LAPORAN TUGAS KECIL 2

IF2211 Strategi Algoritma

Membangun Kurva Bézier dengan Algoritma Titik Tengah berbasis Divide and Conquer

Disusun oleh:

Abdullah Mubarak	13522101
Indraswara Galih Jayanegara	13522119

SEKOLAH TEKNIK ELEKTRO DAN INFORMATIKA

INSTITUT TEKNOLOGI BANDUNG

2024

Daftar Isi

BAB I
DESKRIPSI TUGAS

1.1 Deskripsi Tugas

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.

Tujuan dari tugas kecil kali ini adalah membuat visualisasi kurva bezier dari 3 sampai n titik yang dimasukkan oleh user. Implementasinya harus menggunakan pendekatan divide and conquer, tetapi juga harus mengimplementasikan pendekatan Brute Force sebagai perbandingan untuk pendekatan Divide and Conquer yang telah dibuat.

1.2 Spesifikasi

1) Selain Implementasi Divide and Conquer juga diimplementasikan implementasi dengan pendekatan Brute Force
2) menerima input dengan format dibebaskan
3) Program yang kami buat menggunakan bahasa pemrograman Python

BAB II

PENJELASAN ALGORITMA

1. Bruteforce

   Pada bruteforce langkah yang dilakukan adalah sebagai berikut:

   1. memasukkan titik yang akan dicari kurva beziernya
   2. melakukan pemrosesan melalui persamaan kurva bezier dengan input list titik dan diproses menggunakan persamaan bezier curve dan menghasilkan output adalah list titik

   rumus yang digunakan adalah sebagai berikut:

$$s(t) = \sum_{i=0}^{n} p_i B_{n,i}(t) \quad \text{where}$$

$$B_{n,i}(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

2. Divide and Conquer

   Pada pembuatan kurva bezier dengan pembuatan berdasarkan iterasi memiliki pola sebagai berikut: jumlah titik pada iterasi memiliki pola yang sama tak peduli berapa titik kontrol yang ada, yaitu (2^n) + 1 dengan n adalah nilai iterasi. Langkah-langkah algoritma yang digunakan pada setiap iterasi adalah sebagai berikut:

   1. Mencari titik tengah utama dengan cara mencari titik tengah dari titik-titik kontrol sesuai urutan masukan.
   2. Titik-titik tengah yang dihasilkan disimpan pada array titik tengah.
   3. Menyimpan elemen pertama array titik tengah pada array kiri yang sudah berisi elemen pertama dan terakhir titik masukan.
   4. Menyimpan elemen terakhir array titik tengah pada array kanan yang sudah berisi elemen pertama dan terakhir titik masukan.
   5. Melakukan langkah 1 sampai 4 hingga titik tengah yang dihasilkan berjumlah satu.
   6. Titik tengah yang dihasilkan dimasukkan ke titik-titik hasil.
   7. Melakukan langkah 1 sampai 6 pada setiap sisi hasil baru untuk iterasi berikutnya, dibagi array kiri dan kanan sebagai masukan yang masing-masing mewakili sisi kanan dan kiri.

BAB III

ANALISIS KEEFEKTIFAN

3. 1 Brute Force

Pada Algoritma brute force untuk mencari kurva bezier pada program python yang kami buat membutuhkan kompleksitas sebesar O(n^2) karena pada algoritma ini dilakukan iterasi dari 0 sampai 1 dengan besar sesuai user, dan didalamnya iterasi sebanyak panjang dari list point sehingga kompleksitasnya adalah O(n^2). Space complexity dari brute force adalah O(n) karena hasil dari list akan memiliki panjang n titik.

```python
def bezierCurveNPoint(list_point: list[Point], smoother: float = 0.1) -> list[Point]:
    n: int = len(list_point) - 1
    result: list[Point] = []
    t: float = 0
    while(t <= 1):
        x: float = 0
        y: float = 0
        for k, (point_x, point_y) in enumerate(list_point):
            x += bezierPointCalc(point_x, n, k, t)
            y += bezierPointCalc(point_y, n, k, t)
        result.append((x, y))
        t += smoother
    return result

```

3. 2 Divide and Conquer

Pada algoritma Divide and Conquer ini langkah utamanya adalah mencari titik tengah antara dua titik. Berdasarkan perhitungan, terdapat (1+2+3…x - 1)*k kali jumlah pembuatan titik tengah per iterasi dengan x adalah jumlah titik awal dan k adalah selisih jumlah titik dengan iterasi sebelumnya (jumlah penambahan titik), k bernilai 1 untuk iterasi pertama. Jumlah titik tiap iterasi (n) bisa didapat dengan menggunakan rumus:

$$n = f(iterasi) = \begin{cases} 3, & iterasi = 1 \\ 2 * f(iterasi - 1) - 1, & iterasi > 1 \end{cases}$$

Menggunakan teorema master didapat kompleksitas n adalah 2^n, dan (1+2+3…x-1) dapat diwakili dengan x^2, maka kompleksitas algoritma berdasarkan jumlah penambahan titik tengah adalah O((x^2)*(2^n)) dengan x adalah jumlah titik awal dan n adalah jumlah iterasi.

```python
def makeMiddlePoint(listPoint: list[tuple[float, float]]):
    middlePoint : list[tuple[float, float]] = []
    for i in range(0,len(listPoint) - 1,1):
        middlePoint.append(findMiddlePoint(listPoint[i], listPoint[i+1]))
    return middlePoint

def addListOfPoint(isLeft: bool, iterasi: int, listPoint: list[tuple[float, float]]) -> list[tuple[float, float]]:
    tempPoint : list[tuple[float, float]] = []
    leftSide: list[tuple[float, float]] = []
    rightSide: list[tuple[float, float]] = []
    if(iterasi == 0):
        if (isLeft):
            return [listPoint[0]] + [listPoint[-1]]
        else:
            return [listPoint[-1]]
    else:
        tempList = listPoint
        leftSide = [listPoint[0]]
        while (len(tempList) != 1):
            tempPoint = makeMiddlePoint(tempList)
            leftSide.append(tempPoint[0])
            rightSide.append(tempPoint[-1])
            tempList = tempPoint
        rightSide.reverse()
        rightSide.append(listPoint[-1])
        return addListOfPoint(True, iterasi - 1, leftSide) + addListOfPoint(False, iterasi - 1, rightSide)
```

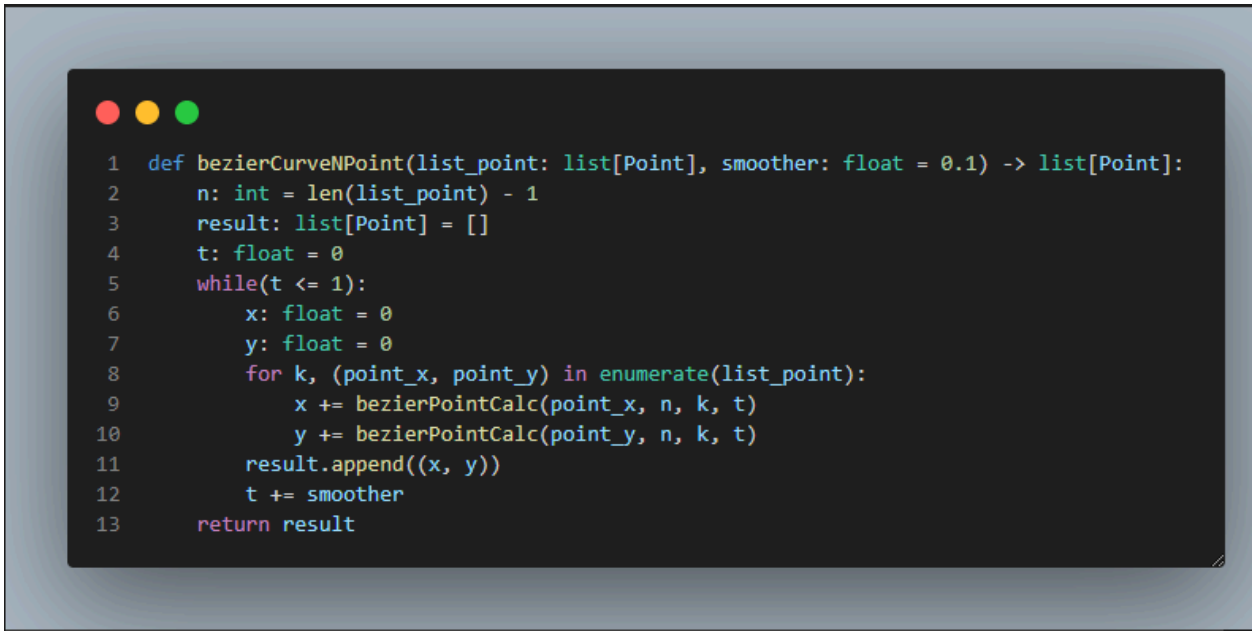
BAB IV

IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi Algoritma Brute Force

Pada bagian algoritma Brute Force berikut adalah beberapa pseudocode dari fungsi-fungsi yang digunakan pada program kami. Diantaranya ada

1. bezierCurveNPoint: Algoritma utama dari pendekatan Brute Force untuk mencari kurva bezier dengan menggunakan persamaan kurva bezier.
2. bezierPointCalc: menghitung point-point dari list_point untuk menghasilkan list_point yang merepresentasikan point dari kurva bezier.
3. binomialCoeff: menghitung kombinasi.
4. fact: menghitung faktorial.
5. bruteforceIterasi: untuk mencari kurva bezier dengan menggunakan iterasi seperti pada algoritma divide and conquer.
6. bezierCurveBruteForce_PointInput: untuk mencari kurva bezier, tapi dengan batasan output yang harus dihasilkan harus berjumlah N titik sesuai dengan input user, misalkan user memasukkan 20 titik, maka titik yang keluar adalah sebanyak 20 titik pula.



```python
def bezierCurveNPoint(list_point: list[Point], smoother: float = 0.1) -> list[Point]:
    n: int = len(list_point) - 1
    result: list[Point] = []
    t: float = 0
    while(t <= 1):
        x: float = 0
        y: float = 0
        for k, (point_x, point_y) in enumerate(list_point):
            x += bezierPointCalc(point_x, n, k, t)
            y += bezierPointCalc(point_y, n, k, t)
        result.append((x, y))
        t += smoother
    return result
```

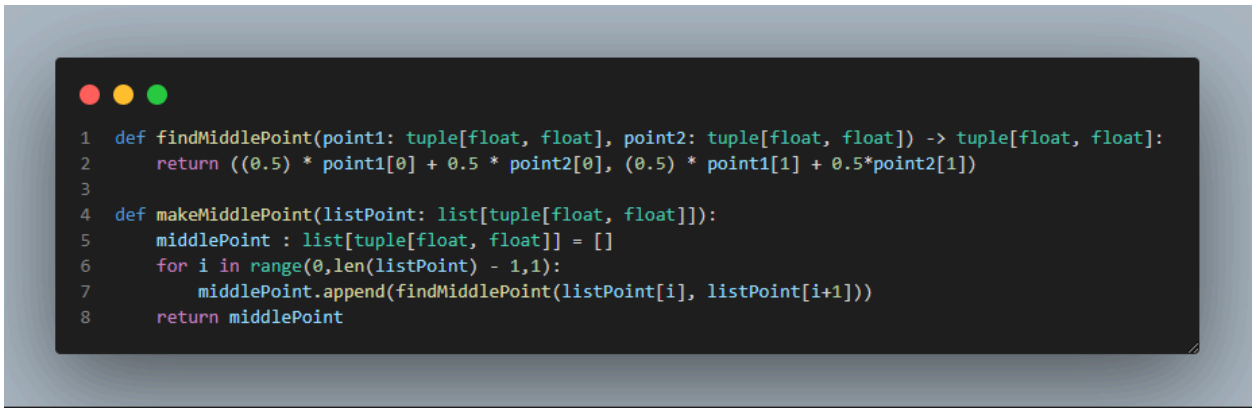
```python
def bezierPointCalc(point_x_or_y: float, n: int, k: int, t: float) -> float:
    return binomialCoeff(n, k) * point_x_or_y * pow((1-t), (n-k)) * pow(t, k)

def binomialCoeff(n: int, k: int) -> float:
    return fact(n) / (fact(k) * fact(n-k))

def fact(n: int) -> int:
    ans: int = 1
    for i in range(1, n + 1):
        ans *= i
    return ans
```

```python
def bruteforceIterasi(list_point: list[Point], iterasi: int):
    temp_list = list(set(list_point))
    if (len(temp_list) < 3):
        return []
    else :
        point_count = pow(2, iterasi) + 1
        return bezierCurveBruteForce_PointInput(list_point, point_count)

def bezierCurveBruteForce_PointInput(list_point: list[Point], result_point_count: int) -> list[Point]:
    t : int = 1/result_point_count
    if (1 % t != 0):
        result_point_count -= 1
    if (1 % t == 0 and t <= 20):
        result_point_count -= 1
    t = 1/result_point_count
    return bezierCurveNPoint(list_point, t)
```

4.2 Algoritma Divide and Conquer

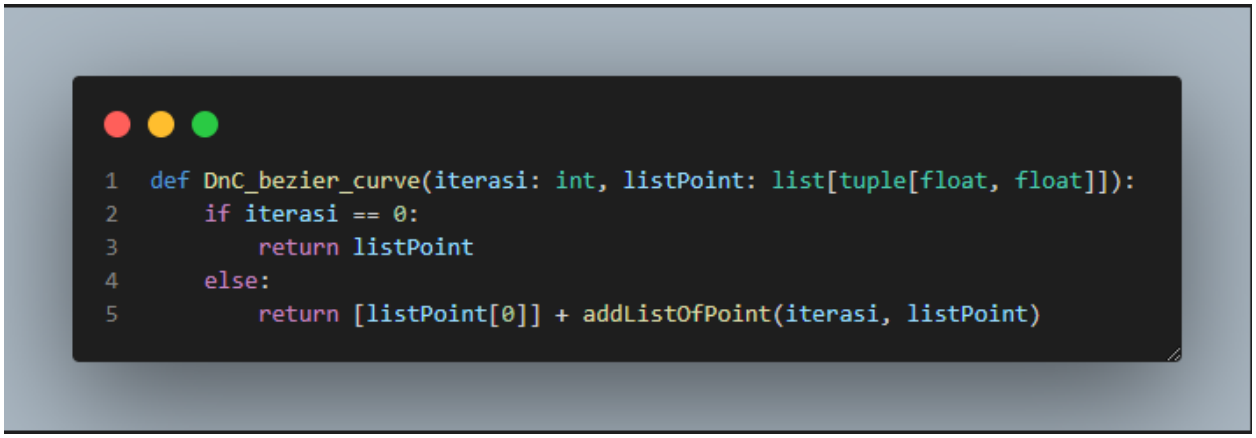
Pada bagian algoritma divide and conquer berikut beberapa pseudocode dari program kami. Diantaranya ada

1. findMiddlePoint: mencari titik tengah dari dua titik
2. makeMiddlePoint: menghasilkan array titik tengah dari setiap 2 elemen dari input list_of_point
3. addListOfPoint: fungsi utama dari pendekatan divide and conquer dari program kami untuk mencari kurva bezier sebanyak n iterasi dengan n sesuai dengan input user.
4. DnC_bezier_curve: fungsi yang memvalidasi input ke fungsi utama sekaligus melengkapi hasil dari fungsi utama



```python
def findMiddlePoint(point1: tuple[float, float], point2: tuple[float, float]) -> tuple[float, float]:
    return ((0.5) * point1[0] + 0.5 * point2[0], (0.5) * point1[1] + 0.5*point2[1])

def makeMiddlePoint(listPoint: list[tuple[float, float]]):
    middlePoint : list[tuple[float, float]] = []
    for i in range(0,len(listPoint) - 1,1):
        middlePoint.append(findMiddlePoint(listPoint[i], listPoint[i+1]))
    return middlePoint
```

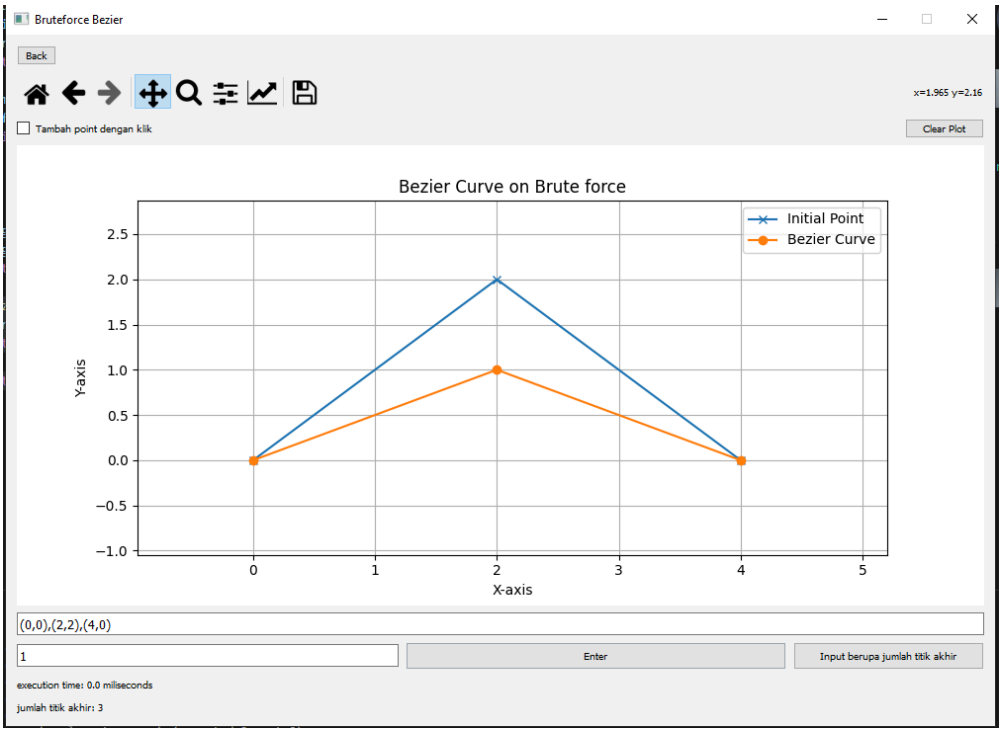
```python
def addListOfPoint(iterasi: int, listPoint: list[tuple[float, float]]) -> list[tuple[float, float]]:
    tempPoint : list[tuple[float, float]] = []
    leftSide: list[tuple[float, float]] = []
    rightSide: list[tuple[float, float]] = []
    if(iterasi == 0):
        return [listPoint[-1]]
    else:
        tempList = listPoint
        leftSide = [listPoint[0]]
        while (len(tempList) != 1):
            tempPoint = makeMiddlePoint(tempList)
            leftSide.append(tempPoint[0])
            rightSide.append(tempPoint[-1])
            tempList = tempPoint
        rightSide.reverse()
        rightSide.append(listPoint[-1])
        return addListOfPoint(iterasi - 1, leftSide) + addListOfPoint(iterasi - 1, rightSide)

```

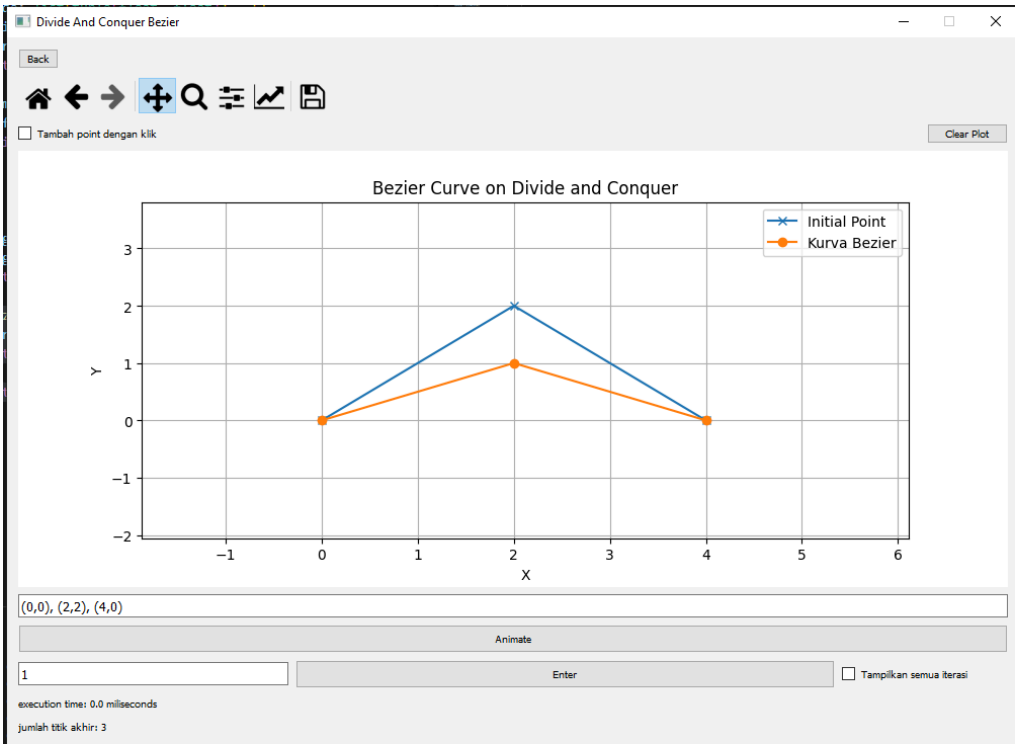


```python
def DnC_bezier_curve(iterasi: int, listPoint: list[tuple[float, float]]):
    if iterasi == 0:
        return listPoint
    else:
        return [listPoint[0]] + addListOfPoint(iterasi, listPoint)
```

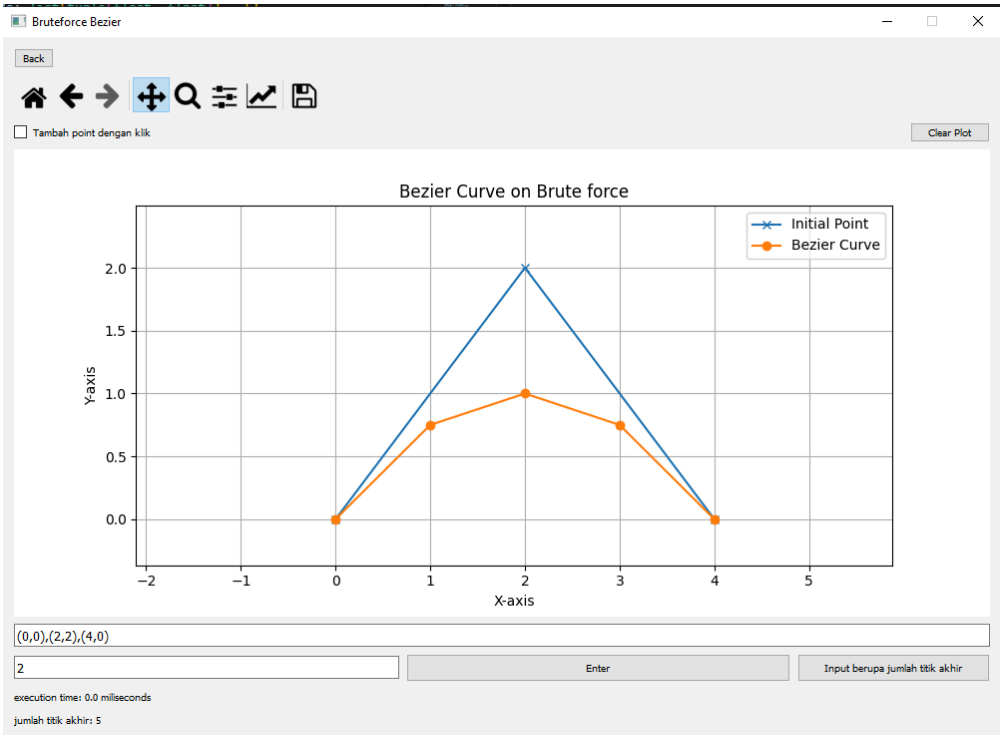
4.3 Pengujian

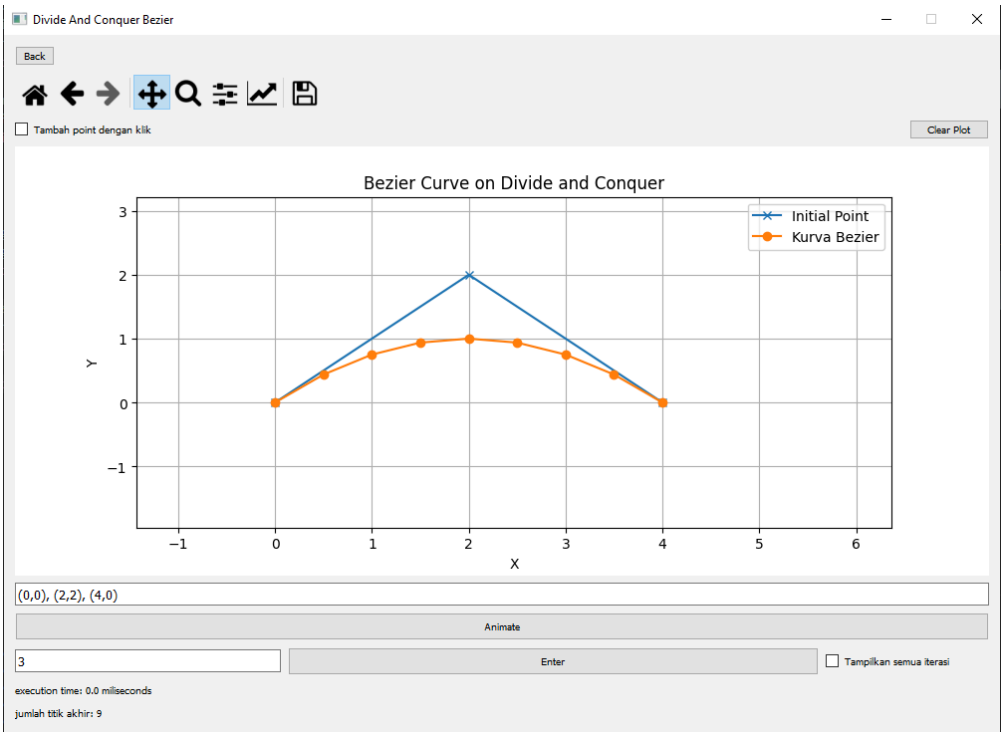
1. Bruteforce
   A. Titik yang diuji: (0,0), (2,2), (4,0)
      a. iterasi 1



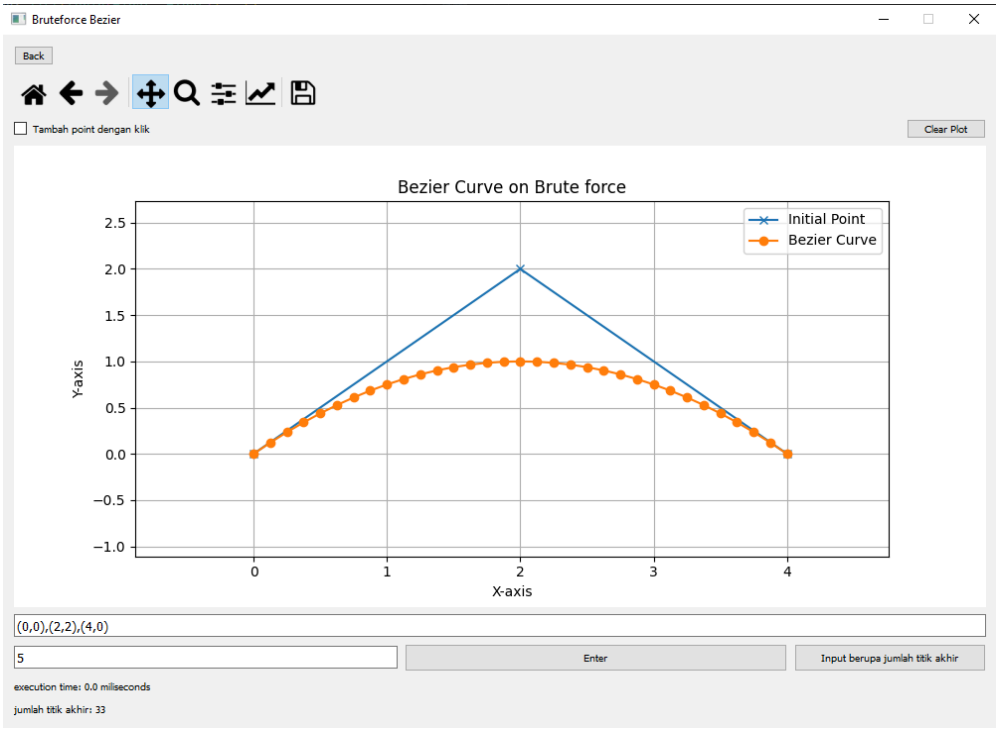


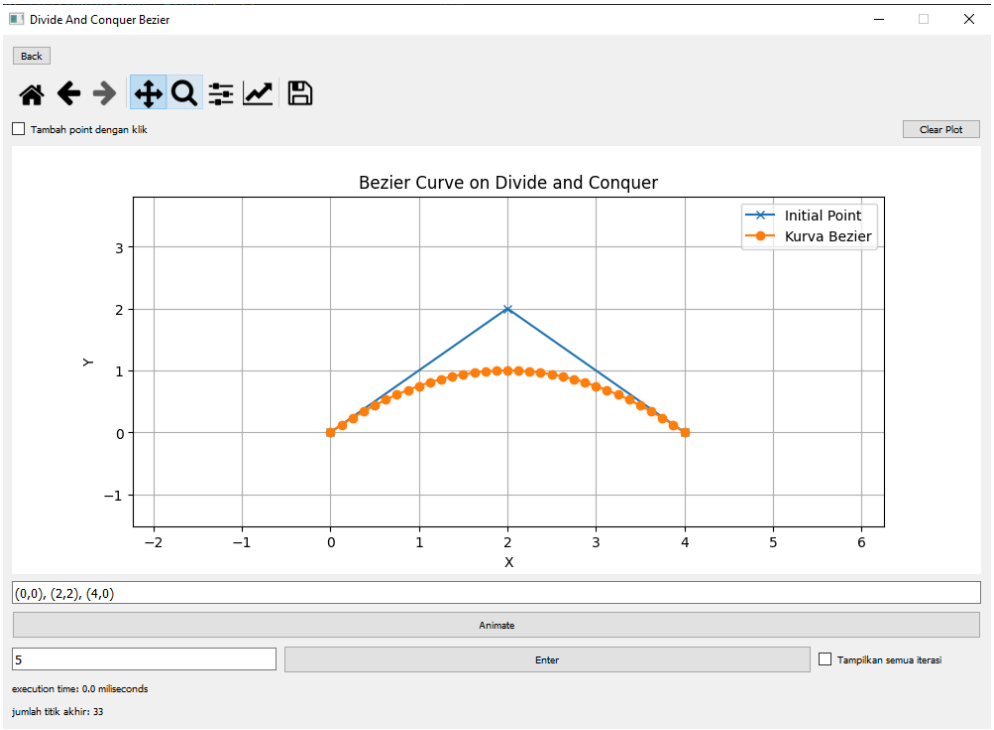
b. iterasi 3



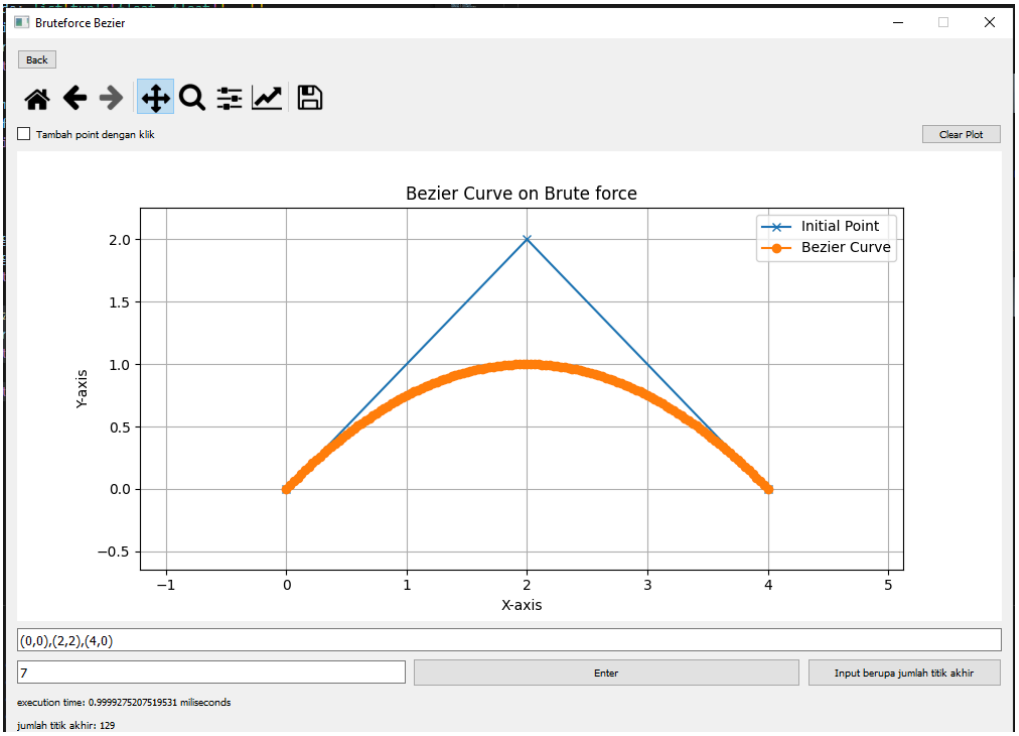


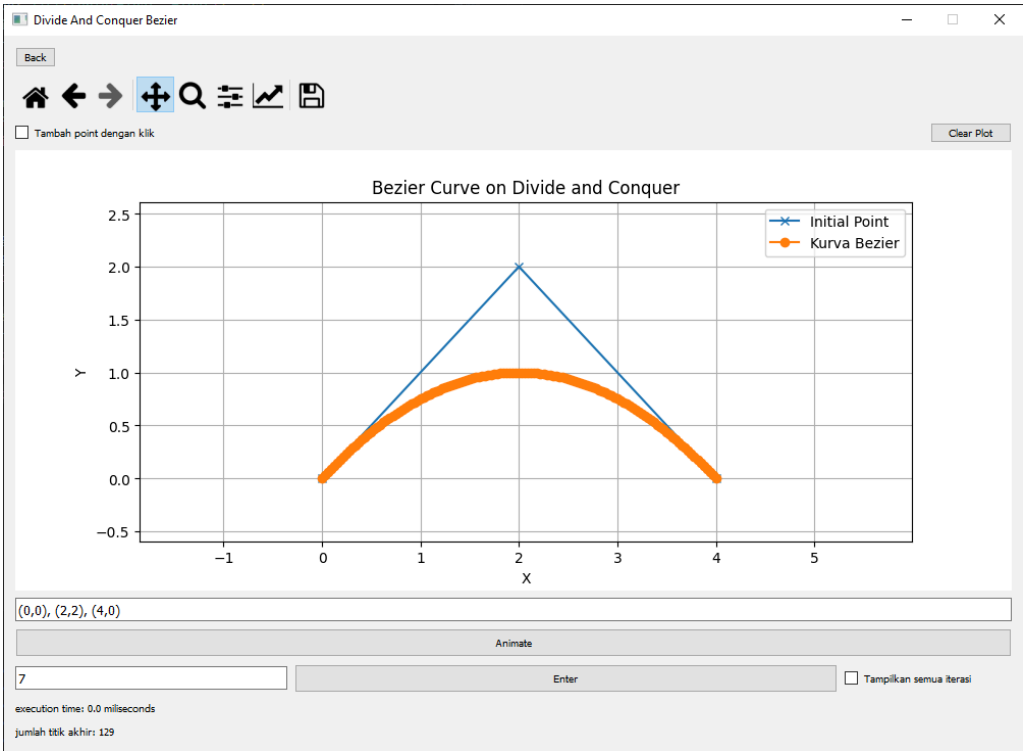
c. iterasi 5


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(0,0),(2,2),(4,0)
5
Enter
Input berupa jumlah titik akhir
execution time: 0.0 milliseconds
jumlah titik akhir: 33

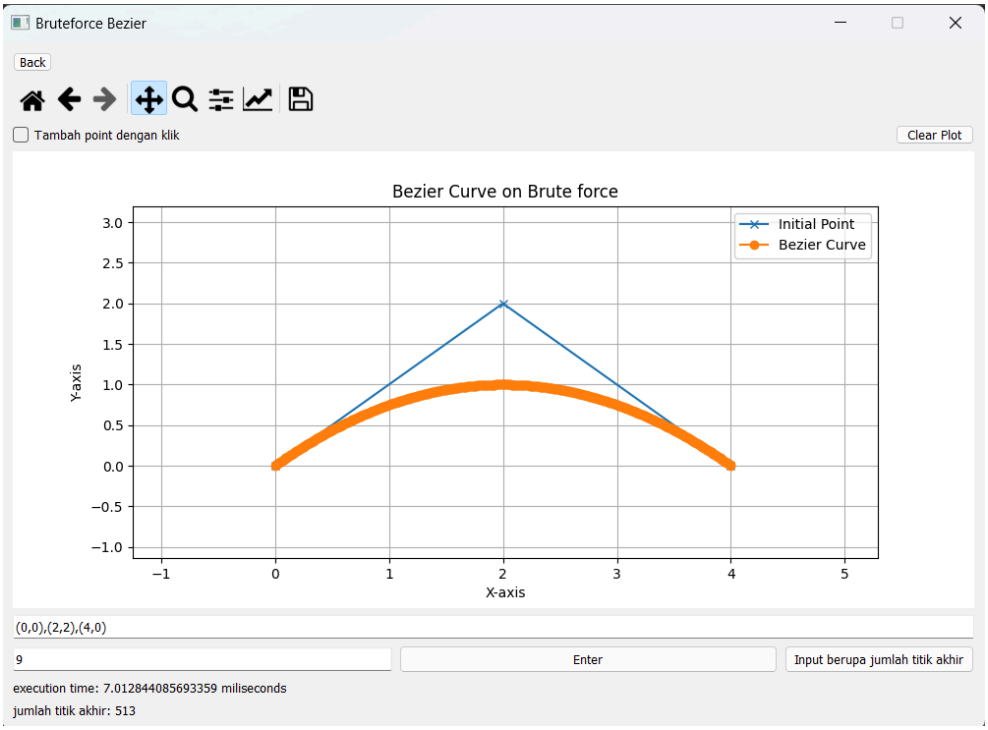

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(0,0), (2,2), (4,0)
Animate
5
Enter
Tampilkan semua iterasi
execution time: 0.0 milliseconds
jumlah titik akhir: 33

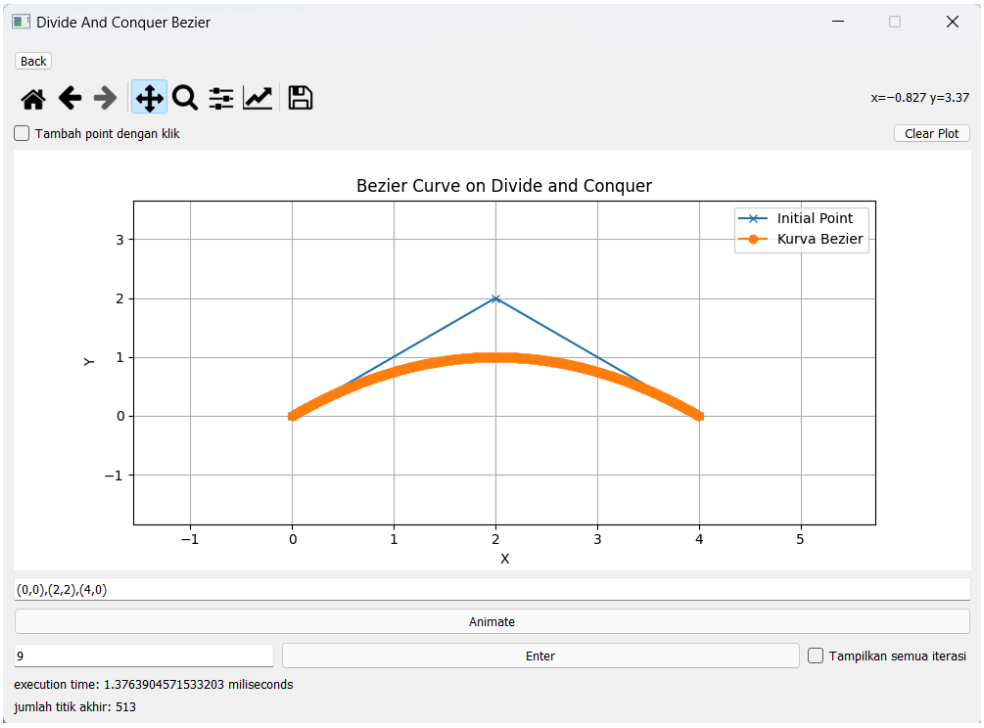
d. iterasi 7


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(0,0),(2,2),(4,0)
7
Enter
Input berupa jumlah titik akhir
execution time: 0.999927520751953 miliseconds
jumlah titik akhir: 129

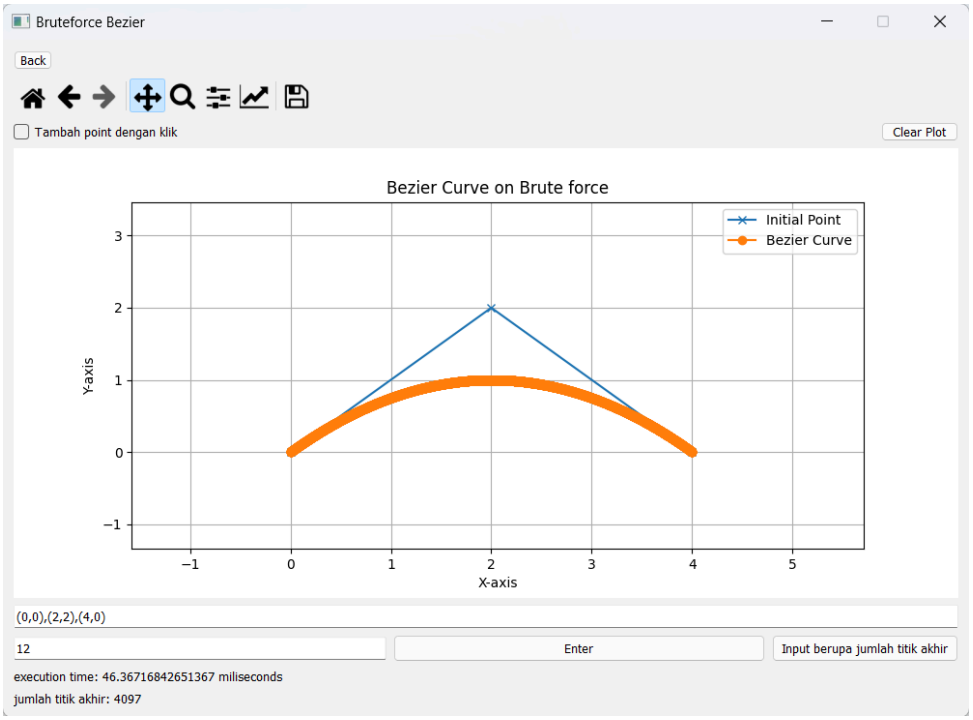

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(0,0), (2,2), (4,0)
Animate
7
Enter
Tampilkan semua iterasi
execution time: 0.0 miliseconds
jumlah titik akhir: 129

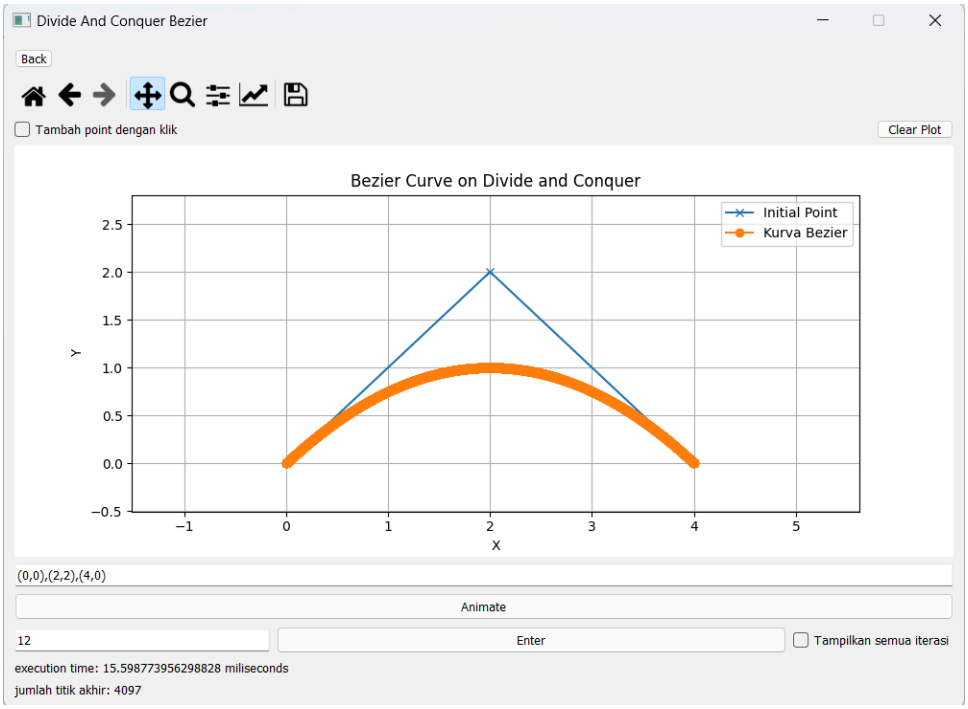
e. iterasi 9


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(0,0),(2,2),(4,0)
9
Enter
Input berupa jumlah titik akhir
execution time: 7.012844085693359 miliseconds
jumlah titik akhir: 513

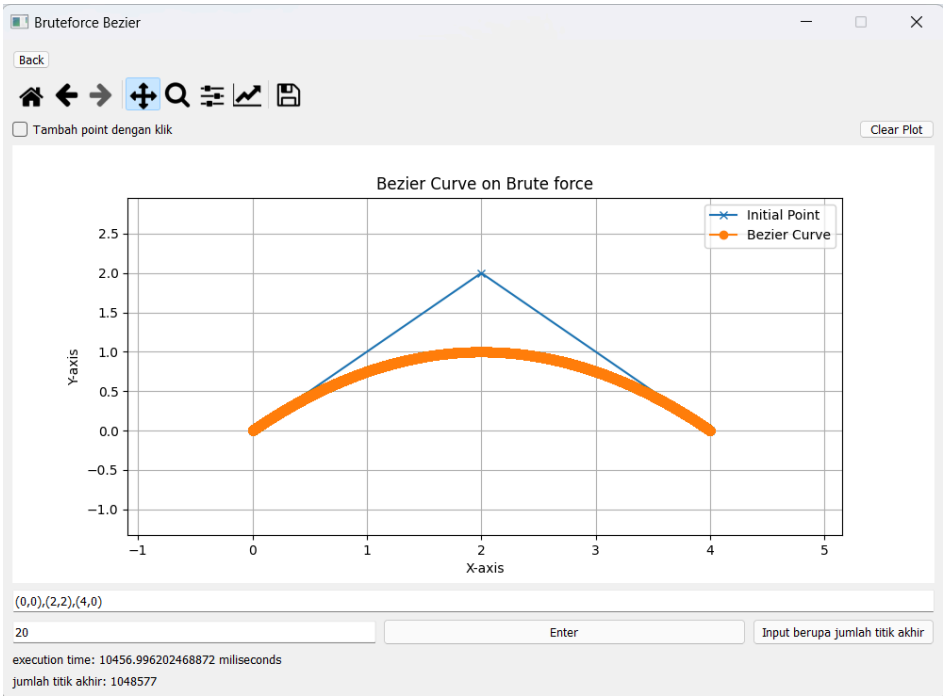

Divide And Conquer Bezier
Back
x=−0.827 y=3.37
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(0,0),(2,2),(4,0)
Animate
9
Enter
Tampilkan semua iterasi
execution time: 1.3763904571533203 miliseconds
jumlah titik akhir: 513

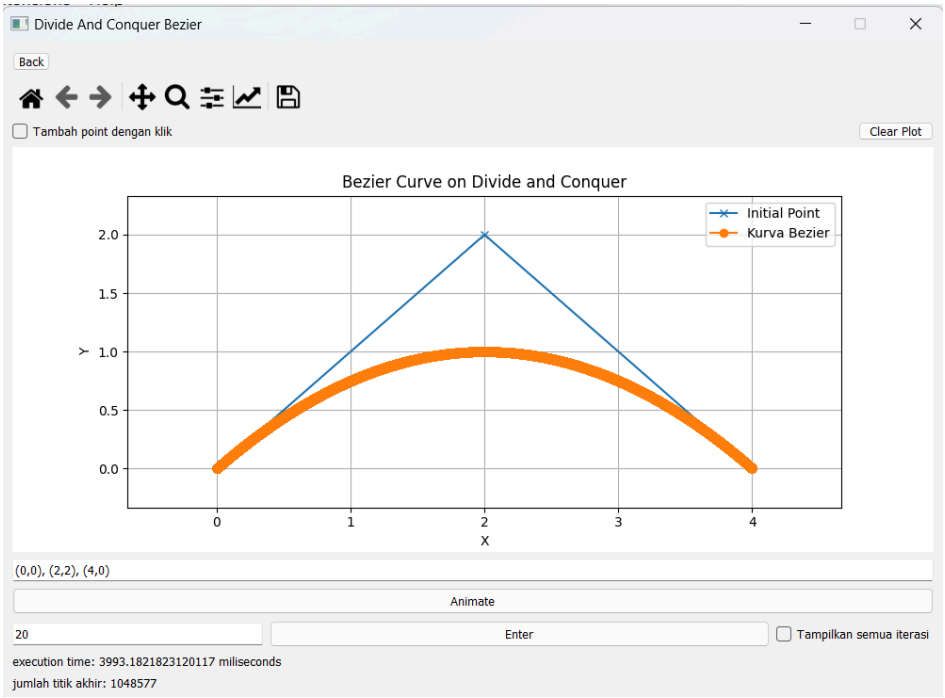
f. iterasi 12


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(0,0),(2,2),(4,0)
12
Enter
Input berupa jumlah titik akhir
execution time: 46.36716842651367 miliseconds
jumlah titik akhir: 4097


Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(0,0),(2,2),(4,0)
Animate
12
Enter
Tampilkan semua iterasi
execution time: 15.598773956298828 miliseconds
jumlah titik akhir: 4097

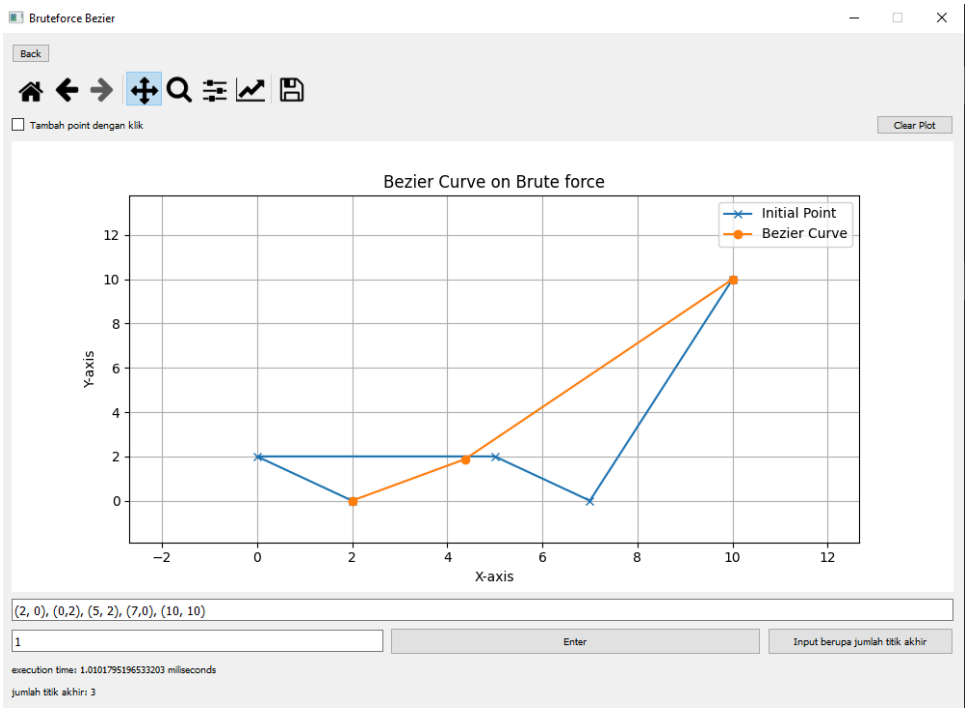
g. iterasi 20

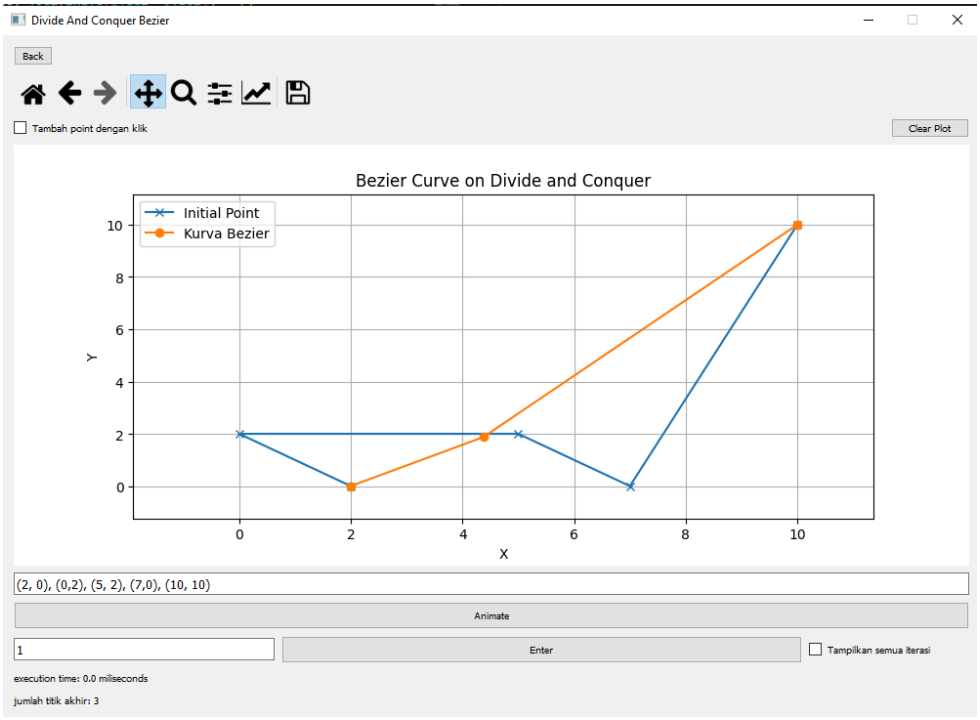




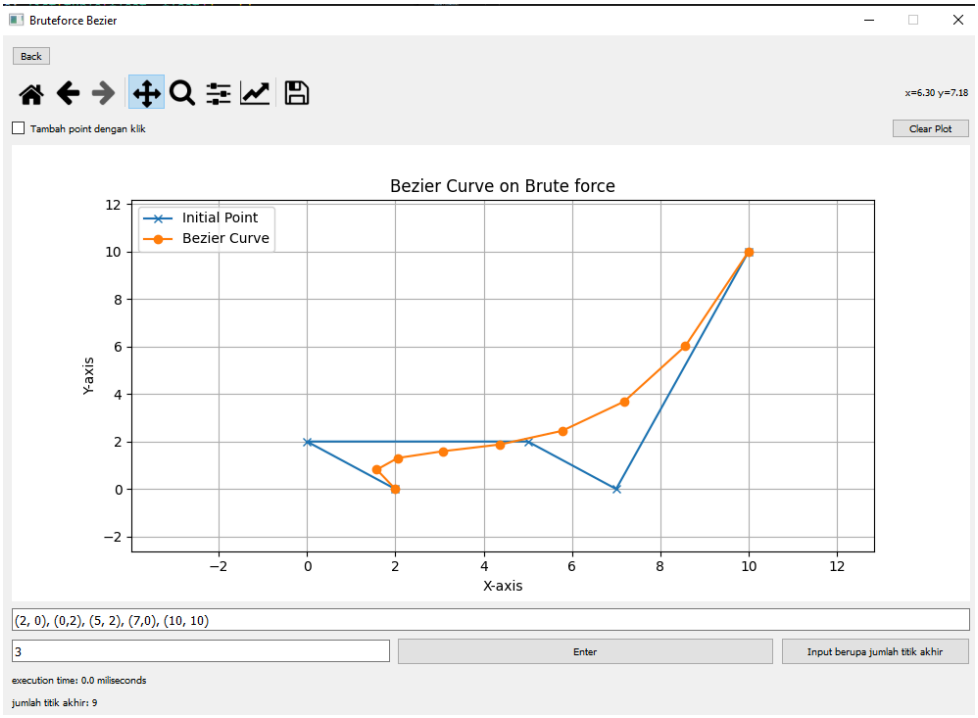
B. Titik yang diuji:

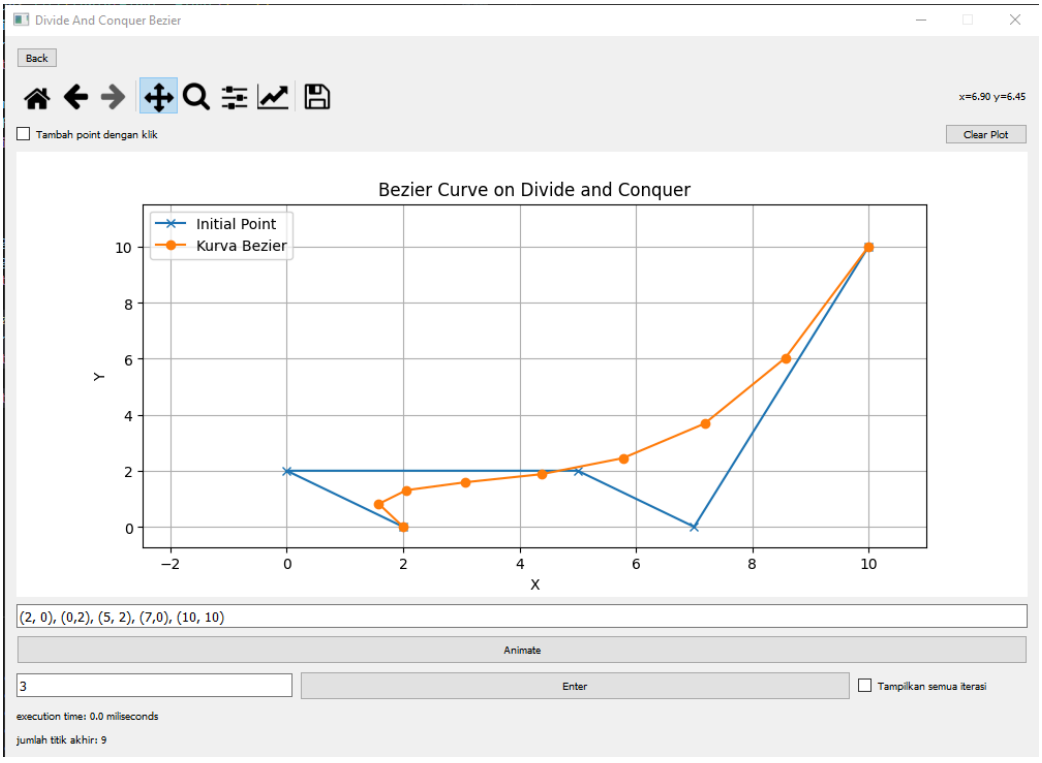
1. iterasi 1: (2, 0), (0,2), (5, 2), (7,0), (10, 10)


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
1
Enter
Input berupa jumlah titik akhir
execution time: 1.010179519653320 milliseconds
jumlah titik akhir: 3

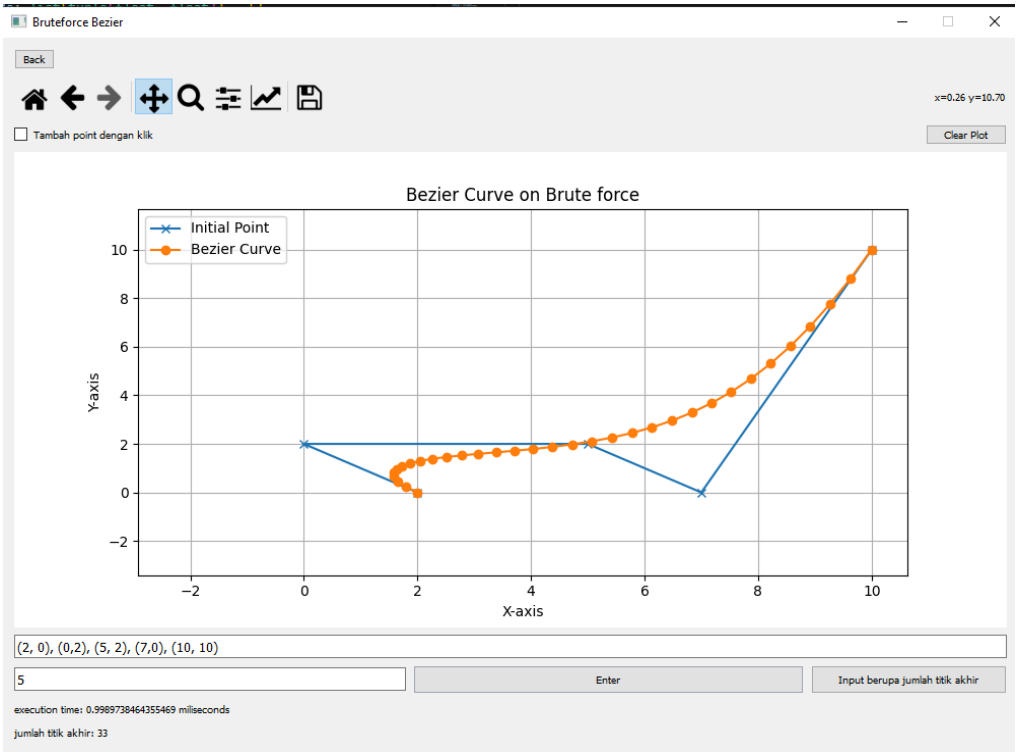

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
Animate
1
Enter
Tampilkan semua iterasi
execution time: 0.0 miliseconds
jumlah titik akhir: 3

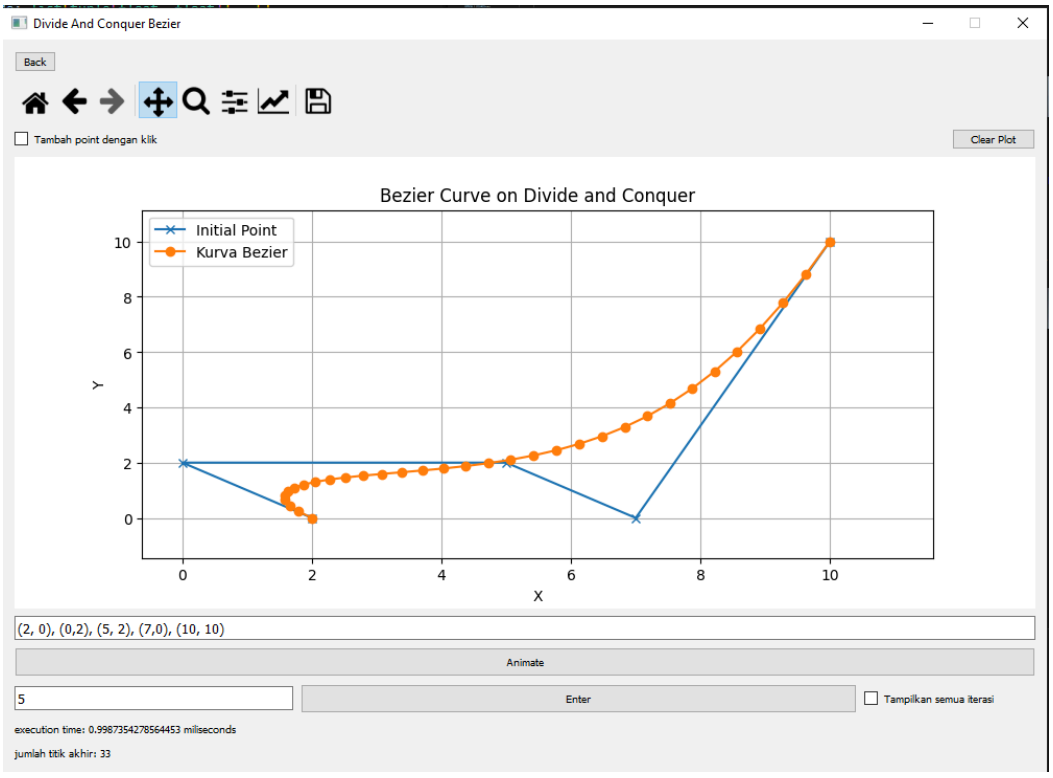
2. iterasi 3



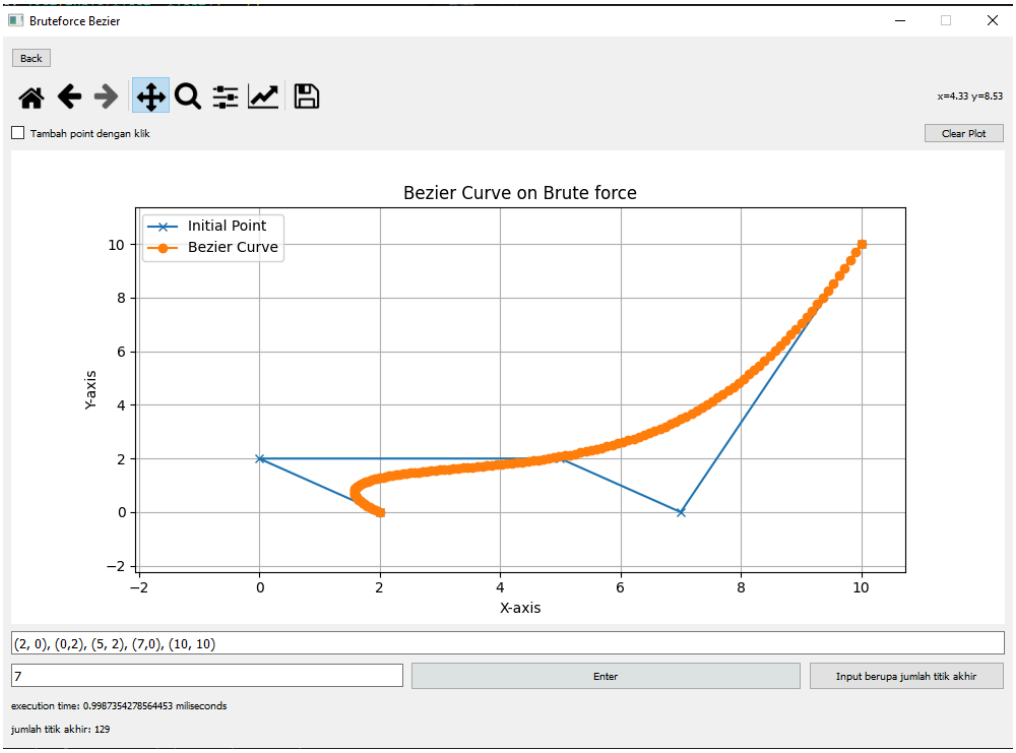


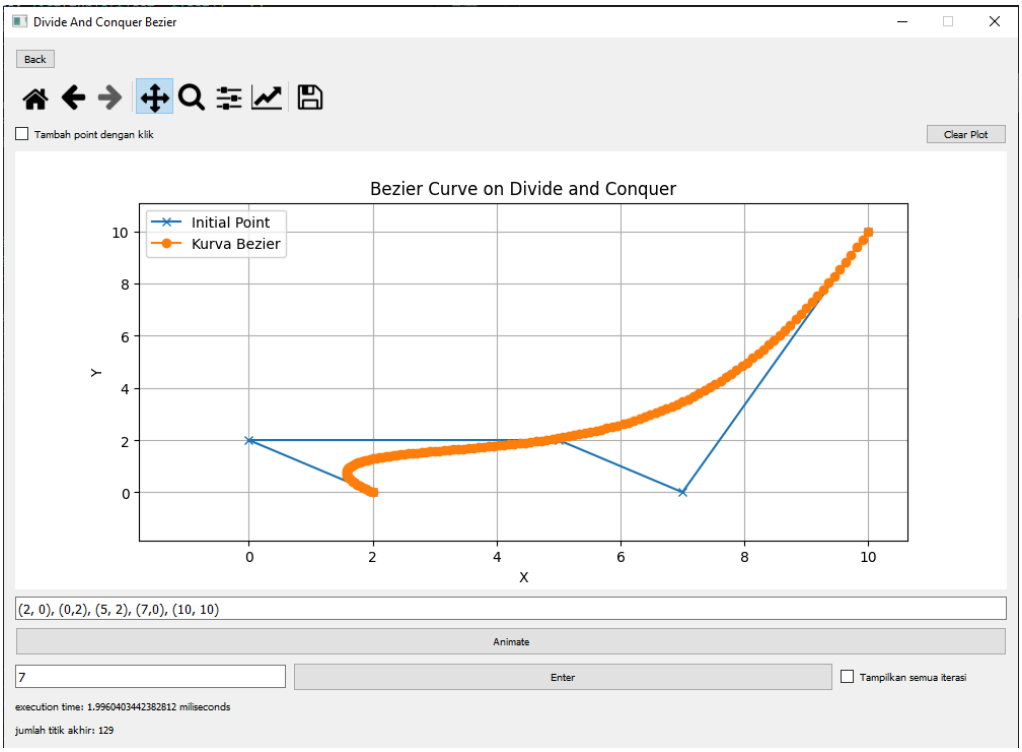
3. iterasi 5


Bruteforce Bezier
Back
x=0.26 y=10.70
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
5
Enter
Input berupa jumlah titik akhir
execution time: 0.998973846435469 miliseconds
jumlah titik akhir: 33

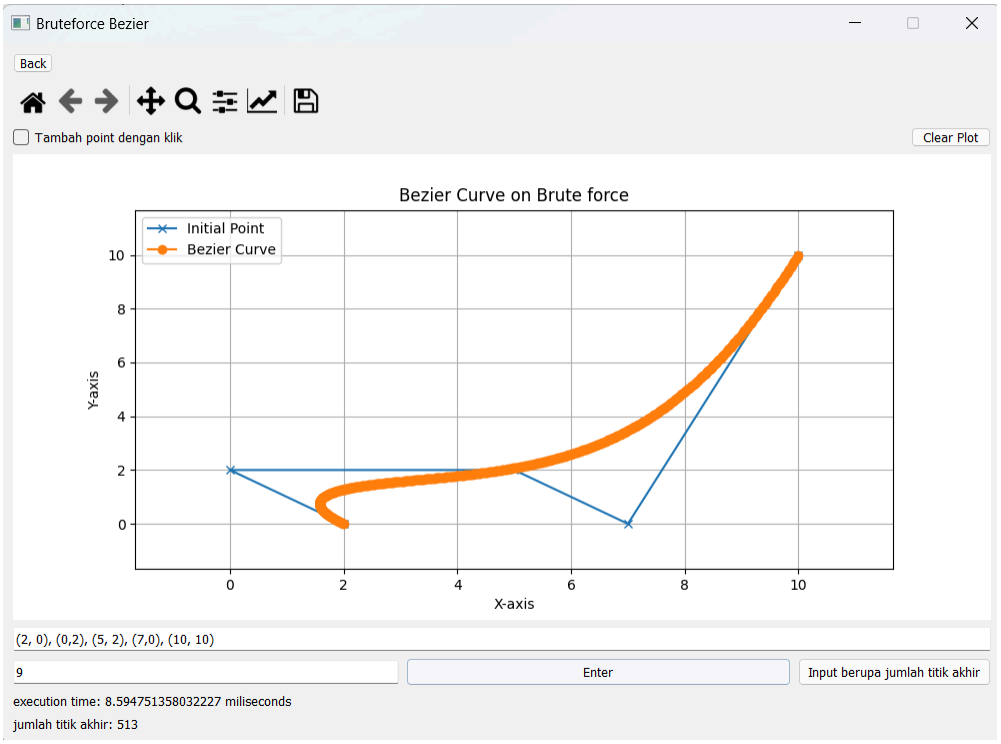

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
Animate
5
Enter
Tampilkan semua iterasi
execution time: 0.998735427856445 miliseconds
jumlah titik akhir: 33

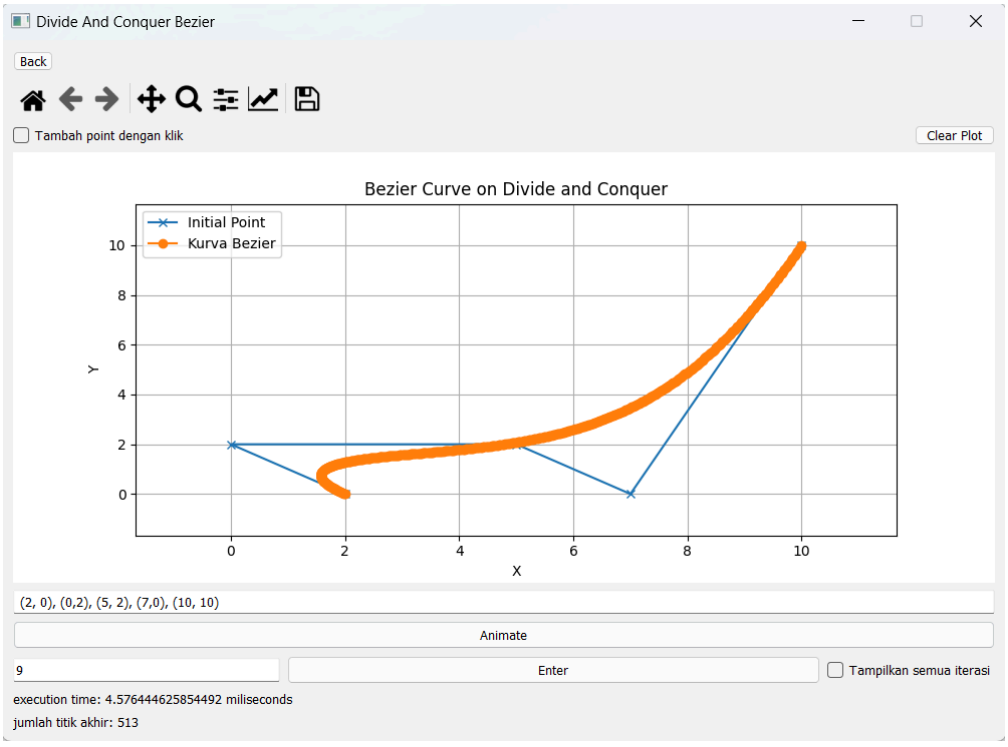
4. iterasi 7



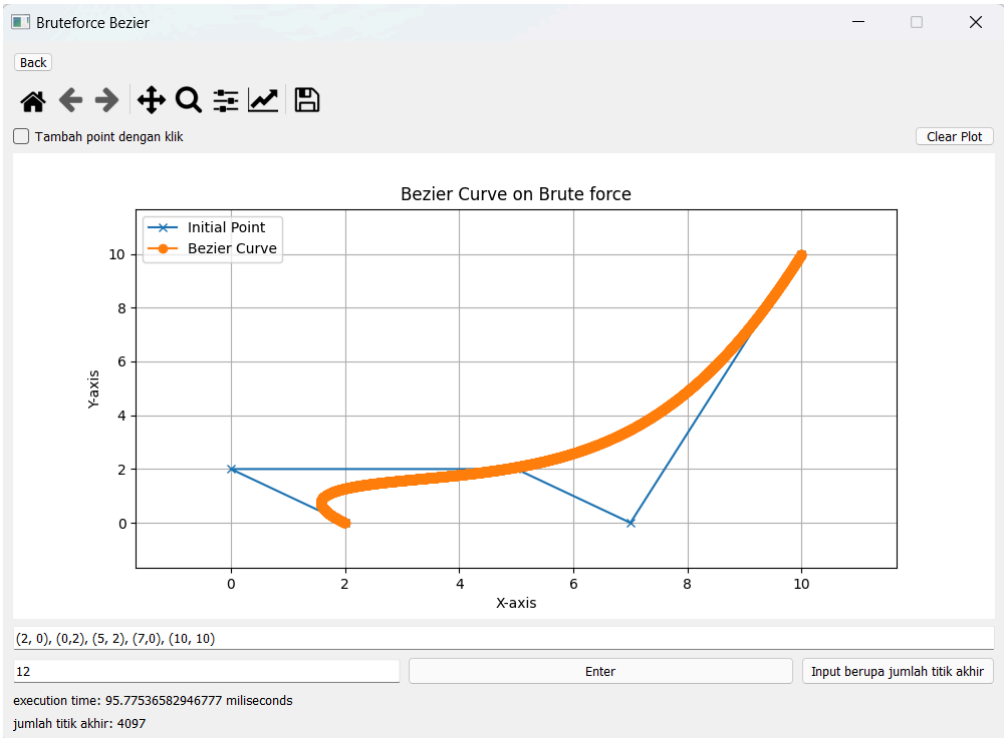


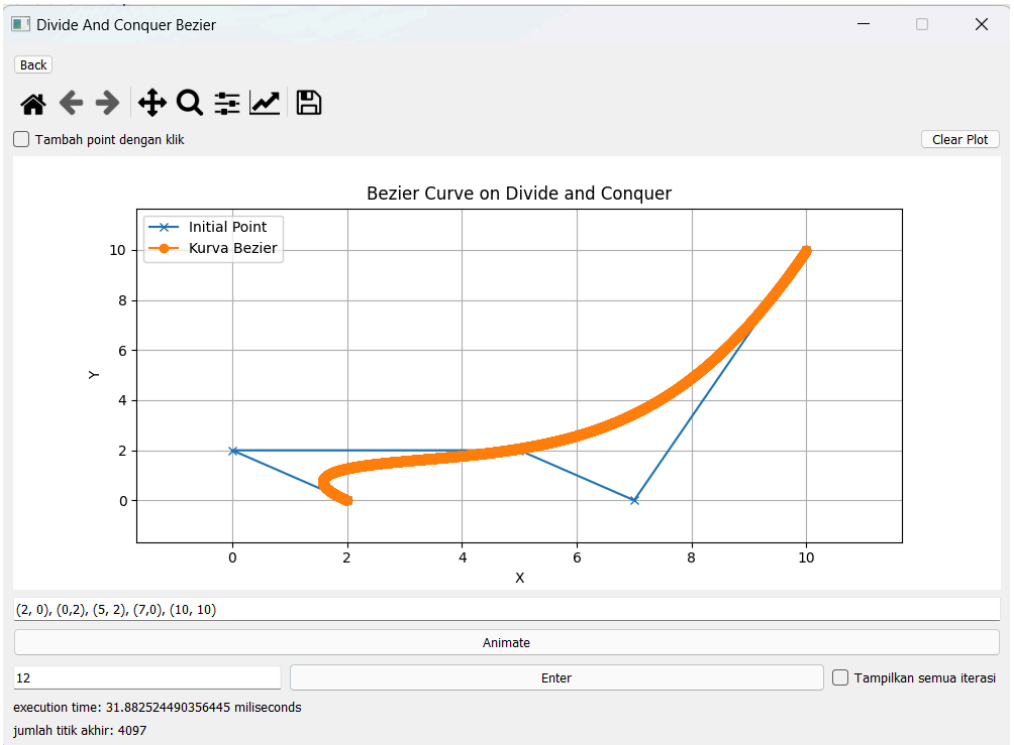
5. Iterasi 9


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
9
Enter
Input berupa jumlah titik akhir
execution time: 8.594751358032227 miliseconds
jumlah titik akhir: 513

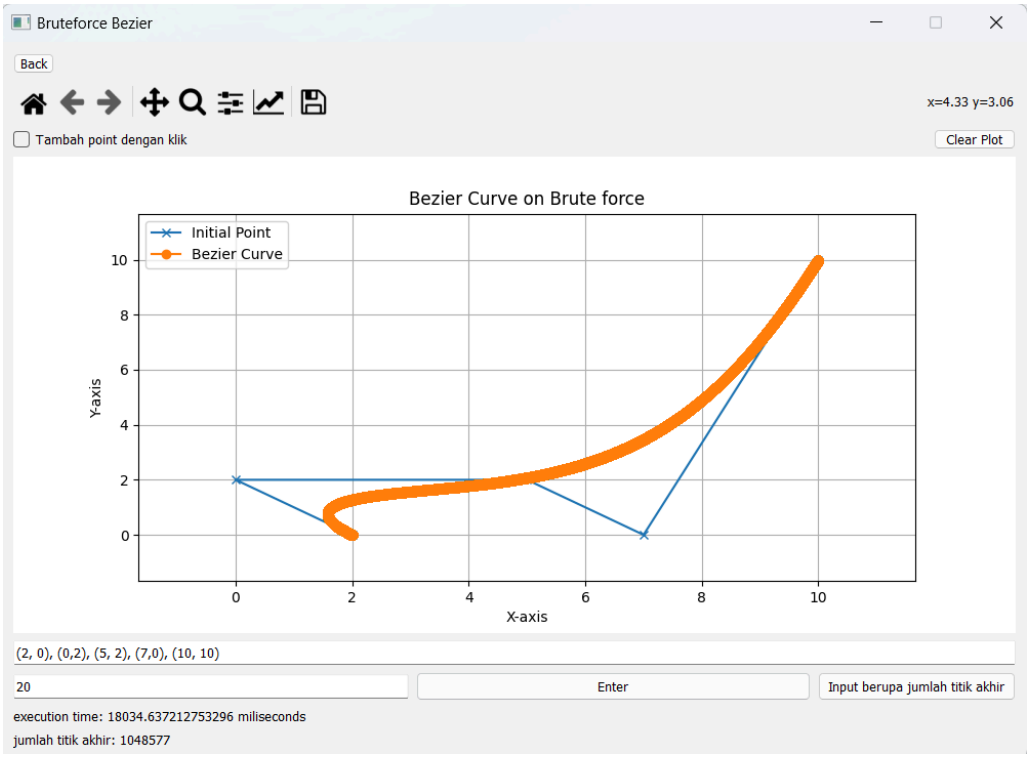

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
Animate
9
Enter
Tampilkan semua iterasi
execution time: 4.576444625854492 miliseconds
jumlah titik akhir: 513

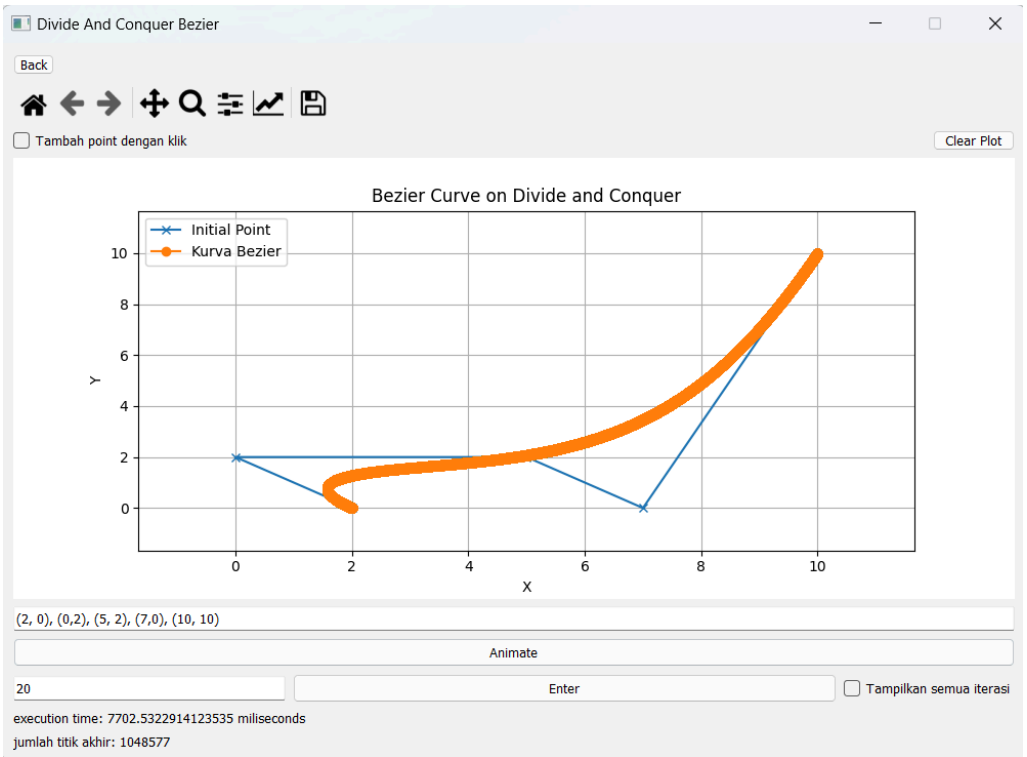
6. Iterasi 12


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
12
Enter
Input berupa jumlah titik akhir
execution time: 95.77536582946777 miliseconds
jumlah titik akhir: 4097


Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(2, 0), (0,2), (5, 2), (7,0), (10, 10)
Animate
12
Enter
Tampilkan semua iterasi
execution time: 31.882524490356445 miliseconds
jumlah titik akhir: 4097

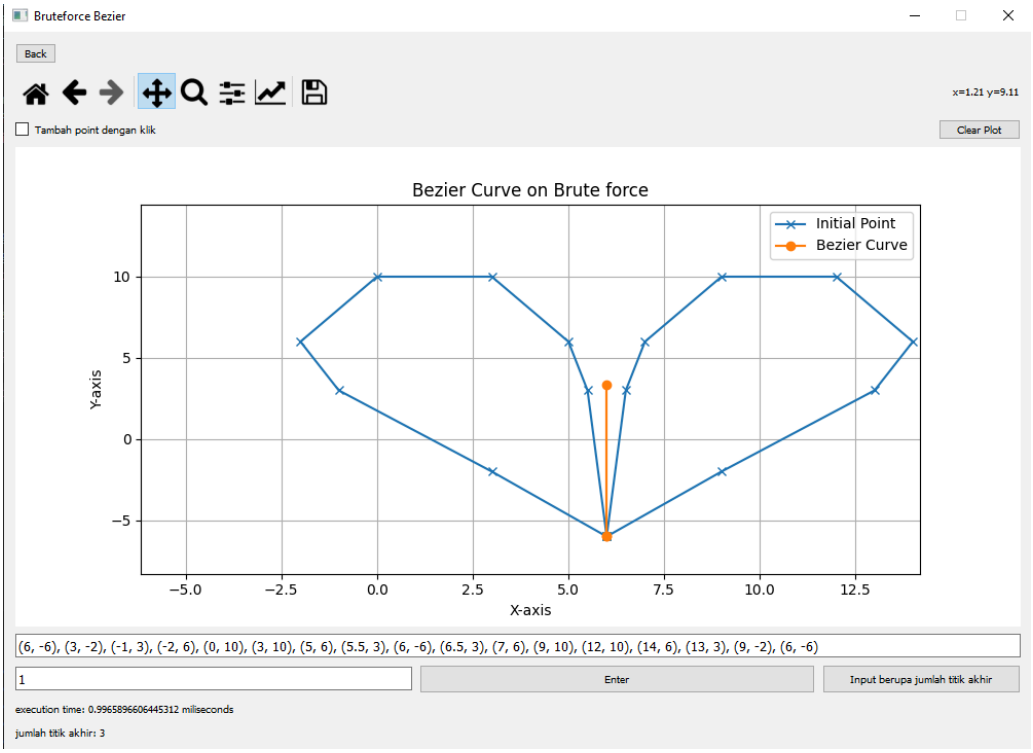
7. Iterasi 20

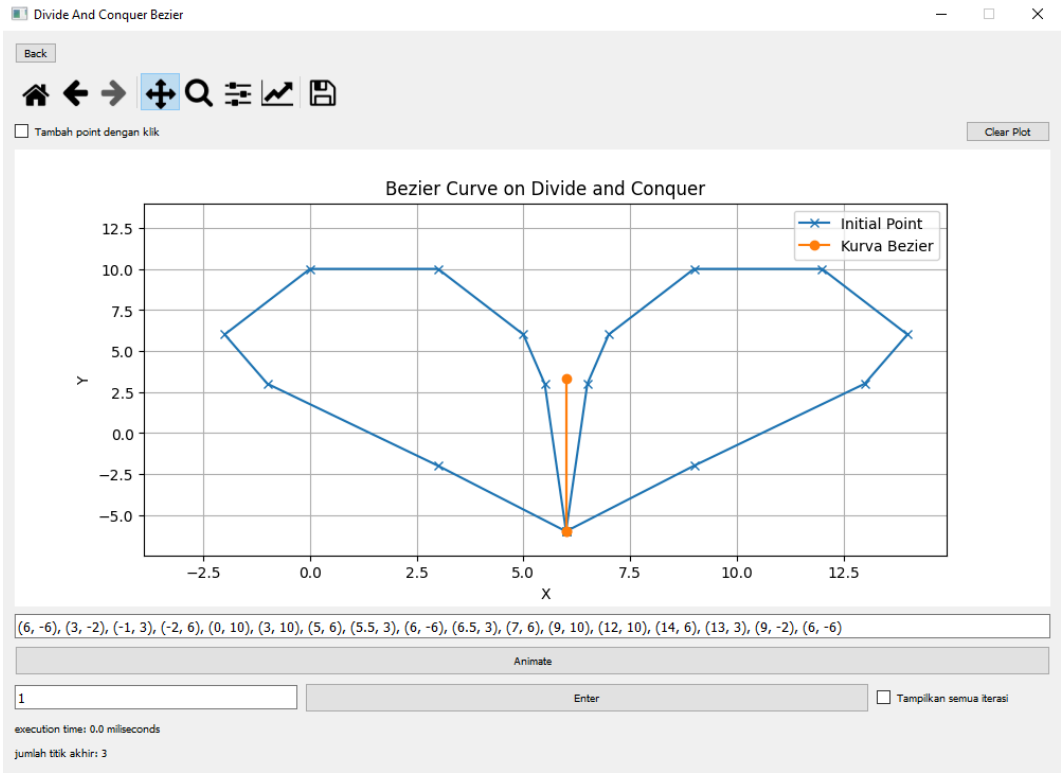




C. Titik yang diuji: (6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)

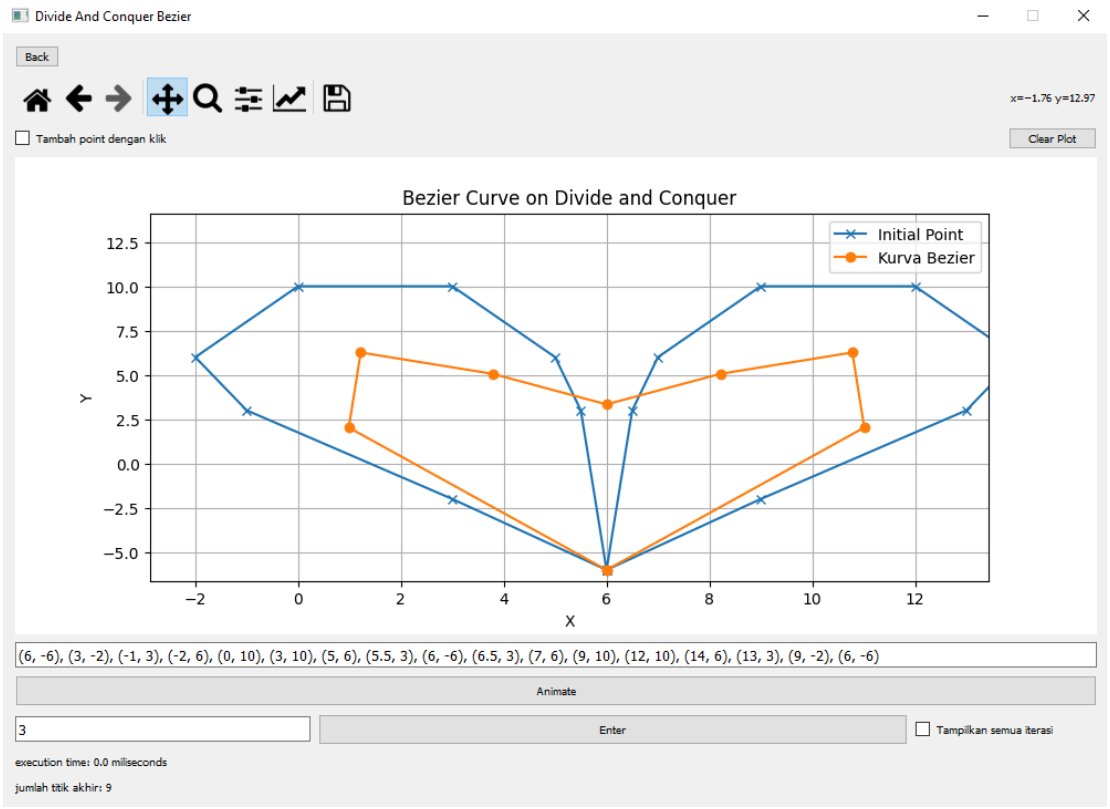
1. Iterasi 1


Bruteforce Bezier
Back
x=1.21 y=9.11
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
1
Enter
Input berupa jumlah titik akhir
execution time: 0.9965896606445312 milliseconds
jumlah titik akhir: 3

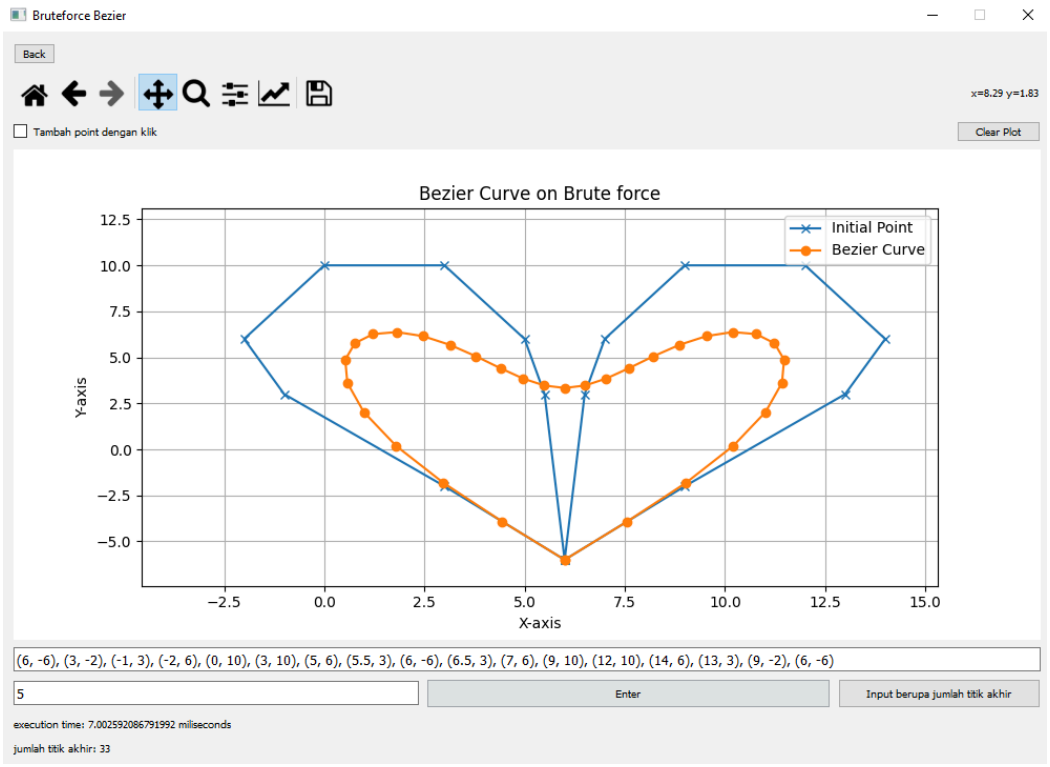

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
1
Enter
Tampilkan semua iterasi
execution time: 0.0 milliseconds
jumlah titik akhir: 3

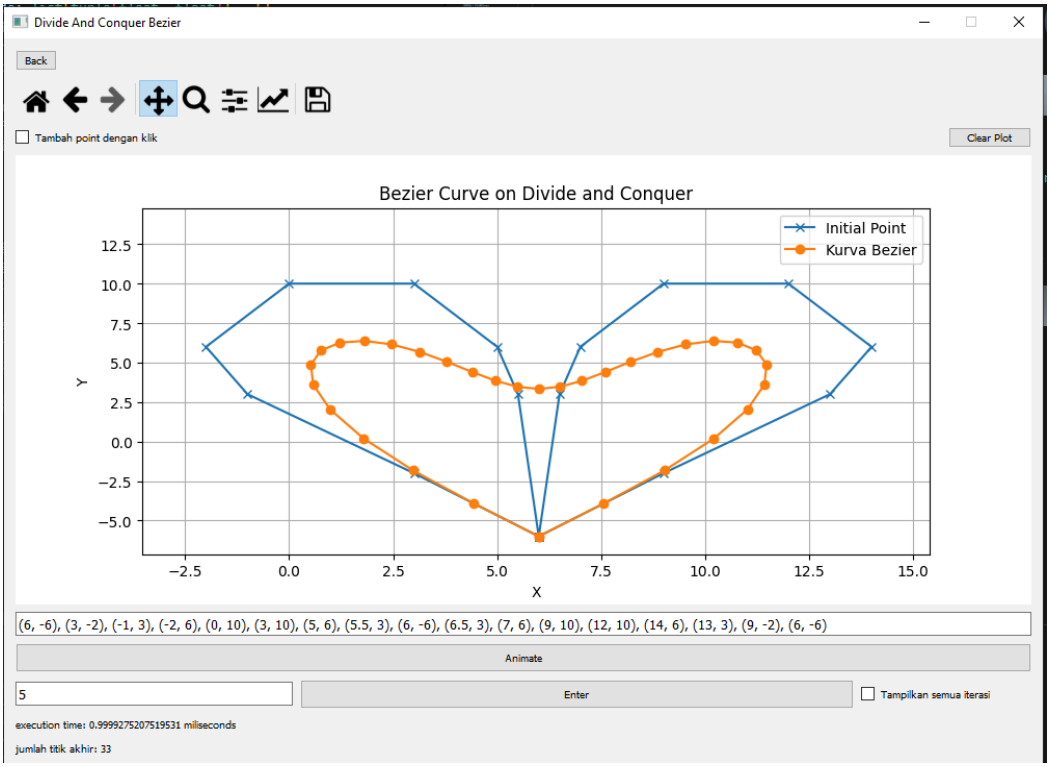
2. Iterasi 3


Bruteforce Bezier
Back
x=−2.67 y=8.42
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
15
10
5
0
−5
−2.5
0.0
2.5
5.0
7.5
10.0
12.5
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
3
Enter
Input berupa jumlah titik akhir
execution time: 1.9986629486083984 miliseconds
jumlah titik akhir: 9

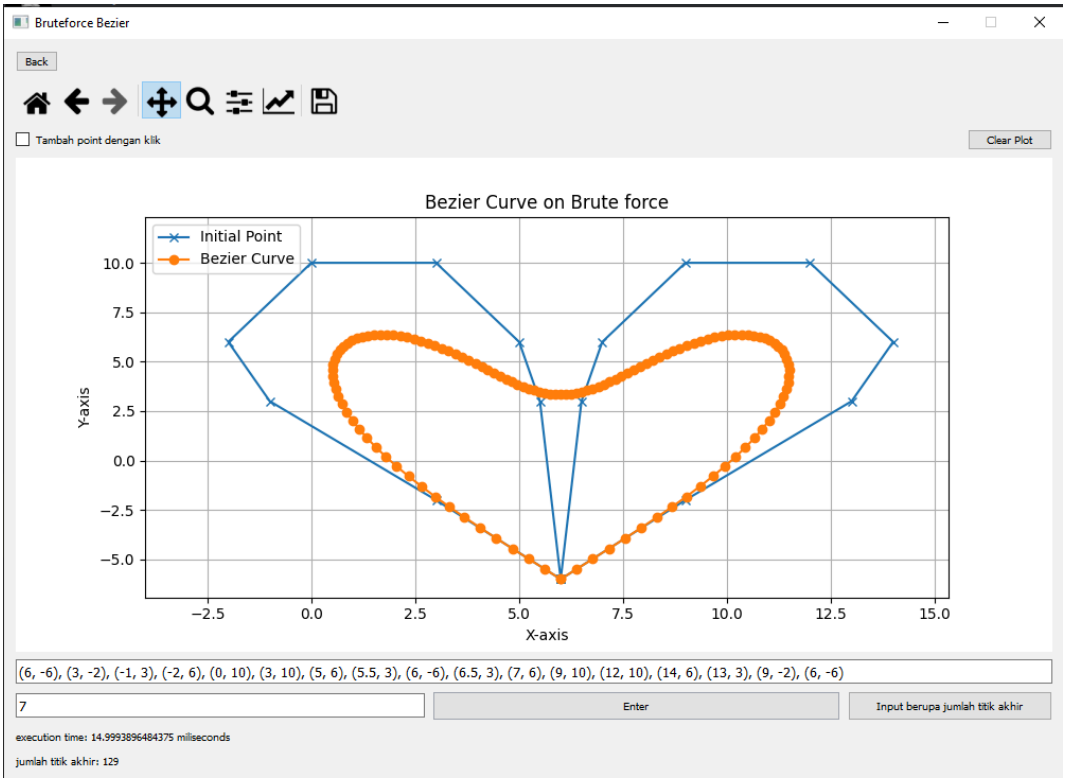

Divide And Conquer Bezier
Back
x=−1.76 y=12.97
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
X
Y
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
3
Enter
Tampilkan semua iterasi
execution time: 0.0 miliseconds
jumlah titik akhir: 9

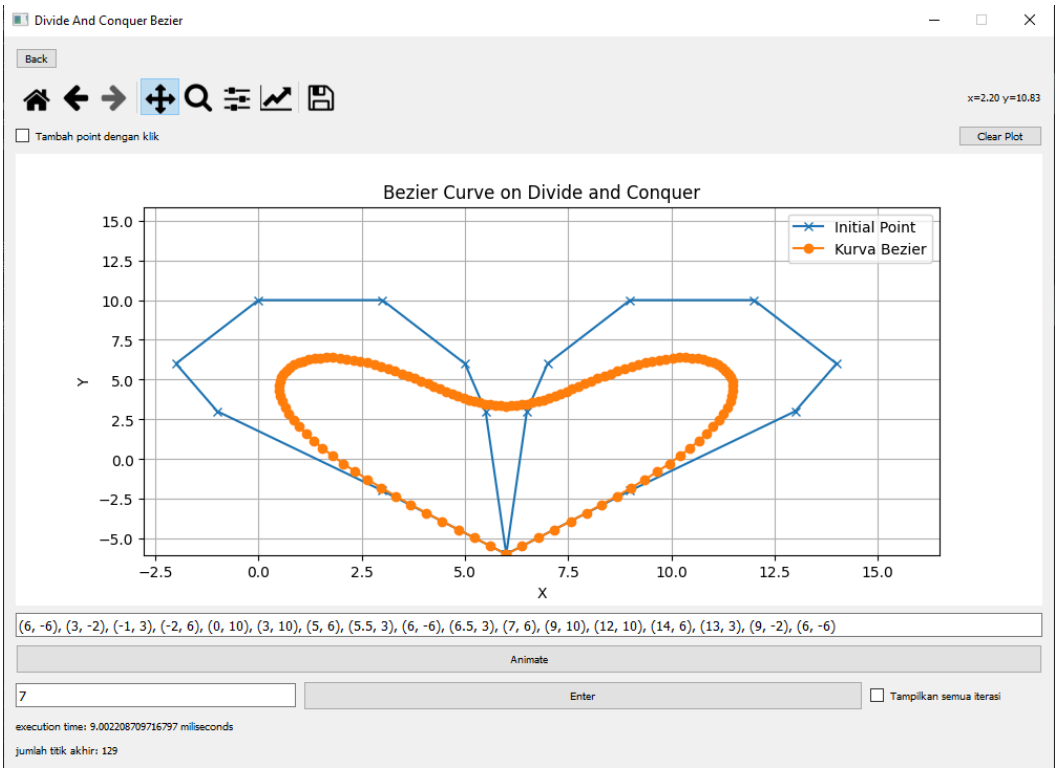
3. Iterasi 5


Bruteforce Bezier
Back
x=8.29 y=1.83
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
5
Enter
Input berupa jumlah titik akhir
execution time: 7.002592086791992 miliseconds
jumlah titik akhir: 33

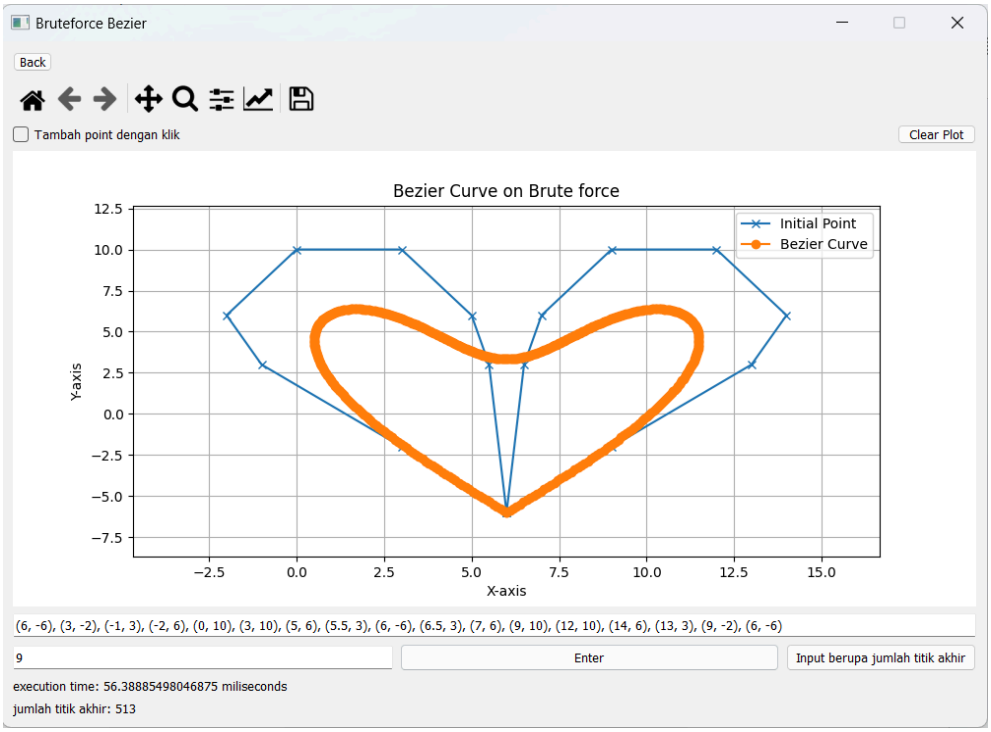

Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
5
Enter
Tampilkan semua iterasi
execution time: 0.9999275207519531 miliseconds
jumlah titik akhir: 33

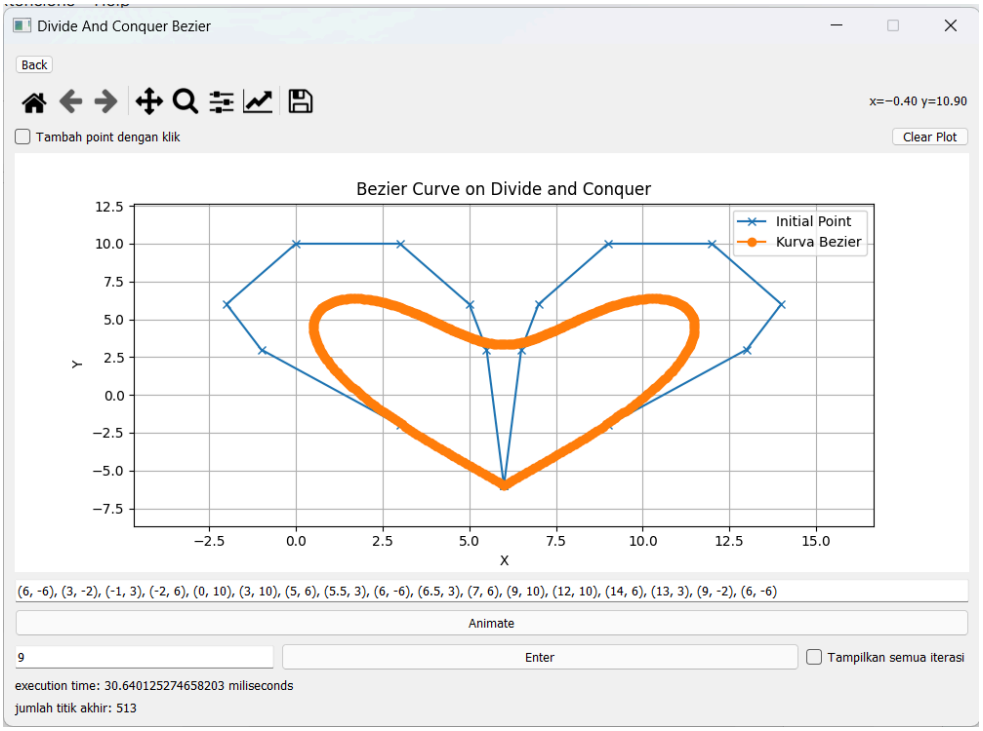
4. Iterasi 7


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
7
Enter
Input berupa jumlah titik akhir
execution time: 14.9993896484375 miliseconds
jumlah titik akhir: 129


Divide And Conquer Bezier
Back
x=2.20 y=10.83
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
7
Enter
Tampilkan semua iterasi
execution time: 9.002208709716797 miliseconds
jumlah titik akhir: 129

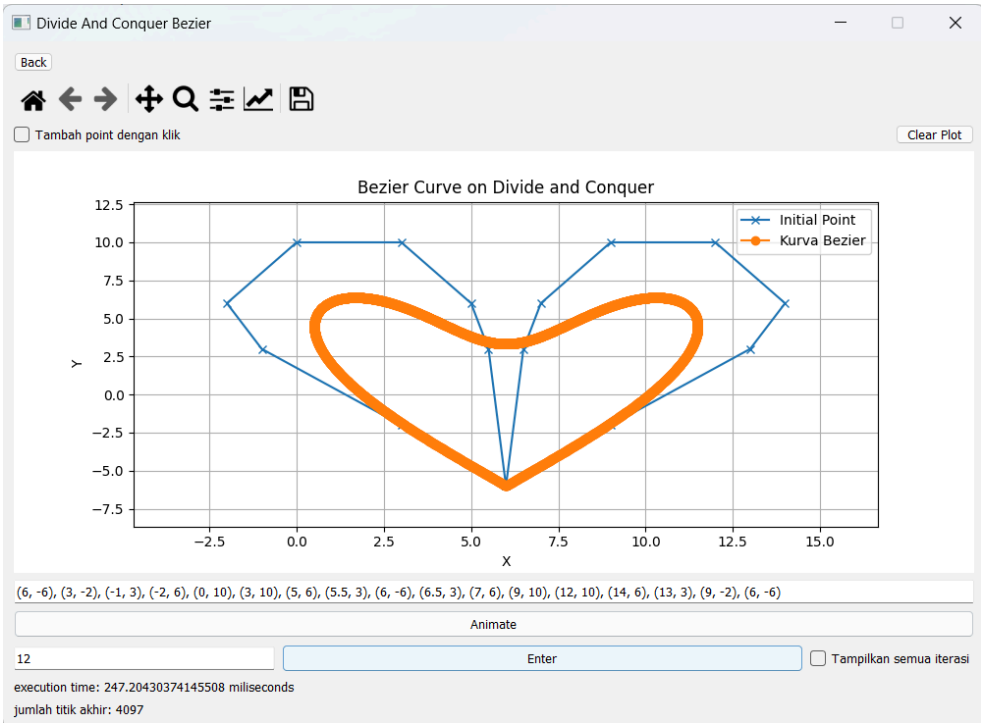
5. Iterasi 9


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
9
Enter
Input berupa jumlah titik akhir
execution time: 56.38885498046875 miliseconds
jumlah titik akhir: 513

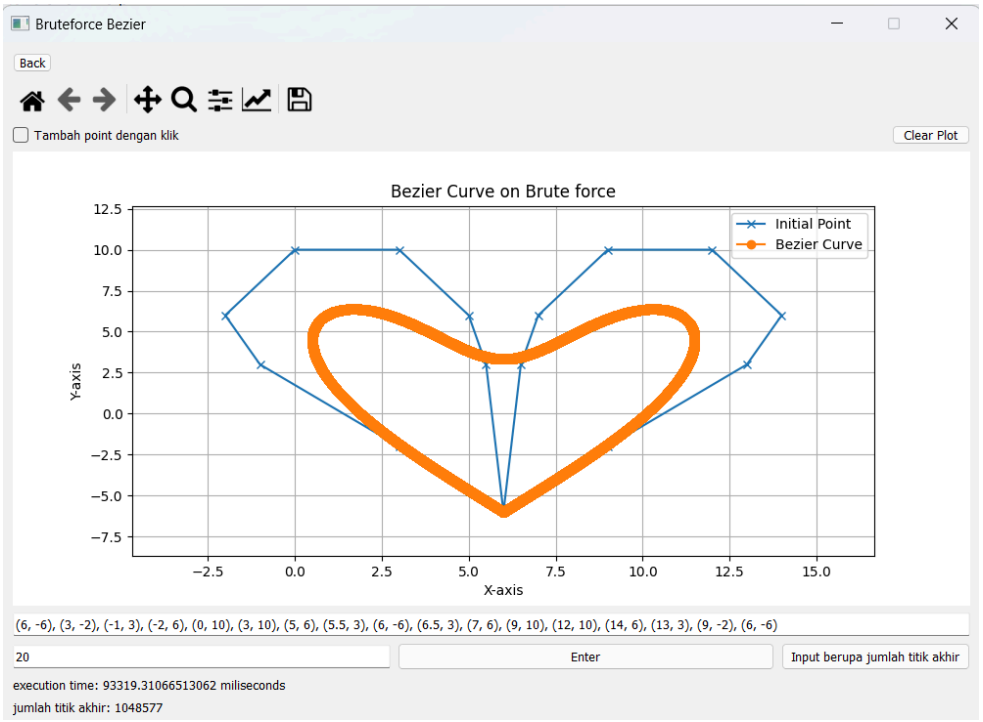

Divide And Conquer Bezier
Back
x=−0.40 y=10.90
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
9
Enter
Tampilkan semua iterasi
execution time: 30.640125274658203 miliseconds
jumlah titik akhir: 513

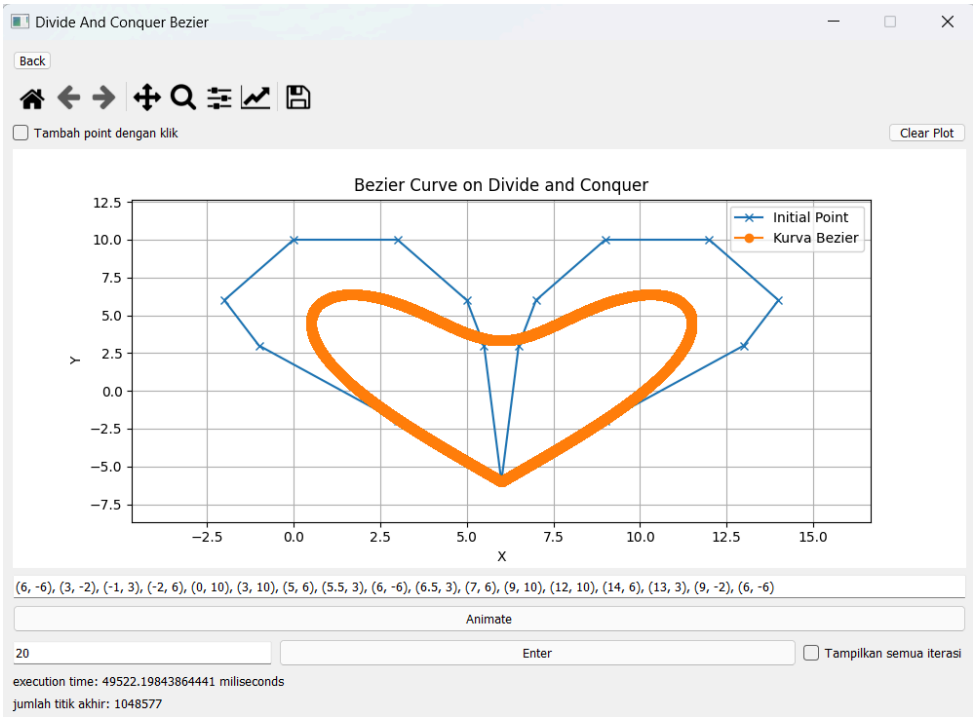
6. Iterasi 12


Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
12
Enter
Input berupa jumlah titik akhir
execution time: 459.35773849487305 miliseconds
jumlah titik akhir: 4097


Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
12
Enter
Tampilkan semua iterasi
execution time: 247.20430374145508 miliseconds
jumlah titik akhir: 4097

7. Iterasi 20

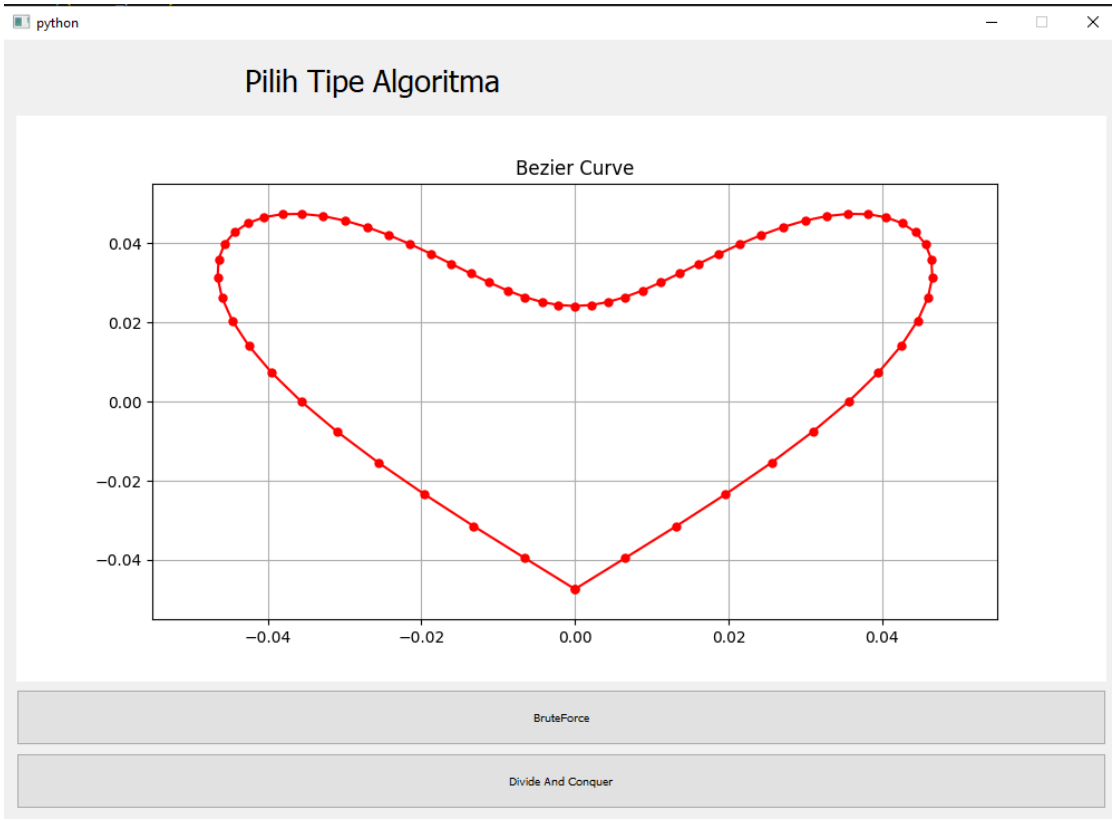

Bruteforce Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Brute force
Initial Point
Bezier Curve
Y-axis
X-axis
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
20
Enter
Input berupa jumlah titik akhir
execution time: 93319.31066513062 miliseconds
jumlah titik akhir: 1048577


Divide And Conquer Bezier
Back
Tambah point dengan klik
Clear Plot
Bezier Curve on Divide and Conquer
Initial Point
Kurva Bezier
Y
X
(6, -6), (3, -2), (-1, 3), (-2, 6), (0, 10), (3, 10), (5, 6), (5.5, 3), (6, -6), (6.5, 3), (7, 6), (9, 10), (12, 10), (14, 6), (13, 3), (9, -2), (6, -6)
Animate
20
Enter
Tampilkan semua iterasi
execution time: 49522.19843864441 miliseconds
jumlah titik akhir: 1048577

4.4 GUI

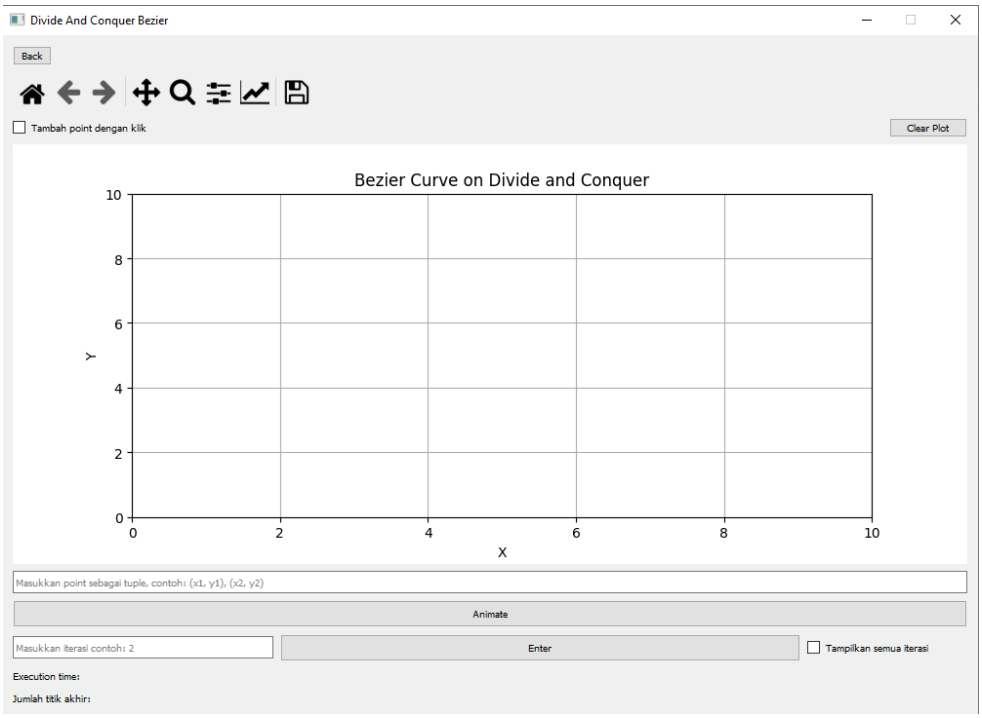
a. Penjelasan Umum

Disini kami menggunakan library GUI dari python yaitu PyQt5. Berikut adalah tampilan depan dari GUI kami.



User bisa memilih tipe algoritma yang akan digunakan. Untuk tampilan dari Algoritma brute force adalah sebagai berikut

dan untuk tampilan dari algoritma Divide and Conquer adalah sebagai berikut:



Untuk masukan Point-nya menggunakan format : (x1, y1), (x2, y2) karena kami sudah menggunakan algoritma yang memproses N point, sehingga tidak diperlukan lagi penerimaan N point langsung saja masukkan Point-nya. Lalu selanjutnya iterasi menggunakan format input integer baik di Divide and Conquer, maupun di Bruteforce,

tetapi ada sedikit perbedaan pada bagian Divide and Conquer, karena pada bagian ini kami memiliki tombol dan checkbox tambahan yaitu animate, dan tampilkan semua titik. Untuk animate ini menampilkan animasi yang menunjukkan proses iterasi dari titik-titik yang kita inputkan mulai dari iterasi 1 sampai iterasi N (iterasi yang kita input). Untuk checkbox tampilkan semua iterasi ini adalah untuk menampilkan keseluruhan iterasi yang ada pada plot.

b. Fitur unik

1. Tambah Point dengan klik

   Pada bagian ini GUI kami juga bisa melakukan input melalui klik pada canvas matplotlib, tetapi harus mencentang terlebih dahulu “Tambah point dengan klik” pada kiri atas dibawah bagian toolbar.

2. Clear Plot

   Tombol ini ditujukan agar menghapus titik-titik yang berada pada canvas matplotlib dan yang berada pada input box Point hilang seketika.

BAB V

KESIMPULAN DAN SARAN

5. 1 Kesimpulan

Dari hasil pengujian pada bab sebelumnya dapat dilihat bahwa dalam masalah pembuatan kurva Bézier dengan n titik kontrol (n >= 3), algoritma Divide and Conquer yang kami buat dapat menghasilkan kurva lebih cepat daripada algoritma *bruteforce* dengan jumlah titik kontrol dan titik akhir yang sama. Di lain sisi, algoritma Divide and Conquer yang kami buat memiliki kekurangan yaitu hanya bisa menerima masukan berupa iterasi sehingga jumlah titik akhir tidak bisa bebas sesuai keinginan. Jadi, pada kasus ini kedua algoritma memiliki kelebihan dan kekurangan masing-masing, Divide and Conquer bisa membuat kurva lebih cepat sedangkan *bruteforce* lebih fleksibel dalam menerima masukan berupa titik akhir.

5.2 Saran

Bagi pembaca yang membaca laporan ini kami harap pembaca bisa mencoba melakukan apa yang kami buat, tetapi menggunakan bahasa yang beda mungkin saja pembaca bisa menjadikan prosesnya menjadi lebih baik karena bahasa yang dipilih untuk menyelesaikan persoalan Bezier Curve.

LAMPIRAN

- Tautan *repository* Github:
  https://github.com/b33rk/Tucil_13522101_13522119
- Penyelesaian

Poin	Ya	Tidak
1. Program berhasil dijalankan	✓	
2. Program dapat melakukan visualisasi kurva bezier	✓	
3. Solusi yang diberikan program optimal	✓	
4. [Bonus] Program dapat membuat kurva untuk n titik kontrol.	✓	
5. [Bonus] Program dapat melakukan visualisasi proses pembuatan kurva	✓	

DAFTAR PUSTAKA