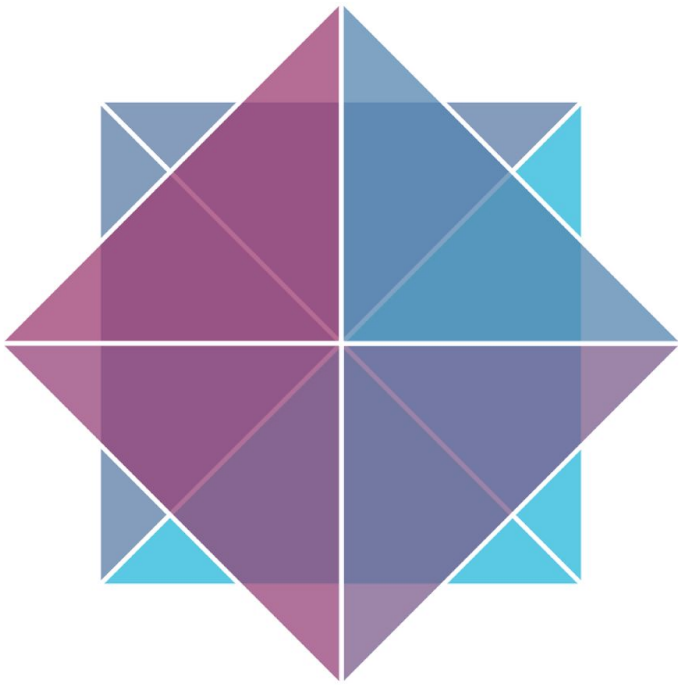
DESIGN

In designing the software, we aimed to create a game that included various levels of difficulty and incentivized users with a creative design rather than a numerical score. The game offers four levels: practice, easy, medium, and difficult. Levels vary in difficulty by the number of vertices in each shape (i.e. the number of limbs users must use to make it) and by the diversity of shapes prompted (all triangles vs. triangles, pentagons, and hearts).

Each level in Geometris consists of 2 game states. In the first game state, users are prompted to recreate 8 shapes (e.g., triangle, pentagon, heart), one at a time, by simultaneously activating particular touchpads on the mat. Each shape appears near the top of the projection area and descends toward a red line at the bottom of the projection area, adjacent to the mat (shown in the figure below). If users fail to reproduce the shape before it reaches the red line, that shape will not appear on the final geometric design. Each shape contains a circle, intended as a frame of reference to indicate the position of the mat's central point relative to the sides and corners of the projected shape.

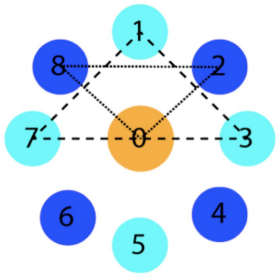
As users complete a shape, the game emits a celebratory sound and a new shape appears from the top of projection. Completed shapes are moved to the lower right hand corner of the screen (shown in the figure), where all successfully reproduced shapes are layered on top of each other in a composite design. Additionally, we vary the shape's color smoothly from blue at the top to red as it approaches the red line. As users complete the shape, the color at the time of completion is recorded and reflected in the composite design.

After the 8th shape, the game moves into its second state where the composite design is enlarged and presented in the middle of the screen, accompanied by a congratulatory music track. A menu is also displayed in the upper left corner, and players can navigate to the home screen by pressing the touchpad at 12 o'clock (see figure below).



DEVELOPMENT PROCESS

We labeled each pressure-sensitive touchpad with an index (1-8 for 8 touchpads) - #1 in the 12 o'clock position proceeding clockwise to #8. We also labeled the origin/central point of reference as index-0 in a 9-element binary array to store which vertices are being activated and also to decide how shape are made. We used the Firmata library to pass the digital inputs to Processing.



The logic to draw shapes defines (x, y) coordinates for each vertex and uses the PShape class to draw the shape given these coordinates. We also wrote supporting functions that allow the program to take in an array of 1s and 0s and return another array of (x, y) coordinate pairs through trigonometric manipulations. Each shape is then constructed as a custom shape object that contains an array of 9 elements (either 1 or 0). For example, the array {0,1,0,1,0,0,0,1,0} would correspond to a right triangle with vertices at positions 1, 3, and 7. Similarly, a triangle that involves the origin (index-0) and vertices 2 and 8 contains the array {1,0,1,0,0,0,0,0,1} (see figure alongside). The custom shape object also stores the color that changes linearly with the y-coordinate of the shape's origin.

By definition, users will have "successfully created" a shape when the required vertices - and only the required vertices - are switched on through their corresponding touchpad. Logically, Processing evaluates if the 9-element array that it is reading is identical to the 9-element array stored for the current shape. The moment this condition is satisfied, the shape object, which also includes its color at the time, is appended to an ArrayList of the custom shape objects. This new array is used to present a

record of shapes completed by the user thus far and also used to present the final composite shape at the end of the level.

Digital inputs from the touchpads are also used to allow navigation across screens. The home screen menu maps pads 8, 1, 2, and 5 to the easy, medium, difficult, and practice levels, respectively. One shortcoming of our current design is that the touchpads could not be used to navigate away from screens when they are already being used to create shapes.

Since a bulk of the software was written while the hardware was being fabricated, we wrote additional mouse functions to debug the code in the absence of the mat and touchpads. Mouse clicks mimic successful shape creation. The practice screen is also used to debug the hardware and software.

The git repo https://github.com/ganesh-iyer/geometris-processing also includes code to the musical instrument that was created using fragments from Geometris.

TIMELINE OF WORK AND AVAILABILITY

Despite an ongoing semester, the team has the bandwidth to log in the required 100 hours of work in from as short a time period as 1.5 months to 4 months at any stage between February and May. The research-driven changes would depend on availability of test subjects and we're tentatively looking at no later than April to obtain test results. We are prepared to set up the groundwork on the platform in parallel or before we implement the changes suggested by research.