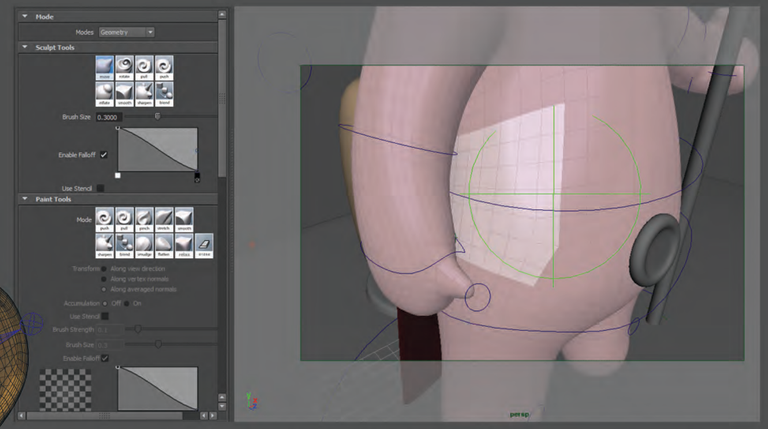
Cornelius the brave

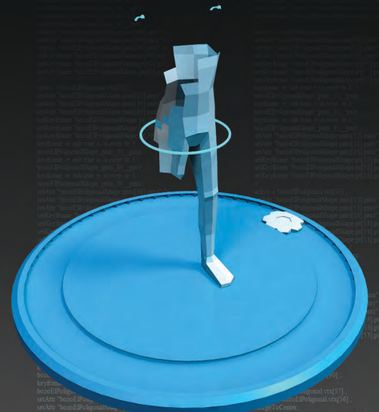
In some of the shots, especially the ones that slow down, I had to hand animate the cape. Using a tool I wrote to help me animate fluidly I was able to sculpt the seemingly correct cloth shapes where Maya would fail. With slower cloth Ncloth will unexpectedly twitch making the effect useless.

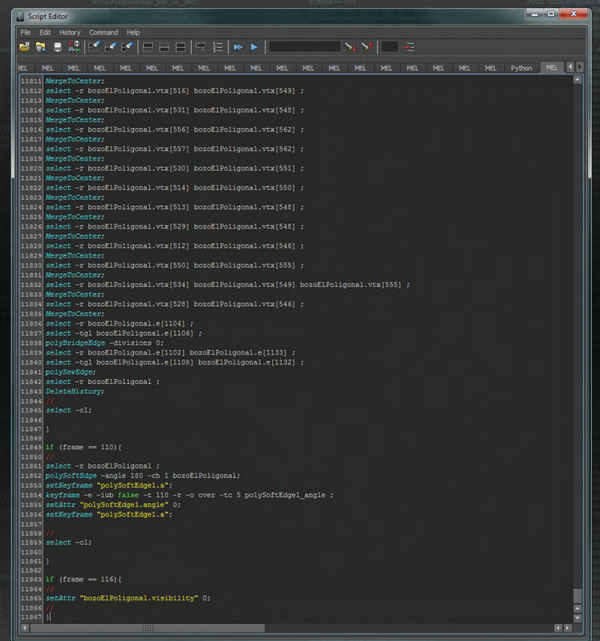


In order to keep the feel of a fluffy character a lot of post animation tweaking was used using Lbrush. After my evaluation period I just wrote a tool that does the same thing to finish the other shots.

Cornelius was an interesting rig, being made of fluff I had to result in creating “ribbons.” These ribbons are what Riggers use to create cartoony rigs. They help in creating physical arcs in the characters arms, legs and torso.

```
//
string $Vnum = 277;
string $FrmNbr = 2;
setAttr "bodyBuild.pnts[277].pntx" -0.011;
setAttr "bodyBuild.pnts[277].pnty" -2.609;
setAttr "bodyBuild.pnts[277].pntz" 0.0363;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntx") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pnty") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntz") ;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntx") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pnty") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntz") 0;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
//
string $Vnum = 264;
string $FrmNbr = 2;
setAttr "bodyBuild.pnts[264].pntx" 0.117;
setAttr "bodyBuild.pnts[264].pnty" -2.605;
setAttr "bodyBuild.pnts[264].pntz" 0.0307;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntx") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pnty") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntz") ;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntx") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pnty") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntz") 0;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
//
string $Vnum = 263;
string $FrmNbr = 2;
setAttr "bodyBuild.pnts[263].pntx" 0.0655;
setAttr "bodyBuild.pnts[263].pnty" -2.602;
setAttr "bodyBuild.pnts[263].pntz" 0.133;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntx") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pnty") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntz") ;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntx") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pnty") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntz") 0;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
//
string $Vnum = 262;
string $FrmNbr = 1;
setAttr "bodyBuild.pnts[262].pntx" 0.56;
setAttr "bodyBuild.pnts[262].pnty" -0.408;
setAttr "bodyBuild.pnts[262].pntz" -0.00733;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntx") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pnty") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntz") ;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntx") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pnty") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntz") 0;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
//
string $Vnum = 250;
string $FrmNbr = 1;
setAttr "bodyBuild.pnts[250].pntx" 0.547;
setAttr "bodyBuild.pnts[250].pnty" -0.405;
setAttr "bodyBuild.pnts[250].pntz" -0.0227;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntx") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pnty") ;
keyframe -e -iub true -r -o over -tc $FrmNbr ("bodyBuild_pnts_" + $Vnum + "__pntz") ;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntx") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pnty") 0;
setAttr ("bodyBuild.pnts[" + $Vnum + "].pntz") 0;
setKeyframe ("bodyBuild.pt[" + $Vnum + "].px");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].py");
setKeyframe ("bodyBuild.pt[" + $Vnum + "].pz");
//
string $Vnum = 249;
```



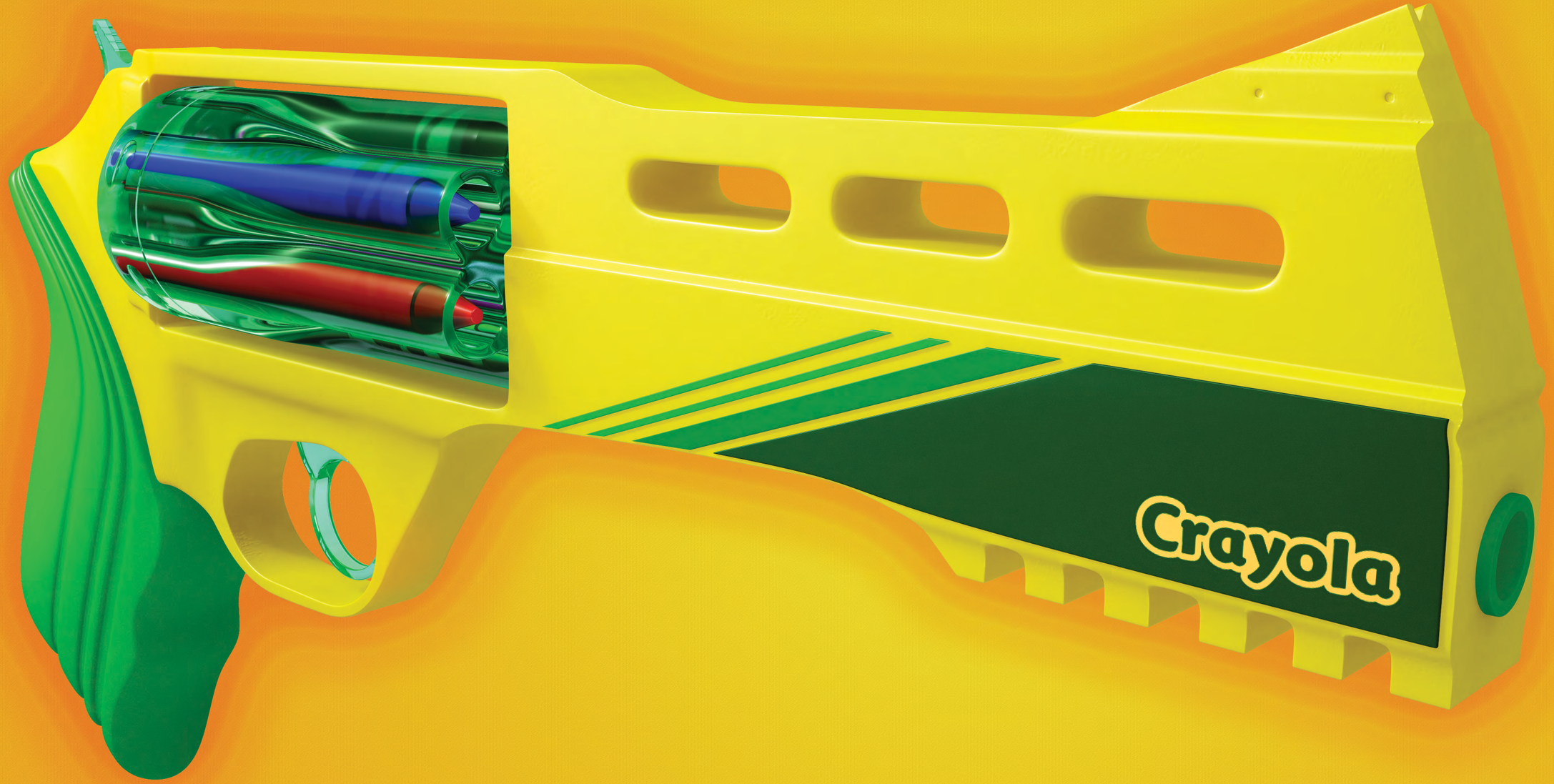


One of my most challenging projects was one centered around time management. My demo reel opener consists of a generic human figure being built one polygon at a time. In other words each individual vertex was created animated and connected to the existing vertices.

A majority of the script was written using a script where I only had to specify the amount of frames for that vertex to be animated in and what vertex it was. This sped up the writing process to help reach my deadline.

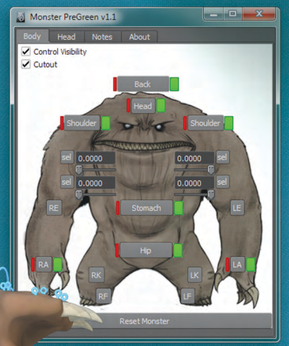
There is no illusion here this script was written containing all of the elements needed to model this character. The final written code was just under 12,000 lines of code or 288 pages (single spaced 12pnt font)

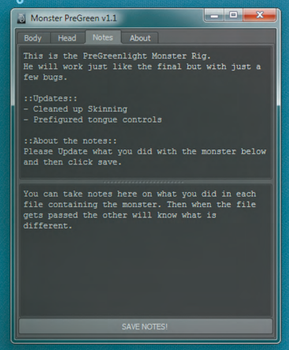
To animate the building process I had to tell Maya to key frames ahead of where the current time frame was while maintaining the main frame for rendering. This was because in previous frames what was being animated never existed and was being created.


Crayola


3/05 2:43 PM
Photo
Camera
SONY
POWER
HOLD
HOME
SELECT
START

***Complete with dual jaws and a bad attitude!**





During the animation phase Animators may get distracted by a large number of controllers. At times some of the controllers become hidden buy the character itself or by other geometry and objects.

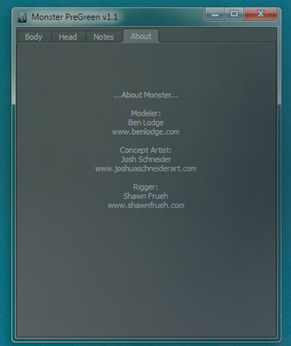
Sometimes the high res mesh can be heavy to calculate and slow down your system. Here the animators have the ability to toggle on and off a cutout version of the high res character to ensure a real time experience.

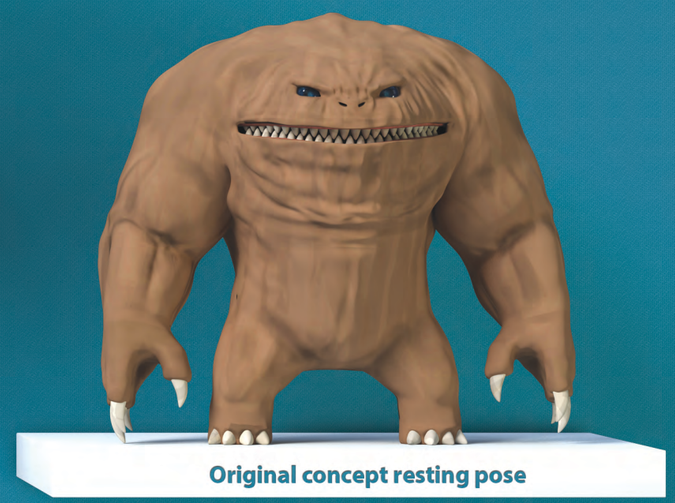
Not only does this GUI, Graphical User Interface, give the animator the ability to select visible controls, but also allows him/her to take notes and save them for the next animator to see when he opens the GUI.

Information about what the current version of the rig is displayed as well as updates and fixes.

For some controls it is better have them hidden at all times like facial controls, where the amount can get insane! This is where a GUI is essential.

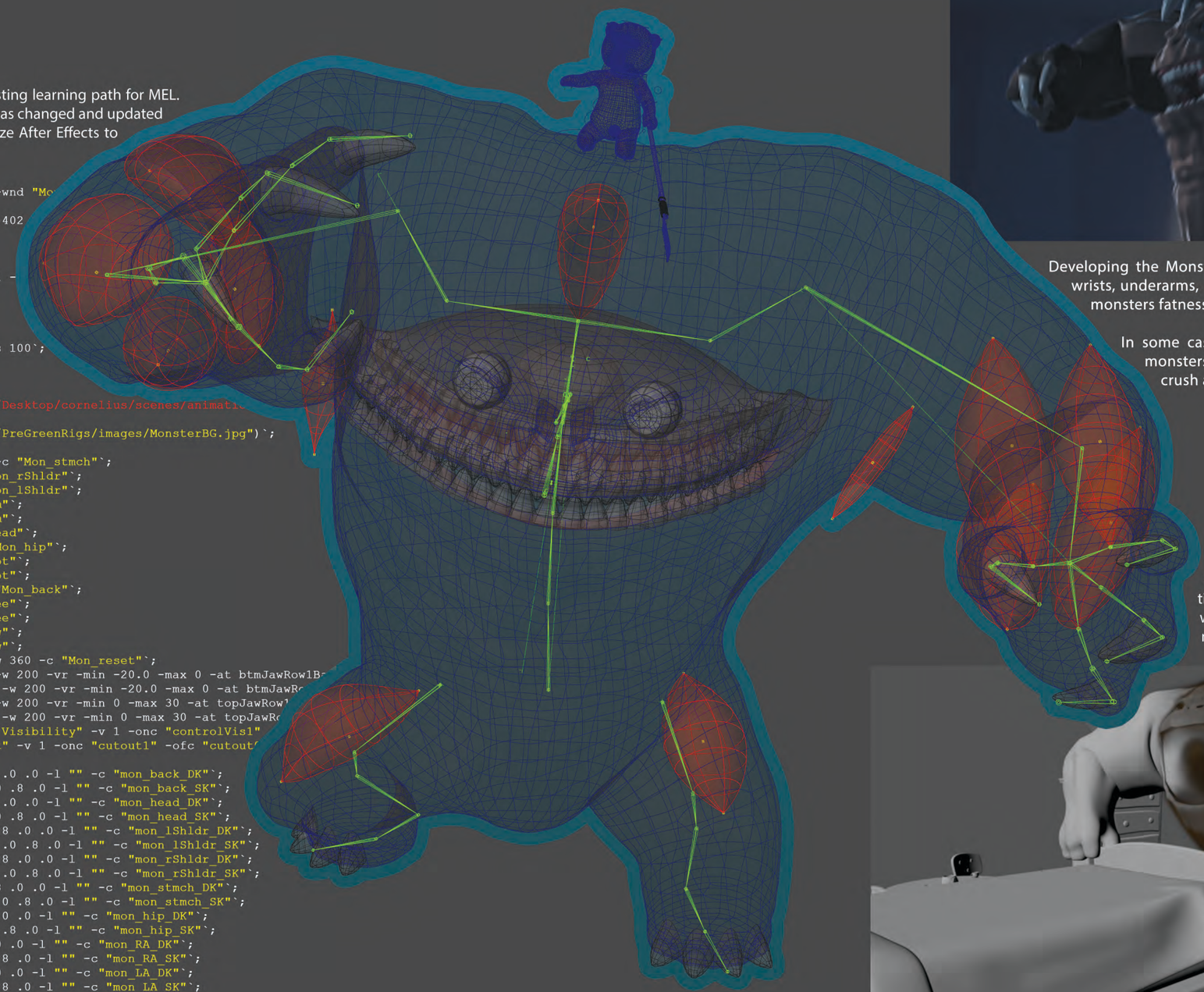
And for credit, the Artists are displayed on the about tab.





Writing the code for the GUI was a fun and interesting learning path for MEL. After the need for sped on GUI creation as the rig was changed and updated so did the GUI. Through this I found a way to utilize After Effects to help with the Layout and design of each GUI.

```
if (`window -exists "MonsterGUI"`) deleteUI -wnd "Mo

window  -t "Monster PreGreen v1.1" -w 360 -h 402
//renameUI window1 MonsterGUI;
//window -exists Monster;

string $tabs = `tabLayout -innerMarginWidth 1 -


string $tab1 = `columnLayout -adj 1`;
//Enter your code here (body)
string $form = `formLayout -numberOfDivisions 100`;
// Set image
string $localUser = `getenv userName` ;

//$img = `image -i ("C:/Users/"+$localUser+"/Desktop/cornelius/scenes/animati
string $wSpace = `workspace -q -rd`;
$img = `image -i ($wSpace+"/scenes/animation/PreGreenRigs/images/MonsterBG.jpg")`;
//Buttons
string $btn_ST = `button -l "Stomach" -w 75 -c "Mon_stmch"`;
string $btn_RS = `button -l "Shoulder" -c "Mon_rShldr"`;
string $btn_LS = `button -l "Shoulder" -c "Mon_lShldr"`;
string $btn_RA = `button -l "RA" -c "Mon_rArm"`;
string $btn_LA = `button -l "LA" -c "Mon_lArm"`;
string $btn_HD = `button -l "Head" -c "Mon_head"`;
string $btn_HP = `button -l "Hip" -w 75 -c "Mon_hip"`;
string $btn_RF = `button -l "RF" -c "Mon_rFoot"`;
string $btn_LF = `button -l "LF" -c "Mon_lFoot"`;
string $btn_BK = `button -l "Back" -w 75 -c "Mon_back"`;
string $btn_RK = `button -l "RK" -c "Mon_rKnee"`;
string $btn_LK = `button -l "LK" -c "Mon_lKnee"`;
string $btn_LB = `button -l "LE" -c "Mon_lBow"`;
string $btn_RB = `button -l "RE" -c "Mon_rBow"`;
string $reset = `button -l "Reset Monster" -w 360 -c "Mon_reset"`;
string $sldr_BJ = `attrFieldSliderGrp -l "" -w 200 -vr -min -20.0 -max 0 -at btmJawRow1Ba
string $sldr_BJi = `attrFieldSliderGrp -l "" -w 200 -vr -min -20.0 -max 0 -at btmJawRo
string $sldr_TJ = `attrFieldSliderGrp -l "" -w 200 -vr -min 0 -max 30 -at topJawRow1
string $sldr_TJi = `attrFieldSliderGrp -l "" -w 200 -vr -min 0 -max 30 -at topJawRo
string $ctrl_vis =` checkBox -label "Control Visibility" -v 1 -onc "controlVis1"
string $cutout_vis =` checkBox -label "Cutout" -v 1 -onc "cutout1" -ofc "cutout
//keying
string $back_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_back_DK"`;
string $back_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_back_SK"`;
string $head_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_head_DK"`;
string $Head_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_head_SK"`;
string $lShldr_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_lShldr_DK"`;
string $lShldr_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_lShldr_SK"`;
string $rShldr_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_rShldr_DK"`;
string $rShldr_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_rShldr_SK"`;
string $stmch_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_stmch_DK"`;
string $stmch_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_stmch_SK"`;
string $hip_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_hip_DK"`;
string $hip_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_hip_SK"`;
string $RA_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_RA_DK"`;
string $RA_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_RA_SK"`;
string $LA_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_LA_DK"`;
string $LA_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_LA_SK"`;
string $LF_DK = `button -w 5 -h 20 -bgc .8 .0 .0 -l "" -c "mon_LF_DK"`;
string $LF_SK = `button -w 13 -h 20 -bgc .0 .8 .0 -l "" -c "mon_LF_SK"`;
string $TJB_S = `button -w 20 -h 20 -l "sel" -c "mon_topJawRow1Base"`;
string $TJB2_S = `button -w 20 -h 20 -l "sel" -c "mon_topJawRow2Base"`;
string $BJB_S = `button -w 20 -h 20 -l "sel" -c "mon_bottomJawRow1Base"`;
string $BJB2_S = `button -w 20 -h 20 -l "sel" -c "mon_bottomJawRow2Base"`;
```

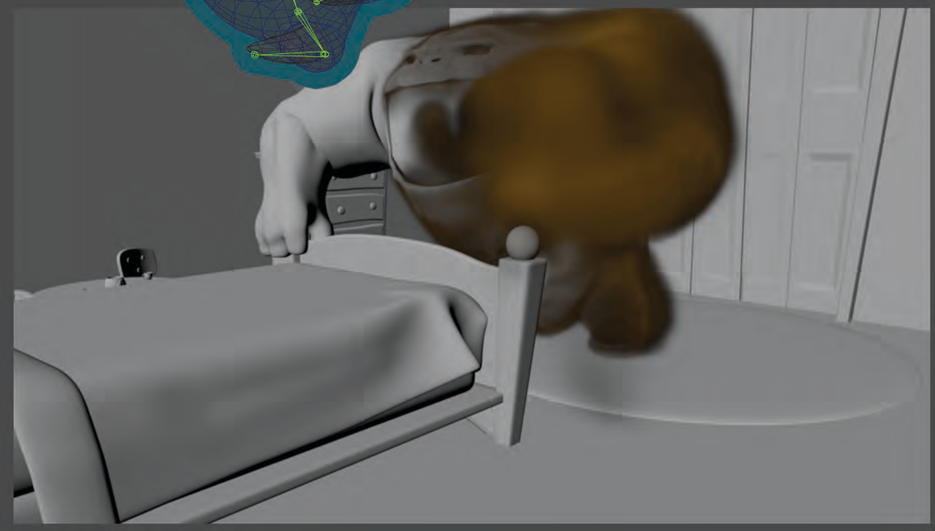


Shot 17_1

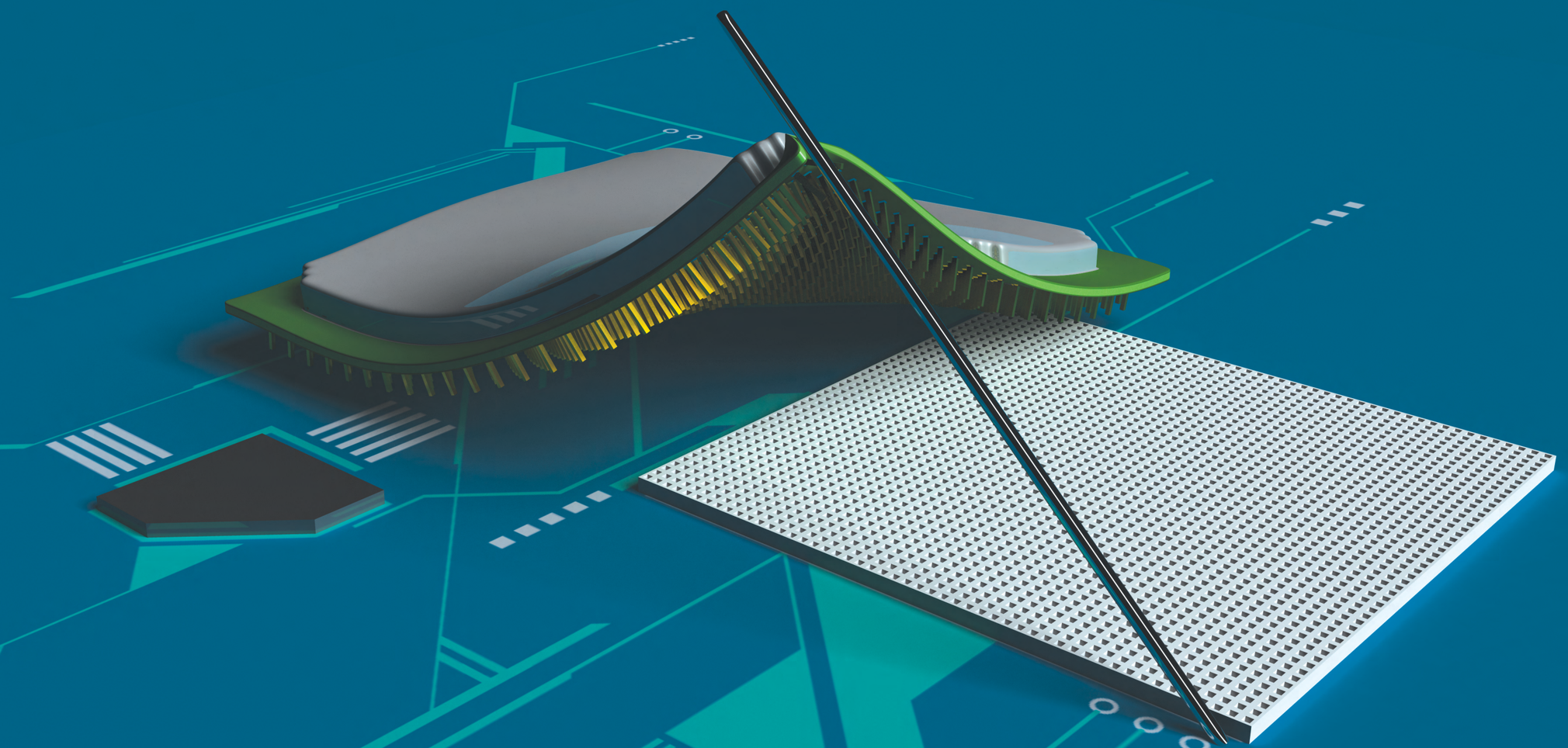
Developing the Monster rig, I chose to use Maya Muscle to articulate the wrists, underarms, neck and upper-legs. Utilizing this system allowed the monsters fatness move in a more realistic way.

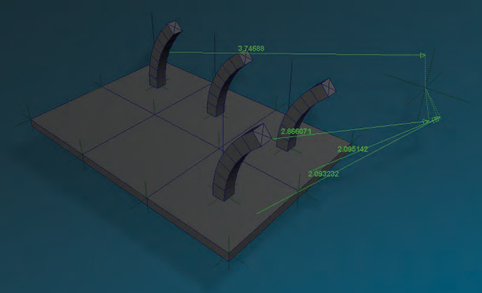
In some cases Mays Muscle was the only way to solve for the monsters underarms. This was because without it the pits would crush and stretch in unattractive ways.

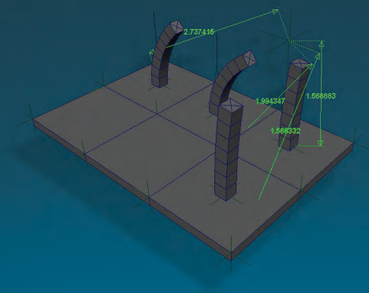
An original concept of the story revolved around the monster being made of shadow or Mist. Countless hours had been put into this development as average calculation times would range from 4-5 hours for every 150 frames. As a way of over comming that challenge small scale tests were made and understood then applied to the final.

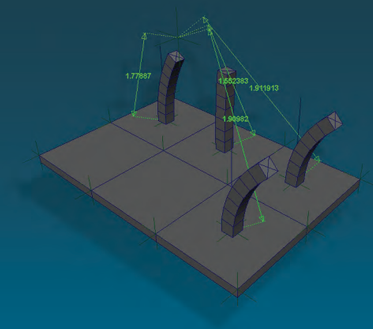


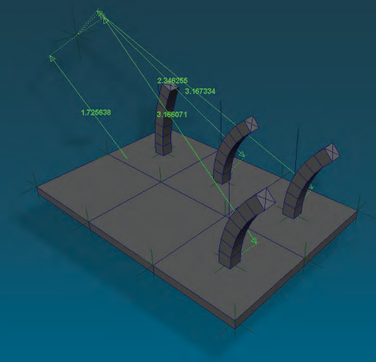
Processor Overload







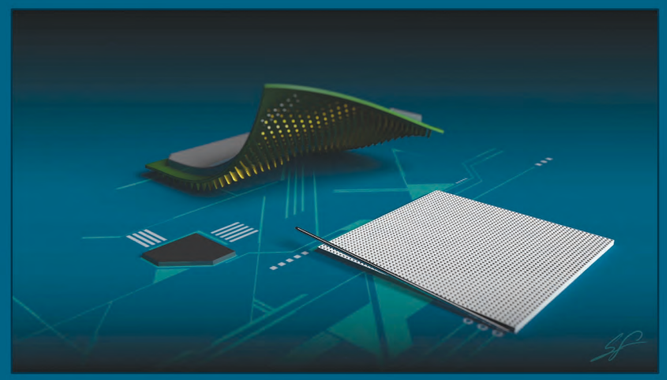


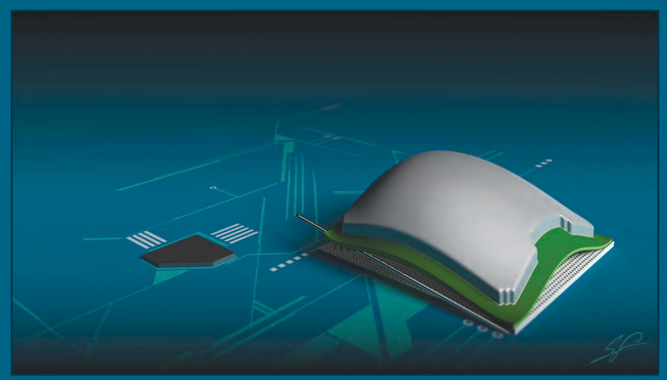


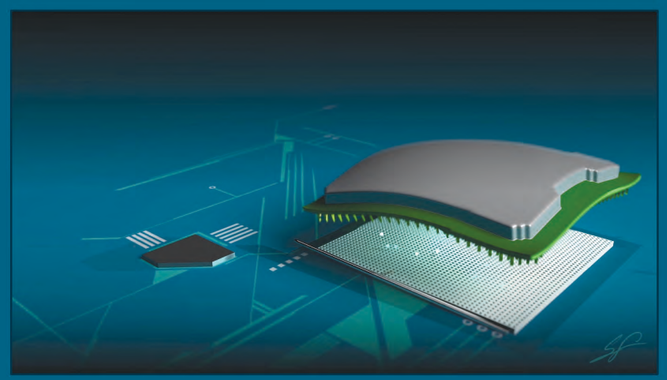

```
// vertLocator1

float $vertPos[] = `pointPosition baseChip.vtx[2502]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator1;

// vetLocator2

float $vertPos[] = `pointPosition baseChip.vtx[2451]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator2;

// vetLocator3

float $vertPos[] = `pointPosition baseChip.vtx[2452]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator3;

// vetLocator4

float $vertPos[] = `pointPosition baseChip.map[2503]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator4;

// vetLocator5

float $vertPos[] = `pointPosition baseChip.map[2399]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator5;

// vetLocator6

float $vertPos[] = `pointPosition baseChip.map[2347]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator6;

//-----------------------------------------------------------------baseChip.vtx[1939:1986]
// vertLocator526

float $vertPos[] = `pointPosition baseChip.vtx[1939]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator526;

// vertLocator527

float $vertPos[] = `pointPosition baseChip.vtx[1940]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator527;

// vertLocator528

float $vertPos[] = `pointPosition baseChip.vtx[1941]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator528;

// vertLocator529

float $vertPos[] = `pointPosition baseChip.vtx[1942]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator529;

// vertLocator530

float $vertPos[] = `pointPosition baseChip.vtx[1943]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator530;

// vertLocator531

float $vertPos[] = `pointPosition baseChip.vtx[1944]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator531;

// vertLocator532

float $vertPos[] = `pointPosition baseChip.vtx[1945]`;
move -ws $vertPos[0] $vertPos[1] $vertPos[2] vertLocator532;
```

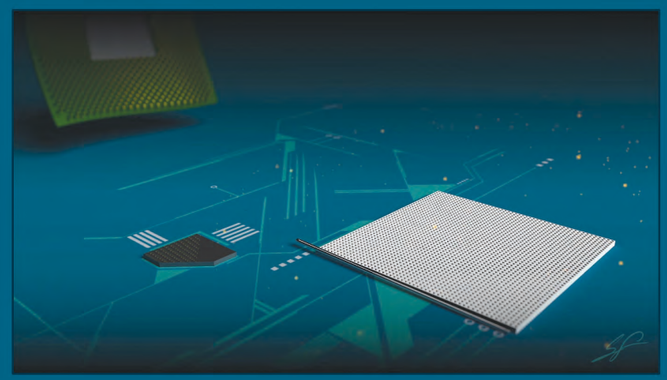
Processor Overload is a short about a processor when finds his way into a socket that's not his. There are 964 pins on the processor all of which are rigged and animated to follow an "eye."

In order to do this, I took this project as a learning curve for procedural setups. Starting small scale I built a system to recognize tagged faces and build the necessary setups to allow each pin to follow the face and distinguish the distance between itself and the "eye" resulting in a wave effect when the eye moves.









```
1: // Editables
2: string $title = "Technical Artist/ Animator";
3: string $name = "Shawn Frueh";
4: string $url = "www.shawnfrueh.com";
5: // Show Title
6: print ($name+"\n");
7: print ($title+"\n\n");
8: print ($url+"\n");
```

```
Shawn Frueh :1
Technical Artist / Animator :2
:3
www.shawnfrueh.com :4
```