Forming | Façade Tiling

```
//For each face set an origin for its cutting plan at either 1/4, 1/2 or 3/4 of its edges
//Check for both edges, their length and set cutting plane orientation
//accordingly (random, horizontal, vertical or no cut)
//Use interpolation of coordinates (UV) to calculate the normal vectors for the cutting planes

//Input: Individual Mesh Faces of the Facade
//Output: Faces with stored cut location and normals for cutting plane

vector min_bb, max_bb, pos;
float riter, rx, ry, rz;
int x_or_z;

getbbox(0,min_bb,max_bb); //Getting the extends of the face
v@min = min_bb;
v@max = max_bb;

riter = detail(-1,'iteration');

//Randomising the position of the cut
rx = onoise(@P+chi('seedx')+riter+@Frame);
ry = onoise(@P+chi('seedy')+riter+@Frame);
rz = onoise(@P+chi('seedz')+riter+@Frame);

//Clipping the cut location to 1/4, 1/2 or 3/4 of the edge length
rx = ceil(rx*3)/4;
ry = ceil(ry*3)/4;
rz = ceil(rz*3)/4;

//Mapping the relative position to the coordinate system
pos.x = fit(rx,-1,1,min_bb.x, max_bb.x);
pos.y = fit(ry,-1,1,min_bb.y, max_bb.y);
pos.z = fit(rx,-1,1,min_bb.z, max_bb.z);

//Get Face Edge Lenghts

vector pt_a = primuv( 0, "P", @primnum, {0, 0});
vector pt_b = primuv( 0, "P", @primnum, {1, 0});
vector pt_c = primuv( 0, "P", @primnum, {1, 1});

vector a = pt_b - pt_a;
vector b = pt_c - pt_b;

float side_a = length(a);
float side_b = length(b);


float w_x = chf("scale_x");
float w_z = chf("scale_z");

//Set Cutting Plane Orientation according to edge lengths (to prevent too small faces)
if (side_a > w_x && side_b > w_z){
    x_or_z = floor(rand(@primnum+riter+chi('seed_x_or_z'))*2); //Random Orientation
}

 if (side_a > w_x && side_b < w_z){
     x_or_z = 0; //Vertical Orientation
 }

if (side_a < w_x && side_b > w_z){
    x_or_z = 1; //Horizontal Orientation
}

if (side_a < w_x && side_b < w_z){
    x_or_z = 2; //No more cuts
    pos.y = 5000000;
}
//Storing Cut Location and Orientation
setdetailattrib(0,'clipcenter',pos);
setdetailattrib(0,'x_or_z',x_or_z);

//Calculating the normal vectors for the cutting planes
vector mid_pt_a = primuv( 0, "P", @primnum, {0.5, 0});
vector mid_pt_b = primuv( 0, "P", @primnum, {1, 0.5});
vector mid_pt_c = primuv( 0, "P", @primnum, {0.5, 1});
vector mid_pt_d = primuv( 0, "P", @primnum, {0, 0.5});

vector cut_a = mid_pt_c – mid_pt_a;
vector cut_b = mid_pt_d – mid_pt_b;
vector n = prim_normal(0, @primnum, {0.5,0.5});

//Storing the normal vectors for later use
v@plane_a = rint(normalize(cross(cut_a,n))*100)/100;
v@plane_b = rint(normalize(cross(cut_b,n))*100)/100
```