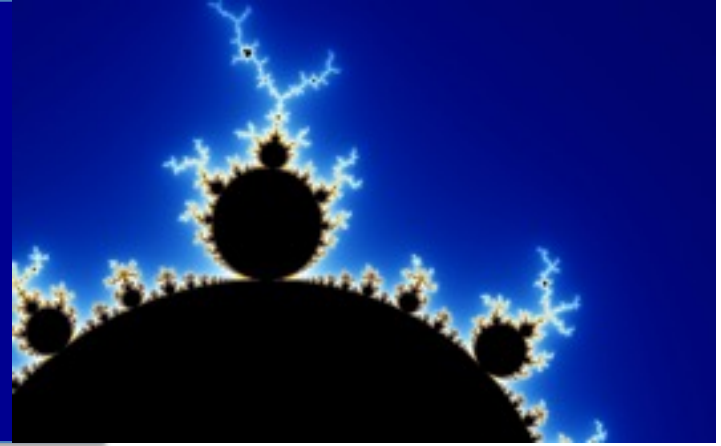
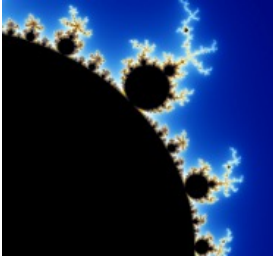


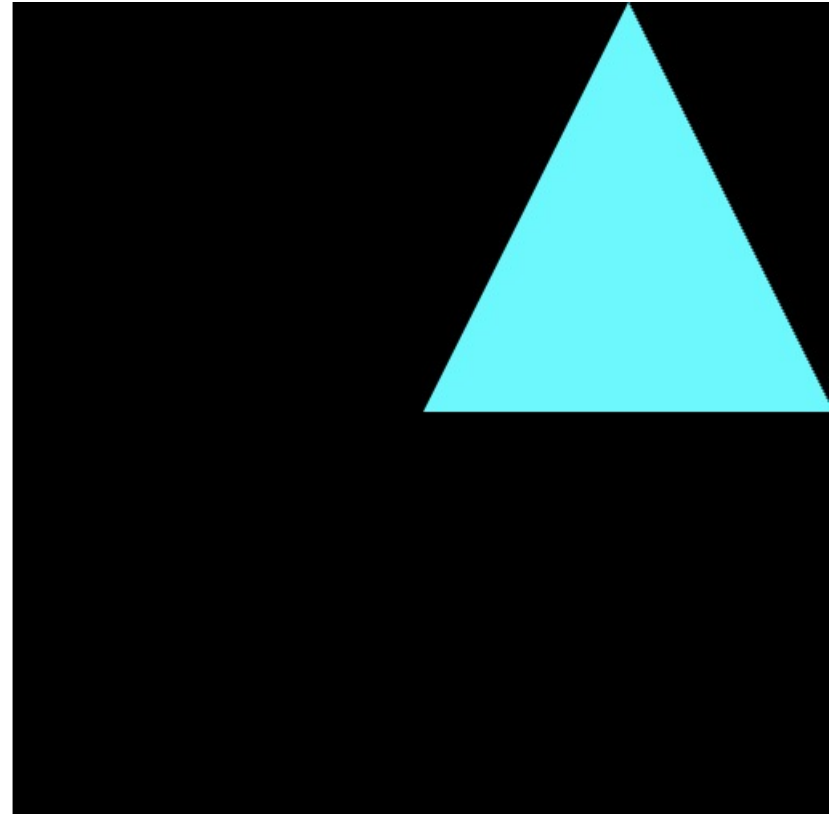
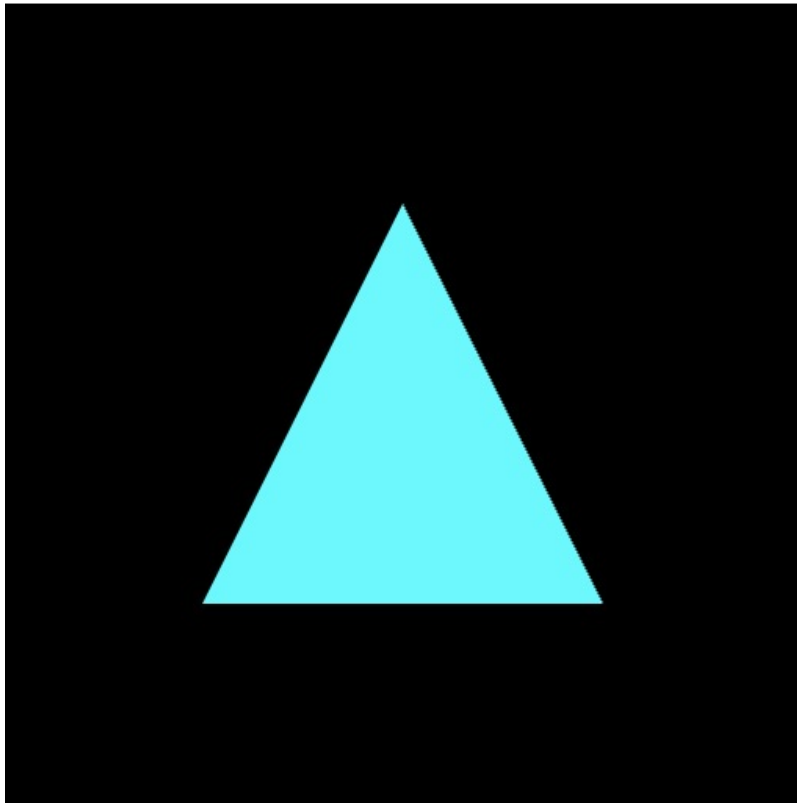
Computer Graphics



A First Look at Transformation



Translate a Triangle

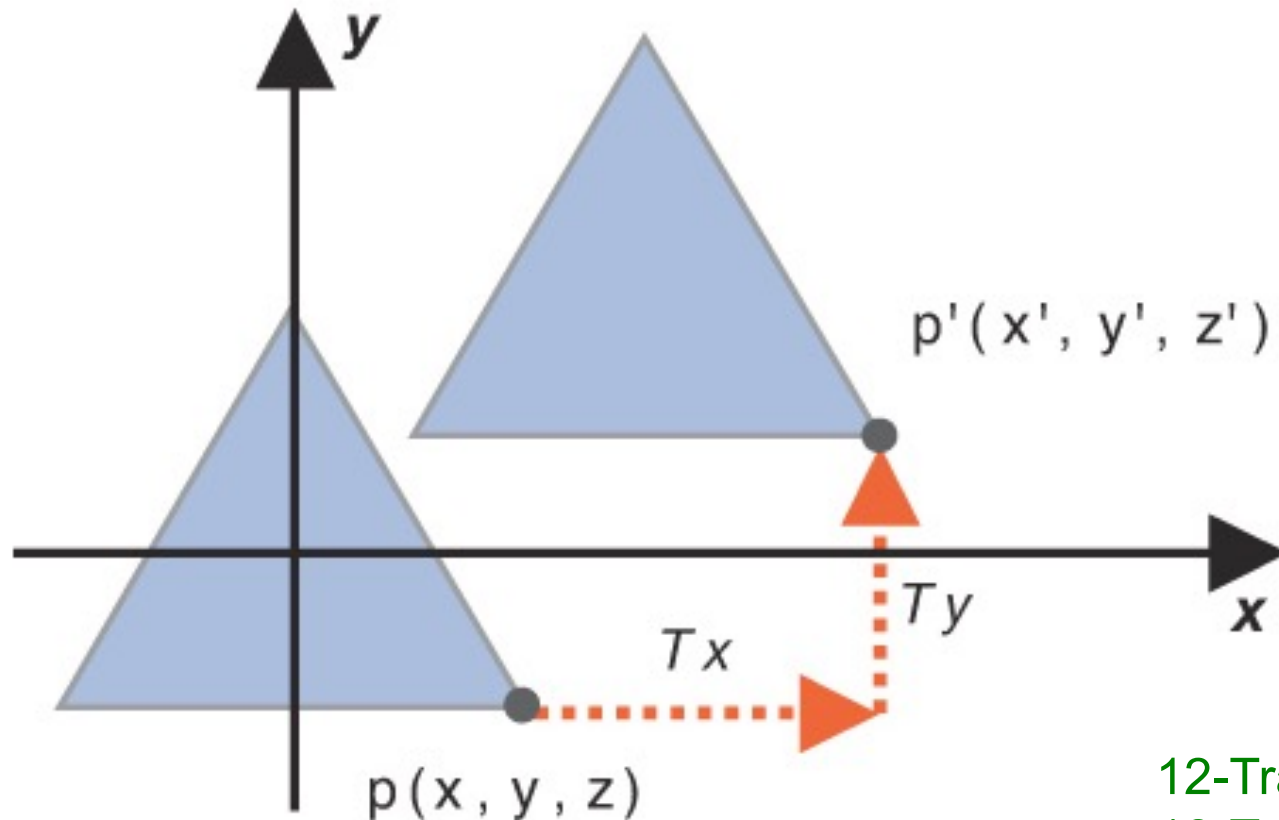


Translate a triangle

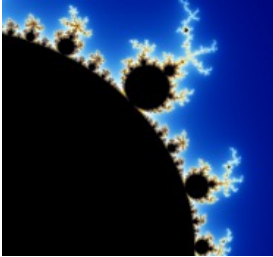
$$x' = x + T_x$$

$$y' = y + T_y$$

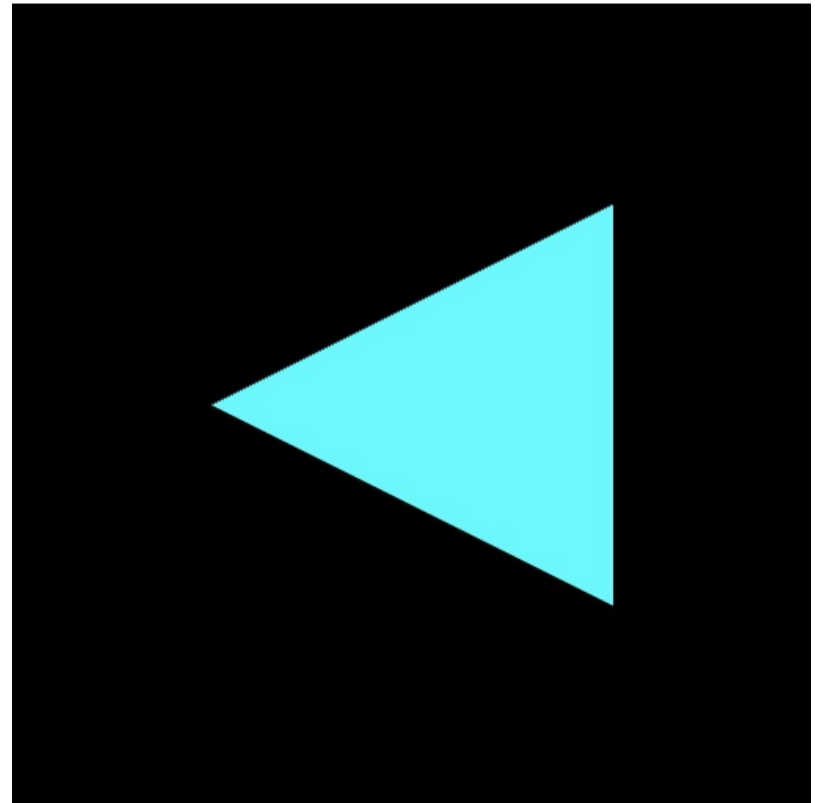
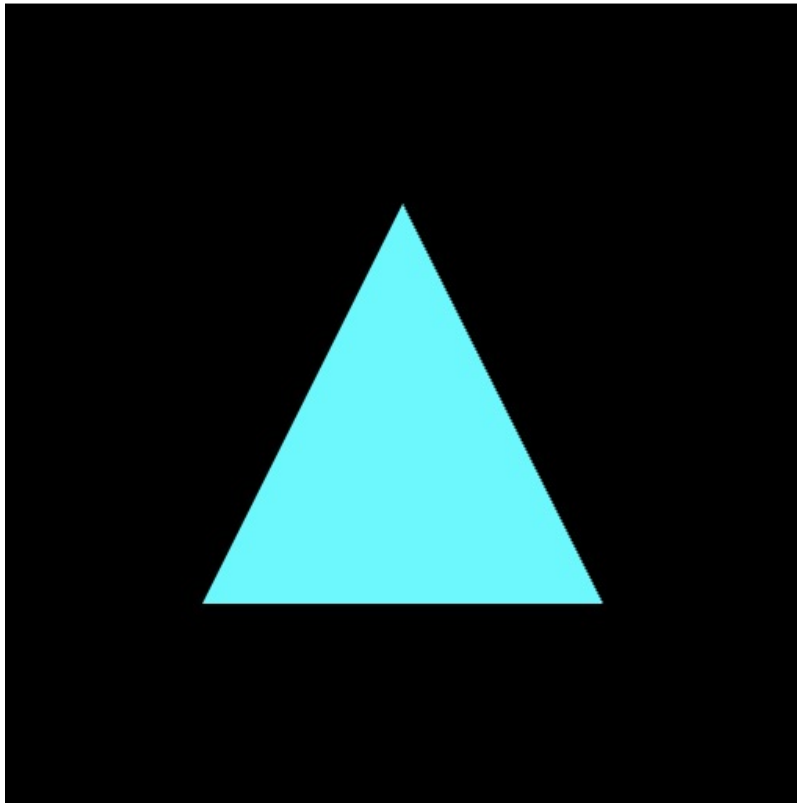
$$z' = z + T_z$$



[12-TranslateTriangle.html](#)
[12-TranslateTriangle.js](#)



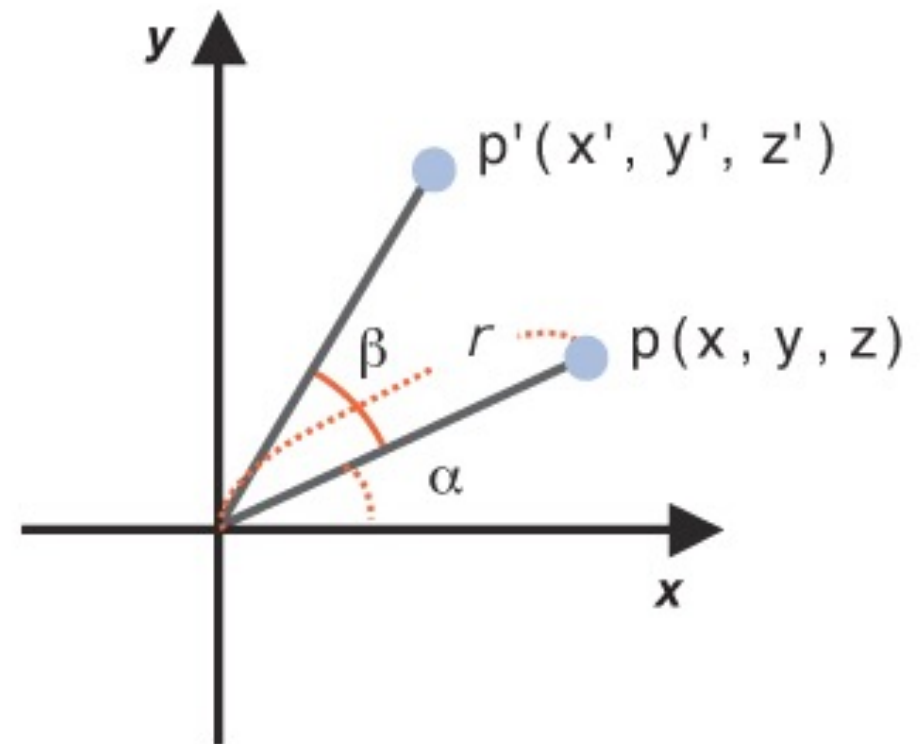
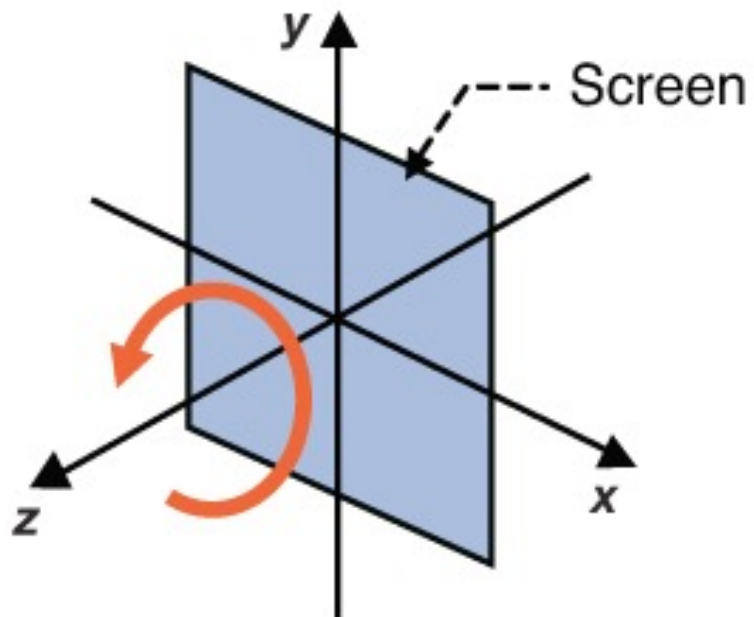
Rotate a Triangle



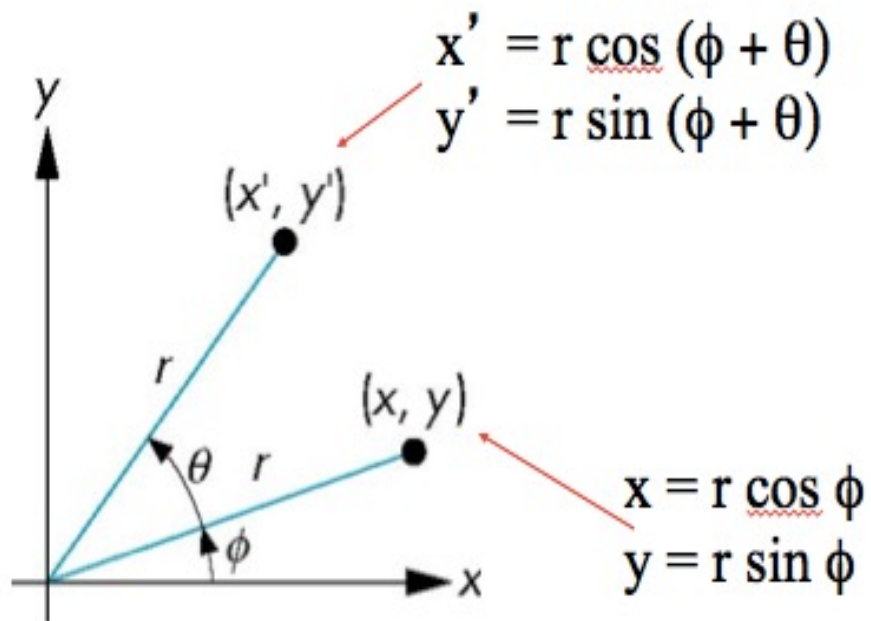
Rotate a Triangle

Rotation Angle β :

Positive \leftrightarrow Counter Clockwise



Rotate a Triangle

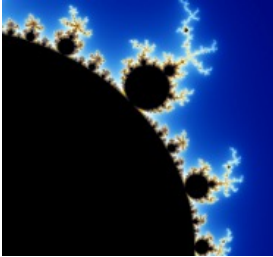


$$\begin{aligned}\cos(\theta + \Phi) &= \cos(\theta) \cos(\Phi) - \sin(\theta) \sin(\Phi); \\ \sin(\theta + \Phi) &= \sin(\theta) \cos(\Phi) + \cos(\theta) \sin(\Phi).\end{aligned}$$

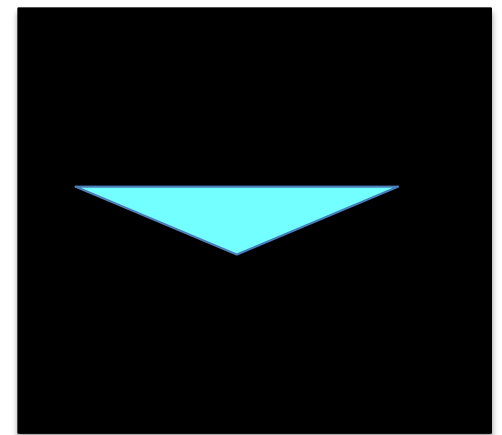
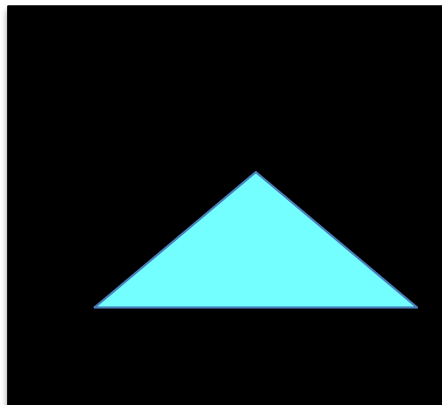
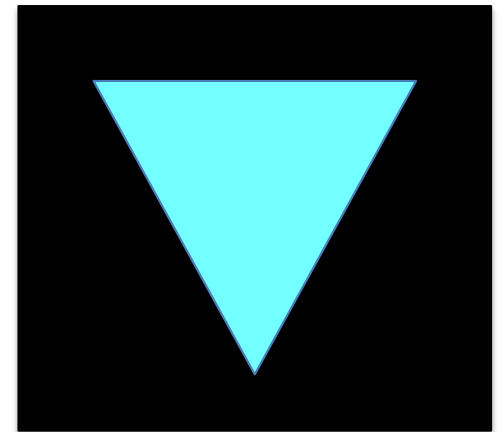
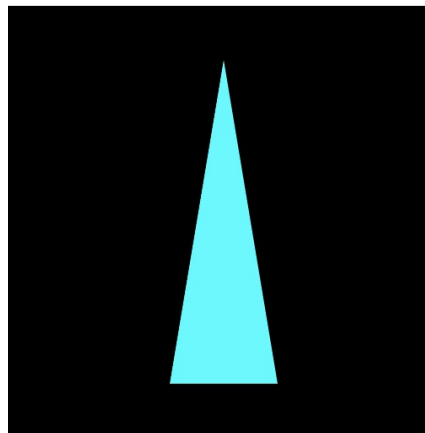
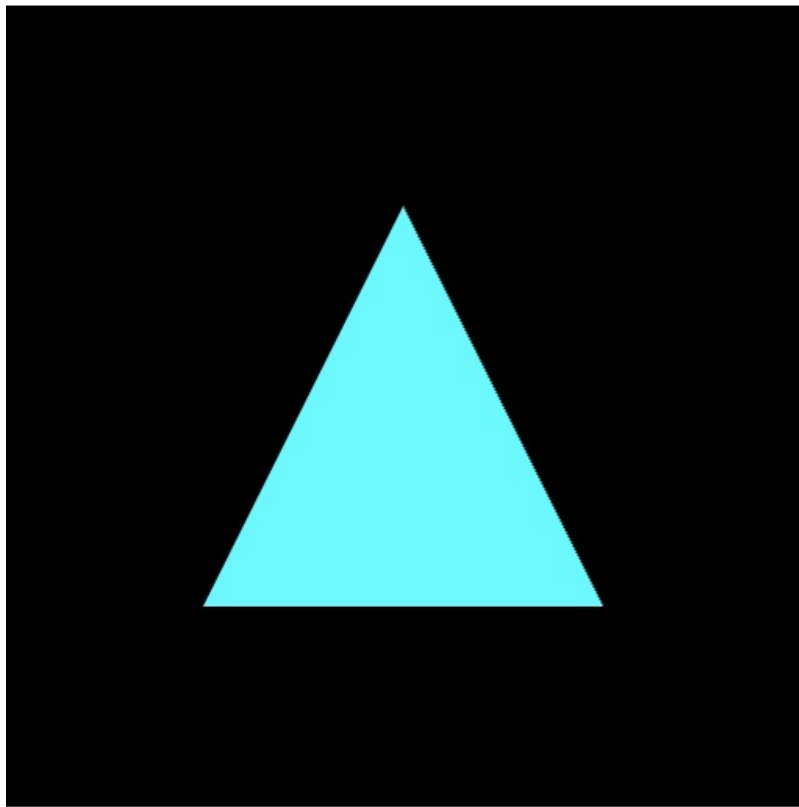


$$\begin{aligned}x' &= x \cos \theta - y \sin \theta \\ y' &= x \sin \theta + y \cos \theta \\ z' &= z\end{aligned}$$

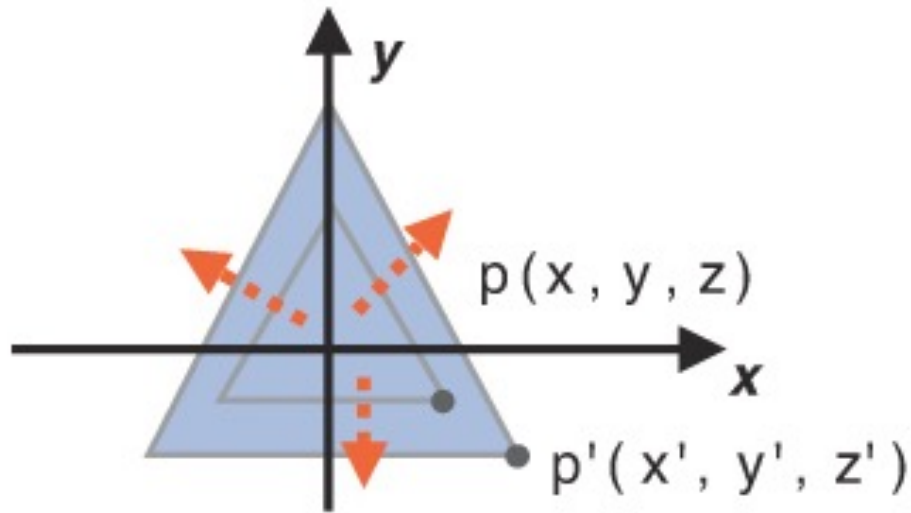
12-RotatedTriangle.html
12-RotatedTriangle.js



Scale a Triangle



Scale a Triangle



$$\begin{aligned}x' &= S_x * x \\y' &= S_y * y \\z' &= S_z * z\end{aligned}$$



$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$



Pass Matrix onto WebGL

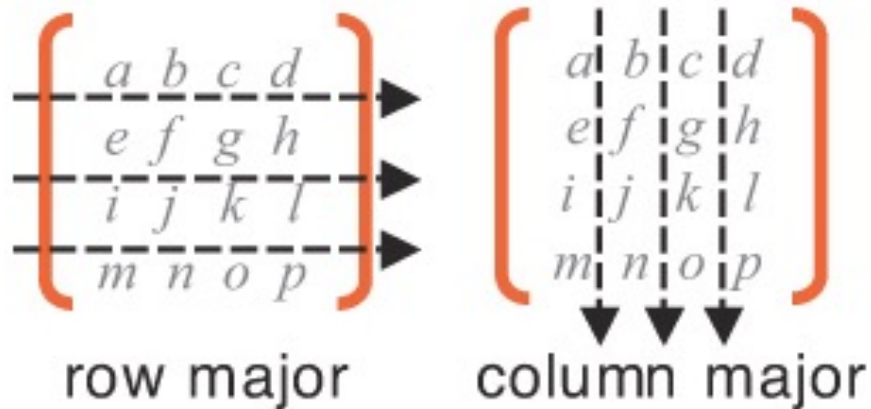
```
gl.uniformMatrix4fv (location, transpose, array)
```

Assign the 4×4 matrix specified by *array* to the uniform variable specified by *location*.

Parameters	location	Specifies the storage location of the uniform variable.
	Transpose	Must be <i>false</i> in WebGL. ³
	array	Specifies an array containing a 4×4 matrix in column major order (typed array).
Return value	None	
Errors	INVALID_OPERATION	There is no current program object.
	INVALID_VALUE	<i>transpose</i> is not <i>false</i> , or the length of <i>array</i> is less than 16.

12-ScaleTriangle.html
12-ScaleTriangle.js

Row or Column major order?



OpenGL/WebGL
Column major order


$$\begin{bmatrix} S_x & 0 & 0 & 0 \\ 0 & S_y & 0 & 0 \\ 0 & 0 & S_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Row and column major the same



Transformation Matrix for Translation?

Translation Matrix:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & T_x \\ 0 & 1 & 0 & T_y \\ 0 & 0 & 1 & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

OpenGL/WebGL
Column major order



$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ T_x & T_y & T_z & 1 \end{bmatrix}$$

[12-TranslationTriangle2.html](#)
[12-TranslationTriangle2.js](#)

Transformation Matrix for Rotation?

Rotation Matrix:

$$\begin{bmatrix} x' \\ y' \\ z' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\beta & -\sin\beta & 0 & 0 \\ \sin\beta & \cos\beta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \times \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

OpenGL/WebGL
Column major order

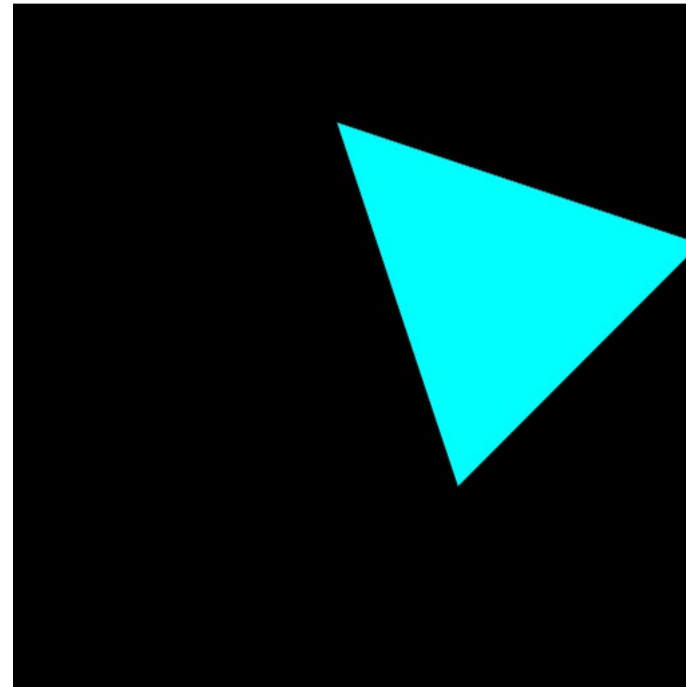
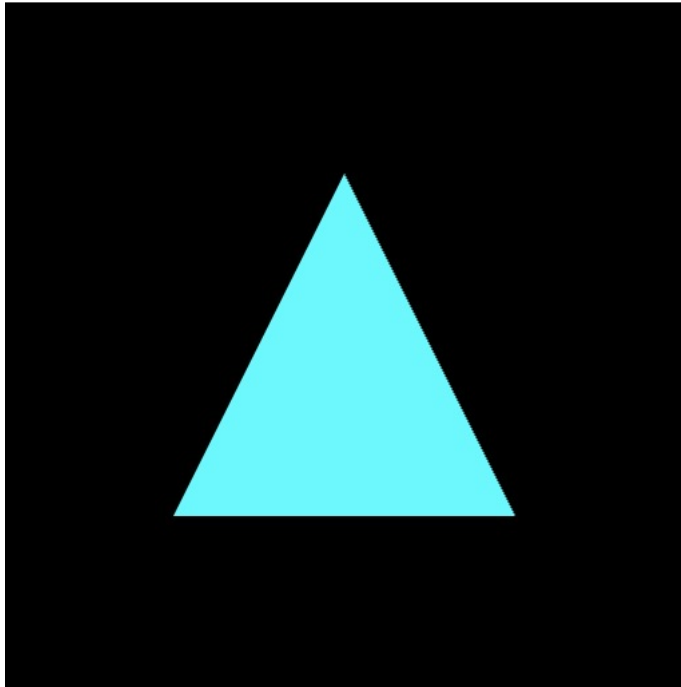


$$\begin{bmatrix} \cos\theta & \sin\theta & 0 & 0 \\ -\sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

[12-RotationTriangle2.html](#)
[12-RotationTriangle2.js](#)



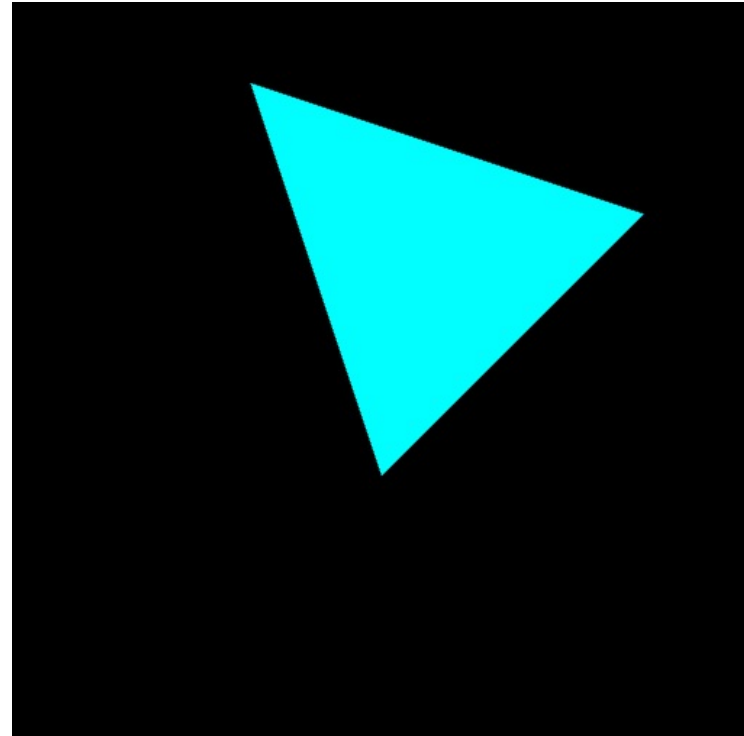
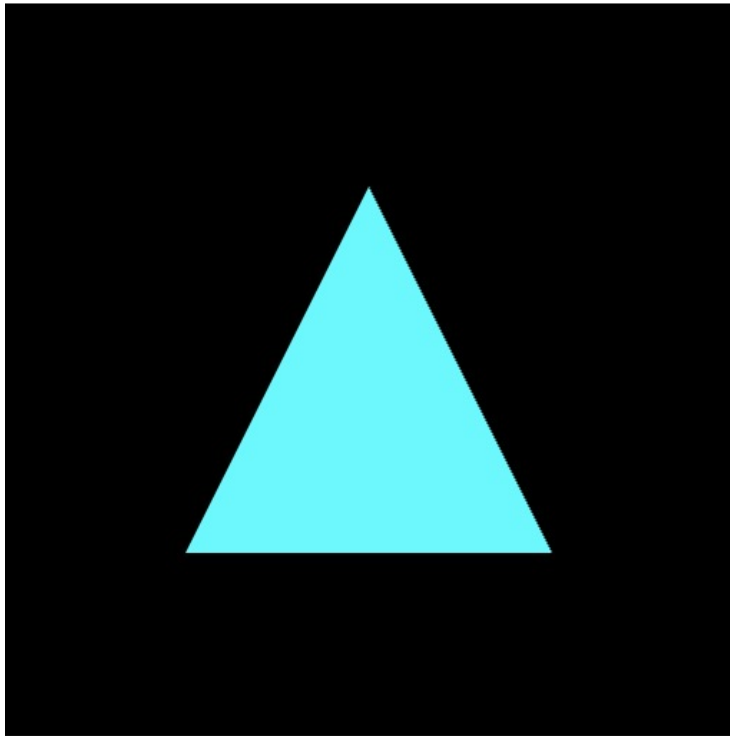
Two Consecutive Transformations



Rotate and then Translate

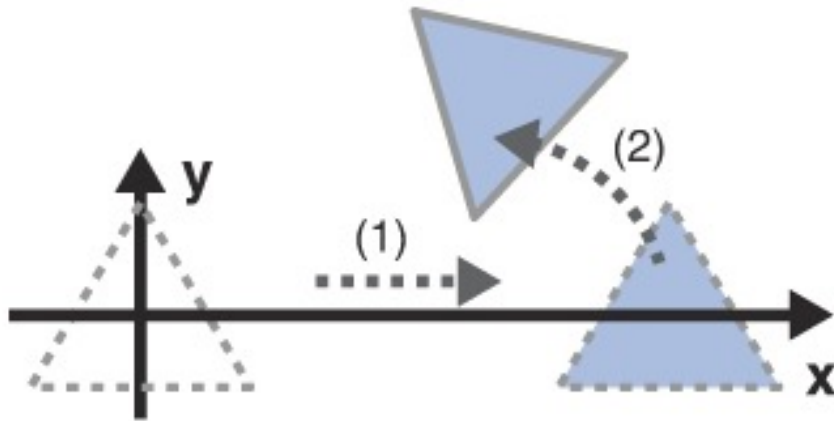


Two Consecutive Transformations



Translate and then Rotate

Two Consecutive Transformations



$\langle \text{"translated" coordinates} \rangle =$

$$\langle \text{translation matrix} \rangle \times \langle \text{original coordinates} \rangle$$

$\langle \text{"translated and then rotated" coordinates} \rangle =$

$$\langle \text{rotation matrix} \rangle \times \langle \text{translated coordinates} \rangle$$

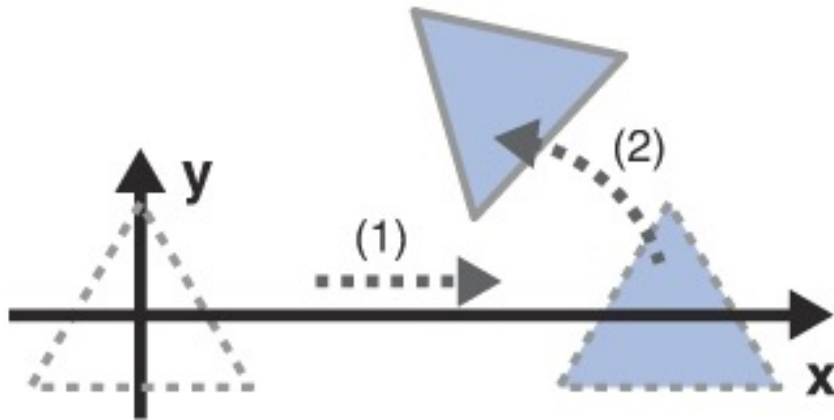
$\langle \text{"translated and then rotated" coordinates} \rangle =$

$$\langle \text{rotation matrix} \rangle \times (\langle \text{translation matrix} \rangle \times \langle \text{original coordinates} \rangle)$$

$$\langle \text{rotation matrix} \rangle \times (\langle \text{translation matrix} \rangle \times \langle \text{original coordinates} \rangle)$$

$$(\langle \text{rotation matrix} \rangle \times \langle \text{translation matrix} \rangle) \times \langle \text{original coordinates} \rangle$$

Two Consecutive Transformations



$$\underbrace{(\langle \text{rotation matrix} \rangle \times \langle \text{translation matrix} \rangle)} \times \langle \text{original coordinates} \rangle$$

```
// perform translation and then rotation
var modelViewMatrix = mat4();
var r=rotate(ANGLE, 0, 0, 1);
var t=translate(0.3, 0.3, 0);
modelViewMatrix = mult ( mult(modelViewMatrix, r), t);

// send over the modelview transformation matrix to vertex shader
gl.uniformMatrix4fv(modelViewMatrixLoc, false, flatten(modelViewMatrix));

// Draw the rectangle
gl.drawArrays(gl.TRIANGLES, 0, n);
```



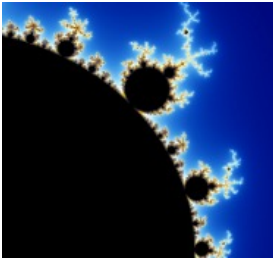

Translate and Rotate

`r = rotate(45, 0, 0, 1)`

`t = translate(0.3, 0.3, 0.0)`

`modelViewMatrix = r * t`

$$= \begin{bmatrix} \cos 45 & -\sin 45 & 0 & 0 \\ \sin 45 & \cos 45 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} 1 & 0 & 0 & 0.3 \\ 0 & 1 & 0 & 0.3 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



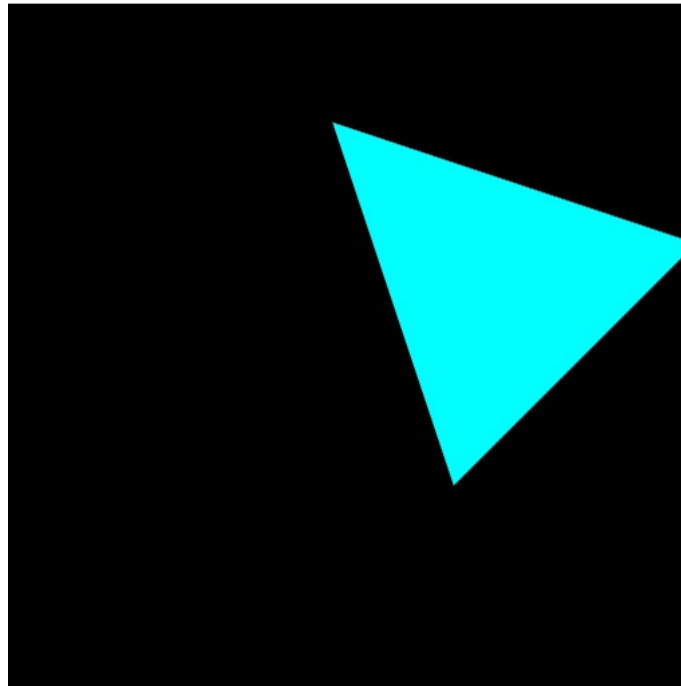
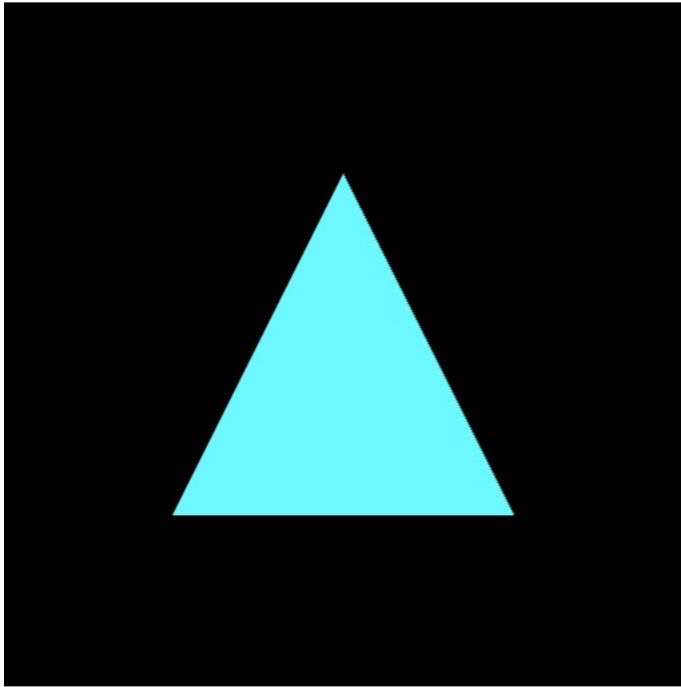
Matrix Multiplication

$$A = \begin{bmatrix} a_{00} & a_{01} & a_{02} \\ a_{10} & a_{11} & a_{12} \\ a_{20} & a_{21} & a_{22} \end{bmatrix}, B = \begin{bmatrix} b_{00} & b_{01} & b_{02} \\ b_{10} & b_{11} & b_{12} \\ b_{20} & b_{21} & b_{22} \end{bmatrix}$$

$$\begin{bmatrix} a_{00} \times b_{00} + a_{01} \times b_{10} + a_{02} \times b_{20} & a_{00} \times b_{01} + a_{01} \times b_{11} + a_{02} \times b_{21} & a_{00} \times b_{02} + a_{01} \times b_{12} + a_{02} \times b_{22} \\ a_{10} \times b_{00} + a_{11} \times b_{10} + a_{12} \times b_{20} & a_{10} \times b_{01} + a_{11} \times b_{11} + a_{12} \times b_{21} & a_{10} \times b_{02} + a_{11} \times b_{12} + a_{12} \times b_{22} \\ a_{20} \times b_{00} + a_{21} \times b_{10} + a_{22} \times b_{20} & a_{20} \times b_{01} + a_{21} \times b_{11} + a_{22} \times b_{21} & a_{20} \times b_{02} + a_{21} \times b_{12} + a_{22} \times b_{22} \end{bmatrix}$$



Two Consecutive Transformations



Rotate and then Translate

```
var modelViewMatrix = mat4();  
var r=rotate(ANGLE, 0, 0, 1);  
var t=translate(0.3, 0.3, 0);  
modelViewMatrix = mult ( mult(modelViewMatrix, t), r);
```