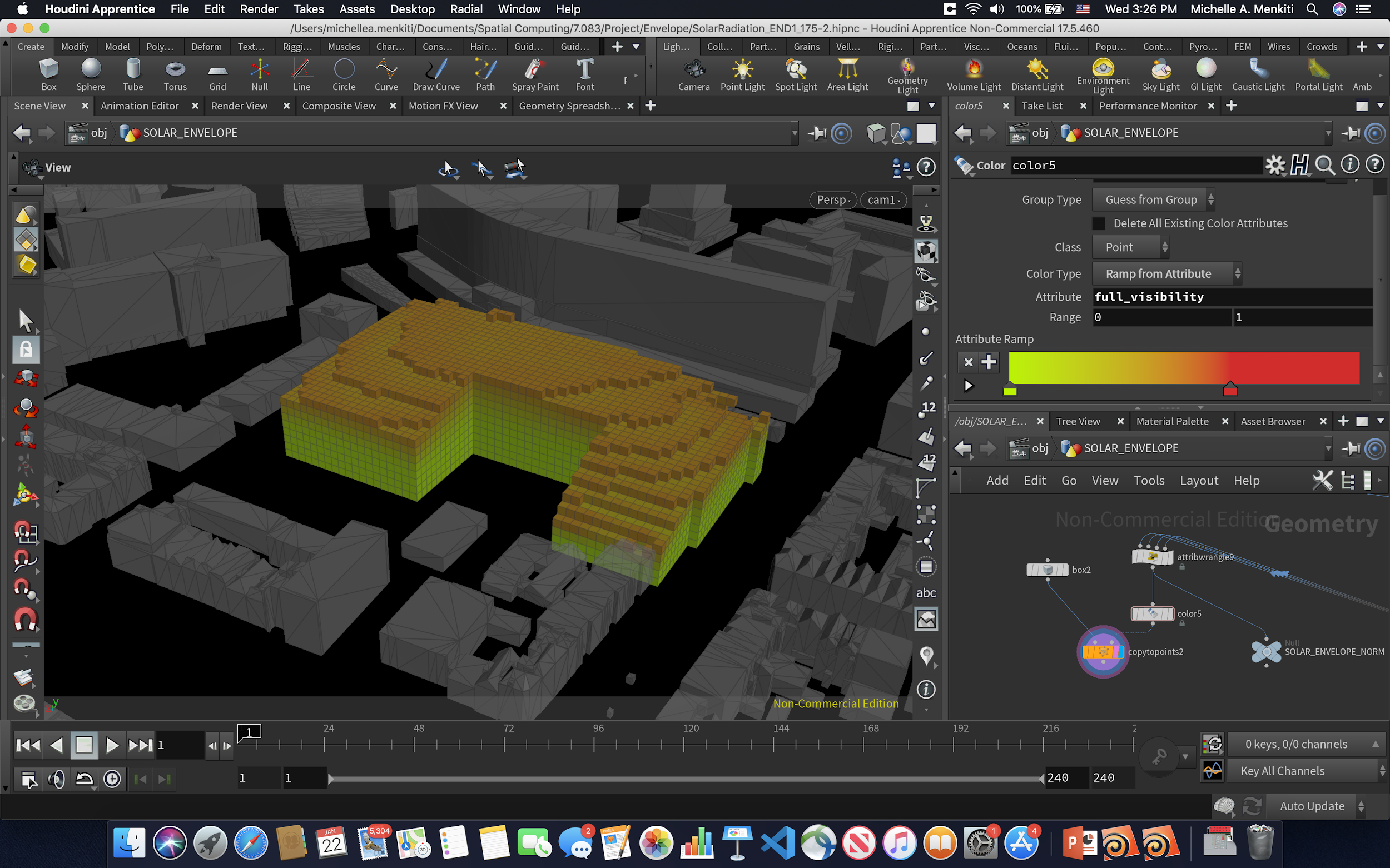
MASSING ENVELOPE

//Date: 24-01-2019

//Authors: Jolt Wiersma, Michelle A. Menkiti, Arthur Masure

//Adapted from work by: Shervin Azadi, Dr.ir. Pirouz Nourian, Hans Hoogenboom

//Purpose: Merging voxel values from Obscurity and Visibility analysis to create one aggregated volume, and solar envelope



THE DESIRED MASSING ENVELOPE BASED ON DESIGN PRINCIPLES / STRATEGY

**WRANGLER 1**

**Normalizing all incoming values**

//INPUT 0 = OBSCURITY OF SKY

//INPUT 1 = OBSCURITY OF SUN

//INPUT 2 = VISIBILITY OF SKY

//INPUT 3 = VISIBILITY OF SUN

//RUN OVER DETAIL

int num\_points = npoints(0);

float max\_vsun = point(2,"visibility\_sun",0);

float max\_vsky = point(3,"visibility\_sky",0);

float max\_osky = point(0,"obscurity\_sky",0);

float max\_osun = point(1,"obscurity\_sun",0);

float min\_vsun = point(2,"visibility\_sun",0);

float min\_vsky = point(3,"visibility\_sky",0);

float min\_osky = point(0,"obscurity\_sky",0);

float min\_osun = point(1,"obscurity\_sun",0);

for (int point = 0; point < num\_points; point++)

{

float vsun\_value = point(2,"visibility\_sun",point);

float vsky\_value = point(3,"visibility\_sky",point);

float osky\_value = point(0,"obscurity\_sky",point);

float osun\_value = point(1,"obscurity\_sun",point);

if (vsun\_value > max\_vsun) max\_vsun = vsun\_value;

if (vsky\_value > max\_vsky) max\_vsky = vsky\_value;

if (osky\_value > max\_osky) max\_osky = osky\_value;

if (osun\_value > max\_osun) max\_osun = osun\_value;

if (vsun\_value < min\_vsun) min\_vsun = vsun\_value;

if (vsky\_value < min\_vsky) min\_vsky = vsky\_value;

if (osky\_value < min\_osky) min\_osky = osky\_value;

if (osun\_value < min\_osun) min\_osun = osun\_value;

}

for (int point = 0; point < num\_points; point++)

{

float vsun\_value = point(2,"visibility\_sun",point);

float vsky\_value = point(3,"visibility\_sky",point);

float osky\_value = point(0,"obscurity\_sky",point);

float osun\_value = point(1,"obscurity\_sun",point);

vsun\_value = fit(vsun\_value, min\_vsun, max\_vsun, 0.001, 0.999);

vsky\_value = fit(vsky\_value, min\_vsky, max\_vsky, 0.001, 0.999);

osky\_value = fit(osky\_value, min\_osky, max\_osky, 0.001, 0.999);

osun\_value = fit(osun\_value, min\_osun, max\_osun, 0.001, 0.999);

//setting normalized values

setpointattrib(2,"visibility\_sun",point,vsun\_value,"set");

setpointattrib(3,"visibility\_sky",point,vsky\_value,"set");

setpointattrib(0,"obscurity\_sky",point,osky\_value,"set");

setpointattrib(1,"obscurity\_sun",point,osun\_value,"set");

}

**WRANGLER 2**

**Normalizing all incoming values**

//INPUT 0 = WRANGLER 1

//RUN OVER DETAIL

//getting inputs from sliders

float O\_sky = ch("obscurity\_sky");

float O\_sun = ch("obscurity\_sun");

float V\_sun = ch("visibility\_sun");

float V\_sky = ch("visibility\_sky");

//going through each input, and changing the value

//multiplying the normal values by the attribute values

int points = npoints(0);

for (int point = 0; point < points; point++)

{

//get attribute, divide it, replace it

//Obscurity\_sky

float value1 = getattrib(0,"point","obscurity\_sky",point,1);

value1 = pow(value1, O\_sky);

//setpointattrib(0,"obscurity\_sky",point,0,value1,"set");

//Obscurity\_sun

float value2 = getattrib(0,"point","obscurity\_sun",point,1);

value2 = pow(value2, O\_sun);

//setpointattrib(0,"obscurity\_sun",point,0,value2,"set");

//Visibility\_sun

float value3 = getattrib(0,"point","visibility\_sun",point,1);

value3 = pow(value3, V\_sun);

//setpointattrib(0,"visibility\_sun",point,0,value3,"set");

//Visibility\_sky

float value4 = getattrib(0,"point","visibility\_sky",point,1);

value4 = pow(value4, V\_sky);

//setpointattrib(0,"visibility\_sky",point,0,value4, "set");

//total

float total = value1 \* value2 \* value3 \* value4;

//setting that value

setpointattrib(0,"full\_visibility",point,total,"set");

}

**WRANGLER 3**

**Normalizing all incoming aggregate values**

//INPUT 0 = WRANGLER 2

//RUN OVER DETAIL

int num\_points = npoints(0);

//setting temp max and min

float max\_total = point(0,"full\_visibility",0);

float min\_total = point(0,"full\_visibility",0);

//iterating through all points

for (int point = 0; point < num\_points; point++)

{

float total\_value = point(0,"full\_visibility",point);

//adjusting values if new min and max are found

if (total\_value > max\_total) max\_total = total\_value;

if (total\_value < min\_total) min\_total = total\_value;

}

//iterating points

for (int point = 0; point < num\_points; point++)

{

float total\_value = point(0,"full\_visibility",point);

total\_value = fit(total\_value, min\_total, max\_total, 0.001, 0.999);

//setting new attribute

setpointattrib(0,"full\_visibility",point,total\_value,"set");

}

**WRANGLER 4**

**Voxel removal**

//INPUT 0 = NORMALIZED VOXELS

//RUN OVER POINTS

//removing points if they are over a certain threshold

if (@full\_visibility > chf("threshold"))

{

removepoint(0, @ptnum);

}