PENERAPAN ALGORITMA DDA (*DIGITAL DIFFIRENTIAL ANALYZER*) MENGGUNAKAN VISUAL BASIC 6.0

Novia Fatimah

Universitas Gunadarma, novia_f@staff.gunadarma.ac.id

ABSTRAK

Garis merupakan salah satu elemen grafik geometri 2D. Garis dapat digambarkan ke dalam komputer dimana cara menggambarnya berbeda dengan menggambar cara manual, karena layar komputer bukanlah bidang kosong seperti bidang kosong dalam lembaran kertas, melainkan bidang yang terdiri dari kotak-kotak berukuran tertentu yang disebut sebagai piksel. Sehingga untuk menggambarkan suatu garis pada komputer adalah dengan cara mengaktifkan piksel-piksel tertentu menggunakan algoritma tertentu sehingga terlihat menyerupai seperti garis. Salah satu algoritma yang dapat digunakan untuk menggambar garis adalah algoritma DDA (Digital Diffirential Analyzer), yaitu suatu algoritma untuk membuat garis sederhana yang memanfaatkan tingkat kemiringan garis. Agar dapat melihat bagaimana suatu garis sesungguhnya terbentuk di dalam layar komputer maka dibuatlah visualisasi algoritma ini dengan menggunakan perangkat lunak Visual Basic 6.0. Secara keseluruhan, penelitian ini menggunakan metode kualitatif studi kasus mengenai salah satu elemen grafik geometri 2D, dan dalam pembuatan aplikasi, penulis menggunakan metode Rapid Application Development (RAD), yaitu suatu metode pengembangan sistem dengan proses model perangkat lunak incremental dengan siklus pengembangan yang singkat, ruang lingkup yang kecil, dan tim yang kecil yang terdiri atas client dan pengembang, dimana baik client maupun pengembangnya adalah penulis sendiri. Hasil akhir penelitian ini adalah suatu program siap pakai yang berupa Aplikasi DDA. Penelitian ini dapat digunakan untuk media ajar Grafika Komputer dengan materi menggambar elemen grafik 2D dan media ajar Pemrograman Komputer, baik sebagai alat bantu menerangkan dasar teori maupun sebagai alat bukti teori yang bersifat interaktif sehingga memudahkan memahami materi yang disampaikan karena imajinatif, tidak monoton dan tidak membosankan.

Kata kunci*: *DDA, Digital Differential Analyzer, Garis, VB 6.0, Visual Basic 6.0

PENDAHULUAN

Grafik geometri terbagi menjadi grafik geometri 2D dan 3D, dimana grafik geometri 3D merupakan pengembangan dari grafik geometri 2D. Grafik geometri 2D terdiri dari beberapa elemen. Elemen terkecilnya adalah elemen titik. Dari elemen titik dapat berkembang menjadi elemen garis, dan dari elemen garis dapat berkembang menjadi tiga elemen yang lebih rumit yaitu poligon, kurva, atau lingkaran. Selanjutnya dari elemen poligon, kurva, dan lingkaran pada grafik geometri 2D dapat berkembang lagi menjadi grafik geometri 3D.

Elemen-elemen grafik geometri dapat digambarkan ke dalam komputer. Cara komputer menggambar elemen-elemen grafik geometri tersebut tidaklah sama dengan cara menggambar secara manual. Misalnya ingin menggambarkan suatu garis ke layar komputer, karena sesungguhnya layar komputer bukanlah bidang kosong, melainkan bidang yang terdiri dari kotak-kotak dengan ukuran tertentu yang disebut sebagai piksel, maka untuk menggambarkan suatu garis tidak dapat langsung begitu saja menggambarnya, melainkan dengan cara mengaktifkan piksel-piksel tertentu

sehingga terlihat menyerupai seperti garis lurus.

Pengaktifan piksel-piksel tertentu pada komputer dilakukan melalui perintah dalam bentuk kode program, dan dibutuhkan algoritma tertentu untuk melakukannya sesuai dengan elemen grafik geometri yang akan digambar, misalnya elemen garis. Buku ataupun berkas komputer (*file*) sebagai media pembelajaran secara manual memiliki kekurangan dalam menyampaikan isinya, yaitu kurang interaktif, sehingga agak sulit membangun imajinasi pembacanya dalam mencapai pemahaman mengenai konsep menggambar elemen grafik geometri pada komputer. Seperti dijelaskan dalam tulisan Edi Ismanto bersama Eka Pandu Cynthia yang berjudul *Drill and Practice Model* dalam Pembuatan Media Pembelajaran Interaktif Pembentukan Objek Primitif Sederhana Dua Dimensi (2017:18) yang mengatakan bahwa buku sebagai media ajar mulai ditinggalkan sebab sangat monoton sehingga menimbulkan kebosanan. Kemudian seperti yang dijelaskan dalam tulisan Aulia Nurul Zahra bersama Mila Rosidana dan Ratna Sari dalam tulisannya yang berjudul Implementasi Algoritma DDA Pada Pemrograman Java Netbeans (2018) yang mengatakan bahwa di kalangan mahasiswa masih sulit untuk memahami terlebih menyelesaikan suatu permasalahan teknologi yaitu pemahaman pembentukan garis pada komputer dalam pembelajaran Grafika Komputer.

Tujuan penelitian ini adalah membuat suatu aplikasi yang dapat memperlihatkan dan menjelaskan bagaimana sebenarnya suatu elemen grafik geometri 2D digambarkan ke dalam layar komputer. Hasil akhir dari aplikasi ini memperlihatkan bagaimana penampakan atau wujud elemen grafik geometri 2D yang digambarkan secara *digital* ke dalam piksel-piksel layar komputer.

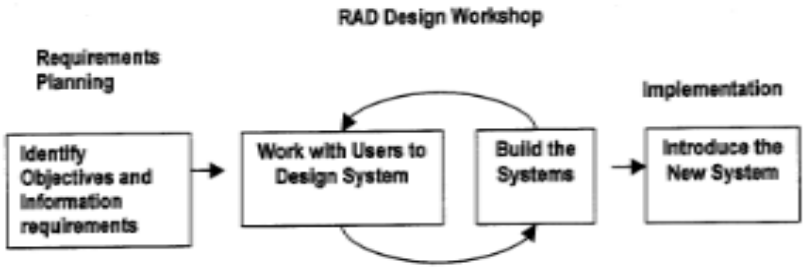
Penulisan ini hanya membahas mengenai elemen garis yaitu bagaimana cara menggambar garis lurus dengan kemiringan tertentu pada layar komputer atau menggambar garis *digital*. Seperti penjelasan pada paragraf tiga di atas bahwa menggambar elemen-elemen grafik geometri 2D pada komputer membutuhkan suatu algoritma. Bressenham dan DDA (*Digital Differential Analyzer*) merupakan algoritma-algoritma untuk menggambar garis lurus pada komputer. Dari dua algoritma tersebut, DDA (*Digital Diffirential Analyzer*) adalah algoritma yang penulis pilih sebagai model pada penelitian ini, dilengkapi dengan Visual Basic 6.0 sebagai perangkat lunak yang penulis gunakan untuk menerapkan model tersebut ke dalam bahasa pemrograman.

METODE PENELITIAN

Metode penelitian ini dilakukan dengan menggunakan metode kualitatif studi kasus salah satu elemen grafik geometri 2D, dimana target penelitian adalah menghasilkan suatu produk berupa aplikasi komputer berbasis visual grafis.

Dalam rangka mencapai target penelitian tersebut, penulis melakukan dua aktivitas utama, yaitu aktivitas penggalian informasi melalui studi pustaka untuk mendapatkan dasar-dasar teori yang berkaitan dengan penelitian ini dari buku-buku, jurnal-jurnal, dan website atau blog yang menjadi referensi yang berhubungan dengan penyusunan penulisan penelitian. Kemudian penulis juga melakukan aktivitas pembuatan dan pengembangan penelitian dengan menggunakan metode *Rapid Application Development* (RAD), yaitu suatu metode pengembangan sistem dengan proses model perangkat lunak

inkremental yang menekankan siklus pengembangan yang singkat. Penulis memilih metode ini karena metode ini relatif lebih sesuai dengan rencana pengembangan aplikasi yang tidak memiliki ruang lingkup yang besar dan dikembangkan oleh tim yang kecil yang terdiri atas pengguna (*client*) dan tim pengembang, dimana yang berperan sebagai pengguna (*client*) dan tim pengembang (*programmer*) adalah penulis sendiri. Siklus hidup Metode RAD digambarkan seperti berikut di bawah ini:



Gambar 1. Tahapan Metode *Rapid Application Development* (RAD)
Sumber: Agustinus Noertjahyana (2002).

Putra (2020) dalam tulisannya yang berjudul 6+ Metode Pengembangan Perangkat Lunak (Waterfall, Rad, Agile, Prototype Dll) menjelaskan bahwa tahapan kerja metode RAD pada Gambar 1 terbagi menjadi tiga tahap. Tahap awal dari Metode RAD adalah Rencana Kebutuhan (*Requirements Planning*) yaitu identifikasi tujuan yang langsung diiringi dengan komunikasi dan perancangan, dimana seluruh pihak baik pengguna maupun pengembang sebagai sebuah tim terlibat aktif dalam setiap perumusannya. Tahap kedua Proses Desain (*Design Workshop*) adalah proses mendesain sistem atau perangkat lunak atau aplikasi sesuai kebutuhan, tahap ini masih melibatkan semua pihak. *Client* atau pengguna ikut terjun dalam menguji coba perangkat lunak. Perbaikan pun langsung diterapkan jika pengguna menemukan kesalahan. Ketika pengguna terpuaskan dengan desain perangkat lunak, setelah melalui berbagai perbaikan, barulah proses kerja menginjak pada tahap terakhir, yaitu Implementasi (*Implementation*), pada tahap ini *programmer* mengembangkan desain menjadi suatu program komputer, setelah program selesai sebagian atau secara keseluruhan, dilakukan pengujian terhadap program tersebut untuk mengetahui tingkat keberhasilan pengimplementasian.

HASIL DAN PEMBAHASAN

Seperti yang dijelaskan di awal bahwa tujuan penelitian ini adalah membuat suatu aplikasi berbasis visual grafis untuk memperlihatkan bagaimana sesungguhnya komputer menggambarkan elemen grafik geometri 2D. Kemudian hasil akhir dari aplikasi ini akan memperlihatkan wujud sesungguhnya dari elemen grafik tersebut ketika digambarkan secara *digital* ke dalam piksel-piksel yang ada pada layar komputer. Aplikasi ini berhasil dibuat dengan mengikuti tahapan-tahapan dalam metode pengembangan sistem RAD seperti yang dijelaskan dalam penjabaran selanjutnya berikut ini.

Rencana Kebutuhan

Tahap pertama pengembangan sistem pada metode RAD adalah melakukan perencanaan kebutuhan

yaitu mengidentifikasi tujuan dan perancangan. Sesuai tujuan pengembangan sistem yang telah disebutkan, maka hal pertama yang dilakukan adalah menentukan elemen grafik geometri 2D yang akan digambar ke layar komputer. Untuk itu, penelitian ini menentukan elemen garis lurus. Selanjutnya menentukan algoritma yang akan digunakan untuk menggambarkan garis lurus tersebut ke dalam layar komputer. Untuk itu, penelitian ini memilih algoritma DDA (*Digital Differential Analyzer*), yaitu suatu algoritma menggambar garis sederhana dengan memanfaatkan tingkat kemiringan garis.

Setelah menentukan elemen grafik geometri 2D yang akan digambarkan dan menentukan algoritma yang digunakan untuk menggambarnya, selanjutnya mengidentifikasi perancangan. Untuk identifikasi tujuan dan perancangan selanjutnya, terlebih dahulu harus mengetahui beberapa konsep dasar mengenai menggambar obyek pada layar komputer dan konsep garis.

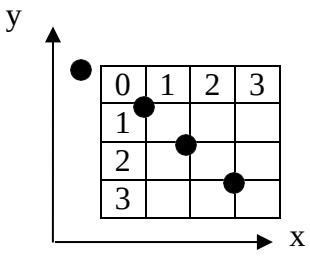
Konsep menggambar obyek pada layar komputer dijabarkan sebagai berikut: Pada layar komputer, suatu titik terletak pada posisi (x, y) di piksel tertentu. Posisi titik tersebut akan menentukan piksel yang aktif untuk membentuk suatu obyek. Misal, posisi titik dalam piksel yang terletak di koordinat (0, 0), (1, 1), (2, 2) dan (3, 3) diperlihatkan seperti ilustrasi gambar berikut ini (berturut-turut dimulai dari bawah ke atas).

Gambar di atas memperlihatkan posisi suatu titik-titik koordinat (0, 0), (1, 1), (2, 2) dan (3, 3) pada piksel-piksel yang membentuk layar komputer. Angka-angka dalam piksel hanyalah ilustrasi untuk memperlihatkan nomor piksel. Dari gambar tersebut di atas, misal akan menggambar obyek titik di koordinat (1, 1) dan (3, 3), maka piksel yang aktif untuk koordinat (1, 1) dan (3, 3) akan terlihat seperti gambar berikut ini (Gambar 3).

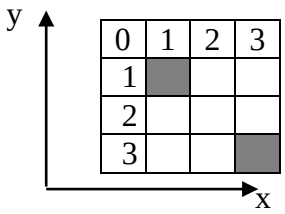
Berdasar konsep tersebut teridentifikasi bahwa nantinya aplikasi akan terdapat tampilan yang memperlihatkan piksel-piksel, sehingga penulis perlu melakukan perancangan untuk menghasilkan tampilan layar yang memperlihatkan piksel.

Berikutnya konsep garis. Sebuah garis merupakan kumpulan titik-titik yang tersusun sedemikian rupa sehingga memiliki pangkal dan ujung. Titik-titik di antara titik pangkal dan ujung akan dibentuk dengan mengacu kepada kedua titik tersebut, yaitu mulai dari titik pangkal hingga ke titik ujung. Misalnya menggambar garis dengan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (4, 5) seperti gambar berikut ini (Gambar 4).

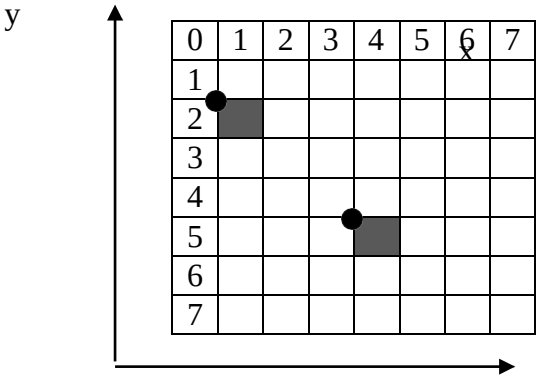
Untuk mendapatkan titik-titik selanjutnya sampai ke titik ujung maka perlu dilakukan peningkatan/inkrementasi atas nilai koordinat sumbu x dan y pada titik sebelumnya. Peningkatan nilai koordinat sumbu x dan y dapat dilihat pada tabel berikut ini (Tabel 1).



Gambar 2. Titik dalam piksel dengan sumbu y relatif terhadap layar komputer



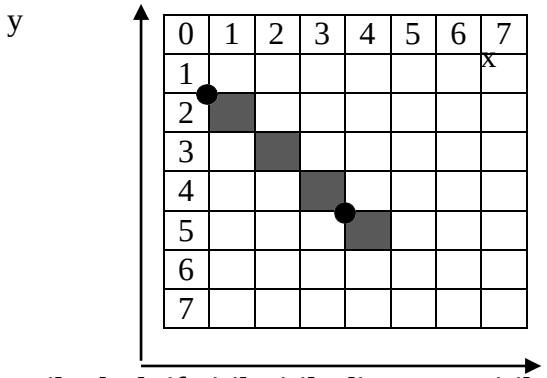
Gambar 3. Piksel aktif titik di koordinat (1, 1) dan (3, 3).



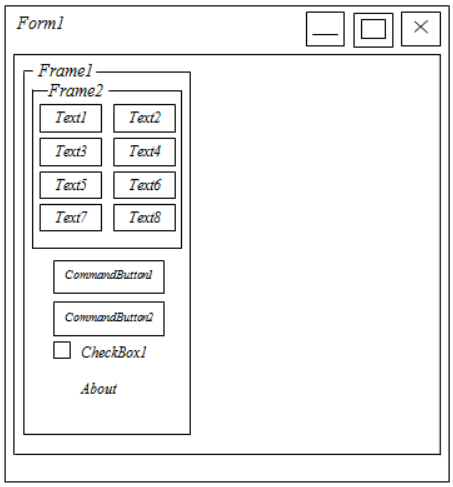
Gambar 4. Titik dan piksel aktif penggambaran garis dengan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (4, 5), sumbu y relatif terhadap layar komputer

Tabel 1.
Peningkatan/inkrementasi nilai koordinat sumbu x dan y.

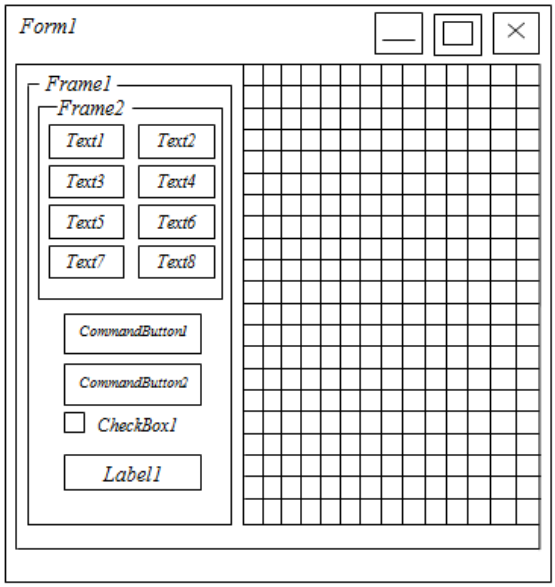
Langkah	x_1	y_1	x_{next}	y_{next}
1	1	2	1 + 1 = 2	2 + 1 = 3
2	2	3	2 + 1 = 3	3 + 1 = 4
3	3	4	3 + 1 = 4	4 + 1 = 5
4	4	5		



Gambar 5. Garis dan piksel aktif titik-titik di antara titik pangkal dan ujung, dengan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (4, 5), sumbu y relatif terhadap layar komputer



Gambar 6. Desain rancangan obyek-obyek masukan-keluaran tanpa *grid*



Gambar 7. Desain rancangan obyek-obyek

Tabel 1 memperlihatkan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (4, 5). Untuk mendapatkan titik-titik selanjutnya dimulai dari titik pangkal hingga mencapai titik ujung dilakukan peningkatan sebesar 1 baik untuk sumbu x maupun sumbu y. Dimulai dari koordinat (1, 2), x dan y berikutnya adalah koordinat (2, 3) didapat dari (1+1, 2+1). Dan begitu seterusnya hingga mencapai titik ujung, dimana titik ujung ini dihasilkan dari koordinat (3, 4). Dengan mengikuti pola penambahan sumbu x dan y yaitu bertambah 1 untuk masing-masing sumbu x dan y pada koordinat (3, 4) maka hasilnya adalah titik ujung (4, 5) yang didapat dari (3+1, 4+1). Setelah didapatkan titik-titik selanjutnya hingga mencapai titik ujung seperti yang terlihat pada Tabel 1, maka Gambar 4 berubah menjadi seperti berikut ini.

Dari penjabaran di atas, teridentifikasi bahwa perancangan masukan-proses-keluaran dimulai dengan menentukan titik pangkal dan ujung sebagai masukan yang akan ditentukan oleh pengguna. Kemudian untuk prosesnya adalah menentukan titik-titik di antara titik pangkal dan

ujung yang didapat dari proses inkrementasi, proses inkrementasi ini akan dilakukan secara otomatis, artinya bukan tugas pengguna menentukan titik-titik di antara titik pangkal dan ujung, melainkan tugas program. Hal ini dilakukan dengan cara mengkodekan algoritma DDA ke dalam bahasa pemrograman. Selanjutnya, sebagai keluaran akan menghasilkan tampilan piksel aktif dan garis berwarna, kedua tampilan tersebut merupakan keluaran utama pada aplikasi ini. Keluaran lainnya adalah tampilan layar yang akan memperlihatkan layar tanpa *grid* dan layar dengan *grid* –*grid* adalah garis-garis pada layar yang memperlihatkan garis bingkai pada piksel-. Karena fokus utamanya bukan pemrograman membuat tampilan grid, maka pemrograman membuat *grid* ini tidak akan dibahas lebih lanjut, namun hasil pemrograman akan diperlihatkan baik dalam tahap desain maupun hasil implementasi.

Selain identifikasi tujuan dan perancangan juga mengidentifikasi komponen-komponen yang akan mendukung pengembangan dan mendukung implementasi aplikasi yang akan dibuat. Komponen-komponen tersebut antara lain komponen perangkat lunak dan perangkat keras minimal yang harus dimiliki untuk dapat mengembangkan dan menjalankan program aplikasi yang akan dibuat. Perangkat lunak yang digunakan untuk menerapkan algoritma DDA pada penelitian ini adalah Visual Basic 6.0, yang merupakan bahasa pemrograman berbasis sistem operasi Windows 9.x atau versi yang lebih tinggi. Oleh karena itu hasil penerapan algoritma DDA ini merupakan suatu program aplikasi yang berbasis Windows yang dapat dijalankan pada sistem operasi 32 bit Windows 9x hingga Windows 7. Sementara perangkat keras yang digunakan untuk menjalankan dengan baik dan optimal aplikasi yang dihasilkan adalah komputer IBM PC atau yang kompatibel dengan prosesor 486 ke atas dengan minimal RAM 16 MB.

Proses Desain

Selanjutnya adalah tahap desain. Ini merupakan tahap kedua dari metode RAD. Sesuai dengan hasil identifikasi tujuan dan perancangan maka selanjutnya dibuat desain rancangan masukan dan keluaran aplikasi yang akan dibuat. Masukan dan keluaran tersebut berfungsi untuk memberikan gambaran secara konseptual fungsi memasukkan data dan menampilkan hasil proses yang dilakukan oleh aplikasi setelah data dimasukkan.

Desain dibagi menjadi dua area, yaitu sisi sebelah kanan dan sisi sebelah kiri. Sisi kanan merupakan area untuk menampilkan keluaran, sementara sisi kiri merupakan area untuk masukan, tombol proses, dan informasi mengenai aplikasi. Pada sisi kanan memiliki dua tampilan, yaitu tampilan tanpa garis dan tampilan bergaris. Garis pada tampilan di sisi kanan ini disebut sebagai *grid*, yaitu garis yang memperlihatkan garis-garis tepi pada piksel. Berikut diperlihatkan tampilan desain rancangan masukan dan keluaran yang memperlihatkan tampilan dengan *grid* dan tanpa *grid*.

Gambar 6 tersebut merupakan desain rancangan aplikasi tanpa menampilkan *grid*, tujuannya agar dapat memperlihatkan tampilan tanpa garis-garis tepi (*border*) yang membentuk piksel. Artinya ketika pengguna menjalankan program untuk menggambar suatu garis dari titik pangkal tertentu hingga titik ujung tertentu, keluaran yang dihasilkan dari program adalah garis dan atau piksel yang membentuk garis tersebut, dimana satu piksel yang membentuk garis diwakili dengan satu kotak, namun kotak-kotak piksel tersebut tidak

memiliki garis-garis tepi (*border*) sehingga dikatakan ini sebagai keluaran garis dan piksel tanpa *grid*.

Gambar 7 tersebut merupakan desain rancangan aplikasi yang menampilkan *grid*, tujuannya agar dapat memperlihatkan tampilan dengan garis-garis tepi (*border*) yang membentuk piksel. Artinya ketika pengguna menjalankan program untuk menggambar suatu garis dari titik pangkal tertentu hingga titik ujung tertentu, keluaran yang dihasilkan dari program adalah garis dan atau piksel yang membentuk garis tersebut, dimana satu piksel yang membentuk garis diwakili dengan satu kotak, dan kotak-kotak piksel tersebut memiliki garis-garis tepi (*border*) sehingga dikatakan ini sebagai keluaran garis dan piksel dengan *grid*.

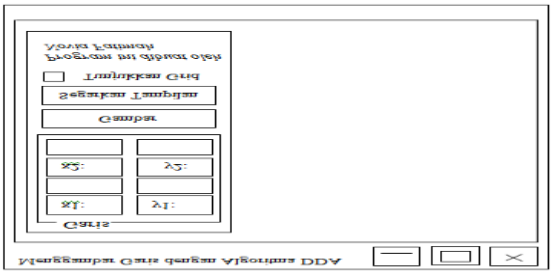
Kedua desain Gambar 6 dan 7 diterapkan pada obyek *Form1*. Obyek *Form1* tersebut dibagi menjadi dua area, yaitu area kiri dan kanan. Untuk membuatnya terlihat rapih maka area sebelah kiri dikelompokkan ke dalam obyek *Frame1*.

Obyek *Frame1* berisi obyek *Frame2*, *CommandButton1*, *CommandButton2*, *CheckBox1*, dan *Label1*. Obyek *Frame2* berfungsi untuk mengelompokkan obyek-obyek *TextBox1* (yang selanjutnya disebut sebagai *Text1*) sampai *TextBox8* (yang selanjutnya disebut sebagai *Text8*). *CommandButton1* sebagai tombol Gambar. Apabila tombol ini ditekan akan menghasilkan gambar garis di area *form* sebelah kanan, garis yang dihasilkan dalam bentuk piksel berwarna biru dan atau garis berwarna magenta (kata "dan atau" menyatakan bahwa piksel dan garis dapat ditampilkan secara bersamaan atau ditampilkan secara bergantian). *CommandButton2* sebagai tombol Segarkan Tampilan, yang jika ditekan akan menghapus tampilan di area kanan *form*. *CheckBox1* digunakan untuk mengaktifkan atau menon-aktifkan *grid* yang akan tampil di area kanan *form*. *Label1* digunakan untuk menampilkan informasi mengenai program Aplikasi DDA, yang berisi informasi nama pembuat atau pengembang aplikasi.

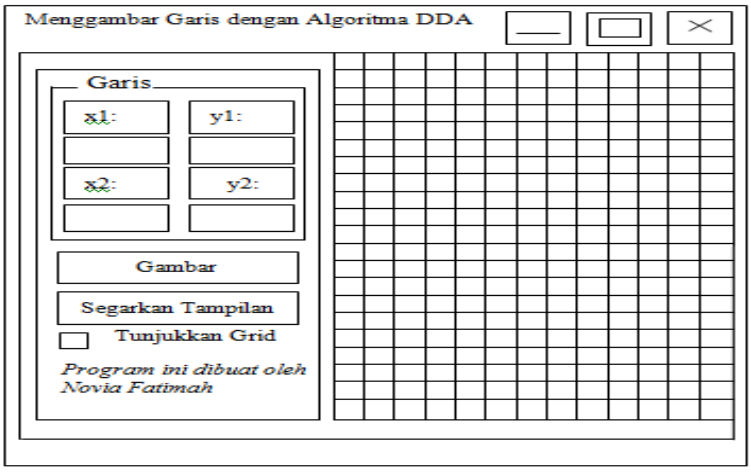
Obyek *Frame2* berisi *Text1* hingga *Text8*. Obyek *Text1*, *Text2*, *Text5*, dan *Text6* digunakan untuk menampilkan keterangan nama koordinat sebagai x_1, y_1, x_2, dan y_2. *Text3*, *Text4*, *Text7*, dan *Text8* adalah fasilitas bagi pengguna untuk memasukkan data koordinat di titik x_1, y_1, x_2, dan y_2 yang diinginkan.

Selanjutnya area kanan *form*, apabila *CommandButton1* atau Tombol Gambar tidak ditekan, maka area ini tidak berisi obyek apapun, karena semua yang ditampilkan di area ini baik *grid*, piksel, maupun garis, terbentuk sebagai hasil eksekusi dari kode-kode program, bukan berupa obyek.

Desain rancangan masukan-keluaran tersebut diharapkan menjadi seperti yang terlihat pada Gambar 8 dan 9 berikut ini.



Gambar 8. Desain rancangan masukan-keluaran tanpa *grid*



Gambar 9. Desain rancangan masukan-keluaran dengan *grid*.

Gambar 8 merupakan desain rancangan tampilan awal masukan-keluaran tanpa garis pada sisi kanan. Desain ini sama seperti Gambar 6 namun dengan area sisi kiri tidak lagi berupa obyek dengan masing-masing nama obyeknya melainkan sudah dalam bentuk keluaran yang diharapkan sebagai hasil menentukan properti *caption* dan *text* untuk masing-masing obyek seperti berikut ini: obyek *Frame1*, properti *caption* adalah null (dikosongkan); obyek *Frame2*, properti *caption* adalah Garis; obyek *Text1*, properti *text* adalah x_1; obyek *Text2*, properti *text* adalah y_1; obyek *Text3* dan *Text4*, properti *text* adalah null (dikosongkan); obyek *Text5*, properti *text* adalah x_2; obyek *Text6*, properti *text* adalah y_2; obyek *Text7* dan *Text8*, properti *text* adalah null (dikosongkan); obyek *CommandButton1*, properti *caption* adalah Garis; obyek *CommandButton2*, properti *caption* adalah Segarkan Tampilan, obyek *CheckBox1*, properti *caption* adalah Tunjukkan Grid, dan obyek *Label1*, properti *caption* adalah Program ini dibuat oleh Novia Fatimah.

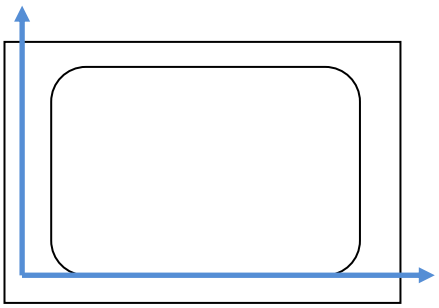
Gambar 9 merupakan desain rancangan tampilan awal masukan-keluaran dengan tampilan bergaris pada sisi kanan. Desain ini sama seperti Gambar 7 namun dengan area sisi kiri tidak lagi berupa obyek dengan masing-masing nama obyeknya melainkan sudah dalam bentuk keluaran yang diharapkan sebagai hasil menentukan properti *caption* dan *text* untuk masing-masing obyek seperti yang dijabarkan pada narasi Gambar 8.

Implementasi

Setelah tahap desain selesai, dilanjutkan dengan tahap terakhir pada metode pengembangan sistem yang penulis pakai –metode RAD-, yaitu tahap implementasi. Pada tahap ini penulis selaku pembuat program mengembangkan desain pada tahap kedua menjadi suatu program komputer sekaligus menerapkan algoritma DDA ke dalam bahasa pemrograman Visual Basic 6.0. Setelah program selesai sebagian atau secara keseluruhan, dilakukan pengujian terhadap program tersebut untuk mengetahui tingkat keberhasilan pengimplementasian.

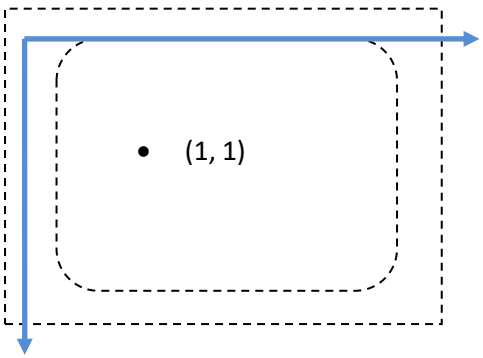
Sumbu Y pada Layar Komputer

Sebelum membuat kode-kode program, perlu diperhatikan bahwa terdapat perbedaan sumbu y antara dunia nyata dan layar komputer. Sumbu y pada dunia nyata dimulai dari bawah ke atas, sehingga apabila terdapat titik di koordinat (1, 1), pada dunia nyata akan digambarkan seperti gambar berikut ini:



Gambar 10a. Koordinat (1, 1) Dunia Nyata

Sumbu y pada layar komputer dimulai dari atas ke bawah, sehingga apabila terdapat titik di koordinat (1, 1), pada layar komputer akan digambarkan seperti gambar berikut ini:



Gambar 10b. Koordinat (1, 1) di layar komputer

Gambar 10b digunakan sebagai acuan oleh penulis untuk melakukan pemrograman sedemikian rupa agar tampilan koordinat di layar komputer akan tampil sesuai dengan koordinat yang digambarkan di dunia nyata seperti yang diperlihatkan pada Gambar 10a. Misal untuk mendapatkan tampilan suatu titik koordinat (1, 1) di layar komputer agar posisinya seperti di dunia nyata (di sudut kiri bawah), maka penulis membuat kode program dengan kode sebagai berikut ini:

PSet(1, *tinggi_layar – 1)*

Menggambar Garis dengan Visual Basic 6.0

Selanjutnya adalah menggambar garis dengan titik pangkal dan titik ujung di suatu koordinat. Kode program untuk menggambar garis dengan warna *default* pada Visual Basic 6.0 adalah:

Line(x1, y1) – (x2, y2)

Sementara kode program untuk menggambar garis dengan warna lain pada Visual Basic 6.0 adalah:

Line(x1, y1) – (x2, y2),
vb_colour_constant

Untuk diperhatikan, hasil dari perintah tersebut di atas tidak dapat langsung terlihat oleh mata dikarenakan resolusi komputer yang digunakan. Semakin tinggi resolusinya maka piksel semakin tidak terlihat oleh mata pengguna. Oleh sebab itu, agar hasilnya dapat terlihat oleh mata pengguna, maka penulis melakukan perbesaran tampilan sebesar 200x dari resolusi komputer yang digunakan, artinya untuk membentuk 1 piksel yang dapat terlihat oleh pengguna maka membutuhkan 200 piksel dalam pemrograman.

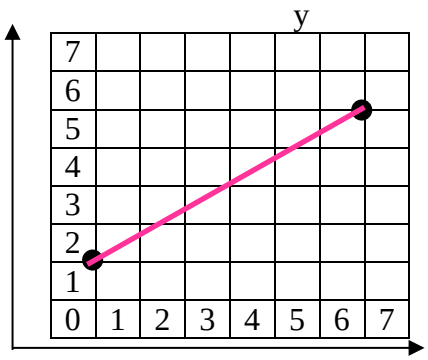
Misalnya untuk menggambar garis berwarna magenta dengan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (7, 6), maka penulis melakukan proses pengkodean dengan ketentuan sebagai berikut: untuk memperbesar piksel sebesar 200x maka sumbu x maupun sumbu y dikalikan 200; untuk menampilkan sumbu x di sebelah obyek *Frame1*, maka yang seharusnya x bernilai 1 menjadi bernilai 1 + lebar *Frame1*; kemudian agar sumbu y ditampilkan sesuai dengan sumbu y nyata (yaitu dimulai dari bawah ke atas) dengan mengacu kepada obyek *Frame1*, maka yang seharusnya y bernilai 2 menjadi bernilai tinggi *Frame1* – 2. Dengan menggunakan ketentuan tersebut di atas maka kode program membuat garis dengan titik pangkal di koordinat (1, 2) dan titik

ujung di koordinat (7, 6) berwarna magenta adalah:

```
x1 = 1 * 200 + Frame1.Width
y1 = Frame1.Height - (2 * 200)
x2 = 7 * 200 + Frame1.Width
y2 = Frame1.Height - (6 * 200)

Line (x1, y1)-(x2, y2), vbMagenta
```

Dengan menggunakan perintah *Line* dan menerapkan algoritma DDA pada bahasa pemrograman Visual Basic 6.0, titik-titik koordinat yang berada di antara keduanya akan terbentuk secara otomatis sehingga menghasilkan tampilan seperti berikut ini:



Gambar 11. Garis berwarna magenta yang terbentuk dengan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (4, 5)

Gambar 11 memperlihatkan hasil eksekusi yang diharapkan dari perintah *Line* pada bahasa pemrograman Visual Basic 6.0 dengan mengikuti algoritma DDA. Perintah *Line* tersebut memiliki atribut vbMagenta yang menghasilkan garis berwarna magenta. Garis yang terbentuk dimulai dari titik pangkal (x_1, y_1) di koordinat (1, 2) dan titik ujung (x_2, y_2) di koordinat (4, 5). Mengingat resolusi komputer yang sudah semakin tinggi, maka garis yang terbentuk dapat terlihat karena masing-masing titik x dan y mendapatkan perbesaran 200 kali. Kemudian garis yang dihasilkan mengikuti posisi sumbu y nyata (yaitu dimulai dari bawah ke atas).

Menggambar Piksel Aktif dengan Visual Basic 6.0

Berikutnya adalah menggambar piksel yang aktif yang akan membentuk sebuah garis. Untuk menggambar piksel aktif, penulis memanfaatkan fungsi membuat kotak dalam Visual Basic 6.0 yaitu dengan perintah:

Line (x_1, y_1)-Step(step_x, -step_y), *vb_colour_constant*, B

Dengan posisi sumbu y relatif terhadap layar komputer (y = 0 berada di posisi atas), keterangan kode program di atas adalah perintah *line* (x_1, y_1) akan memposisikan *pointer* pada komputer di koordinat (x_1, y_1), selanjutnya -Step(step_x, -step_y) akan membuat *pointer* bergerak ke kanan sebanyak step_x titik dan ke atas sebanyak step_y titik, sehingga perintah *line* (x_1, y_1) -Step(step_x, -step_y) memberikan tanda pada koordinat (x_1, y_1) dan ($x_1$$_x$+ step_x, y_1 + step_y). Argumen B pada perintah tersebut berfungsi untuk membuat kotak dengan acuan titik (x_1 + step_x, y_1 + step_y) dan (x_1, y_1) yang sudah ditandai.
Misalnya menggambar piksel berwarna biru di koordinat (1, 2) dengan ukuran 1 piksel-nya sebesar 200, maka perintahnya adalah:

```
Line (1*200, 2*200)-Step(200, -200), vbBlue, B
```

Dengan posisi sumbu y sesuai posisi pada layar komputer (y = 0 berada di posisi atas), keterangan kode program di atas adalah perintah *line* (1*200, 2*200) akan memposisikan *pointer* pada komputer di koordinat (200, 400), selanjutnya -Step(200, -200) akan membuat *pointer* bergerak ke kanan sebanyak 200 titik dan ke atas sebanyak 200 titik, sehingga perintah

line (1*200, 2*200) -Step(200, -200) memberikan tanda pada koordinat (200, 400) dan (400, 200), selanjutnya argumen B pada perintah tersebut membuat kotak dengan acuan titik (400, 200) dan (200, 400) yang sudah ditandai. vbBlue akan menyebabkan garis yang membentuk kotak berwarna biru. Untuk mengisi area dalam pada kotak dilakukan dengan perintah:

FillStyle = vbSolid

Perintah di atas akan menyebabkan area dalam kotak yang terbentuk berwarna gelap. Untuk menyelaraskan warna area dalam kotak dengan warna garis kotak, gunakan perintah:

FillColor = &HFF8888

Perintah di atas akan mengubah warna gelap pada area dalam kotak yang dihasilkan akibat perintah FillStyle = vbSolid menjadi warna biru dengan kode warna biru FF8888 heksadesimal (kombinasi heksadesimal untuk Red FF + Green 88 + Blue 88).

Menggambar Garis dan Piksel Aktif dengan Visual Basic 6.0

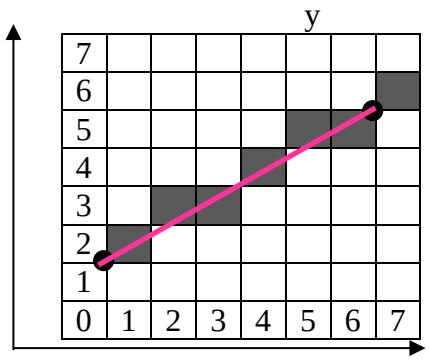
Pemrograman menggambar garis mengikuti langkah-langkah pada algoritma DDA (*Digital Differential Analyzer*) sebagai berikut: tentukan nilai koordinat titik pangkal x_1 dan y_1, kemudian tentukan nilai koordinat titik ujung x_2 dan y_2. Dari nilai koordinat titik pangkal dan ujung tersebut, hitung nilai kemiringan, dengan mengasumsikan m sebagai nilai kemiringan, hitung m dengan cara membagi antara hasil pengurangn y_2 dan y_1 dengan pengurangan x_2 dan x_1 $\left(m = \frac{y_2 - y_1}{x_2 - x_1}\right)$. Setelah didapat nilai m, selanjutnya tentukan nilai m tersebut memenuhi salah satu dari tiga kondisi berikut ini: (1) jika x_1 lebih kecil sama dengan x_2, jika m bernilai lebih besar dari 0 atau m lebih kecil dari 1, maka gambar piksel, tambahkan 1 pada nilai titik x_1 atau $(x_1 = x_1 + 1)$, dan tambahkan m pada nilai titik y_1 atau $(y_1 = y_1 + m)$. (2) jika y_1 lebih kecil sama dengan y_2, jika m bernilai lebih besar dari 1, maka gambar piksel, tambahkan $\frac{1}{m}$ pada nilai titik x_1 atau $\left(x_1 = x_1 + \frac{1}{m}\right)$, dan tambahkan 1 pada nilai titik y_1 atau $(y_1 = y_1 + 1)$. (3) jika x_1 lebih kecil sama dengan x_2, jika m bernilai sama dengan 1, maka gambar piksel, tambahkan 1 pada masing-masing nilai titik x_1 dan y_1 atau $(x_1 = x_1 + 1 \; dan \; y_1 = y_1 + 1)$.
Lakukan pembulatan ke atas pada titik x_1 dan y_1 hanya untuk menggambar piksel yang aktif, sementara untuk perhitungan nilai m dan nilai titik x_1 dan y_1, tetap menggunakan nilai tanpa pembulatan. Ulangi setiap langkah mulai dari langkah menghitung nilai m dengan nilai x_1 dan y_1 adalah nilai yang baru dan nilai x2 dan y2 tidak berubah (tetap titik ujung yang telah ditentukan di awal).

Dengan kombinasi pengkodean menggambar garis dan kotak, langkah-langkah dalam algoritma DDA, dan teknik pemrograman maka untuk membuat garis dan piksel aktif dengan titik pangkal di koordinat (1, 2) dan titik ujung di koordinat (7, 6) akan menghasilkan perhitungan m dan koordinat x_1, y_1, x_2 dan y_2. Seperti berikut ini:

Tabel 2.
Alamat parameter piksel aktif dari mulai titik pangkal di koordinat (1, 2) hingga titik ujung di koordinat (7, 6).

x_1	y_1	x_2	y_2	m	0 < m < 1?	m > 1?	m = 1?
1	2	7	6	(6 – 2) / (7 – 1) = 0,67	ya	-	-
2	2,67	-	-	(6 – 2,67) / (7 – 2) = 0,67	ya	-	-
3	3,34	-	-	(6 – 3,34) / (7 – 3) = 0,67	ya	-	-
4	4,01	-	-	(6 – 4,01) / (7 – 4) = 0,66	ya	-	-
5	4,67	-	-	(6 – 4,67) / (7 – 5) = 0,67	ya	-	-
6	5,34	-	-	(6 – 5,34) / (7 – 6) = 0,66	ya	-	-
7	6	-	-	-	-	-	-

Berdasar hasil perhitungan dan pembulatan nilai x_1 dan y_1 dan x_2 dan y_2 pada tabel di atas, piksel-piksel aktif di antara titik pangkal (1, 2) dan titik ujung (7, 6) dengan sumbu y relatif terhadap dunia nyata, akan secara otomatis terbentuk seperti berikut ini:

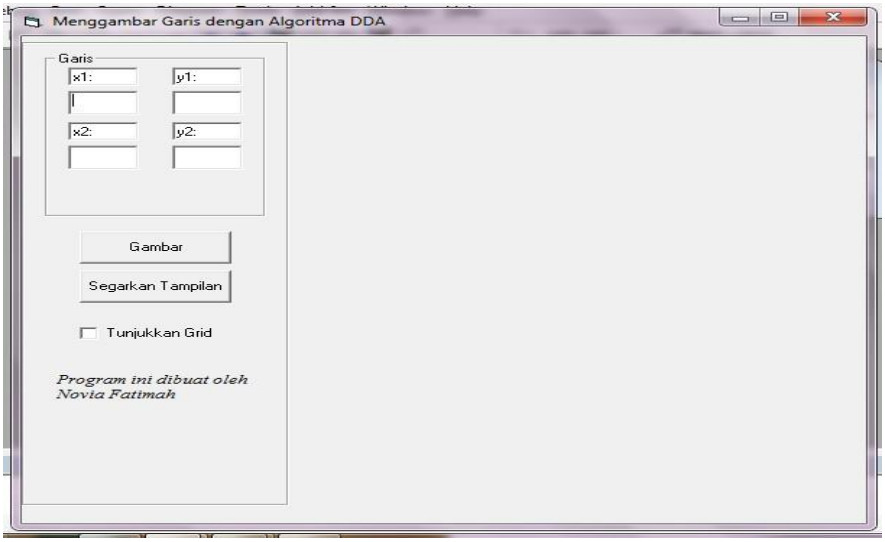


Gambar 12. Piksel aktif dari mulai titik pangkal di koordinat (1, 2) hingga titik ujung di koordinat (7, 6), sumbu y relatif terhadap dunia nyata.

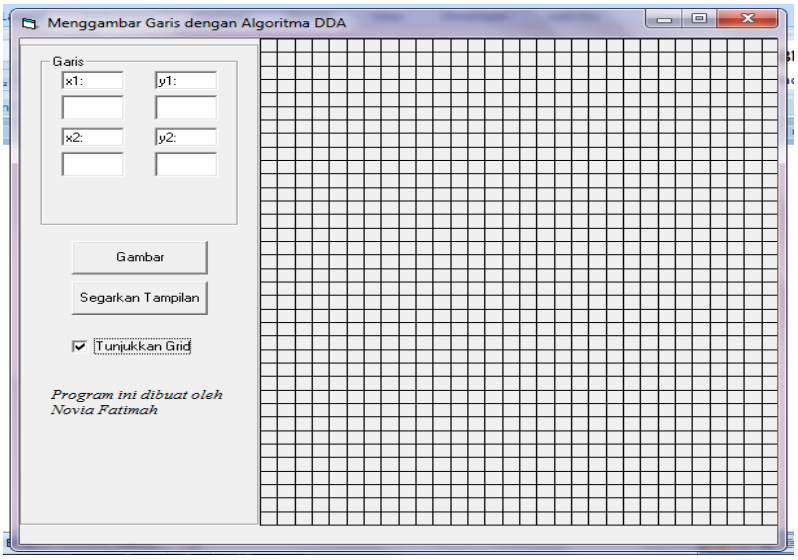
Gambar 12 memperlihatkan hasil eksekusi yang diharapkan dari perintah *Line* pada bahasa pemrograman Visual Basic 6.0 dengan mengikuti algoritma DDA. Algoritma DDA tersebut menentukan kemiringan garis dan piksel-piksel yang aktif. Garis yang terbentuk dimulai dari titik pangkal (x_1, y_1) di koordinat (1, 2) dan titik ujung (x_2, y_2) di koordinat (4, 5). Kemudian garis yang dihasilkan mengikuti posisi sumbu y nyata (yaitu dimulai dari bawah ke atas).

Hasil Keluaran Program

Berikut ini merupakan hasil eksekusi program aplikasi DDA yang berhasil dibuat.



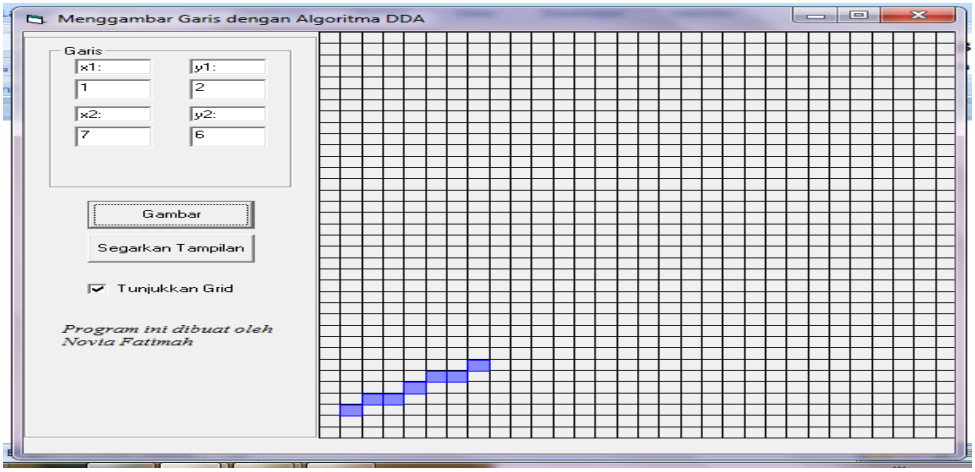
Gambar 13. Tampilan awal tanpa grid



Gambar 14. Tampilan awal dengan grid

Gambar di atas merupakan tampilan awal apabila *CheckBox* **Tunjukkan Grid** tidak dicentang, apabila *CheckBox* tersebut dicentang maka tampilan berubah menjadi seperti Gambar 14. Untuk memberi atau menghilangkan tanda centang pada *CheckBox* cukup lakukan klik kiri *mouse* pada *CheckBox* tersebut.

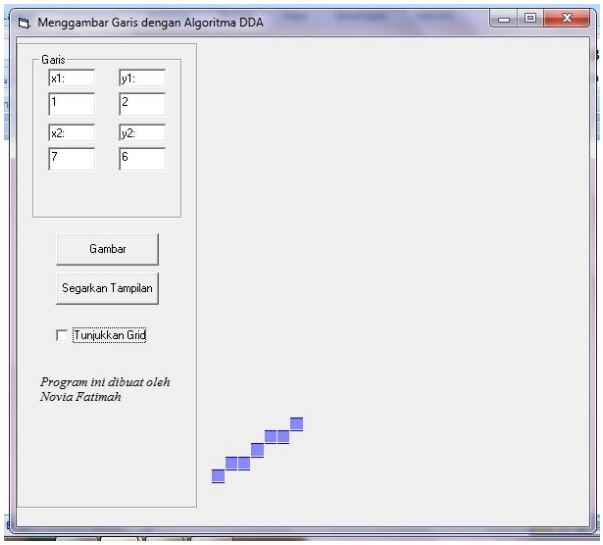
Dari tampilan awal, selanjutnya memasukkan nilai koordinat (x_1, y_1) sebagai titik pangkal dan nilai koordinat (x_2, y_2) sebagai titik ujung. Apabila kotak masukan x_1 diketik 1, kotak masukan y_1 diketik 2, kotak masukan x_2 diketik 7, kotak masukan y_2 diketik 6, *CheckBox* Tunjukkan Grid ditekan, kemudian tombol Gambar ditekan, maka akan menampilkan piksel aktif seperti gambar berikut ini:



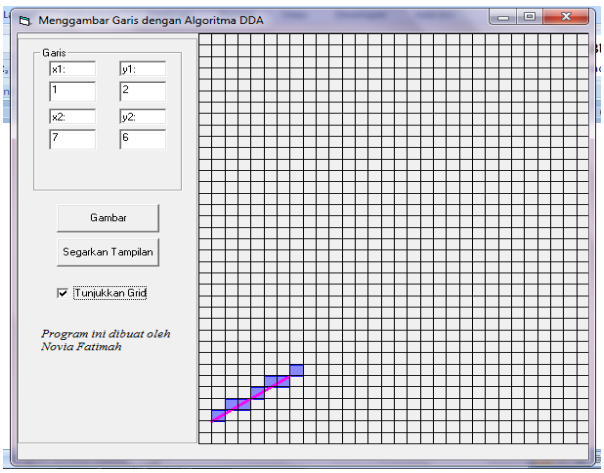
Gambar 15. Tampilan menggambar garis dengan piksel aktif pada koordinat (x1, y1) = (1, 2) hingga koordinat (x2, y2) = (7, 6) dengan *grid* diaktifkan dan sumbu y relatif terhadap dunia nyata.

Gambar 15 memperlihatkan piksel-piksel yang aktif yang membentuk garis dari titik pangkal (1, 2) hingga titik ujung (7, 6). Piksel-piksel yang aktif terlihat berikut dengan garis-garis tepi (*border*) yang membentuk kotak piksel. Garis-garis tepi (*border*) ini terlihat karena centang pada *CheckBox* Tunjukkan Grid diaktifkan dengan cara menekan *CheckBox* Tunjukkan Grid. Piksel yang aktif ditunjukkan dengan kotak berwarna biru. Perhatikan, piksel aktif tersebut sesuai dengan kotak-kotak piksel yang terlihat pada Gambar 12 dan alamat parameter piksel pada Tabel 2.

Kemudian tekan tombol *CheckBox* Tunjukkan Grid untuk menghilangkan tanda centang pada tombol *CheckBox* tersebut, maka piksel aktif tetap terlihat, namun tanpa menampilkan *grid*. Perhatikan gambar berikut ini:



Gambar 16. Tampilan menggambar garis dengan piksel aktif pada koordinat (x1,y1) = (1, 2) hingga koordinat (x2, y2) = (7, 6) dengan *grid* dinon-aktifkan dan sumbu y relatif terhadap dunia nyata



Gambar 18. Tampilan menggambar garis dengan piksel dan garis berwarna magenta aktif pada koordinat (x1, y1) = (1, 2) hingga koordinat (x2, y2) = (7, 6) dengan *grid* diaktifkan dan sumbu y relatif terhadap dunia nyata

Gambar 18 memperlihatkan garis dan piksel-piksel yang aktif yang membentuk garis dari titik pangkal (1, 2) hingga titik ujung (7, 6). Piksel-piksel yang aktif terlihat berikut garis-garis tepi (*border*) yang membentuk kotak piksel. Garis-garis tepi (*border*) ini kembali terlihat karena centang pada *CheckBox* Tunjukkan Grid diaktifkan kembali dengan cara menekan *CheckBox* Tunjukkan Grid. Piksel yang aktif ditunjukkan dengan kotak berwarna biru. Perhatikan, piksel aktif tersebut sesuai dengan kotak-kotak piksel yang terlihat pada Gambar 12 dan alamat parameter piksel pada Tabel 2.

Apabila tombol **Segarkan Tampilan** ditekan, maka tampilan kembali seperti tampilan awal tanpa *grid* (Gambar 15) namun *Text3*, *Text4*, *Text7*, dan *Text8* tetap berisi data masukan koordinat di titik *x1*, *y1*, *x2*, dan *y2* (pada contoh ini *Text3*, *Text4*, *Text7*, dan *Text8* tetap berisi 1, 2, 7, dan 6).

Text3, *Text4*, *Text7*, dan *Text8* dibiarkan tetap terisi dengan alasan agar tidak perlu mengulang pemasukkan data apabila ingin menguji kembali data sebelumnya. Namun apabila ingin mengganti data, lakukan langsung dengan cara menyorotnya dan masukkan data yang baru.

KESIMPULAN DAN SARAN

Penerapan algoritma DDA (*Digital Differential Analyzer*) dengan menggunakan perangkat lunak Visual Basic 6.0 berhasil dilakukan baik untuk pembuatan garis maupun piksel aktif. Algoritma DDA ini merupakan algoritma menggambar garis lurus pada komputer yang memanfaatkan tingkat kemiringan tertentu. Hasil akhir dari penerapan ini berupa program aplikasi yang siap pakai yang diberi nama Aplikasi DDA.

Pengguna dapat mengoperasikan program aplikasi ini dengan mudah, tanpa harus mempelajarinya secara khusus. Melalui program Aplikasi DDA ini, pengguna mendapatkan gambaran bagaimana sesungguhnya penampakan atau wujud garis yang digambarkan ke layar komputer, dapat membantu imajinasi pengguna yang mempelajari konsep menggambar garis pada materi ajar Grafika Komputer, dan dapat juga digunakan sebagai contoh penerapan suatu algoritma ke dalam bahasa pemrograman dalam pelajaran Algoritma & Pemrograman Komputer, sekaligus sebagai contoh aplikasi dalam pelajaran Pemrograman Komputer dengan perangkat lunak Visual Basic. Dengan kata lain Aplikasi DDA ini dapat digunakan sebagai alat bantu menerangkan dasar teori maupun sebagai alat bukti teori yang bersifat interaktif sehingga memudahkan memahami materi yang disampaikan karena imajinatif, tidak monoton dan tidak membosankan.

Pengembangan Aplikasi DDA ini dapat dilakukan melalui penambahan fitur memperbesar dan memperkecil ukuran piksel, sehingga dapat terlihat perbedaan garis yang dihasilkan antara piksel dengan resolusi tinggi dan rendah. Tambahkan juga batas maksimal data masukan agar tidak menghasilkan pesan kesalahan *overflow* jumlah piksel pada layar komputer dikarenakan tiap data masukan akan dikali dengan bilangan 200. Juga dapat ditambahkan fitur tampilan dengan posisi sumbu y titik-titik koordinat yang membentuk garis

sesuai dengan pososi sumbu y pada layar komputer yaitu dimulai dari atas ke bawah seperti yang terlihat pada Gambar 10b.

DAFTAR PUSTAKA

Akhtar727. (2014, Agustus 2). Working with graphics in visual basic 6. Diakses dari https://www.go4expert.com/articles/graphics-visual-basic-6-t30202/

Firdaus. (2007). *64 trik tersembunyi visual basic 6.* Palembang, Indonesia: Maxicom.

Ismanto, E. & Cynthia, E. P. (2017). Drill and practice model dalam pembuatan media pembelajaran interaktif pembentukan objek primitif sederhana dua dimensi. *Jurnal Ilmu Komputer dan Informatika*, 01(01), 18-23. http://jurnal.uinsu.ac.id/index.php/algoritma/article/view/1304

Mansfield, R. (1995). *Panduan berilustrasi visual basic dalam aplikasi*. Jakarta, Indonesia: Dinastindo.

Norton's, P. (1998) *Guide to visual basic 6*. SAMS Publishing.

Nugroho, E. (2005). *Teori dan praktek grafika komputer menggunakan delphi dan opengl*. 1st edn. Yogyakarta, Indonesia: Graha Ilmu.

Noertjahyana, A. (2002). Studi analisis rapid aplication development sebagai salah satu alternatif metode pengembangan perangkat lunak. *Jurnal Informatika*, 2(2). https://ced.petra.ac.id/index.php/inf/article/view/15819

Putra. (2020, Februari 1). 6+ metode pengembangan perangkat lunak (waterfall, rad, agile, prototype dll). Diakses dari https://salamadian.com/metode-pengembangan-perangkat-lunak/

Zahra, A. N.; Rosidana, M. & Sari, R. (2018). Implementasi algoritma DDA pada pemrograman java netbeans. [PDF File]. [Available from Academia https://www.academia.edu/download/56696414/JURNAL_JADI.pdf