T.C.
SAKARYA ÜNİVERSİTESİ
BİLGİSAYAR VE BİLİŞİM BİLİMLERİ FAKÜLTESİ

BSM 401 BİLGİSAYAR MÜHENDİSLİĞİ TASARIMI

GÖRÜNTÜ İŞLEME TABANLI KUMAŞ KALİTESİ TEST YAZILIMI

G131210097 – Mansur MAHMUTOV
G151210307 – Zeynep TORUN
G151210568 – Umut ZARİF

Bölüm : BİLGİSAYAR MÜHENDİSLİĞİ
Danışman : Doç. Dr. Devrim AKGÜN

2018-2019 Güz Dönemi

ÖNSÖZ

Endüstriyel firmalarda hız ve kalite çok önemli bir konudur. Çoğu durumlarda manuel yapılan işlemlerde meydana gelen hatalar sistemdeki üretim hızını ve müşteri memnuniyetini oldukça etkiler. Bu durumla ilgili olarak manuel işlemleri azaltmak, teknolojik gelişmelere uygun şekilde düzenleme yapmak üretim kalitesini, hızını ve buna bağlı olarak maliyetini olumlu yönde etkiler. Amaç hız ve kalite iken firmalar inceleme araçları, robotlar ve görüntü işleme gibi teknolojik gelişmeleri takip ederler. Görüntü işleme teknolojisi çoğu insanın göremediği, hızla hesaplayamacağı veya dikkatsizlik-anlık dalgınlıkla kaçıracağı hesaplamalarda yüksek oranda olumlu sonuçlar çıkardığı görülmektedir.

İÇİNDEKİLER

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Görüntü işleme, Matlab, Java, OpenCV

Görüntü işleme, görüntüyü dijital form haline getirmek ve bazı işlemleri gerçekleştirmek için geliştirilmiş, spesifik görüntü elde etmek veya ondan bazı yararlı bilgiler çıkarmak için kullanılan bir yöntemdir. Bu yöntemin girdisi video kesiti veya fotoğraf gibi bir görüntüdür. Çıktısı ise görüntünün istenilen ya da dikkat edilmesi gereken bölümüne karşılık gelir. Bu kapsamda her sektörde uygulama alanı bulan görüntü işleme tekniklerini endüstriyel bir sorun olan kumaş testinde kullanılması konusunda çalışma yapılacaktır.

Bu çalışma ile yapılmak istenen kumaş üzerinde yapılan çekmezlik testini açıklayıp, nasıl yapıldığını anlattıktan sonra, görüntü işleme teknikleri ile daha net sonuçlar elde eden bir sistem hazırlamaktır. Bu sistemin algoritması, Matlab'ın yaygın kullanılan görüntü işleme komutlarıyla hazırlanmıştır. Daha sonra Netbeans ortamında Java ile OpenCV (Açık Kaynak Bilgisayarlı Görüntü Kitaplığı) kütüphanesi kullanarak gerçeklenmesi tasarlanmaktadır.

BÖLÜM 1. GİRİŞ

Dokuma kumaşlar çok değişik özelliklerde ve çok çeşitli kullanım alanlarında karşımıza çıkarlar. Bu kadar farklı özelliğe ve kullanım alanına hitap eden dokuma kumaşların yapıları da birbirinden farklıdır. Kumaş yapısı, hem kumaşın özelliklerini etkilemesi açısından hem de kumaşın yüzey görünümünü belirlemesi açısından son derece önemlidir.

Kullanım amacına göre dokuma kumaşlarda bulunması istenilen bir takım özellikler vardır. Bunların arasında sağlamlık, esneklik, yumuşaklık, ısı tutma, nem çekme, dökümlülük, hava geçirgenliği, çekmezlik gibi özellikler sıralanabilir. Kumaşlarda bu istenilen özelliklerini olabilmesi için kumaşı oluşturan unsurların ve bunlar arasındaki ilişkilerin çok iyi incelenmesi gerekir.

Çekmezlik testi kumaşın yıkama öncesi ve sonrası değişen boyutlarının hesaplanmasıdır. Görüntü işleme ile bu testin yapılabilmesi için Matlab ile gereken hesaplamaları yapan bir algoritma hazırlanmıştır. Bu algoritma kullanılarak mobil ortama aktarılması veya masaüstü programın yazılması tasarlanmaktadır. Bunların sonucunda ise test sürecinde manuel yapılan işlem hataları minimuma indirmek istenmektedir.

BÖLÜM 2. GÖRÜNTÜ İŞLEME İLE KUMAŞ TESTİ

2.1. Kumaş Testi

Tekstil ürünleri, büyük ölçüde ne kadar dayanıklı olduklarını görmek için test edilirler. Buna bağlı olarak firma için bu test kalite demektir. Kumaş esneklik, dökümlülük, çekmezlik, dönme vb. özellikler kumaşın kalitesini belirler.

Dayanıklılığını ölçmek için yapılan testlerden bazıları ise şunlardır: Yırtılma dayanıklılığı, patlama mukavemeti, dikiş dayanımı, dikiş açılması ve iplik kaymasına karşı mukavemet, boncuklanma (pilling), aşınma dayanımı, hava geçirgenliği, su iticilik, buruşmazlık, statik elektriklenme ve sertlik gibi testler yapılmaktadır.

Bu testler üretim tesislerinin laboratuvarlarında hammadde girdi ve mamulleri kontrol altında tutabilmek için fiziksel uygunluğu test edilip, kalite ve kontrolleri onaylandıktan sonra müşteriye veya bir sonraki aşamada farklı bir departmana sevk edilir.

Aşağıda belirtilen testler her firmada farklılık göstermemesi açısından bazı standartlara uygun yapılır. Kalite kontrol proseslerinden geçemeyen kumaşlar için uygun olmayan ürün prosesi uygulanarak, sonraki aşamaya gönderilmesi engellenir [1].

Tablo 2.1. Bazı kumaş test standartları.

Test Adı	Standart	Test Amacı
Boyut Değişimi (Dimensionel Change)	ISO 6330	Kumaşlarda yıkamadan sonraki boyut değişimi kontrol edilir. Yıkanabilir kumaşlarda, kılıflarda vs. talep edilir.
Boncuklanma (Pilling)	ISO12945-2	Kumaşın zamanla boncuklanma eğiliminin tespiti yapılır.

İplik Sıklık Tayini (Determination of number of threads)	ISO 7211-2	Kumaşta kullanılan atkı ve çözgü iplik sayılarının tespiti yapılır.
pH Tayini (pH Value)	ISO 3071	Kumaşların pH seviyesinin tespiti yapılır.

Tabloda belirtilen standartlara göre kumaşlara testler uygulanır. Bu testler arasında bizim üzerinde çalışma yapacağımız test boyutsal değişim olan çekmezlik testidir. Çekmezlik testinin nasıl yapıldığını öğrendikten sonra geliştirme aşamasına geçilmektedir.

2.2. Kumaşta Çekmezlik nedir?

Çekme testinin bir amacı kumaşın kesim ya da dikimden sonra oluşabilecek çekmelerinin en baştan önüne geçmek için alınacak önlemleri sağlamak içindir. Çekmesi bozuk olan kumaş derhal terbiye işlemlerinin yapıldığı işletmeye gönderilir ve çekmesinin sağlanması istenir. Çekme testi yapılmadan kesilen ya da dikilen kumaş olması gereken beden ölçülerinden daima farklı sonuçların çıkmasına neden olur. Mesela yeni alınan bir giysinin ilk yıkamadan sonra daralması ya da bollaşması çekmezlik denen işlemin sağlıklı yapılmamasından kaynaklanmaktadır [2].

2.3. Çekmezlik Testi Yapılışı

Her gün işletmeden gelen kumaş parçalarının, kumaş örneği takip formunda verilen bilgileri okunur. Firma elemanlarına verilen takip formunda (Şekil 2.4.) müşteri ismi, parti numarası, kumaş kalitesi, renk, kumaş giriş eni, tüm en, besleme değeri bulunaktadır. Bu bilgiler ile test sonucu yorumlanır.

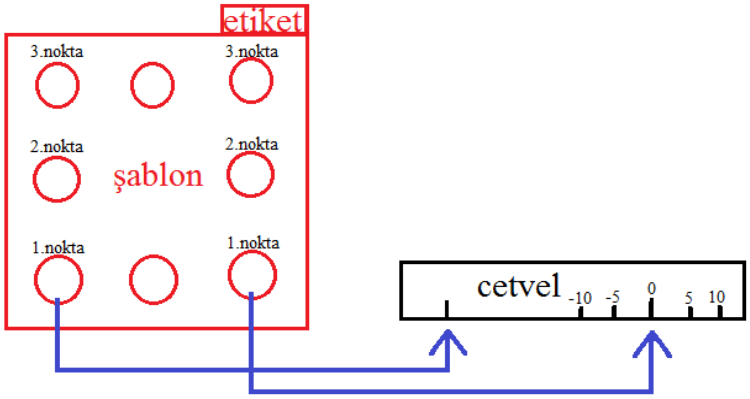
Kumaş üzerine konulan şablonun (Şekil 2.1) 35x35 cm olan çevresine ve içindeki noktalara boyalı bir kalem ile işaretlenir. Daha sonra kumaş çevresi işaretlenen yerden taşarak kumaş kesilir. Kesilen kenarlar dağılmaması amacıyla makinada dikilir ve kumaşlar yıkama makinesine gönderilir. [3]

Şekil 2.1. Vaskator şablonu

Yıkamaya gitmeden önce kumaşlardaki numaralandırılmış noktalar ile onların hizasındaki (en sağdaki) noktalar arası ölçüldüğünde sıfıra sıfırdır. Yıkama işleminden sonra +12 saat beklemeye alınan kumaş kurutma sonunda vaskator cetveli (Şekil 2.2.) ile tekrar ölçülür. Ne kadar sapma olduğu cetveldeki değerden okunarak kayıtlara geçer.

Şekil 2.2. Vaskator cetveli

Takip formunda belirtilen birinci nokta için kumaşın soluna yaslı halde vaskator cetveli (Şekil 1.2) düz bir şekilde konulur, sağ nokta cetvelin 0 noktasında ise kumaş sağlamdır. -15 - 0 arasında ise çekmiş, +15 - 0 arasında ise genişlemiştir.

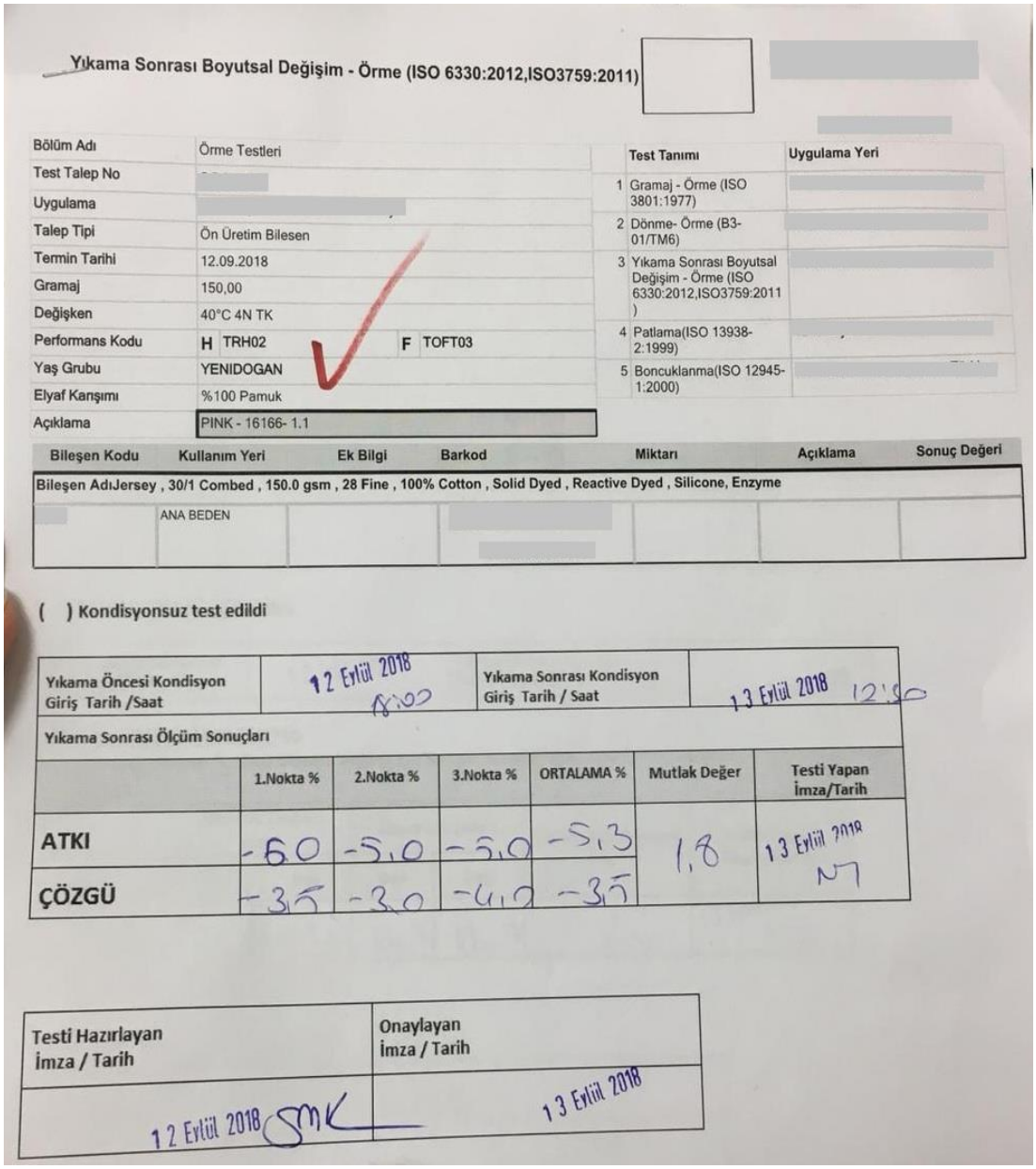


Şekil 2.3. Ölçüm işlemi temsili

Bu işlem etiket sağ üstte olduğunda **Atkı** değerini, kumaştaki etiket sağ yanında olduğunda (kumaş 90 derece sağa döndürüldüğünde) **çözgü** değerlerini verir. Her ikisi için de 3'er noktadan ölçüm yapılarak takip formundaki yerlerine yazılır.

Atkı ve çözgü değerlerine göre mutlak değerin 3'den büyük olması kumaşın testi geçemediğini gösterir (Bazı kumaş türleri için bu değer değişebilir). Eğer kumaş yıkamadan çıktığında çok çekmişse (+15'den büyük veya -15'den küçükse) her bir nokta için *demir cetvel* kullanılarak milimetrik ölçüm yapılır ve şu formul kullanılır: (Denklem 2.1)

$$350 - (\text{ölçü}len\ değer) = Sonuc$$
$$\left(\frac{sonuç}{350}\right) \times 100 = X.nokta\ değeri \quad \textbf{(2.1)}$$



Yıkama Sonrası Boyutsal Değişim - Örme (ISO 6330:2012,ISO3759:2011)

Bölüm Adı	Örme Testleri
Test Talep No	
Uygulama	
Talep Tipi	Ön Üretim Bilesen
Termin Tarihi	12.09.2018
Gramaj	150,00
Değişken	40°C 4N TK
Performans Kodu	H TRH02 F TOFT03
Yaş Grubu	YENIDOGAN
Elyaf Karışımı	%100 Pamuk
Açıklama	PINK - 16166- 1.1

	Test Tanımı	Uygulama Yeri
1	Gramaj - Örme (ISO 3801:1977)	
2	Dönme- Örme (B3-01/TM6)	
3	Yıkama Sonrası Boyutsal Değişim - Örme (ISO 6330:2012,ISO3759:2011)	
4	Patlama(ISO 13938-2:1999)	
5	Boncuklanma(ISO 12945-1:2000)	

Bileşen Kodu	Kullanım Yeri	Ek Bilgi	Barkod	Miktarı	Açıklama	Sonuç Değeri
Bileşen AdıJersey , 30/1 Combed , 150.0 gsm , 28 Fine , 100% Cotton , Solid Dyed , Reactive Dyed , Silicone, Enzyme						
	ANA BEDEN					

( ) Kondisyonsuz test edildi

Yıkama Öncesi Kondisyon Giriş Tarih /Saat	12 Eylül 2018 8:00			Yıkama Sonrası Kondisyon Giriş Tarih / Saat	13 Eylül 2018 12:50	
Yıkama Sonrası Ölçüm Sonuçları						
	1.Nokta %	2.Nokta %	3.Nokta %	ORTALAMA %	Mutlak Değer	Testi Yapan İmza/Tarih
ATKI	-6.0	-5.0	-5.0	-5.3	1,8	13 Eylül 2018 MK
ÇÖZGÜ	-3.5	-3.0	-4.0	-3.5		

Testi Hazırlayan İmza / Tarih	Onaylayan İmza / Tarih
12 Eylül 2018 SMK	13 Eylül 2018

Şekil 2.4. Takip formu örneği

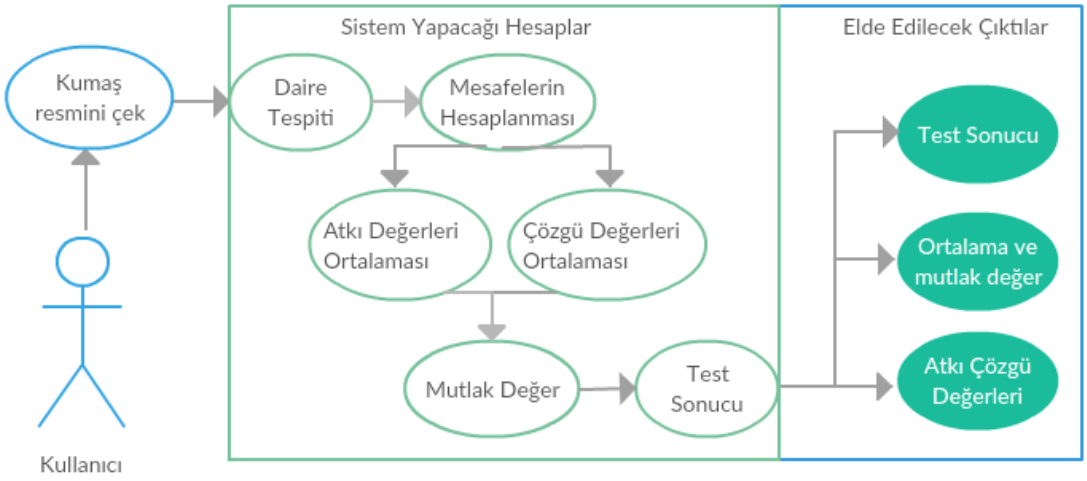
2.4. Çekmezlik Testinde Görüntü İşleme

Anlatılan test adımları sırasında manuel işlemlerde yapılan hatalar nedeniyle testten sağlıklı bir sonuç alınamaz. Yıkama sonrası kumaş üzerinde şablon ile konulmuş noktalar görüntü işleme teknikleri kullanılarak Android arayüz ile bulunduktan sonra manuel testte olduğu gibi noktalar arası mesafe hesaplatılması planlanmaktadır. Hesaplanan değerlere göre çekmezlik hesaplamaları takip formuna doğru ölçümler olarak yazılabilecektir.

Android arayüzden önce kullanılacak algoritma Matlab ile hazırlanmıştır. Daha sonra bu algoritma Netbeans de gerçeklenerek hem mobil hem de masaüstü ortama aktarılması için alt yapı oluşturulması tasarlanmaktadır.

Şablon üzerindeki noktaları bulmak amacıyla uygun algoritmayı yazmak ve işleyişi görmek için öncelikle Octave kullanılmıştır. Octave, Matlab benzeri olan open source ücretsiz GPL lisanslı program olup, kullanımı daha kolaydır. Fakat kaynakların yeterli olmaması nedeniyle Octave'dan vazgeçilmiştir. Matlab'in görüntü işleme konusunda daha fazla kaynağının olması ve daha yaygın olması sebebiyle tercih edilmiştir.

Kumaş testi için yazılacak olan algoritmanın UML diyagramı Şekil 2.5.'de gösterilmiştir.



Şekil 2.5. UML Diyagramı

BÖLÜM 3. TEKNOLOJİLER VE YÖNTEMLER

3.1. Matlab ile Daire Tespiti

Resimdeki dairelerin algılanması işlemlerini sırasıyla anlatmak gerekirse öncelikle cisimlerin ayırt edilebilmesi için resmi tek renk haline getirip üzerindeki gürültüden (anlamsız-küçük pixellerden) arındırmak gerekir. Daha sonra yapılandırarak daha net şekiller elde edilir.

```
RGB = imread('C:\Users\ZEYNEPTORUN\Desktop\dairetespit.png' );
imshow (RGB);

%resme filtre uygulamak ve gürültülerden temizlemek;
I = rgb2gray(RGB);
bw = im2bw(I);
imshow(bw)
bw = bwareaopen(bw,30);
se = strel('disk',2);
bw = imclose(bw,se);
bw = imfill(bw,'holes');
imshow(bw);

%daire sınırlarını bul
[B,L] = bwboundaries(bw,'holes');
imshow(label2rgb(L, @jet, [.5 .5 .5]))

hold on

for k = 1:length(B)
  boundary = B{k};
  plot(boundary(:,2), boundary(:,1), 'w', 'LineWidth', 2)
end
...
```

Resme uygulanan filtreler ve morfolojik işlemler;

- ***rgb2gray()*** gri filtreleme işlemi,
- ***im2bw()*** siyah-beyaz hale gelmesi için, beyaz(1) ile siyah(0) olarak yeniden çizer (Matlab 2016 sürümündeki adı imbinarize).
- ***bwareaopen(bw,30)*** 30 pixele kadar oluşan gürültüyü kadar gidermek için.
- ***strel('disk',2)*** fonksiyonu nesnenin dairesel yapısını korumak için kullanılan 2 pixel yarıçapında yapılandırma elemanıdır.
- ***imclose()*** görüntü üzerinde morfolojik bir kapatma işlemi gerçekleştirir.

- ***imfill(bw,'holes')*** metodu ile yapılandırma işlemi sırasında oluşan boşlukları doldurur.

Böylece olabildiği kadar düzgün bir dairesel yapı elde edilmeye çalışılır.

Bir sonraki işlem dairenin sınırlarını bulmaktır. Bunun için kullanılacak fonksiyonlar ise;

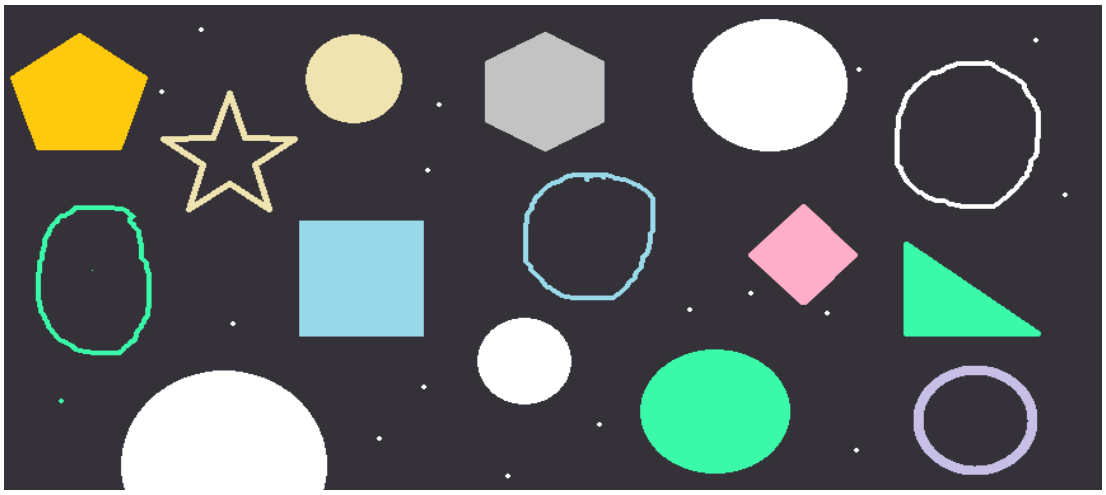
- ***bwboundaries()*** fonksiyonu resimdeki nesneleri bulmak için kullanılır. Nesnelerin dış sınırlarını ve bu nesneler içindeki deliklerin sınırlarını ikili görüntüde izler. Aldığı iki parametreden biri pixel sınır konumlarını içeren b, diğeri nesne içindeki deliklerin sınırları hücre dizisidir. Parametre olarak holes değerinin alınması, nesnelerin içindeki sınırların dahil edilmek istendiği ya da noholes değerinin alınması, dahil edilmek istenmediği anlamına gelir.
- ***label2rgb()*** resmi renk paleti şeklinde renklendirmeyi sağlar.
- ***regionprops(L, 'Area', 'Centroid', 'Perimeter')*** sırasıyla ilk parametre nesne, sonraki alanda bulunan piksel sayısı, bölgenin kütle merkezi, bölge sınırı etrafındaki mesafeyi döndürerek kısmen çevreyi hesaplar.
- ***plot()*** (nesne(:,2), nesne(:,1), çerceve rengi, hat genişliği, çerçeve çizgi boyutu) belirtilen parametlere göre nesneye çerçeve çizer [4]-[9].

Daha sonra for döngüsüyle matristeki her daire elemanının sınırlarını çizer. Döngü, birinci daireden başlayarak lenght(B) değerine (satır ve sütün arasından büyük olan sayıya) kadar döner.

Son işlem ise hangi nesnelerin yuvarlak olduğunu belirlemektir. Nesnelerin ağırlık merkezlerine göre alan ve çevre hesapmaları yapılır. Bu hesaplamalarla hangi orana sahip olduğu nesnenin üstüne yazılır. Test sırasında konulan noktanın boyutu 1cm değerini geçmediği için algoritmanın geliştirme sürecinde bulunan bu değerlerden boyuta göre eleme işlemi yapılacaktır.

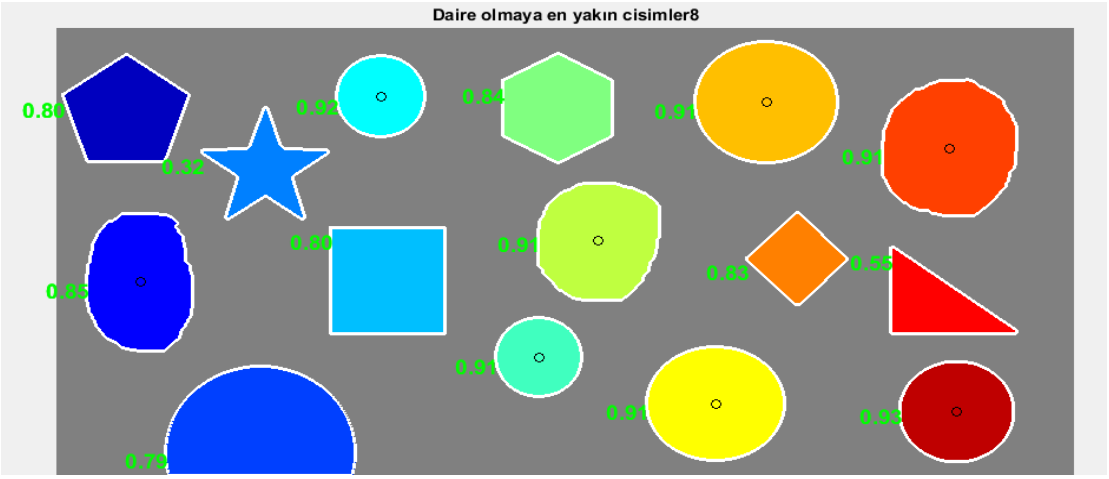
```
nesne_verileri = regionprops(L, 'Area', 'Centroid', 'Perimeter' );
threshold = 0.85;       %Daire sayılabilmesi için ağırlık merkezi
                          eşiği
total=0;                %daire sayısı sayacı
for k = 1:length(B)     %her bir daire için;
  boundary = B{k};      %Boundary; k. dairesinin x,y sınır
                       koordinatlarını tutan 2 boyutlu matrisi tutar
```

```
    delta_sq = diff(boundary).^2;%delta_sq değişkenine dairenin
                                 tahmini çevresini hesaplayıp atadık
   perimeter = sum(sqrt(sum(delta_sq,2)));%çevre = toplam(kök içinde
                                          (toplam delta_sq,5))
   area = nesne_verileri(k).Area;  %k. daireye karşılık gelen alan
                                    hesaplaması
   metric = 4*pi*area/perimeter^2;
   metric_string = sprintf('%2.2f',metric);%sonuçları değişkende
                                           tutmak
   if metric > threshold       %eğer metrik hesap büyükse ağırlık
                                merkezi eşiğinden dairedir
     centroid = nesne_verileri (k).Centroid;
     plot(centroid(1),centroid(2),'ko');
     total=total+1;
   end
   text(boundary(1,2)-35,boundary(1,1)+13,metric_string,
         'Color','g','FontSize',14,'FontWeight','bold');
End
title(['Daire olmaya en yakın cisimler', num2str(total)]);
```



Şekil 3.1. Örnek resim

Örnek resim (Şekil 3.1) üzerinden kod ekran çıktısını gösterilmiştir. (Şekil 3.2)



Şekil 3.2. Resmin algoritmaya göre ekran çıktısı

3.2. Noktalar Arasında Mesafenin Hesaplanması

Şekil 3.3. İki daire arasındaki mesafe

Şekil 3.4. İki daire arasındaki mesafe 2

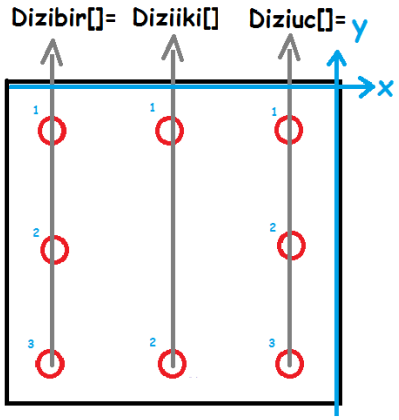
Daire tespiti yaparken bulunan ağırlık merkezlerinden x ve y koordinatları bulunur. Birinci dairenin x koordinatı ile ikinci noktanın x koordinatları arasındaki fark iki dairenin birbiri arasındaki mesafeyi verir.

Sekiz daire için bu işlem yapılmak istenirse, koordinatlar bir diziye atılıp doğru fark elde edilebilmesi için en küçük x ve y koordinat değerine göre sıralama yapılması gerekir. Bu işlemi yaparken Matlab yazma işlemine en küçük x koordinatını yazarak başlamaktadır (Şekil 3.5'de mavi yazılı sıralama ile gösterilmiştir). Yani birinci noktadan bahsedildiğinde sürekli aynı yerdeki daireyi vermesi gerekirken, ortanca veya en alt noktanın x'e olan yakınlığı yüzünden sıralama her resimde değişmektedir. Bizim ihtiyacımız olan ise x ve y'si en küçük olan daireyi bulmaktır (Şekil 3.5' de kırmızı yazılı olan sıralama ile gösterilmiştir).

Şekil 3.5. Daireler arasındaki sıralamanın gösterilmesi

Öncelikle x ve y koordinatların toplamına göre sıralamaya çalışılmıştır. Bu yöntemde resim yukarıda olduğu gibi dikdörtgen iken her nokta arasında belirgin mesafeler olduğundan dolayı sıralamayı doğru yapsa da, resim kare olacağından dolayı hesaplanmak istenen 2 ve 4. Dairelerin toplamlarının birbirine çok yakın çıkması ve bazen numaralandırmada belirsizlik olması ihtimalinden dolayı vazgeçilmiştir.

Diğer bir yöntemde ise daireler x koordinatlarına göre sıralanmışken diziyi 3, 2, 3 elemanlı sayılara ayırarak yeni diziler elde edip, y koordinatlarına göre sıralayarak hesaplamaktır. Bu işlemi 8 noktaya uygulamak istendiğinde şu şema göre göz önüne alınabilir;



Şekil 3.6. Programda kullanılacak dizinin şeması

Her biri dizi iki boyutlu olup dairelerin x ye koordinatlarını içermektedir. Her dizi grubu da kendi içinde y koordinatlarına göre sıralandığında örnek rakamlarla şu şekilde doğru sıralanış elde edildiği gösterilebilir: Soldaki tablo (Tablo3.1.) parçalama işleminden önce x koordinatlarına göre oluşturulan dizi sıralaması iken, sağdaki tablo (Tablo 3.2.) parçalara ayrıldıktan sonraki halinin sıralamasıdır.

Tablo 3.1. X değerine göre sıralamak

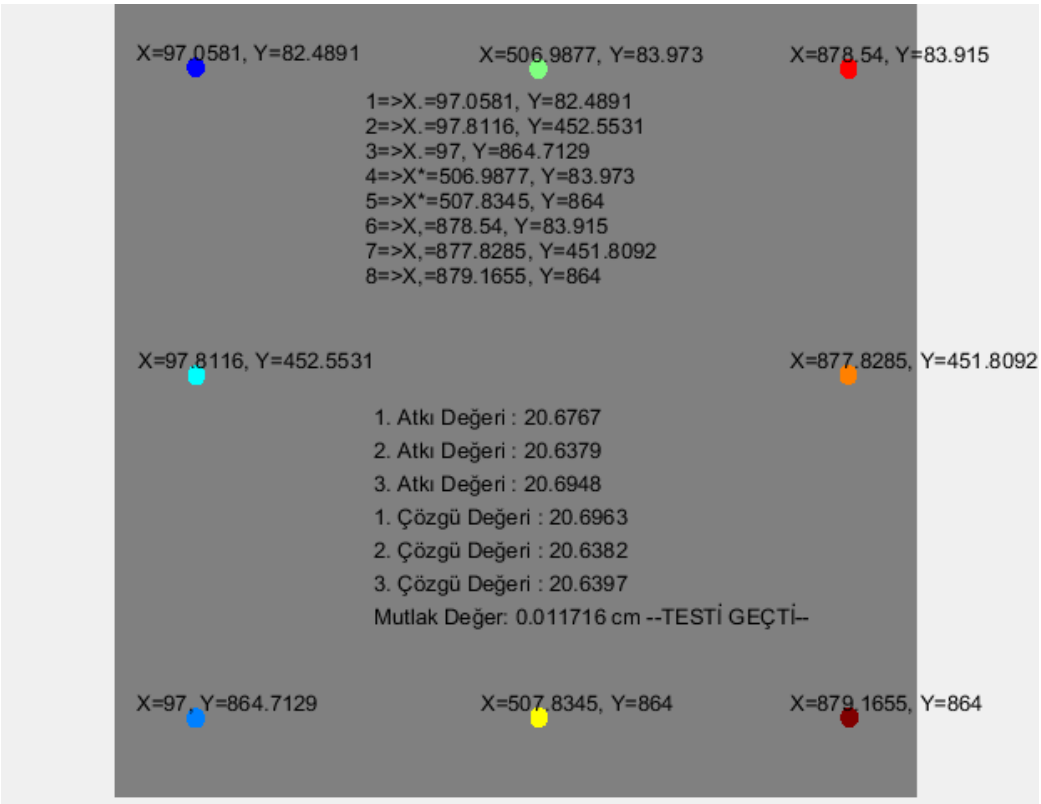
Sıra	X Koordinat	Ykoordinat
1	97,0581	82,4891
2	97	864,7129
3	97,8116	452,5531
4	506,9877	83,973
5	507,8345	864
6	877,8285	451,8092
7	878,54	83,915
8	879,1655	864

Tablo 3.2. Diziyi parçalara ayırarak sıralamak

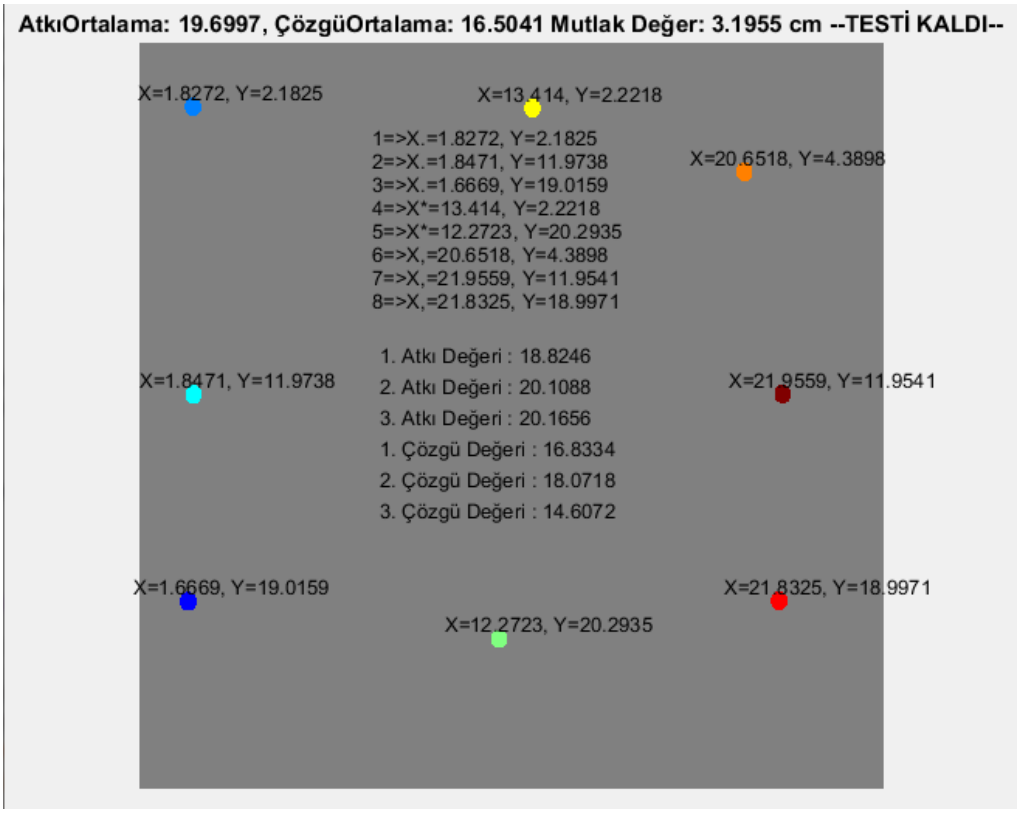
Sıra	X Koordinat	Ykoordinat
1	97,0581	82,4891
2	97,8116	452,7129
3	97	864,5531
4	506,9877	83,973
5	507,8345	864
6	878,54	83,915
7	877,8285	451,8092
8	879,1655	864

Bu elde edilen değerler sonucunda atkı ve çözgü olarak isimlendirilen değerler bulunur. Gerçek test yapılırken atkı değerleri en sağdaki üç noktadan kendilerine karşılık gelen en soldaki üç noktanın çıkartılmasıyla elde edilir, çözgü değerleri ise 90 derece sağa döndürülen kumaşın yine en sağdaki üç noktadan en soldaki üç noktanın çıkarılması işlemi uygulanarak bulunur. Bu işlemler koda dökülürken; atkı değerleri için, "diziuc" isimli diziden "dizibir" isimli dizinin birbirlerine karşılık gelen x değerleri çıkartılırken, çözgü değerleri için, her dizinin son elemanının y değerinden, ilk elemanının y değeri çıkartılarak sonuc bulunur. Bu farklar piksel-cm dönüşümü yapılarak ekrana yazdırılır. (1piksel = 0,02645833 cm)

Test formunda yer alan atkı ve çözgü değerlerinin ortalamaları ve birbirlerinden çıkartılarak elde edilen mutlak değer işlemi sonucunda eğer 3 cm'den küçük bir rakam elde edilmişse kumaş testi geçmiştir demektir, değilse kumaş testi geçemez.

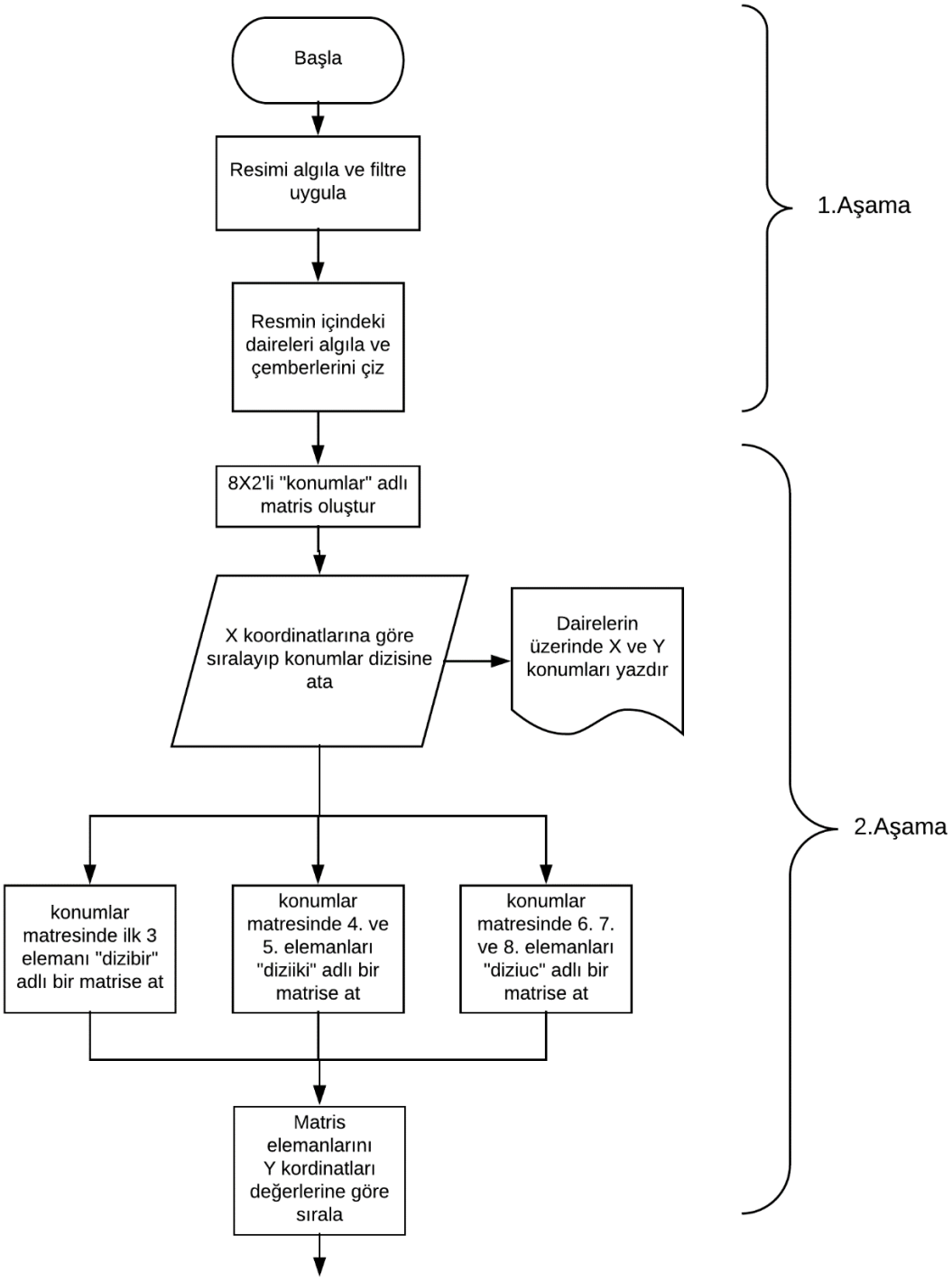


Şekil 3.7. Örnek ekran çıktısı- Testi Geçer

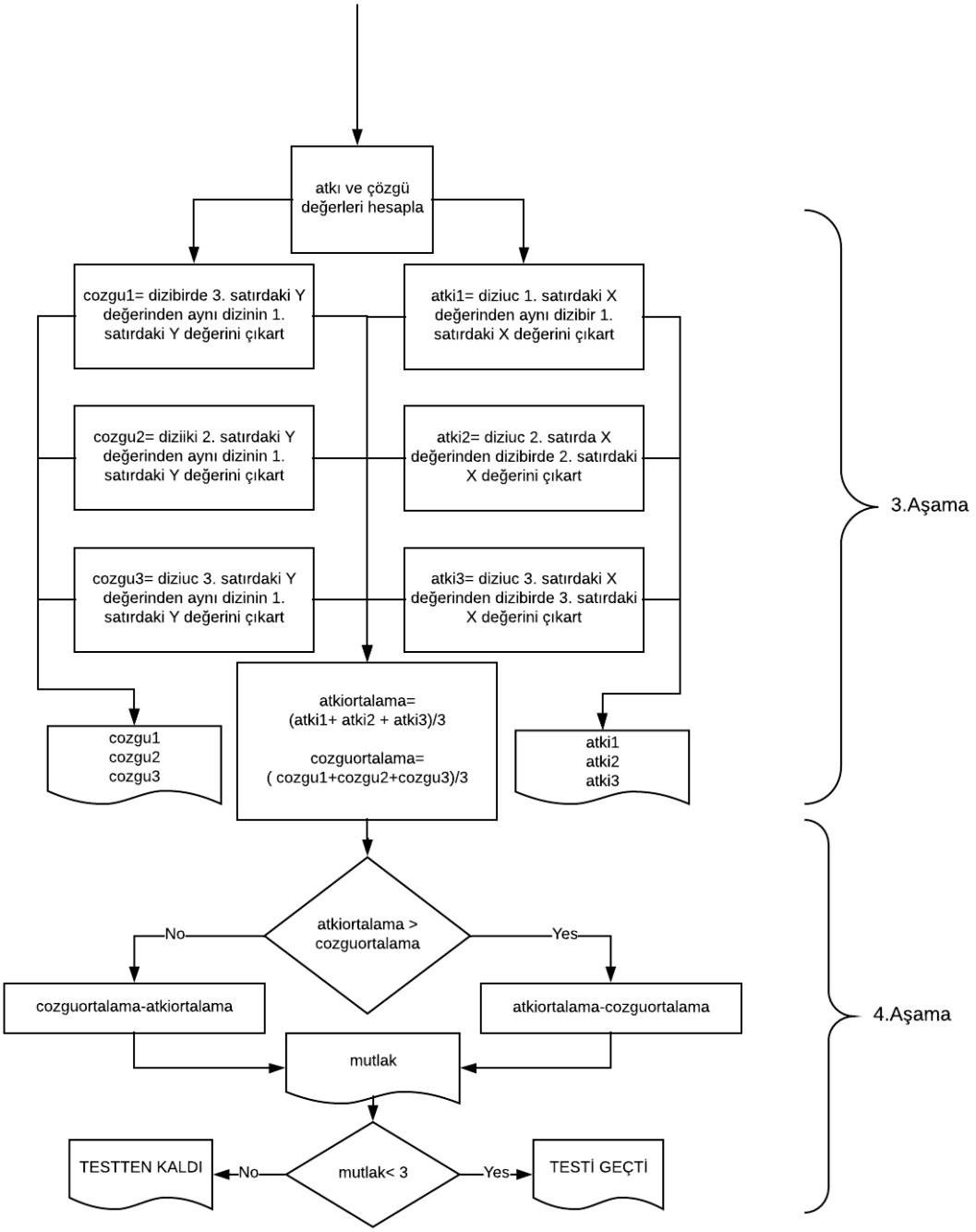


Şekil 3.8. Örnek ekran çıktısı- Testten Kalır

3.3. Akış Diyagramı ve Matlab kodu



Şekil 3.9. Matlab kodunun akış diyagramı -ilk kısım



Şekil 3.10. Matlab kodunun akış diyagramı ikinci kısım

1. Aşama:

Program çalıştırılır ve kullancıdan resim alınır. Alınan resim üzerindeki daireleri tespit etmek ve merkez koordinatlarını daha doğru bir şekilde bulunması amacıyla filtre ve morfolojik işlemler (gri filtre, bitwise, imclose, strel, imfill vs.) uygulanıp daireler bulunur. Dairlerin etrafı çizilip daha belirgin hale getirilir.

2. Aşama:
Kullancıdan alınan her resimde dairelerin konumları belli bir sıralaması olması gerekmektedir. Resimde bulunan dairelerin merkez koordinatlarını tespit edip hem X hem Y koordinatı en küçük olandan başlayarak sıralama yapılmalıdır.

8x2 "konumlar" matrisi oluşturuluğunda X eksenine göre sıralı haldedir. Y koordinatı birbirine yakın olan matris elemanlarının sıralarının sabit olmasından emin olmak için Y koordinatına göre de sıralama yapılmalıdır. Fakat X'e göre sıralamayı bozmamak için "konumlar" matrisin ilk üç elemanı "dizibir", dördüncü ve beşinci elemanları "diziiki", ve son üç elemanları da "diziuc" matrislerine atılır. Bu üç matris için Y koordinat değerlerine göre sıralama yaparak istenilen sıralama elde edilir.

3. Aşama:
Oluşturulan dizibir, diziiki, diziuc matrislerin aracılığıyla yatayda en sol ve en sağdaki dairelerin X koordinatların farklarını hesaplatarak atkı değerleri, dikeyde en üst ve en alttaki dairelerin Y koordinatların farklarını hesaplatarak çözgü değerleri hesaplanır. Ekrana yazdıktan sonra atkı ve çözgü değerlerini toplayıp 3'e bölerek ortalaması alınır.

4. Aşama:
Atkı ve çözgü değerlerinin ortalamalarından büyük olandan küçük olanı çıkartılıp mutlak değeri elde edilir. Mutlak değeri 3'ten küçük ise kumaş testten geçer büyük ise testten kalır. Testten kalan kumaşın kullanılmasına izin verilmez.

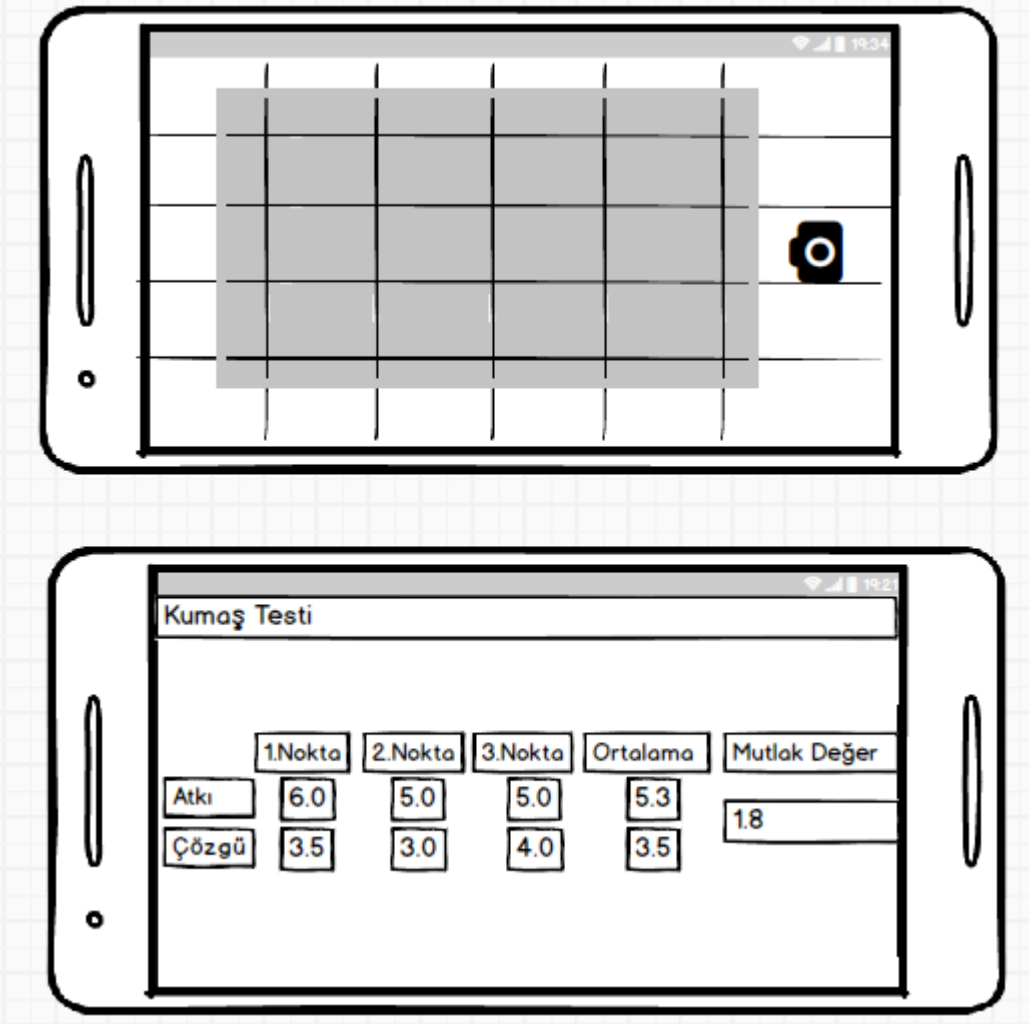
Uygulama kodu; EK A'da yer almaktadır.

3.4. Algoritmanın Gerçeklenme Süreci

Matlab ile yazılan görüntü işleme algoritması Java programlama dili kullanılarak Netbeans platformunda geliştirilecektir. Sebebi ise Java'nın neredeyse her platformda çalışabilen, bol kaynaklara sahip, güvenilir, güncel bir dil olmasıdır. Bu platformda yazılan kod başka platformlara geçiş yapma ihtimaline karşı test amaçlı yazılacaktır. Yani kod Netbeans üzerinde Open CV (Open Source Computer Vision Library) kütüphanesi kullanılarak yazılacak ve denemeler yapılacaktır. Çalışır duruma

geldiğinde birden fazla sistem (Android, Windows veya Linux vs. gibi sistemler) için üzerinde düzenlemeler yapılacak halde geliştirilecektir.

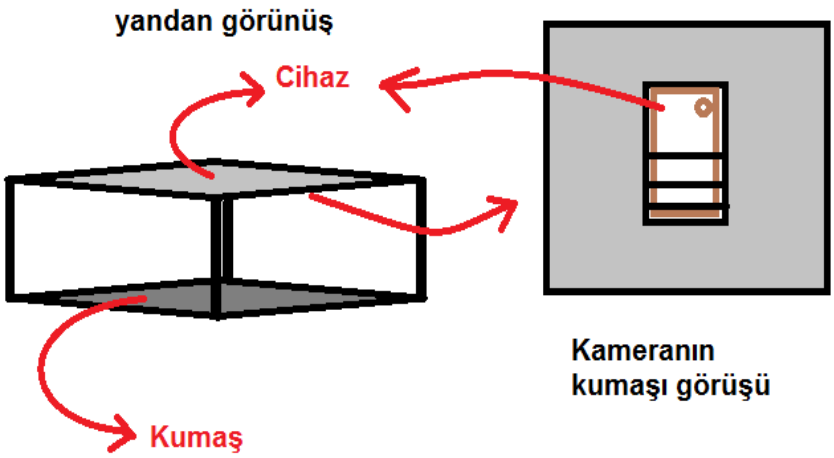
Netbeans platformunda Java dilinde yazılacak uygulama rahatlıkla Android işletim sistemine geçirebileceği için Android işletiminin uygun bir seçenek olacağı düşünülmüştür. Uygulama Android 4.0 da API 14 versiyonu için yazılmasıyla çoğu mobil cihazın uyum sağlayabileceği bir uygulaması yazılacaktır.



Şekil 3.11. Android Mockup görüntüsü

3.5. Projenin Geliştirilmesi

Hesaplama işlemlerinin doğruluğu, çekilen resmin çözünürlüğü ve fotoğrafı çekecek olan cihazın kumaşa olan konumu ile yakından ilgilidir. Eğer fotoğrafta kumaş herhangi şekilde yamuk veya kötü çıkarsa algılama işlemi veya sonuç buna bağlı olarak değişir. Testi uygulayacak kişilerin her biri fotoğrafı kendi çekmeye çalışırsa farklı konumlar, fotoğraf açıları, ışıklandırma, fotoğraf çözünürlüğü gibi sebeplerden test sonuçları hatalı çıkabilir veya hiç algılama yapamayabilir. Bu duruma çözüm olarak cihaz ile kumaş arasında mesafenin sabitlenmesi amacıyla bir aparat düşünüşmüştür.



Şekil 3.12. Temsili donanımsal aparat

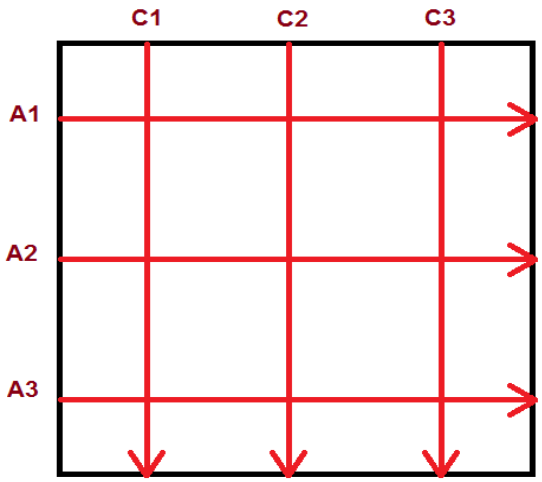
Aparatın alt kısmına konulan kumaş alt bölmede sabitlenmesiyle ve üste konulan cihaz (tablet, telefon vs.) fotoğrafın sabit aynı mesafeden çekilmesini sağlayacaktır. Ayrıca gerekli yerlere ışıklandırma yapılarak ortam ışığından bağımsız hale getirilebilir veya alt bölüme yerleştirilecek olan bant sistemi seri şekilde ölçüm işlemi yapılıp bir sonraki işleme geçişin hızlandırılması sağlanabilir.

Geliştirme sürecinde dikkat edilmesi gereken bazı yazılımsal konular:

- Kumaş arkaplanının çok açık renkli veya beyaz gibi daire tespitini zorlaştıracak renkler olması tasarım aşamasında göz ardı edilmiştir. Bu soruna karşılık Matlab'den sonraki aşamada java kullanıldığında renkler konusunda

kontroller yaptırılması, kullanılan filtrelerin tekrar gözden geçirilmesi gerektiği düşünülmüştür.

- Yazılan kodda kumaşın sabit bir renk olduğu varsayılmıştır. Fakat desen olarak daireler bulunabilir. Eğer ölçüm yaptığımız dairelerden daha büyük boyutlu daireler varsa göz ardı edilmesi kod ile sağlanabilir. Eğer desendeki daireler, şablon üzerinden konulan dairelerin boyutundaysa kod kendine göre seçeceği 8 noktaya göre yanlış sonuçlar üreten işlemler yapacaktır. Bu soruna karşılık ise daire tespitinin tamamen kaldırılması düşünülmüştür. Eğer kumaşın belirli aralıklarla bir kenarından diğer kenarına kadar olan mesafe ölçülürse atkı ve çözgü değerleri daha sağlıklı elde edilebilir.



Şekil 3.13. Kenarlar arası ölçüm

- Sonuçlar telefon ekranında göstermek kullanıcıların o değerler rapor üzerinde tekrar yazıp vakit kaybetmesi demek olabilir. Bunun yerine masaüstü programı haline getirilip çıktılar rapor halinde verilebilir.

BÖLÜM 4. SONUÇLAR

Kumaş üretimindeki en önemli kalite parametrelerinden biri boyutsal değişimdir. Bu çalışmada konfeksiyon işletmelerinde üretim sürecinde yardımcı olabilmek amacıyla bir görüntü işleme algoritması hazırlanmıştır. Bu algoritma ile görüntü işleme teknolojileri kullanarak şablon ile kumaş üzerine konulan noktalar tespit edilip, noktalar arası mesafe hesaplanarak kumaş boyutu üzerindeki değişimler hesaplanmıştır. Algoritmanın geliştirilmesiyle, boyutu değişen sorun oluşturabilecek kumaşlar, ölçü noktalarında yapılan hesaplama hataları ve diğer olası hatalar giderilmek istenmiş, hatasız hızlı ve verimli üretimin gerçekleşmesi adına bir adım atılmıştır.

KAYNAKLAR

[1] Tekstil Dershanesi, “Kumaş Performans Testleri”, link: http://www.boyteks.com/mattress-ticking/quality/testing-standards/, Erişim Tarihi: Ekim 2018

[2] Erol Kara, “Çekmezlik Testi”, link: http://www.erolkara.net/2015/09/sanfor-cekmezlikten-cektigimiz.html, Erişim Tarihi: Kasım 2018

[3] Feriha Demirhan ve Binnaz Meriç, “Örme ve Kumaş Giysilerde Yıkama ve Kurutma Sonrası Boyut Değişimlerinin İncelenmesi”, Sayfa:383-384 Aralık 2004

[4] Derya İşler, “Biçimsel (Morfolojik) Görüntü İşleme”, link: https://slideplayer.biz.tr/slide/10376908/, Erişim Tarihi: Kasım 2018

[5] Chris Solomon and Toby Breckon, “rgb2gray kodu kullanımı”, Fundamentals of Digital Image Processing, Temmuz 2011

[6] George Siogkas, Visual Media Processing Using Matlab Beginner's Guide , Sep 24, 2013

[7] Quanmin Zhu And Ahmad Taher Azar, “bwareaopen kodu kullanımı”, Complex System Modelling and Control Through Intelligent Soft Computations, Sayfa:755 ,Kasım 2014

[8] Matlab Dokumantation, “regionprop property”, Link: https://www.mathworks.com/help/images/ref/regionprops.html

[9] Aykut Gökdemir, “Matlab ile Görüntü İşleme”, Link: https://www.elektrikport.com/makale-detay/matlab-ile-goruntu-isleme-1-elektrikport-akademi/8196#ad-image-0 Erişim Tarihi: Ekim.2018

EK A. Matlab Kodu

```
RGB = imread('C:\Users\ZTRN\Desktop\mesafeHesapla.png');
I =    rgb2gray(RGB);
esik_degeri = graythresh(I);
bw = im2bw(I,esik_degeri);
bw = bwareaopen(bw, 50);
hold on

birlestirici = strel('square',3);
bw = imdilate(bw,birlestirici);
[B,L] = bwboundaries(bw,'noholes'); % nesneleri buluyoruz.

daireler = regionprops(L, 'Area', 'Centroid', 'Perimeter');
imshow(label2rgb(L, @jet, [.5 .5 .5]));
hold on;
konumlar=[0 0;0 0;0 0 ];
tocm=0.02645833;
for n=1:size(daireler,1)
    cent= daireler(n).Centroid;
    X=cent(1);
    Y=cent(2); %dairenin merkez koordinatlar
    if daireler(n).Area>5 %%nesnelere filtre yapmak icin
       text(X-70,Y-20,['X=',num2str(X*tocm),...
                       ', Y=',num2str(Y*tocm)]);
       konumlar(n,1)= X*tocm;
       konumlar(n,2)= Y*tocm;
     end
end

dizibir=[0 0;0 0;0 0];
diziiki=[0 0;0 0];
diziuc=[0 0;0 0;0 0];

for i=1:size(konumlar,1)
    for k=1:2
        if i<4
            dizibir(i,k) = konumlar(i,k);
         elseif i>5
            diziuc(i-5,k) = konumlar(i,k);
         else
            diziiki(i-3,k) = konumlar(i,k);
        end
    end
end
dizibir= sortrows(dizibir,2);
diziiki= sortrows(diziiki,2);
diziuc= sortrows(diziuc,2);
for i=1:size(konumlar,1)

        if i<4
           text(300,90+30*i,[num2str(i),'=>','X.=', ...
num2str(dizibir(i,1)),', Y=',num2str(dizibir(i,2))]);
```

```
        elseif i>5
            text(300,90+30*i,[num2str(i),'=>','X,=', ...
            num2str(diziuc(i-5,1)),', Y=',num2str(diziuc(i-5,2))]);
        else
            text(300,90+30*i,[num2str(i),'=>','X*=',num2str( ...
            diziiki(i-3,1)),', Y=',num2str(diziiki(i-3,2))]);
        end
end
hold on

atki1=(diziuc(1,1)-dizibir(1,1));
atki2=(diziuc(2,1)-dizibir(2,1));
atki3=(diziuc(3,1)-dizibir(3,1));
text(310,400,['1. Atkı Değeri : ',num2str(atki1)]);
text(310,440,['2. Atkı Değeri : ',num2str(atki2)]);
text(310,480,['3. Atkı Değeri : ',num2str(atki3)]);
cozgu1=(dizibir(3,2)-dizibir(1,2));
cozgu2=(diziiki(2,2)-diziiki(1,2));
cozgu3=(diziuc(3,2)-diziuc(1,2));
text(310,520,['1. Çözgü Değeri : ',num2str(cozgu1)]);
text(310,560,['2. Çözgü Değeri : ',num2str(cozgu2)]);
text(310,600,['3. Çözgü Değeri : ',num2str(cozgu3)]);

% atkı ve çözgü değerlerinin ortalaması ve mutlak değer işlemi
atkiort=(atki1+atki2+atki3)/3;
cozguort=(cozgu1+cozgu2+cozgu3)/3;
if atkiort > cozguort
    mutlak =atkiort-cozguort;
else
    mutlak=cozguort-atkiort;
end

 if mutlak>3
    title(['AtkıOrtalama: ',num2str(atkiort),', ÇözgüOrtalama: ',...
            num2str(cozguort),' Mutlak Değer: ', ...
            num2str(mutlak),' cm --TESTİ KALDI--']);
 else
    title(['AtkıOrtalama: ',num2str(atkiort),', ÇözgüOrtalama: ',...
            num2str(cozguort),' Mutlak Değer: ', ...
            num2str(mutlak),' cm --TESTTEN GEÇTİ--']);
 end
```