**LAPORAN TUGAS KECIL 2**

**IF2211 STRATEGI ALGORITMA**

Membangun Kurva Bézier dengan Algoritma Titik Tengah berbasis *Divide and Conquer*



Disusun oleh:

Agil Fadillah Sabri (13522006)

**Program Studi Teknik Informatika**

**Sekolah Teknik Elektro dan Informatika**

**Institut Teknologi Bandung**

**2024**

**DAFTAR ISI**

DAFTAR ISI i

DAFTAR GAMBAR ii

DAFTAR TABEL iii

BAB I DESKRIPSI MASALAH 1

BAB II LANGKAH PEMECAHAN MASALAH

* 1. Analisis dan Implementasi dalam Algoritma *Brute Force*
  2. Analisis dan Implementasi dalam Algoritma *Divide and Conquer*

BAB III IMPLEMENTASI PROGRAM

* 1. Implementasi dalam Algoritma *Brute Force*
  2. Implementasi dalam Algoritma *Divide and Conquer*

BAB IV UJI COBA

* 1. Keterangan Proses Input dan Output
  2. Kurva Bézier Kuadratik
  3. Kurva Bézier N Titik

BAB V ANALISIS SOLUSI

BAB VI IMPLEMENTASI BONUS

* 1. Visualisasi Pembangkitan Kurva
  2. Generalisasi Algoritma

DAFTAR PUSTAKA

LAMPIRAN

**DAFTAR GAMBAR**

Gambar 1. Kurva Bézier Kubik 1

Gambar 2. Implementasi Program (1) 5

Gambar 3. Implementasi Program (2) 6

Gambar 4. Implementasi Program (3) 7

Gambar 5. Implementasi Program (4) 8

Gambar 6. Implementasi Program (5) 9

Gambar 7. Implementasi Program (6) 10

Gambar 8. Tampilan Awal Program 11

Gambar 9. Tampilan Input Tidak Valid 11

Gambar 10. Contoh Test 1 (test1.txt) 12

Gambar 11. Hasil Luaran Test 1 12

Gambar 12. Hasil Luaran Test 1 (*txt file*) 12

Gambar 13. Contoh Test 2 (test2.txt) 13

Gambar 14. Hasil Luaran Test 2 13

Gambar 15. Hasil Luaran Test 2 (*txt file*) 13

Gambar 16. Contoh Test 3 (test3.txt) 14

Gambar 17. Hasil Luaran Test 3 14

Gambar 18. Hasil Luaran Test 3 (*txt file*) 14

Gambar 19. Contoh Test 4 (test4.txt) 15

Gambar 20. Hasil Luaran Test 4 15

Gambar 21. Hasil Luaran Test 4 (*txt file*) 15

Gambar 22. Contoh Test 5 (test5.txt) 16

Gambar 23. Hasil Luaran Test 5 16

Gambar 24. Hasil Luaran Test 5 (*txt file*) 16

Gambar 25. Contoh Test 6 (Terminal) 17

Gambar 26. Hasil Luaran Test 6 17

Gambar 27. Hasil Luaran Test 6 (*txt file*) 17

Gambar 28. Contoh Test 7 (Terminal) 18

Gambar 29.Hasil Luaran Test 7 18

Gambar 30. Hasil Luaran Test 7 (*txt file*) 18

Gambar 31. Contoh Test 8 (Terminal) 19

Gambar 32.Hasil Luaran Test 8 19

Gambar 33. Hasil Luaran Test 8 (*txt file*) 19

Gambar 34. Contoh Test 9 (Terminal) 20

Gambar 35.Hasil Luaran Test 9 20

Gambar 36. Hasil Luaran Test 9 (*txt file*) 20

Gambar 37. Contoh Test 10 (Terminal) 21

Gambar 38.Hasil Luaran Test 10 21

Gambar 39. Hasil Luaran Test 10 (*txt file*) 21

ii

**DAFTAR TABEL**

Tabel 1. Contoh Matriks pada *Cyberpunk 2077 Breach Protocol*  1

Tabel 2. Contoh Masukan 1 dari File Berekstensi txt 12

Tabel 3. Contoh Masukan 2 dari File Berekstensi txt 13

Tabel 4. Contoh Masukan 3 dari File Berekstensi txt 14

Tabel 5. Contoh Masukan 4 dari File Berekstensi txt 15

Tabel 6. Contoh Masukan 5 dari File Berekstensi txt 16

Tabel 7. Contoh Masukan 1 dari Terminal 17

Tabel 8. Contoh Masukan 2 dari Terminal 18

Tabel 9. Contoh Masukan 3 dari Terminal 19

Tabel 10. Contoh Masukan 4 dari Terminal 20

Tabel 11.Contoh Masukan 5 dari Terminal 21

iii

**BAB I**

**DESKRIPSI MASALAH**

Kurva Bézier adalah kurva halus yang sering digunakan dalam desain grafis, animasi, dan manufaktur. Kurva ini dibuat dengan menghubungkan beberapa titik kontrol, yang menentukan bentuk dan arah kurva. Cara membuatnya cukup mudah, yaitu dengan menentukan titik-titik kontrol dan menghubungkannya dengan kurva. Kurva Bézier memiliki banyak kegunaan dalam kehidupan nyata, seperti *pen tool*, animasi yang halus dan realistis, membuat desain produk yang kompleks dan presisi, dan membuat font yang indah dan unik. Keuntungan menggunakan kurva Bézier adalah kurva ini mudah diubah dan dimanipulasi, sehingga dapat menghasilkan desain yang presisi dan sesuai dengan kebutuhan.

**A black background with a black square

Description automatically generated with medium confidence**

**Gambar 1.** Kurva Bézier Kubik

*Sumber:* *https://id.wikipedia.org/wiki/Kurva\_B%C3%A9zier*

Sebuah kurva Bézier didefinisikan oleh satu set titik kontrol P0 sampai Pn, dengan n disebut orde (n = 1 untuk linier, n = 2 untuk kuadrat, dan seterusnya). Titik kontrol pertama dan terakhir selalu menjadi ujung dari kurva, tetapi titik kontrol antara (jika ada) umumnya tidak terletak pada kurva. Pada gambar 1 diatas, titik kontrol pertama adalah P0, sedangkan titik kontrol terakhir adalah P3. Titik kontrol P1 dan P2 disebut sebagai titik kontrol antara yang tidak terletak dalam kurva yang terbentuk.

Sebuah kurva Bézier berderajat n, dapat didefiniskan melalui persamaan berparameter sebagai berikut:

dimana : B(t) : Koordinat titik kurva Bézier pada saat t

dimana : P*i* : Titik kontrol ke-*i*.

dimana : n : Derajat kurva Bézier

dimana : : Koefisien binomial/kombinasi (*n*C*i*)

Selain melalui persamaan di atas, sebuah kurva Bézier berderajat n juga dapat dibentuk dengan memanfaatkan algoritma titik tengah berbasis *divide and conquer*.

**BAB II**

**LANGKAH PEMECAHAN MASALAH**

* 1. **Pembuatan Kurva Bézier dengan Pendekatan *Brute Force***

Pembuatan kurva Bézier menggunakan pendekatan *brute force*, dapat dilakukan dengan mencari satu-persatu titik-titik yang akan membentuk kurva Bézier menggunakan persamaan yang telah disebutkan pada bab sebelumnya, yaitu:

Jika misal ingin dihasilkan kurva Bézier dengan banyak titik sebanyak *k* (termasuk titik kontrol pertama dan titik kontrol terakhir), maka dapat dilakukan dengan langkah-langkah sebagai berikut:

1. Hitung jarak antar titik dalam *t* sebagai *dt*, dengan:
2. Lakukan iterasi sebanyak *k* kali, dimulai dengan nilai *t* = 0 pada saat iterasi pertama.
3. Masukkan nilai *t* tersebut ke dalam persamaan di atas pada setiap iterasinya.
4. Simpan titik yang diperoleh ke dalam sebuah daftar/*list*.
5. Tambahkan nilai *t* sebesar *dt* di akhir setiap iterasi.
6. Pada iterasi yang terakhir, yaitu saat jumlah perulangan telah dilakukan sebanyak *k* kali, nilai *t* akan sama dengan 1.
7. Setelah semua titik dihasilkan, hubungkan setiap titik yang berdekatan dengan sebuah garis, maka terbentuklah kurve Bézier berderajat *n*.
   1. **Pembuatan Kurva Bézier dengan Pendekatan *Divide and Conquer***

Selain menggunakan metode *brute force*, pembuatan kurve Bézier juga dapat dilakukan dengan pendekatan *divide and conquer*, yaitu menggunakan algoritma titik tengah. Pada metode ini, pada setiap langkahnya akan dicari himpunan titik tengah diantara dua titik yang saling berdekatan (pada gambar di bawah, ditunjukkan oleh dua titik yang terhubung oleh sebuah garis secara langsung). Untuk himpunan titik yang baru, proses diulangi kembali berkali-kali hingga pada suatu waktu hanya satu titik yang dihasilkan. Satu titik terakhir inilah yang menjadi titik yang akan membentuk kurva Bézier. Adapun ilustrasinya diberikan oleh gambar dibawah ini.

A graph of a graph

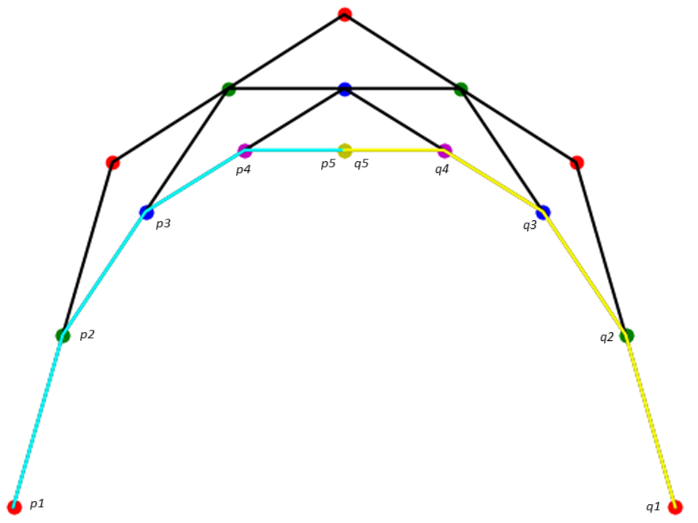
Description automatically generated with medium confidence

**Gambar 2.** Himpunan Titik Tengah (orde = 4)

Pada awalnya, terdapat himpunan titik merah sebagai titik kontrol untuk pembentukan kurva Bézier. Dari himpunan titik merah tersebut, dilakukan pencarian titik tengah antara dua titik yang berdekatan, dihasilkanlah himpunan titik-titik hijau. Ulangi proses untuk himpunan titik hijau untuk membentuk himpunan titik biru, dan seterusnya. Pada satu waktu, hanya akan dihasilkan satu titik tengah, yang pada gambar di atas ditunjukkan oleh titik berwarna kuning. Titik berwarna kuning tersebutlah yang akan menjadi titik pembentuk kurva Bézier.

Melalui cara di atas, baru satu titik yang dihasilkan. Hal ini tentu tidak cukup untuk menghasilkan kurva Bézier yang cukup mulus. Diperlukan pencarian titik-titik yang lain agar dihasilkan kurva Bézier yang mulus. Di sinilah algoritma *divide and conquer* berperan.

Algoritma *divide and conquer* digunakan untuk mencari titik-titik kurva Bézier selanjutnya. Adapun titik kontrol yang digunakan tidak lagi himpunan titik kontrol sebelumnya, namun menggunakan titik kontrol yang baru. Adapun ilustrasi untuk mendapatkan titik kontrol yang baru diberikan oleh gambar di bawah.



**Gambar 3.** Himpunan Titik Kontrol yang Baru

Dari gambar di atas, titik yang berada pada garis berwarna biru dan kuning merupakan himpunan titik-titik yang akan menjadi titik kontrol untuk mendapatkan titik kurva Bézier selanjutnya. Disinilah proses *divide* dilakukan, yaitu dengan membagi himpunan titik tersebut menjadi dua himpunan berbeda, yaitu himpunan titik kiri (yaitu titik-titik yang berada pada garis berwarna biru) dan himpunan titik kanan (yaitu titik-titik yang berada pada garis berwarna kunng). Masing-masing himpunan titik ini akan menjadi titik kontrol untuk memperoleh titik kurva Bézier selanjutnya, melalui cara yang sama dengan yang dijelaskan pada bagian sebelumnya.

Adapun cara untuk memperoleh himpunan titik-titik tersebut dapat dilihat dari pola yang ditunjukkan pada gambar di atas. Himpunan titik kiri, yang beranggotakan titik *p1*, *p2*, *p3*, *p4*, dan *p5*, merupakan titik pertama dari setiap titik tengah pada proses sebelumnya (*p1* adalah titik pertama dari himpunan titik merah, *p2* merupakan titik pertama dari himpunan titik hijau, dan *p3* adalah titik pertama dari himpunan titik biru, dan seterusnya). Adapun himpunan titik kanan, yang beranggotakan titik *q1*, *q2*, *q3*, *q4*, *q5* diperoleh dengan cara yang hampir mirip dengan himpunan titik kiri, yang hanya berbeda dari titik yang diambil. Pada himpunan titik kanan, titik yang diambil merupakan titik terakhir dari setiap titik tengah pada proses sebelumnya (*q1* adalah titik terakhir dari himpunan titik merah, *q2* merupakan titik terakhir dari himpunan titik hijau, dan *q3* adalah titik terakhir dari himpunan titik biru, dan seterusnya). Ilustrasi ini mengasumsikan bahwa titik kontrol juga merupakan himpunan titik tengah.

Dalam pembuatan titik kontrol yang baru, urutan titik-titik yang dibentuk penting. Titik-titik yang bersebalahan harus tidak bisa ditukar dengan titik lain, karena akan mengubah bentuk kurva Bézier. Berdasarkan gambar di atas, maka urutan titik kontrol untuk himpunan titik kiri adalah *p1* 🡪 *p2* 🡪 *p3* 🡪 *p4* 🡪 *p5*. Urutan ini dapat dicerminkan, asalkan titik yang bersebelahan tetap bersebelahan, yaitu *p5* 🡪 *p4* 🡪 *p3* 🡪 *p2* 🡪 *p1*.

Adapun proses *conquer* dilakukan dengan mencari titik pembentuk kurva Bézier dari himpunan titik kontrol yang diberikan seperti yang telah dijelaskan pada halaman 3.

Proses ini secara keseluruhan dapat dilakukan dengan menggunakan pendekatan secara rekursif. Adapun langkah-langkah lengkapnya yaitu:

1. Diberikan himpunan titik kontrol awal yang akan menjadi titik kontrol untuk membentuk kurva Bézier.
2. Cari titik kurva Bézier untuk himpunan titik kontrol tersebut menggunakan cara yang telah dijelaskan pada halaman 3 (bagian *conquer*).
3. Cari himpunan titik kontrol baru sebagaimana yang dijelaskan pada halaman 4 dari himpunan titik kontrol saat ini. Pada bagian ini, akan diperoleh himpunan titik kontrol bagian kiri dan himpunan titik kontrol bagian kanan (bagian *divide*).
4. Untuk masing-masing himpunan titik kontrol yang baru, selesaikan secara terpisah dengan mengulangi proses 2 dan 3 (bagian *divide* sekaligus *conquer*).
5. Ulangi proses di atas sebanyak yang diinginkan.
6. Gabungkan solusi himpunan titik dengan urutan (bagian *combine*):

(hasil dari himpunan titik kontrol baru bagian kiri) + (titik kurva Bézier dari himpunan titik kontrol saat ini) + (hasil dari himpunan titik kontrol baru bagian kanan).

1. Setelah itu, hubungkan setiap titik-titik yang berdekatan dengan garis untuk membentuk kurva Bézier seacara utuh.

Untuk mengontrol kedalaman proses rekursif yang dilakukan, diperlukan sebuah pencatat, misal *i*, yang nilainya selalu berkurang 1 setiap kali prosedur sebelumnya dipanggil. Jika nilai *i* telah mencapai 1, maka prosedur tidak lagi dipanggil dan langsung mengembalikan titik kurva Bézier saat itu.

**BAB III**

**IMPLEMENTASI PROGRAM**

Program pembuatan kurva Bézier ini diimplementasikan menggunakan bahasa pemrograman python (.py). Adapun untuk proses visualisasi memanfaatkan *library* matplotlib.

* 1. **Implementasi dalam Algoritma *Brute Force***

1. **Kurva Bézier Kuadratik**

A screen shot of a computer code

Description automatically generated

**Gambar 4.** Implementasi Pembuatan Kurva Bézier Kuadratik dalam Bahasa Python dengan Algoritma *Brute Force*

1. **Kurva Bézier N-Titik (N ≥ 1)**

A screen shot of a computer code

Description automatically generated

**Gambar 5.** Implementasi Pembuatan Kurva Bézier N Titik (N ≥ 1) dalam Bahasa Python dengan Algoritma *Brute Force*

* 1. **Implementasi dalam Algoritma *Divide and Conquer***

1. **Kurva Bézier Kuadratik**

A screen shot of a computer program

Description automatically generated

**Gambar 6.** Implementasi Pembuatan Kurva Bézier Kuadratik dalam Bahasa Python dengan Algoritma *Divide and Conquer*

1. **Kurva Bézier N-Titik (N ≥ 1)**

A screenshot of a computer code

Description automatically generated

**Gambar 7.** Implementasi Pembuatan Kurva Bézier N Titik (N ≥ 1) dalam Bahasa Python dengan Algoritma *Divide and Conquer* (1)

A screen shot of a computer program

Description automatically generated

**Gambar 8.** Implementasi Pembuatan Kurva Bézier N Titik (N ≥ 1) dalam Bahasa Python dengan Algoritma *Divide and Conquer* (2)

* 1. **Visualisasi**

A computer screen shot of text

Description automatically generated

**Gambar 9.** Prosedur Untuk Melakukan Visualisasi Pembentukan Kurva Bézier

* 1. **Input/Output**
  2. **Main Program**

**BAB IV**

**UJI COBA**

* 1. **Keterangan Proses Input dan Output**

1. **Input**

|  |  |
| --- | --- |
| **Input** | **Gambar** |
| Meminta pengguna memasukkan metode pembentukan kurva Bézier.  Masukan: 1, 2. | **Gambar 10.** Pilihan Metode |
| Meminta pengguna memasukkan jenis kurva Bézier yang akan dibuat. Bergantung pada pilihan sebelumnya.  Masukan: 1, 2, 3 (jika pilihan sebelumnya 1). 1, 2 (jika pilihan sebelumnya 2). | **Gambar 11.** Jenis Kurva Bézier (Pilihan *Brute Force*)    **Gambar 12.** Jenis Kurva Bézier (Pilihan *Divide and Conquer*) |
| Meminta pengguna memasukkan jumlah iterasi yang diinginkan.  Masukan: int ≥ 0. | **Gambar 13.** Input Jumlah Iterasi |
| Meminta pengguna memasukkan jumlah titik kontrol yang diinginkan. Akan dilewatkan jika seandainya memilih Kurva Bézier Kuadratik, karena jumlah titik pasti 3.  Masukan: int ≥ 1. | **Gambar 14.** Input Jumlah Titik Kontrol |
| Meminta pengguna memasukkan koordinat titik kontrol.  Format: x y  Contoh: 12 34  Contoh: 12.5 34.7  Masukan: float float. | **Gambar 15.** Input Koordinat Titik Kontrol |
| Meminta pengguna memasukkan pilihan untuk menganimasikan langkah per langkah pembentukan kurva.  Masukan: y/Y, n/N. | **Gambar 16.** Input Pilihan Penganimasian |
| Meminta pengguna memasukkan jeda antar animasi/*frame.* Akan dilewatkan jika sebelumnya memilih n/N.  Masukan: float > 0. | **Gambar 17.** Input Jeda Antar-*Frame* |

**Tabel 1.** Keterangan Input

1. **Output**

|  |  |
| --- | --- |
| **Output** | **Gambar** |
| Daftar titik kurva Bézier yang dihasilkan (contoh dengan jumlah iterasi 3, jumlah titik kontrol 3). | **Gambar 18.** Keluaran Daftar Titik Kurva Bézier |
| Waktu yang dibutuhkan untuk mencari semua titik kurva Bézier (contoh dengan jumlah iterasi 3, jumlah titik kontrol 3). | **Gambar 19.** Keluaran Lama Waktu Eksekusi |
| Menanyakan pengguna apakah hasil kurva Bézier ingin disimpan ke dalam file (ekstensi .png).  Masukan: y/Y, n/N. | **Gambar 20.** Input Menanyakan Apakah Hasil ingin Disimpan ke File Eksternal |
| Meminta pengguna memasukkan nama file penyimpanan hasil kurva Bézier. Akan dilewatkan jika sebelumnya memilih n/N. | **Gambar 21.** Input Nama File |
| Lokasi penyimpanan berada pada folder test.  Keterangan:   * <nama file> Plot.png : File hasil visualisasi kurva Bézier. * <nama file> Titik Bézier.txt : File tempat menyimpan koordinat titik kurva Bézier yang dihasilkan. * <nama file> Titik Kontrol: File tempat menyimpan titik kontrol yang digunakan. | **Gambar 22.** Lokasi Penyimpanan dan File Hasil Penyimpanan |

**Tabel 2.** Keterangan Output

* 1. **Kurva Bézier Kuadratik**

|  |  |  |
| --- | --- | --- |
| **Keterangan Uji Coba** | **Hasil *Brute Force*** | **Hasil *Divide and Conquer*** |
| Titik: 3  5 0  5 10  15 5  Iterasi: 20 | **Gambar 23.** Hasil Kurva Bézier    **Gambar 24.** Waktu Eksekusi | **Gambar 25.** Hasil Kurva Bézier    **Gambar 26.** Waktu Eksekusi |
| Titik: 3  5 0  5 10  15 5  Iterasi: 15 | **Gambar 27.** Hasil Kurva Bézier    **Gambar 28.** Waktu Eksekusi | **Gambar 29.** Hasil Kurva Bézier    **Gambar 30.** Waktu Eksekusi |
| Titik: 3  -5 0  5 10  15 5  Iterasi: 15 | **Gambar 31.** Hasil Kurva Bézier    **Gambar 32.** Waktu Eksekusi | **Gambar 33.** Hasil Kurva Bézier    **Gambar 34.** Waktu Eksekusi |
| Titik: 3  2.5 0.7  0.55 10.7  7.75 15.95  Iterasi: 10 | **Gambar 35.** Hasil Kurva Bézier    **Gambar 36.** Waktu Eksekusi | **Gambar 37.** Hasil Kurva Bézier    **Gambar 38.** Waktu Eksekusi |
| Titik: 3  2.5 -0.7  -0.55 10.7  -7.75 -15.95  Iterasi: 5 | **Gambar 39.** Hasil Kurva Bézier    **Gambar 40.** Waktu Eksekusi | **Gambar 41.** Hasil Kurva Bézier    **Gambar 42.** Waktu Eksekusi |
| Titik: 3  -1000 0  0 1000  1000 0  Iterasi: 20 | **Gambar 43.** Hasil Kurva Bézier    **Gambar 44.** Waktu Eksekusi | **Gambar 45.** Hasil Kurva Bézier    **Gambar 46.** Waktu Eksekusi |

**Tabel 3.** Perbandingan Hasil Pembentukan Kurva Bézier Kuadratik Menggunakan Metode *Brute Force* dengan *Divide and Conquer*

* 1. **Kurva Bézier N Titik**

|  |  |  |
| --- | --- | --- |
| **Keterangan Uji Coba** | **Hasil *Brute Force*** | **Hasil *Divide and Conquer*** |
| Titik: 5  -10 0  -12 10  2 10  15 0  25 7  Iterasi: 22  \* Kordinat Kurva Bezier untuk test ini tidak disimpan dikarenakan ukuran file besar (> 100 KB) | **Gambar 47.** Hasil Kurva Bézier    **Gambar 48.** Waktu Eksekusi | **Gambar 49.** Hasil Kurva Bézier    **Gambar 50.** Waktu Eksekusi |
| Titik: 13  0 10  5 15  8 12  7 10  4 7  7 0  20 0  26 6  29 13  23 17  16 14  16 11  19 8  Iterasi: 20 | **Gambar 51.** Hasil Kurva Bézier    **Gambar 52.** Waktu Eksekusi | **Gambar 53.** Hasil Kurva Bézier    **Gambar 54.** Waktu Eksekusi |
| Titik: 19  0 1  0 0  1 0  2 0  2 1  2 3  0 3  -3 3  -3 0  -3 -5  2 -5  10 -5  10 3  10 16  -3 16  -24 16  -24 -5  -24 -42  10 -42  Iterasi: 17 | **Gambar 55.** Hasil Kurva Bézier    **Gambar 56.** Waktu Eksekusi | **Gambar 57.** Hasil Kurva Bézier    **Gambar 58.** Waktu Eksekusi |
| Titik: 36  5 0  5 20  20 20  20 17  10 17  10 14  16 14  16 11  10 11  10 0  -5 0  -5 15  -10 10  -15 15  -15 0  -20 0  -25 0  -25 7  -30 7  -30 0  -35 0  -35 20  -30 20  -30 13  -25 13  -25 20  -20 20  -20 0  -20 20  -15 20  -10 16  -5 20  0 20  0 0  0 20  5 20  Iterasi: 20 | **Gambar 59.** Hasil Kurva Bézier    **Gambar 60.** Waktu Eksekusi | **Gambar 61.** Hasil Kurva Bézier    **Gambar 62.** Waktu Eksekusi |
| Titik: 1  1000 1000  Iterasi: 20 | **Gambar 63.** Hasil Kurva Bézier    **Gambar 64.** Waktu Eksekusi | **Gambar 65.** Hasil Kurva Bézier    **Gambar 66.** Waktu Eksekusi |
| Titik: 4  -10 0  -12 10  2 10  15 0  Iterasi: 20 | **Gambar 67.** Hasil Kurva Bézier    **Gambar 68.** Waktu Eksekusi | **Gambar 69.** Hasil Kurva Bézier    **Gambar 70.** Waktu Eksekusi |

**Tabel 4.** Perbandingan Hasil Pembentukan Kurva Bézier N Titik Menggunakan Metode *Brute Force* dengan *Divide and Conquer*

**DAFTAR PUSTAKA**

<https://cyberpunk-hacker.com/> (Diakses pada 8 Februari 2024).

**LAMPIRAN**

1. **Tampilan Awal Program**

Link : <https://github.com/Agil0975/Tucil1_13522006>

1. **Tabel Checkpoint Program**

|  |  |  |
| --- | --- | --- |
| **Poin** | **Ya** | **Tidak** |
| 1. Program berhasil dikompilasi tanpa kesalahan | ✓ |  |
| 2. Program berhasil dijalankan | ✓ |  |
| 3. Program dapat membaca masukan berkas .txt | ✓ |  |
| 4. Program dapat menghasilkan masukan secara acak | ✓ |  |
| 5. Solusi yang diberikan program optimal | ✓ |  |
| 6. Program dapat menyimpan solusi dalam berkas .txt | ✓ |  |
| 7. Program memiliki GUI |  | ✓ |