**NOKTA ÖZELLİK ÇIKARIMI İLE RAYLARDAKİ VİDALARIN TESPİTİ**

Özellikler belirli bir görüntü ile ilgili hesaplama adımlarını çözmekle ilgili olan bir bilgi parçasıdır.

Özellikler görüntülerdeki noktalar kenarlar veya nesneler ve kenarlar gibi belirli yapılar olabilir.

Buradaki özellikler genel olarak bir komşuluk işlemi ile özelliklerin algılanması sonucunda bulunurlar.

Tespit etmede tanımlanan noktaların özelliklerinin diğer resimlerde olması durumunda görselde tespit edilir.

Her özellik noktasının etrafındaki görünümde aydınlatma , ölçek ve düzlemdeki değişiklikler altında değişmeyen özellikler olarak seçilir. Ve her özellik noktası için bir tanımlayıcı vektör elde edilir.

Benzer özellikleri belirlemek için tanımlayıcılar görseller arasında birbirleriyle kıyaslanır.

İki resim için bir çift (Xi,Yi) -->(Xi’,Yi’) şeklinde elde edilir. Burada (Xi,Yi) bir görseldeki bir özelliktir.

(Xi’,Yi’) ise diğer görselle kıyaslanıp onun ile eşleşen özelliktir.

**Özellik Noktası**

Özellik noktası, dokuda ifade edici olan noktadır. Özellik noktası, nesnelerin sınırlarının aniden değiştiği noktadır veya iki veya daha fazla görseldeki parçaların arasında kalan kesişme noktasıdır.

**Özellik Tanımlayıcı**

Özellik tanımlayıcı, bir görseldeki özellik vektörlerini çıkaran bir algoritmadır. Özellik tanımlayıcı görüntünün bir nevi bir dizi sayısal kodlamasıdır görüntüde aradığımız nesnenin parmak izidir.

**Eşleşen Özellikler**

Görsellerin içerisindeki tespit edilmek istenen nesnelerin arasındaki karşılıklı ilişki kurmasıdır.

Görsel eşleştirilmesi için kullanılan yaygın bir yaklaşım şudur;

Her biri görseldeki verilerden görsel tanımlayıcıları ile ilişkili bir dizi özellik noktasının tespit edilmesinden oluşmaktadır. Özellikler ve özellik tanımlayıcıları iki veya birden fazla görüntüden çıkarıldıktan sonra, bir sonraki adım bu görüntüler arasında ön özellik eşleşmeleri oluşturmaktır.

**Surf Algoritmasına genel bakış**

Surf yöntemi Hızlandırılan belirgin özelliklerin benzerliklerinin gösterimi ve görsellerin karşılaştırılması için hızlı ve çok sağlam bir algoritmadır.

Surf yaklaşımının temel ilgilendiği konu kutu filtreleri kullanarak işlemlerin hızlı hesaplanması ve böylece izleme nesne tanıma gibi gerçek zamanlı uygulamalara karşı bir kolaylık sağlamasıdır.

**Surf algoritması iki adımdan oluşmaktadır.**

**Özellik çıkarma**

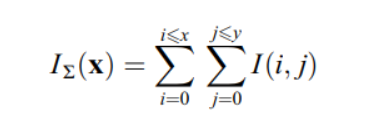
Hedef noktası saptama yaklaşımıdır. Çok basit olarak bir hessian matris yaklaşımı kullanır.

**İntegral görüntüler**

İntegral görüntü veya Toplam Alan Tablosu 1984 yılındı tanıtılmıştır. İntegral görüntü, hedef bir görüntüdeki değerlerin yani piksellerin toplamını hesaplamanın hızlı ve etkili bir yolu olarak kullanılır. Buna ek olarak birde görüntüdeki ortalama yoğunluğun hesaplanması içinde kullanılabilir.

Bu algoritmada kutu şeklinde filtrelemenin hızlı hesaplanmasına olanak tanır X = (x,y)ᵀ

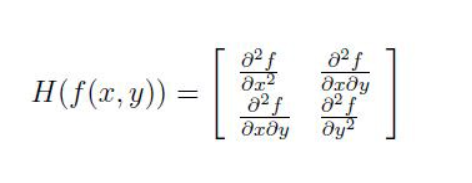
Konumunda bulunan bir integral görüntüsünde I\_∑(x)) girişi, orjin ve x tarafından oluşturulan bir dikdörtgen bölge içindeki girdi görselindeki I tüm piksellerin toplamına eşit olur (Temsil eder).



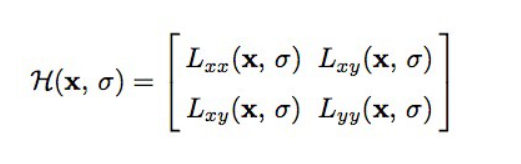
I\_∑ Hesaplandığında, herhangi bir dikdörtgen alan üzerindeki yoğunlukların toplamını ve boyutundan bağımsız olarak hesaplanması için yalnızca dört ekleme yapmak yeterlidir.

Hessian matrisi tabanlı özellik noktaları

Surf yönteminde hesaplama süresi ve doğruluğundan yüksek orandan dolayı hessian matrisi kullanılır. Konum ve ölçeklerin seçmek için farklı bir ölçü kullananmaları yerine (Hessian-Laplace detektörü) surf her ikisi içinde Hessian matrisinin determinantına dayanmaktadır. Bir piksel eğer hesaplanması için algoritmaya verildiğinde bu pikselin değeri şu şekilde olmaktadır.



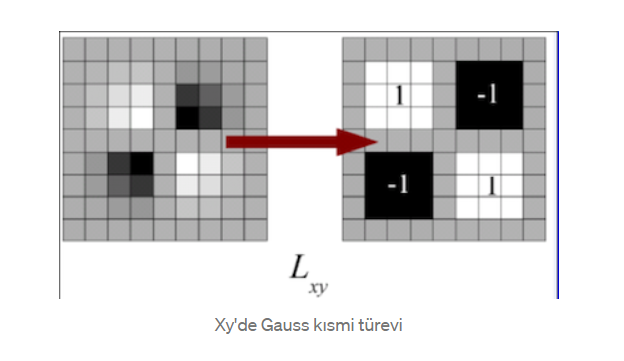
Eğer herhangi bir ölçeğe uyarlanmak istenirse görüntüi, bir Gauss çekirdeğine göre filtrelenir, bu nedenle X = (x,y) noktası verildiğinde σ ölçeğinde x’deki Hessian matrisi H(x,σ) şeklinde tanımlanır.

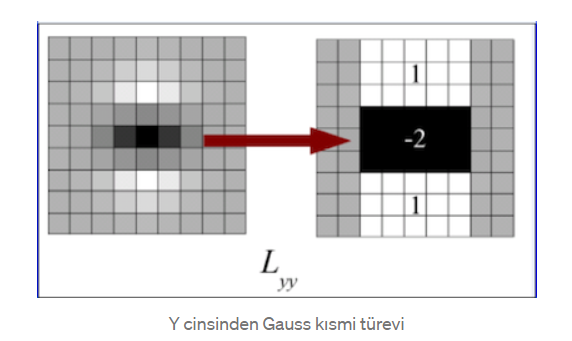


Bu kısımda Lxx (x,σ), Gauss ikinci dereceden türvein x noktasında I ile evrişimidir. Ve benzer şekilde Lxy(x,σ) ve Lyy(x,σ) içinde geçerlidir.

Gauss yöntemi ölçek alanı analizi için idealdir, ancak pratikte, ayrıklaştırılmaları ve kırpılmaları gerekmektedir. Bu 4’ün tek katları etrafından görüntü rotasyonları altında tekrarlanabilirlikle birlikte bir kayba yol açar. Bu zayıflık genel olarak Hessen merkezli dedektörler için geçerlidir. Bununla birlikte, dedektörler hala iy bir performans sergiliyor ve performanstaki hafif kötü gidişat, ayrıklaştırma ve kırpmanın getirdiği hızlı kavramı avantajından daha ağır basmıyor.

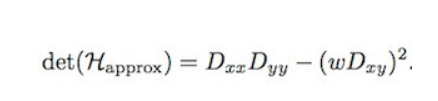
Hessian matrisinin determinantını hesaplamak için, önce Gauss çekirdeği ile evrişimi , ardından ise ikinci dereceden türev uygulamalıyız. Bu şekilde boyuttan bağımsız olarak çok düşük hesaplama maliyetiyle hesaplamalar yapılabilir. Bu sebep aslında temel olarak surf yönteminin hızlı olmasının nedeninin bir parçasıdır.





Görüntülerde 9 x 9 kutu filtreleri σ =1.2 olan Gauss ikinci dereceden türevler için yaklaşık değerdir.

Bu yaklaşımları Dxx ve Dyy ve Dxy ile gösterilir.



**Ölçek alanı gösterimi**

Ölçek uzayları genelde görüntü piramitleri olarak uygulanır. Görüntüler bir Gauss ile tekrar tekrar düzleştirilir. Ve arından daha yüksek bir piramit seviyesi elde edilmesi için alt örnekleme yapılır. Kutu filtrelerinin ve bütünleşik görüntülerin kullanılması nediyle, surf önceden filtrelenmiş bir katman çıktısına aynı filtreyi yinelemeli olarak uygulanması zorunlu değildir. Bunun yerinde herhangi bir boyuttaki uygulayabilir.

**Özellik Tanımlama**

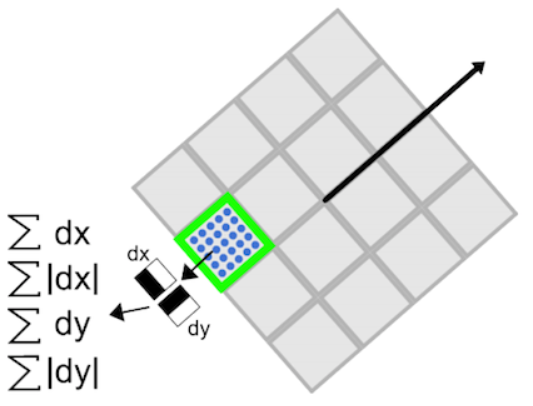
Surf yönteminin oluşturulması iki adımda gerçekleşir. İlk adım, anahtar nokta etrafındaki dairesel bir bölgeden alınan bilgilere dayalı olarak yeniden üretilebilir bir yönelim sabitlemekten oluşur. Ardından, seçilen yöne hizalanmış bir kare bölge oluşturuyoruz ve surf tanımlayıcısını ondan çıkarılır.

**Oryantasyon Ataması**

1. Surf yöntemi ile önce Haar-dalgacıklarını x ve y- yönlerinde hesaplayın ve bu, anahtar noktanın tespit edildiği ölçek s ile anahtar noktasının etrafındaki 6 yarı çaplı dairesel bir alanda ve örnekleme adımı ölçeğine bağlıdır ve s olarak seçilir . Dalgacık yanıtları bu mevcut ölçekte s ‘ de hesaplanır. Buna göre yüksek ölçeklerde dalgacıkların boyutu büyüktür. Bu nedenle, hız filtreleme için yeniden entegre görüntüler kullanır.
2. Daha sonra ise bir tarama alanındaki diket ve yatay dalgacık yanıtlarının toplamı hesaplanır, ardından tarama yönü değiştirilir (π/3) ve en büyük toplam değeri olan yönelimi bulana kadar yeniden hesap yapılır.

**Tanımlayıcı Bileşenler**

1. İlk olarak anahtar nokta etrafında ortalanmış ve oryantasyon boyunca yönlendirilen bir kare bölge oluşmasından ibarettir. Bu bölgenin boyutu 20s’dir.
2. Daha sonrasında ise bölge düzenli olarak daha küçük parçalara mesela 4 x 4 kare bir bölge olacak şekilde bölünür. Bu bölünen bölgelere alt bölge ismi verilir. Bölünen her bir alt bölge için 5 x 5 düzenli aralıklarla yerleştirilmiş örnek noktalarda bir kaç basit özellik hesaplanır. Basit olabilmesi için dx Haar yatay yönde dalgacık ve dy Haar dikey yönde dalgacık olacak şekilde düzenlenir. Oluşabilecek Geometrik deformasyonlara ve lokalizasyon hatalarına karşı dayanıklığı arttırmak için, dx ve dy yanıtları ilk olarak anahtar noktasında ortalanmış bir Gauss(σ = 3.3s)ile ağırlıklandırılmalıdır.



Daha sonra ise dx ve dy dalgacık yanıtları her bir alt bölge üzerinde toplanır ve özellik vektörüne bir ilk girdi kümesi oluşturur. Yoğunluk değişikliklerin kutupsallığı hakkında bilgi getirmek için, yanıtların mutlak değerlerinin toplamından çıkarılır.

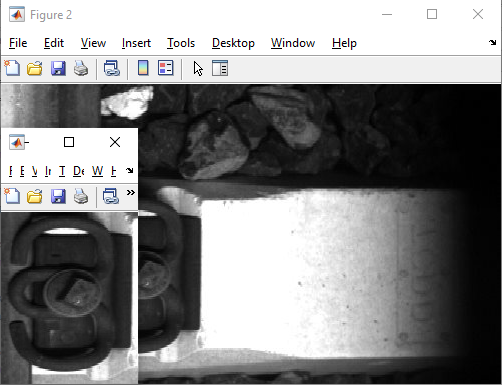
| dx | ve | dy | Bu nedenle her bir alt bölge aslında temelindeki yoğunluk yapısı olan

****V = (∑ dx, ∑ dy, ∑ | dx |, ∑ | dy |)****

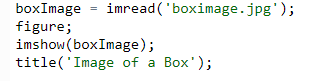
**Için dört boyutlu bir v tanımlayıcı vektörüne sahiptir. Bu , 64 uzunluğundaki tüm 4 x 4 alt bölgeler için bir tanımlayıcı vektörle sonuçlanır Sift yönteminde bu kısımda kullandığımız 64 uzunluğunda vektör 128 boyutunda olduğundan dolayı Surf yönteminin hız bakımından üstünlüğünü görmüş oluyoruz.**



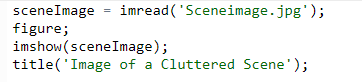
Surf yöntemi akış diyagramı



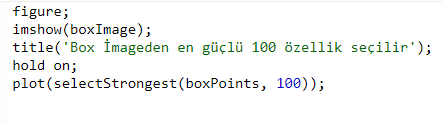
Bu kısımda belirlediğimiz boxImage olan küçük resim boxImage değişkenine atıldı ve ekranda gösterildi.



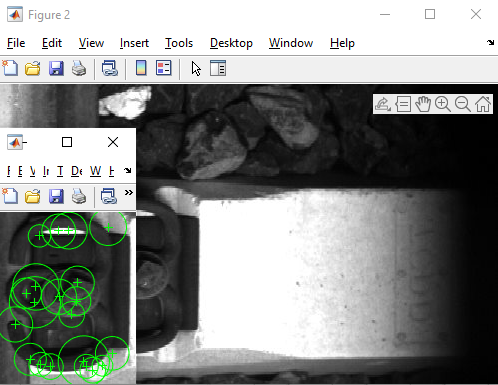
Küçük resim ekranda gösterilir.

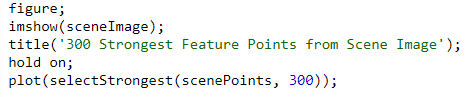


Büyük resim yani sahne resmi sceneImage değişkeninin içerisine atılarak ekranda gösterildi.

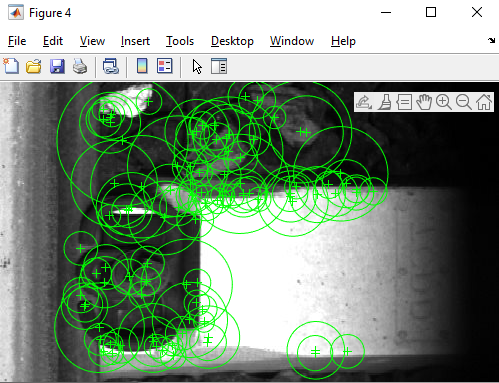


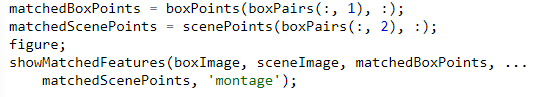
En belirgin 100 özellik seçilerek özellik noktaları belirlendi.



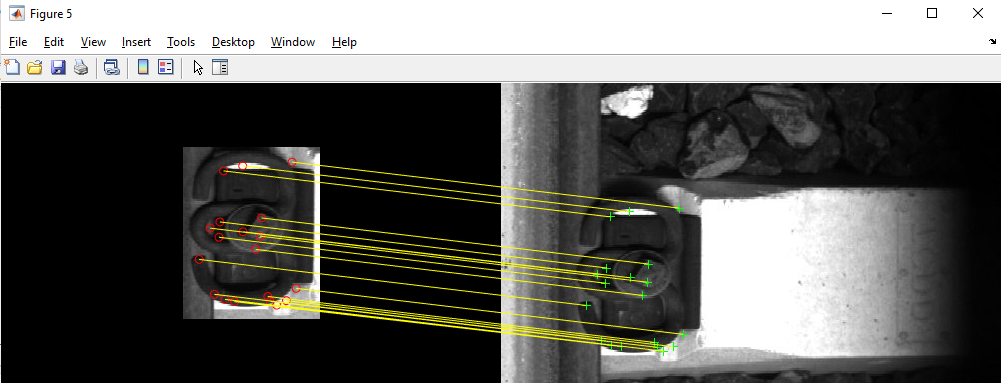


Hedef görüntüde yani sceneImagede bulunan 300 belirgin özellik belirlendi.



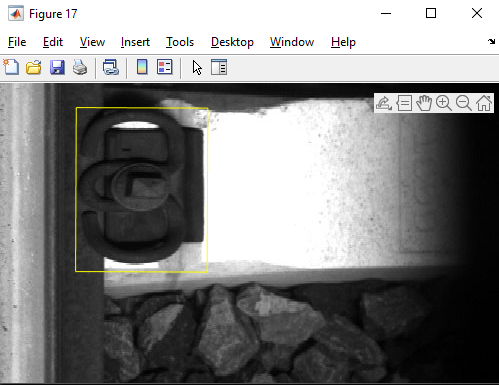


Her iki resimde bulunan ortak özellikler birbirleriyle eşleştirilerek ekranda gösterildi.



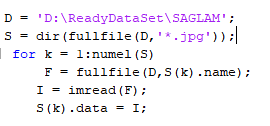
Küçük görsel ve büyük görseldeki eşleşen en belirgin özellik noktaları görselleştirilmiş ve ekranda görünmektedir.

Burada özellik çıkaramında önemli olan nokta en belirgin özellikler belirlenirken fotoğrafın birebir aynısını kullanmakdır. Bu işlem yapılırken kıyaslama tarafında birbirlerinin aynı görüntüleri olduktaları için bu kıyaslama kısmında aynıda olsa farklı bir görsel kullanmak algoritmanın daha verimsiz hale getirecektir.



En belirgin özellikler çıkarıldıktan sonra küçük resimde belirlenen özellikler büyük resimin içerisinde arandı ve büyük resimde küçük resime benzerliği bulunan kısım sayı çizgilerle algoritma yardımıyla belirlendi. Ve bu şekilde tespit edilen nesneyi görüntüledik.

Daha sonrasında dosyanın içerisinde bulunan bütün görseller bu şekilde taranarak büyük resimlerin içerisinden küçük resim tespit edilmeye başlandı.





Örnek olarak burada dosyada bulunan bütün veriler özellikleri çıkarılan küçük resim ile karşılaştırılarak büyük resimlerden tespit edildi.

2 tane resimde tam doğru olmayan iki tespit yapıldı . Yaklaşık 20 veri içerisinden 2 hatalı ray tespiti yapıldı.