Kocaeli Üniversitesi

Bilgisayar Mühendisliği Bölümü

Programlama Laboratuvarı I

Minimum Enclosing Circle

Berk Sunduri

180201145@kocaeli.edu.tr
berksunduri@gmail.com

Projenin Özeti:

Programlama laboratuvarı I olarak bizden en küçük çember problemini çözmemiz istenmektedir

Ben proje için C programlama dilini ve Code::Blocks geliştirme ortamını seçtim.

Proje dokümanında bizden beklenen üç ayrı istek var. Bunlar: N tane noktayı içeren ve çevreleyen en küçük çemberi çizmek. Çizilecek çemberin merkezi koordinatlarını ve yarıçapını bulmak. Son olarakta tüm noktaların yakınından geçecek B-Spline ya da Bezier Curve Spline'ı oluşturmak

Projede ben C programlama dilinde bulunan "Graphics.h" adlı grafik kütüphanesinden yararlandım. Bu kütüphane sayesinde bizden istenen MEC(Minimum Enclosing Circle) ve Bezier Curve'ü çizdim.

1.GİRİŞ

Proje için C programlama dili ve Code::Blocks geliştirme ortamını kullandım.

C programlama dili; genel amaçlı, sözcüksel değişken kapsamını ve özyinelemeyi destekleyen bir programlama dilidir.

Code::Blocks platformu; içerisinde C,C++ ve Fortran yazabileceğiniz ücretsiz bir entegre geliştirme ortamıdır(IDE). Code::Blocks esnek bir IDE'dir ve kişiye özel bir sürü konfigürasyon sunar.

2. TEMEL BİLGİLER

Projemde;Noktaları kaydetmek için kullandığım Struct ve ana kodun çalıştığı main kısmını saymazsak UpperFactorial, Factorial, Bezier_X, Bezier_Y, detA, detD, detE, detF adında 8 tane fonksiyon bulunmaktadır.

Struct noktalar:
Oluşturduğum struct yapısı X ve Y noktalarının değerlerini bir dizi yardımıyla saklıyor.

upperFactorial(int noktaSayısı,int sayı)
Bu fonksiyon üssüyle birlikte yollanan sayının üssünün faktöriyelini alır. Bezier Curve hesaplamasında gerekli.

factorial(int sayı)
Bu fonksiyon yollanan sayının faktöriyelini alır. Bezier Curve hesaplamasında gerekli.

bezier_X(int index, struct nokta A[],float t)
Bu fonksiyon internetten bulduğum formüllerle yollanan integer olan noktanın indexi, struct ile yollanan noktanın bilgileri ve main kısmında çalışacak bir for döngüsünde her döngüde 0.0001 arttırarak

$$x(u) = (1-u)^3 x_0 + 3u^1(1-u)^2 x_1 + 3(1-u)^1 u^2 x_2 + u^3 x_3$$

fonksiyonunu hesaplar.

bezier_Y(int index, struct nokta A[],float t)
Aynı bezier_X gibi çalışır fakat burda ki hesaplanacak fonksiyon altta kidir.

$$y(u) = (1-u)^3 y_0 + 3u^1(1-u)^2 y_1 + 3(1-u)^1 u^2 y_2 + u^3 y_3$$

detA(struct nokta a,struct nokta j,struct nokta k)
Bu fonksiyon altta ki matrisin determinantını hesaplar. Çizilecek çemberin X,Y koordinatlarını ve yarıçapını bulmakta işimize yarar.

$$a = \begin{vmatrix} x_i & y_i & 1 \\ x_j & y_j & 1 \\ x_k & y_k & 1 \end{vmatrix}$$

detD(struct nokta a,struct nokta j,struct nokta k)
Bu fonksiyon altta ki matrisin determinantını hesaplar. Çizilecek çemberin X,Y koordinatlarını ve yarıçapını bulmakta işimize yarar.

$$d = \begin{vmatrix} x_i^2 + y_i^2 & y_i & 1 \\ x_j^2 + y_j^2 & y_j & 1 \\ x_k^2 + y_k^2 & y_k & 1 \end{vmatrix}$$

detE(struct nokta a,struct nokta j,struct nokta k)
Bu fonksiyon altta ki matrisin determinantını hesaplar. Çizilecek çemberin X,Y koordinatlarını ve yarıçapını bulmakta işimize yarar.

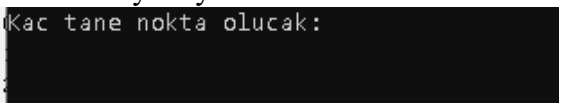
$$e = \begin{vmatrix} x_i^2 + y_i^2 & x_i & 1 \\ x_j^2 + y_j^2 & x_j & 1 \\ x_k^2 + y_k^2 & x_k & 1 \end{vmatrix}$$

detF(struct nokta a,struct nokta j,struct nokta k)
Bu fonksiyon altta ki matrisin determinantını hesaplar. Çizilecek çemberin X,Y koordinatlarını ve yarıçapını bulmakta işimize yarar.

$$f = \begin{vmatrix} x_i^2 + y_i^2 & x_i & y_i \\ x_j^2 + y_j^2 & x_j & y_j \\ x_k^2 + y_k^2 & x_k & y_k \end{vmatrix}$$

3.DENEYSEL SONUÇLAR:

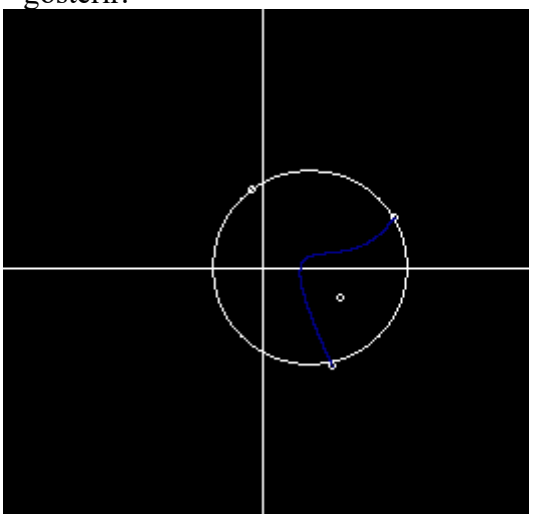
Program başlatıldığında kaç tane nokta için işlem yapıcağımızı girmeliyiz. Bunun amacı sonra ki kısımlarda işlem yapacağım dizilerde yer ayrımıdır.



Kaç nokta olduğunu öğrendiğimiz zaman
program otomatik olarak kendi dosyası içinde olan noktalar.txt'den nokta bilgilerini okur. Noktaları structa aktarır ve en büyük ve en küçük nokta hesaplarını yapar. Bunun yanı sıra yukarda bahsettiğim A,D,E,F nın determinantlarını hesaplayıp bize çizilecek olan çember hakkında bilgi verir.

```
Noktalar:
X:386 Y:215
X:359 Y:255
X:315 Y:201
X:355 Y:289

Kac nokta var:4
Large X:386,215
Small X:315,201
Large Y:355,289
Small Y:315,201
a:-5688

d:-3913480

e:2740520

f:989398032

Cember Merkezi X:344

Cember Merkezi Y:240

Cember Yaricapi:49.34
```

Son olarak program çizilen çemberi ve Bezier eğrisini grafiğe dökerek bize gösterir.



Kaba Kod:

1.Var olan dosyayı aç ve kontrolünü yap.
2.İlk olarak kullanıcıdan array boyutunu ayarlamak için kaç tane nokta olacağını öğren.
3. Dosyadan nokta bilgilerini oku ve structa yazdır.
4. Okunan noktalara göre en büyük X ve en küçük X değerlerine sahip noktaları bul.
5. Okunan noktalara göre en büyük Y ve en küçük Y değerlerine sahip noktaları bul.
6. Bu noktaların structta ki indexini işaretle.
7. Sonuçları kullanıcıya döndür.
8. initGraph() ile bir grafik oluştur ve koordinat sistemini çiz.
9. MEC hesaplamalarında kullanılacak değerlerin hesaplamalarını yap.
10. Yaptığın hesaplamalar sonucu elde ettiğin değerleri circle(x,y,yarıçap) fonksiyonunda kullan. Ve MEC'i elde et.
11. Bezier eğrisini çizdirmek için bir döngü kur ve putpixel fonksiyonuyla döngü sayesinde Bezier eğrisini çizdir.
12. Kullanılan dosyayı kapat.
13.Programı sonlandır.

4.Karşılaşılan Sıkıntılar ve Çözümler:

1.Çember hesaplamaları

En büyük sıkıntım çemberin merkez noktasını bulmaktı. İnternete bakmadan önce geçen yıl verilen ProLab I dersinin ilk projesinde kullandığım birbirine en uzak noktaları bulma fonksiyonunu bu programa entegre etmeye çalıştım. Fakat hesaplamalarda çok yanlışlıklar vardı. Daha sonra internetten yaptığım araştırmalar sonucu Princeton Üniversitesinin yayınladığı Where to Build a School (or Detonate a Bomb) projesine rastladım. Ve hesaplamalarımı altta gösterdiğim formüllere göre yaptım. Fakat

yinede en uzak noktaları bulmak için Brute-Force bir algoritma kullanmak zorunda kaldım.

2.Bezier Hesaplamaları

İlk olarak Bezier'i internette bulduğum kaynaklardan sadece 4 kontrol noktası için olan bir formülle çizdiriyordum. Fakat eğer nokta sayısı daha yüksek olduğunda hesaplamak çok zor oluyordu. Sonradan altta gösterdiğim formülü hesaplamak için programa ekstradan 4 fonksiyon eklemek zorunda kaldım. Sonuç olarak istediğim kadar nokta girebiliyor ve Bezier eğrisini çok daha hızlı hesaplayabiliyordum.

$$B_i^n(u) = \binom{n}{i}(1-u)^{n-i}u^i$$

5.Sonuç:

Bu proje sayesinde C'ye olan hakimiyetimi ve C'nin grafik kütüphanesine olan hakimiyetimi geliştirdim. Sonuç olarak böyle hesaplama projelerinin ne büyük zorluklarla yapıldığını anladım. Yazdığım kodda MEC bulma algoritması O(n) ve Bezier Curve Algoritması $O(n^2)$ karmaşıklığında çalışır. MEC bulmada sadece tek for döngüsü, Bezier Curve da ise 2 ayrı for döngüsü kullanılıyor.

6.Kaynakça:

GeeksforGeeks- Cubic Bezier Curve Implementation in C. (https://www.geeksforgeeks.org/cubic-bezier-curve-implementation-in-c/)

Princeton University-Where to Build a School (or Detonate a Bomb). (https://www.cs.princeton.edu/courses/archive/spring09/cos226/assignments/circle.html)

Colorado University- BGI Documentation. (https://www.cs.colorado.edu/~main/cs1300/doc/bgi/)

JavaScript Info-Bezier Curve(https://javascript.info/bezier-curve)

Lee Mac Programming-MEC(http://www.lee-mac.com/mecfunction.html)

Wikipedia-Smallest Enclosing Circle Problem(https://en.wikipedia.org/wiki/Smallest-circle_problem)