Karaelmas Fen ve Mühendislik Dergisi
Dergi web sayfası: http://fbd.beun.edu.tr

Araştırma Makalesi

C# Kullanarak Mesafeye Bağlı Ters Ağırlık Yöntemi ile Gridleme ve Kontur Çizimi

Gridding and Contouring with Inverse Distance Weight Method by C#

İrfan Duman[1*], Resul Kara[2], Erkan Çetiner[3]

[1]Bülent Ecevit Üniversitesi, Bilgi İşlem Daire Başkanlığı, Zonguldak, Türkiye
[2]Düzce Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Düzce, Türkiye
[3]Bülent Ecevit Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümü, Zonguldak, Türkiye

Öz

Bu çalışmada, herhangi bir alanda yapılacak çalışma için toplanan ham verileri ön işlemlerden geçirip konturlama metodu ile gösterelebilmek için bir yazılım geliştirilmiştir. Toplanan XYZ verileri ön işlemden geçirilirken, öncelikle bir koordinat ağı ya da veri düzlem ağı dediğimiz grid oluşturulur. Grid noktasının her bir düğüm noktası için bir XYZ verisi atanmış ve gridleme işlemi tamamlanmıştır. Veriler ön işlemden geçirilip gridleme tamamlandıktan sonra ise oluşturulan grid üzerine konturlama yapılarak toplanan verilerin kontur haritaları çıkartılmış olur.

Anahtar Kelimeler: Konturlama, Gridleme, C#, CSharp, Mesafenin tersi

Abstract

In this study, a software has been developed to contour pre-processe draw data to be used in any field of studied area. During the pre-processing of collected XYZ raw data, a grid system which is also called mesh data is formed. For all of the nodes on the grid, the XYZ data is attached and the gridding process is completed. After pre-processing all the data, a gridding is completed and a contouring process is conducted on this newly formed grid to generate contour maps of collected XYZ raw data.

Keywords: Contouring, Gridding, C#, CSharp

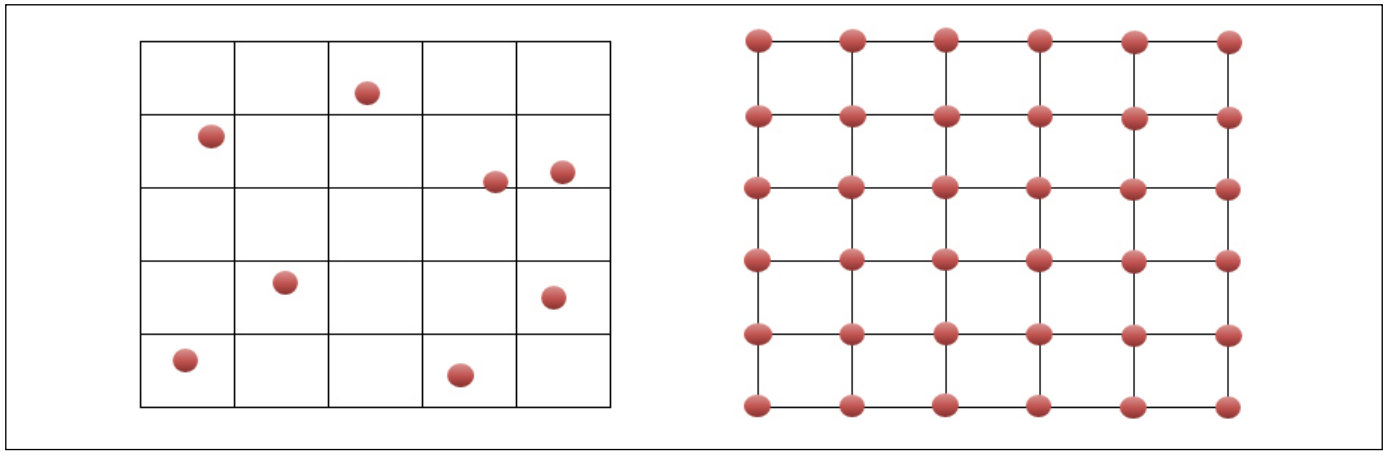
1. Giriş

İnsanlar hayatlarında, cevabını ya da nedenini merak ettiği şeyler ile karşılaşır ve bunun cevabını ararken çevrelerinden veri toplayarak bulurlar. Bazı soruların cevabını tek bir veri cevaplayabilirken bazen tek bir veri bu sorunun cevabını karşılayamamakta, elde edilen birden fazla veriyle istenilen cevaba ulaşılabilmektedir. Araştırmalar sonucunda elde edilen çok fazla miktardaki veriyi daha kolay anlaşılır ve yorumlanabilir hale getirmek için grafikler kullanılır. Kontur grafikleri ya da kontur çizgileri de var olan veya ölçüm yapılarak elde edilen veri setlerindeki verilerin hangi değerler etrafında yoğunlaştığını gösterir. Konturlamanın en çok kullanıldığı yerlerden birisi de topografya haritalarıdır. Topografya haritalarında kontur çizgileri eşyükselti eğrileri olarak yorumlanır. Eşyükselti eğrileri topografya haritalarında kullanılan en yaygın ve en gelişmiş yöntemdir. Eşyükselti eğrileri, adından da anlaşıldığı gibi, belli bir düzeyden (deniz düzeyi) başlayarak, topografyanın eşyükselti noktalarını birleştiren eğrilerdir. Eş yükselti eğrileri aynı zamanda yükselti dağılım ve değişimini de gösterir (Emre 2003). Yeraltı haritaları oluşturulurken yapılan sismik, manyometrik, gravimetrik vb. ölçümlerle elde edilen verilerin analizi sonrasında sahadaki veri değişimleri kontur çizgileri yardımıyla gösterilerek jeofizik haritaları hazırlanır (Dirik 2013).

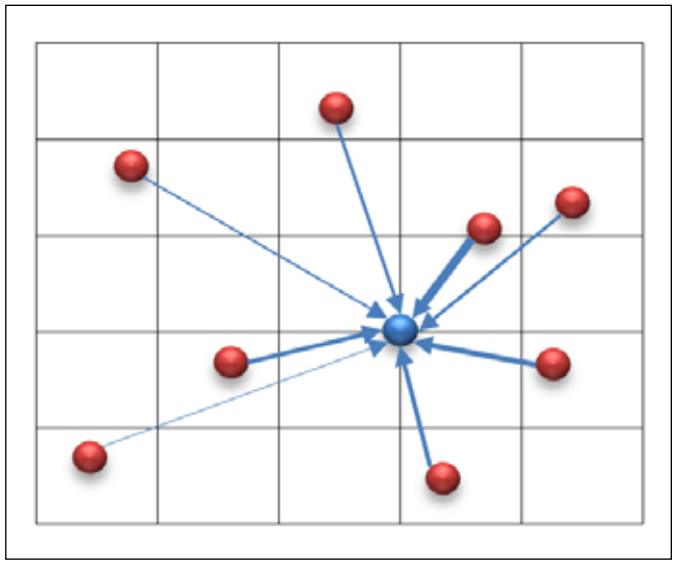
Birçok alanda kullanılan konturlama metodunun genel amacı ise çok sayıda toplanan veriyi değerlendirirken daha kolay ve anlaşılabilir hale getirmektir. Toplanan verilerin ortak noktası ise X, Y konumlarına ya da bir koordinat ağındaki X,Y koordinatlarına bağlı spesifik Z verileridir. Z verileri çalışma alanına göre o konumdaki sıcaklık, basınç, yükseklik, nem, gerilim ya da bir sondaj alanındaki madenin o konumdaki yoğunluğu olabilir. Toplanan çok sayıdaki veri setlerini ön işlemden geçirip gösterimini yapabilmek için bilgisayar yazılımlarına ihtiyaç duyulur. Veri setleri ve

*Sorumlu yazarın e-posta adresi: irfanduman@beun.edu.tr

Geliş tarihi / Received : 27.01.2016
Kabul tarihi / Accepted : 29.01.2016



Şekil 1. Bilinen noktalardan yararlanılarak bilinmeyen noktalar üretilerek çalışma alanının gridlenmesi.



Şekil 2. Ölçüm noktalarının grid düğüm noktalarına uzaklığa bağlı etkileri.

gösterimleri aynı olmasına rağmen her bir çalışma alanı için kullanılan yazılımlar farklılık göstermektedir.

2. Gereç ve Yöntem

Bu kısımda gridleme ve mesafenin tersi yöntemi hakkında bilgi verilmiş C# programlama dili kullanılarak geliştirilen uygulama yardımı ile örnek veriler üzerinde mesafeye bağlı ters ağırlık yöntemi uygulanarak gridleme yapılmış, elde edilen yeni veri seti kullanılarak kontur haritası oluşturulmuştur.

2.1. Gridleme

Araştırmalarda veriler toplanırken her zaman istenilen noktalardan düzenli veriler elde etmek mümkün olmayabilir. Örneğin; jeofiziksel bir harita için manyetik ya da sismik verilerin her 10 metrede bir toplanması hedeflenirken ölçüm yapılacak noktalara yerleştirilecek ölçüm cihazları arazi koşulları nedeniyle yerleştirilemeyebilir. Bu gibi durumlarda arazinin belirli noktalarında yapılan ölçümler arazinin ölçüm yapılamayan noktalarına tahmini değer atamak için yardımcı olacaktır. Bir çalışma alanının belirli noktalarından alınan ölçüm bilgilerinin, alanın ölçüm alınmayan diğer noktalarına tahmini değer atama işlemine enterpolasyon denir. Enterpolasyon yapılırken toplanan üç boyutlu veriler aracılığı ile çalışma alanının tamamı gridlenir (Şekil 1). Grid ifadesi ızgara olarak da kullanılmakta ve kısaca gridleme, alanın dikdörtgenlere bölünüp köşe noktalarına veri atanmasıdır.

Grid çalışma alanının tamamını kapsayacak şekilde olmalıdır. Çalışma alanı bir sondaj alanı ise sondaj noktalarının tamamını, bir sıcaklık dağılım haritası çıkarılacak ise sıcaklık ölçümlerinin yapıldığı alanın tamamını kapsayacak şekilde gridleme yapılmalıdır. Gridleme yapılırken çeşitli enterpolasyon ve jeoistatistik yöntemleri kullanılmaktadır. Bu çalışmada, mesafeye bağlı ters ağırlık yöntemi kullanılmıştır.

2.2. Mesafeye Bağlı Ters Ağırlık Yöntemi

Bu yöntem çalışma alanındaki bilinen noktalardan bilinmeyen grid noktalarına veri elde etmek amacıyla kullanılır. Bilinmeyen bir grid noktasının değeri tahmin edilirken, bilinen tüm veya belirlenen bir alan (etki alanı) içindeki ölçüm noktalarının değerlerinin ardışık olarak o noktaya etkisi hesaplanır. Hesaplamada grid noktasına en yakın olan ölçüm noktasının değerinin etkisi o noktaya en fazla olurken tahmin edilecek grid noktasından uzaklaştıkça hesaba katılacak ölçüm noktalarının ağırlığı azalır (Yang, C. S.2004).

Mesafeye bağlı ters ağırlık yöntemi, (Şekil 2)'deki gibi gridleme yapılırken, grid noktasındaki tahmin edilecek değerin

hesaplanabilmesi için çevredeki bilinen ölçüm noktalarının değerini kullanan bir enterpolasyon yöntemidir.

Tahmin edilecek grid noktasını belirleme işlemi (2.2.1)'de verilen eşitlikle gerçekleştirilir.

$$Z_k = \frac{\sum_{i=1}^{n}\left(\frac{Z_i}{D_{ik}}\right)}{\sum_{n=1}^{n}\left(\frac{1}{D_{ik}}\right)} \quad (2.2.1)$$

Burada, Z_k grid düğüm (dikdörtgen köşe) noktasına atanan değeri belirtirken Z_i ise bilinen değerleri belirtmektedir. Dik mesafesi ise (2.2.2)'de verilen eşitlikle hesaplanır.

$$D_{ik} = \sqrt{(X_k - X_i)^2 + (Y_k - Y_i)^2} \quad (2.2.2)$$

Eşitlikte, X_k, $Y_k$grid düğüm noktasının koordinatlarını, X_i, Y_i değeri bilinen noktanın noktaların koordinatlarını belirtir. D_{ik} tahmin edilecek grid noktasının, ölçüm değeri bilinen her bir noktaya olan doğrusal uzaklığını belirtir.

Her bir düğüm noktasının değerlerini hesaplayan algoritmanın C# programlama dili ile kodlanmış hali aşağıdaki gibidir;

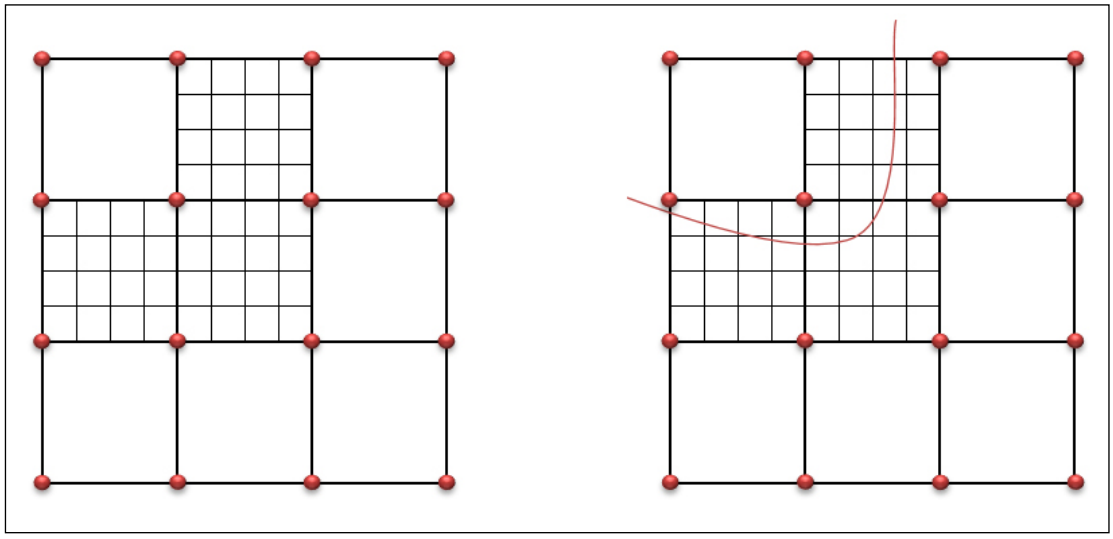
```
private double dugum_noktasi_degeri(double dugum_
noktasi_X, double dugum_noktasi_Y)
    {
    double ilk_toplam = 0, ikinci_toplam = 0;
    KonturlamaEntities _db = new KonturlamaEntities();
    var veri_noktasi_sayisi = _db.Veris.Count(i=> i.KayitId
    == pKayitId);
    var veri_noktalari = _db.Veris.Where(i => i.Id>= 0 &&
    i.KayitId == pKayitId);
    foreach (var item in veri_noktalari)
    {
    ilk_toplam += Convert.ToDouble(item.Z) /
    dugum_veri_noktasi_arasi_mesafe(dugum_noktasi_X,
    dugum_noktasi_Y,
    Convert.ToDouble(item.X),
    Convert.ToDouble(item.Y));
    }
    foreach (var item in veri_noktalari)
    {
    ikinci_toplam += 1 /
    dugum_veri_noktasi_arasi_mesafe(dugum_noktasi_X,
    dugum_noktasi_Y,
    Convert.ToDouble(item.X),
    Convert.ToDouble(item.Y));
    }
    return ilk_toplam / ikinci_toplam;
    }
    private double dugum_veri_noktasi_arasi_mesafe(double
    dugum_noktasi_X,
    double dugum_noktasi_Y, double veri_noktasi_X,
    double veri_noktasi_Y)
    {
    double mesafe;
    mesafe = Math.Sqrt(Math.Pow(dugum_noktasi_X -
    veri_noktasi_X, 2)
    + Math.Pow(dugum_noktasi_Y - veri_noktasi_Y, 2));
    return mesafe;
    }
```

2.3. Konturlama

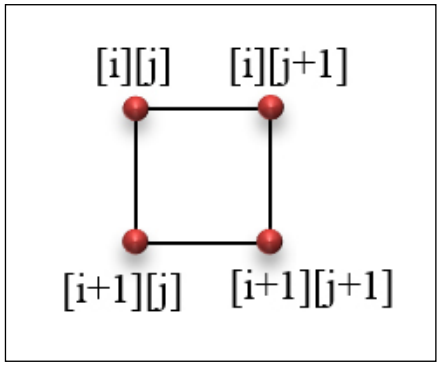
Kontur çizgileri, eş yükselti eğrileri gibi kullanıldığı alanlara göre adlandırılırlar. Kontur çizgileri, bir doğruyu oluşturan sonsuz noktalar gibi eğriler olarak adlandırılıp, eğriler olarak görülse de aslında küçük çizgi parçacıklarından oluşurlar. Ancak kontur çizgisini oluşturan her bir çizgi parçacığı eğer özel bir yumuşatma (smooting) formülü kullanılmıyor ise düzdür (Şekil 3).

Kontur çizgileri çizilirken gridleme sırasında grid düğüm noktalarına atanan veri değerleri kullanılır. İki boyutlu dizi olarak gridlenmiş her XY ikilisi için bir Z değeri enterpolasyon yöntemiyle atanmıştır (Şekil 1). Bu değerlerden XY grid üzerindeki koordinatları, Z değeri ise grid düğüm noktasının değerini belirtir. Kontur çizimine başlandığında gridlenmiş alanlar küçük dikdörtgenlere ayrılır. Her bir dikdörtgen dizi şeklinde değerlendirilerek tek tek Z değerine göre taranır (Şekil 4).

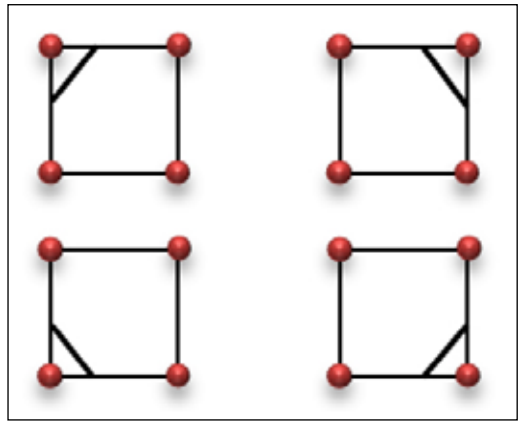
Kontur çizgisinin işlem sırasındaki dikdörtgenden geçip geçmeyeceği kontur çizgisinin taşıdığı değer ve işlem sırasındaki dikdörtgenin kenarlarının uç noktalarındaki değerlere bağlıdır. Biri kontur değerinden büyük ve diğeri küçük ise çizgi bu kenarı kesecektir. Kesme noktası iki uçtaki değerlere kontur değeri orantılanarak belirlenir. Kontur çizgisinin değeri ile dikdörtgen köşesindeki Z değerleri tek tek



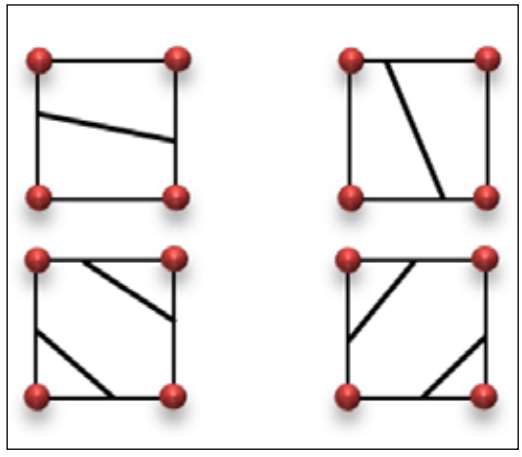
Şekil 3. Kontur eğrisini oluşturan düz çizgi parçacıkları.



Şekil 4. İki boyutlu diziden küçük dikdörtgenler elde edilmesi.



Şekil 5. Kontur çizgisinin geçeceği kenarların belirlenmesi.



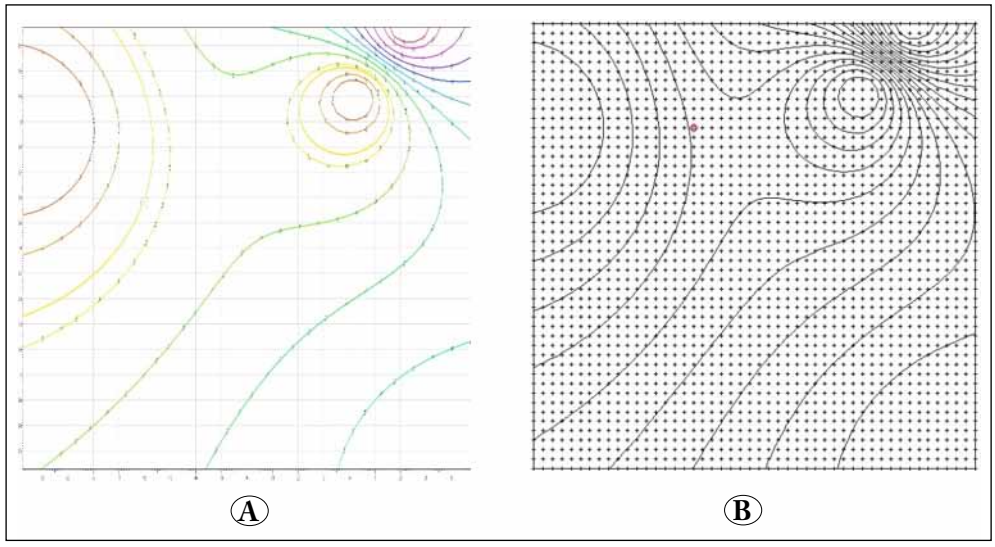
Şekil 6. Eş değer köşelerin kontur çizgisiyle ayrılması.

karşılaştırılır. Dikdörtgenin bir kenarı kontur ile temas etmiş ise mutlaka başka bir kenarı da temas etmelidir. Diğer kenar da aynı şekilde bulununca bu iki nokta bir çizgi ile birleştirilir (Şekil 5).

İşlem yapılan dikdörtgende her zaman tek köşe noktası kontur çizgisinin değerinden büyük olmayabilir. Eğer (Şekil 5)'den farklı olarak kontur çizgisinin değerinden büyük iki değer var ise bu iki değeri ayıracak şekilde kontur çizgisi çekilir. Kontur çizgisinin değerinden büyük olan tek bir grid noktası var ise (Şekil 5)'de olduğu gibi o köşe yalnız bırakılacak şekilde ayrılır. Eğer köşe noktalarından ikisi büyük ise (Şekil 6)'da olduğu gibi büyük olan iki köşe dikdörtgenden ayrılır.

Grid düğüm noktalarının üçünün de değeri kontur çizgisinden büyük ise üç köşe diğer köşeden kontur çizgisi ile kesilerek ayrıldığında (Şekil 5)'deki durum oluşur. Bu durumların hiç birisi gerçekleşmezse, yani dikdörtgenin düğüm noktalarının hepsi kontur çizgisinin değerinden büyük ise bu dikdörtgenden kontur çizgisinin geçmeyeceği anlaşılır.

Bu işlem basamakları, (Şekil 1)'deki gibi gridlenmiş alanın dikdörtgenlere ayrılması ve her bir dikdörtgenin tek tek bu işlem basamaklarından geçirilmesiyle tekrarlanır. Tekrarlanan bu işlemler sonucunda işlem gören her dikdörtgen yan yana geldiğinde birleşen kontur çizgileri kontur eğrilerini, kontur eğrileri de kontur grafiklerini ya da haritalarını mey-



Şekil 7. A) Geliştirilen uygulama ile oluşturulmuş kontur haritası.
B) Surfer yazılımı çıktısı.

dana getirmektedir. Dolayısı ile çok sayıda grid daha çok işlem ve daha yüksek çözünürlük sağlayacaktır.

2.4. Gridleme ve Konturlama Yazılımı Geliştirilmesi

Bilimsel, matematik ve mühendislik alanlarında rapor ve sunumlarda grafiksel verilere oldukça fazla ihtiyaç vardır. Yapılan çalışmalara eklenen tablo ve grafikler sayesinde raporlar görsel ve en önemlisi anlaşılması kolay hale getirilebilir. Microsoft'un nesneye dayalı C# programlama dili kullanılarak hem hesaplamalar yapılıp veri setleri üretilebilir, hem de üretilen bu veri setlerinden Grafiksel Cihaz Arayüzü (Graphics Device Interface-GDI) kütüphanesi kullanılarak tablo ve grafikler oluşturulabilir.

C# programlama dili, temel programlama yeteneklerine sahip kullanıcıların gelişmiş grafik ve tablolar üretmesini sağlayan çok yönlü esnek bir Grafik Kullanıcı Arayüzüne (GUI) sahiptir. C# ile geliştirilebilecek grafik ve tabloların gelişmişlik ve karmaşıklık düzeyi programcının ihtiyacı ve hayal gücüyle sınırlıdır.

Bu çalışmada geliştirilen uygulama yazılımı, var olan XYZ formatında veri setini istenilen boyutta gridleyebilmektedir. Hesaplamalar C#'ın Math kütüphanesi içerisindeki fonksiyonları aracılığıyla yapılmıştır. Hesaplamalar sonucunda elde edilen veriler C#'ın GDI kütüphanesi kullanılarak konturlama yapılıp kontur haritası oluşturulmuştur.

3. Bulgular ve Tartışma

Çalışmada C# programlama dili kullanılarak geliştirilen uygulama sonucunda elde edilen veriler, Golden Software firması tarafından piyasaya sürülen Surfer yazılımı ile karşılaştırılmıştır. Şekil 7a'da C# programı kullanılarak geliştirilen uygulama aracılığıyla oluşturulan kontur haritası, Şekil 7b'de ise aynı veri setleri kullanılarak Surfer uygulamasında üretilen kontur haritası yer almaktadır.

Şekillerde de görüldüğü üzere geliştirilen uygulama ile Surfer uygulamasının oluşturduğu kontur haritası benzerlik göstermektedir. Bu benzerliğe dayanarak kullanılan enterpolasyon yönteminin hesaplamalarının doğru yapıldığı ve çizdirilen kontur haritasının da buna bağlı olarak doğru bir şekilde oluşturulduğu sonucuna varılabilir.

4. Sonuç

Örnek veri seti üzerinde yapılan çalışmada C# programlama dili kullanılarak, mesafeye bağlı ters ağırlık yöntemi ile gridleme işlemi yapılmış ve elde edilen yeni veri seti ile C# programlama dilinde GDI kütüphanesi kullanılarak kontur grafiği oluşturulmuştur. Oluşturulan grafik ile Golden Software firmasına ait Surfer programında aynı veri seti kullanılarak oluşturulan kontur grafiği karşılaştırılmış ve hesaplama sonuçlarının doğru olduğu buna bağlı olarak kontur grafiğinin de doğru bir şekilde oluşturulduğu görülmüştür.

Geliştirilen yazılım, Golden Software firmasının Surfer yazılımı gibi ticari amaçla üretilmemiştir. Geliştirilen yazılım bir çok araştırma alanında elde edilen dağınık verilerin gridlenerek hesaplama ve konturlama amacı ile kullanılabilecektir. Yazılımı geliştirirken kullanılan C# programlama dili sayesinde bir çok mobil ya da web platformlarında da farklı amaçlara yönelik uygulamalar geliştirilebilecektir.

5. Simgeler ve Kısaltmalar

GDI: Graphical device interface; **GUI:** Grafik kullanıcı arayüzü; **C#:** C sharp.

6. Kaynaklar

Dirik, K. 2013. Yeraltı harita çeşitleri, *Hacettepe Üniversitesi Ders Notları*, Ankara.

Emre, T. 2003. Harita çizimi, *Dokuz Eylül Üniversitesi Mühendislik Fakültesi Yayınları Ders Notları*, İzmir.

Krige, D., G. 1951. A statistical approach to some nine valuations and allied problems at the witwatersrand. *Yüksek Lisans Tezi*, Witwatersrand Üniversitesi.

Köroğlu, S. 2006. Farklı Enterpolasyon Yöntemlerinin Hacim Hesabına Etkisinin Araştırılması, *Yüksek Lisans Tezi*, İstanbul Teknik Üniversitesi.

Mcallister, M., Snoeyink, J. 2000. Medial Axis Generalization of River Networks, *Cartography and Geographic Information Science: CAGIS.*, 27: 129-138.

Soycan, A., Soycan, M. 2002. Yol projelerinde sayısal arazi modellerinin kullanılması, *Selçuk Üniversitesi Jeodezi ve Fotogrametri Mühendisliği 30.Yıl Sempozyumu*, Konya.

Yalanak, M. 1997. Sayısal Arazi Modellerinde Hacim Hesaplarında En Uygun Enterpolasyon Yönteminin Araştırılması, *Doktora Tezi*, İTÜ Fen Bilimleri Enstitüsü, İstanbul.

Yalanak, M. 2001. Transformation of Ellipsoid Heights to Local Leveling Heights, *Journal Of Surveying Engineering: ASCE.*, 127: 90-103.

Yang, C. S., Kao, S. P., Lee, F. B., & Hung, P. S. 2004. Twelve different interpolation methods: A case study of Surfer 8.0. In *Proceedings of the XXth ISPRS Congress*., 35: 778-785.