SONDAJ KÖPÜKLERİ ÖZNİTELİKLERİNİN GÖRÜNTÜ İŞLEME TEKNİKLERİ İLE ÇIKARIMI VE YAPAY SİNİR AĞLARI KULLANARAK VERİ ANALİZİ

VELİ MERT ALTAŞ

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ

TOBB EKONOMİ VE TEKNOLOJİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2007
ANKARA

Fen Bilimleri Enstitü onayı

_____

Prof. Dr. Yücel ERCAN
Müdür

Bu tezin Yüksek Lisans derecesinin tüm gereksinimlerini sağladığını onaylarım.

_____

Prof. Dr. Ali YAZICI
Anabilim Dalı Başkanı

Veli Mert ALTAŞ tarafından hazırlanan SONDAJ KÖPÜKLERİ ÖZNİTELİKLERİNİN GÖRÜNTÜ İŞLEME TEKNİKLERİ İLE ÇIKARIMI VE YAPAY SİNİR AĞLARI KULLANARAK VERİ ANALİZİ adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

_____

Yrd. Doç. Dr. A. Murat Özbayoğlu
Tez Danışmanı

Tez Jüri Üyeleri

Başkan : Yrd. Doç. Dr. M. Evren Özbayoğlu _____

Üye : Yrd. Doç. Dr. A. Murat Özbayoğlu _____

Üye : Yrd. Doç. Dr. Bülent TAVLI _____

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada orijinal olmayan her türlü kaynağa eksiksiz atıf yapıldığını bildiririm.

…..……

Veli Mert Altaş

Üniversitesi : **TOBB Ekonomi ve Teknoloji Üniversitesi**
Enstitüsü : **Fen Bilimleri**
Anabilim Dalı : **Bilgisayar Mühendisliği**
Tez Danışmanı : **Yrd. Doç. Dr. Ahmet Murat Özbayoğlu**
Tez Türü ve Tarihi : **Yüksek Lisans – Haziran 2007**

Veli Mert ALTAŞ

SONDAJ KÖPÜKLERİ ÖZNİTELİKLERİNİN GÖRÜNTÜ İŞLEME TEKNİKLERİ İLE ÇIKARIMI VE YAPAY SİNİR AĞLARI KULLANARAK VERİ ANALİZİ

ÖZET

Kimyasal köpükler, bulundukları sıvıya göre düşük yoğunluk ve yüksek akmazlık gibi özelliklerinden dolayı, sıvı yoluyla taşımanın önemli olduğu flotasyon, sondaj gibi endüstriyel işlemlerde ön plana çıkmaktadır. Taşıyıcı sıvıya yardımcı olan köpüklerin boyut, biçim gibi özniteliklerinin taşıma kapasitesini doğrudan etkilediği bugüne kadar yapılan araştırmalarda ortaya konulmuştur. Yapılan çalışmada köpüklerin görsel özniteliklerini görüntü işleme metotlarıyla algılayacak ve YSA kullanılarak veri analizi yapabilecek bir model geliştirilmesi amaçlanmıştır. Bu amaca göre bölgesel eşikleme, Fourier Dönüşüm üzerinden filtreleme gibi metotlar uygulanmış ve elde edilen sonuçlar aktarılmıştır. Modele göre görüntü üzerindeki noktaların potansiyel bir kabarcık merkezi olup olmadığı geliştirilen bir algoritma ile kontrol edilmiştir. Elde edilen potansiyel köpüklere açısal minimum takip, ağırlık merkezi bulma ve sınır takibi gibi metotlar uygulanmıştır. Sınırları belirlenen köpüklerin yarıçap, alan, çevre gibi boyut ve biçim özellikleri çıkarılmış, kimyasal veriler ile birleştirilerek YSA kullanılarak veri analizi yapılmıştır. Yöntemde kullanılan tekniklerin algoritma açısından faydaları zamana dayalı olarak gösterilmiştir. Geliştirilen görüntü işleme modelinin, özellikle gürültülü köpük görüntülerinde literatürde yaygın olarak kullanılan yöntemlere göre daha başarılı olduğu gözlemlenmiştir. YSA kullanılarak yapılan veri analizinin sonuçları incelenmiştir. Yapılan inceleme sonucu görüntü işleme metodu ile elde edilen köpük verilerinin YSA ile analizinin mümkün olduğu gösterilmiştir.

Anahtar Kelimeler: Görüntü işleme, Köpük Görüntülerinin Analizi, Kabarcık Görüntülerinin Analizi, Yapay Sinir Ağları, Örüntü Tanıma.

University : **TOBB University of Economics and Technology**
Institute : **Institute of Natural and Applied Sciences**
Science Programme : **Computer Engineering**
Supervisor : **Assistant Prof. Dr. Ahmet Murat Özbayoğlu**
Degree Awarded and Date : **M.Sc. – June 2007**

Veli Mert ALTAŞ

EXTRACTING THE FEATURES OF DRILLING FOAMS WITH IMAGE PROCESSING TECHNIQUES AND DATA ANALYSIS USING ARTIFICIAL NEURAL NETWORKS

ABSTRACT

The chemical foams have a significant role in industrial processes, which use liquid carrying such as flotation and drilling, because of their low density and high viscosity compared to the liquid that they are in. Up to date research showed that, the features of foam like shape and dimension have a direct effect on the carrying capacity. This study aimed to develop a model that perceive the features of foams with image processing techniques and make a data analysis using artifical neural networks. The methods like local thresholding and filtering from fourier transformation are applied and their results are shown according to this purpose. As to this model, a developed algorithm checks if the pixels on the image are a possible center of a potential froth or not. The techniques like tracing angular minimum, finding center of weight and contour tracing are applied to obtained potential froths. Radius, area, perimeter and other shape and dimension properties of the froths, whose borders are designated, are derived and combined with chemical data to perform a data analysis using artificial neural network.The algorithmic benefits of techniques used in the method, over time are shown. It is observed that, the developed image processing model is more successful than the techniques widely used in literature, especially with noisy froth images. The results of data analysis performed by artifical neural network are studied. Results show that it is possible to perform froth data analysis by neural networks using the extracted froth data by image processing techniques.

Keywords: Image Processing, Analysis of Froth Images, Analysis of Bubble Images, Artificial Neural Network, Pattern Recognition

TEŞEKKÜR

Çalışmalarım boyunca değerli yardım ve katkılarıyla beni yönlendiren tez danışmanım Yrd. Doç. Dr. Ahmet Murat Özbayoğlu'na ve değerli bilgileri ve tecrübeleri ile bana yardımcı olan Yrd. Doç. Dr. Mehmet Evren Özbayoğlu'na, yine kıymetli tecrübelerinden faydalandığım TOBB Ekonomi ve Teknoloji Üniversitesi Bilgisayar Mühendisliği Bölümü öğretim üyeleri ve yüksek lisans arkadaşlarıma, bana verdikleri manevi desteklerden dolayı ailem ve arkadaşlarıma teşekkürü bir borç bilirim.

İÇİNDEKİLER

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

KISALTMALAR

Kısaltmalar	Açıklama
BOP	Belirleyici Olmayan Polinom
ÇKP	Çok Katmanlı Perseptron
YGF	Yüksek Geçirgen Filtre
DHD	Dairesel Hough Dönüşümü
EDT	Elektrikli Direnç Tomografisi
FD	Fourier Dönüşümü
İS	İşletim Sistemi
KLAHE	Kontrast Limitli Adaptif Histogram Eşitlemesi
KYM	Kırmızı Yeşil Mavi
RDD	Renk Doygunluk Değeri
REB	Rasgele Erişimli Bellek
TFD	Ters Fourier Dönüşümü
YRD	Yoğunluk Renk Doygunluğu
YSA	Yapay Sinir Ağı
ZST	Zaman Serisi Tahmini

SEMBOL LİSTESİ

Bu çalışmada kullanılmış olan simgeler açıklamaları ile birlikte aşağıda sunulmuştur.

Simgeler	Açıklama
B	Görüntü noktasının mavi bileşeni ya da yarıçap değerine göre ortalamaların tutulduğu matris
c	Derece adımı
C	Alınan kesitlere ait ortalamaların tutulduğu matris
d	Öklid mesafesi
D	Uzaklık vektörleri matrisi
$g_{\max}$	Azami gri seviye değeri
$g_{\min}$	Minimum gri seviye değeri
G	Görüntü noktasının yeşil bileşeni
G_k	Göreceli kırmızılık değeri
i	Görüntü matrisi üzerinde yatay koordinat
I	Görüntünün gri seviye matrisi
j	Görüntü matrisi üzerinde dikey koordinat
J	KLAHE sonrası elde edilen matris
L	Etiketleme matrisi
m	Görüntüden alınan kesitin yatay bileşeni
M	Görüntünün yatay boyutu
n	Görüntüden alınan kesitin dikey bileşeni
N	Görüntünün dikey boyutu
P	Potansiyel köpükler matrisi
$P(f)$	Kümülatif olasılık dağılımı
r	Köpük yarıçap değeri
r'	Açısal minimuma göre belirlenen ortalama yarıçap
R	Görüntü noktasının kırmızı bileşeni
S	Alınan kesitlere ait standart sapma değerlerinin tutulduğu matris
$\overline{X}_g$	Gri seviye ortalaması
$\overline{X}_k$	Kırmızı seviye ortalaması
ω	Ağırlık katsayısı
α	Görüntüyü yatay boyutlarda ayıran parça değeri
β	Görüntüyü dikey boyutlarda ayıran parça değeri
γ	Yarıçapa göre ortalama aralığı hassasiyeti
δ	Yuvarlaklık oranı
η	Çizilebilen açı oranı
η'	Çizilen açısal oran koşulu
θ	Dairesel açı
$\theta(n)$	Karmaşıklık göstergesi
λ	Tarama hassasiyeti

μ	Belirli açı ve yarıçapa göre minimum değer
Φ	Azalma veya artma koşulu
Ω	Dönüş sayısı

İndisler	**Açıklama**

Üsler	**Açıklama**

BÖLÜM 1

1. GİRİŞ

1.1. Giriş ve Çalışmanın Amacı

Köpük, yüzey gerici bir kimyasal ile güçlendirilmiş bir sıvı faz ile gaz fazının yüksek kayma hızı altında homojen bir şekilde karıştırılması ile elde edilen bir akışkandır. Sıvı faz, yüzey gerici kimyasalın etkisi ile gaz fazını hapsederek, durağan bir süspansiyon oluşturur. Ancak, yerçekimi etkisi ile sıvı faz zamanla hapsedici etkisini yitirir ve köpük bozulmaya başlar. Sürekli olarak bir kayma gerilimi uygulanması durumunda, köpüğün bozulması engellenebilir. Sondaj köpüğü, traş köpüğünü andıran bir yapıya sahip iken, flotasyonda elde edilen köpük, deterjan köpüğüne benzer.

Flotasyon, sondaj gibi endüstriyel işlemlerde kullanılan köpükler, bulundukları sıvıya göre yüksek akmazlık, düşük yoğunluk gibi özelliklerinden dolayı sıvı yoluyla taşıma işlemlerinde, sıvıya yardımcı olmaktadırlar. Köpüklerin boyut, biçim gibi özniteliklerinin taşıma kapasitesini doğrudan etkilediği bugüne kadar yapılan araştırmalarda ortaya konulmuştur. Köpüklerin özelliklerinin taşıma işlemlerine etkisi bilindiği için endüstride bu kontrolü sağlamak üzere deneyimli bir operatöre ihtiyaç duyulmaktadır. Diğer yandan bu kontrol gözlemsel yeteneğe dayalı olduğu ve bilimsel bir analiz yordamı kullanılmadığı için her zaman yeterli sonuç veremeyeceği ortadadır.

Köpüklerin boyut, biçim gibi görsel özelliklerinin çıkarılabilmesi ve bu verilerin işlenmesi için bugüne kadar birçok araştırma yapılmıştır. Ele alınan köpük görüntüleri, araştırmalara göre farklılık gösterdiği için genel bir model ortaya çıkarılamamıştır. Özellikle mikroskobik ya da yakın plan alınan görüntüleri amaçlayan ve kabul edilebilir bir başarım sağlayabilen bir araştırma bugüne kadar geliştirilememiştir.

Yapılan çalışmada bu konudaki eksikliğin giderilmesi ve bu yönde bir modelin oluşturulması amaçlanmıştır. Çalışmanın 2. bölümün ilk kısmında, bu amaca yönelik olarak araştırma yapılmış, kullanılabilecek ve karşılaştırılabilecek yöntemler ve sonuçları bilgisayar bilimleri açısından incelenmiştir.

Bu bölümde temel olarak köpük boyut ve biçimlerini anlamaya yönelik görüntü işleme metotları ele alınmıştır. Öncelikli olarak bölgesel eşikleme ve FD frekans spektrumuna göre filtreleme ve köpük kenarı tespit yöntemleri denenmiştir. Denemeler sonucu istenilen başarıya ulaşılamadığı için, özgün bir algoritma ve buna bağlı bir dizi metotlar geliştirilmiştir. Bu metotlara göre her piksel için potansiyel köpük olup olmadığı, polar koordinatlardaki gri seviye düşüş değeri üzerinden kontrol edilmiştir. Çıkarılan potansiyel köpük merkezlerinin tam tespiti için ortak köpüklere ait ağırlık merkezini hesaplayan yöntem geliştirilmiş ve merkezlerin kayması indirgenmiştir. Diğer yandan yarıçapların tam tespiti için açısal olarak minimum değere göre takip şeklinde açıklanabilen özgün bir metot geliştirilmiştir.

Elde edilen potansiyel merkezlerin birbirleri ile sınır girişimi durumuna karşı kontrol yapılmış ve görüntülere göre seçilebilen bir hassasiyete göre filtreleme uygulanmıştır.

Fazla bölütlenmeyi indirgemek için minimuma yönelen bir sınır takip metodu geliştirilmiş ve görüntülere göre seçilen açısal oranı takip edemeyen köpükler filtrelenmiştir. Ayrıca bu metot ile literatürdeki eksiklik giderilmiş ve köpüklerin biçimlerinin tam olarak algılanması sağlanmıştır.

Biçime göre yapılan sınır takiplerinin ardından her köpüğe ait alan için hücresel olarak etiketleme yapılmıştır. Bu etiketlemeye göre polar koordinatlardan, kartezyen koordinatlara dönüşürken oluşan hücresel boşlukları ortadan kaldıran ve iki köpüğün sınırlarının çakıştığı durumlar için karar verebilen bir düzenek geliştirilmiştir. Bu mekanizmada komşuların salt çoğunluğuna göre doldurma tekniği uygulanmış ve sonuçları incelenmiştir.

Bulunulan bölgeye sığabilecek azami kabarcık değerinin algılanması için uzaklık vektörüne göre kontrol yapılmıştır. Bunun neticesinde algoritmanın hızlandırılması sağlanmış ve bunun etkileri test edilerek zamana dayalı olarak incelenmiştir.

3. bölümde geliştirilen modelin görüntü işleme yöntemleri, belirgin sınırlara ve yuvarlak hatlara ve belirsiz sınırlara ve gürültülü yüzeye sahip iki farklı görüntü tipi üzerinde denenmiş ve sonuçları incelenmiştir. Elde edilen sonuçlar literatürde yaygın kullanılan DHD ve su bendi gibi tekniklerini sonuçları ile karşılaştırılarak sonuç analizi yapılmıştır.

Elde edilen köpük verileri için ÇKP modeli kullanılarak YSA modeli uygulanmıştır. 3. bölümde bu modele göre görüntüler, eğitim, test ve doğrulama aşamalarında kullanılmış ve sonuçları incelenmiştir. İnceleme sırasında elde edilen boyut, biçim gibi değerlerin önceden elde edilen gerçek verilerle ilişkisi ele alınmıştır. Bu ilişkiye göre, aynı görüntüler bu konuda yapılan başka bir çalışmanın sonuçları ile karşılaştırılmış ve neticeleri aktarılmıştır.

Bu konuda yapılacak devam niteliğindeki çalışmalar için modelin eksik yönleri belirtilmiştir. Gelecekte yapılacak çalışmalara ışık tutabilmesi açısından geliştirilebilir ve eklenebilir yanları fikirsel olarak ifade edilmiştir.

BÖLÜM 2

2. KÖPÜK BOYUTLARININ GÖRÜNTÜ İŞLEME YÖNTEMLERİ İLE BULUNMASI

2.1. Kaynak Araştırması

Bilindiği üzere gaz ve buharların sıvı katmanları ile çevrelenmesi sonucu sıvı maddeler içerisinde kabarcıklar oluşmaktadır. Köpükler göreceli olarak düşük yoğunluk ve yüksek akmazlık özelliklerine sahiplerdir. Endüstriyel sondaj işlemlerinde köpükler, yüksek akmazlık özelliğinden dolayı düşük oranlı gaz enjeksiyonlarına göre daha verimli taşıma yapmaktadır [1].

Bir boru içerisinde herhangi bir akışkanın hareketi sırasında, hareket yönünde bir enerji kaybı meydana gelir. Akışkan hızı borunun duvar yüzeyinde "0" iken, borunun merkez ekseninde azami değerini alır. Borunun duvar yüzeyindeki ve borunun merkezindeki bu hız farklılığı, akışkanın kendi içerisinde bir sürtünme kuvveti meydana gelmesine yol açar, ve dolayısıyla, bir enerji kaybı gözlenir. Bu enerji kaybı, akış sırasında basınç azalması olarak tespit edilir. Sonuç olarak, akış yönünde bir basınç düşümü oluşur. Akışkanın fiziksel ve kimyasal özellikleri, borunun çapı, akışkan hızı gibi faktörler, basınç kaybının derecesini etkileyen faktörlerdir.

Kabarcıklar ve köpüklerden oluşan görüntülerden, kabarcıkların ve özelliklerinin çıkarılması konusunda çok sayıda çalışma yapılmıştır. 1992 yılında P.J. Symonds ve G. De Jager tarafından yapılan araştırmada [2] deneyimli operatörlere dayandırılarak, kabarcık görüntülerinin drenaj hakkında bilgi taşıdığı belirtilmiştir. Çevrim yapıldığında drenajın, flotasyon hücresinin derecesi ve geri kazanım göstergesi olarak ele alınabileceği ifade edilmiştir. Diğer yandan görsel veriler olarak ele alınan temel kabarcık özellikleri ise büyüklük, biçim ve akış hızları olarak gösterilmiştir [2,3]. Bu veriler sayesinde kabarcıkların karakteristik özellikleri ortaya koyulabilmekte, optimizasyon ve köpüklerin endüstriyel süreç kontrolü

sağlanabilmektedir [3]. Diğer yandan yapılan çalışmada [2], köpük sınırlarının ve dolaylı bir etken olarak kabarcıklar üzerindeki parlak noktaların görsel olarak ele alınabilecek özellikler olduğu belirtilmiştir. Ele alınan parlak noktaların kameranın göreli konumuna, kabarcıkların boyutları, biçimleri ve yüzey eğimlerine, ışık dağılımına ve köpüğün yansıtıcı özelliğine bağlı olarak kimyasal yapılarının göz önünde bulundurulması gerektiği ifade edilmiştir. Ayrıca köpük sınırlarının çevresindeki diğer bölgelere göre daha koyu bir renk değerine sahip olmasından ötürü köpükler arasındaki sınırların gri renk değerlerine dayalı bir vadi olarak ele alınabileceği önerilmiştir. Kabarcıkların kapladıkları alanları bulmak için sınırların tespitinin önemi vurgulanmıştır. Bunların dışında yine köpük görüntülerini ayırmada karşılaşılan tipik problemler aşağıdaki şekilde sıralanmıştır:

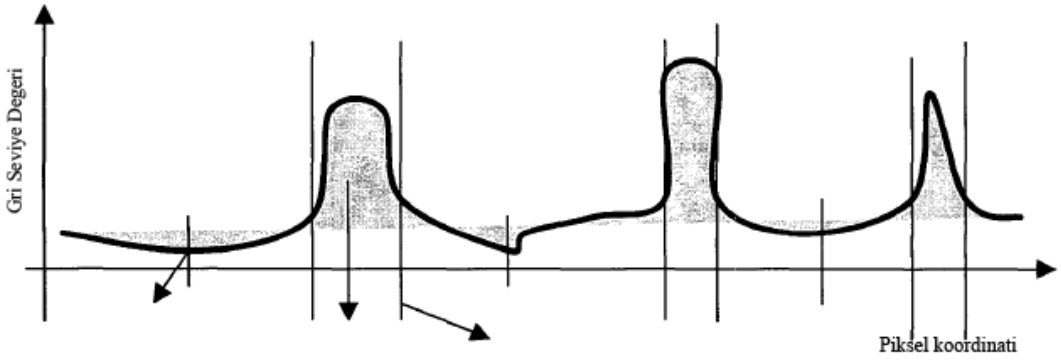
- Işık kaynağının dağınık olmamasından kaynaklanan parlak noktaların doygunluğu.
- Birden fazla ışık kaynağı kullanımında çoklu parlak noktanın elde edilmesi.
- Işıklandırma açısının çok geniş olması durumunda, sınırları belirsizleştiren gölge alanlar oluşması.

Diğer yandan 1997 yılında N. S. Kajemi ve J.J. Cillier ortak çalışmasında köpük görüntülerinin düşük kontrasta sahip ayrı homojen nesnelerden oluştuğu için, gri seviyede uygulanan tekniklerin kabarcıkları zorlukla ayırt edebildiğini vurgulamışlardır [3]. Bunların dışında endüstriyel ve ışık gibi çevresel etmenlerin de bu yöndeki algoritmaların gelişiminde zorluk çıkardığını ifade etmişlerdir.

1996 yılında D. W. Moolman ve ekibi tarafından yapılan diğer bir çalışmada ise görüntü üzerinden çıkarılan sayısal verilerin yararı ve sayısal verilerin kabarcıklar hakkında ne gibi bir fikir verebileceği ortaya konulmuştur [4]. Homojenlik, işlenmemişlik, saflık gibi özellikleri temsil eden sayısal değerlerin, köpük karakteri hakkında “ideal”, “akıcı” ve “yapışkan” gibi işe yarayan özelliklere dönüştürülmesi açıklanmıştır.

W. Wang ve O. Stephansson tarafından 1999 yılında, 4. Avrupa Birliği Bilgi Teknolojileri programına dahil olarak yürütülen ChaCo projesinde [5], önceden ortaya konulan [2-4] köpük görüntülerinden çıkarılabilecek özelliklerin dışında, yüzlerce köpük görüntüsü incelendikten sonra şu sonuçlara varılmıştır:

- Kabarcıklar birbirine değmekte ve kabarcıkların aralarında hiçbir boşluk bulunmamaktadır.
- Kabarcıklar üzerindeki aydınlatma, düzensiz ve eğri bir görüntü sergilemektedir.
- Kabarcıklar arasındaki kenarlar görsel açıdan zayıftır.
- Spekülatif bölgelerde (parlak bölgeler) Şekil 2.1'de görüldüğü üzere keskin renk geçişleri bulunmaktadır.



Şekil 2.1. Köpük görüntüleri üzerindeki parlak noktalar [5]

Yine 1999 yılı içerisinde J.J.Cillliers, M. Wang ve S.J. Neethling tarafından yapılan çalışmada [6] ise, daha önce yapılmış olan araştırmalar bir adım ileriye götürülerek, kabarcıklar arasında oluşan birleşmeler ve EDT yöntemiyle köpük boyutları bulmaya çalışılmıştır. Araştırmada köpük özelliklerinin maden üretimi ve kağıtları mürekkepten arındırma gibi sanayi işlemlerinde kullanışlı olduğuna değinilmiştir. Ele alınan köpük özellikleri boyutlar, hareketlilik ve köpüklerin taşıyabildiği katı miktarına odaklanmıştır. Bunun yanında köpük değerlerinin isabetli ve hızlı çıkarılmasının işlem kontrolü ve hata teşhislerindeki önemine değinilmiştir ayrıca

köpük boyutlarının, kabarcık birleşmelerinin atış oranlarının tahmin edilmesinde önemli rol oynadığına dikkat çekilmiştir.

1999 yılında yapılan bir başka araştırmada [7] Bonifazi ve ekibi ise bu konuyu endüstriyel anlamda ele almış ve flotasyon köpüklerinin kontrolünün incelenmesi için deneyimli bir insana ihtiyaç duyulduğundan bahsetmiştir. Bu incelemenin sonucuna göre, sisteme verilen kimyasal bileşen oranları yeniden ayarlanmakta ve performans artırımı sağlanabilmektedir. Ayrıca bu karmaşık kontrol tekniklerinin uzun süren ayarlamalarını düzenleyebilmek için deneyimli bir mühendise ihtiyaç duyulduğuna fakat hem zaman hem de masraf açısından bunun her zaman mümkün olmadığına dikkat çekilmiştir. Bunun dışında flotasyon işlemine etki eden faktörler arasında maden cevherinin derecesi, parçacık büyüklüğünün dağılımı, parçacık morfolojisi, parçacık yüzeyleri gibi özelliklerin etkisinin olduğu açıklanmıştır [8]. Öte yandan yapılan araştırma tarihine kadar olan çalışmaların yetersiz olduğu dile getirilmiştir.

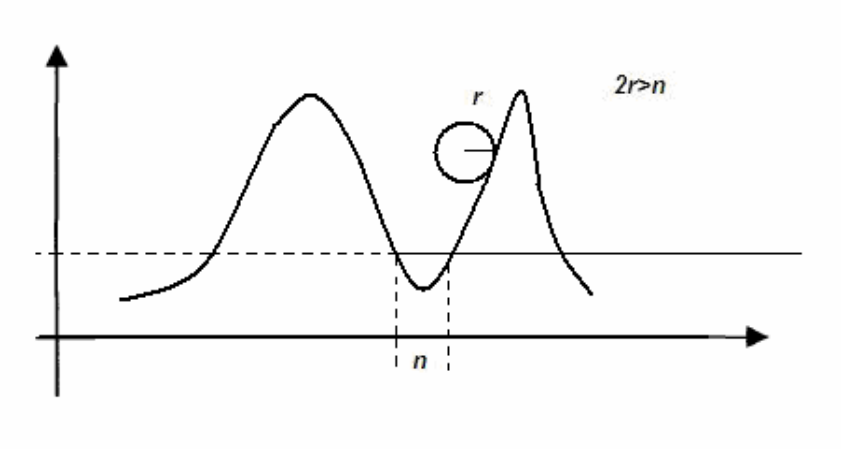
Aynı ekip, çalışmalarına 2000 yılında devam ederek köpük flotasyonunun, yüzdürülen maden cevherinin özelliklerine ve işlem değişkenlerine bağlı olduğunu belirtmiştir [9] ve o ana kadar yapılan çalışmaların genel bir teori ya da kurallar bütünü ortaya koyamadığını bunun yerine özel durumlara göre başarılı spesifik çözümler ürettiğini belirtmiştir.

2003 yılında ise W. Wang, önceden yapmış olduğu çalışmalarına [5,10] devam ederek ekibiyle birlikte çok daha geniş kapsamlı bir Avrupa Birliği destekli "CHACO EC" projesini yürütmüştür [11]. Bu projede, önceki çalışmalarda [2,3,5,10] ortaya konulan görsel kabarcık ve köpük özellikleri ele alınmıştır. Yıllarca bu konuda geliştirilen doku tabanlı ve morfolojik bölütlendirme algoritmalarının yeterli düzeyde başarılı olmadığı vurgulanmıştır; çünkü doku tabanlı algoritmalar görüntü üzerindeki evrensel değerleri bulabilmesine rağmen kabarcıkların teker teker bulunmasında yetersiz kalmaktadır ve su bendi gibi morfolojik metotlar sadece azami değerlerin kolay bulunabilindiği görüntülerde başarılı olabilmektedir. Ayrıca tek tip bir bölütlendirme metodunun farklı büyüklüğe sahip köpük görüntülerinde

düzgün çalışamayacağı ifade edilmiştir, bu yüzden de görüntülerin farklı boyutlara göre ayarlanması gerektiği ortaya konulmuştur. Bunların dışında kabarcıkların karakteristik özelliklerinin ortaya konulabilmesi için yüzlerce köpük görüntüsü incelenmiş ve bölütlendirme testleri uygulanmıştır. Elde edilen sonuçlara göre kabarcıkları ayırabilmek için yoğunluk benzerliği kullanılamamaktadır; çünkü Şekil 2.1'de de görüldüğü gibi köpükler arasındaki gri seviye farkları çok azdır ve bu fazla veya az bölütlenmeye yol açabilir. Köpük sınırları arasındaki değişim farkı az olmasına rağmen parlak noktalarda fazla olduğundan klasik kenar bulma yöntemleri yetersiz kalmaktadır. Morfolojik bölütlendirme algoritmaları sadece azamilerin kolay bulunduğu görüntülerde kullanılabilir [3]. Sonuçta küçük kabarcıklara sahip bir görüntüde çok önemli detay bilgilerin dikkate alınmaması az bölütlenmeye, büyük kabarcıklara sahip görüntülerde fazla detayın kaldırılmamasının fazla bölütlemeye yol açacağı rahatlıkla anlaşılmaktadır. 2006 yılında bu projenin [11] devamı niteliğinde yapılan araştırmada [12] teknolojik gelişmelerden yararlanılarak, yüksek çözünürülüklü dijital kameralardan elde edilen köpük görüntülerinden araştırma yapılmıştır. Bölütleme işlemlerinin hızlandırılması için öncelikle düşük çözünürlüklü hallerinden sınıflandırma yapılmıştır. Ardından kenar bulma yöntemleri ve benzetmeye dayalı bir bölütlendirme ile sonuca ulaşılmıştır.

Yapılan araştırmalarda ortaya konulan özelliklerin tespiti için genel değerlendirmelerin dışında bazı çalışmalarda görüntü işleme anlamında algoritmalar tasarlanmıştır. Örneğin 1992 yılında P.J.Symonds ve G. De Jager tarafından geliştirilen algoritmada [2] klasik sabit değerli eşiklemenin yetersiz kaldığı ispatlanmıştır. Farklı deneme yanılma metotlarıyla ideal değerler üzerinden bir dizi morfolojik işlemler (genleşme, erozyon, kapama, açma) uygulanmış ardından kenar ayrımlarını yakalamak için "yuvarlanan küre" tekniği [13] kullanılmıştır. Yuvarlanan küre tekniği, daha önceden geliştirilmiş vadi kenar metodu [14] temel alınarak tasarlanmıştır. Bu iki tekniğin birleşimi sayesinde gereksiz gürültü olarak varsayılan çukur bölgelere girilmeden gerçek kabarcıklar arasındaki vadi-sınır ayrımları fark edilmiştir. Şekil 2.2'de açıkça ifade edilen sisteme göre, ani düşüşlerin gürültü ya da köpük sınırı olduğu seçilen küre yarıçapına göre belirlenebilmiştir. Dolayısıyla

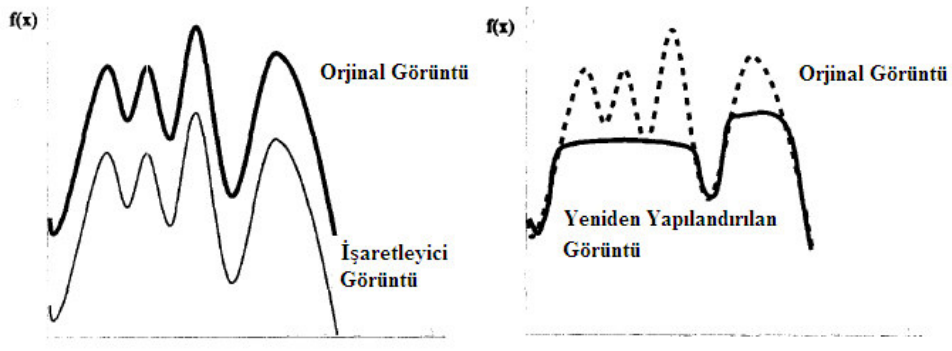
algoritmada yere paralel görüntü yüzeyi üzerinde hayali bir küre gezdirilerek, bu kürenin girdiği çukurların köpükler arasındaki sınırlar olduğu ortaya çıkmıştır.



Şekil 2.2.Yuvarlanan küre ile vadi kenar bulma tekniği

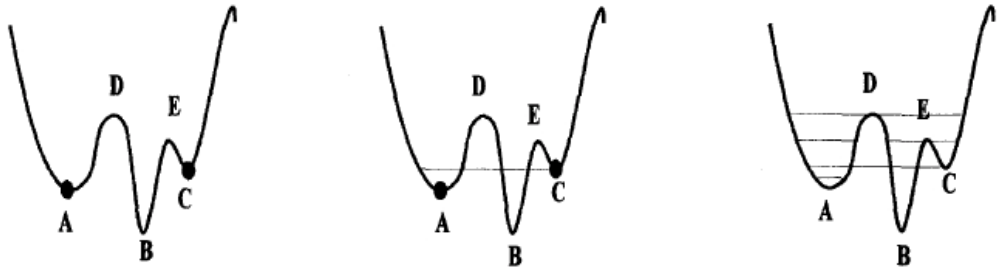
Sonuç olarak çalışmada [2] küresel yapıda bir elemanı, ayırıcı olarak seçmenin güvenilir olabileceği yine de hatalardan tam anlamıyla arındırılamayacağı ortaya çıkmıştır. Bunun için sayım şeklinde ele alınan parlak noktaların daha doğru sonuçlar elde edilmesine yardımcı olduğuna değinilmiştir.

1997 yılında N. S. Kajemi ve J.J. Cillier tarafından geliştirilen algoritmada[3] ise görüntü üzerinde öncelikle histogram denkleştirimi tekniği uygulanmış ve Şekil 2.3'te görüldüğü gibi önceden geliştirilen [15] bir teknik ile sabit bir renk değeri bütün piksel renk değerlerinden çıkarılarak görüntü üzerinde yeniden yapılanma sağlanmıştır.



Şekil 2.3. Gri seviye yeniden yapılanma tekniği [3]

Bölgesel azami değerler tespit edilerek su bendi olarak geliştirilen yöntem [16,17] Şekil 2.4'te görüldüğü gibi uygulanmış ve resim köpüklere göre parçalanmıştır.



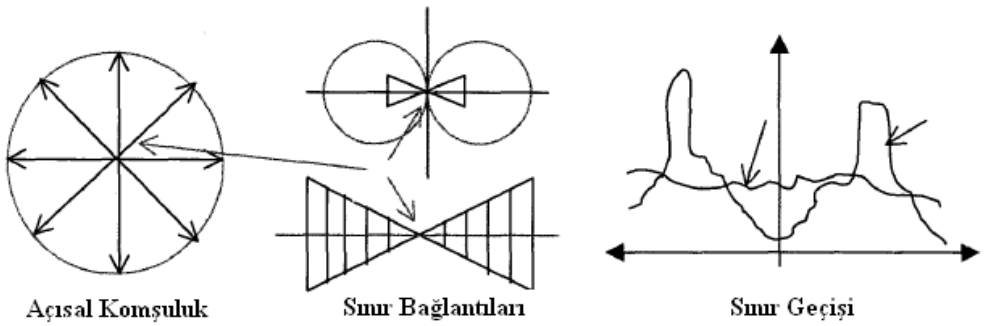
Şekil 2.4. Su bendi kenar bulma yöntemi [3]

Bu tekniğe göre A, B, C noktaları bölgesel minimumlar olarak belirlendikten sonra, uçta kalan A ve C minimumları başlangıç işaretleyicileri olarak seçilmiştir. Bu noktalar arasında kalan azamilerin arasından minimumları geçen D değeri ayırıcı olarak seçilmiş, E gibi taşmayı engelleyemeyen noktalar ise seçilmemiştir. Ayrım yapıldıktan sonra kabarcık alanlarının hesaplanması için köpüklerin kapladıkları piksel sayıları dikkate alınmış ve parçalara ayrılan resim, gerçek resimle üst üste bindirilmiştir. Son olarak elde edilen veriler istatistiksel analiz için uygun hale getirilmiştir. Bunun sonucunda çalışmada [3] kabarcıkların boy ve biçimlerinin flotasyon köpükleri için bir performans göstergesi olacağı vurgulanmıştır. Ayrıca endüstriyel köpük nitelendirme ve kontrol sistemlerinde modelin kullanıcı desteğine ihtiyaç olmaksızın uygulanabileceği belirtilmiştir.

Yaklaşık aynı yıllarda Wang ve Stephansson tarafından yürütülen çalışmada [5] ise, literatürde bilinen vadi-kenar tarama algoritmasına [14] dayalı bir yöntem geliştirilmiştir. Bunun yanında Şekil 2.1'deki gri seviye renk analizinde görüldüğü gibi, parlak noktaların keskin geçişler yaratmasından dolayı klasik kenar bulma yöntemlerinin çalışamayacağı vurgulanmıştır. Dolayısıyla kabarcık kenarlarının net olarak çıkarılabilmesi ve köpük üzerindeki parlak noktalardan etkilenmemesi için algoritmada ilk adımda, Gauss filtresi [18] uygulanarak görüntü üzerinde iyileştirme yapılmıştır ve köpükler arasındaki vadi ayraçlarının bulunmasının gerekliliği ortaya

konulmuştur. Bütün piksellerin bulundukları yere göre en düşük nokta olup olmadığı teker teker araştırılmıştır. Bu düşük nokta analizinde önceden belirlenmiş bir eşik değeri üzerinden koşullama yapılmıştır fakat eşik değerinin vadilerin seçiminde belirsizlik oluşturduğu durumlarda ise bulanık mantıktan yararlanılmıştır.

Şekil 2.5'te görüldüğü gibi her noktanın çevresindeki çizgilerin, dairesel tur biçiminde üçgensel ortalamasına göre tespit edilmesinden sonra vadi kenarlarının büyük çoğunluğu bulunmuş fakat kenarlar arasında küçük aralıklar kalmıştır. Bu açıklıkların kapanması için kenar-takip etme yöntemi [5] geliştirilmiştir. Buna göre, kenarlar 1 piksel boyutunda ve pürüzsüz hale getirilmiş, bitiş noktaları ve yönleri belirlenmiştir. Ayrıca tarama esnasında algoritma yeni bir vadi-kenar pikseline denk gelmesi durumunda, bu yeni pikseli yeni başlangıç noktası olarak kullanmış ve bu işlemi bütün kabarcık etrafında turlayana kadar devam ettirmiştir. Algoritmada hem başta hem sonda olmak üzere uygulanan eşikleme tekniği ile gürültü azaltılmıştır.



Şekil 2.5. Dairesel takip etme tekniği [5]

Sonuç olarak günümüz teknolojisinden alt seviyede bir bilgisayarda 2-3 saniye gibi çok kısa bir sürede sonuca ulaşabilen teknik oldukça başarılı sonuçlar çıkarmıştır. Ayrıca algoritmanın morfolojik işlemler uygulanan bölütlendirme tekniklerine göre daha hızlı çalıştığı gözlemlenmiş ve elle yapılan köpük sayımlarına göre daha isabetli sonuçlar verdiği ortaya konmuştur.

W. Wang önceki çalışmalarına [5] devam ederek 2000 yılında L. Wang ile işbirliği yaparak algoritmalarında bazı düzenlemeler gerçekleştirmiştir [10]. Geliştirilen

yöntem daha önceden ortaya konulan metotlarla birlikte 3 aşamadan oluşturulmaktadır. İlk adımda köpüklerin azami değerlerini bulabilmek için N. Otsu tarafından geliştirilen bir bölgesel eşikleme tekniği [19] kullanılmıştır. Bu tekniği bir sözde kod ile inceleyecek olursak çalışma adımlarını şu şekilde sıralayabiliriz:

ÖP = Ön Plan, AP = Arka Plan, ÖPBölgesi =Ön Plan Bölgesi, APBölgesi =Arka Plan Bölgesi, T = Eşik şeklinde ifade edilirse,

ÖPBölgesi ve APBölgesinin ilk değerlerini belirle.T değişmeden,

Bütün bölgeler için ortalama yoğunluk değerini bul.

T= (ÖP ortalama+AP ortalama)/2.

Bölgeleri tekrar hesapla:

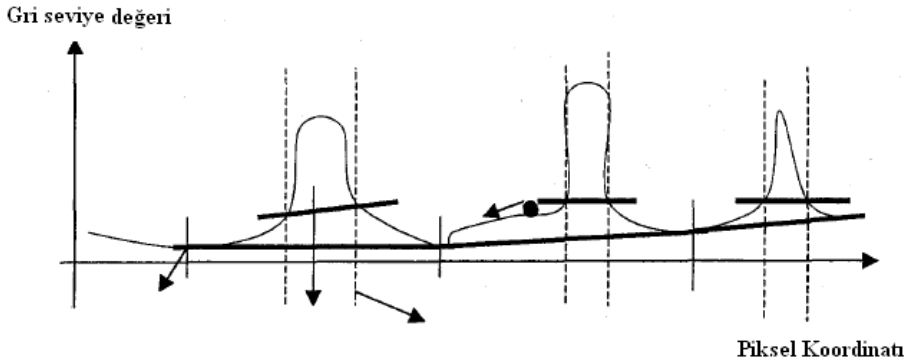
ÖPBölgesi: Yoğunluğu >= T olan piksel.

APBölgesi:Yoğunluğu < T olan piksel.

Çalışmada Otsu'nun eşikleme metodu [19] ile bulunan parlak bölgelerin seçiminden sonra kabarcık alanlarını artırmak için S. Beucher ve F. Meyer tarafından önceden geliştirilen [20] su bendi tekniği [10] kullanılmıştır. Kullanılan bu tekniğe göre incelenen gri seviyedeki görüntü hayali bir topolojik harita olarak ele alınmıştır. Minimum noktalardan delik açılarak su baskını yaratıldığında değişik kaynaklardan gelen suları ayırmak için kullanılan setlerin, bölgeler etrafında turlaması sonucu ortaya çıkan ayırıcı kenarlar incelenir. Su bendi tekniği, taşma işleminin benzetimini yapanlar ve doğrudan su bendi noktalarının bulunmasını hedefleyenler şeklinde ikiye ayrılmaktadır. Bu türler de çözüme yaklaşımı açısından paralel, ardışık ve düzenli algoritmalar olarak alt başlıklara ayrılmaktadır.

Buna ek olarak Şekil 2.6'da görüldüğü gibi kabarcık sınırları bir sonraki yükselmelere kadar genişletilmiş fakat genişleme esnasında fazladan parlak noktaların bulunması sebebiyle fazladan bölütleme oluşmuştur. Tipik bir sorun olan fazla bölütlemeyi engellemek için kabarcık şekillerinin analizine bağlı bir nesne birleştirme ve ardından bölütleme algoritmaları [21, 22, 23] kullanılmıştır. Su bendi metodunun[20] geliştirilen diğer morfolojik metotlarla [23] kombinasyonu ile köpük görüntüleri üzerinde verimli sonuçlar elde edilmiştir. Diğer yandan üretilen algoritma

sadece laboratuar ortamında test edilmiş ve gerçek uygulama alanında test edilme imkanı olmamıştır.



Şekil 2.6. Genleşme algoritması çalışma prensibi [10]

W. Wang ve ekibinin yaptığı çalışmalardan [5,10] 1997 yılındaki J. M. Hargrave ve S.T. Hall tarafından yapılan diğer bir çalışmaya [24] dönüldüğünde görülen, bu çalışmada diğer geliştirilen yöntemlerden farklı olarak kabarcıkların renk özelliği de dikkate alınmış, ayrıca eldeki verileri kullanarak YSA metodu uygulanmıştır. Çalışmada renk değerlerinin kullanılacak endüstriyel alandaki köpük yapılarına göre dikkate alınacağının önemli olduğu vurgulanmıştır.

Sistemde ilk olarak eldeki görüntüye ait kırmızı bileşen (2.1)'de görüldüğü gibi çıkarılmıştır. Burada bahsi geçen G_k, $\bar{X}_g$ ve $\bar{X}_k$ değerleri sırasıyla göreceli kırmızılık, gri seviye ortalaması ve kırmızı seviye ortalamasını ifade etmektedir. Ardından köpükler üzerindeki gölgeli ve parlak noktaların etkisini azaltmak için görüntünün kırmızı halinin ortalaması alınmıştır. Devamında görüntü yeniden gri seviyeye çevrilerek ortalaması alınmıştır. Görüntünün gri seviyeli ve kırmızı halinin ortalamaları farkından hesaplanan göreli kırmızılık oranına göre gri seviyeli görüntü üzerinden normalizasyon işlemi ile ışıklı bölgelerin sağladığı etkiler azaltılmıştır.

$$G_k = \frac{\left(\bar{X}_k - \bar{X}_g\right)}{\bar{X}_g} \tag{2.1}$$

Çalışmada [24] köpüklerin boyut özelliklerini çıkarmak için keskinleştirme metodunun ardından su bendi yöntemi kullanılmıştır. Elde edilen sonuçlara göre köpüklerin büyüklüklerine, uzunluklarının genişliklerine oranına, hangi eksene göre yönelimli olduklarına göre dağılım görüntüleri ortaya çıkarılmıştır. Bu dağılımların sonuçları büyüklük seviyesinin ortalaması, standart sapması ve belli bir yüzde oranı geçebilen köpükler gibi yaklaşımlara göre araştırılmıştır. Bunun yanında her köpüğün ana ekseninin oryantasyonunun yönüne göre üç farklı türe göre ayrımı yapılmıştır. Diğer yandan köpükler, en başta elde edilen renk değerlerine göre gri seviye, kırmızılık ve göreli kırmızılık şeklinde üçe ayrılmıştır. Bu ayrıma göre köpük derecelendirmeleri arasında bazı bağıntılar ve dağılımlar gözlemlenmiştir.

Toplanan çok boyutlu verilerin sınıflandırılabilmesi için YSA modellerinin iyi sonuç vereceği bilinmesine rağmen regresyon analiz teknikleri de model olarak denenmiştir. Sonuç olarak incelenen köpük görüntülerine ait renk ve doku öğelerinin, köpük flotasyonu derecelendirilmesi açısından etkili oldukları gösterilmiştir. Renk değerinin daha çok dereceyi, büyüklüklerin ise köpük hareketlerini ölçmede etkili olduğu saptanmıştır. Bunun dışında çalışmada YSA tekniğinin, köpüklerin görsel özellikleri ile ulaşılabilen verilerle sonuca ulaştırabilecek bir metot olduğu [24] ortaya konulmuştur.

1999 yılında J.J.Cillliers ekibi ile beraber önceki çalışmasını [3] geliştirerek üç aşamalı bir sistem [6] geliştirmiştir. Öncelikle su bendi metodu üzerine kurulu ve önceden yapılan bir çalışmaya [3] dayalı bir görüntü işleme tekniği kullanılmıştır. Ardından akışkan akıcılığını analiz edebilmek için görsel canlandırma ve köpük özelliklerini ortaya çıkarabilmek için EDT üzerine deneyler yapılmıştır. Sonuçta görüntü işlemenin yetersiz kaldığı yerlerde EDT' nin başarılı sonuçlar verdiğine değinilmiş ve ikili bir birleşik kontrol sisteminin iyi sonuçlar vereceği vurgulanmıştır. Fakat son iki adımdaki çalışmalar, araştırma kapsam menzili dışına çıktığı için ciddi bir fikir vermemekle beraber çalışmanın ileri aşamalarına neler eklenebileceği konusunda ışık tutabilmiştir.

Bonifazi ve ekibinin 1999 yılında yaptığı araştırmada [7] konu biraz daha genel olarak ele alınmış ve köpükler öncelikle sınıflara ayrılmaya çalışılmıştır. Sınıflandırmaların oluşturulması için aynı yöntemle çekilen 1500 resim örneği alınmıştır. Ardından görüntü işleme konusunda herhangi bilgisi olmayan fakat belirli bir kültürel ve bilimsel bilgiye sahip üç kişilik bir ekip kurulmuş ve bu ekibin bütün resimleri incelenmesi sağlanmıştır. İnceleme ekibi tarafından homojen dağılmış ilk başta 4 sınıf belirlenmiş fakat ekibe görüntü işleme konusunda uzman bazı araştırmacıların da katılımından sonra, sonuç olarak 5 sınıfta karar kılınmıştır Bu sınıflar aşağıdaki şekildedir:

- elips şekle sahip yeterince küçük köpükler
- düzgün şekle sahip orta boyda köpükler
- düzgün geniş köpükler
- hem geniş hem küçük boyutlarda yuvarlak yapıdaki köpükler
- çamur görünüme sahip çok küçük köpükler

Morfolojik ve morfometrik özelliklerin incelenmesi için su bendi tekniği ele alınmıştır. Ardından köpüklerin renksel özelliklerinin incelenebilmesi için iki yaklaşım denenmiştir. İlk yaklaşıma göre değişik renk uzaylarına ait renk özellikleri analizi bütün görüntü kümesi üzerinden yapılmış, diğer yaklaşıma göre değişik renk sistemlerine göre tek köpüğün renk özellikleri analizi yapılmıştır.

Köpüklerin ayrı ayrı sayısal olarak ele alınabilmesi için görüntü bölütlendirmesinin önemi vurgulanmıştır. Açıklanan ifadeye göre bölütlendirme, ele alınan uzaysal alanı karşılıklı dışlanabilen homojen karakteristik özelliğe sahip alt kümeler olarak açıklanmıştır. Bölütlendirme için üç ayrı yaklaşımdan bahsedilmiştir:

- Alan tabanlı yaklaşım: Pikselleri ayrılan alanlara veya nesnelere atama (eşikleme ve su bendi metotlarında yaygındır.)
- Sınır tabanlı yaklaşım: Bölgesel sınırların yerinin bulunması (sınır-takip ve laplace filtrelerinde)

- Kenar tabanlı yaklaşım: Kenarlardaki piksellerin tanımlanması ve bir sınır oluşturacak şekilde düzene sokulması.

Araştırmada su bendi tekniği uygulandığı için alan tabanlı yaklaşımın kullanıldığı ifade edilmiştir. Bölütlendirme için ele alınan su bendi tekniği için özyineli algoritmaya sahip bir çalışmadan yararlanılmıştır [16]. Bu algoritmada ana yapı, görüntünün her döngüde uygun bir eşik değerinin ayarlanması temeline dayalıdır. Her dönüşte bölgesel minimumlara ait havzalar hesaplanan eşik değerlerine göre ayarlanan bulundukları baskın bölgeye göre genişletilmektedir. Fakat su bendi tekniğinin en önemli handikapı olan fazla bölütlendirme sorunu için önceden kenar bulma ve düzleştirme filtrelerinin uygulanması gerektiği belirtilmiştir. Öncelikle karşılaştırma sağlayabilmek için bütün resim 256X256 boyutlarda alt parçalara ayrılmış ve siyah sınırları kaldırmak için yoğunluk kanalı çıkarılmıştır. Ardından uygulanan keskinleştirme filtresi ile görüntü iyileştirmesi sağlanabilmiştir. Gri seviyeden bütün resimlere yapılan histogram üzerinden en yüksek frekanslar saptanmıştır. Tespit edilen bu değer yüzde oranı olarak ele alınmış ve su bendi filtresi için eşik değer olarak kabul edilmiştir. Sonrasında yapılan su bendi tekniği ile köpük bölütlendirilmesi yapılmıştır [25]. Her bölütlendirme yapılan köpük için geometrik ve morfolojik değerler (örn: alan, görünüş, çap, yarıçap, çevre, yuvarlaklık, uzunluk, genişlik…vb) ortaya konulmuştur.

Bölütlendirme ve özelliklerin çıkarımı sonrasında renk değerleri tespiti için bütün resimde veya bölütlendirilen resimlerde ortalamalar ve standart sapmalar hesaplanmıştır. Bu hesaplamalar KYM, RDD ve YRD üzerinden yapılmış ve önemli bir fark gözlemlenmemiştir.

Son olarak eldeki verilere göre yapılan tahminlerde her zaman olmamakla birlikte bir miktar hata oranı gözlemlenmiştir. Örneğin 30-60 veya 300-340 denek görüntüleri arasında tahminlerle gerçek oranların farkında artış görülmektedir. Bu hata oranını azaltmak için çalışmada [7] önceden belirlenen beş farklı köpük görüntüsü sınıfına göre görüntüler sınıflandırılmış olup her sınıf için korelasyon gözden geçirilmiş ve

doğruluk oranı evrensel değerin altında kaldığı zaman tekrar işleme alınmış ve doğruluk oranı artmıştır.

Sonuçta yapılan çalışmada [7] görüntülerin önceden farklı kümelere ayrılması ve buna göre değerlendirilmeye alınmasının tahminlerde isabet oranını arttırdığı kanıtlanmıştır. Ayrıca köpük bölütlendirme algoritmasının doğru sonuçlara ulaşmak için ne kadar önemli olduğu ortaya konmuştur.

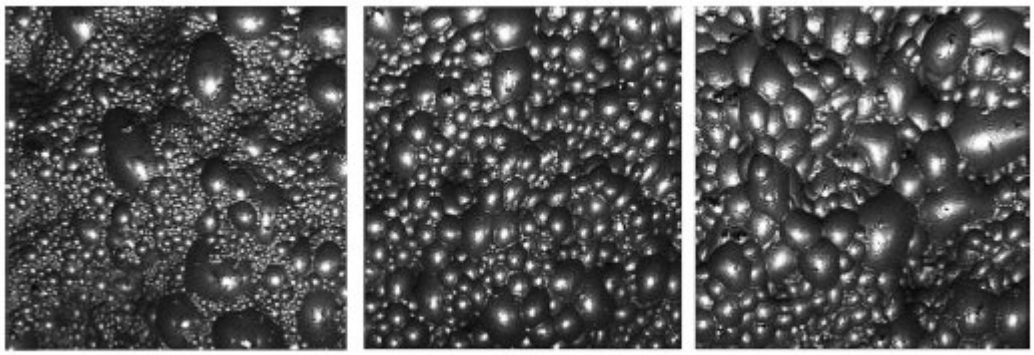
Yine Bonifazi ve ekibinin 1999 yılında yapmış oldukları çalışmanın [7] devamı niteliğinde 2000 yılında yapmış oldukları araştırmada [9] öncekinde olduğu gibi görüntülerin KYM, RDD ve YRD çevrimleri üzerinden ortalamalar ve standart sapmaları alınmış ve bunlar işlem esnasında istatistiksel kontroller için değerlendirilmiştir. Bir önceki çalışmada olduğu gibi bölütlendirme için alan tabanlı modelleme seçilmiş ve su bendi tekniği kullanılmıştır [25]. Ardından değişken ışık etkilerini azaltabilmek için bölütlendirme sonrası inceltme işlemi uygulanmıştır. Elde edilen görüntülerde iyileştirme yapabilmek için keskinleştirme filtresi kullanılmıştır. Sonuçları yanıltmaması için sınırlardaki kabarcıklar ele alınmadan sayım yapılmış ve ortalama, standart sapma gibi değerler hesaplanmıştır. Karşılaştırma yapabilmek için kabarcıkların görsel değerleri üzerinden majör ve minör eksenlerinin oranları çıkarılmıştır. Dolayısıyla bir kabarcığın yuvarlaklığı elde edilen bu oranın 1 değerine yakınlaşması ile gösterilmiştir. Çalışmada [9] kabarcıkların insan gözüyle çıkarılan görünümleri ve yapılan hesaplamalar arasında iyi bir korelasyon yakalanmıştır. İstatistiksel analiz için regresyon metodu kullanılmış ve bulunan sonuçlar karşılaştırılmıştır. Sonuç olarak bu çalışma [9] sisteme etki eden parametreler kadar görüntünün kalitesinin önemini de vurgulamıştır.

Son dönemlerde 2002 yılında, R. Maurus , V. Ilchenko ve T. Sattelmayer tarafından geliştirilen bir başka sistemde [26], yüksek hızlı kameralarla elde edilmiş görüntüler üzerinden kabarcık değerleri (sayıları, boyutları, bölgesel ortalama içerikleri) elde edilmiştir. Sistem kapsamlı olsa da, yaygın bir arka plan üzerine dağınık ve birbirinden ayrı kabarcıklardan oluşan farklı karakteristik özelliklere sahip görüntülerden oluştuğu için ele alınan çalışma ile ana hatlarıyla farklılık

göstermektedir. Fakat bu çalışmanın, dikkat çekici bir özelliği gerçek çalışma alanında resimlerin çekilmesi, fotoğrafların görüntüleme ayarları açısından nasıl bir test ortamı yaratılması konusunda bizlere fikir vermiş olmasıdır. Bunun yanı sıra kabarcık karakterizasyonunun önemli olduğu buharlaşma gibi bir başka alanı da göstermiştir.

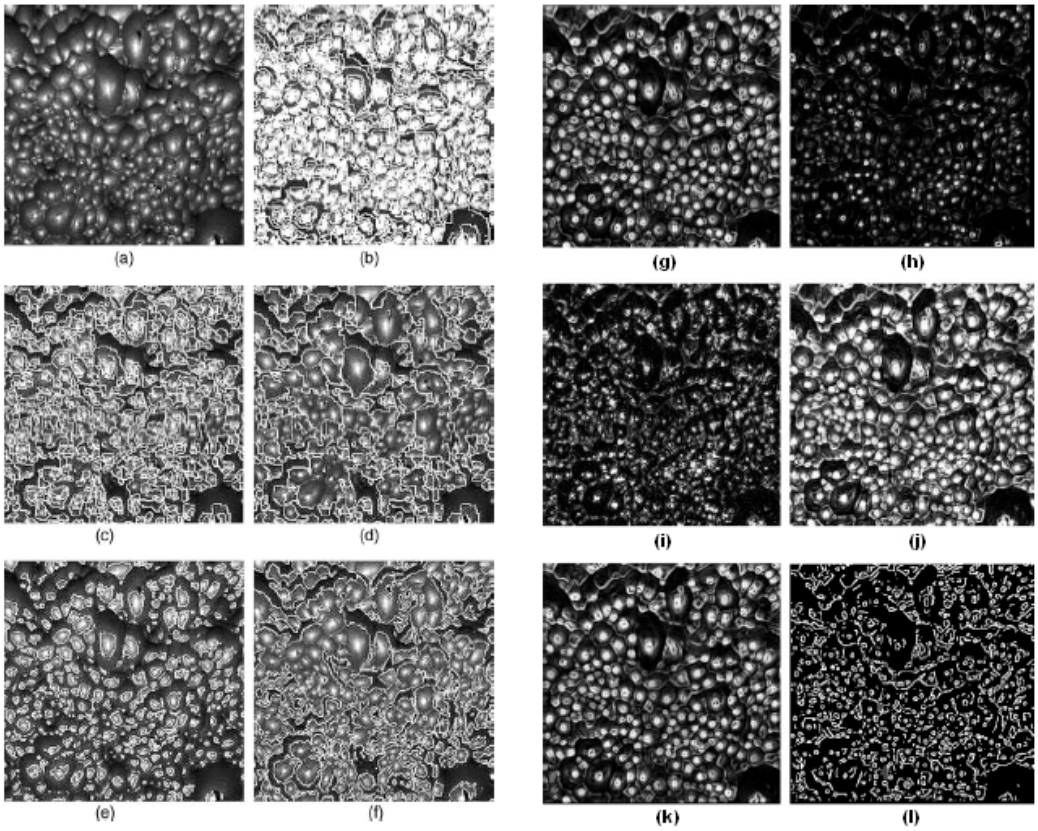
Bu konuda yapılan en geniş projelerden biri olan W. Wang önderliğinde, 2003 yılı önceki çalışmalarının [5] devamı olan ve Avrupa Birliği desteği ile yürütülen projesinde [11] sistem, öncelikle görüntü sınıflandırma ve bundan yararlanılarak geliştirilen vadi-kenar tabanlı bir bölütlendirme algoritması üzerine kuruludur. İlk olarak sınıflandırma algoritmasında görüntünün kalitesi incelenmiş ve bu sonuca göre bir sınıflandırılma yapılmıştır. Yapılan genel izlenim sonucu görüntülerin küçük boyutlu kabarcıklar, orta boyutlu kabarcıklar, büyük boyutlu kabarcıklar ve karışık boyutlara sahip kabarcıklı görüntüler olmak üzere dört farklı grupta incelenebileceği ortaya konulmuştur. Buradaki sınıflandırmanın belirlenmesinde en önemli nokta, her kabarcığın parlak noktaya sahip olması ve kabarcık boyutunun, parlak bölgelerin boyutuyla doğru bir orantıya sahip olmasıdır.

Bu yüzden Wang tarafından daha önce de yararlanılan [10] Otsu'nun dinamik ve otomatik eşikleme algoritması [19] kullanılmıştır. Çıkan sonuçlar sayısal olarak ele alınırsa beyaz bölgelere ait piksel sayısı Şekil 2.7.a için en az , Şekil 2.7.b için ortalama, Şekil 2.7.c için en fazla olacak şekilde parlak bölgelerin ortalama büyüklükleri küçükten büyüğe artmaktadır. Bunların dışında parlak bölgeler sayımı yapıldığında ters orantılı olarak azalma gözlemlenmiştir.



Şekil 2.7. Farklı köpük sınıflarına ait görüntüler (a)küçük, (b)orta, (c)büyük [11]

Dolayısıyla bu analiz sonucunda parlak noktaların kabarcık boyutuyla orantılı olduğu kanıtlanmıştır. Diğer yandan parlak noktaların sayısının kabarcık boyutuyla ters orantılı olduğu ve kabarcık büyüklüğü dağılımının kabaca parlak bölgelerin büyüklük dağılımıyla tahmin edilebileceği ortaya konulmuştur. Çalışmada [11] sınıflandırma aşamasından sonra bölütlendirme aşamasına geçilmiş ve bu konuda benzer çalışmalara ışık tutacak bazı testler yapılmıştır. Buna göre literatürde bilinen klasik kenar bulma yöntemleri denenmiş ve sonuçları gösterilmiştir. Sırayla incelenecek olursa Şekil 2.8.a'da orta boyutlarda sınıflandırılabilecek kabarcıklara sahip bir görüntünün ilk hali görülmektedir. Bölgesel gri seviye farklarına bakılarak "6'dan düşük olduğu durumlarda aynı benzerliğe sahiptir mantığı" ile çalışan metot, Şekil 2.8.b'de görüldüğü gibi fazla bölütlenme ile karşılaşmıştır. Bu fazla bölütlenmenin üstesinden gelebilmek için 15 pikselden daha küçük kabarcıklar kendisine komşu en büyük kabarcıkla birleşsin düşüncesi denenmiş, bölütlenme bir miktar azalma görülse de Şekil 2.8.c'de görüldüğü gibi düzgün sonuç vermemiştir. Sırasıyla Şekil 2.8.d, Şekil 2.8.e, Şekil 2.8.f şıklarında farklı gri seviye eşikleme metotları denenmiş fakat sonuçlar yeterince başarılı olamamıştır. Diğer yandan bazı kenar bulma yöntemleri ile elde edilen sonuçlar karşılaştırılmıştır. Sırasıyla Şekil 2.8.g, Şekil 2.8.h, Şekil 2.8.i, Şekil 2.8.j, Şekil 2.8.k ve Şekil 2.8.l'de görüldüğü üzere, Sobel [27], Robert [28], Laplace [29], Prewitt [30], Canny [31] ve eşiklenmiş Canny kenar bulma metotları denenmiştir. Laplace en kötü sonucu vermiş, diğer yöntemler ise bazı kabarcık parçalarını yakalayabilmiştir.



Şekil 2.8. Gri değer benzerliği ve kenar bulma yöntemleri ile bölütlendirme [11]

Bu metotların parlak noktalara ne kadar bağımlı olduğunu göstermek için 30 değeri üzerinden görüntü eşiklenmiş ve Şekil 2.8.l'de sonuçları görüleceği üzere Canny kenar bulma yöntemi uygulanmıştır. Parlak noktaların azalmasıyla bilinen kenar bulma yöntemleri başarısızlık göstermiştir.

Testler sonucunda farklı bir yöntemin kullanılması gerekliliği ortaya konulmuştur. Bunun dışında klasik yöntemlerle [27-31] istenilen sonuca ulaşılamadığı sonucu ortaya çıkmıştır, çünkü inceleme altına alınan görüntüler burada da bahsedilen parlak bölgelerin azlığı veya dağınıklığı gibi olumsuz etkenler barındırmaktadır. Bu yüzden Wang ve ekibi tarafından önceden de yararlandıkları [5] vadi-kenar bulma algoritması kullanılmıştır.

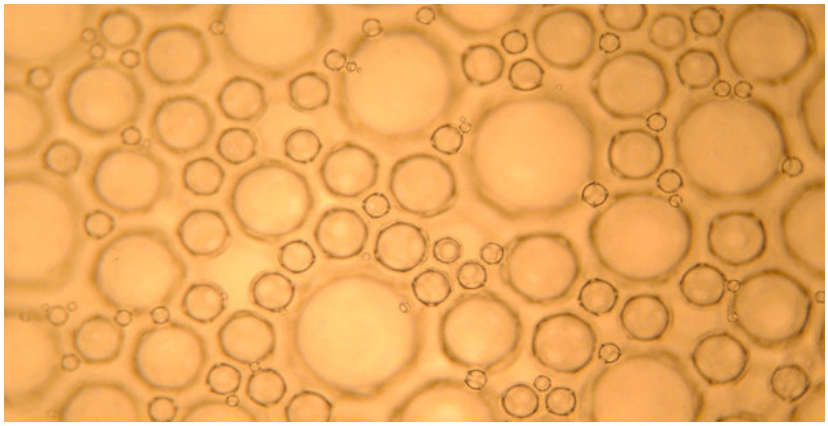
Sonuç olarak gerçek uygulama alanlarında yapılan testler neticesinde projede [11] başarılı sonuçlar elde edilmiştir. Bunun yanında çalışma süresi olarak oldukça hızlı bir teknik olduğu ifade edilmiştir.

Bütün bu çalışmalardan [1-31] anlaşılacağı üzere görüntüler üzerinden köpük özelliklerinin çıkarımı karmaşık bir algoritma gerektirmektedir. Bu konuda yapılan çalışmalarda vadi-kenar ayrımı, su bendi gibi belli başlı bazı metotların kullanımı dışında ciddi bir metot sınıflandırılması gerçekleşmemiştir. Bunun nedeni ise işlemeye alınan köpük görüntülerin farklı biçimlerde ve özelliklerde olması sebebiyle yapılan çalışmalar arasında paralellik sağlanamamasıdır. Geliştirilen tekniklerde her ne kadar ele alınan görüntüler bu çalışmada kullanılanlardan çok farklı olsa da araştırmaya ışık tutabilecek bazı fikirler vermiştir. Bunlar aşağıdaki şekilde sıralanabilir:

- Flotasyon köpüklerinin büyüklük, şekil gibi özellikleri performansla bir korelasyona sahiptir [2,3].
- Klasik görüntü işleme ve kenar bulma teknikleri tek başlarına yetersizdir ancak yapılan işlemlere yardımcı olabilirler [11].
- Flotasyon madenlerinin görsel olarak algılanmasında genel geçerli bir prosedürün ortaya konulamadığı anlaşılmıştır [9].
- Köpük resimleri parçalara ayrılarak karşılaştırma sağlanabilir ve daha kolay işlenebilir [7].
- Görüntülerin çekimi sırasında kullanılan ışıklandırma düzenekleri kabarcık özelliklerinin çıkarımlarını doğrudan etkilemektedir. Çünkü her bir kabarcıkta bir parlak bölge vardır ve bu parlak bölge kabarcığın boyutu ile orantılıdır [11].
- Köpükler genel olarak yuvarlak veya elips şekle sahiptir [2,3].
- Köpüklerin koyu renklerde ve karşılıklı dışlanabilen yapılara sahip sınırları vardır [7].
- Görüntü işleme sonrasında elde edilen verilerin incelenmesinde YSA yapısından yararlanılabilir [7,24].

2.2. Kullanılan Köpük Görüntüleri

Araştırmada iki farklı tipte resim incelenmiştir. T. Eren ve M. E. Özbayoğlu tarafından mikroskop altında elde edilen [1] kabarcık görüntüleri, daha yuvarlak ve belirgin kabarcık hatlara ve düzgün ışık dağılımına sahiptir. Bir örnek görüntü Şekil 2.9'da görülmektedir.

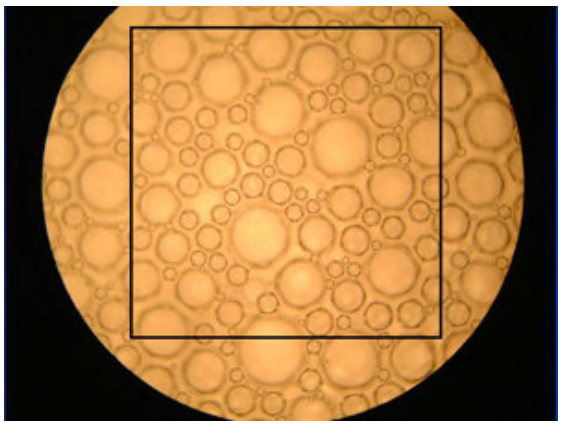


Şekil 2.9. Birinci tip kabarcık görüntüleri

Diğer yandan Strauss. H. tarafından, Freiberg Üniversitesi laboratuarlarına ait köpüklü sondaj cihazından alınan görüntüler ise Şekil 2.10'da bir örneği görüldüğü gibi daha belirsiz ve daha az yuvarlak hatlı kabarcıklara sahiptir.

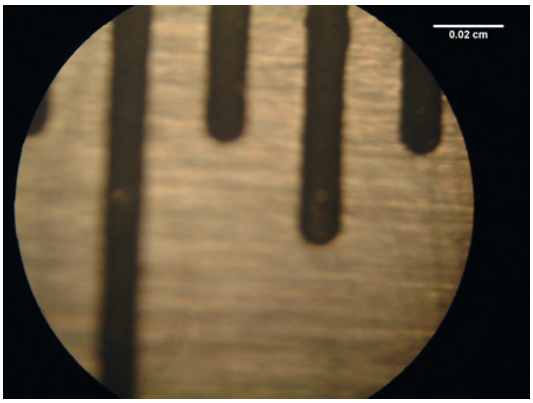
Şekil 2.10. İkinci tip kabarcık görüntüleri

Köpük resimlerinden büyüklük ve şekil özelliklerinin çıkarılması için bilgisayarda seçilen resmi büyük almak, herhangi bir avantaj getirmemesine rağmen hafızada yer yetersizliğine ve yavaşlamaya yol açmaktadır. Bu yüzden 2400x1800 piksel boyutlarındaki görüntülerden Şekil 2.11'de gösterildiği gibi 1350x1350 piksel boyutlarında kesitler alınmıştır. Kesitler daha sonra test bilgisayarlarının RAM yapısına uygun olarak 500x500 piksel boyutlarına küçültülmüştür.



Şekil 2.11. Kesim işlemi

Gerçek boyutlara göre ölçek kalibrasyonu, Şekil 2.12'de görüldüğü üzere 2400-1800 piksel boyutlarına göre hesaplanan çevrimde $320\,piksel = 0,02\,cm$ olarak tespit edilmiştir [1]. Çalışmadaki boyut küçültme işlemlerine göre, piksel boyutu $\frac{10}{27}$ oranında küçülerek $118,5\ piksel \cong 0,02\,cm$ eşitliği esas alınmıştır.



Şekil 2.12. Piksel-metrik çevrim görüntüsü

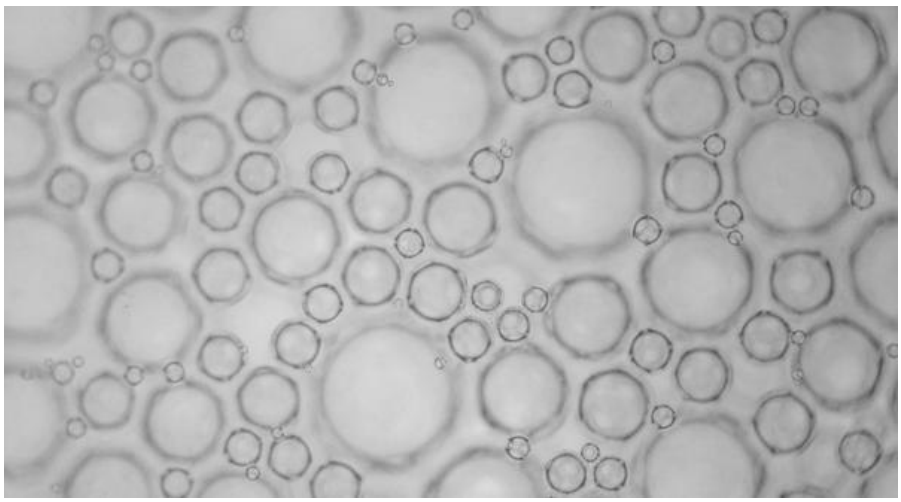
2.3. Geliştirilen Yöntem

2.3.1. Görüntü İşleme Teknikleri

İşleme alınan görüntü, matematiksel değerlerinde işlem yapabilmek için Matlab içerisinde bulunan imread metodu [32] ile I matrisine kopyalanmıştır. I matrisi MxNx3 boyutlarında olup her koordinata göre kırmızı,mavi ve yeşil bileşenlerini taşımaktadır. Fakat çalışmada renk ve parlaklık değerlerinin önemi olmadığı için bu bileşenler sıfırlanmakta ve rgb2gray metodu aracılığıyla resim gri seviyeye dönüştürülmektedir. Dönüşüm için (2.2) denkleminde görülen vektör ağırlıkları kullanılmaktadır [32].

$$(0,2989R) + (0,5870G) + (0,1140B) = I \tag{2.2}$$

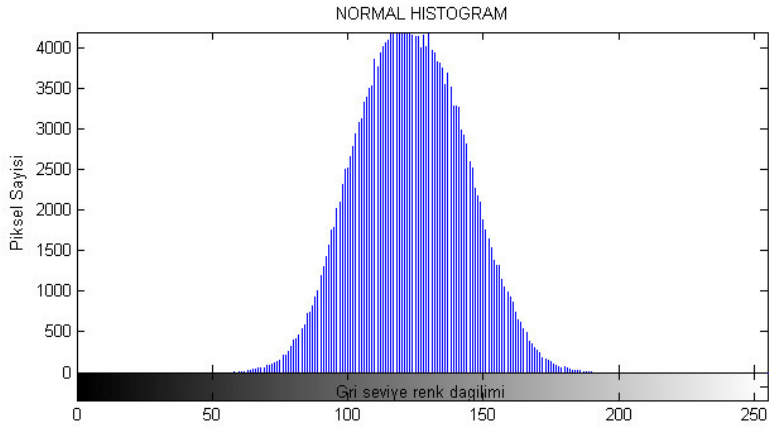
Şekil 2.13'te görülen dönüşüm sonrasında siyah 0 olmak üzere her piksel [0-255] aralığında bir değer almaktadır.



Şekil 2.13. Birinci tip resimlerin gri seviye dönüşümü sonrası hali

Kabarcık sınırlarının tespitini kolaylaştırmak ve değerler arasındaki hassasiyeti artırmak için KLAHE tekniğinden yararlanılarak kontrast farkları artırılmıştır [33]. Matlab yazılımı içinde bulunan adapthisteq ve imadjust metotları [32] ile

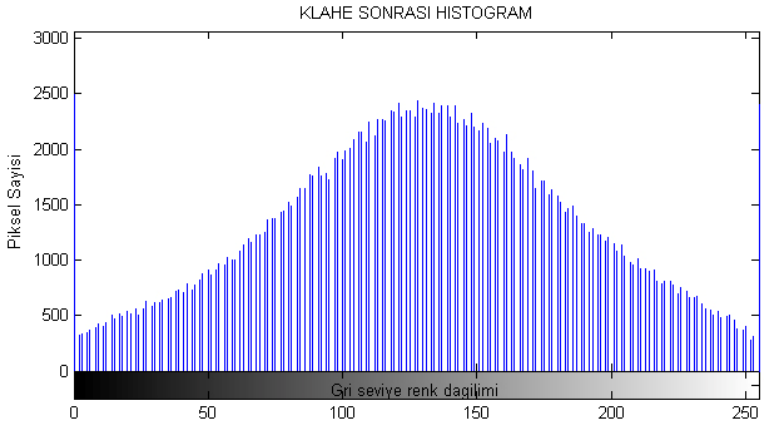
uygulanabilen KLAHE tekniğine göre görüntü küçük parçalara ayrılmaktadır. Şekil 2.14'te normal histogramı görülen görüntünün, standart dağılım histogramına uygun bir şekilde her parçanın kontrastı artırılmıştır.



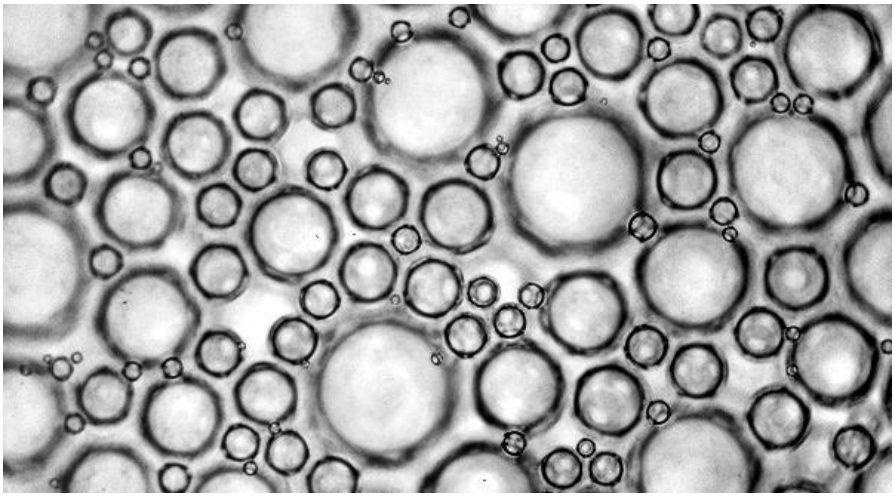
Şekil 2.14. Normal histogram (Örnek olarak 2. tip görüntü alınmıştır)

Elde edilen parçalar arasında sınırlar olmaması için ikili doğrusal interpolasyon ile birleştirme işlemi yapılmıştır. Kontrast artımı limitlenerek gürültülerin artırılmaması sağlanmıştır. (2.3) denklemlinde görüldüğü gibi maksimum ve minimum g değerleri farkı ve P(f) kümülatif olasılık dağılımı yardımıyla g değeri hesaplanmıştır [33]. Sonuçta Şekil 2.15'de histogramı görülen 0-255 aralığına göre normalizasyonu yapılmış görüntüde, Şekil 2.16'da görüldüğü üzere sınırlar belirgin hale gelmiştir.

$$g = \left[g_{\max} - g_{\min} \right] P(f) + g_{\min} \tag{2.3}$$

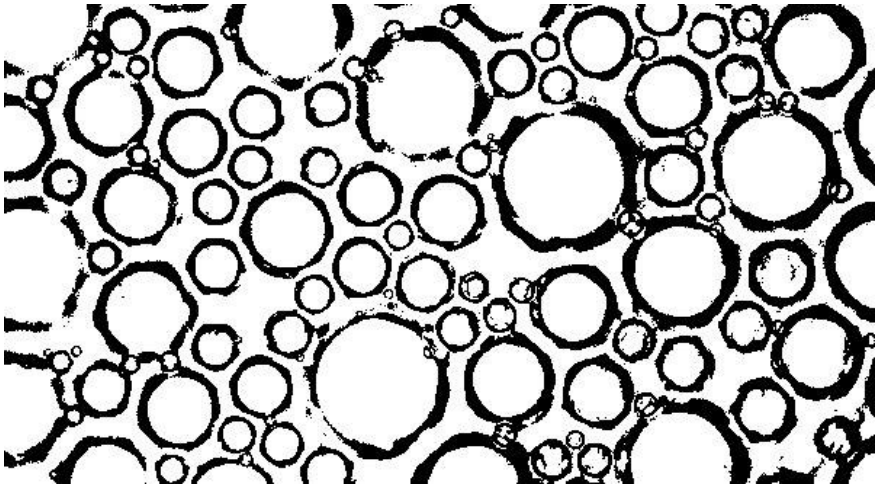


Şekil 2.15. KLAHE sonrası histogram (Örnek olarak 2. tip görüntü alınmıştır)

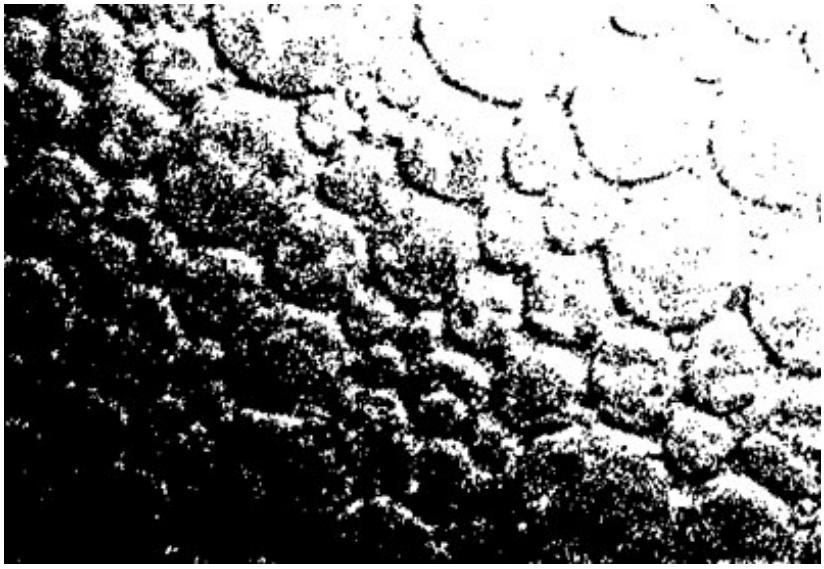


Şekil 2.16. KLAHE metodu sonrası görüntü

Modelin geliştirilmesi sırasında ilk olarak eşikleme teknikleri denenmiştir. Sabit bir değer üzerinden eşikleme tekniği birinci tip resimleri için Şekil 2.17'de görüldüğü üzere iyi sonuç vermesine rağmen ikinci tip ve sınırları belirsiz ve gürültülü yüzeye sahip görüntüler için Şekil 2.18'de görüldüğü üzere iyi sonuç vermemiştir.

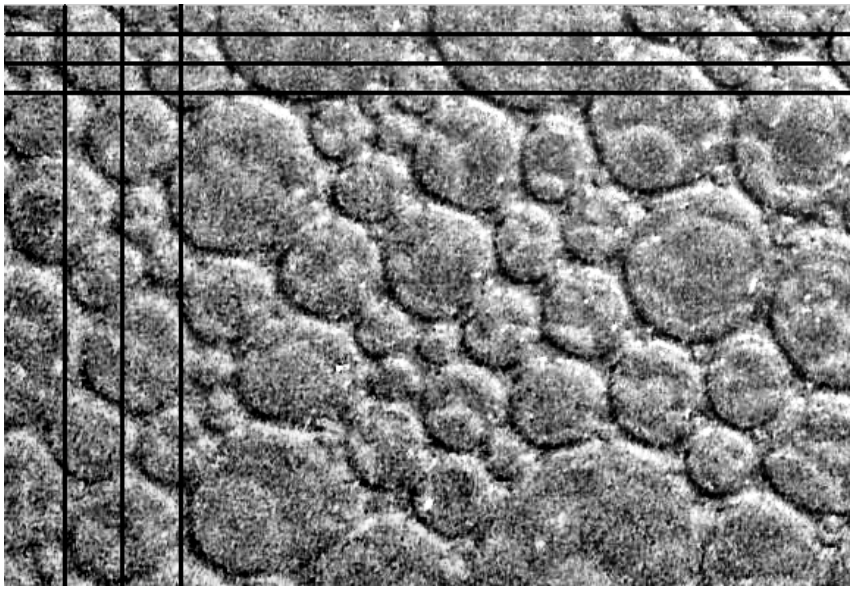


Şekil 2.17. Birinci tip resimler için sabit değer üzerinden eşikleme



Şekil 2.18. İkinci tip resimler için sabit değer üzerinden eşikleme

Genel dağılım üzerinde eşiklemeden istenilen sonuç alınamadığı için sınırları belirginleştirmek için bölgesel eşikleme tekniği denenmiştir. Buna göre görüntü Şekil 2.19'da görüldüğü üzere köpüklerin dikey ve yatay sınır eksenlerini tespit edebilmek için yatay ve dikey doğrultuda dikdörtgen bölgeler olarak ele alınmıştır.



Şekil 2.19. Bölgesel Eşikleme Alan Kesitleri

Bu bölgeler üzerinde (2.4) ve (2.5) denklemlerinde ifade edildiği üzere sırasıyla ortalamalar ve standart sapmaların ortalaması değerleri çıkarılmıştır.

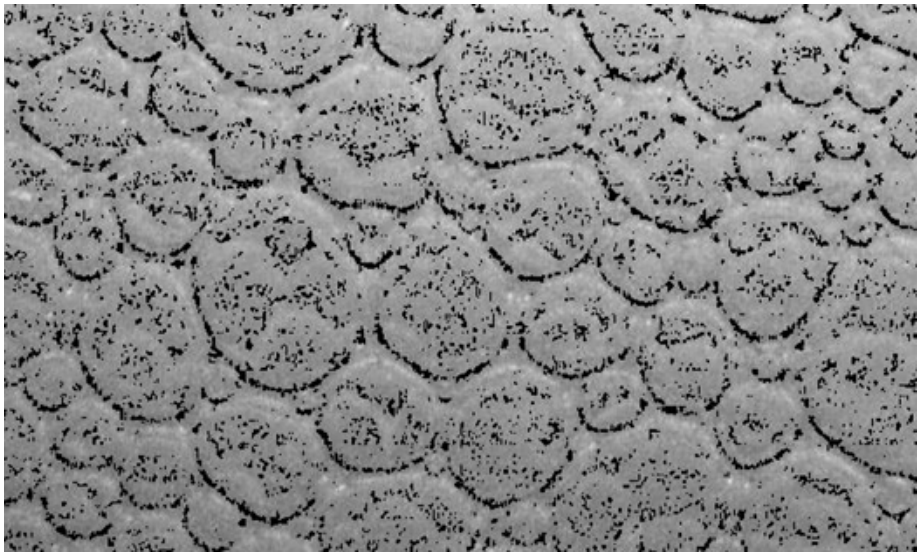
$$m,n\in \mathbf{Z}\,,m=[1,\alpha],n=[1,\beta],\rightarrow C_1(m,n)=\frac{\alpha\beta}{MN}\sum_{i=\frac{(m-1)M}{\alpha}}^{\frac{mM}{\alpha}-1}\sum_{j=\frac{(n-1)N}{\beta}}^{\frac{nN}{\beta}-1}J\left(i,j\right) \quad (2.4)$$

$$S_1(m,n)=\frac{\alpha}{M}\sum_{i=\frac{(m-1)M}{\alpha}}^{\frac{mM}{\alpha}-1}\sqrt{\frac{\beta}{N}\sum_{j=\frac{(n-1)N}{\beta}}^{\frac{nN}{\beta}-1}\left(C(m,n)-J(i,j)\right)^2} \quad (2.5)$$

Denklemlerde kullanılan α ve β değerleri resmin yatay veya dikey uzantıda bölgelere ayrılabilmesi için görüntü boyutlarını ayırdığımız parça boyunu ifade etmektedir. Örneğin, deneylerimizde de kullanıldığı gibi 360X600 boyutları için α ve β değerleri 30 olduğunda 12X20 boyutlarında yatay, $\alpha=15$ ve $\beta=60$ olduğunda 24X10 boyutlarında dikey dikdörtgen bölgeler oluşmaktadır. Her iki seçim için C_1, C_2 ortalamalar matrisi ve S_1, S_2 ortalama standart sapmalar matrisleri elde edilmiştir. Her $J(i,j)$ elemanı için (2.6)'da belirtilen kontrol yapılmış ve uygun durumlarda değer sıfırlanmıştır.

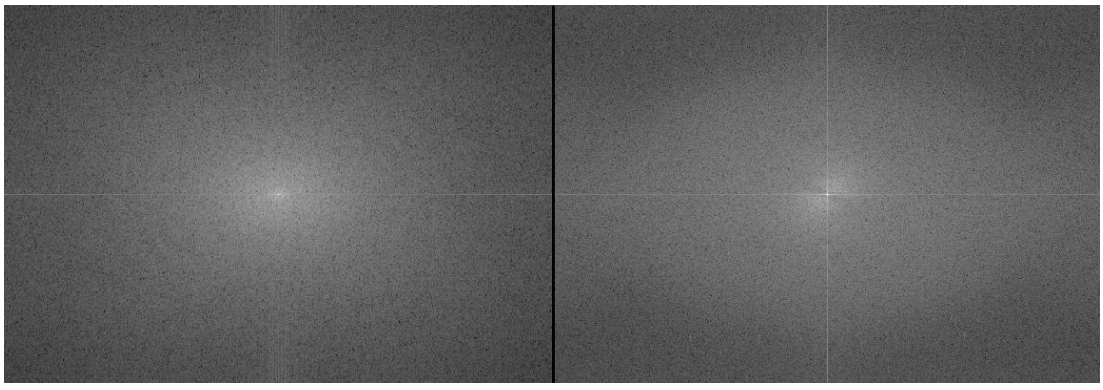
$$\begin{aligned} J(i,j) \;&\leq\; (C_1(k,m)-1,2\,S_1(k,m))\vee \\ J(i,j) \;&\leq\; (C_2(k,m)-1,2\,S_2(k,m))\rightarrow J(i,j)=0 \end{aligned} \quad (2.6)$$

Sonuçta Şekil 2.20'de görüldüğü gibi köpük sınırları biraz daha netleşse de yeterli bir keskinliğe ulaşmamış ve gürültü artmıştır. Dolayısıyla kartezyen koordinatlardan yeterli bir sonuca ulaşılamayacağı kanısına varılmış ve görüntünün sinyal verileri incelenmiştir.



Şekil 2.20. Yatay veya dikey eşikleme sonucu

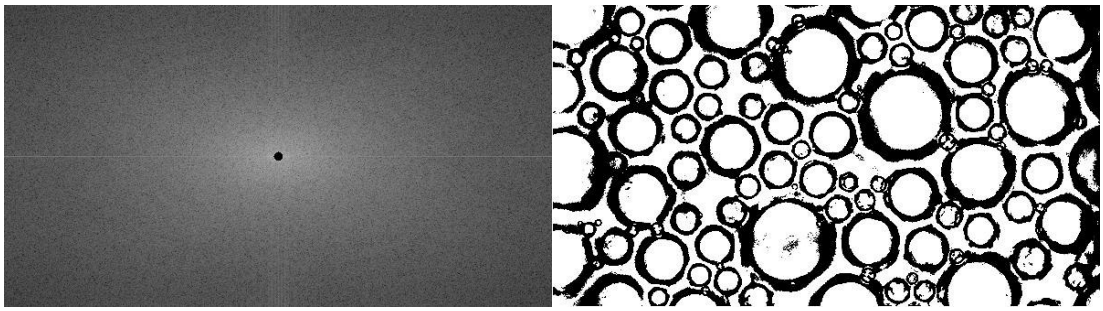
Buna göre Ek A'da görülen model geliştirilmiş ve görüntü üzerinde FD filtreleri uygulanmıştır. Modele göre 2 boyutlu FD için Matlab içerisinde bulunan ve hızlı FD algoritması kullanan fft2 [28,32] metodu ile görüntünün sinyal verisi elde edilmiştir. Bir başka Matlab metodu fftshift [32] ile faz kaydırması yapılmış ve 4 köşedeki bölgeler yer değiştirilerek 0 frekansı spektrumun ortasına alınmıştır. Şekil 2.21.a ve Şekil 2.21.b' de sırasıyla 1. tip ve 2. tip resimlerin spektrumları görülmektedir.



(a) (b)

Şekil 2.21. (a) 1.tip, (b) 2.tip görüntülerde FD spektrumunun mutlak değer görünümü

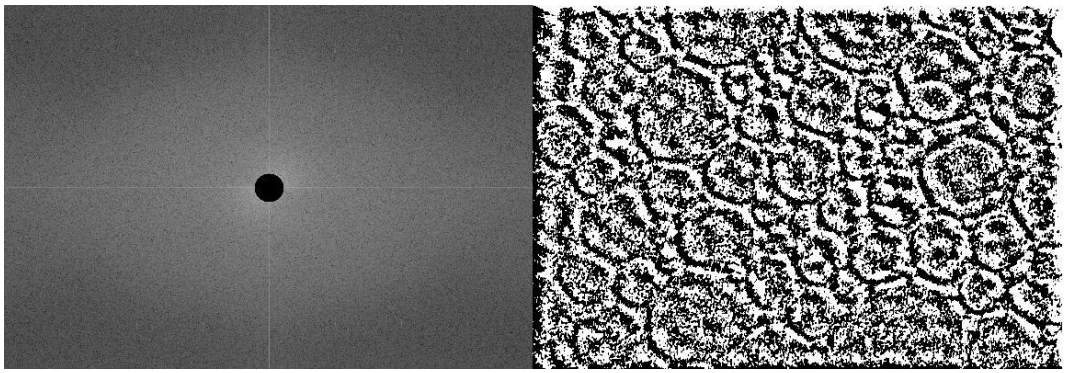
Spektrumlardan anlaşılacağı üzere çok belirgin bir örüntü ortaya çıkmamasına rağmen Şekil 2.22.a ve Şekil 2.23.a'da görülen YGF sırasıyla 1. tip ve 2.tip resimler için uygulanmıştır. Filtre sonrası fftshift ile faz kaydırması yapılmış ve Matlab içerisinde bulunan ve 2 boyutlu TFD sağlayan ifft2 metodu ile [32] ters dönüşüm sağlanmıştır.



(a) (b)

Şekil 2.22 (a) 1. tip resim için uygulanan YGF, (b) YGF sonucu

Sonuçta YGF, Şekil 2.22.b'de görüldüğü gibi 1. tip resimler için sınırlar kabul edilebilir oranda belirginleştirse de 2. tip resimler için Şekil 2.23.b'de görüldüğü gibi iyi bir sonuç vermemiş ve görüntü üzerinde gürültüyü artırmıştır.



(a) (b)

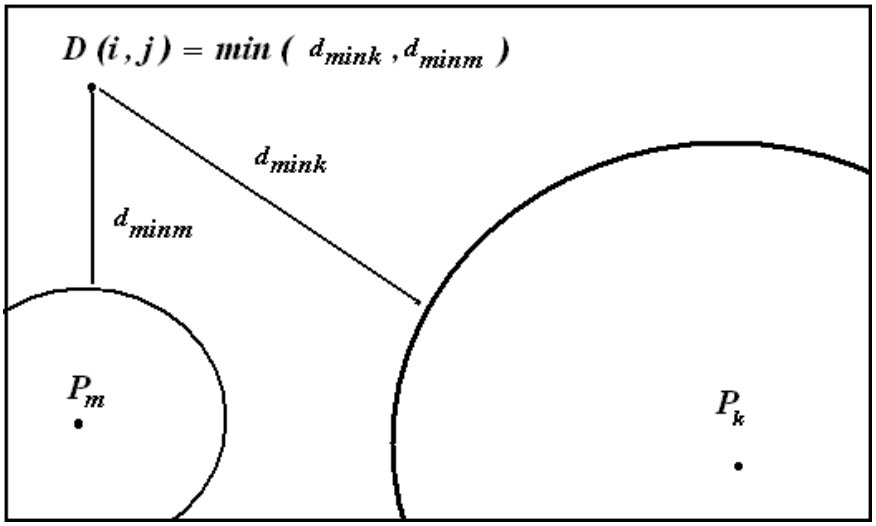
Şekil 2.23 (a) 2. tip resim için uygulanan YGF, (b) YGF sonucu

Anlaşılacağı üzere görüntünün FD sonrası spektrumu dağınık bir görüntüye sahiptir ve buradan uygulanabilecek BGF uygulamaları ile herhangi bir örüntünün yakalanabilmesinin mümkün olmadığı görülmüştür.

Köpüklerin yuvarlak ve eliptik yapıları göz önüne alınarak ve önceden geliştirilen DHD yönteminden [34] esinlenerek deneme yanılmaya dayalı bir görüntü işleme algoritması geliştirilmiştir. Ek B'de ana akış diyagramı görüldüğü üzere algoritma birçok alt işlemden oluşmaktadır. Görüntü KLAHE uygulamasından sonra matris olarak ele alınmaktadır. Ek B'de görüldüğü gibi öncelikle KontrolEyeDrop işleminde, bulunan koordinat ve yarıçap değerine uygun köpüğün sığabilmesi kontrolünü sağlayan uzaklık vektörü güncellemesi yapılmaktadır. Aynı işlem içerisinde, köpük sınırları arası gri seviye düşüş kontrolünün yapıldığı azalmaKoşul değerinin güncellenmesi yapılmaktadır. Güncellenen uzaklık vektörü ve azalmaKoşulu değerlerine göre EyeDrop işlevinde, bulunan koordinatın belirli bir aralığa göre köpük olup olmadığı kontrolü yapılmaktadır Bkz(Ek B-EyeDrop-1. kol). Tespit edilen potansiyel köpükler sırasıyla ağırlık merkezi bulma, açılı minimum takip ve köpükleri ayırma işlevlerinden geçirilerek ön biçimsel düzeltme ve filtreleme işlevlerinden geçirilmektedir Bkz(Ek B-EyeDrop-2. kol). Elde edilen filtreli köpükler sınır takip metoduna gönderilerek bu köpüklerin detaylı biçimsel kontrolü yapılmakta ve sınırlarının çıkarımı sağlanmaktadır Bkz(Ek B-EyeDrop-3. kol). Son olarak görüntüde daha yer olup olmadığı kontrolü yapılmakta, buna göre köpükleri etiketleme ve morfolojik işlemleri yapılmaktadır Bkz(Ek B-EyeDrop-4. kol). En son elde edilen etiketlenmiş matrisine göre çıktılar oluşturulmaktadır.

Keşfedilen köpüklere ait bölgelerde tekrar tarama yapmamak için Öklid uzaklık vektöründen faydalanılmıştır. Şekil 2.24'te görüldüğü üzere her piksel koordinatına göre değeri 0'dan farklı en yakın nokta seçilmiştir. Matlab yazılımı içinde bulunan bwdist [32] metodu yardımıyla seçilen bu noktaya göre Öklid uzaklığı (2.7) denklemindeki gibi hesaplanmış ve D matrisine yazılmıştır. Bu tarama işleminin hızlı olması için önceden geliştirilen algoritmadan [35] yararlanılmıştır. Resimle aynı boyutta ve başlangıçta bütün değerleri 0 olarak ayarlanan D matrisi, bulunan kabarcık alanlarına göre güncellenerek boş alanların tespiti yapılmıştır. Dolayısıyla tarama alanı daraltılarak işlemde hız performansı sağlanmıştır.

$$\mathrm{I}(i_2, j_2) \neq 0 \;\rightarrow\; D(i_1, j_1) = \sqrt{\left|i_1 - i_2\right|^2 + \left|j_1 - j_2\right|^2} \tag{2.7}$$



Şekil 2.24. Uzaklık vektörünün gösterimi

Görüntü iyileştirme ve uzaklık vektörü hesaplanması işlemlerinden sonra iç içe döngüler aracılığıyla potansiyel köpükler taranmıştır. Döngüsel keşfetme yapısı sözde kod olarak aşağıdaki biçimde açıklanabilir.

(r=yarıçap, i = koordinat i, j=koordinat j, c=sayaç, D=uzaklık vektörü, B=3 boyutlu ortalama vektörü)

r ≥ minimum r olduğu sürece,

i ≤ resim boyu olduğu sürece,

j ≤ resim eni olduğu sürece,

r' ye uygun c sayacını hesapla.

Eğer $D(i_1, j_1) \geq r$ r ise,

Resim sınırları dahilinde r+2 ve r-2 arasındaki yuvarlakların ortalama değerini hesapla.

B(i,j,r) değerini B(i,j,r+5) ile karşılaştır.

Eğer orana uygun ise,

B(i,j,r) noktasını potansiyel köpük olarak belirle.

Sözde koddan anlaşılacağı üzere algoritma yarıçap, i, j ve derece döngüleri olmak üzere $\theta(n^4)$ karmaşıklığına sahiptir. Fakat uzaklık vektöründen yararlanılarak, $D(i_1, j_1) \geq r$ koşulu sayesinde algoritma karmaşıklığı gittikçe $\theta(n^3)$'e yaklaşmaktadır. Bu sayede ciddi bir zaman kazanımı sağlanmıştır. Matematiksel olarak ele alındığında, resme uygun olarak (2.8)'de belirtilen belirli bir yarıçap aralığı, $[r_b, r_s]$ seçilmiştir. Bu seçim esnasında görüntü üzerindeki en büyük kabarcık ve dikkate alınacak en küçük köpük değerleri göz önünde tutulmuştur. Diğer yandan (2.9) ve (2.10)'da görüldüğü gibi hız kazanmak için i, j değerleri, görüntü boyutunu aşmayacak şekilde üçer artırılarak tarama işlemi yapılmıştır.

$$r_b \geq r_n \geq r_s \ \wedge \ r_{n-1} - 1 = r_n \tag{2.8}$$

$$1 \leq i_n \leq M \ \wedge \ i_{n-1} + 3 = i_n \tag{2.9}$$

$$1 \leq j_n \leq N \ \wedge \ j_{n-1} + 3 = j_n \tag{2.10}$$

KLAHE metoduna göre hesaplanan $J(i, j)$ koordinatında r yarıçapına uygun bir kabarcığın sığabilmesi için $D(i_1, j_1) \geq r$ koşuluna göre kontrol yapılmaktadır. Eşitsizlik denklemi içinde yer alan $D(i_1, j_1)$ değerinin hesaplanması için (2.7) denklemi kullanılmıştır. $J(i, j)$ koordinatı çevresinde, r yarıçaplı çember tarama için, (2.11) denklemine göre c, derece adımı hesaplanmıştır. Çıkarılan denklem (2.11) için bir çemberi kesintisiz çizebilmek için gerekli minimum derece artışı hakkında denemeler yapılmış ve optimum oran yakalanmıştır.

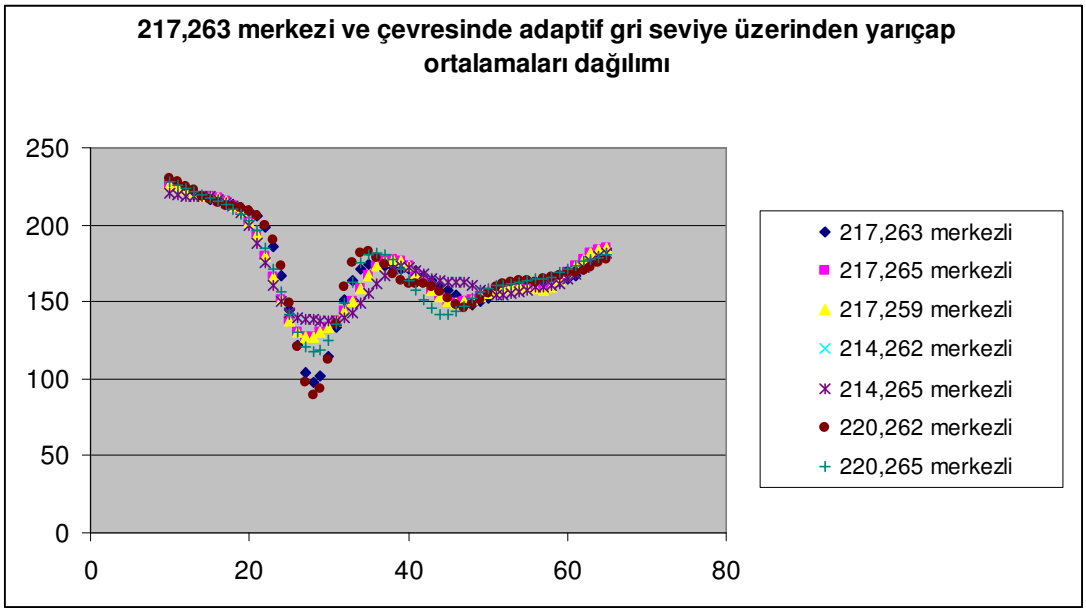
$$c = \frac{360}{10r} \tag{2.11}$$

Aşağıdaki (2.12) denkleminde görüldüğü üzere $J(i, j)$ çevresinde ortalama hesaplanması için polar koordinatlar kullanılmıştır. Ortalama değeri (2.8),(2.9),(2,10) denklemlerine uygun aralıkta hesaplanmakta ve 3 boyutlu B matrisine yazılmaktadır.

$$r_b = r - \frac{\gamma - 1}{2} \wedge r_s = r + \frac{\gamma - 1}{2} \rightarrow$$
$$B(i_n, j_n, r_n) = \frac{1}{\gamma} \sum_{r_b}^{r_s} \sum_{\theta=0^\circ}^{\theta=360^\circ} J\left(\left(i_n - \sin\left(\frac{\theta\,\pi}{180}\right) r_n\right), \left(j_n + \cos\left(\frac{\theta\,\pi}{180}\right) r_n\right)\right) \quad (2.12)$$

Burada ele alınan γ değeri, görüntü üzerindeki kabarcık sınırlarının genişliği ve biçimsel bozukluğu ile orantılıdır. Eldeki görüntüler incelendikten sonra kabarcık sınırlarının ortalama 5 piksel genişliğinde olduğu tespit edilmiş ve γ değeri 5 olarak atanmıştır. İncelenen resimdeki köpük yapılarına göre optimum γ değeri değiştirilebilir.

Köpük sınır bölgelerinden göreceli olarak normal dağılımlı alanlara geçtikçe ortalama değerlerinde değişimler olduğu farkedilmiş ve (2,13), (2,14) denklemlerinde görüldüğü üzere iki tip yaklaşım çıkarılmıştır. Birinci tip yaklaşıma göre Şekil 2.25'te görülen büyük yarıçaplı $B(i_n, j_n, r_{n+5})$ yüzeyinden $B(i_n, j_n, r_n)$ sınır değerine geçtikçe azalma olduğu gözlemlenmiş ve (2.13) denkleminde açıklanan kontrol yapılmıştır.



Şekil 2.25. Değişik merkezlere göre sınırlarda oluşan iniş ve çıkışlar

Diğer yandan kabarcık sınırlarından $B(i_n, j_n, r_{n+5})$ iç alanlardaki daha küçük yarıçaplı $B(i_n, j_n, r_n)$ değerlerine doğru artış olduğu gözlemlenmiş ve (2.14) kontrol denklemi uygulanmıştır. Burada seçilen Φ değeri 1'den büyük olmak üzere, görüntüler üzerinde yapılan kabarcık birim testleri sonucunda tespit edilmiş ve grafikte görüldüğü üzere optimum değer olarak 1.4 oranı seçilmiştir. Her iki koşula (2.13), (2.14) uyan koordinatlar potansiyel köpükleri tutan P kümesine dahil edilmiştir.

$$\Phi > 1 \ \wedge \frac{B(i_n, j_n, r_n)}{B(i_n, j_n, r_{n+5})} \geq \Phi \ \rightarrow B(i_n, j_n, r_n) \in \mathrm{P} \tag{2.13}$$

$$\Phi > 1 \ \wedge \frac{B(i_n, j_n, r_{n+5})}{B(i_n, j_n, r_n)} \geq \Phi \ \rightarrow B(i_n, j_n, r_{n+5}) \in \mathrm{P} \tag{2.14}$$

Kontrolü yapılan orana göre, aslında aynı kabarcığı ifade etmeye çalışan fakat merkezleri ve yarıçapları aynı ya da farklı, birbirine çok yakın olarak iç içe girmiş kabarcıkları tek bir kabarcık olarak ele almak için, ortalama ağırlık merkezlerini bulan bir metot geliştirilmiştir. Şekil 2.25'te bir örnek olarak görünen duruma göre 217,263 merkezine çok yakın uzaklıklardaki noktalarda da aynı (2.13) ve (2.14)'te ifade edilen koşullar oluşmuştur. Bu yüzden bu noktaların arasında bir ağırlık merkezi oluşturulması ve buna göre ortaya bir potansiyel çıkarılması gerekmektedir. Bu metota göre potansiyel bir köpük diğer bütün köpüklerle karşılaştırılmış ve (2.15)'te görülen iki köpük merkezi arasındaki d_{km}, Öklid mesafesi hesaplanmıştır. Mesafeye göre kabarcıkların yakın olarak iç içe olup olmadığı kontrol edilmiş ve (2.16)'da görülen yeni merkez ve yarıçap ortalamaları hesaplanmıştır. Denklemde (2.16) ele alınan ω_m, ağırlık katsayısı bu kabarcığa ait, daha önce iç içe geçtiği belirlenip birleştirilen kabarcıkların toplam sayısını ifade etmektedir. Bu değer ele alınan köpük ile iç içe geçen her köpükte (2.16)'da görüldüğü üzere birer artırılmıştır. Bu sayede hesaplanan yeni ağırlık merkezine, iç içe geçen her kabarcığın etkisi eşit olmuş ve ω_m değerine göre ortalama alınmıştır. Ardından $P(i_k, j_k, r_k)$ köpüğü matristen silinerek filtrelenmiştir.

$$P(i_k, j_k, r_k) \in P,\ P(i_m, j_m, r_m) \in P \rightarrow d_{km} = \sqrt{|i_k - i_m|^2 + |j_k - j_m|^2} \tag{2.15}$$

$$d_{km} < r_m \rightarrow$$

$$i_m = \frac{i_k + (i_m \omega_m)}{\omega_m + 1},\ j_m = \frac{j_k + (j_m \omega_m)}{\omega_m + 1},\ r_m = \frac{r_k + (r_m \omega_m)}{\omega_m + 1},\ \omega_m = \omega_m + 1 \tag{2.16}$$

Diğer yandan P kümesine ait bütün potansiyel köpükler sonucunda her hangi birisi ile iç içe geçmediği tespit edilen köpükler P matrisinde tutulmuş ve ω_k ağırlık katsayısı 1 olarak atanmıştır. Örnek olarak ele alındığında, tarama döngüsü 95 piksel uzunluğunda olan yarıçaplar için Çizelge 2.1'de görülen potansiyel merkezleri bulmuştur. Dikkat edileceği üzere bu merkezler birbirlerine çok yakın mesafede bulunmakta ve (2.15)'te belirtilen koşula uymaktadırlar. Öte yandan Çizelge 2.1'in en alt satırında görüleceği üzere ağırlık merkezi hesaplanmış ve bütün yakın merkezlerin toplamı ağırlık olarak verilmiştir. Bu sayede bütün köpükler eşit ağırlık katsayısına sahip olmaktadır.

Çizelge 2.1. Ağırlık Merkezi Dönüşüm Örneği

Köpük No	i koordinatı	j koordinatı	yarıçap	Ağırlık katsayısı
Köpük 1	487	10	95	1
Köpük 2	487	13	95	1
Köpük 3	487	16	95	1
Köpük 4	490	19	95	1
Köpük 5	493	10	100	1
Ort. Köpük	488.8	13.6	96	5

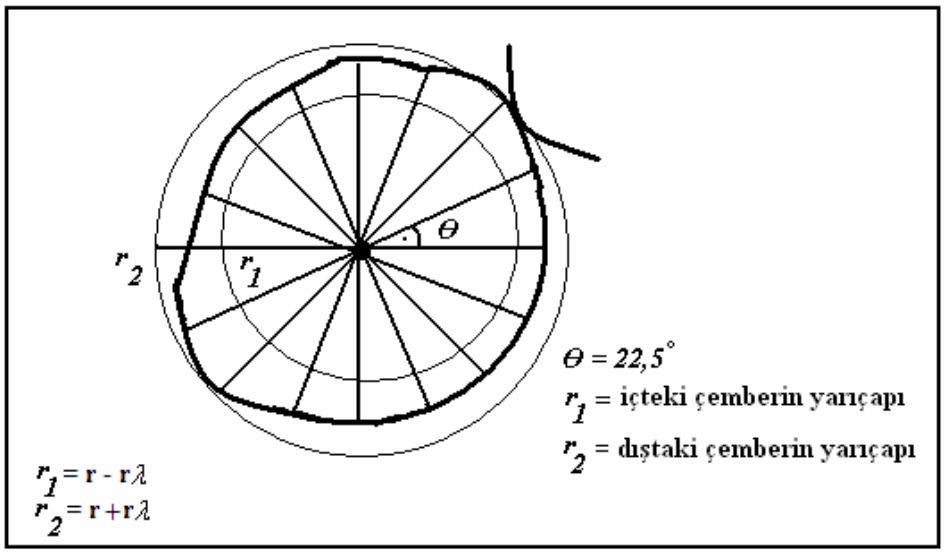
Kabarcıkların biçimsel bozuklukları ve eliptik şekilleri göz önüne alınarak her zaman bir yarıçap değerine sabitlenmediği gözlemlenmiştir. Bu yüzden kontur taramada daha yaklaşık bir yarıçap hesaplanması için (2.17)'de görüldüğü üzere 22.5 derecelik

açılarla, r_n yarıçapına uygun bir aralıkta minimum değerler tespit edilmiştir. (2.18)'de görüldüğü gibi λ tarama hassasiyeti ile r_n minimum için uygun aralığın seçilmesi sağlanmıştır. Şekil 2.26'da görüldüğü üzere λ hassasiyetinin optimum aralıkta seçilmesi gereklidir. Çünkü λ değeri gereğinden büyük seçildikçe taranacak r_n değeri artacak ve sınırların ötesinde diğer köpük sınırlarıyla karışmasına neden olacaktır. Seçilen aralık için (2.19)'da görüldüğü üzere $\mu_{\theta_n r_n}$, minimum değeri kabul edilmiş ve her açı için $\mu_{\theta_n r_n}$ değerini veren r_n seçilmiştir.

$$0^\circ \le \theta < 360^\circ \ \wedge \ \theta_{n-1} + 22.5^\circ = \theta_n \tag{2.17}$$

$$r_b = r_n - \lambda r_n \ \wedge \ r_s = r_n + \lambda r_n \rightarrow r_b \le r_n \le r_s \ \wedge \ r_{n-1} + 1 = r_n \tag{2.18}$$

$$\mu_{\theta_n r_n} = J\left(\left(i - \sin\left(\frac{\theta_n \pi}{180}\right) r_n\right), \left(j + \cos\left(\frac{\theta_n \pi}{180}\right) r_n\right)\right) \rightarrow \mu_{\theta_n r_n} \le \forall \ \mu_{\theta_k r_k} \tag{2.19}$$



Şekil 2.26. Açısal minimum takip

Yeni yarıçap r' değerinin hesaplanması için, (2.20)'de görüldüğü üzere seçilen r_n değerlerinin ortalaması alınmıştır. Diğer yandan daha önceden yapılan ağırlık

merkezi tespiti işlemi nedeniyle, yarıçap dışında *i, j* koordinatları için yeni bir oryantasyona gerek görülmemiştir.

$$r' = \frac{1}{8}\sum_{n=1}^{8} r_n \tag{2.20}$$

Denklem (2.8)'de ifade edildiği gibi istenilen yarıçaplara göre tarama yapıldıktan sonra iç içe geçmiş fakat merkezleri birbirinden uzak ve farklı köpükleri temsil eden potansiyel kabarcıkları filtrelemek için bir metot geliştirilmiştir. Bu metota göre önceden (2.15)'te ifade edildiği gibi merkezler arası Öklid mesafesi hesaplanmış ve (2.21)'de görülen karşılaştırma yapılmıştır. Kabarcıkların karşılıklı olarak ne kadar oranda sınırlardan girdiğini ifade eden η iç içe geçme oranı, görüntü özelliklerine göre değişmektedir. Şekil 2.10'da görüldüğü üzere büyük kabarcıkların yüzeyinde daha küçük ve daha az belirgin kabarcıklar bulunmakta ayrıca köpükler arasında boşluk bulunmamaktadır. Diğer yandan Şekil 2.9'da görüldüğü üzere 1. tip görüntülerde, kabarcıklar arasında boşluklar bulunmaktadır. Dolayısıyla 1. tip görüntülerde katı koşullandırma yapabilmek için daha büyük, 2. tip görüntüler için daha küçük η oranı seçilmiştir.

$$d_{km} \le \eta \left(r_k + r_m \right) \tag{2.21}$$

Denklem (2.21)'de belirtilen koşula uyan köpükler arasında yarıçap karşılaştırılması yapılmıştır. Yapılan incelemeler sonucunda büyük olan köpüklerin daha doğru bulunduğu tespitinden yola çıkılarak, daha küçük yarıçaplı kabarcık filtrelenmiştir.

Filtrelenmiş köpüklerin alanları sınır takibine göre ortaya çıkarılmaktadır. Buna göre öncelikle (2.22)'de ifade edilen Ω,dönüş sayısı en yakın tam sayıya yuvarlanarak hesaplanmakta, ifade edilen *c* değerinin hesaplanması için (2.11)'den yararlanılmaktadır.

$$\Omega \in \mathbf{Z} \rightarrow \Omega = \frac{22,5}{c} \tag{2.22}$$

Her merkez için 45 derecelik açılarla, 22,5 derece pozitif ve negatif yönlere doğru sınır takibi yapılmaktadır. Her derece için pozitif yönde (2.23a) ve negatif yönde (2.23b)'de belirtilen derece artırımları yapılmıştır.

$$n < \Omega \ \rightarrow \theta_{\alpha_{n-1}} + \Omega \ = \ \theta_{\alpha_n} \tag{2.23a}$$

$$n < \Omega \ \rightarrow \ \theta_{\alpha_{n-1}} - \Omega \ = \ \theta_{\alpha_n} \tag{2.23b}$$

Yapılan her artırım için *k* adım sonrasına göre (2.24)'te ifade edilen ortalamalar *r*, *r-1* ve *r+1* için hesaplanmış ve sırasıyla α_r, α_{r-1}, α_{r+1} değerleri elde edilmiştir. Konturun hangi doğrultuda sapma yaptığını tespit edebilmek için α değerleri arasından minimum olanı ve buna bağlı olarak yeni yarıçap değeri seçilmiştir.

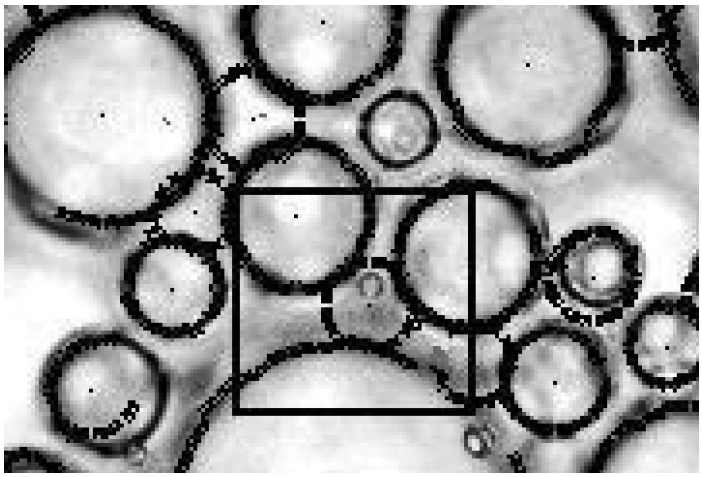
$$\alpha_r = \frac{1}{k}\sum_{n=1}^{k} J\left(\left(i - \sin\left(\frac{\theta_n \pi}{180}\right) r\right), \left(j + \cos\left(\frac{\theta_n \pi}{180}\right) r\right)\right) \tag{2.24}$$

Her adımda güncellenen yarıçapın dışında (2.25)'te açıklanan kontrol gerçekleştirilmiş ve o kabarcığa ait ortalamanın altında olup olmadığı tespit edilmiştir. Buna göre çizilebilen açıyı ifade eden η_k değeri (2.11)'de ifade edilen *c* adımı kadar artırılmaktadır. Fazla bölütlemeyi önleyebilmek için çizilebilen açı değeri (2.26)'da belirtildiği üzere toplanmış ve taranabilen toplam açı oranı ile karşılaştırılmıştır. Burada (2.22)'de yapılan hesaplama sırasında, yuvarlama işlemi sonucu doğan bilgi kaybından dolayı 360 derece yerine 16 $c\,\Omega$ ile oran alınmıştır. Görüntünün ışık dağılımı ve sınırların belirginliği gibi özelliklere göre seçilen η' çizim oranına göre köpüğün filtrelenmesi için geri besleme yapılmıştır.

$$J\left(i_\alpha, j_\alpha\right) \leq B(i, j, r) \rightarrow \eta_{k-1} + c = \eta_k \tag{2.25}$$

$$\eta_t = \sum_{\theta=0^\circ}^{360^\circ} \eta_\theta \rightarrow \frac{\eta_t}{16\ c\,\Omega} \geq \eta' \tag{2.26}$$

Örnek olarak Şekil 2.27'de var olmayan kabarcık için kontur takip tablosunun hücresel yapısı, Ek C'de gösterilmiştir. Buna göre ortalama değeri 116 olarak hesaplanan sözde kabarcık için kontur takibi 45 derecelik açılarla 22,5 derece pozitif ve negatif yönlere doğru yapılmıştır.



Şekil 2.27. Sözde kabarcık kesiti

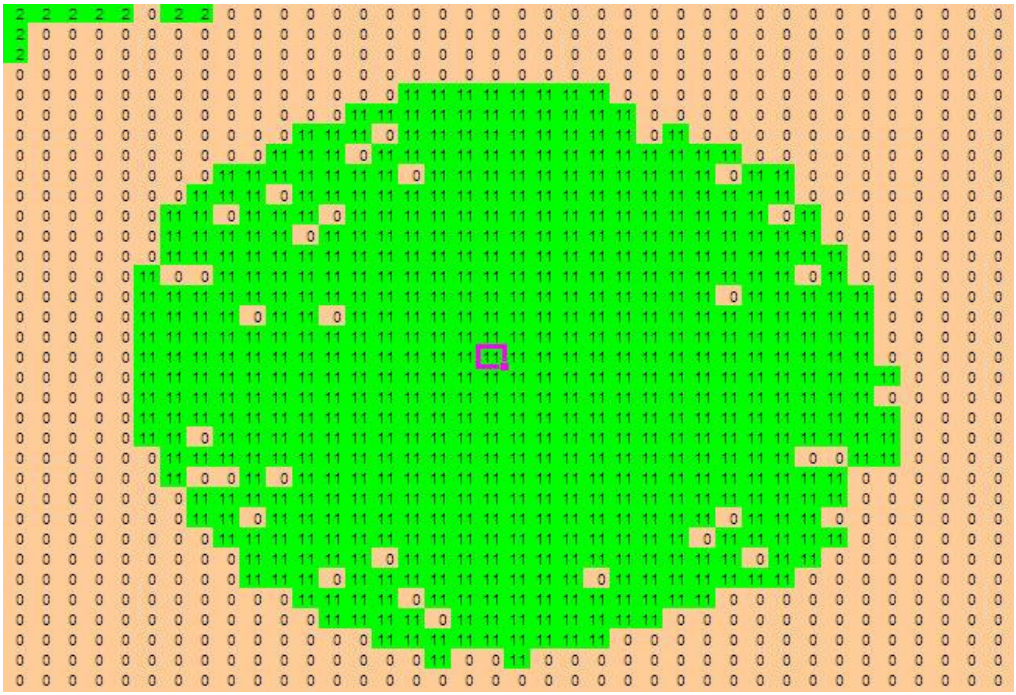
Çizim oranı 0,56 değeri Çizelge 2.2'de görüldüğü üzere η' oranı olarak seçilen 0,65 değerinin altında kalmıştır. Buna göre sözde kabarcık tespit edilip filtrelenmiştir.

Çizelge 2.2. Sözde kabarcık sınır takibi

Yönler	**Çizilebilen açı**	**Toplam açı**	**Çizim Oranı**
Kuzeybatı	44	44	1
Doğu	82	87	0,9375005
Kuzeydoğu	125	131	0,95833292
Güney	169	175	0,9687497
GüneyBatı	180	218	0,82500007
GüneyDoğu	185	262	0,70833311
Kuzey	196	305	0,64285712
Batı	196	349	0,56249991

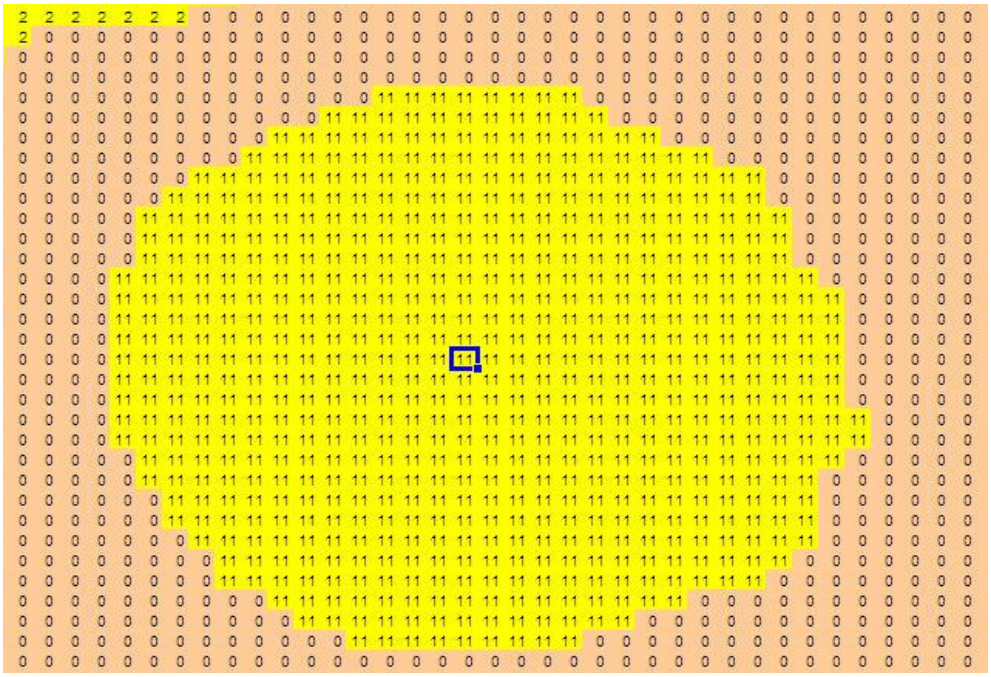
Çalışmada 1. tip görüntüler için daha yüksek η' çizim oranı şartı seçilirken, daha belirsiz sınırlara sahip 2. tip görüntüler için daha düşük η' çizim oranları seçilmiştir. Dolayısıyla köpük sınırları etrafında çizilebilen tur oranı düşük olan ve fazla bölütlemeye yol açan kabarcıklar P kümesinden çıkarılmıştır.

Diğer yandan sınır takibi sırasında, Şekil 2.28'de hücresel yapısı görüldüğü üzere L sıfırlar matrisi, köpüklerin etiketlenmesi için, çizilen yarıçap mesafesinden merkeze kadar kabarcığa ait numara ile işaretlenmiştir. Şekil 2.28'de görüldüğü üzere, polar koordinatlarda yapılan takibin kartezyen koordinatlara dönüşürken kabarcık alanlarının içerisinde olmaması gereken boşluklara neden olduğu gözlemlenmiştir.



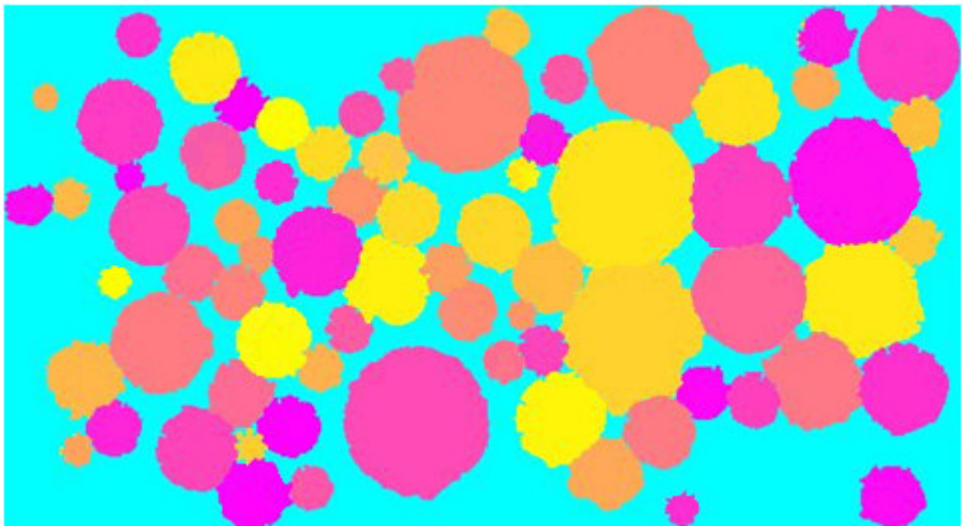
Şekil 2.28. Doldurma öncesi hücresel görünüm

Bu boşlukları doldurmak için 3x3 komşularına göre bir çoğunluğa göre doldurma işlemi gerçekleştirilmiştir. L matrisi üzerinde her piksel, 3x3 komşularında salt çoğunluğu elde eden, diğer bir ifadeyle 5'i geçen etiket numarası ile değiştirilmiştir. Sonuçta Şekil 2.29'da görülen sonuç elde edilmiş ve başarılı olduğu gözlemlenmiştir.



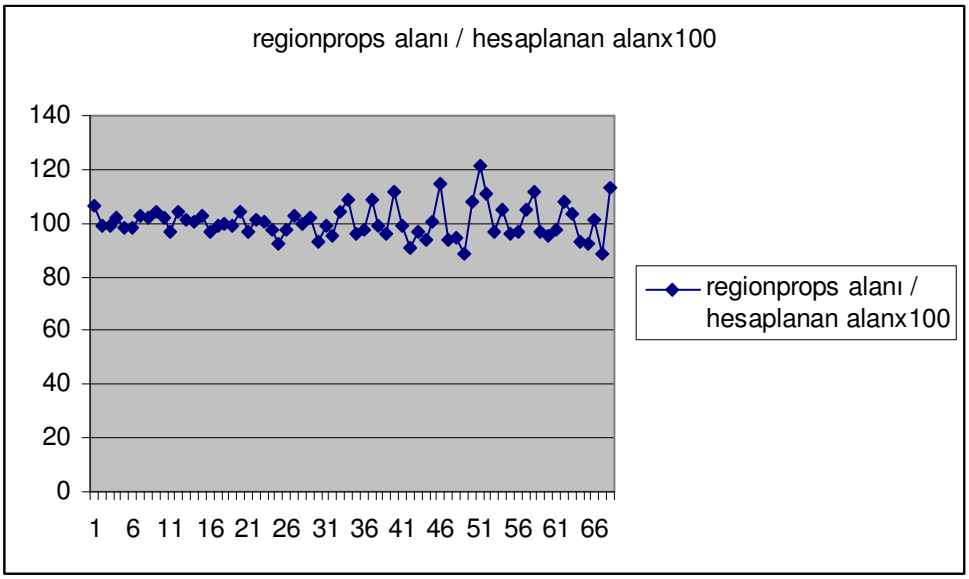
Şekil 2.29. Doldurma sonrası hücresel görünüm

Bütün köpükler için etiketleme yapıldıktan sonra Matlab yazılımı içerisindeki label2rgb [32] yardımı ile *L* matrisinden elde edilen görüntü Şekil 2.30'da görülmektedir.



Şekil 2.30. Köpüklerin Renklendirilmiş Etiketli Görüntüsü

Seçilen köpüklerin çap, alan ve çevre değerlerinin çıkarılması için Matlab yazılımı içerisinde bulunan regionprops metodundan [32] yararlanılmıştır. Bu metoda göre L matrisinde ayrı olarak ifade edilen bütün kabarcıklar için sınır takibi ile çevre hesaplanmıştır. Bu sınır takibi için 3X3 komşuluk bağlantısı esas alınmış ve içte kalan bölge için hücresel sayım metodu ile alan çıkarımı yapılmıştır. Yapılan alan çıkarımı ile tespit edilen r' ortalama yarıçap değerleri ile hesaplanan alan karşılaştırması Şekil 2.31'de görüldüğü üzere yapılmış ve birbirine uyumlu olduğu gözlemlenmiştir.



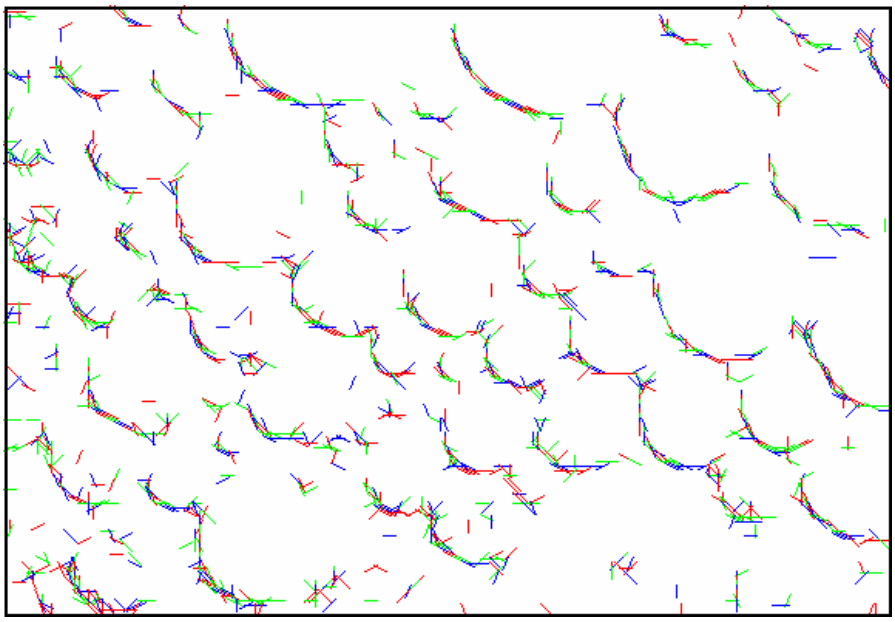
Şekil 2.31. Regionprops alanı, hesaplanan alan karşılaştırması

Her kabarcık için elde edilen çevre ve alan değerlerinden (2.27)'de belirtilen denkleme göre δ yuvarlaklık oranları hesaplanmıştır [1]. Oranların ideal yuvarlaklık oranı δ_i ile karşılaştırması yapılmış ve yuvarlaklık tespiti yapılmıştır.

$$\delta = \frac{çevre^2}{alan} \rightarrow \delta_i = \frac{(2\pi r)^2}{\pi r^2} = 4\pi \tag{2.27}$$

Geliştirilen modele göre köpüklerin boyutsal ve biçimsel bilgileri çıkarılmış fakat işlemsel karmaşık değerinin yüksek olması ve işleme alınan veri kümelerinin çok büyük boyutlarda olmasından dolayı literatürde kullanılan tekniklere göre çok daha uzun sürede sonuca ulaşmaktadır. Bu sürenin azaltılması açısından daha az karmaşıklığa sahip bir model geliştirilmiştir.

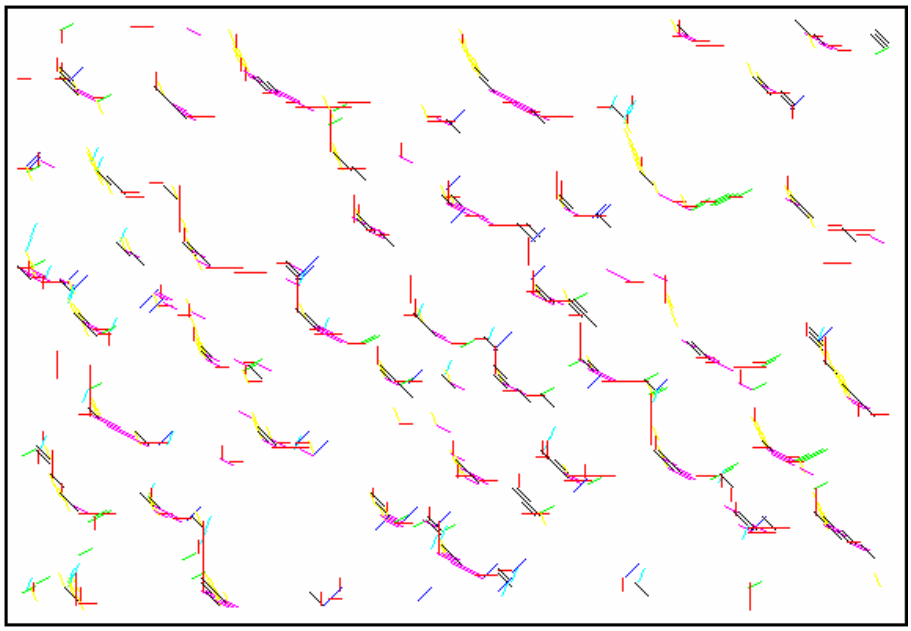
Modele göre ilk önce 3'er piksel aralıklarla, 22,5 derecelik açılarla [0-157,5] arasında sanal eksenler varsayılmıştır. Bu eksenlerin ortalamaları hesaplanmış, bulunan piksele göre aralarındaki en büyük 3 tanesinin ortalaması alınmış ve minimum olan ile karşılaştırılmıştır. Bu karşılaştırmaya göre bulunulan pikselin köpük sınırları üzerinde olup olmadığı kontrolü yapılmıştır. Görüntüdeki gürültü ve sınırların koyuluğuna göre maksimumların ortalaması ve minimum arasında belirli bir düşüş beklenmiş ve buna göre filtreleme yapılmış ve 2. tip görüntülerde Şekil 2.32.'de görülen sonuç elde edilmiştir.



Şekil 2.32. Sınır Eksenleri çıkarımı sonrası

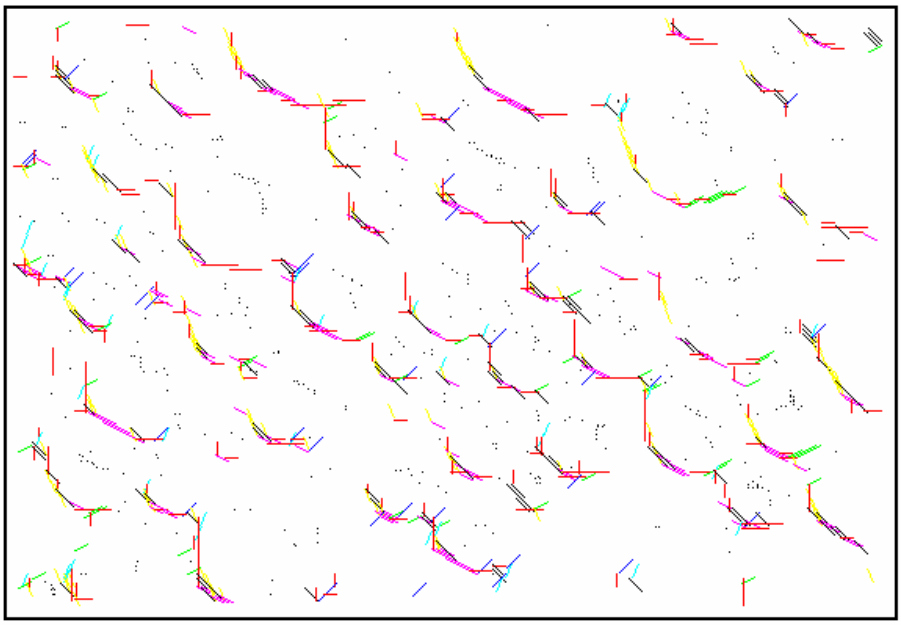
Yapılan incelemeler sonucu fazlalık yaratan ve köpük sınırına ve etrafındaki eksenlere uygun yapıda olmayan eksenler fark edilmiştir. Görüntüde fazlalık yaratan

ve yanlış sonuçlar vermesine sebep olabilecek bu eksenlerin filtrelenmesi için ikili komşu eksen ilişkilerine bakılmıştır. Köpüklerin biçim itibariyle dışbükey çokgen olduğu ve dörtgenden fazla kenarlı dışbükey çokgenlerde ikili çizgiler arasında 90 dereceden fazla açı olması şartlarından yola çıkılarak bir filtreleme uygulanmıştır. Buna göre her eksen, komşusuyla açısına göre kontrol edilmiş ve aralarındaki açının en az 112,5 derece olması şartıyla şablonlar oluşturulmuştur. Elde edilen şablonlara göre yapılan filtrelemeler sonucunda Şekil 2.33.'te görülen daha az yanlış bilgi içeren sonuç elde edilmiştir.



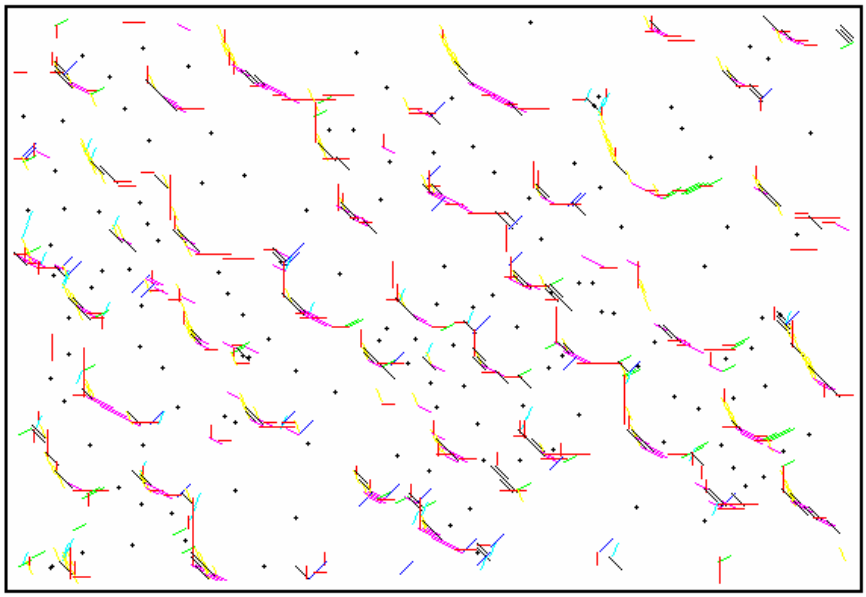
Şekil 2.33. İkili şablon filtreleri sonrası

Şekil 2.10.'da bir kesiti görüldüğü üzere 2. tip görüntülerden sağ üst taraftan verilen ışık sonucu köpüklerin sol alt sınırları gölgeler ile ortaya çıkmış fakat sağ üst kısımdaki sınırlar parlamış ve belirsizleşmiştir. Diğer yandan köpüklerin sağ üstünde yer alan diğer köpüklerin sol alt sınırları, köpüklerin karşılıklı sınırları olarak varsayılmıştır. Buna göre karşılıklı eksenler arasındaki orta noktaların köpük merkezine yakın olacağı yaklaşımı ile orta noktalar tespit edilmiş ve Şekil 2.34'te görülen merkez dağılımları ortaya çıkarılmıştır.



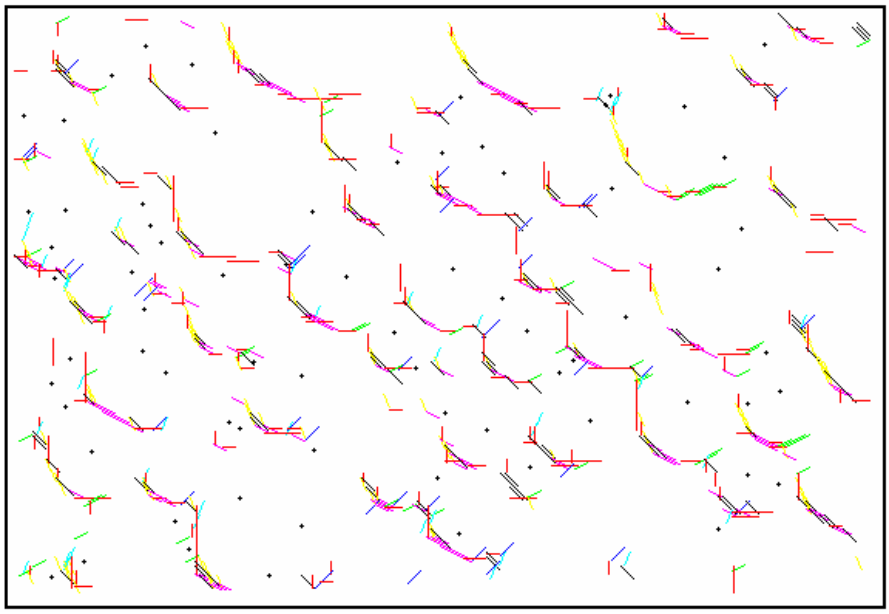
Şekil 2.34. Eksenler arasındaki orta noktalar

Birbirine yakın ve aynı köpüğü ifade etmeye çalışan orta noktalar için Bölüm 2.3.1.'de anlatılan ağırlık merkezi bulma metoduna göre köpüklerin merkezleri çıkarılmıştır. Sonuçta Şekil 2.35'de görülen merkezler elde edilmiştir.



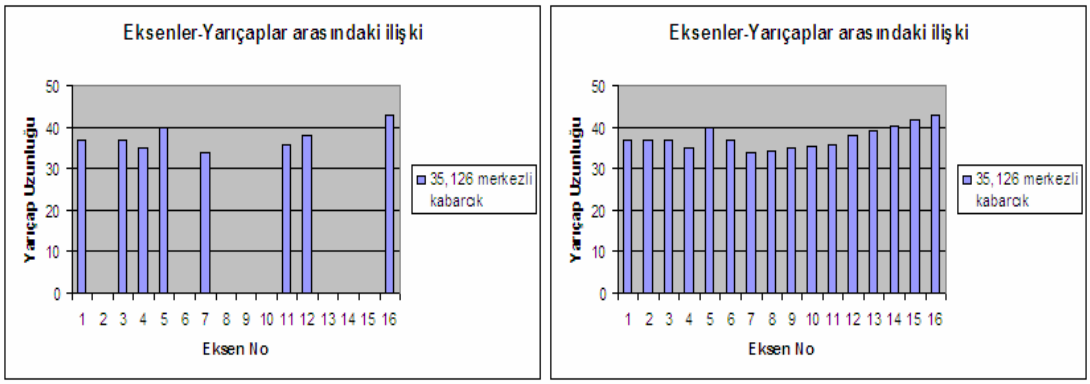
Şekil 2.35. Orta noktalar üzerinden Ağırlık Merkezi Hesaplamaları Sonucu

Elde edilen merkez noktaları gerçek merkezlere göre yakın koordinatlarda bulunmasına rağmen Şekil 2.35'te görüldüğü üzere bazı kabarcık alanları içerisinde birden fazla hatalı merkez çıkarılmıştır. Yanlış bilgiler içeren bu sözde köpükleri filtrelemek için bulunan her merkezden 10 derecelik doğrultularla tarama yapılmıştır. Her doğrultu için dik, 67,5 veya 112,5 derecelik bir eksene çarpanların sayısı kontrol edilmiş ve belli bir oranın altında kalanlar elenmiştir. Bu oranın belirlenmesinde görüntünün kirlilik oranı ve sınırların belirginliği göz önüne alınmıştır. Şekil 2.36'da eksenler üzerinden görülen merkezler sonuç olarak elde edilmiştir.



Şekil 2.36. Doğrultulara göre yapılan tarama sonrası merkezler

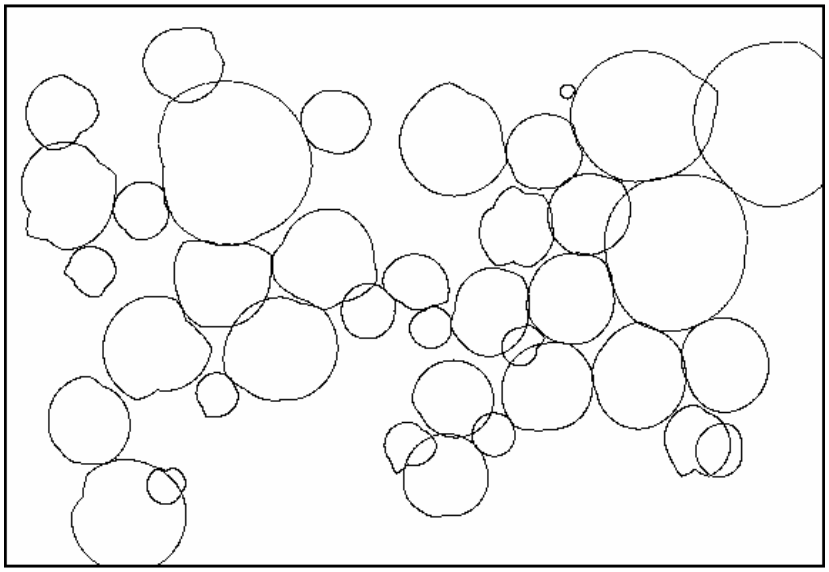
Bulunan merkezler gerçek değerlere göre yakın olmasına rağmen kabarcıkların bazı sınırlarında eksenler bulunmadığı için yarıçap değerlerine göre interpolasyon işlemi uygulanmaktadır. Şekil 2.37'de bir merkeze ait 10 derecelik açılarla tespit edilen değerlerin bir kısmı görüldüğü üzere, 0 olan değerlerin yerine kendisine en yakın olanlar arasında bir interpolasyon işlemi ile yeni değerler atanmıştır.



(a) (b)

Şekil 2.37. Eksenlere ait yarıçaplar arasında interpolasyon işlemi

Dolayısıyla sınırlarda oluşabilecek eğrilikleri kaldırarak gerçeğe yakın sınırlar ortaya çıkarılmıştır. Diğer yandan hala iç içe geçmiş kabarcık merkezleri bulunuyorsa, eksenlere ait yarıçap değerlerine göre standart sapma değerleri kontrol edilmiş ve en çok sapmaya sahip olan merkez filtrelenmiştir. Son olarak Şekil 2.38'de ikinci tip görüntüler için sonucu görüldüğü üzere, eldeki yarıçap değerlerine göre sınırlar, 1 derecelik açılarla interpolasyon yapılarak çizilmiştir. Kalan bölümde diğer modelde geliştirilen etiketleme işlemleri uygulanmış ve kabarcık alanları çıkarılmıştır.



Şekil 2.38. İkinci tip görüntüler için sınırların görünümü

Her iki görüntü analizi modeli sonucunda köpüklere ait merkez koordinat bilgileri, yarıçap ve çap değerleri, çevre, alan ve yuvarlaklık bilgileri elde edilmiştir. Bu verileri işlemek ve kimyasal ve fiziksel verilerle ilişkisini ortaya koyabilmek için bir YSA modeli kullanılmıştır.

2.3.2. Kullanılan YSA Modeli

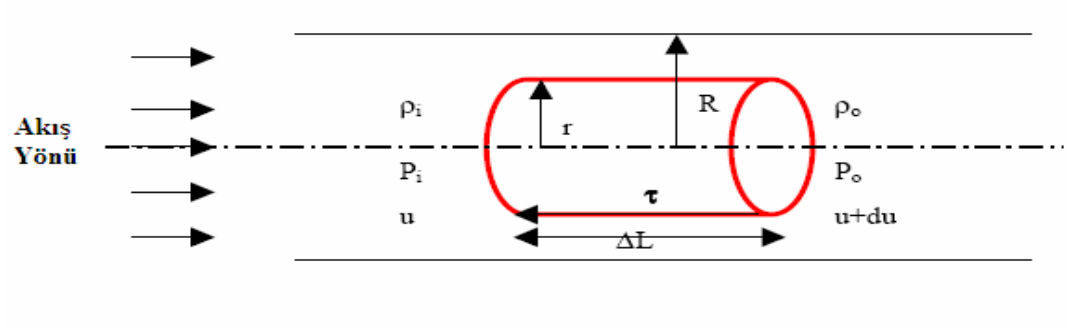
YSA, biyolojik nöronları referans alan ve yapay zeka işlevlerini yerine getirmek için ortaya konulmuş bir modeldir. Birçok nöronun belli bir düzende bir araya getirilmesi ve uygun öğrenme algoritmaları sayesinde sinir ağları kurulabilmekte ve bu ağlar çok karmaşık görevleri başarıyla yerine getirebilmektedir. Robot denetiminden, ses tanıma uygulamalarına kadar birçok alanda kullanılan YSA'nın dikkat çekici özelliği kendini oluşturan alt öğelerin hatalarını telafi edebilmesi ve karışık eşleştirmeleri, belli koşullar sağlandığı takdirde önceden tanımlanabilen bir hata sınır içerisinde gerçekleyebilmesidir. Öte yandan bir problemi çözmek için YSA'dan yardım alırken, çözülebilecek problem uzayının, insan beyninin çözebildiği problem uzayının alt kümesi olduğu gözden kaçırılmamalıdır [36].

YSA'nın dört temel özelliği bulunmaktadır. Birinci özellik, sistemin paralelliği ve toplamsal işlevin yapısal olarak dağılmışlığıdır. Bu durum birçok nöronun eşzamanlı çalışarak karmaşık bir işlevi yerine getirmesi ile açıklanabilir. İkinci temel özellik ise eğitim esnasında kullanılan verilerden eşleştirmeyi sağlayan kaba özellikleri çıkarması ve böylelikle eğitim esnasında kullanılmayan girdiler için de anlamlı yanıtlar üretebilmesidir. Diğer bir özellik ise ağ fonksiyonlarının nonlineer oluşudur. Diğer bir ifadeyle doğrusal olmayacak şekilde dağılmış alt birimler olmasına rağmen işlevin doğru biçimde yerine getirilebilmesini matematiksel olarak olası kılarlar. Son olarak diğer bir özellikleri ise, sayısal ortamda tasarlanan YSA modelinin sayısal ortamda gerçeklenebilir olmasıdır [36].

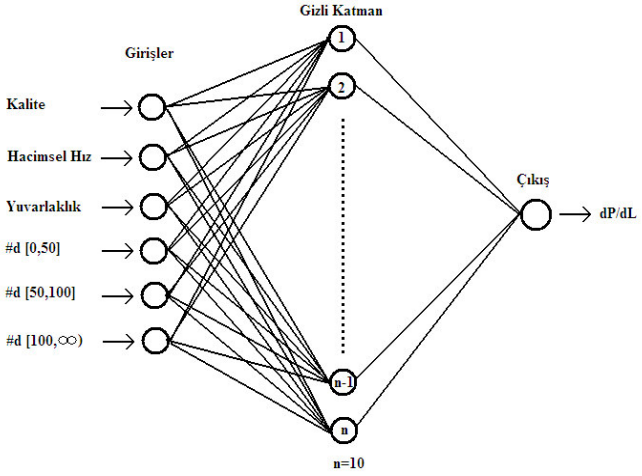
Diğer yandan YSA mimarisi ele alınırsa, nöronlar arası her girdi koluna ait bir ağırlık katsayısı bulunmaktadır. Tüm girdiler kendi ağırlık katsayıları ile çarpılır. Çarpım

değeri, YSA yapılarının temel taşı olan eşikleme mantık ünitesini geçtiği anda sonuç 1 çıkmaktadır. Bu şekilde eş zamanlı çalışan nöronların girdi ve çıktı ilişkisini kurabilecek şekilde bağlanması ile YSA oluşmaktadır. Örüntü tanıma, ZST ve veri işleme gibi bilimsel alanlarda kullanılan bu yapı sayesinde tıbbi, finansal, eposta filtreleme gibi bir çok ciddi uygulamada, polinom ve BOP sorunlara çözümler oluşturulmuştur [37].

Elde edilen verilere uygun kullanılacak YSA modeli için öğreticili öğrenme mekanizması seçilmiştir. Çünkü istenilen çıkış değerleri daha önce yapılan çalışmada [1] çıkarılmış olup YSA modelinde kullanılmaya hazır durumdadır. Diğer yandan ağ üzerindeki bilgi akışının sürekli ileri doğru yapıldığı ileri sürümlü mimari seçilmiştir. Birinci tip görüntüler için gazın toplam hacme oranını ifade eden köpük kalitesi, toplam debiyi ifade eden akışın boru hacmine göre hızı verileri girdi olarak T. Eren ve M. E. Özbayoğlu tarafından yapılan çalışmadan alınmıştır. Boru içerisinde L uzunluktaki silindirik kesit üzerindeki stresi çıkarmaya yarayan dP/dL verileri, YSA için öğretici çıktı olarak kullanılmıştır. Diğer yandan görüntü üzerindeki genel köpük boyutu dağılımını ortaya çıkarmak için [0,50], [50,100], [100,∞) olmak üzere çap aralıkları seçilmiş ve bu aralıklara göre sayım yapılmıştır. Ayrıca (2.27)'de ifade edilen yuvarlaklık oranları ortalaması çıkarılmıştır. Diğer yandan dP/dL, Şekil 2.39'da görüldüğü üzere boru içerisinde boruya paralel L uzunluğundaki silindirik bir kesit üzerinde, dairesel alanlar arasındaki basınç farkı olarak YSA modeline eklenmiştir. Model için dP/dL istenilen öğretici çıktı, kalan veriler girdi olarak Şekil 2.40'ta görülen YSA oluşturulmuştur.



Şekil 2.39. dP/dL görünümü



Şekil 2.40. Kullanılan YSA modeli

Sonuçta köpük kalitesi, boru hacmine göre hız, yuvarlaklık, 3 farklı çapa göre sayım değerleri olmak üzere 6 girdi nöronu bulunmaktadır. Öte yandan 1 gizli katman içerisinde 10 nörona sahip ve 1 çıktı değeri olmak üzere YSA modeli oluşturulmuştur. İkinci tip görüntüler, köpüklere ait kimyasal verilerin eksikliğinden dolayı YSA modeline sokulmamıştır.

BÖLÜM 3

3. SONUÇLAR VE TARTIŞMA

3.1. Geliştirme ve Test Ortamı

Görüntüler üzerinde yapılan işlemlerde matematiksel işlem yoğunluğu ve bu konuda yardımcı metotlar içerdiğinden dolayı Matlab yazılımı kullanılmıştır. Görüntü işleme teknikleri sonucu çıkarılan verilerin tabloda tutulması için Microsoft Excel 2003 programından yararlanılmıştır. Elde edilen verileri girdi şeklinde YSA ortamında eğitim ve test işlemlerinde ise Neuro Solutions yazılımı kullanılmıştır.

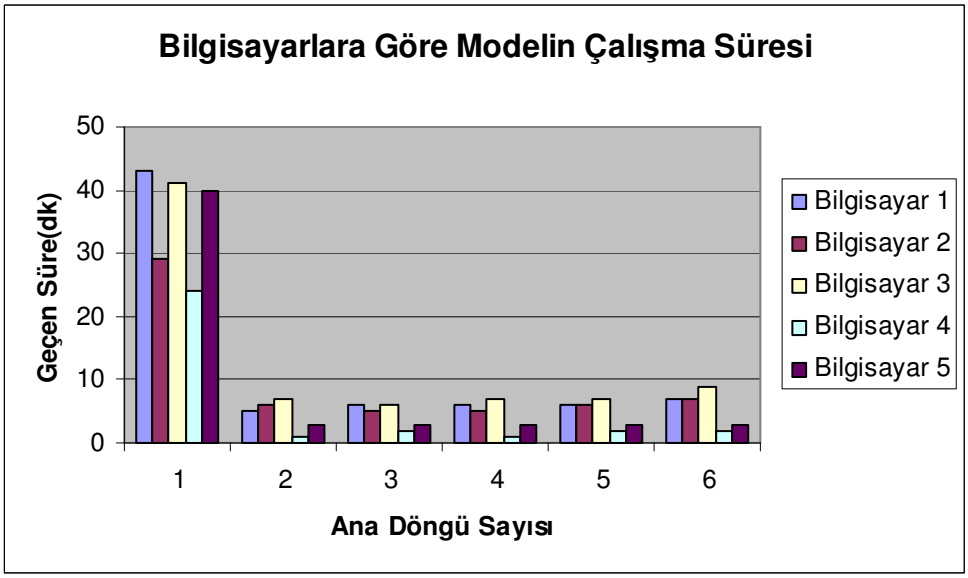
Bütün geliştirme ve test işlemleri Çizelge 3.1'de ifade edilen konfigürasyona sahip bilgisayarlar üzerinde yapılmıştır:

Çizelge 3.1. Geliştirme ve test bilgisayar konfigürasyonları

Bilgisayar No	İşlemci	REB	İS
1	Intel Pentium 4 2.26 GHz	896 MB	WindowsXP-32 bit- Service Pack 2
2	Intel Pentium M 1.6 GHz	512 MB	WindowsXP-32 bit- Service Pack 2
3	Intel Pentium 4 2.8 GHz	1 GB	WindowsXP-32 bit- Service Pack 2
4	Core 2 Duo 2.4 GHz, çift çekirdek	2 GB	WindowsXP-64 bit- Service Pack 2
5	Intel Pentium 4 3 GHz	1 GB	WindowsXP-32 bit- Service Pack 2

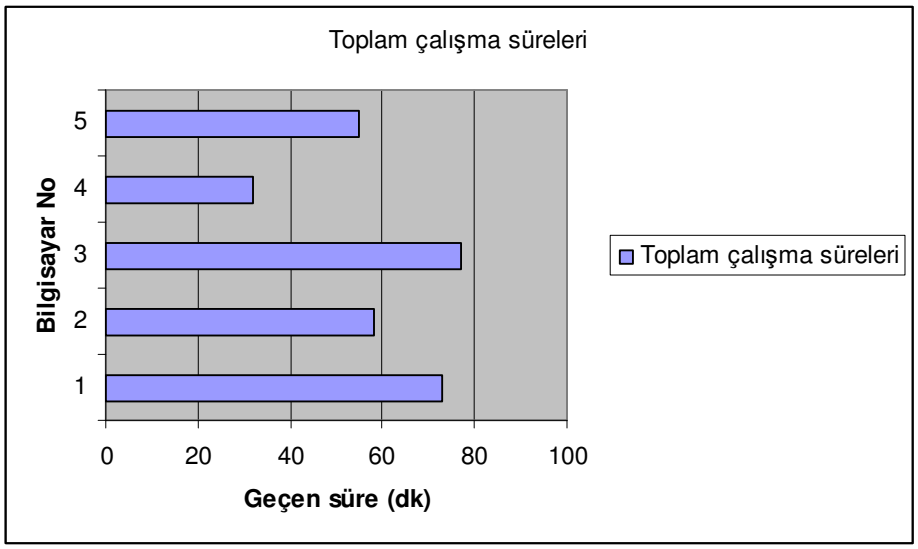
3.2. Uygulanan Testler ve Sonuçları

Geliştirilen modelin görüntü analizi işlevi, birinci tip için 97, ikinci tip için 10 farklı görüntü üzerinde test edilmiştir. Modelde yarıçap [10,65] aralığında tarama yapıldığı süre, zamana dayalı incelendiğinde Şekil 3.1'de görülen grafik elde edilmiştir. Grafikte ifade edilen ana döngü (2.8), (2.9), (2.10) ve sonrası yapılan açısal çevrim sonucu $\theta(n^4)$ karmaşıklığına sahiptir.



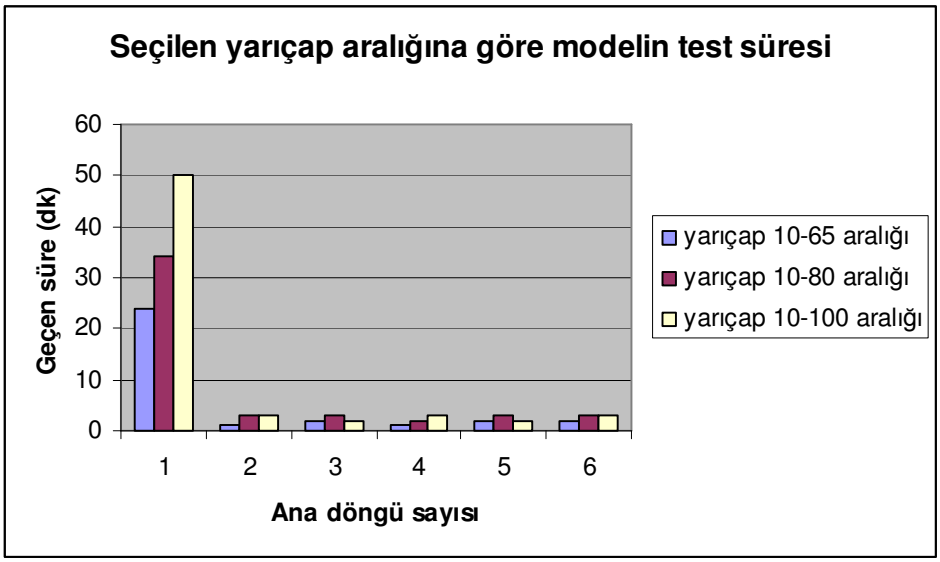
Şekil 3.1. Bilgisayarlara göre modelin çalışma süreleri

Modelde kullanılan uzaklık vektörünün etkisi neticesinde 1. ana döngüden sonra büyük düşüş yaşanmıştır. Bu sayede yapılan kontrol sonucu karmaşıklık $\theta(n^3)$'e yaklaşmış ve zaman anlamında ciddi bir kazanç sağlanmıştır. Diğer yandan bilgisayarlara göre toplam geçen süre olarak ele alındığında Şekil 3.2'de görülen sonuç elde edilmiştir.



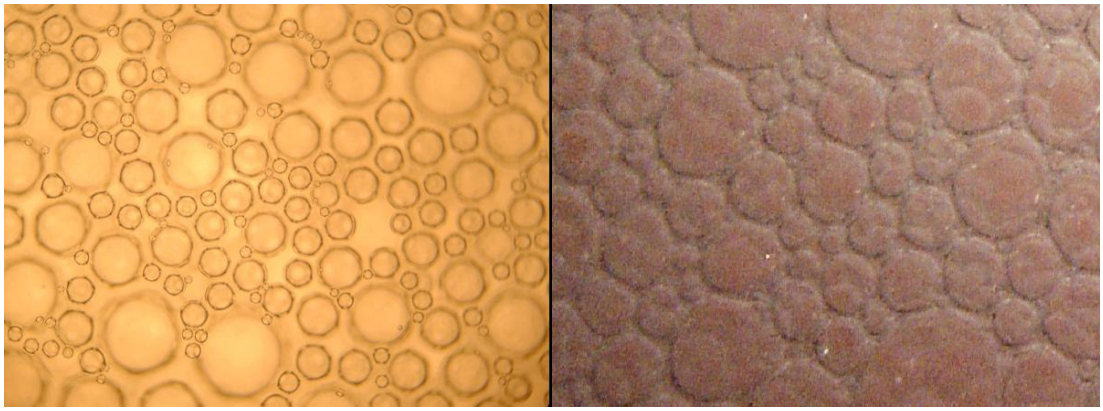
Şekil 3.2. Toplam çalışma sürelerinin bilgisayarlara göre dağılımı

Seçilen yarıçap aralığı değiştirilerek Bilgisayar 4 üzerinde yapılan testler sonucu geçen süre Şekil 3.3'te gösterilmiştir. Buna göre ele alınan görüntüde seçilen yarıçap aralığının, işlem süresini doğrusal olarak etkilediği gözlemlenmiştir. Dolayısıyla görüntülerin öncelikle içerdiği en büyük yarıçaplı kabarcığa göre sınıflandırılmasının gerektiği anlaşılmıştır.



Şekil 3.3. Seçilen yarıçap aralığına göre modelin test süresi

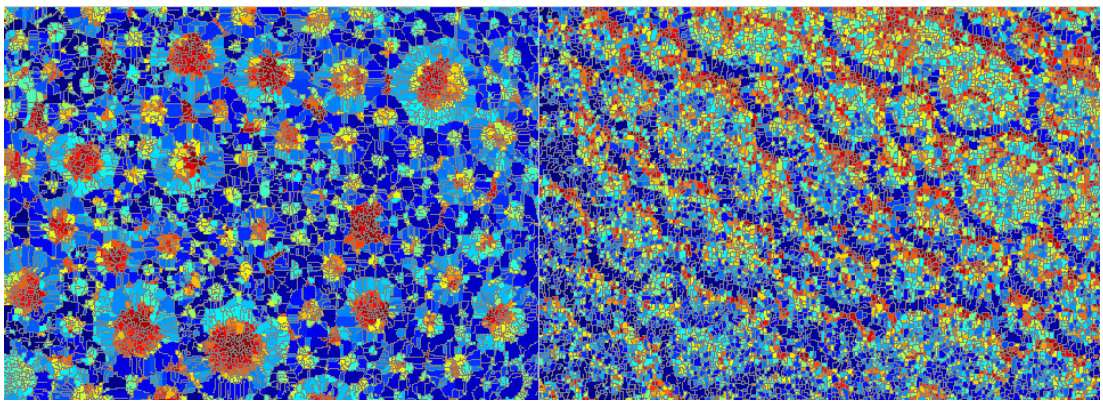
Geliştirilen model, literatürde bu konuda yaygın olarak kullanılan su bendi [16] yöntemi ile başarım açısından karşılaştırılmıştır. Başarımı artırmak için su bendi yöntemi öncesinde görüntülere, gri seviye değerleri üzerinden KLAHE metodu uygulanmıştır. Elde edilen matris, Matlab yazılımı içerisinde bulunan watershed metoduna [32] girdi olarak verilmiştir. Birinci ve ikinci tip resimler için denek olarak Şekil 3.4.a ve 3.4.b'deki görüntüler seçilmiş ve su bendi tekniği sonucu renklendirilmiş hali sonrası sırasıyla Şekil 3.5.a ve Şekil 3.5.b'de görülen sonuçlar elde edilmiştir.



(a) (b)

Şekil 3.4. Birinci ve ikinci tip için denek olarak seçilen görüntüler

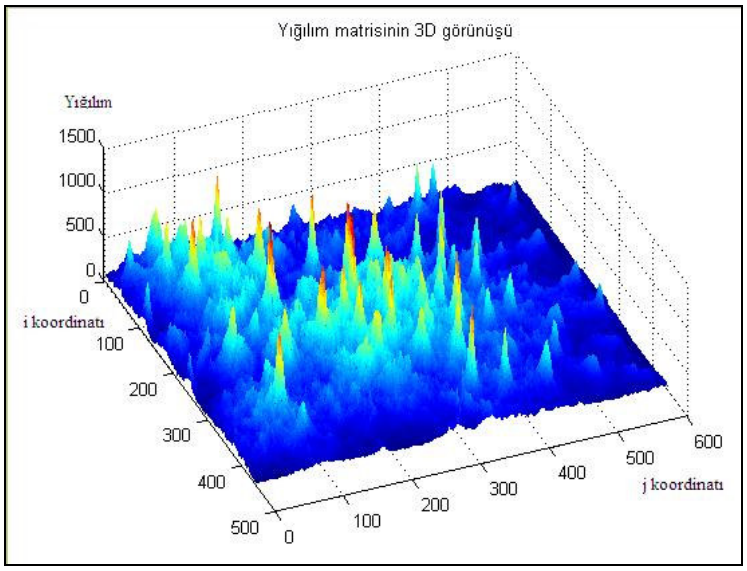
Şekil 3.5.a ve Şekil 3.5.b görüntülerinden anlaşılacağı üzere köpük hatları biraz belirginleşse de özellikle ikinci tip görüntüler için boyutlara ilişkin hesaplama yapmak mümkün değildir.



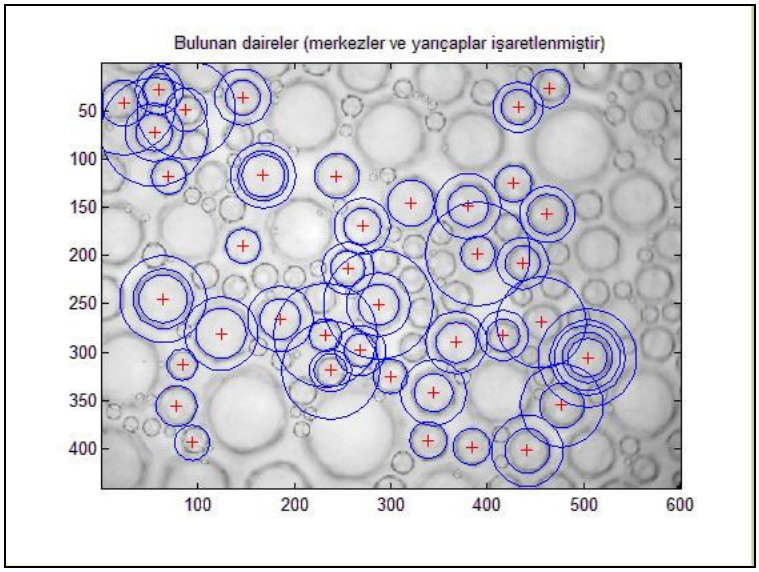
(a) (b)

Şekil 3.5. Birinci ve ikinci tip görüntülerde su bendi tekniği sonrası

Su bendi tekniği dışında köpük sınırlarını ortaya çıkarabilmek için T. Peng. Tarafından geliştirilen DHD metodu [38] Şekil 3.4.a'da görülen birinci tip görüntü üzerine uygulanmıştır. Sonuçta görüntüde, birçok yuvarlak hatta sahip kabarcık bulunmasına rağmen Şekil 3.6'da görülen 3 boyutlu yığılım matrisine göre Şekil 3.7'de bulunan merkezleri ve yarıçapları görülen sonuç elde edilmiştir.



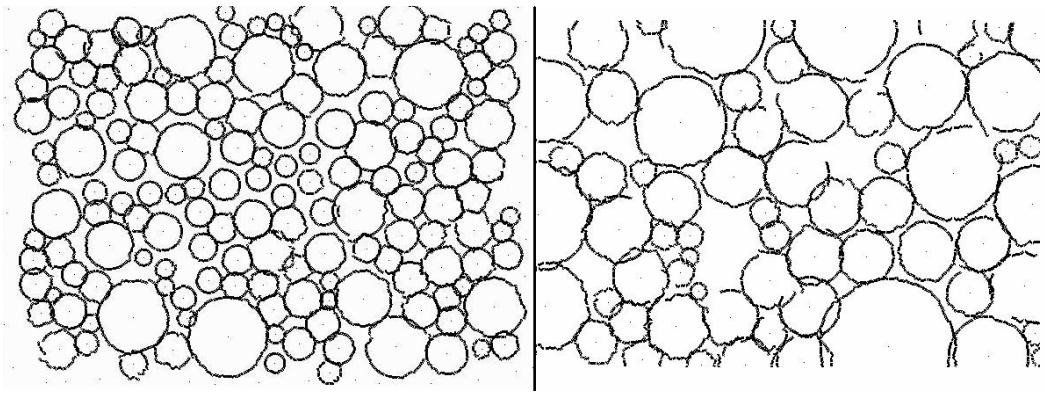
Şekil 3.6. Yığılım matrisinin 3 boyutlu görüntüsü



Şekil 3.7. DHD sonucu bulunan kabarcık merkezleri ve yarıçapları

DHD sonucu elde edilen merkezlerin gerçek görüntü ile görsel karşılaştırması teker teker sayım yapıldığında, bulunması gereken 97 kabarcıktan 14 tanesi tam doğru, 23 tanesi yanlış yarıçaplı toplam 37 adet bulunmuştur. Tam yuvarlak hatlara ve belirgin sınırlara sahip birinci tip resimler için bile DHD metodu ile çok düşük başarım oranı yakalanmıştır. Öte yandan DHD yöntemi ile köpük boyutu ortaya çıkarılabilmekte fakat köpüklerin her zaman tam yuvarlak hatta sahip olmayacağı göz önüne alınırsa biçimleri algılamak için yetersiz kaldığı ortadadır. Çok daha gürültülü ve belirsiz sınırlara sahip ikinci tip resimler için de DHD denenmiş fakat birkaç kabarcık bulması dışında sonuç verememiştir.

Diğer yandan aynı görüntülere geliştirilen model uygulanmış ve sırasıyla Şekil 3.8.a ve Şekil 3.8.b'de görülen sonuçlar elde edilmiştir. Sonuçta su bendi [16] ve DHD [38] teknikleri geliştirilen modelden çok daha hızlı işlemi bitirmesine rağmen köpükleri net olarak ortaya çıkaramamış ya da iyi bir başarım oranı yakalayamamıştır. Görüntülerden anlaşılacağı üzere geliştirilen model ile çok daha başarılı sonuçlar elde edilmiştir. Özellikle sınırları belirsiz ve gürültü oranı yüksek ikinci tip görüntüler için geliştirilen model ile nispeten başarılı sayılabilecek sonuçlar ortaya konulmuştur.

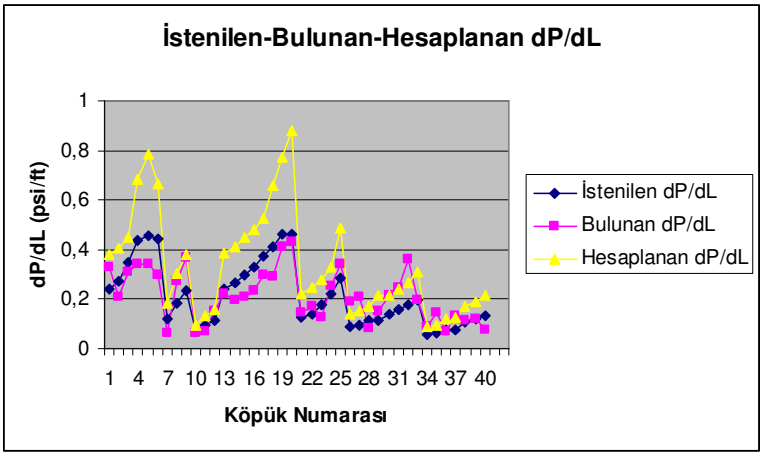


(a) (b)

Şekil 3.8. Birinci ve ikinci tip görüntüler için geliştirilen model sonrası sınırların görünümü

Geliştirilen modelin görsel olarak elde edilen değerlere göre başarım oranı gösteriminde denek olarak birinci tip için, Şekil 3.4.a'da görülen görüntü ele alınmış ve modelde ortaya konulan yöntemlere göre test edilmiştir. Buna göre kenarlara değmeyen köpüklere ait merkez koordinatları ve yarıçapları ile görsel olarak karşılaştırıldığında, bulunması gereken 97 kabarcıktan 78'i tam doğru, 7 tanesi yanlış yarıçaplı olarak bulunmuş, 12 tanesi ise bulunamamıştır. Diğer yandan sınır takip filtreleri sonucunda indirgenmiş olmasına rağmen 32 tane olması gerekenden fazla kabarcık bulunmuştur. Aynı teknik, Şekil 3.4.b'de görülen ve merkezi ışık dağılımı sonucu belirsiz yarım daire şeklinde sınırlara sahip ikinci tip görüntülerde, bulunması gereken 56 köpükten 34'ünü tam doğru, 2 tanesini yanlış yarıçaplı bulmuş, 20 tanesini ise bulamamıştır. Öte yandan zaman açısından daha hızlı olarak geliştirilen ikinci model, ikinci tip görüntülerde 46 köpükten 30'unu doğru olarak tespit etmiştir. İkinci model için toplamda 10 köpüğün dikkate alınmamasının sebebi ikinci modelin sınırlara yakın köpükleri dikkate almamasından kaynaklıdır. Ayrıca köpükler arasında boşluklar bulunduğundan dolayı birinci tip görüntülerde hıza yönelik olarak geliştirilen ikinci teknik uygulanmamıştır. Çünkü eksenler arasında kalan bu boşluklar geliştirilen modele göre köpük olarak algılanmakta ve fazla bölütlenmeye yol açmaktadır. Sonuçta yakalanan köpük oranı tam isabetli sonuç vermemiş olsa da, su bendi ve DHD gibi literatürde bulunan tekniklerin yetersiz olduğu ortamda iyi bir başarım sağlamıştır.

Genel olarak köpük boyutları ve karakteristiği korelasyonu karşılaştırmasını yapabilmek için birinci tip görüntülerden 97 tanesi geliştirilen model ile test edilmiş ve köpük boyutları, yuvarlaklık gibi özellikleri çıkarılmıştır. Bu görüntü değerlerinin 40 tanesi eğitim, 40 tanesi test ve 17 tanesi doğrulama adımlarında kullanılmak üzere ÇKP modeli uygulanmıştır. Elde edilen sonuç Şekil 3.9'de görüldüğü gibi birer birer incelendiği zaman farklar olmasına rağmen istenilen ve bulunan dP/dL arasında genel ilişkiyi yakalamıştır.



Şekil 3.9. İstenilen, hesaplanan ve model ile bulunan dP/dL arasındaki ilişki

Şekil 3.9'da bulunan grafikten görüldüğü gibi geliştirilen model, T. Eren ve M. E. Özbayoğlu tarafından yapılan çalışmanın [1] nihai hedefi olmamasına rağmen bir yan işlem olarak yapılan hesaplamalardaki sonuçlara göre, istenilen dP/dL değerlerine göre daha yakın sonuçlar vermiştir.

Diğer yandan yapılan çalışma ve T. Eren tarafından geliştirilen metodun [1] istenilen değerlere göre yüzde hata oranları ortalaması karşılaştırılmıştır. Çalışmada bir yan sonuç olarak hesaplanan dP/dL için % 57,19 oranında hata payı bulunmasına rağmen geliştirilen model ile bu oran % 36,09 seviyesine indirilmiştir.

Bütün olarak elde edilen sonuçlar ele alıındığında, geliştirilen yöntemin görüntü analizi yöntemleri ile literatürde yaygın olarak kullanılan su bendi [16], DHD [38] gibi tekniklere göre başarım açısından daha olumlu sonuçlar elde edildiği gözlemlenmiştir. Diğer tekniklerin geliştirilen modele göre çok daha hızlı çalışması önemli bir fark oluşturmakta fakat bunun nedeni geliştirilen yöntemin eldeki bilgiyi doğru filtrelemeye yönelik daha karmaşık bir algoritmaya sahip olmasından kaynaklanmaktadır. Diğer yandan aynı köpük görüntü kümesi ile yapılan araştırmada elde edilen yan sonuçlara [1] göre köpük boyutları ve dP/dL arasındaki ilişki açısından daha olumlu sonuçlar elde edilmiştir.

3.3. Geleceğe Yönelik Çalışmalar

Geliştirilen yöntem ile literatürde yaygın olarak kullanılan yöntemlere göre başarılı sonuçlar elde edilmesine rağmen daha isabetli sonuçlar alınabilmesine yönelik çalışmalar öncelikli olarak ele alınabilir. Bu yöndeki araştırmalar, daha önceden denenmiş [7] ve olumlu sonuçlar vermiş, eldeki görüntülerin sınıflandırılmasına yönelik olabilir. Sınıflandırılan görüntülere göre, sınırlar arası gri seviyedeki düşüklük oranı, sınır hassasiyeti gibi seçilen parametreler değiştirilerek, doğru oranda köpük bulmaya yönelik başarım oranı artırılabilir. Ayrıca kabarcıklar sınırları arası geçişlerdeki farkın değişken yapıya sahip olması bulanık mantık yöntemleri ile araştırılabilir ve dinamik bir geçiş oranı yakalanabilir.

Geliştirilen model, algoritmik olarak ele alınıp daha hızlı sonuca ulaşabilen teknikler geliştirilebilir. Deneme yanılma metodu yerine daha az karmaşıklığa sahip bir çözüm üretilmesi ya da kullanılan algoritmanın paralel çalışan bilgisayarlarda çalıştırılabilmesi, bilgisayar bilimleri açısından önem taşıyacaktır.

Elde edilen verileri eğiterek, daha iyi sonuç verebilecek değişik YSA modeli kullanılacak ve sonuçlar incelenecektir.

Ayrıca model içerisinde, yüzeysel gaz ve sıvı oranına göre de hesaplanabilen ve köpük kalitesini gösteren bir düzenleme yapılacak, hesaplanan değer ile ilişkisi ölçülecektir. Bu sayede kalite değerinin ortaya çıkması için hesaplama yapılması yerine eldeki görüntü işleme sonrası ortaya çıkan veriler kullanılmış olacaktır.

Diğer yandan sınırlarda görülen yoğunluk farkı, kuş bakışı ele alınan köpük görüntüsünde, 3. boyutta uzaklık farkını gösterdiği için bu yönde 3 boyutlu bir modelleme çalışması yapılabilir. Bu sayede köpüklerin gerçek izdüşümlerine göre hesaplama yapılacak ve perspektif bakış etkisi de sisteme dahil olabilecektir. Ayrıca incelenen köpük görüntülerinde sıvı ve gaz maddelerin dışında katı maddelerin de

bulunacağı göz önüne alınarak işlev artırılabilir fakat algoritmada, katı maddeleri ifade eden koyu renkli sınırlar için bir algoritmik düzenleme yapılması gerekecektir.

Bütün bunların dışında durağan görüntüler yerine dinamik köpük görüntüleri üzerinden köpüklerin akış hızı, boyut ve biçimsel değişimleri, alınan kesitlere göre incelenebilir ve gerçek zamanlı bir uygulanabilen bir çalışma haline getirilebilir. Böylece endüstriyel anlamda katkı artırılmış olacaktır.

KAYNAKLAR

[1] Eren, T., 2004, Foam Characterization: Bubble Size and texture Effects, *Master Tezi, Middle East Technology University*, Ankara, Türkiye.

[2] Symonds, P.J., De Jager, G., A technique for automatically segmenting images of the surface froth structures that are prevalent in flotation cells, Proceedings of the 1992 South African Symposium on Communications and Signal Processing COMSIG '92, 111-115, Rondebosch, Güney Afrika, Eylül 1992.

[3] Sadr-Kazemi, N., Cilliers, J.J., An image processing algorithm for measurement of flotation froth bubble size and shape distributions, Mineral Engineering, 10(10), 1075-1083, 1997.

[4] Moolman, D.W., Aldrich, C., Schmitz, G.P.J., van Deventer, J.S.J., The interrelationship between surface froth characteristics and industrial flotation performance, Minerals Engineering, 9(8), 837-854, 1996.

[5] Wang, W., Stephansson, O., A robust bubble delineation algorithm for froth images, The seconf International Conference on Intelligent Porcessing and Manufacturing of Materials IPMM'99, 471-476, Havai, Amerika Birleşik Devletleri, Temmuz 1999.

[6] Cilliers, J.J., Wang, M., Neethling, S.J., Measuring Flowing Foam Density Distributions Using ERT, 1st World Congress on Industrial Process Tomography, 108-112, Buxton, İngiltere, Nisan 1999.

[7] Bonifazi, G., Serranti, S., Volpe, F., Zuco, R., A combined morphological and color based approach to characterizeflotation froth bubbles, Intelligent Processing and Manufacturing of Materials IPMM '99, 465-470, Honolulu, Amerika Birleşik Devletleri, Temmuz 1999.

[8] Mckee, D.J., Automatic flotation control - A review of 20 years of effort, Minerals Engineering, 4(7-11), 653-666, 1991.

[9] Bonifazi, G., Serranti, S., Volpe, F., Zuco, R., Characterisation of flotation froth colour and structure by machine vision, Computers and Geosciences, 27(9), 1111-1117, 2001.

[10] Wang, W., Wang, L., Froth image segmentation algorithm, Signal Processing Proceedings, 2000. WCCC-ICSP 2000. 5th International Conference, 1, 480-483, 2000.

[11] Wang, W., Bergholm, F., Yang, B., Froth delineation based on image classification, Minerals Engineering, 16(11), 1183-1192, 2003.

[12] Wang, W., Colony Delineation on Image Classification, Icpr, 3, 705-708, 2006.

[13] Sternberg, S.R., Grayscale morphology, Computer vision, graphics, and image processing, 35, 333-355, 1986.

[14] Pearson, D.E., Robinson, J.A., Visual communication at very low data rates, Proceedings of the IEEE, 73(4), 795-812, 1985.

[15] Vincent, L., Morphological grayscale reconstruction in image analysis: applications and efficient algorithms, Image Processing, IEEE Transactions, 2(2), 176-201, 1993.

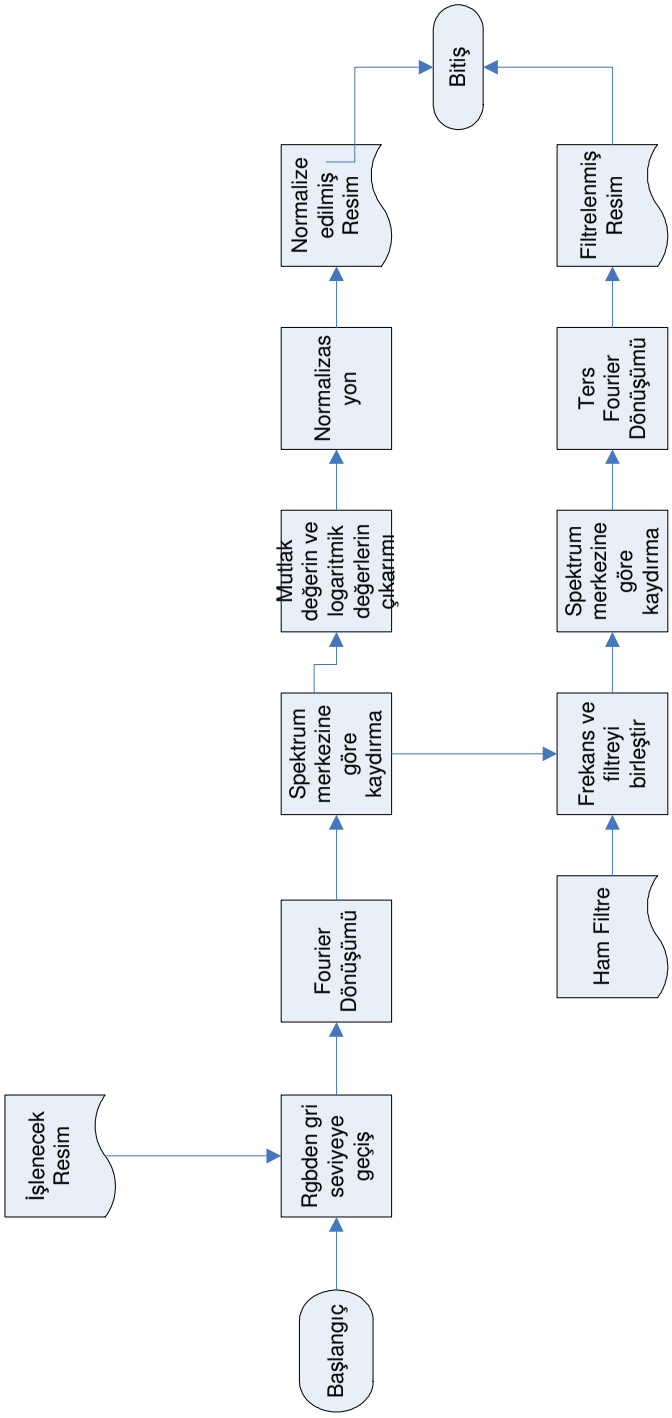
[16] Vincent, L., Soille, P., Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations, IEEE Transactions on Pattern Analysis and Machine Intelligence, 13(6), 583-598, 1991.

[17] Beucher, S., Lanteujoul, C., Use of watersheds in contour detection, International Workshop Image Processing, 17-21, Rennes, Fransa, Eylül 1979
[18] Ito, K., Xiong, K., Gaussian filters for nonlinear filtering problems, IEEE Transactions on Automatic Control, 45(5), 910-927, 2000.
[19] Otsu, N., A threshold selection method from graylevel histogram, IEEE Transactions on Systems, Man, and Cybernetics, 9, 62-66, 1979.
[20] Dougherty, E. R, Mathematical Morphology in Image Processing, *CRC*, New York, 1992.
[21] Suk, M., Chung, S.M., A new image segmentation technique based on partition mode test, Pattern Recognition, 16(5), 469-480,1983.
[22] Stephansson, O., Wang, W.X., Dahlhielm, S., Automatic image processing of aggregates, ZSRM Symposium: EUROCK '92, 14-17, Chester, Birleşik Krallık, 1992.
[23] Katsabanis, T., Franklin, J.A, Measurement of Blast of Fragmentation, *Taylor & Francis*, Rotterdam, 1996.
[24] Hargrave, J.M., Hall, S.T., Diagnosis of concentrate grade and mass flowrate in tin flotation from colour and surface texture analysis, Minerals Engineering, 10(6), 613-621, 1997.
[25] Beucher, S., Meyer, F., Morphological approach to segmentation: the watershed transformation, 433-482, 1993.
[26] Maurus, R., Ilchenko, V., Sattelmayer, T., Study of the bubble characteristics and the local void fraction in subcooled flow boiling using digital imaging and analysing techniques, Experimental Thermal and Fluid Science, 26(2-4), 147-155, 2002.
[27] Duda, R., Hart, P.E., Pattern Classification and Scene Analysis, John Wiley & Sons, 1973.
[28] Gonzalez, R.C., Woods, R.E., Digital Image Processing, Addison-Wesley Publishing Company, New Jersey, 1992.
[29] Jahne, B., Digital Image Processing, Springer, Berlin, 2002.
[30] Acharya, T., Ray, K.R., Image Processing Principles and Applications, Wiley-Interscience Publication, 2005.
[31] Canny, J., A Computational Approach To Edge Detection, IEEE Trans. Pattern Analysis and Machine Intelligence, 8, 679-714, 1986.
[32] "Documentation for MathWorks Products, R2007a" erişim adresi: http://www.mathworks.com/access/helpdesk/help/helpdesk.html, erişim tarihi: 20Mayıs 2007.
[33] Pratt, W.K., Digital Image Processing, John Wiley & Sons, New York, 1978.
[34] Xu, L., Oja, E., Kultanen, P., A new curve detection method: randomized Hough transform (RHT), Source Pattern Recognition Letters archive, 11(5), 331-338, 1990.
[35] Breu, H., Gil, J., Kirkpatrick, D., Werman, M., Linear Time Euclidean Distance Transform Algorithms, IEEE Transactions on Pattern Analysis and Machine Intelligence, 17(5), 529-533, 1995.
[36] Efe, Ö., Kaynak, O., Yapay Sinir Ağları ve Uygulamaları, Boğaziçi Üniversitesi Yayınevi, İstanbul, 2004.
[37] "The Free Encyclopedia" erişim adresi: http://en.wikipedia.org/wiki/Main_Page, erişim tarihi:20 Mayıs 2007.

[38] "An open exchange for the MATLAB and Simulink user community", erişim adresi: http://www.mathworks.com/matlabcentral/fileexchange, erişim tarihi: 10 Mayıs 2007.

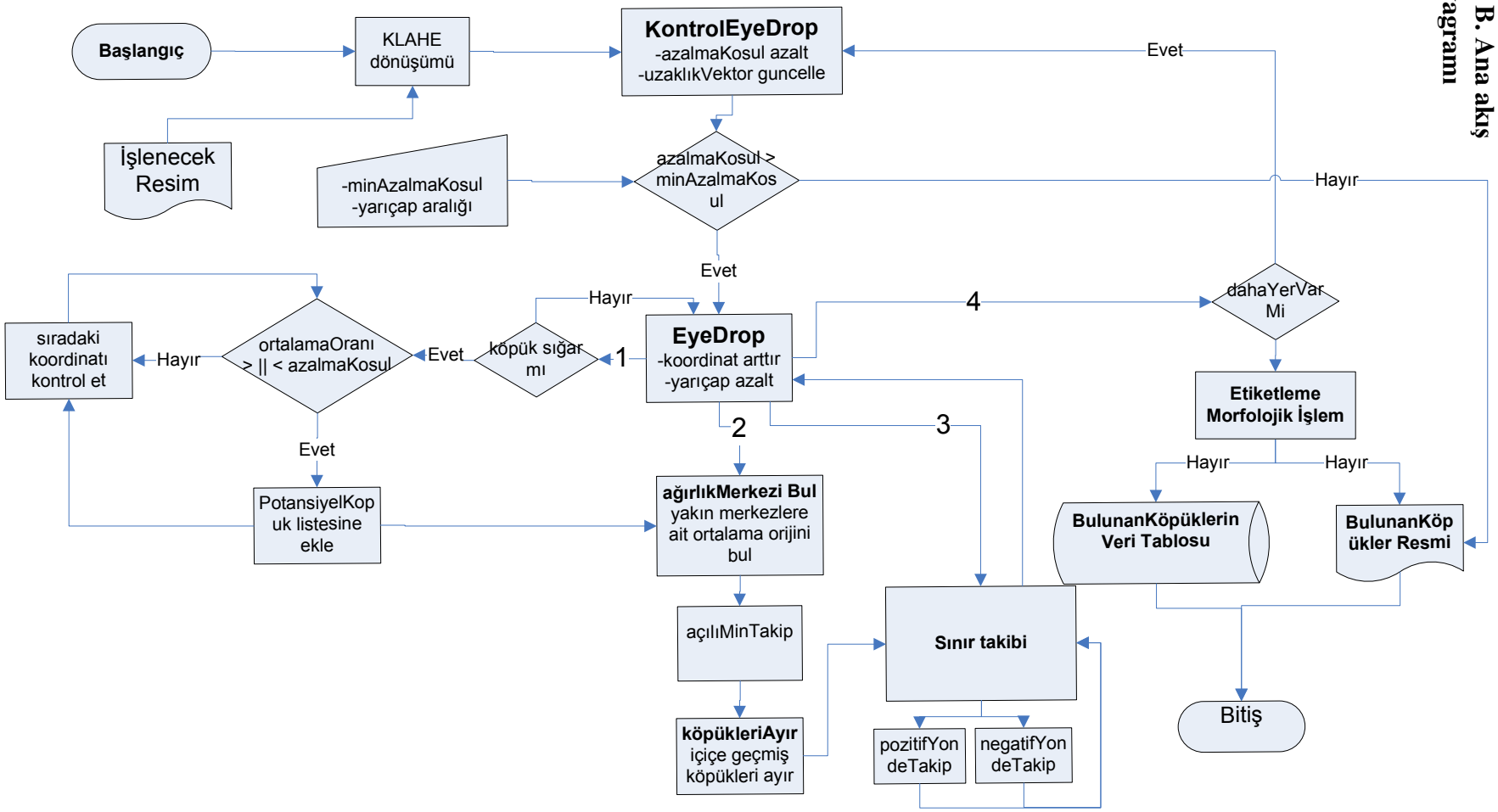
EKLER

Ek A. FD ve TFD İşlemleri



MMM

Ek B. Ana akış diyagramı



J(i,j)	251	252	253	254	255	256	257	258	259	260	261	262	263	264	265	266	267	268	269	270	271	272	273	274	275	276	277	278
194	233	224	210	205	190	176	150	118	77	43	15	11	61	130	180	203	205	199	205	145	78	31	17	79	162	167	157	146
195	229	229	223	203	170	135	109	74	49	15	11	42	116	181	192	203	203	198	184	125	47	10	7	64	150	166	167	146
196	217	219	209	184	139	99	65	35	20	7	25	77	171	206	171	187	213	203	188	130	29	0	11	70	150	166	166	146
197	222	208	167	134	98	61	32	11	6	22	71	135	164	100	60	98	109	188	203	157	36	3	40	84	127	156	156	160
198	183	157	111	79	50	26	17	24	36	72	110	155	103	82	107	131	74	105	196	157	75	33	79	99	107	139	155	166
199	95	81	57	39	31	33	47	72	89	113	138	110	74	160	176	190	131	100	146	181	100	54	86	95	82	96	145	170
200	46	42	36	40	60	85	118	127	152	153	153	86	75	184	174	164	150	84	141	187	109	45	53	67	57	67	121	159
201	46	60	81	123	156	176	187	187	192	183	167	153	91	169	174	164	139	61	135	180	114	52	18	18	31	50	99	149
202	135	155	170	185	185	206	210	201	196	187	162	162	89	118	217	178	77	106	164	170	135	68	10	0	6	31	74	144
203	199	209	215	215	210	215	210	201	181	171	166	162	176	148	157	128	120	163	169	160	145	92	43	8	1	14	52	109
204	235	240	233	229	219	215	199	174	174	174	176	176	176	187	177	183	177	153	159	159	149	125	98	53	21	10	22	67
205	234	231	229	219	203	188	180	174	174	174	180	170	164	150	162	146	132	132	138	163	155	149	160	121	71	33	17	26
206	212	203	203	198	183	173	188	169	163	170	159	164	159	139	141	117	117	118	148	138	138	138	174	180	130	75	40	17
207	192	187	187	171	167	169	163	153	157	120	149	149	149	139	130	98	113	117	132	127	128	138	163	209	195	135	85	54
208	181	177	181	156	148	157	162	148	153	132	148	144	124	134	124	99	105	113	127	132	118	142	159	188	215	185	135	93
209	160	166	156	152	156	131	152	162	157	137	137	142	144	132	120	106	120	121	121	131	127	132	142	173	199	203	195	164
210	145	150	150	135	125	141	150	146	146	142	146	152	157	148	144	124	130	124	135	135	146	146	142	163	173	203	224	210
211	135	135	130	130	131	141	135	156	156	150	160	146	142	157	157	132	138	139	130	145	150	162	156	152	142	159	203	224
212	139	134	139	149	139	145	160	150	141	141	141	135	135	135	131	152	148	153	149	164	160	156	152	131	127	142	184	203
213	149	159	159	159	155	149	139	130	120	116	106	106	110	107	117	131	148	153	153	149	159	150	146	146	152	152	177	184
214	132	138	144	134	128	134	110	110	92	85	82	82	85	79	102	103	117	127	128	134	144	145	135	131	152	152	167	183
215	109	113	109	114	114	114	109	100	92	84	74	74	78	70	71	99	91	99	109	109	124	138	134	130	150	166	167	173
216	99	95	103	107	103	118	113	113	99	91	84	81	74	74	82	74	72	72	86	92	109	114	120	139	145	146	156	157
217	107	117	137	131	142	142	131	132	123	107	113	103	107	88	88	85	75	75	75	84	81	95	105	120	125	139	141	127
218	150	162	166	171	181	166	171	166	156	152	156	142	137	123	118	109	100	106	102	93	91	95	95	96	105	114	125	127

Ortalama altında kalanlar | Ortalamayı geçenler | Mil taşları (22,5 derecelik açılarla) | Yarıçap Uzantısı

Ek D. Geliştilen Kod – Birinci Model

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Kodlayan:Veli Mert Altas-Bilgisayar Muhendisligi-Yuksek Lisans Ogr.
% Danışman:Dr. A. Murat Özbayoğlu
% TOBB Ekonomi ve Teknoloji Universitesi,Ankara-Turkiye,2007.
%%%% email:mertaltas@gmail.com, maltas@etu.edu.tr%%%%
% Coder:Veli Mert Altas-Computer Science-Grad Student
% Instructor:Dr. A. Murat Özbayoğlu
% TOBB University of Economy and Technology ,Ankara-Turkiye,2007.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function KontrolEyeDropV206(dizin, dosyaIsmi, uzanti,settings)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% AÇIKLAMALAR
% Ornek Cagirim:
% KontrolEyeDropV206 ('D:\Matlab\Arastirma\','10 min','.jpg',settings);

% settings icin asagidaki sekilde parametreler girilmelidir
% settings(1) = azalmaKosul baslangic değeri
% default 1.4 olarak ayarlanmıştır.
% settings(2) = azalmaKosul bitis değeri
% default 1.15 olarak ayarlanmıştır.
% settings(3) = Resim icindeki maksimum yaricapa gore ayarlanan baslangic
% değeri default 70 olarak ayarlanmıştır.
% settings(4) = Resim icindeki minimum yaricapa gore ayarlanan bitis
% değeri default 10 olarak ayarlanmıştır.
% settings(5) = Kabul edilen cizim orani
% default 0.5 olarak ayarlanmıştır.
% settings(6) = Min Kontur kontrol tarama orani
% default 20 olarak ayarlanmıştır.

% Ornek:
% Settings=[1.4, 1.15, 70, 10, 0.5, 20];
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

dosya=strcat(dizin,dosyaIsmi,'.',uzanti);

% dosya='D:\Matlab\Arastirma\10 min.jpg' seklinde birlestiriliyor.
ciktiDizini= strcat(dizin,dosyaIsmi,'Test Sonuclari','\Cizim Orani'...
    ,num2str(settings(5)),'\');
mkdir(ciktiDizini);
%%%%giris timestamp%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ts=1;
tBaslik={'yil','ay','gun','saat','dakika','saniye','cputime'};

timestamp(ts,1:6)=clock;
timestamp(ts,7)=cputime;

tStamp=num2cell(timestamp);
timexls = [tBaslik; tStamp]
xlswrite(strcat(ciktiDizini,'Time Stamps.xls'),timexls);
ts=ts+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Igray=isikFiltreV206(dosya,ciktiDizini);
% Isık parlamaları icin filtreleme islemi
```

```
[nRows nCols]=size(Igray);
% Resmin en, boy oranlarinin cikarilmasi

kontrolTrigger=1;
% Resimde herhangi bir kopuge daha yer olup olmadigina dair trigger

PotansiyelKopuk=ones(1,9);
% Potansiyel Kopuklerin tutuldugu matris
% default olarak 1ler matrisine set ediliyor.
FiltreliPotMerkez=PotansiyelKopuk;
% KonturTrace sonrasi filtrelenen kabarcıkların tutuldugu matris

Ipot = zeros(nRows,nCols);
Idist=bwdist(Ipot);
% Distance vektörünün kullanılabilmesi icin resimle
% ayni buyuklukte bir Ipot matrisi yaratilip distance matrisi cikariliyor.
% Daha sonra bu matrise bakılarak bos yer olup olmadığı taranacak.

B(1,1,1)=0;
% Butun ortalamalarin tutuldugu 3 boyutlu matris
Icember=0;
% Bulunan potansiyel merkezlerin ciziminin yapilacagi resimle
% ayni boyutlarda matris

Byazilditrigger=0;
% B matrisinin Excel tablosuna bir kere cikarilmasi icin trigger
% (B matrisi butun ortalamalari tutan matris)
ButOrtTrig=0;
%
azalmaKosul=settings(1);
azalmaKosulBitis=settings(2);
radBas=settings(3);
radSon=settings(4);
cizimOrani=settings(5);
minKonturTaramaYaricapi=settings(6);

%%%%%%%%buyuk yaricapli kopuklerin cikarimi%%%%%%
while(kontrolTrigger==1 && azalmaKosul>=azalmaKosulBitis)

        [FiltreliPotMerkez,PotansiyelKopuk,B,ButOrtTrig,Icember]=...
             EyeDropAzalmaDistanceV206(B,Byazilditrigger,Igray,Idist,...
             azalmaKosul,PotansiyelKopuk,ButOrtTrig,Icember,...
             radBas,radSon,FiltreliPotMerkez,cizimOrani,...
             minKonturTaramaYaricapi,ciktiDizini,dosyaIsmi);

        [Ipot]=MerkezUzaklikBulV206(PotansiyelKopuk,Igray,Ipot,ciktiDizini);
         %Uzaklık vektörünün güncellenmesi
         Idist=bwdist(Ipot);

         Byazilditrigger=1;
         kontrolTrigger=dahaYerVarMi(Idist,nRows,nCols);
         azalmaKosul=azalmaKosul-0.05;

         timestamp(ts,1:6)=clock;
```

```
        timestamp(ts,7)=cputime-timestamp(ts-1,7);
        ts=ts+1;
        tStamp=num2cell(timestamp);
        timexls = [tBaslik; tStamp]
        xlswrite(strcat(ciktiDizini,'Time Stamps.xls'),timexls);
end

function [Igray]=isikFiltreV206(image,ciktiDizini)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Görüntü üzerinde patlayan ışık değerlerinin filtrelenmesini sağlayan bir
%metottur. İncelenen görüntüye göre kullanılabilir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Gri Seviye Dönüşümü%%%%
Igray=imread(image);
Igray=rgb2gray(Igray);

imwrite (Igray,strcat(ciktiDizini,'GriSeviye.jpg'),'jpg');
imwrite (imhist (Igray,256),strcat(ciktiDizini,...
    'HistogramGriSeviye.jpg'),'jpg');

%%Görüntünün Gri Seviye Dönüşümü
meanVal=mean(mean(Igray));
[nRows nCols]=size(Igray);

%% Parlak Işık noktalarının filtrelenmesi
FiltreDegeri=220;
for i=1:nRows
    for j=1:nCols

        if(Igray(i,j)>=FiltreDegeri)

            Igray(i,j)=meanVal;
        end

    end
end

imwrite (Igray,strcat(ciktiDizini,'IsikFiltre.jpg'),'jpg');

function [FiltreliPotMerkez,PotansiyelKopuk,B,butunOrtalamayaziTrigger,...
    Icember]=EyeDropAzalmaDistanceV206(B,Byazilditrigger,Igray,Idist,...
    azalmaKosul,PotansiyelKopuk,butunOrtalamayaziTrigger,...
    Icember,radBas,radSon,FiltreliPotMerkez,cizimOrani,minKTr,...
    ciktiDizini,img_input)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR.
%Belirli koordinat ve yarıçap aralığına göre Potansiyel Kopuk kontrolünü
%sağlayan metot, bütün alt filtreleme metotları buradan çağırılmaktadır
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[nRows nCols]=size(Igray);
[z u]=size(PotansiyelKopuk);

if(PotansiyelKopuk(1,1)~=1)
    z=z+1;
```

```
end
% z=1;%PotansiyelKopuk matrisi uzerinde bulunan merkezleri eklememize
% yarayan indeks/sayac
C=uint32(0);
% Her 2D koordinatina ait ortalamayi tutabilmek icin olusturulan aratoplam
ctr=0;
% Her 2D koordinata ait ortalamayi tutabilmek icin olusturulan sayac

claheI = adapthisteq (Igray);
Iadjust = imadjust(claheI);

imwrite (Iadjust,strcat(ciktiDizini,'Clahe.jpg'),'jpg');
imwrite (imhist (Iadjust,256),strcat(ciktiDizini,'HistogramClahe.jpg'),...
    'jpg');

% adaptif thresholding elimizdeki resme uygulanmasi

 for i=1:nRows
     for j=1:nCols
         Icember(i,j)=255;
     end
 end
 IfiltreCember=Icember;
 % bulunan potansiyel ve filtrelenen merkezleri resim üzerinde gorebilmek
 % icin olusturulmus matrisler

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%her koordinat icin ortalama bulma ve potansiyel kopuk kesfi donguleri

for rad=radBas:-1:radSon
    sayac=1;
    % butun ortalamalari excelle cikti olmak icin olusturulan Tablo
    % matrisinde hangi rowa yazilacagini tutan sayac
    for i=1:3:nRows

        for j=1:3:nCols


           if(Idist(i,j)>=rad )
               %distance vektörüne göre yeterli yer var mi?

              if(Byazilditrigger==0)
                degCtr=360/(rad*10);
                % derece artisinin ne kadar yapilacaginin hesaplanmasi

                for degree=0:degCtr:360

                    idegeri=round(i-sin(degree*pi/180)*(rad+2));
                    jdegeri=round(j+cos(degree*pi/180)*(rad+2));
                    if(idegeri>=1 && idegeri<=nRows && jdegeri>=1 &&...
                            jdegeri<=nCols )
                    % resim boyutlari disina cikilip cikilmadigi kontrolu
                    C=C+uint32(Iadjust(round(i-sin(degree*pi/180)*rad),...
                        round(j+cos(degree*pi/180)*rad)));
                    C=C+uint32(Iadjust(round(i-sin(degree*pi/180)*...
```

```
            (rad+1)),round(j+cos(degree*pi/180)*(rad+1))));
        C=C+uint32(Iadjust(idegeri,jdegeri));
        % rad +2 hesabi,zaten yukarida hesaplandigi icin
        % tekrar hesaplanmamakta
        C=C+uint32(Iadjust(round(i-sin(degree*pi/180)*...
            (rad-1)),round(j+cos(degree*pi/180)*(rad-1))));
        % toplam deger
        C=C+uint32(Iadjust(round(i-sin(degree*pi/180)*...
            (rad-2)),round(j+cos(degree*pi/180)*(rad-2))));

        ctr=ctr+5;
        %ortalama alabilmek icin sayac

        end %if (idegeri>=1 && id...

    end%for degree

    B(i,j,rad)=  C / ctr;
    % ortalamanin B matrisine yazilmasi

    Tablo(sayac,1)=i;
    Tablo(sayac,2)=j;
    Tablo(sayac,rad)=B(i,j,rad);
    % butun ortalamalarin Excel tablosuna cikarilmasi icin
    % olusturulan matrise yazilmasi

    end %if(Byazilditrigger==0)



    if(rad<=radBas-5)

    if((B(i,j,rad+5)/B(i,j,rad))>=azalmaKosul && ...
            B(i,j,rad+5)~=0)



    PotansiyelKopuk(z,1)=i; %Potansiyel Kopugun i degeri
    PotansiyelKopuk(z,2)=j; %Potansiyel Kopugun j degeri
    PotansiyelKopuk(z,3)=rad;%Potansiyel Kopugun yarıçap degeri
    PotansiyelKopuk(z,4)=0;%İlerde cizilebilen aci tutulacaktir
    PotansiyelKopuk(z,5)=1;
    %İlerde ortak köpük sayısını tutacak sayaç
    PotansiyelKopuk(z,6)=0;
    %İlerde resmin icine dahil toplam aci tutulacak
    PotansiyelKopuk(z,7)=B(i,j,rad);%rad degerine göre ortalama
    PotansiyelKopuk(z,8)=B(i,j,rad+5);
    %rad+5 degerine göre ortalama
    PotansiyelKopuk(z,9)=0;
    %daha sonra label isleminde kullanilacaktir

    z=z+1;
    elseif ((B(i,j,rad)/B(i,j,rad+5))>=azalmaKosul...
            && B(i,j,rad+5)~=0)
```

```
                PotansiyelKopuk(z,1)=i;
                PotansiyelKopuk(z,2)=j;
                PotansiyelKopuk(z,3)=rad+5;
                PotansiyelKopuk(z,4)=0;
                PotansiyelKopuk(z,5)=1;
                 %İlerde ortak köpük sayısını tutacak sayaç
                PotansiyelKopuk(z,6)=0;
                PotansiyelKopuk(z,7)=B(i,j,rad+5);
                PotansiyelKopuk(z,8)=B(i,j,rad);
                PotansiyelKopuk(z,9)=0;
                z=z+1;

                end %if ((B(i,j,rad+5)...

                end %if (rad<=radBas-5)

            end%if (Idist(i,j)>=rad)
          sayac=sayac+1;
          C=uint32(0);
          ctr=0;

        end%for j

    end%for i

%%%%%%%%Ağırlık Merkezi Hesapla%%%%%%%%
[PotansiyelKopuk]=AgirlikMerkeziHesapla(PotansiyelKopuk);
% birbirine cok yakin bulunan merkezleri ayirt etmek

%%%%%%%%Açılı Minimum takip%%%%%%%%
[u g]=size(PotansiyelKopuk);
w=1
while (w<=u)

[PotansiyelKopuk(w,3)]=aciliMinTakip(PotansiyelKopuk(w,1),...
    PotansiyelKopuk(w,2),PotansiyelKopuk(w,3),Iadjust)

w=w+1;
end
% Açısal olarak minimum değer takibi yapılıyor
% elde edilen minimum radius secilecek
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[z pCols]=size(PotansiyelKopuk);
if(PotansiyelKopuk(1,1)~=1)
    z=z+1;
end

rad
end %end rad
%İç içe donguler sonu
```

```
%%%%%%%%%%Köpüklerin Ayrılması%%%%%%%%%%%

[PotansiyelKopuk]=KopukleriAyir(PotansiyelKopuk);

[z pCols]=size(PotansiyelKopuk);

filtresayac=1;

%%%%%%%%%%Kontür Takip%%%%%%%%%%%%%%%%%%
Ilabel = zeros(nRows,nCols);
maxRad=0;
 for x=1:z
    [Icember,feedBackTrigger,PotansiyelKopuk(x,4),PotansiyelKopuk(x,6),...
        Ilabel]=KonturTakipV206(img_input,azalmaKosul,...
        Icember,Iadjust,PotansiyelKopuk(x,1),PotansiyelKopuk(x,2),...
        PotansiyelKopuk(x,3),PotansiyelKopuk(x,7),cizimOrani,minKTr,...
        ciktiDizini,x,Ilabel);
    %%%%%%%%%Filtrelenmis Potansiyel Köpükler%%%%%%%%%%%
    if(feedBackTrigger==1)
        FiltreliPotMerkez(filtresayac,1)=PotansiyelKopuk(x,1);%i
        FiltreliPotMerkez(filtresayac,2)=PotansiyelKopuk(x,2);%j
        FiltreliPotMerkez(filtresayac,3)=PotansiyelKopuk(x,3);%rad
        FiltreliPotMerkez(filtresayac,4)=PotansiyelKopuk(x,4);%cizilenDerec
        FiltreliPotMerkez(filtresayac,5)=PotansiyelKopuk(x,5);
        FiltreliPotMerkez(filtresayac,6)=PotansiyelKopuk(x,6);%toplamDerece
        FiltreliPotMerkez(filtresayac,7)=PotansiyelKopuk(x,7);%rad
        FiltreliPotMerkez(filtresayac,8)=PotansiyelKopuk(x,8);%rad+5
        FiltreliPotMerkez(filtresayac,9)=PotansiyelKopuk(x,9);%label no
        filtresayac=filtresayac+1;

    elseif(feedBackTrigger==2 && PotansiyelKopuk(x,3)>maxRad )
        maxRad=round(PotansiyelKopuk(x,3));

    end
 end
 PotansiyelKopuk=FiltreliPotMerkez;
 [f pCols]=size(FiltreliPotMerkez);

 %%%%%%%%Filtrelenen Köpüklerin Çizimi
 for r=1:f
     FiltreliPotMerkez(r,9)=r;
     [IfiltreCember,feedBackTrigger,FiltreliPotMerkez(r,4),...
     FiltreliPotMerkez(r,6),Ilabel]=KonturTakipV206...
     ('filtre',azalmaKosul,IfiltreCember,Iadjust,FiltreliPotMerkez(r,1),...
     FiltreliPotMerkez(r,2),FiltreliPotMerkez(r,3),...
     FiltreliPotMerkez(r,4),cizimOrani,minKTr,ciktiDizini,r,Ilabel);
 end

%%%%%%%%%Raster to Label Islemleri%%%%%%%%%%%%
  Ilabel=KomsuDoldurma8n(Ilabel);

  ortalamaYuvarlaklik=0;
  ortalamaCap=0;
  renklendirme = label2rgb(Ilabel, @spring, 'c', 'shuffle');
  imwrite(renklendirme,strcat(ciktiDizini,'Label',num2str(azalmaKosul),...
```

```
      '.jpg'),'jpg');
  STATS = regionprops (Ilabel,'all');
  %%%Köpüklere ait Öznitelikler in Çıkarılması
  for i =1:f
    Foam(i,5) = STATS(i).Area;
    Foam(i,4) = STATS(i).Perimeter;
    Foam(i,1) = STATS(i).Centroid(1);
    Foam(i,2) = STATS(i).Centroid(2);
    Foam(i,3) = STATS(i).EquivDiameter;
    Foam(i,6) = (Foam(i,4)*Foam(i,4))/Foam(i,5);

  end
%Yuvarlaklık ve Çapa ait mode ve ortalama değerlerinin hesaplanması
    ortalamaYuvarlaklik=mean(Foam(:,6));
    modeYuvarlaklik=mode(round(Foam(:,6)));
    ortalamaCap=mean(Foam(:,3));
    modeCap=mode(round(Foam(:,3)));

%gaz alani -sivi alaninin hesaplanmasi
  [gazAlani,siviAlani,toplamAlan]=alanlarOrani(Ilabel,maxRad,...
      ciktiDizini,Igray);


  %%%%%%%%%Excel Dosyalarına Yazma İşlemleri%%%%%%%%%
  if (butunOrtalamayaziTrigger==0)
  T=num2cell(Tablo);

  xlswrite( strcat(ciktiDizini,'ButunOrt',int2str(radBas),'-',...
      int2str(radSon),'.xls'),T);
  butunOrtalamayaziTrigger=1;
  end

  baslik={'i degerleri','j degerleri','rad','cizilen Derece',...
      'fark','Toplam Aci','rad ort','rad+5 ort','label'};
  PK=num2cell(PotansiyelKopuk);
  pp = [baslik; PK];
  xlswrite ( strcat(ciktiDizini,'PotMerk',int2str(radBas),'-',...
      int2str(radSon),'.xls'),pp);

  baslik2={'i','j','rad','cizilenDerece','sayac','toplamDerece',...
      'rad','rad+5','label'};
  FT=num2cell(FiltreliPotMerkez);
  filtre = [baslik2; FT];
  xlswrite ( strcat(ciktiDizini,'FiltreliMerk',int2str(radBas),'-',...
      int2str(radSon),'.xls'),filtre);

  baslik3={'j','i','cap','cevre','alan','yuvarlaklik'};
  FoamCell=num2cell(Foam);
  baslikliFoam = [baslik3; FoamCell];
  xlswrite ( strcat(ciktiDizini,'FoamStats',int2str(radBas),'-',...
      int2str(radSon),'.xls'),baslikliFoam);

  oranlar= {'gazAlani','siviAlani','toplamAlan','gazOrani','siviOrani',...
      'gaz/sivi-Orani';gazAlani,siviAlani,toplamAlan,...
      gazAlani/toplamAlan,siviAlani/toplamAlan,gazAlani/siviAlani};
```

```
  xlswrite ( strcat(ciktiDizini,'Gaz-Sivi Oranlari.xls'),oranlar);

  %%%%%%%Text dosyalarina yazma İşlemleri%%%%%%%%
  dlmwrite(strcat(ciktiDizini,'FoamStats.txt'), Foam, 'delimiter', '\t',...
      'precision', '%8.3f', 'newline', 'pc');

  summary=[ortalamaYuvarlaklik,modeYuvarlaklik,ortalamaCap,modeCap];
  dlmwrite(strcat(ciktiDizini,'Ozet.txt'), summary, 'delimiter', '\t',...
      'precision', '%8.3f', 'newline', 'pc');

function [Ipot]=MerkezUzaklikBulV206(PotKopuk,Igray,Ipot,ciktiDizini)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Euclid Mesafesinin Çıkarımını sağlayan metot, bütün köpük alanları beyaza
%set edilmektedir
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[nRows nCols]=size(Igray);
[pRows pCols]=size(PotKopuk);

yaricapHassasiyet=1;

for i=1:nRows
    for j=1:nCols


        for k=1:pRows


           euclidDist=sqrt((i-PotKopuk(k,1))^2+(j-PotKopuk(k,2))^2);
           if(euclidDist<=PotKopuk(k,3)*yaricapHassasiyet)
               Ipot(i,j)=255;

           end



        end

    end
end
imwrite (Ipot,strcat(ciktiDizini,'EuclidDist.bmp'),'bmp');


function [trigger]=dahaYerVarMi(Idist,nRows,nCols)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
% Görüntü matrisi üzerinde her hangi bir köpüğün sığabileceği yerin olup
% olmadığı kontrolünü yapan metot.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:nRows
    for j=1:nCols
```

```
            if(Idist(i,j)>=10)
                trigger=1;
                return;
             elseif (Idist(i,j)<10)
                trigger=0;
            end
    end
end

function [C]=AgirlikMerkeziHesapla(PotOrigin)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci birbirine cok yakin bulunan merkezleri tek bir
%merkezde toplamak(ortalama) ve ustuste binmis kopuklerin olusmasini
%engellemektir. Her radiusa gore taramadan sonra bu
%kontrol yapilarak yanlis kopuk sonuclarinin cikarilmasi engellenir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[nRows nCols]=size(PotOrigin);

%Agirlik Merkezinin hesaplandığı matris
C(1,1)=PotOrigin(1,1);%i
C(1,2)=PotOrigin(1,2);%j
C(1,3)=PotOrigin(1,3);%rad
C(1,4)=PotOrigin(1,4);%
C(1,5)=PotOrigin(1,5);%sayac
C(1,6)=PotOrigin(1,6);
C(1,7)=PotOrigin(1,7);
C(1,8)=PotOrigin(1,8);
C(1,9)=PotOrigin(1,9);

%bulundu=0;
for z=2:nRows
    k=1;
    bulundu=0;
    [kRows kCols]=size(C);
    %bütün köpüklerin bulunulan köpükle ortak bileşik olup olmadığı kontrol
    %edilmekte
    while(k<=kRows)

          if ((sqrt(abs((PotOrigin(z,1)-C(k,1)))^2+abs((PotOrigin(z,2)...
                  -C(k,2)))^2) < (C(k,3)/2))  )

             C(k,1)=((C(k,1)*C(k,5))+PotOrigin(z,1))/(C(k,5)+1);
             C(k,2)=((C(k,2)*C(k,5))+PotOrigin(z,2))/(C(k,5)+1);
             C(k,3)=((C(k,3)*C(k,5))+PotOrigin(z,3))/(C(k,5)+1);
             C(k,7)=((C(k,7)*C(k,5))+PotOrigin(z,7))/(C(k,5)+1);
             C(k,8)=((C(k,8)*C(k,5))+PotOrigin(z,8))/(C(k,5)+1);
             C(k,9)=0;
             % her bulunan ortak köpük merkezi için ağırlık katsayısı
             % artırılmakta
             C(k,5)=C(k,5)+1;

            bulundu=1;
            break;
```

```
        end
    k=k+1;

    end
    %herhangi bir köpükle bileşik olmayan köpüklerin matriste set edilmesi
    if(bulundu==0)

     C(kRows+1,1)=PotOrigin(z,1);
     C(kRows+1,2)=PotOrigin(z,2);
     C(kRows+1,3)=PotOrigin(z,3);
     C(kRows+1,4)=PotOrigin(z,4);
     C(kRows+1,5)=PotOrigin(z,5);
     C(kRows+1,6)=PotOrigin(z,6);
     C(kRows+1,7)=PotOrigin(z,7);
     C(kRows+1,8)=PotOrigin(z,8);
     C(kRows+1,9)=PotOrigin(z,9);
     PotOrigin(z,5)=kRows+1;

    end

end


function [minRadOrtalama]=aciliMinTakip(i,j,yaricap,Iadjust)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci, orijini belli köpüğün tam sınırlarını hassas olarak
%ortaya çıkarabilmek için açısal minimum değerler üzerinden ortalama
%yarıçapı hesaplamak
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
taramaHassasiyet=0.1;
%radiusa uygun minimum tarama yapilirken secilecek hassasiyet
radBas=yaricap-round(yaricap*taramaHassasiyet);
%ve secilen bas son rad degerleri
radSon=yaricap+round(yaricap*taramaHassasiyet);

[nRows nCols]=size(Iadjust);
%resim sinirlari icinde olup olmadigi takibi icin gerekli
radToplam=0;
%elde edilen min yaricaplar toplamini tutan deger
radSayac=0;
%radToplami bolerek ortalama minimum rad degerini cıkaracaktir

%%22.5 derecelik açılarla
for degree=0:22.5:337.5
%0
 idegeri=round(i-sin(degree*pi/180)*yaricap);
 jdegeri=round(j+cos(degree*pi/180)*yaricap);

if(idegeri>=1 && idegeri<=nRows && jdegeri>=1 && jdegeri<=nCols )
 radSayac=radSayac+1;
 min=Iadjust(idegeri,jdegeri);
 minRad=yaricap;

 %belirli bir yarıçap aralığında
```

```
 for rad=radBas:1:radSon

    idegeri=round(i-sin(degree*pi/180)*rad);
    jdegeri=round(j+cos(degree*pi/180)*rad);
    if(idegeri>=1 && idegeri<=nRows && jdegeri>=1 && jdegeri<=nCols )
    %minimum kontrolü
    if(Iadjust(idegeri,jdegeri)<min)
        min=Iadjust(idegeri,jdegeri);
        minRad=rad;

    end %end if (Iadjust(ideger...
    end%end if (idegeri>=1...

 end%end for rad
else
minRad=0;

% minRad

end
%minimum değerli açılara sahip yarıçaplar toplanmakta
radToplam=minRad+radToplam;
end %end for degree

%minimum değerli açılara sahip yarıçapların ortalaması alınmakta
minRadOrtalama=radToplam/radSayac;

function ayrilmisPot=KopukleriAyir(Pot)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci, iki köpüğün sınırlardan birbirine girişimi durumunda
%küçük yarıçapa sahip olanı filtrelemektir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[z pCols]=size(Pot);
trigger=1;
ayrilmisPot(1,1)=Pot(1,1);
ayrilmisPot(1,2)=Pot(1,2);
ayrilmisPot(1,3)=Pot(1,3);
ayrilmisPot(1,4)=Pot(1,4);
ayrilmisPot(1,5)=Pot(1,5);
ayrilmisPot(1,6)=Pot(1,6);
ayrilmisPot(1,7)=Pot(1,7);
ayrilmisPot(1,8)=Pot(1,8);
ayrilmisPot(1,9)=Pot(1,9);
a=1;
x=1;

while(x<=z)
    y=x+1;
    while(y<=z)

            %sınırlardan girişim var mı?
           if ( sqrt((Pot(x,1)-Pot(y,1))^2+ (Pot(x,2)-
Pot(y,2))^2)<=0.8*(Pot(y,3)+Pot(x,3)))
```

```
                %büyük olanı tercih et
                if(Pot(x,3)>=Pot(y,3))
                     %küçük olan filtrelenir
                     Pot(y,:) = [];
                     [z pCols]=size(Pot);


                     g=0;
                elseif(Pot(x,3)<Pot(y,3))
                    trigger=0;
                end


            end %end if
    y=y+1;
    end %end while

    if (trigger==1)
        ayrilmisPot(a,1)=Pot(x,1);
        ayrilmisPot(a,2)=Pot(x,2);
        ayrilmisPot(a,3)=Pot(x,3);
        ayrilmisPot(a,4)=Pot(x,4);
        ayrilmisPot(a,5)=Pot(x,5);
        ayrilmisPot(a,6)=Pot(x,6);
        ayrilmisPot(a,7)=Pot(x,7);
        ayrilmisPot(a,8)=Pot(x,8);
        ayrilmisPot(a,9)=Pot(x,9);
        a=a+1;

    end

    trigger=1;
 x=x+1;
end  %end while

function [Icember,feedBackTrigger,cizilenDerece,toplamAci,Ilabel] = ...
    KonturTakipV206(imgName,azalmaKosul,Icember,Iadjust,iMerkez,...
    jMerkez,yaricap,ortalama,cizimOrani,minKTr,ciktiDizini,w,Ilabel)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci, orijini ve ortalama yarıçapı belli köpüğün sınır
%takibini 22.5 derecelik açılarla pozitif ve negatif yönlerde
%gerçeklemektir
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
w;%kopuge ait label
cizilenDerece=0;
toplamAci=0;
geciciAciAraligi1=0;
geciciAciAraligi2=0;
[nRows nCols]=size(Iadjust);
x=1;
acisayac=1;
degCtr=360/(yaricap*10);

if(iMerkez>=1 && iMerkez<=nRows && jMerkez>=1 && jMerkez<=nCols )
```

```
Icember(round(iMerkez),round(jMerkez))=0;
else
feedBackTrigger=2;
return;
end
%Açısal sınır taşma kontrolü
for degree=0:22.5:337.5
iKontrol=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
jKontrol=round(jMerkez+cos(degree*pi/180)*(yaricap+1));
if(iKontrol<1 || iKontrol>nRows || jKontrol<1 || jKontrol>nCols)
feedBackTrigger=2;
return;
end
end
%45 derecelik açılarla 22,5pozitif-22.5 negatif yönde olmak üzere sınır
%takibi gerçeklenmekte
for degree=0:45:315

        [trigger1,geciciAciAraligi1,cizilenSayac1,Icember,Ilabel]...
            =degreePosTracingCokluKarar(Iadjust,Icember,iMerkez,jMerkez,...
            yaricap,ortalama,degCtr,w,Ilabel,degree);
        [trigger2,geciciAciAraligi2,cizilenSayac2,Icember,Ilabel]=...
            degreeNegTracingCokluKarar(Iadjust,Icember,iMerkez,jMerkez,...
            yaricap,ortalama,degCtr,w,Ilabel,degree);
        cizilenDerece=cizilenDerece+cizilenSayac1+cizilenSayac2;
        toplamAci=toplamAci+geciciAciAraligi1+geciciAciAraligi2;
        if(trigger1==0 || trigger2==0)
           feedBackTrigger=2;
           return;
        end
 end

%girişte alınan çizim oranı değerinin altında kaldığı zaman kabarcık sözde
%köpük olarak ele alınmakta ve filtrelenmektedir.
if ( (cizilenDerece/toplamAci)>=cizimOrani || yaricap>=minKTr)
feedBackTrigger=1;
else
feedBackTrigger=0;
end

%Sınır takipleri sonrası köpük görüntülerinin çıkarımı
Icember=uint8(Icember);
[Ibirles]=birlestir (Iadjust,Icember);
imwrite(Ibirles,strcat(ciktiDizini,imgName,'-Birlesik Ak-',...
    num2str(azalmaKosul),'.jpg'),'jpg');
imwrite(Icember,strcat(ciktiDizini,imgName,'-Bulunan Ak-',...
    num2str(azalmaKosul),'.bmp'),'bmp');


function [trigger,geciciAciAraligi1, cizilenSayac,Icember,Ilabel]=...
    degreePosTracingCokluKarar(Iadjust,Icember,iMerkez,jMerkez,...
    yaricap,ortalama,degCtr,w,Ilabel,degree)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amacı, polar koordinatlar yardımıyla pozitif yönde sınır
```

```
%takibi yapmak ve çizilebilen oranı geri iletmektir. Bu sınır takibi
%esnasında minimum aralığa göre yönelim gerçeklenmekte ve kabarcığın
%biçimsel eğrileri ortaya çıkarılmaktadır.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
yaricapIlk=yaricap;
aciHassasiyet=0.2;
trigger=1;
%gelinen noktanin resim boyutlarini kontrol trigger
cizilenSayac=0;
%baslangic noktalari merkezden radius kadar uzaklikta belirleniyor
iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
jStart=round(jMerkez+cos(degree*pi/180)*yaricap);
donusSayisi=floor(22.5/degCtr);
sayac=1;%kac derece ilerledigini tutan sayac
[nRows nCols]=size(Iadjust);

if(iStart<1 || iStart>nRows || jStart<1 || jStart>nCols )
%baslangic noktasi resim boyutlarindan disarida ise return ediliyor
geciciAciAraligi1=0;
trigger=0;
return;
else
geciciAciAraligi1=degCtr*donusSayisi;
end

Icember(iStart,jStart)=0;
iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));

iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));

if (iradMinus>1 && iradMinus<nRows && jradMinus>1 && ...
        jradMinus<nCols && iradPlus>1 && iradPlus<nRows && jradPlus>1 ...
        && jradPlus<nCols)
Icember(iradMinus,jradMinus)=0;
Icember(iradPlus,jradPlus)=0;
end

karar=5;%karar verebilmek icin kac piksel ileriye bakacagimiz
%gosteren menzil


while( sayac<=donusSayisi && trigger==1)

  for r=yaricap:-1:0
     ilabel=round(iMerkez-sin(degree*pi/180)*(r));
     jlabel=round(jMerkez+cos(degree*pi/180)*(r));
     Ilabel(ilabel,jlabel)=w;
  end

 if(Iadjust(iStart,jStart)<=ortalama)
    if( cizilenSayac+degCtr<=22.5)
    cizilenSayac=cizilenSayac+degCtr;
    end
```

```
%Karar meknizmasi icin bulunan ortalamalar
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%r=yaricap icin bulunan ortalama
normalRadToplam=0;
posRadToplam=0;
negRadToplam=0;


eskiDegree=degree;

for x=1:karar
    iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
    jStart=round(jMerkez+cos(degree*pi/180)*yaricap);
    if(iStart>=1 && iStart<=nRows && jStart>=1 && jStart<=nCols )

    normalRadToplam(x)=Iadjust(iStart,jStart);
    end %en if
    degree=degree+degCtr;
end %end for

normalRadOrt=mean(normalRadToplam);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%r=yaricap-1 icin bulunan ortalama
degree=eskiDegree;

for x=1:karar
    iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
    jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));
    if(iradMinus>=1 && iradMinus<=nRows && jradMinus>=1 && ...
            jradMinus<=nCols )

    negRadToplam(x)=Iadjust(iradMinus,jradMinus);
    end %end if
    degree=degree+degCtr;
end%end for
negRadOrt=mean(negRadToplam);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%r=yaricap+1 icin bulunan ortalama
degree=eskiDegree;

for x=1:karar
    iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
    jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));
     if(iradPlus>=1 && iradPlus<=nRows && jradPlus>=1 && ...
             jradPlus<=nCols )

    posRadToplam(x)=Iadjust(iradPlus,jradPlus);
     end%end if
    degree=degree+degCtr;
end%end for
posRadOrt=mean(posRadToplam);
```

```
      degree=eskiDegree; %aci ilk haline getiriliyor

      %sirayla r=yaricap,r=yaricap+1,r=yaricap-1 icin koordinatlar
      %hesaplaniyor
      degree=degree+degCtr;
      iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
      jStart=round(jMerkez+cos(degree*pi/180)*yaricap);

      iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
      jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));

      iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
      jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));

      %herhangi bir boyut tasma durumunda metod duruyor
      if (iStart<1 || iStart>nRows || jStart<1 || jStart>nCols || ...
              iradMinus<1 || iradMinus>nRows || jradMinus<1 || ...
              jradMinus>nCols || iradPlus<1 || iradPlus>nRows ||...
              jradPlus<1 || jradPlus>nCols)
       trigger=0;
      return;
      end%end if

      %cizgi 3 piksel kalinliginda ciziliyor
      Icember(iStart,jStart)=0;
      Icember(iradMinus,jradMinus)=0;
      Icember(iradPlus,jradPlus)=0;

      %minimum olan ortalamaya gore yeni koordinat seciliyor
      if(normalRadOrt<=negRadOrt && normalRadOrt<=posRadOrt )
         %yaricap=yaricap;
      elseif(negRadOrt<=normalRadOrt && negRadOrt<=posRadOrt && ...
              ((yaricap-1)>=yaricapIlk-(aciHassasiyet*yaricapIlk)) )
         yaricap=yaricap-1;
         iStart=iradMinus;
         jStart=jradMinus;
      elseif(posRadOrt<=normalRadOrt && posRadOrt<=negRadOrt && ...
              ((yaricap+1)<=yaricapIlk+(aciHassasiyet*yaricapIlk)))
         yaricap=yaricap+1;
         iStart=iradPlus;
         jStart=jradPlus;
      end %end if(normalRadOrt<=neg

else
        %sirayla r=yaricap,r=yaricap+1,r=yaricap-1 icin koordinatlar
     %hesaplaniyor
      degree=degree+degCtr;
      iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
      jStart=round(jMerkez+cos(degree*pi/180)*yaricap);

      iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
      jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));

      iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
      jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));
```

```
      %herhangi bir boyut tasma durumunda metod duruyor
       if (iStart<1 || iStart>nRows || jStart<1 || jStart>nCols ...
              || iradMinus<1 || iradMinus>nRows || jradMinus<1 || ...
              jradMinus>nCols || iradPlus<1 || iradPlus>nRows || ...
              jradPlus<1 || jradPlus>nCols)
       trigger=0;
      return;
      end %end if

      %cizgi 3 piksel kalinliginda ciziliyor
      Icember(iStart,jStart)=0;
      Icember(iradMinus,jradMinus)=0;
      Icember(iradPlus,jradPlus)=0;


 end %end


      %ilerlenen her koordinatta derece sayaci önceden belirlenen arttirici
     %kadar arttırılıyor.
sayac=sayac+1 ;

end

 function [trigger,geciciAciAraligi2, cizilenSayac,Icember,Ilabel]=...
    degreeNegTracingCokluKarar(Iadjust,Icember,iMerkez,jMerkez,...
    yaricap,ortalama,degCtr,w,Ilabel,degree)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amacı, polar koordinatlar yardımıyla negatif yönde sınır
%takibi yapmak ve çizilebilen oranı geri iletmektir. Bu sınır takibi
%esnasında minimum aralığa göre yönelim gerçeklenmekte ve kabarcığın
%biçimsel eğrileri ortaya çıkarılmaktadır.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
yaricapIlk=yaricap;
aciHassasiyet=0.2;
trigger=1;
%sinirlari tasip tasmadigini kontroleden trigger
cizilenSayac=0;
%baslangic noktalari merkezden radius kadar uzaklikta belirleniyor
iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
jStart=round(jMerkez+cos(degree*pi/180)*yaricap);

donusSayisi=floor(22.5/degCtr);
sayac=1;%kac derece ilerledigini tutan sayac
[nRows nCols]=size(Iadjust);

if(iStart<1 || iStart>nRows || jStart<1 || jStart>nCols )
%baslangic noktasi resim boyutlarindan disarida ise return ediliyor
geciciAciAraligi2=0;
trigger=0;
return;
else
geciciAciAraligi2=degCtr*donusSayisi;
```

```
end

Icember(iStart,jStart)=0;
iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));

iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));

if (iradMinus>1 && iradMinus<nRows && jradMinus>1 && ...
        jradMinus<nCols && iradPlus>1 && iradPlus<nRows && jradPlus>1 ...
        && jradPlus<nCols)
Icember(iradMinus,jradMinus)=0;
Icember(iradPlus,jradPlus)=0;
end

karar=5;%karar verebilmek icin kac piksel ileriye bakacagimiz
%gosteren menzil

while(sayac<=donusSayisi && trigger==1)

   for r=yaricap:-1:0
     ilabel=round(iMerkez-sin(degree*pi/180)*(r));
     jlabel=round(jMerkez+cos(degree*pi/180)*(r));
     Ilabel(ilabel,jlabel)=w;
   end

   if(Iadjust(iStart,jStart)<=ortalama)
    if( cizilenSayac+degCtr<=22.5)
       cizilenSayac=cizilenSayac+degCtr;
    end
       %Karar meknizmasi icin bulunan ortalamalar

    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %r=yaricap icin bulunan ortalama
    normalRadToplam=0;
    posRadToplam=0;
    negRadToplam=0;


    eskiDegree=degree;
    for x=1:karar

        iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
        jStart=round(jMerkez+cos(degree*pi/180)*yaricap);
        if(iStart>=1 && iStart<=nRows && jStart>=1 && jStart<=nCols )

        normalRadToplam(x)=Iadjust(iStart,jStart);
        end
        degree=degree-degCtr;
    end

     normalRadOrt=mean(normalRadToplam);
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    %r=yaricap-1 icin bulunan ortalama
```

```
degree=eskiDegree;

for x=1:karar
    iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
    jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));
    if(iradMinus>=1 && iradMinus<=nRows && jradMinus>=1 && ...
            jradMinus<=nCols )

    negRadToplam(x)=Iadjust(iradMinus,jradMinus);
    end
    degree=degree-degCtr;
end
negRadOrt=mean(negRadToplam);


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%r=yaricap+1 icin bulunan ortalama
degree=eskiDegree;

for x=1:karar
    iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
    jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));
    if(iradPlus>=1 && iradPlus<=nRows && jradPlus>=1 && ...
            jradPlus<=nCols )

    posRadToplam(x)=Iadjust(iradPlus,jradPlus);
    end
    degree=degree-degCtr;
end
posRadOrt=mean(posRadToplam);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

 degree=eskiDegree; %aci ilk haline getiriliyor

 %sirayla r=yaricap,r=yaricap+1,r=yaricap-1 icin koordinatlar
 %hesaplaniyor
 degree=degree-degCtr;
 iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
 jStart=round(jMerkez+cos(degree*pi/180)*yaricap);

 iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
 jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));

 iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
 jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));


 %herhangi bir boyut tasma durumunda metod duruyor
 if (iStart<1 || iStart>nRows || jStart<1 || jStart>nCols ||...
         iradMinus<1 || iradMinus>nRows || jradMinus<1 ||...
         jradMinus>nCols || iradPlus<1 || iradPlus>nRows ||...
         jradPlus<1 || jradPlus>nCols)
  trigger=0;
 return;
```

```
  end

  %cizgi 3 piksel kalinliginda ciziliyor
   Icember(iStart,jStart)=0;
   Icember(iradMinus,jradMinus)=0;
   Icember(iradPlus,jradPlus)=0;

%minimum olan ortalamaya gore yeni koordinat seciliyor
 if(normalRadOrt<=negRadOrt && normalRadOrt<=posRadOrt)
%yaricap=yaricap;
 elseif(negRadOrt<=normalRadOrt && negRadOrt<=posRadOrt && ...
         ((yaricap-1)>=yaricapIlk-(aciHassasiyet*yaricapIlk)) )
          yaricap=yaricap-1;
      iStart=iradMinus;
      jStart=jradMinus;
  elseif(posRadOrt<=normalRadOrt && posRadOrt<=negRadOrt && ...
          ((yaricap+1)<=yaricapIlk+(aciHassasiyet*yaricapIlk)))
       yaricap=yaricap+1;
      iStart=iradPlus;
      jStart=jradPlus;
 end

else
  %sirayla r=yaricap,r=yaricap+1,r=yaricap-1 icin koordinatlar
  %hesaplaniyor
  degree=degree-degCtr;
  iStart=round(iMerkez-sin(degree*pi/180)*yaricap);
  jStart=round(jMerkez+cos(degree*pi/180)*yaricap);

  iradMinus=round(iMerkez-sin(degree*pi/180)*(yaricap-1));
  jradMinus=round(jMerkez+cos(degree*pi/180)*(yaricap-1));

  iradPlus=round(iMerkez-sin(degree*pi/180)*(yaricap+1));
  jradPlus=round(jMerkez+cos(degree*pi/180)*(yaricap+1));


  %herhangi bir boyut tasma durumunda metod duruyor
  if (iStart<1 || iStart>nRows || jStart<1 || jStart>nCols ...
          || iradMinus<1 || iradMinus>nRows || jradMinus<1 ||...
          jradMinus>nCols || iradPlus<1 || iradPlus>nRows || ...
          jradPlus<1 || jradPlus>nCols)
   trigger=0;
  return;
  end

  %cizgi 3 piksel kalinliginda ciziliyor
   Icember(iStart,jStart)=0;
   Icember(iradMinus,jradMinus)=0;
   Icember(iradPlus,jradPlus)=0;


end


  %ilerlenen her koordinatta derece sayaci önceden belirlenen arttirici
```

```
      %kadar arttırılıyor.
      sayac=sayac+1;

   % end
end
```

Ek E. Geliştilen Kod – İkinci Model

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Kodlayan:Veli Mert Altas-Bilgisayar Muhendisligi-Yuksek Lisans Ogr.
% Danışman:Dr. A. Murat Özbayoğlu
% TOBB Ekonomi ve Teknoloji Universitesi,Ankara-Turkiye,2007.
%%%% email:mertaltas@gmail.com, maltas@etu.edu.tr%%%%
% Coder:Veli Mert Altas-Computer Science-Grad Student
% Instructor:Dr. A. Murat Özbayoğlu
% TOBB University of Economy and Technology ,Ankara-Turkiye,2007.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [EksenlerMatrisi]=OrientV104(dizin, dosyaIsmi,uzanti)

%%%%Dosyanin okunmasi%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
dosya=strcat(dizin,dosyaIsmi,'.',uzanti);

%Orn: dosya='D:\Matlab\Arastirma\10 min.jpg' seklinde birlestiriliyor.
ciktiDizini= strcat(dizin,dosyaIsmi,'Test Sonuclari','\shift','\');
mkdir(ciktiDizini);

%%%%Giris timestamp%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ts=1;
tBaslik={'yil','ay','gun','saat','dakika','saniye','cputime'};

timestamp(ts,1:6)=clock;
timestamp(ts,7)=cputime;%cpu zamani duruma gore olculebilir

tStamp=num2cell(timestamp);
timexls = [tBaslik; tStamp]
xlswrite(strcat(ciktiDizini,'Time Stamps.xls'),timexls);
ts=ts+1;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%Gri Seviye ve adaptif threshold donusumu%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Irgb=imread(dosya);
Igray=rgb2gray(Irgb);
claheI = adapthisteq (Igray);
adjust = imadjust(claheI);

%%%%Parametrelerin Atanması%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
adjust=medfilt2(adjust,[3 3]);%median filtre uygulanıyor
cizimOrani=0.5;%Kontur taramada aranan cizim orani
minKTr=20;%min kopuk tarama orani Orn:'20den buyuklerde arama'
dinamikKosul=0.4;%her eksen sekizlisinde en buyuk 3 ortalamanın en kucuk...
                 %olana gore oranlanması icin aranan kosul. Sınırlardaki...
                 %eksenlerin cıkartılması icin gerekli
[nR nC nZ]=size(Irgb);
EksenlerMatrisi=0;%Eksenleri tutan matris
eksenSayaci=0;   %dinamik kosulun guncellenmesi icin eksenler sayaci

for i=1:nR
    for j=1:nC
         HangiEksen(i,j)=0;  %her pikselin hangi tip eksene dahil oldugunu
                             %tutan matris
```

```
            Icember(i,j)=255;%Siyah Beyaz sınırların cizildigi 2D
                             %goruntu matrisi (beyaz olarak ataniyor)
            SinirlarinGoruntusu(i,j)=255;
            for z=1:nZ
            Goruntu(i,j,z)=255;%Cizimlerin yapildigi 3D goruntu matrisi
                             %(beyaz olarak ataniyor)
        end
    end
end

adjust16=uint16(adjust); %ilerde uint16 toplami icin gerekli cevrimi
pRow=floor(nR/3);%3er 3er piksel ilerlemek icin toplam i'nin 3te biri
                 %Orn:396/3=132
pCol=floor(nC/3);%3er 3er piksel ilerlemek icin toplam j'nin 3te biri
                 %Orn:576/3=192

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[eksenSayaci,HangiEksen,EksenlerMatrisi,Goruntu]=EksenHesaplaV103(...
    HangiEksen,pRow,pCol,EksenlerMatrisi,Goruntu,adjust16,dinamikKosul,...
    eksenSayaci);
imwrite (Goruntu,strcat(ciktiDizini,'oryantasyon1.bmp'),'bmp');



xlswrite(strcat(ciktiDizini,'CizgilerFiltresiz.xls'),EksenlerMatrisi);

%%%%İkili Şablonlara gore Filtreleme%%%%%%%%%%%%%%%%%%%%%%%%%%
FiltreliEksenler=EksenFiltrelemeV103(EksenlerMatrisi);
xlswrite(strcat(ciktiDizini,'CizgilerFiltreli1.xls'),FiltreliEksenler);

for i=1:nR
    for j=1:nC

        HangiEksen(i,j)=0;%her pikselin hangi tip eksene dahil oldugunu
                          %tutan matris
        for z=1:nZ
        Goruntu(i,j,z)=255;%filtrelenmis haline gore tekrar cizim
                         %yapabilmek icin goruntu matrisi beyaza cevriliyor
        end
    end
end
[fR fC]=size(FiltreliEksenler);
dereceEkseni=[1,2,3,4,5,6,7,8];
for k=0:1:fR-3
    for m=0:1:fC-3
         [HangiEksen,EksenlerMatrisi,Goruntu]=EksenCizimiV103(...
             FiltreliEksenler,k,m,Goruntu,FiltreliEksenler(k+1,m+1),...
             dereceEkseni,HangiEksen);
         %dikkat EksenlerMatrisinin uzerine yazılıyor,
    end
end
%%%%%%%Merkez-rad Cikarimi%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

imwrite (Goruntu,strcat(ciktiDizini,'oryantasyonFiltreli.bmp'),'bmp');
```

```
Y=Goruntu;
U=Y;
[PotMerk,Goruntu]=MerkezBulmaV103(FiltreliEksenler,Goruntu,HangiEksen,...
    adjust,ciktiDizini);
%%%%%%%Agirlik Merkezi Bulma%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
PotMerk=AgirlikMerkeziHesapla(PotMerk);
xlswrite(strcat(ciktiDizini,'PotMerk.xls'),PotMerk);

[z pCols]=size(PotMerk);
for x=1:z
    Y(round(PotMerk(x,1)),round(PotMerk(x,2)),:)=[0 0 0];
    Y(round(PotMerk(x,1)+1),round(PotMerk(x,2)),:)=[0 0 0];
    Y(round(PotMerk(x,1)-1),round(PotMerk(x,2)),:)=[0 0 0];
    Y(round(PotMerk(x,1)),round(PotMerk(x,2)-1),:)=[0 0 0];
    Y(round(PotMerk(x,1)),round(PotMerk(x,2)+1),:)=[0 0 0];
end
imwrite (Y,strcat(ciktiDizini,'AgirlikMerkezleri.bmp'),'bmp');
%%%%%%%%%%Merkezlerin Kontrolu%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[PotMerkDogrular,PotMerk]=MerkezKontrol(PotMerk,U,HangiEksen);
xlswrite(strcat(ciktiDizini,'MerkezKontrolSonrasi1.xls'),PotMerk);
xlswrite(strcat(ciktiDizini,'YaricapDogrulari1.xls'),PotMerkDogrular);


%%%%%%%%%%Kopuklerin Ayrilmasi%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[PotMerkDogrular,IPAciArtimi]=YaricapInterPolasyon(PotMerkDogrular)


[pR pC]=size (PotMerk)
[Ir Ic]=size(IPAciArtimi)
[Mr Mc]=size(PotMerkDogrular)

[PotMerk,PotMerkDogrular,StSapma,IPAciArtimi]=KopukleriAyirStDev...
    (PotMerk,PotMerkDogrular,IPAciArtimi);

xlswrite(strcat(ciktiDizini,'MerkezKontrolSonrasi2.xls'),PotMerk);
xlswrite(strcat(ciktiDizini,'YaricapDogrulari2.xls'),PotMerkDogrular);
[pR pC]=size (PotMerk)
[Ir Ic]=size(IPAciArtimi)
[Mr Mc]=size(PotMerkDogrular)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[z pCols]=size(PotMerk);
for x=1:z
    U(round(PotMerk(x,1)),round(PotMerk(x,2)),:)=[0 0 0];
    U(round(PotMerk(x,1)+1),round(PotMerk(x,2)),:)=[0 0 0];
    U(round(PotMerk(x,1)-1),round(PotMerk(x,2)),:)=[0 0 0];
    U(round(PotMerk(x,1)),round(PotMerk(x,2)-1),:)=[0 0 0];
    U(round(PotMerk(x,1)),round(PotMerk(x,2)+1),:)=[0 0 0];
end
imwrite
(U,strcat(ciktiDizini,'AgirlikMerkezleriKontrolSonrasi.bmp'),'bmp');


%%%%%%%%%%Sınır Tamamlama%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[SinirlarinGoruntusu]=SinirTamamlama(PotMerkDogrular,PotMerk,...
```

```
    SinirlarinGoruntusu,IPAciArtimi);
imwrite (SinirlarinGoruntusu,strcat(ciktiDizini,'Sinirlar.bmp'),'bmp');



%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
timestamp(ts,1:6)=clock;
timestamp(ts,7)=cputime;%cpu zamani duruma gore olculebilir

tStamp=num2cell(timestamp);
timexls = [tBaslik; tStamp]
xlswrite(strcat(ciktiDizini,'Time Stamps.xls'),timexls);
ts=ts+1;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

imwrite (adjust,strcat(ciktiDizini,'adaptifresim.jpg'),'jpg');
imwrite (Goruntu,strcat(ciktiDizini,'oryantasyonMerkezli.bmp'),'bmp');



function [eksenSayaci,R,C,X]=EksenHesaplaV103(R,pRow,pCol,C,X,J,...
    dinamikKosul,eksenSayaci)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amacı, Görüntü üzerinden eksenlerin çıkarılmasıdır. 3 piksel
%aralikla eksenleri hesaplayarak sinirlar uzerinde olanlari tespit etmek ve
% aralarindan minimum olani bastirmaktir
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

axis=ones(1,8);

top=uint16(0);
for k=0:1:pRow-3
    for m=0:1:pCol-3

          d=1;

         %d=1    0-180 derece
          top=uint16(J(((k*3)+5),((m*3)+9))+J(((k*3)+5),((m*3)+8))+...
              J(((k*3)+5),((m*3)+7))+J(((k*3)+5),((m*3)+6))+...
              J(((k*3)+5),((m*3)+5))+J(((k*3)+5),((m*3)+4))+...
              J(((k*3)+5),((m*3)+3))+J(((k*3)+5),((m*3)+2))+...
              J(((k*3)+5),((m*3)+1)));

%            C(k+1,m+1,d)=top/9;
          axis(d)=top/9;

          d=d+1;

          %d=2    22.5-202.5 derece

            top=uint16(J(((k*3)+3),((m*3)+9))+ J(((k*3)+4),((m*3)+7))+...
                J(((k*3)+5),((m*3)+5))+J(((k*3)+6),((m*3)+3))+...
```

```
        J(((k*3)+7),((m*3)+1))...
         +((J(((k*3)+3),((m*3)+8))+J(((k*3)+4),((m*3)+8)))/2)...
         +((J(((k*3)+4),((m*3)+6))+J(((k*3)+5),((m*3)+6)))/2)...
         +((J(((k*3)+5),((m*3)+4))+J(((k*3)+6),((m*3)+4)))/2)...
         +((J(((k*3)+6),((m*3)+2))+J(((k*3)+7),((m*3)+2)))/2));


%           C(k+1,m+1,d)=top/9;
        axis(d)=top/9;
        d=d+1;

        %d=3    45-225 derece
          top=J(((k*3)+1),((m*3)+9))+J(((k*3)+2),((m*3)+8))+...
              J(((k*3)+3),((m*3)+7))+J(((k*3)+4),((m*3)+6))+...
              J(((k*3)+5),((m*3)+5))+J(((k*3)+6),((m*3)+4))+...
              J(((k*3)+7),((m*3)+3))+J(((k*3)+8),((m*3)+2))+...
              J(((k*3)+9),((m*3)+1));
%             C(k+1,m+1,d)=top/9;
          axis(d)=top/9;
          d=d+1;

        %d=4     67.5-247.5 derece
          top=J(((k*3)+1),((m*3)+7))+ J(((k*3)+3),((m*3)+6))+...
              J(((k*3)+5),((m*3)+5))+J(((k*3)+7),((m*3)+4))+...
              J(((k*3)+9),((m*3))+3)+...
              ((J(((k*3)+2),((m*3)+6))+J(((k*3)+2),((m*3)+7)))/2)+...
              ((J(((k*3)+4),((m*3)+5))+J(((k*3)+4),((m*3)+6)))/2)+...
              ((J(((k*3)+6),((m*3)+4))+J(((k*3)+6),((m*3)+5)))/2)+...
              ((J(((k*3)+8),((m*3)+3))+J(((k*3)+8),((m*3)+4)))/2);
%             C(k+1,m+1,d)=top/9;
          axis(d)=top/9;
          d=d+1;

        %d=5     90-360 derece
           top=J(((k*3)+9),((m*3)+5))+J(((k*3)+8),((m*3)+5))+...
               J(((k*3)+7),((m*3)+5))+J(((k*3)+6),((m*3)+5))+...
               J(((k*3)+5),((m*3)+5))+J(((k*3)+4),((m*3)+5))+...
               J(((k*3)+3),((m*3)+5))+J(((k*3)+2),((m*3)+5))+...
               J(((k*3)+1),((m*3)+5));
%              C(k+1,m+1,d)=top/9;
           axis(d)=top/9;
           d=d+1;

         %d=6    112.5-292.5 derece
            top=J(((k*3)+1),((m*3)+3))+ J(((k*3)+3),((m*3)+4))+...
                J(((k*3)+5),((m*3)+5))+J(((k*3)+7),((m*3)+6))+...
                J(((k*3)+9),((m*3))+7)+...
                ((J(((k*3)+2),((m*3)+4))+J(((k*3)+2),((m*3)+3)))/2)+...
                ((J(((k*3)+4),((m*3)+5))+J(((k*3)+4),((m*3)+4)))/2)+...
                ((J(((k*3)+6),((m*3)+6))+J(((k*3)+6),((m*3)+5)))/2)+...
                ((J(((k*3)+8),((m*3)+7))+J(((k*3)+8),((m*3)+6)))/2);
%               C(k+1,m+1,d)=top/9;
            axis(d)=top/9;
            d=d+1;

         %d=7    135-315 derece
```

```
                top=J(((k*3)+9),((m*3)+9))+J(((k*3)+8),((m*3)+8))+...
                    J(((k*3)+7),((m*3)+7))+J(((k*3)+6),((m*3)+6))+...
                    J(((k*3)+5),((m*3)+5))+J(((k*3)+4),((m*3)+4))+...
                    J(((k*3)+3),((m*3)+3))+J(((k*3)+2),((m*3)+2))+...
                    J(((k*3)+1),((m*3)+1));
%                 C(k+1,m+1,d)=top/9;
                axis(d)=top/9;
                d=d+1;

              %d=8   157.5-347.5 derece
                 top=J(((k*3)+3),((m*3)+1))+ J(((k*3)+4),((m*3)+3))+...
                     J(((k*3)+5),((m*3)+5))+J(((k*3)+6),((m*3)+7))+...
                     J(((k*3)+7),((m*3)+9))+...
                     ((J(((k*3)+3),((m*3)+2))+J(((k*3)+4),((m*3)+2)))/2)+...
                     ((J(((k*3)+4),((m*3)+4))+J(((k*3)+5),((m*3)+4)))/2)+...
                     ((J(((k*3)+5),((m*3)+6))+J(((k*3)+6),((m*3)+6)))/2)+...
                     ((J(((k*3)+6),((m*3)+8))+J(((k*3)+7),((m*3)+8)))/2);
%                    C(k+1,m+1,d)=top/9;
                 axis(d)=top/9;
                 d=d+1;


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
minimum=min(axis);
%%%%3 maksimumun bulunmasi%%%%%%%%
           denekAxis=axis;
           topMax=0;
           [z pCols]=size(denekAxis);
     for o=1:3
       azami=max(denekAxis);
       topMax=topMax+azami;
       for p=1:pCols

           if(azami==denekAxis(p))
              denekAxis(1,p) = 0;
              break;
           end

       end

     end
     ortMax=topMax/3;


       if((ortMax-minimum)/ortMax >= dinamikKosul)
           %%%%%%%%%%%%%%%%%%%%%%%%%%%%
           eksenSayaci=eksenSayaci+1;
           [R,C,X]=EksenCizimiV103(C,k,m,X,minimum,axis,R);
       end
    end
end
```

```
function [R,C,X]=EksenCizimiV103(C,k,m,X,minimum,axis,R)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amacı, Görüntüye ait eksen haritasına göre sınırları ve
%eksenleri çizmektir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


            if(minimum==axis(1))    %d=1    0-180 derece

                 for g=1:9
                     X(((k*3)+5),((m*3)+g),:)=[255,0,0];%kirmizi

                     R(((k*3)+5),((m*3)+g),:)=1;
                 end
                 C(k+1,m+1)=1;

            elseif(minimum==axis(2))%d=2    22.5-202.5 derece
                 X(((k*3)+3),((m*3)+9),:)=[0,255,0];%yesil
                 X(((k*3)+4),((m*3)+7),:)=[0,255,0];
                 X(((k*3)+5),((m*3)+5),:)=[0,255,0];
                 X(((k*3)+6),((m*3)+3),:)=[0,255,0];
                 X(((k*3)+7),((m*3)+1),:)=[0,255,0];
                 X(((k*3)+4),((m*3)+8),:)=[0,255,0];
                 X(((k*3)+5),((m*3)+6),:)=[0,255,0];
                 X(((k*3)+6),((m*3)+4),:)=[0,255,0];
                 X(((k*3)+7),((m*3)+2),:)=[0,255,0];

                 R(((k*3)+3),((m*3)+9),:)=2;%yesil
                 R(((k*3)+4),((m*3)+7),:)=2;
                 R(((k*3)+5),((m*3)+5),:)=2;
                 R(((k*3)+6),((m*3)+3),:)=2;
                 R(((k*3)+7),((m*3)+1),:)=2;
                 R(((k*3)+4),((m*3)+8),:)=2;
                 R(((k*3)+5),((m*3)+6),:)=2;
                 R(((k*3)+6),((m*3)+4),:)=2;
                 R(((k*3)+7),((m*3)+2),:)=2;

                 C(k+1,m+1)=2;

            elseif(minimum==axis(3)) %d=3    45-225 derece

                 for g=1:9
                     X(((k*3)+g),((m*3)+(10-g)),:)=[0,0,255];%mavi

                     R(((k*3)+g),((m*3)+(10-g)),:)=3;
                 end
                 C(k+1,m+1)=3;

            elseif(minimum==axis(4))%d=4     67.5-247.5 derece

                 X(((k*3)+1),((m*3)+7),:)=[0,255,255];%turkuaz
                 X(((k*3)+3),((m*3)+6),:)=[0,255,255];
                 X(((k*3)+5),((m*3)+5),:)=[0,255,255];
                 X(((k*3)+7),((m*3)+4),:)=[0,255,255];
```

```
        X(((k*3)+9),((m*3)+3),:)=[0,255,255];
        X(((k*3)+2),((m*3)+6),:)=[0,255,255];
        X(((k*3)+4),((m*3)+5),:)=[0,255,255];
        X(((k*3)+6),((m*3)+5),:)=[0,255,255];
        X(((k*3)+8),((m*3)+4),:)=[0,255,255];


        R(((k*3)+1),((m*3)+7),:)=4;%turkuaz
        R(((k*3)+3),((m*3)+6),:)=4;
        R(((k*3)+5),((m*3)+5),:)=4;
        R(((k*3)+7),((m*3)+4),:)=4;
        R(((k*3)+9),((m*3)+3),:)=4;
        R(((k*3)+2),((m*3)+6),:)=4;
        R(((k*3)+4),((m*3)+5),:)=4;
        R(((k*3)+6),((m*3)+5),:)=4;
        R(((k*3)+8),((m*3)+4),:)=4;
        C(k+1,m+1)=4;

    elseif(minimum==axis(5)) %d=5     90-360 derece
        for g=1:9
            X(((k*3)+g),((m*3)+5),:)=[255,0,0];%pembe

            R(((k*3)+g),((m*3)+5),:)=5;
        end
        C(k+1,m+1)=5;

    elseif(minimum==axis(6)) %d=6    112.5-292.5 derece
        X(((k*3)+1),((m*3)+3),:)=[255,255,0];%sari
        X(((k*3)+3),((m*3)+4),:)=[255,255,0];
        X(((k*3)+5),((m*3)+5),:)=[255,255,0];
        X(((k*3)+7),((m*3)+6),:)=[255,255,0];
        X(((k*3)+9),((m*3)+7),:)=[255,255,0];
        X(((k*3)+2),((m*3)+4),:)=[255,255,0];
        X(((k*3)+4),((m*3)+5),:)=[255,255,0];
        X(((k*3)+6),((m*3)+5),:)=[255,255,0];
        X(((k*3)+8),((m*3)+6),:)=[255,255,0];

        R(((k*3)+1),((m*3)+3),:)=6;%sari
        R(((k*3)+3),((m*3)+4),:)=6;
        R(((k*3)+5),((m*3)+5),:)=6;
        R(((k*3)+7),((m*3)+6),:)=6;
        R(((k*3)+9),((m*3)+7),:)=6;
        R(((k*3)+2),((m*3)+4),:)=6;
        R(((k*3)+4),((m*3)+5),:)=6;
        R(((k*3)+6),((m*3)+5),:)=6;
        R(((k*3)+8),((m*3)+6),:)=6;

        C(k+1,m+1)=6;

    elseif(minimum==axis(7)) %d=7    135-315 derece
        for g=1:9
            X(((k*3)+g),((m*3)+g),:)=[0,0,0];%siyah

            R(((k*3)+g),((m*3)+g),:)=7;
        end
```

```
                C(k+1,m+1)=7;

            elseif(minimum==axis(8))%d=8   157.5-347.5 derece
                X(((k*3)+3),((m*3)+1),:)=[255,0,255];%turuncu
                X(((k*3)+4),((m*3)+3),:)=[255,0,255];
                X(((k*3)+5),((m*3)+5),:)=[255,0,255];
                X(((k*3)+6),((m*3)+7),:)=[255,0,255];
                X(((k*3)+7),((m*3)+9),:)=[255,0,255];
                X(((k*3)+4),((m*3)+2),:)=[255,0,255];
                X(((k*3)+5),((m*3)+4),:)=[255,0,255];
                X(((k*3)+6),((m*3)+6),:)=[255,0,255];
                X(((k*3)+7),((m*3)+8),:)=[255,0,255];

                R(((k*3)+3),((m*3)+1),:)=8;%turuncu
                R(((k*3)+4),((m*3)+3),:)=8;
                R(((k*3)+5),((m*3)+5),:)=8;
                R(((k*3)+6),((m*3)+7),:)=8;
                R(((k*3)+7),((m*3)+9),:)=8;
                R(((k*3)+4),((m*3)+2),:)=8;
                R(((k*3)+5),((m*3)+4),:)=8;
                R(((k*3)+6),((m*3)+6),:)=8;
                R(((k*3)+7),((m*3)+8),:)=8;

                C(k+1,m+1)=8;
            else
                C(k+1,m+1)=0;
            end

function [F]=EksenFiltrelemeV103(C)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amacı, görüntüdeki eksenlerin ikili şablonlara göre kontrolü
%sonrasında filtrelemektir
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[Fr Fc]=size(C);
F=zeros(Fr,Fc)
for i=4:1:Fr-4
    for j=4:1:Fc-4
        %%180 derece cizgiler icin kontrol
        if(C(i,j)==1)
            if(C(i,j-3)==1 || C(i,j+3)==1 || C(i+1,j-3)==2 || ...
                    C(i-1,j+3)==2 || C(i+1,j-3)==3 || C(i+2,j-3)==3 ||...
                    C(i-1,j+3)==3 || C(i-2,j+3)==3  || C(i+1,j-2)==4 || ...
                    C(i+2,j-2)==4 || C(i-1,j+2)==4 || C(i-2,j+2)==4 ...
            || C(i-2,j-2)==6 || C(i-1,j-2)==6 || C(i+2,j+2)==6 || ...
            C(i+1,j+2)==6 || C(i-1,j-3)==7 || C(i-2,j-3)==7|| ...
            C(i+2,j+3)==7 || C(i+1,j+3)==7 || C(i-1,j-3)==8 || ...
            C(i+1,j+3)==8 )

                F(i,j)=C(i,j);
            else
                F(i,j)=0;
            end

        elseif(C(i,j)==2)
```

```
    if(C(i+2,j-3)==2 || C(i+1,j-3)==2 || C(i-2,j+3)==2 || ...
            C(i-1,j+3)==2 || C(i+2,j-3)==3 || C(i-2,j+3)==3 || ...
            C(i+2,j-2)==4 || C(i-2,j+2)==4 || C(i-1,j-3)==7 || ...
            C(i+1,j-3)==7 || C(i,j-3)==8 || C(i,j+3)==8 ||...
            C(i+1,j-3)==1 || C(i-1,j+3)==1 || C(i-2,j+1)==5 || ...
            C(i+2,j-1)==5 || C(i+2,j-2)==5 || C(i-2,j+2)==5)

        F(i,j)=C(i,j);
    else
        F(i,j)=0;
    end
 elseif(C(i,j)==3)
    if(C(i+1,j-3)==1 || C(i+2,j-3)==1 || C(i-1,j-3)==1 || ...
            C(i-2,j-3)==1 || C(i-2,j+3)==2 || C(i+2,j-3)==2 ...
            ||C(i-3,j+3)==3 || C(i+3,j-3)==3 || C(i-3,j+2)==4 ...
            || C(i+3,j-2)==4 || C(i+3,j-2)==5 || C(i+3,j-1)==5 ...
            || C(i-3,j+2)==5 || C(i-3,j+1)==5 || ...
       C(i+1,j-3)==8 || C(i,j-3)==8 || C(i-1,j+3)==8 || ...
       C(i,j+3)==8 || C(i+3,j-1)==6 || C(i-3,j+1)==6)

        F(i,j)=C(i,j);
    else
        F(i,j)=0;
    end
elseif(C(i,j)==4)
    if(C(i+3,j-1)==4 || C(i+3,j-2)==4 || C(i-3,j+2)==4 || ...
            C(i-3,j+1)==4  || C(i+3,j-1)==5 || C(i-3,j+1)==5 ||...
            C(i-3,j)==6 || C(i+3,j)==6 ...
            || C(i-3,j)==7 || C(i+3,j)==7 || C(i-1,j+2)==1 || ...
            C(i-2,j+2)==1 || C(i+1,j-2)==1 || C(i+2,j-2)==1 ...
            || C(i-2,j+2)==2 || C(i+2,j-2)==2 || C(i-3,j+2)==3 ...
            || C(i+3,j-2)==3)

        F(i,j)=C(i,j);
    else
        F(i,j)=0;
    end
elseif(C(i,j)==5)
    if(C(i-3,j)==5 || C(i+3,j)==5 || C(i-3,j-1)==6 || ...
            C(i+3,j+1)==6 ||C(i-3,j-1)==7 || C(i+3,j+1)==7 ||...
            C(i-3,j-2)==7|| C(i+3,j+2)==7  || C(i-2,j-1)==8 || ...
        C(i+2,j+1)==8 || C(i-2,j+1)==2 || C(i+2,j-1)==2 ||...
        C(i+2,j-2)==2 || C(i-2,j+2)==2 || C(i+3,j-2)==3 ||...
        C(i+3,j-1)==3 || C(i-3,j+2)==3 || C(i-3,j+1)==3 || ...
        C(i+3,j-1)==4 || C(i-3,j+1)==4)

        F(i,j)=C(i,j);
    else
        F(i,j)=0;
    end

elseif(C(i,j)==6)
    if(C(i-3,j-2)==6 || C(i-3,j-1)==6 || C(i+3,j+2)==6 || ...
            C(i+3,j+1)==6 ...
            || C(i-3,j-2)==7 || C(i+3,j+2)==7 || C(i-2,j-2)==8||...
            C(i+2,j+2)==8 || C(i-2,j-2)==1 || C(i-1,j-2)==1 || ...
```

```
                    C(i+1,j+2)==1 || C(i+2,j+2)==1 || C(i+3,j-1)==3 || ...
                    C(i-3,j+1)==3 || C(i-3,j)==4 || C(i+3,j)==4 || ...
                    C(i-3,j-1)==5 || C(i+3,j+1)==5)

                F(i,j)=C(i,j);
            else
                F(i,j)=0;
            end
        elseif(C(i,j)==7)
            if(C(i-3,j-3)==7 || C(i+3,j+3)==7 || C(i-2,j-3)==8 || ...
                    C(i+2,j+3)==8 || C(i-1,j-3)==1 || C(i-2,j-3)==1 ||...
                    C(i+1,j+3)==1 || C(i+2,j+3)==1 || C(i-1,j-3)==2 ...
                    || C(i+1,j-3)==2 || C(i-3,j)==4 || C(i+3,j)==4 || ...
                    C(i-3,j-1)==5 || C(i+3,j+1)==5 || C(i-3,j-2)==5 || ...
                    C(i+3,j+2)==5 || C(i-3,j-2)==6 || C(i+3,j+2)==6)
                F(i,j)=C(i,j);
            else
                F(i,j)=0;
            end
         elseif(C(i,j)==8)
            if(C(i-2,j-3)==1 ||C(i+2,j+3)==1 || C(i,j-3)==2 ...
                    || C(i,j+3)==2 || C(i+1,j-3)==3 || C(i,j-3)==3 ...
                    || C(i-1,j+3)==3 || C(i,j+3)==3|| C(i-2,j-1)==5 ...
                    ||C(i+2,j+1)==5|| C(i-2,j-2)==6|| C(i+2,j+2)==6 ...
                || C(i-2,j-3)==7 || C(i+2,j+3)==7 || C(i-2,j-3)==8 ...
                || C(i+2,j+3)==8 )
                F(i,j)=C(i,j);
            else
                F(i,j)=0;
            end
         end

    end
end

function[PotMerk,X]= MerkezBulmaV103(F,X,R,Iadjust,ciktiDizini)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amacı, görüntüdeki eksenlerin karşılıklı taramalarına göre orta
%noktaların tespit edilmesidir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[fR fC]=size(F);
[xR xC xZ]=size(X);
maxCap=160;
minCap=10;
pSayac=1;
O=1;
D(1,1)=1;%bitiş noktalarını tutan matris
PotMerk=[1,1,1,1,1,1,1,1,1];

for k=0:1:fR-3
    for m=0:1:fC-3
        if(F(k+1,m+1)~=0)%~=

            dikAci=mod((((F(k+1,m+1)-1)*22.5)+90),360);
```

```
          %aciya 90 derece dik aci


          iE=((k*3)+5);
%eksenin merkez alindigi pikselin i koordinati (9luk eksenin tam ortasi)
          jE=((m*3)+5);
%eksenin merkez alindigi pikselin j koordinati (9luk eksenin tam ortasi)
          d=minCap;
          i=round(iE-sin(dikAci*pi/180)*(d));
          j=round(jE+cos(dikAci*pi/180)*(d));
          cap=sqrt((i-((k*3)+5))^2+(j-((m*3)+5))^2);


          while ( R(i,j)==0 && i>1 && j>1 && i<xR && j<xC && cap<maxCap)


               cap=sqrt((i-iE)^2+(j-jE)^2);

               d=d+1;
               i=round(iE-sin(dikAci*pi/180)*(d));
               j=round(jE+cos(dikAci*pi/180)*(d));

          end



          if(i>1 && j>1 && i<xR && j<xC && cap~=maxCap-1 && ...
                  (R(i,j)==R(iE,jE) ||(R(i,j)==mod(R(iE,jE)+1,8)) ...
                  || (R(i,j)==mod(R(iE,jE)-1,8))))

               rad=cap/2;
               ri=round(iE-sin(dikAci*pi/180)*(rad));
               rj=round(jE+cos(dikAci*pi/180)*(rad));

              degCtr=360/(rad*10);
              % derece artisinin ne kadar yapilacaginin hesaplanmasi
              C=uint32(0);
              ctr=0;

              for degree=0:degCtr:360

                  idegeri=round(ri-sin(degree*pi/180)*(rad+2));
                  jdegeri=round(rj+cos(degree*pi/180)*(rad+2));
                  if(idegeri>=1 && idegeri<=xR && jdegeri>=1 &&...
                          jdegeri<=xC )
                  % resim boyutlari disina cikilip cikilmadigi kontrolu
                  C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*rad),...
                      round(rj+cos(degree*pi/180)*rad)));
                  C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*...
                      (rad+1)),round(rj+cos(degree*pi/180)*(rad+1))));
                  C=C+uint32(Iadjust(idegeri,jdegeri));
                  % rad +2 hesabi,zaten yukarida hesaplandigi icin
                  % tekrar hesaplanmamakta
                  C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*...
```

```
                    (rad-1)),round(rj+cos(degree*pi/180)*(rad-1))));
                % toplam deger
                C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*...
                    (rad-2)),round(rj+cos(degree*pi/180)*(rad-2))));

                ctr=ctr+5;
                %ortalama alabilmek icin sayac

                end %if (idegeri>=1 && id...

            end%degree=0:degCtr:360
            %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
            kucukRadOrt=C/ctr;

            C=uint32(0);
            ctr=0;
            radB=rad+5;

            for degree=0:degCtr:360

                idegeri=round(ri-sin(degree*pi/180)*(radB+2));
                jdegeri=round(rj+cos(degree*pi/180)*(radB+2));
                if(idegeri>=1 && idegeri<=xR && jdegeri>=1 &&...
                        jdegeri<=xC )
                % resim boyutlari disina cikilip cikilmadigi kontrolu
                C=C+uint32(Iadjust(round(ri-
sin(degree*pi/180)*radB),...
                    round(rj+cos(degree*pi/180)*radB)));
                C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*...
                    (radB+1)),round(rj+cos(degree*pi/180)*(radB+1))));
                C=C+uint32(Iadjust(idegeri,jdegeri));
                % rad +2 hesabi,zaten yukarida hesaplandigi icin
                % tekrar hesaplanmamakta
                C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*...
                    (radB-1)),round(rj+cos(degree*pi/180)*(radB-1))));
                % toplam deger
                C=C+uint32(Iadjust(round(ri-sin(degree*pi/180)*...
                    (radB-2)),round(rj+cos(degree*pi/180)*(radB-2))));

                ctr=ctr+5;
                %ortalama alabilmek icin sayac

                end %if (idegeri>=1 && id...

            end%for degree=0:degCtr:360
            buyukRadOrt=C/ctr;
             O(pSayac,1)=ri;
             O(pSayac,2)=rj;
             O(pSayac,3)=rad;
             O(pSayac,4)=0;
             O(pSayac,5)=1;
             O(pSayac,6)=0;
             O(pSayac,7)=kucukRadOrt;
             O(pSayac,8)=buyukRadOrt;
             O(pSayac,9)=pSayac;
```

```
                X(ri,rj,:)=[0 0 0];


                  pSayac=pSayac+1;


            end% if(i>1 && j>1 && i<xR &

        end% if(F(k+1,m+1)~=0)
    end%for m=0:1:fC-3
end%for k=0:1:fR-3
PotMerk=0;
xlswrite(strcat(ciktiDizini,'OrtaNoktalar.xls'),0);

function [C]=AgirlikMerkeziHesapla(PotOrigin)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci birbirine cok yakin bulunan merkezleri tek bir
%merkezde toplamak(ortalama) ve ustuste binmis kopuklerin olusmasini
%engellemektir. her radiusa gore taramadan sonra bu
%kontrol yapilarak yanlis kopuk sonuclarinin cikarilmasi engellenir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[nRows nCols]=size(PotOrigin);

%Agirlik Merkezinin hesaplandığı matris
C(1,1)=PotOrigin(1,1);%i
C(1,2)=PotOrigin(1,2);%j
C(1,3)=PotOrigin(1,3);%rad
C(1,4)=PotOrigin(1,4);%
C(1,5)=PotOrigin(1,5);%sayac
C(1,6)=PotOrigin(1,6);
C(1,7)=PotOrigin(1,7);
C(1,8)=PotOrigin(1,8);
C(1,9)=PotOrigin(1,9);

%bulundu=0;
for z=2:nRows
    k=1;
    bulundu=0;
    [kRows kCols]=size(C);
    %bütün köpüklerin bulunulan köpükle ortak bileşik olup olmadığı kontrol
    %edilmekte
    while(k<=kRows)

          if ((sqrt(abs((PotOrigin(z,1)-C(k,1)))^2+abs((PotOrigin(z,2)...
                  -C(k,2)))^2) < (C(k,3)/2))  )

             C(k,1)=((C(k,1)*C(k,5))+PotOrigin(z,1))/(C(k,5)+1);
             C(k,2)=((C(k,2)*C(k,5))+PotOrigin(z,2))/(C(k,5)+1);
             C(k,3)=((C(k,3)*C(k,5))+PotOrigin(z,3))/(C(k,5)+1);
             C(k,7)=((C(k,7)*C(k,5))+PotOrigin(z,7))/(C(k,5)+1);
             C(k,8)=((C(k,8)*C(k,5))+PotOrigin(z,8))/(C(k,5)+1);
             C(k,9)=0;
             % her bulunan ortak köpük merkezi için ağırlık katsayısı
             % artırılmakta
```

```
            C(k,5)=C(k,5)+1;

           bulundu=1;
           break;

       end
    k=k+1;

    end
    %herhangi bir köpükle bileşik olmayan köpüklerin matriste set edilmesi
    if(bulundu==0)

     C(kRows+1,1)=PotOrigin(z,1);
     C(kRows+1,2)=PotOrigin(z,2);
     C(kRows+1,3)=PotOrigin(z,3);
     C(kRows+1,4)=PotOrigin(z,4);
     C(kRows+1,5)=PotOrigin(z,5);
     C(kRows+1,6)=PotOrigin(z,6);
     C(kRows+1,7)=PotOrigin(z,7);
     C(kRows+1,8)=PotOrigin(z,8);
     C(kRows+1,9)=PotOrigin(z,9);
     PotOrigin(z,5)=kRows+1;

    end

end

function [PotMerkDogrular,Pot]=MerkezKontrol(Pot,U,HangiEksen)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci merkezlerden dogrular cikarilarak eksenlerin kontrolünün
%yapılmasını sağlamaktır. Bu sayede yanlis cikarilan merkezler
%filtrelenecektir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[pR pC]=size (Pot);
[xR xC xZ]=size(U);

x=1;                    % Potansiyel Merkezleri tutan sayac
while (x<=pR)

    aciSayaci=1;
    acisalYaricap=1;
    count=0;            %Merkezden gonderilen isinlarin sayisini tutansayac
    ri=Pot(x,1);        %Merkezin i koordinati
    rj=Pot(x,2);        %Merkezin j koordinati
    rad=Pot(x,3);    %Merkezin yarıcapi
    yakinEksenVar=0;    %yakinda eksen olup olmadigin kontrol eden trigger

    for aci=0:10:350%22.5 derecelik acilarla dogru gonderiyoruz
        r=1;            %r=1 den baslayarak rad*1.2ye kadar arttırılıyor
        di=round(ri-sin(aci*pi/180)*(r));%aci ve r degerine gore gidilen
                                         %pikselin i koordinati
        dj=round(rj+cos(aci*pi/180)*(r));%aci ve r degerine gore gidilen
                                         %pikselin j koordinati
```

```
%       yaricap=round(sqrt((di-ri)^2+(dj-rj)^2));%di ve dj degerlerinegore

    while (di>1 && dj>1 && di<xR && dj<xC && HangiEksen(di,dj)==0 ...
                           && r<rad*1.2 )


       r=r+1;
       di=round(ri-sin(aci*pi/180)*(r));
       dj=round(rj+cos(aci*pi/180)*(r));
     %  yaricap=round(sqrt((di-ri)^2+(dj-rj)^2));


    end


     if (di>1 && dj>1 && di<xR && dj<xC && r<=((rad*1.2)-1) && r>=...
             ((rad*0.6)) &&...
         ( HangiEksen(di,dj)==(round(mod(aci+90,180)/22.5)+1) ...
         ||HangiEksen(di,dj)==(round(mod(aci+112.5,180)/22.5)+1) || ...
         HangiEksen(di,dj)==(round(mod(aci+67.5,180)/22.5)+1)))


     acisalYaricap(aciSayaci)=r;

       count=count+1;

     elseif (r<((rad*0.6)))

       yakinEksenVar=1;
       acisalYaricap(aciSayaci)=0;

     else

       if(aciSayaci==1)
           acisalYaricap(aciSayaci)=rad;
       else
           acisalYaricap(aciSayaci)=0;
       end
     end
       aciSayaci=aciSayaci+1;
   end



  if(count<=4 || yakinEksenVar==1  )

       Pot(x,:) = [];

  else

       PotMerkDogrular(x,:)=acisalYaricap;
```

```
        skoko=0;
        x=x+1;

   end


   [pR pC]=size (Pot);
end

function[PotMerkDogrular,IPAciArtimi]=YaricapInterPolasyon(PotMerkDogrular)
[pR pC]=size (PotMerkDogrular);
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci cikarilan yaricap degerleri arasinda interpolasyon
%yapilarak 0 olan degerlerin atanmasidir.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


for x=1:pR
    y=1;


    while(y<=36)

         siradaki=y+1;%kendisine en yakin 0 olmayan yaricap ile mesafesini
                      %tutan sayac

        modSiradaki=mod(siradaki,36);
        if(modSiradaki==0)
            modSiradaki=36;
        end%  if(modSiradaki==0)

        while(PotMerkDogrular(x,modSiradaki)==0)
            %0 olmayana kadar bir sonraki degeri kontrol ediyor,
            siradaki=siradaki+1;
            modSiradaki=mod(siradaki,36);
            if(modSiradaki==0)
                modSiradaki=36;
            end% if(modSiradaki==0)

        end%while(PotMerkDogrular

        yaricaplarFarki=PotMerkDogrular(x,modSiradaki)-
PotMerkDogrular(x,y);
        kacKesit=siradaki-y;
        araSayac=0;
        while(araSayac<kacKesit)
            PotMerkDogrular(x,y+araSayac)=PotMerkDogrular(x,y)+...
                (araSayac*(yaricaplarFarki/kacKesit));
            IPAciArtimi(x,y+araSayac)=yaricaplarFarki/(kacKesit*10);
            araSayac=araSayac+1;
        end%while(araSayac<kacParca)
```

```
        if(siradaki<=36)
            y=siradaki;
        else
            break;
        end

    end%while y<=16
end

function [Pot,PotMerkDogrular,StSapma,IPAciArtimi]=KopukleriAyirStDev...
    (Pot,PotMerkDogrular,IPAciArtimi)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci, iki köpüğün cok yakin olmasi durumunda standart sapma
%ortalamasina gore yuksek olani filtrelemektir
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[nRows nCols]=size(PotMerkDogrular);


x=1;
StSapma=std(PotMerkDogrular,0,2);

while(x<=nRows-1)

    y=x+1;

    while(y<=nRows)

        if(sqrt((Pot(x,1)-Pot(y,1))^2+(Pot(x,2)-Pot(y,2))^2)<0.8*Pot(x,3))

            if(StSapma(x)>StSapma(y))

            Pot(x,:) = [];
            PotMerkDogrular(x,:)=[];
            StSapma(x)=[];
            IPAciArtimi(x,:)=[];
            break

            elseif(StSapma(x)<=StSapma(y))

            Pot(y,:) = [];
            PotMerkDogrular(y,:)=[];
            StSapma(y)=[];
            IPAciArtimi(y,:)=[];

            end
        else
            y=y+1;

        end
```

```
    [nRows nCols]=size(Pot);

    end
    x=x+1;
    [nRows nCols]=size(Pot);
end

function [SG]=SinirTamamlama(PotMerkDogrular,PotMerk,SG,IPAciArtimi)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%AÇIKLAMALAR
%Bu metodun amaci, sinirin cizilmesini saglamaktir.
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[sR sC]=size(SG);
[pR pC]=size (PotMerkDogrular);

for x=1:pR

    yaricap=PotMerkDogrular(x,1);

    for degree=0:1:359

            i=round(PotMerk(x,1)-sin(degree*pi/180)*yaricap);
            j=round(PotMerk(x,2)+cos(degree*pi/180)*yaricap);


            if(i>0 && i<sR && j>0 && j<sC)
            SG(i,j)=0;
            end
            yaricapArtisi=IPAciArtimi(x,floor(degree/22.5)+1);
            yaricap=yaricap+yaricapArtisi;

    end% for degree=0:1:360
end%for x=1:pR
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, Adı : ALTAŞ, Veli Mert
Uyruğu : T.C.
Doğum tarihi ve yeri : 03.04.1982 Ankara
Medeni hali : Bekar
Telefon : 0 (312) 292 40 76
Faks : 0 (312) 292 40 91
e-mail : maltas@etu.edu.tr

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Lisans	Çankaya Üniversitesi/Bilgisayar Mühendisliği	2005

İş Deneyimi

Yıl	Yer	Görev
2005-2007	TOBB Ekonomi ve Teknoloji Üniversitesi	Araştırma Görevlisi
2004-2005	Bisay Yazılım	Proje Sorumlusu

Yabancı Dil

İngilizce

Yayınlar