PENGGUNAAN *PROCESSING* DALAM KOMPUTER GRAFIK

Meiyanto Eko Sulistyo
Jurusan Informatika
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Sebelas Maret, Surakarta
Email : mekosulistyo@uns.ac.id

Abstract

Computer graphics is the study of how to create images and animations (image sequences) involving computers, both hardware (hardware) and software (the software). In this paper, we will discuss the basics of using the Processing programming language computer graphics. The author conducted basic experiments using Processing on making size (size) in the window display, point (point), line (line), triangle (triangle), quadrilateral (quad), rectangular (rect), and ellipse (ellipse), as well as the shape coordinate system. Of the experiments that have been performed, resulting in a positive y coordinate position starting from zero which is located in the top left corner heading down, and look up the appropriate forms with the dharapkan.

Keywords: *ellipse, computer graphics line, point, processing, quad, rect, size, triangle*

Komputer grafik adalah ilmu yang mempelajari bagaimana membuat gambar dan animasi (urutan gambar) dengan melibatkan komputer, baik perangkat keras (*hardware*) maupun perangkat lunak (*software*). Dalam makalah ini, akan dibahas tentang dasar-dasar penggunaan bahasa pemrograman *Processing* dalam komputer grafik. Penulis melakukan percobaan-percobaan dasar menggunakan Processing tentang membuat ukuran (*size*) pada *display window*, titik (*point*), garis (*line*), segitiga (*triangle*), segiempat (*quad*), persegi panjang (*rect*), dan ellips (*ellipse*), serta bentuk sistem koordinatnya. Dari percobaan-percobaan yang telah dilakukan, menghasilkan posisi koordinat y positif dimulai dari titik nol yang terletak di pojok kiri paling atas menuju ke bawah, serta tampilan bentuk bangun yang sesuai dengan yang dharapkan.
Kata Kunci : *ellipse*, komputer grafik *line*, *point*, *processing*, *quad*, *rect*, *size*, *triangle*

1. PENDAHULUAN

Komputer grafik merupakan salah satu mata kuliah di bidang komputer seperti program studi ilmu komputer, teknik informatika, teknik komputer, dan teknologi informasi. Untuk mempermudah dosen, peneliti, dan mahasiswa mempelajari dan melakukan penelitian di bidang komputer grafik, maka perlu penggunaan suatu perangkat lunak (*software*). Dalam makalah ini, akan dibahas dasar-dasar penggunaan bahasa pemrograman *Processing* dalam komputer grafik.

Penulis akan melakukan percobaan-percobaan yang sangat mendasar bagaimana membuat ukuran (*size*) pada *display window*, titik (*point*), garis (*line*), segitiga (*triangle*), segiempat (*quad*), persegi panjang (*rect*), dan ellips (*ellipse*), serta jika bentuk-bentuk tersebut dilihat dalam grafik sistem koordinat.

2. KOMPUTER GRAFIK

Komputer grafik berkaitan dengan ilmu yang menghasilkan gambar [4][3] dan animasi (atau urutan gambar) [4] dengan menggunakan komputer [4][3]. Hal ini mencakup sistem perangkat keras (*hardware*) dan perangkat lunak (software) yang digunakan untuk membuat gambar. Menghasilkan gambar foto-realistis adalah salah satu yang sangat kompleks dan bidang yang sangat diminati, karena hampir tak terbatas dari aplikasi. Bidang komputer grafik telah berkembang pesat selama 10-20 tahun terakhir, dan banyak sistem perangkat lunak telah dikembangkan untuk menghasilkan berbagai macam komputer grafik. Hal ini dapat mencakup sistem untuk menghasilkan model tiga dimensi dari adegan yang akan diambil, perangkat lunak

render untuk membuat gambar, dan melewati perangkat lunak antarmuka pengguna (*user interface*) dan perangkat keras [4].

Kata "gambar" harus dipahami dalam arti yang lebih abstrak di sini. Sebuah gambar dapat mewakili adegan realistis dari dunia nyata, tapi grafik seperti *pie chart* atau histogram serta antarmuka pengguna grafik dari sebuah perangkat lunak juga dianggap sebagai gambar [3].

3. PROCESSING

Processing adalah bahasa pemrograman open source [1][2] dan lingkungan bagi orang-orang yang ingin membuat gambar, animasi, dan interaksi. Awalnya dikembangkan untuk melayani sebagai sketsa perangkat lunak dan mengajarkan dasar-dasar pemrograman komputer dalam konteks visual, *Processing* juga berkembang menjadi alat untuk menghasilkan pekerjaan profesional. Saat ini, ada puluhan ribu mahasiswa, seniman, desainer, peneliti, dan penggemar yang menggunakan *Processing* untuk belajar, *prototyping*, dan produksi [1][5].

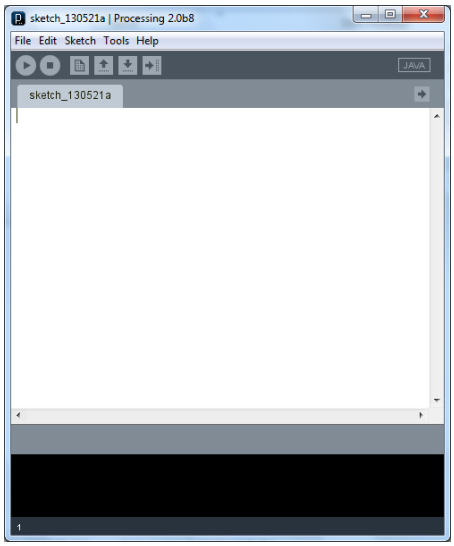
Processing juga termasuk dalam Lingkungan Pengembangan Terpadu (*Integrated Development Environment*, IDE) yang dibangun untuk seni elektronik, seni media baru, dan komunitas desain visual dengan tujuan mengajarkan dasar-dasar pemrograman komputer dalam konteks visual, dan untuk melayani dasar sketsa elektronik. Proyek ini dimulai pada tahun 2001 oleh Casey Reas dan Benyamin Fry, dari Kelompok Estetika dan Perhitungan di MIT Media Lab. Salah satu AIMS dinyatakan *Processing* bertindak sebagai alat untuk mendapatkan *non-programmer* dimulai dengan pemrograman, melalui kepuasan instan umpan balik visual. Bahasa ini bukan hanya dibangun di atas bahasa Java, melainkan menggunakan sintaks sederhana dan model pemrograman grafik [2].

Konsep perangkat lunak *Processing* yang berhubungan dengan prinsip-prinsip bentuk visual, gerak, dan interaksi. Hal ini terintegrasi dengan metodologi bahasa pemrograman, lingkungan pengembangan, dan pengajaran menjadi sistem ed Unifi. *Processing* diciptakan untuk mengajarkan dasar-dasar pemrograman komputer dalam konteks visual, melayani sebagai sketsa perangkat lunak, dan digunakan sebagai alat produksi [5].

Bahasa *Processing* adalah bahasa pemrograman teks yang khusus dirancang untuk menghasilkan dan memodifikasi gambar. *Processing* berusaha mencapai keseimbangan antara kejelasan dan fitur canggih. Pemula dapat menulis program sendiri setelah hanya beberapa menit dari instruksi, namun pengguna yang lebih maju dapat mempekerjakan dan menulis *library* dengan fungsi tambahan. Sistem ini difasilitasi teknik mengajar komputer grafik dan banyak interaksi Termasuk vektor / raster gambar, pengolahan citra, model warna, *mouse*, dan *keyboard*, jaringan komunikasi, dan pemrograman berorientasi obyek (*object oriented programming*). Library dengan mudah memperpanjang *Processing* untuk menghasilkan suara, mengirim / menerima data dalam format yang beragam, dan untuk impor / ekspor dari format file 2D dan 3D [5].

4. PEMBAHASAN

Sebelum menggunakan *Processing*, terlebih dahulu penulis mendowloadnya dari situs: http://www.processing.org/download/. Perangkat lunak Processing yang tersedia untuk Linux, Macintosh, dan Windows [5]. Dalam penelitian ini, penulis menggunakan Processing versi 2.0 Beta 8 (http://processing.googlecode.com/files/processing-2.0b8-windows32.zip) dengan sistem operasi Windows 7 Enterprise 32 bit dan lakukan ekstrak file processing-2.0b8-windows32.zip. Di dalam folder processing-2.0b8-windows32 terdapat file processing.exe dan jalankan file processing.exe dengan mengklik dua kali, maka akan muncul tampilan berikut ini :



Gambar 1. Tampilan Awal Processing 2.0b8

Pada gambar 1 di atas terdapat kotak berbentuk persegi panjang berwarna putih yang disebut editor teks (*text editor)*, yang merupakan salah satu bagian dari *Processing Development Environment* (PDE). PDE terdiri dari editor teks yang sederhana untuk menulis kode, area pesan (*message area*), konsol teks (*text console*), tab untuk mengelola file, toolbar dengan tombol untuk melakukan tindakan, dan serangkaian menu. Ketika program dijalankan, akan membuka jendela baru yang disebut sebagai *display window* [5].

Dalam pembahasan ini, penulis akan memperkenalkan sistem koordinat dari *display window* dan berbagai bentuk geometris. Sintaks yang akan diperkenalkan, yaitu : size(), point(), line(), triangle(), quad(), rect(), dan ellipse() [5].

4.1. size()

Ukuran sebuah *display window* dapat diatur dengan menggunakan fungsi size() :

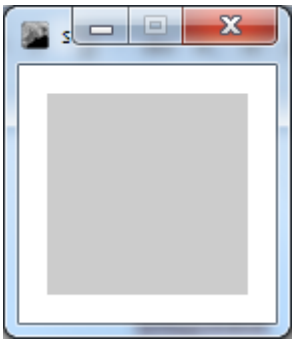
```
size(lebar, tinggi);
```

Fungsi size() memiliki dua parameter, yaitu yang pertama menetapkan lebar jendela dan yang kedua menetapkan tingginya [5].

Dalam percobaan fungsi size() ini, yaitu membuat *display window* dengan ukuran lebar 100 piksel dan tinggi 100 piksel, dengan menggunakan sintaks :

```
size(100, 100);
```

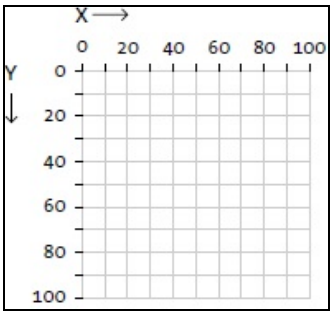
kemudian sintaks tersebut dijalankan, maka akan diperoleh tampilan sebagai berikut :



Gambar 2. Tampilan *Display Window* dengan Lebar 100 Piksel dan Tinggi 100 Piksel

Posisi pada layar terdiri dari koordinat x dan koordinat y. Koordinat x adalah jarak horizontal dari titik 0 dan koordinat y adalah jarak vertical dari titik 0. Dalam *Processing*, titik 0 adalah sudut kiri atas dari *display window* dan mengkoordinasikan nilai menuju ke bawah dan ke kanan [5]. Hal ini berarti koordinat x terletak dari titik 0 menuju ke kanan secara horizontal dan koordinat y terletak dari titik 0 menuju ke bawah secara vertical.

Gambar 2 di atas, jika diubah menjadi sistem koordinat akan terlihat sebagai berikut :



Gambar 3. Sistem Koordinat pada *Display Window* dengan Lebar 100 Piksel dan Tinggi 100 Piksel [5]

4.2. point()

Sebuah titik merupakan elemen visual yang sederhana dan dapat digambar dengan menggunakan fungsi point() :

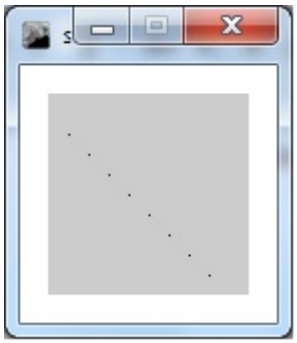
point(x, y);

Fungsi point memiliki dua parameter, yaitu yang pertama adalah koordinat x dan yang kedua adalah koordinat y. Kecuali ditentukan lain, yaitu sebuah titik adalah ukuran sebuah piksel tunggal [5].

Percobaan yang dilakukan penulis dengan menggunakan fungsi point() ini, yaitu untuk menampilkan 8 buah titik dengan satu kali eksekusi, dengan koordinat setiap titiknya, yaitu: (10,20), (20,30), (30,40), (40,50), (50,60), (60,70), (70,80), dan (80,90). Percobaan tersebut menggunakan sintaks berikut ini :

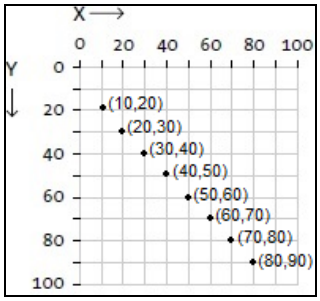
```
point(10,20);
point(20,30);
point(30,40);
point(40,50);
point(50,60);
point(60,70);
point(70,80);
point(80,90);
```

selanjutnya sintaks pada percobaan fungsi point() di atas dijalankan, akan menghasilkan tampilan sebagai berikut :



Gambar 4. Tampilan 8 Buah Titik: (10,20), (20,30), (30,40), (40,50), (50,60), (60,70), (70,80), dan (80,90)

Sistem koordinat dari Gambar 4 di atas dapat ditunjukkan oleh gambar berikut ini :



Gambar 5. Sistem Koordinat 8 Buah Titik: (10,20), (20,30), (30,40), (40,50), (50,60), (60,70), (70,80), dan (80,90)

4.3. line()

Sebuah garis merupakan serangkaian titik, garis yang lebih sederhana digambar dengan fungsi line(). Fungsi ini memiliki empat parameter, dua untuk setiap titik akhir:

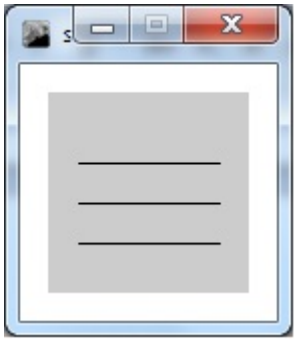
line(x_1, y_1, x_2, y_2);

Dua parameter yang pertama digunakan untuk mengatur posisi di mana garis dimulai dan dua parameter yang terakhir digunakan untuk mengatur posisi di mana garis berhenti [5].

Pada fungsi line() ini, penulis melakukan percobaan dengan membuat 3 buah garis dengan satu kali eksekusi, dengan koordinat setiap garisnya, yaitu: (15, 35, 85, 35), (15, 55, 85, 55), dan (15, 75, 85, 75). Sintaks yang dapat dituliskan adalah :

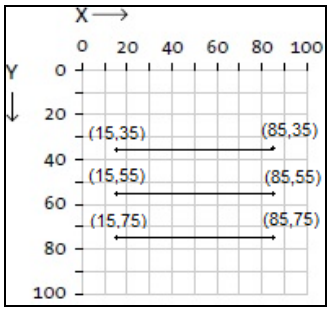
```
line(15, 35, 85, 35);
line(15, 55, 85, 55);
line(15, 75, 85, 75);
```

kemudian sintaks-sintaks fungsi line() tersebut dijalankan, akan menampilkan 3 buah garis sebagai berikut :



Gambar 6. Tampilan 3 Garis: (15, 35, 85, 35), (15, 55, 85, 55), dan (15, 75, 85, 75)

Gambar 6 di atas merupakan bentuk dari sistem koordinat di bawah ini :



Gambar 7. Sistem Koordinat 3 buah Garis : (15, 35, 85, 35), (15, 55, 85, 55), dan (15, 75, 85, 75)

4.4. triangle()

Fungsi triangle() digunakan untuk menggambar segitiga. Fungsi ini memiliki enam parameter, dua untuk setiap titik:

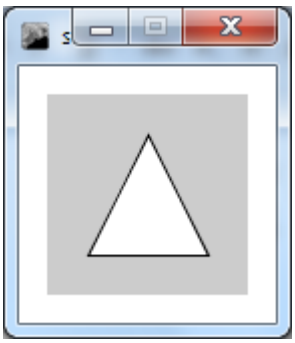
triangle(x_1, y_1, x_2, y_2, x_3, y_3);

Pasangan pertama mendefinisikan titik yang pertama, pasangan tengah mendefinisikan titik yang kedua, dan pasangan terakhir mendefinisikan titik yang ketiga. Setiap segitiga dapat ditarik dengan menghubungkan tiga baris, tetapi fungsi triangle() memungkinkan untuk menggambar sebuah bentuk yang terisi. Segitiga dari segala bentuk dan ukuran dapat dibuat dengan mengubah nilai parameter [5].

Penulis melakukan percobaan untuk membuat bentuk segitiga sama sisi dengan titik-titik kooordinat: (50, 20), (20, 80), dan (80, 80). Dalam percobaan ini, penulis menggunakan sintaks sebagai berikut:

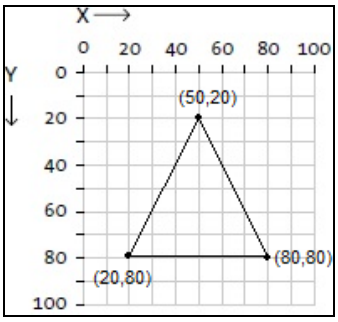
triangle(50, 20, 20, 80, 80, 80);

jika sintaks tersebut dieksekusi, akan menampilkan bentuk segitiga sama sisi:



Gambar 8. Tampilan Segitiga Sama Sisi dengan Titik-Titik Koordinat: (50, 20), (20, 80), dan (80, 80)

Grafik sistem koordinat dari gambar 8 ditunjukkan oleh gambar berikut ini:



Gambar 9. Sistem Koordinat Segitiga Sama Sisi dengan Titik-Titik Koordinat: (50, 20), (20, 80), dan (80, 80)

4.5. quad()

Fungsi quad() digunakan untuk menggambar bentuk segi empat, poligon bersisi empat. Fungsi ini memiliki delapan parameter, dua untuk setiap titik:

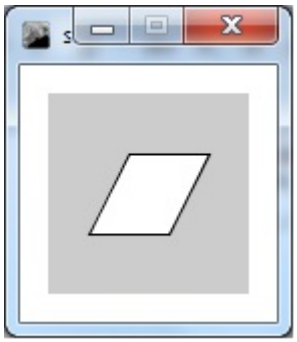
quad(x_1, y_1, x_2, y_2, x_3, y_3, x_4, y_4);

Mengubah nilai parameter dapat menghasilkan persegi panjang, bujur sangkar, jajaran genjang, dan segiempat yang beraturan [5].

Dalam percobaan fungsi quad() ini, penulis membuat bentuk jajaran genjang dengan titik-titik koordinat: (40, 30), (80, 30), (60, 70), dan (20, 70) dan menggunakan sintaks:

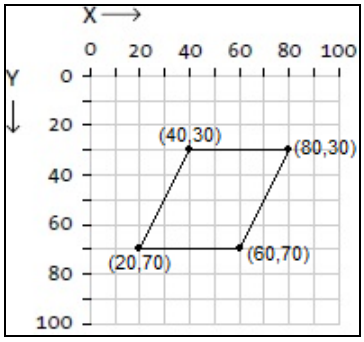
quad(40, 30, 80, 30, 60, 70, 20, 70);

maka menghasilkan keluaran berupa bentuk jajaran genjang:



Gambar 10. Tampilan Jajaran Genjang dengan Titik-Titik Koordinat: (40, 30), (80, 30), (60, 70), dan (20, 70)

Tampilan pada gambar 10 di atas merupakan bentuk dari sistem koordinat yang ditunjukkan oleh gambar di bawah ini:



Gambar 11. Sistem Koordinat Jajaran Genjang dengan Titik-Titik Koordinat: (40, 30), (80, 30), (60, 70), dan (20, 70)

4.6. rect()

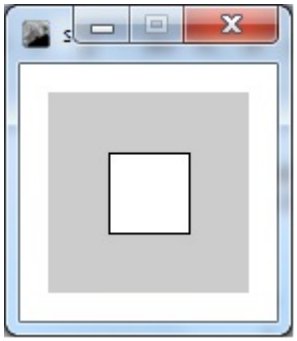
Fungsi rect() digunakan untuk menggambar bentuk persegi panjang:
rect(x, y, lebar, tinggi);

Dua parameter pertama digunakan untuk menetapkan lokasi pojok kiri atas, set yang ketiga adalah lebar, dan set yang keempat adalah tinggi. Menggunakan nilai yang sama untuk parameter lebar dan tinggi untuk menggambar bentuk persegi [5].

Dengan fungsi rect() ini, penulis melakukan percobaan untuk membuat bentuk bangun bujursangkar dengan melibatkan sebuah titik koordinat (30,30), lebar 40 piksel, dan tinggi 40 piksel. Sintaks yang dapat dituliskan adalah sebagai berikut:

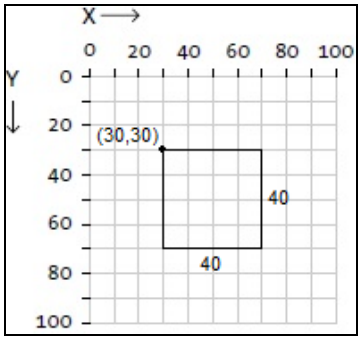
rect(30, 30, 40, 40);

selanjutnya sintak rect(30, 30, 40, 40); ini dijalankan dan menampilkan bentuk bangun bujursangkar seperti berikut ini:



Gambar 12. Tampilan Bujursangkar dengan Titik Koordinat (30, 30), Lebar 40 Piksel, dan Tinggi 40 Piksel

Bentuk sistem koordinat bujursangkar dari gambar 12 di atas ditunjukkan oleh gambar di bawah ini:



Gambar 13. Sistem Koordinat Bujursangkar dengan Titik Koordinat (30, 30), Lebar 40 Piksel, dan Tinggi 40 Piksel

4.7. ellipse()

Fungsi ellipse() digunakan untuk menggambar bentuk ellips pada *display window*:

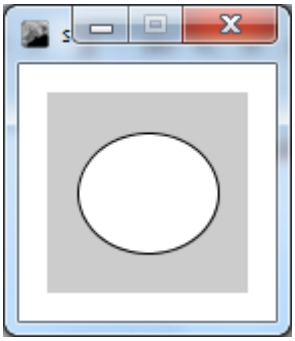
ellipse(x, y, lebar, tinggi);

Dua parameter yang pertama untuk menetapkan lokasi pusat ellips, set yang ketiga adalah lebar, dan set yang keempat adalah tinggi. Menggunakan nilai yang sama untuk parameter lebar dan tinggi akan membentuk sebuah lingkaran [5].

Percobaan fungsi ellipse() ini merupakan percobaan terakhir yang dilakukan penulis dengan melibatkan sebuah titik koordinat (50,50), lebar 70 piksel, dan tinggi 60 piksel.

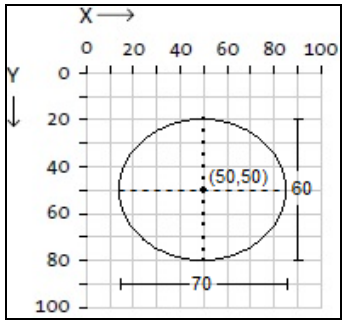
ellipse(50, 50, 70, 60);

selanjutnya sintaks ellipse(50, 50, 70, 60); di-*running*, akan menghasilkan tampilan berikut ini:



Gambar 14. Tampilan Ellips dengan Titik Koordinat (50, 50), Lebar 70 Piksel, dan Tinggi 60 Piksel

Gambar 14 di atas merupakan bentuk dari sistem koordinat yang ditunjukkan oleh gambar berikut ini:



Gambar 15. Sistem Koordinat Ellips dengan Titik Koordinat (50, 50), Lebar 70 Piksel, dan Tinggi 60 Piksel

5. PENUTUP

5.1. KESIMPULAN

Berdasarkan percobaan-percobaan yang telah dilakukan oleh penulis di atas, maka dapat diambil beberapa kesimpulan, yaitu:

1. Posisi koordinat y positif dimulai dari titik nol yang terletak di pojok kiri paling atas menuju ke bawah.
2. Percobaan-percobaan bahasa pemrograman Processing di atas menghasilkan tampilan bentuk bangun yang sesuai dengan yang dharapkan.

5.2. SARAN

Percobaan-percobaan yang telah dilakukan oleh penulis di atas masih sangat sederhana, maka masih perlu dilakukan :

1. Percobaan untuk pembuatan animasi, gambar 2D, gambar 3D, dan percobaaan-percobaan lain yang dapat dilakukan dengan menggunakan bahasa pemrograman *Procesing*.
2. Penelitian yang mengarah ke pengolahan citra (*image processing*), pengenalan pola (*pattern recognition*), dan teknologi multimedia.

DAFTAR PUSTAKA

Klawonn, F., 2008, *Introduction to Computer Graphics Using Java 2D and 3D*, Springer-Verlag London Limited.

Mount, D.M., 2004, *CMSC 427 Computer Graphics*, Department of Computer Science, University of Maryland.

Reas, C. and Fry, B., 2007, *Processing: A Programming Handbook For Visual Designers and Artists*, The MIT Press Cambridge, Massachusetts London, England.

_____, [online]: http://www.processing.org, diakses pada tanggal 7 April 2013.

_____, 2013, *Processing Programming Language*, [online]: http://en.wikipedia.org/wiki/Processing_(programming_language), diakses pada tanggal 17 Mei 2013.