การแปลงใน 2 มิติ
2D transformation

การแปลงใน 2 มิติ

- การแปลงของวัตถุใน 2 มิติคือ
  - การเปลี่ยนตำแหน่ง (การเลื่อน)
  - การเปลี่ยนขนาด (การย่อขยาย)
  - การเปลี่ยนมุม (การหมุน)
  - การเปลี่ยนรูปร่าง (การเฉือน)
- คำนวณได้จากการคูณเมตริกซ์

การแทนจุดใน 2 มิติ

- ใช้เวกเตอร์แนวตั้งแทนจุดใน 2 มิติ $\begin{vmatrix} x \\ y \end{vmatrix}$

- รูปการแปลงเชิงเส้น

$x' = ax + by + c$

หรือ $\begin{vmatrix} X' \\ Y' \\ 1 \end{vmatrix} = \begin{vmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$

$y' = dx + ey + f$

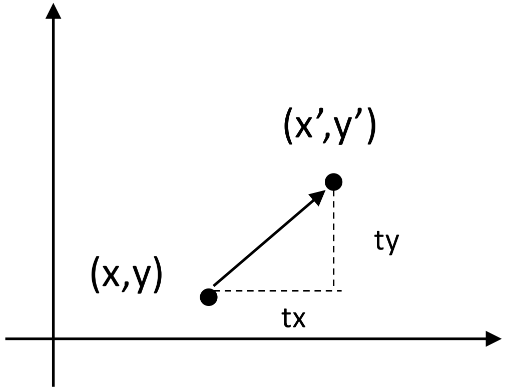
การเลื่อน

- เปลี่ยนตำแหน่งของจุดด้วยการเลื่อนบนเส้นตรง
- กำหนดจุด (x,y) และเวกเตอร์การเลื่อน (tx,ty)

จุดที่เกิดจากการเลื่อน : (x’, y’)

$$x' = x + tx$$

$$y' = y + ty$$



หรือ $P' = P + T$ เมื่อ $P' = \begin{vmatrix} x' \\ y' \end{vmatrix}$ $p = \begin{vmatrix} x \\ y \end{vmatrix}$ $T = \begin{vmatrix} tx \\ ty \end{vmatrix}$

การเลื่อนใน 2 มิติในรูป 3x3 เมตริกซ์

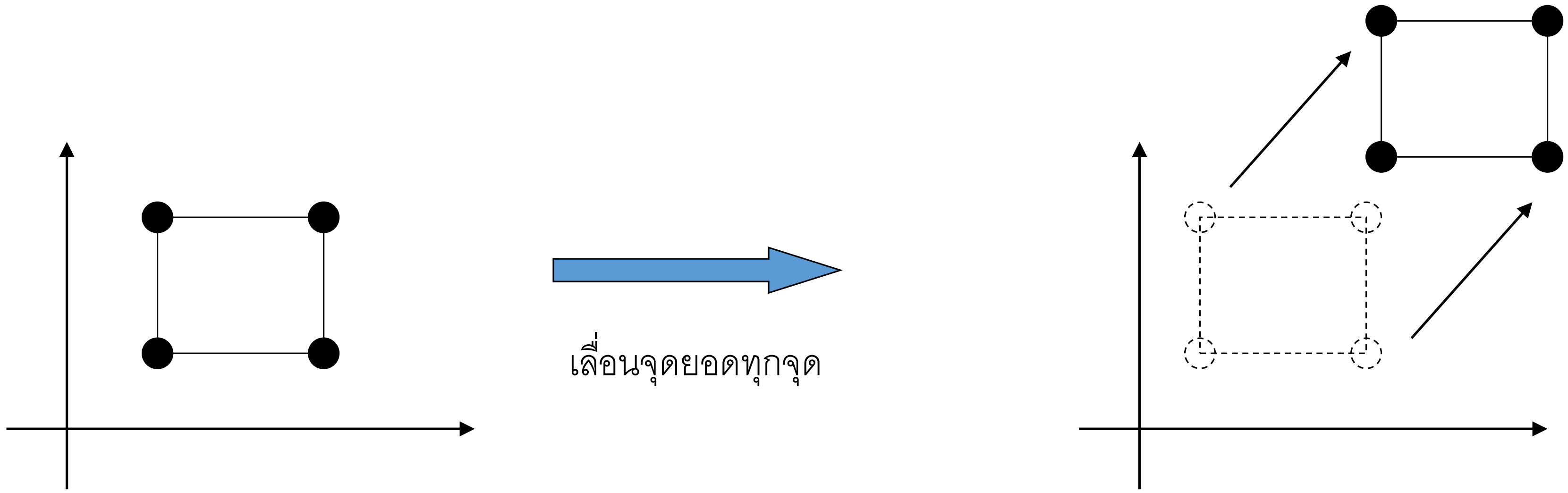
$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} x \\ y \end{vmatrix} + \begin{vmatrix} tx \\ ty \end{vmatrix}$$

$\Downarrow$ เวกเตอร์ 3 x 1

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

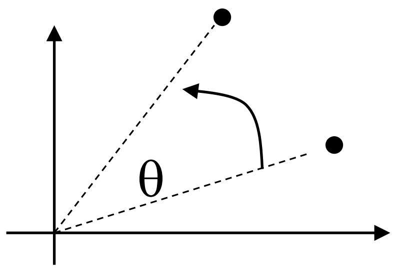
- อยู่ในรูป 3x3 เมตริกซ์คูณเวกเตอร์ 3x1

การเลื่อนวัตถุ

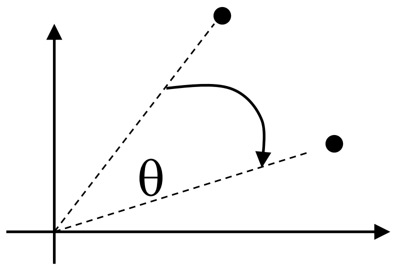


การหมุนใน 2 มิติ (2D rotation)

- หมุนรอบจุดกำเนิด (0,0)



$\theta > 0$: หมุนทวนเข็มนาฬิกา

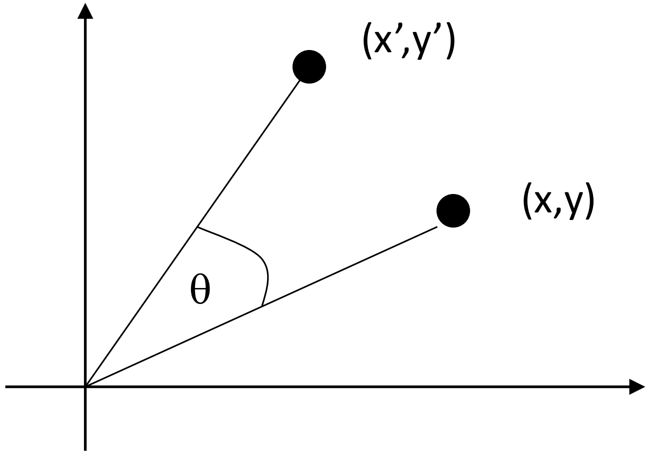


$\theta < 0$: หมุนตามเข็มนาฬิกา

การหมุนใน 2 มิติ

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

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



รูปการคูณเมตริกซ์

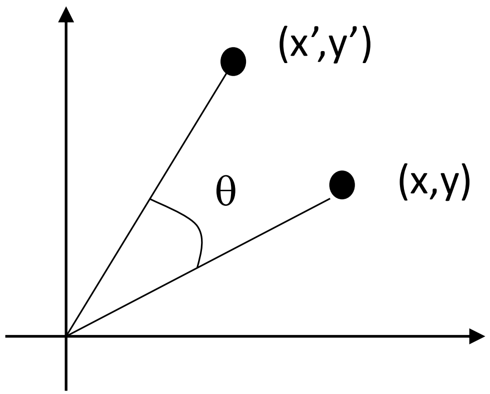
$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix}$$

รูป 3 x 3?

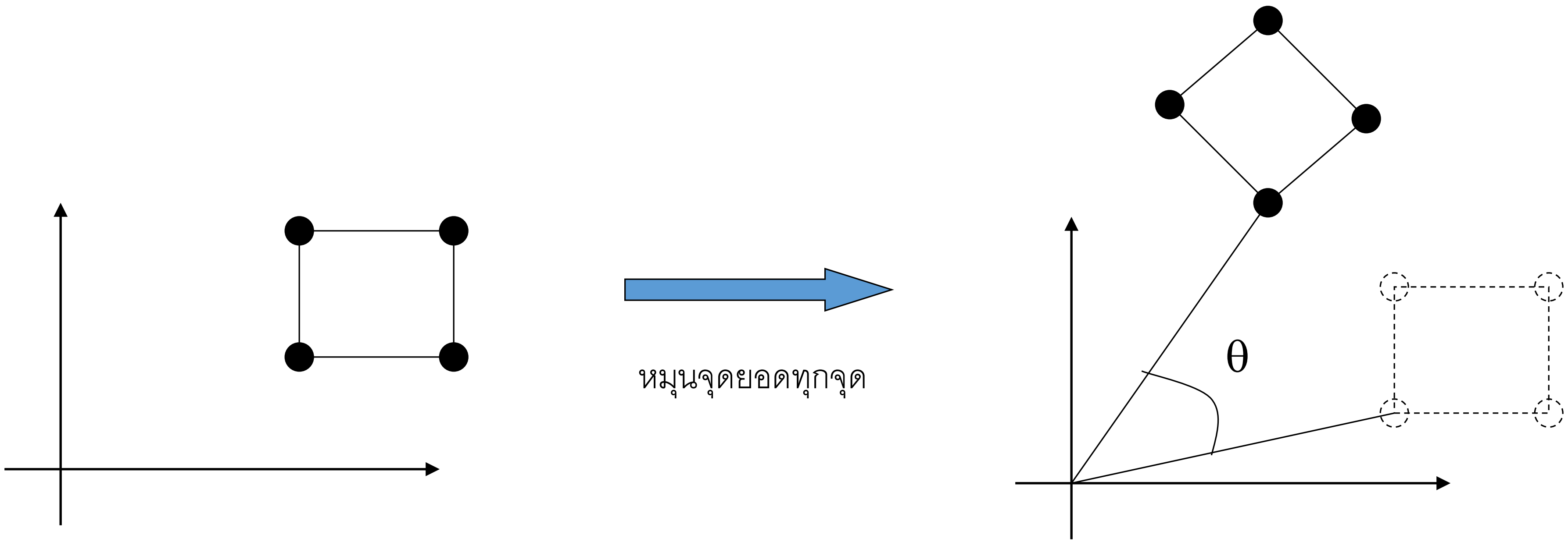
การหมุนใน 2 มิติในรูปเมตริกซ์ 3x3

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix}$$

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



การหมุนวัตถุ



การย่อขยายใน 2 มิติ (2D scaling)

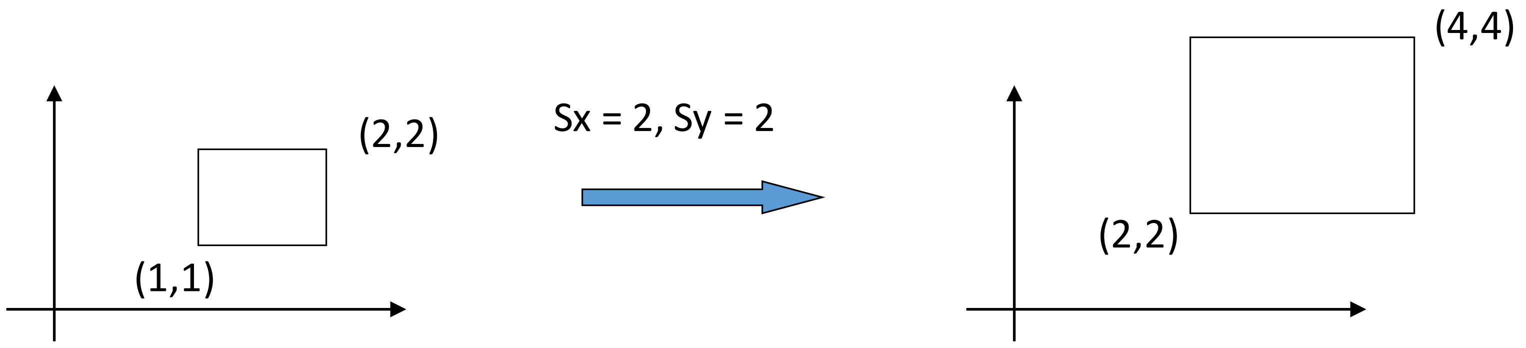
การย่อขยาย : เปลี่ยนขนาดของวัตถุด้วยตัวคูณ (Sx, Sy)

ค่าตัวคูณ $> 1 \rightarrow$ ขยาย

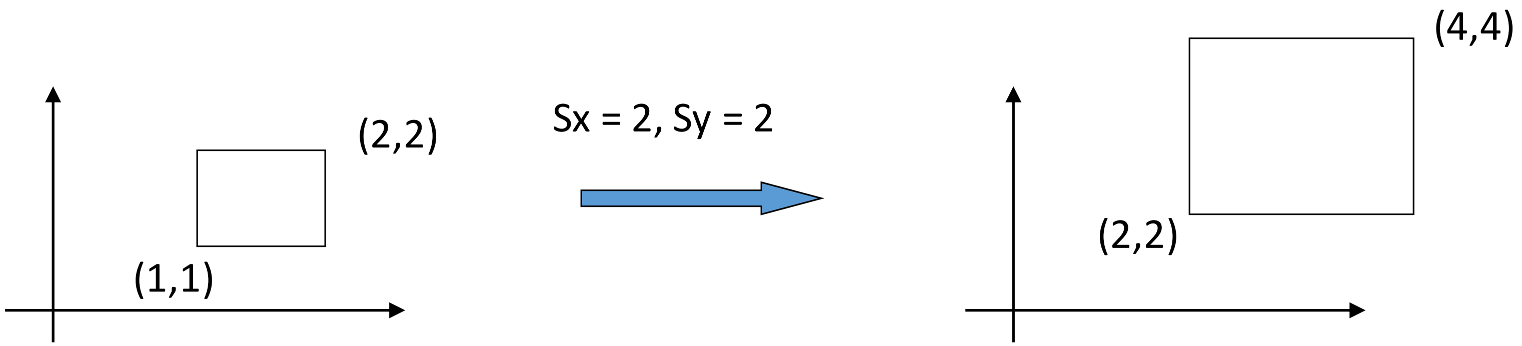
$0 <$ ค่าตัวคูณ $< 1 \rightarrow$ ย่อ

$$x' = x \,.\, Sx$$
$$y' = y \,.\, Sy$$

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} Sx & 0 \\ 0 & Sy \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix}$$



2D Scaling



- ขนาดเปลี่ยนแต่ตำแหน่งเปลี่ยนตามด้วย

การย่อขยายใน 2 มิติในรูปเมตริกซ์ 3x3

$$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} Sx & 0 \\ 0 & Sy \end{vmatrix} \begin{vmatrix} x \\ y \end{vmatrix}$$

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

สรุป

- การเลื่อน: $$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} x \\ y \end{vmatrix} + \begin{vmatrix} tx \\ ty \end{vmatrix}$$

- การหมุน: $$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{vmatrix} * \begin{vmatrix} x \\ y \end{vmatrix}$$

- การย่อขยาย: $$\begin{vmatrix} x' \\ y' \end{vmatrix} = \begin{vmatrix} Sx & 0 \\ 0 & Sy \end{vmatrix} * \begin{vmatrix} x \\ y \end{vmatrix}$$

การแปลงด้วยเมตริกซ์ 3x3

- การเลื่อน:

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

- การหมุน:

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

- การย่อขยาย:

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{vmatrix} * \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

ทำไมถึงใช้เมตริกซ์ 3x3

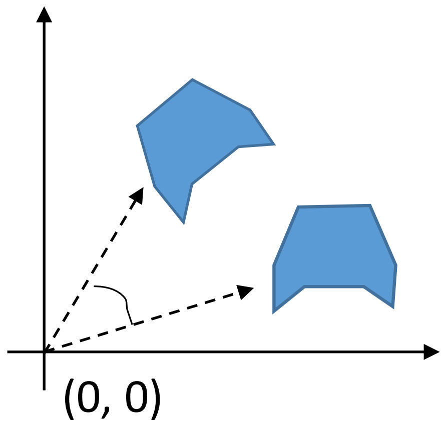
- สามารถคำนวณการแปลงทุกชนิดได้ด้วยการคูณเมตริกซ์
- สามารถคูณเมตริกซ์การแปลงทั้งหมดก่อนคูณเวกเตอร์จุด

- จุด (x,y) ต้องเพิ่ม 1 เข้ามาอีกแถว → (x,y,1)
  - พิกัดเอกพันธ์ (Homogeneous coordinates)

การหมุนใน 2 มิติ

- หมุนรอบจุดกำเนิด (0,0)

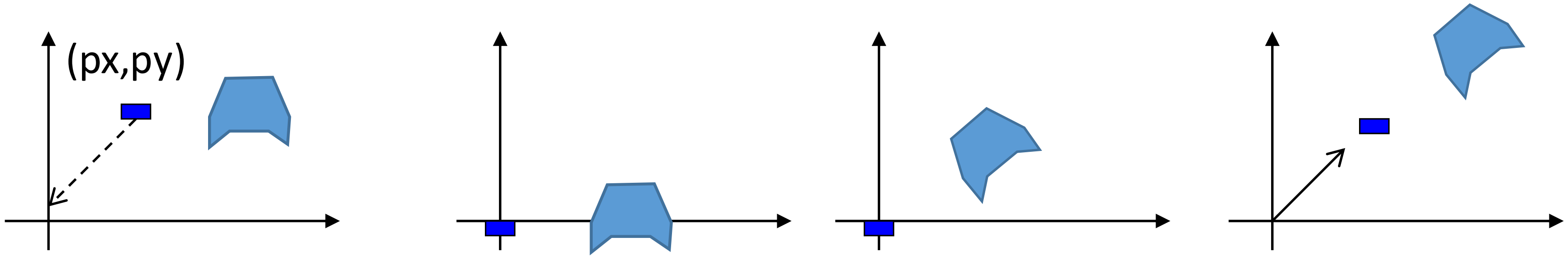
$$\begin{vmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{vmatrix}$$



- ถ้าจะหมุนรอบจุดอื่นทำอย่างไร?

การหมุนรอบจุดใดๆ

- หมุนรอบจุด P (px,py) ด้วยมุม θ:
  - เลื่อน P ไปที่จุดกำเนิด นั่นคือเลื่อนวัตถุไปด้วย : T(-px, -py)
  - หมุนวัตถุรอบจุดกำเนิดด้วยมุม θ : R(θ)
  - เลื่อน P และวัตถุกลับไปที่เดิม : T(px,py)



การหมุนรอบจุดใดๆ

เลื่อน P ไปที่จุดกำเนิด นั่นคือเลื่อนวัตถุไปด้วย : T(-px, -py)
หมุนวัตถุรอบจุดกำเนิดด้วยมุม θ : R(θ)
เลื่อน P และวัตถุกลับไปที่เดิม : T(px,py)

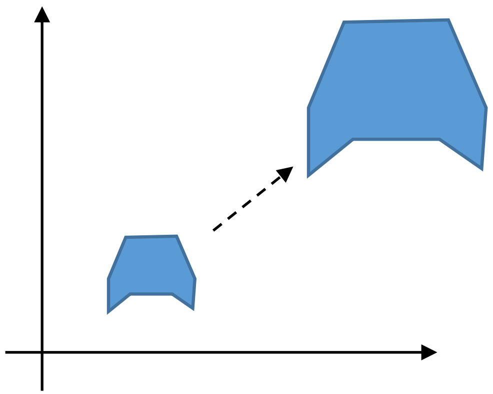
- จัดอยู่ในรูปการคูณเมตริกซ์ : T(px,py) R(θ) T(-px, -py) * P

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} 1 & 0 & px \\ 0 & 1 & py \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} 1 & 0 & -px \\ 0 & 1 & -py \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

การย่อขยายรอบจุดใดๆ

- การย่อขยายปกติมีหมุด (pivot) อยู่ที่จุดกำเนิด (0,0)

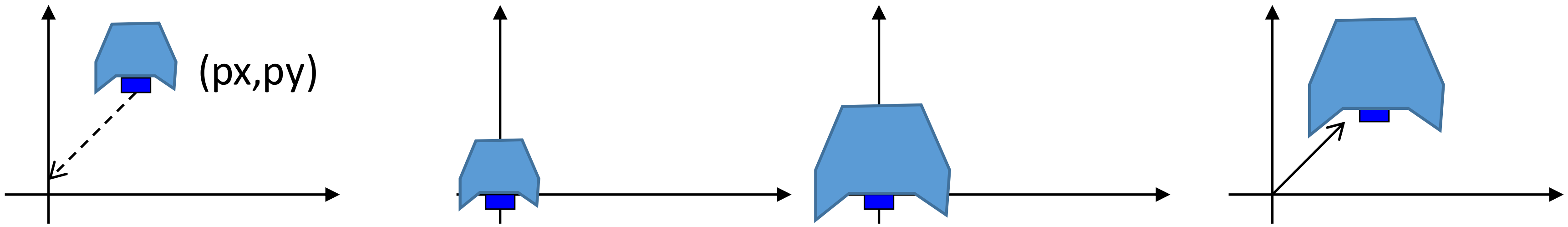
$$\begin{vmatrix} Sx & 0 & 0 \\ 0 & Sy & 0 \\ 0 & 0 & 1 \end{vmatrix}$$



- ถ้าจะย่อยขยายเทียบหมุดอื่นทำอย่างไร?

การย่อขยายรอบจุดใดๆ

- การย่อขยายเทียบหมุด P (px,py):
  - เลื่อน P ไปที่จุดกำเนิด นั่นคือเลื่อนวัตถุไปด้วย : T(-px, -py)
  - ย่อขยายวัตถุ : S(sx, sy)
  - เลื่อน P และวัตถุกลับไปที่เดิม : T(px,py)



การแปลงสัมพรรค (affine transform)

- คือการแปลงที่จุดผลลัพธ์ P’ เกิดจากผลรวมเชิงเส้นของจุดตั้งต้น P

$$\begin{vmatrix} x' \\ y' \\ 1 \end{vmatrix} = \begin{vmatrix} m11 & m12 & m13 \\ m21 & m22 & m23 \\ 0 & 0 & 1 \end{vmatrix} \begin{vmatrix} x \\ y \\ 1 \end{vmatrix}$$

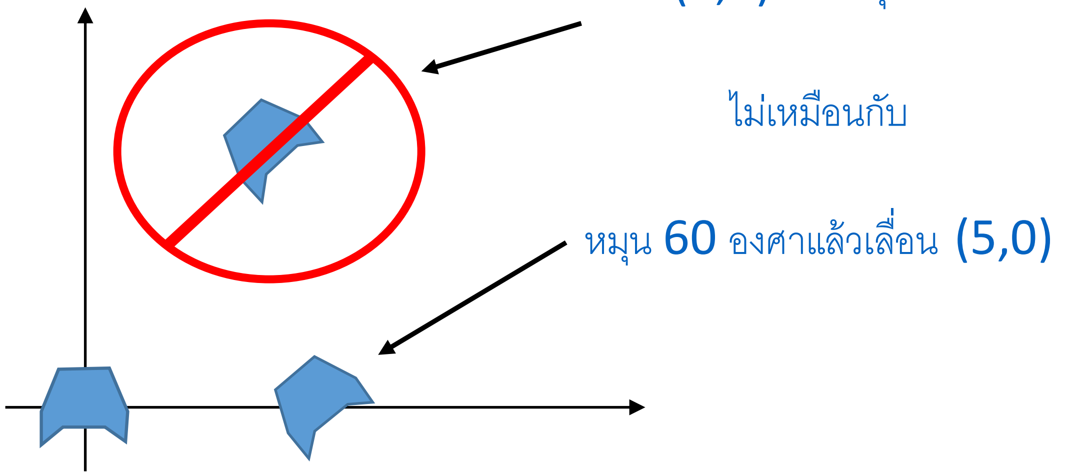
- การเลื่อน การหมุน การย่อขยายล้วนเป็นการแปลงสัมพรรคทั้งสิ้น
- การแปลงสัมพรรคใดๆ สามารถแบ่งเป็นการหมุนต่อด้วยการย่อขยายต่อด้วยการเลื่อนได้
    - A = T * S * R

การรวมการแปลง

- การแปลงจุด P ด้วย M_1 ตามด้วย M_2 ตามด้วย M_3
  - $P' = (M_3 * (M_2 * (M_1 * P)))$
  - $P' = (M_3 * M_2 * M_1) * P$
- การคูณเมตริกซ์จัดกลุ่มได้
  - $(M_3 * M_2 * M_1) = (M_3 * M_2) * M_1 = M_3 * (M_2 * M_1)$
- สลับที่ไม่ได้ $M_2 * M_1$ อาจจะไม่เท่ากับ $M_1 * M_2$
  - มีบางกรณีที่เท่าเช่น
    - M_2 เป็นการเลื่อนและ M_1 เป็นการเลื่อน
    - M_2 เป็นการหมุนและ M_1 เป็นการหมุน
    - ฯลฯ

ลำดับการแปลง

- หมุนก่อนเลื่อนและเลื่อนก่อนหมุนไม่เหมือนกัน

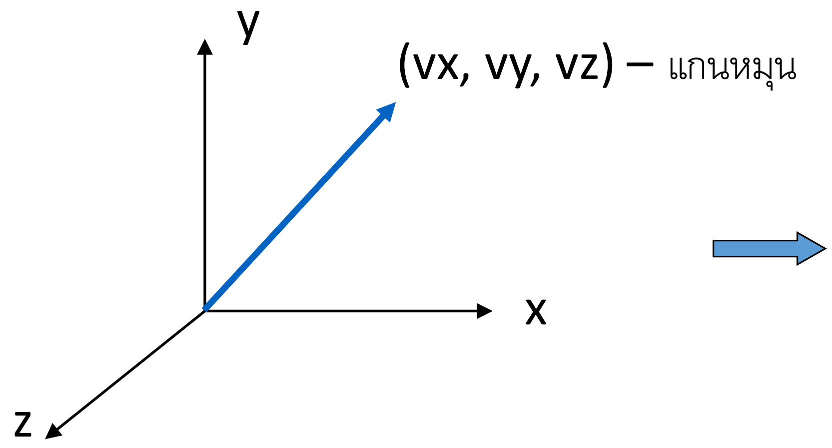


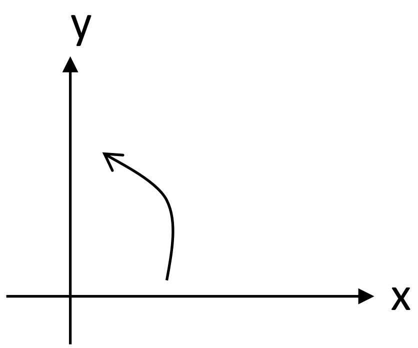
การเลื่อนใน OpenGL

- ฟังก์ชันการแปลงใน OpenGL ใช้ในการแปลง 3 มิติ
- นำมาใช้ใน 2 มิติได้โดยมองค่า z เป็น 0
- การเลื่อน :
  - glTranslatef(tx, ty, tz) -> glTranslateftx,ty,0) สำหรับ 2 มิติ

การหมุนใน OpenGL

- การหมุน:
  - glRotatef(angle, vx, vy, vz) -> glRotatef(angle, 0,0,1) สำหรับ 2 มิติ





แกนหมุนคือแกน z วิ่งเข้าหาตัวเรา

เมตริกซ์การแปลงใน OpenGL

- ใช้วิธีย้ายตำแหน่งการวาด M ด้วยการตั้งโหมด GL_MODELVIEW
  glMatrixMode(GL_MODELVIEW)
- กลับตำแหน่งการวาดมาที่จุดกำเนิด

  glLoadIdentity()

  -> M = 1 0 0

  0 1 0

  0 0 1

เมตริกซ์การแปลงใน OpenGL

- การแปลงในโหมด GL_MODELVIEW จะเป็นการคูณเมตริกซ์การแปลงต่อท้าย
- เช่น

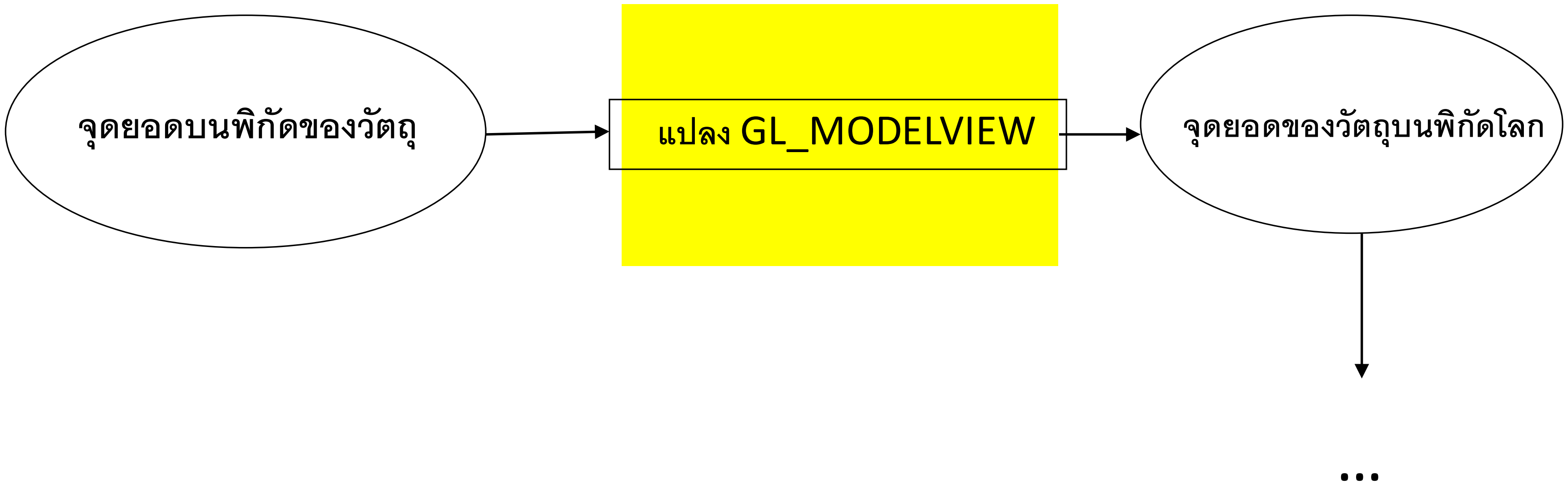
glTranslated(1,1 0);

$$M = M \times \begin{vmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{vmatrix}$$

- ทุกการวาดจุด P ภายใต้โหมด GL_MODELVIEW จะผ่านการแปลง M

$$P' = M \times P$$

ขั้นตอนการแปลงใน OpenGL



ข้อควรระวังสำหรับการแปลงใน OpenGL

- เมื่อสั่งให้มีการแปลงการแปลงนั้นจะถูกคูณต่อท้าย !!!!

$M = M \times M_{new}$

- ตัวอย่าง : เลื่อนแล้วหมุน

0) M = I (เมตริกซ์เอกลักษณ์)

1) เลื่อน $T(tx,ty,0)$ -> $M = M \times T(tx,ty,0)$

2) หมุน $R(\theta)$ -> $M = M \times R(\theta)$

3) แปลงจุด P -> $P' = M \times P$

$= T(tx, ty, 0) \times R(\theta) \times P$

สิ่งที่เคยเข้าใจคือ $P' = R(\theta) \times T(tx, ty, 0) \times P$ ไม่เท่ากัน !!!!

จัดลำดับความคิดใหม่

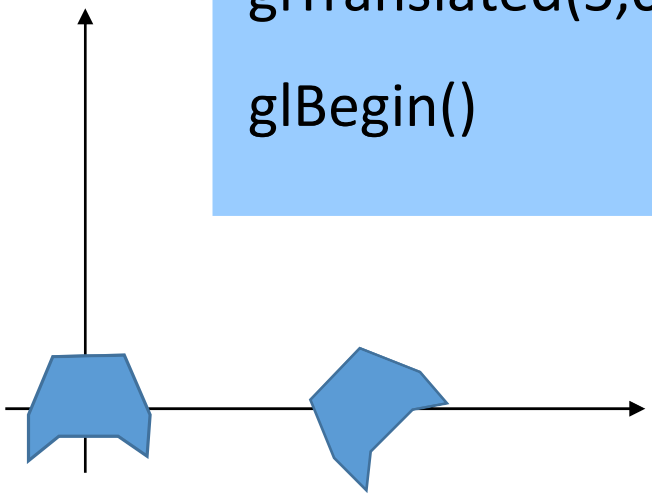
- ต้องการหมุนแล้วค่อยเลื่อน

```
glRotated(60,0,0,1);

glTranslated(5,0,0);

glBegin()
```

```
glTranslated(5,0,0);
glRotate(60,0,0,1);
glBegin()
…
```



จัดลำดับความคิดใหม่

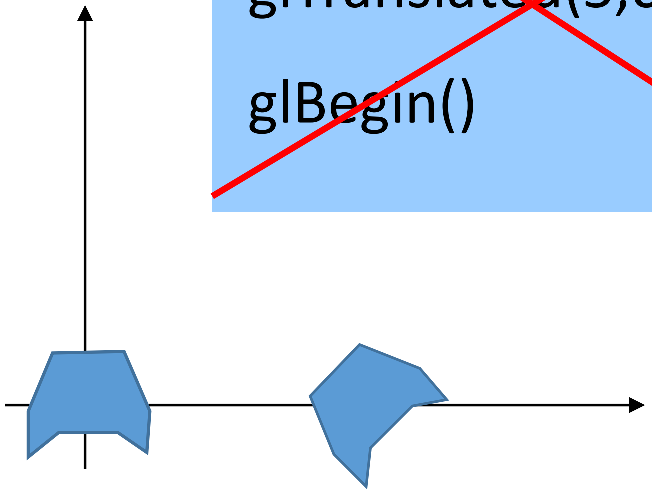
- ต้องการหมุนแล้วค่อยเลื่อน

~~glRotated(60,0,0,1);~~

~~glTranslated(5,0,0);~~

~~glBegin()~~

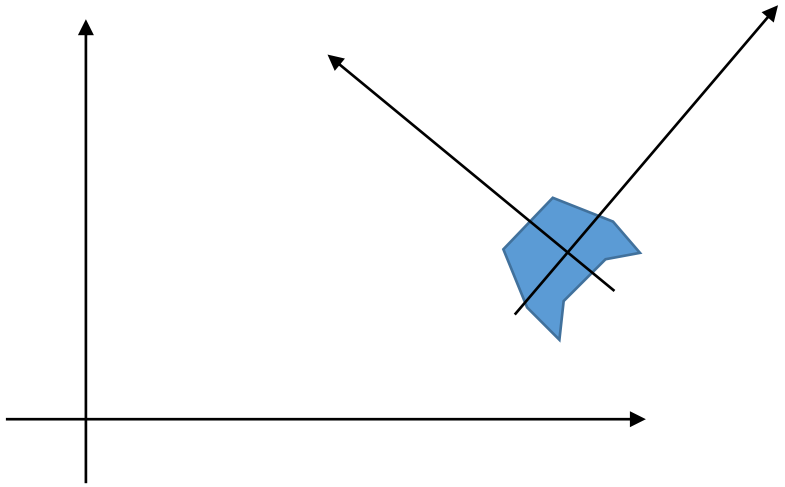
```
glTranslated(5,0,0);
glRotate(60,0,0,1);
glBegin()
…
```



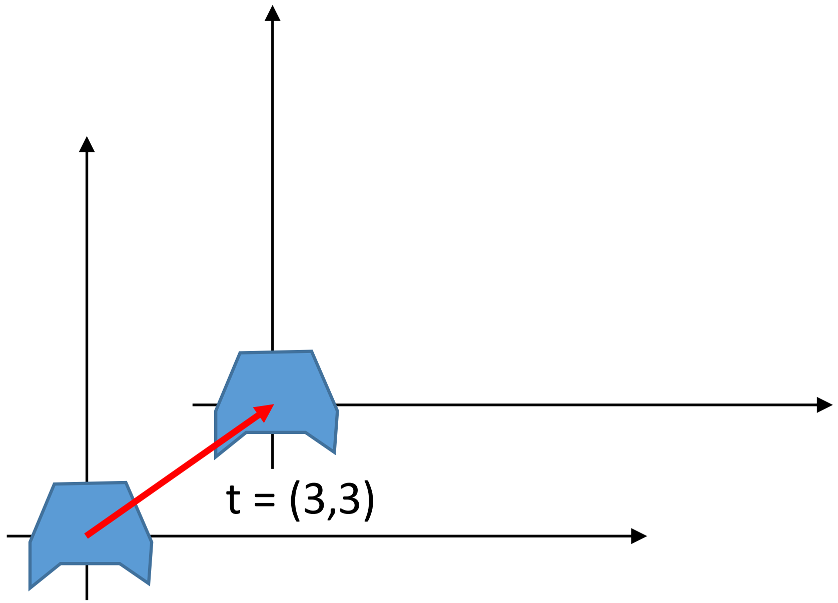
กำหนดลำดับให้สลับกันคือเลื่อนก่อนแล้วค่อยหมุน

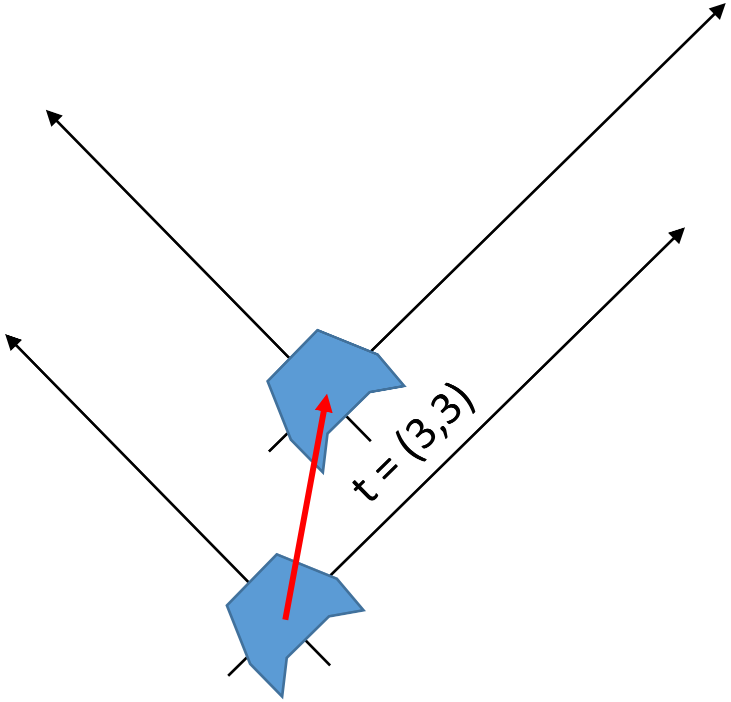
การมองพิกัดตัวเอง

- ก่อนหน้านี้เรามองว่าการแปลงทั้งหมดอ้างอิงกับจุดกำเนิด
- ลำดับการแปลงแบบ OpenGL เป็นการมองการแปลงเทียบกับตัวเอง
  - จุดกำเนิดของตัวเอง
  - แกนของตัวเอง
  - การแปลงแกนตัวเองเทียบกับตัวเอง

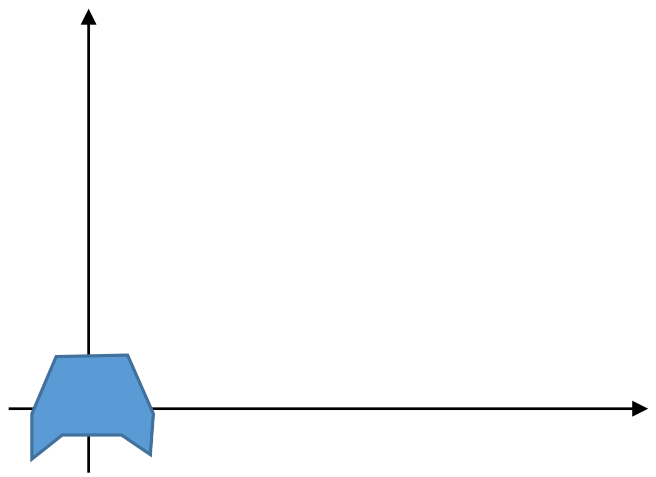


การเลื่อนเทียบแกนตัวเอง

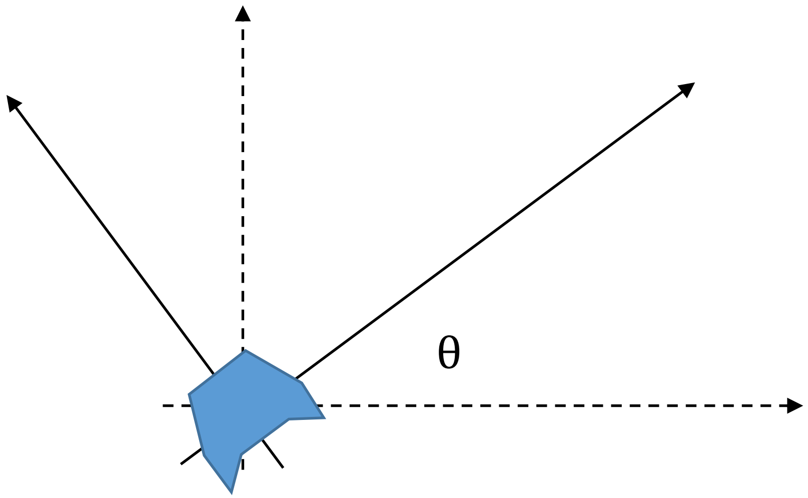




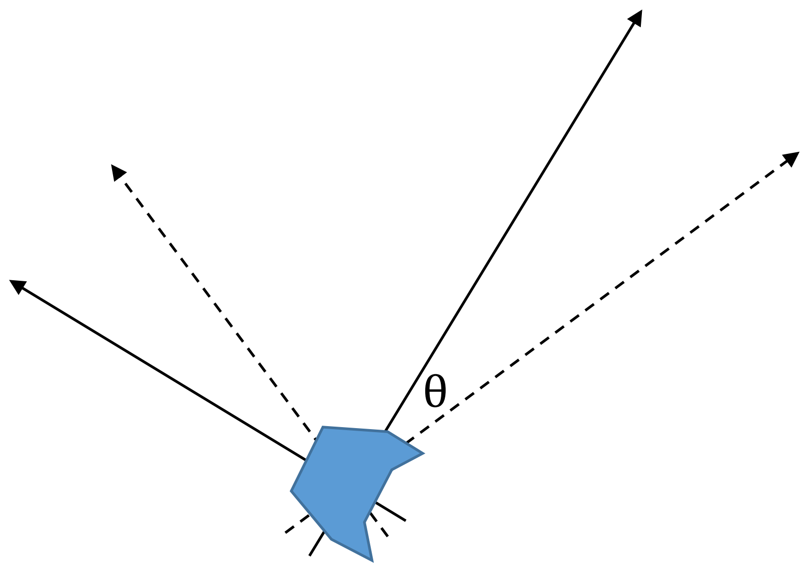
การหมุนเทียบแกนตัวเอง



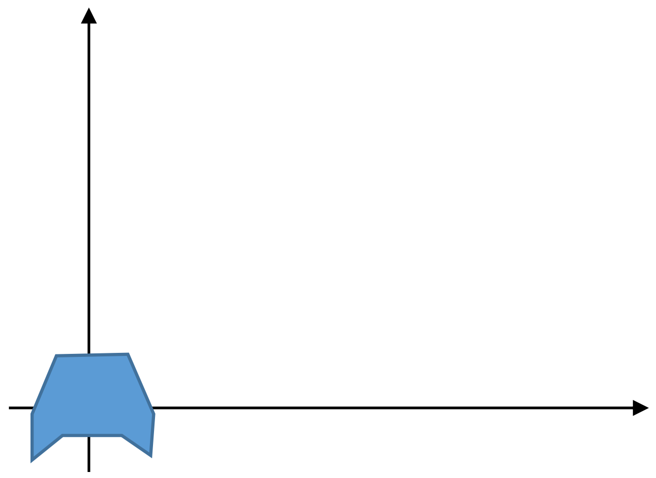
การหมุนเทียบแกนตัวเอง



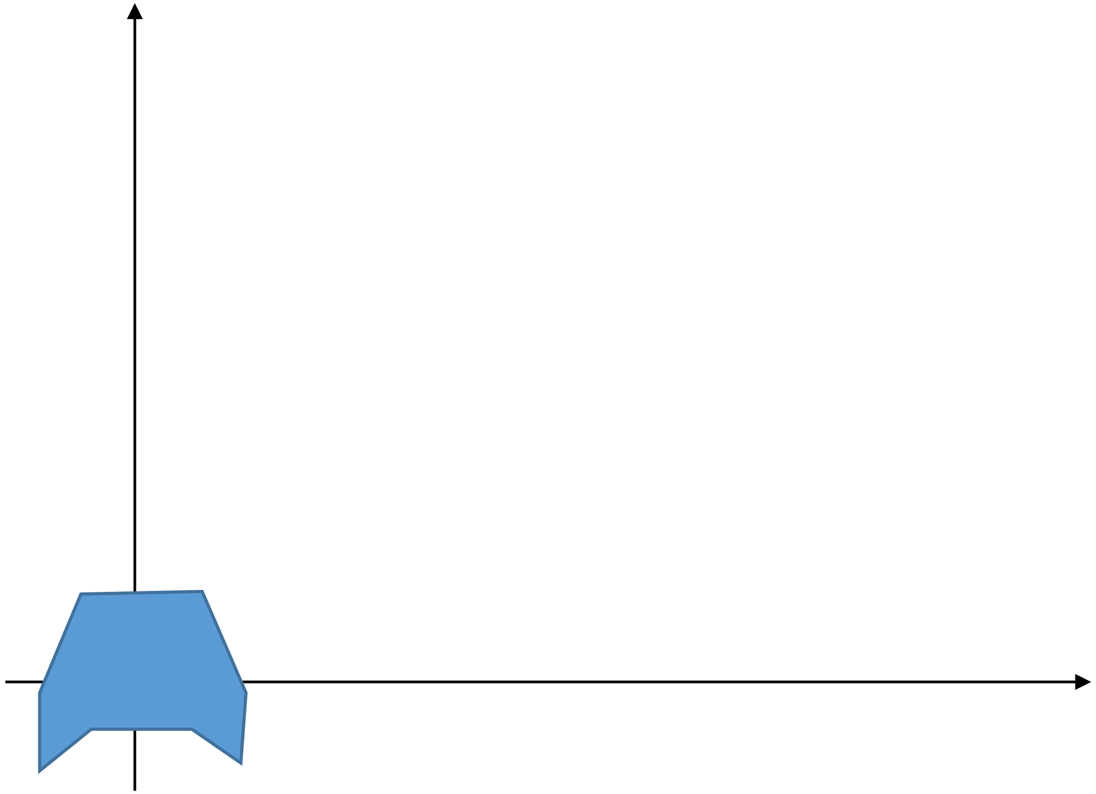
การหมุนเทียบแกนตัวเอง



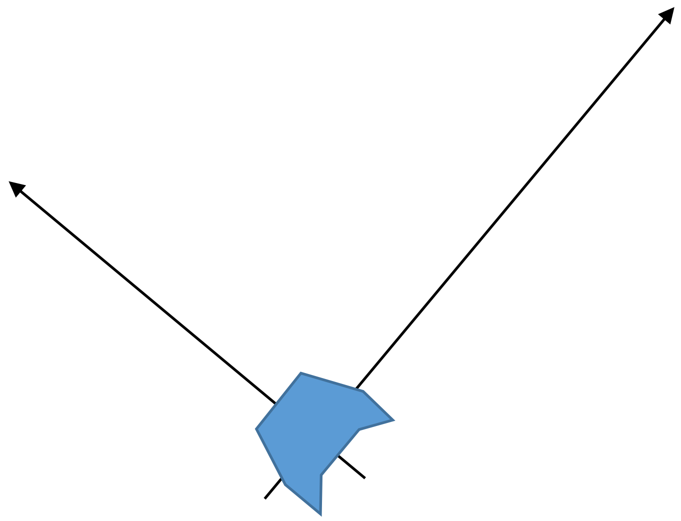
การย่อขยายเทียบแกนตัวเอง



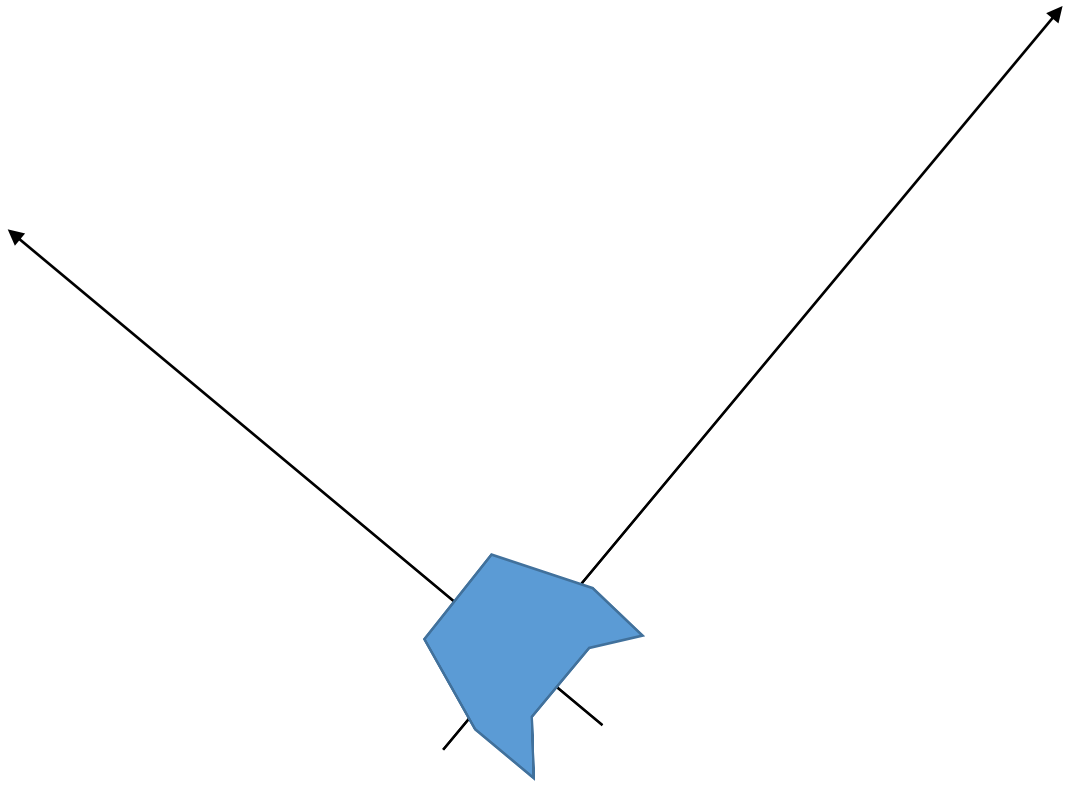
การย่อขยายเทียบแกนตัวเอง



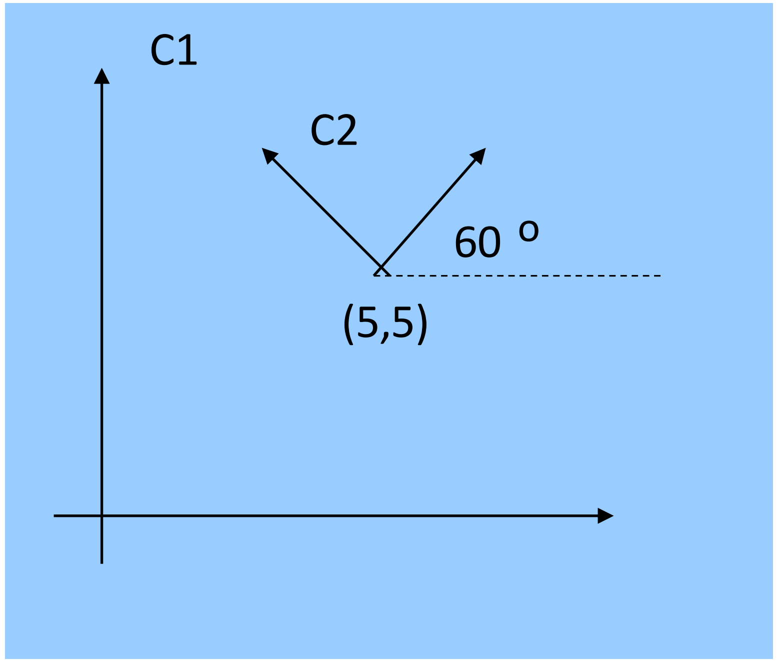
การย่อขยายเทียบแกนตัวเอง



การย่อขยายเทียบแกนตัวเอง



การรวมการแปลง



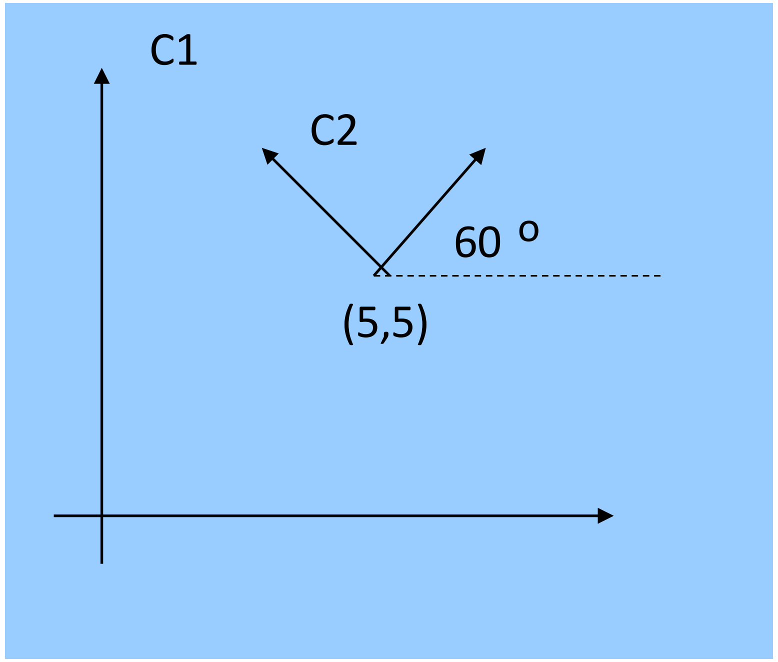
แปลงแกน C1 ไปเป็นแกน C2 ได้อย่างไร?

เลื่อน (5,5) แล้วหมุน (60)

หรือ

หมุน (60) แล้วเลื่อน (5,5) ???

การรวมการแปลง



แปลงแกน C1 ไปเป็นแกน C2 ได้อย่างไร?

เลื่อน (5,5) แล้วหมุน (60)

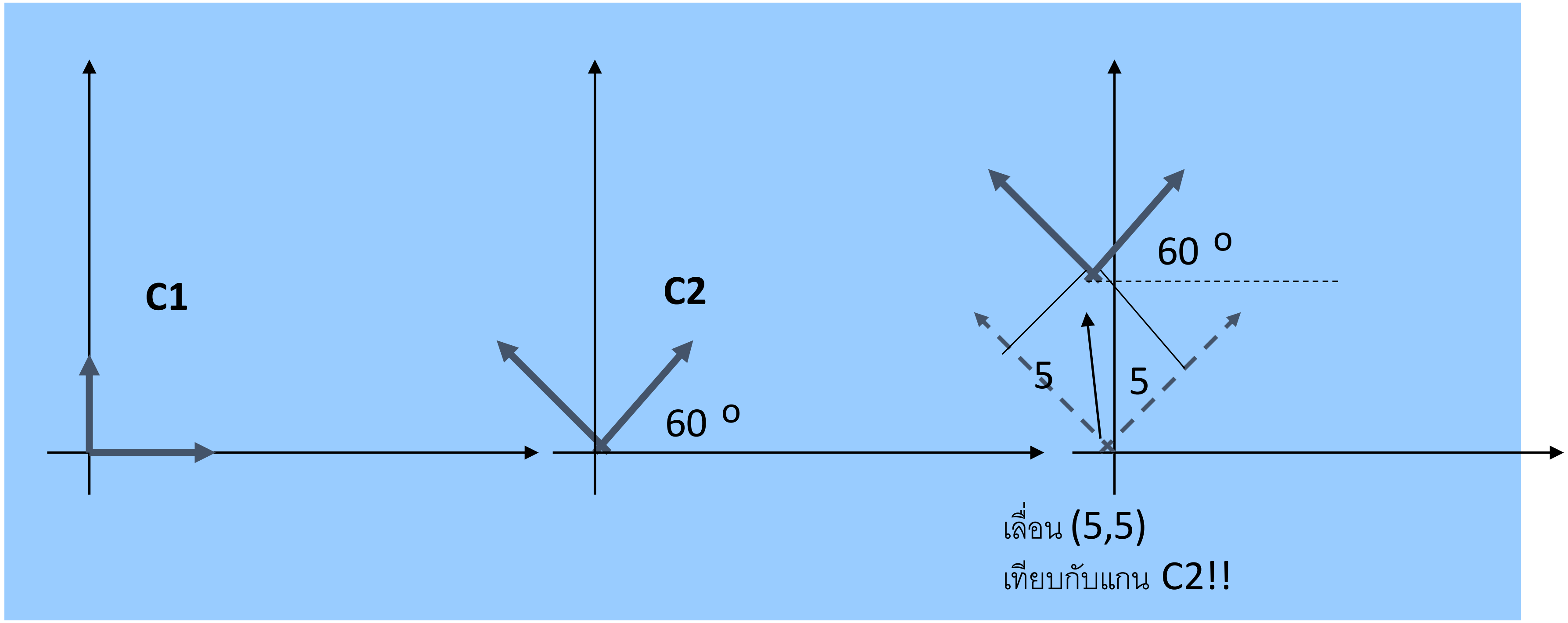
หรือ

หมุน (60) แล้วเลื่อน (5,5) ???

เฉลย: เลื่อน (5,5) แล้วหมุน (60)

การรวมการแปลง

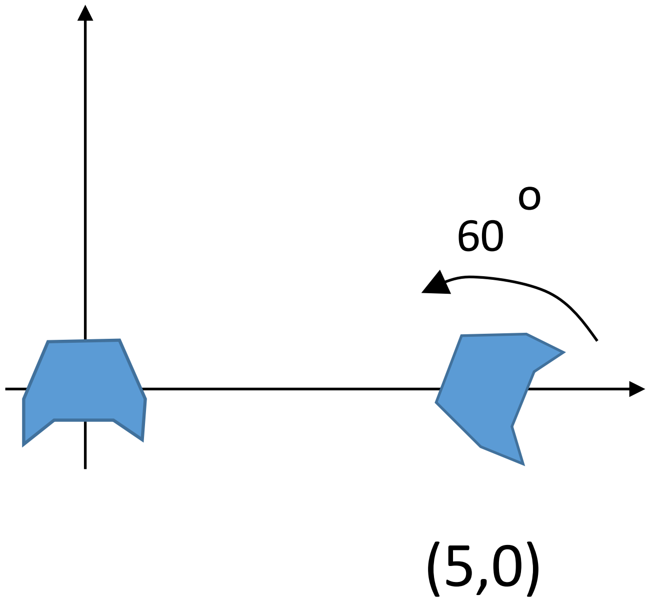
ถ้าหมุน (60) แล้วเลื่อน (5,5) ...



การแปลงวัตถุ

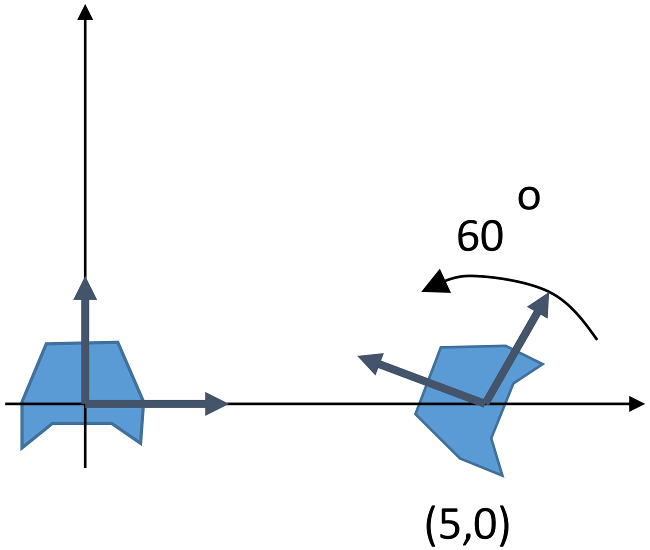
- การแปลงแกนเกี่ยวข้องอะไรกับการแปลงวัตถุ?
  - เราสามารถมองว่าวัตถุยึดติดอยู่กับแกนตัวเอง
  - การแปลงวัตถุก็คือการแปลงแกนของวัตถุแล้วนำการแปลงทั้งหมดนั้นมาทำกับจุดยอดของวัตถุ

การแปลงวัตถุแบบเทียบจุดกำเนิด



1) หมุน (60)
2) เลื่อน (5,0)

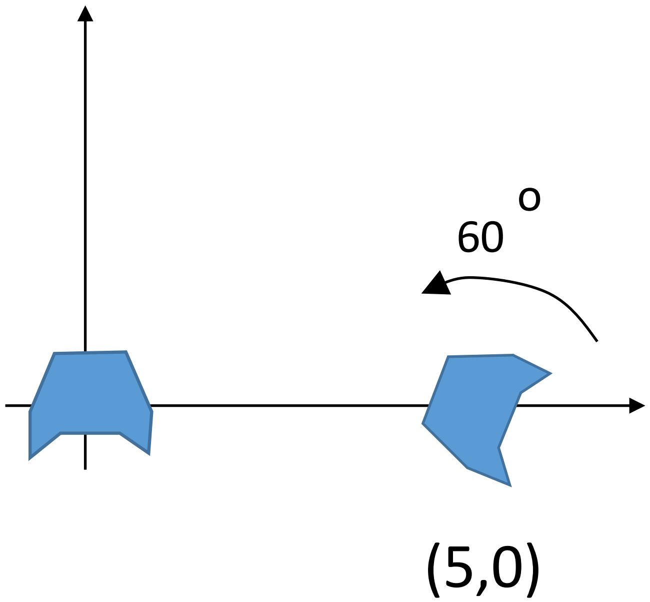
การแปลงวัตถุแบบเทียบแกนตัวเอง



1) เลื่อน (5,0)
2) หมุน (60)

ลำดับจะย้อนกลับจากแบบที่เทียบจุดกำเนิด

มองทั้งสองแบบ



ถ้ามองเป็นการแปลงเทียบจุดกำเนิด

1) หมุน (60) - M_R

2) เลื่อน (5,0) - M_T

$P' = M_R \times P$

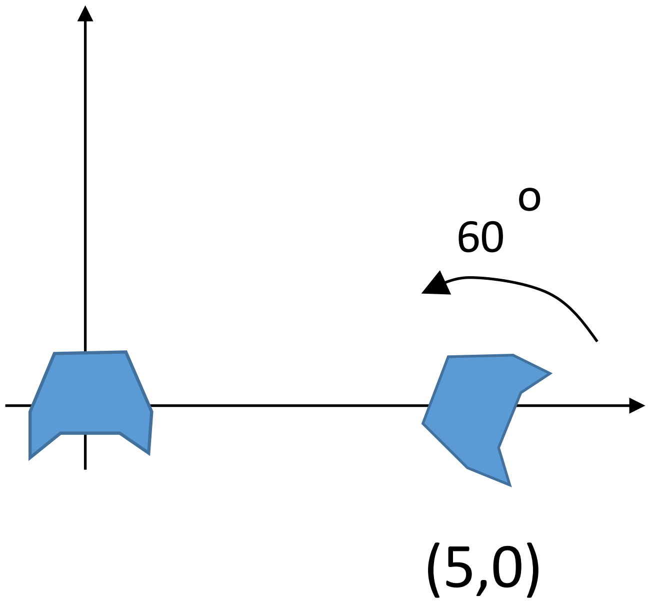
ถ้ามองเป็นการแปลงเทียบแกนตัวเอง

1) เลื่อน (5,0) - M_T

2) หมุน (60) - M_R

$P' = M_T \times P$

มองทั้งสองแบบ



ถ้ามองเป็นการแปลงเทียบจุดกำเนิด

1) หมุน (60) - M_R
2) เลื่อน (5,0) - M_T

$P' = M_T \times M_R \times P$

ถ้ามองเป็นการแปลงเทียบแกนตัวเอง

1) เลื่อน (5,0) - M_T
2) หมุน (60) - M_R

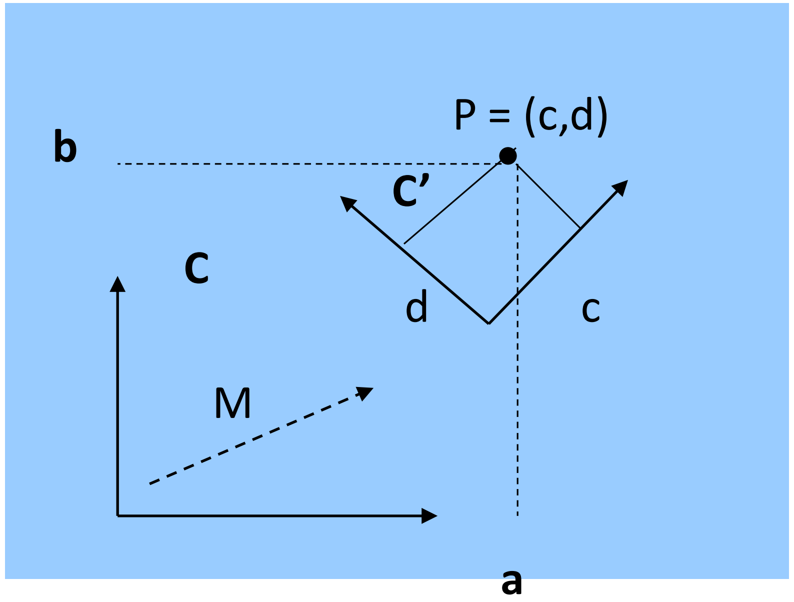
$P' = M_T \times M_R \times P$

มองแบบ OpenGL เรียก glTranslate()
ตามด้วย glRotate()

การแปลงแกนและการเทียบจุดต่างแกน

- กำหนดให้ P (c, d) ในแกน C’ และ C’ เกิดจากการแปลง C ด้วย M

$$C' = M \times C$$

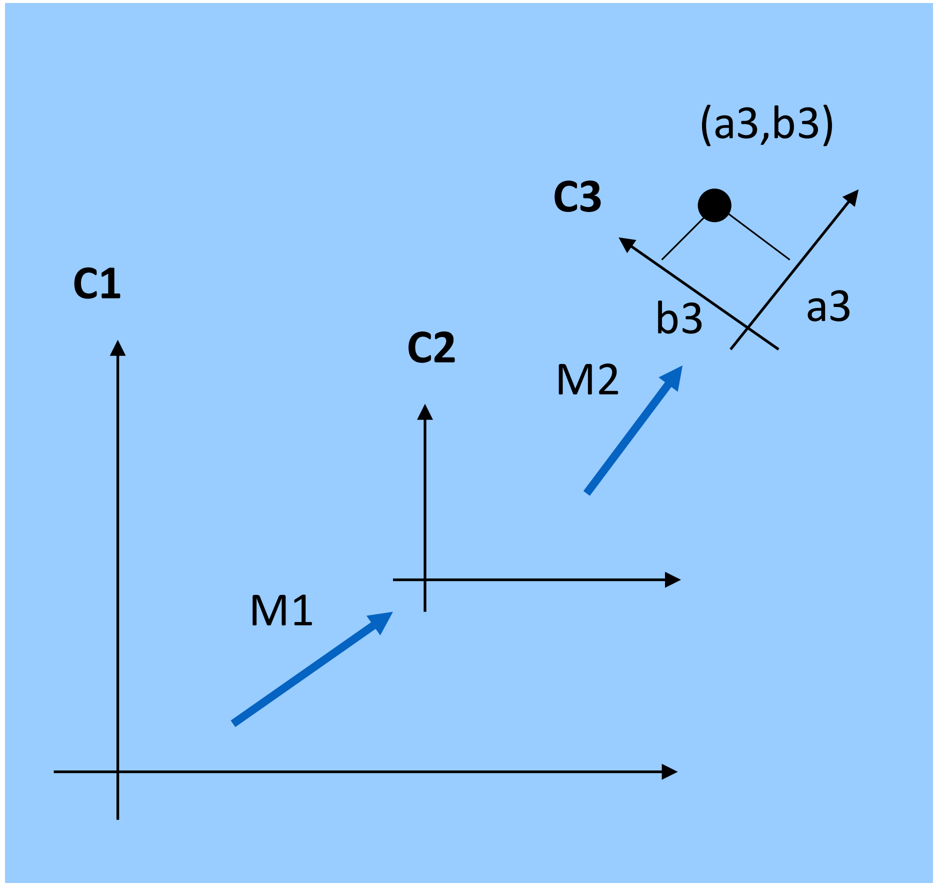


ตำแหน่งของ P ใน C คือ

$$P' = M \times P$$

$$(a,b,1) = M \times (c,d,1)$$

การเทียบจุดต่อๆ กัน



$$C1 \xrightarrow{M1} C2 \xrightarrow{M2} C3$$

กำหนดจุด P (a3,b3) ในแกน C3
ตำแหน่งของ P ในแกน C1 คือเท่าไหร่?

1) ตำแหน่งของ P ในแกน C2

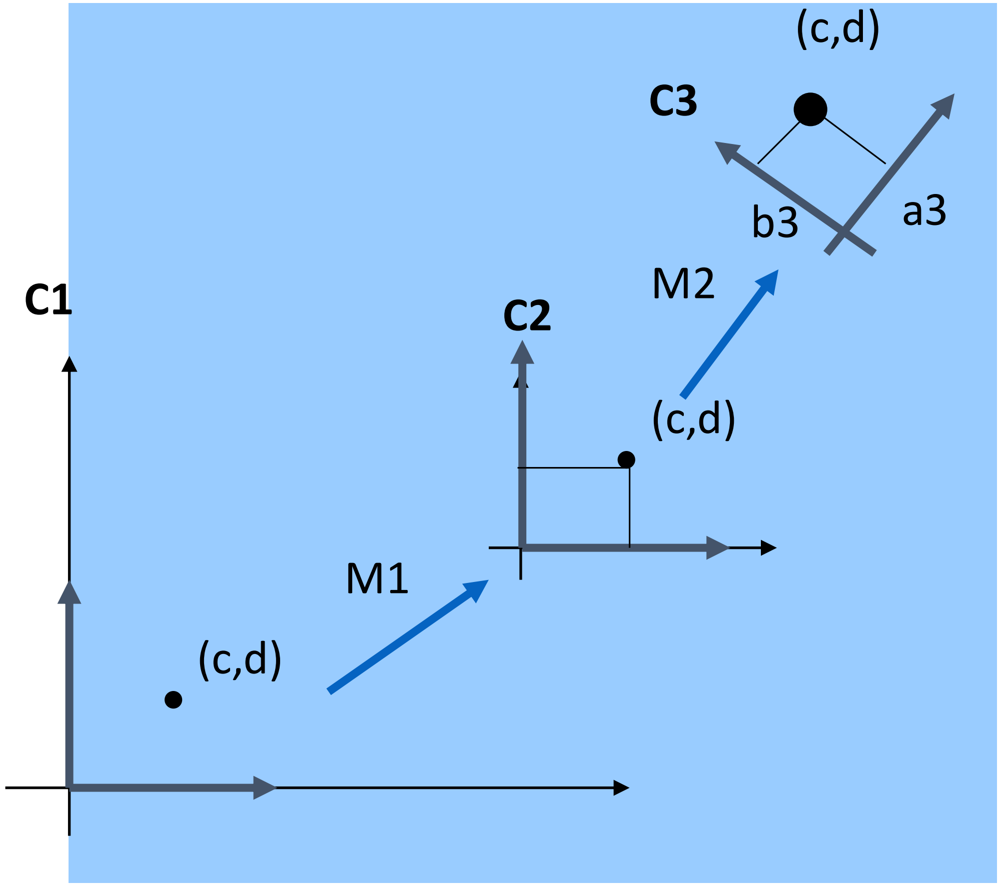
P_c2 = M2 x P

2) ตำแหน่งของ P_c2 ในแกน C1

P_c1 = M1 x P_c2

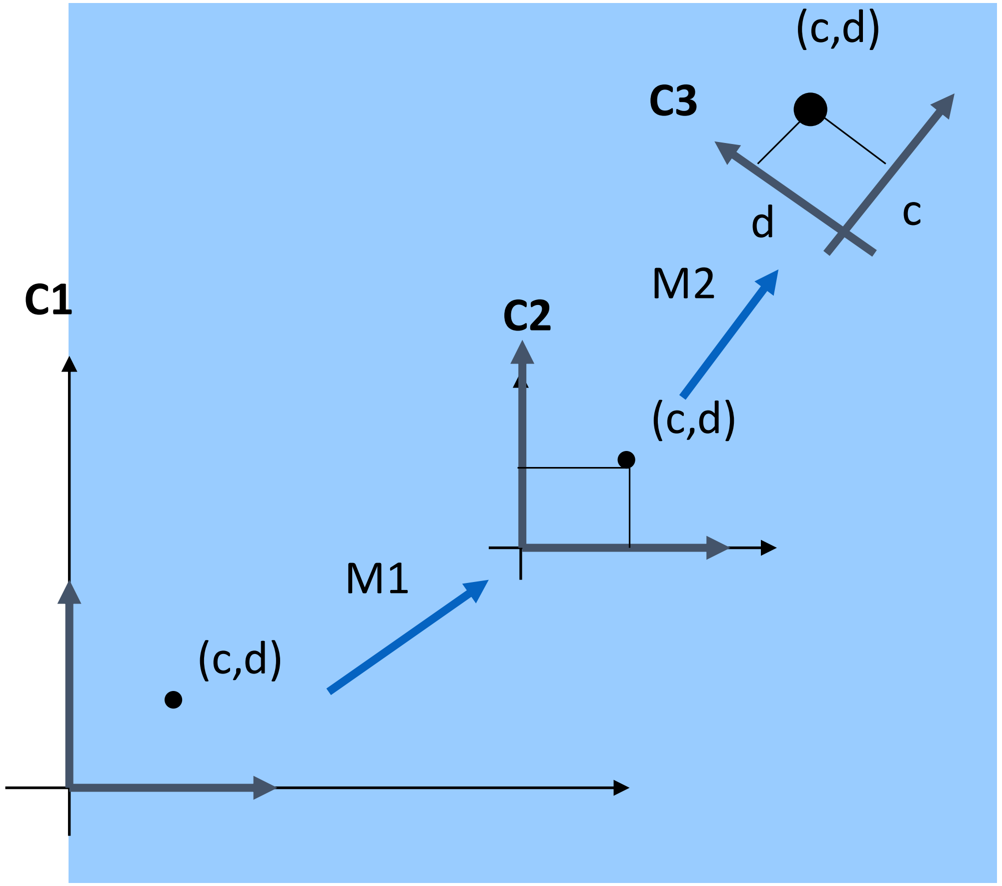
ดังนั้น P_c1 = M1 x M2 x P

มองเป็นการแปลงแกน



- มองการแปลง P (c,d) เป็นลำดับของการแปลงแกนที่มันอาศัยอยู่
- P (c,d) ติดอยู่กับแกนตัวเองตลอดเวลา
- ตำแหน่งสุดท้ายของ P หลังผ่านการแปลงแกนทั้งหมดคือตำแหน่งใน C1

มองแบบ OpenGL



สั่งการแปลงใน OpenGL ด้วย:

M1 (แปลง C1 ไปยัง C2)
M2 (แปลง C2 ไปยัง C3)

ตำแหน่งสุดท้ายของ P =
ตำแหน่งของ P ในแกน C1 =

M1 x M2 x P