Minimum Kapsayan Daire Problemi

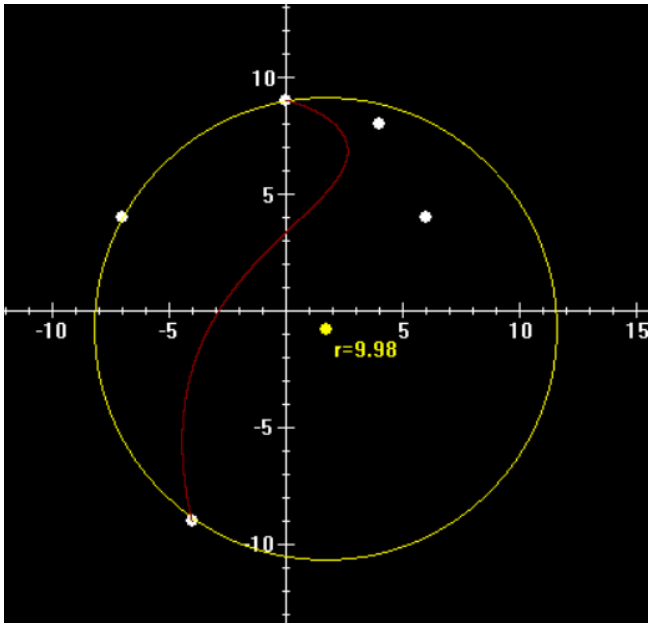
Muhammed Taha METİN, Burak YAVUZKILIÇ

1. ÖZET

Proje, koordinatı verilen N tane noktayı çevreleyen en küçük çemberi ve bu noktalara en yakın geçen eğriyi çizdirmeyi amaçlar.

2. GİRİŞ

Belirlenen metin dosyasına nokta koordinatları girilir. Bu koordinatlar programdaki bir tam sayı dizisine kaydedilir. Ardından noktaların koordinat düzleminde belirtilmesiyle birlikte noktaları çevreleyen en küçük daire ve noktalara en yakın geçen eğri çizilir.




```
cemberin merkezi --> 1.735849 , -0.830189
cemberin yaricapi --> 9.982273
```

3. YÖNTEM

Program C dilini ve “Graphics.h” kütüphanesini kullanmaktadır.

Minimum kapsayan daire problemini çözmek için önce verilen noktaların tüm 2’li kombinasyonlarını dairenin çapının 2 ucu kabul edip dışarda nokta kalıp kalmadığını kontrol eder. Dışarda nokta kalmayan daire kombinasyonlarından en küçük yarıçaplı daireyi cevap olarak belirler.

2 noktadan çemberin yarıçapı ve merkezini bulmak için 2 noktanın merkezini çemberi merkezi olarak, iki nokta arasındaki uzaklığın yarısını da yarıçap olarak kabul eder.

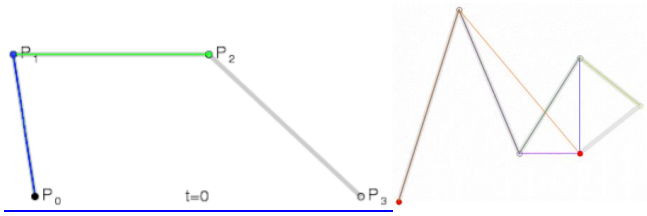
Eğer dışarda nokta kalmayan 2’li nokta kombinasyonu yok ise tüm noktaların 3’lü nokta kombinasyonlarını alıp her kombinasyon için dışarda nokta kalıp kalmadığına ve yarıçapın dışarda nokta bırakmayan diğer çemberlerden daha küçük olup olmadığına bakar. Bulunan en küçük yarıçaplı ve dışarda nokta bırakmayan daireyi cevap olarak belirler

3 noktadan çemberin merkezi ve yarıçapını bulmak için

- $x^2 + y^2 + 2gx + 2fy + c = 0$
- $(x – h)^2 + (y -k)^2 = r^2$

çember denklemleri kullanılır.

Noktaların en yakınından geçen eğriyi Bezier Curve eğri biçimi ile çizdirir.



Açık tanım:

$$\mathbf{B}(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n-i} t^i \mathbf{P}_i$$

$$= (1-t)^n \mathbf{P}_0 + \binom{n}{1}(1-t)^{n-1} t \mathbf{P}_1 + \cdots + \binom{n}{n-1}(1-t) t^{n-1} \mathbf{P}_{n-1} + t^n \mathbf{P}_n$$

$$0 \leqslant t \leqslant 1$$

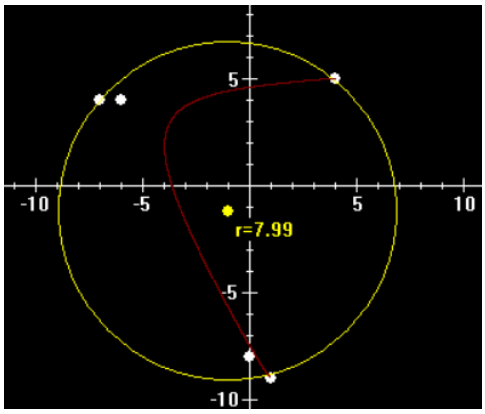
Yinelemeli tanım:

$$\mathbf{B}_{\mathbf{P}_0}(t) = \mathbf{P}_0, \text{ and}$$
$$\mathbf{B}(t) = \mathbf{B}_{\mathbf{P}_0\mathbf{P}_1\ldots\mathbf{P}_n}(t) = (1-t)\mathbf{B}_{\mathbf{P}_0\mathbf{P}_1\ldots\mathbf{P}_{n-1}}(t) + t\mathbf{B}_{\mathbf{P}_1\mathbf{P}_2\ldots\mathbf{P}_n}(t)$$

Bezier Curve ile eğri çizdirilirken her ardışık nokta arasında t’nin değerinin 0’dan 1’e gitmesi ile ilerleyen bir kontrol noktası oluşturulur. Bu kontrol noktalarından ardışık olanlar arasında bir çizgi çekilir ve o çizginin de başlangıcından sonuna t’nin 0’dan 1’e gitmesi ile ilerleyen bir kontrol noktası yerleştirilir. Bu çizgi ve kontrol noktası yerleştirme işlemi sadece 1 kontrol noktası kalana kadar devam eder. En son üretilen kontrol noktasının, t 0’dan 1’e giderken geçtiği yerler bize eğriyi verir.

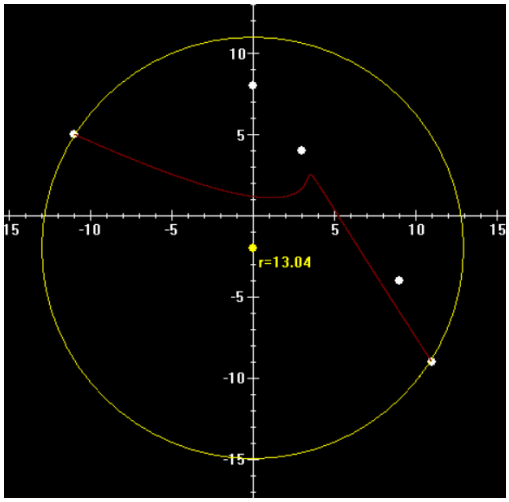
4. DENEYSEL SONUÇLAR

Noktalar: (4,5),(-7,4),(-6,4),(0,-8),(1,-9)




```
cemberin merkezi --> -0.976821 , -1.254967
cemberin yaricapi --> 7.993332
```

Noktalar: (-11,5),(9,-4),(3,4),(0,8),(11,-9)




```
cemberin merkezi --> 0.000000 , -2.000000
cemberin yaricapi --> 13.038405
```

5. KABA KOD (PSEUDO CODE)

void DrawXYAxis:

- X-Y düzlemini çizer.
- X-Y düzlemindeki 25 ile -25 arasındaki bir tam sayı ifade eden yerlere kısa çizgi çizer.
- X-Y düzlemindeki 25 ile -25 arasındaki 5 in katı sayılara bir uzun çizgi çizer ve sayıyı yazdırır.

void DrawCircle:

- Verilen değerlere göre çemberi çizer.
- Çemberin merkezine bir nokta çizer.
- Çemberin yarıçapını noktanın yanına yazdırır.

void DrawPoints:

- Noktaları koordinat düzleminde çizer.

double FindTheDistanceBetweenTwoPoints:

- Verilen iki nokta arasındaki mesafeyi hesaplayıp geri döndürür.

double MidPointOfTwoPoint:

- Verilen iki değerin ortalamasını geri döndürür.

void FindCircleFromThreePoint:

- Noktaların her 3'lü kombinasyonunu çemberin üzerindeki 3 nokta olarak kabul edip çemberin değerlerini bulur.
- Bu değerler ile çizilen çemberin dışında kalan nokta olup olmadığını kontrol eder.
- Dışarda nokta yoksa ve yarıçap, o ana kadar bulunan yarıçaptan daha küçükse (yarıçapın ilk değeri için yeterince yüksek bir değer girilmiştir) çemberin merkez ve yarıçap değerleri atanır.
- Bu işlemler tüm noktalar için uygulandıktan sonra çember çizdirilir.

void FindCircleFromTwoPoint:

- Noktaların her 2'li kombinasyonunu çemberin çapının üzerindeki 2 nokta olarak kabul edip çemberin değerlerini bulur.
- Bulunan değerlere göre dışarda nokta kalıp kalmadığı kotrol edilir.
- Dışarda nokta kalmıyor ve yarıçap, o ana kadar bulunan yarıçaptan daha küçükse (yarıçapın ilk değeri için yeterince yüksek bir değer girilmiştir) çemberin merkez ve yarıçap değerleri atanır.
- Bu işlemler tüm noktalar için uygulandıktan sonra dışarda nokta kalıp kalmadığı tekrar kontrol edilir.
- Dışarda nokta kalmadıysa ve en küçük yarıçap değeri kontrolden önceki değeriyle aynı
  - Değilse bulunan değerlerle çember çizdirilir.
  - İse FindCircleFromThreePoint fonksiyonunu çağır.

int fact:

- N sıfıra eşit ise fonksiyondan çık:
- N sıfırdan farklı ise n*fac(n-1) değerini döndür.

void comb:

- (fact(n) / (fact(n-r) * fact(r)) değerini döndür

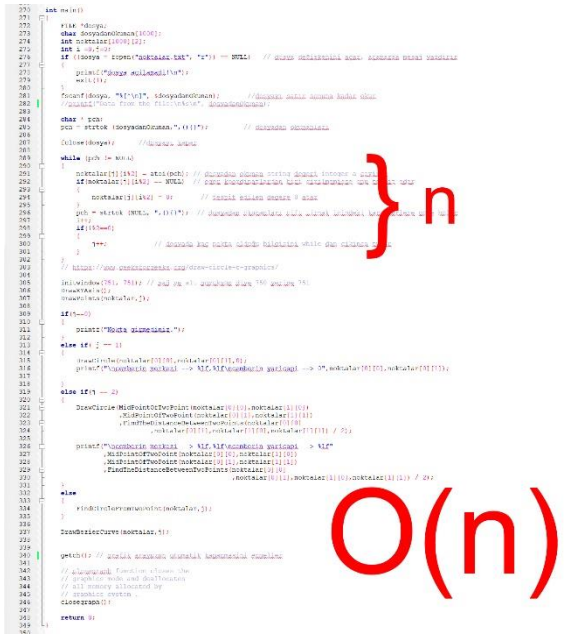
void DrawBezierCurve:

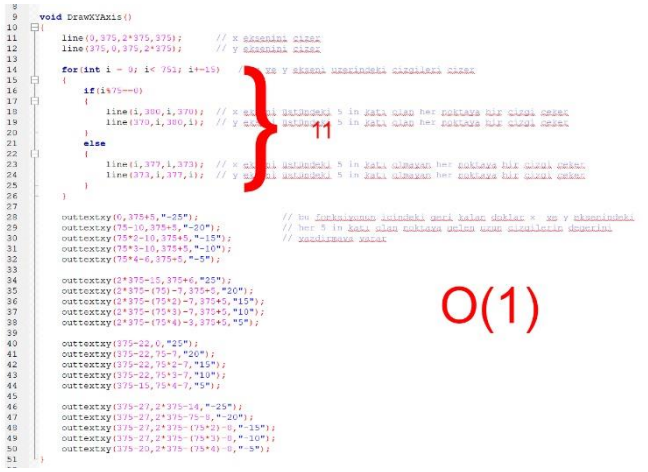
- T değerini 0'dan 1'e götürürken her T değeri için her noktaya $B(t) = \sum_{i=0}^{n} \binom{n}{i} (1-t)^{n+i} t^i P_i$ formülünü uygulayıp çıkan toplamı çizilecek noktaya atıyor.
- Her t değeri için belirlenen değerler ekrana nokta olarak çiziliyor.

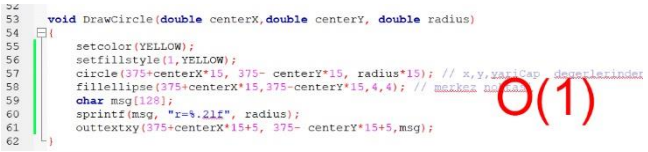
int main:

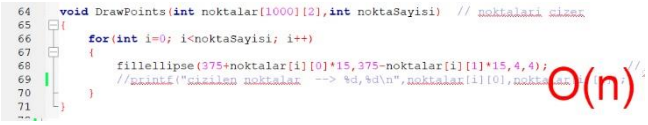
- Başla
- Dosyadan koordinatları oku.
- Okunan koordinatları noktalar dizisine yaz.
- Grafik penceresini aç.
- DrawXYAxis() fonksiyonunu çağır.
- DrawPoints() fonksiyonunu çağır.
- Nokta sayısı 0 ise "nokta girmediniz" yazdır.
- Nokta sayısı 1 ise merkezi girilen nokta olan 0 yarıçaplı çemberi çiz.
- Nokta sayısı 2 ise çapının iki ucu bu noktalara denk gelen çemberi çiz.
- Nokta sayısı 0, 1, ya da 2 değilse FindCircleFromTwoPoint() fonksiyonunu çağır.
- DrawBezierCurve() fonksiyonunu çağır.
- Bitir.

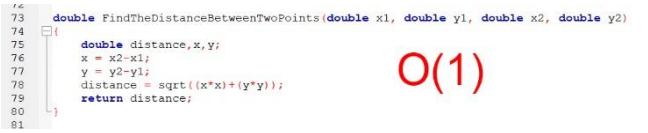
6. KARMAŞIKLIK

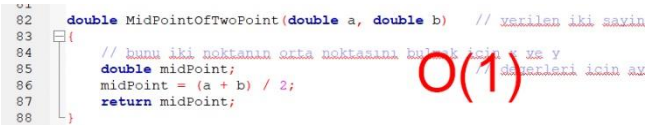


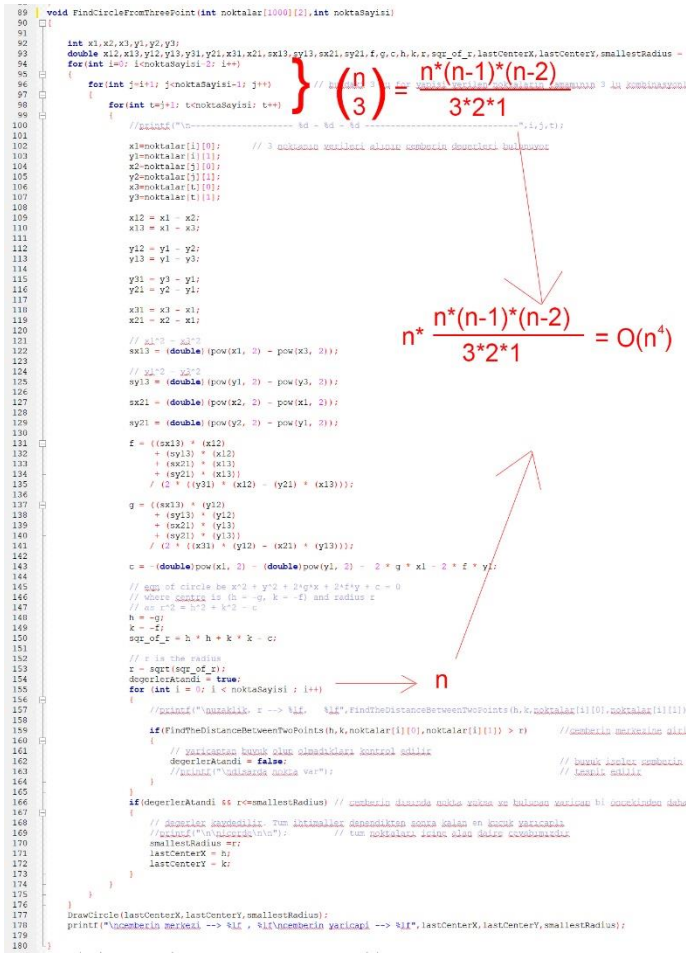


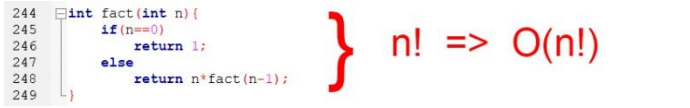


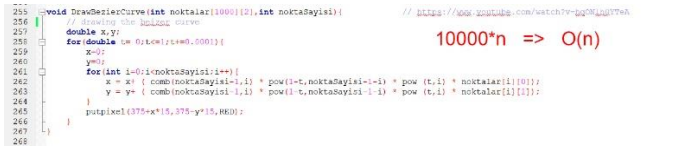












Karmaşıklığı en büyük olan fonksiyon fact() fonksiyonudur. Bu yüzden programın karmaşıklığını bu fonksiyon belirler ve karmaşıklık O(N!)'dir.

7. SONUÇ

Program dosyadan alınan koordinatlara göre en küçük çevreleyen çember ve noktalara en yakın geçen eğri problemini bezier eğrisini ve çember denklemlerinden yararlanarak çözümlemiştir.

8. KAYNAKÇA

- https://www.geeksforgeeks.org/draw-circle-c-graphics/
- https://www.youtube.com/watch?v=hqONlh0YTeA
- https://www.geeksforgeeks.org/setfillstyle-floodfill-c/
- https://www.geeksforgeeks.org/equation-of-circle-when-three-points-on-the-circle-are-given/
- https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1126/materials/cppdoc/graphics.html
- https://tr.wikipedia.org/wiki/B%C3%A9zier_e%C4%9Frisi#:~:text=B%C3%A9zier%20e%C4%9Frisi%2C%20%C3%B6zellikle%20bilgisayar%20grafikleri,yer%20almaz%20(interpolasyon%20e%C4%9Frisi).
- https://en.wikipedia.org/wiki/B%C3%A9zier_curve#:~:text=A%20B%C3%A9zier%20curve%20(%2F%CB%88b,the%20bodywork%20of%20Renault%20cars.