**LAPORAN TUGAS KECIL 2**

Semester II tahun 2023/2024

Membangun Kurva Bézier dengan Algoritma Titik Tengah berbasis Divide and Conquer



NIM 13522112

Nama : Dimas Bagoes Hendrianto Kelas : K02

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

**2024**

DAFTAR ISI

DAFTAR ISI 1

BAB I Deskripsi Masalah 2

BAB II Teori Singkat 5

BAB III Implementasi 6

BAB IV Eksperimen 8

Lampiran 10

# BAB I

**Dekripsi Masalah**

**A black background with a black square

Description automatically generated with medium confidence**

**Gambar 1.1.** Kurva Bézier Kubik

(Sumber: <https://id.wikipedia.org/wiki/Kurva_B%C3%A9zier>)

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.

Sebuah kurva Bézier didefinisikan oleh satu set titik kontrol *P0*sampai *Pn*,  dengan *n* disebut order (*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.

Mengulas lebih jauh mengenai bagaimana sebuah kurva Bézier bisa terbentuk, misalkan diberikan dua buah titik *P0* dan *P1* yang menjadi titik kontrol, maka kurva Bézier yang terbentuk adalah sebuah garis lurus antara dua titik. Kurva ini disebut dengan **kurva Bézier linier**. Misalkan terdapat sebuah titik *Q0* yang berada pada garis yang dibentuk oleh *P0* dan *P1*, maka posisinya dapat dinyatakan dengan persamaan parametrik berikut.



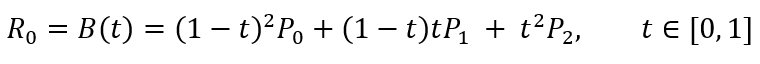
dengan *t* dalam fungsi kurva Bézier linier menggambarkan seberapa jauh *B*(*t*) dari *P0* ke *P1*. Misalnya ketika *t* = 0.25, maka *B*(*t*) adalah seperempat jalan dari titik *P0* ke *P1*. sehingga seluruh rentang variasi nilai *t* dari 0 hingga 1 akan membuat persamaan *B*(*t*) membentuk sebuah garis lurus dari *P0*  ke *P1*.

Misalkan selain dua titik sebelumnya ditambahkan sebuah titik baru, sebut saja *P2*, dengan *P0* dan *P2* sebagai titik kontrol awal dan akhir, dan *P1* menjadi titik kontrol antara. Dengan menyatakan titik *Q1* terletak diantara garis yang menghubungkan *P1* dan *P2*, dan membentuk kurva Bézier linier yang berbeda dengan kurva letak *Q0* berada, maka dapat dinyatakan sebuah titik baru, *R0* yang berada diantara garis yang menghubungkan *Q0* dan *Q1* yang bergerak membentuk **kurva Bézier kuadratik** terhadap titik *P0* dan *P2*. Berikut adalah uraian persamaannya.

A math equations on a white background

Description automatically generated

dengan melakukan substitusi nilai *Q0* dan *Q1*, maka diperoleh persamaan sebagai berikut.



Berikut adalah ilustrasi dari kasus diatas.

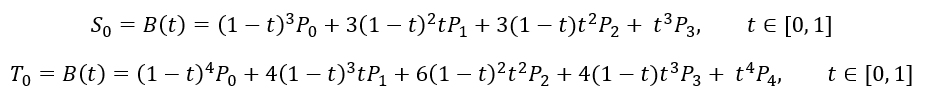
A graphic of a curved object

Description automatically generated with medium confidence

**Gambar 1.2.** Pembentukan Kurva Bézier Kuadratik.

(Sumber: <https://simonhalliday.com/2017/02/15/quadratic-bezier-curve-demo/>)

Proses ini dapat juga diaplikasikan untuk jumlah titik yang lebih dari tiga, misalnya empat titik akan menghasilkan **kurva Bézier kubik**, lima titik akan menghasilkan **kurva Bézier kuartik**, dan seterusnya. Berikut adalah persamaan kurva Bézier kubik dan kuartik dengan menggunakan prosedur yang sama dengan yang sebelumnya.



Tentu saja persamaan yang terbentuk sangat panjang dan akan semakin rumit seiring bertambahnya titik. Oleh sebab itu, dalam rangka melakukan efisiensi pembuatan kurva Bézier yang sangat berguna ini, maka Anda diminta untuk mengimplementasikan **pembuatan kurva Bézier** dengan algoritma titik tengah berbasis ***divide and conquer***

# BAB II

**Teori Singkat**

Divide didefiniksan menjadi membagi persoalan menjadi beberapa upa-persoalan yang memiliki kemiripan dengan persoalan semula namun berukuran lebih kecil (idealnya setiap upa-persoalan berukuran hampir sama). Conquer didefiniksan menjadi menyelesaikan (solve) masing-masing upa-persoalan ( secara langsung jika sudah berukuran kecil atau secara rekursif jika masih berukuran besar). Combine didefiniksan menjadi mengabungkan solusi masing-masing upa-persoalan sehingga membentuk solusi persoalan semula.

A diagram of a structure

Description automatically generated

**Gambar 2.1.** Diagram Proses Divide and Conquer

(Sumber: <https://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2020-2021/Algoritma-Divide-and-Conquer-%282021%29-Bagian1.pdf>)

Obyek persoalan yang dibagi : masukan (input) atau instances persoalan yang berukuran n seperti tabel (larik), matriks, eksponen, polinom, dll, bergantung persoalannya. Tiap-tiap upa-persoalan memiliki karakteristik yang sama (*the same type*) dengan karakteristik persoalan semula namun berukuran lebih kecil sehingga metode Divide and Conquer lebih natural diungkapkan dalam skema rekursif.

# BAB III

**Implementasi**

Algoritma Brute Force yang saya gunakan (spesifikasi wajib dan bonus):

A screen shot of a computer program

Description automatically generated

1. Membuat fungsi generateMid terlebih dahulu sebagai default “titik tengah” atau “titik antara” yang merupakan titik tengah di antara dua buah titik kontrol.
2. Pada fungsi generateBezierBF akan dilakukan pembuatan titik-titik antara yang titik kontrolnya disimpan dalam dua buah array yang pertama untuk absis dan yang kedua untuk oordinat.
3. Akan ditambahkan terlebih dahulu titik kontrol pertama basis sebelum dilakukan pengulangan untuk titik kontrol lain
4. Untuk setiap 3 titik kontrol yang berurutan di dalam array akan dihasilkan 3 titik antara yaitu antara tengah kiri, antara tengah tengah, dan antara tengah kanan. Antara tengah tengah dan antara tengah kanan yang kemudian akan ditambahkan ke array hasil karena tidak ingin adanya pengulangan titik-titik yang sama pada array hasil nanti. Pengecualian untuk pengulangan yang pertama akan ditambahkan antara tengah kiri akan ditambahkan ke dalam array hasil
5. Pada pengulangan yang terakhir akan ditambahkan titik kotrol terakhir karena tidak ingin kehilangan titik kontrol ini.

Algoritma Divide and Conquer yang saya gunakan (spesisfikasi wajib) :

A screen shot of a computer program

Description automatically generated

1. Membuat fungsi midPoint terlebih dahulu sebagai default “titik tengah” atau “titik antara” yang merupakan titik tengah di antara dua buah titik kontrol
2. Pada fungsi generateBezier akan dilakukan pembuatan titik-titik antara yang titik kontrolnya ditambahkan ke dua buah array yang pertama untuk absis dan yang kedua untuk oordinat.
3. Dengan adanya iterasi yang menjadi salah satu parameter pada fungsi berguna untuk mengetahui kapan pengulangan akan berhenti.
4. Pada dasarnya akan dihasilkan 3 titik antara yaitu antara tengah kiri, antara tengah tengah, dan antara tengah kanan dari 3 titik kontrol input. Tidak lupa untuk menambahkan nilai pada parameter iterasi sehingga perulangan yang terjadi akan terhingga dan mendapat hasil yang sesuai. Kemudian akan dilakukan DIVIDE yang pertama adalah titik kontrol 1, antara tengah kiri, serta antara tengah tengah, DIVIDE yang kedua adalah antara tengah tengah, antara tengah kanan, serta titk kontrol 3. Titik kontrol 2 tidak lagi digunakan karena telah “tergantikan” oleh titik-titik antara tengah yang menjadi “titik kontrol” baru untuk pengulangan selanjutnya.
5. Pada proses COMBINE akan ditambahkan titik antara tengah tengah ke dalam array yang telah ada.
6. Hal ini akan berulang sesuai parameter iterasi yang dimasukkan oleh pengguna.

.

# BAB IV

**Eksperimen**

|  |  |
| --- | --- |
| Divide And Conquer | Bruteforce |
|  |  |
| **T ms** | T ms |
|  |  |
| **T ms** | T ms |
|  |  |
| **T ms** | T ms |
|  |  |
| **T ms** | T ms |
|  |  |
| **T ms** | T ms |
|  |  |
| **T ms** | T ms |

**Kesimpulan**

Dari pengujian yang telah dilakukan bisa dilihat kalau waktu yang diperlukan untuk memperoleh kurva Bezier dengan algoritma Devide and Conquer dengan angka yang cukup konsisten yaitu 2 kai lebih cepat dua kali lipat apabila dibandingkan dengan algoritma Brute Force

# Lampiran

**A screenshot of a computer

Description automatically generated**

**Link**

https://github.com/dimasb1954/Tucil2\_13522112.git