NOKTA-PARSEL SORGULAMASI (POINT IN POLYGON TEST)

Doç. Dr. Ahmet KAYA ve Arş. Gör. Faruk YILDIRIM
Jeodezi ve Fotogrametri Mühendisliği Bölümü
Karadeniz Teknik Üniversitesi, 61080 Trabzon

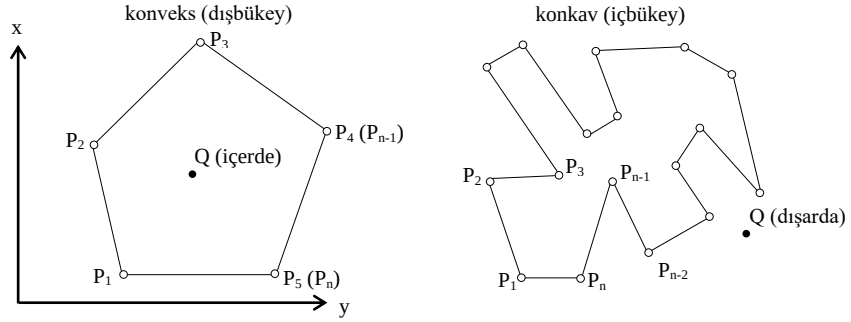
ÖZET: *Nokta parsel sorgulaması;konumu belirli bir noktanın köşe koordinatları ile belirli bir parselin içinde olup olmadığının sorgulanmasıdır. Bu sorgulama için farklı matematiksel ve geometrik yaklaşımlardan yola çıkılarak yöntemler geliştirilmiştir. Geliştirilen yöntemin konveks (dışbükey) ve konkav (içbükey) parsellerin her ikisi için de uygulanabilir olması arzu edilir. Sorgulamanın verimli ve hızlı olması; bir çok nokta sorgulamasının gerekeceği düşünülürse oldukça önem kazanır. Bu bildiride çözüm yöntemleri incelenecek, alternatif bir algoritmanın detayları verilecektir.*

1. GİRİŞ

Kent Bilgi Sistemleri ya da genel olarak Konumsal Bilgi Sistemi(CBS-GIS) oluşturulmasının temellerinde doğru ve güncel bir harita alt bilgisine ihtiyaç duyulur. Her türlü objenin öznitelik bilgileri yanında, koordinat yani konum bilgileri önemlidir ve bilgi sisteminin kullanılması birçok konuma dayalı sorgulama gerektirecektir. Burada öncelikle bir noktanın tanımlı bir parsele göre sorgulanmasında ortaya çıkan problemler ve çözüm yolları ele alınacaktır. Bu işlemden yüzlerce, binlerce tekrar olabileceği düşünülürse; algoritmanın-hesap tekniğinin hızı büyük önem kazanır. Tek nokta sorgulaması algoritmaları, iki veya çoklu nokta sorgulaması için de geliştirilerek kullanılabilirler.

Nokta parsel sorgulamasında; keyfi bir Q noktasının konumu ve verilen bir P parseli köşe noktalarının koordinatları ile bellidir. P parseli konveks (dışbükey) veya konkav (içbükey) olabilir. P parselinin kırık noktaları (P_1,P_2,...,P_n) ve Q noktası (x,y) koordinat çiftleriyle tanımlıdır. Nokta parsel sorgulamasında; eğer Q noktası P parselinin köşeleri içinde kalıyorsa Q noktası P parselinin içindedir, değilse dışındadır (Şekil. 1).

Sorgulama çok zor olmamakla beraber parselin konkav ve çok köşeli olması, sorgulanacak parsel sayısının fazla olması durumlarında, bilgisayarla bu sorgulama algoritmasının programlanması gerekir. Sorgulama bir çok parsel ve noktalar için yapılacağından program algoritmasının hızlı ve sade olması istenir. Bu nedenle konveks ve konkav parsellerin her ikisi ve sadece konveks parseller için birçok yöntem geliştirilmiştir.



Şekil 1. Konveks ve konkav parseller

2. SORGULAMA YÖNTEMLERİ

Yöntemler; konveks ve konkav parsellerin her ikisi ve sadece konveks parseller için kullanılan yöntemler olmak üzere iki başlık altında toplanmıştır [Bowyer&Woodwark, 1983; Preparate&Shamos, 1985; Kaya, 1992; Huang&Shih, 1997].

2.1. Konveks ve Konkav Parseller İçin Yöntemler

Bu başlık altında Işın Kestirme (Ray Intersection), Açıların Toplamı (Sum of Angles) ve Şerit (Swath) yöntemleri incelenecektir. Yöntemlerin şekilleri ve algoritma adımları verilecek ayrıca karşılaşılan güçlükler de anlatılacaktır.

2.1.1. Işın Kestirme Yöntemi

Q noktasından geçen bir doğru (ℓ) çizilir. Bu doğru genellikle koordinat eksenlerinden birine paralel olarak seçilir. ℓ doğrusu ile parsel kenarlarının kesişim sayıları hesaplanır. Kesişim için $ax+by+c=0$ doğru denklemi kullanılır. P_i-P_{i+1} parsel kenarları için $a_1x+b_1y+c_1=0$ (burada $a_1=dy$, $b_1=-dx$ ve $c_1=-x_idy+y_idx$ dir) ve ℓ için $a_2x+b_2y+c_2=0$ ($Q(x_0,y_0)$ olmak üzere burada $a_2= b_2=0$ ve $c_2=-x_0$ dir. Şayet ℓ doğrusu x eksenine paralel seçilseydi $c_2=-y_0$ olacaktı) doğru denklemleri kullanılır. Parsel kenarlarına ait doğru denklemlerinin ℓ 'nin doğru denklemiyle yaptığı kesişim koordinatları hesaplanır ($xkes_i=(b_1c_2-b_2c_1)/(a_1b_2-a_2b_1)$, $ykes_i=(a_2c_1-a_1c_2)/(a_1b_2-a_2b_1)$) Bu koordinatların Q' nun herhangi bir tarafında ($ykes_i<y_0$ veya $ykes_i>y_0$) ve parsel kenarlarının arasında olanların sayısı hesaplanır. Bu sayı tek ise Q noktası parselin içindedir, çift ise Q noktası parselin dışındadır (Şekil. 2).

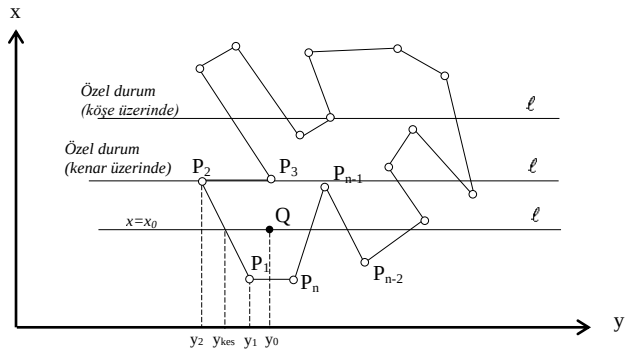
Işın ve parsel kenarları arasındaki kesişim sayısı daha kısa zamanda sorgulanabilir. Bunun için Q noktasının parsel kenarının köşe koordinatları ile yaptığı eğim açısı formülleri aşağıdaki gibi yazılarak, irdeleme de sadece eşitliğin sağ tarafları kullanılır.

$$tan(QP_i) = \frac{y_i - y_0}{x_i - x_0} = \frac{dy_i}{dx_i}$$

Q ve kenarlar arasında;şayet $dx_i dy_{i+1} > dy_i dx_{i+1}$ ise saat ibresi yönünde bir dönüş (dönüş=+1), $dx_i dy_{i+1} < dy_i dx_{i+1}$ ise saat ibresinin tersi yönünde bir dönüş (dönüş=-1) vardır. Q noktası ile parsel kenarının aynı doğrultu üzerinde olması durumunda ($dx_i dy_{i+1} = dy_i dx_{i+1}$); eğer ($dx_i dx_{i+1} < 0$) veya ($dy_i dy_{i+1} < 0$) ise dönüş=-1, eğer ($dx_i dx_i + dy_i dy_i >= dx_{i+1} dx_{i+1} + dy_{i+1} dy_{i+1}$) ise dönüş=0 ve diğer durumlarda ise dönüş=+1 dir.

Dönüş fonksiyonu ile kesişim (Q, Q_1, P_i, P_{i+1}) fonksiyonu da tanımlanır. Burada Q ve Q_1 ışının, P_i, ve P_{i+1} parsel kenarlarının uç noktalarını temsil etmektedir. Her iki doğru parçasının(ışın ve parsel kenarı) uç noktaları bir diğerinin farklı tarafında ise (farklı dönüş değerlerine sahipse) kesişir ve kesişme noktası vardır. Dolayısıyla kesişmeye karar vermek için dört tane dönüş fonksiyonu (a=dönüs(Q, Q_1, P_i,), b= dönüs(Q, Q_1, P_{i+1}), c=dönüs(P_i, P_{i+1}, Q) ve d=dönüs(P_i, P_{i+1}, Q_1)) tanımlamak gerekir. Eğer a*b ve c*d çarpımları sıfırdan küçük eşit ise kesişme vardır. Böylece tüm parsel kenarları için kesişme sayısı bulunarak Q noktasının parsel içinde olup olmadığına karar verilir.

Kesişme sayısını hesaplayan iki algoritma da bazı özel durumlar için geçerli değildir. Bunlar; Q' dan geçen ışının parsel kenarlarından birinin üzerinde olması ve parsel köşelerinden geçmesi durumlarıdır(Şekil 2).

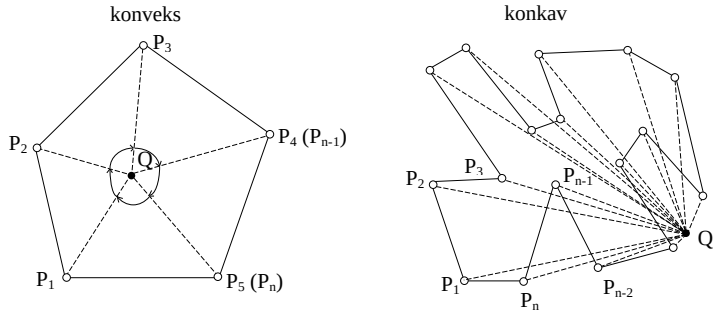


Şekil 2: Işın kestirme

2.1.2. Açıların Toplamı Yöntemi

Q noktası ile parsel köşeleri birleştirilir. Böylece QP_iP_{i+1} üçgenleri oluşur. Her üçgenin Q noktasındaki iç açısı hesaplanır ($P_i\hat{Q}P_{i+1}$ açısı (QP_{i+1}) ve (QP_i) semtler farkından). Bu açıların toplamı 360° ise Q noktası parselin içindedir. Açıların toplamı 360° den farklı ise Q noktası parselin dışındadır (Şekil. 3). Hesaplanan

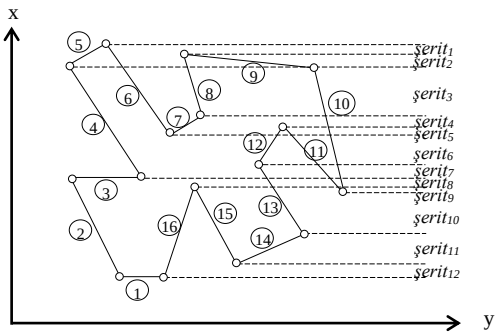
açının negatif veya 180° den büyük çıkması özel durumlardır. Açının +180° den büyük olması (konveks ve konkav parsellerin her ikisinde de) durumunda açı 360° den çıkartılır. Açının -180° den büyük olması durumunda; konveks parsellerde açı mutlak değerce hesaba katılır, konkav poligonlarda ise açı işaretiyle beraber hesaba katılır. Açının -180° den küçük olması (konveks ve konkav parsellerin her ikisinde de) durumunda ise açıya 360° eklenir. Bu yöntem açıların toplamından gelen yuvarlatma hatalarından etkilenebilir. Işın kestirme yöntemine göre sonuca daha çok zamanda ulaşır. Fakat ışın kestirme yöntemindeki özel durumlar bu yöntemde problem oluşturmaz.



Şekil 3: Açıların toplamı yöntemi

2.1.3. Şerit Yöntemi

Bu yöntemde; parselin tüm köşelerinden y eksenine paralel çizilen doğrularla, parsel şeritlere bölünür (Şekil. 4). Parsel köşelerinin x koordinatları büyükten küçüğe sıralanarak, şeritlerin alt ve üst x koordinatları tayin edilir. Q noktasının x_0 koordinatından yararlanılarak, Q noktasının hangi şeride düştüğü irdelenir. Daha sonra Q noktasını içeren şeritdeki parsel kenarları bulunur. Bu kenarlara ışın kestirme yöntemi uygulanarak, noktanın parselin içinde olup almadığı tayin edilir.



Şekil 4: Şerit Yöntemi

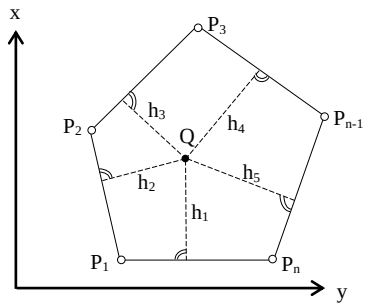
Bu yöntemde işlem adımlarının artmasına karşılık ışın kestirme yönteminde sorgulanacak özel durumlar ortaya çıkmaz.

2.2. Konveks Parseller İçin Yöntemler

Bu başlık altında İşaret Karşılaştırma (Sign of Offset), Alan Toplama (Sum of Area), Dönüş Yönü (Orientation) ve Kama (Wedge) yöntemleri incelenecektir. Yöntemlerin algoritmik adımları verilecek ayrıca karşılaşılan güçlüklere dikkat çekilecektir.

2.2.1. İşaret Karşılaştırma Yöntemi

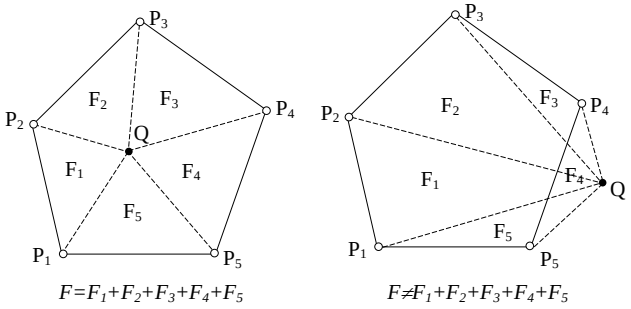
Q noktasının parselin tüm P_iP_{i+1} kenarlarına olan mesafeleri (h yüksekliği ki bu değer Q ve kenarın köşe koordinatlarından dik boyu hesabından bulunabilir) hesaplanır. Bütün h mesafeleri aynı işarete sahipse Q noktası parselin içindedir. h mesafeleri işaret değişikliğine uğruyorsa Q noktası parselin dışındadır (Şekil. 5).



Şekil 5: İşaret Karşılaştırma

2.2.2. Alan Toplama Yöntemi

Parselin her bir köşesi ile Q noktası birleştirilerek üçgenler oluşturulur. Bu üçgenlerin alanları toplamı parselin alanına(F) eşitse Q noktası parselin içinde değilse dışındadır (Şekil. 6).



Şekil 6: Alan Toplama

Bu yöntem programlama açısından işaret karşılaştırma yöntemine karşı daha kullanışlıdır. Çünkü alan hesabında kullanılan matematiksel işlemler, h yüksekliğinin hesabına nazaran daha azdır.

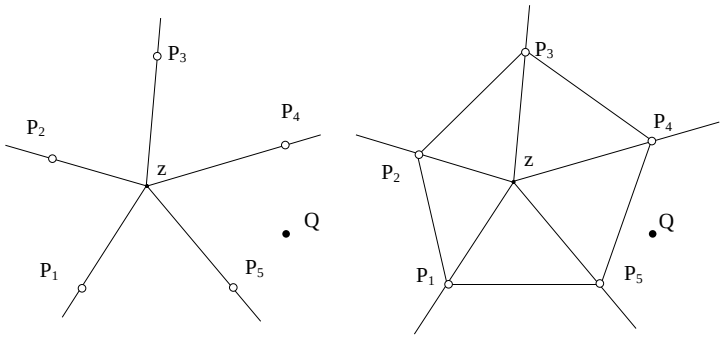
2.2.3. Dönüş Yönü Yöntemi

Parselin her bir köşesi ile Q noktası birleştirilerek üçgenler oluşturulur. Bu üçgenlerin QP_iP_{i+1} köşelerinin dönüş yönüne (işaretine) bakılır. İşaret değişikliği yoksa Q noktası parselin içindedir. İşaret değişikliği varsa Q noktası parselin dışındadır.

Bu yöntemle alan toplama yönteminin her ikisinde de alan hesaplaması yapılmasına rağmen bu yöntem daha avantajlıdır. Çünkü alan toplama yönteminde alanlar hesaplanarak biriktirilmektedir, bu yöntemde biriktirmeye gerek olmayıp sadece işarete bakılır.

2.2.4. Kama Yöntemi

Parselin içine düşen bir z noktası tayin edilir. z ile tüm parsel köşeleri birleştirilerek üçgen bölgeler oluşturulur (Şekil. 7). Q noktasının bu bölgelerden hangisine düştüğü irdelenir. Q noktasının bulunan üçgenin içinde olup olmadığı diğer üç yöntemden, sonuca ulaşma en kısa zamanda gerçekleşen dönüş yönü yöntemi tercih edilerek yapılmalıdır. Bu yöntem konveks parseller için kullanılan yöntemler içinde sorgulama zamanı en az olan yöntemdir.



Şekil 7: Kama Yöntemi

3. ALTERNATİF YÖNTEMLER

Burada anlatılacak yöntem konveks ve konkav parsellerin her ikisi içinde geçerlidir [Taylor, 1994; Sedgewick, 1988].

3.1. Nokta-Vektör Gösterim Yöntemi

Işın kestirme yönteminin algoritması (Şekil 2) de gösterilen özel durumlar için (köşe noktasından geçme ve kenar üzerinde olma) yeterli değildi. Nokta-Vektör gösterimi ile ışın belirlenerek bu özel durumları da içeren bir algoritma elde edilmektedir.

Düzlemde bir doğrunun bir çok gösterim tarzı vardır. Baş (x_0,y_0) ve son (x_1,y_1) noktasının koordinatları ile tanımlı bir L doğru parçasının vektörel formu

$$L = \left\{ (x_0, y_0) + (r[x_1 - x_0], r[y_1 - y_0] \mid 0 < r < 1 \right\}$$

denklemiyle tanımlıdır.

Nokta-Vektör Algoritması

Düzlemde L ve L′ doğru parçaları verilmiştir. L ve L′ doğru parçaları sırasıyla P_0 , P_1 ve P'_0 , P'_1 noktaları arasında tanımlanmıştır (Şekil. 8). İki doğrunun kesişim problemlerinde doğru parçalarının P_0 , P_1, P'_0 ve P'_1 noktalarının koordinatları bellidir. L ve L′ doğrularının kesişmesi için

$$P_0 + r(P_1 - P_0) = P'_0 + r'(P'_1 - P'_0)$$

eşitliğinin sağlanması gerekir. Noktaların koordinat değerlerinden

$$x_0 + r(x_1 - x_0) = x'_0 + r'(x'_1 - x'_0)$$
$$y_0 + r(y_1 - y_0) = y'_0 + r'(y'_1 - y'_0)$$

denklemleri yazılır. r ve r′ değerleri bu denklemden bir tek çözüme sahiptir. Bu değerler aşağıdaki determinantların hesabıyla bulunabilir.

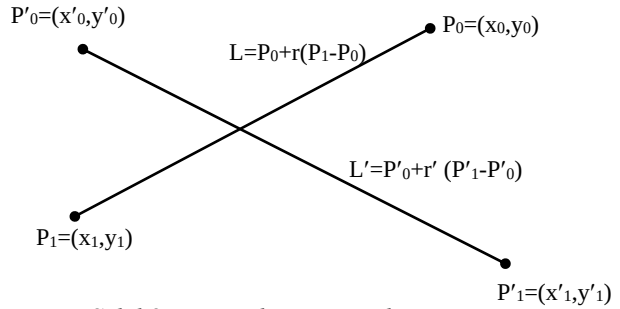
$$D = \begin{vmatrix} (y_1 - y_0) & -(y'_1 - y'_0) \\ (x_1 - x_0) & -(x'_1 - x'_0) \end{vmatrix}$$

$$D_1 = \begin{vmatrix} (y'_0 - y_0) & -(y'_1 - y'_0) \\ (x'_0 - x_0) & -(x'_1 - x'_0) \end{vmatrix}$$

$$D_2 = \begin{vmatrix} (y_1 - y_0) & (y'_0 - y_0) \\ (x_1 - x_0) & (x'_0 - x_0) \end{vmatrix}$$

Buradan $r=D_1/D$ ve $r'=D_2/D$ hesaplanır. r ve r′ değerleri [0,1] arasında ise L ve L′ doğru parçaları bir noktada kesişir. Eğer *D* determinantı sıfıra eşitse L ve L′ doğru parçaları paraleldir. Eğer *D*, D_1 ve D_2 determinantları sıfıra eşitse L ve L′ doğru parçaları çakışıktır yani üst üstedir. Programlama açısından paydanın sıfır olma ihtimali olduğundan bölme işleminden kurtulmak için sadeleştirmeye gidilebilir. Bunun için; D determinantı sıfıra eşit değil ve r sayısı [0,1] aralığında değer alıyorsa $r(1-r)=(D_1D-D_1^2)/D^2$ değeri her zaman pozitiftir. Payda (D^2) her durumda pozitif olduğundan $(D_1D-D_1^2)$ değeri de pozitiftir.

Dolayısıyla $D_1(D-D_1)$ *ve* $D_2(D-D_2)$ değerleri negatif değilse doğru parçaları kesişir.



Şekil 8:L ve L′doğru parçaları

Işın Seçimi ve Sorgulama

Tüm parsel kenarlarıyla yapılacak kesişimleri sorgulamak için, ışınında vektörel olarak ilk ve son koordinatları belli bir doğru parçası olarak tanımlanması gerekir. Sorgulama için verilen Q noktasının (x_0,y_0) koordinatları ışının ilk noktasıdır. Işın vektörel olarak

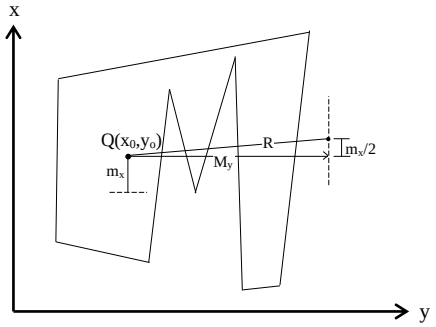
$$R = \left\{ (x_0, y_0) + r(m_x, 2M_y) \,\middle|\, r > 0 \right\}$$

tanımlanırsa ışının son nokta koordinatları r=1 olmak üzere

$$(x', y') = (x_0 + m_x, y_0 + 2M_y)$$

elde edilir. Buradaki M_y ve m_x ışına belli bir eğim vermek için kullanılır. Böylece ışın (şekil 2) deki özel durumların hiç biri gibi olmayacaktır. M_y için $|y_0 - y_i|$ mutlak değer farkı en büyük değer alınır. m_x için $|x_0 - x_i|$ mutlak değer farkı en küçük olan sıfırdan farklı değer alınır (Şekil. 9).

Nokta parsel sorgulamasında (x_0,y_0) dan (x′,y′) ye kadar olan R ışını, parselin tüm S_i kenarları sorgulanarak kesişme sayısı bulunur. Kesişimlerin sayısına



Şekil 9: Işın seçimi

D_i (D_i - D_i) (1)
D_i D_i (2)

determinant çarpımlarının hesaplanmasıyla varılır. Burada ki değerler

$$D_i = \begin{vmatrix} 2M_y & -(y_{i+1} - y_i) \\ m_x & -(x_{i+1} - x_i) \end{vmatrix}$$

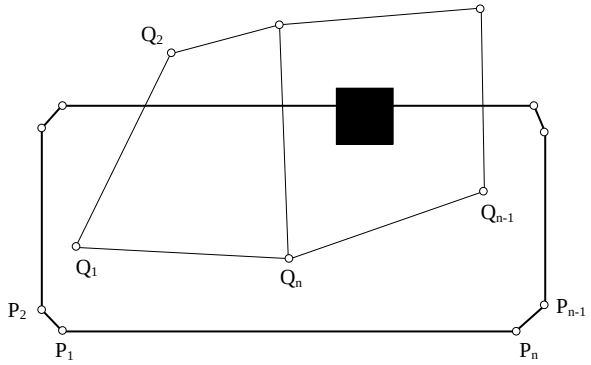
$$D_i' = \begin{vmatrix} 2M_y & (y_i - y_0) \\ m_x & (x_i - x_0) \end{vmatrix}$$

$$D_i'' = \begin{vmatrix} (y_i - y_0) & -(y_{i+1} - y_i) \\ (x_i - x_0) & -(x_{i+1} - x_i) \end{vmatrix}$$

determinantlarıyla hesaplanır. Eğer (1) ve (2) pozitifse R ışını parselin S_i kenarını keser. Diğer durumlar da kesmez. Böylece kesişim sayısı hesaplanır. Bu sayı tek ise Q noktası parselin içinde, çift ise parselin dışındadır.

4. SONUÇ

Jeodezi uygulamalarında nokta parsel sorgulaması, imar uygulamalarında parsel, bina ve yol gibi detayların imar adalarının içinde kalıp kalmadığı, kalıyorsa tamamı mı yoksa ne büyüklükte bir alanının kaldığının irdelenmesinde ve hatta arsa ve arazi düzenleme uygulamasında düzenleme sınırı ile kadastro parsellerinin sorgulamasında kullanılır(Şekil. 10).



Şekil 10:

Uygulamalar da çok noktanın sorgulanacağı düşünülürse; sorgulamanın, kısa sürede bitmesi, tüm şekiller de(konveks veya konkav) yapılması ve bütün özel durumlar için problem oluşturmaması istenir. Bunun için bir çok yöntem geliştirme yoluna gidilmiştir. Alternatif yöntemler başlığı altında verilen yöntem ve ışın seçiminin bu yöntemde belirtilen şekilde seçilmesi durumunda ışın kestirme yönteminde verilen ikinci algoritma tüm bu özellikleri kapsamaktadır.

KAYNAKLAR

Bowyer, A., and Woodwark, J. (1983) *A Programmer's Geometry*, Butter and Tanner Ltd, London.

Huang, C. W., and Shih, T. Y. (1997) *On The Complexity Of Point-in-Polygon Algorithms,* Vol. 23, No. 1, pp. 109-118.

Kaya, A. (1992) *Parsellerin Bölünmesinde Doğrudan Çözüm Yöntemleri ve Programlama için Algoritmalar*, İmar Planlarının Uygulanması Semineri, KTÜ Mühendislik Mimarlık Fakültesi, s.167-178, Trabzon.

Preparate, F. P., and Shamos, M. I. (1985) *Computational Geometry-an Introduction*, Springer-Verlag, New York.

Sedgewick, R. (1988) *Algorithms*, Addison-Wesley Publ. Company, New York.

Taylor, G. (1994) *Point in Poligon Test*, Survey Review, 32, 254, pp. 479-484.