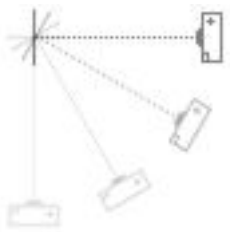
Billboards explained

2004 by Peter Grundmann

Introduction



Billbording is a fast and simple way used to orient a polygon always face the camera. This technic is the basic used for decals, light flares, lens flares, particles, aso. By a given center position, the size of the decal and the view matrix we can calculate the position of the decal's edge points.

Get the Orientation

As we know, the orientation of a billboard is the reversed orientation as the viewer/camera. So we have to get the current view matrix, extract the Right- and Up-Vector and normalize them to get unit vectors (length is one).

```
ViewMatrix  = GetCurrentViewMatrix();
vRight      = Normalize (  ViewMatrix._11, ViewMatrix._21, ViewMatrix._31 );
vUp         = Normalize (  ViewMatrix._12, ViewMatrix._22, ViewMatrix._32 );
```

If you have problems to understand, than take a look at the DirectX view matrix:

	vRight	*vUp*	*vDirection*	-
Rotation	$\_11 = R_X$	$\_12 = U_X$	$\_13 = D_X$	0
Vectors	$\_21 = R_Y$	$\_22 = U_Y$	$\_23 = D_Y$	0
	$\_31 = R_Z$	$\_32 = U_Z$	$\_33 = D_Z$	0
vPosition	$\_41 = P_X$	$\_42 = P_Y$	$\_43 = P_Z$	1

Size the Billboard

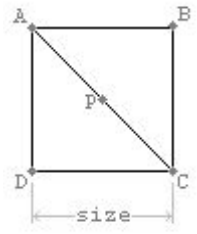
Knowing the direction of the billboard, we also need the dimension for each direction. We can size the two extracted vectors to the half size of the billboard:

```
vRight  = vRight * Size / 2.0f;   // where Size is the size of the billboard
vUp     = vUp    * Size / 2.0f;
```

Calculate the edge vertices

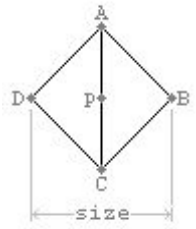
Now, as we know the Position of the billboard we can create the four edge vertices by two differend ways:

a) a square shape




```
A = P - Right + Up
B = P + Right + Up
C = P + Right - Up
D = P - Right - Up
```

b) a diamond shape




```
A = P + Up
B = P + Right
C = P - Up
D = P - Right
```

Render the billboard

The billboard can now simply rendered as a triangle fan and have to be realigned everytime the view matrix (camera) change.

Some hints

If you understand this simple code you wont have any problems to calculate texture coordinates. In the case you need a normal vector for the vertices, simple use the reversed (multiply each element by -1.0f) direction vector from the view matrix.

On todays GPU's, billbording is supported directly on hardware via Point-Sprites.