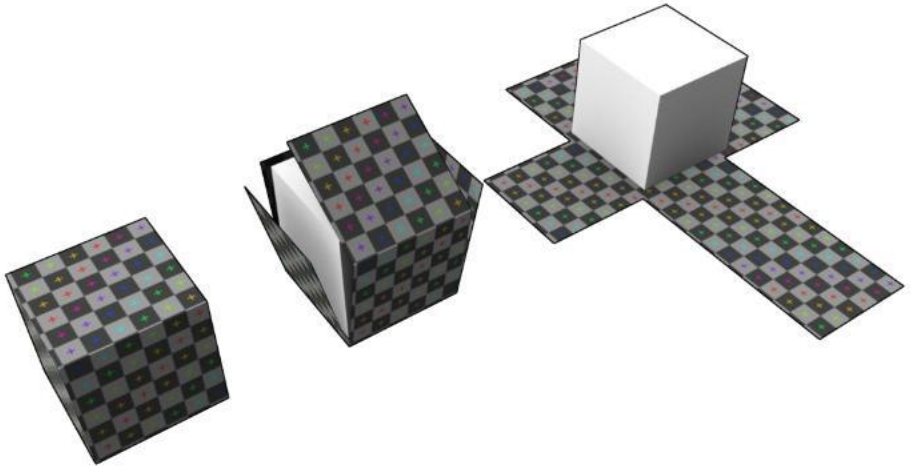
ШУТИС, Мэдээлэл Холбооны Технологийн Сургууль

F.CS209 Компьютерийн график

Лекц – Texture mapping

Боловсруулсан багш: Х.Хулан, Ч.Цэнд-Аюуш

2022 он

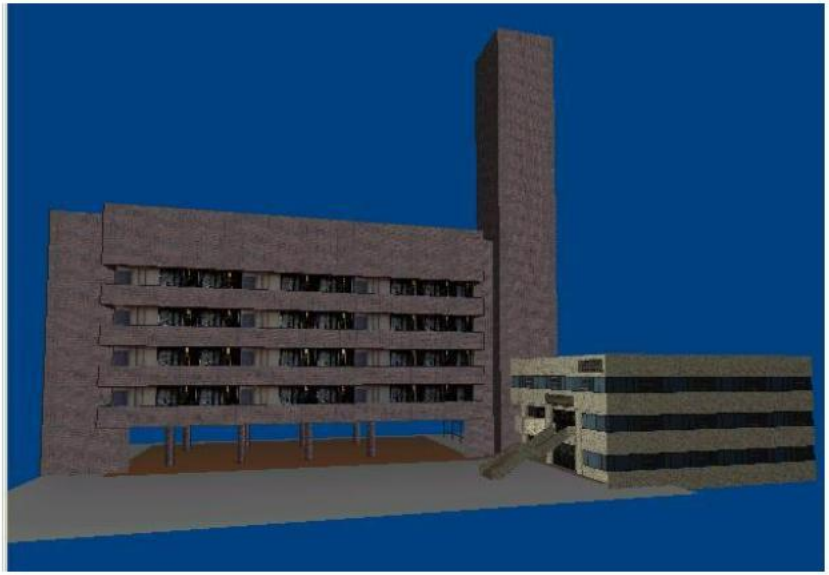


Texture Mapping

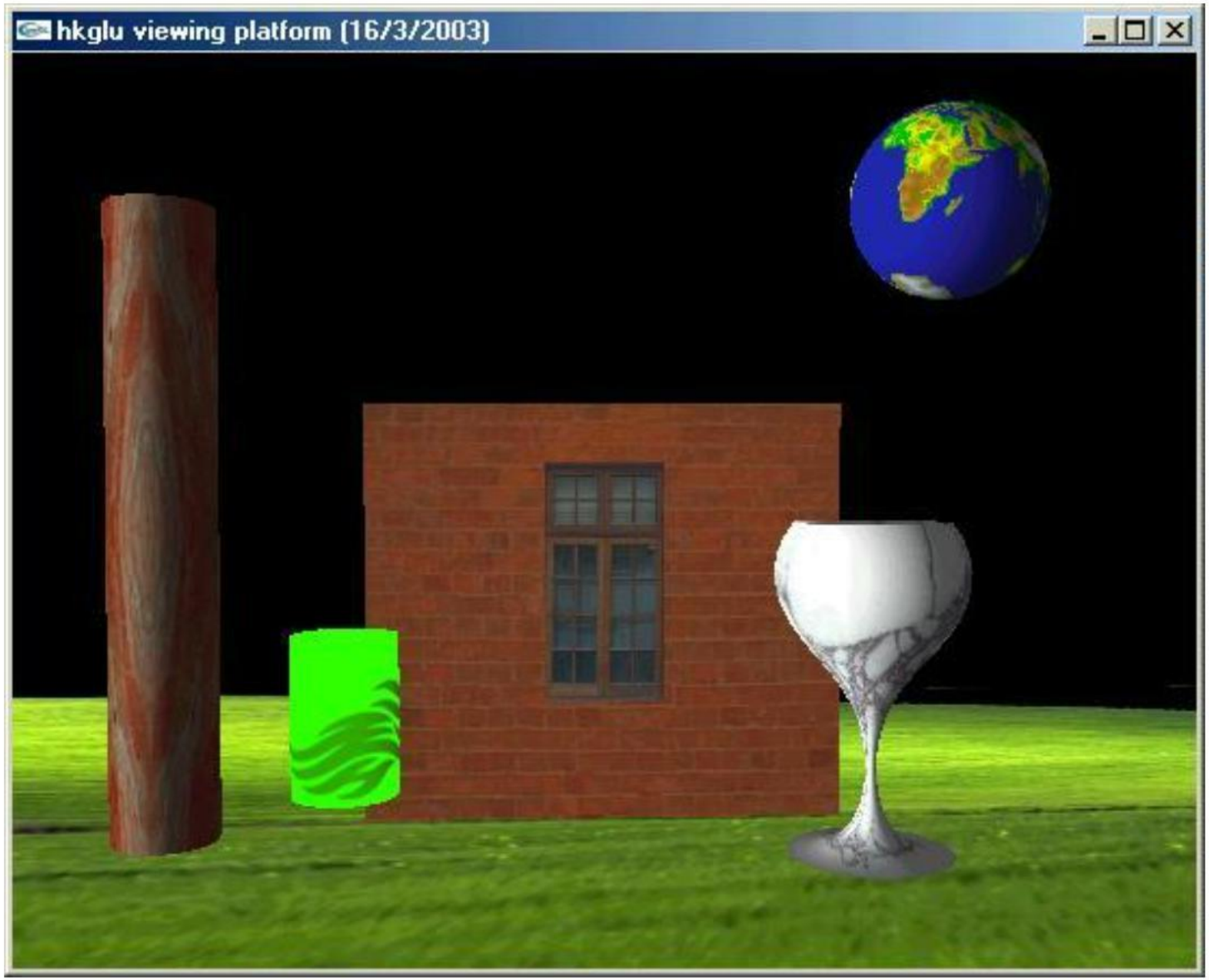
Гадаргууг боловсронгуй болгох нь

Texture Mapping

- Зургийг гадаргууд буулгах
  - Тэгш өнцөгт (Rectangles)
  - Олон өнцөгт (Polygons)
  - Цилиндр (Cylinders)
  - Бөмбөрцөг (Spheres)

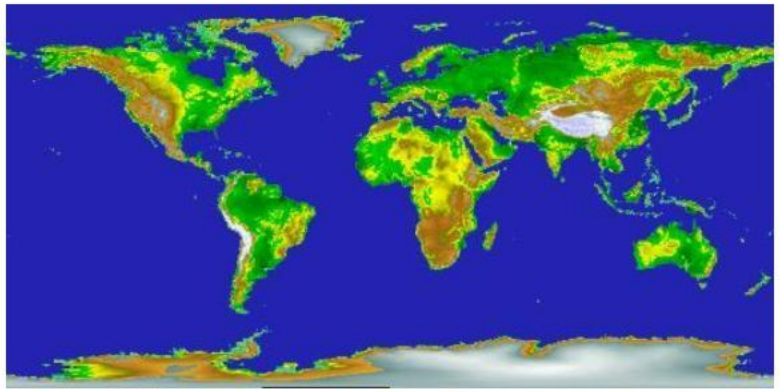


Haking

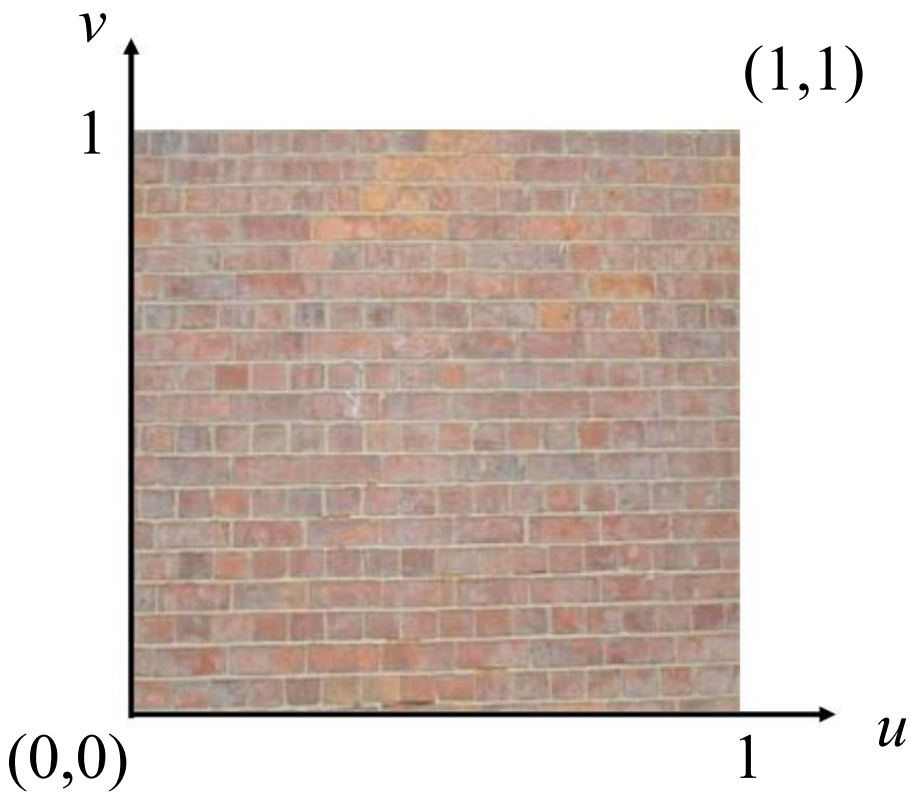


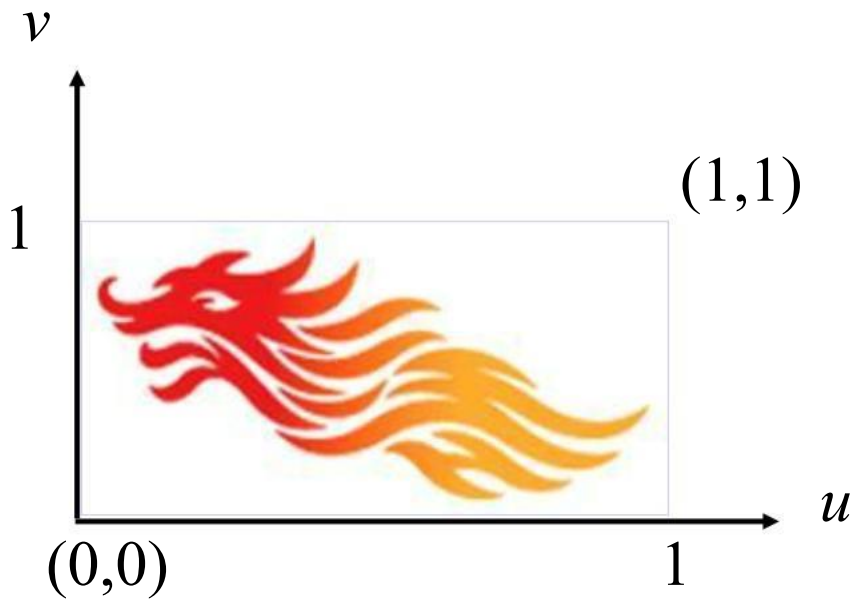
Mapping 1

Текстүр зургийн жишээ

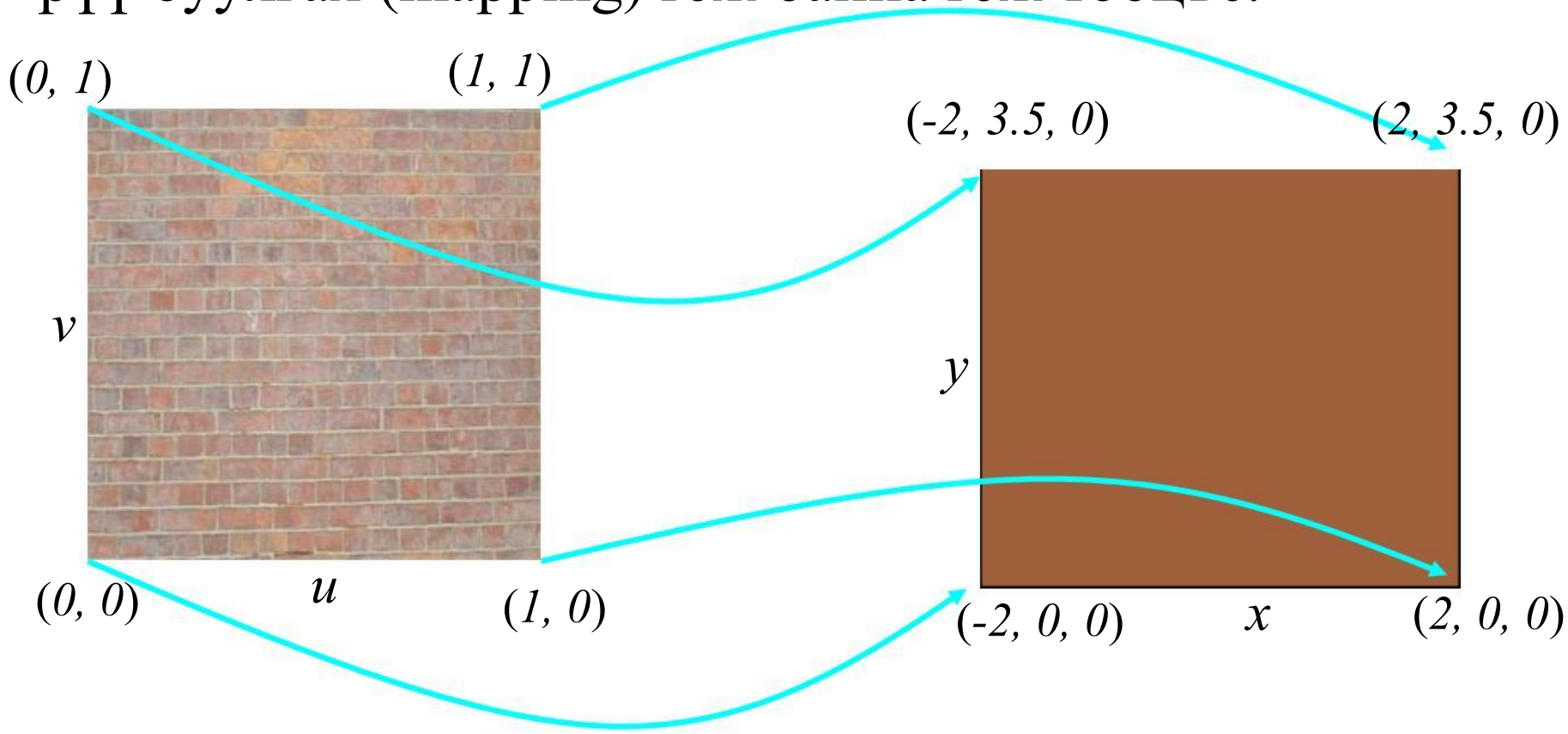


- 2-D текстүр зурагт хэвтээ тэнхлэгийг *u*, ба босоо тэнхлэгийг *v* гэж авч үзнэ. Зургийн зүүн доод булан нь координатын эх дээр байна (0, 0).
- Тэгш өнцөгт зураг ямар хэмжээтэй байхаас үл хамааран тэнхлэгийг масштаблан, баруун дээд буланг үргэлж (1, 1) байх ёстой.





- Тоосгон хананы зургийг бүхэлд нь тэгш өнцогт дүрс рүү буулгах (mapping) гэж байна гэж тооцъё.



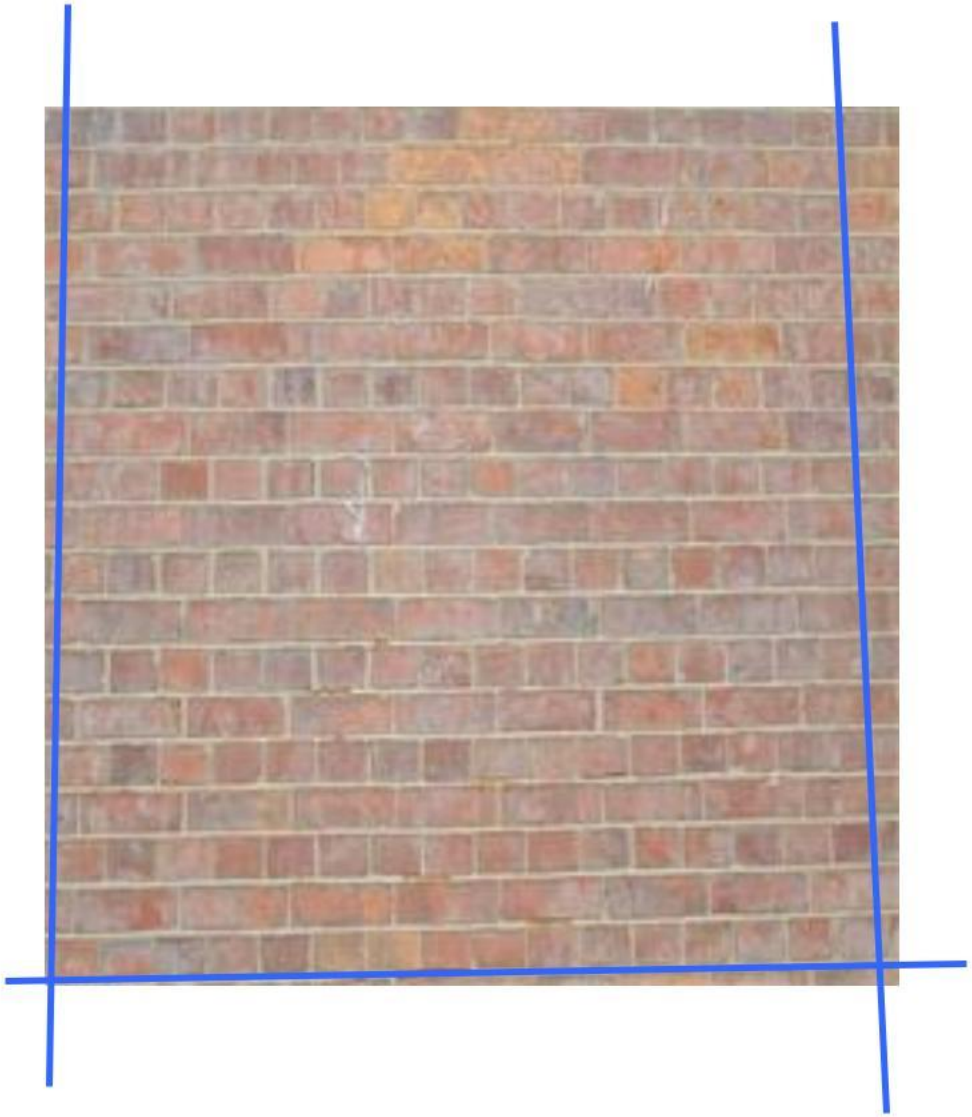
$0 \le u \le 1 \longrightarrow -2 \le x \le 2$

$0 \le v \le 1 \longrightarrow 0 \le y \le 3.5$

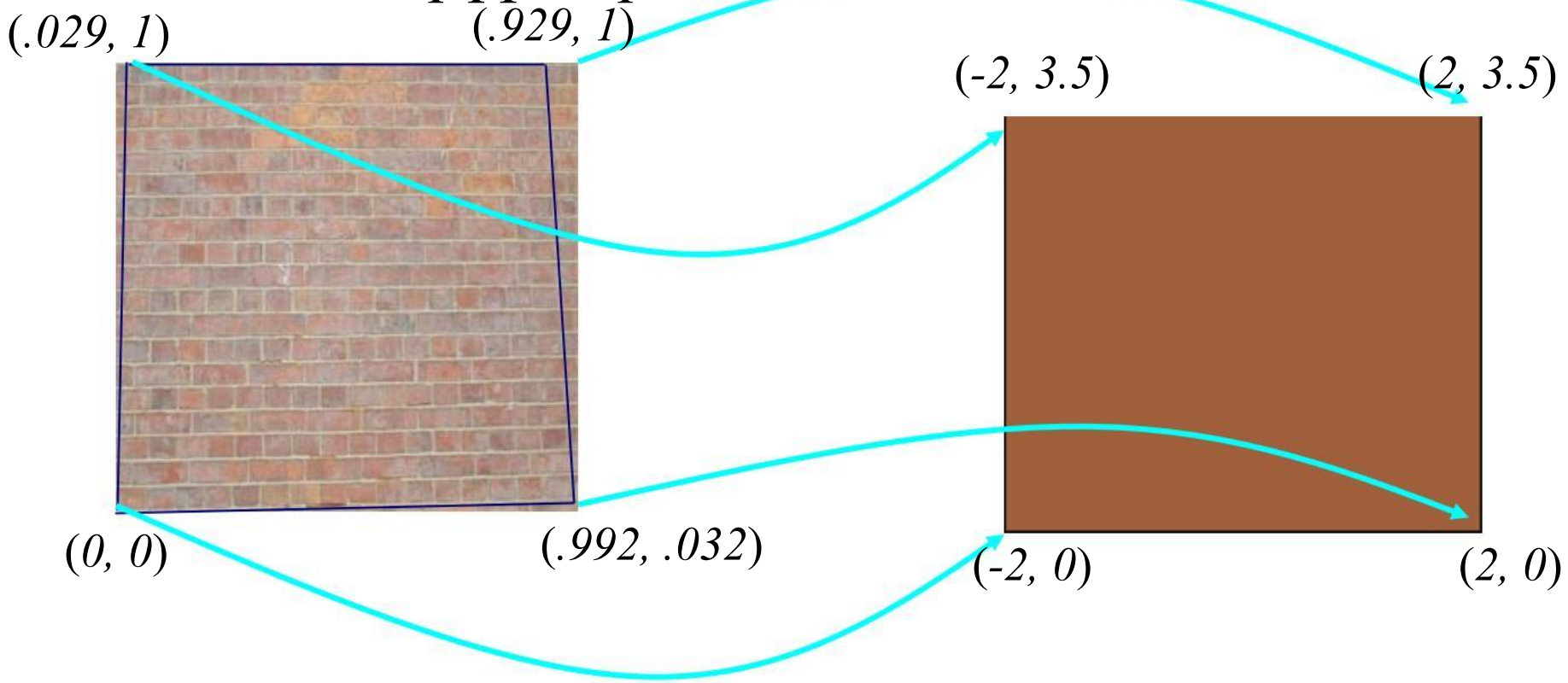
- Зураг ба тэгш өнцөгт хооронд орой, оройгоор нь mapping хийх код

```
glBegin( GL_QUADS);
        glTexCoord2f( 0., 1.);     glVertex3f( -2., 3.5, 0. );
        glTexCoord2f( 0., 0.);     glVertex3f( -2., 0.0, 0.);
        glTexCoord2f( 1., 0.);     glVertex3f(  2, 0.0, 0.);
        glTexCoord2f( 1., 1.);     glVertex3f(  2., 3.5, 0.);
glEnd();
```

- Зурган дээрх тоосго нь эгц хэвтээ болон босоо биш байх тохиолдол гардаг.



- Хурдан засах боломж нь хэсэгчилсэн текстүрийг тэгш өнцөгт рүү map хийх явдал юм



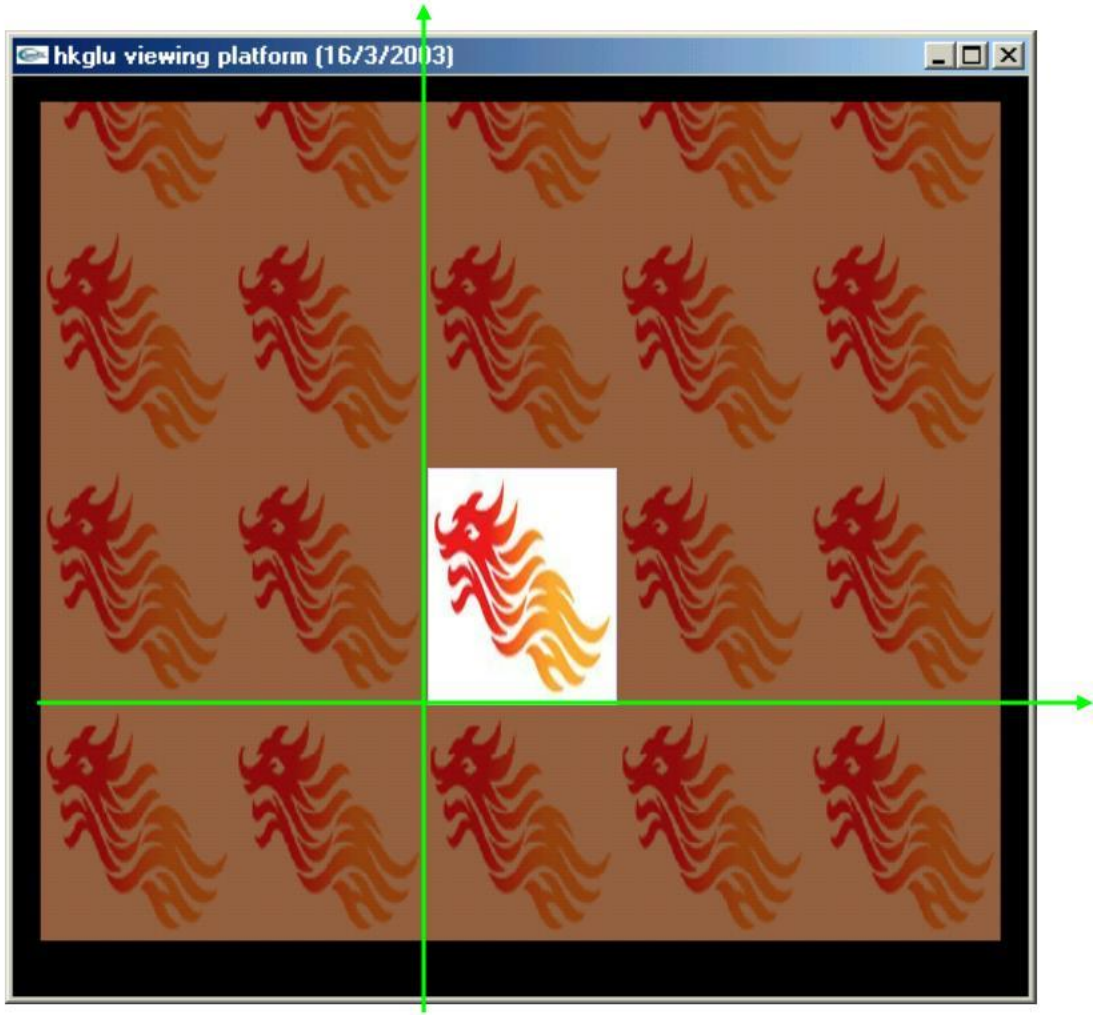

```
glBegin( GL_QUADS);
        glTexCoord2f( .029, 1.);      glVertex3f( -2., 3.5, 0. );
        glTexCoord2f( 0., 0.);        glVertex3f( -2., 0.0, 0.);
        glTexCoord2f( .992, .032);    glVertex3f(  2, 0.0, 0.);
        glTexCoord2f( .929, 1.);      glVertex3f(  2., 3.5, 0.);
glEnd();
```

- Өөрөөр хэлбэл текстур зургийн олон хувилбарыг тэгш өнцөгт дээр map хийж болно.

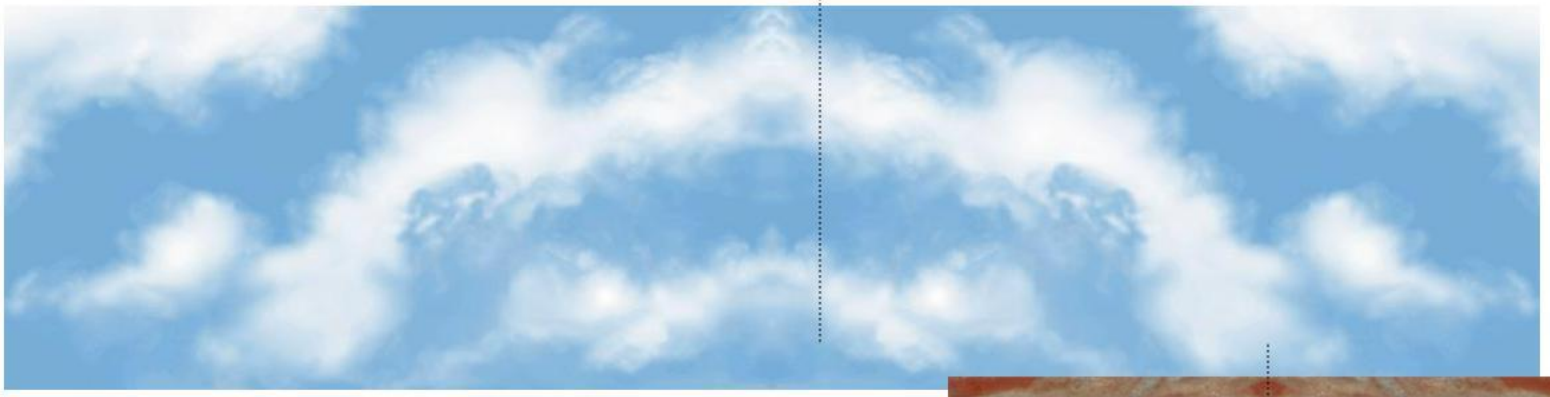

```
glBegin( GL_QUADS);
        glTexCoord2f( 0., 1.5);     glVertex3f( -2., 1.5, 0. );
        glTexCoord2f( 0., 0.);      glVertex3f( -2., 0.0, 0.);
        glTexCoord2f( 4., 0.);      glVertex3f(  2, 0.0, 0.);
        glTexCoord2f( 4., 1.5);     glVertex3f(  2., 1.5, 0.);
glEnd();
```

- Текстүр координат *u* ба *v* нь (0, 1) мужаас хэтэрсэн тохиолдолд зургийг дахин давтана.

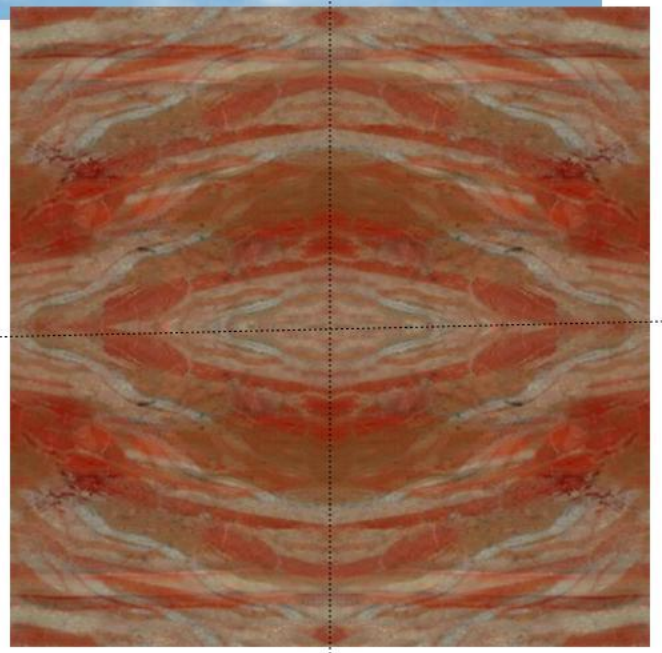
```
glBegin( GL_QUADS);
    glTexCoord2f( -2., 2.5);
    glVertex3f( -2., 3.5, 0. );
    glTexCoord2f( -2., -1.);
    glVertex3f( -2., 0.0, 0.);
    glTexCoord2f( 3., -1.);
    glVertex3f(  2, 0.0, 0.);
    glTexCoord2f( 3., 2.5);
    glVertex3f(  2., 3.5, 0.);
glEnd();
```



- Зургийг хэвтээ тэнхлэгийн дагуу давтах үед хүссэнээс өөр дүрс үүсэхээс зайлсхийхийн тулд зүүн тал нь баруун талын толин дүрс байхаар дүрс хурдан засварлаж өгөх явдал юм.



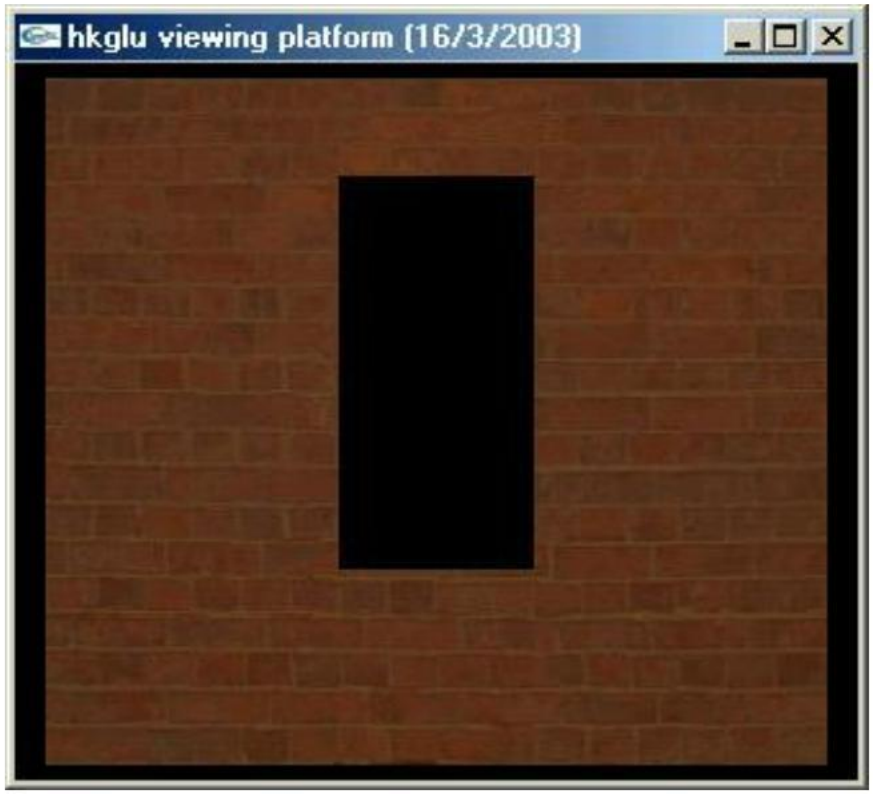
- Зургийг босоо чиглэлд давтах үед дүрсийн хүрээг устгахад ижил арга ашиглаж болно.



- Гүний туршилт (depth) дахь тойргийн алдаанаас болж хоёр гадаргууг бие биетэйгээ маш ойрхон зурсан тохиолдолд алдаа гарч болзошгүй.

Defect 1

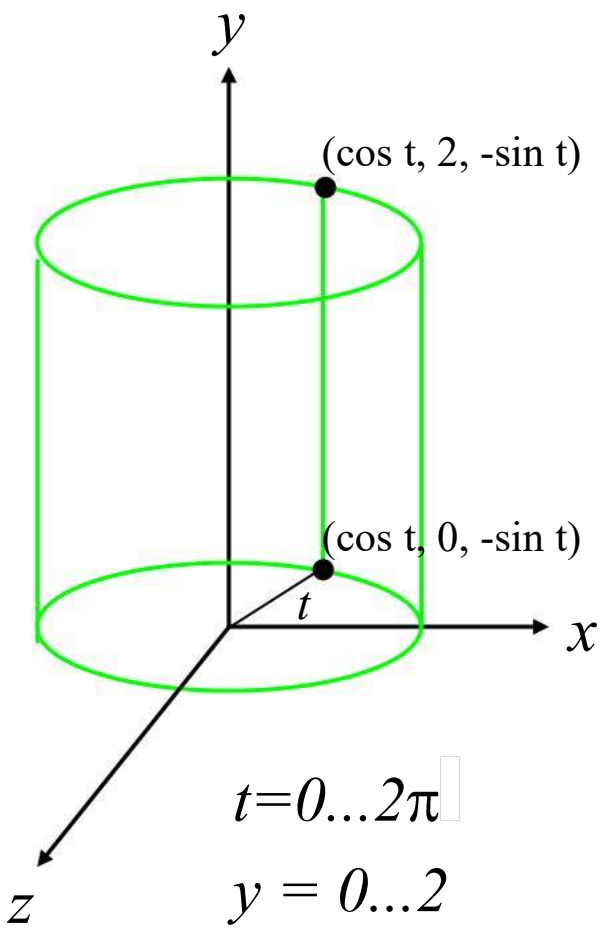
- Цонхны ард тоосгон хана зурах нь зөв шийдэл биш байна.



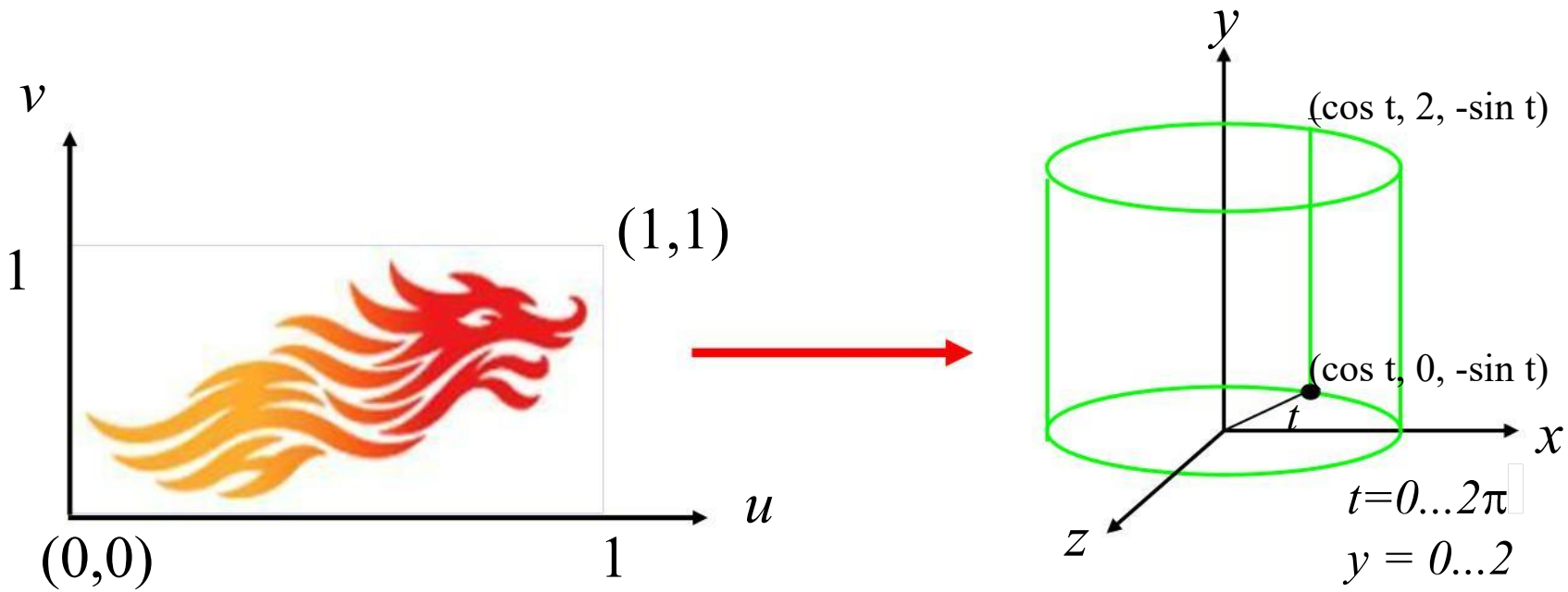
Window

- Дараах код нь x-z хавтгай дээр цилиндр зурах код юм.

```
glBegin( GL_QUAD_STRIP);
    t = 0.;
    dt = 2 * PI / nslice;
    for (j = 0; j <= nslice; ++j) {
        glVertex3f( cos( t), 2., -sin( t));
        glVertex3f( cos( t), 0., -sin( t));
        t = t + dt;
    }
glEnd();
```



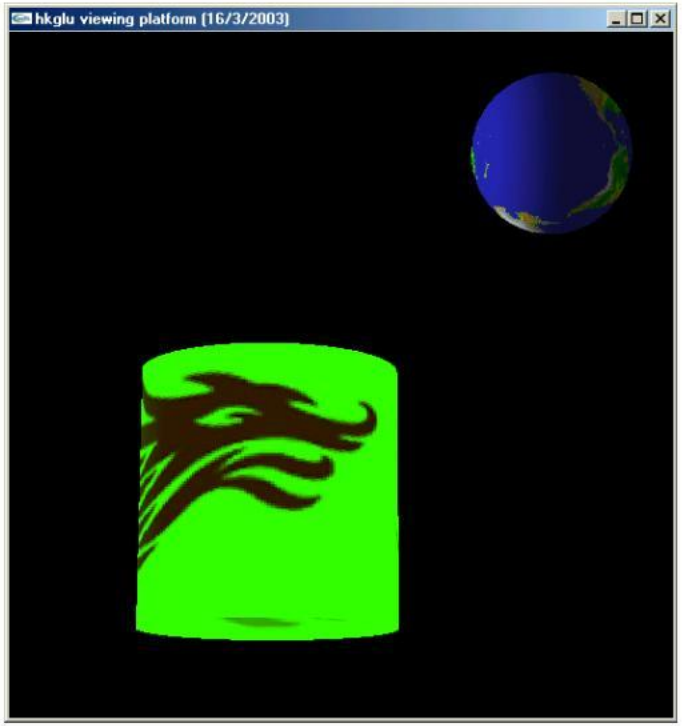
- Цилиндр дээр зургийг mapping хийх нь цилиндрийг цаасаар бүрсэнтэй адил юм.



$0 \le u \le 1 \longrightarrow 0 \le t \le 2\pi$

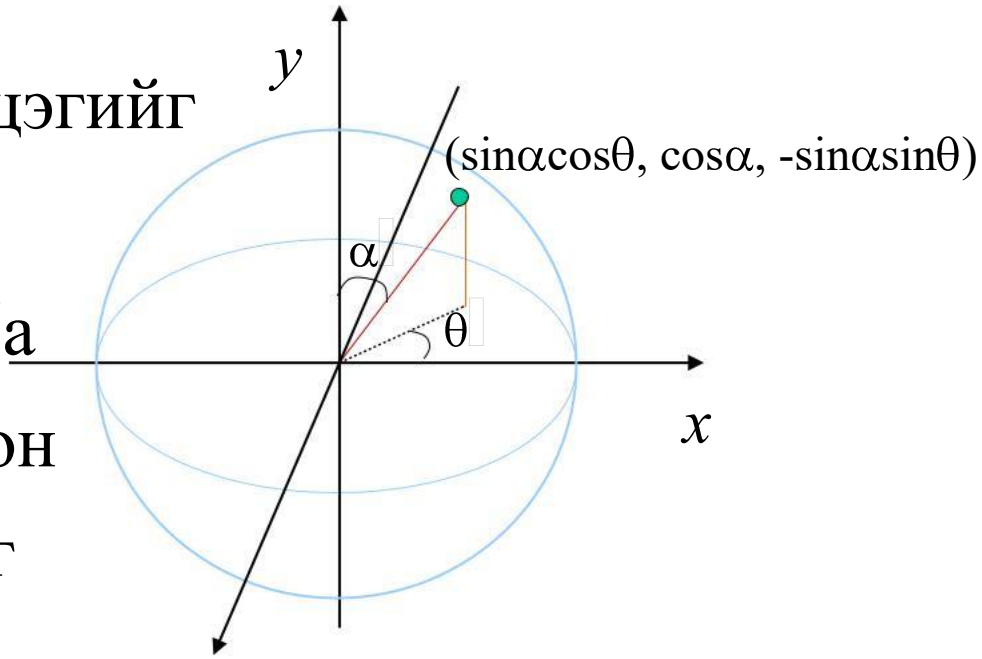
$0 \le v \le 1 \longrightarrow 0 \le y \le 2$

```
glBegin( GL_QUAD_STRIP);
    t = 0.;  dt = 2*PI/nslice;
    for (j = 0; j <= nslice; ++j) {
        glTexCoord2f( t/(2*PI), 1.);  glVertex3f( cos( t), 2., -sin( t));
        glTexCoord2f( t/(2*PI), 0.);  glVertex3f( cos( t), 0., -sin( t));
        t = t + dt;
    }
glEnd();
```



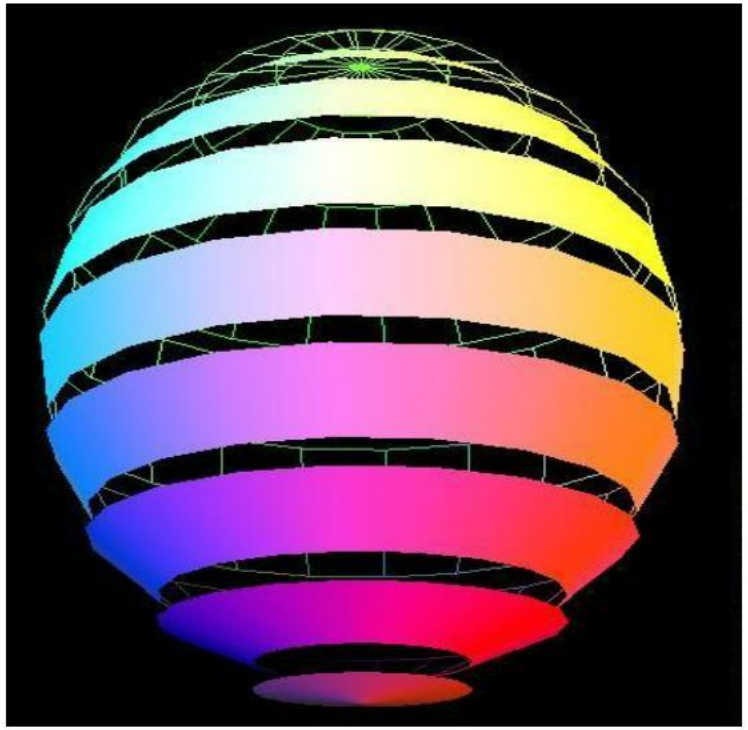
Lantern

- Туйлын координат
  Нэгж бөмбөрцөг дээрх цэгийг
  (α, θ) гэж тэмдэглэв.

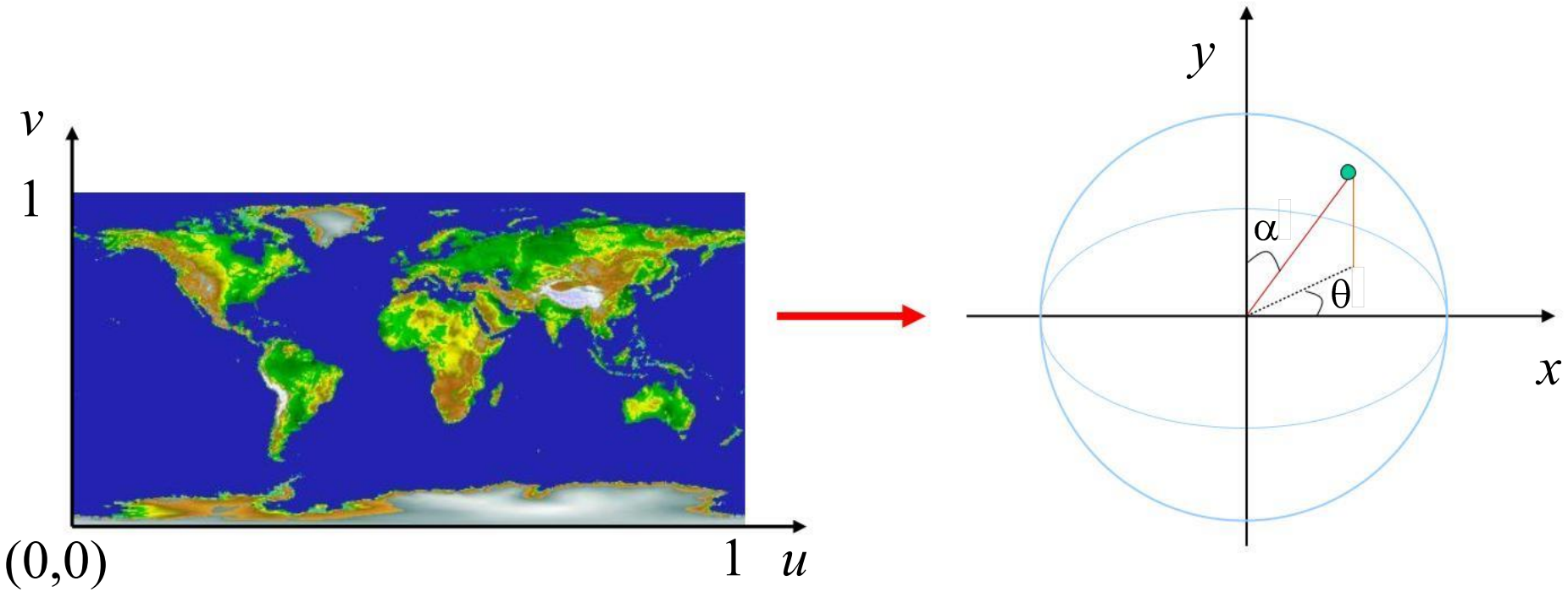


- $0 \le \alpha \le \pi$ нь у тэнхлэг ба коор.эхээс цэг рүү холбосон шулууны хоорондох өнцөг
- $0 \le \theta \le 2\pi$ х тэнхлэг ба x-z хавтгай дээрх шулуунуудын проекцийн хоорондох зайг цагийн зүүний эсрэг хэмжсэн өнцөг юм.
- Нэг бөмбөрцөгийн хувьд
  - у тэнхлэг дээрх цэгийн проекц нь $\cos(\alpha)$
  - х тэнхлэг дээрх цэгийн проекц нь $\sin(\alpha)\cos(\theta)$
  - z тэнхлэг дээрх цэгийн проекц нь $-\sin(\alpha)\sin(\theta)$

```
for each stack, top to bottom
{
     glBegin( GL_QUAD_STRIP);
     for each slice in the stack.
    {
         specify two end points

     }
    glEnd();
}
```



- газрын зургийг бөмбөрцөг рүү mapping хийх

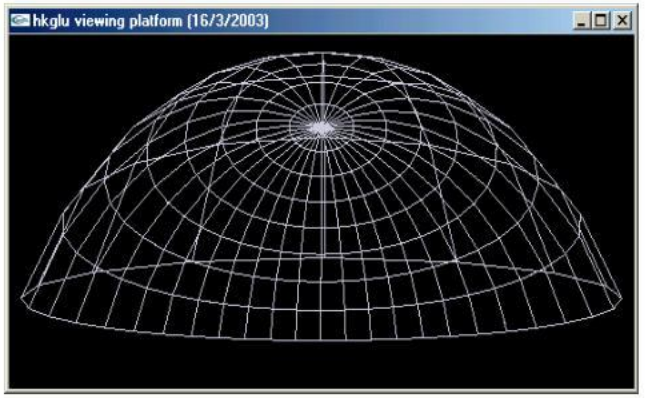


$0 \le u \le 1 \longrightarrow 0 \le \theta \le 2\pi$

$1 \ge v \ge 0 \longrightarrow 0 \le \alpha \le \pi$

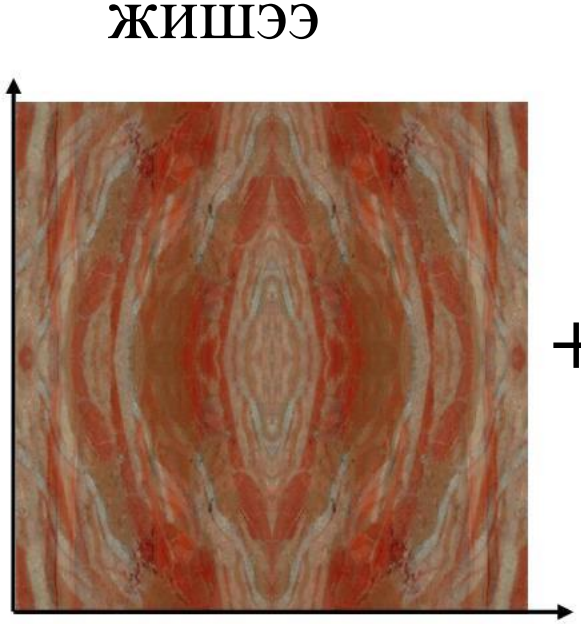
Бөмбөрцөгийн гадаргуу $(\theta/(2\pi), 1- \alpha/\pi)$ дээрх цэгийн текстүр координат $(\sin\alpha \cos\theta, \cos\alpha, -\sin\alpha \sin\theta)$, байна.

- Дараахь демо нь бөмбөрцгийн дотоод гадаргуу дээр тэнгэрийн зургийг mapping хийхийг харуулсан.

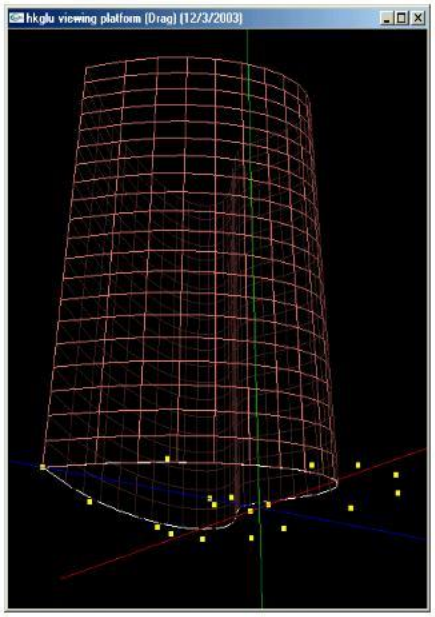


Sky

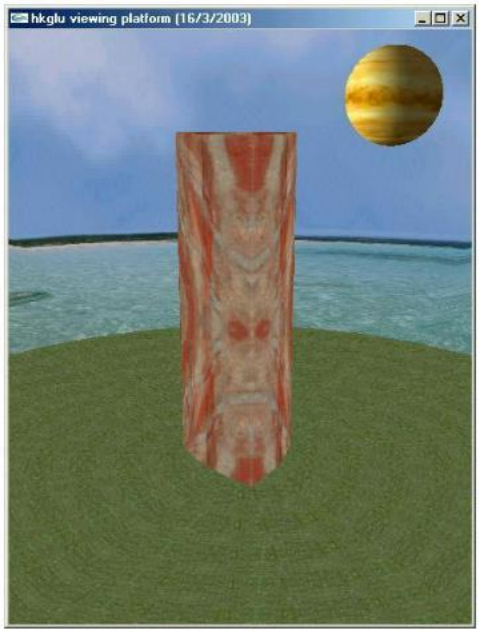
- extrusion гадаргуу руу текстүр mapping хийсэн жишээ



+



=



$0 \le u \le 1 \longrightarrow 0 \le t \le 1$

$0 \le v \le 1 \longrightarrow 0 \le y \le height$

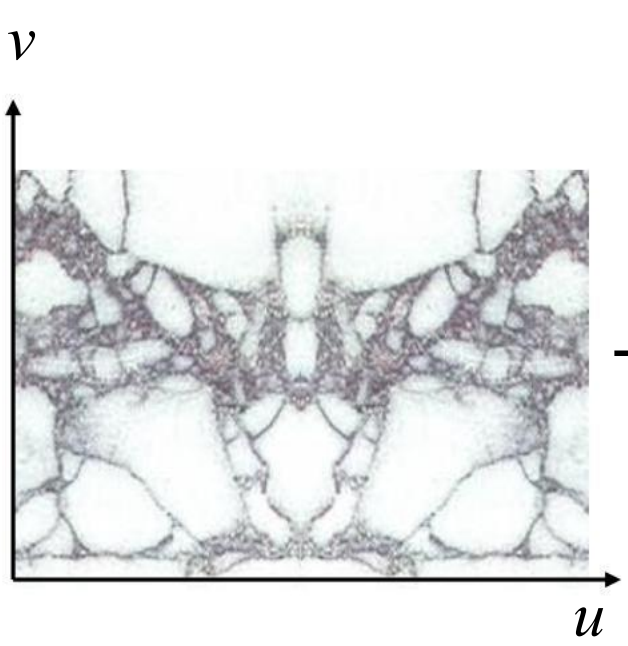
Love Pillar

Bezier муруй дээрх цэгийн байрлалыг *t* параметрээр тодорхойлно. *t* нь 0 –ээс 1 болж өөрчлөгдөхөд цэг нэг төгсгөлөөс нөгөө төгсгөл рүү шилжинэ.

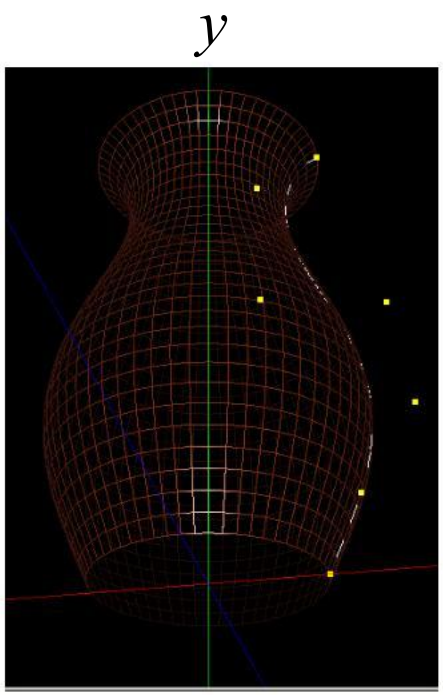
Тектүр координатыг Mapping хийх

```
for (int i = 1;  i <=nstack;  ++i) {
        y1 = (i-1) * height / nstack;       //шатны өндөр
        y2 = i * height / nstack;           //таазны өндөр

       // Стекийг quad_strip-ээр зурах
        glBegin( GL_QUAD_STRIP);

              for (int j = 0;  j <= nSegment;  ++j) {
                  t = (float) j / nSegment;
                  < t-ээс муруй дээрх цэгийг (xc, zc) тооцоолох >

                  glTexCoord2f( t, y2/height);
                  glVertex3f( xc, y2, zc);

                  glTexCoord2f( t, y1/height);
                  glVertex3f( xc, y1, zc);
              }
         glEnd();
}
```

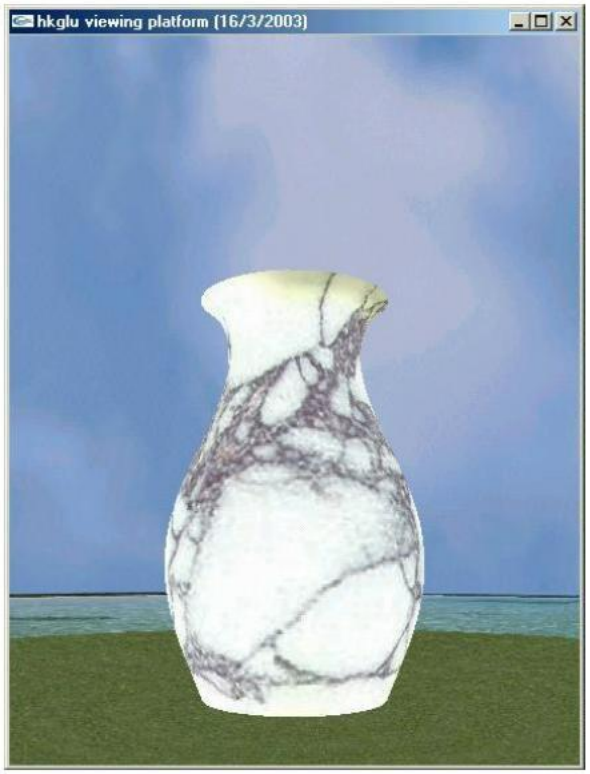
эргүүлэгтэй гадаргуу дээр текстүр map хийх жишээ



+



=



$0 \le u \le 1 \longrightarrow 0 \le \theta \le 2\pi$

$0 \le v \le 1 \longrightarrow 0 \le t \le 1$

Marble Vase

Bezier муруй дээрх цэгийн байрлалыг t параметрээр тодорхойлно. t нь 0 –ээс 1 болж өөрчлөгдөхөд цэг нэг төгсгөлөөс нөгөө төгсгөл рүү шилжинэ.

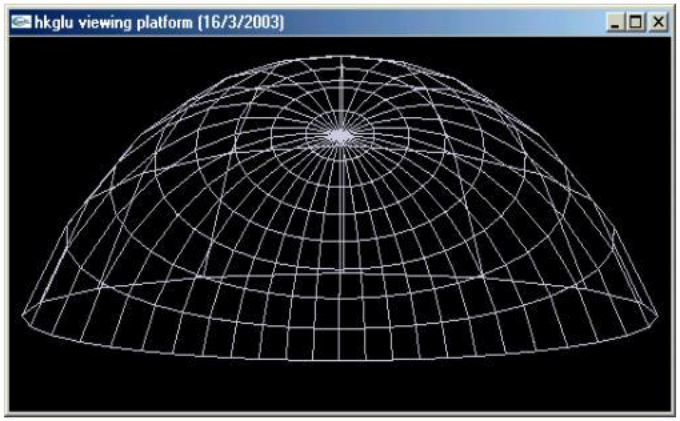
Програмын үр дүн

```
for (int i = 1;  i <=nstack;  ++i) {



    glBegin( GL_QUAD_STRIP);
        for (int j = 0;  j <= nSegment;  ++j)  {
            theta = j * 2 * PI / nSegment;
            < (xa, za) = эргүүлэх (x1, 0) тета радианаар>
            < (xb, zb) = (x2, 0) тета радианаар эргүүлнэ >
            glTexCoord2f( theta/(2*PI), y2/height);
            glVertex3f( xb, y2, zb);
            glTexCoord2f( theta/(2*PI), y1/height);
            glVertex3f( xa, y1, za);
        }
    glEnd();
}
```

Барзгар гадаргууг харуулах үйл явц нь төвөгтэй байдаг. Дэлгэц дээр харуулсан жишээнд Backward mapping харуулсан.

Backward mapping

Орой тус бүрийн бүтэцтэй координатыг тодорхойлох замаар зураглал хийх.



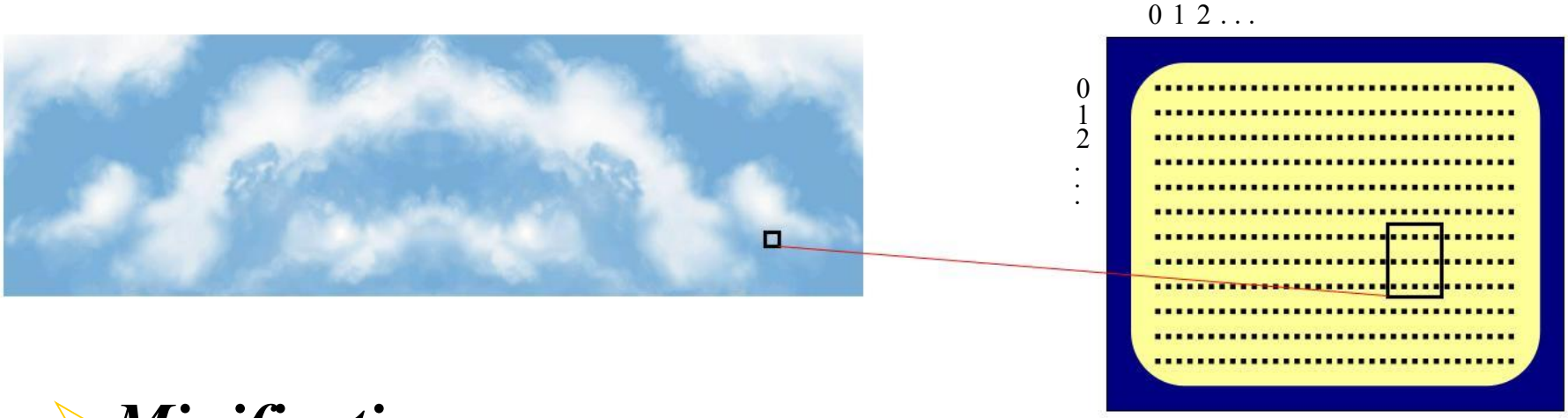
Projection

0 1 2 . . .

0
1
2
.
.

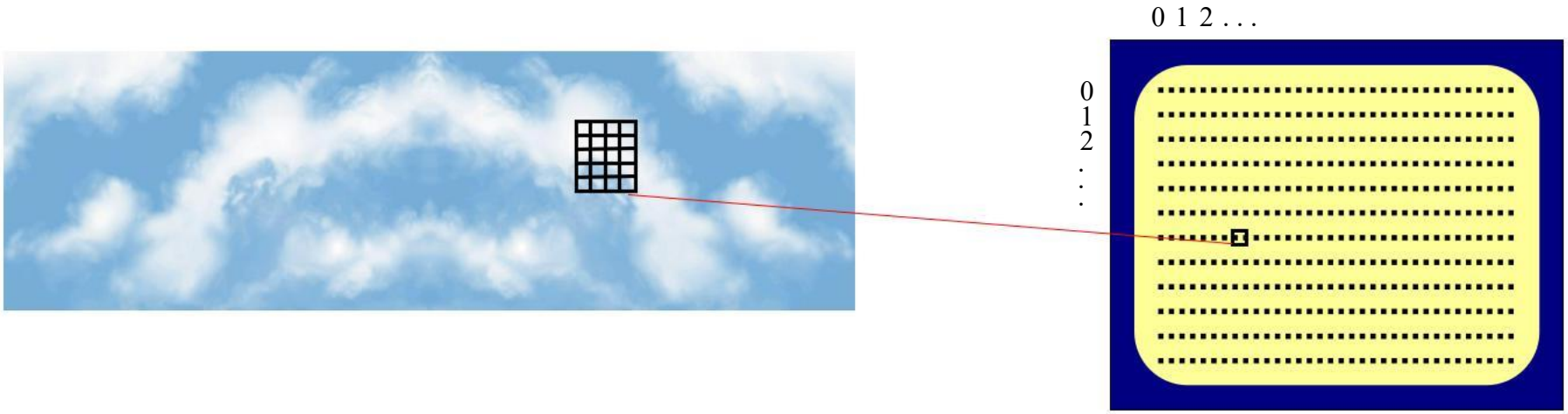
- ***Magnification***

  texel дэлгэц дээрх олон пикселийг хамарч болно

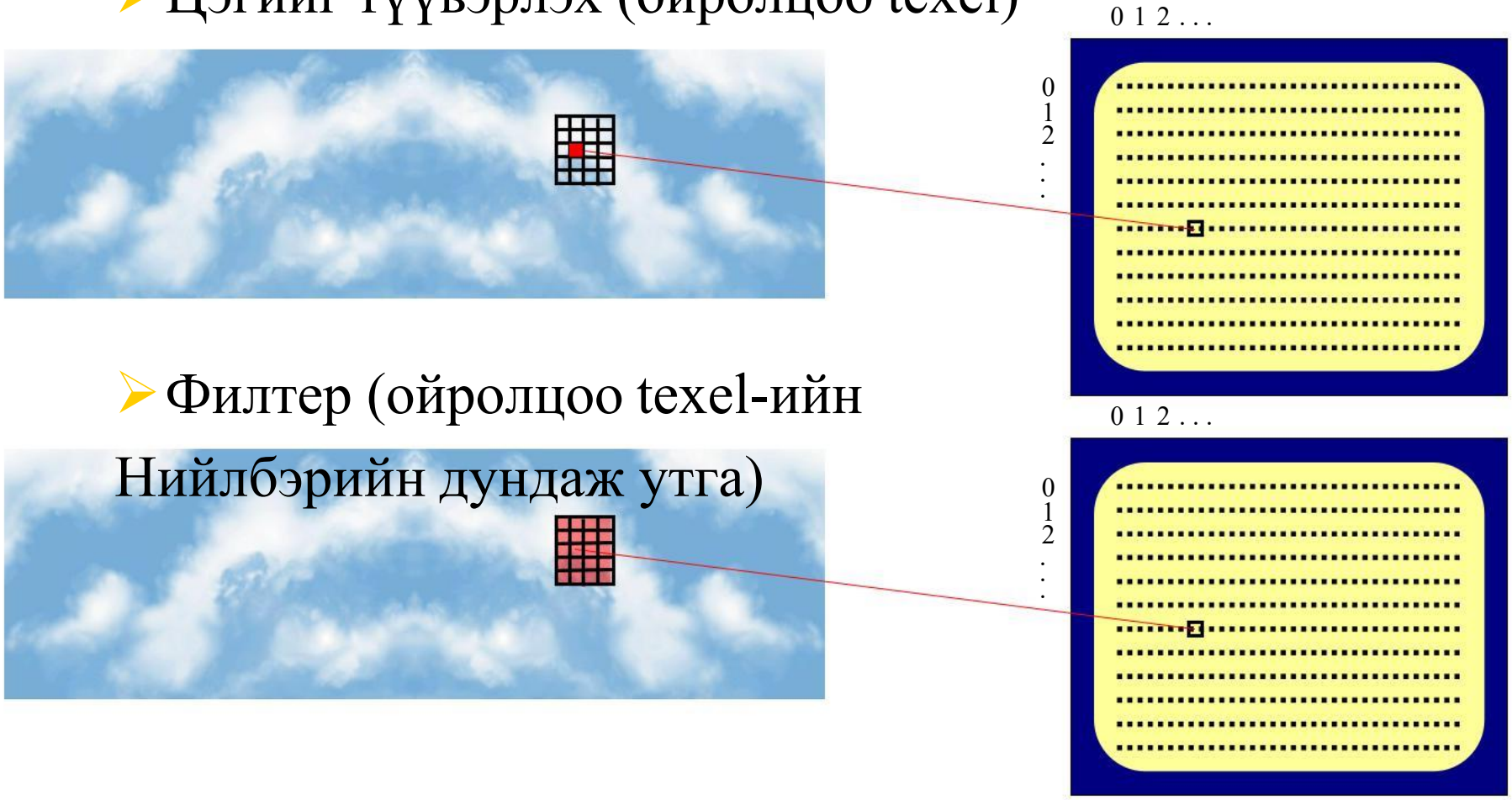


- ***Minification***

  Олон texels дэлгэц дээрх нэг пикселийг хамарч болно



- Дараах тохиолдолд хоёуланд нь пикселийн утгыг тооцолоход стандарт зургийн боловсруулалтын аргыг ашигладаг.
  - Цэгийг түүвэрлэх (ойролцоо texel)



  - Филтер (ойролцоо texel-ийн Нийлбэрийн дундаж утга)

- *Mipmapping* нь текстүр гадаргууг харуулах чанар, үр ашгийг дээшлүүлэх арга техник юм.

- Жижиг хэмжээтэй текстүр зургийг эх зургаас нь үүсгэдэг. Рендерлэх явцад texel хэмжээ бүхэлдээ сонгосон зургийн пикселийн хэмжээтэй ойролцоо байх зургийг сонгоно.

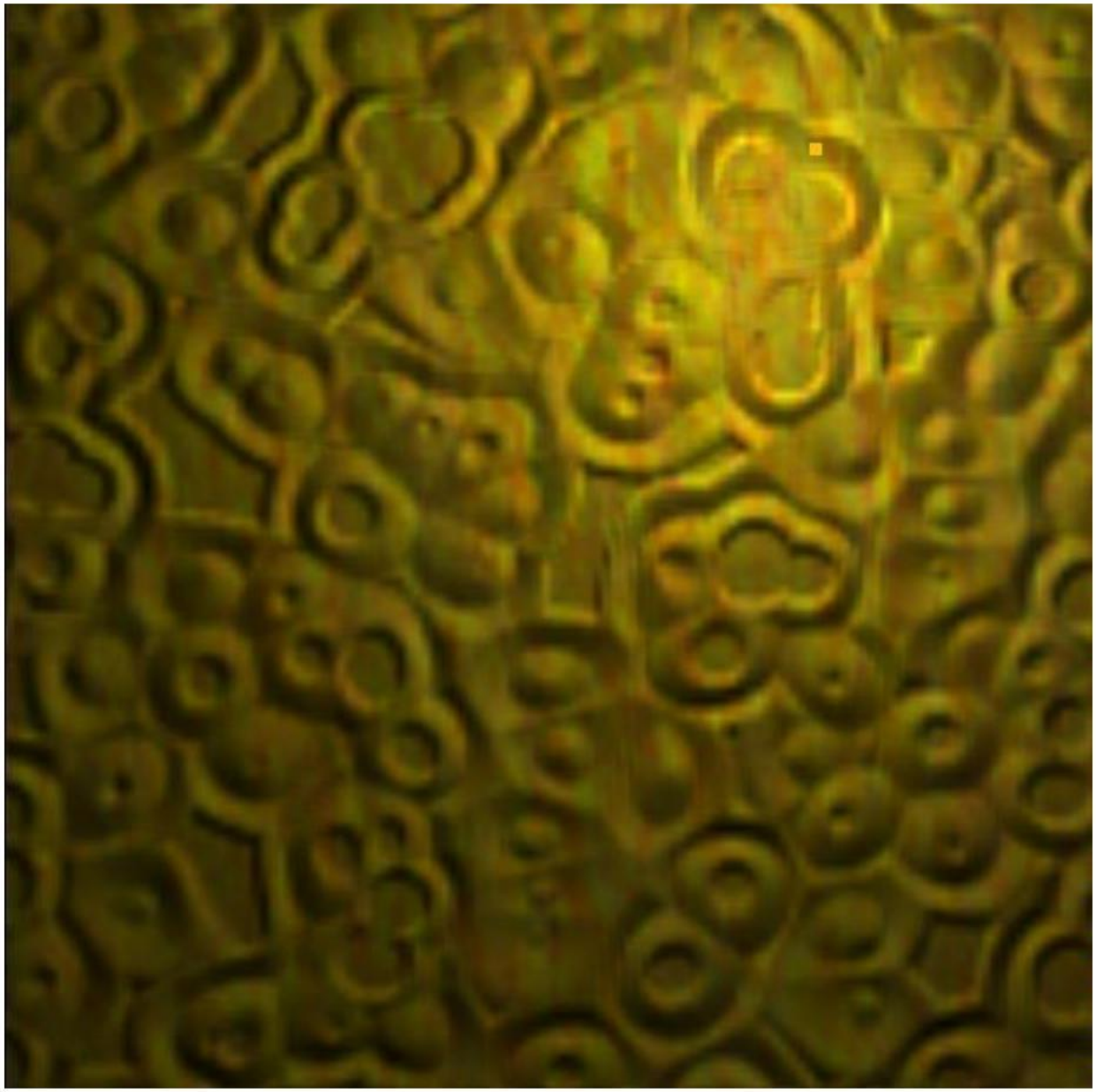
- *OpenGL* –д жижиг хэмжээтэй зургийг рендерлэхэд тохирох хэмжээний зургийг сонгох процессийг автоматжуулсан

Bump mapping

Синтетик зураг дээр илүү бодит байдлыг нэмж өгдөг арга техник юм. Энэ нь гадаргууг товойлгоход илүү нарийвчлалтай shading тооцоолдог.

Ердийн texture mapping (bump mapping хэрэгжүүлээгүй)

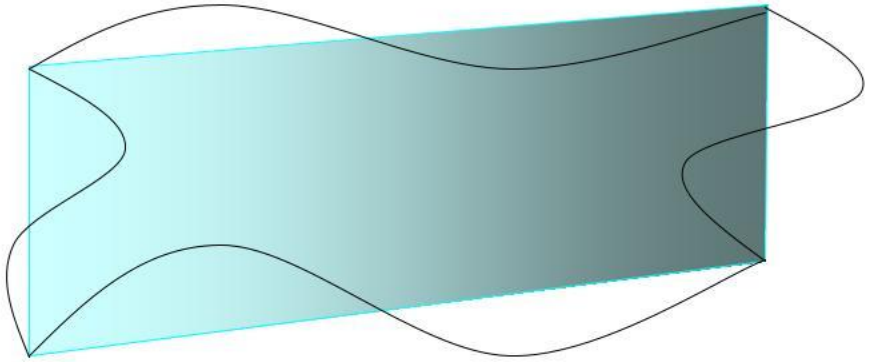
bump mapping хэрэгжүүлсэн байдал



Bump Mapping

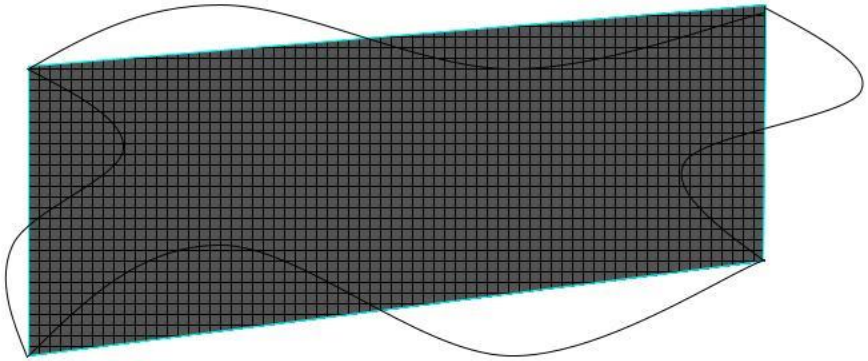
Bump mapping

- Энэхүү аргыг Жим Блин 1978 онд зохион бүтээжээ.
- Жижиг муруй гадаргуугийн хэлтэрхийг олон өнцөгтөөр ойролцоолъё.



- Gouraud shading үед гэрэлтүүлгийн тооцоог зөвхөн оройн цэг дээр хийдэг. Олон өнцөгтийн хоорондох пикселийн эрчим нь оройн хэсгээс холилдоно.

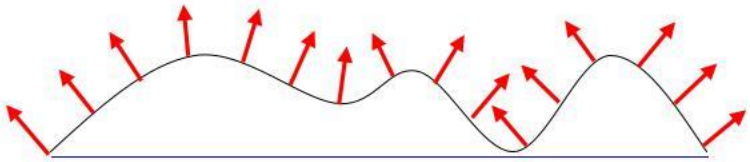
- bump mapping аргад гэрлийн тооцооллыг цэг тус бүрт авч үзнэ. Анхдагч гадаргуугийн нормалийг тооцоололд ашиглана.



- Гадаргуугийн сүүдрийн деталиудыг олон өнцөгтийг subdividing хийх шаардлаггүйгээр сайжруулна.

1D жишээ

- Муруйг шулууны сегментүүдээр ойролцоолно.

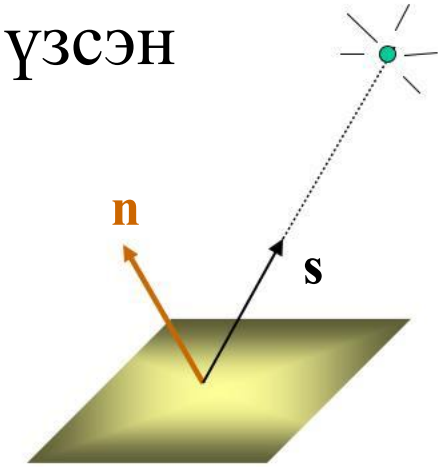


- Gouraud shading аргад light intensity оройн цэгүүд дээр тооцоолно. Шулуун дээрх бусад цэгүүдийн эрчимийг хоёр оройн цэгээс холилдоно (blended).

- bump mapping аргад light intensity цэг бүрт тооцоолно. Анхны гадаргуугийн нормалыг ашиглана. (Муруйг шулуунаар ойролцоолж байгааг анхаарна уу.)

Цэгийн гэрлийн эх үүсвэрээс diffuse reflection-ны эрчимийг тооцоолох тухай өмнө үзсэн

$$I_d = \frac{R_d I_i}{a + bd + cd^2}(\mathbf{s} \bullet \mathbf{n})$$



$a = 1, b = c = 0$ байна гэж үзье (No attenuation). $R_d I_i$ нь гадаргуу дээрх цэг бүрт тогтмол байна. $k = R_d I_i$ гэж үзье. Дараах томъёо бий болно.

$$I_d = k(\mathbf{s} \bullet \mathbf{n}) \qquad (1)$$

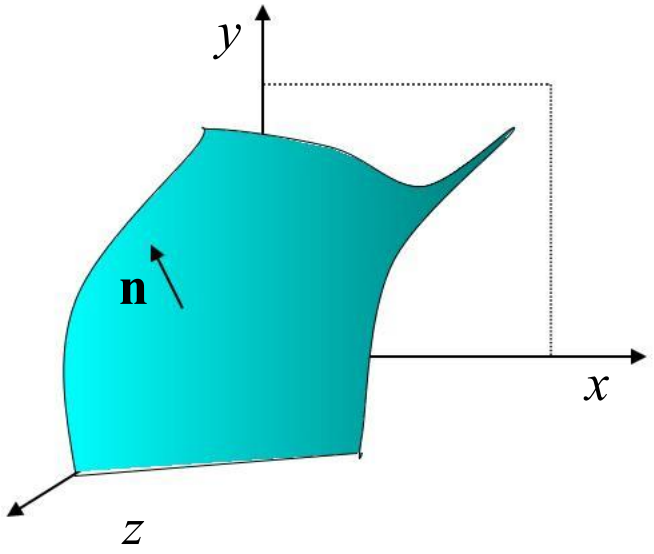
- Гадаргууг $d(x, y)$ height (depth) функцээр тодорхойлно гэж үзье. Иймд $z - d(x,y) = 0$ гадаргуугийн тэгшитгэл.
- Гадаргуу дээрх цэг $(x, y, d(x,y))$ –ийн нормал вектор нь

$$\mathbf{n} = (-\partial d/\partial x\ ,\ -\partial d/\partial y,\ 1)$$

- x чиглэлийн шүргэгч нь $Px = (1, 0, \partial d/\partial x)$.

  y чиглэлийн шүргэгч нь $Py = (0, 1, \partial d/\partial y)$.

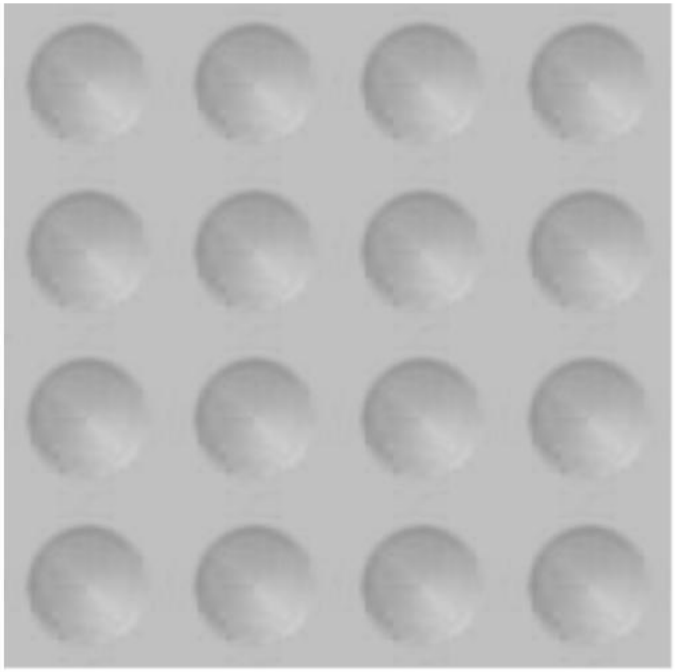
  Нормаль: $\mathbf{n} = Px \times Py$.



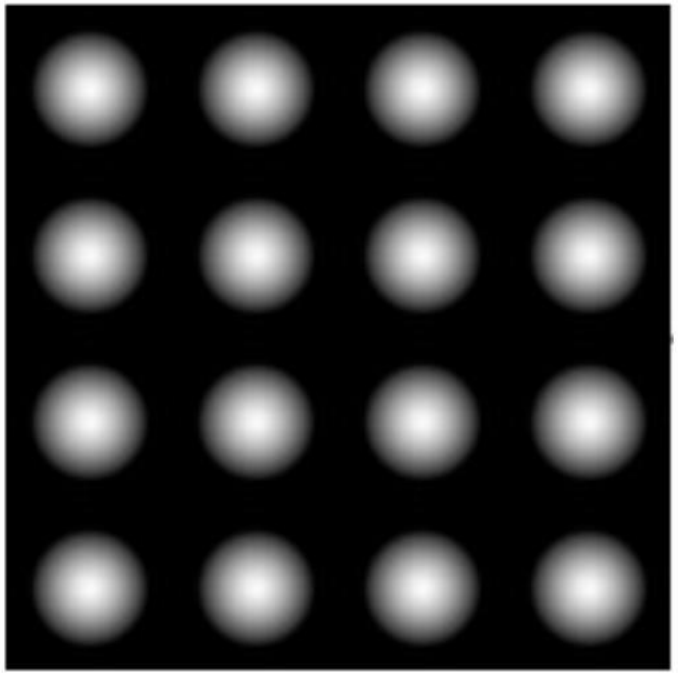
Энэ аргаар олж авсан n-ийг ашиглан бид пикселийн гэрлийн эрчмийг тооцоолж болно. Уг тооцооллыг гадаргууг дүрсэлсэн пиксел бүрт хийдэг. (Удаан.)

Height Map

Текстүр зургийн height map хувьд height саарал түвшинд хадгална.



Товгор гадаргуу



Гадаргуугийн height map

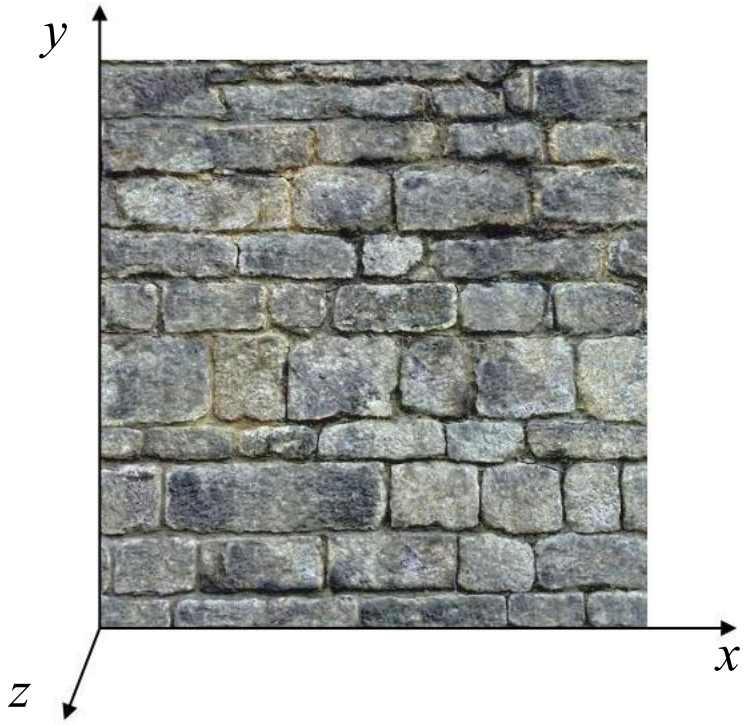
- Орчин үеийн график боловсруулах нэгж (*gpu*) нь пиксел бүрт shading тооцоолох shader компоненттэй байдаг
- Пикселийн нормалийг normal maps гэж нэрлэгдэх color buffer -т тодорхойлж өгдөг.
- Normal map дахь normal (Nx, Ny, Nz), $-1 \leq Nx, Ny, Nz \leq 1$ утга нь өнгөний утгыг илэрхийлнэ.Жнь,
  $R = floor[(Nx + 1) * 127.5] , 0 \leq R \leq 255$
  $G = floor[(Ny + 1) * 127.5] , 0 \leq G \leq 255$
  $B = floor[(Nz + 1) * 127.5] , 0 \leq B \leq 255$

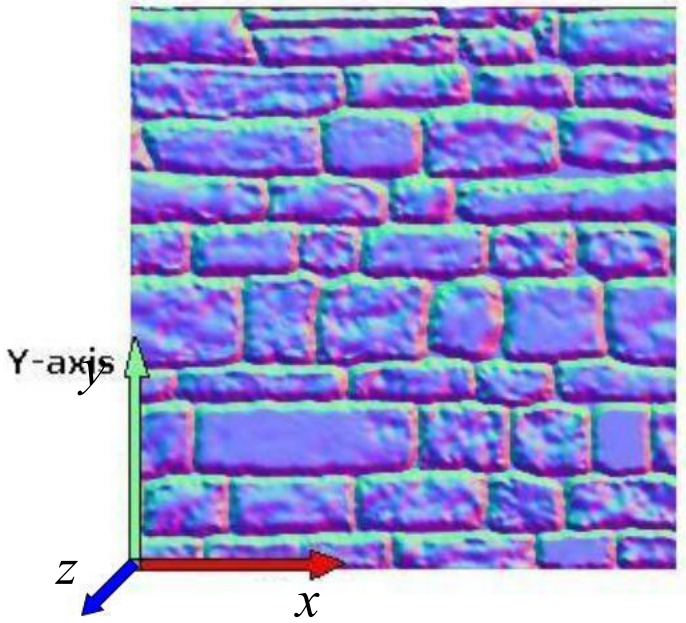
Баруун талын зурагт зүүн талын зураг дээр normal map хэрэгжүүлсэн жишээ юм. Normal map-г өндрөөс нь тооцоолж авна.

x чиглэлийн шүргэгч нь Px = (1, 0, $\partial d/\partial x$).

y чиглэлийн шүргэгч нь Py = (0, 1, $\partial d/\partial y$).

n = Px × Py = (- $\partial d/\partial x$, - $\partial d/\partial y$, 1).





normal map ашиглан Bump mapping рендерлэсэн жишээ

http://members.shaw.ca/jimht03/normal.html

Normal mapping

- Өнөө үед бодит хугацаанд зургийг рендерлэх чанарыг дээшлүүлэхэд normal maps ашигладаг.
  1. Маш сайн нарийвчилсан 3D загварыг бүтээж, рендерлэнэ.
  2. Загвараас normal map гаргаж авна
  3. Геометр загвар дээр normal map хэрэгжүүлнэ

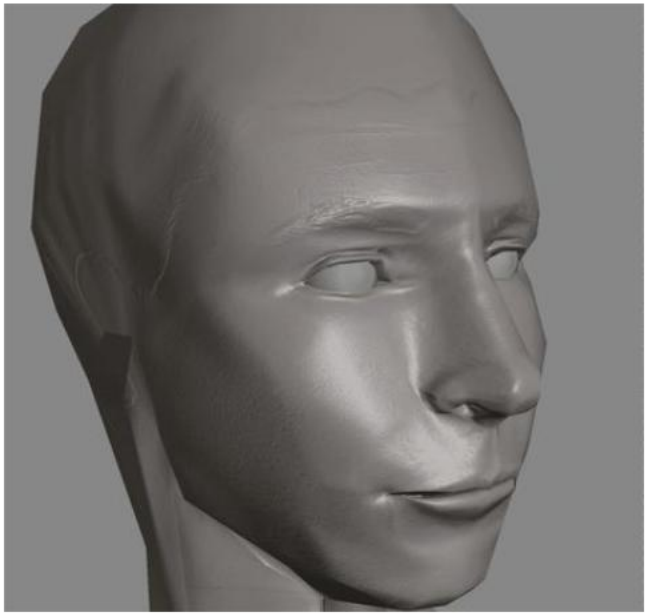
Rendered with
30000 polygons

normal map үүсгэсэн байдал

Ижил толгойн моделийг 632 олон өнцөгтөөр рендерлэсэн (normal map-гүй)

normal map хэрэгжүүлсэн

original 30000-head

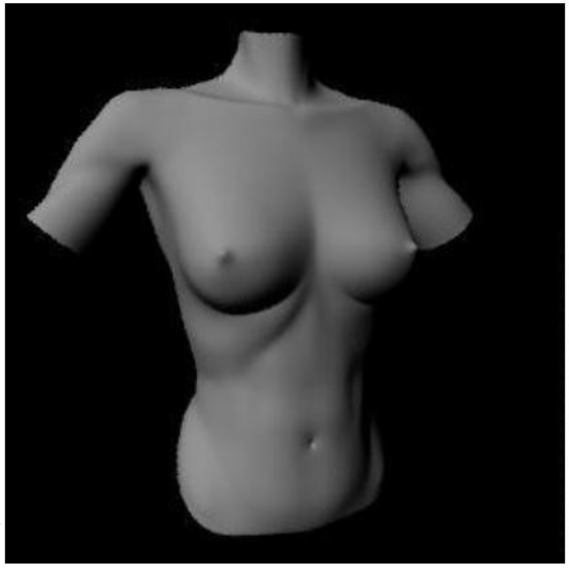


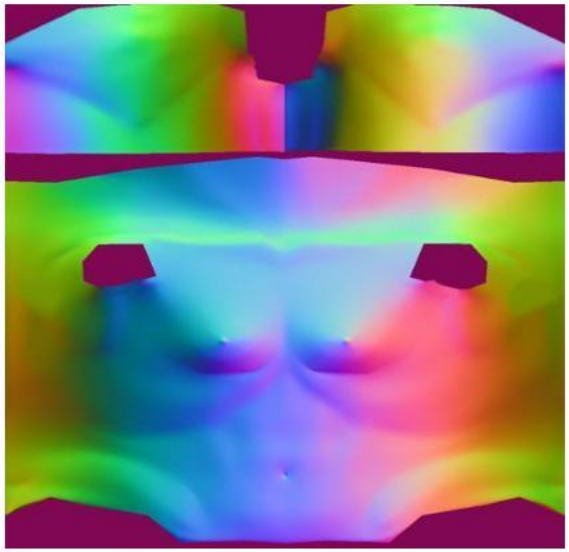
http://www.monitorstudios.com/bcloward/tutorials_normal_maps5.html

Normal mapping жишээ

http://www.drone.org/tutorials/normal_maps.html

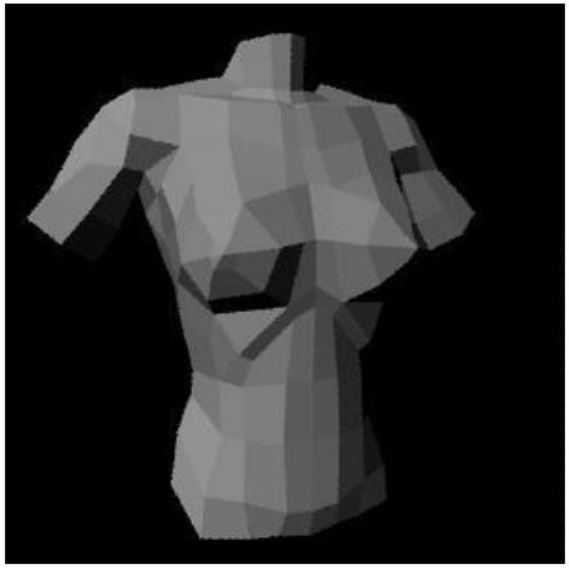
Fine model

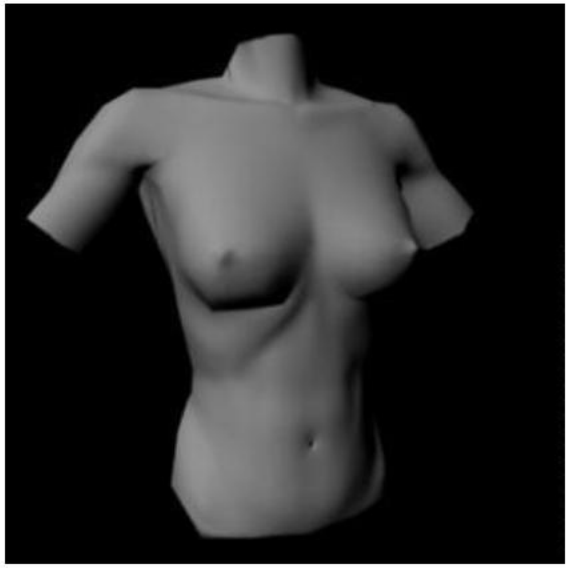




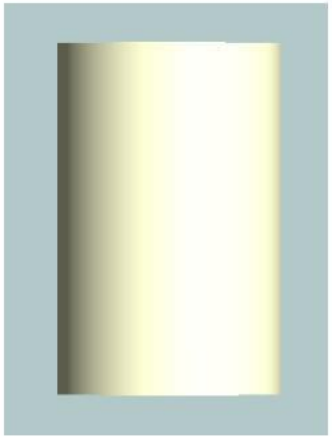
Normal map

Coarse model



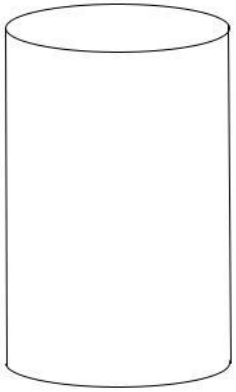
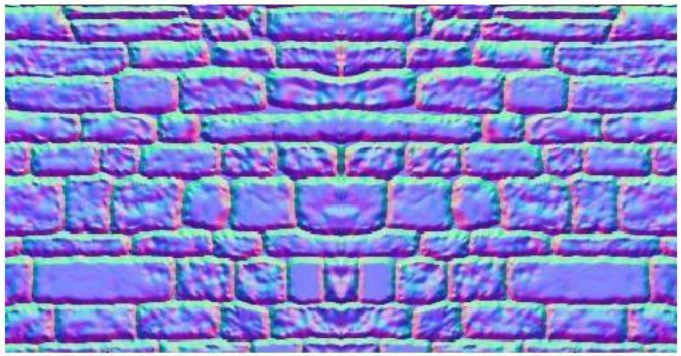


Coarse model with normal map

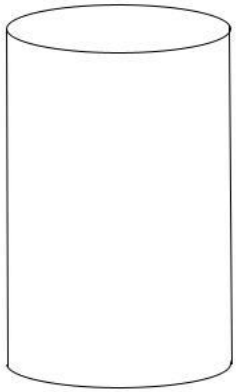

+

=

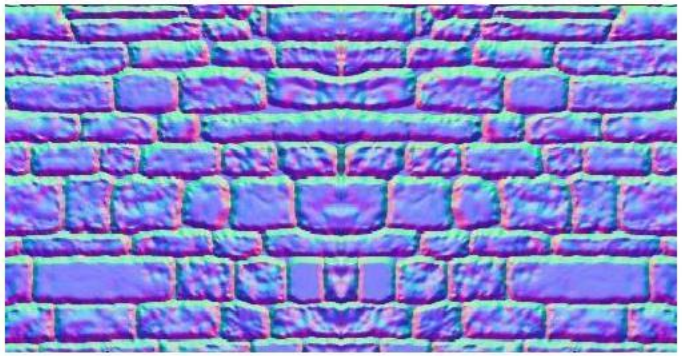

Texture map


+

=


Normal map



+



+

=