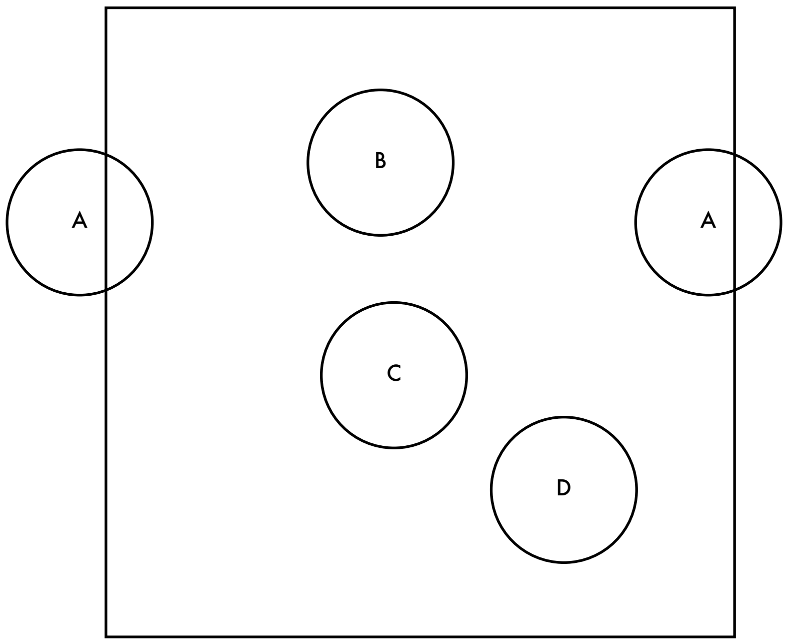
Assignment 2 — Sound Field — Simon Demeule

Concept

The concept behind my interactive piece was simply to link visual movement with sound location, timbral quality and harmonic movement.

The piece fills the screen with a set of orbs — some blurry, some that are only seen through their "refraction" of the background. The blurry orbs all possess an associated voice, synthesized through two-operator frequency modulation. Their panning (sound spatialization) is given by their left/right position on screen, while their modulation index (in other words the complexity of their spectrum) is given by the up/down position on screen.

When the user's mouse is not over the canvas, the balls roam freely, driven by a randomly calculated speed vector. When the user's mouse is over the canvas, the balls converge to it's location. Flicking the cursor in and out of the canvas allows the user to throw the balls across the screen. Rather than bouncing, the balls wrap across the screen, as illustrated below. Upon wrapping, the circles can trigger a change in the pitch they hold, changing the overall harmony. Clicking will cause all circles to change pitch.

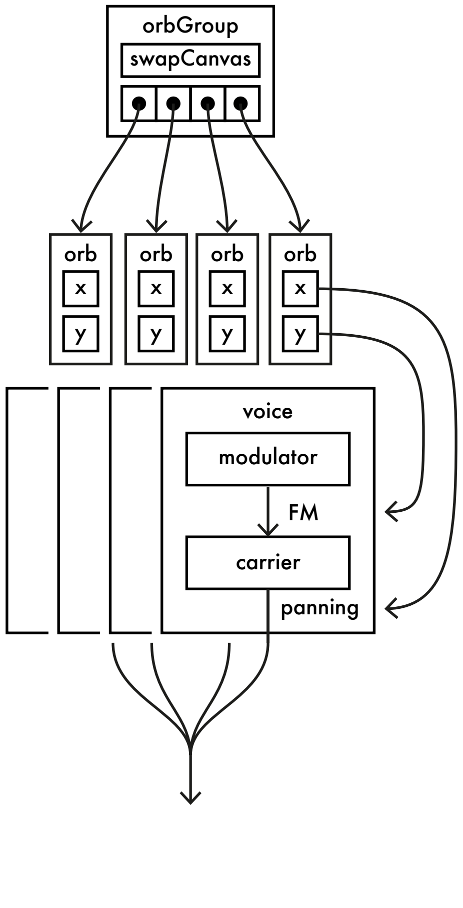


Process

The project was derived from the code produced in exercise 2, which already contained most of the visual part of the code: orb objects that contained functions to draw themselves and interact with the mouse.

A new orbGroup object was created to allow different types of visual effects applied to the fill of the orbs. The filling process relies on creating a duplicate canvas in memory (called swapCanvas) prior to the painting of the orbs that contains the pixels currently inside the canvas for each orbGroup. (Put explicitly, the change was to go from a global swapCanvas allowing one effect only to a swapCanvas per group.) On this duplicate canvas, an effect is applied (blur in the case of blurry orbs, none for the other orbGroup), and then, for every orb of each orbGroup, circle clipping masks are dawn at each orb location, and then filled with the image contained in swapCanvas, displaced by an amount inverse of the speed of each orb. This is how "refraction" is implemented for the other orbGroup.

The main addition is the audio part of the project. All audio is synthesized live through AudioContext. The illustration below represents the object architecture of orbGroup and orb (only containing some properties) and the voice architecture of the project. (not illustrated is a flanger implemented with a short modulated delay line).



The harmonies are generated by taking a tonic frequency and multiplying it or dividing it iteratively with number chosen at random from a list of small primes. This yields simple chords that are relatively free of dissonances.