APLIKASI GRAFIKA KOMPUTER MENGGUNAKAN *PASCAL*

Tugas Mata Kuliah
GRAFIKA KOMPUTER
KELAS B
Semester Genap TA. 2018/2019

Oleh:

1.	Ninditya Salma Nur Aini	123160175	(Ketua)
2.	Dymas Surya Mukti	123160177	(Anggota)
3.	Scipo Surbakti	123160041	(Anggota)

JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
YOGYAKARTA

2018

HALAMAN PENGESAHAN PRESENTASI

APLIKASI GRAFIKA KOMPUTER MENGGUNAKAN BAHASA *PASCAL*

Disusun oleh:

KELOMPOK 3

Nama	No.Mhs.	Tanda Tangan
Ninditya Salma Nur Aini	123160175	
Dymas Surya Mukti	123160177	
Scipo Surbakti	123160041	

telah dipresentasikan pada tanggal 30 April 2019

Menyetujui
Dosen Pengampu

Herry Sofyan, S.T., M.Kom

DAFTAR ISI

BAB I
DASAR TEORI

1.1 Pendahuluan

Grafika Komputer adalah bagian dari ilmu komputer yang berkaitan dengan pembuatan dan manipulasi gambar (visual) secara digital. Bentuk sederhana dari grafika komputer adalah grafika komputer 2D yang kemudian berkembang menjadi grafika komputer 3D, pemrosesan citra (image processing), dan pengenalan pola (pattern recognition). Grafika komputer sering dikenal juga dengan istilah visualisasi data.

Grafika komputer merupakan bagian dari ilmu komputer karena dalam setiap pembuatan grafik membutuhkan perhitungan dan skala pada setiap pembuatannya, serta membutuhkan peralatan – peralatan baik itu berupa hardware maupun software, karena jika tidak maka pada setiap pembuatannya tidak dapat dilakukan bergitu juga dengan keluarannya maka tidak dapat melihat hasil yang dibuat.

Manfaat dari Grafika Komputer itu sendiri meliputi :

1. Entertainment, misalnya dalam pembuatan film animasi. Terminator II dan Titanic merupakan contoh film non animasi yang memanfaatkan efek-efek grafika komputer yang sangat canggih.
2. Visualisasi Proses, misalnya menggambarkan layout kinerja proses pabrik, atau proses-proses yang dalam modul ajar.
3. Visualisasi Hasil Penelitian, seperti menggambar grafik performance, grafik perubahan bahkan grafik model hasil simulasi dan implementasi program.
4. Bersama-sama dengan image processing digunakan sebagai algoritma identifikasi, seperti yang dilakukan pada kepolisian untuk menggambarkan wajah seseorang secara 3D dan identifikasi seseorang.

1.2 Sistem Grafika

Grafika komputer memiliki banyak sistem baik itu pada perangkat lunak maupun perangkat keras. Untuk perangkat lunak operator tidak dapat berinteraksi dengan komputer atau bersifat statis, operator juga bisa mengendalikan, isi, format, bentuk, ukuran serta warna. Selain itu juga sistem pada grafika komputer bisa mensimulasikan real world pada layar komputer.

Pada perangkat kerasnya sendiri berupa teknologi display, raster-scan display, random-scan display dan juga berupa alat input interaktif yang lain.

1.2.1 Teknologi Display

Teknologi display sebagai alat utama untuk menampilkan output pada sistem grafika adalah video monitor. Pada umumnya menggunakan perancangan cathode-ray-tube (CRT).

Cara kerja monitor dengan teknologi CRT adalah sebagai beriku:

1. Elektron dipancakan dari elektron gun yang melewati *focusing system* (sistem untuk menentukan fokus) dan diteruskan ke *deflection system* (sistem untuk mengatur pembelokan) sehingga pancaran elektro mencapai posisi tertentu pada layar monitor yang dilapisi fosfor.
2. Lapisan fosfor yang ada dimonitor yang dikenai pancaran elektron pada posisi tertentu akan memancarkan sinar kecil pada setiap posisi yang berhubungan dengan pancaran elektron. Pancaran sinar dari lapisan fosfor ini cepat hilang, untuk mempertahankannya diperluka proses *refreshing* (menembakan eletron berulang kali pada posisi yang sama).

Jumlah maksimum titik yang dapat ditampilkan pada layar monitor disebut resolusi. Resolusi yaitu jumlah titik yang dapat ditampilkan per senti meter menurut arah horisontal dan vertikal. Semakin tinggi resolusinya semakin bagus dan lebih realistis terutama untuk menampilkan citra dan grafik.

1.2.2 Raster-Scan System

Pada umumnya terdiri dari beberapa unit pemroses. Kecuali CPU, digunakan processor khusus video controler atau display controler yang berfungsi untuk mengontrol operasi dari peralatan display.

Video Controler

Operasi dasar pada video controler, dua register digunakan untuk menyimpan koordinat pixel pada layar. Nilai dari posisi pixel yang disimpan pada frame buffer diambil dan digunakan untuk mengatur intensitas dari pancaran elektorn. Kecuali refreshing dasar, beberapa operasi dapat dilakukan. Video controler dapat mengambil intensitas pixel dari area memori yang berbeda pada siklus refreshing yang berbeda.

Pada sistem dengan kualitas tinggi, sering digunakan dua frame buffer, sehingga satu buffer digunakan untuk refreshing, sedangkan yang lain diisi dengan nilai intensitas. Kedua buffer dapat saling tukar untuk melakukan fungsi tersebut, sehingga dapat memenuhi kebutuhan mekanisme yang cepat, seperti animasi real-time. Untuk merefresh suatu display dengan ukuran 1024 x 768 pixel pada suatu refresh rate 60 hz memerlukan suatu akses memori setiap 1/(1024*768)60) seconds = 21 ns. Sedangkan untuk men-set komponen

warna red, green, dan blue video controler menggunakan suatu look-up Table yang digunakan untuk mengkonversi warna ke kekuatan signal.

1.2.3 Random Scan System

Program aplikasi dimasukkan dan disimpan dalam sistem memori dari suatu perangkat lunak aplikasi grafika. Perintah grafik pada program aplikasi diterjemahkan ke dalam display file yang disimpan dalam sistem memori. Kemudian display file diakses oleh display processor untuk ditampilkan pada layar monitor. Display processor mengulang kembali setiap perintah dari program pada saat dilakukan refreshing. Pola grafik digambar pada random-scan system dengan menembakkan elektron langsung sesuai komponen garis pada layar monitor. Garis ditentukan oleh nilai dari dua koordinat titik awal dan titik akhir.

1.2.4 Peralatan Input Interaktif

Beberapa macam alat input interaktif yang melengkapi grafika dapat dibagi dalam lima kelompok :

1. Keyboard : Untuk memasukkan karakter atau string.
2. Locator : Untuk mengenali posisi atau orientasi
   (mouse, trackball dan spaceball, joystick, glove, Digitizer)
3. Pick : Untuk menyeleksi entity suatu tampilan (touch screen, light pen)
4. Valuator : Digunakan untuk memasukkan bilangan real
5. Choice : Untuk menyeleksi dari suatu action atau pilihan yang tersedia.

1.3 Output Primitif

Output/Grafis primitif adalah bentuk geometri dasar yang dapat digunakan untuk membentuk obyek yang lebih komplek. Dengan memasukkan output primitif tersebut sebagai stuktur yang lebih kompleks. Setiap output primitif mempunyai data koordinat dan informasi lain tentang bagaimana cara object ditampilkan pada layar. Titik dan garis lurus adalah bentuk geometri paling sederhana dan komponen gambar.

1.3.1 Titik dan Garis

1.3.1.1 Titik

Titik merupakan satuan gambar/grafis yang terkecil. Dengan menggambar titik maka kita dapat menggambar obyek apapun. Termasuk bentuk geometri dibawah merupakan bentuk –bentuk yang pada dasarnya berasal dari titik-titik. Operasi titik ini sering digunakan pada pengolahan citra (Image processing). Setiap titik pada monitor memiliki parameter : koordinat dan warna.

Sistem koordinat yang dipakai bisa *Polar Coordinates* atau *Cartesian Coordinates*. Biasanya dalam pemrograman grafis, yang paling umum digunakan adalah *Cartesian Coordinates*. Dalam *Cartesian Coordinates*, titik didefinisikan sebagai kombinasi dua bilangan yang menentukan posisi tersebut dalam koordinat x dan y (2D).

Pembentukan titik dilakukan dengan mengkonversi sebuah koordinat tunggal yang diberikan oleh sebuah program aplikasi dalam suatu operasi tertentu dengan menggunakan peralatan output, sebagai contoh monitor. Pada Random scan-system, instruksi tersebut dikonversikan menjadi arah pancaran sinar pada posisi layar. Untuk raster system hitam – puti, sebuah titik dengan lokasi tertentu pada layar dibuat dengan cara memberi nilai bit 1 pada frame buffer. Untuk system RGB, frame buffer diisi dengan nilai intensitas warna pixel yang sesuai dengan yang akan ditampilkan pada posisi tertentu di layer.

1.3.1.2 Garis

Garis adalah kumpulan titik-titik/pixel yang tersusun secara lurus dan linier dari titik awal sampai titik akhir. Peralatan output diarahkan untuk mengisi posisi – posisi tersebut antara kedua ujung. Untuk peralatan analog seperti plotter dan random-scan display garis lurus dapat dihasilkan dengan halus. Pada peralatan digital setiap segmen garis lurus dibuat dalam bentuk titik diskrit sepanjang garis lurus dan setiap posisi koordinat diskrit dihitung dengan menggunakan persamaan garis lurus.Perhitungan titik yang menghasilkan nilai pecahan dibulatkan menjadi bilngan integer dengan sisterm pembulatan nilai keatas. Posisi pixel dapat digambarkan sesuai nilai scan-line dan nilai kolom. Nilai scan-line dimulai dari 0 pada bagian bawah layar dan kolom pixel dimulai dari 0 pada bagian kiri layar.

1.3.2 Algoritma Pembentuk Garis

Persamaan garis lurus pada koordinat kartesius diwujudkan dalam persamaan garis adalah **y=m.x+b** dimana : *m* : adalah kemiringan garis ; *b* : intercept y. Jika dimisalkan pada dua titik (x_0,y_0 dan x_1,y_1) akan dibuat sebuah garis lurus, kita dapat menentukan nilai *m* dan *b* dengan persamaan berikut:

$$m = \frac{y1 - y0}{x1 - x0}$$

$$b = y_1 - m.x_1$$

Algoritma untuk menggambar garis pada komputer didasarkan pada dua persamaan di atas. dimana m adalah gradien atau kemiringan garis tersebut.

1.3.2.1 Algoritma Digital Diferential Analyzer (DDA)

Digital Diferential Analyzer (DDA) adalah algoritma pembentukan garis berdasarkan perhitungan □*y* atau □*x*, menggunakan perasamaan :

$$\Delta y = m.\Delta x \quad \text{atau} \quad \Delta x = \frac{\Delta y}{m}$$

Garis dibuat menggunakan dua ujung garis, yaitu titik awal (x_1, y_1) dan titik akhir (x_2, y_2). Setiap koordinat titik (x_k, y_k) yang membentuk garis diperoleh dari perhitungan, kemudian hasil perhitungan dikonversikan menjadi nilai integer. Prinsip algoritma ini adalah mengambil nilai integer terdekat dengan jalur garis berdasarkan atas sebuah titik yang telah ditentukan sebelumnya (titik awal garis).

Langkah- langkah pembentukan garis algoritma DDA :

1. Tentukan dua titik yang akan dihubungkan dalam pembentukan garis.
2. Tentukan salah satu sebagai titik awal (x_1, y_1) dan titik akhir (x_2, y_2).
3. Hitung $dx = x_2 - x_1$ dan $dy = y_2 - y_1$
4. Tentukan *step*, yaitu jarak maksimum jumlah penambahan nilai *x* atau nilai *y*, dengan ketentuan:
   - bila $|dx| > |dy|$ maka $step = |dx|$
   - bila tidak, maka $step = |dy|$
5. Hitung penambahan koordinat pixel dengan persamaan:
   - $x\_inc = dx / step$
   - $y\_inc = dy / step$
6. Koordinat selanjutnya ($x+x\_inc$, $y+y\_inc$)
7. Plot pixel pada layar, nilai koordinat hasil perhitungan dibulatkan
8. Ulangi nomor 6 dan 7 untuk menentukan posisi pixel berikutnya sampai $x = x_1$ atau $y = y_1$.

1.3.2.2 Algoritma Bressenham

Algoritma Bressenham tidak membulatkan nilai posisi pixel setiap waktu. Algoritma Bressenham hanya menggunakan penambahan nilai integer yang juga dapat diadaptasi untuk menggambar lingkaran. Algoritma garis Bressenham disebut juga MidPoint Line Algorithm adalah algoritma konversi penambahan nilai integer yang juga dapat diadaptasi untuk menggambar sebuah lingkaran.

Langkah – langkah pembentukan garis berdasarkan algoritma Bressenham

1. Tentukan dua titik yang akan dihubungkan dalam pembentukan garis.
2. Tentukan salah satu sebagai titik awal (x_1, y_1) dan titik akhir (x_2, y_2).
3. Hitung *dx*, *dy*, 2*dy* dan 2*dy* - 2*dx*
4. Hitung parameter : $p_o = 2dy - dx$
5. Untuk setiap x_k sepanjang jalur garis, dimulai dengan k=0

- bila $p_k < 0$ maka titik selanjutnya adalah: (x_k+1, y_k) dan $p_k+1 = p_k + 2dy$
- bila tidak, titik selanjutnya adalah: (x_k+1, y_k+1) dan $p_k+1 = p_k + 2dy – 2dx$

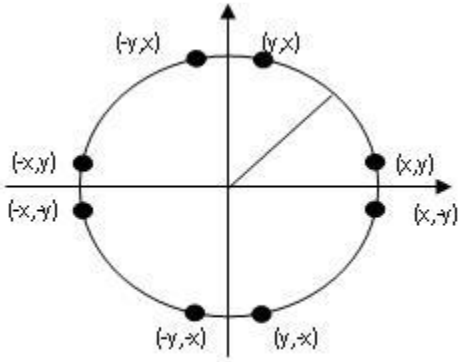
6. Ulangi nomor 5 untuk menentukan posisi pixel berikutnya, sampai $x = x_1$ atau $y = y_1$.

1.3.3 Algoritma pembentuk lingkaran

`Secara umum, pembentukan lingkaran dapat dirumuskan sebagai berikut : x2+y2=R2. Dalam membentuk sebuah lingkaran terdapat beberapa cara, hanya bayka pula yanng tidak efisien. Lingkaran dapat dibentuk dengan mengggambar seperempat, ini karena bagian lain dapat dibuat sebagai bagian yang simetris.

1.3.3.1 Simetris Delapan titik

Disini pembuatan lingkaran dapat dilakukan dengan menentukan satu titik awal. Anggaplah satu titik awal tersebut (X, y), maka akan terdapat tiga posisi lain, sehingga dapat diperoleh delapan titik. Dengan demikian sebenarnya hanya perlu menghitung segmen 45’ dalam menentukan keseluruhan lingkaran. Lewat pusat titik tertentu, delapan titik simetris dapat ditampilkan dengan prosedur circle point dibawah ini:



Gambar Delapan Titik Simetris Pada Lingkaran

1.3.3.2 Algoritma Lingkaran Midpoint

Algoritma Lingkaran Midpoint juga disebut algoritma lingkaran Bressenham. Bressenham mengembangkan generator lingkaran yang cukup efisien. Algoritma yang digunakan membentuk semua titik berdasarkan titik pusat dengan penambahan semua jalur sekeliling lingkaran. Algoritma ini diturunkan dari algoritma Midpoint untuk pembentukan garis. Dalam hal ini hanya diperhatikan bagian 45’ dari suatu lingkaran, yaitu oktan kedua dari x=0 ke x= r/√2, dan menggunakan CirclePoints untuk menampilkan titik dari seluruh lingkaran.

Langkah langkah untuk membentuk lingkaran algoritma Circle Midpoint:

1. Tentukan radius r dengan titk pusat lingkaran(x_c,y_c) kemudian diperoleh (x_o,y_o)=(0,r)
2. Hitung nilai dari parameter P_0=5/4 r
3. Untuk setiap posisi x_k dimulai dengan k=0 berlaku ketentuan :

- Bila $p_k < 0$ maka titik selanjutnya adalah (x_k+1,y_k) dan $p_{k+1} = p_k + 2\ x_{k+1} + 1$
- Bila tidak, titik selanjutnya adalah (x_k + 1,y_k – 1) dan $p_{k+1} = p_k + 2\ x_{k+1} + 1 – 2\ y_{k+1}$ dimana $2\ x_{k+1} = 2x_k + 2$ dan $2\ y_{k+1} = 2y_k – 2$

4. Tentukan titik simetris pada ketujuh oktan lain.
5. Gerakan setiap posisi pixel (x,y) pada garis lingkaran dengan titik pusat (x_c,y_c) dan plot nilai koordinat : $x = x + x_c$, $y = y + y_c$
6. Ulangi langkah 3 sampai dengan 5 hingga $x \geq y$

1.3.4 Fill Area Primitif

Fill area (pengisian area) output primitif standar umumnya adalah warna solid atau pola raster. Bentuk poligon paling mudah untuk diproses karena memiliki garis batas yang jelas. Ada dua macam dasar pendekatan fiil area pada sistem raster :

- Dengan menentukan overlap interval untuk scan line yang melintasi area (digunakan untuk bentuk – bentuk dasar seperti poligon, lingkarang, elips, dan kurva sederhana lainnya)
- Dengan memulai dari titik tertentu pada posisi di dalam poligon dan menggambar secara menyebar kepinggir sampai batas poligon.

1.3.4.1 Algoritma Scan Line

Titik potong diurutkan dari kiri ke kanan. Posisi yang berhubungan pada frame buffer antara sepasang titik potong diberi warna tertentu. Posisi empat pixel sebagai titik potong antara garis batas polygon ditentukan oleh dua buah pixel pada koordinat dari x=10 ke x=15 dan dari x=23 ke x=35.

Garis scan line yang melalui sebuah vertex (titik) yang merupakan perpotongan dua buah garis poligon dihitung sebagai dua titik perpotongan pada scan line. Algoritma untuk metode scan line ini cukup rumit karena harus memperhatikan setiap titik potong poligon dengan scan line.

1.3.4.2 Inside – Outside Test

Algoritma area filling dan proses grafika seringkali memerlukan identifikasi ruang bagian dalam suatu objek. Pada poligon standar tidak ada perpotongan pada objek tersebut. Untuk mengidentifikasi bagian dalam dari poligon biasanya menggunakan metode straightward, tetapi dalam aplikasi grafik dapat dibuat spesifikasi suatu sekuen vertices suatu fill area, termasuk yang menghasilkan garis potong.

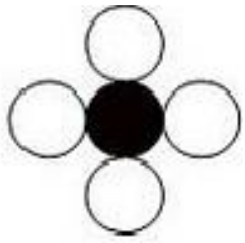
Aturan odd-even yang juga disebut dengan aturan odd parity atau even-odd, secara konseptual membuat gambar dengan garis dari posisi P ke titik dengan jarak tertentu diluar koordinat objek dan menghitung edge poligon yang berpotongan dengan garis.

1.3.4.3 Algoritma Boundary Fill

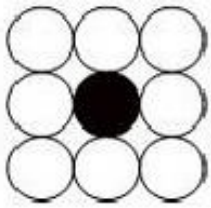
Metode ini bermanfaat untuk paket aplikasi grafik interaktif dimana titik dalam dapat dengan mudah ditentukan. Prosedur boundary fill menerima input koordinat suatu titik(x,y), warna isi dan garis batas. Dimulai dari titik (x,y), kemudian prosedur memeriksa posisi titik tetangga, yaitu apakah merupakan warna batas. Bila tidak, maka titik tersebut digambar dengan warna isi. Selanjutnya periksa lagi posisi warna dan titik tetangganya. Proses ini diulangi sampai semua titik pada batas diperiksa.

Ada dua metode untuk mengecek pixel tetangga dari suatu titik, yaitu :

- Metode 4-connected dimana posisi tetangga yang dicek berada di kiri, kanan, atas, dan bawah dari titik yang diketahui.



- Metode 8-connected dimana posisi pixel tetangga yang dicek berada di kiri, kanan, atas, bawah, dan keempat posisi diagonal dari titik yang diketahui.



1.3.4 Fill Area Primitif

Prosedur berikut menampilkan metode rekursif mengisi 4 bidang dengan intensitas pada parameter fill.

1.3.4.4 Algoritma Flood Fill

Metode ini dimulai pada titik (x,y) dan mendefinisikan seluruh pixel pada bidang tersebut dengan warna yang sama. Pada metode ini warna pixel yang diganti dengan warna isi yang baru adalah warna lama yang telah ditentukan, tanpa melihat warna batas. Bila bidang yang akan diisi warna memiliki beberapa warna, pertama tama yang dibuat adalah membuat nilai pixel baru, sehingga smua pixel memiliki warna yang sama.

Prosedur Algoritma Flood Fill berikut ini menampilkan metode rekursif untuk pengisian area dengan menggunakan metode 4-connection dengan parameter posisi titik (x,y), warna isi baru dan warna asal yang diganti.

1.4 Atribut Output Primitif

Pada umumnya, setiap parameter yang memberi pengaruh pada output primitif ditampilkan sesuai dengan parameter atribut.

1.4.1 Atribut Titik

Titik merupakan satuan gambar/grafis yang terkecil. Dengan menggambar titik maka kita dapat menggambar obyek apapun. Termasuk bentuk geometri dibawah merupakan bentuk –bentuk yang pada dasarnya berasal dari titik-titik. Operasi titik ini sering digunakan pada pengolahan citra (Image processing). Setiap titik pada monitor memiliki parameter : koordinat dan warna.

1.4.2 Atribut Kurva

Parameter untuk atribut kurva sama dengan atribut segmen garis. Kurva dapat ditampilkan dengan berbagai warna, tebal, dot-dash(Style garis) dan pilihan pen atau brush. Selain itu untuk pengisian suatu bidang tertentu termasuk memilih warna antara solid dan pattern tertentu dan memilih warna pattern yang ada.

1.4.3 Warna dan Grayscale

Ketika suatu sistem menyediakan opsi warna, suatu parameter akan memberikan indeks warna awal yang dimasukkan ke dalam daftar nilai atribut sistem. suatu prosedur polyline kemudian akan menampilkan garis dengan warna tersebut dengan cara mengatur warna tersebut d frame buffer pada lokasi piksel sepanjang garis tersebut dengan menggunakan prosedur setPixel.

1.5 Transformasi Dua Dimensi

- Langkah-Langkah Transformasi Viewing 2 Dimensi

1. Pembentukan scene pada WC menggunakan output primitif atribut.
2. Untuk mendapatkan orientasi tertentu dari window, maka ditentukan sistem VC dua dimensi pada WC. Frame digunakan untuk melengkapi penentuan orientasi dari windows persegi empat . Setelah frame dibentuk dapat ditransformasikan ke dalam WC untuk menampilkan koordinat.
3. Kemudian viewport ditentukan dalam normalized NVC (pada batasan antara 0 dan 1) dan memetakan deskripsi VC dari scene pada Normalized Coordinate.
4. Akhirnya dilakukan clipping (pemotongan) pada semua gambar yang ada diluar viewport.

- Macam-macam transformasi:

1. Transformasi Objek, yang ditransformasikan titik-titik yang menyusun objek tersebut.
2. Transformasi Koordinat, yang diubah system koordinatnya sehingga objek mengalami transformasi dikarenakan perubahan system koordinat tersebut.

- Tujuan Transformasi :

1. Merubah atau menyesuaikan komposisi pandangan.
2. Memudahkan membuat objek yang simetris
3. Melihat objek dari sudut pandang berbeda
4. Memindahkan satu atau beberapa objek dari satu tempat ke tempat lain, biasanya digunakan pada animasi computer.

1.5.1 Translasi

Adalah memindahkan suatu objek sepanjang garis lurus dari suatu lokasi koordinat tertentu ke lokasi yang lain. Translasi dilakukan dengan penambahan translasi pada suatu titik koordinat dengan translation vector atau shift, yaitu(t_x, t_y), dimana :

t_x : translasi vector sumbu x

t_y : translasi vector sumbu y

Koordinat baru titik yang ditranslasi dapat diperoleh dengan menggunakan rumus :

$x' = x + t_x$

$y' = y + t_y$

Dimana (x, y) adalah koordinat asal suatu objek sedangkan (x', y') koordinat baru objek setelah ditranslasi.

Contoh :

Untuk menggambarkan translasi suatuobjek yang berupa segitiga dengan koordinat A(5,6) , B(8,9), dan C(4,7) dengan translation vector (2,3), pertama-tama dihitung koordinat hasil translasi satu demi satu.

Titik A (5,6) Vektor (2,3)

$x' = x + t_x$ $y' = y + t_y$

$= 5 + 2$ $= 6 + 3$

$= 7$ $= 9$

Hasil translasi titik A' (7,9)

Titik B (8,9) Vektor (2,3)

$x' = x + t_x$ $y' = y + t_y$

$= 8 + 2$ $= 9 + 3$

$= 10$ $=12$

Hasil translasi titik B' (10,12)

Titik C (4,7) Vektor (2,3)

$x' = x + t_x$ $y' = y + t_y$

$= 4 + 2$ $= 7 + 3$

$= 6$ $= 10$

Hasil translasi titik C' (6,10)

Dengan demikian hasil translasi segitiga dengan koordinat A'(7,9) B'(10,12) C'(6,10). Segitiga yang baru sama bentuknya dengan segitiga yang lama.

1.5.2 Rotasi

Rotasi dua dimensi pada suatu objek akan memindahkan objek tersebut menurut garis melingkar. Untuk melakukan rotasi diperlukan sudut rotasi θ dan pivot point (x_p,y_p). Nilai positif dari sudut rotasi menentukan arah rotasi berlawanan dengan arah jarum jam. Sedangkan sudut rotasi negatif menurut objek searah dengan jarum jam. Rotasi suatu titik terhadap pivot point (xp, yp) menggunakan bentuk trigonemetri, sebagai berikut :

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

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

contoh :

Untuk menggambarkan rotasi suatu objek yang berupa segitiga dengan koordinat A(12,13) B(22,23) C(22,13) dengan sudut rotasi 30^0 terhadap titik pusat koordinat Cartesian (2,3), dilakukan dengan menghitung koordinat hasil rotasi tiap titik satu demi satu.

Perhitungan :

Titik A (12,13) K. Cartesian (2,3)

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

$= 2 + (12-2) \cos 30 - (13-3) \sin 30$

$= 2 + 10 . 0,9 - 10 . 0,5$

$= 2 + 9 - 5$

$=17$

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

$= 3 + (12-2) \sin 30 + (13-3) \cos 30$

$= 3 + 10 . 0,5 + 10 . 0.9$

$= 3 + 5 + 9$

$= 6$

Hasil rotasi titik A' (6.17)

Titik B (22,23) K. Cartesian (2,3)

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

$= 2 + (22-2) \cos 30 - (23-3) \sin 30$

$= 2 + 20 . 0,9 - 20 . 0,5$

$= 2 + 18 - 10$

$= 10$

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

$= 3 + (22-2) \sin 30 + (23-3) \cos 30$

$= 3 + 20 . 0,5 + 20 . 0.9$

$= 3 + 10 + 18$

$= 31$

Hasil rotasi titik B' (10,31)

Titik C (22,13) K. Cartesian (2,3)

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

$= 2 + (22-2) \cos 30 - (13-3) \sin 30$

$= 2 + 20 . 0,9 - 10 . 0,5$

$= 2 + 18 - 5$

$= 15$

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

$= 3 + (22-2) \sin 30 + (13-3) \cos 30$

$= 3 + 20 . 0,5 + 10 . 0.9$

$= 3 + 10 + 9$

$= 22$

Hasil rotasi titik C' (15,22)

Dengan demikian hasil rotasi segitiga dengan koordinat A'(6,17) B'(10,31) C'(15,22) Segitiga yang baru sama bentuknya dengan segitiga yang lama.

1.5.3 Skala

Transformasi skala adalah perubahan ukuran suatu objek. Koordinat baru diperoleh dengan melakukan perkalian nilai koordinat dengan skala factor, yaitu (s_x,s_y) dimana

s_x = skala factor untuk sumbu x

s_y = skala factor untuk sumbu y.

Koordinat baru titik yang diskala dapat diperoleh dengan :

$x' = x . s_x$; $y' = y . s_y$;

Skala factor s_x dan s_y dapat diberikan sembarang nilai positif. Nilai lebih dari 1 menandakan bahwa sebuah objek diperbesar sedangkan nilai-nilai kurang dari 1 menunjukkan bahwa objek diperkecil.

Contoh :

Untuk menggambarkan skala suatu objek yang merupakan segi empat dengan koordinat A(3,4), B(6,4), C(3,3), D(6,3) diskala dengan skala faktor (2,2), pertama-tama dihitung koordinat hasil skala satu demi satu. Perhitungan :

Titik A (3,4) S.Vektor (2,2)

$x' = x . s_x$ $y' = y . s_y$

$= 3 . 2$ $= 4 . 2$

$= 6$ $= 8$

Hasil skala titik A' (6,8)

Titik B (6,4) S.Vektor (2,2)

$x' = x \cdot s_x$ $y' = y \cdot s_y$

$= 6 \cdot 2$ $= 4 \cdot 2$

$= 12$ $= 8$

Hasil skala titik B' (12,8)

Titik C (3,3) S.Vektor (2,2)

$x' = x \cdot s_x$ $y' = y \cdot s_y$

$= 3 \cdot 2$ $= 3 \cdot 2$

$= 6$ $= 6$

Hasil skala titik C' (6,6)

Titik D (6,3) S.Vektor (2,2)

$x' = x \cdot s_x$ $y' = y \cdot s_y$

$= 6 \cdot 2$ $= 3 \cdot 2$

$= 12$ $= 6$

Hasil skala titik D' (12,6)

Dengan demikian hasil skala persegi panjang dengan koordinat A'(6,8) B'(12,8) C'(6,6) D' (12,6). Persegi panjang yang baru sama bentuknya dengan persegi panjang yang lama.

1.5.4 Matriks Transformasi

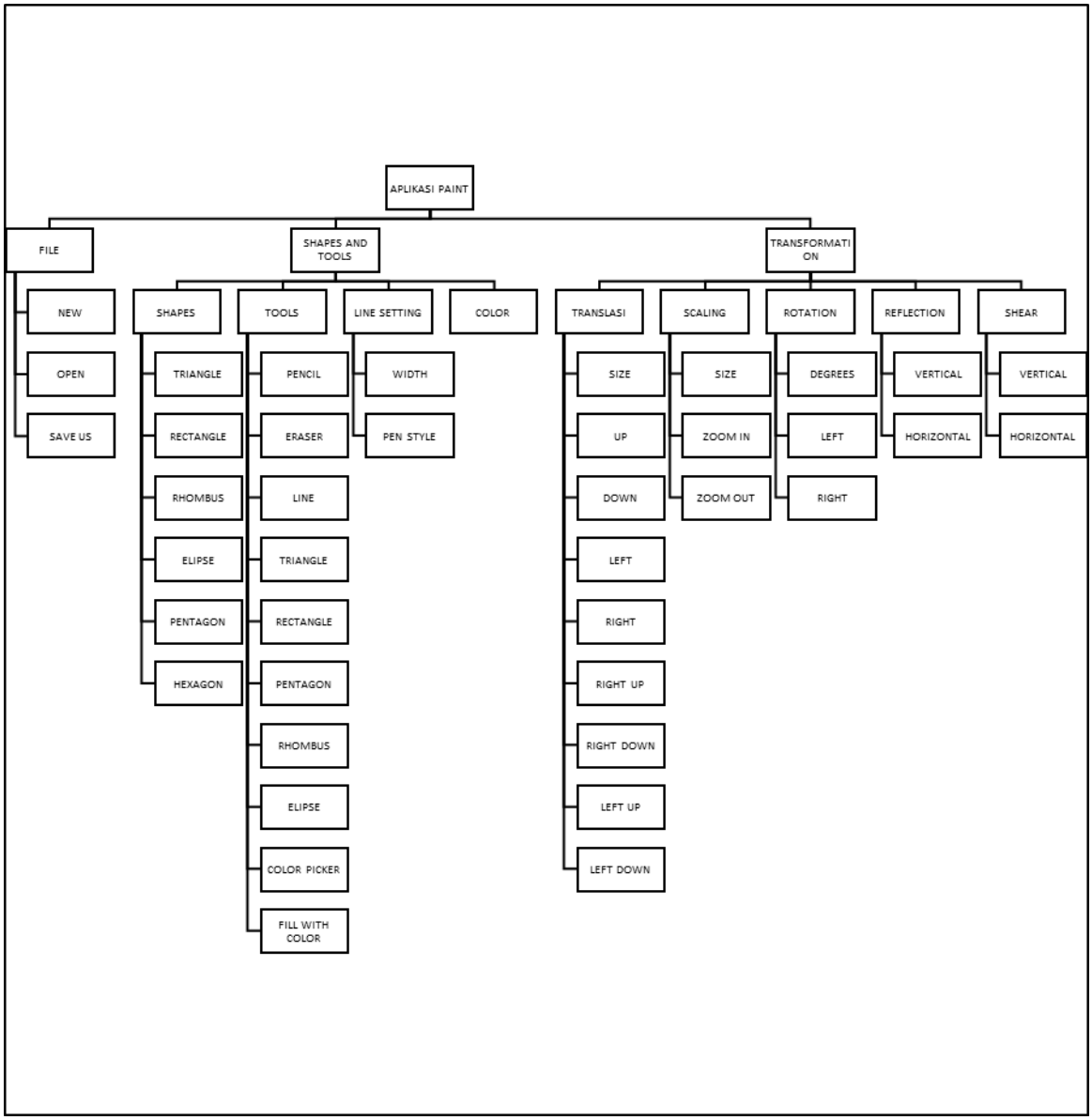
Matrik transformasi adalah matrik yang membuat sebuah obyek mengalami perubahan baik berupa perubahan posisi, maupun perubahan ukuran. Matrik transformasi 2D dinyatakan dalamukuran 3x3, dimana kolom ke-3 digunakan untuk menyediakan tempat untuk proses translasi.

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

BAB II

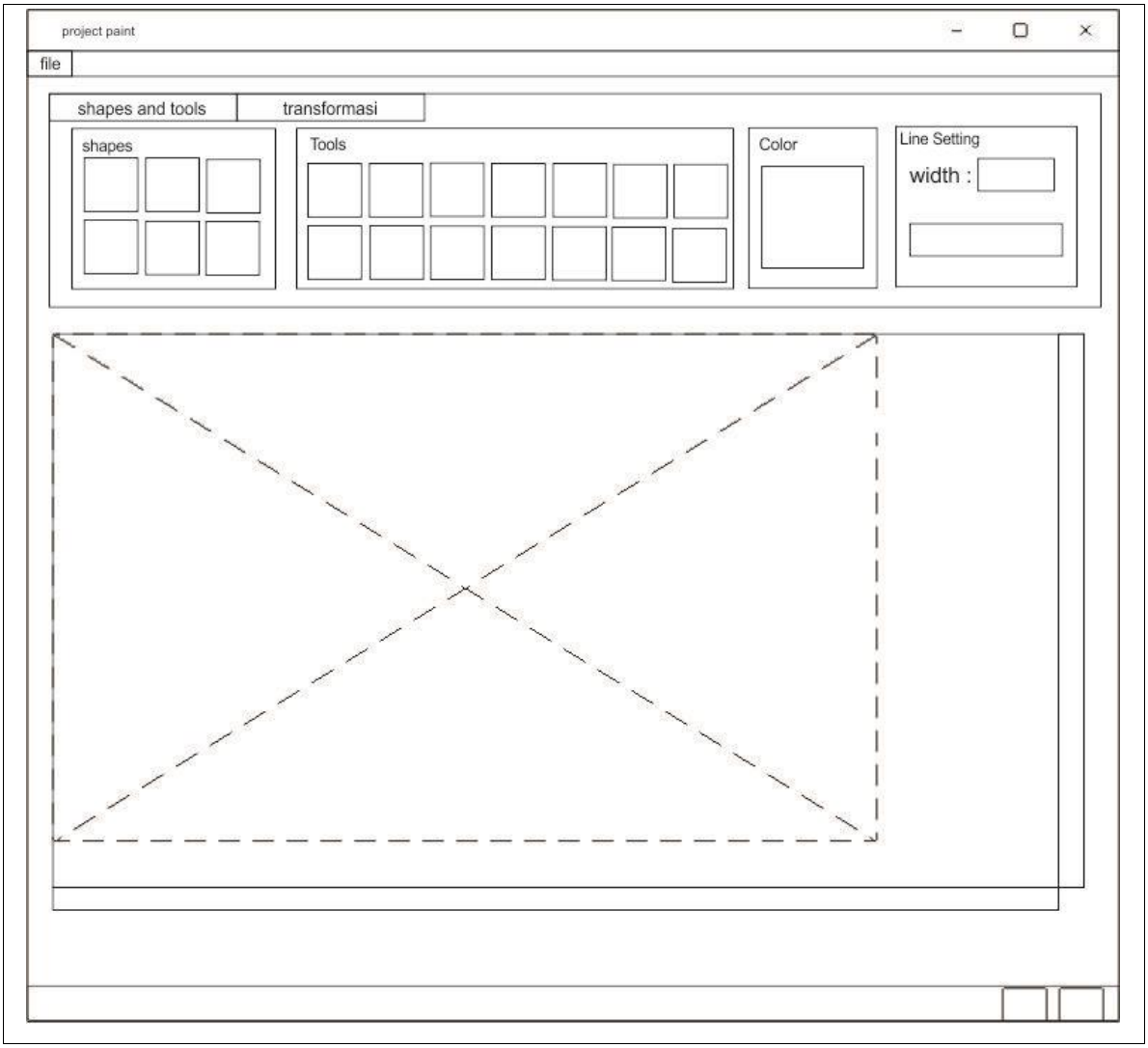
PERANCANGAN APLIKASI

2.1. Perancangan Menu

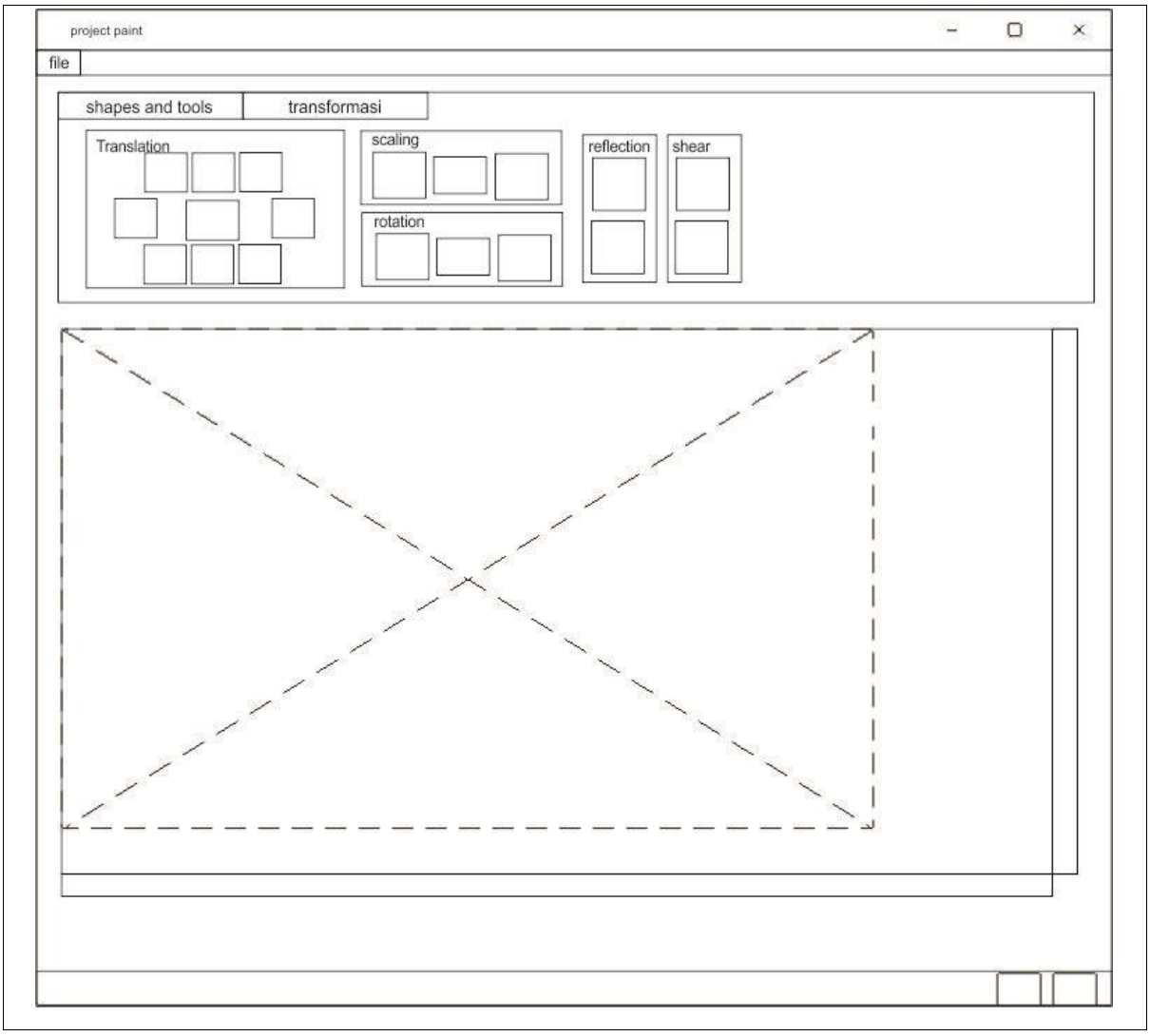


Gambar 2.1 Perancangan Menu Aplikasi Paint

2.2. Perancangan Antar Muka Pengguna



Gambar 2.2.1 Perancangan Antar Muka Pengguna



Gambar 2.2.2 Perancangan Antar Muka Pengguna

Lembar kerja terletak pada bagian tengah. Sedangkan sisi atas merupakan menu-menu yang dapat digunakan untuk menggambar. Bagian bawah terdapat keterangan koordinat yang ditunjuk oleh krusor.

BAB III

PERANCANGAN APLIKASI

3.1 Perangkat Keras Yang Digunakan

Penyelesaian program ini dibuat dengan didukung oleh komponen perangkat keras dengan spesifikasi sebagai berikut :

1. Komputer Pesonal dengan *processor* min 2.00 GHz atau yang lebih tinggi.
2. RAM 1 GB atau lebih tinggi.
3. VGA minimal 1GB atau lebih.
4. *Mouse* dan *Keyboard*.

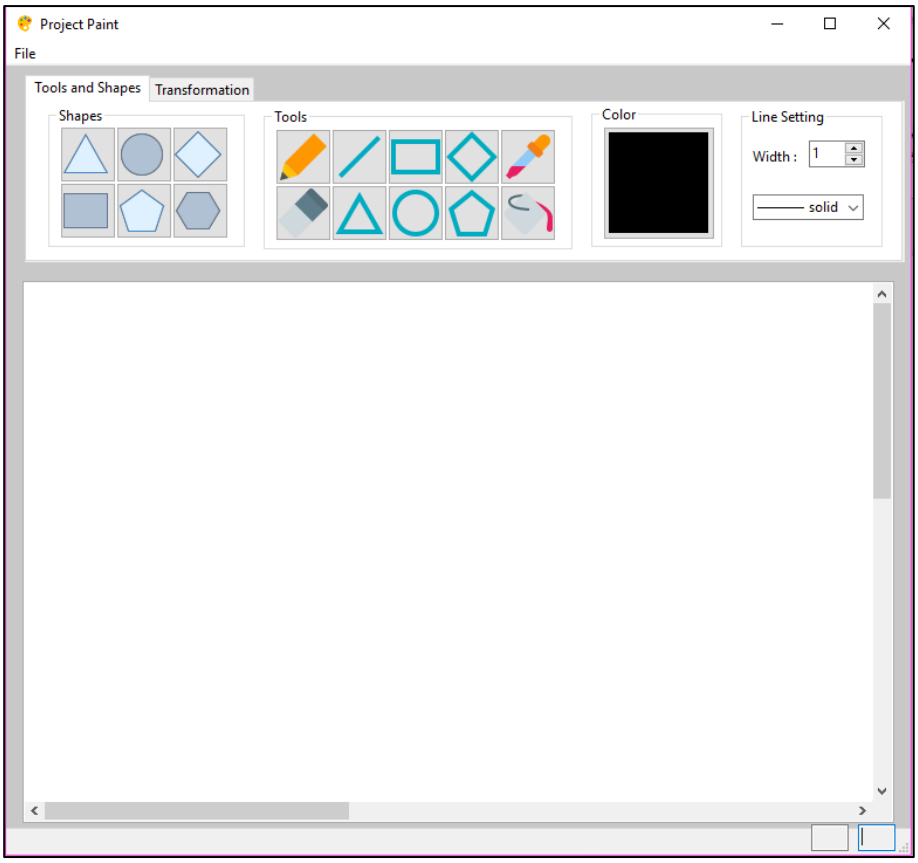
3.2 Perangkat Lunak Yang Digunakan

Untuk merancang aplikasi ini dibutuhkan beberapa perangkat lunak pendukung. Adapun perangkat lunak yang digunakan antara lain :

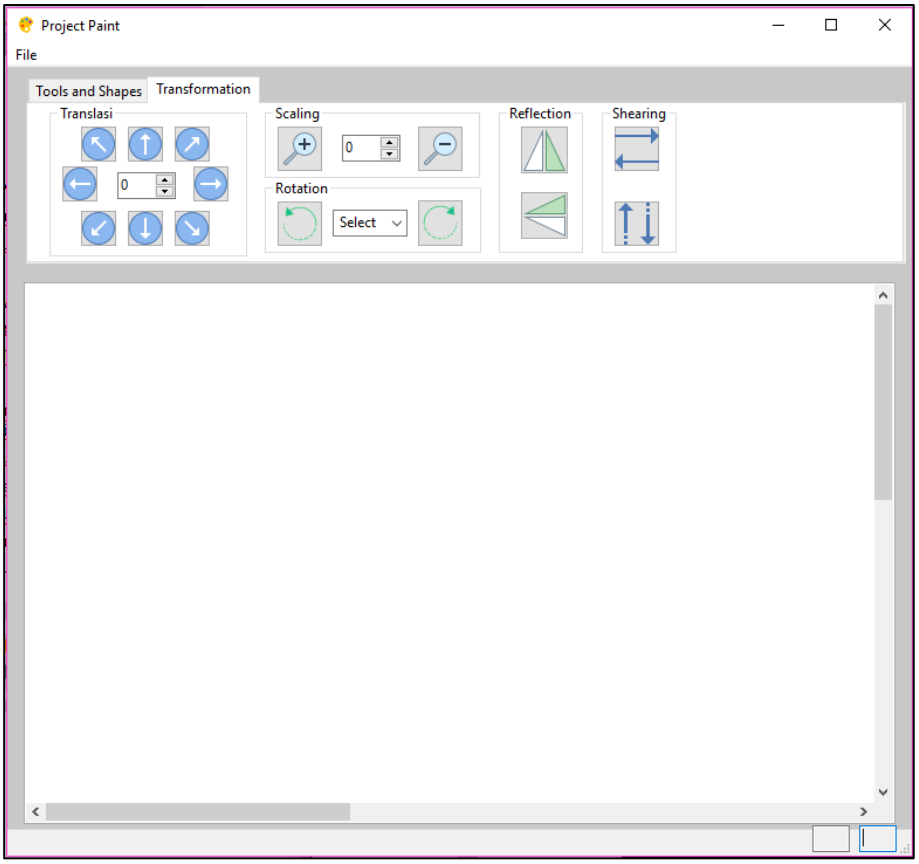
1. Sistem Operasi yang digunakan adalah Windows 7 dan Windows 8.
2. Program pendukung yang digunakan Lazarus dan Paint.

3.3 Tampilan dan Modul Program

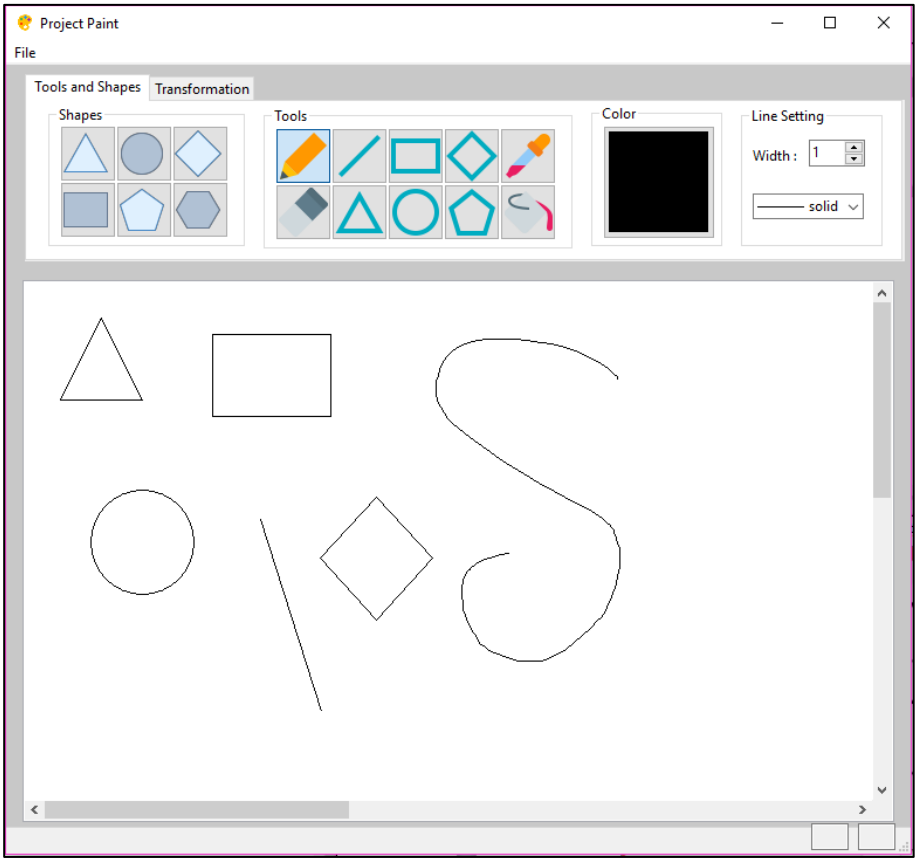
3.3.1 Tampilan



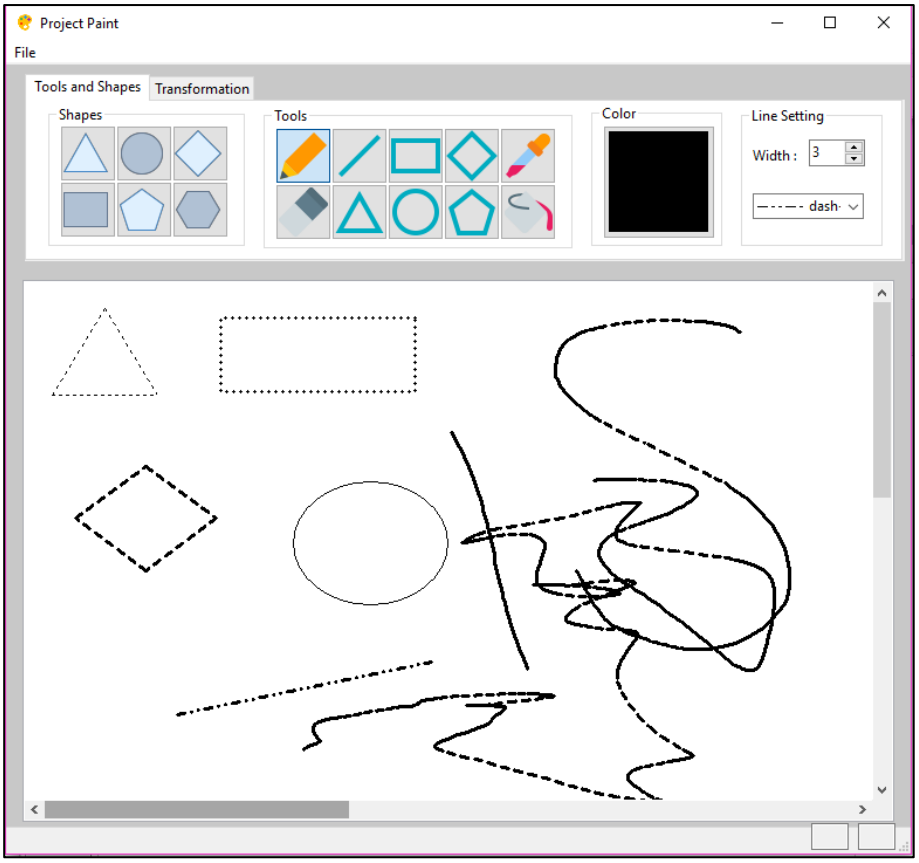
Gambar 3.1 Output Program



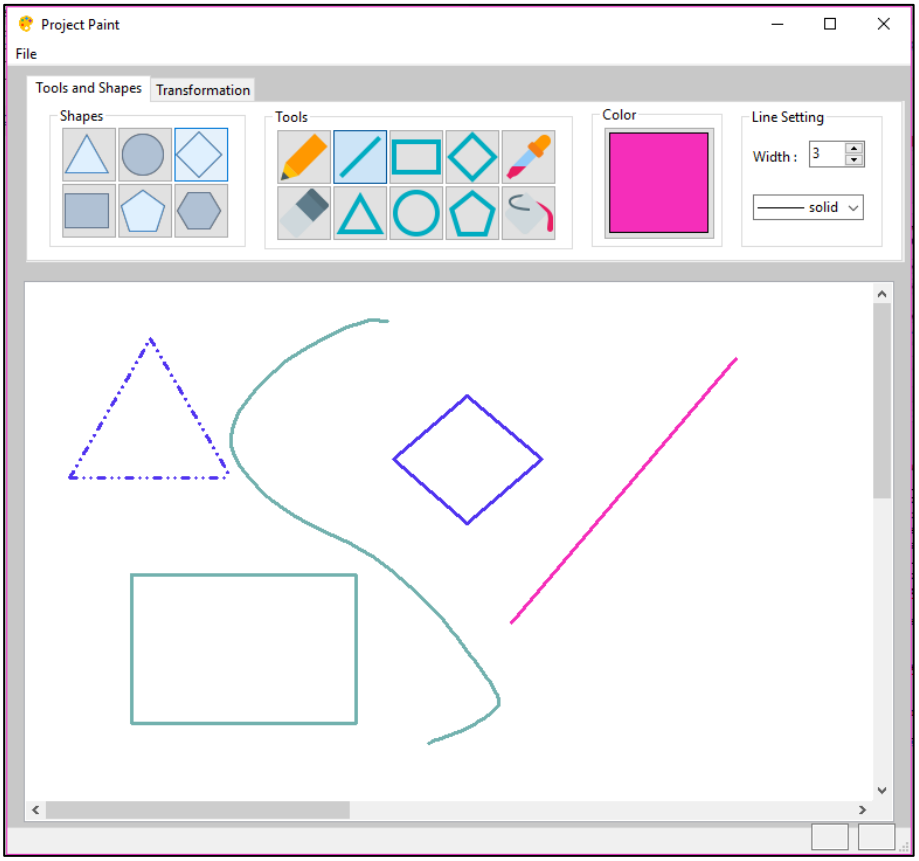
Gambar 3.2 Output Program



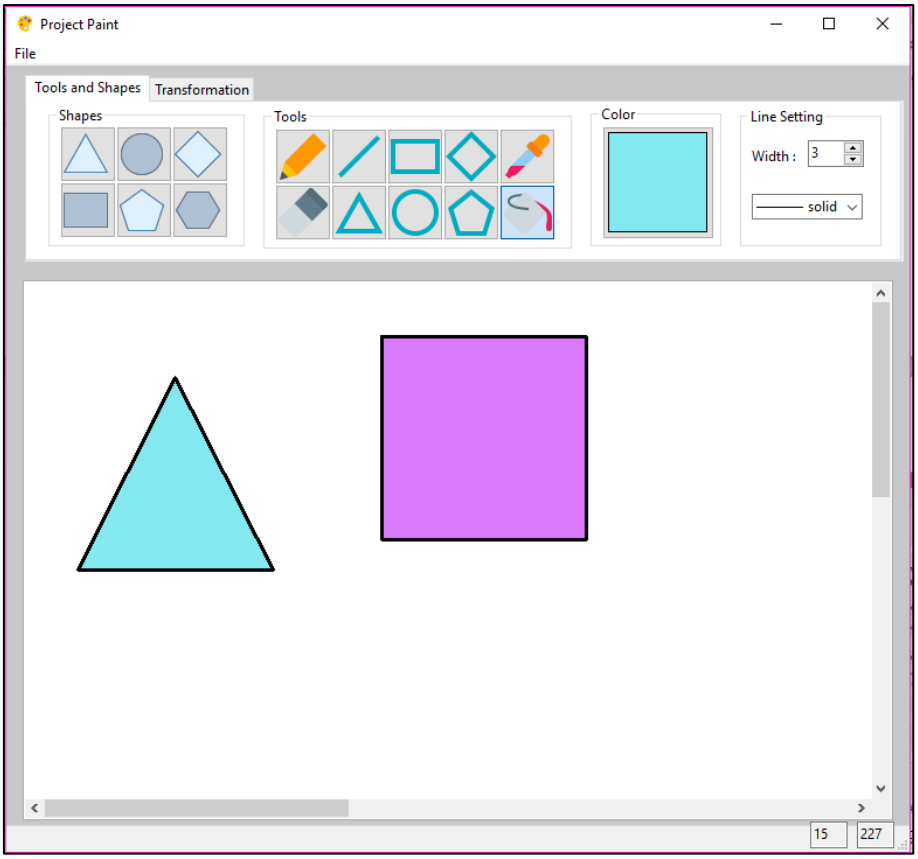
Gambar 3.3 Output Program



Gambar 3.4 Output Program



Gambar 3.5 Output Program



Gambar 3.6 Output Program

3.3.2 Modul Program

```
unit frm1;
{$mode objfpc}{$H+}
interface
uses
  Classes, SysUtils, FileUtil, TAChartCombos, Forms, Controls,
Graphics,
  Dialogs, ExtCtrls, Buttons, Clipbrd, LCLIntf, LCLType, StdCtrls,
Spin, Menus,
  ActnList, ComCtrls, Types;
type
    Elemen=record
      a,b:integer;
     end;
  { TForm1 }
  TForm1 = class(TForm)
    ComboBox1: TComboBox;
    GroupBox1: TGroupBox;
    GroupBox2: TGroupBox;
    GroupBox3: TGroupBox;
    GroupBox4: TGroupBox;
    GroupBox5: TGroupBox;
    GroupBox6: TGroupBox;
    GroupBox7: TGroupBox;
    GroupBox8: TGroupBox;
    GroupBox9: TGroupBox;
    korx: TEdit;
    kory: TEdit;
    Label5: TLabel;
    LineColor: TColorButton;
```

```
MainMenu1: TMainMenu;
MenuItem1: TMenuItem;
MenuItem3: TMenuItem;
MenuItem4: TMenuItem;
MenuItem5: TMenuItem;
MyCanvas: TPaintBox;
OpenDialog1: TOpenDialog;
PageControl1: TPageControl;
PenStyle: TChartComboBox;
SaveDialog1: TSaveDialog;
CanvasScroller: TScrollBox;
SpeedButton1: TBitBtn;
SpeedButton10: TSpeedButton;
SpeedButton11: TSpeedButton;
SpeedButton12: TSpeedButton;
SpeedButton13: TSpeedButton;
SpeedButton14: TSpeedButton;
SpeedButton15: TSpeedButton;
SpeedButton16: TSpeedButton;
SpeedButton17: TSpeedButton;
SpeedButton18: TSpeedButton;
SpeedButton19: TSpeedButton;
SpeedButton2: TBitBtn;
SpeedButton20: TSpeedButton;
SpeedButton21: TSpeedButton;
SpeedButton22: TSpeedButton;
SpeedButton3: TBitBtn;
SpeedButton4: TBitBtn;
Button5: TBitBtn;
SpeedButton6: TBitBtn;
SpeedButton7: TSpeedButton;
SpeedButton8: TSpeedButton;
SpeedButton9: TSpeedButton;
SpinEdit1: TSpinEdit;
SpinEdit2: TSpinEdit;
SpinEdit3: TSpinEdit;
StaticText1: TStaticText;
StatusBar1: TStatusBar;
TabSheet1: TTabSheet;
TabSheet2: TTabSheet;
TabSheet3: TTabSheet;
ToolColorDropper: TSpeedButton;
ToolDiamond: TSpeedButton;
ToolEraser: TSpeedButton;
ToolFill: TSpeedButton;
ToolLine: TSpeedButton;
ToolOval: TSpeedButton;
ToolPencil: TSpeedButton;
ToolPoligon: TSpeedButton;
ToolRect: TSpeedButton;
ToolTriangle: TSpeedButton;
procedure Button5Click(Sender: TObject);
procedure FormActivate(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure Label1Click(Sender: TObject);
procedure LineColorColorChanged(Sender: TObject);
procedure MenuItem3Click(Sender: TObject);
procedure MenuItem4Click(Sender: TObject);
```

Listing Program 3.1 Lanjutan Aplikasi Grafika Komputer

```
procedure MenuItem5Click(Sender: TObject);
    procedure MyCanvasMouseDown(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure MyCanvasMouseLeave(Sender: TObject);
    procedure MyCanvasMouseMove(Sender: TObject; Shift: TShiftState; X,
      Y: Integer);
    procedure MyCanvasMouseUp(Sender: TObject; Button: TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure MyCanvasPaint(Sender: TObject);
    procedure PageControl1Change(Sender: TObject);
    procedure SpeedButton10Click(Sender: TObject);
    procedure SpeedButton11Click(Sender: TObject);
    procedure SpeedButton12Click(Sender: TObject);
    procedure SpeedButton13Click(Sender: TObject);
    procedure SpeedButton14Click(Sender: TObject);
    procedure SpeedButton15Click(Sender: TObject);
    procedure SpeedButton16Click(Sender: TObject);
    procedure SpeedButton17Click(Sender: TObject);
    procedure SpeedButton18Click(Sender: TObject);
    procedure SpeedButton19Click(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure SpeedButton20Click(Sender: TObject);
    procedure SpeedButton21Click(Sender: TObject);
    procedure SpeedButton22Click(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure SpeedButton4Click(Sender: TObject);
    procedure SpeedButton6Click(Sender: TObject);
    procedure SpeedButton7Click(Sender: TObject);
    procedure SpeedButton8Click(Sender: TObject);
    procedure SpeedButton9Click(Sender: TObject);
    procedure TabSheet3ContextPopup(Sender: TObject; MousePos: TPoint;
      var Handled: Boolean);
    procedure ToolEraserClick(Sender: TObject);
    procedure SpinEdit1Change(Sender: TObject);
    procedure ToolPencilClick(Sender: TObject);
    procedure ToolRectClick(Sender: TObject);
    procedure TitikTengah;
  Private
{ private declarations }
  public
    { public declarations }
  end;
var
  Form1: TForm1;
  objek:array[1..100]of elemen;
  TempObjek:array[1..14]of Elemen;
  l:integer;
  paintbmp: TBitmap;
  bidang: string;
  filename:string;
  MouseIsDown: Boolean;
  minimize, maximize: Boolean;
  PrevX, PrevY, X, Y, tx, ty: Integer;
  jmlsdt,m,n,r:Integer;
implementation
{$R *.lfm}
{ TForm1 }
procedure TForm1.FormCreate(Sender: TObject);
```

Listing Program 3.2 Lanjutan Aplikasi Grafika Komputer

```
begin
  // We create a new file/canvas/document when
  // it starts up
  MenuItem3Click(Sender);
end;
procedure TForm1.Button5Click(Sender: TObject);
begin
  bidang := 'Persegi';
    Objek[1].a:=150;    Objek[1].b:=150;
    Objek[2].a:=250;    Objek[2].b:=150;
    Objek[3].a:=250;    Objek[3].b:=250;
    Objek[4].a:=150;    Objek[4].b:=250;
  FormShow(sender);
end;
procedure TForm1.FormActivate(Sender: TObject);
begin
end;
procedure TForm1.FormShow(Sender: TObject);
begin
  MyCanvasPaint(Sender);
  if bidang = 'Persegi'  then
  begin
    MyCanvas.Canvas.MoveTo(Objek[4].a,Objek[4].b);
    for l:=1 to 4 do
    MyCanvas.Canvas.LineTo(Objek[l].a,Objek[l].b);
  end else
  if bidang = 'Segitiga' then
  begin
     MyCanvas.Canvas.MoveTo(Objek[3].a,Objek[3].b);
     for l:=1 to 3 do
     MyCanvas.Canvas.LineTo(Objek[l].a,Objek[l].b);
  end else
  if bidang = 'Oval'  then
  begin
     MyCanvas.Canvas.MoveTo(Objek[1].a,Objek[1].b);
     MyCanvas.Canvas.Ellipse(Objek[1].a,Objek[l].b,
Objek[2].a,Objek[2].b);
  end else
  if bidang = 'Rhombus' then
  begin
     MyCanvas.Canvas.MoveTo(Objek[4].a,Objek[4].b);
     for l:=1 to 4 do
     MyCanvas.Canvas.LineTo(Objek[l].a,Objek[l].b);
  end  else
  if bidang = 'Pentagon' then
  begin
     MyCanvas.Canvas.MoveTo(Objek[5].a,Objek[5].b);
     for l:=1 to 5 do
     MyCanvas.Canvas.LineTo(Objek[l].a,Objek[l].b);
  end else
  if bidang = 'hexagon' then
  begin
     MyCanvas.Canvas.MoveTo(Objek[1].a,Objek[1].b);
     for l:=1 to 7 do
```

Listing Program 3.3 Lanjutan Aplikasi Grafika Komputer

```
MyCanvas.Canvas.LineTo(Objek[l].a,Objek[l].b);
  end;
end;
procedure TForm1.Label1Click(Sender: TObject);
begin
end;
procedure TForm1.LineColorColorChanged(Sender: TObject);
begin
  paintbmp.Canvas.Pen.Color:=LineColor.ButtonColor;
  MyCanvas.Canvas.Pen.Color:=LineColor.ButtonColor;
end;
procedure TForm1.MenuItem3Click(Sender: TObject);
begin
    // if our bitmap is already Create-ed (TBitmap.Create)
    // then start fresh
    if paintbmp <> nil then
      paintbmp.Destroy;
    paintbmp := TBitmap.Create;
    paintbmp.SetSize(Screen.Width, Screen.Height);
    paintbmp.Canvas.FillRect(0,0,paintbmp.Width,paintbmp.Height);
    paintbmp.Canvas.Brush.Style:=bsClear;
    MyCanvas.Canvas.Brush.Style:=bsClear;
    paintbmp.Canvas.Pen.Color:=LineColor.ButtonColor;
    paintbmp.Canvas.Pen.Width:=SpinEdit1.Value;
    MyCanvasPaint(Sender);
end;
procedure TForm1.MenuItem4Click(Sender: TObject);
begin
      if OpenDialog1.Execute then
  begin
    OpenDialog1.Filter:= 'Bitmap(*.bmp)|*.BMP';
    FileName:= OpenDialog1.FileName;
    paintbmp.LoadFromFile(FileName);
    MyCanvasPaint(Sender);
  end;
end;
procedure TForm1.MenuItem5Click(Sender: TObject);
begin
    SaveDialog1.Execute;
  if SaveDialog1.Files.Count > 0 then begin
    // if the user enters a file name without a .bmp
    // extension, we will add it
    if RightStr(SaveDialog1.FileName, 4) <> '.bmp' then
      SaveDialog1.FileName:=SaveDialog1.FileName+'.bmp';
    paintbmp.SaveToFile(SaveDialog1.FileName);
  end;
end;
procedure TForm1.MyCanvasMouseDown(Sender: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
 MouseIsDown := True;
  PrevX := X;
  PrevY := Y;
```

Listing Program 3.4 Lanjutan Aplikasi Grafika Komputer

```
procedure TForm1.MyCanvasMouseLeave(Sender: TObject);
begin
  korx.Text:=' ';
  kory.Text:=' ';
end;
procedure TForm1.MyCanvasMouseMove(Sender: TObject; Shift: TShiftState;
X,
  Y: Integer);
begin
  if MouseIsDown = true then begin
    // Pencil Tool
    if ToolPencil.Down = true then begin
      paintbmp.Canvas.Line(PrevX, PrevY, X, Y);
      paintbmp.Canvas.Pen.Style:=PenStyle.PenStyle;
      MyCanvas.Canvas.Line(PrevX, PrevY, X, Y);
      PrevX:=X;
      PrevY:=Y;
    // Eraser Tool
    end else if ToolEraser.Down = true then begin
       paintbmp.Canvas.Pen.Color := clWhite;
       paintbmp.Canvas.Line(PrevX, PrevY, X, Y);
       MyCanvas.Canvas.Line(PrevX, PrevY, X, Y);
       PrevX:=X;
       PrevY:=Y;
    // Line Tool
    end else if ToolLine.Down = true then begin
      // we are clearing previous preview drawing//
      MyCanvasPaint(Sender);
      // we draw a preview line //
      paintbmp.Canvas.Pen.Style:=PenStyle.PenStyle;
      MyCanvas.Canvas.Line(PrevX, PrevY, X, Y);
    // Rectangle Tool //
    end else if ToolRect.Down = true then begin
      MyCanvasPaint(Sender);
      paintbmp.Canvas.Pen.Style:=PenStyle.PenStyle;
      MyCanvas.Canvas.Rectangle(PrevX, PrevY, X, Y);
    //Oval Tool //
    end else if ToolOval.Down = true then begin
      MyCanvasPaint(Sender);
      paintbmp.Canvas.Pen.Style:=PenStyle.PenStyle;
      MyCanvas.Canvas.Ellipse(PrevX, PrevY, X, Y);
    // Diamond Tool //
 end else if ToolDiamond.Down = true then begin
      MyCanvasPaint(Sender);
      paintbmp.Canvas.Pen.Style:=PenStyle.PenStyle;
      MyCanvas.Canvas.Line(PrevX+((X-PrevX) div 2), PrevY,X,PrevY+((Y-
PrevY) div 2));
      MyCanvas.Canvas.Line(X, PrevY+((Y-PrevY) div 2), PrevX+((X-PrevX)
div 2),Y);
      MyCanvas.Canvas.Line(PrevX+((X-PrevX) div 2),Y,PrevX,PrevY+((Y-
PrevY) div 2));
      MyCanvas.Canvas.Line(PrevX,PrevY+((Y-PrevY) div 2),PrevX+((X-
PrevX) div 2), PrevY);
    // Triangle Tool //
```

Listing Program 3.5 Lanjutan Aplikasi Grafika Komputer

```
end else if ToolTriangle.Down = true then begin
      MyCanvasPaint(Sender);
      paintbmp.Canvas.Pen.Style:=PenStyle.PenStyle;
      MyCanvas.Canvas.Line(PrevX,Y,PrevX+((X-PrevX) div 2), PrevY);
      MyCanvas.Canvas.Line(PrevX+((X-PrevX) div 2),PrevY,X,Y);
      MyCanvas.Canvas.Line(PrevX,Y,X,Y);
    end;
  end;
      ////Koordinat////
      korx.Text:=IntToStr(X);
      kory.Text:=IntToStr(Y);

end;
procedure TForm1.MyCanvasMouseUp(Sender: TObject; Button: TMouseButton;
  Shift: TShiftState; X, Y: Integer);
var
  TempColor: TColor;
begin
  if MouseIsDown then begin
    // Line tool
    if ToolLine.Down = true then begin
      paintbmp.Canvas.Line(PrevX, PrevY, X, Y);
    // Rect tool
    end else if ToolRect.Down = true then begin
      paintbmp.Canvas.Rectangle(PrevX, PrevY, X, Y);
    // Oval tool
    end else if ToolOval.Down = true then begin
      paintbmp.Canvas.Ellipse(PrevX, PrevY, X, Y);
    // Diamond tool
end else if ToolDiamond.Down = true then begin
      paintbmp.Canvas.Line(PrevX+((X-PrevX) div 2), PrevY,X,PrevY+((Y-
PrevY) div 2));
      paintbmp.Canvas.Line(X, PrevY+((Y-PrevY) div 2), PrevX+((X-PrevX)
div 2),Y);
      paintbmp.Canvas.Line(PrevX+((X-PrevX) div 2),Y,PrevX,PrevY+((Y-
PrevY) div 2));
      paintbmp.Canvas.Line(PrevX,PrevY+((Y-PrevY) div 2),PrevX+((X-
PrevX) div 2), PrevY);
    // Triangle tool
    end else if ToolTriangle.Down = true then begin
      paintbmp.Canvas.Line(PrevX,Y,PrevX+((X-PrevX) div 2), PrevY);
      paintbmp.Canvas.Line(PrevX+((X-PrevX) div 2),PrevY,X,Y);
      paintbmp.Canvas.Line(PrevX,Y,X,Y);
    // Color Dropper Tool
    end else if ToolColorDropper.Down = true then begin
      LineColor.ButtonColor:=MyCanvas.Canvas.Pixels[X,Y];
    // (Flood) Fill Tool
    end else if ToolFill.Down = true then begin
      TempColor := paintbmp.Canvas.Pixels[X, Y];
      paintbmp.Canvas.Brush.Style := bsSolid;
      paintbmp.Canvas.Brush.Color := LineColor.ButtonColor;
      paintbmp.Canvas.FloodFill(X, Y, TempColor, fsSurface);
      paintbmp.Canvas.Brush.Style := bsClear;
      MyCanvasPaint(Sender);
```

Listing Program 3.6 Lanjutan Aplikasi Grafika Komputer

```
procedure TForm1.MyCanvasPaint(Sender: TObject);
begin
  paintbmp.Canvas.Pen.Color:=LineColor.ButtonColor;
  if MyCanvas.Width<>paintbmp.Width then begin
    MyCanvas.Width:=paintbmp.Width;
    // the resize will run this function again
    // so we skip the rest of the code
    Exit;
  end;
  if MyCanvas.Height<>paintbmp.Height then begin
    MyCanvas.Height:=paintbmp.Height;
    // the resize will run this function again
    // so we skip the rest of the code
    Exit;
  end;
  MyCanvas.Canvas.Draw(0,0,paintbmp);
  end;
procedure TForm1.PageControl1Change(Sender: TObject);
begin
end;
procedure TForm1.SpeedButton10Click(Sender: TObject);
begin
    TitikTengah;
    for l:=1 to 7 do
    begin
      Objek[l].a:=Objek[l].a-SpinEdit2.Value;
      Objek[l].b:=Objek[l].b+SpinEdit2.Value;
    end;
    m:=m-SpinEdit2.Value;
    n:=n-SpinEdit2.Value;
    FormShow(Sender);
end;
procedure TForm1.SpeedButton11Click(Sender: TObject);
begin
    TitikTengah;
  for l:= 1 to 7 do
     Objek[l].b:= Objek[l].b+SpinEdit2.value;
  FormShow(Sender);
end;
procedure TForm1.SpeedButton12Click(Sender: TObject);
begin
    TitikTengah;
    for l:=1 to 7 do
    begin
      Objek[l].a:=Objek[l].a+SpinEdit2.Value;
      Objek[l].b:=Objek[l].b-SpinEdit2.Value;
    end;
    m:=m-SpinEdit2.Value;
    n:=n-SpinEdit2.Value;
    FormShow(Sender);
end;
procedure TForm1.SpeedButton13Click(Sender: TObject);
begin
    TitikTengah;
```

Listing Program 3.7 Lanjutan Aplikasi Grafika Komputer

```
for l:=1 to 7 do
    begin
      Objek[l].a:=Objek[l].a-SpinEdit2.Value;
    end;
    FormShow(Sender);
end;
procedure TForm1.SpeedButton14Click(Sender: TObject);
begin
    TitikTengah;
    for l:=1 to 7 do
    begin
      Objek[l].a:=Objek[l].a+SpinEdit2.Value;
    end;
    FormShow(Sender);
    ;
end;
procedure TForm1.SpeedButton15Click(Sender: TObject);
begin
   TitikTengah;
   for l:= 1 to 7 do
   begin
        Objek[l].a:=Objek[l].a-m;
        Objek[l].b:=Objek[l].b-n;
        TempObjek[l].a:=Objek[l].a * SpinEdit3.Value;
        TempObjek[l].b:=Objek[l].b * SpinEdit3.Value;
        Objek[l]:=TempObjek[l];
        Objek[l].a:=Objek[l].a+m;
        Objek[l].b:=Objek[l].b+n;
   end;
  FormShow(Sender);
end;
procedure TForm1.SpeedButton16Click(Sender: TObject);
begin
   TitikTengah;
   for l:= 1 to 7 do
   begin
        Objek[l].a:=Objek[l].a-m;
        Objek[l].b:=Objek[l].b-n;
        TempObjek[l].a:=Objek[l].a div SpinEdit3.Value;
        TempObjek[l].b:=Objek[l].b div SpinEdit3.Value;
        Objek[l]:=TempObjek[l];
        Objek[l].a:=Objek[l].a+m;
        Objek[l].b:=Objek[l].b+n;
   end;
  FormShow(Sender);
end;
procedure TForm1.SpeedButton17Click(Sender: TObject);
var
  sdt : real;
begin
  if ComboBox1.ItemIndex=0 then sdt:= 15 mod 360 / -180*PI;
  if ComboBox1.ItemIndex=1 then sdt:= 45 mod 360 / -180*PI;
  if ComboBox1.ItemIndex=2 then sdt:= 90 mod 360 / -180*PI;
  if ComboBox1.ItemIndex=3 then sdt:= 180 mod 360 / -180*PI;
```

Listing Program 3.8 Lanjutan Aplikasi Grafika Komputer

```
  TitikTengah;
  for l:=1 to 7 do
  begin
  Objek[l].a  := Objek[l].a - m;
  Objek[l].b  := Objek[l].b - n;
  TempObjek[l].a := round(Objek[l].a * cos(sdt) - Objek[l].b *
sin(sdt));
  TempObjek[l].b := round(Objek[l].a * sin(sdt) + Objek[l].b *
cos(sdt));
  Objek[l]   := TempObjek[l];
  Objek[l].a := Objek[l].a + m;
  Objek[l].b := Objek[l].b + n;
  end;
  FormShow(Sender);
end;
procedure TForm1.SpeedButton18Click(Sender: TObject);
var
  sdt : real;
begin
  if ComboBox1.ItemIndex=0 then sdt:= 15 mod 360 / 180*PI;
  if ComboBox1.ItemIndex=1 then sdt:= 45 mod 360 / 180*PI;
  if ComboBox1.ItemIndex=2 then sdt:= 90 mod 360 / 180*PI;
  if ComboBox1.ItemIndex=3 then sdt:= 180 mod 360 / 180*PI;
  TitikTengah;
  for l:=1 to 7 do
  begin
  Objek[l].a  := Objek[l].a - m;
  Objek[l].b  := Objek[l].b - n;
  TempObjek[l].a := round(Objek[l].a * cos(sdt) - Objek[l].b *
sin(sdt));
  TempObjek[l].b := round(Objek[l].a * sin(sdt) + Objek[l].b *
cos(sdt));
  Objek[l]   := TempObjek[l];
  Objek[l].a := Objek[l].a + m;
  Objek[l].b := Objek[l].b + n;
  end;
  FormShow(Sender);
end;
procedure TForm1.SpeedButton19Click(Sender: TObject);
begin
  for l:=1 to 7 do
    begin
      Objek[l].a:=Objek[l].a;
      Objek[l].b:=-Objek[l].b+500;
    end;
    m:=m-500;
    n:=n-500;
    FormShow(Sender);
end;
procedure TForm1.SpeedButton1Click(Sender: TObject);
begin
  bidang :='Segitiga';
  Objek[1].a:=30;          Objek[1].b:=100;
  Objek[2].a:=65;          Objek[2].b:=30;
```

Listing Program 3.9 Lanjutan Aplikasi Grafika Komputer

```
  Objek[3].a:=100;         Objek[3].b:=100;
end;
procedure TForm1.SpeedButton20Click(Sender: TObject);
begin
  TitikTengah;
    for l:=1 to 7 do
    begin
      Objek[l].a:=-Objek[l].a+500;
      Objek[l].b:=Objek[l].b;
    end;
    m:=-m+500;
  FormShow(Sender);
end;
procedure TForm1.SpeedButton21Click(Sender: TObject);
begin
  TitikTengah;
  if (bidang='Persegi') or (bidang='Rhombus') then
  begin
      Objek[1].a:=Objek[1].a+15;
      Objek[2].a:=Objek[2].a+15;
      Objek[3].a:=Objek[3].a-15;
      Objek[4].a:=Objek[4].a-15;
    FormShow(Sender);
  end else
  if bidang='Segitiga' then
  begin
      Objek[1].a:=Objek[1].a+15;
      Objek[2].a:=Objek[2].a-15;
      Objek[3].a:=Objek[3].a-15;
    FormShow(Sender);
  end else
  if bidang='hexagon' then
  begin
      Objek[1].a:=Objek[1].a+15;
      Objek[2].a:=Objek[2].a+15;
      Objek[4].a:=Objek[4].a-15;
      Objek[5].a:=Objek[5].a-15;
    FormShow(Sender);
  end else
if bidang='Oval' then
  begin
      Objek[1].a:=Objek[1].a+15;
      Objek[2].a:=Objek[2].a-15;
    FormShow(Sender);
  end else
  if bidang='Pentagon' then
  begin
      Objek[1].a:=Objek[1].a+15;
      Objek[2].a:=Objek[2].a+15;
      Objek[3].a:=Objek[3].a-15;
      Objek[4].a:=Objek[4].a-15;
      Objek[5].a:=Objek[5].a+15;
    FormShow(Sender);
  end;
```

Listing Program 3.10 Lanjutan Aplikasi Grafika Komputer

```
end;
procedure TForm1.SpeedButton22Click(Sender: TObject);
begin
  TitikTengah;
  if (bidang='Persegi') or (bidang='Rhombus') then
  begin
      Objek[1].b:=Objek[1].b+15;
      Objek[2].b:=Objek[2].b-15;
      Objek[3].b:=Objek[3].b-15;
      Objek[4].b:=Objek[4].b+15;
    FormShow(Sender);
  end else
  if bidang='Persegipanjang' then
  begin
      Objek[1].b:=Objek[1].b+15;
      Objek[2].b:=Objek[2].b-15;
      Objek[3].b:=Objek[3].b-15;
      Objek[4].b:=Objek[4].b+15;
    FormShow(Sender);
  end else
  if bidang='Segitiga' then
  begin
      Objek[1].b:=Objek[1].b+15;
      Objek[2].b:=Objek[2].b-15;
      Objek[3].b:=Objek[3].b+15;
    FormShow(Sender);
  end else
  if bidang='hexagon' then
  begin
      Objek[1].b:=Objek[1].b+15;
      Objek[2].b:=Objek[2].b-15;
      Objek[3].b:=Objek[3].b-15;
      Objek[4].b:=Objek[4].b+15;
      Objek[5].b:=Objek[5].b+15;
      Objek[6].b:=Objek[6].b+15;
    FormShow(Sender);
  end else
  if bidang='Lingkaran' then
  begin
      Objek[1].b:=Objek[1].b+15;
      Objek[2].b:=Objek[2].b-15;
    FormShow(Sender);
  end else
  if bidang='Pentagon' then
  begin
      Objek[2].b:=Objek[2].b-15;
      Objek[3].b:=Objek[3].b-15;
      Objek[4].b:=Objek[4].b+15;
      Objek[5].b:=Objek[5].b+15;
    FormShow(Sender);
    end;
    end;
    procedure TForm1.SpeedButton2Click(Sender: TObject);
    begin
```

Listing Program 3.11 Lanjutan Aplikasi Grafika Komputer

```
    bidang := 'Pentagon';
      Objek[1].a:=200;          Objek[1].b:=200;
      Objek[2].a:=250;          Objek[2].b:=250;
      Objek[3].a:=225;          Objek[3].b:=300;
      Objek[4].a:=175;          Objek[4].b:=300;
      Objek[5].a:=150;          Objek[5].b:=250;
    end;
    procedure TForm1.SpeedButton3Click(Sender: TObject);
    begin
      bidang := 'Rhombus';
      Objek[1].a:=100;          Objek[1].b:=100;
      Objek[2].a:=150;          Objek[2].b:=150;
      Objek[3].a:=100;          Objek[3].b:=200;
      Objek[4].a:=50;           Objek[4].b:=150;
    end;
    procedure TForm1.SpeedButton4Click(Sender: TObject);
    begin
      bidang := 'Oval';
      Objek[1].a:=300;          Objek[1].b:=300;
      Objek[2].a:=450;          Objek[2].b:=450;
    end;
    procedure TForm1.SpeedButton6Click(Sender: TObject);
    begin
      bidang := 'hexagon';
      Objek[1].a:=150;          Objek[1].b:=50;
      Objek[2].a:=200;          Objek[2].b:=50;
      Objek[3].a:=225;          Objek[3].b:=100;
      Objek[4].a:=200;          Objek[4].b:=150;
      Objek[5].a:=150;          Objek[5].b:=150;
      Objek[6].a:=125;          Objek[6].b:=100;
      Objek[7].a:=150;          Objek[7].b:=50;
    end;
    procedure TForm1.SpeedButton7Click(Sender: TObject);
    begin
      TitikTengah;
        for l:=1 to 7 do
        begin
          Objek[l].a:=Objek[l].a-SpinEdit2.Value;
          Objek[l].b:=Objek[l].b-SpinEdit2.Value;
        end;
        m:=m-SpinEdit2.Value;
        n:=n-SpinEdit2.Value;
        FormShow(Sender);
    end;
    procedure TForm1.SpeedButton8Click(Sender: TObject);
    begin
      TitikTengah;
      for l:= 1 to 7 do
         Objek[l].b:= Objek[l].b-SpinEdit2.value;
      FormShow(Sender);
end;
procedure TForm1.SpeedButton9Click(Sender: TObject);
begin
    TitikTengah;
```

Listing Program 3.12 Lanjutan Aplikasi Grafika Komputer

```
for l:=1 to 7 do
    begin
      Objek[l].a:=Objek[l].a+SpinEdit2.Value;
      Objek[l].b:=Objek[l].b+SpinEdit2.Value;
    end;
    m:=m-SpinEdit2.Value;
    n:=n-SpinEdit2.Value;
    FormShow(Sender);
end;
procedure TForm1.TabSheet3ContextPopup(Sender: TObject; MousePos:
TPoint;
  var Handled: Boolean);
begin
end;
procedure TForm1.SpinEdit1Change(Sender: TObject);
begin
  paintbmp.Canvas.Pen.Width:=SpinEdit1.Value;
  MyCanvas.Canvas.Pen.Width:=SpinEdit1.Value;
end;
procedure TForm1.ToolPencilClick(Sender: TObject);
begin
end;
procedure TForm1.ToolEraserClick(Sender: TObject);
begin
end;
procedure TForm1.ToolRectClick(Sender: TObject);
begin
end;
procedure Tform1.TitikTengah;
var
  tx,ty : integer;
begin
  tx:=0;    ty:=0;
  if bidang='Persegi' then
  begin
  for l:=1 to 4 do
  begin
    tx:=tx+Objek[l].a;
  end;
  for l:=1 to 4 do
  begin
    ty:=Objek[l].b+ty;
  end;
  m:=round(tx/4);
  n:=round(ty/4);
  end else
if bidang='Rhombus' then
  begin
  for l:=1 to 4 do
  begin
    tx:=tx+Objek[l].a;
    end;
      for l:=1 to 4 do
      begin
```

Listing Program 3.13 Lanjutan Aplikasi Grafika Komputer

```
ty:=Objek[l].b+ty;
  end;
  m:=round(tx/4);
  n:=round(ty/4);
  end else
  if bidang='Segitiga' then
  begin
  for l:=1 to 3 do
  begin
    tx:=tx+Objek[l].a;
  end;
  for l:=1 to 3 do
  begin
    ty:=Objek[l].b+ty;
  end;
  m:=round(tx/3);
  n:=round(ty/3);
  end else
  if bidang='Oval' then
  begin
  for l:=1 to 2 do
  begin
    tx:=tx+Objek[l].a;
  end;
  for l:=1 to 2 do
  begin
    ty:=Objek[l].b+ty;
  end;
  m:=round(tx/2);
  n:=round(ty/2);
  r:=Objek[l].a-m;
  end else
  if bidang='Pentagon' then
  begin
  for l:=1 to 5 do
  begin
    tx:=tx+Objek[l].a;
  end;
  for l:=1 to 5 do
  begin
    ty:=Objek[l].b+ty;
  end;
  m:=round(tx/5);
  n:=round(ty/5);
  end else
  if bidang='hexagon' then
  begin
  for l:=1 to 6 do
  begin
    tx:=tx+Objek[l].a;
    end;
      for l:=1 to 6 do
      begin
        ty:=Objek[l].b+ty;
```

Listing Program 3.14 Lanjutan Aplikasi Grafika Komputer

```
    end;
      m:=round(tx/6);
      n:=round(ty/6);
      end;
      end;
    end.
```

Listing Program 3.15 Lanjutan Aplikasi Grafika Komputer

BAB IV

KESIMPULAN DAN SARAN

1.1 Kesimpulan

Berdasarkan hasil analisis dan perancangan, telah dihasilkan suatu Aplikasi Grafika Komputer yang dapat membantu dan mempermudah *user* dalam membuat bangun dalam bentuk dua Dimensi. Aplikasi ini pun dapat membuat bangun tersebut bertipekan garis lurus dengan tebal dan jenis garis yang di inginkan. Untuk sistem perwarnaan bangun dua dimensi dapat di beri warna pada saat membentuk garis (line color) dan dapat di warnai bangun ruangnya (fill color). Pada aplikasi ini dapat melakukan transformasi diatara lain yaitu translasi, rotasi, perubahan skala dengan nilai yang di inginkan user, *reflection* dan *shear*. Terdapat bermacam tools yang dapat digunakan dengan kendali *mouse* atau krusor seperti pencil , eraser , fill color , color picker dan pembentuk bangun dua dimensi..

1.2 Saran

Aplikasi ini hanya menampilkan fungsi-fungsi seperti yang di sebutkan pada kesimpulan dan pada BAB II. Diharapkan untuk kedepannya dapat memperbaiki bahkan melengkapi fungsi-fungsi yang kurang yang terdapat pada aplikasi grafika komputer ini seperti menambah bangun 2 dimensi lebih banyak bahkan atau menambahkan bangun ruang pada aplikasi ini, memindahkan objek dengan kendali *mouse* atau krusor dan memiliki fitur *select* gambar pada kanvas yang akan di berikan transformasi.

DAFTAR PUSTAKA

_____,https://elib.unikom.ac.id/download.php?id=14270 (diakses 26 April 2019)

_____,http://lazplanet.blogspot.com/2013/06/how-to-create-basic-paint-software.html?m=1 (diakses 16 April 2019)

_____, MODUL PRAKTIKUM Grafika Komputer (diakses 12 April 2019)