A close-up of a screen

Description automatically generated

In this assignment the goal is to replicate the computer vision-based interaction seen in the image above. The assignment is a simplified version of the work done by Memo Akten in his *WebCam Piano 2.0*. More information on that can be found [here](http://www.memo.tv/portfolio/webcam-piano-2/).

**Steps to complete**

Your starting code should be the backgroundSubtraction example code provided in class. We have to turn this example from a background subtraction example to a frame differencing one.

**Step 1:** Start off by renaming the *backImg* to *prevImg* everywhere in the code so that it’s more intuitive for frame differencing. Run the code and make sure it runs just like before.

**Step 2:** In background subtraction whenever we press a key we save the current snapshot to use as a background. In frame differencing we want save the previous frame in every loop. To do that, move the code updating the *prevImg* from the keyPressed() function to the end of the draw() loop. If you’ve done things right you should have a working copy of a frame differencing example. The image on the right should show you where the movement in the frame is.

A person holding up their hand

Description automatically generated

**Step 3:** Download the Grid.js file file included on this page. This file contains the class that allows for the circles to be drawn on screen where there’s movement. Include a reference to it in the index.html file, create a global variable called grid and initialise it in the bottom of the setup() function by calling this line:

*grid = new Grid(640,480);*

and at the bottom of the draw() function call this line:

*grid.run(diffImg);*

If you’ve done things right the grid should appear on top of the colour image on the right wherever movement is detected, like in the image below. However, notes are activated even in the areas where tiny bits of movement (or even noise) triggers them. This won’t make for a very good webcam piano. We’ll fix that next.

A person taking a selfie

Description automatically generated

**Step 4:** If your camera is anything like mine you should notice that there’s a lot of activation of the “notes” of the piano even when we don’t quite intend it. Playing with the threshold can’t really fix it. Let’s address this by running the blur filter on the *currImg* at the top of the draw() function, right after we have copied the contents of the video into it. See the filter documentation [here](https://p5js.org/reference/#/p5.Image/filter). Use the BLUR filter with 3 iterations (i.e. pass 3 as a 2nd parameter in the filter function). If you’ve done things right you should see something like the image below where the noise is removed due to the blur, but the sketch will be running very slowly. We’ll fix that next.



**Step 5:** Right before using the blur filter on the *currImg,* use the resize command to scale it down to a quarter of the size it was. [Documentation here](https://p5js.org/reference/#/p5.Image/resize). Make sure you do the same for the *diffImg* a couple of lines later or things will break! The sketch should now be running quite fast (>40fps) and your sketch should look like the animation at the top of this page.

**Step 6:**Extend the sketch. This is your last graded assignment and is your chance to make use of techniques and knowledge learnt throughout the course. Select two ideas for further development aiming to write clean and modular code. Include comments about how and why you extended the sketch in the way you did. Get inspiration from Memo Atken’s webcam piano [here](http://www.memo.tv/portfolio/webcam-piano-2/).

**Ideas for further development:**

* Customize the graphics. In Grid.js you could customize the base grid of graphics thinking about color, opacity or shape. Can you make use of *noteState* to drive different effects that change over time after the note has been activated?
* Trigger secondary graphics effects or animations when an active note is drawn. Can you include rules that only trigger the effects sometimes, perhaps making use of noise or randomness?
* Implement the core p5.js sound library to play sounds depending on which “note” in the grid is activated. You should read and use the p5.js documentation about [starting the audio context on user interaction](https://p5js.org/reference/#/p5/userStartAudio) to make sure your audio works across all browsers.
* Implement a custom “Note” class that is used in Grid.js. Instead of an array of values for *noteSize*, *notePos* and *noteState* you would have an array of notes. Think about what parameters you would need for the Note constructor method and what other methods the Note class would need, both the methods needed to adapt the existing functionality from the Grid.js code and any custom methods you would like to add.

**Marking rubric**

Step 1 - [1 point]: Renaming of *backImg* to *prevImg* has taken place.

Step 2 - [2 points]: Frame differencing implemented by moving *prevImg* around.

Step 3 - [2 points]: Learner has included Grid.js correctly and grid activates with movement.

Step 4 - [2 points]: Learner has included blur in order to reduce the amount of noise that activates the grid.

Step 5 - [2 points]: Learner has scaled down images processed (*currImg*, *diffImg*) so that the sketch runs fast after blurring has slowed it down.

Step 6 - [3 points]: How much has the learner extended the sketch? Learner has included comments about the extension and shows understanding of techniques learnt throughout the course.