Kathmandu University

Department of Computer Science and Engineering Dhulikhel, Kavre



Lab report 4

[Code No: COMP 342]

Submitted by Ashutosh Singh Khadka(22)

Submitted to:

Department of Computer Science and Engineering

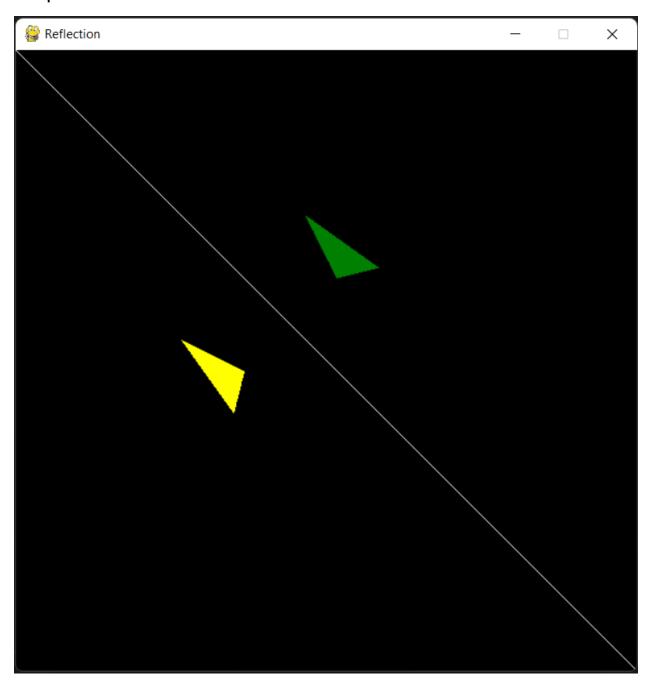
Submission Date: 9th August, 2021

Table of Contents

Reflection:	2
Implementation:	2
Output:	3
Algorithm:	4
Shearing:	5
Implementation:	5
Output:	6
Algorithm:	7
Translation:	8
Implementation:	8
Output:	9
Algorithm:	10
Rotation:	11
Implementation:	11
Output:	12
Algorithm:	13
Scaling:	14
Implementation:	14
Output:	15
Algorithm:	16

Reflection:

```
import pygame
     from pygame import gfxdraw
     import numpy as np
    WHITE = (255, 255, 255)
    YELLOW = (255, 255, 0)
    GREEN = (0,128,0)
    isp = False
    x1 = y1 = x2 = y2 = 0
    ps = (x1, y1)
    pe = (x2, y2)
    def prepare_screen():
         Create the initial screen.
        pygame.init()
        screen = pygame.display.set_mode((600, 600))
         screen.fill((0,0,0))
         pygame.display.set_caption("Reflection")
         return screen
    def reflection(x,y):
         mat = ([x],
            [y],
         transformMat = ([0,1,0],[1,0,0],[0,0,1])
         translatedPoints = np.dot(transformMat, mat)
        return translatedPoints[0],translatedPoints[1]
    screen = prepare_screen()
    gfxdraw.line(screen,0,0,1000,1000, WHITE)
38 a = (160,280)
39 b = (210,350)
40 c = (220,310)
    gfxdraw.filled_polygon(screen, [a,b,c], YELLOW)
    a = reflection(a[0],a[1])
    b = reflection(b[0],b[1])
    c = reflection(c[0],c[1])
    gfxdraw.filled_polygon(screen, [a,b,c], GREEN)
    while True:
         for event in pygame.event.get():
             if event.type == pygame.QUIT:
                pygame.quit()
                quit()
         pygame.display.update()
```

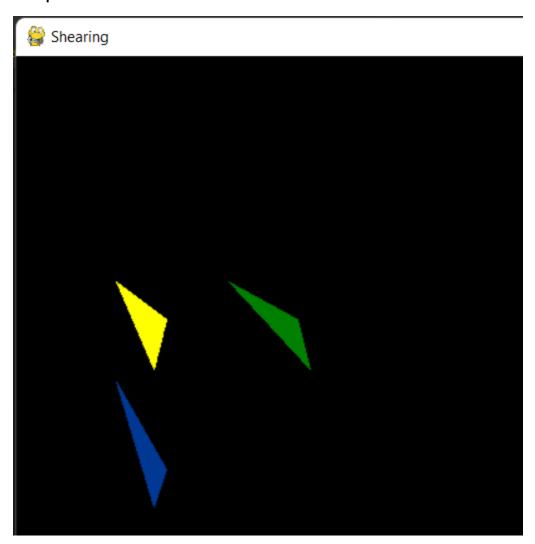


25 M. T	Saller from the Control of the Contr
	Date Page
	20 Reflection:
	ni attania
step 1:	Given points starting point (MIIYI) and end point (MIIY)
step7:	leadertion about 7-0713 (MOTHIN LOIM) 13.
	1 . 6 6
	10 -1 0 at a all all said 14.
	0 0 1
step 3:	Matin borm of starting and end points are:
	$\lceil m_1 \rceil \lceil m_2 \rceil$
	yr and yz
	1) 2) 12 12 12 12
	All states 11
16b A:	Matrin multiplication for reflection:
	T 0 0 1 1 m2
	0 -1 0 J+ y21 11 11 11 11 11 11 11
	001).[11]
	i.e
	$n_1' = n_1 & n_2' = n_2$
	y' = -y1 y2' = -y2
	1 = + 1 = +
	50,
10	o-ordinates are (11. yill) and (12, yz . 1).
	and the state of the second of the
-	

Shearing:

```
import pygame
from pygame import gfxdraw
import numpy as np
      (constant) YELLOW: tuple[Literal[255], Literal[255], Literal[0]]
     YELLOW = (255, 255, 0)
GREEN = (0,128,0)
    BLUE = (0, 56, 147)
11  x1 = y1 = x2 = y2 = 0
12  ps = (x1, y1)
      def prepare_screen():
         screen = pygame.display.set_mode((800, 800))
screen.fill((0,0,0))
          return screen
      def shearX(x,y,shx):
          mat =([x],
          transformMat = ([1,shx,0],[0,1,0],[0,0,1])
          translatedPoints = np.dot(transformMat, mat)
          return translatedPoints[0],translatedPoints[1]
      def shearY(x,y,shy):
          mat =([x],
              [y],
[1])
          transformMat = ([1,0,0],[shy,1,0],[0,0,1])
          translatedPoints = np.dot(transformMat, mat)
          return translatedPoints[0],translatedPoints[1]
```

```
screen = prepare_screen()
a = (80, 180)
b = (110,250)
c = (120,210)
gfxdraw.filled_polygon(screen, [a,b,c], YELLOW)
ax = shearX(a[0],a[1], 0.5)
bx = shearX(b[0],b[1], 0.5)
cx = shearX(c[0],c[1], 0.5)
gfxdraw.filled_polygon(screen, [ax,bx,cx], GREEN)
ay = shearY(a[0],a[1], 1)
by = shearY(b[0],b[1], 1)
cy = shearY(c[0],c[1], 1)
gfxdraw.filled_polygon(screen, [ay,by,cy], BLUE)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.quit()
    pygame.display.update()
```

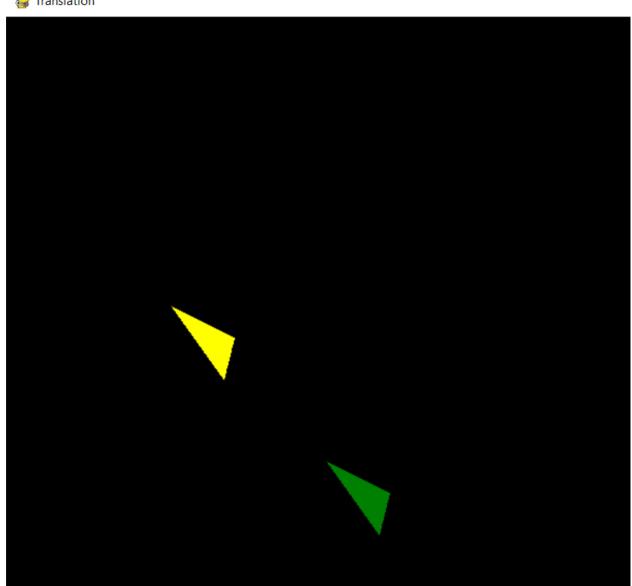


Step 1: Given points staying point (m1, y1) and end points as (m2, y2) Step 2: Given points staying point (m1, y1) and end points as (m2, y2) Step 3: Matrin pin q stearing along y-anis = shy Step 3: Matrin pin q stearing along shm and shy are = 1 L shn 0 D 1 0 D 1 0 D 1 0 D 1 0 D 1 0 Shy 1 0 D 0 1 Step 4: Matrin prom q starting point and end point. Step 5: The shearing along m-amis Lor starting 1 shm 0 m1 for end 1 0 0 m2 point 0 1 0 y1 points 0 1 0 y2 i.e. m' = n_1 + shm y1 yi = y1 yi = y2 yi = y2 new co-ordinates are (n1 y1 1) and (n2 1 y2 1) eise for starting 1 0 y1 point shy 1 0 g2 i.e. m' = m_1 yi = m_1 yi = m_1 yi = m_1 yi = m_1 so, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) i.e. m' = m_2 yi = y1 yi = y1 yi = y1 yi = y1 yi = y2 yi = y2 1 = 1 So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) yi = y1 yi = y1 yi = y2 yi = y2 + shy y2 1 = 1 So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1) So, new co-ordinates are (n1 y1 1) and (m2 1 y2 1)	1	II
Step 1: Given points stayting point (n_{1}, y_{1}) and end points as (n_{1}, y_{2}) Step 2: Given sharing along y -anis = shn Shearing along y -anis = shn Shearing along y -anis = shn Step 3: Matrin form of shearing along shn and shy are $sharing$ Step 4: Matrin form of $sharing$ point and end point. Step 4: Matrin from of $sharing$ point and end point. Step 5: T_{0} shearing along m -anis bor starting 1: shn 0: m_{1} for end 1: m_{2} 0: m_{2} point 0: m_{1} 1: m_{1} 1: m_{1} 1: m_{1} 2: m_{2} 1: m_{2} 2: m_{2} 1: m_{2} 2: m_{2} 1: m_{2} 3: m_{2} 4: m_{2} 3: m_{2} 4:		2D sheaving
Step? Given shearing along n-anis = shin Shearing along y-anis = shin Shearing along y-anis = shin Step 3 Motion prim of shearing along shin and shy are y- L shin 0 D L o and Shy L o D O L o Bry L o Step 4. Mathon from of starting point and end point. Step 5. To shearing along m-anis Lor starting L shin 0 71 Yu and Yu Step 5. To shearing along m-anis Lor starting L shin 0 71 I to 0 0 1 Step 5. To shearing along m-anis Lor starting L shin 0 71 I to 0 0 1 I to 0 0 1 Shin 0 72 I to 0 0 1 I to 0 0 72 I to sharing the same (n' y' y' 1) and (n' y' y' 1) else Point Shy L o y' = y' + shy y' 2 I to n' = m' = m Y' = y' + shy y' = y' = y' + shy y' 2 I to m' = m Y' = y' + shy y' = y' + shy y' 2		Almyth
Step 2 Given shearing along n -anis = shn Shearing along y -anis = shn Shearing along y -anis = shn Step 3 Motrin form q shearing along shn and shy are shn Step 4 Motrin form q shearing along shn and shy are shn Step 4 Motrin form q shearing point and shy are shn Step 5 The shearing along shn Step 5 The shearing along shn Step 5 The shearing shn Sharing shn Step 5 The shearing shn Sharing shn Shar	Step1	
Step 3 Matrin form of shearing along y-anis = shy step 3 Matrin form of shearing along shm and shy are significantly and shy 1 and	Step 2	Point Stayting maint (on, un) and end points as (12,42)
Step 3 Mothin form of shearing along shin and shy are 2 to $0 ext{ } ext{ }$		shearing along n-axis = share in the
Step 4: Mathin grow of starting point and end point. Step 4: Mathin grow of starting point and end point. Step 5: To shearing along m-amis box starting 1 ship 0 mi for end 1 0 0 mi point 0 1 0 yi point 0 1 0 yi i.e. $n_1' = n_1 + ship y_1$ $y_1' = y_1$ $y_2' = y_2$ $y_1' = y_1$ $y_1' =$	Sten 3	snearing along y-anis = shydra the mark Allithat
Step 4: Mathing from of starting point and end point. Step 5: The shearing along m-amis bor starting 1 ship 0 mi for end 1 0 0 mi i.e. $n_1' = n_1 + ship y_1$ $y_1' = y_2$ $y_1' = y_2$ $y_1' = y_1$ $y_1' $		Mothin form of shearing along shin and shy are
Step 4: Mathin from of starting point and end point. Step 5: T_b shearing along m -amis bor starting 1 shin 0 m_1 for end 1 m_2 1:e $m_1' = m_1$ for end 1 m_2 point $m_1' = m_1$ i.e $m_2' = m_2$ 1:e $m_2' = m_2$		1 shn 0 / [1 0 0]
Step 4: Mathon from of starting point and end point. Step 5: T_b shearing along m -amis Lor starting T_b shearing T_b shapped T_b shearing T_b shapped T_b shearing T_b shapped T_b shearing T_b shapped $T_$		- 1 13 Ny 1 0
Step 5: T_b shearing along m -amis bor starting T_b sharp T_b T_b shearing T_b sharp T_b T_b shearing T_b		
Step 5: T_b shearing along m -amis bor starting T_b sharp T_b T_b shearing T_b sharp T_b T_b shearing T_b	Stenu	- Aller Calin I aller I al Aller I al All I am
Step 5 T_b shearing along m -amis bor starting T_b shearing T_b shappy $T_$	- ch 41:	Matrin from of starting point and end point.
Step 5 Tb shearing along m-amis bor starting 1 ship 0 mi for end 1 mo mi point 0 1 0 yi point 0 1 0 yz i.e. $n_1' = n_1 + ship y_1$ i.e. $n_2' = n_2 + ship y_2$ $1 = 1$		WIT WE WANTED
Step 5: T_b shearing along m -amis bor starting 1 ship 0 m_1 for end 1 0 m_2 point 0 1 0 m_1 point 0 1 0 m_2 i.e. $m_1' = m_1$ + ship m_1' i.e. $m_2' = m_2$ + ship m_2' point $m_1' = m_1$ for end $m_2' = m_2$ point $m_1' = m_1$ i.e. $m_2' = m_2$		y and yz
Step 5 The shearing along m-amis bor starting 1 Shm 0 m_1 for end 1 m_2 point 0 1 0 m_1 for end 1 m_2 point 0 1 0 m_2 i.e. $m_1' = m_1$ + Shm y_1 i.e. $m_2' = m_2$ + Shm y_2 $m_1' = m_2$ + Shm m_2' $m_2' = m_2$ + Shm m_2' $m_1' = m_2$ + Shm m_2' $m_2' = m_2' = m_2$ $m_2' = m_2' = m_2$ $m_1' = m_2' = m_2$ $m_2' = m_2' = m_2$ $m_2' = m_2' = m_2$	01. 6	TO TO THE TAX PROPERTY AND A STATE OF THE PARTY OF THE PA
point $0 ext{ } ext$	Step 5	0 0 4 5 7 4 1 3
point 0 1 1 1 1 1 1 1 1 1 1	Tiel	The state of the s
i.e $n_1' = n_1 + shn y_1$ i.e $n_2' = n_2 + shn y_2$ $y_1' = y_1$ $y_2' = y_2$ $y_2' = y_2$ $y_2' = y_2$ $y_2' = y_2$ $y_1' = y_1$ $y_2' = y_2$ $y_2' = y_2$ $y_2' = y_2$ $y_1' = y_1$ for end $y_2' = y_2$ $y_1' = y_1 + shy = y_2$ $y_1' = y_1 + shy = y_1$ $y_2' = y_2 + shy = y_2$ $y_1' = y_1 + shy = n_1$ $y_2' = y_2 + shy = y_2$ $y_2' = y_2 + shy = y_2$ $y_2' = y_2 + shy = y_2$		point 0 1 0 1 yell points
i.e $n_1 = n_1 + shn y_1$ $y_1 = y_1$ $y_2 = y_2$ for starting $y_2 = y_2$ $y_1 = y_1$ $y_2 = y_2$ $y_1 = y_1$ $y_2 = y_2$		· · · · · · · · · · · · · · · · · · ·
new co-ordinates are $(n_1', y_1', 1)$ and $(n_2', y_2', 1)$ else for starting $(n_2', y_2', 1)$ else point $(n_2', y_2', 1)$ else $(n_2$		ie ni = ni + shnyi ie ni = n + shn
new co-ordinates are $(n_1', y_1', 1)$ and $(n_2', y_2', 1)$ else for starting $(1 \ 0 \ 0)$ (n_1) for end $(1 \ 0 \ 0)$ (n_2) $($		H = H = H
point g		
point g		new co-ordinates are (n' 1, y', 1) and (n', y', y', 1) pice
$ \begin{array}{c ccccccccccccccccccccccccccccccccccc$		10. 31411 0 1 1 0 0 7 72
$ \begin{array}{ccccccccccccccccccccccccccccccccccc$		7 7 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8 7 8
$y_1' = y_1 + shy \cdot m_1$ $y_2' = f_2 + shy \cdot y_2$		
1=1		$1.e n_1 = n_1 \qquad \qquad 1.e n_2 = n_2$
So, new co-ordinates are (n1', y1', 1) and (n2, y2', 1)		J 1 2
1100 00 010 (11 11 11 11 11 11 11 11 11 11 11 11 11		So new co-ordinates are (71', v. 1) and (21')
		11 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
	-	Free

Translation:

```
import pygame
from pygame import gfxdraw
import numpy as np
WHITE = (255, 255, 255)
YELLOW = (255, 255, 0)
GREEN = (0,128,0)
isp = False
x1 = y1 = x2 = y2 = 0
ps = (x1, y1)
pe = (x2, y2)
def prepare_screen():
    pygame.init()
    screen = pygame.display.set_mode((800, 800))
    screen.fill((0,0,0))
    pygame.display.set_caption("Translation")
    return screen
def translate(x,y, tx, ty):
    mat = ([x],
        [y],
    transformMat = ([1,0,tx],[0,1,ty],[0,0,1])
    translatedPoints = np.dot(transformMat, mat)
    return translatedPoints[0],translatedPoints[1]
screen = prepare_screen()
a = (160, 280)
b = (210,350)
c = (220,310)
gfxdraw.filled_polygon(screen, [a,b,c], YELLOW)
a = translate(a[0],a[1], 150, 150)
b = translate(b[0],b[1], 150, 150)
c = translate(c[0],c[1], 150, 150)
gfxdraw.filled_polygon(screen, [a,b,c], GREEN)
while True:
    for event in pygame.event.get():
         if event.type == pygame.QUIT:
            pygame.quit()
            quit()
    pygame.display.update()
```

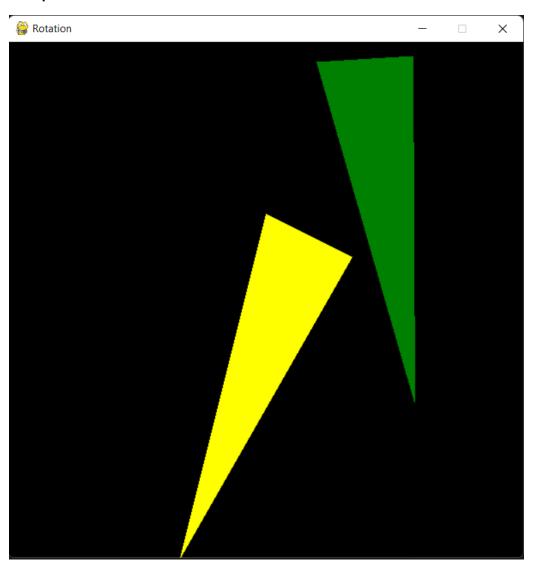
Fanslation



The second of th	
2D Translation:	
Algorithm:	
Step 1: Given points of line (71, 41) as starting point and (72 142)	03
- end point	
Given translation rolly 1 hours	
Step 3. Arranging obove translation point into 3x3 matrin.	
[1 0 ta]	
0 1 ty	
001	
Step 4. Arranging about starting & end points into 3x1 motion	
for starting MI for end [72]	
Point yL Points yz	
Steps: Carry out matrin multiplication:	
Lota M	
0 1 ty 3-	
$ i \in n_1' = n_1 + t_n $ $ y_2' = y_1 + t_y $	
$\frac{1}{1} = \frac{1}{1}$	
new co-ordinates of starting point (ni, yi, 1)	
in the state of stating point (M. gr, 1)	
Stepb: Carry out translation multiplication for end points.	
1 0 tn 7 [72]	
0 L ty Y2	
0.01	
n' = tn + n2 So, new co-adinates for end	
$y_2' = y_2 + t_y$ points $(\eta_2', y_2', \underline{1})$.	
1 = 1	

Rotation:

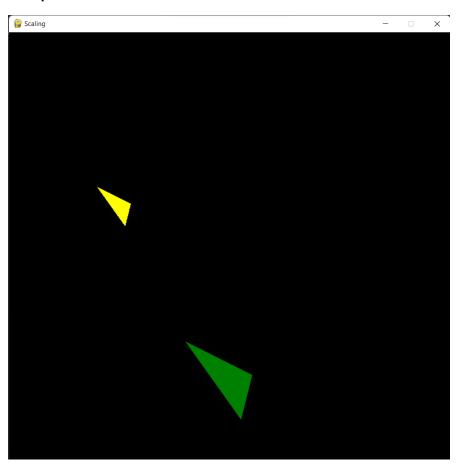
```
import pygame
     from pygame import gfxdraw
     import numpy as np
     WHITE = (255, 255, 255)
     YELLOW = (255, 255, 0)
    GREEN = (0,128,0)
     isp = False
    ps = (x1, y1)
     pe = (x2, y2)
     def prepare_screen():
       pygame.init()
        screen = pygame.display.set_mode((600, 600))
        screen.fill((0,0,0))
        pygame.display.set_caption("Rotation")
        return screen
    screen = prepare_screen()
     def rotat(x,y,a):
        a = np.math.radians(a)
        m = ([x])
            [y],
         transformM = ([np.math.cos(a),-np.math.sin(a),0],[np.math.sin(a),np.math.cos(a),0],[0,0,1])
        translatedPoints = np.dot(transformM, m)
        return translatedPoints[0],translatedPoints[1]
   m = (200,600)
    n = (300, 200)
     o = (400, 250)
     gfxdraw.filled_polygon(screen, [m,n,o], YELLOW)
    m = rotat(m[0],m[1], 330)
    n = rotat(n[0],n[1], 330)
     o = rotat(o[0],o[1], 330)
     gfxdraw.filled_polygon(screen, [m,n,o], GREEN)
     while True:
38
         for event in pygame.event.get():
             if event.type == pygame.QUIT:
                pygame.quit()
                 quit()
         pygame.display.update()
```



	Date
	20 Rotation:
Should	Alanotan
Step 2:	- Startion and 6 1 1 1 1 0 1 (a. 11)
Step 3:	Rotation by angle (0) Acronging above rotation angle in 3x3 matrix
	(630 3410 0
,	Sino COSO O
Step 4.	Starting point & end point in mathon.
	$\frac{1}{3}$ $\frac{1}$
Step 5:	Carry out matrin multiplication for rotation
•	For starting [coso -sino o] 71
	SIND COSO 0 . JI
	ie: MI = MICOSO - GISINO
	$y_1' = m_1 \sin \phi + y_1 \cos \phi$ $+ = 1$
	and 7 = 7
	For end cose - sine o nz Points sine cose o 42
	(9 0 1) (1).
	i.e $n_2' = n_2 \cos \Theta - y_2 \sin \Theta$ $y_2' = 10 n_2 \sin \Theta + y_2 \sin \Theta$
	So, co-ordinates are (n1'141'11) & (n2142', 1)

Scaling:

```
from pygame import gfxdraw
import numpy as np
WHITE = (255, 255, 255)
YELLOW = (255, 255, 0)
GREEN = (0,128,0)
ps = (x1, y1)
pe = (x2, y2)
def prepare_screen():
    Create the initial screen.
   pygame.init()
    screen = pygame.display.set_mode((800, 800))
    screen.fill((0,0,0))
    pygame.display.set_caption("Scaling")
    return screen
def scal(x,y,sx,sy):
    mat = ([x],
        [y],
    transformMat = ([sx,0,0],[0,sy,0],[0,0,1])
    translatedPoints = np.dot(transformMat, mat)
    return translatedPoints[0],translatedPoints[1]
screen = prepare_screen()
a = (160, 280)
b = (210,350)
c = (220,310)
gfxdraw.filled_polygon(screen, [a,b,c], YELLOW)
a = scal(a[0],a[1], 2, 2)
b = scal(b[0],b[1], 2, 2)
c = scal(c[0],c[1], 2, 2)
gfxdraw.filled_polygon(screen, [a,b,c], GREEN)
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
           pygame.quit()
            quit()
    pygame.display.update()
```



	Date Page
	2D scaling:
	Ole and the mi
Stepl	
	(Siven scaling ractor (Smisy)
Step3	
	Snow of the one of the last of the
	0 Sy 0
4	
stepy	Matin form of starting and end points ore:
	[J]
1	y 1 and yz
Steps	Matria multiplication for scaling:
этер.	for starting point: [sn 0 0] [on1] se mil = myroso-yisin 0
	Bor starting point: Son O O MI GO MI = myros & - yisin & yi= misin & + yios &.
	. 0 0 1 (1)
	h!
	points $\cos \theta$ - $\sin \theta$ 0 η_2 i.e $\eta_2' = \eta_2 \cos \theta - y_2 \sin \theta$ $\sin \theta$ $\cos \theta$ 0 $y_2' = \eta_2 \sin \theta + y_2 \sin \theta$
	Sino coso o y2 y2 = n2sino fyzsino
	the same of the Figure 1 and the same of t
	So, co-ordinates are (ni yi, L) f (nz, yz, 1).
	· // · · · · · · · · · · · · · · · · ·

Conclusion:

In this way, we used python programming language and pygame environment to reflect, shear, translate, rotate and scale a triangle in this lab segment.