

# flipV()

Flips the geometry’s texture v-coordinates.

In order for `texture()` to work, the geometry needs a way to map the points on its surface to the pixels in a rectangular image that's used as a texture. The geometry's vertex at coordinates `(x, y, z)` maps to the texture image's pixel at coordinates `(u, v)`.

The `myGeometry.uvs` array stores the `(u, v)` coordinates for each vertex in the order it was added to the geometry. Calling `myGeometry.flipV()` flips a geometry's v-coordinates so that the texture appears mirrored vertically.

For example, a plane's four vertices are added clockwise starting from the top-left corner. Here's how calling `myGeometry.flipV()` would change a plane's texture coordinates:

```
// Print the original texture coordinates.  
// Output: [0, 0, 1, 0, 0, 1, 1, 1]  
console.log(myGeometry.uvs);  
  
// Flip the v-coordinates.  
myGeometry.flipV();  
  
// Print the flipped texture coordinates.  
// Output: [0, 1, 1, 1, 0, 0, 1, 0]  
console.log(myGeometry.uvs);  
  
// Notice the swaps:  
// Left vertices: [0, 0] <--> [1, 0]  
// Right vertices: [1, 0] <--> [1, 1]
```

## Examples

▶

■

```
let img;  
  
function preload() {  
  img = loadImage('/assets/laDefense.jpg');  
}  
  
function setup() {  
  createCanvas(100, 100, WEBGL);  
  
  background(200);  
  
  // Create p5.Geometry objects.  
  let geom1 = buildGeometry(createShape);  
  let geom2 = buildGeometry(createShape);  
  
  // Flip geom2's V texture coordinates.  
  geom2.flipV();  
  
  // Left (original).  
  push();  
  translate(-25, 0, 0);  
  texture(img);  
  noStroke();  
  model(geom1);  
  pop();  
  
  // Right (flipped).  
  push();  
  translate(25, 0, 0);  
  texture(img);  
  noStroke();  
  model(geom2);  
  pop();  
  
  describe(
```

This page is generated from the comments in [src/webgl/p5.Geometry.js](#) . Please feel free to edit it and submit a pull request!

## Related References

### calculateBoundingBox

Calculates the position and size of the smallest box that contains the geometry.

### clearColors

Removes the geometry’s internal colors.

### computeFaces

Computes the geometry's faces using its vertices.

### computeNormals

Calculates the normal vector for each vertex on the geometry.

