Colour Theremin

Premise

Colour is something we can’t actually touch, so why do we search for it by touching colour samples? This project’s medium-fidelity prototype is an attempt to re-imagine colour searching as a paint-chip device. It takes location input instead based on distance and outputs a relevant colour. It is aimed at professional-c level creatives and those who aspire to be. Specifically, it was envisioned for anyone involved in paint colour selection such as interior designers, painters, or other artists.

Inspirations and Intentions

This project clearly owes some inspiration to the theremin[1], except that it produces colours rather than sounds. However, that was not its original inspiration. The original inspiration was Colorigins on the Sifteo Cubes platform[2] by Brad Tober[3]. It used small colour-screened cubes that could be physically tilted into each other to mix each cube’s colours virtually. However, I wanted to build a colour selection system instead of a mixing one.

Originally, this was envisioned using an accelerometer to measure motion that the user would use to move a paint chip through a colour space. This was intended to suggest to the user that they themselves were inside the space when making a selection. This was not as feasible as originally thought for the micro-controller being used and the time available.

After some iteration of ideas, I kept the virtual paint chip idea but replaced the motion through space with the motion used by tint adding machine for paints. These are the same ones you still see in paint stores where each measured amount of dye is pulled into a plunger first for accuracy and then added to the paint itself before the next is added, according to a colour’s recipe. This could be done and undone instead by raising and lowering your hand on to a virtual paint chip instead, and through this the idea for Colour Theremin was born.

This new method is intended to promote more colour exploration by the end user rather than just searching. They can quickly see nearby colours along a colour axis with only slight hand adjustments rather than manual searching through real paint chips. It is also hoped that the user may get a muscle memory for a range of colour they use regularly through the motions they use when selecting colours this way.

How to Use

This prototype has no on-off switch and begins working from power-on.

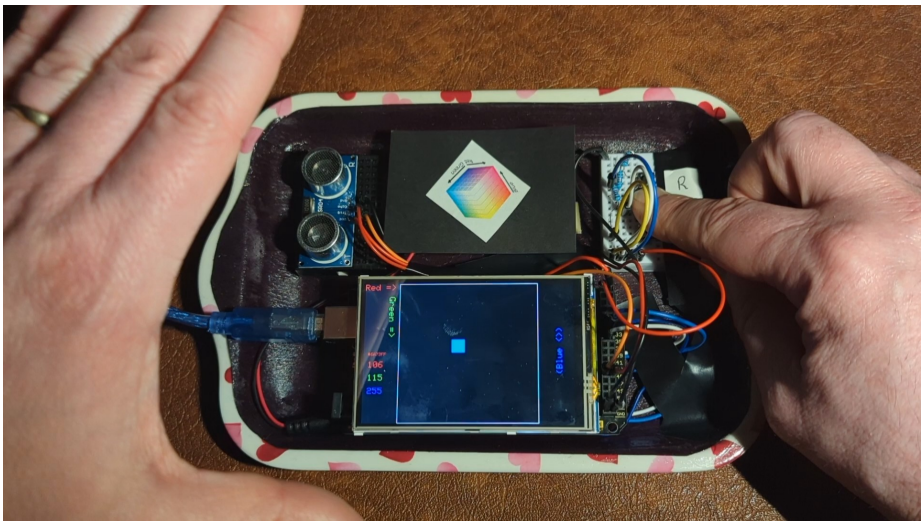
It has 2 modes: colour selection and paint chip mode when locked.

If no button is pressed, it will be in paint chip mode where it will show the current colour selection and its associated colour values. The handles on the side are intended to allow the user to put the paint chip screen against other surfaces for comparison when selecting colours.

Picture 1: Paint chip mode.

From power-on, the colour-state is full white #FFFFFF.

If the user, presses the R, G, or B buttons at right, colour selection mode is enabled and the user can begin to adjust that colour level by raising or lowering their hand above the ultrasonic distance sensor at the top.



Picture 2: Colour selection mode: hand at left adjusts level while the hand at right selects the RGB channel.

While the user is doing this, they get two channels of feedback on the display screen. First is the numeric feedback that gives specific values for the visual colour space. Second is the 2 dimensional motion highlighting of where the selected colour is in the RGB colour cube. Red and green are shown on the x and y axis respectively. But due to hardware limitations, the blue z axis had to be represented conceptually with up and down arrows relative to the red and green position.

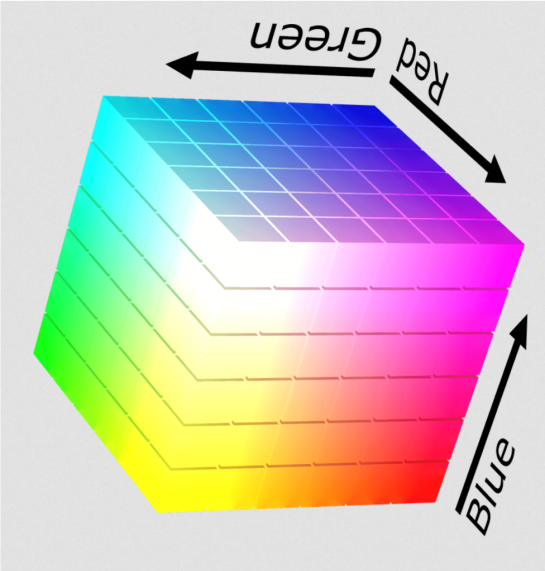


Figure 1: RGB colour space cube, taken from Wikimedia[4]. Here it is rotated such that the red, green, and blue axis better represent the Colour Theremin's space visualization.

Building the Colour Theremin

Components

- Arduino Mega2560 or Mega2560 clone
- TFT 3.5” 480x320 screen (touch screen not necessary but used here)
- 4 pin HC-SR04 ultrasonic depth sensor
- 3 momentary push switches
- 3 @ 10kOhm resistor
- wiring and breadboards as desired
- 9V battery adapter for portable power
- mounting board or tray as desired

Libraries

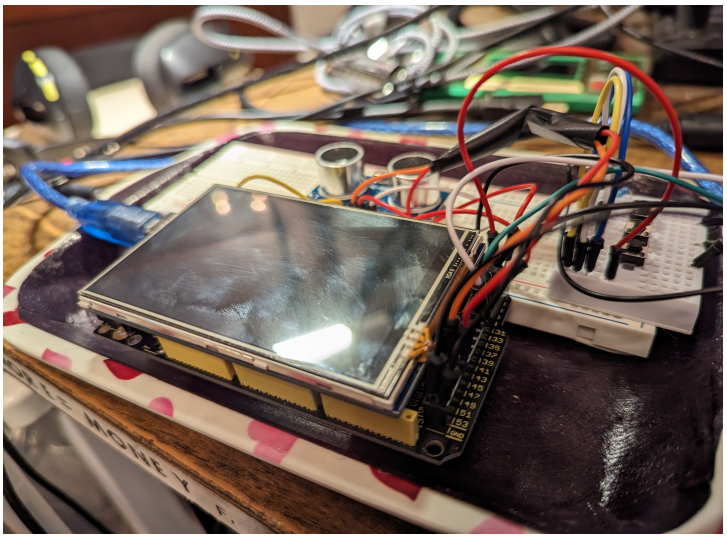
This requires:

- Adafruit GFX library for graphics
- MCUFRIEND_kbv library for the screen
- SPI library to talk to the distance sensor

These are all available in the standard Arduino IDE Library Manager

Assembly

Assembly is fairly straight forward.
1) The TFT connects to the Mega2560 board in only one manner, so do that first.



Picture 3: You will now see why the 2560 is necessary from this build picture. The TFT shield takes almost all the pins except the pins 22-53 and the power and grounds on the right hand side of the board.

2) Wire the buttons and ultrasonic sensor as follows:

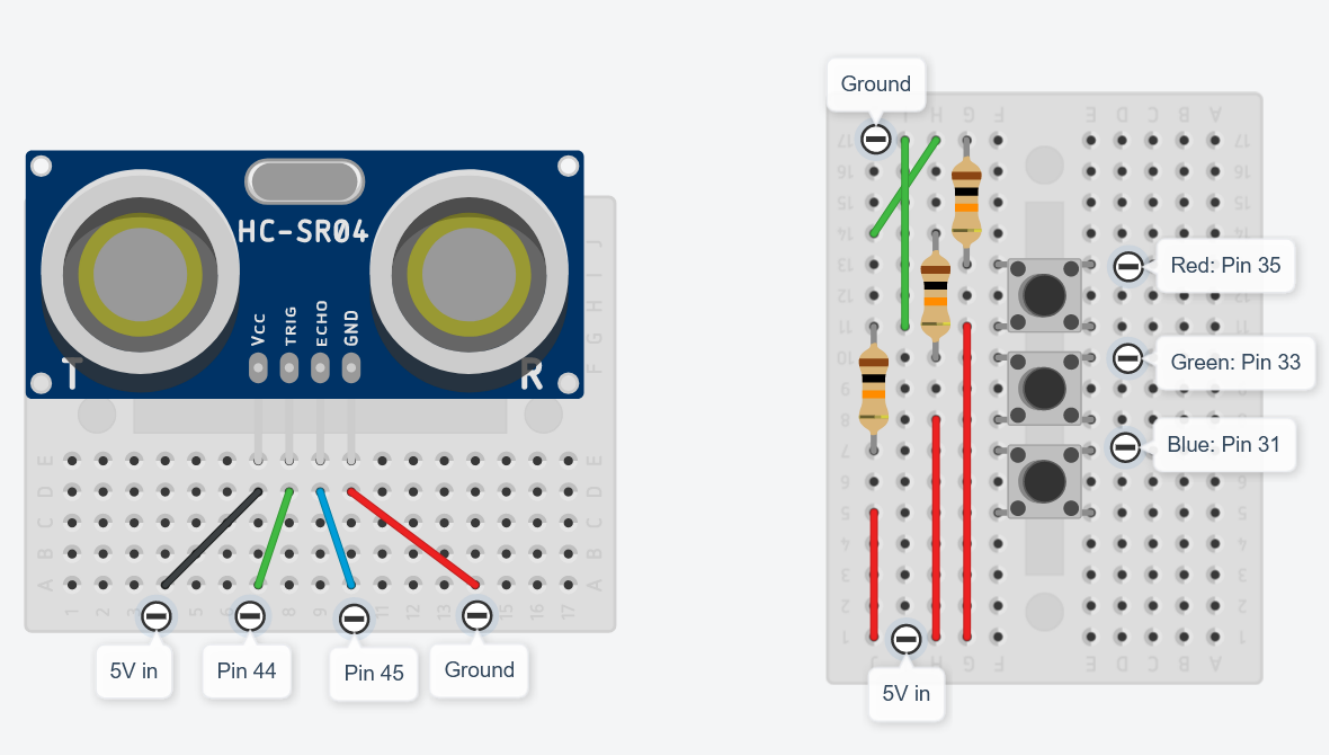


Figure 2: Wiring diagram for the sensors. As the TFT shield basically snaps in and covers the Mega2560, it is not included here.

3) Connect your board via USB and upload the Arduino code from source here:
https://github.com/mdaikman/ColourTheremin
You can adjust this code if desired and the main variables worth adjusting are:
- refresh_rate to set how sensitive the sampling time is.
- dist_floor to adjust the minimum distance from the sensor. Too close is not good.
- dist_scale to adjust the amount of range considered from the sensor. Higher means more condensed.

4) Mount and power as desired. I chose to have the RGB buttons as easily accessible to the right side as possible for a prototype. I found it to be more beneficial to have the sensor off to the left side though, so having your hand over the sensor does not interfere with seeing the screen. It also gives a bit more of that classic theremin playing pose.

Limitations

At this stage of development, there are several limitations that were either accepted or learned along the way that are worth noting.

As mentioned, originally this was intended to be moved fully through space rather than hand-waved at for motion selection. Although accelerometers exist for the Arduino environment, it became clear that they were great at detecting the existence of motion and its axis but had a limitation in detecting the direction that could not be solved in the timeline of this project.

The TFT itself has three major limitations other than its enormous footprint on the Arduino board: draw speed, colour depth, and colour system.

First issue is the draw speed. The TFT display can not move a large number of pixels on to the screen at once. You can see this by how slowly it clears to black at the startup. As such, it means that the colour square you see changing during colour selection can only be so big, as it has to erase and redraw it without too much delay or flicker. On lock screen though, the whole frame can be filled since there is no rapid redrawing.

The second is the lack of colour depth. Full RGB colour has 2^{24} potential colours, but Arduino TFT displays have 2^{16} at best and this only uses 2^{15} for simplicity, which is 32 colour shades for each RGB channel. However, the human eye is complex and a discussion of how limiting this actually is well-beyond the scope of this document. It is worth acknowledging it explicitly though and hope that the 32,768 this prototype does show are sufficient for demonstration.

Additionally, the code has a conceit in it to make the range of colour appear smoother numerically. Each colour channel can show a value between 0 (full off) to 31 (2^5 bits for full on). You can't simply multiply this by 8 to get the same respective value in 0 – 255 for RGB. At max value 31 * 8 = 248, so there is an algorithm to smooth in-between values of those extra 7 bits through interpolation as it approaches max value. This is done purely to give the user the feeling of 0-255 RGB colour values at this prototype stage.

Finally, I will state the obvious. As a colour chip system intended for paint design, this prototype should not be using an additive light colour space like RGB to get reflected light colours. Something additive like CMYK would be far more appropriate, and with the advent of e-ink that should be possible. However, cheap, fast, and colour accurate e-ink is not yet readily available, so that would need to be added in a later iteration.

References

[1]*Theremin* (2023) *Wikipedia*. Wikimedia Foundation. Available at: https://en.wikipedia.org/wiki/Theremin (Accessed: January 31, 2023).

[2] *Colorigins* (no date) *Colorigins by Brad Tober | International Design Awards™ Winners*. Available at: https://www.idesignawards.com/winners/zoom2.php?eid=9-8728-15 (Accessed: January 31, 2023).

[3] Tober, B. (no date) *Brad Tober*. Available at: https://bradtober.notion.site/ (Accessed: January 31, 2023).

[4] *File:RGB color solid cube.png - Wikimedia Commons* (no date). Wikimedia. Available at: https://commons.wikimedia.org/wiki/File:RGB_color_solid_cube.png (Accessed: January 31, 2023).