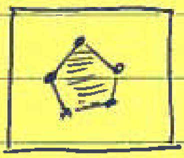
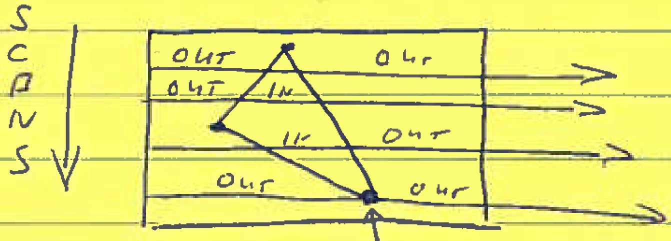
LEC 3

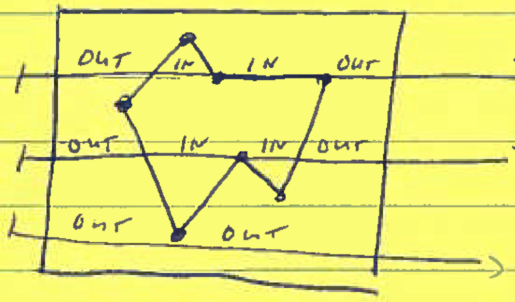
- **RASTERIZING POLYGONS** – Fill in a polygon's interior



- INPUT:
  vertices / edges

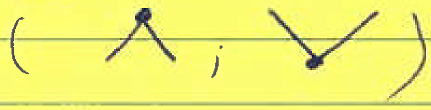
IDEA: "INSIDE-OUTSIDE TEST"
→ using semi-infinite rays





– ray on edge ⇒ "IN"

ray passing through local extremum ⇒ "VOID"



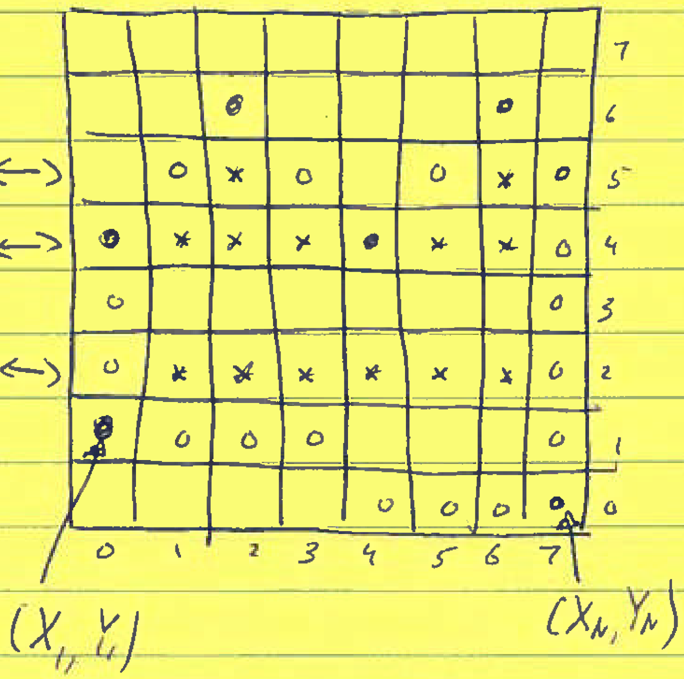
- **PIX Example**

edge lists

(1,3) → (5,7)

(0,4) → (4,7)

(0,7)



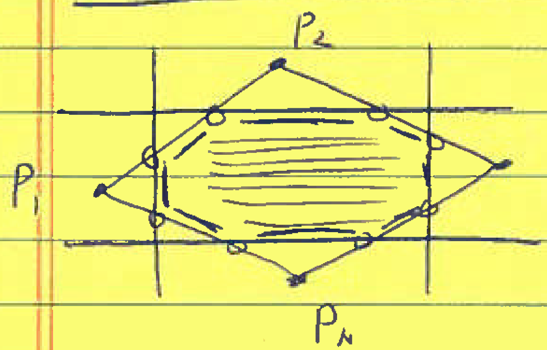
- IN: $\{(X_i, Y_i)\}$, poly vertices
- OUT: all pixels on edges and interior

i) edge pixels "o"

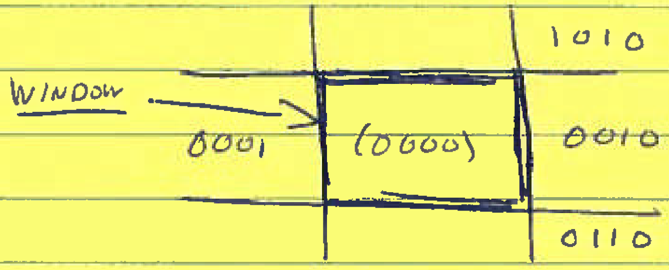
ii) edge lists for all scan lines "*"

CLIPPING

- Cohen-Sutherland Line Clipping



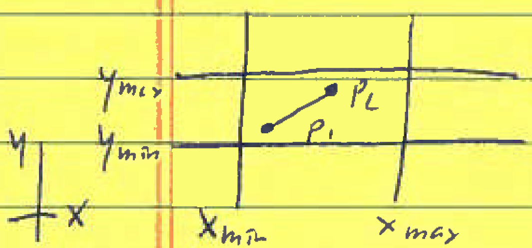
9 "regions" – use 4-bit code:



"Clip all edges $\overline{P_i P_{i+1}}$ against boundary edges of window; resulting new poly entirely INSIDE window"

(A, B, R, L) = (above, below, right, left)

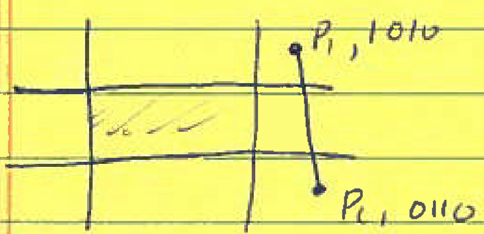
- 4-bit codes associated with all vertices; CASES:



$\mathbb{C}_1 = 0000 \wedge \mathbb{C}_2 = 0000$

$\Rightarrow$ $\overline{P_1 P_2}$ INSIDE (entirely)

$\Rightarrow$ ACCEPT



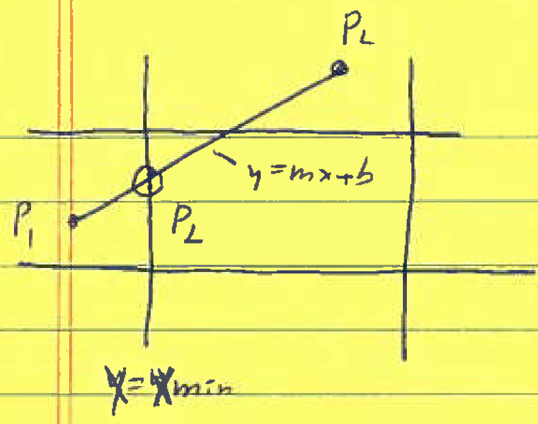
here: $1010 \wedge 0110 = \underline{0010}$

"bitwise AND"

$\mathbb{C}_1 \wedge \mathbb{C}_2 \neq 0000$

$\Rightarrow$ $\overline{P_1 P_2}$ OUTSIDE (entirely)

$\Rightarrow$ REJECT



- MUST SPLIT $\overline{P_1 P_2}$:

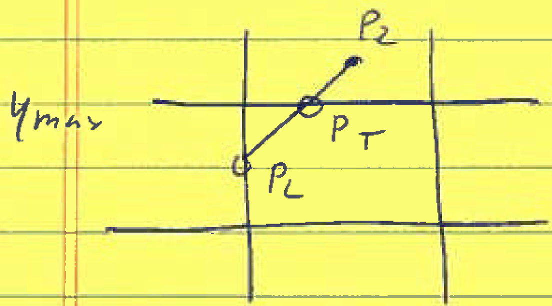
i) Clip against $x = x_{min}$

$$\boxed{P_L = (x_{min},\ m x_{min} + b)}$$

$\Rightarrow$ Segs. $\overline{P_1 P_L}$, $\overline{P_L P_2}$

ii) Check $\overline{P_1 P_L}$, $\overline{P_L P_2}$

a) $\overline{P_1 P_L}$: OUTSIDE ✓
b) $\overline{P_L P_2}$: MUST SPLIT



Clip against $y = y_{max}$

$$\boxed{P_T = \left(\frac{y_{max} - b}{m},\ y_{max}\right)}$$

$\Rightarrow$ Segs $\overline{P_L P_T}$, $\overline{P_T P_2}$

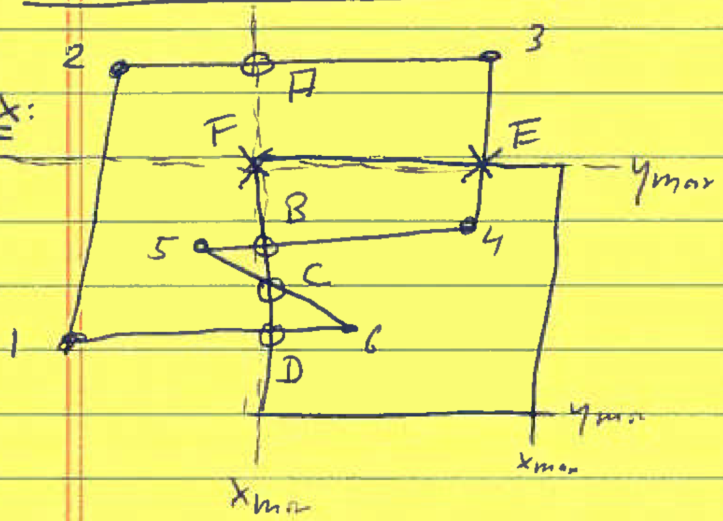
iii) Check $\overline{P_L P_T}$, $\overline{P_T P_2}$

a) $\overline{P_L P_T}$: INSIDE ✓ ACCEPT
b) $\overline{P_T P_2}$: OUTSIDE ✓

DONE

• SUTHERLAND – Hodgeman Polygon Clipping

Ex:



IN : Sequence of Poly vertices $P_1, \dots, P_N$

OUT : Sequence of Poly vertices defining "sub-poly" entirely inside window

ALG • 4-times line clipping:
- against $x = x_{min}$, ...
- " $y = y_{max}$

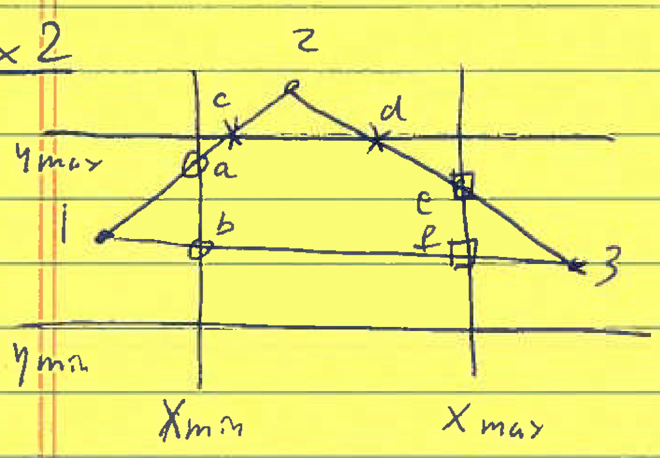
Here:

Orig : (1, 2, 3, 4, 5, 6)
x_{min}-clip : (A, 3, 4, B, C, 6, D)
y_{max}-clip : (E, ~~4~~, B, C, 6, D, F)
x_{max}-clip : "
y_{min}-clip : "

DONE

⇒ How about edge $\overline{BC}$?

Ex 2



orig : (1, 2, 3)
x_{min} : (a, 2, 3, b)
y_{max} : (a, c, d, 3, b)
x_{max} : (a, c, d, e, f, b)
y_{min} : "

Done

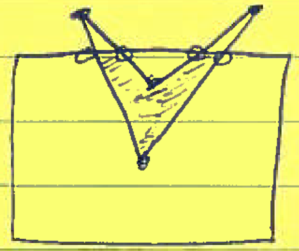
LEC 3

- When does this ALGO work?

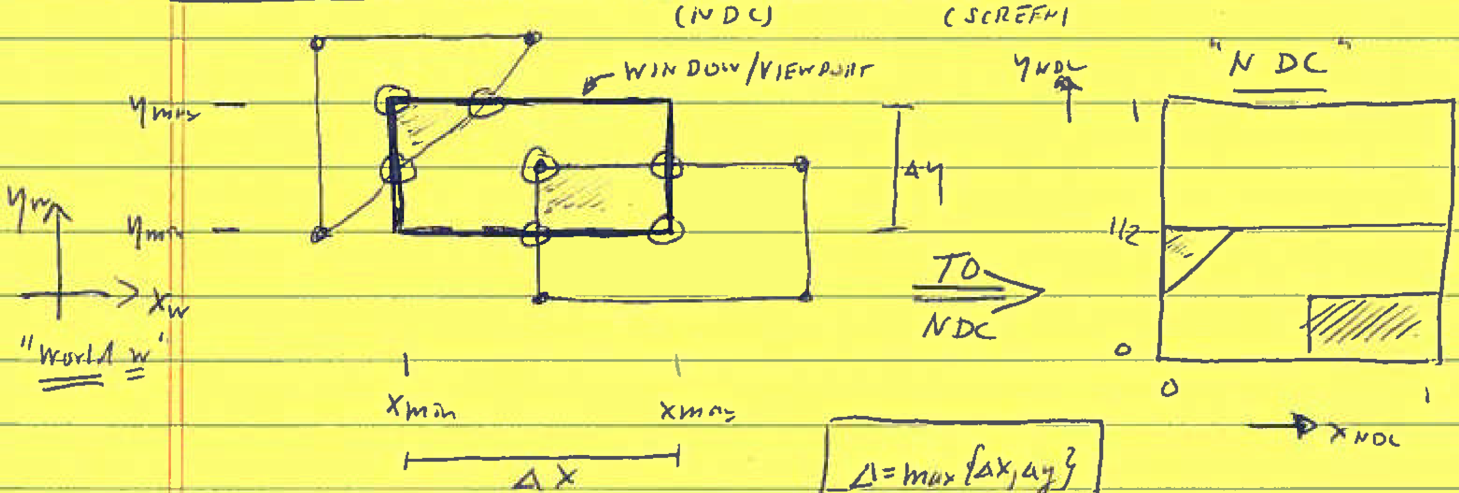
(i) WINDOW = ANY CONVEX POLY REGION:

"Apply Scissors to all edges/lines defining the WINDOW."

(ii) POLY TO BE CLIPPED: NO NEED TO BE CONVEX:



- WINDOWS — NORMALIZATION — PIXELS
(NDC) (SCREEN)



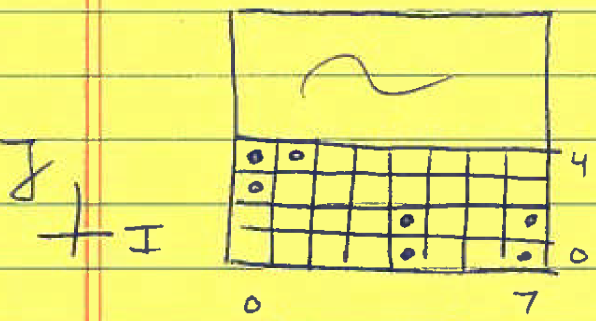
$\boxed{\Delta = \max\{\Delta x, \Delta y\}}$

→ CLIP IN WORLD THAN MAP TO NDC (DOES IT MATTER?)

— World-To-NDC:

$$\boxed{\begin{aligned} x_{NDC} &= \frac{x_w - x_{min}}{\Delta} \\ y_{NDC} &= \frac{y_w - y_{min}}{\Delta} \end{aligned}}$$

- NDC_TO_PIX

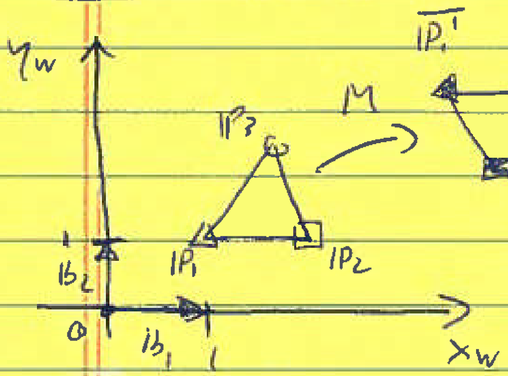


Res: $N=8$

$$\boxed{\begin{aligned} I &= (\text{int})\,(x_{NDC}\cdot(N-1)) \\ J &= (\text{int})\,(y_{NDC}\cdot(N-1)) \end{aligned}}$$

($\rightarrow$ Use int or round?)

▣ 2D Transforms

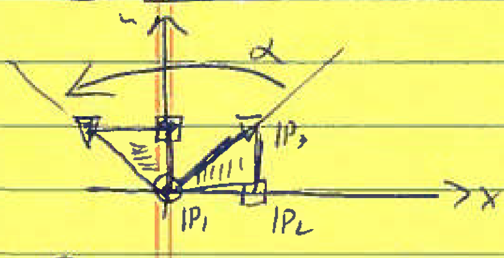


$$\boxed{\begin{aligned} \bar{\mathbb{P}} = \begin{pmatrix}\bar{x}\\ \bar{y}\end{pmatrix} &= \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22}\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}t_x\\ t_y\end{pmatrix} \\ &= M\,\mathbb{x} + \mathbb{t} \end{aligned}}$$

$\rightarrow$ 3 $\mathbb{P}$'s & 3 $\bar{\mathbb{P}}$'s $\Rightarrow$ $M, \mathbb{t}$ defined

$\rightarrow$ images of $\mathbb{0}, \mathbb{b}_1, \mathbb{b}_2 \Rightarrow M, \mathbb{t}$ "

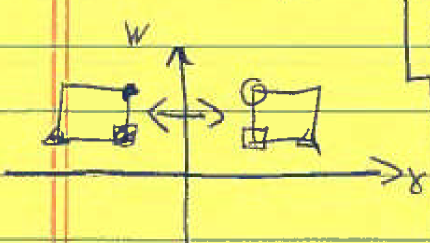
Ⓘ ROT:



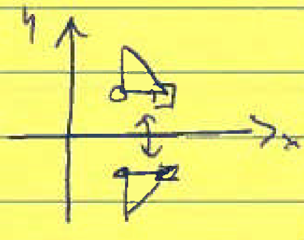
∘ ROT by α about origin:

$$\boxed{M = \begin{pmatrix}\cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha\end{pmatrix} = \underbrace{\begin{pmatrix} c\alpha & -s\alpha \\ s\alpha & c\alpha\end{pmatrix}}_{Rot(\alpha)}, \quad \mathbb{t} = \begin{pmatrix}0\\0\end{pmatrix}}$$

Ⓘⓘ REFL – x-/y-axis:

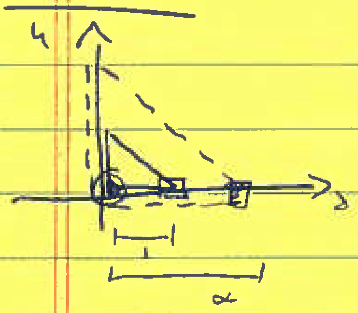


$$\boxed{\begin{aligned} &\text{Refl}_y \\ &= \begin{pmatrix}-1 & 0\\ 0 & 1\end{pmatrix} \\ &\mathbb{t} = \begin{pmatrix}0\\0\end{pmatrix} \end{aligned}}$$



$$\boxed{\begin{aligned} &\text{Refl}_x \\ &= \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}, \\ &\mathbb{t} = \begin{pmatrix}0\\0\end{pmatrix} \end{aligned}}$$

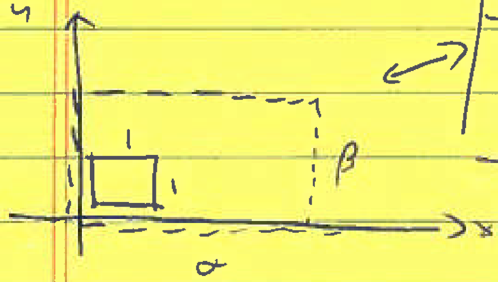
- SCALE



- Scale w.r.t. origin: (UNIFORM)

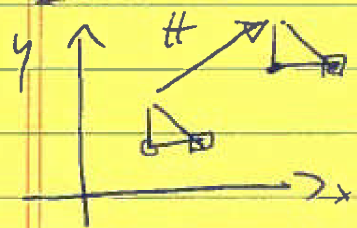
$$\boxed{Scale(\alpha) = \begin{pmatrix} \alpha & 0 \\ 0 & \alpha \end{pmatrix}, \quad t = \begin{pmatrix} 0 \\ 0 \end{pmatrix}}$$

- NON-UNIFORM scale w.r.t. origin:



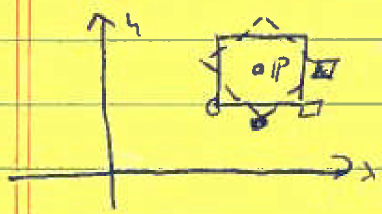
$$\boxed{Scale(\alpha, \beta) = \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}, \quad t = \begin{pmatrix} 0 \\ 0 \end{pmatrix}}$$

- TRANSLATION

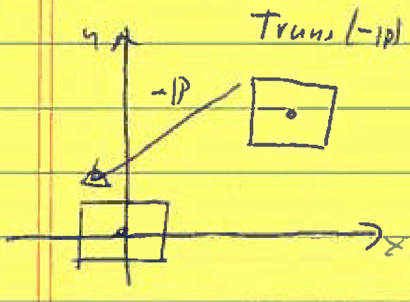


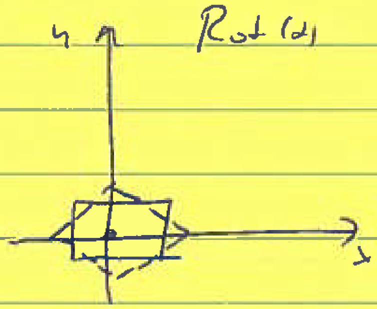
$$\boxed{Trans(t): \quad M = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad t = \begin{pmatrix} t_x \\ t_y \end{pmatrix}}$$

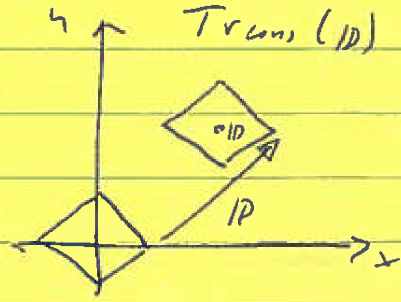
- MORE COMPLEX: $Rot(\alpha, P)$ "rot. about P by α"



1) Trans(−P)
2) Rot(α)
3) Trans(+P)







⇒ Sequence

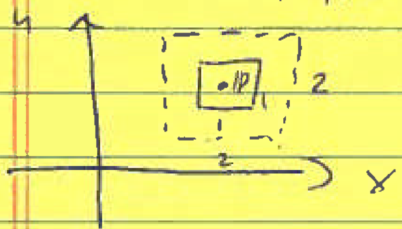
1) $\begin{pmatrix} x \\ y \end{pmatrix} \longmapsto \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} -p_x \\ -p_y \end{pmatrix} = \begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix}$

2) $\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} \longmapsto \begin{pmatrix} c\alpha & -s\alpha \\ s\alpha & c\alpha \end{pmatrix}\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix}$

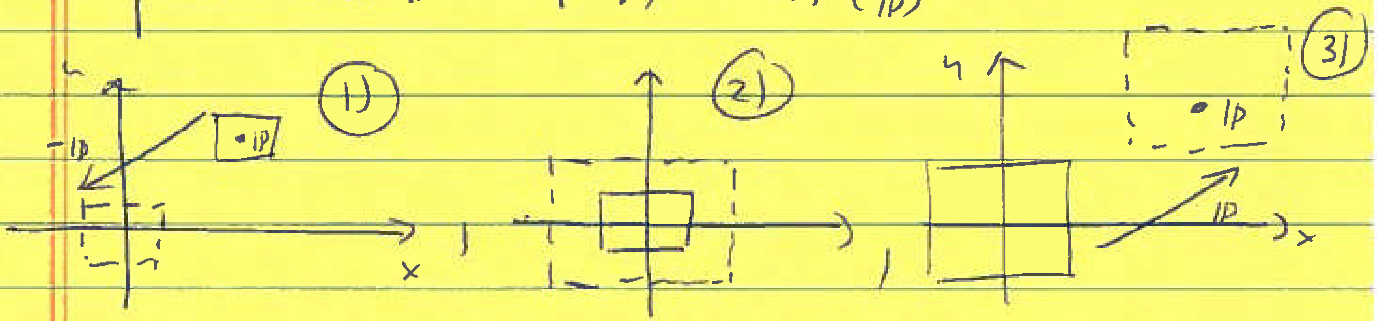
3) $\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \longmapsto \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} + \begin{pmatrix} p_x \\ p_y \end{pmatrix} = \begin{pmatrix} \bar{x} \\ \bar{y} \end{pmatrix}$

⇒ CAN WRITE THIS AS $\underline{1}$ LINEAR TRANSFORM, USING $\underline{1}$ 2-by-2 matrix and $\underline{1}$ transl. vector!

- SCALE(α, $\mathbb{P}$) "Scale uniformly w.r.t. $\mathbb{P}$"



1) Trans($-\mathbb{P}$)
2) Scale(α)
3) Trans($\mathbb{P}$)



- How to "elegantly" concatenate 'atomic' transformations to write compactly more complex ones?

⇒ transl. vector t becomes 'part of M'

⇒ use HOMOGENEOUS COORDINATES!