the desired output duration which is set by default to 2 sec. This yields 50 msec of playback time for each column.