KOCAELİ ÜNİVERSİTESİ Bilgisayar Mühendisliği Bölümü

PROGRAMLAMA LAB. I- I. Proje Minimum Enclosing Circle(MEC)-B Spline

Fatih Güner - 190201074

Projenin Özeti: 1. GİRİŞ

Dönemin 1. projesi olarak bizden istenen bu proje ‘en küçük çember’ ve ‘spline’ adı altında iki bölümden oluşmaktadır.

Öncelikle kullanıcı tarafından verilen ‘.txt’ dosyası içindeki koordinatları okuyarak bunları ekrana yazdırmamız, sonrasında kendi algoritmamızı kullanarak bu noktalardan geçen en küçük alana sahip çemberi oluşturmamız isteniyor. İkinci kısımda ise bu noktalardan geçen eğriyi oluşturmamız gerekli.

Noktaları arayüz üstünde oluşturacağımız koordinat düzlemine uygun olarak yerleştirmeli ve bulduğumuz çemberin merkezi ile yarıçapını ekranda göstermeliyiz.

Ben bu projeyi oluştururken Devc++ geliştirme ortamını kullandım. Arayüz tasarımı için ‘Graphics.h’ kütüphanesini tercih ettim. ‘Graphics.h’ kütüphanesini seçmemin nedeni eski bir kütüphane olmasına rağmen sadeliği ve pratik oluşu ile bana kolaylık sağlıyor olması.

2. YÖNTEM

Adım 1:

Kodlamaya .txt dosyasındaki koordinat bilgilerini ‘Struct Coords’ ile tanımladığım diziye aktararak başladım.

Struct Coords { int x; int y; };

.txt dosyamın düzeni ‘ 1,2 14,-5’ Şeklinde olup her boşluk tuşundan sonra bir sonraki dizi elemanının x ve y koordinatlarını tutmaktadır.

Adım 2:

Ekrana koordinat sistemini çizdirdim ve üzerine noktaları yerleştirdim.

Adım 3:

Minimum çevreleyen çemberi sırası ile şu işlemleri uygulayarak buldum:

1-Birbirine en uzak iki noktayı bul.(n1,n2)

for(i=0;i<N;i++){ for(h=0;h<N;h++){

if(uzaklikbul(i,h)>tut) tut=uzaklikbul(i,h);

}}

2-Bu iki noktanın merkezini bul.(m) m.x=(n1.x+n2.x)/2; m.y=(n1.y+n2.y)/2;

3-Merkeze en uzak noktayı tespit et.(n3)

for(i=0;i<N;i++){ if(uzaklikbul(i,m)>tut)

tut=uzaklikbul(i,m);}

5- Eğer çemberimiz 2 nokta tarafından tanımlanacaksa çemberi çok basit bir matematik kullanarak oluşturabiliriz. Bu durumda çemberin merkezi Adım3.2’de belirlediğimiz m olacaktır ve yarıçap ise n1 ile m arasındaki mesafe olacaktır.

Fakat çember 3 nokta tarafından oluşturulacaksa uzun denklemler işin içine giriyor. Böyle bir durumda çemberi belirleyebilmek için çember denklemini kullanmaya karar verdim.

**x²** **+** **y²** **+** **2gx** **+** **2fy** **+** **c** **=** **0**

Bu denklemi her üç nokta için yazıp üç ayrı denklem elde ediyoruz. Sonrasında denklemleri birbirine ekleme ve çıkarma işlemleri yaparak g ve f değerlerini buluyoruz. g ve f değerleri merkezin sırasıyla x ve y koordinatlarını verecektir.

Yarıçapı bulmak için ise **(x** **–** **h)²** **+** **(y** **-** **k)²** **=** **r²** denklemini kullanıyoruz.

4-Üçgeni oluşturacak noktaları belirle

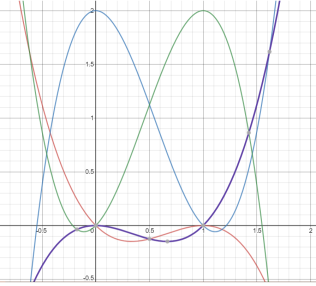
if(uzaklikbul(n3,m)<uzaklikbul(n1,m)) ucgenciz(n1,n2);

else ucgenciz(n1,n2,n3);

Adım 4:

En küçük çemberi oluşturduğumuza göre artık sıra eğrimizi çizdirmede.

Eğriyi çizdirmek için yeni bir fonksiyon oluşturuyoruz. T değişkenimizi 0’dan başlatarak çok minik artışlarla arttırdığımız bir döngünün içinde fonksiyona noktaların koordinatlarını ve t değerimizi yolluyoruz.

Bu denklemlerin kullanılması tesadüfi değildir ve dikkatle seçilmelilerdir.

for (float t = 0; t < i; t += 0.005) {

tut=splinecoords(t,&noktalar[0]); çemberçiz(tut);

}

splinecoords fonksiyonunda x=t olacak şekilde

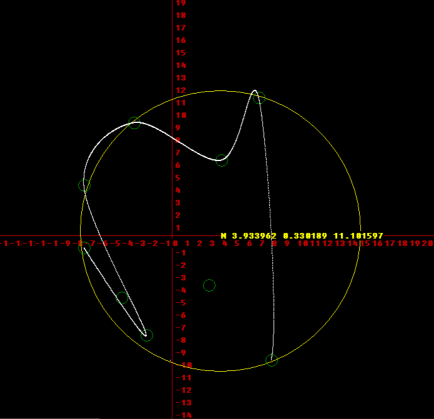
-x^{3}+2x^{2}-x \*p0 1 3x^{3}-5x^{2}+2 \*p1 2 -3x^{3}+4x^{2}+x \*p2 3 x^{3}-x^{2} \*p3 4

denklemler eğrimizi belirleyecek olan 4 nokta ile sırasıyla çarpılır. (p0,p1,p2,p3)

2 ve 3. denklemler birbirinin tersi şeklinde davranıyor. Bu iki denklem x=0 ve x=1 aralığındaki büyük değerleri ile eğrinin iki ucunu oluşturuyor iken 1 ve 4 denklemleri küçük değerleri ile eğrinin kendilerinden uzaklaşıp yakınlaşmasını sağlıyorlar.

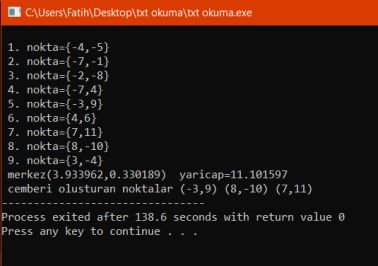
Sonuç olarak bu dört denklemi topladığımızda bulduğumuz değeri eğriyi çizdirmek için kullanıyoruz.

Deneysel Sonuçlar :

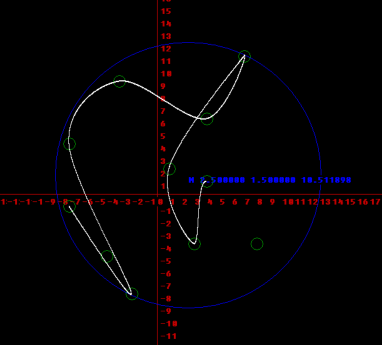
Zaman karmaşıklığı : Algoritmamızın zaman

karmaşıklığı noktalar arası uzaklığı hesaplarken kullandığımız iç içe iki döngü tarafından belirlenir. Burada n\*n adet işlem yapıldığı için o(n^2) olur.

Sonuç :

Sonuç olarak noktalarımızı ekrana yazdırarak en küçük çemberi ve eğrimizi bulmuş olduk. Deneysel sonuçlardan da görüldüğü üzere kodumuz düzgün bir şekilde çalışmaktadır.

Bu proje sayesinde arayüz tasarımının nasıl yapıldığını ve ağır matematiksel denklemlerin koda nasıl dönüştürüldüğünü öğrendim.

Kaynaklar : <https://en.wikipedia.org/wiki/B-spline>

[https://matkafasi.com/83144/cemb](https://matkafasi.com/83144/cemberin-denklemi) [erin-denklemi](https://matkafasi.com/83144/cemberin-denklemi)

[https://www.codeproject.com/Articl](https://www.codeproject.com/Articles/1165267/Coding-Challenge-Smallest-Circle-Problem) [es/1165267/Coding-Challenge-Smallest-Circle-Problem](https://www.codeproject.com/Articles/1165267/Coding-Challenge-Smallest-Circle-Problem)

[https://www.desmos.com/calculato](https://www.desmos.com/calculator?lang=tr) [r?lang=tr](https://www.desmos.com/calculator?lang=tr)