

Tugas Kecil 2 IF2211 Strategi Algoritma  
Semester II tahun 2023/2024

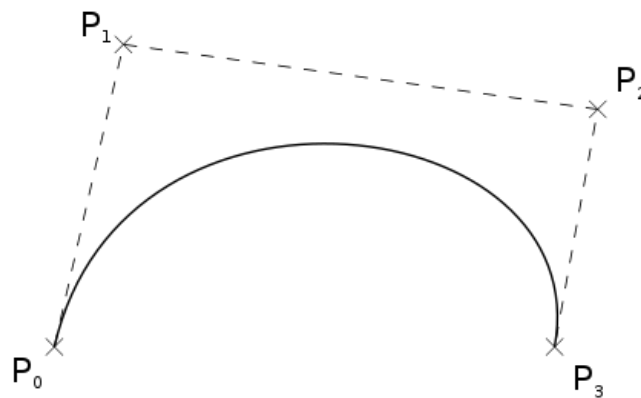
**Membangun Kurva Bézier dengan Algoritma Titik Tengah berbasis *Divide and Conquer***

**Batas pengumpulan** : Hari Selasa, 19 Maret 2024 pukul 12.59 WIB (sebelum kelas kuliah)

**Arsip pengumpulan** :

- *Source* program yang dapat dijalankan disertai README
- Laporan (*soft copy*)

**Deskripsi Tugas**



**Gambar 1.** Kurva Bézier Kubik

(Sumber: [https://id.wikipedia.org/wiki/Kurva\\_B%C3%A9zier](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  $P_0$  sampai  $P_n$ , 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  $P_0$ , sedangkan titik kontrol terakhir adalah  $P_3$ . Titik kontrol  $P_1$  dan  $P_2$  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  $P_0$  dan  $P_1$  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  $Q_0$  yang berada pada garis yang dibentuk oleh  $P_0$  dan  $P_1$ , maka posisinya dapat dinyatakan dengan persamaan parametrik berikut.

$$Q_0 = B(t) = (1 - t)P_0 + tP_1, \quad t \in [0, 1]$$

dengan  $t$  dalam fungsi kurva Bézier linier menggambarkan seberapa jauh  $B(t)$  dari  $P_0$  ke  $P_1$ . Misalnya ketika  $t = 0.25$ , maka  $B(t)$  adalah seperempat jalan dari titik  $P_0$  ke  $P_1$ . sehingga seluruh rentang variasi nilai  $t$  dari 0 hingga 1 akan membuat persamaan  $B(t)$  membentuk sebuah garis lurus dari  $P_0$  ke  $P_1$ .

Misalkan selain dua titik sebelumnya ditambahkan sebuah titik baru, sebut saja  $P_2$ , dengan  $P_0$  dan  $P_2$  sebagai titik kontrol awal dan akhir, dan  $P_1$  menjadi titik kontrol antara. Dengan menyatakan titik  $Q_1$  terletak diantara garis yang menghubungkan  $P_1$  dan  $P_2$ , dan membentuk kurva Bézier linier yang berbeda dengan kurva letak  $Q_0$  berada, maka dapat dinyatakan sebuah titik baru,  $R_0$  yang berada diantara garis yang menghubungkan  $Q_0$  dan  $Q_1$  yang bergerak membentuk **kurva Bézier kuadrat** terhadap titik  $P_0$  dan  $P_2$ . Berikut adalah uraian persamaannya.

$$Q_0 = B(t) = (1 - t)P_0 + tP_1, \quad t \in [0, 1]$$

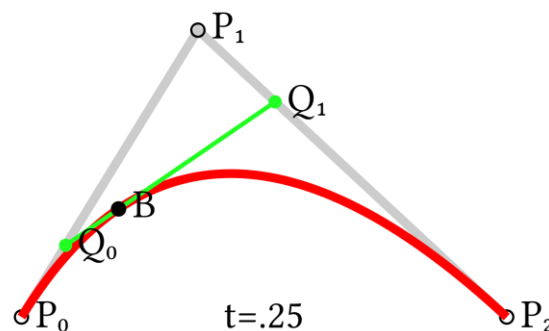
$$Q_1 = B(t) = (1 - t)P_1 + tP_2, \quad t \in [0, 1]$$

$$R_0 = B(t) = (1 - t)Q_0 + tQ_1, \quad t \in [0, 1]$$

dengan melakukan substitusi nilai  $Q_0$  dan  $Q_1$ , maka diperoleh persamaan sebagai berikut.

$$R_0 = B(t) = (1 - t)^2P_0 + (1 - t)tP_1 + t^2P_2, \quad t \in [0, 1]$$

Berikut adalah ilustrasi dari kasus diatas.



**Gambar 2.** Pembentukan Kurva Bézier Kuadrat.

(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.

$$S_0 = B(t) = (1 - t)^3 P_0 + 3(1 - t)^2 t P_1 + 3(1 - t) t^2 P_2 + t^3 P_3, \quad t \in [0, 1]$$

$$T_0 = B(t) = (1 - t)^4 P_0 + 4(1 - t)^3 t P_1 + 6(1 - t)^2 t^2 P_2 + 4(1 - t) t^3 P_3 + t^4 P_4, \quad t \in [0, 1]$$

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*.

### Ilustrasi kasus

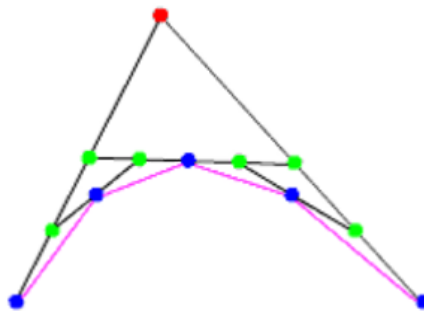
Idenya cukup sederhana, relatif mirip dengan pembahasan sebelumnya, dan dilakukan secara iteratif. Misalkan terdapat tiga buah titik,  $P_0$ ,  $P_1$ , dan  $P_2$ , dengan titik  $P_1$  menjadi titik kontrol antara, maka:

- Buatlah sebuah titik baru  $Q_0$  yang berada di tengah garis yang menghubungkan  $P_0$  dan  $P_1$ , serta titik  $Q_1$  yang berada di tengah garis yang menghubungkan  $P_1$  dan  $P_2$ .
- Hubungkan  $Q_0$  dan  $Q_1$  sehingga terbentuk sebuah garis baru.
- Buatlah sebuah titik baru  $R_0$  yang berada di tengah  $Q_0$  dan  $Q_1$ .
- Buatlah sebuah garis yang menghubungkan  $P_0 - R_0 - P_2$ .

Melalui proses di atas, telah dilakukan 1 buah iterasi dan diperoleh sebuah “kurva” yang belum cukup mulus dengan aproksimasi 3 buah titik. Untuk membuat sebuah kurva yang lebih baik, perlu dilakukan iterasi lanjutan. Berikut adalah prosedurnya.

- Buatlah beberapa titik baru, yaitu  $S_0$  yang berada di tengah  $P_0$  dan  $Q_0$ ,  $S_1$  yang berada di tengah  $Q_0$  dan  $R_0$ ,  $S_2$  yang berada di tengah  $R_0$  dan  $Q_1$ , dan  $S_3$  yang berada di tengah  $Q_1$  dan  $P_2$ .
- Hubungkan  $S_0$  dengan  $S_1$  dan  $S_2$  dengan  $S_3$  sehingga terbentuk garis baru.
- Buatlah dua buah titik baru, yaitu  $T_0$  yang berada di tengah  $S_0$  dan  $S_1$ , serta  $T_1$  yang berada di tengah  $S_2$  dan  $S_3$ .
- Buatlah sebuah garis yang menghubungkan  $P_0 - T_0 - R_0 - T_1 - P_2$ .

Melalui iterasi kedua akan tampak semakin mendekati sebuah kurva, dengan aproksimasi 5 buah titik. Anda dapat membuat visualisasi atau gambaran secara mandiri terkait hal ini sehingga dapat diamati dan diterka dengan jelas bahwa semakin banyak iterasi yang dilakukan, maka akan membentuk sebuah kurva yang tidak lain adalah kurva Bézier.



**Gambar 3.** Hasil pembentukan Kurva Bézier Kuadratik dengan *divide and conquer* setelah iterasi ke-2.

## Spesifikasi Tugas Kecil 2

- Buatlah program sederhana dalam bahasa C++/Python/JavaScript/Go yang dapat membentuk sebuah **kurva Bézier kuadratik** dengan menggunakan algoritma titik tengah berbasis *divide and conquer*.
- Selain implementasi dalam algoritma *divide and conquer*, program **juga diminta untuk diimplementasikan dalam algoritma *brute force***. Solusi ini ditujukan sebagai pembandingan dengan solusi sebelumnya.
- Tugas dapat dikerjakan individu atau berkelompok dengan anggota **maksimal 2 orang** (sangat disarankan). Boleh lintas kelas dan lintas kampus, tetapi **tidak boleh sama** dengan anggota kelompok pada tugas-tugas Strategi Algoritma sebelumnya.
- **Input :**

Format masukan **dibebaskan**, dengan catatan dijelaskan pada README dan laporan. Komponen yang perlu menjadi masukan yaitu.

  1. **Tiga buah pasangan titik.** Sebagai catatan, titik yang paling awal dimasukkan akan menjadi titik awal kurva, begitu juga dengan titik yang paling akhir.
  2. **Jumlah iterasi** yang ingin dilakukan.
- **Output :**

Berikut adalah luaran dari program yang diekspektasikan.

  1. Hasil **kurva Bézier yang terbentuk** pada iterasi terkait. Kakas yang digunakan untuk visualisasi dibebaskan.
  2. **Waktu eksekusi** program pembentukan kurva.
- **Bonus :**

Pastikan sudah mengerjakan spesifikasi wajib sebelum mengerjakan bonus.

  1. Melakukan generalisasi algoritma, ide, serta melakukan implementasinya sehingga program dapat membentuk kurva Bézier kubik, kuartik, dan selanjutnya dengan 4, 5, 6, hingga  $n$  titik kontrol.
  2. Memberikan visualisasi proses pembentukan kurva, sehingga tidak hanya hasil akhirnya saja. Tentu saja proses visualisasinya perlu melibatkan *Graphical User Interface* (GUI) yang kakasnya dibebaskan.
- **Laporan :**

Berkas laporan yang dikumpulkan adalah laporan dalam bentuk PDF yang setidaknya berisi:

  1. Analisis dan implementasi dalam algoritma *brute force* sebagai algoritma pembandingan.
  2. Analisis dan implementasi dalam algoritma *divide and conquer* (berisi deskripsi langkah-langkahnya, **bukan notasi pseudocode**). Perlu untuk diperhatikan bahwa prosedur yang disampaikan pada ilustrasi kasus **belum memuat konsep *divide and conquer***. Anda diminta untuk melakukan analisis dan implementasi *divide and conquer* pada solusi tersebut.
  3. *Source code* program implementasi keduanya dalam bahasa pemrograman yang dipilih (pastikan bahwa program telah dapat dijalankan).

4. Tangkapan layar yang memperlihatkan *input* dan *output* (minimal 6 contoh untuk masing-masing algoritma).
5. Hasil analisis perbandingan solusi *brute force* dengan *divide and conquer*. Analisis dilakukan dalam bentuk paragraf dan minimal memuat analisis kompleksitas algoritma.
6. Penjelasan mengenai implementasi bonus jika mengerjakan.
7. Pranala ke *repository* yang berisi kode program.

- **Struktur Kode :**

Program disimpan dalam repository yang bernama Tucil2\_NIM jika mengerjakan secara individu atau Tucil2\_NIM1\_NIM2 jika dikerjakan berkelompok. Berikut merupakan struktur dari isi repository tersebut:

1. Folder **src** berisi *source code*.
2. Folder **bin** berisi *executable* (jika ada).
3. Folder **test** berisi data uji yang digunakan pada laporan.
4. Folder **doc** berisi laporan tugas kecil (dalam bentuk PDF).
5. **README** yang minimal berisi:
  - a. Deskripsi singkat program yang dibuat.
  - b. *Requirement* program dan instalasi tertentu bila ada.
  - c. Cara mengkompilasi program (bila dikompilasi).
  - d. Cara menjalankan dan menggunakan program.
  - e. Identitas pembuat program.

Silahkan gunakan template [berikut](#), sebagai referensi untuk implementasi struktur dalam README.

- Laporan dikumpulkan hari **Selasa, 19 Maret 2024** pada alamat Google Form berikut paling lambat pukul **12.59** (sebelum kelas kuliah):  
[LINK]
- Adapun pertanyaan terkait tugas kecil ini bisa disampaikan melalui QnA berikut:  
<https://bit.ly/qnastima24>

Perhatikan:

- **Dilarang keras copy paste** program dari internet (AI, repository lain, ataupun teman). Program harus dibuat sendiri, kecurangan akan mengakibatkan nilai tugas menjadi nol.
- Pastikan program **dapat setidaknya dikompilasi** pada *windows* dan *linux*.
- Apabila program **tidak bisa dijalankan** maka tidak akan dinilai oleh asisten.
- Pastikan **repository program sudah public** saat melewati **deadline**.
- Pastikan penamaan **repository** dan **laporan** sesuai dengan **format**.
- Tambahkan *checklist* (centang dengan ✓) berikut pada bagian lampiran laporan Anda.

Poin	Ya	Tidak
------	----	-------

1. Program berhasil dijalankan.		
2. Program dapat melakukan visualisasi kurva Bézier.		
3. Solusi yang diberikan program optimal.		
4. <b>[Bonus]</b> Program dapat membuat kurva untuk $n$ titik kontrol.		
5. <b>[Bonus]</b> Program dapat melakukan visualisasi proses pembuatan kurva.		