

**LAPORAN TUGAS BESAR II**  
**IF2123-ALJABAR GEOMETRI**  
**“Simulasi Transformasi Linier pada Bidang 2D Dengan**  
**Menggunakan OpenGL API”**

disusun oleh:

Eric Jonathan	13516117
Christian Wibisono	13516147



**PROGRAM STUDI TEKNIK INFORMATIKA**  
**SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA**  
**INSTITUT TEKNOLOGI BANDUNG**  
**2017**

## BAB I

### DESKRIPSI MASALAH

#### A. Deskripsi Umum

Pada tugas ini, mahasiswa diminta untuk membuat program yang dapat mensimulasikan transformasi linier untuk melakukan operasi translasi, refleksi, dilatasi, rotasi, *shear*, *stretch*, serta transformasi menggunakan matriks pada sebuah bidang dua dimensi (2D). Bidang dibuat dengan mendefinisikan sekumpulan titik sudut lalu membuat bidang dari titik-titik tersebut.

Setelah titik-titik pada bidang didefinisikan maka program akan menampilkan hasil visualisasi bidang dua dimensi menggunakan layanan dan fitur yang disediakan OpenGL API pada layar. Pengguna kemudian dapat mensimulasikan transformasi linier yang dikehendaki melalui masukan perintah pada *command prompt*.

#### B. Spesifikasi Program

Program akan memiliki dua buah window, window pertama (*command prompt*) berfungsi untuk menerima masukan dari pengguna, sedangkan window kedua (*GUI*) berfungsi untuk menampilkan output berdasarkan input dari user. Kedua window ini muncul ketika user membuka file *executable*. Saat program baru mulai dijalankan, program akan menerima input **N**, yaitu jumlah titik yang akan diterima. Berikutnya, program akan menerima input **N** buah **titik** tersebut (pasangan nilai **x dan y**). Setelah itu program akan menampilkan output sebuah bidang yang dibangkitkan dari titik-titik tersebut. Selain itu juga ditampilkan dua buah garis, yaitu **sumbu x** dan **sumbu y**. Nilai **x** dan **y** memiliki rentang minimal **-500 pixel** dan maksimum **500 pixel**. Pastikan window *GUI* yang Anda buat memiliki ukuran yang cukup untuk menampilkan kedua sumbu dari ujung ke ujung.

Berikutnya, program dapat menerima input yang didefinisikan pada tabel dibawah.

Input	Keterangan
<code>translate &lt;dx&gt; &lt;dy&gt;</code>	Melakukan translasi objek dengan menggeser nilai $x$ sebesar $dx$ dan menggeser nilai $y$ sebesar $dy$ .
<code>dilate &lt;k&gt;</code>	Melakukan dilatasi objek dengan faktor scaling $k$ .
<code>rotate &lt;deg&gt; &lt;a&gt; &lt;b&gt;</code>	Melakukan rotasi objek secara berlawanan arah jarum jam sebesar $deg$ derajat terhadap titik $a,b$
<code>reflect &lt;param&gt;</code>	Melakukan pencerminan objek. Nilai $param$ adalah salah satu dari nilai-nilai berikut: $\mathbf{x}$ , $\mathbf{y}$ , $\mathbf{y=x}$ , $\mathbf{y=-x}$ , atau $(\mathbf{a}, \mathbf{b})$ . Nilai $(a,b)$ adalah titik untuk melakukan pencerminan terhadap.
<code>shear &lt;param&gt; &lt;k&gt;</code>	Melakukan operasi <i>shear</i> pada objek. Nilai $param$ dapat berupa $x$ (terhadap sumbu $x$ ) atau $y$ (terhadap sumbu $y$ ). Nilai $k$ adalah faktor <i>shear</i> .
<code>stretch &lt;param&gt; &lt;k&gt;</code>	Melakukan operasi <i>stretch</i> pada objek. Nilai $param$ dapat berupa $x$ (terhadap sumbu $x$ ) atau $y$ (terhadap sumbu $y$ ). Nilai $k$ adalah faktor <i>stretch</i> .
<code>custom &lt;a&gt; &lt;b&gt; &lt;c&gt; &lt;d&gt;</code>	Melakukan transformasi linier pada objek dengan matriks transformasi sebagai berikut: $\begin{bmatrix} a & b \\ c & d \end{bmatrix}$

<pre>multiple &lt;n&gt; ... // input 1 ... // input 2 ... ... // input n</pre>	<p>Melakukan transformasi linier pada objek sebanyak <math>n</math> kali berurutan. Setiap baris input 1..<math>n</math> dapat berupa <i>translate</i>, <i>rotate</i>, <i>shear</i>, <i>dll</i> tetapi bukan <i>multiple</i>, <i>reset</i>, <i>exit</i>.</p>
<pre>reset</pre>	<p>Mengembalikan objek pada kondisi awal objek didefinisikan.</p>
<pre>exit</pre>	<p>Keluar dari program.</p>

## BAB II

### LANDASAN TEORI

#### A. Transformasi Linier

Secara umum transformasi (pemetaan) didefinisikan dari suatu himpunan ke himpunan lain Transformasi geometri merupakan salah satu bahasan dalam geometri mengenai perubahan bentuk, letak, dan penyajian berdasarkan pada suatu gambar dan matriks.. Transformasi ini dapat dipandang sebagai fungsi bernilai vektor yang berasal dari peubah yang vektor juga. Jadi, domain dan kodomain fungsi ini adalah berupa vektor. Jika  $V$  dan  $W$  adalah ruang vektor dan  $F$  adalah suatu fungsi yang mengasosiasikan vektor unik di  $W$  dengan setiap vektor yang terletak di  $V$ , maka dikatakan  $F$  memetakan  $V$  di dalam  $W$ .

$$F: V \rightarrow W$$

Jika  $F$  mengasosiasikan vektor  $w$  dengan vektor  $v$ , maka  $w = F(v)$ .  $w$  adalah bayangan dari  $v$  dibawah  $F$ . Ruang vektor  $V$  dikatakan domain  $F$

Misalkan  $V$  dan  $W$  adalah ruang vektor atas bilangan real  $R$ , dan misalkan  $t: V \rightarrow W$  adalah suatu fungsi.

Fungsi  $t$  disebut transformasi linier jika

$$1) \quad T(\mathbf{u} + \mathbf{v}) = T(\mathbf{u}) + T(\mathbf{v})$$

$$2) \quad T(k \cdot \mathbf{u}) = k T(\mathbf{u})$$

Untuk setiap  $\mathbf{u}, \mathbf{v} \in V$  dan  $k \in R$ .

#### B. Matriks Transformasi

Untuk transformasi linear, secara umum  $T: R^n \rightarrow R^m$  dapat didefinisikan sebagai berikut

$$w_1 = a_{11}x_1 + a_{12}x_2 + \dots + a_{1n}x_n$$

$$w_2 = a_{21}x_1 + a_{22}x_2 + \dots + a_{2n}x_n$$

...

$$w_m = a_{m1}x_1 + a_{m2}x_2 + \dots + a_{mn}x_n$$

dan dapat diringkas menjadi

$$W = A.x$$

Dimana A adalah matriks standar untuk transformasi linear T

### C. Operasi Pada Transformasi Linier

#### Translasi

Translasi adalah transformasi yang memindahkan titik-titik dengan jarak dan arah tertentu.

$$T(x, y) + (\Delta x, \Delta y) = (x + \Delta x, y + \Delta y)$$

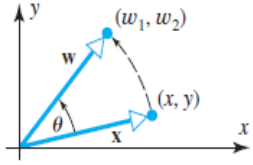
#### Refleksi

Suatu transformasi yang memindahkan tiap titik pada bidang dengan menggunakan sifat bayangan cermin dari titik-titik yang dipindahkan.

Operator	Illustration	Images of $e_1$ and $e_2$	Standard Matrix
Reflection about the y-axis $T(x, y) = (-x, y)$		$T(e_1) = T(1, 0) = (-1, 0)$ $T(e_2) = T(0, 1) = (0, 1)$	$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$
Reflection about the x-axis $T(x, y) = (x, -y)$		$T(e_1) = T(1, 0) = (1, 0)$ $T(e_2) = T(0, 1) = (0, -1)$	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Reflection about the line $y = x$ $T(x, y) = (y, x)$		$T(e_1) = T(1, 0) = (0, 1)$ $T(e_2) = T(0, 1) = (1, 0)$	$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

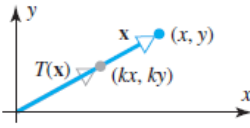
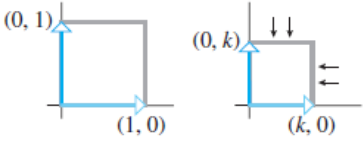
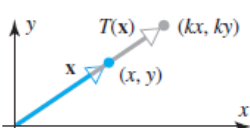
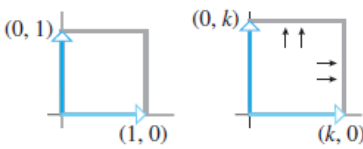
## Rotasi

Rotasi adalah perputaran yang di tentukan oleh pusat sumbu putar dan besar sudut putar.

Operator	Illustration	Rotation Equations	Standard Matrix
Rotation through an angle $\theta$		$w_1 = x \cos \theta - y \sin \theta$ $w_2 = x \sin \theta + y \cos \theta$	$\begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$

## Dilatasi

Dilatasi (pembesaran atau perkalian) ialah suatu transformasi yang mengubah ukuran (memperkecil atau memperbesar) suatu bangun tetapi tidak mengubah bentuk bangun yang bersangkutan. Dilatasi ditentukan oleh titik pusat dan faktor (faktor skala)

Operator	Illustration $T(x, y) = (kx, ky)$	Effect on the Standard Basis	Standard Matrix
Contraction with factor $k$ on $R^2$ $(0 \leq k < 1)$			$\begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix}$
Dilation with factor $k$ on $R^2$ $(k > 1)$			

## Shear

Operator	Effect on the Standard Basis	Standard Matrix
Shear of $R^2$ in the $x$ -direction with factor $k$ $T(x, y) = (x + ky, y)$		$\begin{bmatrix} 1 & k \\ 0 & 1 \end{bmatrix}$
Shear of $R^2$ in the $y$ -direction with factor $k$ $T(x, y) = (x, y + kx)$		$\begin{bmatrix} 1 & 0 \\ k & 1 \end{bmatrix}$

## Custom

Operasi transformasi matriks menggunakan matriks transformasi 2x2

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$x' = ax + by$$

$$y' = cx + dy$$

## Stretch

Operator	Illustration $T(x, y) = (kx, y)$	Effect on the Standard Basis	Standard Matrix
Compression of $R^2$ in the $x$ -direction with factor $k$ $(0 \leq k < 1)$			$\begin{bmatrix} k & 0 \\ 0 & 1 \end{bmatrix}$
Expansion of $R^2$ in the $x$ -direction with factor $k$ $(k > 1)$			
Operator	Illustration $T(x, y) = (x, ky)$	Effect on the Standard Basis	Standard Matrix
Compression of $R^2$ in the $y$ -direction with factor $k$ $(0 \leq k < 1)$			$\begin{bmatrix} 1 & 0 \\ 0 & k \end{bmatrix}$
Expansion of $R^2$ in the $y$ -direction with factor $k$ $(k > 1)$			



#### D. OpenGL API

Open Graphics Library (OpenGL) adalah API yang berfungsi untuk melakukan *rendering* grafik 2D dan 3D. OpenGL bersifat *cross-language*, *cross-platform*, dan *open source*. OpenGL umumnya digunakan untuk melakukan interaksi dengan GPU (*graphics processing unit*) untuk mencapai hasil *render* yang diakselerasi dengan hardware. OpenGL mendefinisikan berbagai instruksi untuk menggambar objek, image (umumnya 3D) dan melakukan berbagai operasi terhadap objek-objek tersebut.

Fungsi dasar dari OpenGL adalah untuk mengeluarkan koleksi perintah khusus atau executable ke sistem operasi. Dengan demikian, program ini bekerja dengan perangkat keras grafis yang ada yang berada pada hard drive atau sumber tertentu lainnya. Setiap perintah dalam dirancang untuk melakukan tindakan tertentu, atau memulai efek khusus tertentu yang terkait dengan grafis.

Seiring dengan kemampuan interface dari sistem operasi, OpenGL juga menyediakan beberapa built-in protokol yang mungkin berguna bagi pengguna akhir. Di antaranya fitur alat seperti alpha blending, pemetaan tekstur, dan efek atmosfer. Alat ini dapat berinteraksi dengan sistem operasi yang sedang digunakan.

Awalnya dikembangkan oleh Silicon Graphics, OpenGL kini dianggap standar industri. Interface program aplikasi yang aktif didukung oleh Microsoft ini, menawarkan download gratis daftar OpenGL untuk digunakan pada sistem Windows. OpenGL juga bekerja sangat baik dengan Inventor Open, sebuah pemrograman berorientasi obyek alat juga diciptakan oleh Silicon Graphics.

## BAB III

### IMPLEMENTASI PROGRAM

#### A. Bahasa Pemrograman dan Pustaka

Pada pengerjaan tugas ini kami mengimplementasikan program menggunakan bahasa pemrograman Python 2.7. Adapun kami membuat 2 file yaitu file program utama (main.py) sebagai *controller window* OpenGL dan file modul transformation.py untuk pengaplikasian fungsi-fungsi pada transformasi linier seperti *translate*, *rotate*, *dilate*, dan sebagainya.

Untuk membantu penggunaan OpenGL API, kami menggunakan beberapa pustaka yaitu GL, GLU, dan GLUT. Dalam implementasi fungsi-fungsi untuk melakukan transformasi serta animasi kami menggunakan pustaka math, sleep, dan thread untuk penggunaan fitur *multi-threading*.

```
from __future__ import print_function
from OpenGL.GL import *
from OpenGL.GLU import *
from OpenGL.GLUT import *
from math import pi, sin, cos
import thread
from time import sleep
import copy
```

#### B. Modul dan Fungsi

Program utama simulasi transformasi linier pada bidang 2D ini kami implementasikan dengan mengimpor modul transformasi.py yang merupakan kumpulan fungsi-fungsi transformasi beserta animasi yang telah kami buat. Adapun dalam modul ini terdapat prosedur-prosedur

- def translate(vert, NbVert, dx, dy): merupakan prosedur untuk melakukan translasi bangun vert sejauh dx dan dy

- `def dilate(vert,NbVert,k)`: merupakan prosedur untuk melakukan dilatasi bangun vert sebesar  $k$  sebagai faktor skalar
- `def rotate(vert,NbVert,deg,a,b)`: merupakan prosedur untuk melakukan rotasi bangun vert dengan derajat  $deg$  dan titik pusat  $(a,b)$  .
- `def shear(vert,NbVert,axis,k)`: merupakan prosedur untuk melakukan operasi *shear* pada bangun vert dengan parameter sumbu *axis* dan faktor skalar  $k$ .
- `def stretch(vert,NbVert,axis,k)`: merupakan prosedur untuk melakukan operasi *stretch* pada bangun vert dengan parameter sumbu *axis* dan factor skalar  $k$ .
- `def reflect(vert,NbVert,params)`: merupakan prosedur untuk melakukan operasi pencerminan bangun vert terhadap suatu parameter( $\mathbf{x}$ ,  $\mathbf{y}$ ,  $\mathbf{y}=\mathbf{x}$ ,  $\mathbf{y}=-\mathbf{x}$ , atau  $(\mathbf{a},\mathbf{b})$ ) .
- `def custom(vert,NbVert,matrix)`: merupakan prosedur untuk menghasilkan bayangan bangun vert sesuai input matriks transformasi oleh pengguna

Pada file *main.py* juga terdapat beberapa prosedur yang digunakan untuk menggambar bangun dan melakukan *controlling window* antara lain:

Variabel Global

- `quited`: boolean untuk melakukan quit window untuk OpenGL dan command prompt
- `NbVertex`: integer untuk menampung banyaknya titik yang ingin dimasukkan oleh pengguna
- `Vertices`: adalah list untuk menampung nilai titik-titik yang mendefinisikan suatu bangun
- `initVertices`: adalah list untuk menampung nilai titik-titik pertama yang didefinisikan oleh pengguna

Prosedur

- `def terminal_display()`: prosedur untuk menampilkan tampilan awal program dan judul program.
- `def init_draw2d()`: prosedur untuk menginisialisasi perintah menggambar, melakukan setup pada OpenGL.
- `def uninit_draw2d()`: prosedur untuk menyelesaikan/menutup inisialisasi perintah menggambar
- `def print_text(str, x, y, r, g, b)`: prosedur untuk menggambar suatu text pada layer *window* openGL.
- `def draw_grid()`: prosedur untuk menggambar *grid* beserta mencetak skala pada sumbu.
- `def draw_axis()`: prosedur untuk menggambar sumbu x dan y pada layer.
- `def drawPolygon(list_point)`: prosedur untuk menggambar polygon sesuai dengan titik-titik yang telah didefinisikan oleh pengguna.
- `def init_gl()`: prosedur untuk menginisialisasi windows baru serta menginisialisasi eksekusi openGL.
- `def shape_input()`: prosedur untuk menerima dan mengelola masukan titik-titik yang didefinisikan pengguna
- `def draw()`: prosedur *controller* yang berguna untuk mengatur apa saja yang akan ditampilkan/digambar pada windows.
- `def command_input()`: prosedur untuk menerima, mengelola dan mengarahkan masukan perintah transformasi yang didefinisikan pengguna ke prosedur yang telah disediakan.

### C. Pembagian Tugas

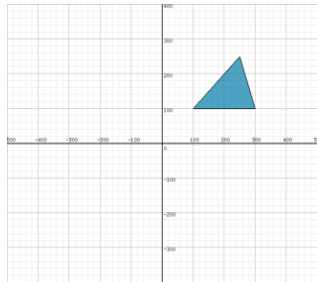
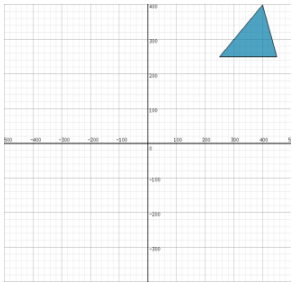
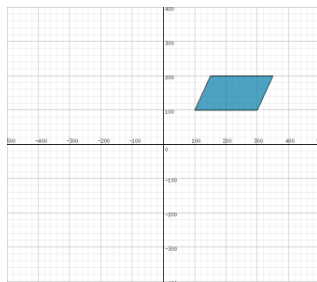
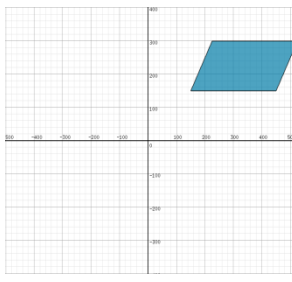
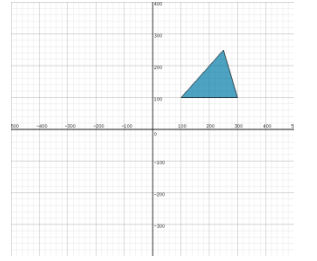
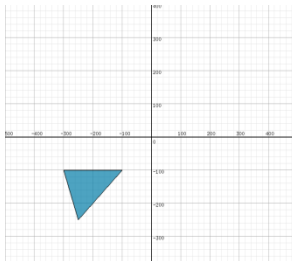
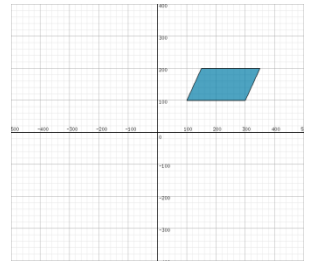
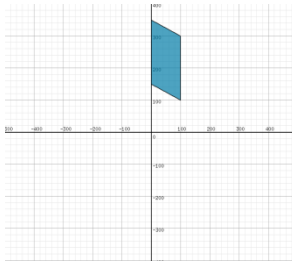
Tugas	Jenis	Kontributor
Implementasi prosedur <code>terminal_display</code>	Program	13516147
Implementasi prosedur <code>shape_input</code>	Program	13516147 13516117

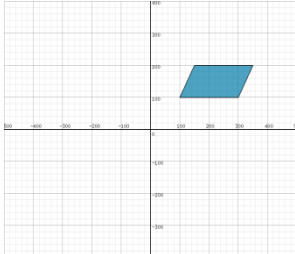
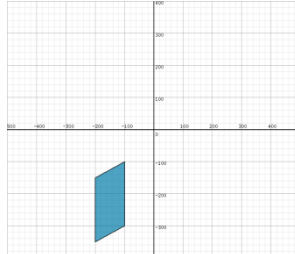
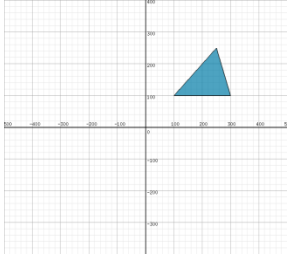
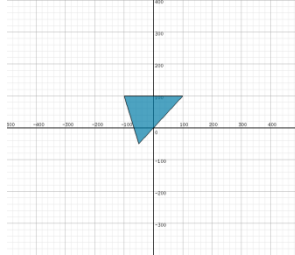
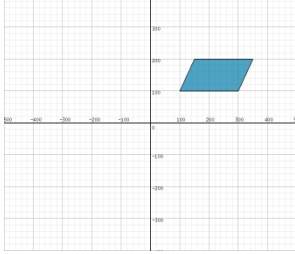
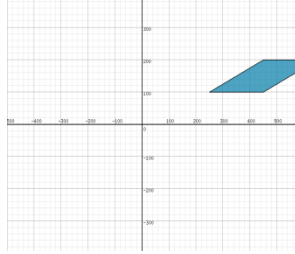
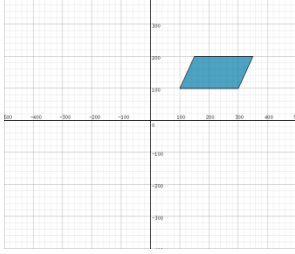
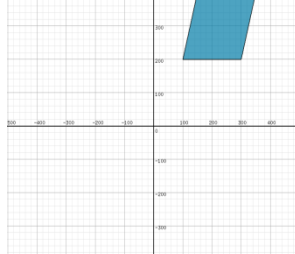
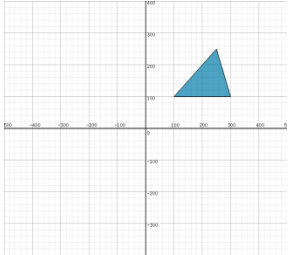
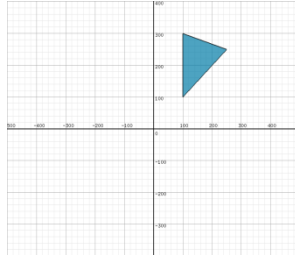
Implementasi prosedur init_draw2d dan uninit_draw2d	Program	13516117 13516147
Implementasi prosedur drawPolygon	Program	13516117
Implementasi prosedur print_text	Program	13516147
Implementasi prosedur draw_axis dan draw_grid	Program	13516147
Implementasi prosedur draw	Program	13516147 13516117
Implementasi prosedur command_input	Program	13516147 13516117
Implementasi prosedur init_gl	Program	13516147 13516117
Implementasi prosedur translate	Program	13516147 13516117
Implementasi prosedur dilate	Program	13516147 13516117
Implementasi prosedur reflect	Program	13516117 13516147
Implementasi prosedur rotate	Program	13516117 13516147
Implementasi prosedur shear	Program	13516147 13516117
Implementasi prosedur stretch	Program	13516117 13516147
Implementasi prosedur custom	Program	13516117 13516147
Penulisan Bab 1, Bab 2, Bab 3, Bab 5	Laporan	13516147 13516117
Penulisan Bab 4	Laporan	13516117

## BAB IV

### EKSPERIMEN

#### A. Interaksi Input Output

Input	Before	Command	After
Translate <150> <150>		<pre>&gt; Insert number of points : 3 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 250 &gt; Y2 : 250 &gt; Point-2 : &lt; 250.0 , 250.0 &gt; &gt; X3 : 300 &gt; Y3 : 100 &gt; Point-3 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : translate 150 150</pre>	
Dilate <1.5>		<pre>&gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : dilate 1.5</pre>	
Rotate <180> <0> <0>		<pre>&gt; Insert number of points : 3 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 250 &gt; Y2 : 250 &gt; Point-2 : &lt; 250.0 , 250.0 &gt; &gt; X3 : 300 &gt; Y3 : 100 &gt; Point-3 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : rotate 180 0 0 &gt; Insert command :</pre>	
Rotate <90> <100> <100>		<pre>&gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : rotate 90 100 100</pre>	

Reflect $\langle y = -x \rangle$		<pre> &gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : reflect y=-x </pre>	
Reflect $\langle (100,100) \rangle$		<pre> &gt; Insert number of points : 3 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 250 &gt; Y2 : 250 &gt; Point-2 : &lt; 250.0 , 250.0 &gt; &gt; X3 : 300 &gt; Y3 : 100 &gt; Point-3 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : reflect (100,100) </pre>	
Shear $\langle x \rangle \langle 1.5 \rangle$		<pre> &gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : shear x 1.5 </pre>	
Stretch $\langle y \rangle \langle 2 \rangle$		<pre> &gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : stretch y 2 </pre>	
Custom $\langle 0 \rangle \langle 1 \rangle$ $\langle 1 \rangle \langle 0 \rangle$		<pre> &gt; Insert number of points : 3 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 250 &gt; Y2 : 250 &gt; Point-2 : &lt; 250.0 , 250.0 &gt; &gt; X3 : 300 &gt; Y3 : 100 &gt; Point-3 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : custom 0 1 1 0 </pre>	

<p>Multiple &lt;3&gt;</p> <p>-Rotate &lt;90&gt; &lt;0&gt;</p> <p>&lt;0&gt;</p> <p>-Translate -100 -100</p> <p>-Reflect &lt;y&gt;</p>		<pre> &gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : multiple 3 &gt; rotate 90 0 0 &gt; translate -100 -100 &gt; reflect y </pre>	
<p>Reset</p>		<pre> &gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : multiple 3 &gt; rotate 90 0 0 &gt; translate -100 -100 &gt; reflect y &gt; Insert command : reset </pre>	
<p>Exit</p>		<pre> &gt; Insert number of points : 4 &gt; X1 : 100 &gt; Y1 : 100 &gt; Point-1 : &lt; 100.0 , 100.0 &gt; &gt; X2 : 150 &gt; Y2 : 200 &gt; Point-2 : &lt; 150.0 , 200.0 &gt; &gt; X3 : 350 &gt; Y3 : 200 &gt; Point-3 : &lt; 350.0 , 200.0 &gt; &gt; X4 : 300 &gt; Y4 : 100 &gt; Point-4 : &lt; 300.0 , 100.0 &gt; &gt; Insert command : multiple 3 &gt; rotate 90 0 0 &gt; translate -100 -100 &gt; reflect y &gt; Insert command : reset &gt; Insert command : exit &gt; Goodbye ! </pre>	<p>Goodbye!</p>

## B. Analisis Hasil

Hasil perhitungan tiap transformasi pada eksperimen

### a) Translasi 150 150

$$P1(100,100) \rightarrow P1'(250,250)$$

$$P2(250,250) \rightarrow P2'(400,400)$$

$$P3(300,100) \rightarrow P3'(450,250)$$

### b) Dilatasi

$$P1(100,100) \rightarrow P1'(150,150)$$

$$P2(150,200) \rightarrow P2'(275,300)$$



$$P3(350,200) \rightarrow P3'(525,300)$$

$$P4(300,100) \rightarrow P4'(450,150)$$

c) **Rotasi**

$$P1(100,100) \rightarrow P1'(-100,-100)$$

$$P2(250,250) \rightarrow P2'(-250,-250)$$

$$P3(300,100) \rightarrow P3'(-300,-100)$$

d) **Refleksi**

$$P1(100,100) \rightarrow P1'(-100,-100)$$

$$P2(150,200) \rightarrow P2'(-200,-150)$$

$$P3(350,200) \rightarrow P3'(-200,-350)$$

$$P4(300,100) \rightarrow P4'(-100,-300)$$

e) **Shear**

$$P1(100,100) \rightarrow P1'(250,100)$$

$$P2(150,200) \rightarrow P2'(450,200)$$

$$P3(350,200) \rightarrow P3'(650,200)$$

$$P4(300,100) \rightarrow P4'(450,100)$$

f) **Stretch**

$$P1(100,100) \rightarrow P1'(100,200)$$

$$P2(150,200) \rightarrow P2'(150,400)$$

$$P3(350,200) \rightarrow P3'(350,400)$$

$$P4(300,100) \rightarrow P4'(300,200)$$

g) **Custom**

$$P1(100,100) \rightarrow P1'(100,100)$$

$$P2(250,250) \rightarrow P2'(250,250)$$

$$P3(300,100) \rightarrow P3'(100,300)$$

Berdasarkan hasil perhitungan manual dan hasil visualisasi output dari program dapat dibandingkan bahwa hasil transformasi bidang dua dimensi yang ditampilkan sudah sesuai dengan perhitungan. Dapat disimpulkan bahwa program sudah memiliki akurasi yang cukup baik dalam menyimulasikan transformasi linier pada bidang dua dimensi.

## **BAB V**

### **KESIMPULAN DAN SARAN**

#### **A. Kesimpulan**

Pada tugas ini kami telah berhasil meyimulasikan transformasi linier pada bidang dua dimensi dengan baik disertai dengan animasi yang mendukung tampilan program ini.

Melalui pembuatan program simulasi transformasi linier pada bidang dua dimensi kami mendapatkan beberapa kesimpulan. Hal pertama yang dapat kami simpulkan adalah bahwa transformasi linier akan lebih mudah dipahami ketika terdapat visualisasi grafis. Selain itu, transformasi linier juga dapat diterapkan dalam kehidupan sehari-hari seperti pada pengaturan tata letak cermin dan kaca spion, pembentukan pola ukiran/ karya seni, serta dalam teknologi pemrosesan citra/gambar.

#### **B. Saran**

Menurut kami program simulasi transformasi linier yang kami buat dapat dikembangkan lebih lagi. Program sejenis dapat segera diimplementasikan sebagai salah satu sarana pembelajaran interaktif untuk mata kuliah aljabar geometri atau pelajaran geometri di tingkat SMA. Simulasi visual dari transformasi sangat memudahkan pelajar serta mahasiswa untuk memahami proses yang sebenarnya terjadi dalam transformasi linier.

Selain itu program ini dapat dikembangkan lebih lagi dengan meyimulasikan transformasi linier pada ruang vektor 3 dimensi. Dengan demikian program ini dapat dimanfaatkan sebaik-baiknya untuk membantu proses pembelajaran transformasi linier.

## REFERENSI

Anton, H., & Rorres, C. (2013). *Elementary Linear Algebra*. Wiley

Rosita Ayu (2012,November). *Transformasi Geometri Matematika*

Retrieved from <http://rositayu1.blogspot.co.id/2012/11/transformasi-geometri-matematika.html> diakses pada 9 November 2017 pukul 20.17.

Agus (2011,Juli) *Pengenalan OpenGL*

Retrieved from <http://agussale.com/penjelasan-mengenai-apa-itu-opengl> diakses pada 7 November 2017 pukul 08.32.