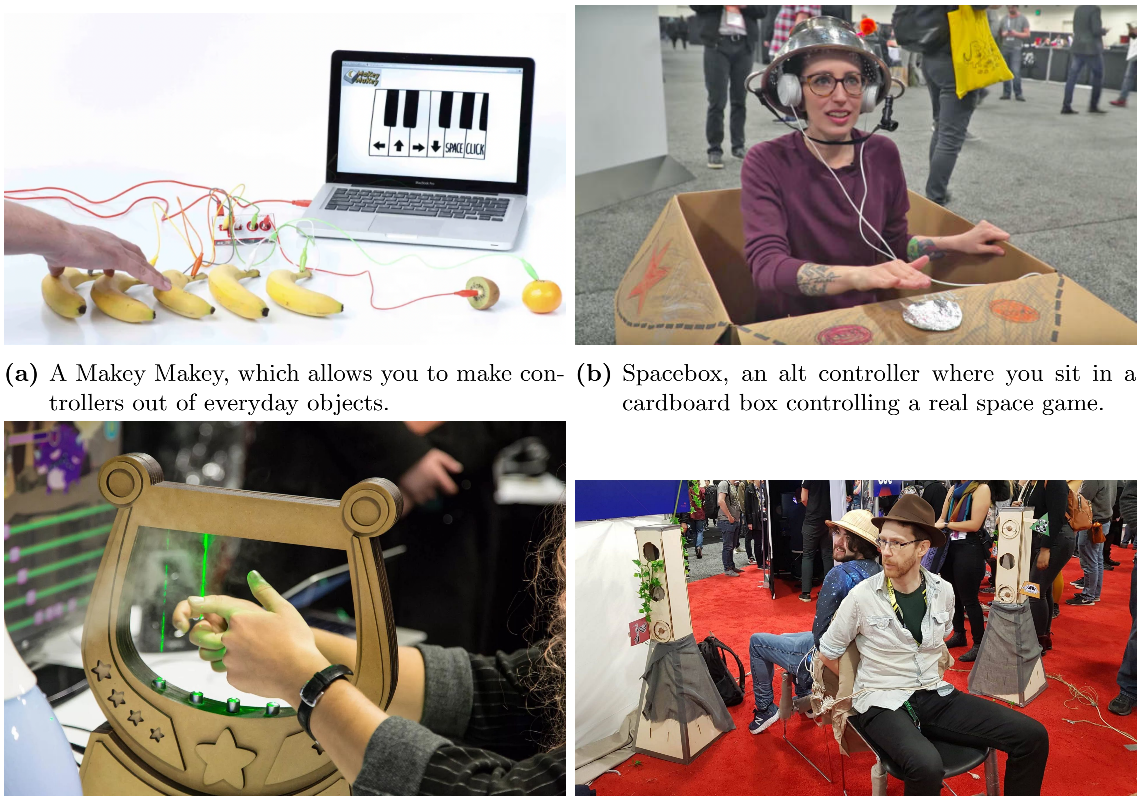
# Creative Coding - Week 2

Processing, Part 2

**Lecturer:** Dr Michael Cook (mike.cook@kcl.ac.uk)



**(a)** A Makey Makey, which allows you to make controllers out of everyday objects.

**(b)** Spacebox, an alt controller where you sit in a cardboard box controlling a real space game.

**(c)** Orpheus Quest, an alt controller using motion sensors built into a fake harp.

**(d)** Tied Escape: The Curse of Cortez, in which two players are tied to chairs.

**Figure 1:** Four examples of alternative or 'alt' controllers.

In this week's session we're going to look at building more interactive sketches with p5.js. If you've not attended Part 1, I recommend checking it out first, as it'll give you some familiarity with the tool when we use other features today. As before, p5.js has a browser-based editor, which you can find here: editor.p5js.org

I recommend registering an account on the p5.js website, which will let you save and load your scripts from any device. You can find a login/signup link in the top right of the editor screen.

## 1 Input in p5.js

Making sketches that run in a browser gives you all sorts of opportunities to get input from the user. We can take traditional inputs like keyboard buttons, less traditional inputs like audio and video, or get external devices to make custom controls. Figure 1a) shows a photo of someone using a Makey Makey, which lets you accept input to your computer from anything conductive (like a banana). Figures 1b), 1c) and 1d) all show games that were exhibited at the 'alt.ctrl.gdc' show at the Game Developers Conference - all of them are real controllers that provide input to a computer!

## 1.1 Keys and Buttons

In p5.js, the most basic ways to get input are to check when keys are pressed or mouse buttons are pressed. You can do this by defining two methods called `keyPressed()` and `mousePressed()`. Here's a quick preview of how they work:

```
function keyPressed(){
    //the key variable contains what was just pressed as a string
    if (key === 'c') {
        //do something when c is pressed
    }
    //the keycode variable refers to special system values for which key was just
    pressed (check the docs!)
    if(keyCode === UP_ARROW) {
        //do something when the up arrow is pressed
    }
}

function mousePressed(){
    //we can check which button has been pressed
    if (mouseButton == LEFT){
        //do something
    }

    //we can also get the position of the mouse cursor (at any time)
    if (mouseX < 100 && mouseY < 100){
        //do something
    }
}
```

There are a lot of similar methods/flags that you can check for key and mouse input, so do check the online documentation. For example, you can check when a key or button is initially pressed down, you can check when a key or button is released, and you can have a function that is called every frame a key or button is held down. These all let you capture different kinds of interaction. By default, someone tapping on their phone or tablet will be interpreted as a mouse button click, but there are also special functions for detecting taps specifically, too.

## 1.2 Audio and Video

You can also get input from a webcam or microphone easily:

```
function setup(){
    //Create an audio in from the microphone
    mic = new p5.AudioIn();
    mic.start();

    //Create a video in from the webcam (can also capture audio)
    capture = createCapture(VIDEO);
    capture.hide();
}

function draw(){
    //Draw the output from the webcam on the screen
    image(capture, 0, 0, 360, 400);

    //Print the volume of the microphone input
    print(mic.getLevel())
}
```

For some audio and video things you'll need to import additional libraries. In the top-left corner of your sketch is a caret arrow pointing right. Clicking this will show you the files in your sketch. You can upload files here, but there's also an index.html file that lets you import libraries. p5js imports p5.Sound by default, but certain browsers don't like it. If you have trouble running any of these examples, ask me, or check out the imports in my Mic Game example later.

We can use this basic raw information as it comes in - for example using the volume of a microphone to change colours in our sketch, or displaying the webcam image on the screen. But we can also get the raw data from these sources to do more interesting things with them.

### 1.2.1 Analysing Audio

For example, we can create a Fast Fourier Transform (FFT) of any audio signal, including microphone input. An FFT is a way of breaking down audio into many different smaller components. Each component tracks a particular frequency range in the original audio signal, letting you separate high frequencies from low ones, and is useful for analysing sounds of all kinds.

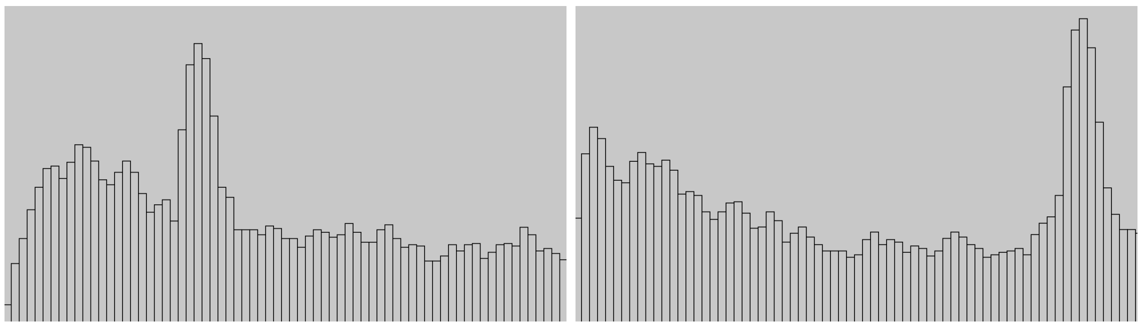
```
function start(){
    //...
    //Create a microphone input
    mic = new p5.AudioIn();
    mic.start();
    //Create an FFT analyser and connect the microphone to it
    fft = new p5.FFT();
    fft.setInput(mic);
    //...
}
```

```
function draw(){
    //...
    let fft_data = fft.analyze();

    for (i = 0; i < spectrum.length; i++) {
        //Draw a rectangle for the i'th part of the frequency analysis (max value
    is 255, so we divide by that)
        rect(i*10, height, 10, -(fft_data[i]/255)*height)
    }
    //...

    //Or you can get values for ranges
    //for example, this is between 300hz and 500hz (I think)
    let low_freqs = fft.getEnergy(300,500);
}
```

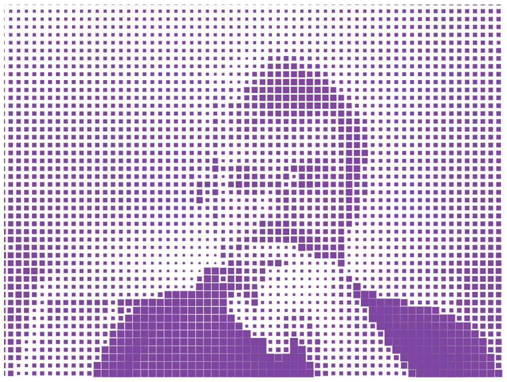
This looks very complicated, but it makes a lot more sense when you see a screenshot of this in action. Figure 2 shows two outputs from this waveform analysis of my microphone. The two peaks you can see are me whistling notes of different pitches. There's a very nice example of visualising FFT information for a song clip here. FFTs have been used in games, art and other interactive programs for ages!



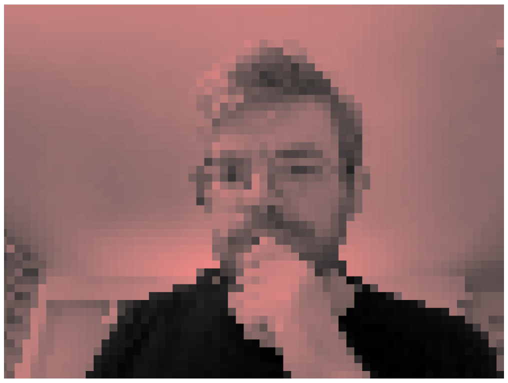
**(a)** An FFT of mic audio, whistling a low note. **(b)** An FFT of mic audio, whistling a high note.

**Figure 2:** The data from an FFT analysis of my laptop microphone. Lower frequency sound is on the left of the image, higher frequency is on the right. In both images, a narrow peak is formed by me whistling a note of different pitches.

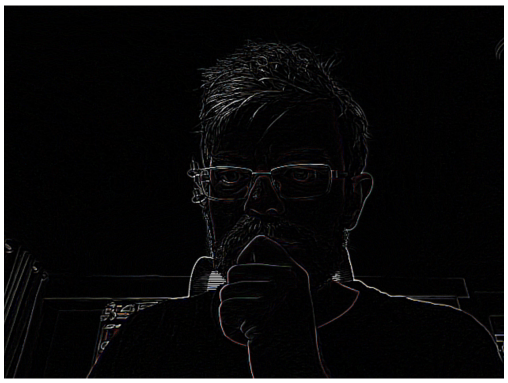
**(a)** Raw webcam input.



**(b)** Pointillism.



**(c)** Duotone colour mapping



**(d)** Edge detection.

**Figure 3:** Raw webcam input and three different ways to process it.

### 1.2.2 Analysing Video

We can also get all the pixels from our webcam image and examine or change them one-by-one. Note that the bigger the video feed image is, the more pixels there are, and the slower this process is. If you do something complicated with the pixel data, it might be too slow to run in real-time.

```
//Load the pixel data from our webcam
capture.loadPixels();
//Iterate over the pixel data (in this case, we only draw one in every twenty-five pixels, which makes it more efficient)
for (let y = 0; y < capture.height; y +5) {
    for (let x = 0; x < capture.width; x +5) {
        //Get the colour from the pixel
        //The indexing here is a little strange: the pixel data is a one-dimension array, so we need to convert (x,y) to a single index first
        var pix = (y * width + x) * 4
        //For each pixel, four numbers are stored for the R, G, B and A values of the pixel. We take the first three.
        fill(
            capture.pixels[pix],
            capture.pixels[pix+1],
            capture.pixels[pix+2])
        //Draw a 5x5 rect
        rect(x, y, 5, 5)
    }
}
```

The above code is an example of simply taking the pixel data and redrawing it exactly, albeit at a lower resolution. Figure 3 shows some ways pixel data can be drawn differently - to modify shapes, colours, or to reprocess the pixel data entirely to perform image filters. Pixel data is just data, just numbers that we can manipulate, change, interpret and display like any other information. There's a lot of cool creative stuff you can do!

https://editor.p5js.org/mtrc/sketches/rq6ndXaMW

### 1.2.3 Complex Image Processing

p5js has a lot of cool libraries, plugins and example code you can use to do more complex things. Searching online for p5js plus a feature of your choice (like facial recognition) will often get you exciting things back! Here's a really neat example: hand points detection. You can find an example from the p5js folks here:

https://p5js.org/tutorials/speak-with-your-hands/

This can do realtime detection and tracking of a single hand in a webcam feed, and give you points of the different fingers.

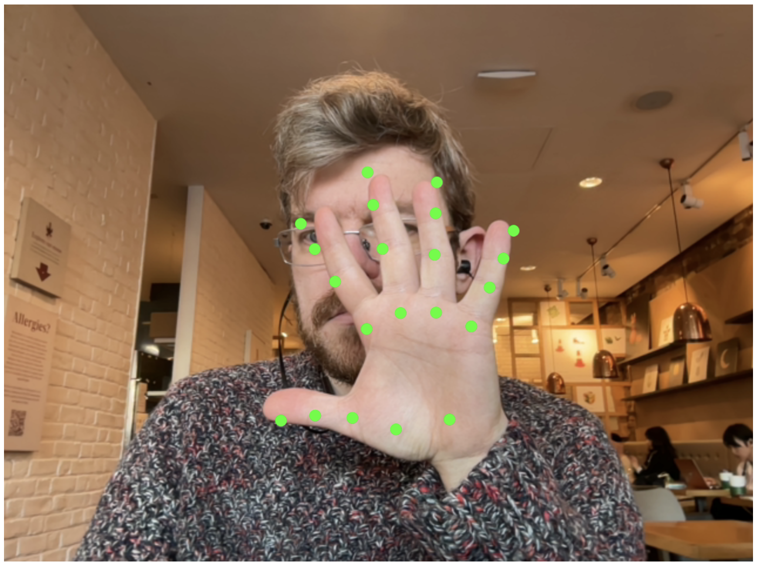
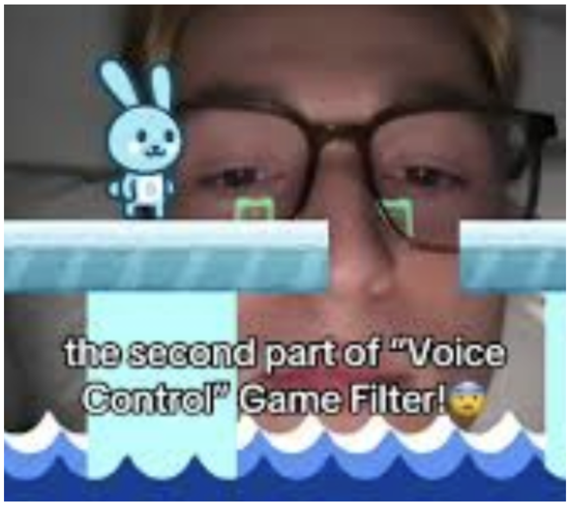
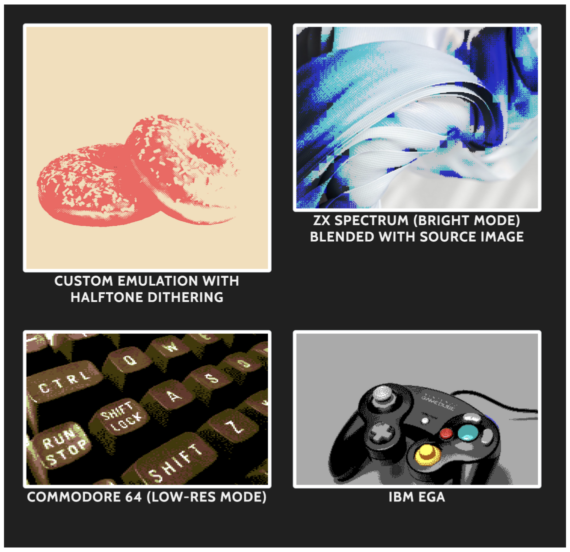


**Figure 4:** Hand detection live via webcam. The circles mark points on the palm and fingers. It's not perfect but it's pretty quick!

## 2 Challenge: Weird Interaction Design

This week, I'd like you to make a sketch that is interactive in some way, especially in a way that is unusual, delightful, expressive or surprising. A good starting point for this is to take one of the digital art sketches you made last time and make it interactive somehow - connect input to the size, shape, rotation, colour or position of some of the elements in your sketch!




CUSTOM EMULATION WITH HALFTONE DITHERING
ZX SPECTRUM (BRIGHT MODE) BLENDED WITH SOURCE IMAGE
COMMODORE 64 (LOW-RES MODE)
IBM EGA

**(a)** Voice activated games on TikTok use microphone input to control a game character.

**(b)** Retrospecs is an iOS app that uses filters to recreate old rendering tech.

**Figure 5:** Two ideas for interactive p5js sketches - voice control games and retro image filters.

If you don't have a sketch from last time, or you don't want to use what you made then, I've linked to some of my own sketches based on some of the artworks we saw last week. Feel free to take one of these, and then connect some part of the code to keyboard, mouse or other input.

https://editor.p5js.org/mtrc/sketches/EL9aqi0FB
https://editor.p5js.org/mtrc/sketches/63okVX3Wp
https://editor.p5js.org/mtrc/sketches/OtD25iqjr
https://editor.p5js.org/mtrc/sketches/j68irYnF

The last one isn't from last week - it's actually something I made in 2023 for Genuary. If you want something a bit different (and undocumented) check that out!

If you want some more ideas, Figure 5 shows two examples of interesting interactive apps that use sound or image processing. The first is from a genre of games popular on TikTok that use microphone input (usually volume, sometimes pitch) to control a game in some way. I don't recommend making a platformer in p5js (yet) but you could create something light and fun quite easily with a similar idea! I've created two quick templates that show a sort of obstacle avoidance game using microphones and hand detection. In the first, hold your hand up to your webcam (once the script has loaded) and move it around to dodge obstacles:

https://editor.p5js.org/mtrc/sketches/FCmAZEdqO

In the second, use your microphone (uh, maybe quietly, while we're together in the lab) to control the ball and move it up and down:

https://editor.p5js.org/mtrc/sketches/jrjhZER9J

Feel free to take either of these templates and extend them too.

The second image in Figure 5p shows outputs from an iOS app called Retrospecs. This app recreates rendering techniques from dozens of historical computing platforms, including all their glitches and restrictions. This includes dithering types, palette restrictions, and other rendering quirks. You could make a p5js sketch that takes a photo using the webcam when you press a button, and then applies a filter to it.

## 3 What Next

### 3.1 Share your work with me!

I'd love to see what you make, and I'd also love to be able to share people's work at future department events, or show to students in future years for inspiration! I would love it if you shared what you make - anonymously if you prefer, or with your name for full credit.

Ways you can share your work:

- Email the code, or a link to your p5.js sketch (mike.cook@kcl.ac.uk) - you can find sharing options in File → Share on the p5js editor.
- Post it to GitHub and share with me (username: gamesbyangelina or just email a link)
- If you want to submit anonymously, put your code somewhere like pastebin and send it anonymously in this form: https://forms.gle/qh8UWnqW1fNJyfxz8.

### 3.2 Sharing with other people

p5.js allows you to save and share your work with others, if you register an account and log in. You can also use the `save()` method to capture screenshots. There are lots of ways to share your work online, including the usual subreddits and Discord servers, as well as tags/keywords like #generative and #p5js. Every January there is also a generative art event called #genuary where people post digital art daily. See `https://genuary.art` for more information, and for optional daily challenges.

### 3.3 Further Reading

p5.js is a web adaptation of Processing, a Java-based offline tool that has almost identical syntax and structure. Because p5.js is so lightweight and can be run in a browser, it has become more popular over time, but you can find the original Processing and documentation online at https://processing.org/. Both Processing and p5.js are maintained by the Processing Foundation.

p5.js has a lot of libraries and add-ons written, including code for using machine-learned models for things like gesture recognition, video process, games and more. We'll be looking at some of these in future weeks.

The Coding Train is a series of programming tutorials by Dan Shiffman, an NYU Professor. Although some of his content is aimed at new programmers, there's hundreds of videos and projects that extend to very advanced topics and interesting creative projects, and most of them use p5.js. I highly recommend checking it out: https://thecodingtrain.com/.

## 3.4 Alt Controllers and Experimental Game Design

We'll be covering some very basic game development in the second half of this course, but if you're interested in more experimental design with physical devices, here are some good places to check out:

- **Now Play This**, an exhibition of experimental game design, held every April in Somerset House (just over the road from King's). https://nowplaythis.net/
- **London Pattern Club**, a group of people who are interested in crafting (knitting, crochet, etc) and technology. They often run workshops and meetups. https://patternclub.org/london/.
- **A.MAZE**, an international games festival held in Berlin every year, with a lot of experimental games as well as talks, concerts and more. https://2025.amaze-berlin.de/