

ここでは、環境構築、プロジェクトファイルの実行方法、入力ファイルの作り方などを説明する。研究室の掲示板の卒論のページの私の論文や、スライドなどのリンクが張られているところがあり、そこに「プログラム」というリンクがあるためそれをクリックし、ダウンロードしておくこと。

まず、環境構築について説明する。

今回私は「Visual studio 2017」という環境にてプログラムを作成している。

そのため、まずは「Visual studio 2017」をインストールする必要がある。

「Visual studio 2017」のインストール方法についてはネットの方が詳しいため、そちらを参照したほうが良い。途中でインストールするもの選ぶ場面があるが C++の機能は入れといたほうがいいかもしれない。参考になりそうなものを下に添付する(少し古いが)。

<https://www.softantenna.com/wp/tips/visual-studio-2017-install/>

また、研究室の掲示板の「プログラミング環境の構築」っていうところに「Visual studio 2017」の環境構築の動画のリンクがあるためそちらも参考に。

「Visual studio 2017」を無事インストールしたら、私のプロジェクトファイルにあるソリューションファイル(.sln)をダブルクリックで開く。開いたときに、「ソリューションの再ターゲット～」みたいなのが出てきたときは、再ターゲットをすること。開いたら、試しに実行(Ctrl + f5 で実行できます)すること。もし、普通に実行できればそのままでよいが、もし実行できなければ、再度環境構築する必要がある。今回のプログラムでは「OpenGL」と「CLAPACK」というものを用いている。研究室の掲示板の「プログラミング環境の構築」っていうところの動画(「Visual studio 2017」の時と同じ)に「OpenGL」と「CLAPACK」の環境構築も説明しているのでそれを参考に。

また、入力ファイルを作成するにあたって、「blender」というソフトを用いている。

これもインストール方法はネットの方が詳しく掲載されているため今回は省略する。

いかに参考になるリンクを貼る

<https://blender-cg.net/download-install/>

<https://blender.jp/modules/xfsection/article.php?articleid=277>

<https://www.kunihikokaneko.com/dblab/toolchain/blender.html>

次にプロジェクトファイルの使い方について説明する。

- ・ソースコードの概要

main.cpp

→ここで、どのような入力ファイルを読み込むかなどの設定や描画などを行っている。

Function1.cpp

→ここにはファイル読み込み用関数や、内積などを計算する関数を定義している。

Function2.cpp

→長方形の自動生成や、外周の変形、Obj ファイル読み込みなどの関数を定義している。

Arap.cpp

→As Rigid As Possible 法の第 1 段階、第 2 段階、第 3 段階の処理の関数が定義している。

Class.h

→オリジナルのクラスを定義している

変更可能な部分は **main.cpp** の上部にすべてまとめている。以下にその概要を記す。

TYPE…ファイルを読み込むか、長方形のメッシュを自動生成するかを決める。

0 に設定するとファイル読み込み、1 に設定すると自動生成を行う。

WNUM…重みをつける要素の数(長方形メッシュにのみ有効)

WEIGHT…重み付けの強さ(長方形メッシュにのみ有効)

height…TYPE=1 の時に用いられる。長方形の縦の分割数

width…TYPE=1 の時に用いられる。長方形の横の分割数

str1…TYPE=0 の時に用いられる。転写前のメッシュの情報を格納した
ファイルの名前をここで入力する。

str2…TYPE=0 の時に用いられる。既存のファイル(.txt)の場合は外周上の点の番号と
移動後の外周上の点の座標を格納したファイルの名前を入力。

Obj ファイルの場合外周上の節点のみの情報を格納したファイルの
名前を入力する。

str3…TYPE=0 の時に用いられる。既存ファイルの場合は重みをつける要素の番号を
格納したファイルの名前を入力。Obj ファイルの場合は
変形させた外周上の節点の座標を格納

weightF…重みをつけるかどうかのフラグ。

true であれば重みをつける、false であれば重みをつけない。

transform…外周をどのように変形させるのか。

また、どのようなファイルを読み込むのかなどをここで指定する。

例えば、**transform="BEND"**と入力すると **BEND** の変形を行う。

以下にどんな入力をするとどの変形を行うかを記す。

TYPE=1 のとき、

“BEND”…長方形のメッシュを曲げる。

rad という変数に曲げる角度を入力する。例えば **30** 度曲げたい場合、

rad=30.0f * (float)M_PI / 180.0f; と入力する。

“SKEW”…長方形のメッシュをせん断変形させる。

skew という変数に値を入力することで長方形の上底を

x 軸方向にどのくらい平行移動させるかを決める。

例えば **skew=3.0f** とすると

x 軸方向に **3.0f** 長方形の上底を平行移動させる

“SCALE”…長方形のメッシュを拡大縮小させる。

scale_x にどのくらい x 軸方向に拡大縮小させるのかを、

scale_y にどのくらい y 軸方向に拡大縮小させるのかを入力する。

例えば、**scale_x=0.5f; scale_y=1.5f;** とすると x 軸方向に **0.5** 倍縮小し、

y 軸方向に **1.5** 倍拡大する変形が行われる。

“B_SCALE”…長方形の上底、下底を両側から拡大縮小させたり、

右辺左辺を両側から拡大縮小させる。

まず、**direction** という変数に”X”か”Y”を入力。

direction="X" なら上底下底を、

direction="Y" なら右辺左辺を拡大縮小変形させる。

“X”的ときは **scale_x** を用いてどのくらい拡大縮小させるのか、

“Y”的ときは **scale_y** を用いてどのくらい拡大縮小させるのかを決める。

使い方は”SCALE”と同じ。

“O_SCALE”…長方形の上底、下底を片側から拡大縮小させたり、

右辺左辺を片側から拡大縮小させる。

まず、**direction** という変数に”X”か”Y”を入力。

direction="X" なら上底下底を、

direction="Y" なら右辺左辺を拡大縮小変形させる。

“X”的ときは **scale_x** を用いてどのくらい拡大縮小させるのか、

“Y”的ときは **scale_y** を用いてどのくらい拡大縮小させるのかを決める。

使い方は”SCALE”と同じ。

transform(続き)…

“**TRANSLATE**”…長方形メッシュを平行移動させる。

trans_x,trans_y に値を入力し, x 方向, y 方向に平行移動させる。

例えば **trans_x=1.0f;trans_y=0.5f;** にした時,

x 方向に **1.0**, y 方向 **0.5** 動く。

“**ROTATION**”…長方形メッシュを回転させる。

Cx, Cy にどこに回転の中心を持ってくるかを入力。

rad に回転する角度を入力する。

例えば座標が **x=1.0f, y = 1.0f** を中心に **45** 度回転させたい場合,

Cx=1.0;Cy=1.0, rad=45.0f * (float)M_PI / 180.0f; と入力する。

“**R_ANGLE**”…長方形メッシュを直角に曲げる変形を行う。

TYPE=0 の時,

“**MON**”…門型のメッシュの内側の線を動かす変形を行う。

入力するファイルは以下のように入力すること。

str1 = “MON(main).txt”

str2 = “sample(mon)2-2.txt”

str3 = “ArmBend1-3.obj” (このファイルは何でもよいが

入力は必要, 入力しないとバグ起きる可能性あり)

rate に値を入力することで内側の線をより外側,

またはより内側に寄せる変形を行う。

“**OBJ**”…Obj ファイルを入力する。

str1 = “~.obj” (転写前のメッシュの情報)

str2 = “~.obj” (転写前のメッシュの外周の情報)

str3 = “~.obj” (転写先のメッシュの外周の情報)

流れとしては **str1, str2** のファイルに格納されたメッシュを

str3 のファイルに格納された外周に反映させる。

ファイルの詳しい構成, 作り方は後に記述する。

output1…場所, 名前を入力することで, 転写前のメッシュ情報を出力する。

今回のプロジェクトを実行する上では必要ない。

用途としてはメッシュの座標がどうなってるかを確認するときなど。

output2…場所, 名前を入力することで, 転写先のメッシュの外周情報を出力する。

今回のプロジェクトを実行する上では必要ない。用途は上と同じ。

Output3…場所, 名前を入力することで, 重み付けを行う要素の情報を出力する。

今回のプロジェクトを実行する上では必要ない。用途は上と同じ。

以上が、プロジェクトファイルを実行する上で変更してもよい箇所である。
それ以外は変更しないことを進める。(バグが起きる可能性大)

```
//-----変更可能なグローバル変数-----
#define TYPE 0 //TYPEが1の時は長方形を自動生成して、それを転写させる。TYPEが0の時はファイルを読み込んでその情報を元に転写を行う
#define VNUM 144 //40 //重みをつける要素の数
#define WEIGHT 10000.0f //重みの数値
int height = 20; //長方形の縦の長さ(分割数)
int width = 10; //長方形の横の長さ(分割数)
const char* str1 = "Input/otameshi/sample1-1.obj"; //MON(main).txt
const char* str2 = "Input/otameshi/sample1-2.obj"; //sample(mon)2-2.txt
const char* str3 = "Input/otameshi/sample1-3.obj"; //ArmBend1-3.obj
bool weightF = false; //重み付けを行うかどうかのフラグ
const char* transform = "OBJ";
float rate = 0.0f; //門変形を行う際、どのくらい内側または外側に寄せるかの値
float rad = 90.0f * (float)M_PI / 180.0f; //曲げるまたは回転させる際の角度
float skew = 3.0f; //せん断変形で長方形の上底をどのくらい平行移動させるか
float scale_x = 1.5f; //拡大縮小変形する際のx軸方向の割合
float scale_y = 8.0f; //拡大縮小変形する際のy軸方向の割合
const char* direction = "Y"; //どちらの方向に変形させるか
float trans_x = 3.0f; //平行移動のx軸方向の値
float trans_y = 1.5f; //平行移動のy軸方向の値
float Cx = width / 2.0f; //回転の軸のx座標
float Cy = height / 2.0f; //回転の軸のy座標
const char* output1 = "Output/hit01-0.txt"; //メッシュの情報を出力(転写前の節点座標、要素を形成する節点番号など)
const char* output2 = "Output/hit01-1.txt"; //メッシュの情報を出力(外周上の移動後の節点座標、節点番号など)
const char* output3 = "Output/hit01-2.txt"; //メッシュの情報を出力(重みをつける要素の番号)
```

図 1:main.cpp の変更可能な部分

次に、入力ファイル(.obj 形式)の作成方法について説明する。

今回入力ファイルの作成に当たって、「blender」というソフトを用いる。

インストールした後、blender を開くとこのような画面になる。

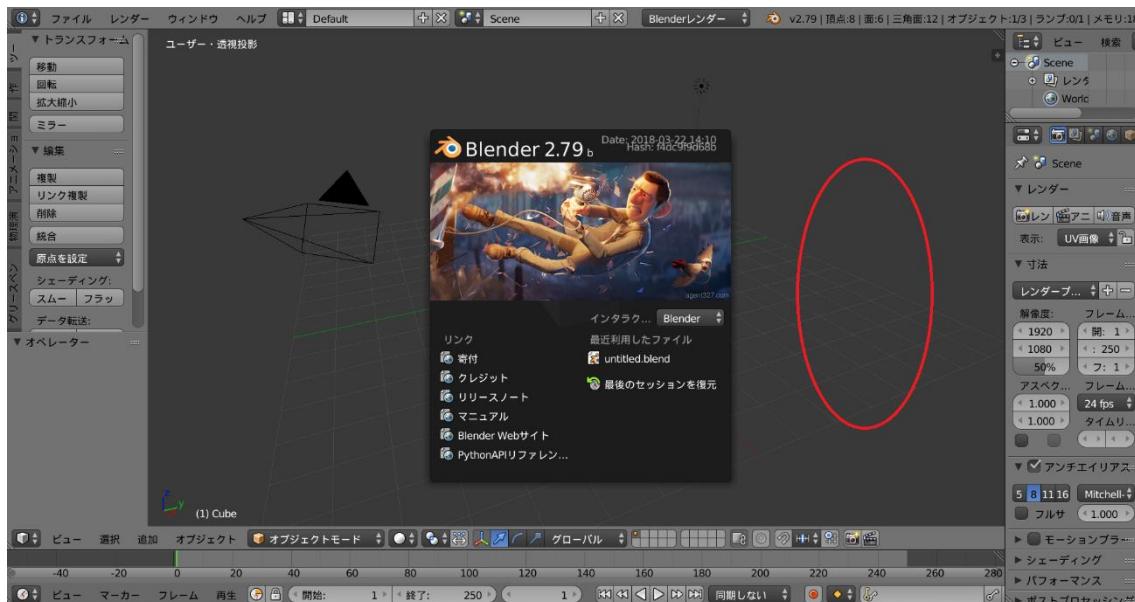
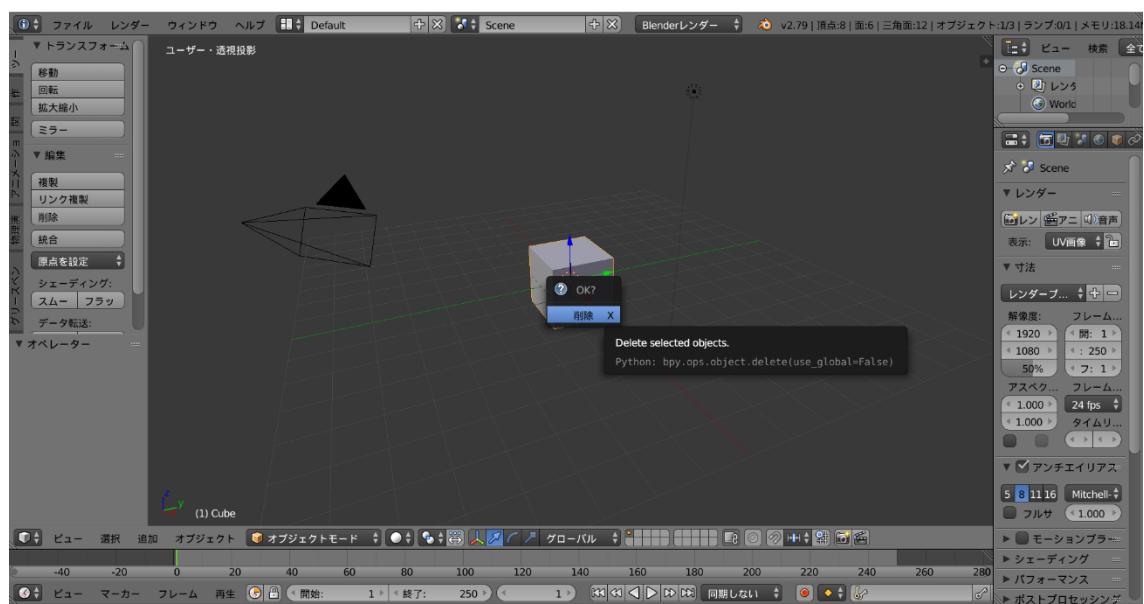
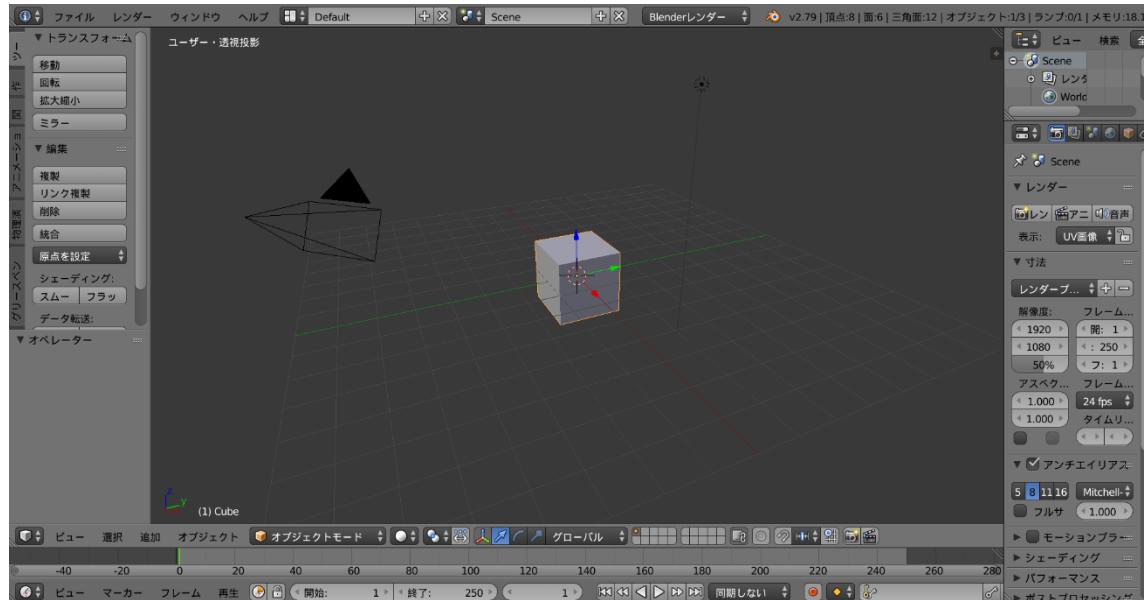


図 2:Blender のデフォルト画面

この広告みたいなのは赤い丸部分あたりをクリックすれば消えるため消しておく。

(適当に広告以外のところクリックしておけばそれでいい)

広告みたいなのを消した後、「Delete」キーを押すと「削除」という項目が出るためクリックすると目の前にあった箱が消える。



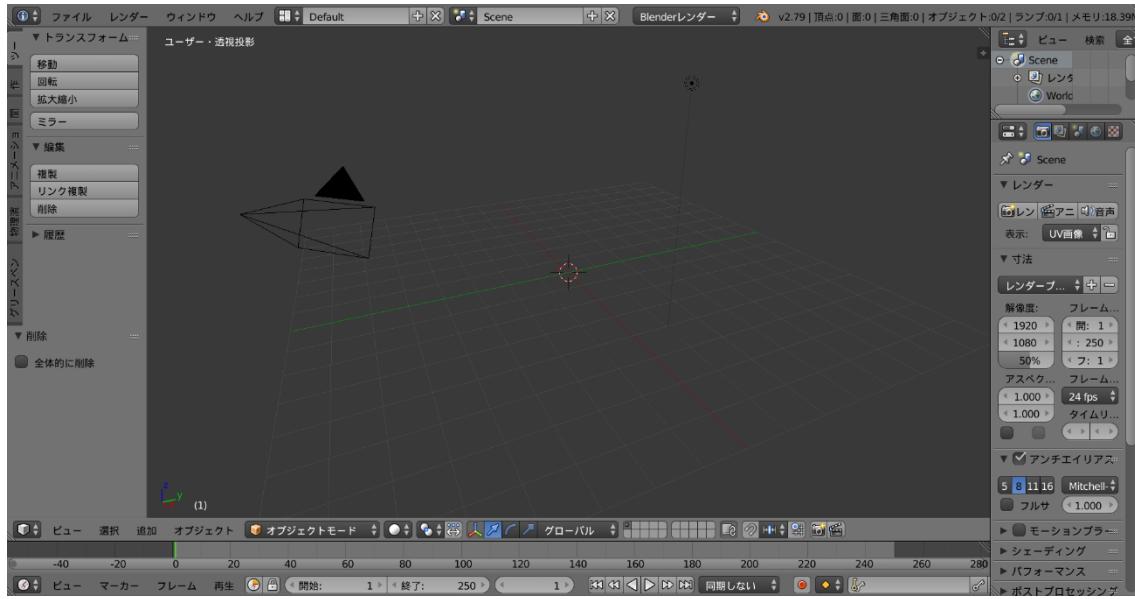


図 3:目の前の箱が消えるまで

入力ファイルを作成する前に作業をより簡単に行えるようにいくつか設定をするが、ネットの説明の方が簡単なため、参考になりそうな URL を以下に貼っておく。

- ・日本語設定… <https://blender-cg.net/nihonngoka/>
- ・テンキーがない人のための設定… <https://blender-cg.net/tennki-daitai/>

今回の入力ファイルは x 座標,y 座標を持つ節点(頂点)で構成されたメッシュ情報が必要である。つまり、2 次元のメッシュであるため、デフォルトの視点でメッシュを変形しようとすると少し、面倒である。それを少しでも改善するため、視点を変える。テンキーの「5」(または、普通のキーボードの「5」)を押し、透視投影から平行投影に変える。続いて、テンキーの「7」(または、普通のキーボードの「7」)を押せば、横方向が x 軸、縦方向が y 軸に見える。この視点で編集すると少しは楽になる。もちろん、blender を使い慣れている人であれば、この操作をしなくても編集できると思うので、この操作をするかどうかは自由である。

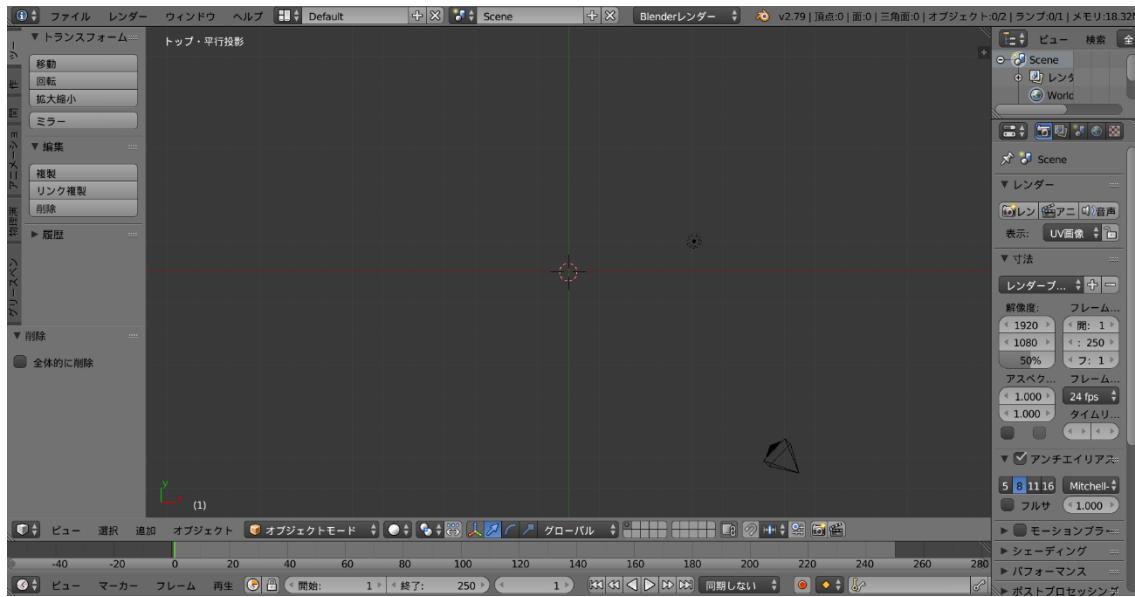


図 4: 視点を変えたときの図

最初にオリジナルの転写前のメッシュを作成する。操作方法についてはこちらでも簡単に説明するがネットで調べたほうがより詳しくのっているのでもしわからなければそれを参考に。(readme の最後の方に URL を貼っておく)まず、「Shift + A」キーでメニューを表示、「メッシュ」→「Plane」(平面)を選択する。すると、正方形のメッシュが生成される。今回はこのメッシュを変形させて、オリジナルのメッシュを作成する。

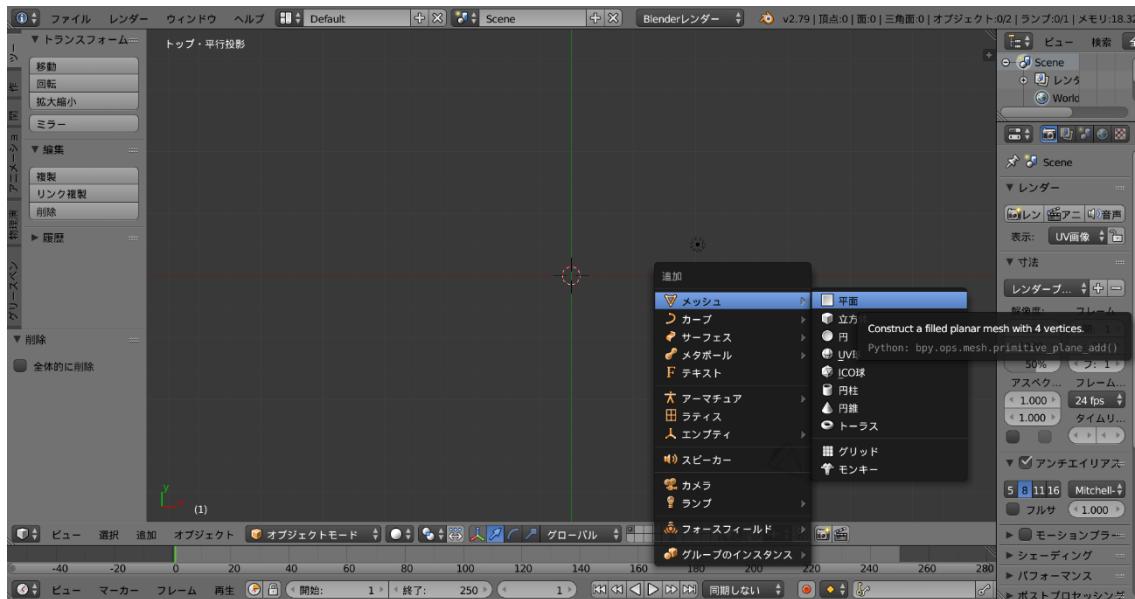


図 5: メッシュ作成

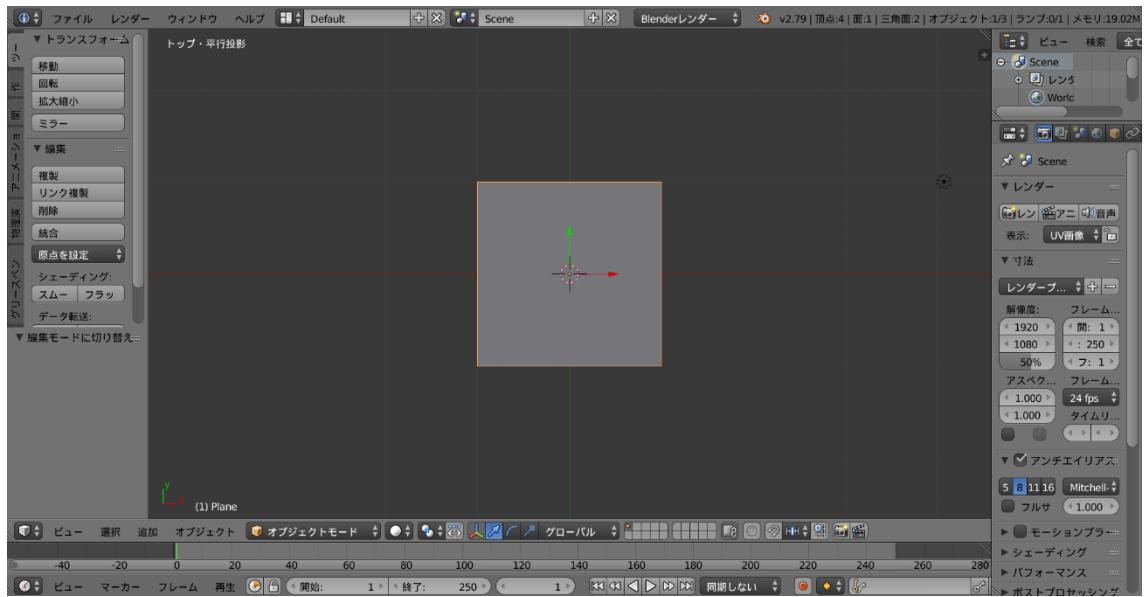


図 6:メッシュ作成した後

まず、「tab」キーを押して「オブジェクトモード」から「編集モード」に変更。

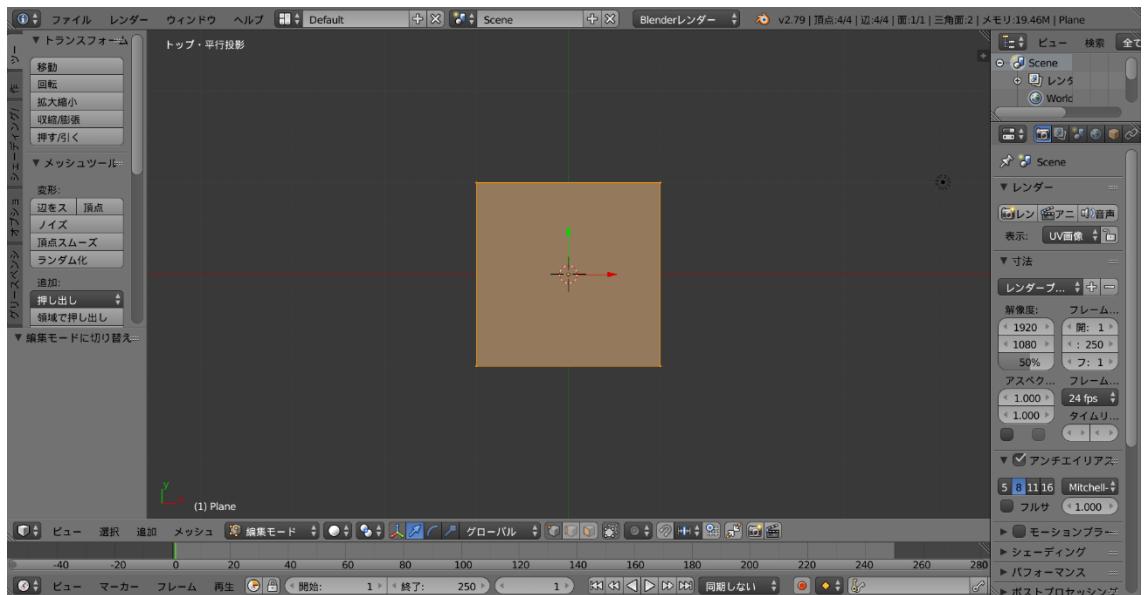


図 7:編集モード

以降、編集モードで作業を行う。「w」キーを押して、「sudvide」(細分化)を選択。選択した直後、左下の部分に「分割数」と書かれているところがあるが、この矢印の部分をクリックすると分割数を増やすことができる。

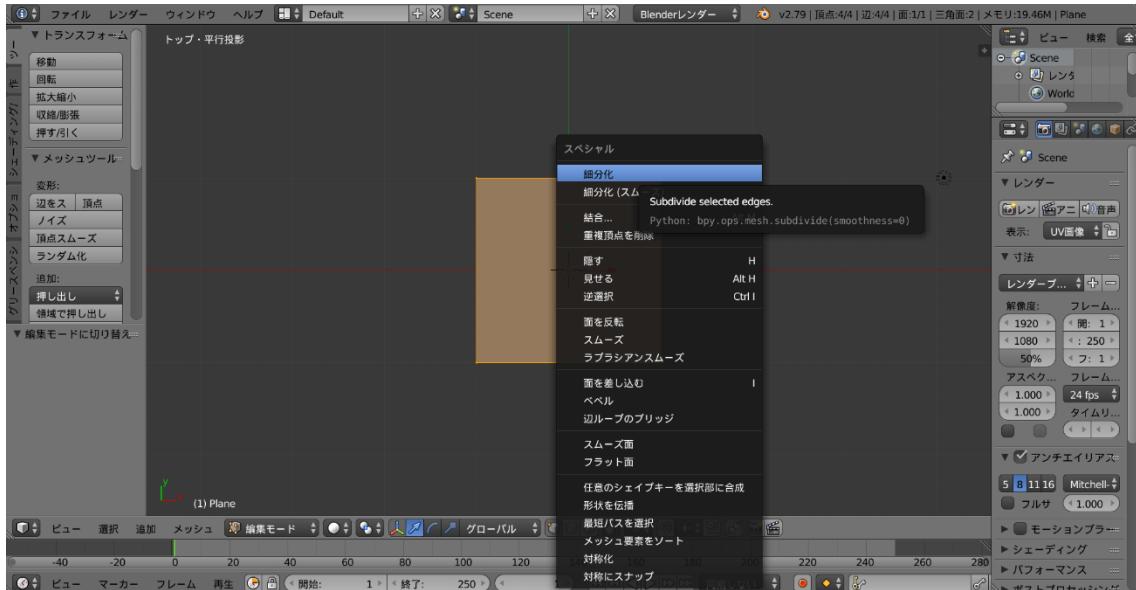


図 8:細分化を選択する図

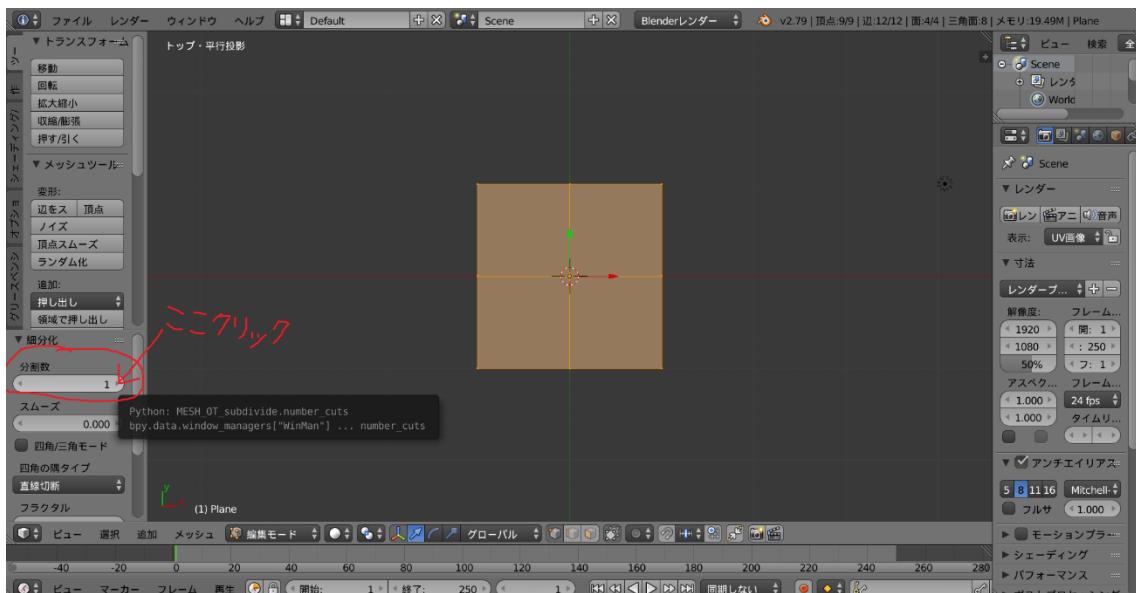


図 9:分割数を示す図

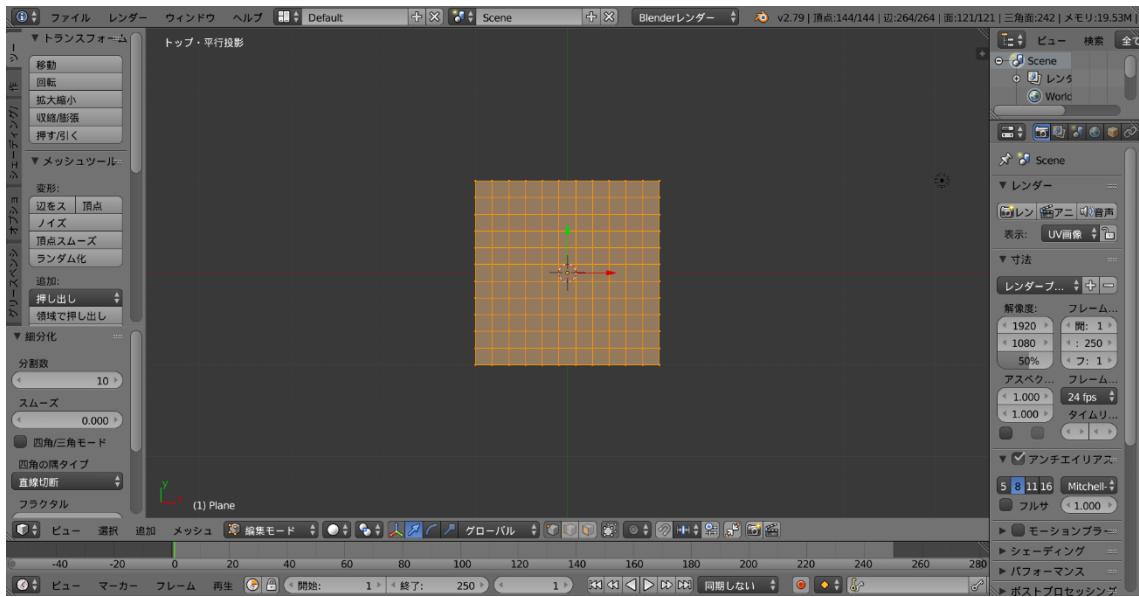


図 10:分割後

続いてメッシュの頂点を右クリックで選択する。選択すると選択部分がオレンジ色になる。
 そのあと「G」キーを押してマウスを動かすと頂点を動かせる。
 その状態のまま、さらに「X」キーまたは、「Y」キーを押すと、
 その方向にのみ動かせるようになる。ほかの節点に重なるように動かしてしまうとメッシュがおかしくなるため、注意が必要。移動させたいところまで動かしたら、
 左クリックをするとその変更が反映される。

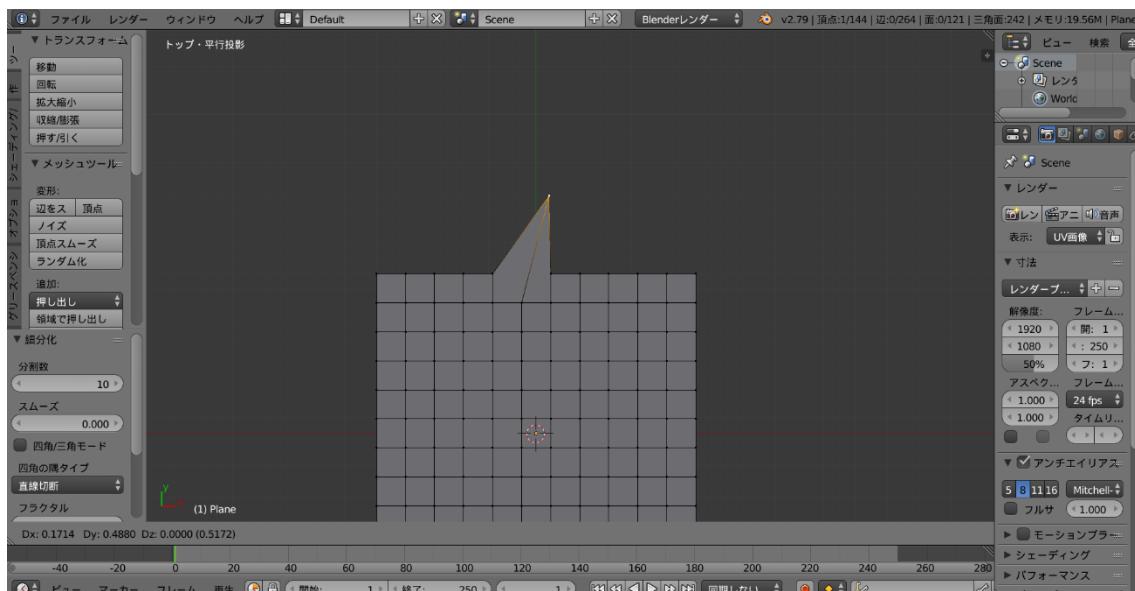


図 11:頂点を動かしている図

また、頂点を選択した状態で「E」キーを押し、
マウスを動かすと選択した頂点を維持しつつ、そこから新たな頂点を作成して
移動させることができる、押し出しという操作ができる。

これも「G」キーとおなじように方向を固定したりすることができる。移動後、
左クリックを押すと、その変形が反映される。

また、操作を誤ってしまい、前の状態に戻りたいときは「Ctrl + Z」で戻ることが可能。

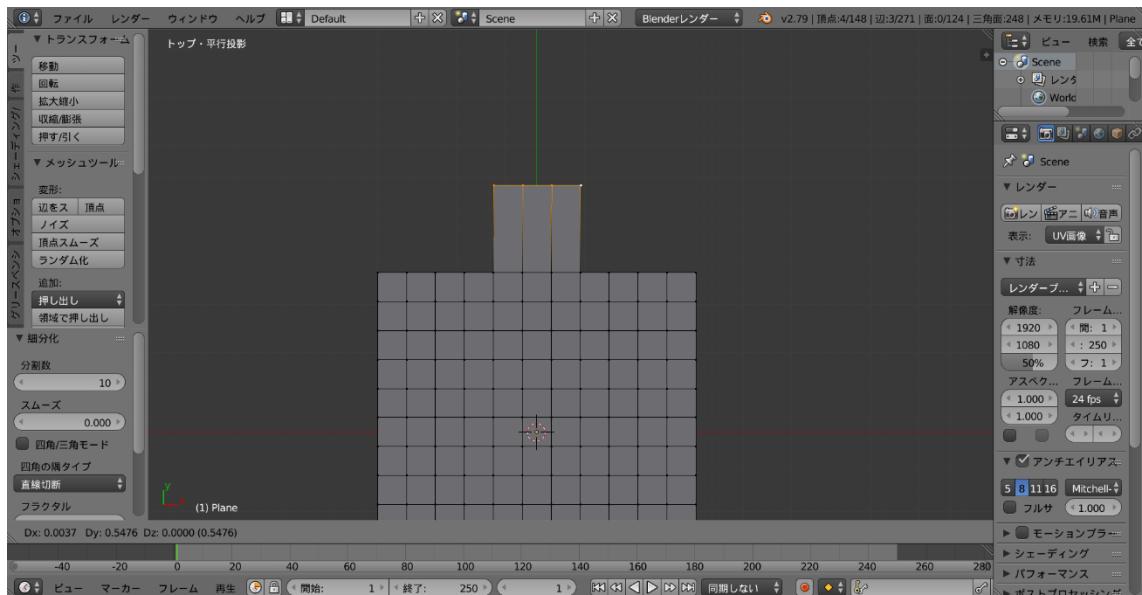


図 12:押し出しをしている様子

頂点を移動した後、メッシュが伸びてしまうため、これをさらに細分化していく。

「Ctrl + R」キーで伸びた部分にマウスを重ねると紫の線が間に表示される。

この状態で左クリックを2回押すと、紫の線通りに線が引かれる。

マウスホイールで紫の線を増やすこともできる。

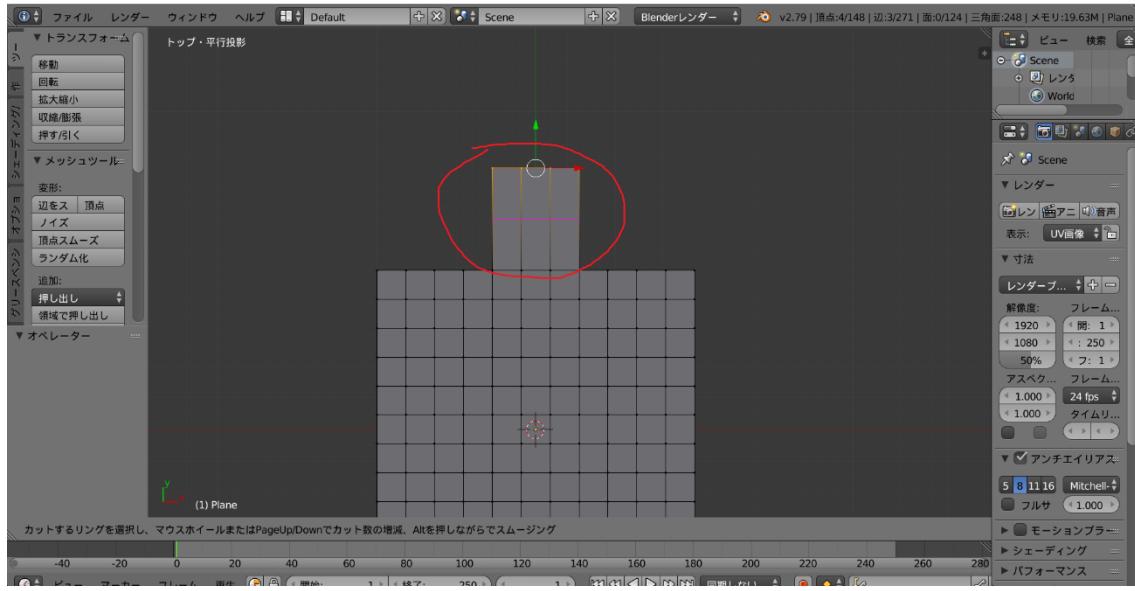


図 13:Ctrl+R キーを押したとき

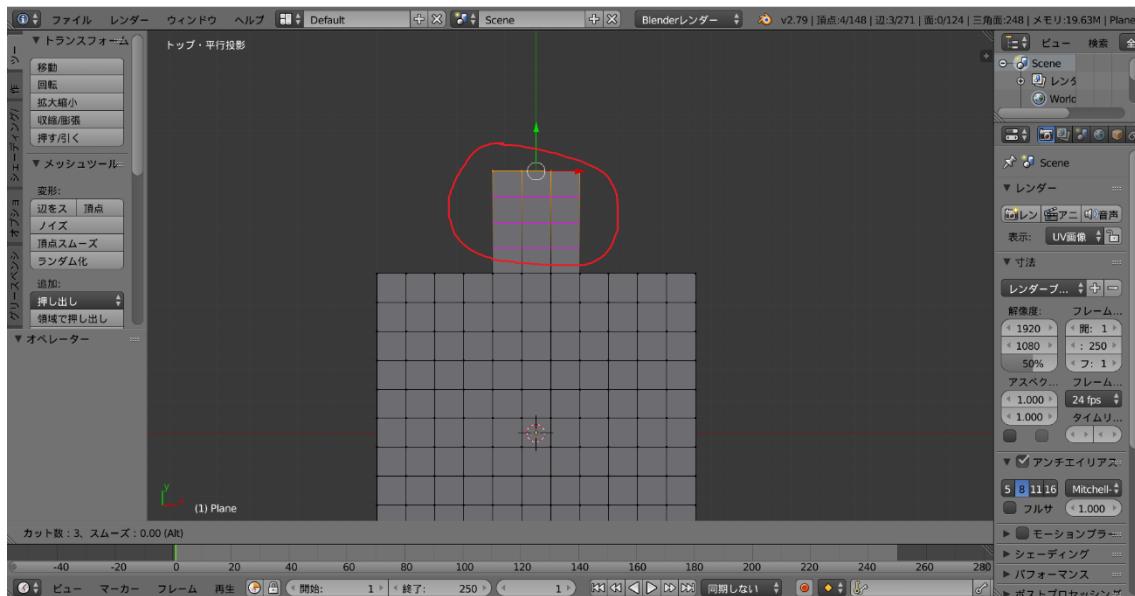


図 14:マウスホイールを回したとき

オリジナルのメッシュを作成出来たら、まず「A」キーを押してメッシュを全選択する。

(もし、ほかの頂点を先に選択していたら、「A」キーで全選択解除になるため、

もう一回押す必要あり)

「Ctrl + T」キーで今まで四角形の要素で構成されたものが三角形に分割される。

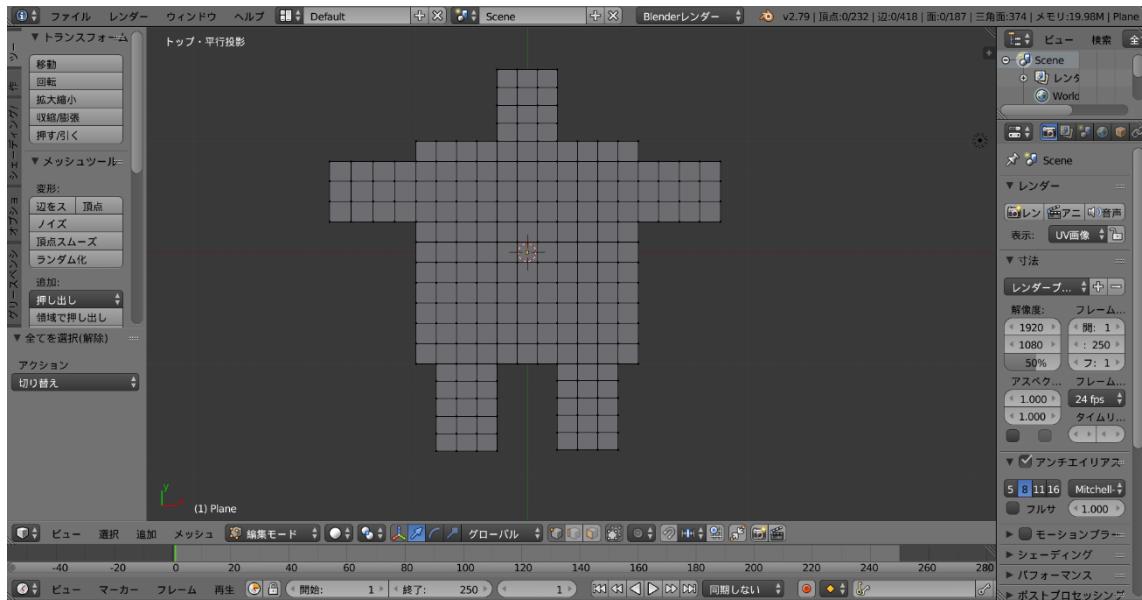


図 15:作成したメッシュ

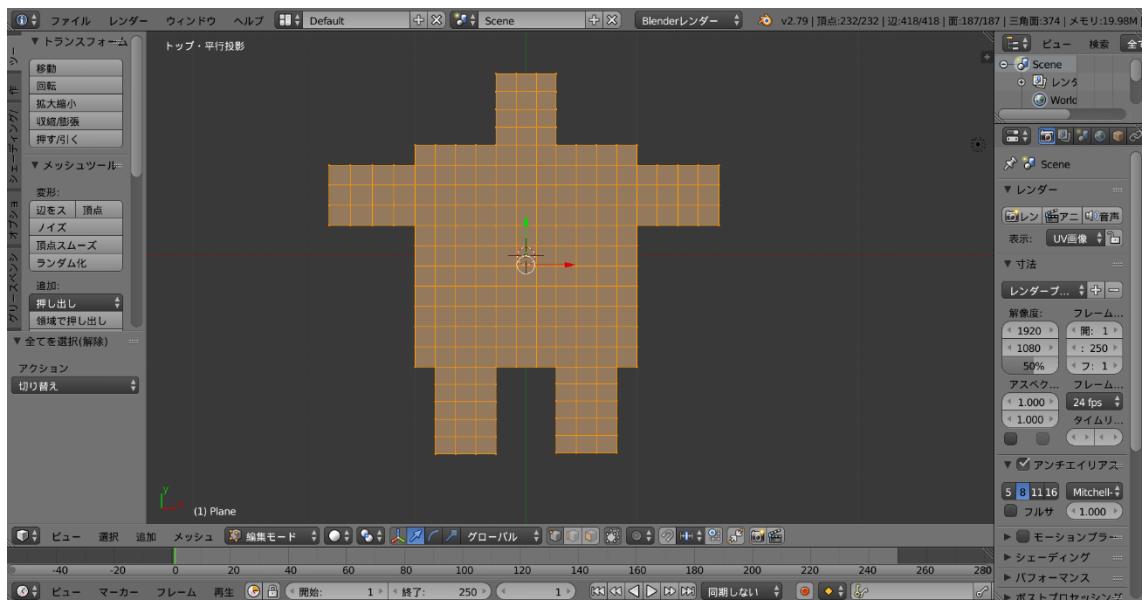


図 16:全選択した状態

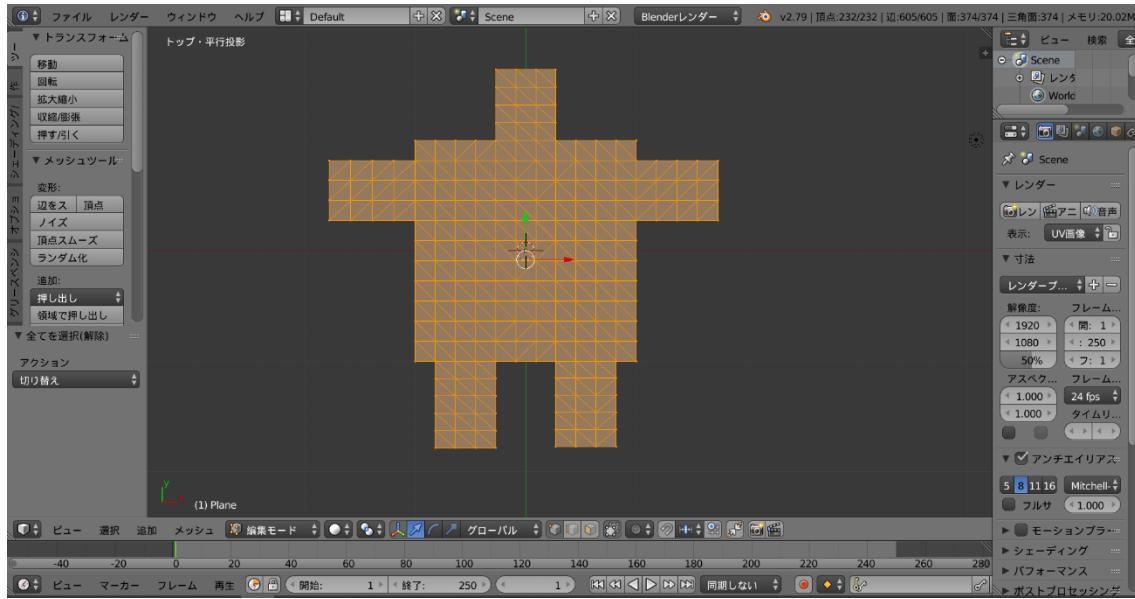


図 17: 「Ctrl + T」 キーを押して、面を三角形に分割

これで、転写前のメッシュを作成することができたため、次は OBJ ファイルに出力する。まず、「ファイル」タブをクリックし、「エクスポート」→「Wavefront(.obj)」の順に選択。すると出力設定画面に変わる。この時、メッシュを選択している状態にしないといけないため、メッシュ全体がオレンジ色になっていなければ「A」キーを 2 回おして全選択状態にしておく。

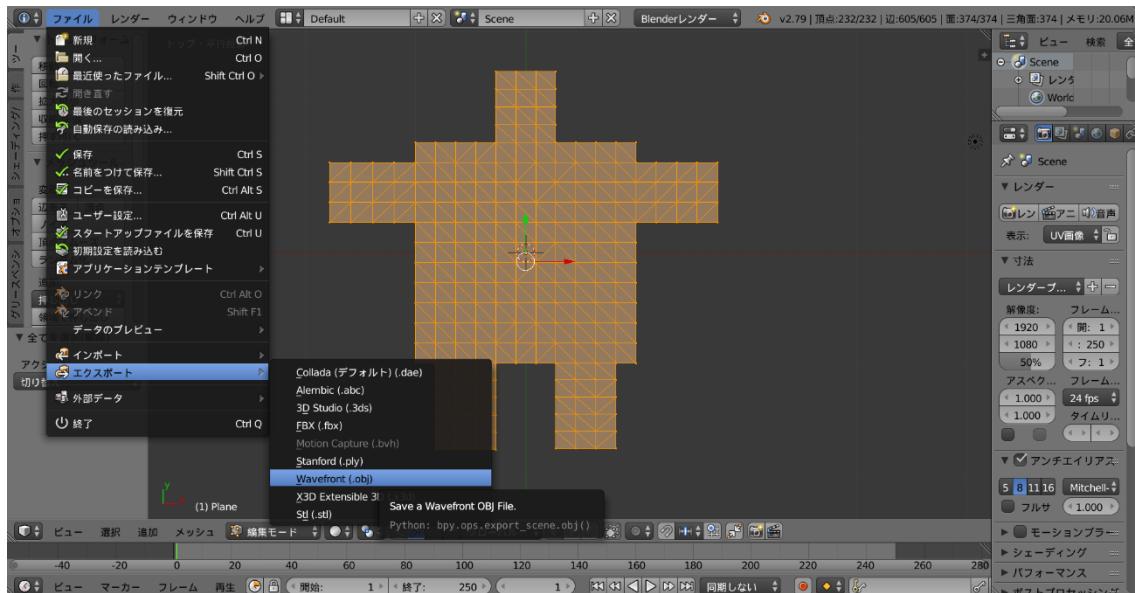


図 18: ファイルメニューを開いている様子

まず右下あたりに「OBJをエクスポート」という欄があり、その下に「前方」、「上」という欄がある。「前方」の右の矢印の部分をクリックし、「Yが前方」を選択。「上」の右の矢印の部分をクリックし、「Zが上」を選択。続いてその下にいくつかチェックボックスがあるが、「選択物のみ」というチェックボックスにだけチェックをつけ、あとは外しておく。その後、一番右上にある「OBJをエクスポート」というボタンの左側に出力先を、その下に出力ファイル名を入力し、「OBJをエクスポート」というボタンをクリックすると、出力ファイルが生成される。これで、転写前のメッシュの情報を出力できた。この時のファイルを「ファイル1」とする。続いて、転写前の外周上の点の情報を出力する。

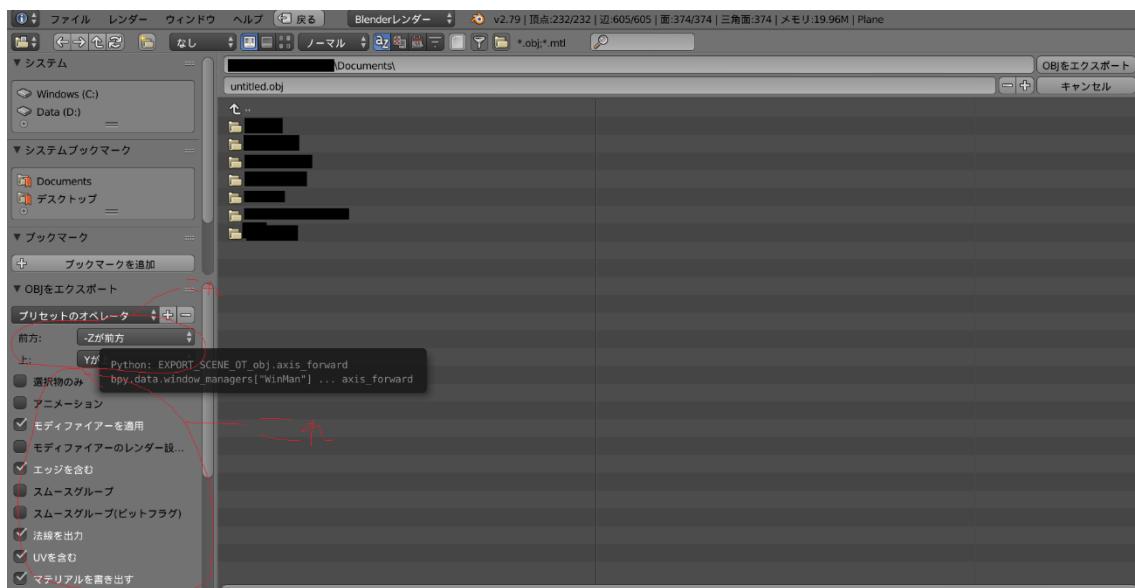


図 19:出力設定画面

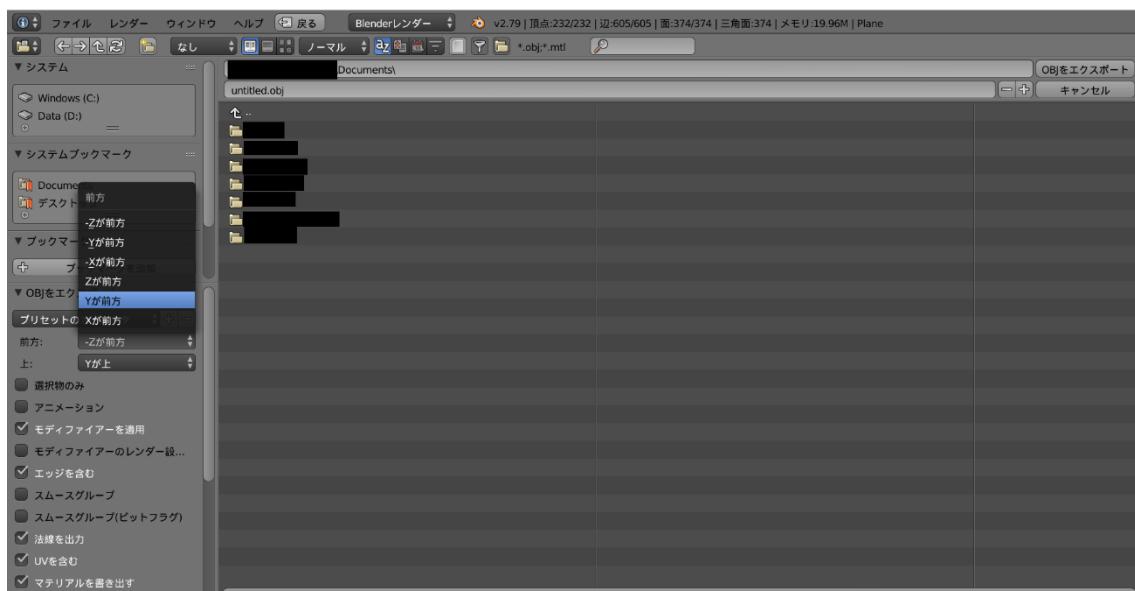


図 20:「前方」「上」を変更

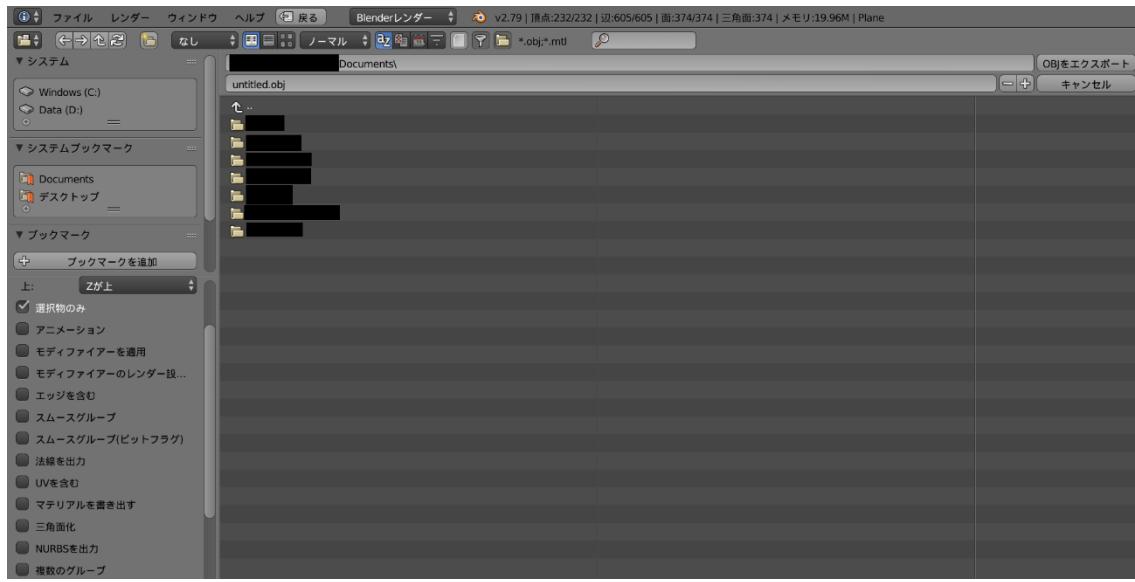


図 21: チェックボックスを「選択物のみ」にチェックを入れる

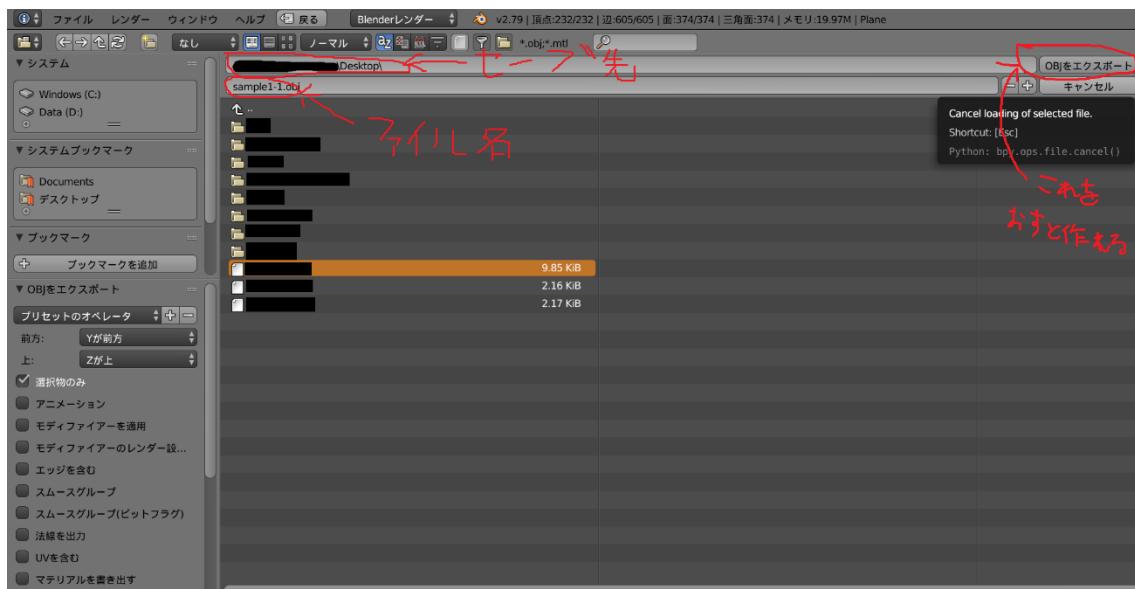


図 22: 保存先, ファイル名を決めて, エクスポート

「OBJをエクスポート」というボタンをクリックすると、元の画面に戻る。

この時、「オブジェクトモード」になっているため、「tab」キーを押して「編集モード」に変更。外周上の点を一つ右クリックで選択する。

外周のみを選択したいため、「Alt + Shift + 右クリック」で外周を選択する。

この時、間隔が広いと選択したくないところまで選択してしまうため、間隔を広げすぎないようにする。

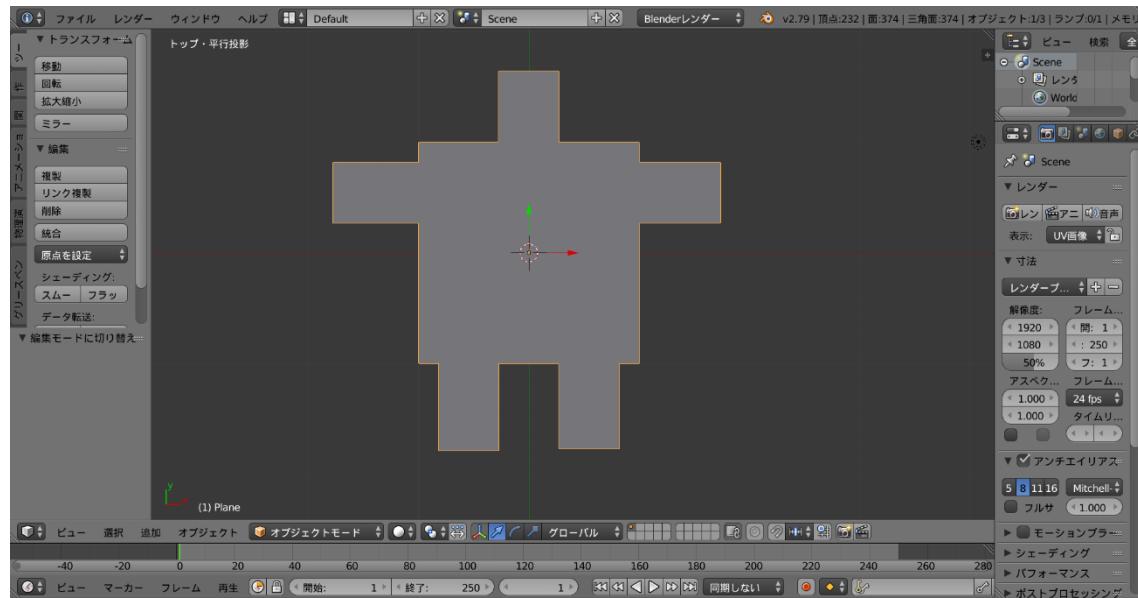


図 23: 「Objをエクスポート」をクリックした直後の状態

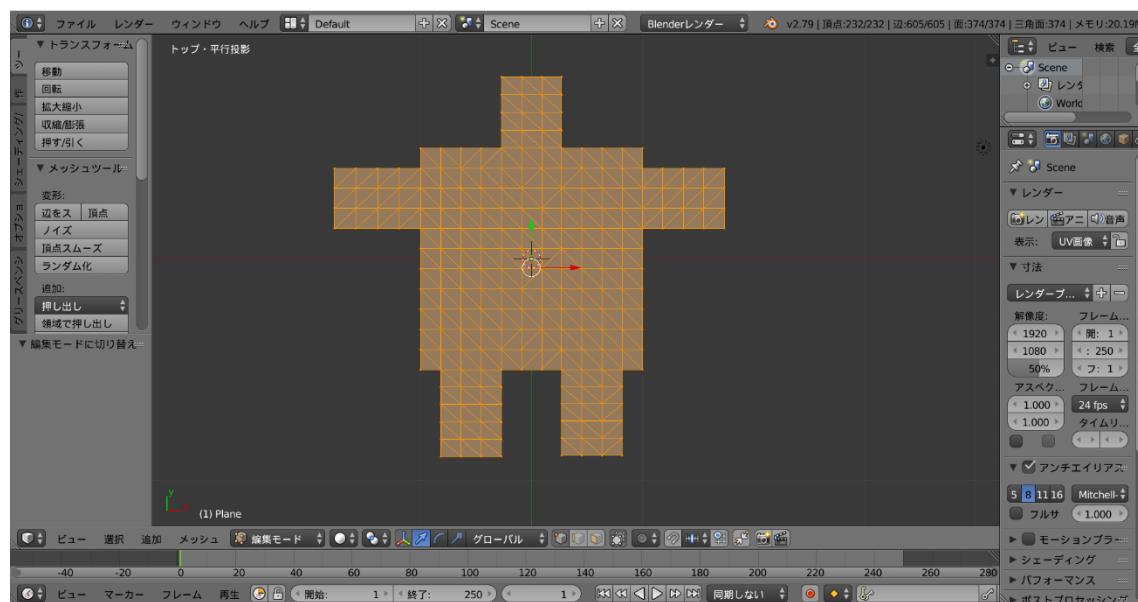


図 24: 「tab」キーを押して編集モードに設定

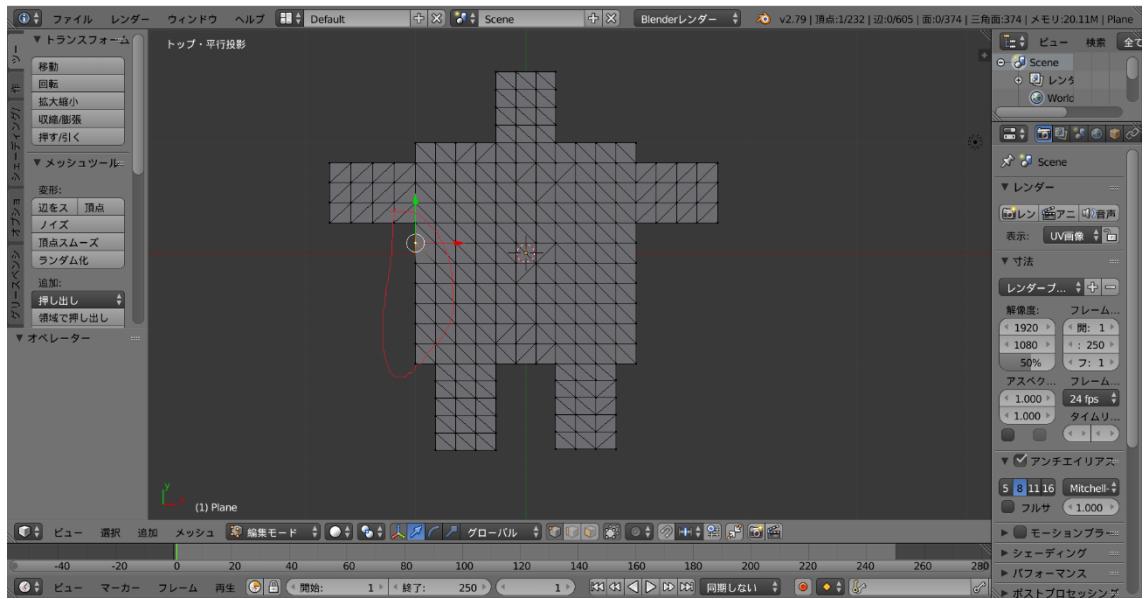


図 25:ある頂点を選択

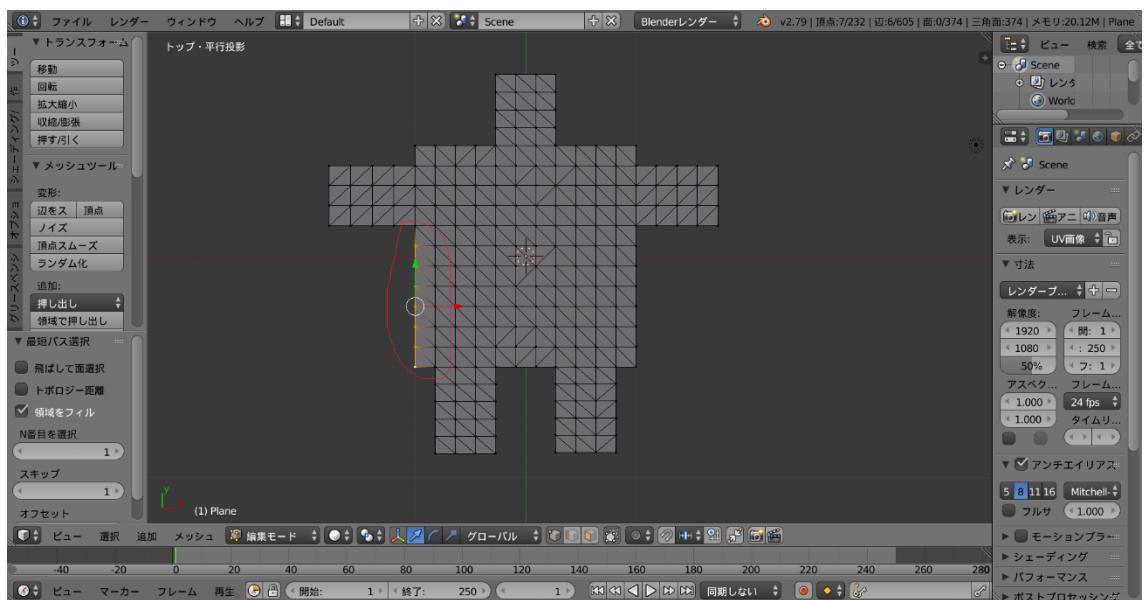


図 26:「Alt + Shift + 右クリック」で選択している図

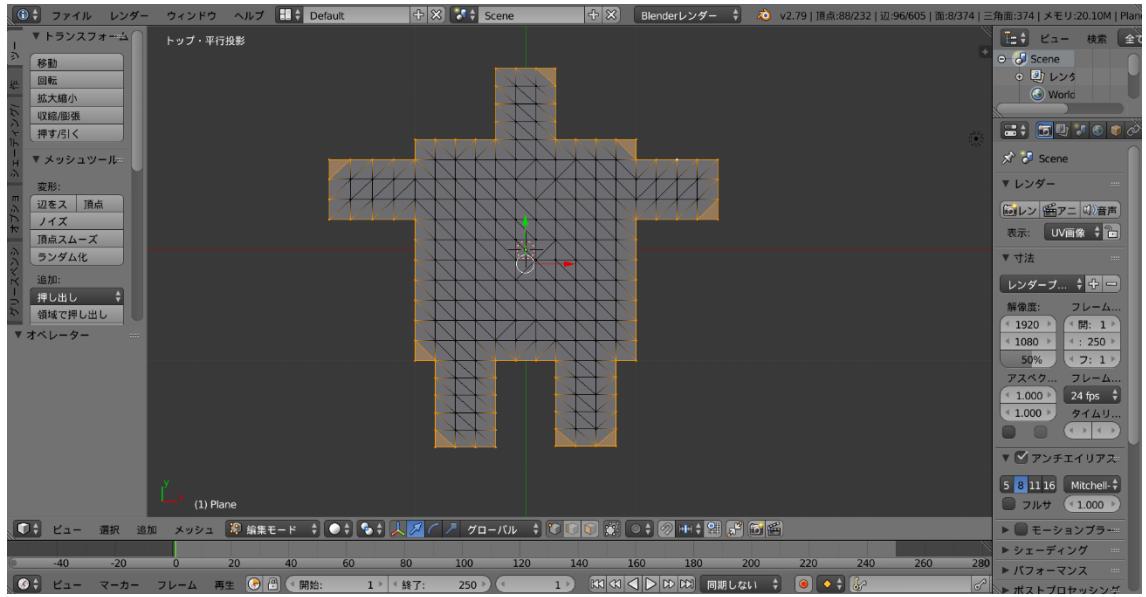


図 27:外周を選択した状態

外周を選択し終わったら、「**Ctrl + I**」キーで先ほど選択した頂点以外を選択する。そのまま「**Delete**」キーを押し、「頂点」を選択すると、内部の頂点が削除されて、外周のみ残る。しかし、外周上の頂点で構成された面は残ってしまうため、これらを削除する必要がある。「**Ctrl + tab**」キーで「メッシュ選択モード」のメニューを開き、「辺」を選択する。外周上ではない辺を「右クリック」で選択し、「**Delete**」キーを押して、「辺」を選択すると、外周のみが残る。これを繰り返していく、最終的に外周のみの状態にする。続いて、これを obj ファイルで出力する。この時のファイルを「**ファイル 2**」とする。出力の仕方、設定は転写前のメッシュを出力したときと同じである。出力が終わって元の画面に戻ったら、「**tab**」キーで「編集モード」に切り替える。

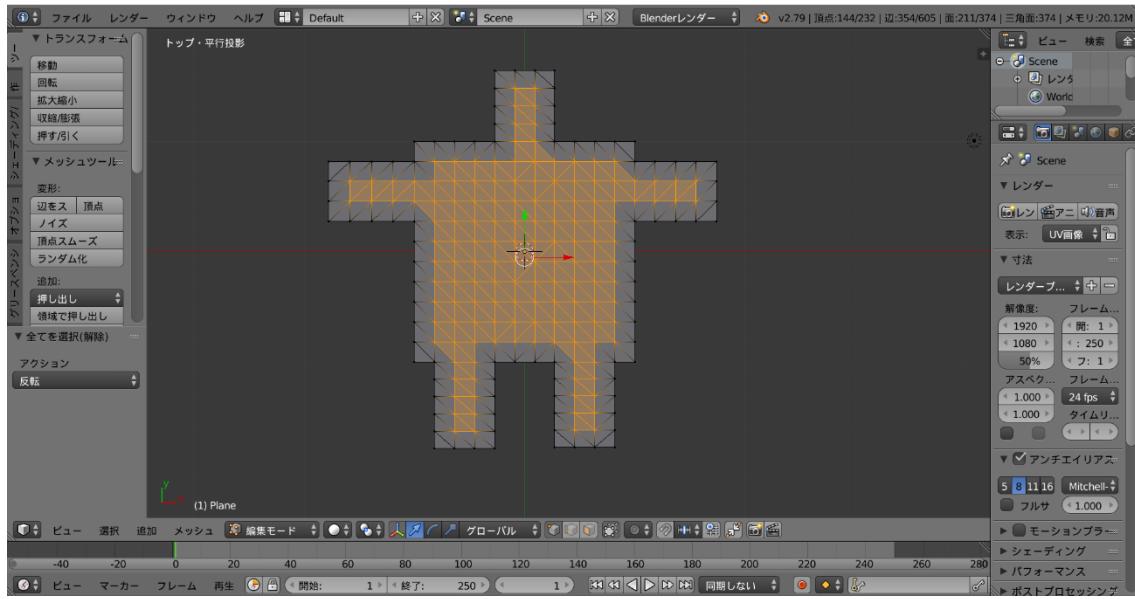


図 28:内部のメッシュを選択したとき

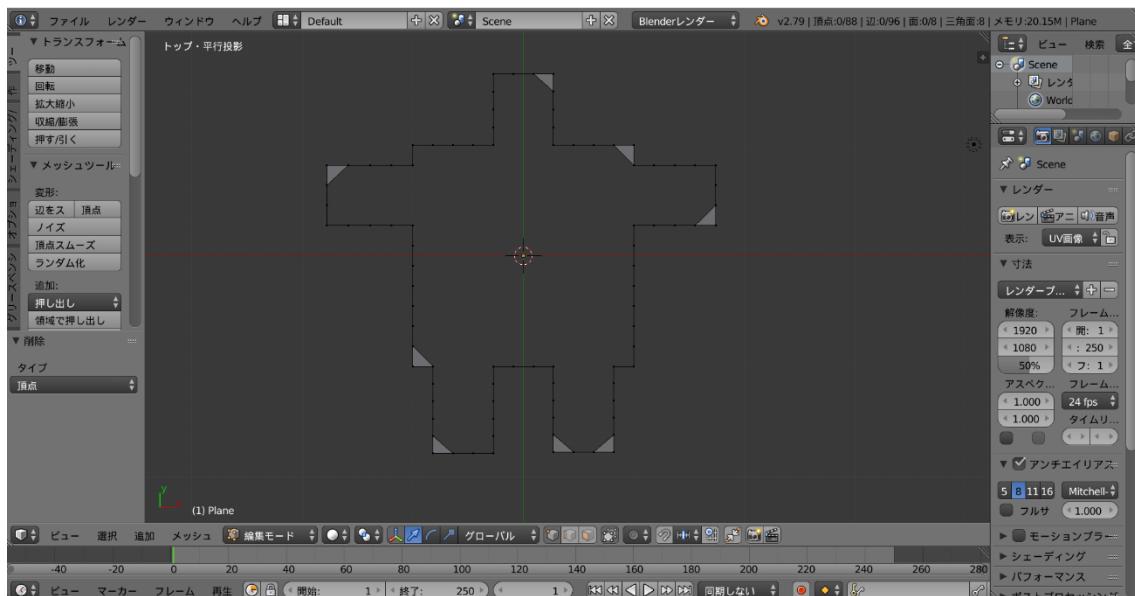


図 29:内部のメッシュを削除したとき

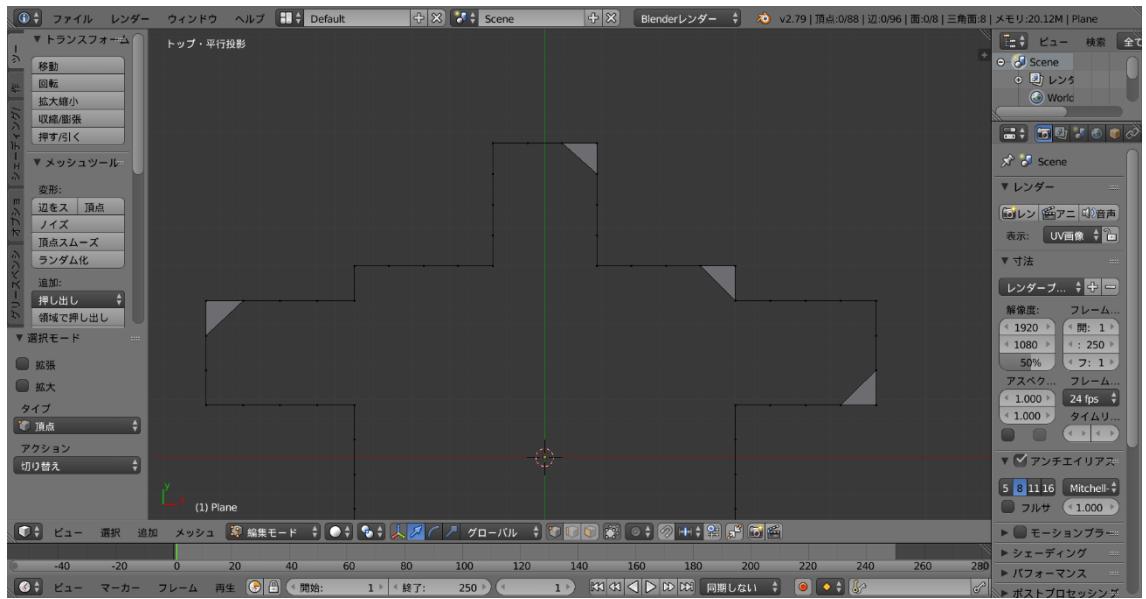


図 30:頂点選択モード

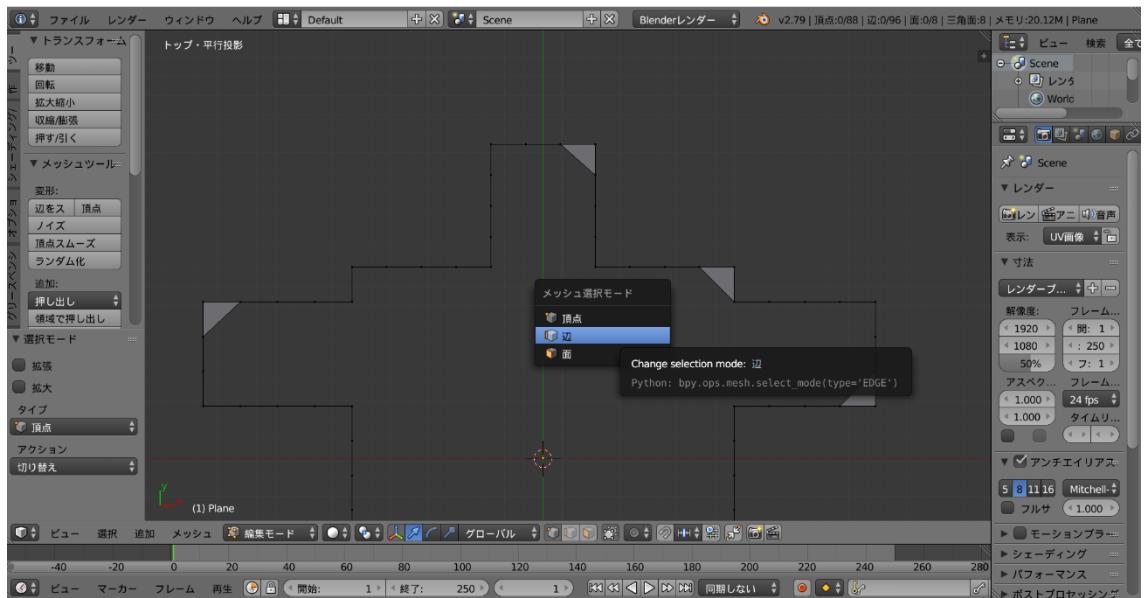


図 31:選択モードのメニュー

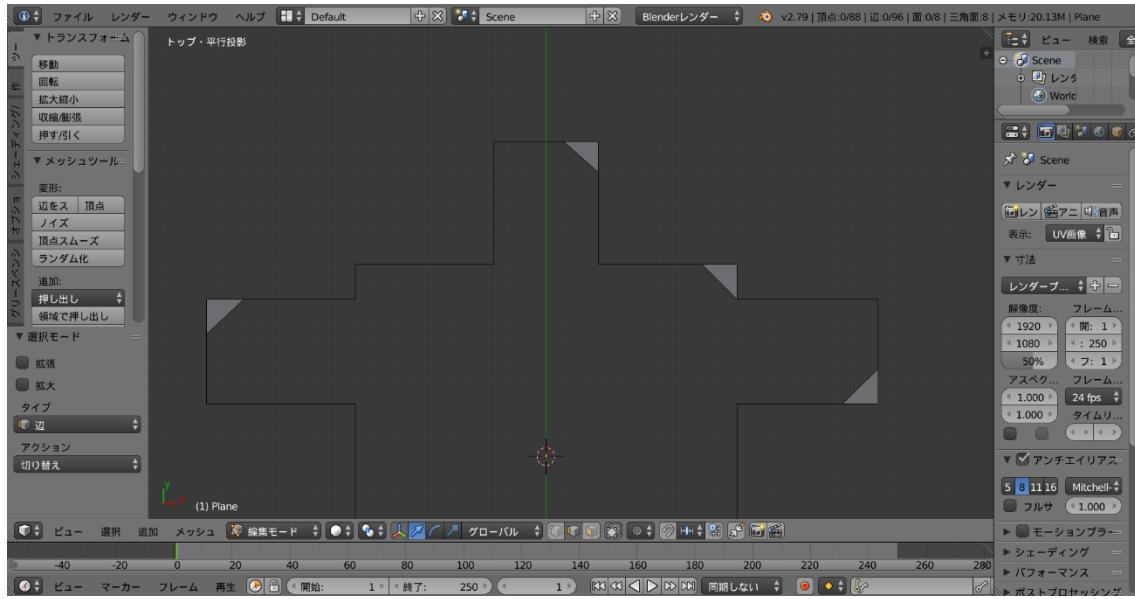


図 32:辺選択モード

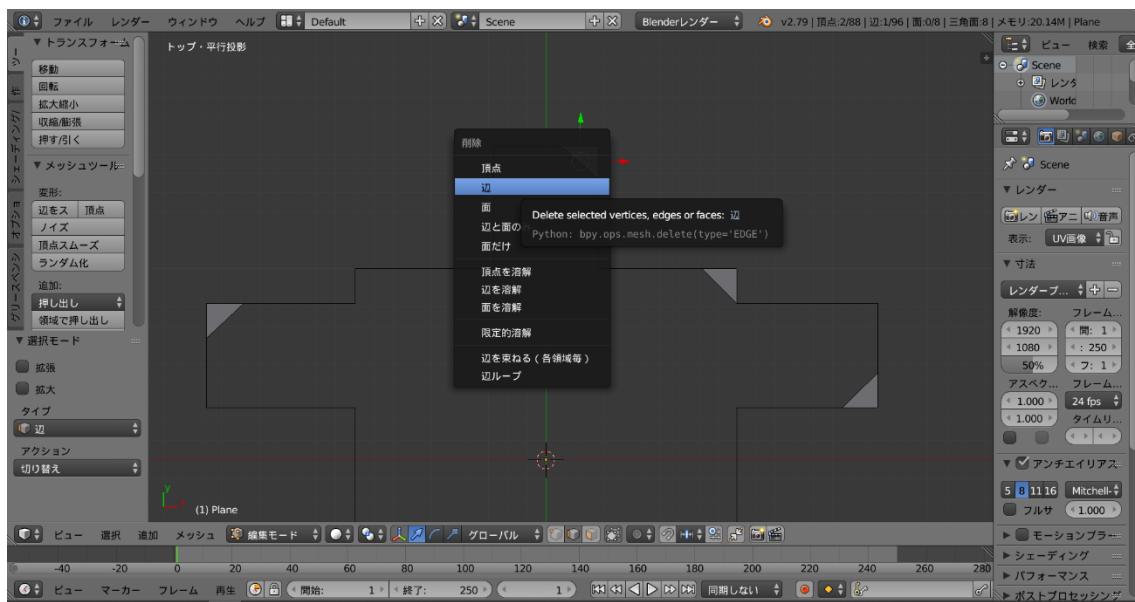


図 33:辺のみを削除している様子

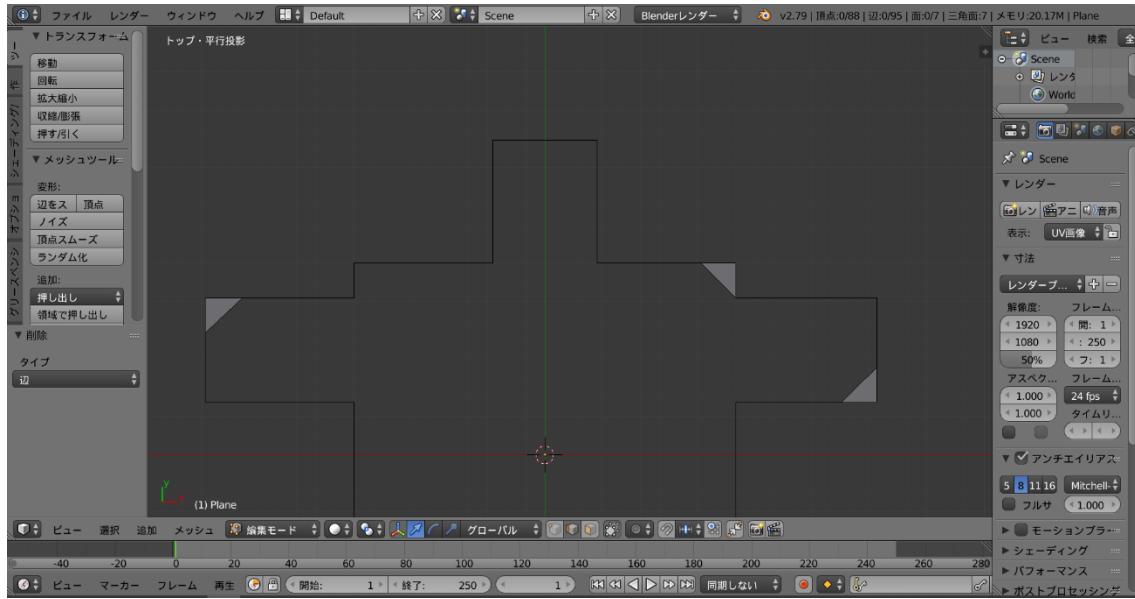


図 34: 内部に存在した辺を削除

最後に、転写先となる外周を作成する。「Ctrl + tab」キーで頂点を選択し、先ほどの外周上の頂点を選択、移動させて、変形させる。この時、元の外周の頂点数と、変形させた後の頂点数は変わらないまま変形させることに注意しておく。また、辺、頂点を消したり、つなぎなおしたりしないこと。これをしてしまうと、今まで出力したメッシュの情報、外周の情報が無駄になるので絶対にしないこと。変形が終わったら、また OBJ ファイルに出力すること(設定、方法などは今までと一緒)。この時のファイルを「ファイル 3」とする。

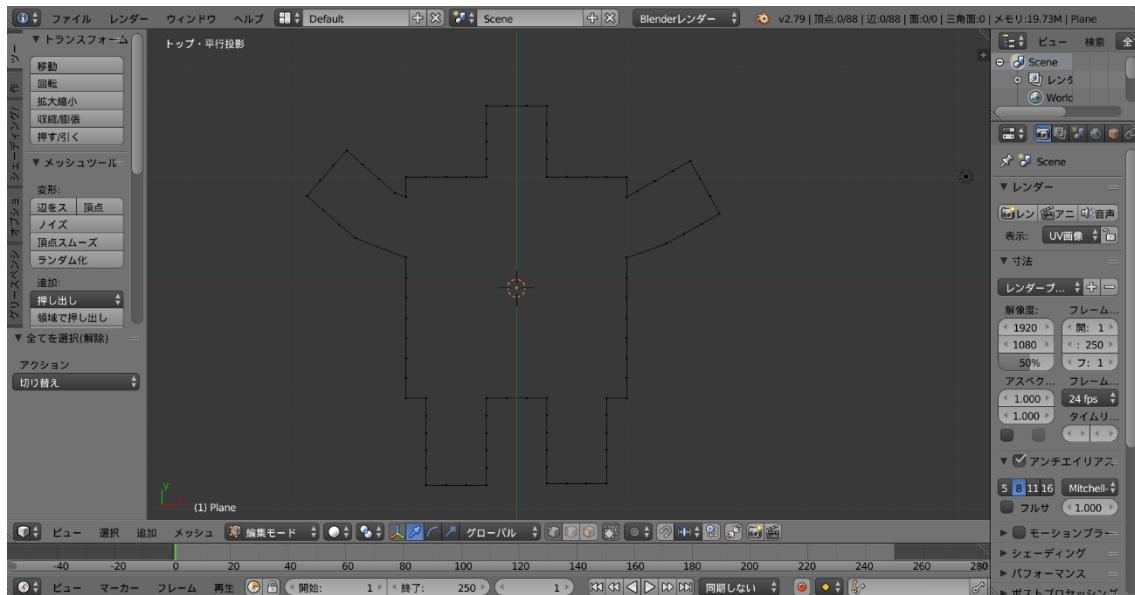


図 35: 外周を変形させた後の様子

以上で入力用ファイルを作成することに成功した。

続いてこれをどのように用いるかを説明する。

プロジェクトファイルを開き、「Input」フォルダを開き、「otameshi」というフォルダを開く。そこに先ほど作成したファイル1, ファイル2, ファイル3, を入れる。

次に Main.sln を開いてプロジェクトを開き、「main.cpp」ファイルを開く。

その中の変更可能なグローバル変数の部分に str1, str2, str3 が定義されている部分がある。

そこに先ほど作成したファイル1, 2, 3, を入力する。str1 にファイル1, str2 にファイル2, str3 にファイル3 を入力すること(ちゃんと読み込めない)。

この時、「Input」フォルダの中に新たなフォルダ作成して、

入力ファイルを入れていたとしてもパスを指定すれば読み込める。

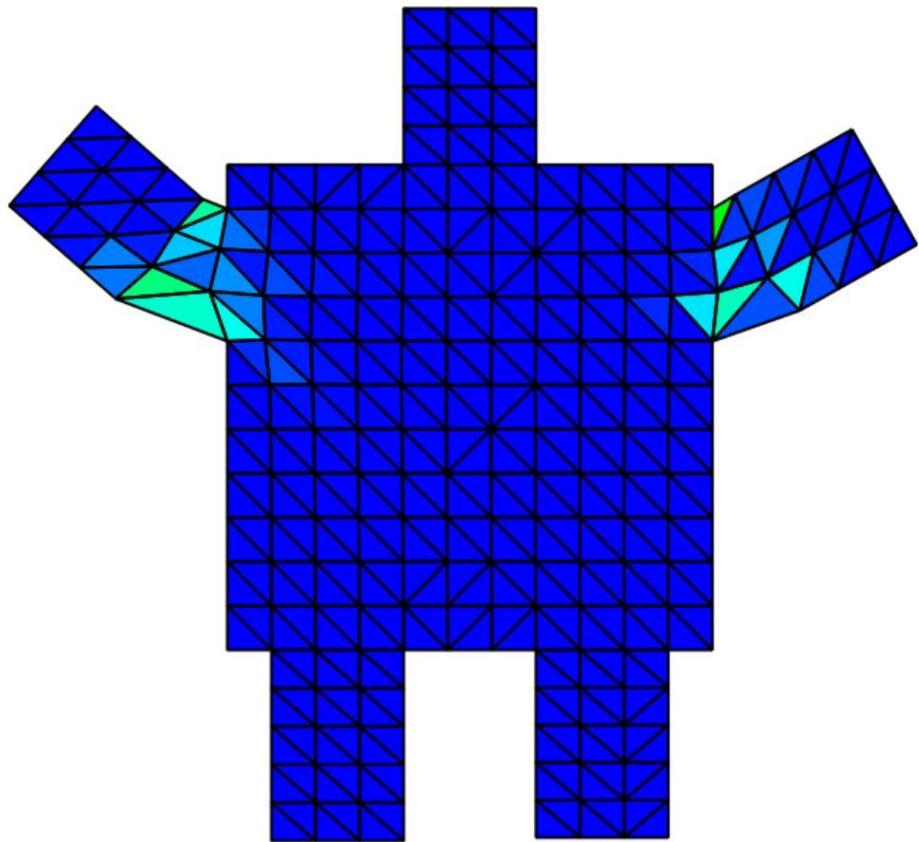
また、「TYPE = 0」, 「transform = "OBJ"」になっていることも確認すること。

```
-----変更可能なグローバル変数-----  
#define TYPE 0 //TYPEが1の時は長方形を自動生成して、それを転写させる。TYPEが0の時はファイルを読み込んでその情報を元に転写を行う  
#define VNUM 144 //40 //重みをつける要素の数  
#define WEIGHT 10000.0f //重みの数値  
int height = 20; //長方形の縦の長さ(分割数)  
int width = 10; //長方形の横の長さ(分割数)  
const char* str1 = "Input/otameshi/sample1.obj"; //MON(main).txt  
const char* str2 = "Input/otameshi/sample1-2.obj"; //sample(mon)2-2.txt  
const char* str3 = "Input/otameshi/sample1-3.obj"; //ArmBend1|3.obj  
bool weight = false; //重みつけをするかどうかのフラグ  
const char* transform = "OBJ";  
float rate = 0.0f; //門変形を行う際、どのくらい内側または外側に寄せるかの値  
float rad = 90.0f * (float)M_PI / 180.0f; //曲げるまたは回転させる際の角度  
float skew = 3.0f; //せん断変形で長方形の上底をどのくらい平行移動させるか  
float scale_x = 1.5f; //拡大縮小変形する際のx軸方向の割合  
float scale_y = 8.0f; //拡大縮小変形する際のy軸方向の割合  
const char* direction = "Y"; //どちらの方向に変形させるか  
float trans_x = 3.0f; //平行移動のx軸方向の値  
float trans_y = 1.5f; //平行移動のy軸方向の値  
float Cx = width / 2.0f; //回転の軸のx座標  
float Cy = height / 2.0f; //回転の軸のy座標  
const char* output1 = "Output/hit01-0.txt"; //メッシュの情報を出力(転写前の節点座標、要素を形成する節点番号など)  
const char* output2 = "Output/hit01-1.txt"; //メッシュの情報を出力(外周上の移動後の節点座標、節点番号など)  
const char* output3 = "Output/hit01-2.txt"; //メッシュの情報を出力(重みをつける要素の番号)
```

図 36:入力ファイルの名前、パスを入力する部分

ファイル名を指定したら、実行してみる。

すると、転写前のメッシュを転写先に反映させる処理が行われる。



Blender の使い方の参考 URL

- ・<https://blender-cg.net/>
- ・<https://www.cgradproject.com/tutorials/blender%E5%85%A5%E9%96%80/>
- ・<https://www.blender3d.biz/>

Blender で調べる際のキーワード

- ・頂点選択, 移動, 回転, 拡大
- ・Obj ファイルの出力
- ・細分化
- ・押し出し
- ・ループカット

・最後に

かなり急いで書いたので、不備やわからないことがあると思います！

その時は以下のメアドに連絡してください！できるだけ早急に返事します。

is0328ff@ed.ritsumei.ac.jp