



Uydu görüntülerinden zincir kod yöntemi kullanılarak yarı otomatik yol çıkarımı

Muhammed TEKİN¹, Müfit ÇETİN¹

¹Yalova Üniversitesi, Mühendislik Fakültesi, Bilgisayar Müh. Bl., Yalova

Özet

Gerek navigasyon gerekse de Coğrafi Bilgi Sistemlerinin doğru çalışabilmesi açısından güncel kara yolu haritalarına sahip olmak son yılların önemli araştırma konularından birisidir. Verilerin büyüklüğünden kaynaklı iş yükü dolayısıyla da işlemlerin otomatize edilmesi büyük önem taşımaktadır. Bundan dolayı bu çalışmada uydu görüntüsündeki kara yollarının tespitinde yeni bir yarı otomatik yöntem geliştirilmeye çalışılmıştır. Sistem beş ana aşamadan oluşmaktadır. İlk aşama, uydu görüntüsündeki kenarlıkların ortaya çıkarılma işlemidir. İkinci aşama, zincir kod yönteminin (chaincode) kenar nesnelere uygulanarak her bir kenar nesnesine ait zincir kodun elde edilmesidir. Üçüncü aşama, en uzun ortak alt dizi (longest common subsequence) yöntemi ile zincir kod benzerlikleri karşılaştırılarak birbirine benzer kenarların otomatik tespit edilmesidir. Bu aşama sonunda benzer kenarların arasındaki alanlar olası yol pikselleri olarak işaretlenecektir. Dördüncü aşama, önceki aşamada yol olarak belirlenmiş piksellerin ortalama gri renk değerlerinin hesaplanıp bu değerden küçük piksellerin elimine edilmesidir. Son aşama ise önceki adımda yol olması muhtemel olarak belirlenmiş piksellerin bölge büyütme algoritmasına besleme değeri olarak verilmesi ile gerçek yol piksellerinin belirlenmesidir. Geliştirilen yöntem, yerleşim yeri dışı bölgelerdeki yolları içeren dört farklı uydu görüntüsü üzerinde uygulanmış ve bunların ortalama başarımların değerleri sırasıyla duyarlılık (sensitivity) %88, özgüllük (specifity) %96, kesinlik (accuracy) %94, F1 puanı %86 olarak elde edilmiştir. Söz konusu sonuçlar aynı uydu görüntülerini kullanan literatürdeki bir başka yöntem ile karşılaştırılmış ve önerilen sistemin daha az adım ile yaklaşık aynı sonuçları elde edebildiği görülmüştür

Anahtar Kelimeler: Uydu görüntüleri, yol tespiti, zincir kod, en uzun ortak alt küme, görüntü işleme

Semi automatic road extraction from satellite images by using chaincode method

Abstract

It is one of the most important research subjects of recent years to have current road maps in terms of both navigation and Geographic Information Systems. Because of data size it takes a long time to process and it makes automating this process is very important. Therefore, in this study, a new semi-automatic method has been tried to be developed for the detection of land roads in satellite images. The system consists of five main stages. The first stage is the edge detection in the satellite image. The second step is to obtain the chaincode for each edge by applying chaincode method. The third stage is the automatic detection of similar edges by comparing their chaincodes by longest common subsequence algorithm. At the end of this step

Makale Bilgisi

Başvuru:
04/12/2018
Kabul:
07/05/2019

the areas between the similar edges will be marked as possible road pixels. The fourth stage is to calculate the average gray color values of the pixels specified as the road in the previous stage and eliminate the pixels smaller than this value. The last step is to determine the actual road pixels by giving the pixels as the feed value to the region growing algorithm, which is likely to be the road in the previous step. The developed method was applied on four different satellite images those includes roads in the non-residential areas. Their average performance values were 88%, specificity 96%, accuracy 94%, F1 score 86%, respectively. The results were compared with another method in the literature using the same satellite images, and the proposed system was able to achieve approximately the same results with fewer steps..

Keywords: *Satellite images, road detection, chain code, longest common subsequence, image processing*

1 Giriş

Güncel ve doğru haritalara sahip olma konusu özellikle navigasyon sistemlerinde, şehir planlamada, acil durumlara müdahalelerde önemli bir konudur [1]. Bu işlem için gerekli olan görüntülerin elde edilmesi için 20.yy'ın başlarında uçaklardan faydalanılsa da zamanla uydu teknolojilerinin gelişmesi ile uydular uçakların yerini almıştır. Bununla birlikte her gün uydulardan gelen görüntülerin sayısı insanların kısa bir zamanda işleyemeyeceği kadar çoktur. Bu yüzden söz konusu işlemlerin otomatikleştirilmesi hem zaman hem de iş gücü açısından ciddi faydalar sağlamaktadır. Ancak henüz bu konuda tamamen insandan bağımsız ve güvenilir bir sistem geliştirilememiştir[2]. Bu konuda tam başarımın elde edilememesine neden olan ve problemi karmaşıklaştıran bazı faktörler kısaca şöyle özetlenebilir[3]:

- Uydu görüntülerinin çözünürlüklerinin farklı oluşu (46 cm'den 25 m'ye kadar farklı çözünürlükler)[4]
- Görüntülerdeki bazı bölgelerinin bulutlarla kaplı olması
- Ağaç, ev ve araçlar gibi istenmeyen nesnelere yol görüntüsünü engellemesi
- Yolların uzunluk ve genişliklerinin farklı olması

- Kavşakların ve özellikle döner kavşakların düz yollara göre şekilsel olarak farklı olmaları
- Bazı binaların düz kenarlarının yol şeklinde yanıltıcı olması

Bu çalışmada geliştirilen yöntem elle yapılan güncellemelerdeki hataların ve zamansal kayıpların önüne geçmeyi hedefleyen yarı otomatik bir işlemdir. Bu yöntem, kullanıcının sadece yol genişlik bilgisi girerek gerçekleştirilen çeşitli morfolojik işlemlerin ardından zincir kod ve en uzun ortak alt dizi (longest common subsequence – LCS) algoritmaları ile yolları ortaya çıkaracaktır. Sistemin başarımını daha da iyileştirilmesi için bölge büyütme yöntemi de kullanılacaktır. Gelinmek istenen nokta ise verilen bir uydu görüntüsündeki kara yolların başarıyla tanınabilmesidir. Ancak problemin geniş kapsamlı olmasından dolayı burada kara yolları olarak şehirlerarası ana yollar ile çalışma sınırlandırılmıştır.

2 Literatür araştırması

Uydu görüntülerindeki yolların tespiti için literatürde yarı otomatik ve tam otomatik çalışmalar mevcuttur. Bu bölümde her iki tipteki sistemler için de örnekler verilecektir.

Fateme ve diğ. (2008) [5] yayınlarında bulanık C – Ortalamalar kümeleme analizi yöntemi ile spektral olarak görüntüleri verilen bir eşik değerine göre

sınıflandırmışlardır. Ardından sonuçları daha iyi hale getirmek için de farklı morfolojik işlemler uygulayıp (genişleme, erozyon, açma, kapama vb.) graf teorisi ve en küçük tarama ağacı yöntemlerinin de yardımıyla kara yollarının orta noktalarını %98 doğrulukla tespit etmişlerdir.

Das ve diğ. (2011) [3] yayınlarında olasılıklı destek vektör makinesi yöntemi ile beraber konvüsyonel yapay sinir ağı modelini birlikte kullanmışlardır. Ağın eğitimi için toplamda 12200 adet 21x21 boyutlarında görüntü kullanılmıştır. Bu görüntüler, bir uzman tarafından şehir içi, şehir dışı ve kırsal alan görüntülerinden oluşan 100 adetlik 512x512 çözünürlükteki veri seti içerisinden seçilmiştir. Görüntülerin 7200 tanesi yol olmayan, 5000 tanesi de yol olan alanlardan oluşmaktadır. Geliştirdikleri yöntemi 100 adetlik veri setine uygulayarak ortalama %89 başarımla elde etmişlerdir.

Kirthika ve Mookambiga (2011) [6] ise yapay sinir ağlarından faydalanarak yol tespitine çalışmışlardır. Her bir pikselin farklı spektraldeki değerlerini kullanarak kontrast, enerji, entropi ve homojenlik değerlerini elde etmişler ve yapay sinir ağına öğrenme için kullanmışlardır. Bu şekilde %80 doğrulukla tespit yapabilmışlardır.

Abraham ve Sasikumar (2013) [7] dalgacık tabanlı bir segmentasyon yöntemini bulanık mantık ile birleştirerek % 80 üzerinde doğrulukla yol tespiti yapmışlardır. Ayrıca, geliştirilen sistemde görüntüdeki gürültü etkisini azaltmak için bir önışlem aşamasından faydalanmışlardır.

Daha yeni çalışmalardan olan Alshehhi ve Marpu (2017) [8]'ün geliştirdikleri sistemde, yollar ile yol olmayan görüntülerin ayrıştırılması için önce Gabor ve morfolojik filtreleme yapılarak renk ve şekil bilgilerine göre graf tabanlı bir segmentasyon yöntemi ile %90 üzeri doğrulukla yollar elde edilebilmiştir.

Bir başka yeni çalışma da Maboudi ve diğ. (2018) [9] aittir. Geliştirdikleri sistemde karınca kolonisi optimizasyonundan ve bulanık mantık yönteminden faydalanarak %89 üzeri bir başarımla elde etmeyi başarmışlardır.

Liu ve diğ. (2016) [10] ise kesme dönüşüm yöntemi ve iskeletlendirme algoritması kullanarak %90'a yakın bir başarımla elde etmişlerdir.

Literatürdeki çalışmaların hemen hepsinde başlangıç aşamasında çeşitli önışlem metotlarının kullanıldığı görülmektedir [11]. Giriş kısmının sonunda bahsedilen uydu görüntülerindeki bazı

istenmeyen gürültülerin yok edilerek bilgisayar öğrenmesi aşamalarının daha sağlıklı gerçekleştirilmesi için ön işlem metotları mutlaka kullanılmak zorundadır.

Bu çalışmada geliştirilen önışlem metodu literatürdeki diğer önışlem metotlarından farklı olarak zincir kod ve LCS yöntemlerini kullanarak söz konusu probleme çözüm oluşturmaya çalışmaktadır. Zincir kod yöntemi görüntü işlemede sıklıkla [12][13][14][15][16] kullanılan bir yöntemdir. Bu yöntem her bir yol kenar nesnesine ait zincir kodun çıkarılmasını sağlamaktadır. Bu kodların benzerliklerinin karşılaştırılması ve benzer olanlarının yol olduğunun belirlenmesi içinse LCS yöntemi tercih edilmiştir. Bu yöntem, sinyal işlemede iki sinyalin benzerliğinin bulunmasında ve metin işleme de de iki metnin benzerliğinin bulunmasında kullanılan bir yöntemdir [17][18][19][20][21]. Her iki yöntemde uygulaması basit ve hızlı çalışan yöntemler olduğundan tercih edilmişlerdir.

Bu önışlem kısmında sonra başarımla daha da iyileştirilmesi için bölge genişleme algoritması kullanılmıştır [22]. Bu yöntemin çalışması için gerekli olan başlangıç piksel değeri için ise üstteki yöntemlerin çıktısı verilmiştir. Böylelikle hem sonuçlar iyileştirilmiş hem de söz konusu yöntemin ihtiyaç duyduğu bilgi için kullanıcının değer girme zorunluluğunun da önüne geçilmiştir.

Sonuç olarak literatürdeki benzer çalışmalara yakın başarımla elde edebilen ve bunu da daha az adımda başarabilen bir sistem oluşturulmuştur.

3 Geliştirilen sistemin yapısı

Literatürdeki birçok çalışmadan da görüleceği üzere uydu görüntüleri üzerinde her hangi bir uygulama yapılacağı zaman öncelikle uygun bir ön işlem süreci yapılmalıdır. Aksi takdirde geliştirilmek istenen sistemin başarımla ciddi oranda düşecektir. Bu kapsamda geliştirilmek istenen önışlem yönteminin ana yapısı ise 8 adımdan oluşmaktadır.

1. Çok bantlı uydu görüntü dosyasının gri renk uzayına dönüştürülmesi
2. Yumuşatma
3. Morfolojik kapama
4. Canny [23] kenar bulma yönteminin uygulanması
5. İskelet çıkarma

6. Kenarlara ait zincir kodların (Chaincode) [24] çıkarılması
7. En Uzun Ortak Alt Dizi (LCS) [25][26] yöntemi ile kenarlara ait zincir kodların benzerliklerinin çıkarılması
8. Yol piksellerine ait dinamik gri renk eşği bulunarak bu deęerin altındaki piksellerin temizlenmesi

Yöntemin başarımının test edilmesi için de Das ve dię. (2011) [3]'de ki yayınlarındaki açık kaynak olarak sunulan uydu görüntüleri kullanılmıştır. Bahsi geçen yayında bu görüntülerin <http://wikimapia.org> sitesinden alındığı ve 512x512 büyüklükte 1-m/piksel çözünürlüklü çok bantlı (multispectral, RGB) oldukları belirtilmiştir.

Aşağıda bu adımlarda belirtilen işlemler sırayla ayrıntılı bir biçimde açıklanmaktadır

3.1 Çok bantlı uydu görüntü dosyasının gri renk uzayına dönüştürülmesi

İşlem zamanının kısaltılması ve kenar bulma algoritmasının çalışabilmesi için RGB renk uzayındaki uydu görüntüleri gri renk uzayına dönüştürülmektedir. Bu işlem için denklem (1)'deki ifade kullanılır[27].

$$I_g = I_R * 0.2989 + I_G * 0.5870 + I_B * 0.1140 \quad (1)$$

Burada, I_R , I_G , ve I_B deęerleri sırasıyla görüntüdeki kırmızı, yeşil ve mavi piksellerin deęerini gösterir. I_g ise resmin gri renge dönüşmüş halini gösterir.

3.2 Yumuşatma

Her hangi bir görüntü içerisinde var olması istenmeyen piksellere gürültü (noise) denir [28]. Bu gürültü pikselleri, görüntünün alındığı kameradan, görüntünün iletildiği ortamdan veya çevresel faktörlerden kaynaklanmış olabilir. Hatta gerçekte bu tarz bir gürültüde olmayıp sadece yapılacak çalışmada var olması istenmeyen her hangi nesneye ait pikseller de olabilir.

Bu çalışmada, basitliği ve hızlı çalışmasından dolayı çok tercih edilen bir yöntem olan ortalama filtreleme yöntemi kullanılmıştır. Bu yöntemde, $m \times n$ boyutlarında bir çekirdek (kernel) matris (bu çalışmada 3×3 boyutlarında tercih edilmiştir) resmin tüm pikselleri üzerinde soldan sağa ve yukarıdan aşağıya doğru birer birer gezdirilir. Her adımda, matrisin altında kalan alandaki piksellerin gri renk deęerlerinin ortalaması hesaplanır ve ortadaki piksele deęer olarak atanır. Kenar noktaları için konvülyasyon yapılırken ise

görüntünün sınırını aşan noktalarla işlem yapılacağı zaman en yakında bulunan piksel dikkate alınır. Böylece f resmi g filtresi ile konvülyasyon işlemine tabi tutularak filtrelenmiş olur.

Konvülyasyon işlemi sonunda görüntüde, çekirdek pencerenin büyüklüğü ile doğru orantılı bir bulanıklık oluşacak ve bu da bazı gürültülerin önüne geçilebilmesini sağlayacaktır. Konvülyasyon işlemi için denklem (2)'deki gibi ifade kullanılmıştır [24]. Burada f filtrelenecek resmi, g filtreyi ve $*$ operatörü de konvülyasyon işlemi gösterir. Denklemdeki h ise konvülyasyon işleminin sonucunda yumuşatılmış resmi ifade eder

$$h(h, x) = f(x, y) * g(x, y) \quad (2)$$

3.3 Morfolojik kapama

Görüntü işlemede morfolojik kapama işlemi önce genişleme sonra aşınma diye de adlandırılır. Burada amaç, sonraki adımda yapılacak olan kenar bulma işleminde daha az gürültüye sahip bir görüntü sunarak başarımı arttırabilmektir. Morfolojik genişleme işlemi bir küme işlemi olup denklem (3)'teki gibi ifade edilir [29].

$$A \oplus B = \{z | (B^{\wedge})_z \cap A \neq \emptyset\} \quad (3)$$

Burada, A deęeri işlem yapılmak istenen resmi, B ise genişleme işleminde kullanılacak yapısal elementi gösterir. Bu elementin merkezi A resmindeki bir pikselin üzerine getirildiğinde en az bir ya da daha fazla piksel altta kalıyorsa bu piksel yeni oluşacak görüntüye dâhil edilir. İşlem A resmindeki tüm pikseller için gerçekleştirilir. Morfolojik aşınma işlemi de bir küme işlemi olup denklem (4)'teki gibi ifade edilir [29].

$$A \ominus B = \{z | (B)_z \cap A^{\wedge} \neq \emptyset\} \quad (4)$$

Burada, yine A deęeri işlem yapılmak istenen resmi, B ise aşınma işleminde kullanılacak yapısal elementi gösterir. Bu elementin merkezi A resmindeki bir pikselin üzerine getirildiğinde, elementin her hangi bir pikseli A 'nın tümleyenine yani arka plana dokunuyorsa A görüntüsündeki o piksel yeni görüntüde olmayacaktır. Morfolojik kapama işlemi de kısaca denklem (5)'teki gibi yazılabilir [29].

$$A \cdot B = (A \oplus B) \ominus B \quad (5)$$

3.4 Kenar bulma

Yol bulma işlemlerinde kenar bilgisi en önemli verilerden biridir. Kenar, görüntüde belirli bir

değerin üzerinde değişimin olduğu yerdir diye tanımlanabilir [30]. Bu değişim, birinci derece türevin ($\nabla f=f'$) yerel maksimumu bulunarak elde edilebilir.

Bu çalışmada Canny kenar bulma algoritması kullanılmıştır. Bu algoritmayı diğerlerinden farklı kılan yönleri ise şunlardır [29]

- Düşük hata oranı
- Gerçek kenar noktasına en yakın noktada kenar tespiti yapılabilmesi
- Her bir gerçek kenar noktası için yalnızca tek bir kenar noktası bulabilme

Canny kenar bulma algoritması aşağıda verilen 5 adımda çalışmaktadır:

- Yumuşatma : Canny algoritması Gauss filtresi kullanarak yumuşatma işlemini gerçekleştirir. Bu filtre denklem (6)'da verilmiştir.

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (6)$$

Burada σ işareti standart sapmayı gösterir.

- Gradyan büyüklüğünün ve yönünün bulunması :

Görüntünün, Gauss fonksiyonun x ve y doğrultusunda birinci derece türevi ile konvülyasyona tabi tutulması işlemidir. Bu işlem denklem (7) ve (8)'de belirtilmiştir

$$f_x(x, y) = f(x, y) * \left(\frac{-x}{\sigma^2}\right) \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (7)$$

$$f_y(x, y) = f(x, y) * \left(\frac{-y}{\sigma^2}\right) \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \quad (8)$$

Ardından büyüklük değeri için denklem (9) ve yön için de denklem (10) kullanılır.

$$|f(x, y)| = \sqrt{f_x(x, y)^2 + f_y(x, y)^2} \quad (9)$$

$$\theta(x, y) = \arctg\left(\frac{f_y(x, y)}{f_x(x, y)}\right) \quad (10)$$

- Gradyan vektörü yönünde zayıf kenar noktalarını baskılama (non-maximum suppression) : Bu işlem için önceki adımda bulunan θ açısı 4 ana bölgeye indirgenir. Bu bölgeler 8 temel yönü içermektedir. Diğer bir deyişle eldeki açı değeri en yakın 45

dereceye yuvarlanarak ait olduğu bölge belirlenmiş olur. Daha sonra geçerli pikselin değeri o bölgenin negatif yönündeki piksel ile ve pozitif yönündeki piksel ile karşılaştırılır. Eğer en büyük değer geçerli pikselin ise değişim yapılmaz aksi halde geçerli piksel silinir. Örneğin yön değeri kuzey ise, geçerli pikselin bir üstündeki ile bir altındaki piksel değerleri kontrol edilecektir.

- İkili eşik değeri yöntemi ile kenarların birleştirilmesi : Gradyan vektörünün büyüklük değeri yüksek eşik değerinden fazla olan pikseller güçlü (yani kesin kenar pikseli) olarak kabul edilirken gradyan vektörünün büyüklük değeri düşük eşik değerinden daha küçük olan pikseller de zayıf (yani kesin kenar olmayan piksel) olarak kabul edilir. Gradyan vektörünün büyüklük değeri iki eşik değeri arasında kalan piksellerin her biri için komşusu olan 8 yöne de bakılır. En az bir tane güçlü piksel bulunursa geçerli piksel de güçlü kabul edilir yoksa silinir

3.5 İskelet çıkarma

Zincir kod işlemine başlamadan önce bir önceki adımda bulunan kenarların iskeletlerinin çıkarılması ve dallanma noktalarının koparılması gerekmektedir. Bu işlem yapılmadan kenarlara ait zincir kodlar çıkarılacak olursa elde edilen kodlar yanlış olacak ve istenen başarımla elde edilemeyecektir.

İskelet çıkarma işlemi, bir A görüntüsünün aşınma ve yayılma işlemi olarak ifade edilebilir ve denklem (11)'deki gibi gösterilebilir [31]

$$S(A) = \bigcup_{k=0}^K [(A \ominus kB) - (A \ominus kB) \circ B] \quad (11)$$

Burada B yapısal elementi gösterirken, $A \ominus kB$ işlemi ise A görüntüsünün k defa aşınmaya tabi tutulacağını ifade etmektedir. K ise A görüntüsünün boş küme olmadan önceki son adımını göstermektedir. Bu ifade denklem (12)'de gösterilmiştir

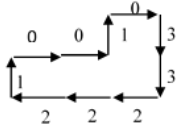
$$K = \max\{k | (A \ominus kB) \neq \emptyset\} \quad (12)$$

Bu işlemden sonra çatallanmış olan kenarlar dallanmanın olduğu piksel renk değeri silinerek yok edilir ve görüntüde sadece tekil kenarların kalması sağlanır.

3.6 Zincir kod

Zincir kod (chaincode) yöntemi, harflerin [12], nesnelerin [13][14][15], ya da sinyallerin [16] incelenmesinde kullanılan yöntemlerden biridir. Bu yöntemde, nesnenin sırayla kenarlarındaki tüm piksellerde dolaşılır. Her bir pikselin bir öncekine göre yön durumuna ait kod kaydedilir. Burada kod değerinin alabileceği değerler dört yön (kuzey-güney-doğu-batı) kullanıldıysa 0 - 3 arasında, 8 yön (kuzey kuzey doğu, kuzey batı, vb.) kullanıldıysa 0 - 7 arasında olacaktır. Bu çalışmada daha hassas ve doğru bir kod dizisi sunduğu için 8 yön kullanılması tercih edilmiştir.

Şekil 1'de bir nesne için zincir kod örneği verilmiştir. Bu örnek için oluşturulabilecek zincir kodlardan biri {0, 0, 1, 0, 3, 3, 2, 2, 2, 1} olacaktır. Bu zincir kod bilgisi kullanılarak aynı şekil yeniden elde edilebilir



Şekil 1. Nesnenin 4 yönlü zincir kodu

Birden fazla zincir kod oluşturulabilmesinin nedeni ise başlangıç noktasının istenildiği yerden seçilebilmesindedir. Bundan dolayı zincir kod yöntemi uygulanırken seçilen başlangıç noktasından elde edilen kod sola doğru bir adım ötelenerek her iki kodun fark değerleri alınır. Böylece başlangıç noktasından bağımsız bir zincir kod elde edilmiş olunur.

Bu çalışmada yol kenarları için zincir kod çıkarılacağı için nesnenin çevresinde dönmek istediğimiz sonucu bize vermeyecektir. Çünkü yol kenarları kapalı nesnelere değil açık uçlu çizgilerden oluşmaktadır. Bu yüzden bir yol kenarının her hangi bir ucundaki ilk pikselden başlanıp son piksele kadarki zincir kod çıkarılır. Yukarıda açıklanan başlangıç noktası sorunundan dolayı aynı işlem diğer uçtan başlanılarak yapılır ve sonuçta n tane pikselden oluşan bir yol kenarı için n - 1 uzunluğunda iki adet zincir kod elde edilir.

3.7 En uzun ortak alt dizi

Zincir kod yöntemi ile yol kenarlarına ait kodları çıkardıktan sonra sıra bunların karşılaştırmasına gelmektedir. Fakat zincir kod yönteminin gürültüye ve nesnenin büyüklüğüne son derece hassas oluşundan dolayı basit bir karşılaştırma sonucu büyük oranda benzeyen iki yol kenarı benzeşmiyor

gibi gözükülecektir. Bundan dolayı bu çalışmada karşılaştırma için LCS yönteminden faydalanılmıştır.

LCS yöntemi, yazım düzeltme, dosya karşılaştırma, genetik mühendisliği ve sinyal işlemede kullanılan bir yöntemdir [17][18][19][20][21]. Burada hedef, verilen iki kümeni her ikisine de soldan sağa doğru bakıldığında bulunabilecek en uzun boyutlu alt kümenin bulunmasıdır. Benzeşmenin aynı indislerde olması da gerekmez. Örneğin "abcdefg" ve "klaxcyg" şeklindeki iki kümeden "a", "c", "g", "ac", "ag" ve "acg" gibi alt kümeler çıkarılabilir. Ama burada aranan en uzun benzeşen küme olduğundan sonuç "acg" olacaktır.

Yönteme ait algoritma literatürde genellikle özyinelemeli olarak verilmektedir [32]. Denklem (13) bu tarz bir gösterimi ifade etmektedir. Burada karmaşıklık değeri ise $O(2^n)$ olmaktadır (n kümenin boyutudur)

$$LCS(X_i, Y_i) = \begin{cases} \text{eğer } i = 0 \text{ veya } j = 0 & \emptyset \quad (1) \\ \text{eğer } X_i = Y_i & LCS(X_{i-1}, Y_{i-1}) \\ \text{eğer } X_i \neq Y_i & \max(LCS(X_i, Y_{i-1}), L) \end{cases}$$

Bu çalışmada LCS yönteminin yol kenarlarının benzerliğinin tespitinde kullanılabilmesi için öncelikle her bir kenar için kullanıcının istediği yol büyüklüğünde bir çerçeve oluşturulmuştur. Burada verilecek en az ve en çok yol genişliği değerleri geliştirilen sistemin kullanıcıdan istediği tek parametrelerdir. Referans alınan her bir kenar için oluşturulan bu çerçeveler içerisindeki kenarların zincir kodları ile referans kenarların zincir kodları karşılaştırılır. Referans kenar ile en çok benzeşen kenarın bir yol oluşturduğu belirlenir ve benzeşmenin başladığı ve bittiği alan yol olarak işaretlenir. Ancak burada dikkat edilmesi gereken nokta benzeşmenin karşılaştırılan kenarların zıt uçlarında olmaması gerektiğidir. Diğer bir deyişle bir kenar nesnesinin sol ucu diğer kenar nesnesinin sağ ucuna benziyor olabilir ama bu onların yol kenarı olabileceklerini göstermez. Çünkü yol kenarlarında benzeşmenin yakın bir bölüt içinde olması beklenir. Geliştirilen sistemde bu sorunun çözümü için öncelikle benzeşen iki kenar nesnesinin her pikselinin diğerine olan Manhattan mesafesi ölçülür. Bu hesaplama için denklem (14) kullanılır.

$$ManhattanMesafesi = \sqrt{(y_2 - y_1)^2 + (x_2 - x_1)^2} \quad (12)$$

Ardından benzeşen bir piksel bulunduğunda 20 piksel öncesi ve sonrasında başka benzeşen olup olmadığına da bakılır. Yani sadece 20 piksellik alan içinde benzeşme varsa kabul edilir. Daha uzaktaki bir pikselle benzeşiyorsa bu benzeşme geçersiz kabul edilir.

LCS yöntemi sonucunda kenar nesnelere zincir kodu benzeşen pikselleri arasındaki alanlar yol olarak işaretlenmiş olacaktır.

3.8 Dinamik renk

Önceki adımda kenarlara ait zincir kodlarının benzeşmesinden dolayı yol olabilecek alanlar tespit edilmişti. Bu adımda ise ilk olarak söz konusu alanlara ait piksellerin ortalama gri renk değerleri hesaplanacaktır. Test için kullanılan görüntülerin hepsinde yolların gri renk değeri genel olarak koyu renkten ziyade daha çok açık renge doğru kaydığından ikinci adım olarak da görüntüde bu değerden düşük olan pikseller silinecektir. Bu işlemin sabit ve elle verilen bir sayı ile değil de sistemin bu şekilde belirlediği o resme ait dinamik gri renk eşik değeri ile yapılmış olması söz konusu işleme ait oluşabilecek hataları en aza indirmek içinidir.

Böylece kenar bulma algoritmasının ormanlık alanlarda ya da açık arazilerde tespit ettiği ve zincir kodları da benzeşen hatalı alanların temizlenmesi sağlanmış olacaktır.

3.9 Bölge büyütme algoritması

Önceki adımların sonucunun daha da başarılı hale getirmek için burada ek bir adım daha yapılacaktır. Bu adımda bir tür görüntü bölütleme yöntemi olan bölge büyütme algoritması kullanılacaktır.

Görüntü bölütleme, bilgisayarlı görme sahasındaki önemli konulardan biridir [33]. Tıbbi görüntü işleme, görüntüde nesne bulma, otomatik trafik kontrol sistemleri vb. birçok farklı sahadaki görüntülerde kullanılan yöntemler geliştirilmiştir. Bu yöntemlerin genel amacı görüntüde belirli karakteristik özellikleri ortak olan (homojen) alanlar oluşturmaktır [34]. Burada ortak özellikler (homojenlik), renk, şekil ve yüzey kaplaması gibi özelliklerden oluşabilir [35].

Bölge büyüme (region growing) algoritması da Adams ve Bischof (1994) [22] tarafında geliştirilmiş piksel bazında yarı otomatik bir bölge tabanlı görüntü bölütleme yöntemidir.

Kolay, hızlı ve başarımı yüksek bir yöntem olduğundan birçok farklı sahadaki çalışmada

[36][37][38] kullanılmıştır. Ancak yöntemin başarımın etki eden en önemli faktör başlangıç besleme (seed) değeridir. Kötü verilecek bir besleme değeri sonuçları olumsuz etkileyecektir. Bu değerlerin elle verilmesini engellemek için de bazı otomatize etme çalışmaları [33][35] yapılmıştır

Bu çalışmada ise bölge büyütme algoritmasının ihtiyaç duyduğu ilk besleme değeri elle değil zincir kod ve LCS yöntemlerinin sonucundan verilmektedir. Böylece hem otomatize edilmiş bir bölge büyütme yapılmış hem de önceki bölümlerde elde edilen sonuçlar daha da iyileştirilmiş olacaktır.

Burada, önceki adımda yol olarak belirlenen piksellerin ortalama gri renk değerine en yakın ilk piksel seçilerek bölge büyütme işlemi uygulanır. İşlem sonunda yeniden önceki adımdan gelen yol piksellerine bakılır ve henüz değerlendirilmemiş piksel varsa yine ortalama gri renk değerine en yakın olandan başlanarak bölge büyütme işlemi devam ettirilir. Tüm pikseller belirli bir bölgeye dâhil olana kadar işleme devam edilir.

Bölge büyütme işlemi esnasında seçili pikselin dört yönüne yani üst, alt, sol ve sağ komşu piksellere bakılır. Başlangıçta ortalama gri renk değeri geçerli pikselin rengi kabul edilir. Komşu piksellerden bu değere en yakın gri değer içeren bölgenin elemanı olarak belirlenir. Eğer bu piksel bölgeye katıldığında bölgenin ortalama gri renk değerindeki değişim %10'dan az ise yeni eklenen piksel başlangıç olarak alınıp aynı işlem adımları oradan devam ettirilir. %10'dan fazla fark oluştuğunda bölge oluşmuş demektir. İşlem böylece sonlandırılır.

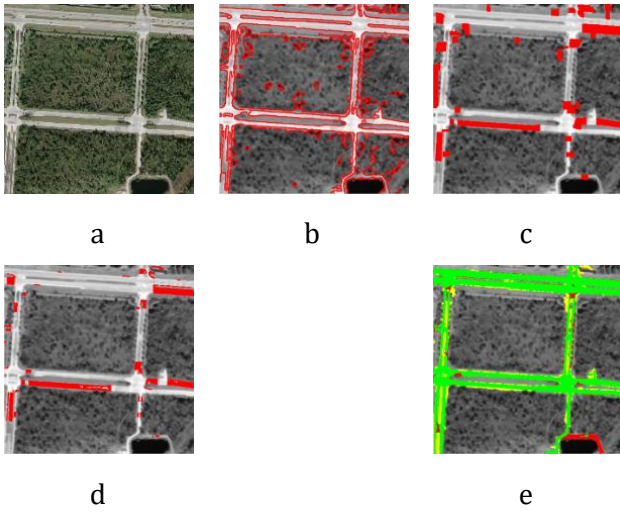
4 Sonuçlar

Bu bölümde geliştirilen sistemin örnek görüntüler üzerinde çalışması sunulacaktır. Bu görüntüler, Das ve diğ. (2011)[3]'de ki yayınlarında kullanmış oldukları şehir dışı uydu görüntülerinin bazılarıdır. Görüntüler, 512x512 RGB ve JPG formatındadırlar. Uygulama Matlab 2016 programı ile gerçekleştirilmiştir.

Şekil 2 – 5 arasında geliştirilen sistemin beş ana adımına ait çıktılar dört farklı uydu görüntüsü üzerinde sunulmuştur. Bu adımların ilki (a) uydu görüntüsünün orijinal halini göstermektedir. İkinci adım (b) Canny kenar bulma algoritması sonucunda bulunan kenarları kırmızı ile çizerek göstermektedir. Üçüncü adım (c) LCS yöntemi sonucunda benzer olarak tespit edilen yol bloklarını kırmızı ile boyayarak göstermektedir. Dördüncü

adım (d) renk eşik değeri kontrolü sonrası elde kalan pikselleri göstermektedir. Beşinci ve son adım da (e) bölge büyütme algoritması ile el çizimine ait karşılaştırmayı göstermektedir.

Bu karşılaştırmada yeşil renkli alanlar gerçekte yol olan ve geliştirilen sistemin de yol olarak bulduğu pikselleri, kırmızı alanlar gerçekte yol olduğu halde yol olarak tespit edilemeyen pikselleri ve sarı alanlar da gerçekte yol olmadığı halde sistemin yol olarak tespit ettiği pikselleri göstermektedir



Şekil 2. Görüntü 1 için çıktılar : (a):Orijinal görüntü, (b):Canny, (c):LCS, (d):Renk eşik değeri, (e): Bölge büyütme sonucu karşılaştırma

Şekil 2'de geliştirilen yöntemin uygulanması için kullanılacak olan ilk görüntü dosyası verilmiştir. Şekil 2.a' da, kara yolları ve ağaçlık alanlar görülebilmektedir. Fakat görüntünün sağ üst kısmındaki araçlar ve kara yolunun orta kısmındaki refüj alanları yolların tespitini zorlaştıran etmenlerden biridir. Bu görüntüde yol genişliği parametresi en az 20 ve en çok 30 olarak alınmıştır.

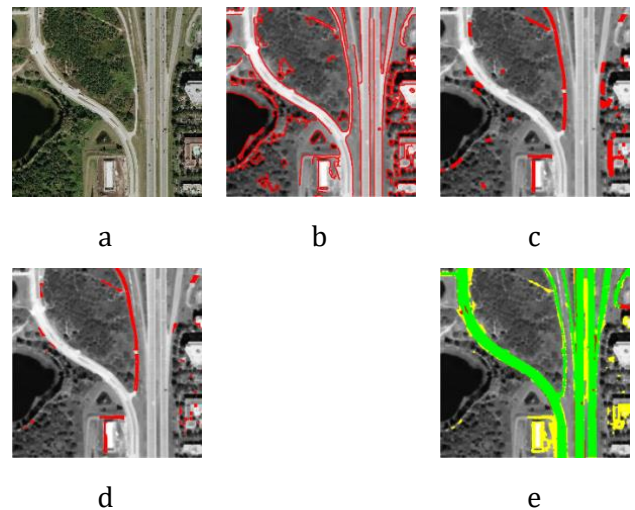
Şekil 2.b'de orijinal görüntü önce gri renk uzayına dönüştürülmüş ardından da önce yumuşatma sonra da morfolojik kapama işlemi uygulanmıştır. Ardından Canny kenar bulma algoritması eldeki görüntü üzerinde çalıştırılmış ve algoritmanın kenar olarak tespit ettiği alanlar kırmızı ile çizilmek suretiyle Şekil 2.b oluşturulmuştur.

Şekil 2.c'de LCS yöntemi sonucu muhtemel olarak bir yola ait olabileceği belirlenen pikseller kırmızı ile boyanarak gösterilmektedir. Bu işlem için öncelikle zincir kod yöntemi Şekil 2.b'ye uygulanarak her bir kenar için zincir kodlar elde edilmiştir. Daha sonra her bir kenar nesnesi üzerine

başta belirtilmiş olan yol genişliği boyutunda bir çerçeve yerleştirilmiştir. Her çerçeve içindeki kenarların zincir kodlarının, üzerine çerçeve yerleştirilen geçerli kenarın zincir kodu ile olan benzerliği LCS yöntemiyle kontrol edilmiştir. Sonuç olarak geçerli kenarla en çok benzeşen kenarın benzeştikleri pikseller ve arasında kalan alan kırmızı olarak gösterilebilmiştir.

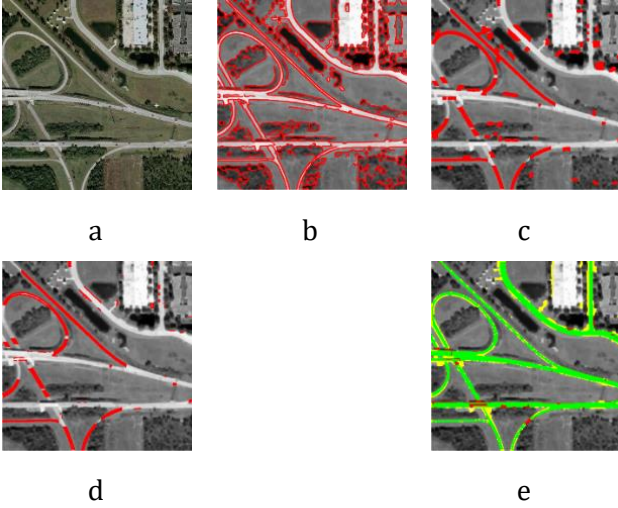
Şekil 2.d'de ise dinamik renk eşiği uygulanarak elde edilen yol pikselleri gösterilmektedir. Burada ormanlık, tarla vb. alanlarda aslında yol olmayan ama zincir kodları benzeşen kenarlar ve aralarındaki alanlar silinmektedir. Dinamik renk değeri bilgisi ise LCS yöntemi sonucu muhtemel yol olarak belirlenen piksellerin ortalama gri renk değeridir. Bu değerden küçük gri rengi olan pikseller silindiğinde Şekil 2.c'de ki görüntü Şekil 2.d'de ki hale gelmektedir.

Son olarak Şekil 2.d üzerinde bölge büyütme algoritması uygulanmış ve Şekil 2.e elde edilmiştir. Söz konusu algoritmanın ihtiyaç duyduğu besleme değeri Şekil 2.d'de kırmızı olarak işaretlenmiş piksellerdir. Diğer bir deyişle muhtemel yol olabilecek alanlardan bölge büyütme işlemi yapılmış ve Şekil 2.e elde edilmiştir. Burada yeşil ile boyanan yerler gerçekte yol olan ve geliştirilen yöntemin de yol olarak bulduğu pikselleri göstermektedir. Kırmızı alanlar ise gerçekte yol olupta geliştirilen yöntem tarafından yol olarak bulunamayan pikselleri göstermektedir. Sarı alanlar da gerçekte yol olmayan ama yol olarak tespit edilen alanları göstermektedir.



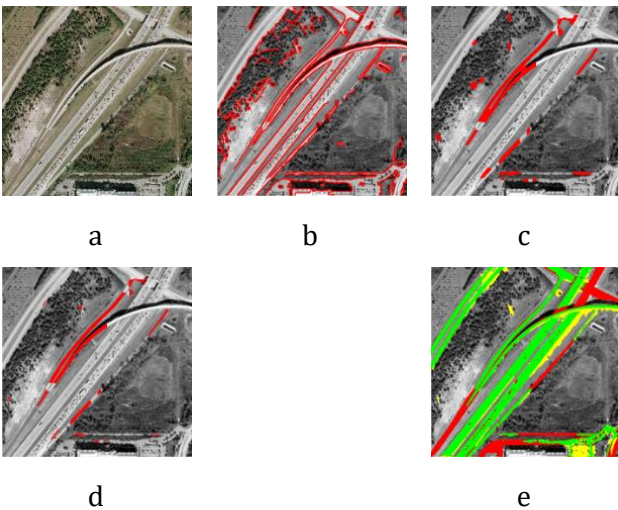
Şekil 3. Görüntü 2 için çıktılar : (a):Orijinal görüntü, (b):Canny, (c):LCS, (d):Renk eşik değeri, (e): Bölge büyütme sonucu karşılaştırma

Şekil 3'de geliştirilen yöntemin uygulanması için kullanılacak olan ikinci görüntü dosyası verilmiştir. Şekil 3.a'da kara yolları, ağaçlık alanlar ve binalar görülebilmektedir. Bu görüntüde orta alt kısımdaki yapı hem şekil hem de renk olarak kara yoluna çok benzediğinden yöntemin başarımını olumsuz etkilemektedir. Bu görüntüde yol genişliği parametresi en az 5 ve en çok 15 olarak alınmıştır.



Şekil 4. Görüntü 3 için çıktılar : (a):Orijinal görüntü, (b):Canny, (c):LCS, (d):Renk eşik değeri, (e): Bölge büyütme sonucu karşılaştırma

Şekil 4'de geliştirilen yöntemin uygulanması için kullanılacak olan üçüncü görüntü dosyası verilmiştir. Şekil 4.a'da kara yolları, ağaçlık alanlar ve binalar görülebilmektedir. Bu görüntüde üst sağ bölgedeki yapılar hem şekil hem de renk olarak kara yoluna çok benzediğinden yöntemin başarımını olumsuz etkilemektedir. Bu görüntüde yol genişliği parametresi en az 7 ve en çok 15 olarak alınmıştır.



Şekil 5. Görüntü 4 için çıktılar : (a):Orijinal görüntü, (b):Canny, (c):LCS, (d):Renk eşik değeri, (e): Bölge büyütme sonucu karşılaştırma

Şekil 5'de geliştirilen yöntemin uygulanması için kullanılacak olan dördüncü ve son görüntü dosyası verilmiştir. Şekil 5.a'da kara yolları, ağaçlık alanlar ve binalar görülebilmektedir. Bu görüntüde orta alt bölgedeki yapı ve otopark alanı hem şekil hem de renk olarak kara yoluna çok benzediğinden yöntemin başarımını olumsuz etkilemektedir. Aynı şekilde karayolunun sol kısmındaki kara yolu ile benzer şekil ve renkteki alanda yanlış tespit yapılmasına neden olmaktadır. Bu görüntüde yol genişliği parametresi en az 7 ve en çok 15 olarak alınmıştır.

Geliştirilen yöntemin başarımını değerlendirebilmek için aşağıda listelenen istatistiksel kıstaslar kullanılmıştır. Buradaki değerler oran oldukları için en yüksek değer 1, en düşük değer de 0'dır.

Tablo 1'de söz konusu kıstasların hesaplanma yöntemini gösteren karşılaştırma (confusion) matrisi verilmiştir.

Tablo 1. Karşılaştırma matrisi

		Referans		
		Pozitif (P)	Negatif (N)	
Test Çıktısı	Pozitif	Doğru Pozitif (DP)	Yanlış Pozitif (YP)	Hassasiyet DP / (DP + YP)
	Negatif	Yanlış Negatif (YN)	Doğru Negatif (DN)	Negatif Öngörü Değeri DN / (DN + YN)
		Duyarlılık DP / (DP + YN)	Özgüllük DN / (YP + DN)	Kesinlik (DP + DN) / (P + N)

Kıstasların ifade ettikleri anlamlar ise şöyledir

- Duyarlılık (Sensitivity) : Yol olan pikselleri yol olarak tanımlayabilme oranıdır. Bu değer, bazı kaynaklarda [3][39] bütünlük (completeness) olarak da geçmektedir. Test verisinin çoğunluğunu pozitif kabul eden bir yöntemde bu değer çok yüksek çıkabilir.
- Özgüllük (Specifity) : Yol olmayan piksellerin yol olmadığını tanımlayabilme oranıdır. Bu değer, bazı kaynaklarda [3][39] doğruluk (correctness) olarak da geçmektedir. Test verisinin çoğunluğunu negatif kabul eden bir yöntemde bu değer çok yüksek çıkabilir.
- Hassasiyet (Precision) : Gerçekte yol olan ve yol olarak bulunan piksel sayısının tüm yol olarak bulunan piksel sayısına oranıdır. Bu değer yüksek olması sistemin güvenilir bir pozitiflik kapasitesine sahip olduğunu

gösterir. Ancak bu değer başarımın dar bir alanda mı yoksa tüm görüntüde mi olduğunu göstermez. Diğer bir deyişle az sayıda piksel hatasız olarak bulunsa da, daha tespit edilemeyen birçok piksel olduğundan tüm görüntü için bu değer güvenilir olmayabilir.

- Negatif Öngörü Değeri (NÖD) : Gerçekte yol olmayan ve yol değil olarak bulunan piksellerin tüm yol değil olarak bulunan piksellere oranıdır. Bu değer yüksek olması sistemin güvenilir bir negatiflik kapasitesine sahip olduğunu gösterir. Ancak hassasiyetteki ölçek boyutunun belirsizliği sorunu burada da mevcuttur.
- Kesinlik (Accuracy) : Doğru olarak bulunanların (yol ve yol olmayan) sayısının toplam görüntüdeki piksel sayısına oranıdır. Bu değer yüksek olması sistemin hem başarımının yüksek olmasını hem de kapsamının geniş olmasını gösterir.
- F1 : Hassasiyet ve duyarlılık değerlerinin harmonik ortalamasıdır. Bazı kaynaklarda [39] kalite diye de geçmektedir.

Tablo 2' de zincir kod yönteminin sonucunun LCS ile karşılaştırması neticesinde elde edilen başarım sonuçları verilmiştir. Tablo 3' de ise bu sonuçların üzerine bölge büyütme algoritması kullanılması suretiyle elde edilen son başarım değerleri verilmiştir.

Uydu görüntülerindeki kara yollarının tespiti problemi açısından bu çalışmada sunulan zincir kod ve LCS yöntemlerinin Tablo 2' deki kesinlik değerlerine bakıldığında makul denebilecek bir başarım görülse de F1 skorlarının düşük olması tek başlarına söz konusu problemin çözümünde yetersiz kaldıklarını ortaya koymaktadır. Bununla birlikte söz konusu çıktıların bölge büyütme yöntemi gibi başlangıç değerinin kullanıcı tarafından el ile girilmesi gereken yerlerde kullanılması suretiyle artırılması ise mümkündür. Böylece hem el ile giriş yapma gereksinimi ortadan kaldırılırken hem de genel sistemin başarımı daha da iyileştirilebilmektedir.

Tablo 2. Zincir kod ve benzerlik karşılaştırma sonrası başarım.

	Görüntü 1	Görüntü 2	Görüntü 3	Görüntü 4
Duyarlılık	18.37%	4.69%	23.73%	9.44%
Özgüllük	96.44%	97.04%	96.54%	97.62%
Hassasiyet	55.22%	27.72%	51.46%	52.98%
NÖD	83.19%	80.79%	89.11%	79.17%
Kesinlik	81.40%	79.04%	86.79%	78.13%
F1	27.57%	8.03%	32.48%	16.02%

Tablo 3. Bölge büyütme yöntemi sonrası başarım.

	Görüntü 1	Görüntü 2	Görüntü 3	Görüntü 4
Duyarlılık	90.86%	91.27%	91.09%	80.31%
Özgüllük	97.94%	94.96%	98.08%	91.20%
Hassasiyet	91.31%	81.41%	87.98%	72.15%
NÖD	97.82%	97.82%	98.61%	94.23%
Kesinlik	96.57%	94.24%	97.14%	88.80%
F1	91.08%	86.06%	89.51%	76.01%

Geliştirilen yöntemin başarımı aynı görüntüleri kullandığımız Das ve diğ. (2011) [3] makalesindeki başarılara oldukça yakındır. Bu başarım karşılaştırması Tablo 4'te verilmiştir. Söz konusu makalede duyarlılık ve özgüllük değerlerinin 25 adet görüntünün başarımının ortalaması olarak elde edildiği belirtilmiştir. Bu çalışmadaki değerler için ise kullanılan dört resmin başarım ortalamaları alınmıştır. Yukarıda çizelge kısıtlarının anlamlarında belirtildiği gibi kesinlik ve F1 değerleri başarıma ait daha güvenilir değerler sunabiliyor olsa da söz konusu makalede bu değerler verilmediğinden karşılaştırmada kullanılmamıştır.

Tablo 4. Literatür ile sonuç karşılaştırma.

	Duyarlılık	Özgüllük
Das ve diğ. (2011)	93%	97%
Geliştirilen Metot	88%	96%

Tablo 4'de görülebileceği gibi başarım oranları oldukça yakın elde edilebilmiştir. Bununla birlikte işlem adımları önemli ölçüde azaltılmıştır.

Geliştirilen yöntem her ne kadar şehirlerarası yollarda başarım sağlayabilmiş olsa da şehir içi yollar özellikle de sokaklar için yetersiz kalmaktadır. Bunun sebebi zincir kod yönteminin uzun kenarlar da daha başarılı sonuçlar vermesinden dolayıdır. Yine de başarımın iyileştirilmesi için Das ve diğ. (2011) [3] yöntemindeki gibi makine öğrenmesi yöntemlerinden istifade edilebilir.

5 Değerlendirme

Çalışmanın başında da bahsedildiği gibi uydu görüntülerindeki kara yollarının tespiti karmaşık ve

tek bir yöntemle başarılamayacak bir problemdir. Güncel yol bilgisinin hızlı bir şekilde alınıp haritalara yansıtılması gerekliliği navigasyon ve CBS sistemleri açısından elzemdir. Ancak yüksek miktardaki verinin insan tarafından hızlıca işlenmesi mümkün olmadığından bilgisayarlı görme yöntemlerine ihtiyaç duyulmaktadır.

Bu çalışmada da söz konusu sorunun çözümü için literatürde birçok farklı alanda uygulanan yöntemlerden olan zincir kod ve LCS yöntemlerinin bir çözüm adımı olarak kullanılabilirliği gösterilmiştir. Sistemin olabildiğince otomatize edilebilmesi için ise kullanıcıdan sadece yol genişlik bilgisi girilmesi istenmiştir. Girilecek bu değer sistemin başarımına direkt etki eden en önemli faktördür. Bununla birlikte zincir kod yönteminin kenar bulma algoritması sonucuna göre başarım göstermesinde dolayı kenar bulma algoritmasında yapılacak iyileştirmeler de sonuçlar üzerinde etki bırakmaktadır.

Zincir kod ve LCS yöntemlerinden ortaya çıkan başarım sonuçları her ne kadar yüksek olmasa da özellikle el ile öğrenim verisi girişi yapılması gereken adımlar için (makine öğrenmesi, bölge büyütme yöntemi vb.) bu işlemin otomatize edilmiş olması açısından uygulamada faydası olacaktır.

Bununla birlikte geliştirilen sistemin çıktıları bölge büyütme yöntemiyle daha da iyileştirilmiş ve literatürde çok daha fazla adımdan oluşan ve makine öğrenmesi içeren yöntemlere yakın başarım elde edilmiştir. Her ne kadar makine öğrenmesi üzerine yoğunlaşmış olan çalışmaların başarım değerleri daha yüksek olabiliyor olsa da sistemin eğitilmesi noktasında ciddi bir veri büyüklüğüne ve eğitim süresine ihtiyaç duyulmaktadır [8]. Bu çalışmada sunulan yöntem başarımı yüksek tutarken bu tarz süreçleri de azaltacaktır.

Geliştirilen yöntem sadece kara yolları için değil matematiksel olarak paralel kabul edilemeyen ama insan gözüyle bakıldığında büyük çoğunlukla paralel diye tespit ettiğimiz nehir, damar vb. birçok farklı nesnelere tespitinde kullanılabilir.

Yöntemin başarımının daha da artırılması ve şehir içi yolların da tespit edilebilmesi açısından makine öğrenmesi yöntemlerinin çalışmaya eklenmesi yapılabilecek şeylerden biridir. Bir başka iyileştirme yöntemi de üçten fazla kanala sahip uydu görüntüleri kullanmak suretiyle makine öğrenmesi adımı için daha fazla özellik elde edilmesidir.

Kaynaklar

- [1] M. Rajeswari, K. S. Gurumurthy, S. N. Omkar, J. Senthilnath, and L. P. Reddy, "Automatic road extraction using high resolution satellite images based on level set and mean shift methods," in *ICECT 2011 - 2011 3rd International Conference on Electronics Computer Technology*, 2011.
- [2] R. S. Babu, B. Radhakrishnan, and L. Padma Suresh, "Detection and extraction of roads from satellite images based on Laplacian of Gaussian operator," in *Proceedings of IEEE International Conference on Emerging Technological Trends in Computing, Communications and Electrical Engineering, ICETT 2016*, 2017.
- [3] S. Das, T. T. Mirnalinee, and K. Varghese, "Use of salient features for the design of a multistage framework to extract roads from high-resolution multispectral satellite images," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 10, pp. 3906–31, 2011.
- [4] C. Ünsalan and K. L. Boyer, *Multispectral Satellite Image Understanding*. 2011.
- [5] F. Ameri, M. Mobaraki, Z. Valadan, and M. Javad, "Semi-automatic extraction of different-shaped road centerlines from MS and pan-sharpened IKONOS images," *Remote Sens. Spat. Info. Sci.*, p. 37, 2008.
- [6] A. Kirthika and A. Mookambiga, "Automated road network extraction using artificial neural network," in *International Conference on Recent Trends in Information Technology, ICRTIT 2011*, 2011, pp. 1061–65.
- [7] L. Abraham and M. Sasikumar, "A fuzzy based road network extraction from degraded satellite images," in *Proceedings of the 2013 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2013*, 2013, pp. 2032–36.
- [8] R. Alshehhi and P. R. Marpu, "Hierarchical graph-based segmentation for extracting road networks from high-resolution satellite images," *ISPRS J. Photogramm. Remote Sens.*, vol. 126, pp. 245–60, 2017.
- [9] M. Maboudi, J. Amini, S. Malihi, and M. Hahn, "Integrating fuzzy object based image analysis and ant colony optimization for road extraction from remotely sensed images," *ISPRS J. Photogramm. Remote Sens.*, vol. 138, pp. 151–63, 2018.
- [10] R. Liu, J. Song, Q. Miao, P. Xu, and Q. Xue, "Road centerlines extraction from high resolution images based on an improved directional segmentation and road probability," *Neurocomputing*, vol. 212, pp. 88–95, 2016.
- [11] X. Liu, J. Tao, X. Yu, J. J. Cheng, and L. Q. Guo, "The rapid method for road extraction from high-resolution satellite images based on USM algorithm," in *Proceedings of 2012 International Conference on Image Analysis and Signal Processing, IASP 2012*, 2012.
- [12] H. Grailu, M. Lotfizad, and H. Sadoghi-Yazdi, "1-D chaincode pattern matching for compression of Bi-level printed farsi and arabic textual images," *Image Vis. Comput.*, vol. 27, pp. 1615–25, 2009.
- [13] H. Sánchez-Cruz, E. Bribiesca, and R. M. Rodríguez-Dagnino, "Efficiency of chain codes to represent binary objects," *Pattern Recognit.*, vol. 40, pp. 1660–74, 2007.
- [14] G. Q. Lu, H. G. Xu, and Y. B. Li, "Line detection based on chain code detection," in *2005 IEEE International Conference on Vehicular Electronics and Safety*

- Proceedings*, 2005, pp. 98–103.
- [15] E. Bribiesca, "A new chain code," *Pattern Recognit.*, vol. 32, pp. 235–51, 1999.
- [16] B. S. Anami and V. B. Pagi, "Acoustic signal based detection and localisation of faults in motorcycles," *IET Intell. Transp. Syst.*, vol. 8, no. 4, p. 345, 2013.
- [17] C. Gruber, T. Gruber, S. Krininger, and B. Sick, "Online signature verification with support vector machines based on LCSS kernel functions," *IEEE Trans. Syst. Man, Cybern. Part B Cybern.*, 2010.
- [18] G. Bourque and P. A. Pevzner, "Genome-scale evolution: Reconstructing gene orders in the ancestral species," *Genome Res.*, 2002.
- [19] Q. Wang, D. Korkin, and Y. Shang, "A fast multiple longest common subsequence (MLCS) algorithm," *IEEE Trans. Knowl. Data Eng.*, 2011.
- [20] D. Frolova, H. Stern, and S. Berman, "Most probable longest common subsequence for recognition of gesture character input," *IEEE Trans. Cybern.*, 2013.
- [21] D. Jarchi, C. Wong, R. M. Kwasnicki, B. Heller, G. A. Tew, and G. Z. Yang, "Gait parameter estimation from a miniaturized ear-worn sensor using singular spectrum analysis and longest common subsequence," *IEEE Trans. Biomed. Eng.*, 2014.
- [22] R. Adams and L. Bischof, "Seeded Region Growing," *IEEE Trans. Pattern Anal. Mach. Intell.*, 1994.
- [23] J. Canny, "A Computational Approach to Edge Detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 679–98, 1986.
- [24] M. Shah, "Fundamentals Of Computer Vision." Comp. Science Dep. University Of Central Florida, Orlando, 1997.
- [25] Ics.uci.edu, "Design and Analysis of Algorithms Lecture notes," *Dept. Information & Computer Science -- UC Irvine*. [Online]. Available: <https://www.ics.uci.edu/~eppstein/161/960229.html>. [Accessed: 06-Sep-2018].
- [26] Algorithmist.com, "Longest Common Subsequence," 2018. [Online]. Available: http://www.algorithmist.com/index.php/Longest_Common_Subsequence. [Accessed: 06-Sep-2018].
- [27] C. Poynton, *Digital Video and HD: Algorithms and Interfaces*. Morgan Kaufmann, 2003.
- [28] O. Marques, *Practical Image and Video Processing Using MATLAB®*. 2011.
- [29] R. C. Gonzalez, R. E. Woods, and S. L. Eddins, "Digital Image Processing Using Matlab - Gonzalez Woods & Eddins.pdf," *Education*. 2013.
- [30] J. S. Lim, "Two-Dimensional Signal and Image Processing," *Englewood Cliffs, NJ, Prentice Hall*, 1989.
- [31] J. Serra, *Image Analysis and Mathematical Morphology*. Academic Press Inc., 1982.
- [32] G. Kawade, S. Sahu, S. Upadhye, N. Korde, and M. Motghare, "An analysis on computation of longest common subsequence algorithm," in *2017 International Conference on Intelligent Sustainable Systems (ICISS)*, 2017, pp. 982–987.
- [33] C. Huang, Q. Liu, and X. Li, "Color image segmentation by seeded region growing and region merging," in *Proceedings - 2010 7th International Conference on Fuzzy Systems and Knowledge Discovery, FSKD 2010*, 2010.
- [34] D. Kaur and Y. Kaur, "Various Image Segmentation Techniques: A Review," *Int. J. Comput. Sci. Mob. Comput.*, 2014.
- [35] S. S. Kumar, M. Moorthi, M. Madhu, and R. Amutha, "An improved method of segmentation using Fuzzy-Neuro logic," in *2nd International Conference on Computer Research and Development, ICCRD 2010*, 2010.
- [36] P. K. Jain and S. Susan, "An adaptive single seed based region growing algorithm for color image segmentation," in *2013 Annual IEEE India Conference, INDICON 2013*, 2013.
- [37] J. Dehmeshki, H. Amin, M. Valdivieso, and X. Ye, "Segmentation of pulmonary nodules in thoracic CT scans: A region growing approach," *IEEE Trans. Med. Imaging*, 2008.
- [38] P. Lu, K. Du, W. Yu, R. Wang, Y. Deng, and T. Balz, "A new region growing-based method for road network extraction and its application on different resolution SAR images," *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.*, 2014.
- [39] C. Heipke, H. Mayer, H. Wiedemann, and H. Jamet, "Evaluation of automatic road extraction," *Int. Arch. Photogramm. Remote Sens.*, 1997.