KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

RENK ANALİZİ TABANLI ÖNERİ SİSTEMLERİ İÇİN MOBİL UYGULAMA GELİŞTİRME

GÜREL YILDIZ

KOCAELİ 2012

KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

YÜKSEK LİSANS TEZİ

RENK ANALİZİ TABANLI ÖNERİ SİSTEMLERİ İÇİN MOBİL UYGULAMA GELİŞTİRME

GÜREL YILDIZ

Prof.Dr. Adnan KAVAK
Danışman, Kocaeli Üniv. .....

Doç.Dr. Hacı Ali MANTAR
Jüri Üyesi, GYTE .....

Yrd. Doç.Dr. Ahmet SAYAR
Jüri Üyesi, Kocaeli Üniv. .....

Tezin Savunulduğu Tarih: 18.07.2012

ÖNSÖZ VE TEŞEKKÜR

Günümüzde akıllı mobil cihazların geliştirilmesi bilgisayarlardan aldığımız birçok hizmeti cebimize taşımaktadır. Mobil cihazların hafıza, hız ve güç kaynağı kısıtlamalarından dolayı mobil uygulama geliştirme sürecinde kullanılacak yöntem ve algoritmalar önem kazanmaktadır. Bu çalışmada Türkiye' de önemli bir yeri olan mobilya ve dekorasyon sektörüne yönelik renk analizi tabanlı öneri sisteminden oluşan mobil uygulama gelişirtirilmiştir. Renk analizi işlemleri için skaler ve vektörel nicemleme yöntemleri kullanılarak android işletim sistemine sahip mobil cihazlarda hata oranları, zaman ve hafıza kullanma parametrelerine göre performansları karşılaştırılarak uygun yöntem uygulanmıştır. Mobil uygulamada seçilen renk kodu, en, boy ve yükseklik bilgilerine göre sunucudaki veriler ile doğrusal eşleştirme algoritması kullanarak öneriler oluşturulmuştur.

Yüksek lisans eğitimim ve tez çalışmam süresince büyük yardımları olan ve öğrencisi olmaktan onur duyduğum danışman hocam sayın Prof. Dr. Adnan KAVAK başta olmak üzere tüm hocalarıma ve çalışma arkadaşlarıma teşekkür ederim.

Ayrıca hayatım boyunca benden emeğini, sevgisini ve güvenini hiç esirgemeyen aileme ve eğitim sürecim boyunca hep yanımda olan, sevgisinden güç aldığım eşim Çiğdem YILDIZ'a içten teşekkürlerimi sunarım.

Temmuz – 2012 Gürel YILDIZ

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER DİZİNİ VE KISATLMALAR

dB : Decibel
MB : Megabyte
ms : Milisaniye
s : Saniye

Kısaltmalar

ADT : Android Development Tools (Android Geliştirme Araçları)
AIDL : Android Interface Definition Language (Android Arayüz Tanımlama Dili)
CPU : Central Processing Unit (Merkezi İşlem Birimi)
DDMS : Dalvik Debugging Monitor Server (Dalvik Hata Ayıklama Monitör Sunucusu)
DVM : Dalvik Virtual Machine (Dalvik Sanal Makinesi)
HSV : Hue Saturation Value (Renk tonu Doygunluk Parlaklık değeri)
JDK : Java Development Kit (Java Geliştirme Ortamı)
JIT : Just-In-Time (Tam Zamanında)
JSON : JavaScript Object Notation (JavaScript Nesne Notasyonu)
IDE : Integrated Development Environment (Bütünleşik Geliştirme Ortamı)
MSE : Mean Squared Error (Ortalama Karesel Hata)
PSNR : Peak Signal-to-Noise Ratio (Doruk Sinyal Gürültü Oranı)
RGB : Red, Green, Blue (Kırmızı, Yeşil, Mavi)
SCIL : Center for Industrial Studies (Endüstriyel Çalışma Merkezi)
SDK : Software Development Kit (Yazılım Geliştiştirme Ortamı)
SE : Software Environment (Yazılım Ortamı)

RENK ANALİZİ TABANLI ÖNERİ SİSTEMLERİ İÇİN MOBİL UYGULAMA GELİŞTİRME

ÖZET

Bu çalışmada ev, ofis gibi ortamların dekorasyonunu yaparken kullanıcılara yardımcı olmak amacıyla mobilya, boya ve halı gibi dekorasyon ürünlerinin seçimini sağlayacak android işletim sistemli mobil cihazlar için mobil uygulama gerçeklenmiştir. Mobil uygulama, kullanıcının mobil telefonu ile çektiği görüntünün renk nicemleme metodlarını kullanarak renk analizini yapmaktadır ve kullanıcının belirlediği kıstaslara göre sunucu tarafındaki veritabanında yer alan ürünlerden önerilerde bulunulmaktadır. Görüntü renk analizi için skaler ve vektör nicemleme metodlarının performansları mobil cihaz üzerinde hata oranı, zaman ve hafıza kullanma açısından incelenmiştir. Yapılan teorik karşılaştırmalar sonucunda skaler nicemleme yönteminin çalışma zamanı ve hafıza gereksinimi açısından mevcut mevcut android platform üzerinde gerçeklenebilir olduğuna karar verilmiştir. Skaler nicemleme yöntemi Android 4.0.2 için java dili kullanılarak kodlanmış ve çalışabildiği örnek uygulamalarla gösterilmiştir.

Anahtar Kelimeler: Android, Mobil Uygulama, Nicemleme Metodları, Renk Analizi

MOBILE APPLICATION DEVELOPMENT FOR COLOR ANALYSIS BASED PROPOSAL SYSTEM

ABSTRACT

In this study, a mobile application that assist user to select decoration products like furniture, paint and carpet has been implemented on mobile device with android operating system. The mobile application analyzes the image which is taken by mobile phone using the color quantization methods and offers products from database based on parameters defined by user. The performance of scalar and vector quantization methods for the analysis of color content of image has been made in terms of mean square error, time and memory occupancy. As a result of the theoretical comparisons, it was decided that scalar quantization method can be implemented on android platform in terms of time and memory requirements. Scalar quantization method was implemented by using java programming language for Android 4.0.2 operating system and it has been shown with sample application that the implementation can run properly.

Keywords: Android, Mobile Application, Quantization Methods, Color Analysis

GİRİŞ

Mobil telefon sistemleri ve mobil uygulamalara olan ilgi gün geçtikçe artmaktadır. Artık insanlar web üzerinden aldıkları hizmetleri mobil telefonlardan da alabilmek istemektedirler. Televizyon, gazete, web gibi mobil uygulamalarda artık şirketler için ürünlerini tanıtmada ve satmada yeni bir kanal olmuştur. Günümüzde mobil çağı yaşatan platformların başında Android gelmektedir. Android Google ve Open Handset Allience tarafından kodlanmış açık kaynak kodlu geliştirilmiş bir işletim sistemidir. Android işletim sisteminin açık kaynak kodlu olması farklı mobil cihaz markalarının cihazlarının kullanmasına sebep olmuş ve böylece android için geliştirilen uygulamaların çok sayıda kullanıcıya ulaşmasını sağlamaktadır [1].

Mobilya sanayi ve dekorasyon sektörü yarattığı katma değer ile ülkemizde stratejik bir önem taşımaktadır. CSIL 2008 verilerine göre 74 milyon nüfuslu Türkiye' de 5.636 milyon $ mobilya üretimi varken 5.184 milyon $ mobilya tüketimi olmuştur. MOBSAD 2008-yılı sektör raporuna göre Türkiye' de faaliyet gösteren toplam mobilya firma sayısı 64780 dir [2].

Günümüzde firmalar rekabet, tanıtım ve satış için web, mobil ve sosyal medya bağlantılı uygulamalara ciddi yatırımlar yapmaktadırlar [3]. Ülkemiz sanayisinde önemli yeri olan mobilya ve dekorasyon sektörünü incelediğimizde mobil ve web uygulama teknolojilerini yeterince kullanmadıkları gözlemlenmiştir. Bundan dolayı bu çalışmada mobilya ve dekorasyon sektörüne yönelik android platform üzerinde çalışan renk analizine dayalı ve kullanıcının belirlediği ilave parametreler göre kullanıcıya öneri yapan mobil uygulama geliştirilmiştir. Kullanıcıya ürün seçiminde yardımcı olacak öneri sisteminin bu işlemi en kısa sürede ve kullanıcının en fazla ilgisini çekebilecek nitelikte başarması gerekir. Bu amaçla hata oranı, zaman ve hafıza kullanımı gibi parametreler göz önünde bulundurularak renk analizi algoritmaları incelenmiş ve en uygun algoritma ile android işletim sistemli akıllı telefonlar için mobil uygulama geliştirilmiştir.

1. GENEL BİLGİLER

1.1. Tez Çalışmasının Amacı ve Başlatılma Sebepleri

Günümüzde insanların akıllı cihazlara hızlı adaptasyonu sonucunda mobil uygulamalarda şirketler için müşteriye ulaşmada yeni bir yol olmuştur. Akıllı cihazların sağladığı mobil uygulama özelliği sayesinde ortam bağımsız, hızlı ve kişi ile ilgili uygulamalar tasarlanabilmektedir [4]. Yapılan araştırmalar sonucunda mobilya ve dekorasyon sektörüne yönelik sınırlı sayıda uygulama olduğu gözlemlenmiştir. Bu tespitler sonucunda çalışmada, Türkiye' de önemli bir yeri olan mobilya ve dekorasyon sektörüne yönelik bir mobil uygulama gerçekleştirilmesi amaçlanmaktadır.

Bu çalışmada ev, ofis gibi ortamların dekorasyonunu yaparken kullanıcılara yardımcı olmak amacıyla mobilya, boya ve halı gibi dekorasyon ürünlerinin seçimini sağlayacak mobil uygulama gerçeklenmesi hedeflenmektedir. Çalışmada mobil uygulaması kullanıcının bulunduğu ortamdan bağımsız olarak yüklediği veya mobil telefonu ile çektiği resimdeki renk tonlarının analizini çıkartarak kullanıcının belirlediği kriterlere göre sistemde yer alan ürünlerden önerilerde bulunacaktır. Kullanıcı aldığı önerilerden arşivler oluşturabilecek ya da sosyal medya ve arkadaşları ile paylaşabilecektir. Ayrıca kullanıcıya sistemde yer alan ürünleri çeşitli filtreleme seçeneklerine göre inceleyebilme imkânı da sunulacaktır.

Proje sonucunda üretilen ürünün kullanıcılar için ortamdan bağımsız, kolay kullanılabilir, anında cevap alınabilir ve ihtiyaca göre sonuç vermesi amaçlanırken, mobilya ve dekorasyon sektöründeki firmalar için reklam ve tanıtım giderlerinin azaltılması satış ve müşteri ilişkilerinin arttırılmasını teknolojik yollar kullanarak amaçlanmaktadır.

1.2. Tez Çalışmasının Katkıları

Tez çalışmalarının katkıları dört ana başlık altında ifade edilebilir:

- Görüntü işleme konularında farklı skaler nicemleme ve vektör nicemleme yöntemleri incelenerek hata oranı, zaman ve hafıza kullanımı açısından renk analiz sonuçları karşılaştırılmıştır.
- Doğrusal eşleştirme algoritmasının mobil cihaz üzerindeki performansı incelenmiştir.
- Renk analiz algoritmaları ile android işletim sistemli mobil telefon üzerinde çalışan uygulama geliştirilmiştir.
- Geliştirilen uygulam gerçek görüntüler üzerinde test edilmiştir.

1.3. Literatür Taraması

Literatürde mobil platformlar ile ilgili çalışmalarda son yıllarda mobil cihaz sektöründe meydana gelen değişmelerden, android cihazların bu sektörün hangi bölümünde bulunduğunun kısa bir özeti verilmiştir ve android sitemler pek çok özelliği bakımından değerlendirilmiştir. İlk olarak android yazılım platform hakkında bilgi verilmiş, android sitemler güvenlik ve gizlilik bakımından diğer markalar ile karşılaştırılmış, android için geliştirilmiş olan uygulama geliştirici yazılımı değerlendirilmiş ve son olarakta uygulama geliştirinin teknolojiyi geliştirmede yaratıcılığı geliştirmek için kullanılabilecek yönlerinden bahsedilmiştir [1].

Orchard ve Bouman [5] bir görüntü üzerinde yer alan sınırlı sayıda rengin seçimi için kullanılacak nicemleme algoritmaları incelenmiş ve sonrasında da nicemleme işlemi için bir algoritma önerilmiştir. Çalışmada nicemleme işlemi sırasında karşılaşılan iki temel sorundan bahsedilmiştir. Bu sorunlardan ilki görüntünün doğal renkleri ile uyumlu olacak en uygun renk paletinin seçilmesi ve sonrasında paletteki her rengin resim içerisinde yer alan uygun piksellerle eşleştirilmesi işlemidir.

Renk paletini oluşturmak için hiyerarşik ağaç yapısı kullanılmıştır. Oluşturulan algoritma sonrasında görüntü kalitesi yüksek resimler elde edilmiş ve daha önce kullanılan algoritmalar göre yeni üretilen algoritmada daha az işlem ihtiyacı olduğu gözlemlenmiştir.

Bloomberg [6] RGB görüntüleri için renk nicemlemesi yapan octree veri yapısı kullanmıştır. Octree algoritması genel olarak renk uzayını bölmek için iyi bir yöntemdir. Ters bir indeks tablosundan indeksleme yaparak hızlı bir şekilde pikseli elde etmeyi sağlar. Çalışmada octree algoritması ile renk alanını bölümleme işlemi yapılırken kullanılanılabilecek dört farklı yöntemden bahsedilmiş. Bir-geçişli algoritma titremesiz, bir-geçişli algoritma titremeli, iki-geçişli algoritma titremesiz ve iki-geçişli algoritma titremeli olarak görüntülere uygulayarak sonuçlarını karşılaştırmıştır. Sonuçta iki-geçişli algoritma titremeli olarak daha iyi sonuç verdiğini göstermiştir.

Octree renk nicemlemesi ile ilgili başka bir çalışmada octree algoritmasının performansını arttırmak için algoritma üzerinde yapılan değişiklikler incelenmiştir. Buradaki amaç işlem zamanını kısaltmak ve ihtiyaç duyulan büyük hafıza alanlarını indirgemektir. İlk olarak nicemleme işlemi sırasında olşturulacak ağaç sayısının derinliği 4 ile sınırlandırılmış sonrasında da kullanılacak budama algoritması değiştirilerek hız kazanılmaya çalışılmıştır. Bu işlemler sonrasında olışturulan algoritma ile ortalama işlem süresi 87.36 milisaniyeden 14.43 milisaniyeye düşmüştür [7].

Literatürde Median-Cut algoritması ile ilgili çalışmada Median-Cut algortiması geliştirilmiş ve algoritmanın diğer renk nicemleme algoritmalarından olan octree ve neural network algoritmalarına göre daha kısa sürede sonuç verdiği gösterilmiştir. Median cut algoritmasının temeli resmi kutulara bölmeye dayanmaktadır. RGB uzayında yer alan renk histogramını hesaplamak için geniş bir belleğe ihtiyaç duymaktadır. Belleğe kaydetmek yerine bit odaklı kesim işlemi yapılması tavsiye edilmiştir. Kutulara bölme işlemini gerçekleştirirken kullanılabilecek pek çok kesme yöntemi bulunmaktadır. Klasik olarak en uzun R, G, B yoğunluğuna sahip uzunluk seçilirken çalışmada farklı bir matematik hesaplama yöntemi geliştirilerek hesaplama

yapılmıştır. Kutulama işlemi sonrasında elde edilen matrislere yakın renk değerlerinin hesaplanması yerine oluşturulmuş olan histogramdaki renk değerleri atanmıştır [8].

1.4. Tez Yapısı

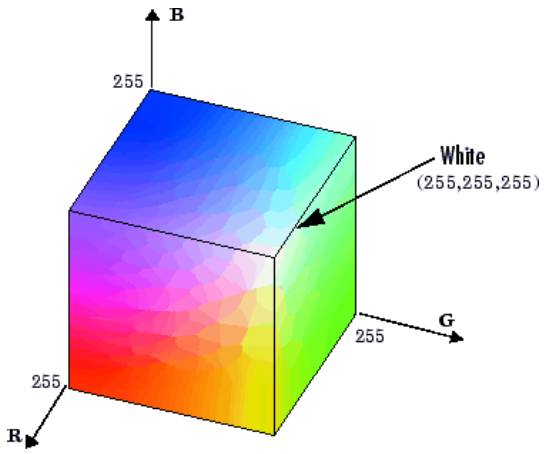
Tez çalışmaları, dört ana bölümde oluşmaktadır. Bölüm 1'de resim renk analiz yöntemleri hakkında genel bilgiler verilmekte, performans karşılaştırmaları yapılmaktadır. Bölüm 2'te mobil uygulama geliştirme için kullanılan yazılım ve donanım ile ilgili bilgiler yer almaktadır. Bölüm 3'te ise gerçekleştirilen renk analiz tabanlı mobil uygulama anlatılmaktadır. Son bölümde tez çalışmasının sonuçları hakkında bilgi yer almaktadır.

2. RENK ANALİZ YÖNTEMLERİ

2.1. Giriş

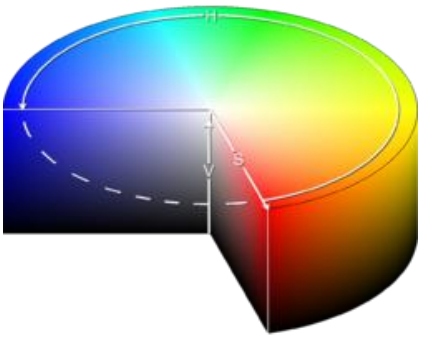
Bilgisayarların yavaş ve hafızanın pahalı olduğu bilgisayar teknolojinin ilk zamanlarında, bilgisayarların renkli görüntüleri göstermesi zordu çünkü görüntünün her bir pixel renginin bilgisayar hafızasında depolanması gereklidir ama o zamanlar büyük miktar renge sahip görüntüleri arabellekleyecek ekran kartları mevcut değildi. Bundan dolayı zaman içerisinde görüntü içerisindeki renk sayılarını azaltan renk nicemleme işlemleri geliştirilmiştir. Teknolojinin ilerlemesi sonucunda günümüzde akıllı mobil telefonlar için uygulamalar geliştirilebilmektedir. Mobil cihazların sınırlı hafıza, işlemci hızı ve güç kaynağına sahip olmasından dolayı görüntü işleme içeren uygulamalarda renk görüntü nicemleme yöntemlerine ihtiyaç duyulmaktadır [9].

Görüntülerde her renk kırmızı(R), yeşil(G) ve mavi(B) renk değerlerinin karışımından oluşmaktadır. Her renk değeri 8 bit ile tanımlanır ve değeri 0 ile 255 arasında değişir. Bir RGB rengi 24 bittir. RGB renk küpü, bir veri tipi için tanımlanan renklerin hepsinin üç boyutlu dizisidir. Her bir renk yüzeyi (red, blue ve green) için 256 değer olan renk küpünde 224 (ya da 16.777.216) renk tanımlıdır.



Şekil 2.1. RGB renk küpü

Görüntü pixel renklerinin RGB den başka bir diğer tanımlanma şekli HSV dir. Şekil 2.2 deki gibi geometrik olarak silindir şeklinde renk uzayıdır. HSV renk uzayı renk özü, doygunluk ve değer olarak tanımlanır. Renk özü, rengin baskın dalga uzunluğunu belirler. Açısal olarak 0° ile 360° arasında değişir. Doygunluk, rengin canlılığını belirler. 0 ile 100 arasında değişir ve düşük ve yüksek doygunluğa göre rengin griliği değişir. Parlaklık rengin içindeki beyaz oranını belirler ve değeri 0 ile 100 arasından değişir.



Şekil 2.2. HSV renk silindiri

Büyük miktar renge sahip indexli görüntüler problemlere sebep olabilir. Genelde indexli görüntüleri 256 renge sınırlamak gerekebilir. 8-bit görüntülü sistemlerde, 256 renkden fazla renge sahip görüntüler dithered(renk azaltma işlemi sonucunda azalan detayları düzeltme işlemidir) edilmelidir ve bundan dolayı görüntü düzgün görülmez. Bununla beraber bazı platformlarda, renk haritası 256 girdiyi geçmez. Cihazlarda görüntüyü gösterebilmek için görüntünün renk dizisi cihazın renk dizisinden büyük olamaz. Bundan dolayı resimler 8 bite nicemlenir [9].

Bu çalışmada mobil platform üzerinde renk analizi işlemleri için görüntü boyutunu yeniden boyutlandırma ve renk nicemleme yöntemleri uygulanmıştır. Renk nicemleme yöntemleri skaler nicemleme (scalar quantization) ve vektör nicemleme (vector quantization) yöntemleri olarak ikiye ayrılmaktadır.

2.2. Skaler Nicemleme

Skaler niceleme işleminde görüntünün her bir pixelinin renk değeri bir skaler değere bölünür ve çıkan sonuca yuvarlama işlemi uygulanır. Bu işlem sırasında renk değerleri skaler değerin katlarından hangisine yakınsa o değere atanır böylece görüntünün renk sayısı azaltılmış olur. Skaler değer 51 için 0 ile 255 arasında değişen RGB renk kodlarının değerleri 0, 51, 102, 153, 204 ve 255 değerlerine dönüştürülür. C değerleri m farklı renge sahip orijinal görüntünün renk kodu değerleri ve C' değerleri skaler nicemleme sonucu oluşan n farklı renge sahip görüntünün renk kodu değerleridir.

Orjinal görüntü: m farklı renk

$$C=\{C_1,C_2,\ldots,C_m\} \tag{2.1}$$

Görüntü kalitesindeki değişimi minimum tutarak n farklı renge azaltma:

$$C'=\{C'_1,C'_2,\ldots,C'_n\} \tag{2.2}$$

Her $\mathbf{C_i'}$ elemanını orjinal resimdeki $\mathbf{C_i}$ değeri ile değiştiriyoruz:

$$C'_i \leftarrow \left\lfloor C_i . \frac{n}{m} \right\rfloor \tag{2.3}$$

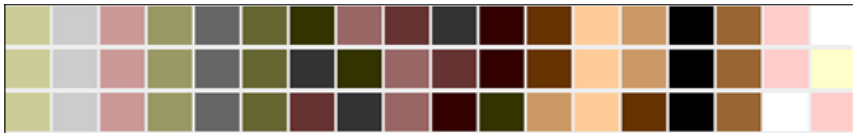
Şekil 2.3. Sistem analizi için yüklenen görüntü

Tablo 2.1. Görüntünün özellikleri

YÜKSEKLİK (Pixel)	GENİŞLİK (Pixel)	TOPLAM RENK SAYISI	TEK RENK SAYISI
172	239	41108	25354

Tablo 2.1'de skaler nicemleme metodu uygulanacak olan Şekil 2.3'deki görüntünün yükseklik, genişlik, toplam renk sayısı ve tek renks sayısı bilgileri verilmiştir. Toplam renk sayısı görüntüdeki toplam pixel renk sayısıdır. Tek renk sayısı toplam pixel renk sayısı içinde farklı renk sayısıdır.

Tablo 2.2. Görüntünün 51 skaler değerine göre renk analizi

TANESELLİK	RENK NİCEMLEMESİ (Skaler Değer=51)
4	
8	
16	

Tablo 2.3. Görüntünün 17 skaler değerine göre renk analizi

TANESELLİK	RENK NİCEMLEMESİ (Skaler Değer=17)
4	

Tablo 2.4. Görüntünün 5 skaler değerine göre renk analizi

TANESELLİK	RENK NİCEMLEMESİ (Skaler Değer=5)
4	

Şekil 2.3'deki örnek resim üzerinde MATLAB kullanarak uygulanan skaler niceleme işleminin analiz sonuçları Tablo 2.2, Tablo 2.3 ve Tablo 2.4'de gösterilmektedir. Tablo 2.2'de, resim pixel renk R, G ve B değerleri 51'e bölünerek tamsayıya yuvarlanmıştır. Bu işlem sonucunda 8'er bit olan R, G ve B değerleri 3 bit'e düşürülmektedir. R, G ve B için 2^8= 256 durum varken, $2^6 > 51 > 2^5$ değerine bölerek 2^3 duruma düşürülür.

Resim piksellerinin taranması işleminde örnek olarak 4, 8 ve 16 lık tanesellik taraması seçilmiştir. Tanesellik, nicemleme işlemi algoritmasının uygulandığı pixellerin atlama değeridir. Böylece 4 tanesellik değerinde 8 tanesellik değerine göre görüntünün daha çok pixeline algoritma işlemleri uygulanmış olur. Nicemleme işleminden sonra resmin yeni pixel renk değerleri sayılarak en yoğun renkten en az renge doğru bir renk histogramı oluşturulur. Renkler tablolarda soldan sağa en çok

yoğun renkten en az yoğun renge doğru sıralanmıştır. Analiz işlemi sırasındaki amaç analiz süresini azaltarak yoğun renkleri elde etmektir.

MSE hesaplama:

$$MSE = \frac{1}{m} . \sum_{j=1}^{m} (C_j^{'} - C_j)^2 \qquad (2.4)$$

Tablo 2.5. Görüntünün skaler nicelemeye göre hata ve zamana bağlı sonuçları

SKALER DEĞER	TANESELLİK DEĞERİ	MSE	PSNR (dB)	ZAMAN (s)
	4	0.189	55.353	0.010217
5	8	0.050	61.127	0.000250
	16	0.013	66.718	0.000067
	4	2.251	44.606	0.000952
17	8	0.568	50.580	0.000250
	16	0.138	56.728	0.000067
	4	14.515	36.512	0.000952
51	8	3.673	42.480	0.000249
	16	0.926	48.462	0.000067

Orijinal görüntü skaler nicemleme işlemine 5, 17 ve 51 skaler değerleri kullanarak uygulanmıştır. Algoritmanın işlemi sırasında görüntünün pixellerinde 4, 8 ve 16 tanesellik değerine göre iterasyon yapılmıştır. Nicemleme işlemi sonucunda orijinal görüntü ile nicemleme uygulanmış görüntü arasındaki MSE, PSNR ve nicemleme işlemi zamanı hesaplanmıştır. MSE Eşitlik (2.4) ile hesaplanır. Tablo 2.5'e göre tanesellik değeri düşünüldüğünde küçük skaler değer daha büyük PSNR değerinde sonuçlanır. Skaler değer sabitlendiğinde büyük tanesellik değeri seçilmesi daha büyük PSNR değeri vermektedir. Tanesellik değeri arttıkça işlem süresi azalmaktadır.

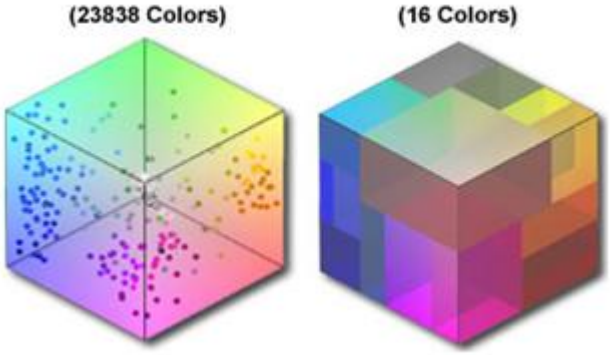
2.3. Vektör Nicemleme

Vektör nicemleme, skaler nicemleme gibi her bir pixel renk değeri olarak değil renk değerlerinin oluşturuduğu vektörler gibi davranır. Bundan dolayı görüntü renk sayısı

azaltma işlemleri sonucunda görüntüdeki bozulmalar skaler nicemlemeye göre daha azdır [10]. Bu çalışmada median-cut ve octree algoritmaları incelenmiştir.

2.3.1. Median-Cut Algoritmasi

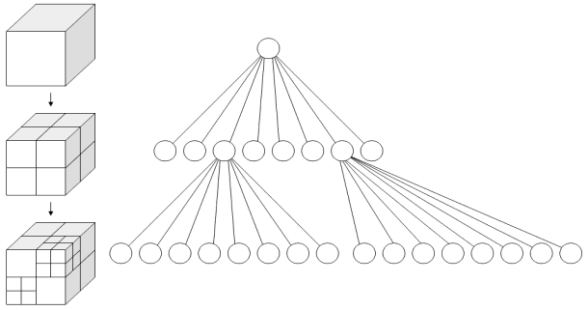
RGB renk küpünde görüntüdeki tüm renkleri içeren en küçük hücre seçilir. Hücrenin uzun olan ekseni boyunca renkleri küçükten büyüğe doğru sıralanır. Renkleri sıraladıktan sonra hücreyi uzun ekseni boyunca ortadan iki parçaya bölünür. Yukarıdaki işlemi renk uzayını istediğiniz sayıya düşürene kadar devam edilir.



Şekil 2.4. RGB renk küpü ve median-cut uygulanmış renk küpü

2.3.2. Octree Algoritması

Renkler veri yapısı ağacında sıralanır. Dallanma faktörü olarak 8 kullanılır. En fazla K düğümde farklı renk içeren ağaç yapısı oluşturulması hedeflenir. Ağaç dallarına yeni renk eklenirken en benzer olanıyla ortalaması alınarak yeni düğüm oluşturulur [4]. Octree algoritması iyi sonuç verir ama çok hafıza ve zaman tüketir.



Şekil 2.5. Octree ağaç yapısı

2.4. Skaler ve Vektör Nicemleme Karşılaştırması

Şekil 2.6. Nicemleme uygulanacak örnek görüntü-1

Tablo 2.6. Görüntü-1'in özellikleri

YÜKSEKLİK (Pixel)	GENİŞLİK (Pixel)	TOPLAM RENK SAYISI	TEK RENK SAYISI
216	297	64152	35865

Şekil 2.7. Nicemleme uygulanacak örnek görüntü-2

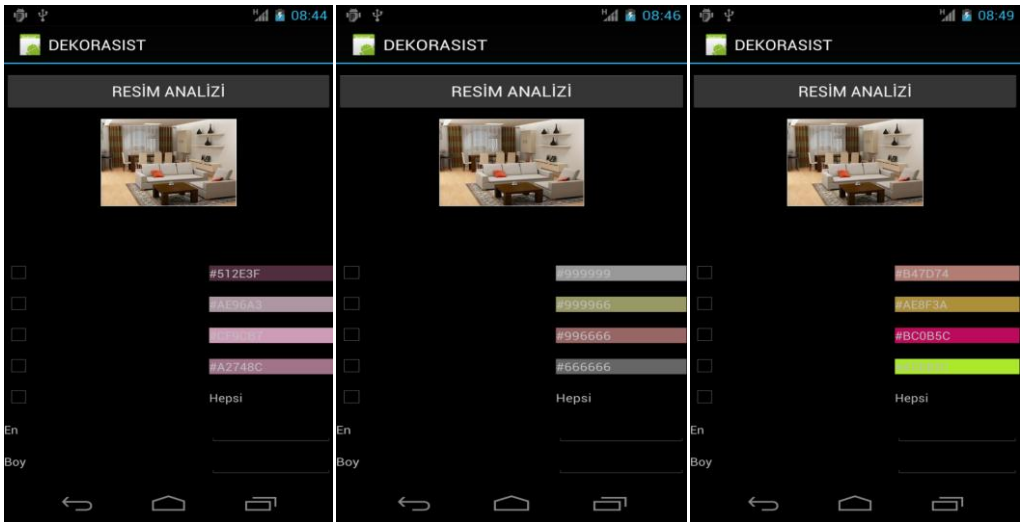
Tablo 2.7. Görüntü-2'nin özellikleri

YÜKSEKLİK (Pixel)	GENİŞLİK (Pixel)	TOPLAM RENK SAYISI	TEK RENK SAYISI
216	289	62424	18192

Skaler nicemleme, median-cut nicemleme ve octree nicemleme algoritmalarının mobil cihaz üzerindeki performansları şekil 2.6 ve şekil 2.7 deki görüntülerle test edilmiştir. Test sonucunda orijinal görüntü ile nicemleme algoritması uygulanmış görüntü arasındaki MSE, zaman(algoritmanın nicemleme işlem süresi) ve hafıza

kullanımı değerleri elde edilmiştir. Testler için mobil cihaz olarak Android 4.0.2 işletim sistemli Samsung Nexus marka mobil cihazda java dili ile programlanmış uygulama ortamı kullanılmıştır. Uygulamanın java kodu Ek-A'da verilmiştir.

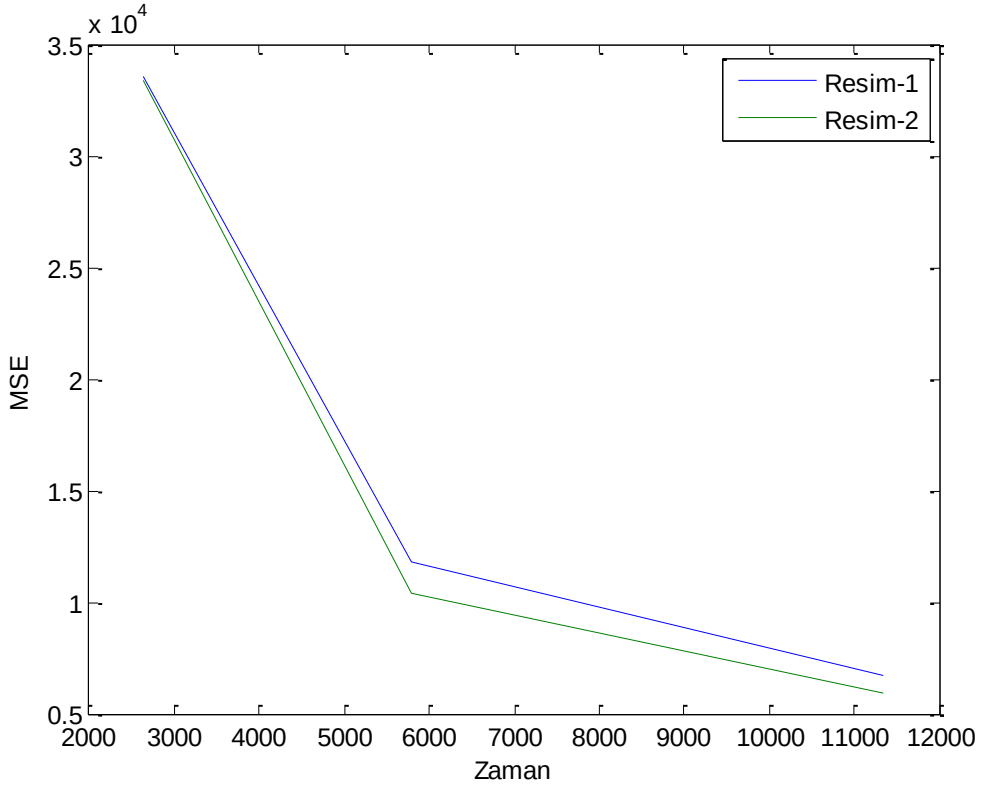
Şekil 2.7'de median-cut, skaler ve octree algoritmalarının mobil uygulama üzerindeki renk analizi sonuçları görülmektedir. Mobil uygulamada görüntünün pixel renklerine algoritmalar uygulanmış ve elde edilen görüntünün renkleri en çok yoğunluklarına göre sıralanarak renk kodları uygulamada görsel olarak gösterilmiştir. Mobil telefonlarda renk analiz işlemlerinde hafıza kullanımı ve performans için optimum algoritmanın belirlenmesi önemlidir [11]. Bundan dolayı algoritmaların farklı parametrelere göre test işlemleri yapılmıştır.



Şekil 2.8. Median-cut, skaler ve octree algoritmalarının renk analiz sonuçları

Tablo 2.8. Nicemleme algoritmalarının mobil cihazdaki test sonuçları

		MSE	Zaman(ms)	Hafıza Kullanımı (MB)
Skaler Nicemleme	**Görüntü-1**	33515,14	2625	16072
	Görüntü-2	33404,22	1761	15317
Median-cut Nicemleme	**Görüntü-1**	11825,23	5786	30118
	Görüntü-2	10427,46	3676	29787
Octree Nicemleme	**Görüntü-1**	6676,47	11347	21120
	Görüntü-2	5943,52	6412	18580



Şekil 2.9. Görüntülerin MSE ve zaman grafiği

Tablo 2.8'de test sonuçlarına göre mobil cihaz üzerinde en iyi nicemleme sonuçları için daha çok yürütüm süresi ve hafıza gerektiği görülüyor. Bu da nicemleme algoritması kullanılarak geliştirilecek olan mobil uygulamada performans açısından başarısız olmaktadır. Octree ve median-cut algoritmaları görüntü renk nicemlemesinde iyi sonuçlar çıkartmasına rağmen fazla hafıza ve çalışma zamanı gerektirmektedir. Geliştirilecek olan mobil uygulamada görüntü kalitesinden çok görüntüdeki yoğun olan ana renklerin elde edilmesi daha önemli olduğu için zaman

ve hafıza açısından daha verimli olan skaler nicemlemenin kullanılması tercih edilmiştir. Skaler nicemlemenin mobil uygulama için bir diğer seçilme sebebi mobil uygulamada skaler nicemleme algoritması uygularken farklı skaler değerler kullanarak ana renkler ve ana renklerin tonlarının çıkartılmasını daha az sürede sağlamasıdır.

3. MOBİL UYGULAMA GELİŞTİRME

3.1. Mobil Uygulama Platformu

Günümüzde mobil işletim sistemlerinin market payları artarken daha fazla mobil cihazlarda çalışan uygulamalar geliştirilmektedir [12]. Tez çalışmasında mobil uygulama platformu olarak Android işletim sistemli mobil cihaz kullanılmıştır. Uygulamanın geliştirme aşamaları Android emulatör üzerinde gerçeklenmiştir.

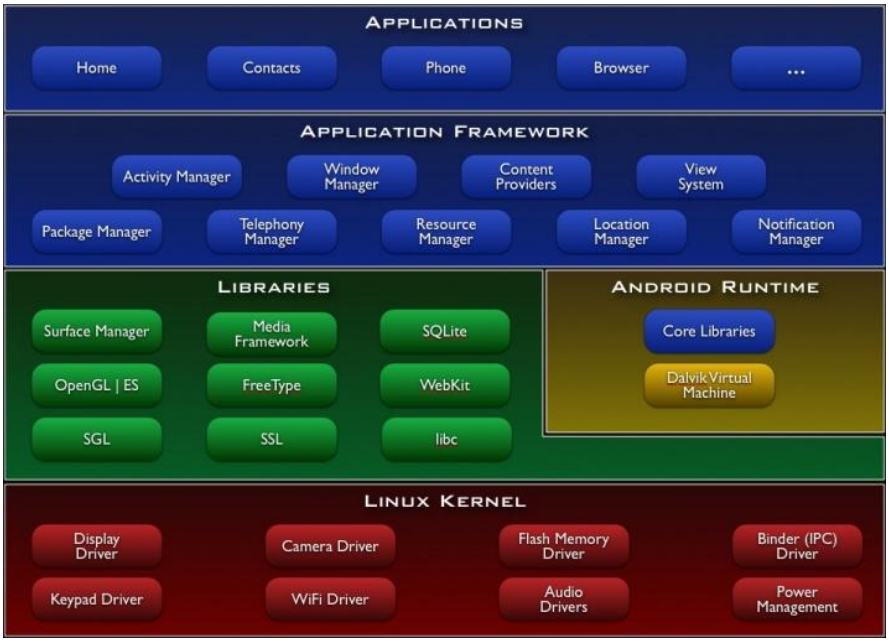
3.1.1 Android İşletim Sistemi

Android Google ve Open Handset Allience[13] tarafından kodlanmış linux işletim sistemi tabanlı mobil cihazlar için tasarlanmış açık kaynak kodlu geliştirilmiş bir işletim sistemidir. Android 2003 yılında Silikon Vadisinde Andy Rubin, Rick Miner, Nick Sears ve Chris White bir araya gelerek kurdukları ANDROID INC. adında bir şirkettir. Android Inc.'in kurucuları Danger, Wildfire Communication, T-Mobile, WebTV gibi şirketlerden gelmektedir. Bu şirket 2005 yılının Temmuz ayında Google tarafından satın alınmıştır ve Google bu kişileri Android takımı altında toplayarak 2007'nin Kasım ayında Open Handset Allience kurarak bugünün en çok tercih edilen mobil işletim sisteminin geliştirilmesine öncülük etmiştir [14].

Şekil 3.1. Android bileşenleri

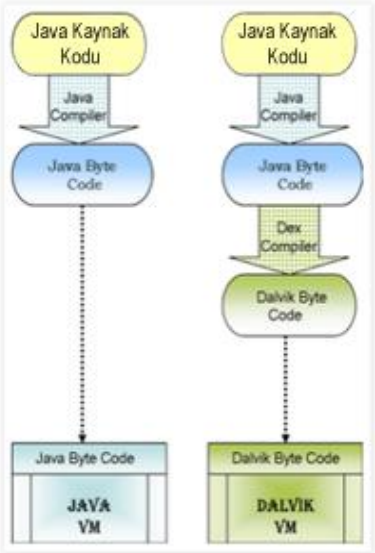
3.1.2 Android Mimarisi

Android işletim sistemi Şekil 3.1'de görüldüğü gibi Linux Kernel ve JIT derleme özellikli Dalvik sanal makinesi üstünde çalışan çekirdek kütüphanelerin üstündeki Java tabanlı, nesneye dayalı uygulama sistemi üzerinde çalışan telefon, tarayıcı ve haritalar gibi bazı anahtar uygulamardan oluşur [15].



Şekil 3.2. Android mimarisi [14]

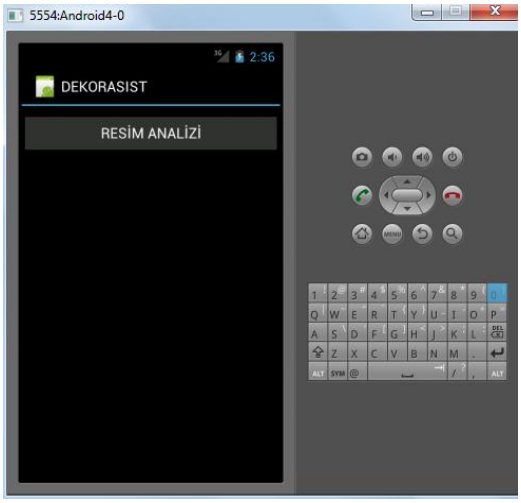
Android işletim sisteminde geliştirilen uygulamalar java dilinde yazılır. Şekil 3,2'de görüldüğü gibi java dilinde yazılan kod derlendiği zaman bytecode'a dönüşür. Cihaza uygulama yüklenmeden önce. class uzantılı bytecode dosyaları. dex dosyalarına java sanal makinesi tarafından döndürülür. Java sanal makinesi stack makine olmasına rağmen Dalvik sanal makinesi kayıt-tabanlı mimariye sahiptir. Dalvik sanal makinesi yavaş CPU daha az hafıza kullanımına yönelik optimize edilmişitir. Dalvik sanal makinesi sayesinde java dili kullanılarak mobil cihaz için yazılan uygulama sadece mobil cihazlarda çalışmaktadır.



Şekil 3.3. JVM ve DVM mimarisi

3.1.3 Android Emülatör

Android emülatör bilgisayarda çalışan bir sanal mobil cihazdır. Emülatör bir fiziksel cihaz kullanmadan android uygulamalarını geliştirmeyi ve test etmeyi sağlar. Emülatörü başlatırken kullanıcının özelleştireceği görüntü yada davranış gibi bir çok özelliği destekler.



Şekil 3.4. Android emülatörü

3.1.3.1. Android emülatör kurulumu

Bilgisayar ortamında android emülatör kurulum aşamaları:

- JDK kurulumu
- Eclipse kurulumu
- Android SDK kurulumu
- Android ADT Plugin ayarlanması

3.1.3.2. JDK kurulumu

Java geliştirme kiti kurulumu için en son verisiyonu olan Java Platform(JDK) web adresinden(http://www.oracle.com/technetwork/java/javase/downloads/index.html) indirilir. Java SE olarak Linux, solaris, windows işletim sistemleri için geliştirme kitini seçilir.

3.1.3.3. ECLIPSE kurulumu

Eclipse açık kaynak kodlu bir tümleşik geliştirme ortamıdır. Android SDK ile bir bütün olarak çalışabilen eclipse'in içinde java ile yazılan programları test etmek ve çalıştırabilmek için emülator kurulabilmektedir. Eclipse IDE'yi kullanılan ortama göre web adresinden(http://www.eclipse.org/downloads/) indirilir.

3.1.3.4. ANDROID SDK kurulumu

Android yazılım geliştirme kitini kullanılan platforma göre web adresinden (http://developer.android.com/sdk/index.html) indirilir. Kurulum işleminden sonra eclipse platformunda üst menüde Window sekmesinden Android SDK Manager seçerek android işletim sisteminin versiyonları ve araçları kurulur.

3.1.3.5. ANDROID ADT plugin ayarlanması

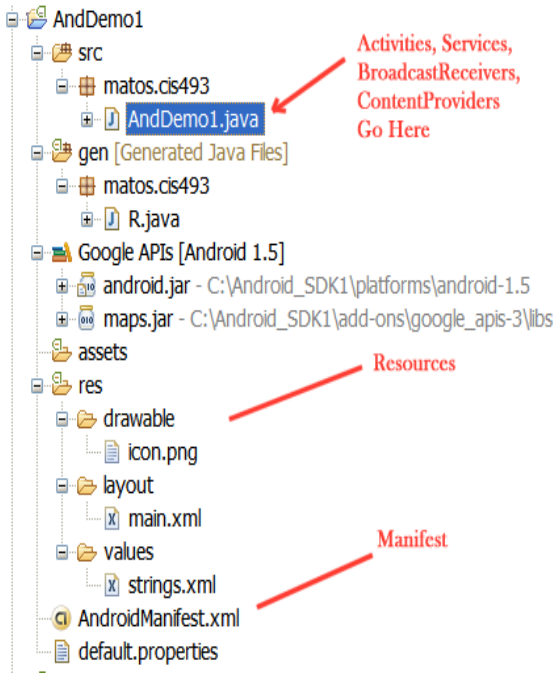
Android geliştirme araçları eklentilerini ayarlamak için eclipse platformunda üstteki menüde Help sekmesinden Install New Software bölümü seçilir. Açılan pencerede Work with alanına ayarlar için web adresi(https://dl-ssl.google.com/android/eclipse/)

girilir. Seçilen android geliştirme araçları(Android DDMS, Android Development Tools, Android Hierarchy Viewer ve Android traceview) kurulur.

DDMS bir hata ayıklama aracıdır. Bağlantı noktası yönlendirme servisi, cihazın ekran görüntüsünü kaydetme, cihazdaki thread ve heap bilgisi, logcat, process, gelen arama, konum bilgisi belirleme gibi birçok özelliği DDMS sağlar. DDMS eclipse platformuna bütünleşiktir.

3.2. Mobil Uygulama Yapısı

Eclipse platformunda yeni android proje oluşturulduğunda Şekil 3.5'deki gibi src, gen, Google APIs, assets, res ve AndroidManifest.xml gibi klasör ve dosyaları proje içinde oluşturur.



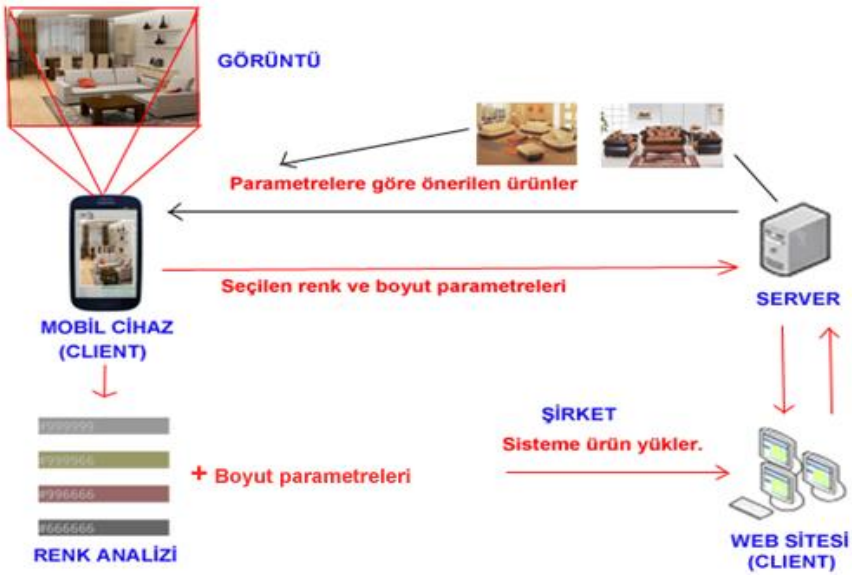
Şekil 3.5. Android uygulama yapısı

Proje içindeki src/ klasörü Activity Java dosyasını içerir. Bütün uygulama için yazılan diğer Java dosyalarıda burada yer alır. Klasör <Android Version>/ (e.g., Android 1.5/) içerisinde uygulamayı oluşturan android.jar dosyasını içerir. Klasör gen/ içinde ADT tarafından oluşturulmuş Java dosyaları yer alır örneğin, R.java dosyası ve AIDL dosyalarından üretilen arayüzler. Klasör assets/ boştur. İçinde değerli dosyaları depolanır. Klasör res/ uygulama için kaynaklar yer alır örneğin layout dosyaları, string değerleri, drawable dosyaları. AndroidManifest.xml projenin android manifestosudur. Dosya default.properties proje ayarlarını içerir örneğin hedef cihazlar gibi.

4. GELİŞTİRİLEN MOBİL UYGULAMA

4.1. Sistemin Mimarisi

Tez çalışmasında uygulaması gerçekleştirilen mobil uygulama sistemin mimarisi Şekil 4.1'de görülmektedir.

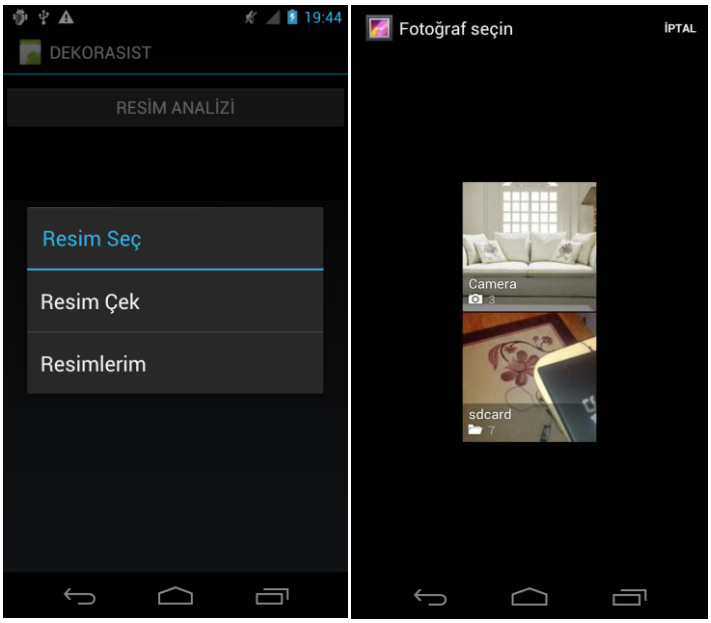


Şekil 4.1. Sistemin mimarisi

Sistem mobil ve web uygulamasından oluşmaktadır. Mobil uygulama bölümünde kullanıcı ortam görüntüsünü mobil cihazı ile çeker ya da resim galerisinden bir görüntü seçer. Uygulama görüntüyü analiz ederek renk paleti sunar. Renk paleti en yoğun renk, en az yoğun renk, ortalama renk, karşıt renk ve renk tonlarından oluşmaktadır. Kullanıcı renk paletinden renk seçer ve öneri almak istediği ürünün boyut bilgilerini girer. Mobil uygulama bu parametreleri server'a gönderir. Server paremetrelere göre veritabanındaki uygun önerilen ürünleri mobil uygulamaya göndererek kullanıcaya sunar. Ürün şirketleri web uygulama üzerinden ürünlerini sisteme yüklerler.

4.2. Mobil Uygulama İsterleri

Tez çalışmasında mobil cihaz olarak Android 4.0.2 işletim sistemli Samsung Nexus marka mobil cihaz kullanılmıştır. Programlama dili olarak JAVA kullanılmıştır. Şekil 4.2'de mobil uygulamanın resim seç bölümü görünmektedir. Resim seç bölümünden kullanıcılar resim çek veya resimlerim işlemini yapar. Resim çek işleminde kullanıcı mobil cihazın kamerasını kullanarak istediği ortamın fotoğrafını çeker ve uygulama bu görüntü üzerinde renk analizini yapar. Resimlerim bölümünden kullanıcı mobil cihazının resim galerisindeki görüntülerden seçer ve uygulama bu görüntü üzerinde renk analizini yapar.



Şekil 4.2. Mobil uygulama resim seç bölümü

Resim seç bölümü ile seçilen görüntü üzerinde aşağıdaki işlemler uygulanır:

- Görüntü boyutlandırma
- Görüntü pixel renklerinin RGB değerlerinin çıkartılması
- Renk değerlerine nicemleme algoritmasının uygulanması
- RGB renk kodlarının hexadecimal koda çevrilmesi
- Her bir rengin hex kodunun sayılması
- Sayılan renk kodlarının büyükten küçüğe sıralanması

Şekil 4.4'de görüldüğü gibi mobil uygulama görüntüye yukardaki işlemleri uygulayarak kullanıcıya maximum yoğunluktaki rengi, minimum yoğunluktaki rengi, renklerin ortalaması olan rengi ve diğer çoğunluklu olan renkleri gösterir. Kullanıcı bu bölümden istediği renk tonunu seçer ve öneri için ürünün en, boy ve yükseklik değerlerini girer. En, boy ve yükseklik değerleri ürünün ortam içerisinde yerleştirileceği alanın boyut sınırlarıdır. Kullanıcı Şekil 4.5'de öneri al butonunu kullanarak seçtiği parametreleri sunucuya gönderir. Doğrusal eşleştirme algoritması ile veritabanıyla eşleştirilen parametrelere göre önerilen ürünler mobil uygulama tarafından kullanıcıya Şekil 4.6 deki gibi gösterilir. Kullanıcı seçtiği renk kodlarını ve girdiği en, boy ve yükseklik bilgilerini değiştirerek yeni ürün önerileri alabilir.

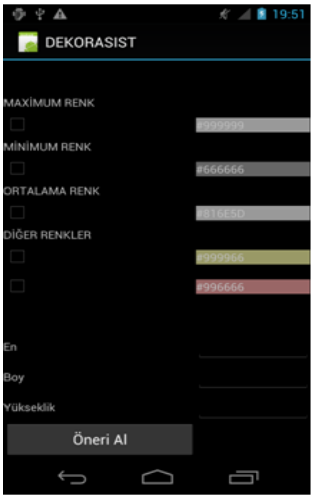
Doğrusal eşleştirme algoritması:

Cm renk analizi yapılan görüntünün seçilen renk kodları, Dm renk kod sayısı, Dk,w k sırası ile sıralanmış sunucu veritabanındaki ürünlerin sayısı, Ck,w sunucu veritabanındaki ürünlerin renk kodları olmak üzere en büyük benzerlik Popt değerini hesaplamak için Şekil 4.3'deki algoritma kullanılır [16]. Doğrusal eşleştirme algoritmasında işlem hızını arttırmak için parametre indexleme işlemi uygulanır. Kullanıcının seçtiği renk türüne göre sunucu tarafındaki veritabanında ürünler tablosunda sorgu çalıştırılır. Böylece kullanıcı alacağı öneri ürünlerini maximum renk, minimum renk ve ortalama renk değerlerine göre filtrelemiş olur.

for k=1:N;
if $D_{k,w} < D_m$
$p_k = |C_m - C_{k,w}|$
end
end
$p_{opt} = \min\{p_k\}$

Şekil 4.3. Doğrusal eşleştirme algoritması

Şekil 4.4. Mobil uygulama renk analiz bölümü



Şekil 4.5. Mobil uygulama öneri al bölümü

Şekil 4.6. Mobil uygulama önerilen ürünler bölümü

4.3. Mobil Uygulama Yazılım Bileşenleri

4.3.1. AndroidManifest

Uygulamanın manifestosudur. Uygulamanın paket, uygulama versiyonu, SDK versiyonu, internet erişim izni ve ana aktivite hakkındaki bilgileri verir. SDK versiyonun kapsadığı cihazlarda uygulama çalışabilir.

4.3.2. Main layout

Uygulamanın ekran görüntüsünü oluşturur. ScrollView ile uygulama görüntülerinin cihaz ekranında düşeyde yukarı aşağı hareket edebilmesi sağlanır. TableLayout ile ekran tablo olarak satır ve sütünlara bölünür. Button ile ekrana buton eklenmesi sağlanır. Button olarak "Renk Analizi" butonudur. ImageView ile ekrana görüntü eklenir. Eklenen görüntü kullanıcının uygulama vasıtası ile cihazından seçtiği görüntülerdir. Layout içerisinde konum ve renklendirme işlemleri yapılabilir.

4.3.3. Görüntü boyutlandırma

Görüntüye renk analizi işlemleri uygulanmadan önce hata payını azaltmak ve hızı arttırabilmek için kullanıcının cihazından seçtiği görüntü decode edilerek optimum boyuta indirgenir.

4.3.4. Skaler nicemleme bölümü

Skaler niceleme algoritmasının görüntüye uygulandığı bölümdür. Skaler değer olarak 51, tanesellik olarak 4 seçilmiştir. Görüntünün her pixelinin R, G, B renkleri çıkartılarak 51 değeriyle nicelenmiştir. Elde edilen her yeni renk map dizisi içine renk adı ve sayısı olarak yerleştirilmiştir. Map içerisindeki renk sayıları indisleri ile birlikte büyükten küçüğe doğru sıralanmıştır. Sıralı renk kodlarından ilk dört tanesi kullanıcıya uygulamada gösterilmektedir.

4.3.5. Renk analiz sonuçları

Bu bölümde elde edilen renk kodları checkboxlar ile birlikte ekrana görüntü olarak yerleştirilmektedir. Her renk koduna göre o renkte dikdörtgen kullanıcıya gösterilmektedir. Her renk satırı tablonun satırı olarak eklenmektedir.

4.3.6. HTTP iletişim

Bu bölümde server ile asenkronize task olarak bağlantı kurulmaktadır. Kullanıcının uygulamadan seçtiği renk ve parametre bilgilerini gönderme ve serverdan gelen cevapları alma işlemi yapılmaktadır. Veriler JSON formatında gönderilmekte ve alınmaktadır. JSON formatının seçilmesinin sebepi gönderilecek parametrelerin az ve ardışık yapıda gönderilebilir olmasıdır. Böylece mobil cihazın işlemler için daha az efor harcaması sağlanmıştır.

5. SONUÇLAR VE ÖNERİLER

Bu tez çalışmasında, ilk olarak renk analiz yöntemlerinin performans karşılaştırmaları yapılmıştır. Renk nicemleme yöntemlerinden skaler nicemleme ve vektörel nicemleme algoritmaları incelenmiştir. Vektör nicemleme algoritmalarından median-cut ve octree algoritmalarının görüntü işleme tarafında iyi sonuç verdiği ancak zaman ve hafıza tüketimi açısından kötü olduğu gözlemlenmiştir. Skaler nicemleme algoritması görüntü işleme konusunda vektör nicemleme yöntemlerine göre daha fazla hata oranına sahipken hafıza ve zaman konusunda başarılı olduğu gözlemlenmiştir. Geliştirilen yazılımın kullanıcılara öneri oluşturacağından ve hafızası sınırlı olan mobil cihaz üzerinde çalışacağından dolayı skaler nicemleme yöntemi kullanılarak mobil uygulama geliştirilmiştir. Hata oranını azaltmak için görüntülerin boyutlarında küçültme işlemi uygulanmıştır.

Mobil uygulama platformu olarak Android işletim sistemi seçilmiştir. Android işletim sistemi üzerinde JAVA dili ile uygulama geliştirilmiştir. Açık kaynak kodlu olan Android platformu ve görüntü işleme üzerine zengin kütüphaneleri olan JAVA dilinden dolayı uygulama Android 4.0.2 işletim sistemli mobil cihaz için geliştirilmiştir.

Geliştirilen mobil uygulamada kullanıcı mobil telefondaki resim galerisinden bir resim seçer ya da bulunduğu ortamın resmini çeker. Kullanıcı resmi seçtiği zaman mobil uygulama skaler nicemleme algoritmasını kullanarak resmin maximum, minimum ve ortalama renk değerlerini kullanıcıya sunmaktadır. Kullanıcı seçtiği renk değeri ve belirlediği en, boy ve yükseklik bilgileri ile öneri alır. Uygulama bu parametrelerle sunucudan doğrusal eşleştirmeye göre önerilen ürünleri kullanıcıya sunmaktadır. Çalışmada ileri süreçlerde renk analiz sonuçlarını iyileştirmeye yönelik yeni algoritmalar üzerinde çalışılabilir. Diğer mobil işletim sistemleri(iOS ve Windows Mobile Phone) içinde uygulama geliştirilerek performans karşılaştırması yapılabilir.

KAYNAKLAR

[1] Butler M., Android: Changing the Mobile Landscape, *Pervasive Computing*, 2010, **10**, 4–7.

[2] Türkiye Cumhuriyeti Sanayi ve Ticaret Bakanlığı Mobilya Sektörü Raporu, http://www.sanayi.gov.tr/Files/Documents/mobilya_sektoru_raporu_oc-14022011110007.pdf/ (Ziyaret Tarihi: 10 Haziran 2012).

[3] Stephen D. D., Embracing Next-Generation Mobile Platforms to Solve Business Problems, 2008, *Sybase White Paper*.

[4] Gavalas D., Economou D., Development Platforms for Mobile Applications: Status and Trends, *IEEE Software*, 2011, **28**, 77–86.

[5] Orchard M. T., Bouman C. A., Color quantization of images, *IEEE Transaction on Signal Processing*, 1991, **39**, 2677–2690.

[6] Bloomberg D. S., Color quantization using octrees, http://www.leptonica.org/papers/colorquant.pdf (Ziyaret Tarihi: 10 Haziran 2012).

[7] Baolong G., Xiang F., A Modified Octree Color Quantization Algorithm, *First International Conference on Communications and Networking*, 2006.

[8] Chen W., Ding W., An Improved Median-Cut Algorithm of Color Image Quantization, *International Conference on Computer Science and Software Engineering*, 2008.

[9] Heckbert P., Color image quantization for frame buffer display, *Computer Graphics*, 1982, **16**, 297–307.

[10] Abut H., *Vector quantization*, IEEE Press, 1990.

[11] Wei-Chao C., Yingen X., Gao, J., Gelfand N., Radek G., Efficient Extraction of Robust Image Features on Mobile Devices, *6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, 2007.

[12] Chi C. T., Richard H., Mobile Application Development: Essential New Directions for IT, School of Technology, Brigham Young University, 2010.

[13] URL–1: http://www.openhandsetalliance.com/ (Ziyaret Tarihi: 10 Haziran 2012).

[14] URL–2: http://developer.android.com/ (Ziyaret Tarihi: 10 Haziran 2012).

[15] URL–3: http://sites.google.com/site/io/anatomy-physiology-of-an-android/ (Ziyaret Tarihi: 10 Haziran 2012).

[16] Yıldız G., Kaya F., Kavak A., A Mobile Application and Web Recommendation System for Assisting Indoor Decoration, *International Symposium on INnovations in Intelligent SysTems and Applications (INISTA2012)*, Trabzon, Turkey, 2012.

EKLER

Ek−A GELİŞTİRİLEN SİSTEM İÇİN YAZILAN KAYNAK KODLAR

AndroidManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
package="gurel.dekorasist"
  android:versionCode="1"
  android:versionName="1.0">

   <uses-sdk android:minSdkVersion="14" ></uses-sdk>

   <uses-permission android:name="android.permission.INTERNET"></uses-
permission>

   <application android:icon="@drawable/icon" android:debuggable="true"
android:label="@string/app_name">

      <activity android:name=".MainActivity"
            android:label="@string/app_name">

         <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
         </intent-filter>

      </activity>

   </application>

</manifest>
```

MainLayout.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ScrollView xmlns:android="http://schemas.android.com/apk/res/android"
    android:scrollbars="vertical"
    android:layout_height="fill_parent"
    android:layout_width="fill_parent">
<TableLayout    xmlns:android="http://schemas.android.com/apk/res/android"
android:layout_height="wrap_content"
android:layout_width="fill_parent"
android:id="@+id/color_table"
android:stretchColumns="*"

android:background="#000000"
>
<TableRow android:layout_width="fill_parent"
      android:layout_height="fill_parent" >

  <Button android:id="@+id/btn_choose"
android:text="@string/resim_sec"
android:layout_marginTop="10dp"
android:layout_weight="1"
/>
</TableRow>
<TableRow android:layout_width="wrap_content"
      android:layout_height="wrap_content" >
<ImageView
   android:contentDescription="@string/image"
android:id="@+id/iv_pic"
android:layout_marginTop="10dp"
android:layout_weight="1"
android:layout_width="0px"
/>
</TableRow>
<TableRow android:layout_width="wrap_content"
      android:layout_height="wrap_content" >
<TextView android:id="@+id/TextView01" android:layout_marginTop="2dp"
android:layout_width="wrap_content" android:layout_height="wrap_content"
></TextView>
<ImageView
   android:contentDescription="@string/image"
android:id="@+id/image1"
android:layout_marginTop="2dp"
android:layout_width="0px"
/>
</TableRow>
<TableRow android:layout_width="wrap_content"
      android:layout_height="wrap_content" >
<TextView android:id="@+id/TextView02" android:layout_marginTop="2dp"
```

```
android:layout_width="wrap_content" android:layout_height="wrap_content" >
</TextView>
<ImageView
android:contentDescription="@string/image"
android:id="@+id/image2"
android:layout_marginTop="2dp"
android:layout_width="0px"
/>
</TableRow>
<TableRow android:layout_width="wrap_content"
      android:layout_height="wrap_content" >
<TextView android:id="@+id/TextView03" android:layout_marginTop="2dp"
android:layout_width="wrap_content" android:layout_height="wrap_content"
></TextView>

<ImageView
android:contentDescription="@string/image"
android:id="@+id/image3"
android:layout_marginTop="2dp"
android:layout_width="0px"

/>
</TableRow>
</TableLayout>
</ScrollView>
```

GÖRÜNTÜ BOYUTLANDIRMA

```
private Bitmap decodeFile(File f,int x){
      try {

          BitmapFactory.Options o = new BitmapFactory.Options();
          o.inJustDecodeBounds = true;
          BitmapFactory.decodeStream(new FileInputStream(f),null,o);

          final int REQUIRED_SIZE=x;

          int scale=1;
          while(o.outWidth/scale/2>=REQUIRED_SIZE &&
o.outHeight/scale/2>=REQUIRED_SIZE)
              scale*=2;


          BitmapFactory.Options o2 = new BitmapFactory.Options();
          o2.inSampleSize=scale;
```

```
        return BitmapFactory.decodeStream(new FileInputStream(f), null, o2);
    } catch (FileNotFoundException e) {}
    return null;
5.1.1. }
```

SKALER NİCEMLEME

```
int R, G, B, index = 0;
    int j=0;
for(int y = 0; y < height; y=y+4) {
for(int x = 0; x < width; x=x+4) {

index = y * width + x;

R = Math.round(Color.red(pixels[index])/51)*51;
G = Math.round(Color.green(pixels[index])/51)*51;
B = Math.round(Color.blue(pixels[index])/51)*51;

pixels[index]=Color.rgb(R,G,B);

hexColor[j] = String.format("#%06X", (0xFFFFFF & pixels[index]));


if(map.containsKey(hexColor[j])){
   val=map.get(hexColor[j]);
val=val+1;
   }
else
{
   val=1;
}

map.put(hexColor[j],val);

    j=j+1;
}
}

mImageView.setImageBitmap(bitmap_second);

sorted = sortHashMap(map);

String renkkodlari="";
```

```
        int k=0;

for (String key : sorted.keySet()){
        if(k<4){
           renkkodlari[k]=key;
        }else
        break;
        k=k+1;
}
```

RENK ANALİZ SONUÇ GÖRÜNTÜLEME

```
TableRow row1 = new TableRow(this);

        TextView goruntule = new TextView(this);
        c1 = new CheckBox(this);
    row1.addView(c1);
        goruntule.setText(renkkodlari[0]);

goruntule.setBackgroundColor(Integer.parseInt(renkkodlari[0].replaceFirst("^#",""),
16)+0xFF000000);
    row1.addView(goruntule);

    table.addView(row1,new
TableLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT));

        TableRow row2 = new TableRow(this);
        c2 = new CheckBox(this);
    row2.addView(c2);
    TextView goruntule1 = new TextView(this);
        goruntule1.setText(renkkodlari[1]);

goruntule1.setBackgroundColor(Integer.parseInt(renkkodlari[1].replaceFirst("^#","")
, 16)+0xFF000000);
    row2.addView(goruntule1);

    table.addView(row2,new
TableLayout.LayoutParams(LayoutParams.WRAP_CONTENT,
LayoutParams.WRAP_CONTENT));

    TableRow row3 = new TableRow(this);
    c3 = new CheckBox(this);
    row3.addView(c3);
    TextView goruntule2 = new TextView(this);
```

```
        goruntule2.setText(renkkodlari[2]);

goruntule2.setBackgroundColor(Integer.parseInt(renkkodlari[2].replaceFirst("^#","")
, 16)+0xFF000000);
    row3.addView(goruntule2);

    table.addView(row3,new
TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT));

    TableRow row4 = new TableRow(this);
    c4 = new CheckBox(this);
    row4.addView(c4);
    TextView goruntule3 = new TextView(this);
        goruntule3.setText(renkkodlari[3]);

goruntule3.setBackgroundColor(Integer.parseInt(renkkodlari[3].replaceFirst("^#","")
, 16)+0xFF000000);
    row4.addView(goruntule3);

    table.addView(row4,new
TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT));

    TableRow row10 = new TableRow(this);
    c5 = new CheckBox(this);
    row10.addView(c5);
    TextView goruntule10 = new TextView(this);
        goruntule10.setText("Hepsi");
    row10.addView(goruntule10);

    table.addView(row10,new
TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT));

    TableRow row5 = new TableRow(this);
    TextView goruntule4 = new TextView(this);
        goruntule4.setText("En");
        row5.addView(goruntule4);
        text1=new EditText(this);
        row5.addView(text1);
        table.addView(row5,new
TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT));

    TableRow row6 = new TableRow(this);
    TextView goruntule5 = new TextView(this);
        goruntule5.setText("Boy");
        row6.addView(goruntule5);
```

```
        text2=new EditText(this);
        row6.addView(text2);
        table.addView(row6,new
TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT));
    TableRow row7 = new TableRow(this);
    TextView goruntule6 = new TextView(this);
        goruntule6.setText("Yükseklik");
        row7.addView(goruntule6);
        text3=new EditText(this);
        row7.addView(text3);
        table.addView(row7,new
TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,
LayoutParams.MATCH_PARENT));
        TableRow row8 = new TableRow(this);
        Button btn = new Button(this);
        btn.setText("Öneri Al");
        row8.addView(btn);
        table.addView(row8,new

TableLayout.LayoutParams(LayoutParams.MATCH_PARENT,

LayoutParams.MATCH_PARENT));
```

HTTP İLETİŞİM

```
class task extends AsyncTask<String, Integer, Void>
    {

        InputStream is = null ;
        String result = "";

          @Override
    protected Void doInBackground(String... params) {

          String param = params[0];
      String url_select = "http://www.dekorasist.com/";


      HttpClient httpClient = new DefaultHttpClient();
      HttpPost httppost = new HttpPost(url_select);
      JSONObject json = new JSONObject();


        try {
```

```
json.put("renkkod",param);

JSONArray postjson=new JSONArray();
postjson.put(json);

httppost.setHeader("json",json.toString());
httppost.getParams().setParameter("jsonpost",postjson);

HttpResponse response=httpClient.execute(httppost);


      HttpEntity entity = response.getEntity();


       is = entity.getContent();



   } catch (Exception e) {

   Log.e("log_tag", "Error in http connection "+e.toString());
   }

      try{


            BufferedReader reader = new BufferedReader(new
InputStreamReader(is));


            StringBuilder sb = new StringBuilder();


            String line = null;


           while ((line = reader.readLine()) != null) {


                 sb.append(line + "\n");
            }

            is.close();

            result=sb.toString();


      }catch(Exception e){
```

```
            Log.e("log_tag", "Error converting result "+e.toString());


        }

        try{

            JSONArray jArray = new JSONArray(result);

            for(int i=0;i<jArray.length();i++){


                JSONObject json_data = jArray.getJSONObject(i);


                Log.i("log_tag","id: "+json_data.getString("renkkod"));
                   id=json_data.getString("renkkod");

            }


        }catch(JSONException e){


            Log.e("log_tag", "Error parsing data "+e.toString());


        }

    return null;

    }
}
```

KİŞİSEL YAYINLAR VE ESERLER

A. Uluslararası Bilimsel Toplantılarda Sunulan ve Bildiri Kitabında Basılan Bildiriler

1. **Yıldız G**., Altıntaş L., Medical Education Program Development Application of Kocaeli University, *Proceeding of International Science and Technology Conference (ISTEC11),* Istanbul, Turkey, 2011.

2. **Yıldız G.**, Kaya F., Kavak A., A Mobile Application and Web Recommendation System for Assisting Indoor Decoration, *International Symposium on INnovations in Intelligent SysTems and Applications (INISTA2012),* Trabzon, Turkey, 2012.

B. Ulusal Bilimsel Toplantılarda Sunulan ve Bildiri Kitaplarında Basılan Bildiriler

1. **Yıldız G.**, Kavak A., Mobil Telefon için J2ME ile Öğrenci Bilgi Sistemi Web Servis Uygulaması Geliştirme, *II. İleri Teknoloji Çalıştayı*, 2011.

2. Altıntaş L, Alimoğlu M. K., Alvur T. M., **Yıldız G.**, Diri S., Ulusal Çekirdek Eğitim Programının Tıp Eğitimi Programına Entegrasyonunda Yazılım Destekli Uygulama Örneği Kocaeli Tıp Fakültesi Deneyimi, *VII. Ulusal Tıp Eğitimi Kongresi Özet Kitabı (UTEK 2012),* 2012.

ÖZGEÇMİŞ

1982 yılında Sinop'da doğdu. İlk, orta ve lise öğrenimini Kocaeli'de tamamladı. 2002 yılında girdiği Işık Üniversitesi Elektronik Mühendisliği bölümünden 2007 yılında mezun oldu. 2010 yılından itibaren Kocaeli Üniversitesi Bilgisayar Mühendisliği'nde araştırma görevlisi olarak çalışmaktadır. 2009 yılında başladığı Kocaeli Üniversitesi Fen Bilimleri Bilgisayar Mühendisliği Anabilim Dalı'ndaki Yüksek Lisans eğitimine devam etmektedir.