Lecture 04

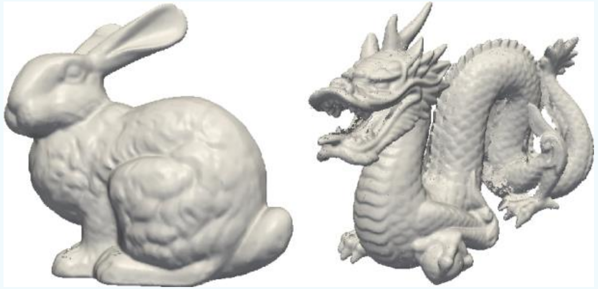
Modelling & Transforms (Part 1)

Prepared by Ban Kar Weng (William)

3D Modelling

3D Modelling

- Process of developing an **mathematical representation** of a 3D object.
- **3D Model** (*or simply **model***) refers to the geometric model enhanced with various other attributes (e.g. colour, texture, material reflectance, etc.)



All image are taken from the Internet. To take the address, right-click on the image and click the "Open Hyperlink" context menu item.

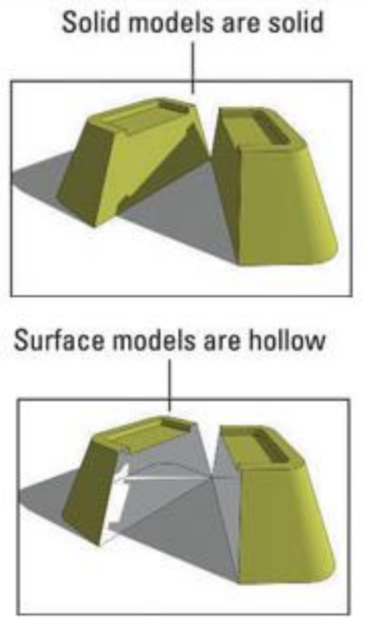
3D Model | Solid Model v.s. Surface Model

Solid Model

- Defines the **volume** of the object they represent
- **Applications**: Engineering and medical simulations
- **Example**: Voxel

Surface Model

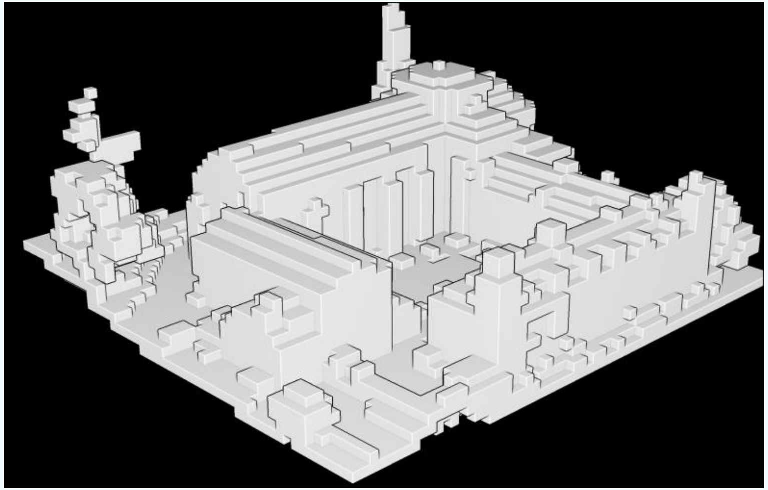
- Represents the **surface** (i.e. the object's boundary)
- **Applications**: Games and Film
- **Example**: Mesh, Point Cloud, etc.



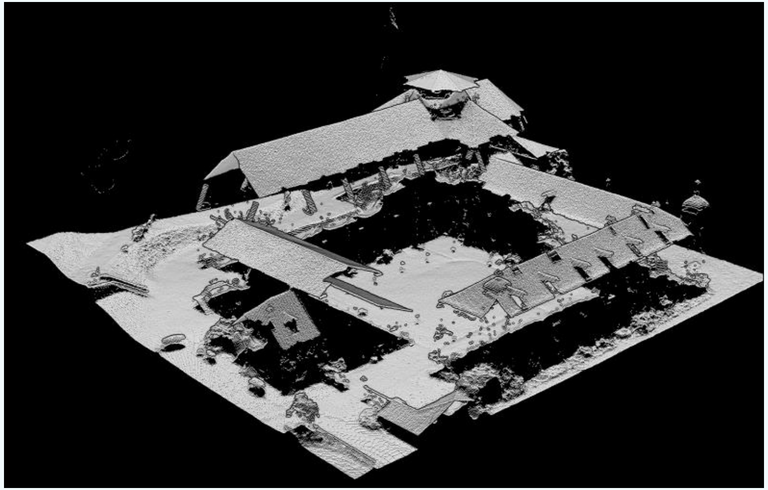
NOTE: Both can create functionally identical objects. Differences between them lies in the operations (e.g. creation, editing, etc.) they support.

Click the image to go to image source. Content source: https://towardsdatascience.com/how-to-represent-3d-data-66a0f6376afb.

3D Model | Solid Model v.s. Surface Model



Voxel

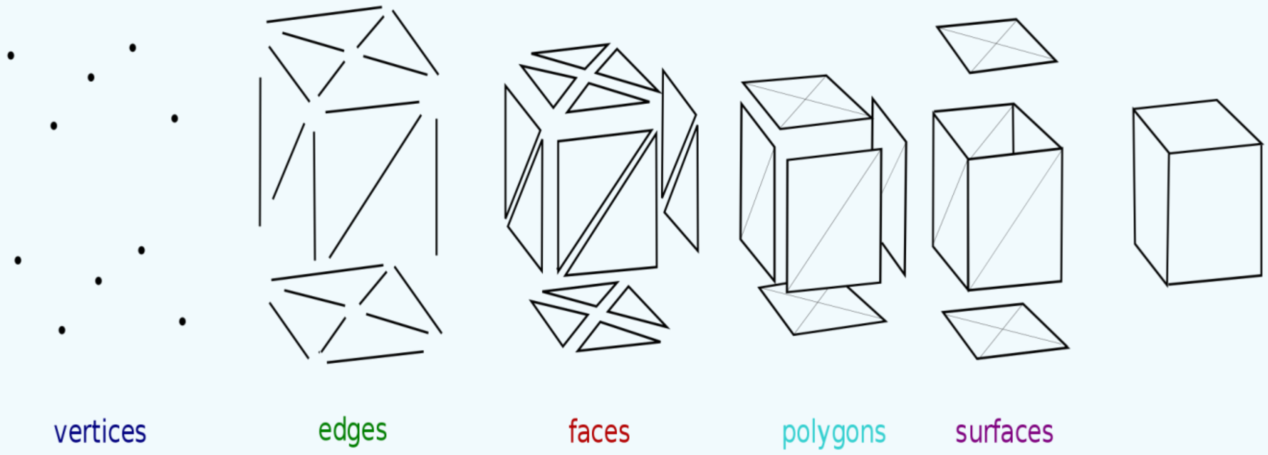


Point Cloud

Mesh

Mesh

A collection of **vertices**, **edges**, and **faces** that defines the surface of an object.



Source: https://en.wikipedia.org/wiki/Polygon_mesh

Mesh

A collection of **vertices**, **edges**, and **faces** that defines the surface of an object.

Element	Definition
Vertex	A point (usually in 3D) along with other attributes (e.g. colour, normal vector, texture coordinates).
Edge	A connection between two vertices.
Face	A closed set of edges • Triangle face : 3 edges • Quadrilateral face : 4 edges

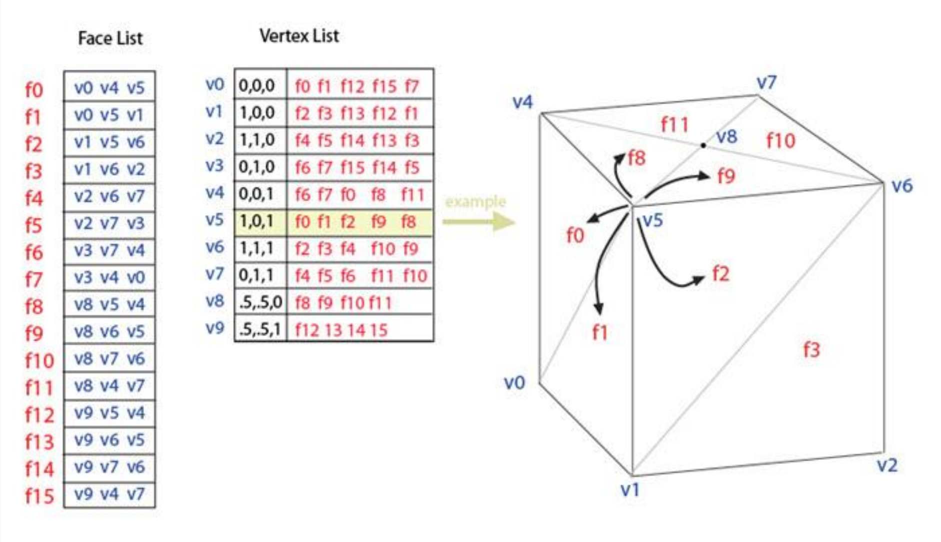
Source: https://en.wikipedia.org/wiki/Polygon_mesh

Mesh | Representations

Face-vertex meshes:

<u>Description (Part 1):</u>

- Represent an object as a set of faces and a set of vertices.
- Most widely used representation, with built-in support in modern graphics hardware.



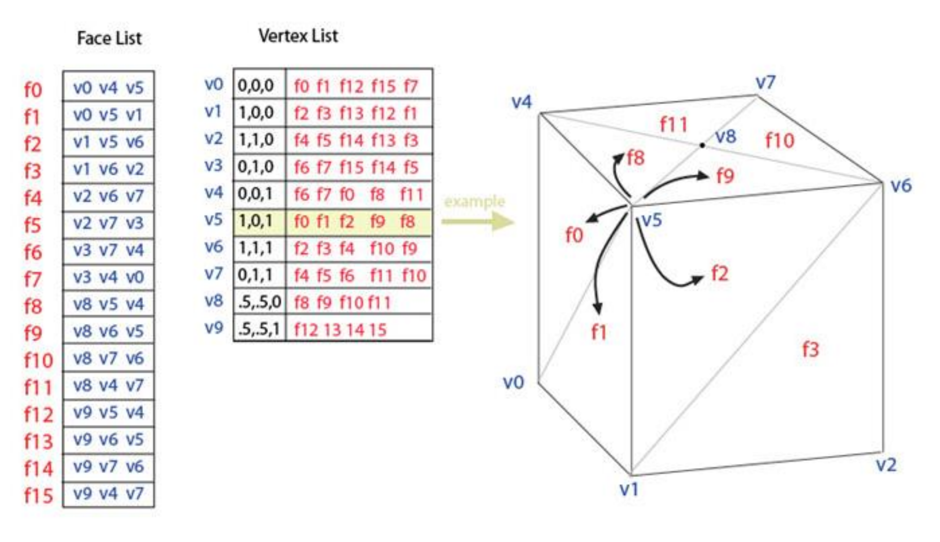
Source: https://en.wikipedia.org/wiki/Polygon_mesh

Mesh | Representations

Face-vertex meshes:

<u>Description (Part 2):</u>

- For **rendering**, the **<u>face list</u>** is usually transmitted to the GPU as **a set of indices to the vertices**, and the **<u>vertex list</u>** are sent to **include all attributes** (i.e. position/colour/normal).


Face List

f0	v0 v4 v5
f1	v0 v5 v1
f2	v1 v5 v6
f3	v1 v6 v2
f4	v2 v6 v7
f5	v2 v7 v3
f6	v3 v7 v4
f7	v3 v4 v0
f8	v8 v5 v4
f9	v8 v6 v5
f10	v8 v7 v6
f11	v8 v4 v7
f12	v9 v5 v4
f13	v9 v6 v5
f14	v9 v7 v6
f15	v9 v4 v7

Vertex List

v0	0,0,0	f0 f1 f12 f15 f7
v1	1,0,0	f2 f3 f13 f12 f1
v2	1,1,0	f4 f5 f14 f13 f3
v3	0,1,0	f6 f7 f15 f14 f5
v4	0,0,1	f6 f7 f0 f8 f11
v5	1,0,1	f0 f1 f2 f9 f8
v6	1,1,1	f2 f3 f4 f10 f9
v7	0,1,1	f4 f5 f6 f11 f10
v8	.5,.5,0	f8 f9 f10 f11
v9	.5,.5,1	f12 13 14 15

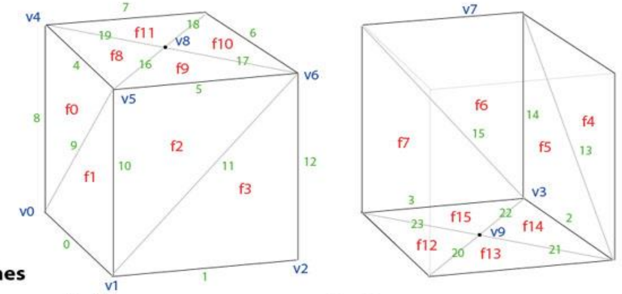
Source: https://en.wikipedia.org/wiki/Polygon_mesh

Mesh | Representations

Winged-Edge meshes:

<u>Description (Part 1):</u>

- Explicitly represent the **vertices**, **faces**, and **edges** of a mesh.
- Widely used in modelling software.



Winged-Edge Meshes

Face List

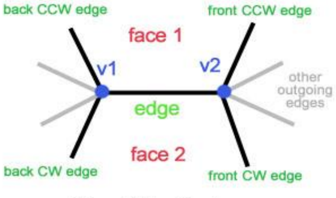
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



Winged Edge Structure

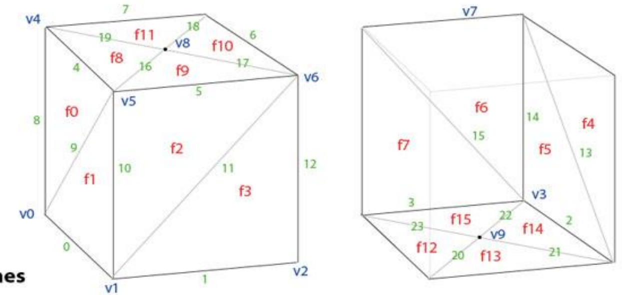
Source: https://en.wikipedia.org/wiki/Polygon_mesh

Mesh | Representations

Winged-Edge meshes:

<u>Description (Part 2):</u>

- **<u>Strength</u>**:
  Suited for dynamic geometry.
- **<u>Drawback</u>**:
  Large storage requirements and increases complexity.



Winged-Edge Meshes

Face List

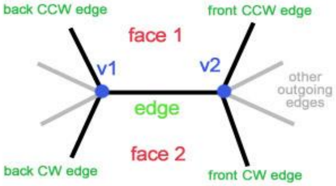
f0	4 8 9
f1	0 10 9
f2	5 10 11
f3	1 12 11
f4	6 12 13
f5	2 14 13
f6	7 14 15
f7	3 8 15
f8	4 16 19
f9	5 17 16
f10	6 18 17
f11	7 19 18
f12	0 23 20
f13	1 20 21
f14	2 21 22
f15	3 22 23

Edge List

e0	v0 v1	f1 f12	9 23 10 20
e1	v1 v2	f3 f13	11 20 12 21
e2	v2 v3	f5 f14	13 21 14 22
e3	v3 v0	f7 f15	15 22 8 23
e4	v4 v5	f0 f8	19 8 16 9
e5	v5 v6	f2 f9	16 10 17 11
e6	v6 v7	f4 f10	17 12 18 13
e7	v7 v4	f6 f11	18 14 19 15
e8	v0 v4	f7 f0	3 9 7 4
e9	v0 v5	f0 f1	8 0 4 10
e10	v1 v5	f1 f2	0 11 9 5
e11	v1 v6	f2 f3	10 1 5 12
e12	v2 v6	f3 f4	1 13 11 6
e13	v2 v7	f4 f5	12 2 6 14
e14	v3 v7	f5 f6	2 15 13 7
e15	v3 v4	f6 f7	14 3 7 15
e16	v5 v8	f8 f9	4 5 19 17
e17	v6 v8	f9 f10	5 6 16 18
e18	v7 v8	f10 f11	6 7 17 19
e19	v4 v8	f11 f8	7 4 18 16
e20	v1 v9	f12 f13	0 1 23 21
e21	v2 v9	f13 f14	1 2 20 22
e22	v3 v9	f14 f15	2 3 21 23
e23	v0 v9	f15 f12	3 0 22 20

Vertex List

v0	0,0,0	8 9 0 23 3
v1	1,0,0	10 11 1 20 0
v2	1,1,0	12 13 2 21 1
v3	0,1,0	14 15 3 22 2
v4	0,0,1	8 15 7 19 4
v5	1,0,1	10 9 4 16 5
v6	1,1,1	12 11 5 17 6
v7	0,1,1	14 13 6 18 7
v8	.5,.5,0	16 17 18 19
v9	.5,.5,1	20 21 22 23



Winged Edge Structure

Source: https://en.wikipedia.org/wiki/Polygon_mesh

File Formats

File Formats

Some of the many file formats for storing polygon mesh data:

File Suffix	Format Name	Description
.fbx	Autodesk Filmbox Format	• Proprietary. • ASCII + Binary encodings. • High interoperability between major third-party software. • **Data**: Geometry, appearance (colour and textures), skeletal animations, morphs. • **Industry**: Video game and film industry.
.blend	Blender File Format	• Open source, binary-only format. • **Data**: scenes, objects, materials, textures, sounds, images, post-production effects.
.dae	Digital Asset Exchange (COLLADA)	• A universal XML format designed to prevent incompatibility. • **Data**: Geometry, appearance (colour, material, textures), animations, kinematics, physics. • **Industry**: Video game and film industry.

File Formats

Some of the many file formats for storing polygon mesh data:

File Suffix	Format Name	Description
.obj	Wavefront OBJ	• ASCII + Binary encodings (Only ASCII encoding is open source). • **Data**: objects, colour, textures (in a separate MTL file) • Does not support animation. • Industry: 3D graphics, 3D printing.
.ply	Polygon File Format	• Binary + ASCII encodings • Designed by the Stanford University.
.stl	Stereolithography	• **Data**: Geometry (triangular mesh) • Does not support appearance, scene, and animations. • **Industry**: 3D printing, rapid prototyping, CAM.

Transformation (Part 1)

[Introduction]

Transformation

A transformation is a **function** that maps a point (or vector) into another point (or vector).

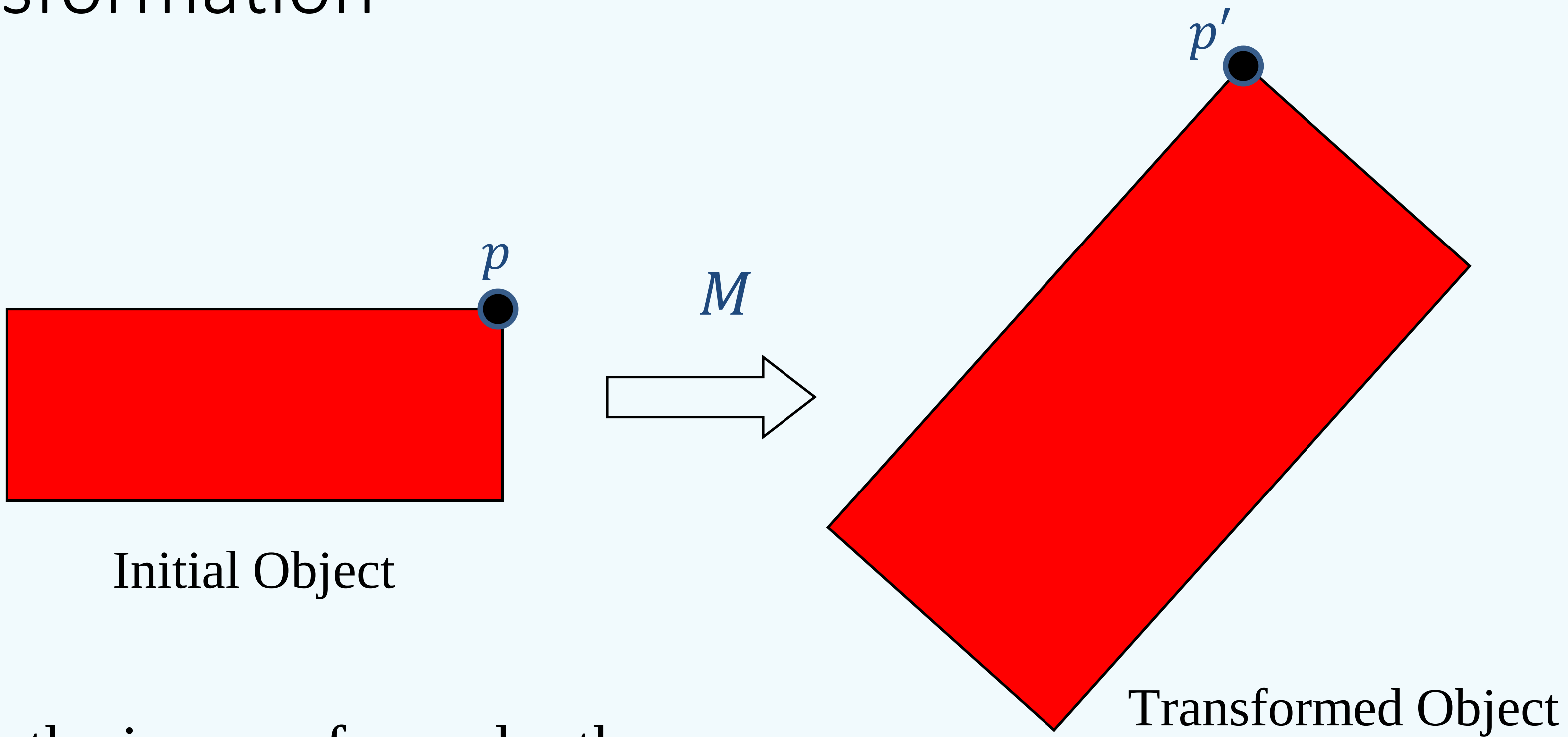
More formally, a transformation on $\mathbb{R}^n$ is any mapping

$$M: \mathbb{R}^n \mapsto \mathbb{R}^n$$

such that $p \in \mathbb{R}^n$ is mapped to a unique point, $M(p)$, also in $\mathbb{R}^n$

NOTE: $n = 2$ means 2D Transformation, $n = 3$ means 3D Transformation.

Transformation



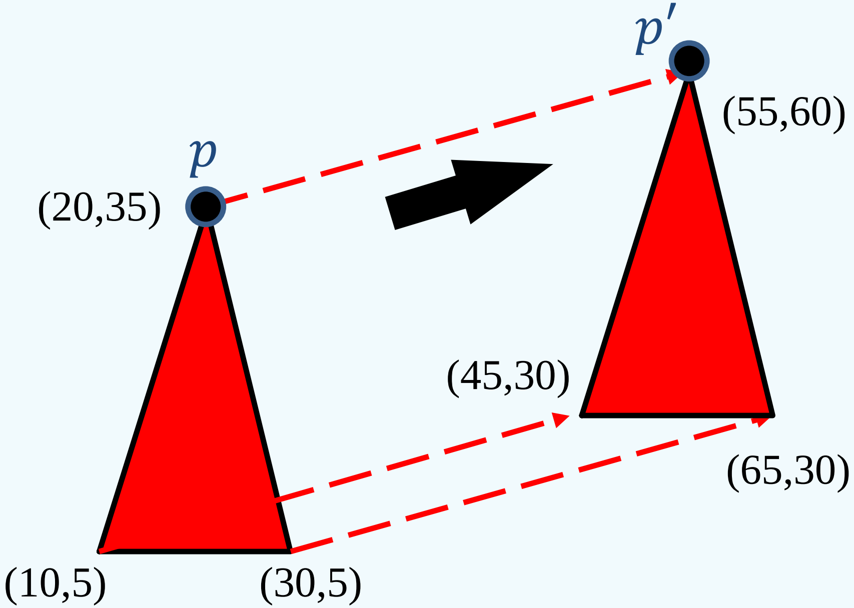
p' is the image of p under the transformation M.

Transformation (Part 1)

[2D Transformation]

2D Translation

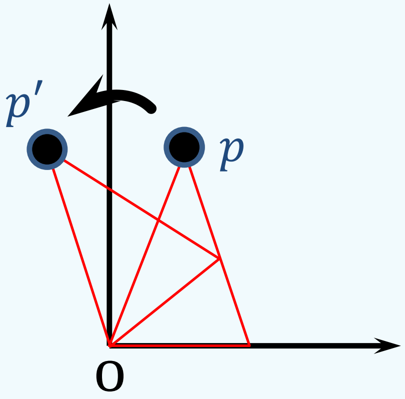
The process of moving a vector based on a translational vector.



$$p' = p + t$$

2D Rotation

- A circular movement of an object around the Z-axis.
- The direction of rotation can be determined using the right-hand rule.



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

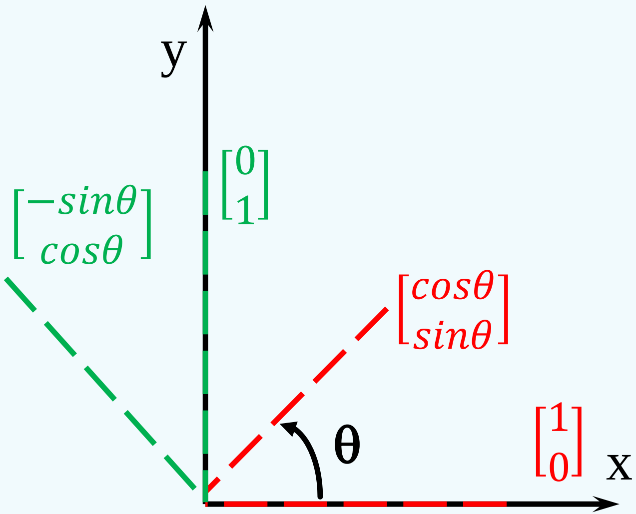
2D Rotation

Proof of $R_z(\theta)$

Let $R_z(\theta) = \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix}$.

$$R_z(\theta)\begin{bmatrix}1\\0\end{bmatrix} = \begin{bmatrix}cos\theta\\sin\theta\end{bmatrix}, R_z(\theta)\begin{bmatrix}0\\1\end{bmatrix} = \begin{bmatrix}-sin\theta\\cos\theta\end{bmatrix}$$

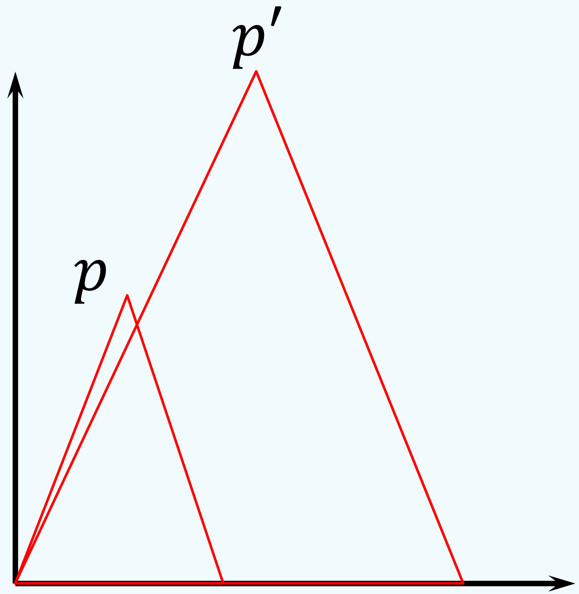
$$\begin{aligned} R_z(\theta) &= \begin{bmatrix} r_{11} & r_{12} \\ r_{21} & r_{22} \end{bmatrix} \\ &= R_z(\theta)I_2 = R_z(\theta)\begin{bmatrix}1 & 0\\0 & 1\end{bmatrix} \\ &= \begin{bmatrix} cos\theta & -sin\theta \\ sin\theta & cos\theta \end{bmatrix} \end{aligned}$$



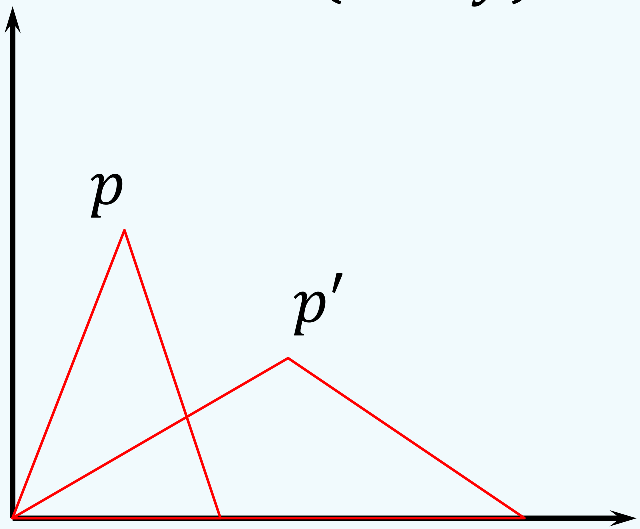
Definition of Identity Matrix and its images are taken from here.

2D Scaling

$$S(s_x, s_y) = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$



Uniform $s_x = s_y$

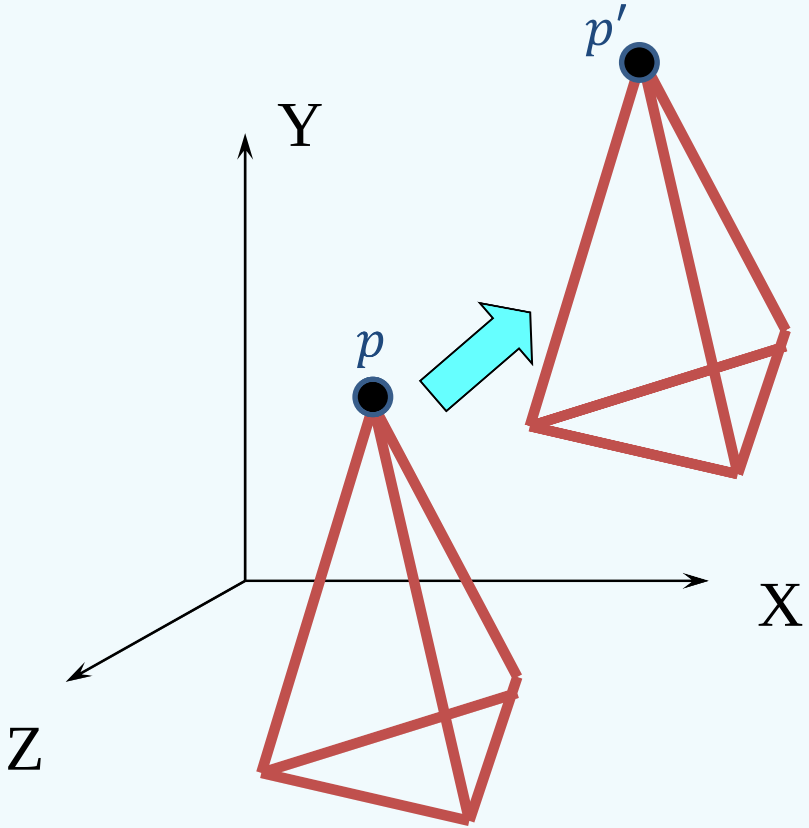


Non-Uniform $s_x \neq s_y$

Transformation (Part 1)

[3D Transformation]

3D Translation

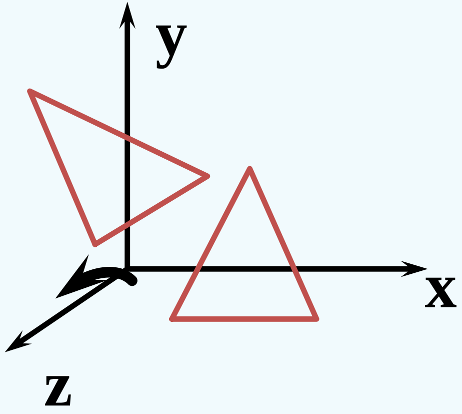


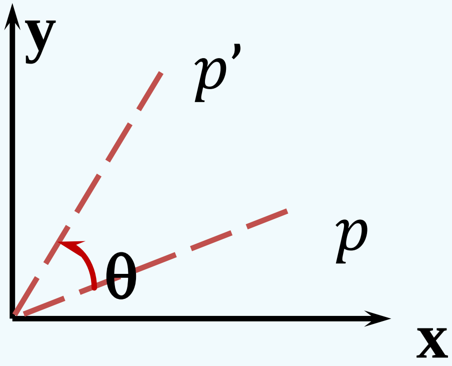
$$p' = p + t$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} + \begin{bmatrix} x_t \\ y_t \\ z_t \end{bmatrix}$$

3D Rotation

Rotation About the Z-Axis

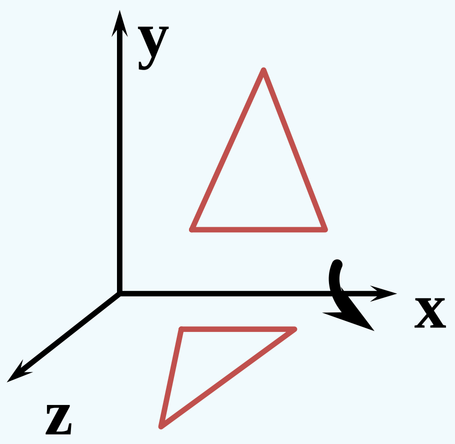


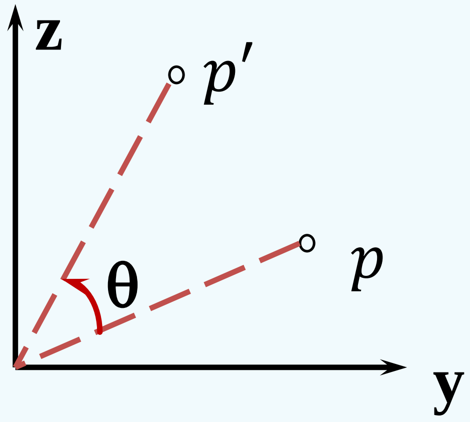


$$R_z(\theta) = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3D Rotation

Rotation About the X-Axis

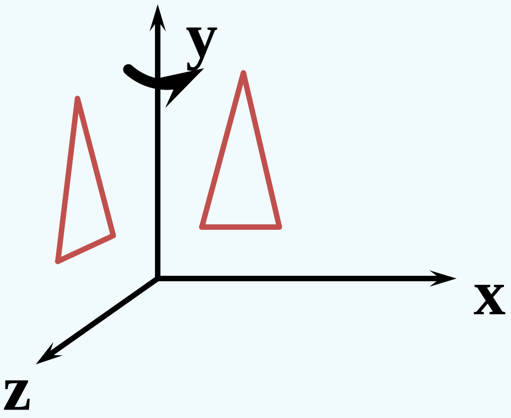


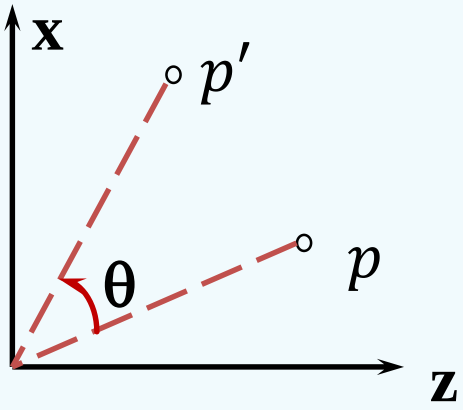


$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos\theta & -sin\theta \\ 0 & sin\theta & cos\theta \end{bmatrix}$$

3D Rotation

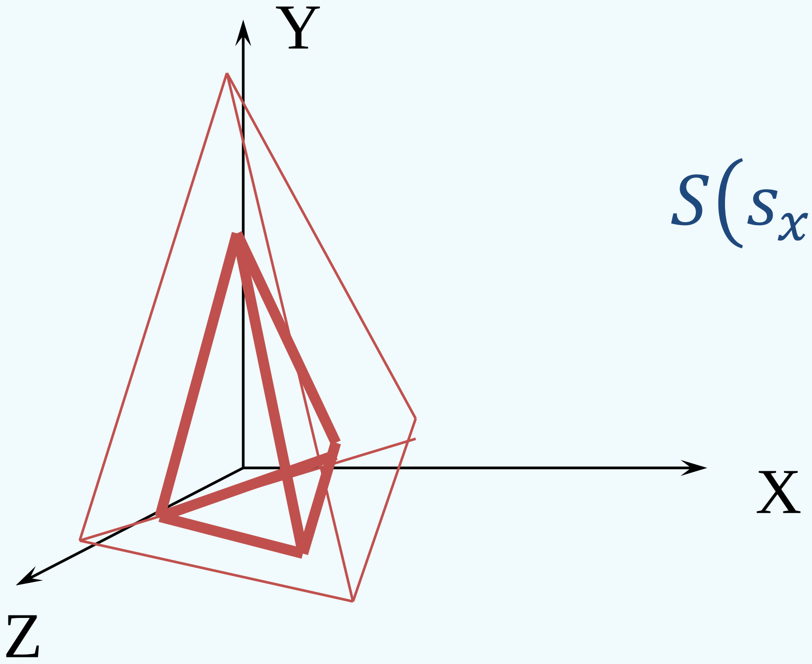
Rotation About the Y-Axis





$$R_y(\theta) = \begin{bmatrix} cos\theta & 0 & sin\theta \\ 0 & 1 & 0 \\ -sin\theta & 0 & cos\theta \end{bmatrix}$$

3D Scaling



$$S(s_x, s_y, s_z) = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & s_z \end{bmatrix}$$

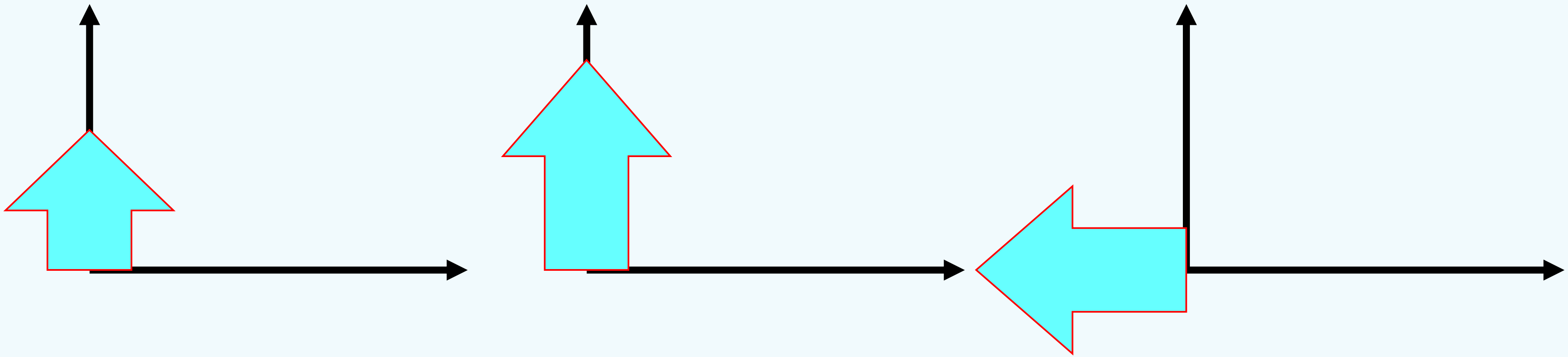
Transformation (Part 1)

[Concatenation of Transformation]

Concatenation of Transformation

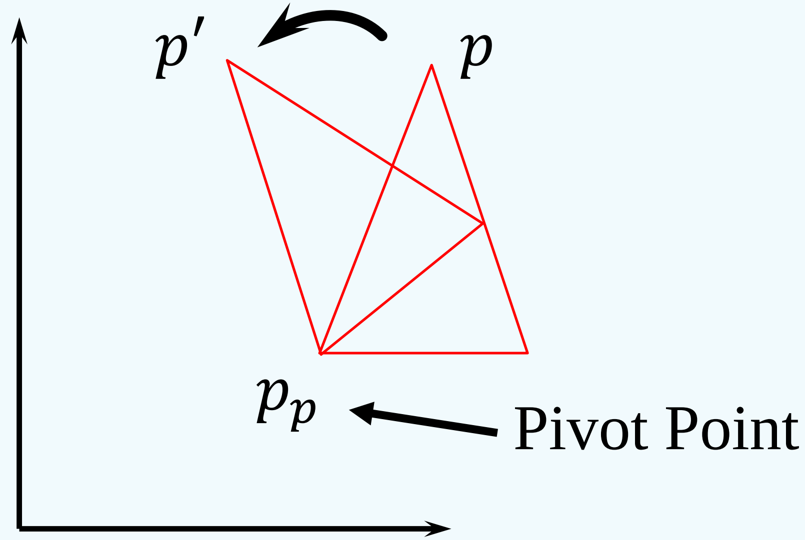
A series of transformations can be represented by the product of the corresponding individual transformation. For example:

$$p' = R_z(90°)S(1,1.5)p$$



Concatenation of Transformation

Special Example 1: Rotation About a Pivot Point

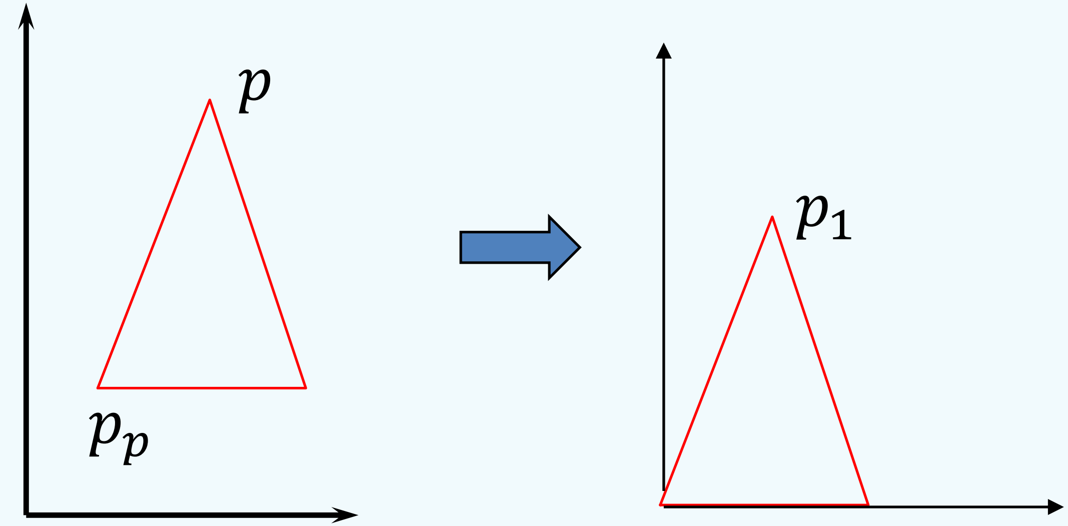


Description:

- Pivot point is the point of rotation
- Pivot point need not necessarily be on the object

Concatenation of Transformation

Special Example 1: Rotation About a Pivot Point



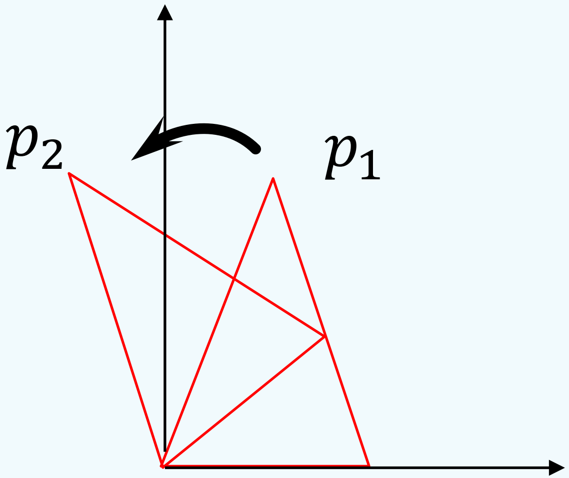
Step 1:

Translate the pivot point to the origin:

$$p_1 = p - p_p$$

Concatenation of Transformation

Special Example 1: Rotation About a Pivot Point



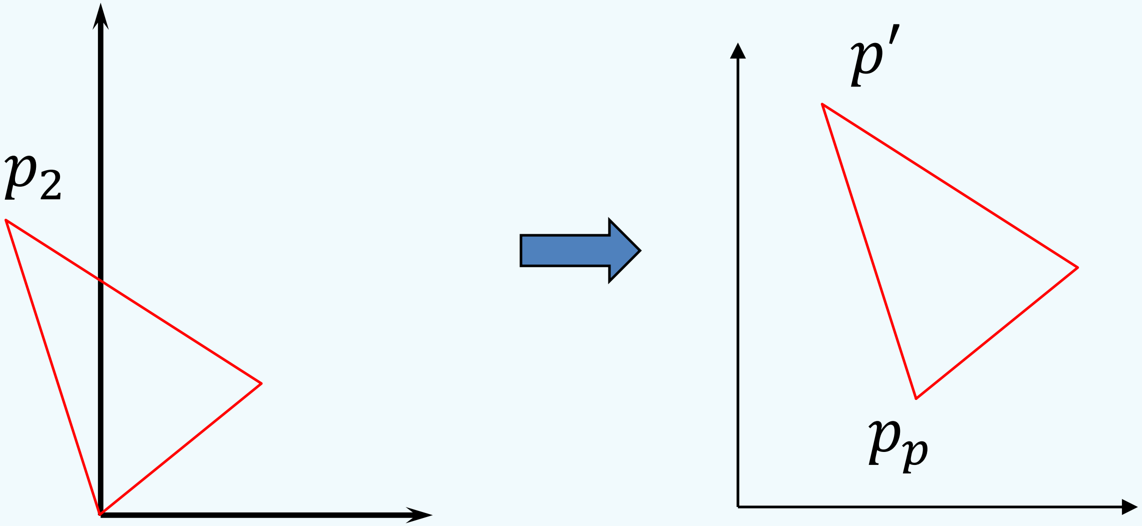
Step 2:

Rotate about the origin.

$$p_2 = R_z(\theta)p_1$$

Concatenation of Transformation

Special Example 1: Rotation About a Pivot Point



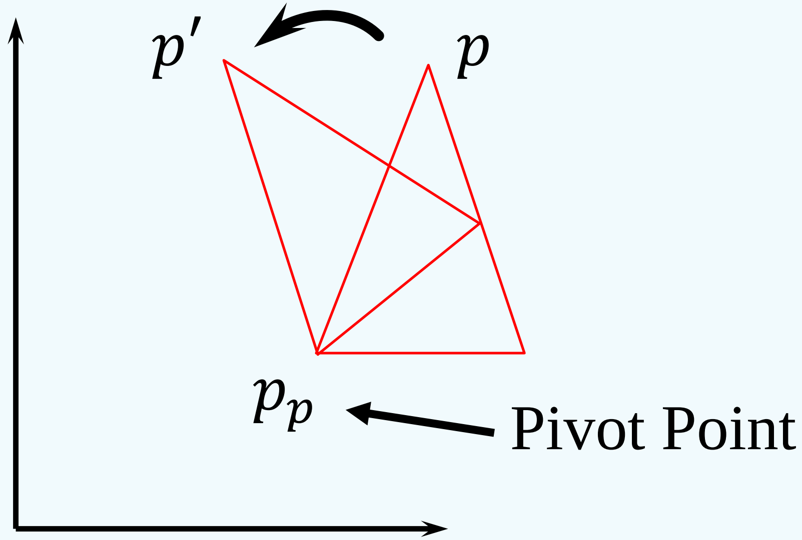
Step 3:

Translate the pivot point back to its original location.

$$p' = p_2 + p_p$$

Concatenation of Transformation

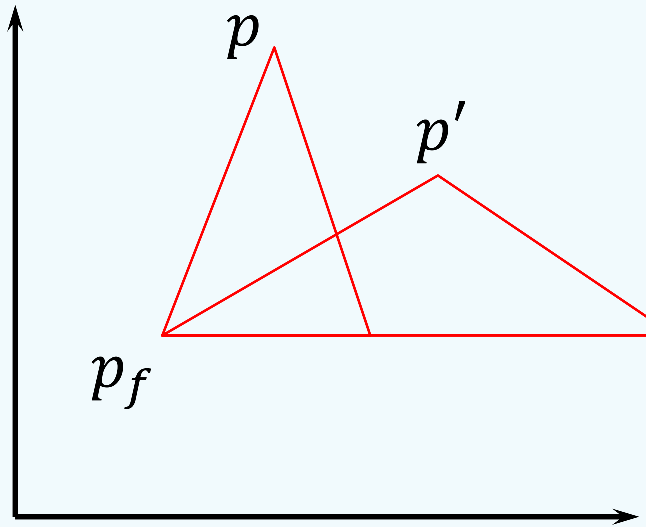
Special Example 1: Rotation About a Pivot Point



$$p' = p_2 + p_p$$
$$= R_z(\theta)p_1 + p_p$$
$$p' = R_z(\theta)\left[p - p_p\right] + p_p$$

Concatenation of Transformation

Special Example 2: Scaling About a Fixed Point

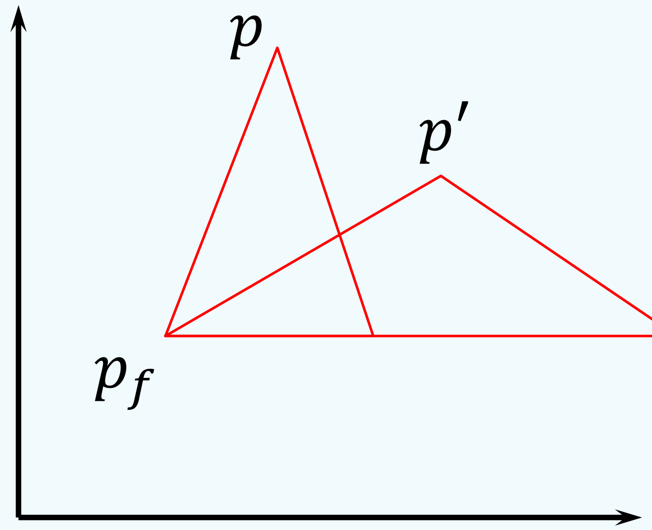


Steps:

1. Translate the fixed point (p_f) to origin.
2. Scale with respect to origin.
3. Translate the fixed point to its original position.

Concatenation of Transformation

Special Example 2: Scaling About a Fixed Point



$$p' = p_2 + p_f$$
$$= S(s_x, s_y)p_1 + p_f$$
$$p' = S(s_x, s_y)[p - p_f] + p_f$$

Q & A

Acknowledgement

- This presentation has been designed using resources from PoweredTemplate.com