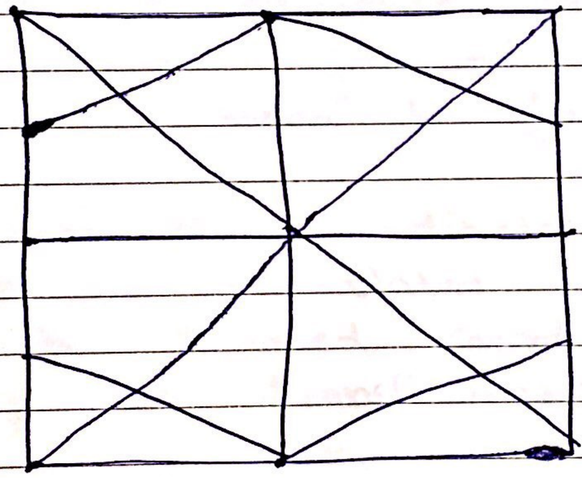
combination of 24 bit line segments.
→ In 24 bit line segment code each bit rep-
-resent a single line.
→ To highlight a line we put corresponding bit 1 in 24 bit line segment code & 0 otherwise.

Drawbacks:→
1) 24 bit segment code is required to put in memory for generating character. Hence extra memory is required in the method.
2) Character quality is poor due to limited face.



★ Anti-Aliasing:→
→ is a s/w technique for diminishing jaggies- stairstep like lines that should be smooth. Jaggies occur because the output device, the monitor or printer does not have high enough resolution to represent a smooth line. Antialiasing reduces the prominence of jaggies by surrounding the stairsteps with intermediate shades of gray (for gray scaling device) or color

Title

Date

Notes

Mo Tu We Th Fr Sa Su

(for Color devices). Although it reduces the jagged appearance of the lines it also make them fuzzier.

→ Another method for reducing jaggies is called smoothing in which the printer changes the size & horizontal allignment of dots to make curves smoother.

→ Anti-aliasing is sometimes called as over-sampling.

2-D Transformations :-

→ Transformation means changing some graphic into something else by applying rules. We can have various types of transformations such as translation, scaling up or down, rotation, shearing etc. When a transformation takes place on a 2-D plane it is called as 2-D transformation.

→ Transformations play an important role in computer graphics to reposition the graphics on the screen & change their size or orientation.

⇒ Homogenous Coordinates :-

→ To perform a sequence of transformations such as translation followed by rotation & scaling we need ~~need~~ to follow a sequential process :-

- Translate the coordinates
- Rotate the translated coordinates
- Scale the rotated coordinates to complete the composite transformation.

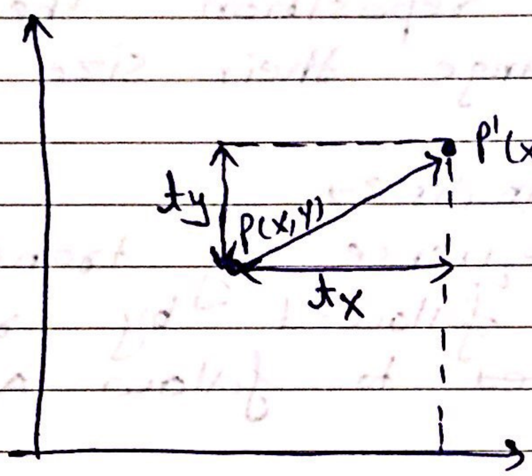
→ To shorten this process, we have to use 3x3 transformation matrix instead of 2x2 transformation matrix. To convert a 2x2 matrix to a 3x3 matrix we have to add an extra dummy coordinate W.

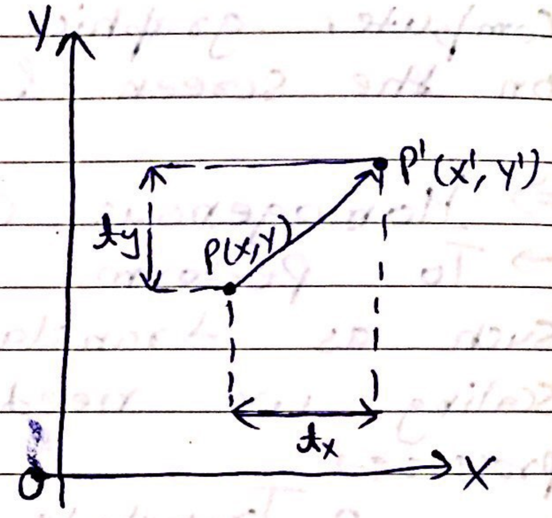
→ In this way we can represent the point by 3 numbers instead of two numbers, which is called homogenous coordinate system. In this system we can represent all the transformation equations in matrix multiplication.

Any Cartesian point $P(x,y)$ can be converted to homogenous coordinate by $P'(x_h, y_h, h)$.

Translation :→

→ A translation moves an object to a diff. position on the screen. You can translate a in 2D by adding translation coordinate (t_x, t_y) to the original coordinate (X, Y) to get the new coordinate (x', y').





From the above figure, we can write:→

$$x' = x + t_x$$
$$y' = y + t_y$$

→ The pair (t_x, t_y) is called the translation vector or shift vector. The above equation can also be represented using the column vectors :→

$P = \begin{bmatrix} x \\ y \end{bmatrix}$ $\quad P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$ $\quad T = \begin{bmatrix} t_x \\ t_y \end{bmatrix}$

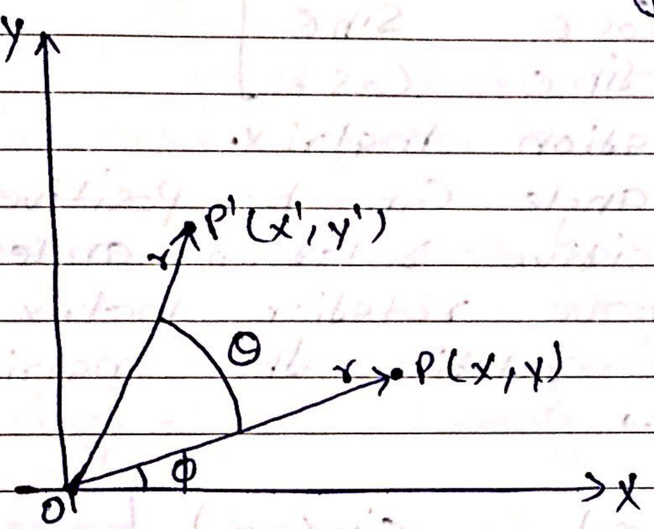
It can be written as:→

$\boxed{P' = P + T}$ $\quad \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix} \Rightarrow \begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}$

Rotation :→

→ In rotation we rotate the object at particular angle θ (theta) from its origin. From the following fig. we can see that the point $P(X, Y)$ is located at angle ϕ from the horizontal X coordinate with distance r from the origin.

→ Let us suppose you want to rotate it at the angle θ (theta). After rotating it to a new location, we will get a new point $P'(x', y')$.



→ Using standard trigonometry the original coordinate of point $P(X, Y)$ can be represented as:→

$$X = r\cos\phi \quad — ①$$
$$Y = r\sin\phi \quad — ②$$

→ In the same way we can represent the point $P'(x', y')$:→

$$x' = r\cos(\theta + \phi)$$
$$= r\cos\theta\cos\phi - r\sin\theta\sin\phi \quad — ③$$
$$y' = r\sin(\theta + \phi)$$
$$= r\cos\phi\sin\theta + r\sin\phi\cos\theta \quad — ④$$

Substitute eq ① & ② in eq ③ & ④ :→

$$x' = x\cos\theta - y\sin\theta$$
$$y' = x\sin\theta + y\cos\theta$$

Representing the above eq. in matrix form

$$[x'\ y'] = [x\ y]\begin{bmatrix}\cos\theta & \sin\theta \\ -\sin\theta & \cos\theta\end{bmatrix}$$

$$P' = P \cdot R$$

where $R = \begin{bmatrix}\cos\theta & \sin\theta \\ -\sin\theta & \cos\theta\end{bmatrix}$

→ R is called rotation matrix.

→ The rotation angle can be positive or negative. For positive rotation angle we can use the above rotation matrix. However for negative angle rotation the matrix will change as below:→

$$R = \begin{bmatrix}\cos(-\theta) & \sin(-\theta) \\ -\sin(-\theta) & \cos(-\theta)\end{bmatrix}$$

As, $\begin{cases}\cos(-\theta) = \cos\theta \\ \sin(-\theta) = -\sin\theta\end{cases}$

$$R = \begin{bmatrix}\cos\theta & -\sin\theta \\ \sin\theta & \cos\theta\end{bmatrix}$$

Download WinBGIm from
http:// winbgim.codecutter.org/

→ It will Contain :→

→ graphics.h
→ winbgim.h
→ libbgi.a

→ graphics.h & winbgim.h in include folder.
of Codeblocks folder.

→ Copy & Paste libbgi.a file in lib folder
of codeblocks folder.

→ Codeblocks → Setting → Compiler setting → Linker Setting → Add lib (libbgi.a) & in other link Options → Add Command :→
-lbgi -lgdi32 -lcomdlg -luuid -loleaut32 -lole32

→ If error Appears then download the library from :→
http:// github.com/tanvir002700/Blog-Post/blob/master/ winBGIm_GCC47.zip?raw=true

Scanned with CamScanner

→ Scaling :→ To change the size of an object Scaling transformation is used. In the Scaling process we either expand or Compress the dimensions of the object. Scaling can be achieved by multiplying the Original Coordinates of the object with the Scaling factor to get the desired result.

→ Let us assume that the original Coordinates are (x, y), the scaling factor are (Sx, Sy) and the produced coordinates are (x', y'). This can be mathematically represented as:-

$$x' = x \cdot S_x \quad \& \quad y' = y \cdot S_y$$

→ The scaling factor S_x, S_y scales the object in x and y direction respectively. The above equation can also be represented in matrix form as below:→

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

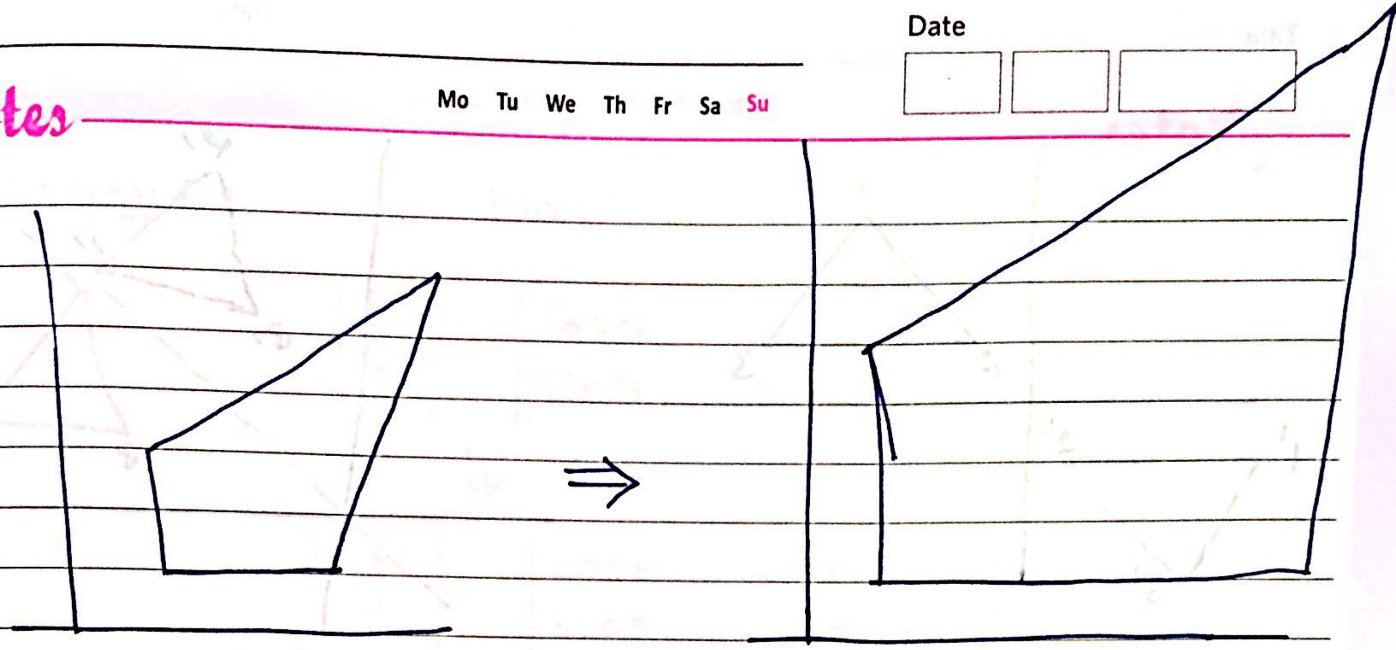
$$\begin{bmatrix} x' & y' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix}$$

Or

$$P' = P \cdot S$$

where S is the scaling matrix.

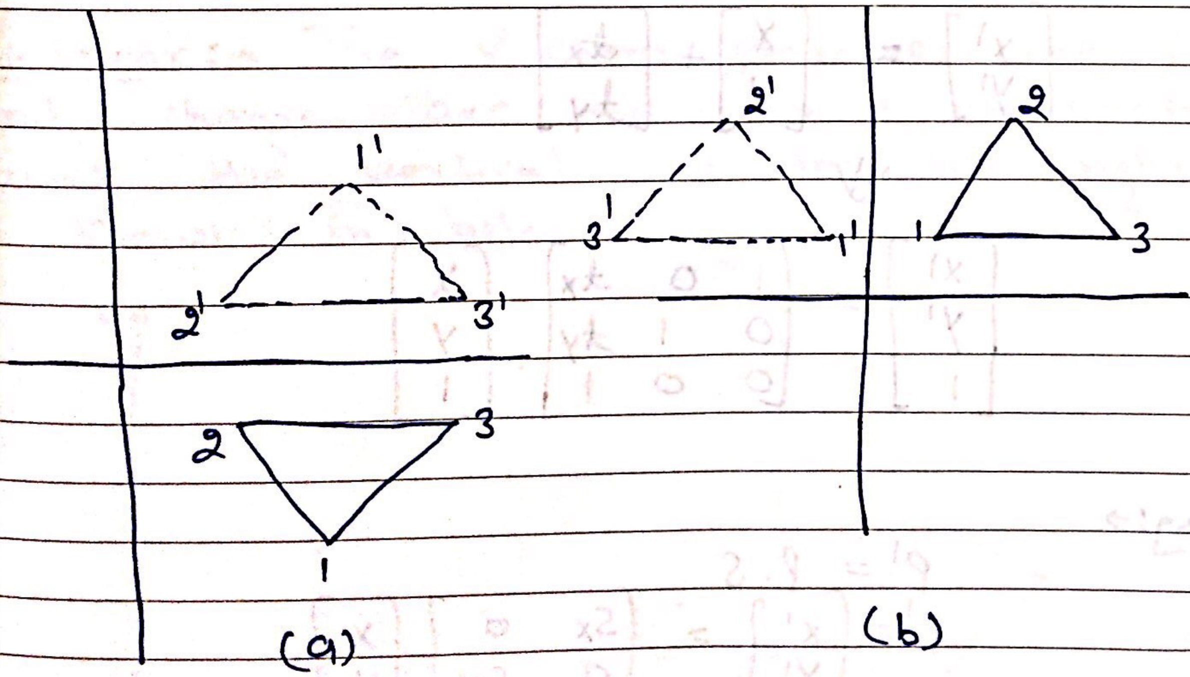
→ If we provide values less than 1 to the scaling factor S, then we can reduce the size of the object. If we provide values



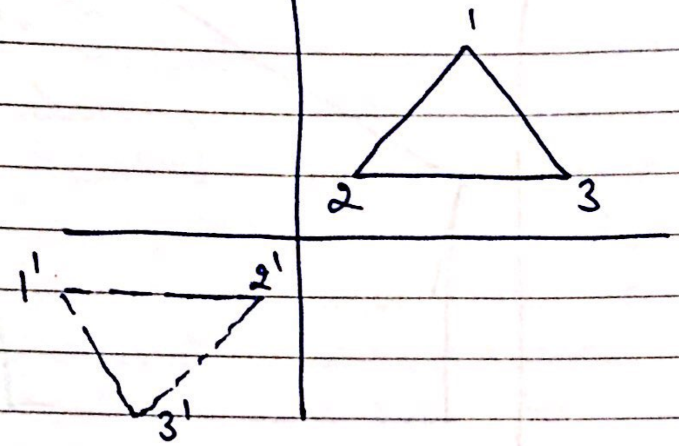
greator than 1, then we can increase the size of the object.

Reflaction:→

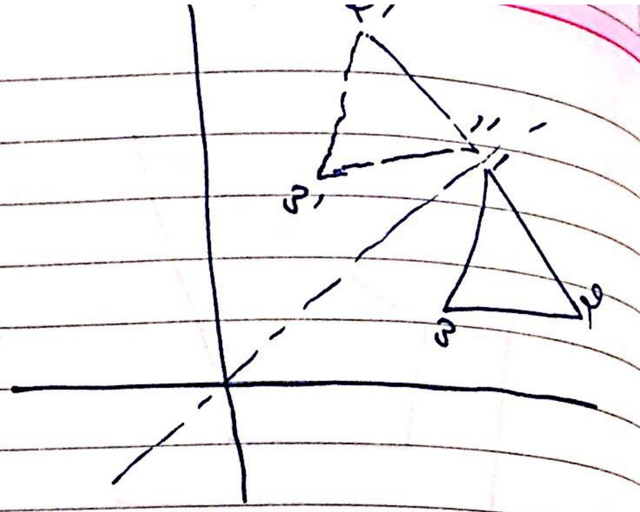
→ is the mirror image of original object. In other words, we can say that it is a rotation operation with 180°. In reflection transformation, the size of the object does not change.



(a) (b)



(C)



(d)

✩ Homogenous Coordinates:→

$$P' = PM_1 + M_2$$

→ M_1 → Multiplicative Matrix
M_2 → transactional term.

Translation:→

$$P' = P + M_1$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

$$\Downarrow$$

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Scaling:→

$$P' = P \cdot S$$

$$P' = \begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} S_x & 0 \\ 0 & S_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \end{bmatrix}$$

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

Rotation :-> $P' = R(\theta) \cdot P$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\Downarrow$$

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

☆ **Shearing :->** A transformation that slants the shape of an object is called shear transformation There are two shear transformations X-shear & Y-shear. One shifts X-coordinates values & other shifts Y coordinate values. However in both the cases only one coordinate changes its coordinates & other preserves its values. Shearing is also termed as "Skewing".

X-Shear :-> The X-shear preserves the Y coordinates and changes are made to X coordinates which causes the vertical lines to tilt right or left as shown in below fig.

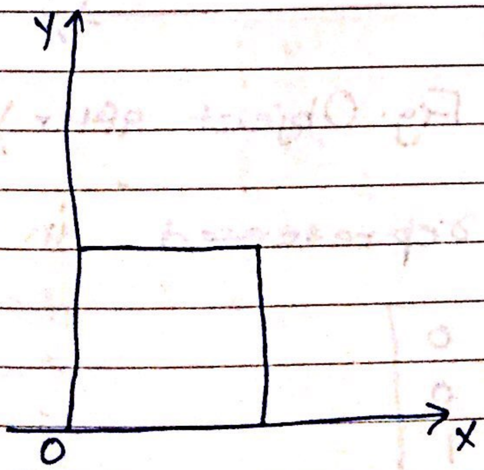


fig. Original object

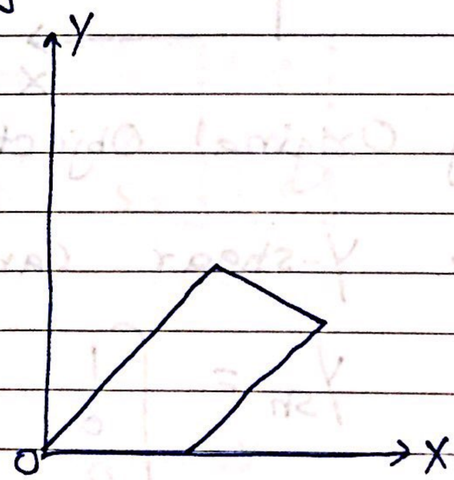


fig. Object after X-shear

→ The transformation matrix for X-shear Can be represented as:→

$$X_{sh} = \begin{bmatrix} 1 & 0 & 0 \\ Shx & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$X' = X + Sh_x \cdot Y$$
$$Y' = Y$$

Y-shear:→

→ The Y-shear preserves the X-coordinates & changes the Y coordinates which causes the horizontal lines to transform into lines which slopes up or down as shown in following figure:→

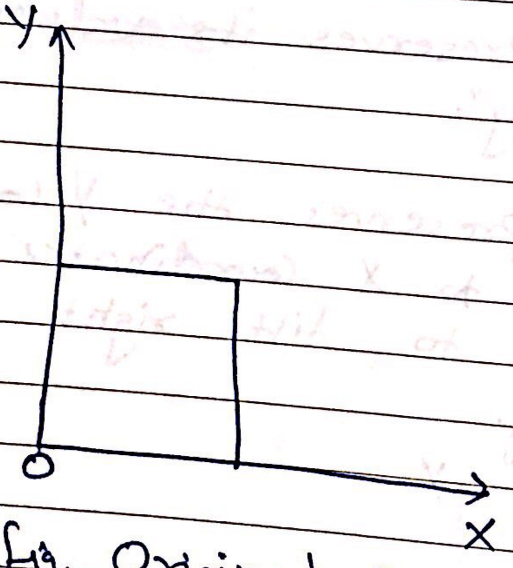


Fig. Original Object

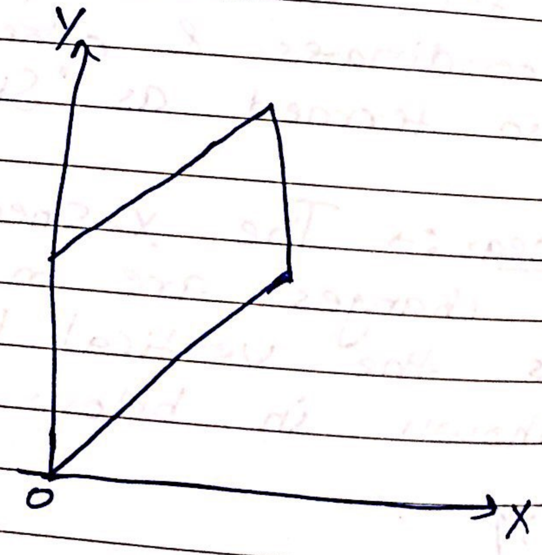


Fig. Object after Y-shear

→ The Y-shear can be represented in matrix

$$Y_{sh} = \begin{bmatrix} 1 & Shy & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$Y' = Y + Sh_y \cdot X \quad \& \quad X' = X.$$

Scanned with CamScanner

3D Transformations :→

- Translation → Moving | Reposition
- Scaling → Change in shape | Size
- Rotation → Rotate an object → θ angle
- Reflection → Rotate with 180°
- Shearing → Tilting along x | y axis.

Translation in 3D :→

$$x' = x + t_x$$
$$y' = y + t_y$$
$$z' = z + t_z$$

Matrix form :→

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Scaling :→

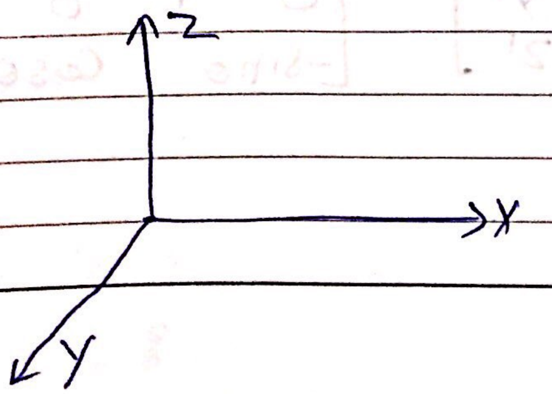
$$x' = x \cdot s_x$$
$$y' = y \cdot s_y$$
$$z' = z \cdot s_z$$

Matrix form :→

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

Rotation :→

- → X-roll
- → Y-roll
- → Z-roll



Z-roll :→ $z' = z$

$x' = x\cos\theta - y\sin\theta$

$y' = x\sin\theta + y\cos\theta$

matrix :→

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

X-Roll :→

$x' = x$

$y' = y\cos\theta - z\sin\theta$

$z' = y\sin\theta + z\cos\theta$

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

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

Y-Roll :→

$y' = y$

$z' = z\cos\theta - x\sin\theta$

$x' = z\sin\theta + x\cos\theta$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\theta \; \cancel{\sin\theta} & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

3D Reflection :→

XY Plane→

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

YZ plane :→

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

ZX Plane :→

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

3D Shearing :→

Shear – Z

:→ Sh_x, Sh_y

$z' = z$

$x' = x + z \cdot sh_x$

$y' = y + z \cdot sh_y$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & sh_x \\ 0 & 1 & Sh_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Shear – x :→

$x' = x$

$y' = y + Sh_y \cdot x$

$z' = z + Sh_z \cdot x$

Notes

Mo Tu We Th Fr Sa Su

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ sh_y & 1 & 0 \\ sh_z & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

Shear - y :→

$$y' = y$$
$$x' = x + Sh_x \cdot y$$
$$z' = z + Sh_z \cdot y$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & Sh_x & 0 \\ 0 & 1 & 0 \\ 0 & Sh_z & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$