
GAZİ ÜNİVERSİTESİ
Gazi
1926

TEKRARLI UZUNLUK KODLAMASI (RLE) VE AYRIŞTIRMA ESASLI GÖRÜNTÜ SIKIŞTIRMA

Mustafa Burak KALKAN

YÜKSEK LİSANS TEZİ
BİLGİSAYAR MÜHENDİSLİĞİ ANA BİLİM DALI

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

HAZİRAN 2019

Mustafa Burak KALKAN tarafından hazırlanan "TEKRARLI UZUNLUK KODLAMASI (RLE) VE AYRIŞTIRMA ESASLI GÖRÜNTÜ SIKIŞTIRMA" adlı tez çalışması aşağıdaki jüri tarafından OY BİRLİĞİ ile Gazi Üniversitesi Bilgisayar Mühendisliği Ana Bilim Dalında YÜKSEK LİSANS TEZİ olarak kabul edilmiştir.

Danışman: Prof. Dr. Recep DEMİRCİ

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi ...……

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum.

Başkan: Prof. Dr. Remzi YILDIRIM

Bilgisayar Mühendisliği Ana Bilim Dalı, Yıldırım Beyazıt Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Üye: Dr. Öğr. Üyesi Cemal KOÇAK

Bilgisayar Mühendisliği Ana Bilim Dalı, Gazi Üniversitesi

Bu tezin, kapsam ve kalite olarak Yüksek Lisans Tezi olduğunu onaylıyorum. ...……

Tez Savunma Tarihi: 14/06/2019

Jüri tarafından kabul edilen bu tezin Yüksek Lisans Tezi olması için gerekli şartları yerine getirdiğini onaylıyorum.

……..…….

Prof. Dr. Sena YAŞYERLİ

Fen Bilimleri Enstitüsü Müdürü

ETİK BEYAN

Gazi Üniversitesi Fen Bilimleri Enstitüsü Tez Yazım Kurallarına uygun olarak hazırladığım bu tez çalışmasında;

- Tez içinde sunduğum verileri, bilgileri ve dokümanları akademik ve etik kurallar çerçevesinde elde ettiğimi,
- Tüm bilgi, belge, değerlendirme ve sonuçları bilimsel etik ve ahlak kurallarına uygun olarak sunduğumu,
- Tez çalışmasında yararlandığım eserlerin tümüne uygun atıfta bulunarak kaynak gösterdiğimi,
- Kullanılan verilerde herhangi bir değişiklik yapmadığımı,
- Bu tezde sunduğum çalışmanın özgün olduğunu,

bildirir, aksi bir durumda aleyhime doğabilecek tüm hak kayıplarını kabullendiğimi beyan ederim.

Mustafa Burak KALKAN

14/06/2019

TEKRARLI UZUNLUK KODLAMASI (RLE) VE AYRIŞTIRMA ESASLI GÖRÜNTÜ SIKIŞTIRMA

(Yüksek Lisans Tezi)

Mustafa Burak KALKAN

GAZİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

Haziran 2019

ÖZET

Teknolojinin gelişimiyle birlikte her geçen gün boyutları büyümekte olan sayısal görüntüleri saklamak ve transfer etmek problem haline gelmiştir. Dolayısıyla görüntüleri saklamak için gerekli bellek alanı artmakta ve aynı zamanda transfer süreleri de uzamaktadır. İlgili problemlerin çözümü için geliştirilen çözümlerden biri de görüntülerin sıkıştırılması olmuştur. Bu çalışmada renkli görüntülerin renk uzayı yardımıyla ayrıştırılması yaklaşımı ve tekrarlı uzunluk kodlaması (run length encoding: RLE) algoritması birleştirilerek yeni bir kayıplı sıkıştırma algoritması geliştirilmiştir. Temel olarak kodlayıcı ve kod çözücü olmak üzere iki simetrik yapıdan oluşan yöntemin kodlayıcı aşamasında, orijinal görüntü seçilen eşik sayısına göre renk uzayı yardımıyla ayrıştırılmakta ve renk indirgemesi yapılmaktadır. Sınıflandırma sonucunda görüntüdeki her bir pikselin ait olduğu sınıfı gösteren bir etiket haritası elde edilmektedir. Ayrıca ilgili sınıflara atanan renk değerlerini içeren bir kod kitabı üretilmektedir. Akabinde ise etiket haritasındaki bilgiler ve RLE algoritması yardımıyla sıkıştırma yapılmıştır. Kodlayıcı tarafından üretilen sıkıştırılmış veri biçimlendirilmiş bir dosya halinde belleğe kaydedilebilmekte veya kod çözücü ile bellekten okunabilmektedir. Geliştirilen metodun farklı sınıflandırma çözümleri ile çalışabilme esnekliği de bulunmaktadır. Bu nedenle çok yaygın olarak kullanılan K-ortalamalar yöntemi ile önerilen algoritma birleştirilmiş ve sonuçlar alınmıştır. İlave olarak önerilen yaklaşımın JPEG ve GIF sıkıştırma formatları karşısındaki başarımı da incelenmiştir. Deneysel sonuçlar geliştirilen yöntemin renk geçişleri ve değişimlerinin az olduğu veya renk dağılımının homojen olduğu görüntülerde daha başarılı sonuçlar üretebildiğini göstermiştir.

Bilim Kodu	:	92418
Anahtar Kelimeler	:	Görüntü sıkıştırma, görüntü ayrıştırma, tekrarlı uzunluk kodlaması
Sayfa Adedi	:	79
Danışman	:	Prof. Dr. Recep DEMİRCİ

RUN-LENGTH ENCODING AND SEGMENTATION BASED IMAGE COMPRESSION

(M. Sc. Thesis)

Mustafa Burak KALKAN

GAZİ UNIVERSITY

GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES

June 2019

ABSTRACT

With the development of technology, it has become a problem to store and transfer the digital images that are growing in size every day. Therefore, the memory space required to store images increases and also the transfer times are extended. One of the strategies developed for the solution of related problems is the compression of the images. In this study, a new lossy compression algorithm was developed by combining the segmentation of images with color space approach and the run length encoding (RLE) algorithm. In the encoding stage of the method, which consists mainly of two symmetrical structures as encoder and decoder, the original image is segmented with the color space according to the selected number of thresholds and color reduction is performed. As a result of the classification, a label map is obtained that shows the class of each pixel in the image. In addition, a code book containing the color values assigned to the related classes is produced. Then, the information on the label map was compressed using the RLE algorithm. The compressed data generated by the encoder can be stored in the memory as a formatted file or read from the memory by the decoder. The developed method has the flexibility to work with different classification solutions. Therefore, the commonly used K-means method and the proposed algorithm were combined, and the results were taken. In addition, the performance of the proposed approach against JPEG and GIF compression formats was also tested. The experimental results showed that the developed method can produce successful results in images which has homogeneous color distribution or has a small amount of color transitions and changes.

Science Code : 92418
Key Words : Image compression, image segmentation, run-length encoding
Page Number : 79
Supervisor : Prof. Dr. Recep DEMİRCİ

TEŞEKKÜR

Tez çalışması sürecinde rehberliğini ve yardımlarını esirgemeyen değerli danışman hocam Prof. Dr. Recep DEMİRCİ başta olmak üzere, destekleriyle beni hiçbir zaman yalnız bırakmayan değerli eşim Behice Nur KALKAN'a ve biricik kızım Erva Beren'e; ayrıca deneyimlerinden faydalandığım değerli arkadaşlarım Mahmut KILIÇARSLAN, Haydar TUNA, Ahmet Selim KAHRAMAN, Taymaz Rahkar FARSHİ'ye teşekkürlerimi sunmayı bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

ÇİZELGELERİN LİSTESİ

Çizelge **Sayfa**

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

Şekil **Sayfa**

SİMGELER VE KISALTMALAR

Bu çalışmada kullanılan simgeler ve kısaltmalar, aşağıda açıklamaları ile birlikte verilmiştir.

Simgeler	**Açıklamalar**
dB	Desibel
kB	Kilobyte

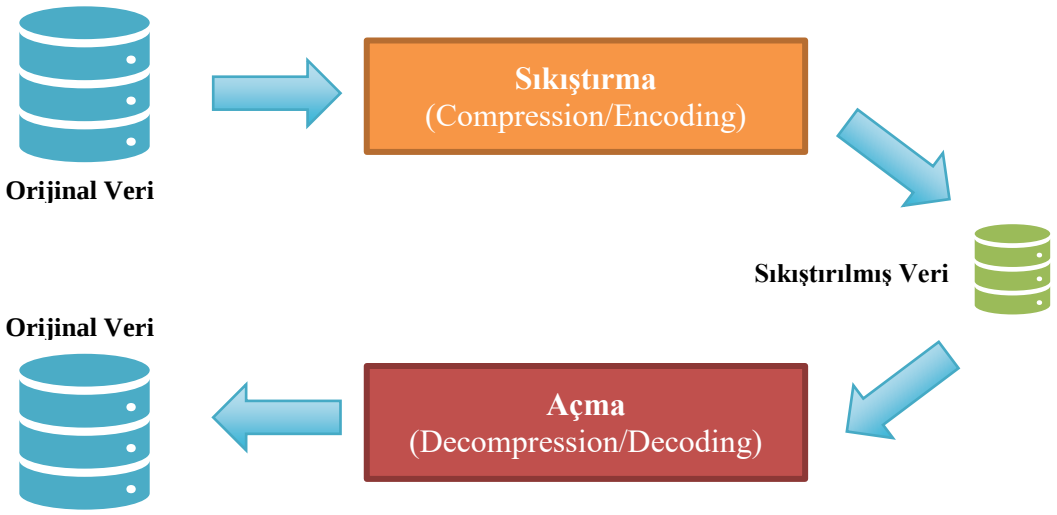
Kısaltmalar	**Açıklamalar**
CMYK	Cyan-Magenta-Yellow-Black renk modeli
CR	Compression Ratio (Sıkıştırma Oranı)
DCT	Discrete Cosine Transform (Ayrık Kosinüs Dönüşümü)
GIF	Graphics Interchange Format görüntü biçimi
HSV	Hue-Saturation-Value renk modeli
JPEG	Joint Photographic Experts Group görüntü biçimi
LBG	Linde-Buzo-Gray algoritması
LSB	Least Significant Byte (En Düşük Değerlikli Byte)
LZW	Lempel-Ziw-Welch sıkıştırma algoritması
MSE	Mean Square Error (Ortalama Kare Hatası)
MSB	Most Significant Byte (En Yüksek Değerlikli Byte)
PSNR	Peak Signal to Noise Ratio (Tepe Sinyal-Gürültü Oranı)
RGB	Red-Green-Blue renk modeli
RGBA	Red-Green-Blue-Alpha renk modeli
RLE	Run-Length Encoding sıkıştırma algoritması
SOM	Self-Organizing Maps algoritması
YUV	Y: Luminance, U-V: Chrominance renk modeli

1. GİRİŞ

Teknolojinin gelişmesiyle beraber her geçen gün daha kapsamlı hale gelen ve boyutları büyümekte olan sayısal verilerin depolanması ve aktarımı önemli bir problem haline gelmiştir. Depolama birimlerinin kapasitelerinin sonlu olması, bilginin günümüzde daha dinamik ve taşınabilir duruma gelmesi, aktarım hızlarının önem kazanması, ayrıca veri büyüdükçe bütünlüğünün korunmasının zorlaşması gibi birçok faktör de problemin seviyesini artırmaktadır. Dolayısıyla veri boyutunun bir şekilde küçültülmesi, depolama ve aktarım maliyetlerinin düşürülmesi gerekmektedir. Bu noktada veri sıkıştırma yöntem ve teknikleri öne çıkmaktadır.

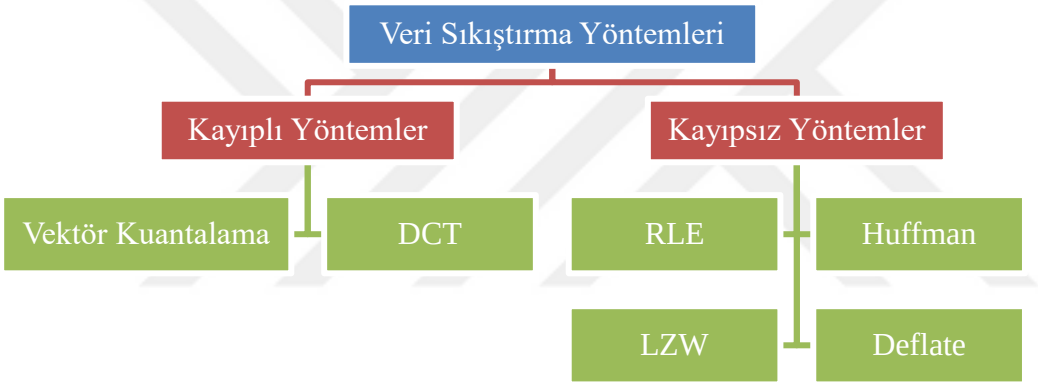
Veri sıkıştırma, veriyi elektronik ortamda depolayabilmek için gerekli sayısallaştırılmış bilgi miktarını azaltmayı amaçlayan yöntemler bütünüdür. Başka bir deyişle veri sıkıştırma bilgiyi olduğundan daha kompakt bir formda temsil etme tekniğidir [1].

Sıkıştırma işleminde orijinal veri boyutu, kullanılan sıkıştırma yöntemine ait kodlayıcı (compressor/encoder) tarafından sıkıştırılarak küçültülür. Sıkıştırılmış haldeki veri tek başına kullanılabilir değildir. Sıkıştırılmış veri tekrar kullanılmak istendiğinde, sıkıştırıldığı yönteme ait çözücü (decompressor/decoder) ile orijinal haline geri dönüştürülmesi gerekmektedir. Şekil 1.1'de sıkıştırma ve açma işleminin diyagramı verilmiştir.



Şekil 1.1. Veri sıkıştırma ve açma işlemi

Veri sıkıştırma yaklaşımları biçimlerine göre kayıpsız ve kayıplı yaklaşımlar olmak üzere iki kısımda incelenebilir [1]. Kayıpsız sıkıştırma yaklaşımlarında orijinal veri, sıkıştırılmış veri içerisinden hiçbir kayıp olmadan aynı şekliyle geri getirilebilir. Bu özelliği dolayısıyla kayıpsız teknikler her alanda kullanılabilir. Kayıplı metotlarda ise daha yüksek sıkıştırma oranlarına ulaşabilmek için verinin bütünlüğünü en az düzeyde etkileyecek olan veri kümeleri orijinal veriden çıkartılır ve geriye kalan kümeler kayıpsız sıkıştırmaya tabi tutulur [2]. Dolayısıyla kayıplı yöntemlerde orijinal veri ile geri getirilen veri birebir aynı değildir. Oluşan bilgi kaybı dolayısıyla metin sıkıştırma gibi alanlarda kayıplı sıkıştırmanın kullanılması uygun olmaz. Fakat görüntü, ses, video vb. alanlarda kayıplara müsaade edilerek daha yüksek sıkıştırma oranlarına ulaşılabilir. Bazı kayıplı ve kayıpsız sıkıştırma algoritmaları Şekil 1.2'de gösterilmiştir.



Şekil 1.2. Kayıplı ve kayıpsız sıkıştırma yöntemleri

Görüntü, üç boyutlu uzaydaki nesnelerin iki boyutlu düzlemdeki iz düşümlerinden meydana gelen bir resim bilgisidir [3]. Sayısal görüntü ise ilgili resim bilgisinin optik-elektrik görüntü algılayıcıları ile elektriksel sinyallere, sonrasında da dönüştürücüler yardımıyla sayısal sinyallere dönüştürülmesi ile elde edilir [4]. Sayısal görüntüler satır ve sütunlardan oluşan iki boyutlu diziler veya tablolar şeklinde temsil edilebilirler. Satır ve sütunların kesişiminde yer alan her bir elemana piksel adı verilir ve pikseller görüntü içerisinde bulundukları konuma ait aydınlık veya renk bilgisini tutarlar.

Renk bilgilerinin saklanmasında ise çeşitli modeller ve renk uzayları kullanılmaktadır. RGB (red, green, blue) renk modeli bunlardan biridir. RGB modelinde renkler, görünür ışığın ana renk bileşenleri olan kırmızı, yeşil ve mavinin farklı yoğunluklarda karıştırılması ile elde edilmektedir. RGB uzayının kullanıldığı sayısal görüntülerde piksellere ait renk bilgileri 8'er

bitlik üç ayrı renk kanalı ile temsil edilmektedir. Buna göre her piksel için 24 bit boyutunda bir bellek alanına ihtiyaç duyulur.

Görüntü sıkıştırma, veri sıkıştırmanın sayısal görüntüler üzerinde uygulanmasıdır. Görüntü sıkıştırma ile sayısal görüntünün bellekte işgal edeceği yerin küçültülmesi amaçlanmaktadır. Görüntü sıkıştırma için kullanılan yaklaşımlar, veri sıkıştırmada olduğu gibi kayıplı ve kayıpsız sıkıştırma yaklaşımlar olarak iki grupta ele alınabilir. Kayıpsız tekniklerde orijinal görüntü herhangi bir değişiklik yapılmadan sıkıştırılırken, kayıplı tekniklerde ise görüntü kalitesinden ödün verilerek daha yüksek sıkıştırma oranlarına ulaşılması hedeflenmektedir. Kayıplı sıkıştırma ile sıkıştırılmış görüntüdeki kayıpların ne düzeyde kabul edilebilir olacağı kullanım ve uygulama alanına göre değişmektedir. Ayrıntıların sonuçları etkilemediği alanlarda, daha yüksek sıkıştırma oranlarına ulaşabilmek adına kayıplı tekniklere başvurulabilir.

Kayıplı sıkıştırma, esasen bir vektör kuantalama işlemidir. Vektör kuantalamada orijinal veri sınıflandırılarak daha küçük boyutlu bir veri setine indirgenir. Bu esnada orijinal veri içerisinde bilgi kaybı meydana gelir fakat set boyutu küçüldüğü için depolama için gereken bellek alanından tasarruf sağlanmış olur. Vektör kuantalama sonucunda elde edilen yeni veri seti kod kitabı olarak adlandırılır.

Vektör kuantalama yaklaşımları rekabetçi öğrenme tabanlı stratejiler ve K-ortalamalar tabanlı stratejiler olmak üzere iki kategoride incelebilir [5]. Rekabetçi öğrenme tabanlı kuantalama metotları için ateşböceği algoritması (firefly algorithm), eşit bozulma prensibi (equidistortion principle), bulanık mantık (fuzzy-soft), özdüzenleyici haritalar (self-organizing maps: SOM) ve yapay gaz ağları (neural gas networks) tabanlı yaklaşımlar örnek verilebilir [6-10]. Ortalama tabanlı metotlar içinse K-ortalamalar ve LBG (Linde-Buzo-Gray) sınıflandırma algoritmaları örnek gösterilebilir [11, 12].

Vektör kuantalama alanında birçok farklı çalışma yapılmıştır. Kekre ve arkadaşları [13] klasik LBG algoritmasına göre daha hızlı çalışan, hibrit dalgacık dönüşümü ve vektör kuantalama tabanlı bir görüntü sıkıştırma yaklaşımı önermişlerdir. Bu yaklaşımda görüntüye hibrit dalgacık dönüşümü uygulanarak görüntüyü ilgili domenine aktarmışlar, sonrasında ise düşük frekans katsayılarına vektör kuantalama uygulayarak sıkıştırma yapmışlardır. Chiranjeevi ve arkadaşları [14] guguk kuşu optimizasyon algoritması ve vektör kuantalama

tabanlı bir görüntü sıkıştırma tekniği önermişlerdir. İlgili çalışmada Levy uçuşu fonksiyonu kullanılarak LBG algoritması için kod kitabı oluşturma aşamasının optimize edilmesi amaçlanmıştır. Celebi [15], çalışmasında K-ortalamalar algoritmasının renk kuantalayıcı performansını incelemiştir. Farklı görüntü setleri ve yaklaşımlarla yapılan deneylerin sonucunda, başlangıç merkez noktalarının seçiminde iyi bir strateji uygulandığı takdirde K-ortalamalar'ın etkili sonuçlar verebildiğini tespit etmişlerdir. Dursunoğlu [16], yaptığı çalışmada K-ortalamalar ve SOM algoritmaları ve bunların farklı güncelleme metotlarını kullanarak Lena görüntüsü üzerinde vektör kuantalama performansını incelemiş, aynı zamanda kuantalanan görüntü üzerinde fark kodlaması ve Huffman kodlaması kullanarak sıkıştırma çalışmaları gerçekleştirmiştir.

Veri sıkıştırma performans ölçümünde sıkıştırma oranı kullanılmaktadır. Sıkıştırma oranı, verinin sıkıştırılmamış haldeki boyutunun sıkıştırılmış haldeki boyutuna oranıdır [17]. Kayıpsız sıkıştırmada sıkıştırma oranı, performans ölçütü olarak tek başına kullanılabilmektedir. Fakat kayıplı sıkıştırmada, üretilen görüntünün orijinal görüntüden ne kadar farklılaştığının da değerlendirmeye alınması gerekmektedir. Söz konusu değerlendirmenin yapılabilmesi için benzerlik ölçeklerinden faydalanılabilir. PSNR, kaybolan bilgi miktarının veya kodlanan görüntünün orijinal görüntüden ne kadar farklılaştığının ölçümünde kullanılan bir metriktir. Desibel ile ifade edilen PSNR değeri artıkça, iki görüntü arasındaki benzerlik oranı da artacaktır [18]. PNSR kriteri gri ölçekli görüntülerin veya renkli görüntülerin her bir kanalının benzerliğini test etmek için kullanılmaktadır. Bununla birlikte renkli görüntülerde üç kanalın ortalaması da tercih edilen bir başka yaklaşım olmuştur [19].

Literatürde birçok görüntü sıkıştırma algoritması ve bunları kullanan dosya formatları mevcuttur. Bunlardan bazıları Çizelge 1.1'de listelenmiştir [2, 20-23].

Çizelge 1.1. Bazı görüntü dosya formatları ve sıkıştırma özellikleri

Dosya Biçimi	Desteklediği Sıkıştırma Algoritma ve Yöntemleri	Sıkıştırma Türü	Sıkıştırma Oranına Müdahale Edebilme
BMP	RLE (kısıtlı)	Kayıpsız	(uygulanamaz)
PNG	Deflate	Kayıpsız	(uygulanamaz)
JPEG	Ayrık Kosinüs Dönüşümü + Huffman	Kayıplı	Var
GIF	Lempel-Ziw-Welch (LZW)	Kayıplı	Yok

BMP, Microsoft tarafından geliştirilmiş bir görüntü formatıdır. 32 bit derinliğe kadar renk derinliği desteği vardır. 4 bit ve 8 bit derinliğindeki görüntüler için RLE sıkıştırması kullanabilir. Diğer renk derinliklerinde ise sıkıştırma desteklenmemektedir [23].

PNG, 90'lı yılların ortalarında geliştirilmiş kayıpsız sıkıştırmalı bir dosya biçimidir. LZ77 algoritması ile Huffman kodlamasının bir birleşimi olan Deflate algoritmasını kullanmaktadır. GIF'in kullanımındaki yasal kısıtlamalar sebebiyle ortaya çıkmıştır. Taşınabilirlik ve kolay geliştirilebilirlik amaçlanmıştır. Patent koruması olmadığı için geniş bir kullanım alanı mevcuttur. Tam alfa şeffaflık desteği vardır. Simgesel boyutlardaki resimlerde sıkıştırma başarımı yüksektir [21, 24].

GIF, sıkıştırma algoritması olarak LZW kullanan, CompuServe Inc. tarafından geliştirilmiş kayıplı bir görüntü formatıdır [25]. Maksimum 8 bit renk derinliğine, başka bir ifade ile 256 renge kadar renk desteği vardır. Dolayısıyla, daha fazla renk içeren bir görüntü GIF formatı kullanılarak sıkıştırılmak istenirse renk indirgemeye (color reduction) tabi tutularak 256 renk seviyesine indirgenir. Şeffaflık (alfa) desteği vardır. Tek dosyada birden çok resmi saklayabilir ve bunları bir animasyon şeklinde gösterebilir. GIF formatında kullanılan LZW ise sözlük tabanlı bir kayıpsız sıkıştırma algoritmasıdır. Unisys Corp. tarafından patentli olarak geliştirilmiş, 2004 yılında ise patent süresi sona ermiştir [26]. Hızı ve sıkıştırma performansı yüksektir. LZW kayıpsız olmasına rağmen GIF formatının maksimum 256 renge kadar renk desteği olması sebebiyle daha fazla renk içeren görüntüler için çıkış görüntüsü kayıplı olmaktadır [22].

JPEG formatı, ayrık kosinüs dönüşümü (Discrete Cosine Transform: DCT) ve Huffman RLE tabanlı kayıplı bir sıkıştırma metodu kullanmaktadır. Sıkıştırma aşamalarında ilk olarak görsele RGB- YUV renk uzayı dönüşümü uygulanır. Sonrasında görsel 8×8 bloklara ayrılır ve her bloğa DCT uygulanır. DCT uygulanmış blok, 1-100 arasında, 1 en düşük kalite ve 100 en yüksek kalite olmak üzere kullanıcı tarafından seçilebilen bir kalite parametresine karşılık gelen kuantizasyon matrisi kullanılarak kuantalanır. Söz konusu kuantizasyon matrisleri psiko-görsel araştırmalar sonucunda elde edilen veriler doğrultusunda hazırlanmıştır. Kuantalama sonrasında çıkan sonuç matrisi de Huffman ve RLE algoritması kullanılarak sıkıştırılır [27].

Görüntü ayrıştırma görüntüyü birbiriyle çakışmayan anlamlı bölgelere ayırma işlemidir [28, 29]. Bu yönüyle görüntü ayrıştırmaya, renk indirgeme ve vektör kuantalama işlemlerinde de başvurulmaktadır. Ayrıştırma için literatürde birçok farklı yöntem ve yaklaşım bulunmaktadır. Bunlar eşikleme tabanlı, bölge tabanlı ve kenar tabanlı yaklaşımlar olarak üç grupta incelenebilir [30, 31]. K-ortalamalar, bölge büyümesi, LBG ve bulanık c-ortalamalar bölge tabanlı yaklaşımlara; Roberts, Sobel, Prewitt, Kirsch, Robinson, Marr-Hildret, Watershed ve Canny yaklaşımları ise kenar algılama tabanlı ayrıştırma yaklaşımlarına örnek gösterilebilir [11, 12, 31-33]. Eşikleme tabanlı yaklaşımlara örnek olarak ise Otsu, Kapur ve bunları temel alan görüntülerin renk uzayı yardımıyla ayrıştırılması yaklaşımı örnek verilebilir [28, 34, 35]. Demirci ve arkadaşları tarafından geliştirilen görüntülerin renk uzayı yardımıyla ayrıştırılması yöntemi, gri seviyeli görüntülerde başarılı sonuçlar üreten, Otsu ve Kapur algoritmaları temel alınarak geliştirilmiş bir yaklaşımdır. Üç boyutlu RGB renk uzayı esas alınmaktadır. Süreçte öncelikle her bir kanal için eşikler hesaplanmakta, sonrasında ise renk uzayı alt küp veya prizmalar şeklinde kesilmektedir. Her bir alt küp veya prizma içinde kalan pikseller aynı sınıfa atanmaktadır. Ayrıca aynı sınıfa atanan piksellerin renk ortalaması alınarak renk indirgemesi yapılmaktadır. Renk indirgemesi sebebiyle görüntüde kayıplar oluşmaktadır fakat görüntü sıkıştırmaya uygun hale getirilmektedir.

Bu çalışmada görüntülerin renk uzayı yardımıyla ayrıştırılması metodu ve kayıpsız bir sıkıştırma algoritması olan tekrarlı uzunluk kodlaması (RLE) birleştirilerek renkli görüntüler için yeni bir kayıplı sıkıştırma tekniği geliştirilmiştir. Geliştirilen algoritmanın, içerisindeki renklerin uzaysal olarak homojen dağıldığı görüntülerde başarılı olduğu görülmüştür. Ayrıca farklı ayrıştırma yöntemleriyle de çalışabilme esnekliği bulunmaktadır.

2. SAYISAL GÖRÜNTÜ İŞLEME VE AYRIŞTIRMA YÖNTEMLERİ

Sayısal görüntü işleme, girdisinin görüntü, çıktısının ise yeni bir görüntü veya girdi görüntüsüyle ilgili birtakım bilgiler olabileceği bir sinyal işleme türüdür. Fotoğrafçılık, tıp, savunma sanayi, güvenlik, havacılık, otomotiv, coğrafi sistemler, meteoroloji, biyometrik sistemler, trafik kontrol sistemleri, artırılmış gerçeklik sistemleri ve birçok farklı alanda görüntü işleme tekniklerinden faydalanılmaktadır. Görüntü işleme süreci, sayısal görüntünün elde edilmesiyle başlayan; değiştirilmesi, iyileştirilmesi, parçalanması, birleştirilmesi, sıkıştırılması, ayrıştırılması veya ondan anlamlı bilgiler çıkarılması şeklinde devam eden çeşitli metodolojilerden oluşmaktadır.

2.1. Sayısal Görüntü

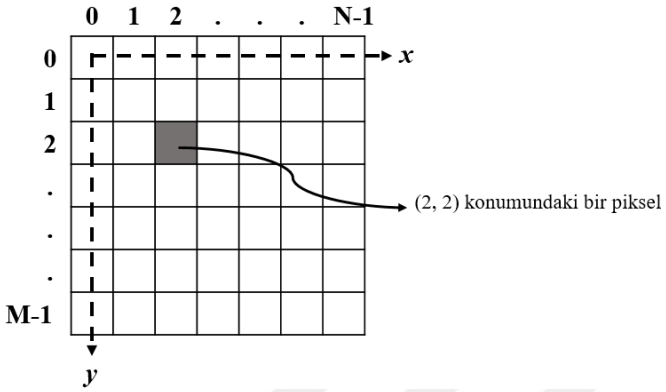
Sayısal görüntü, görüntünün optik-elektrik algılayıcılar ve dönüştürücüler kullanılarak elektrik sinyalleri ve sonrasında da sayısal sinyaller ile temsil edilmiş halidir [3, 4]. Sayısal görüntü teknolojisi, görüntülerin elektronik ortamlarda depolanabilmesini ve aktarılabilmesini sağlamıştır. İlk sayısal görüntü uygulaması 1920 yılında gazete endüstrisinde kullanılmıştır. Bartlane adı verilen iletişim sistemi ile Londra ve New York şehirleri arasında denizaltı kabloları aracılığıyla gazetelerin sayısal görüntülerinin aktarımı yapılmıştır [36].

Sayısal görüntüler; satırlar, sütunlar ve bunların kesişimleri olan piksellerden meydana gelen iki boyutlu matris yapılarıdır. Başka bir ifade ile sayısal görüntüler, *x* değişkeninin sütun indisini, *y* değişkeninin ise satır indisini gösterdiği *f(x, y)* şeklinde ifade edilebilen iki değişkenli ve sonlu fonksiyonlardır [36]. *f(x, y)* fonksiyonunun $M \times N$ boyutlarındaki matris ifadesi $0 \le x \le N-1$ ve $0 \le y \le M-1$ olmak koşuluyla,

$$f(x,y) = \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,N-1) \\ f(1,0) & f(1,1) & \dots & f(1,N-1) \\ \vdots & \vdots & \ddots & \vdots \\ f(M-1,0) & f(M-1,1) & \dots & f(M-1,N-1) \end{bmatrix} \tag{2.1}$$

şeklinde gösterilebilir. Burada *f(x, y)* fonksiyonunun her bir bileşeni piksel olarak adlandırılır. Pikseller renk veya aydınlık bilgisi taşırlar ve görüntüyü oluşturan yapı taşı niteliğindedirler. İkili görüntülerde yalnızca 0 ve 1 değerini alabilirken, gri ölçekli

görüntülerde ise aydınlık derecesine göre 0 ile 255 arasında değerler alabilirler. Bir pikselin bulunduğu konum, sütun ve satır ikilisi *(x, y)* şeklinde ifade edilir. Buna göre $M \times N$ boyutlarındaki bir görüntü için *(0, 0)* ifadesi görüntünün sol üst köşesindeki, *(M-1, N-1)* ifadesi ise sağ alt köşesindeki pikselin konumunu temsil etmektedir. Sayısal görüntünün ve görüntü içerisindeki bir pikselin gösterimi Şekil 2.1'de verilmiştir. Renkli görüntülerde piksele ait renk bilgisi, rengi meydana getirecek olan farklı renkteki ışık bileşenlerinin aydınlık derecelerinin birleşiminden oluşmaktadır.

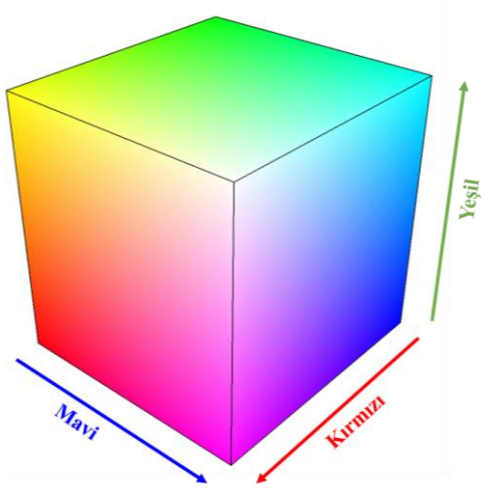


Şekil 2.1. Sayısal görüntü ve pikselin gösterimi

2.2. RGB Renk Modeli

Sayısal görüntülerde sık kullanılan renk modelleri olarak RGB (Red, Green, Blue), CMYK (Cyan, Magenta, Yellow, Black) ve HSV (Hue, Saturation, Value) örnek verilebilir. Her model farklı kullanım alanlarının ihtiyaçlarına göre geliştirilmiştir. RGB modeli sayısal görüntülerin saklanması ve işlenmesi alanında sıklıkla kullanılırken, CMYK modeli boya renklerinin karışım özelliklerine uygunluğu sebebiyle daha çok baskı endüstrisinde kullanılmaktadır. HSV modeli ise insan görüsüne yatkın bir renk modeli geliştirme fikriyle ortaya çıkmıştır [37].

RGB modelinde renkler kırmızı, yeşil ve mavi olmak üzere üç renk bileşeninin farklı oranlarda karıştırılmasıyla elde edilir. RGB uzayındaki kırmızı, yeşil ve mavi kanalları 3 boyutlu olarak Şekil 2.2'de gösterilmiştir. Her kanala ait değerler 0-255 aralığında (8 bit) olmakla beraber RGB uzayı, 24 bit derinliğindeki 2^{24} = 16 777 216 adet farklı rengi içerisinde barındırmaktadır.



Şekil 2.2. RGB renk uzayının 3 boyutlu gösterimi

RGB modelinde renkler opak olarak temsil edilmekte olup piksele ait şeffaflık (alfa) bilgisi yer almamaktadır. Bu ihtiyaçlar için RGB modelinin kırmızı, yeşil ve mavi kanallarının yanında şeffaflık bilgisi de içerebilen RGBA varyasyonu da bulunmaktadır. Alfa değeri azaldıkça renk şeffaflaşır. Alfa kanalı da diğer kanallar gibi 8 bit ile temsil edilir. Bu durumda RGBA modelini kullanan görüntülerde her piksel 32 bit bellek alanına ihtiyaç duyar.

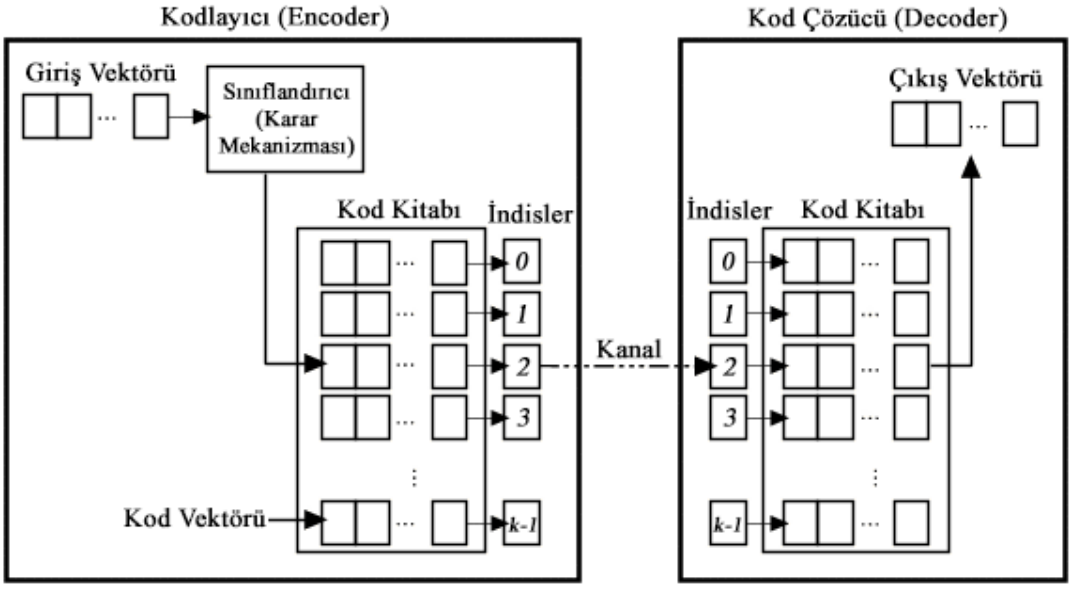
2.3. Vektör Kuantalama

Vektör kuantalama, depolanacak veya transfer edilecek verinin, daha az miktarda veri ile temsil edilebildiği kayıplı bir kodlama tekniğidir. Vektör kuantalama orijinal veri üzerinde aynı zamanda bir kümeleme (sınıflandırma) işlemi yapmaktadır. Bu noktada temel hedef veri boyutu düşürülür iken oluşacak veri kaybının en aza indirilmesi, bununla beraber aynı kümeye ait verilerin benzerliğinin olabildiğince yüksek, farklı kümelere ait verilerin ise benzerliklerinin olabildiğince farklı olmasıdır [16].

Vektör kuantalama, n boyutlu bir R^n vektör uzayındaki giriş vektörlerini k elemanlı ve $C = \{x_i : i = 0,1,2,3,...,k-1\}$ şeklinde ifade edilen sonlu bir vektör kümesine indirgeme işlemini yapar. İşlem sonucu oluşan C kümesine kod kitabı, kümedeki her x_i elemanına ise kod vektörü denir.

Renkli görüntülerde giriş vektörleri piksellere ait renk değerlerinin 3 boyutlu RGB uzayındaki konum bilgileridir. Kuantalama işlemi söz konusu konum bilgileri ve

aralarındaki uzaklıklar üzerinden yapılmaktadır. Giriş vektörleri Şekil 2.3'te gösterilen sınıflandırıcı (karar mekanizması) ile ayrıştırılmak istenen küme sayısına (*k*) tasnif edilir. Renklerin konumları arasındaki uzaklıklara göre benzer olduğuna karar verilen renkler aynı kümeye alınır ve ilgili kümedeki tüm piksellerin, küme merkezindeki tek bir renk (kod vektörü) ile temsil edilebilmesi sağlanır. Elde edilen her kod vektörü de belirlenen bir indis ile kod kitabına işlenir. Böylece sayısal görüntü, söz konusu indislerden meydana gelen bir matris ile temsil edilebilmiş olur. Sonuçta görüntü içerisinde veri kaybı kaçınılmaz olsa da toplam boyut indirgenmiş olur. Başka bir deyişle veri sıkıştırılmış olur. Diğer taraftan ayrıştırılmak istenen küme sayısı *k* arttıkça sıkıştırma başarısı düşer.

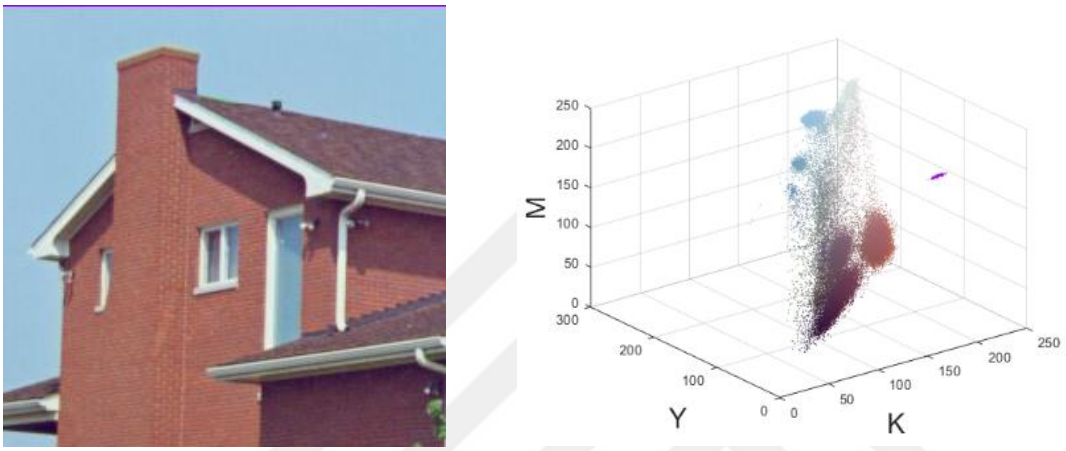


Şekil 2.3. Vektör kuantalama kodlama/kod çözme diyagramı

Vektör kuantalama ile kodlanmış veri kendi başına kullanılabilir halde değildir. Veri tekrar kullanılmak isteniyorsa öncelikle kod çözücü (decoder) ile çözme işlemine tabi tutulup giriş vektörü formatında bir çıkış vektörü elde edilmelidir. Kod çözme işleminde de kodlama işleminin tersi yapılmaktadır. İndislerden meydana gelen kodlanmış veri, indislerin kod kitabındaki karşılık gelen kod vektörü değerlerine göre tekrar yapılandırılır ve giriş vektöründeki format ile çıkışa verilir.

Kuantalama işleminin görüntüler üzerinde uygulanmasında ise sınıflandırma yöntemleri kullanılarak görüntü içerisindeki renk sayısının düşürülmesi (renk indirgemesi) amaçlanmaktadır. Renk indirgemesi sayesinde her renk için temsili bir etiket tanımlanabilir. Etiketleme işlemi resmin sıkıştırılmasında büyük avantaj sağlamaktadır. Atanan etiketlerin

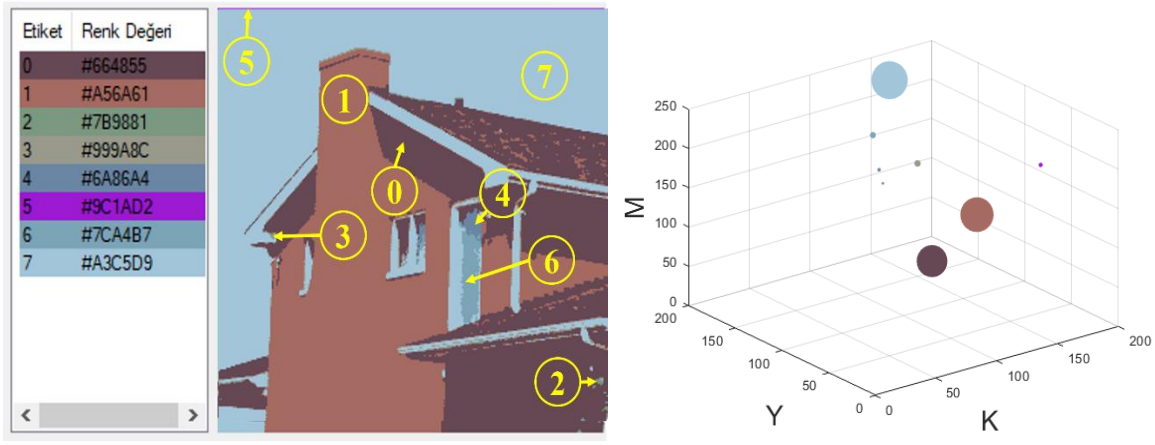
hangi renge karşılık geldikleri kod kitabı içerisinde bir kereliğine eklenmektedir. Böylece renkli bir görüntü sıkıştırılırken resimdeki her piksele ait renk bilgisinin ihtiyaç duyduğu 3'er byte'lık alanlar, 1'er byte'lık etiket numaraları ile temsil edilerek bellek alanından tasarruf sağlanabilir. Örnek olarak, Ev görüntüsünün Şekil 2.4 (a)'da orijinali ve Şekil 2.4 (b)'de ise RGB uzayındaki renk dağılımı verilmiştir. Orijinal görüntünün renk dağılımı incelendiğinde içerisinde birçok farklı rengi barındırdığı görülebilmektedir.



a) b)

Şekil 2.4. Ev görüntüsü a) Orijinal b) Renk dağılımı

Şekil 2.5 (a)'da ise Ev görüntüsüne ayrıştırma uygulanması sonucunda oluşan kümeler ve atanan renk etiketleri, Şekil 2.5 (b)'de ise ayrıştırılmış görüntüye ait renk dağılımı verilmiştir. Ayrıştırılmış görüntünün renk dağılımı incelendiğinde görüntü üzerinde bir kuantalama işlemi yapıldığı, dolayısıyla renk indirgemesi oluştuğu açıktır.



a) b)

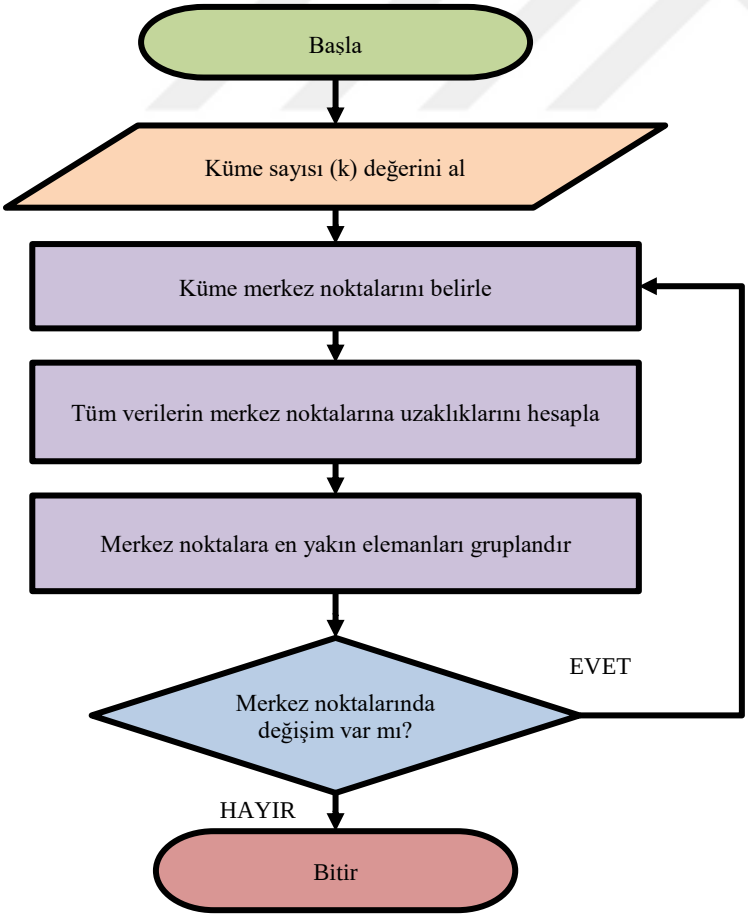
Şekil 2.5. İndirgenmiş ev görüntüsü a) Ayrıştırma işlemi ve atanan renk etiketleri b) Renk dağılımı

2.4. K-Ortalamalar Algoritması

K-Ortalamalar algoritması 1967 yılında MacQueen tarafından geliştirilmiş, veri kümelemede kullanılan en yaygın gözetimsiz öğrenme yöntemlerinden biridir [38]. Veriyi istenen sayıda (*k* adet) kümeye ayırır. K-Ortalamalar algoritması için önerilen amaç fonksiyonu,

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \left\| X_i^j - C_j \right\| \tag{2.2}$$

şeklindedir. Burada *k*, bölünmek istenilen küme sayısını, *n* toplam veri sayısını, *i* ise *i*'inci veri noktasını belirtmektedir. $\left\| X_i^j - C_j \right\|$ ifadesi ise veriler arasındaki uzaklık ölçüsüdür. Burada C_j *j*' inci küme merkezini belirtmektedir [39]. K-Ortalamalar algoritmasının akış şeması Şekil 2.6'daki gibidir.



Şekil 2.6. K-ortalamalar algoritması akış şeması

K-ortalamalar algoritması, elde edilecek kümeleri, küme içi benzerliklerin en yüksek, kümeler arası benzerliklerin ise en düşük olacak şekilde belirleme eğilimindedir. Başka bir deyişle *k* adet kümeyi belirlerken amaç fonksiyonunu minimize etmeyi hedefleyerek ideal kümeleme sonucuna ulaşmaya çalışır. Küme benzerliği kümedeki nesnelerin ortalama değeri ile ölçülür ve kümenin ağırlık merkezidir [38].

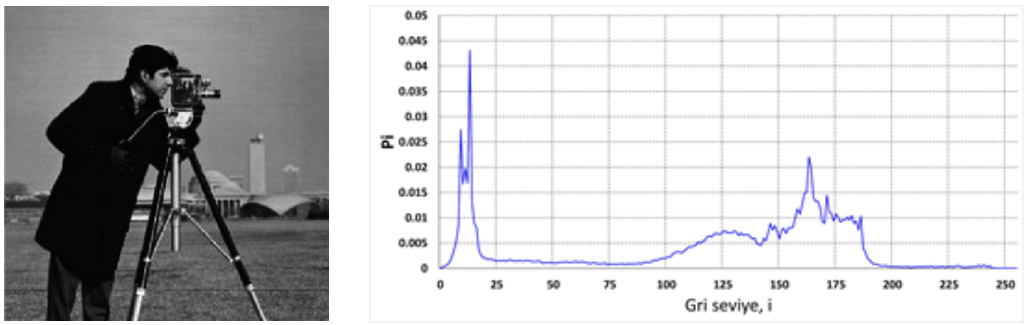
2.5. Görüntü Histogramı

Histogram, sayısal görüntülerdeki piksellerin dağılımını gösteren bir frekans grafiği ve önemli bir bilgi temsil yöntemidir [40]. Başka bir ifade ile histogram, $\{0,1,2,...,(L-1)\}$ yeğinlik değerine sahip bir sayısal görüntünün ayrık olasılık dağılım fonksiyonudur. Buna göre $M \times N$ boyutlu gri ölçekli bir görüntüde *i.* gri seviyenin olasılığı,

$$p_i = \frac{h_i}{(M \times N)} \tag{2.3}$$

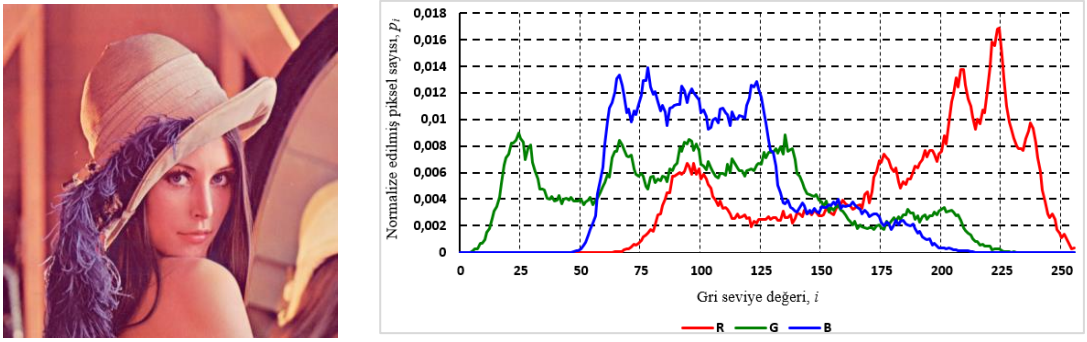
şeklinde hesaplanır. Burada *M*, görüntünün piksel cinsinden yüksekliğini, *N* piksel cinsinden genişliğini, h_i, ise *i.* gri seviyesine sahip piksel sayısını belirtmektedir. Gri seviyeli görüntülerde tek kanal olduğundan histogram tek bir diziden oluşurken, RGB tabanlı renkli görüntülerde ise üç ayrı renk kanalı olması dolayısıyla üç diziden oluşmaktadır.

Şekil 2.7(a)'da gri ölçekli kameraman görüntüsü ve Şekil 2.7(b)'de görüntüye ait histogram, Şekil 2.8(a)'da renkli Lena görüntüsü ve Şekil 2.8(b)'de ise görüntüye ait üç kanallı histogram verilmiştir.



a) b)

Şekil 2.7. Kameraman a) Orijinal b) Histogram



a) b)

Şekil 2.8. Lena a) Orijinal b) Histogram

Bir sayısal görüntünün yalnızca histogramı incelenerek görüntü hakkında önemli bilgilere ulaşılabilir. Bu yönüyle histogram, görüntü işlemenin birçok uygulamasında başvurulan bir istatistik bilgisidir.

2.6. Görüntü Eşikleme

Eşikleme, görüntü bölütlemede kullanılan temel bir sınıflandırma tekniğidir. Görüntü eşiklemede genel amaç bir renk kanalına ait eşik değerleri belirleyerek ilgili eşik değerlerine göre görüntüyü sınıflara ayırmaktır.

Gri seviye görüntüler için geliştirilmiş eşikleme algoritmalarından olan Otsu ve Kapur algoritmaları bu alanda yaygın olarak tercih edilen yöntemlerdir [41]. Her iki yöntemde de eşik sayısı kullanıcı tarafından seçilmekte olup, eşik noktalarının yerleri hesaplanırken histogram bilgisinden faydalanılmaktadır [34, 35].

Kullanıcı tarafından seçilen *m* adet eşik ile gri seviyeli görüntü *m+1* adet kümeye ayrıştırılır. Fakat eşik sayısının artması, başka bir deyişle ayrıştırma sonucunda istenen küme sayısının artması, hesaplamaya logaritmik olarak artan bir maliyet getirmektedir [41]. Eşik değerinin seçiminde Otsu yöntemi, sınıflar arası varyansı maksimum yapan değeri temel alınmaktadır. Tek eşikli Otsu yönteminin amaç fonksiyonu

$$J\left(t_1\right)=\sigma_0+\sigma_1 \tag{2.4}$$

olarak tanımlanmıştır.

Burada σ_0 ve σ_1 parametreleri sınıflar arası varyansı temsil etmekte olup,

$$\sigma_0 = \omega_0 \left(\mu_0 - \mu_T \right)^2 \text{ ve } \sigma_1 = \omega_1 \left(\mu_1 - \mu_T \right)^2 \tag{2.5}$$

şeklinde hesaplanır. Sınıfların Eş. 2.5'te verilen istatistiksel bilgileri ise

$$\omega_0 = \sum_{i=0}^{t_1-1} p_i \text{ ve } \mu_0 = \sum_{i=0}^{t_1-1} \frac{ip_i}{\omega_0} \tag{2.6}$$

$$\omega_1 = \sum_{i=t_1}^{L-1} p_i \text{ ve } \mu_1 = \sum_{i=t_1}^{L-1} \frac{ip_i}{\omega_1} \tag{2.7}$$

şeklinde olmaktadır. Ayrıca görüntünün gri seviye ortalaması μ_T

$$\mu_T = \sum_{i=0}^{L-1} ip_i \tag{2.8}$$

olarak hesaplanmaktadır. Otsu yöntemi iki veya çok seviyeli eşikleme uygulamalarına da genişletilebilmektedir [42]. Otsu yöntemine göre gri seviyeli görüntüde çok seviyeli eşikler, Eş. 2.4'teki amaç fonksiyonunun,

$$J\left(t_1, t_2, ..., t_m\right) = \sigma_0 + \sigma_1 + \sigma_2 + ... + \sigma_m \tag{2.9}$$

$$\sigma_0 = \omega_0 \left(\mu_0 - \mu_T \right)^2 \tag{2.10}$$

$$\sigma_1 = \omega_1 \left(\mu_1 - \mu_T \right)^2 \tag{2.11}$$

$$\sigma_2 = \omega_2 \left(\mu_2 - \mu_T \right)^2 \tag{2.12}$$

$$\sigma_m = \omega_m \left(\mu_m - \mu_T \right)^2 \tag{2.13}$$

$$\omega_0 = \sum_{i=0}^{t_1-1} p_i \text{ ve } \mu_0 = \sum_{i=0}^{t_1-1} \frac{ip_i}{\omega_0} \tag{2.14}$$

$$\omega_1 = \sum_{i=t_1}^{t_2-1} p_i \text{ ve } \mu_1 = \sum_{i=t_1}^{t_2-1} \frac{ip_i}{\omega_1} \tag{2.15}$$

$$\omega_2 = \sum_{i=t_2}^{t_3-1} p_i \text{ ve } \mu_2 = \sum_{i=t_2}^{t_3-1} \frac{ip_i}{\omega_2} \tag{2.16}$$

$$\vdots \qquad \vdots \qquad \vdots$$

$$\omega_m = \sum_{i=t_m}^{L-1} p_i \text{ ve } \mu_m = \sum_{i=t_m}^{L-1} \frac{ip_i}{\omega_m} \tag{2.17}$$

şeklinde genişletilmesi ile hesaplanabilir. Burada sınıflar arası varyansı en büyük yapan eşik değerleri çözüm kümesi olarak belirlenmiş olur. Her durumda $t_1 < t_2 < \ldots < t_m$ koşulunun sağlanmış olması gerekmektedir [42].

Temelde gri ölçekli görüntüler için önerilen eşikleme metodunun renkli görüntülerde kullanılması üzerine farklı çalışmalar yapılmıştır. Görüntülerin renk uzayı yardımıyla ayrıştırılması yöntemi bunlardan biridir [28].

2.7. Görüntülerin Renk Uzayı Yardımıyla Ayrıştırılması

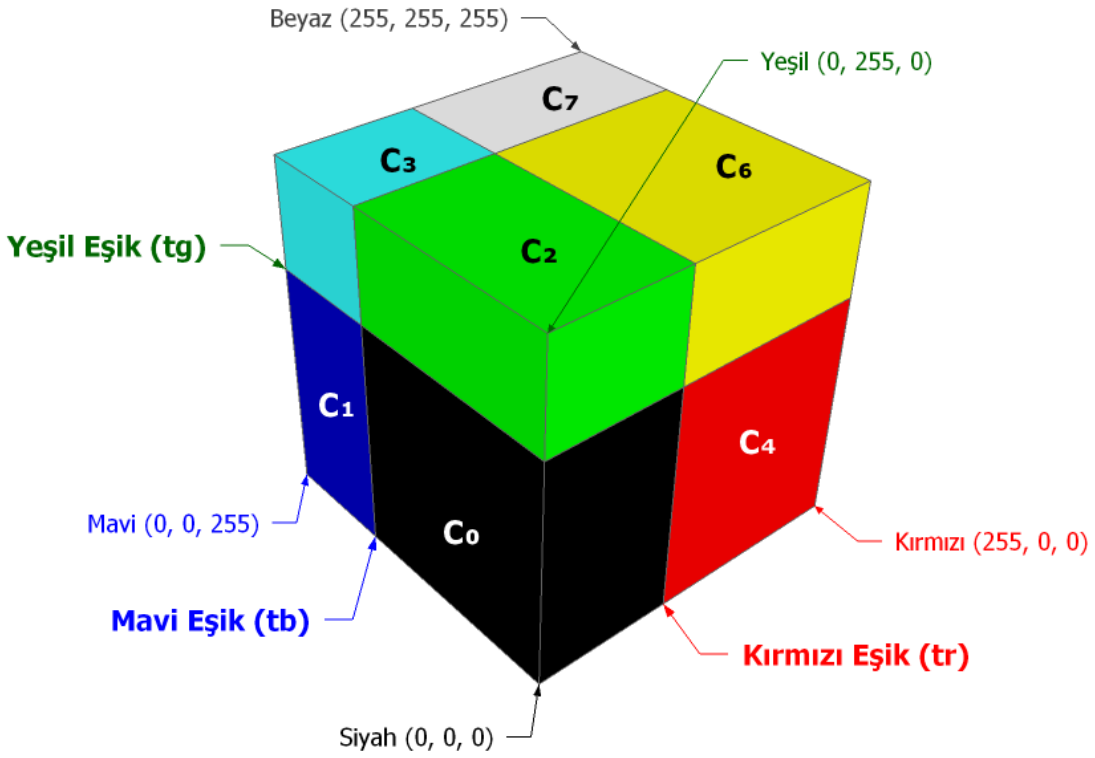
Renk uzayı yardımıyla ayrıştırma algoritması, gri seviyeli görüntüler için ayrıştırma problemlerinde olumlu sonuçlar veren Otsu ve Kapur metotları temel alınarak geliştirilmiş bir renkli görüntü ayrıştırma tekniğidir. Bu yöntemde eşik değerleri her bir kanal için ayrı ayrı hesaplanmakta ve bunların kesişimleri kullanılarak RGB uzayı alt küp veya prizmaya bölümlenmektedir. Oluşan her bir parça da bir küme (sınıf) olarak yorumlanmıştır [28].

Renk uzayı ile ayrıştırmada kanal başına *m* adet eşik için $k = (m+1)^3$ adet küme oluşmaktadır. Oluşan her küme, kendi prizma sınırları içerisinde kalan tüm piksellerin renk ortalaması alınarak tek bir renk ile temsil edilmektedir. Böylece görüntü üzerinde kuantalama, başka bir ifade ile renk indirgemesi yapılmış olur.

Şekil 2.9'da tek eşikli (m=1) bölümlenmiş bir renk uzayı örneği verilmiştir. Görüntüye ait her piksel, her üç renk kanalı için de hesaplanmış olan eşik değerlerine göre Çizelge 2.1'de

belirtilen kurallar üzerinden bir sınıfa dahil edilmiştir. Buna göre k=8 adet küme oluştuğu görülmektedir. Burada R, G ve B değerleri mevcut piksele ait sırasıyla kırmızı, yeşil ve mavi kanal değerlerini; t_r, t_g ve t_b ise bu kanallar için hesaplanmış olan eşik değerlerini temsil etmektedir.

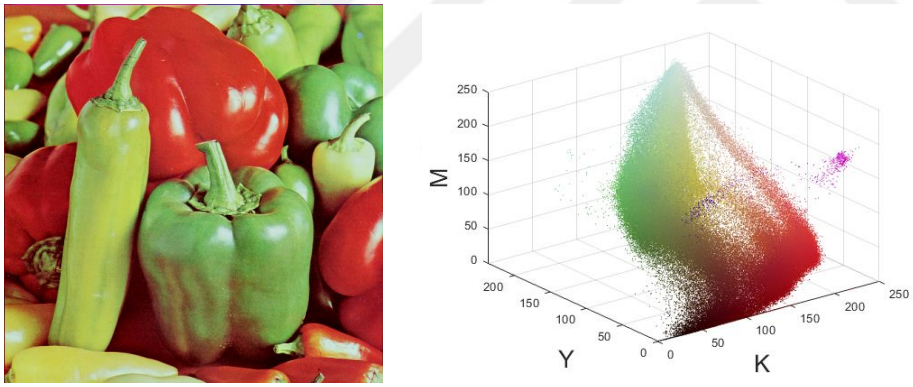
Şekil 2.10(a) ve Şekil 2.8(a)'da Biberler ve Lena orijinal görüntüleri, Şekil 2.10(b) ve Şekil 2.11'de ise görüntülere ait renk dağılım grafikleri verilmiştir. Orijinal görüntülerin renk dağılımları incelendiğinde birçok farklı renkten meydana geldikleri görülebilmektedir.



Şekil 2.9. Renk uzayının bölünmesi ve oluşan kümeler (C_0 – C_7)

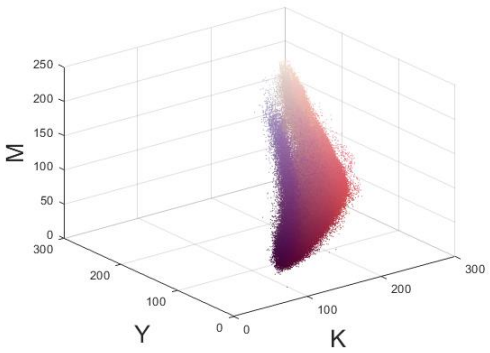
Çizelge 2.1. Renk uzayının bölümlenmesi

Kümeye Atanan Etiket	Atama Kuralı	Atanan Renk	İkili kod
C_0	$R \leq t_r$ && $G \leq t_g$ && $B \leq t_b$	0x000000	000
C_1	$R \leq t_r$ && $G \leq t_g$ && $B \geq t_b$	0x0000FF	001
C_2	$R \leq t_r$ && $G \geq t_g$ && $B \leq t_b$	0x00FF00	010
C_3	$R \leq t_r$ && $G \geq t_g$ && $B \geq t_b$	0x00FFFF	011
C_4	$R \geq t_r$ && $G \leq t_g$ && $B \leq t_b$	0xFF0000	100
C_5	$R \geq t_r$ && $G \leq t_g$ && $B \geq t_b$	0xFF00FF	101
C_6	$R \geq t_r$ && $G \geq t_g$ && $B \leq t_b$	0xFFFF00	110
C_7	$R \geq t_r$ && $G \geq t_g$ && $B \geq t_b$	0xFFFFFF (Beyaz)	111



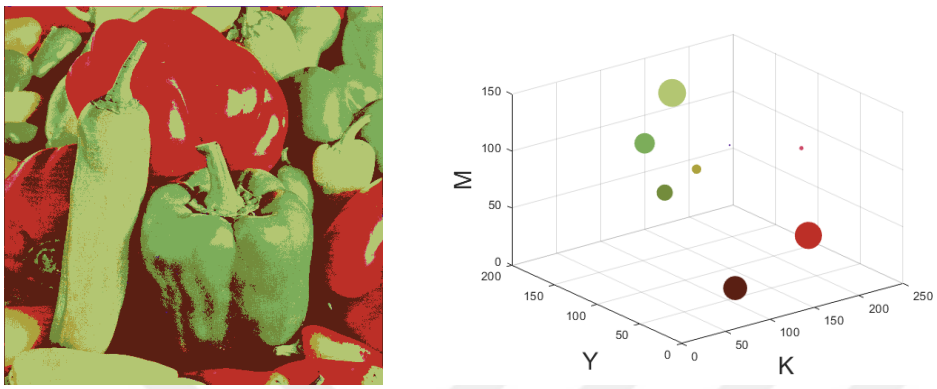
a) b)

Şekil 2.10. Biberler a) Orijinal b) Renk dağılımı



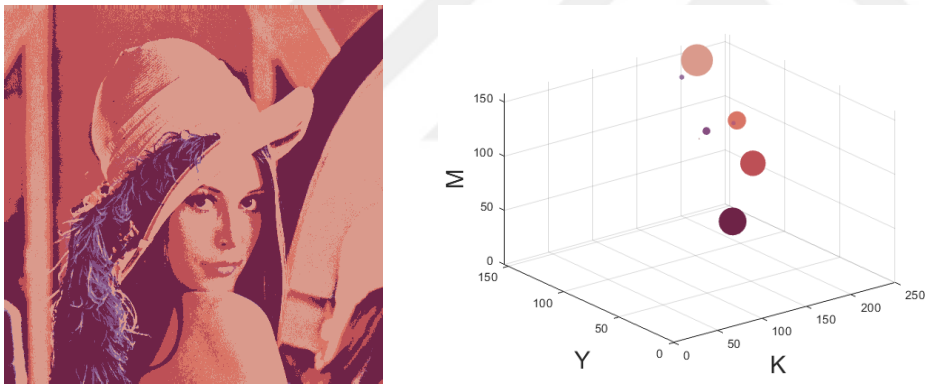
Şekil 2.11. Lena renk dağılımı

Şekil 2.12(a) renk indirgemesi yapılmış Biberler görüntüsünü, Şekil 2.12(b) ise renk dağılımını göstermektedir. Renk indirgemesi yapılmış Lena görüntüsü Şekil 2.12(a)'da, renk dağılımı ise Şekil 2.12(b)'de gösterilmiştir. Renk uzayı yardımıyla ayrıştırılmış görüntülerin renk dağılımları incelendiğinde resimlerin renk sayısının indirgenmiş olduğu açıktır.



a) b)

Şekil 2.12. Sınıflandırılmış biberler a) İndirgenmiş b) Renk dağılımı



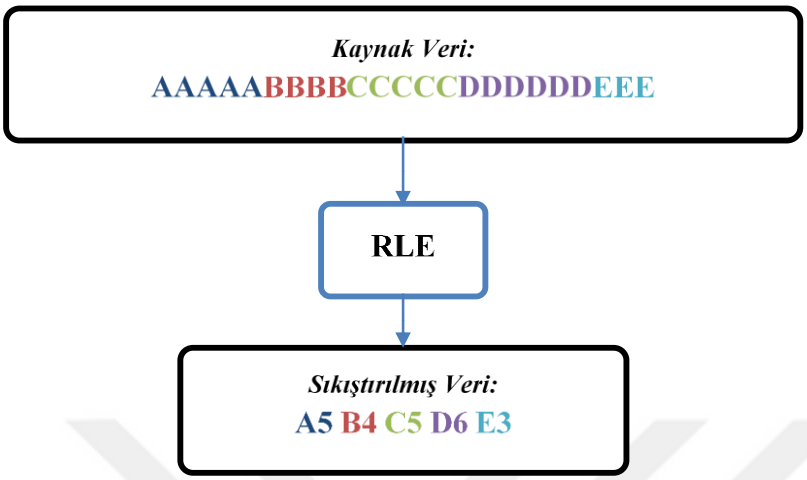
a) b)

Şekil 2.13. Sınıflandırılmış Lena a) İndirgenmiş b) Renk dağılımı

2.8. Tekrarlı Uzunluk Kodlaması (Run-Length Encoding - RLE)

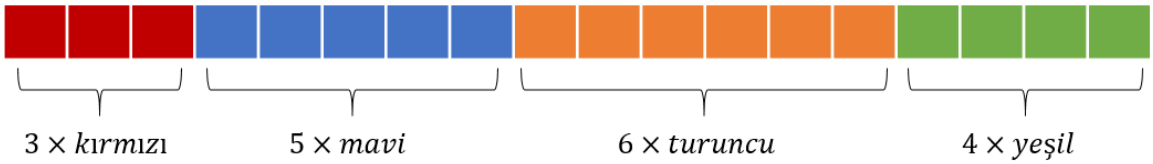
Tekrarlı uzunluk kodlaması, ilk olarak 1967 yılında televizyon sinyallerinin iletiminde kullanılmış olan, birbirini takip eden aynı sembollerin hepsinin saklanması yerine, sadece sembol ve tekrar sayılarının saklanması mantığına dayalı bir kayıpsız sıkıştırma algoritmasıdır [20]. Şekil 2.14'teki örnek, RLE algoritmasının basit bir metin üzerinde çalışma mantığını göstermektedir. Buna göre toplam 23 karakter uzunluğundaki kaynak veri, içerdiği tekrarlı karakterler ve karakterlerin tekrar sayıları kullanılarak RLE ile sıkıştırılmış

ve toplamda 10 karaktere indirilmiştir. Kaynak verinin art arda gelen sıralı tekrarlar içermesi RLE algoritmasının başarımını olumlu yönde etkilemektedir.



Şekil 2.14. RLE algoritması

RLE algoritmasının görüntü sıkıştırmada kullanımına ise Şekil 2.15'deki yapı örnek verilebilir. Şekil 2.15'de verilen resmin 24 bit RGB renk modunda, bir satırının 1x18 boyutunda olan bir görüntü olduğu varsayılırsa, bu boyuttaki bir ham görüntü için gereken bellek alanı 18x3 = 54 byte'tır.



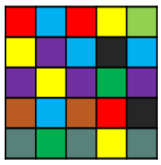
Şekil 2.15. 1x18 boyutunda 24 bit'lik görüntü örneği

Görüldüğü üzere resimdeki pikseller sıralı ve ardışık tekrarlar içermektedir. Her renk için bir etiket belirlenmiş olursa, renkli görüntü RLE algoritması yardımıyla etiket-tekrar ikilileri şeklinde temsil edilebilir. Başka bir ifade ile daha küçük boyutlara sıkıştırılmış olur. Örneğin, K, M, T ve Y etiketlerinin sırasıyla kırmızı, mavi, turuncu ve yeşili temsil ettiği varsayılırsa, sıkıştırılmış görüntü "K3 M5 T6 Y4" şeklinde 8 byte olarak saklanabilir.

Sıralı renk tekrarlarının olduğu homojen renk dağılımlı görüntülerde RLE algoritması başarılı bir sıkıştırma yapabilmektedir. Diğer taraftan tekrarların az olduğu veya renklerin homojen bir dağılım göstermediği görüntülerde algoritmanın başarısı düşük kalabilmektedir.

Şekil 2.16(a) yüksek başarıma bir örnek iken, Şekil 2.16(b) ise düşük başarıma bir örnek teşkil etmektedir.

a) K3 Y2 S2 M3 C5 V4 F1 D5 (CR: 75/16 = **4.69)**



b) K1 M1 K1 S1 Y1 S1 C1 M1 F1 M1 C1 S1 C1 Y1 C1 V1 M1 V1 K1 F1 D1 Y1 D1 S1 D1 (CR: 75/50 = **1.5)**

Şekil 2.16. Tekrarlı renkler ve etiketleri (K: kırmızı, Y: açık yeşil, S: sarı, M: mavi, C: mor, G: kahverengi, F: siyah, D: su yeşili) a) yüksek başarım b) düşük başarım

2.9. Performans Ölçütleri

Görüntü sıkıştırma yaklaşımlarında performans değerlendirme kriterleri, yöntemin kayıplı veya kayıpsız olmasına göre değişmektedir. Kayıpsız sıkıştırma yapılan görüntülerde, sıkıştırma oranı (compression ratio: CR) tek başına bir performans ölçütü olabilmektedir ve sıkıştırma başarımının dosya boyutu esaslı tanımlanmasında kullanılan basit bir ölçüm yöntemidir. Genel anlamda verinin sıkıştırılmamış haldeki boyutunun, sıkıştırılmış haldeki boyutuna oranıdır [17] ve

$$Sıkıştırma\ Oranı\ (CR) = \frac{Sıkıştırılmamış\ Boyut}{Sıkıştırılmış\ Boyut} \tag{2.18}$$

şeklinde hesaplanır. Örneğin, sıkıştırılmamış boyutu 60 byte ve sıkıştırılmış boyutu 5 byte olan bir veri için sıkıştırma oranı 60/5 = 12.00 olarak ifade edilir.

Diğer taraftan, kayıplı sıkıştırma yapılan görüntülerde sıkıştırma oranının yanında orijinal görüntü ile sıkıştırılmış görüntü arasındaki benzerliğin de ölçülmesi gerekmektedir. Ortalama kare hatası (mean squared error: MSE) ve tepe sinyal-gürültü oranı (peak signal-to-noise ratio: PSNR) benzerlik ölçüm yöntemlerinden bazılarıdır [43]. PSNR desibel (dB) ile ifade edilir ve iki görüntü arasındaki benzerlik arttıkça PSNR da artar [18]. MSE ve PSNR ölçütleri sırasıyla,

$$MSE = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} [f(x_i, y_j) - g(x_i, y_j)]^2 \tag{2.19}$$

$$PSNR = 10 \log \frac{255^2}{MSE} \tag{2.20}$$

şeklinde tanımlanmıştır. Burada *f(x, y)* orijinal referans görüntüyü; *g(x, y)* ise sıkıştırılmış görüntüyü ifade etmekte olup, *i* ve *j* ise mevcut pikselin koordinatlarını göstermektedir [44]. Denklemlerden anlaşılacağı üzere MSE sıfıra yaklaştıkça PSNR değeri de sonsuza yaklaşır. Benzerlik derecesinin arttığı durumda PSNR değerinin de arttığı buradan görülebilmektedir.

PSNR tek kanallı çalıştığı için renkli görüntülerde kullanılmak istenirse her renk kanalı için hesaplanmış MSE değerlerinin ortalaması PSNR denkleminde yerine konulur. Buna göre üç kanallı RGB görüntülerde MSE ve PSNR hesaplamaları

$$MSE_{rgb} = \frac{MSE_r + MSE_g + MSE_b}{3} \tag{2.21}$$

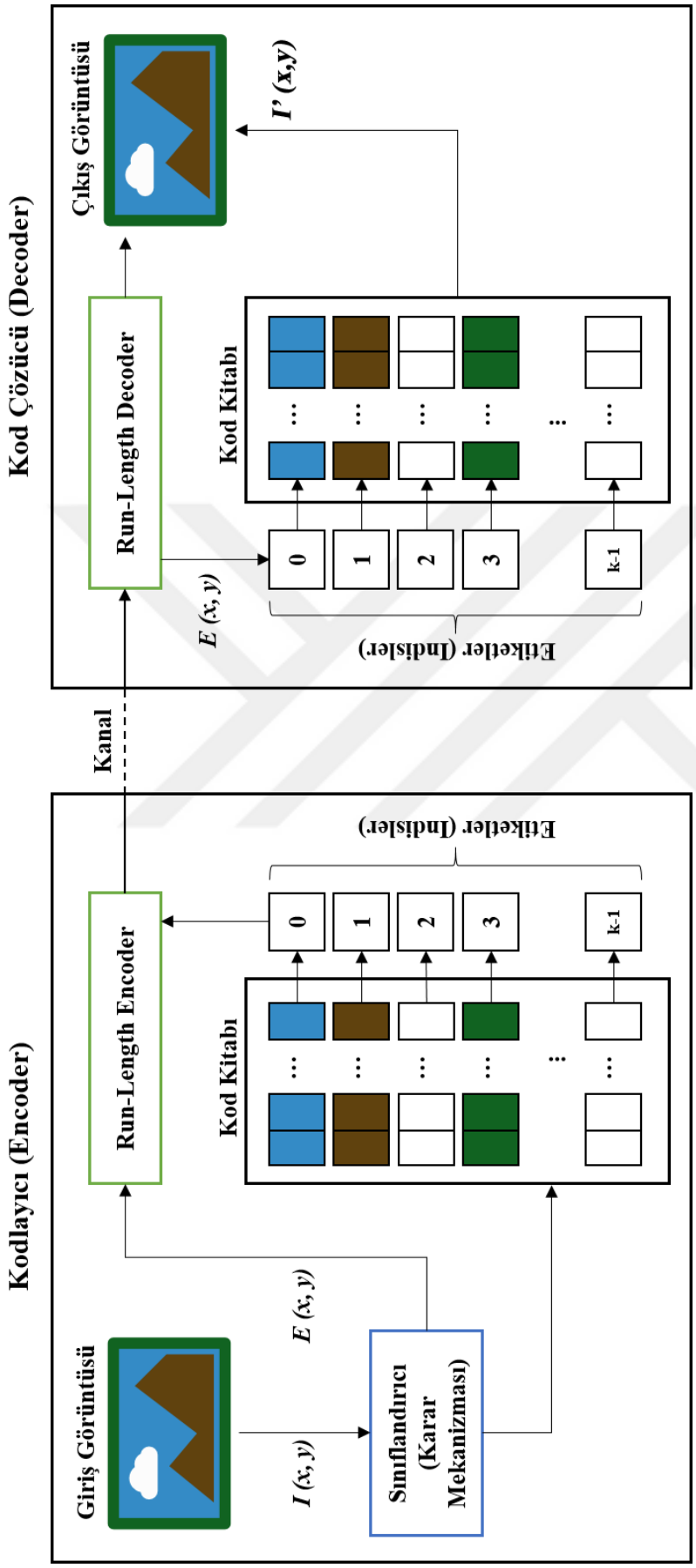
$$PSNR_{rgb} = 10 \log \frac{255^2}{MSE_{rgb}} \tag{2.22}$$

şeklinde olmaktadır [19]. Eş. 2.21'de verilen MSE_r, MSE_g ve MSE_b sırasıyla kırmızı, yeşil ve mavi kanallarına ait MSE değerlerini, MSE_{rgb} ise üç kanala ait MSE değerlerinin ortalamasını göstermektedir. Eş. 2.22'deki $PSNR_{rgb}$ ise görüntüye ait ortalama PSNR değeridir.

3. RLE VE AYRIŞTIRMA TABANLI GÖRÜNTÜ SIKIŞTIRMA

Bu çalışmada RLE ve görüntü ayrıştırma temelli yeni sıkıştırma algoritması tasarlanmıştır. Öncelikle, sınıflandırma metotları kullanılarak görüntüde renk indirgemesi yapılmış ve işlem sonucunda kod kitabı ve etiket bilgileri elde edilmiştir. Sonrasında ise etiket bilgileri RLE algoritması ile sıkıştırılarak kod kitabı ile birlikte bir dosyaya kaydedilmiş ve dosya boyutu küçültülmüştür. Başka bir ifade ile bilgi kayıplı renk indirgeme yaklaşımları ile kayıpsız bir sıkıştırma tekniği olan RLE birleştirilerek yeni bir kayıplı sıkıştırma yöntemi geliştirilmiştir [45]. Geliştirilen yöntem, Şekil 3.1'de gösterildiği üzere kodlayıcı ve kod çözücü şeklinde iki modülden oluşmaktadır. Kodlayıcı modülde öncelikle orijinal görüntüye ayrıştırma teknikleri uygulanarak görüntüdeki pikseller sınıflara ayrıştırılır. Süreç sonunda hangi pikselin hangi sınıfa ait olduğunu gösteren bir $E(x, y)$ etiket haritası ve etiketlere karşılık gelen renkleri belirten bir kod kitabı elde edilir. Akabinde, elde edilen $E(x, y)$ etiket haritası, RLE sıkıştırma algoritması uygulanarak sıkıştırılır. Nihai aşamada görüntüye ait genişlik-yükseklik bilgisi, kod kitabı ve sıkıştırılmış veri birleştirilerek biçimli bir dosya haline getirilmektedir. Elde edilen dosya istendiği takdirde belleğe kaydedilebilmekte ve kod çözücü modül ile bellekten tekrar okunup Bitmap görüntüsü olarak görüntülenebilmektedir. Önerilen yaklaşım, sınıflandırıcı algoritma olarak renk uzayı ile ayrıştırma yöntemini esas almaktadır. Sıkıştırma başarımı, renk uzayı yönteminde seçilen eşik sayısından etkilenmektedir. Eşik sayısı arttığında sıkıştırma oranı düşerken PSNR artmakta, başka bir deyişle çıkış görüntüsü orijinale daha benzer hale gelmektedir. Dolayısıyla önerilen yaklaşımın, eşik sayısı vasıtasıyla sıkıştırma oranına müdahale edilebilirlik özelliği de bulunmaktadır. Bununla beraber önerilen yöntem, diğer sınıflandırıcı çözümleri ile de çalışabilir esnekliktedir. Örneğin, K-ortalamalar, bulanık C-ortalamalar (FCM), LBG gibi algoritmalar önerilen yönteme kolayca entegre edilebilir nitelikte sınıflandırıcılardır.

Tasarlanan strateji RLE algoritması ile sıkıştırma yapmaktadır. Sıkıştırma aşamasında renk bilgileri değil, ayrıştırma sonucu elde edilen etiket bilgileri kullanılmıştır. Sıkıştırmada etiket bilgilerinin kullanılması başarımın artmasında önem arz etmektedir. Çünkü renk bilgileri kullanıldığı takdirde her piksel için 3 byte bellek alanına ihtiyaç duyulacak iken, etiket bilgilerinin esas alınması sayesinde söz konusu uzunluk 1 byte seviyesine inmektedir. Bu sayede sıkıştırma aşaması başlamadan dahi bellek alanından tasarruf sağlanmakta, ayrıca daha küçük boyutlu veri ile çalışılacağı için işlem performansı da artmaktadır.

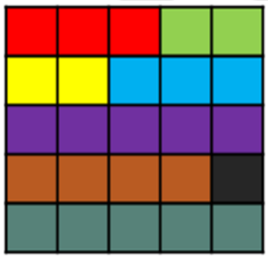


Şekil 3.1. RLE ve ayrıştırma tabanlı görüntü sıkıştırma algoritması

3.1. Kodlayıcı (Encoder) Modül

Kodlayıcı (encoder) modülün görevi *I(x, y)* giriş görüntüsünü seçilen sınıflandırıcı metot (karar mekanizması) ve RLE kodlayıcı algoritması yardımıyla sıkıştırmak ve biçimlendirilmiş bir dosya halinde kanal üzerinden belleğe kaydetmektir.

Kodlayıcı modülün birinci aşamasında giriş görüntüsü olan *I(x, y)*, seçilen sınıflandırıcı algoritma yardımıyla ayrıştırılmakta ve işlem sonucunda piksellerin atandığı bölgelere ait etiketlerden meydana gelen, görüntü ile aynı boyutlara sahip iki boyutlu bir *E(x, y)* etiket haritası elde edilmektedir. *E(x, y)* etiket haritasında etiketler 0'dan başlamak üzere sayısal değerlerden oluşmaktadır. Bu durum etiket-renk ilişkilendirmesinde tutarlılık ve performans sağlamasının yanı sıra 255 adede kadar farklı etiketin 1'er byte ile temsil edilebilmesini de sağlamaktadır. Böylece bellek ve kaynak kullanımını en aza inmektedir. Şekil 2.16(a)'daki görüntünün sayısal etiketler ile kullanımına ait örnek Şekil 3.2'de gösterilmiştir. Şekil 3.2(a)'da renk indirgemesi yapılmış 5×5 boyutlarında renkli bir görüntü ve etiket-renk ilişkileri, Şekil 3.2(b)'de ise elde edilen *E(x, y)* etiket haritası gösterilmiştir.



a) 0: Kırmızı, **1**: Yeşil, **2**: Sarı, **3**: Mavi, **4**: Mor, **5**: Kahverengi, **6**: Siyah, **7**: Su Yeşili

$$E(x,y)=\begin{bmatrix}0&0&0&1&1\\2&2&3&3&3\\4&4&4&4&4\\5&5&5&5&6\\7&7&7&7&7\end{bmatrix}$$

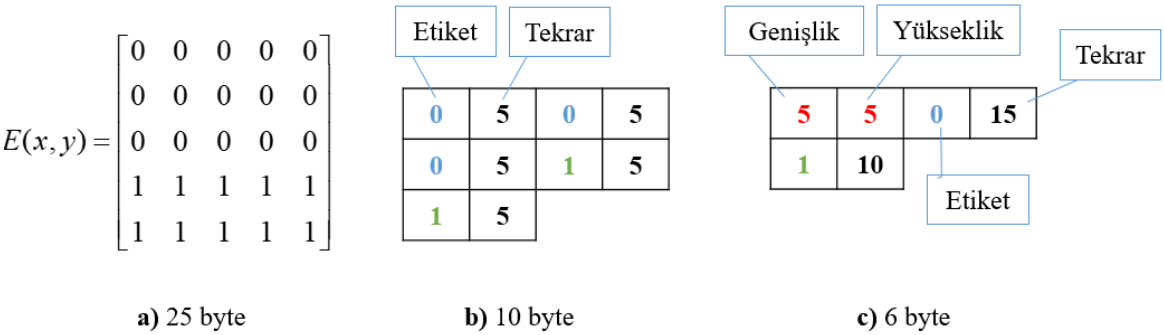
b) Etiket haritası

Şekil 3.2. Sayısal etiketleme a) Görüntü ve etiket-renk ilişkileri b) *E(x, y)* etiket haritası

Kodlayıcı modülün ikinci aşamasında kod kitabı hazırlanmaktadır. Kod kitabı, sayısal etiketlere sırasıyla karşılık gelen renk değerlerinden oluşan bir sözlük veya renk dizisi olarak da düşünülebilir. Burada renk değerleri kod kitabına 32 bit tamsayı olarak işlenmektedir.

Üçüncü aşamada ise *E(x, y)* etiket haritası, RLE algoritması kullanılarak sıkıştırılır. Diğer taraftan, bu aşamada görüntüye ait genişlik ve yükseklik bilgisinden de faydalanılmaktadır. Genişlik ve yükseklik bilgisi RLE sıkıştırma başarımının artmasında önem arz etmektedir. Özellikle genişlik bilgisi, etiket haritasında birden çok satıra ulaşan tekrarlar oluşması

durumunda, tekrar sayısının tamamının tek bir etiket-tekrar ikilisi şeklinde temsil edilebilmesini sağlamaktadır. Aksi takdirde her satırdaki tekrar sayıları için ayrı ayrı etiket-tekrar ikilileri oluşturulması gerekeceğinden sıkıştırma başarımı olumsuz etkilenecektir. Bu durum Şekil 3.3'te gösterilmiştir. Kodlayıcı modülün son aşamasında ise görüntüye ait genişlik ve yükseklik bilgisi, oluşturulan kod kitabı ve RLE ile sıkıştırılmış etiket haritası birleştirilerek biçimli bir dosya halinde belleğe kaydedilmektedir.

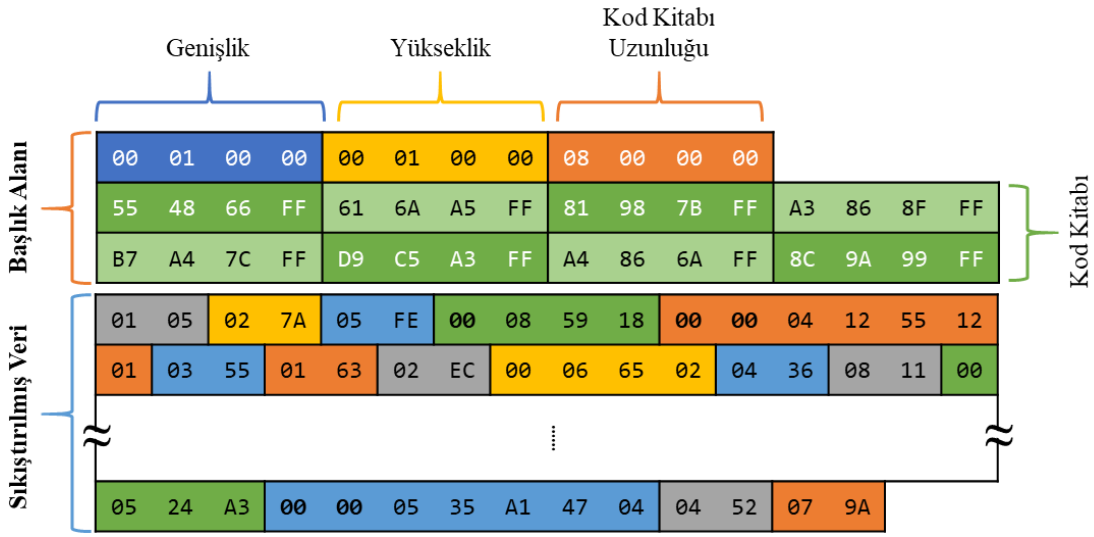


a) 25 byte **b)** 10 byte **c)** 6 byte

Şekil 3.3. Genişlik-yükseklik bilgisinin RLE sıkıştırmasına etkisi a) E(x, y) etiket haritası b) Genişlik-yükseklik bilgisinin olmadığı durumdaki sıkıştırma c) Genişlik-yükseklik bilgisinin olduğu durumdaki sıkıştırma

3.2. Dosya Yapısı

Kaydedilen dosya yapısında, Şekil 3.4'te örneklendirildiği üzere başlık alanı ve sıkıştırılmış veri alanı olmak üzere 2 kısım bulunmaktadır. Başlık alanında görüntünün genişlik-yükseklik bilgisi ve kod kitabı bilgileri, sıkıştırılmış veri alanında ise görüntüyü oluşturacak etiket haritasının etiket-tekrar ikilileri şeklinde RLE ile kodlanmış hali yer almaktadır. Dosya içerisinde sayısal veriler depolanırken x86 tabanlı işlemcilerin çalışma prensibine uygun olarak küçük sonlu (little endian) byte sıralaması kullanılmıştır [46]. Küçük sonlu byte sıralamasında en küçük değerlikli byte (least significant byte: LSB) en küçük bellek adresinde, en büyük değerlikli byte (most significant byte: MSB) ise en büyük bellek adresinde olacak şekilde ters sıralı olarak depolanırlar [46]. Örneğin, onaltılık tabanlı 0x26FE1C sayısı küçük sonlu sıralamaya göre 1C FE 26 şeklinde saklanmaktadır.



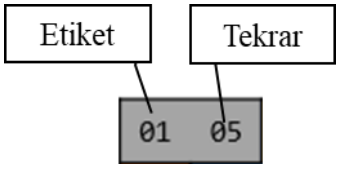
Şekil 3.4. Kaydedilen dosya yapısı

Başlık alanı, sıkıştırılmış görüntüye ait genel bilgilerin bulunduğu kısımdır. Bu alan sırasıyla genişlik bilgisini, yükseklik bilgisini, görüntüde yer alan renk listesini içeren kod kitabının uzunluk bilgisini ve kod kitabının kendisini barındırmaktadır. Burada genişlik, yükseklik ve kod kitabı uzunluğu bilgileri her biri 32 bit tamsayı olarak saklanmakta olup toplam 12 byte bellek alanı kullanmaktadırlar.

Kod kitabı uzunluk bilgisi, devamında yer alan kod kitabının uzunluğunu belirtmektedir. Başka bir deyişle kod kitabında kaç adet renk olduğu bilgisidir. Bu sayede kod çözücü, kod kitabından renk listesini hatasız bir şekilde ayrıştırabilir.

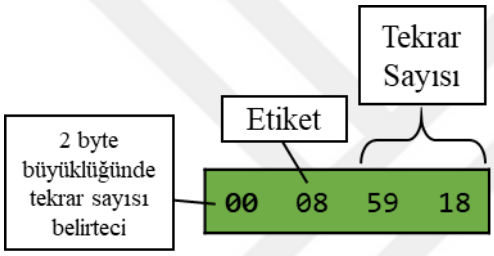
Kod kitabındaki her renk, 32 bit uzunluğunda bir tamsayı olarak, karşılık gelen sayısal etiket numarasına göre sıralı bir şekilde saklanmaktadır. Örneğin, kod kitabındaki ilk sırada bulunan tamsayı, 0 etiketine ait renk değeri iken, sonraki tamsayı ise 1 etiketine ait renk değeridir. Renklerin etiket numarasına göre sıralı olarak saklanması sayesinde etiket-renk eşleştirmelerinin ayrıca belirtilmesine gerek kalmamaktadır.

Sıkıştırılmış veri alanında ise, görüntüye ait *E(x,y)* etiket haritasının RLE algoritması kullanılarak etiket-tekrar ikilileri şeklinde sıkıştırılmış hali yer almaktadır. Etiket-tekrar ikililerinin depolanmasında tekrar sayısının büyüklüğü depolama biçimini etkilemektedir. Bir etiket-tekrar ikilisi için, eğer tekrar sayısı 255 veya daha küçükse, başka bir ifade ile 1 byte'ı aşmayacak büyüklükte ise ilk byte etiket, sonraki byte ise tekrar sayısı olmak üzere, Şekil 3.5'te olduğu gibi toplam 2 byte ile bellekte temsil edilebilmektedir.



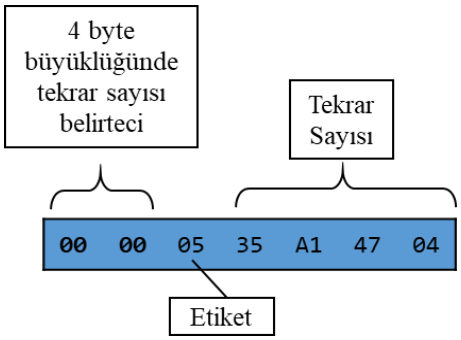
Şekil 3.5. 1 byte büyüklüğünde tekrar içeren etiket-tekrar ikilisi

Tekrar sayısının 2 byte büyüklüğünde olması durumunda, etiket-tekrar ikilisinin önüne bir byte uzunluğa sahip 0 (sıfır) değerinde bir belirteç eklenmektedir. Belirteç sayesinde, kod çözücü sıfır değeri ile karşılaştığında tekrar sayısını elde etmek için 2 byte'lık değer okuyacaktır. Etiket-tekrar ikilisinin bellekte işgal ettiği toplam alan ise Şekil 3.6'da gösterildiği gibi 4 byte olmaktadır.



Şekil 3.6. 2 byte büyüklüğünde tekrar içeren etiket-tekrar ikilisi

Tekrar sayısının 3 byte veya 4 byte büyüklüğünde olması durumunda ise, etiket-tekrar ikilisinin önüne iki byte uzunluğunda 0 (sıfır) değeri ile eklenmektedir. Böylece kod çözücü, ardışık iki byte'lık sıfır değeri ile karşılaştığında tekrar sayısını elde etmek için 4 byte okuyacaktır. Etiket-tekrar ikilisinin bellekte işgal ettiği toplam alan ise Şekil 3.7'de gösterildiği gibi 7 byte olmaktadır.



Şekil 3.7. 4 byte büyüklüğünde tekrar içeren etiket-tekrar ikilisi

Sıkıştırılmış veri alanında belirteç amaçlı kullanılmakta olan 0 değerinin, 0 numaralı etiket ile karışıklık oluşturmaması adına, etiket değerleri depolanırken 1 artırılarak depolanmaktadır.

3.3. Kod Çözücü (Decoder)

Kod çözücü (decoder) modülün görevi kaydedilen sıkıştırılmış dosyayı tekrar çözerek *I'(x, y)* çıkış görüntüsünü meydana getirmektedir. Kod çözücü modülün ilk aşamasında kaydedilen dosya, bir byte dizisi olarak bellekten okunmakta ve başlık alanı ayrıştırılarak görüntüye ait genişlik-yükseklik bilgisi ve kod kitabı uzunluğu elde edilmektedir.

İkinci aşamada öncelikle kod kitabı uzunluğu ile aynı uzunlukta bir renk dizisi oluşturulmaktadır. Kod kitabındaki her renk değeri 32 bit tamsayı olarak temsil edildiği için kod çözücü, kod kitabı uzunluğu adedince 32 bit tamsayı okumakta ve bunları renk nesnesine dönüştürerek oluşturulan renk dizisine sırasıyla eklemektedir.

Üçüncü fazda ise RLE çözücü algoritma yardımıyla sıkıştırılmış veri alanı ayrıştırılarak iki boyutlu *E(x,y)* etiket haritası tekrar elde edilmektedir. Burada ilave olarak, etiket değerleri depolanırken 1 artırılarak depolandığı için çözme aşamasında 1 eksiltilerek orijinal değerine getirilmektedir. Ayrıca, genişlik yükseklik bilgisinin işleme dâhil edilmesi de bu süreçte olmaktadır. Kod çözücü, genişlik miktarından daha uzun bir tekrar değeri ile karşılaştığında $T > N$ olmak koşuluyla,

$$S = \frac{T}{N} \tag{3.1}$$

$$A = T \bmod N \tag{3.2}$$

ifadesine göre çözme işlemi gerçekleştirmektedir. Eş. 3.1 ve Eş. 3.2'de yer alan *T* değişkeni mevcut etiketin tekrar sayısını, *N* ise görüntüye ait genişlik bilgisini göstermektedir. Bununla beraber, *S* değeri etiketin *E(x, y)* içerisinde kaç adet tam satırlık alana açılacağını belirtir iken, *A* ise tam satırlar dolduktan sonra arta kalan tekrar miktarını temsil etmektedir.

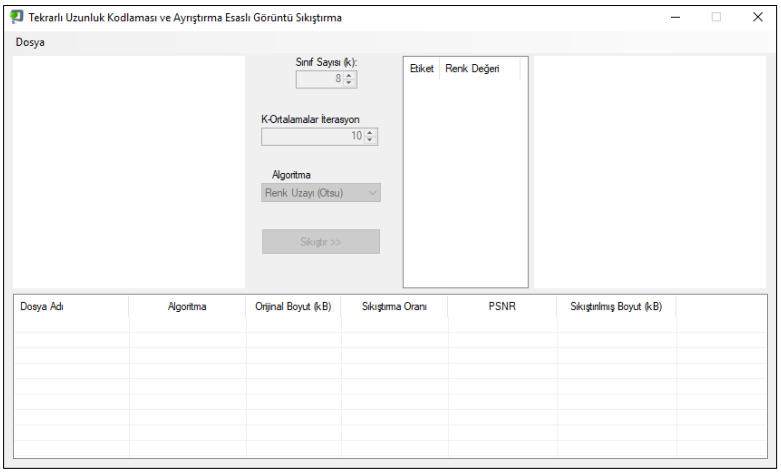
Son durumda ise, etiket bilgileri kod kitabından okunan renk değerleri ile eşleştirilerek *I'(x,y)* çıkış görüntüsü üretilmekte ve genişlik-yükseklik bilgisi de doğrulandıktan sonra Bitmap biçiminde bir çıktı olarak döndürülmektedir. Şekil 2.4'teki Ev görüntüsünün önerilen algoritma ile sıkıştırıldığında elde edilen dosya içeriği Ek-1 ve Ek-2'de gösterilmiştir.

4. DENEYSEL SONUÇLAR VE YORUMLAR

Test amaçlı seçilen orijinal görüntüler 24 bit'lik BMP formatındadır. Test edilen renkli görüntüler öncelikle Otsu algoritması yardımıyla her bir kanal için elde edilen eşikler ile ayrıştırılmıştır. Akabinde sıkıştırma yapılarak dosya oluşturulmuştur. Elde edilen dosya boyutları ve ortalama PSNR değerleri hesaplanmıştır. Ayrıca BMP formatındaki orijinal görüntüler JPEG ve GIF biçimleriyle kaydedilerek karşılaştırma yapılmıştır. Test ortamı C# üzerinde geliştirilmiştir. Kayıt yapılırken .NET Framework içerisinde gelen Image, ImageFormat ve Encoder sınıfına ait metotlar kullanılmıştır ve JPEG için 100 kalite faktörü seçilmiştir. GIF formatı için ise 8 bit renk derinliği bulunmaktadır. Önerilen yöntem kayıplı sıkıştırma yaptığı için kayıpsız sıkıştırma yöntemleri testlere dahil edilmemiştir. JPEG biçimi yaygın kullanımı sebebiyle, GIF biçimi ise renk indirgeme tabanlı bir yöntem olması bir sebebiyle karşılaştırmada tercih edilmiştir. Bunlara ilave olarak önerilen metodun farklı sınıflandırıcılar ile performansını görmek için K-ortalamalar algoritması da testlere dâhil edilmiştir. K-ortalamalar algoritmasında iterasyon sayısı 20 olarak sabit tutulmuştur.

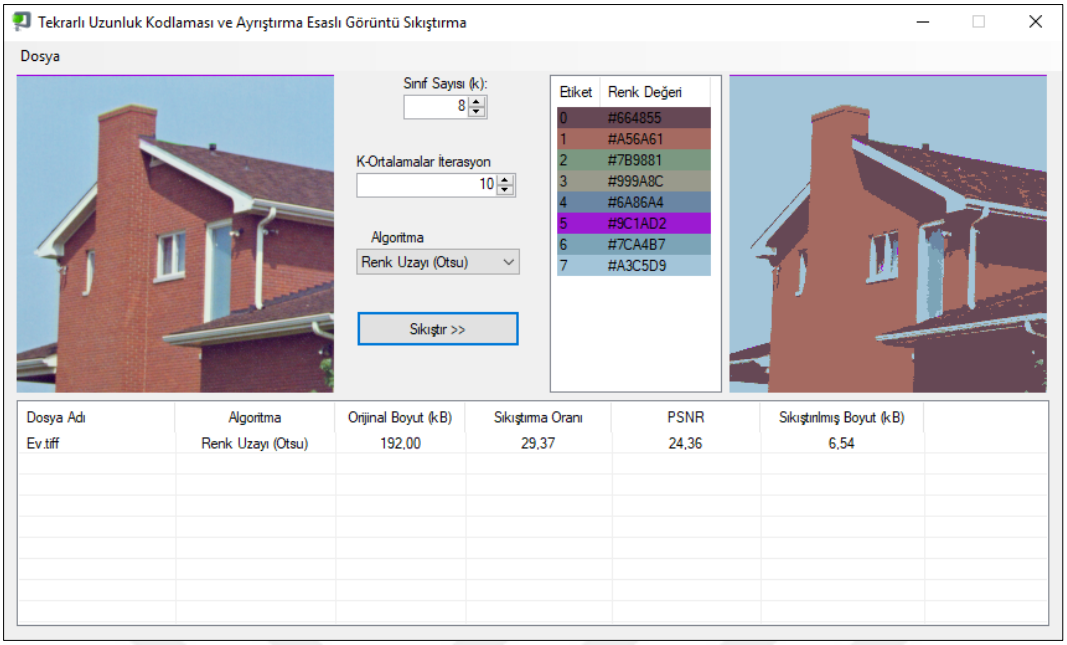
4.1. Geliştirilen Arayüz

Geliştirilen algoritma için C# (.NET Framework 4.5) ortamında bir arayüz yazılımı tasarlanmıştır. Geliştirilen arayüzün giriş ekranı görüntüsü Şekil 4.1(a)'da verilmiştir.

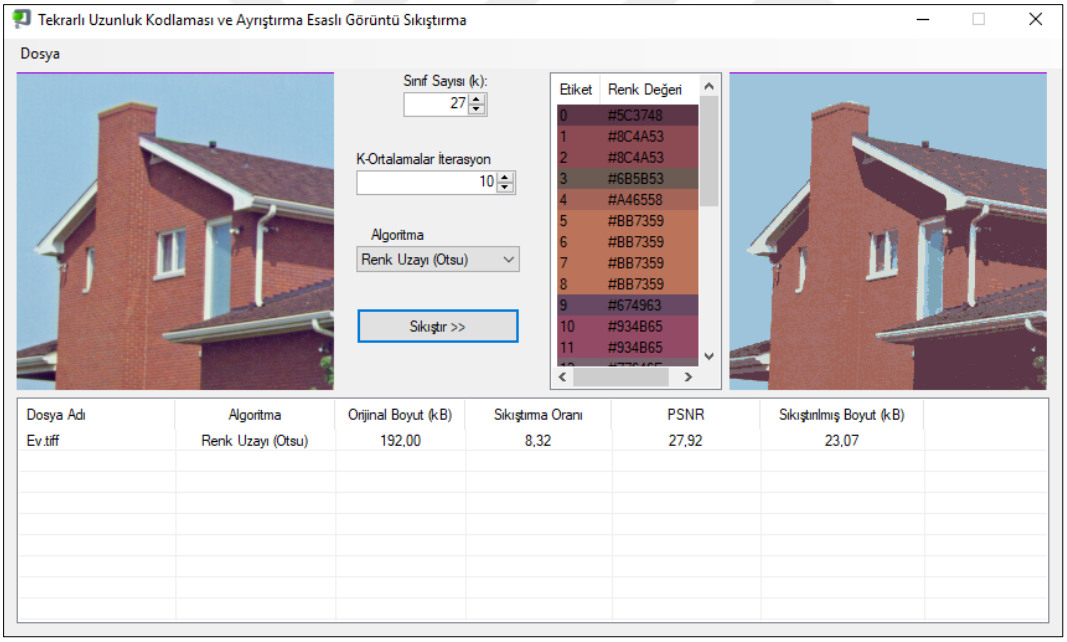


a)

Şekil 4.1. Geliştirilen arayüz a) Giriş b) Renk uzayı yöntemi ile *k=8* (*m=1*) durumundaki sonuç c) Renk uzayı yöntemi ile *k=27 (m=2)* durumundaki sonuç d) K-ortalamalar yöntemi ile *k=8* ve 20 iterasyon durumundaki sonuç e) Sonuç dosya içeriğinin onaltılık biçimde görüntülenmesi

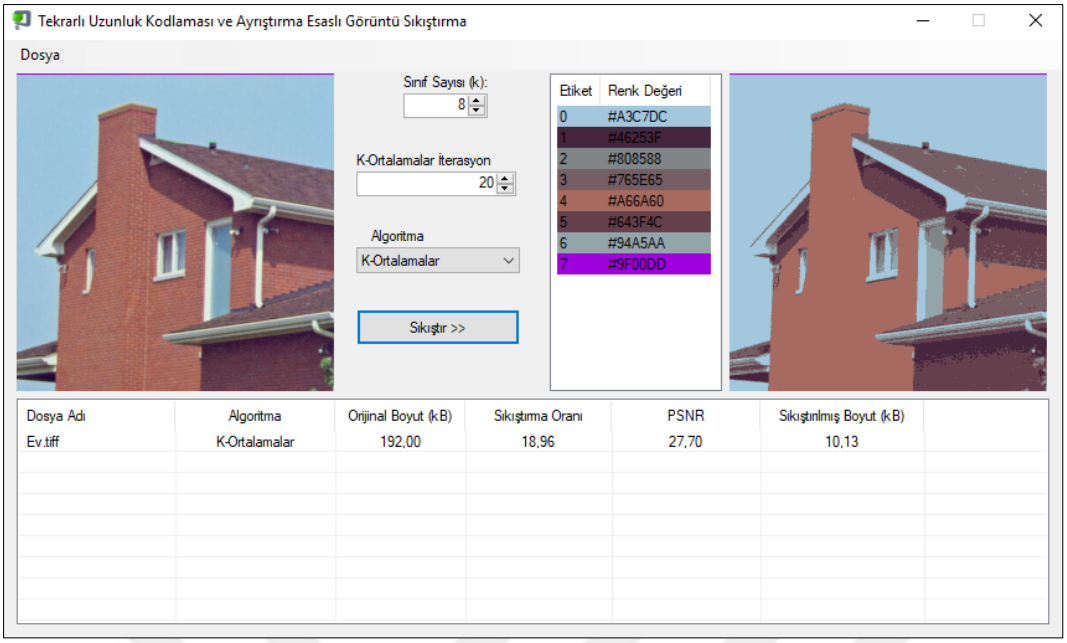


b)

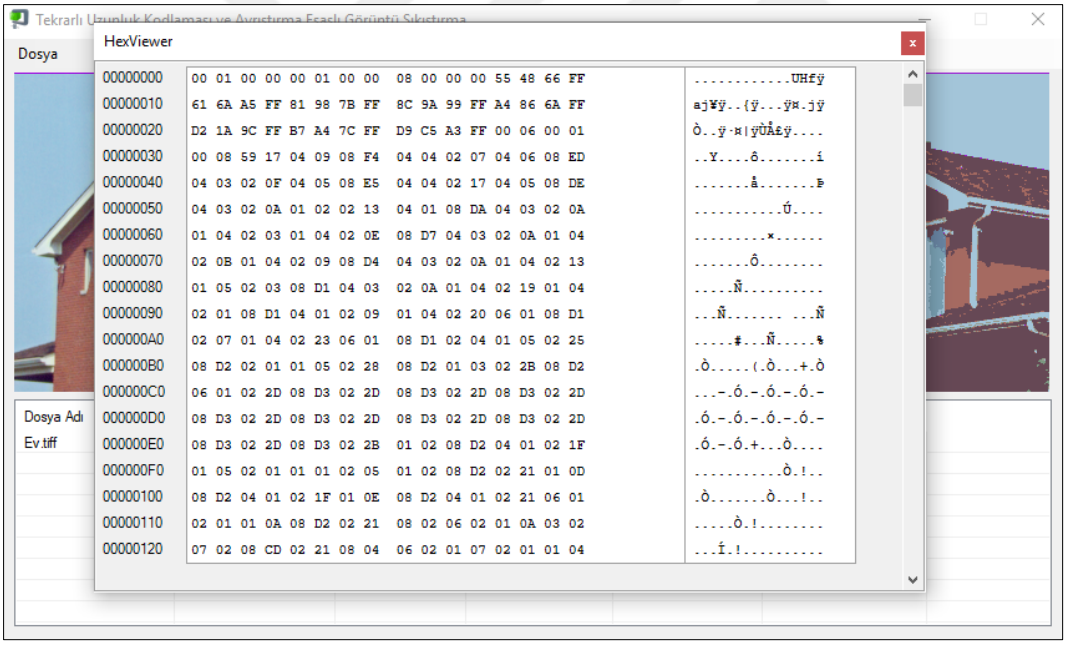


c)

Şekil 4.1. (devam) Geliştirilen arayüz a) Giriş b) Renk uzayı yöntemi ile *k=8* (*m=1*) durumundaki sonuç c) Renk uzayı yöntemi ile *k=27 (m=2)* durumundaki sonuç d) K-ortalamalar yöntemi ile *k=8* ve 20 iterasyon durumundaki sonuç e) Sonuç dosya içeriğinin onaltılık biçimde görüntülenmesi



d)



e)

Şekil 4.1. (devam) Geliştirilen arayüz a) Giriş b) Renk uzayı yöntemi ile *k=8* (*m=1*) durumundaki sonuç c) Renk uzayı yöntemi ile *k=27 (m=2)* durumundaki sonuç d) K-ortalamalar yöntemi ile *k=8* ve 20 iterasyon durumundaki sonuç e) Sonuç dosya içeriğinin onaltılık biçimde görüntülenmesi

Uygulama başlatıldığında ilk olarak “Dosya > Aç” menüsü izlenerek sıkıştırılmak istenen görüntü seçilir. Seçilen orijinal görüntü formun sol tarafındaki resim alanına aktarılmaktadır. Sonrasında ise “Algoritma” açılır menüsünden sıkıştırmada kullanılmak istenen ayrıştırma algoritması ve “Sınıf Sayısı” alanından ise ayrıştırılmak istenen sınıf adedi seçilebilmektedir. Sınıf Sayısı için seçilebilecek *k* değerleri 8 ve 27 ile sınırlandırılmıştır. “Renk Uzayı (Otsu)” algoritması tercih edildiğinde eğer *k=8* seçilmişse tek eşikli ayrıştırma, *k=27* seçilmişse iki eşikli ayrıştırma yapılmaktadır. “İterasyon Sayısı” alanı K-Ortalamalar algoritması seçildiği takdirde dikkate alınmaktadır. Buradan K-Ortalamalar algoritması için istenen maksimum iterasyon sayısı ayarlanabilmektedir.

Parametreler seçildikten sonra ise “Sıkıştır” butonuna tıklanarak sıkıştırma işlemi başlatılır. İşlem tamamlandığında sıkıştırılmış görüntü Şekil 4.1(b), Şekil 4.1(c) ve Şekil 4.1(d)'de olduğu gibi formun sağ tarafındaki resim alanında görüntülenir. Elde edilen sonuçlar ise formun alt kısmındaki tabloda listelenir ve istendiği takdirde CSV formatında kaydedilebilir. Sonuçları kaydetmek için liste alanına sağ tıklanarak “Tümünü Kaydet” seçilir.

Formun sağ tarafındaki sonuç görüntüsüne çift tıklanarak Şekil 4.1(e)'de olduğu gibi kaydedilecek dosya içeriğinin onaltılık biçimdeki önizlemesi açılabilir. Sıkıştırılan dosya kaydedilmek istenirse “Dosya > Kaydet” yolu izlenir.

4.2. Deneysel Sonuçlar ve Yorumlar

Önerilen yöntem, geliştirilen arayüz kullanılarak Intel Core i7 2630-QM 2,00 GHz işlemci ve 8GB RAM donanımına sahip bir bilgisayarda, SIPI ve Wisconsin-Madison Üniversitesi resim veritabanlarından alınan 11 adet resim üzerinde test edilmiştir [47, 48].

Geliştirilen yaklaşım ilk olarak Şekil 4.2(a)'da orijinali verilen Ağaç görüntüsü üzerinde test edilmiş ve teste ait sonuçlar Çizelge 4.1'de verilmiştir. Çıkış görüntüsü Şekil 4.2(b)'de verilen JPEG sıkıştırmasının en yüksek benzerlik değerini elde ettiği fakat sıkıştırma oranı bazında diğer yöntemlere göre geride kaldığı görülmektedir. Şekil 4.2(c)'de çıkış görüntüsü verilen GIF sıkıştırması ise düşük PSNR değerine rağmen yeterli sıkıştırma oranına ulaşamamıştır. Önerilen yöntemin tek eşikli ayrıştırılmasındaki çıkış görüntüsü ise Şekil 4.2(d)'de verilmiş olup, en yüksek sıkıştırma seviyesini elde etmiştir. Fakat bu durum yüksek bilgi kaybını beraberinde getirmiş ve dolayısıyla PSNR açısından yeterli başarıma

ulaşılamamıştır. Şekil 4.2(e)'de ise geliştirilen yaklaşımın sınıflandırıcısı, K-ortalamalar algoritması ile değiştirilmiş ve renk uzayı yöntemindeki tek eşikli ayrıştırma şartlarına denk olması adına *k=8* seçilerek test edilmiştir. Benzer durum görüntüsü Şekil 4.2(f)'de verilmiş olan iki eşikli ayrıştırma ve görüntüsü Şekil 4.2(g)'deki gibi olan K-ortalamalar algoritmasının 27 sınıflı ayrıştırması için de geçerlidir. K-ortalamalar algoritması PSNR bakımından başarılı olsa da sıkıştırma oranı noktasında önerilen yöntemin gerisinde kalmıştır. Ağaç görüntüsü için K-ortalamalar yöntemi PSNR açısından daha iyi başarım gösterirken, yeterli sıkıştırma oranına ulaşamamıştır. Ayrıca K-ortalamalar metodunda sınıflandırma için başlangıç şartları rastgele oluşturulduğundan algoritma her seferinde farklı sonuçlar üretmekte ve yöntemin güvenilirliğini düşürmektedir. Şekil 4.2(h)'de ise sonuçların değişimleri grafik halinde sunulmuştur.

Şekil 4.3(a)'da Babun görüntüsünün orijinali, Çizelge 4.2'de deney sonuçları verilmiştir. Şekil 4.3(b)'deki JPEG sonucu yüksek PSNR değerine sahipken sıkıştırma oranı en düşük seviyede olmuştur. Şekil 4.3(c)'deki GIF sıkıştırmasının ise hem sıkıştırma oranında hem de benzerlik açısından Babun görüntüsü için iyi bir başarım gösterdiği söylenebilir. Önerilen algoritma tek eşikli durumda Şekil 4.3(d)'de verilen çıkış görüntüsünü üretmiştir. Sıkıştırma oranı 5,36 seviyesinde kalmış olsa da diğer yöntemlere nazaran daha yüksek bir seviyededir. Fakat PSNR tarafında yeterli başarıma ulaşamamıştır. Babun görüntüsü renklerin yeterince homojen olarak dağılmadığı karmaşık bir görüntü örneğidir. Bu sebeple hem ayrıştırma yöntemi hem de RLE sıkıştırması açısından dezavantajlı bir yapıdadır. Dolayısıyla en düşük sıkıştırma oranları Babun görüntüsünde elde edilmiştir. K-ortalamalar metodunun 8 sınıflı durumunda Şekil 4.3(e)'deki görüntü elde edilmiştir. Burada GIF sıkıştırmasına yakın sonuçlar elde edilmiş olsa da yeterli sıkıştırma oranına ulaşılamamıştır. Şekil 4.3(f)'de ise önerilen iki eşik durumundaki, Şekil 4.3(g)'de ise K-ortalamalar için 27 sınıf durumundaki sonuç görüntüleri verilmiştir. Burada sınıf sayıları arttığı için daha yüksek PSNR değerlerine çıkılmış fakat sıkıştırma oranı açısından yeterli başarım elde edilememiştir. Deney sonuçlarına ait değişim grafiği Şekil 4.3(h)'de verilmiştir.

Orijinali Şekil 2.10(a)'da verilmiş olan Biberler görüntüsünün deneysel sonuçları Çizelge 4.3'te gösterilmiştir. Şekil 4.4(a)'deki JPEG sıkıştırması yüksek PSNR seviyelerine çıkarken yeterli sıkıştırma oranına ulaşamamıştır. GIF sıkıştırma sonucu ise Şekil 4.4(b)'de verilmiştir. Biberler'in Şekil 4.4(c)'de görüntüsü verilen renk uzayı tek eşikli ayrıştırma sonucunun, Şekil 4.4(d)'de çıktısı verilmiş olan K-ortalamalar 8 sınıflı sınıflandırmasına

göre hem sıkıştırma oranı ve hem de bilgi kaybı ölçütü açısından geride kaldığı görülmektedir. Bununla beraber, Şekil 4.4(e)'de görüntüsü verilen iki eşikli ayrıştırma sonucu, GIF sıkıştırmasına göre hem benzerlik ölçütü hem de sıkıştırma oranı noktasında daha iyi başarım göstermiştir. Şekil 4.4(f)'de verilen k=27 durumundaki K-ortalamalar algoritmasında ise GIF sıkıştırmasına yakın sıkıştırma oranı elde edilmiştir. K-ortalamalar algoritmasının Biberler görüntüsünde genel olarak iyi bir başarım yakaladığı söylenebilir. Bunun sebebi olarak özellikle Şekil 4.4(d) incelendiğinde, görüntüde RLE performansını olumlu etkileyecek sıralı ve uzun renk tekrarlarının meydana gelmiş olması, başka bir ifade ile görüntünün renk dağılımında homojenlik derecesinin artmış olması gösterilebilir. Sonuçlara ait grafiksel gösterim ise Şekil 4.4(g)'den incelenebilir.

Şekil 4.5(a)'da Bonibon orijinal görüntüsü, Çizelge 4.4'te ise Bonibon'a ait deneysel sonuçlar yer almaktadır. Sonuçlara göre, Şekil 4.5(b)'deki JPEG sıkıştırması yüksek PSNR elde etmiş olsa da sıkıştırma performansında geride kalmıştır. Şekil 4.5(c)'de verilen GIF sıkıştırması ise hem PSNR hem de sıkıştırma oranında en düşük başarımı elde ettiği görülmektedir. Geliştirilen metodun, çıkış görüntüsü Şekil 4.5(d)'de verilmiş olan tek eşikli durumu için 40 seviyelerinin üzerinde bir oran ile hayli yüksek bir sıkıştırma performansı ortaya koyduğu görülmektedir. Burada görüntüye ait arka planın büyük ölçüde tek bir sınıf olarak ayrıştırılmış olmasının etkisi yüksektir. Bu durum görüntü içerisinde uzun ve sıralı renk tekrarların meydana gelmesini ve RLE sıkıştırmasının iyi performans göstermesini sağlamıştır. Şekil 4.5(e)'de görüntüsü verilmiş 8 sınıflı K-ortalamalar algoritması sonucu ise geliştirilen algoritmanın tek eşikli durumuna benzer şekilde sıkıştırma oranında öne çıkmaktadır. Şekil 4.5(f)'de verilen iki eşikli renk uzayı yöntemi sonucu ve Şekil 4.5(g)'de verilen K-ortalamalar yönteminin 27 sınıflı ayrıştırma sonuçları ise birbirine yakın değerlerdedir. Sonuçların değişimleri ise Şekil 4.5(h)'de sunulmuştur.

a)

b)

c)

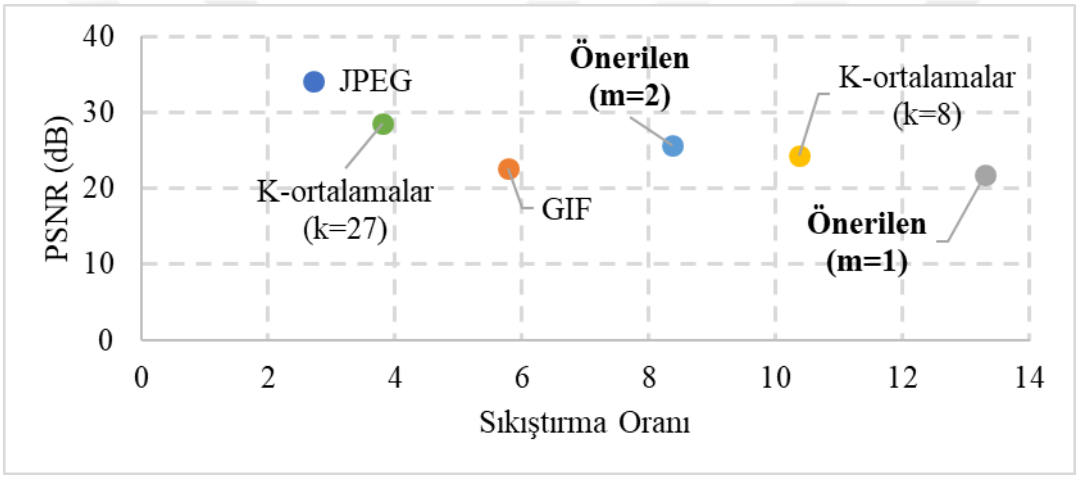
d)

e)

f)

Şekil 4.2. Ağaç a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

g)

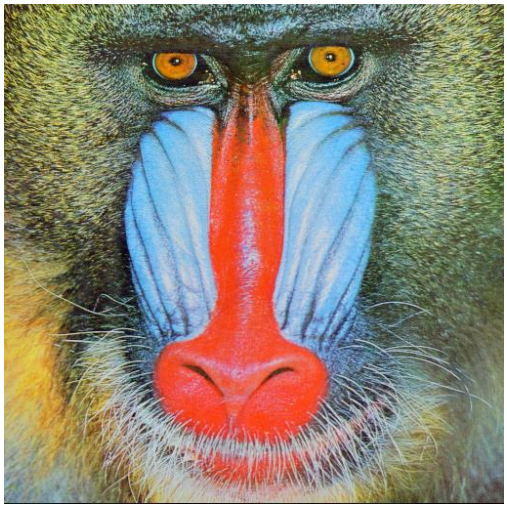


h)

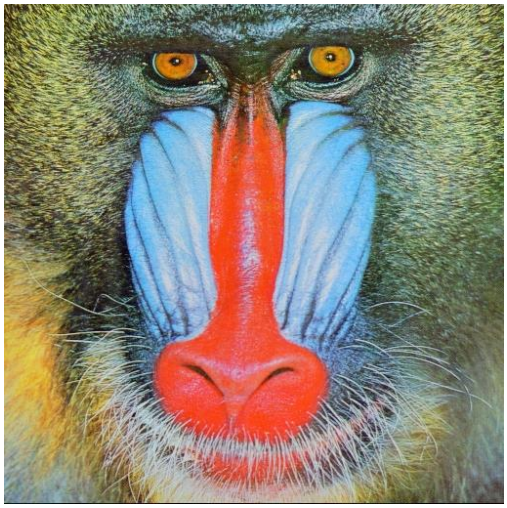
Şekil 4.2. (devam) Ağaç a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.1. Performans ölçütleri: Ağaç

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Ağaç, 256x256 (192,00 kB)	JPEG	2,71	34,21	70,79
	GIF	5,79	22,62	33,17
	Önerilen (m=1)	13,30	21,90	14,43
	K-ortalamalar (k=8)	10,37	24,30	18,51
	Önerilen (m=2)	8,37	25,67	22,93
	K-ortalamalar (k=27)	3,80	28,51	50,58



a)



b)

c)

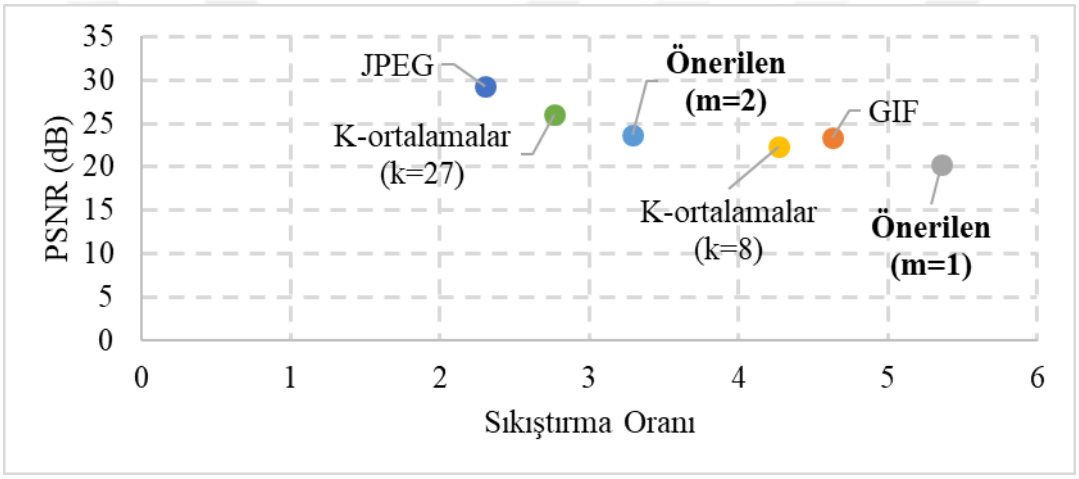
d)

e)

f)

Şekil 4.3. Babun a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

g)



h)

Şekil 4.3. (devam) Babun a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.2. Performans ölçütleri: Babun

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Babun, 512x512 (768,00 kB)	JPEG	2,30	29,29	333,64
	GIF	4,63	23,33	165,88
	Önerilen (m=1)	5,36	20,23	143,41
	K-ortalamalar (k=8)	4,27	22,34	180,02
	Önerilen (m=2)	3,29	23,70	233,34
	K-ortalamalar (k=27)	2,76	26,05	278,11

a)

b)

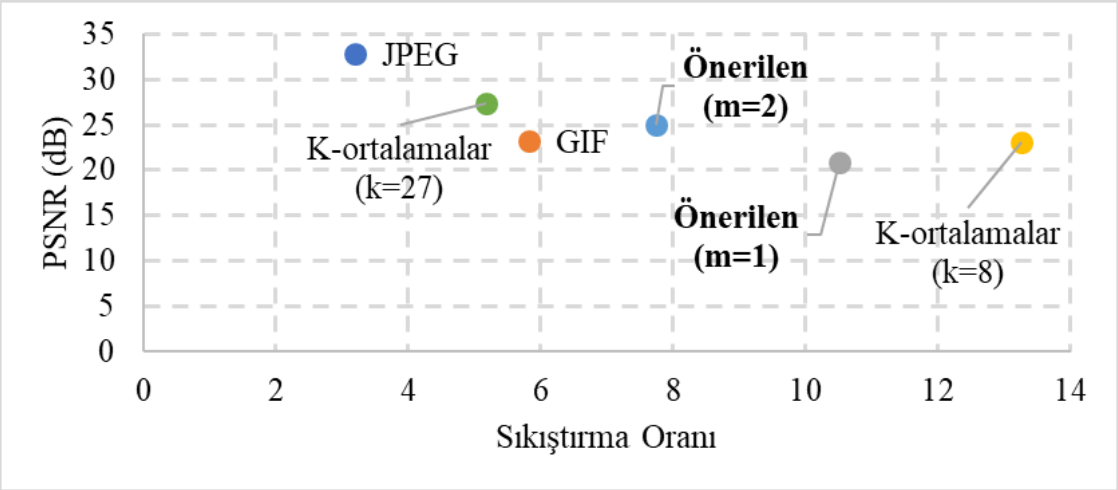
c)

d)

e)

f)

Şekil 4.4. Biberler a) JPEG b) GIF c) Önerilen (m=1) d) K-ortalamalar (k=8) e) Önerilen (m=2) f) K-ortalamalar (k=27) g) Sıkıştırma oranı – PSNR değişimi

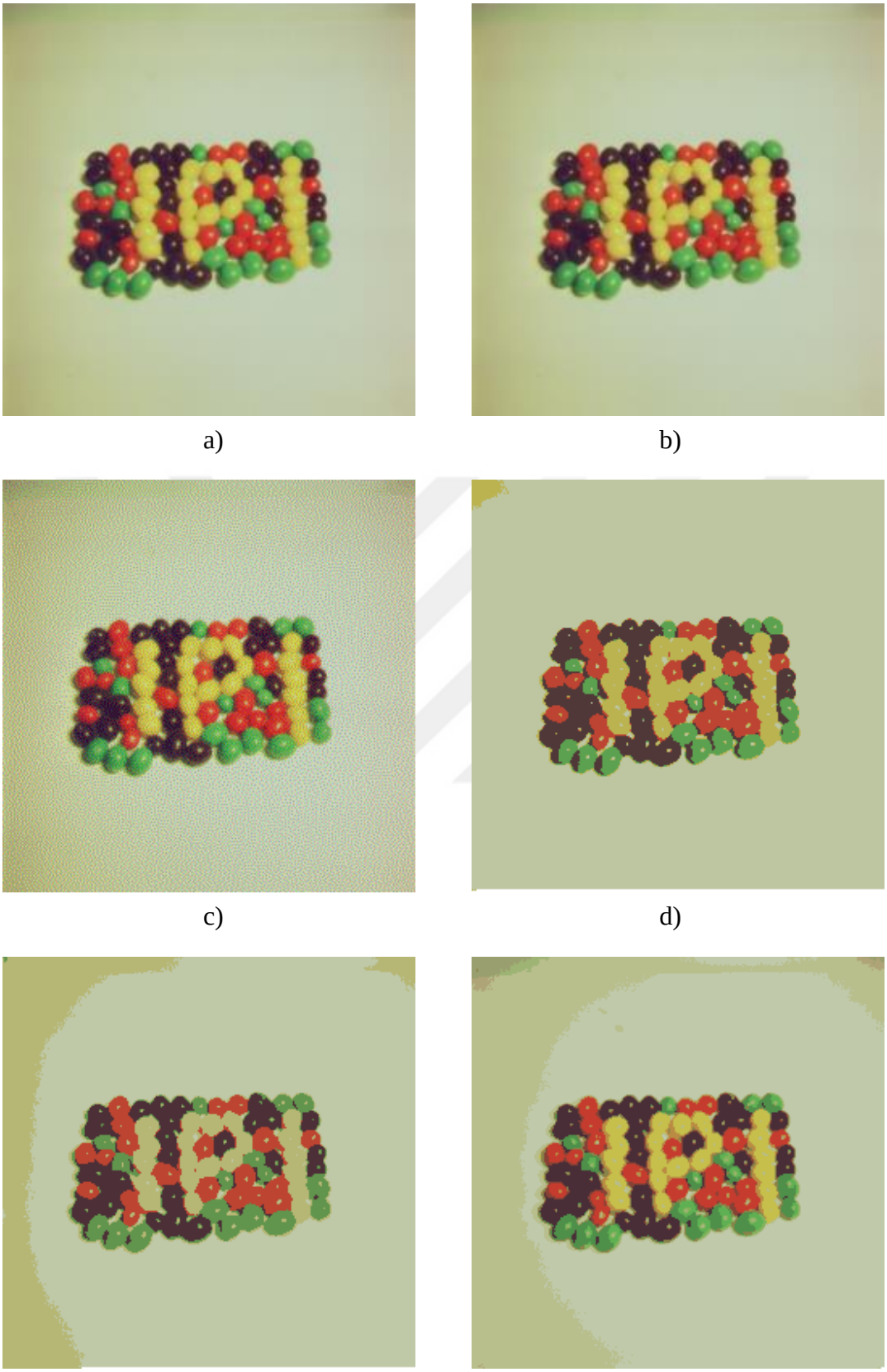


g)

Şekil 4.4. (devam) Biberler a) JPEG b) GIF c) Önerilen (m=1) d) K-ortalamalar (k=8) e) Önerilen (m=2) f) K-ortalamalar (k=27) g) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.3. Performans ölçütleri: Biberler

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Biberler, 512x512 (768,00 kB)	JPEG	3,19	32,80	240,55
	GIF	5,82	23,16	131,87
	Önerilen (m=1)	10,50	20,85	73,15
	K-ortalamalar (k=8)	13,26	23,08	57,91
	Önerilen (m=2)	7,75	24,96	99,10
	K-ortalamalar (k=27)	5,17	27,40	148,60



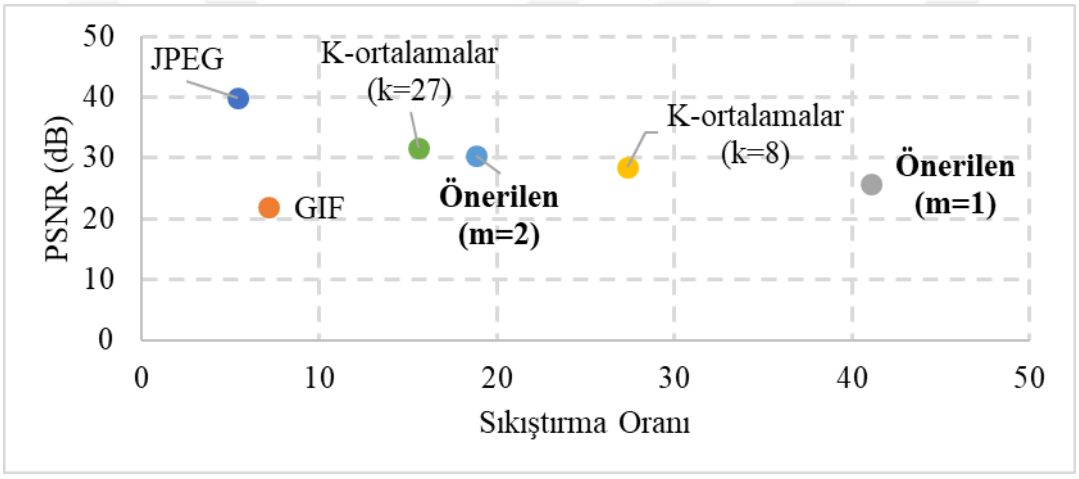
a) b)

c) d)

e) f)

Şekil 4.5. Bonibon a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

g)



h)

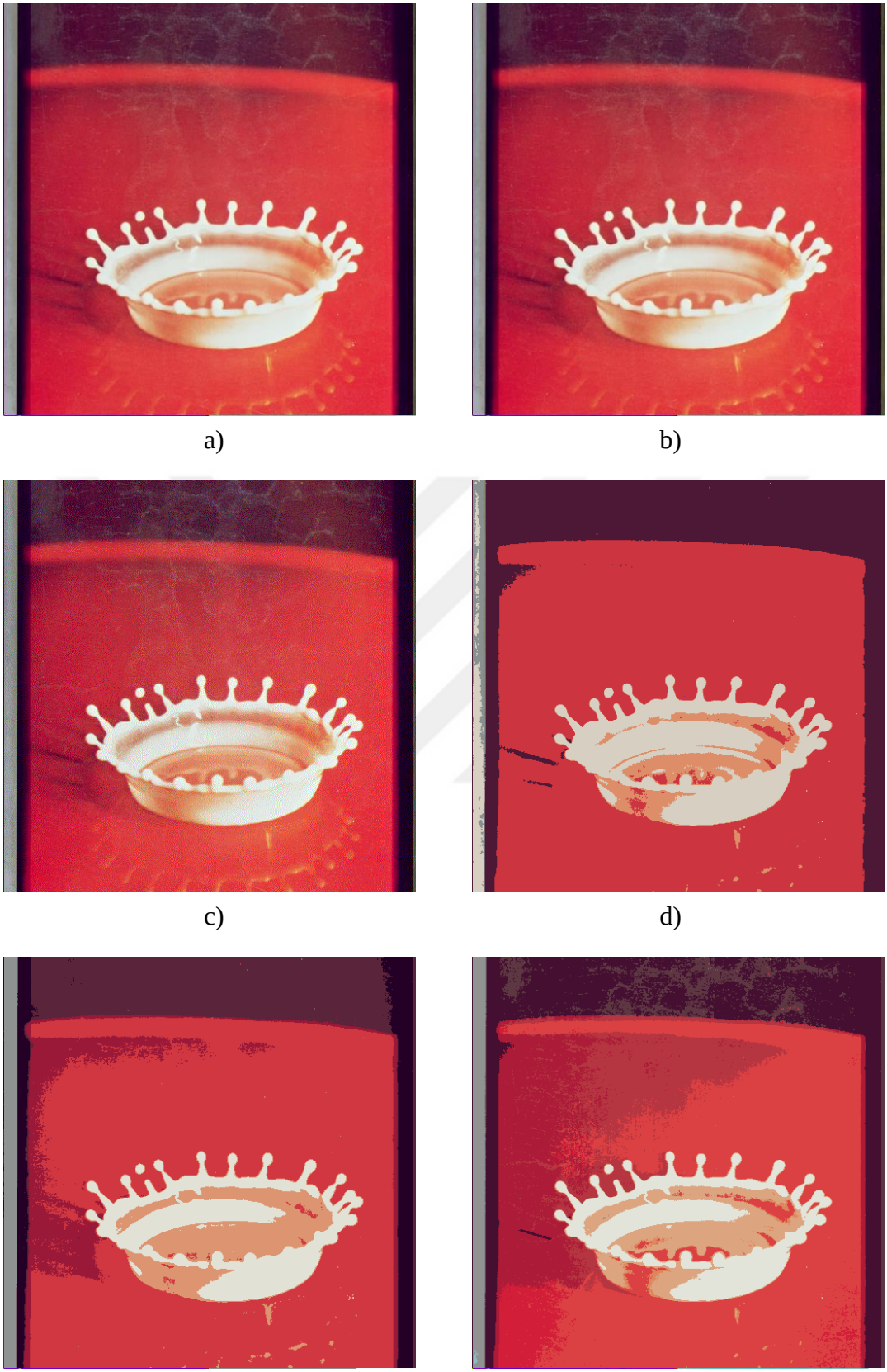
Şekil 4.5. (devam) Bonibon a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.4. Performans ölçütleri: Bonibon

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Bonibon, 256x256 (192,00 kB)	JPEG	5,39	39,79	35,59
	GIF	7,14	21,94	26,90
	Önerilen (m=1)	41,05	25,76	4,68
	K-ortalamalar (k=8)	27,33	28,40	7,03
	Önerilen (m=2)	18,81	30,39	10,21
	K-ortalamalar (k=27)	15,56	31,64	12,34

Orijinali Şekil 4.6(a)'da verilen Damla görüntüsünün Çizelge 4.5'te verilen test sonuçlarına bakıldığında, Şekil 4.6(b)'deki JPEG sıkıştırmasının yüksek PSNR'ın yanında düşük sıkıştırma oranı elde ettiği, Şekil 4.6(c)'deki GIF sıkıştırmasının ise hem düşük PSNR hem de düşük sıkıştırma oranı ile sonuçlandığı görülmektedir. Tasarlanan algoritmanın Şekil 4.6(d)'de çıkış görüntüsünün verildiği tek eşikli durumdaki sıkıştırma oranının 50 seviyelerinin üzerine çıkarak yüksek performans göstermiş olduğu açıkça görülmektedir. Bunun sebebi olarak Damla etrafındaki kırmızı alanın algoritma tarafından tek bir renk sınıfı olarak atanması ve sonucunda meydana gelen sıralı renk tekrarlarının RLE algoritmasını avantajlı hale getirmiş olması söylenebilir. Fakat PSNR bazında yeterli seviyeye ulaşılamamıştır. Aynı zamanda sonuç görüntüsü incelendiğinde orijinal görüntüde bulunan dumanların kaybolduğu, dolayısıyla yüksek bir bilgi kaybı oluştuğu görülmektedir. Dumanların kaybolduğu benzer durum ile Şekil 4.6(e)'deki K-ortalamalar yaklaşımının 8 sınıflı ayrıştırmasında da karşılaşılmış olup, buna rağmen K-ortalamalar için yeterli sıkıştırma oranının elde edilemediği görülmektedir. Şekil 4.6(f)'de verilmiş olan önerilen algoritmanın iki eşikli sonucu ile Şekil 4.6(g)'de verilen K-ortalamalar'ın 27 sınıflı ayrıştırması incelendiğinde ise kabul edilebilir benzerlik değerleri elde edilmiş olsa da sıkıştırma oranı açısından yeterli başarıya ulaşılamadığı görülmüştür. Şekil 4.6(h) ise sonuçların dağılımının grafiksel bir gösterimi olarak verilmiştir.

Şekil 2.4(a)'da Ev görüntüsünün orijinali verilmiştir. Görüntüye ait deneysel sonuçlar ise Çizelge 4.6'da verilmiştir. Sonuçlara göre, Şekil 4.6(a)'da verilmiş olan JPEG sıkıştırması yüksek PSNR değerlerini elde etmiş olmasına rağmen sıkıştırma oranında zayıf kalmıştır. Şekil 4.6(b)'deki GIF sıkıştırması ve Şekil 4.6(c)'deki önerilen yöntemin m=1 olduğundaki durumu karşılaştırıldığında, yöntemin yüksek bir sıkıştırma elde etmesinin yanı sıra, PSNR tarafında GIF sıkıştırmasına göre öne geçmiş olduğu görülmektedir. Ev görüntüsünün ayrıştırma sonucunda RLE sıkıştırmasının başarımını artıracak şekilde sıralı renk tekrarları içeren bir yapı elde etmiş olması bu duruma bir sebep olarak gösterilebilir. Şekil 4.6(d)'deki K-ortalamalar algoritmasının k=8 durumundaki sonucu incelendiğinde iyi bir PSNR seviyesine ulaşabildiği fakat yeterli sıkıştırma oranı elde edemediği görülmektedir. Şekil 4.6(e) ve Şekil 4.6(f)'de çıkış görüntüleri verilen K-ortalamalar algoritmasının k=27 durumu ile renk uzayı algoritmasının m=2 durumlarında ise algoritmaların birbirlerine yakın sonuçlar ettikleri görülmektedir. Şekil 4.7(g)'de ise sonuçlara ait değişim grafiği sunulmuştur.

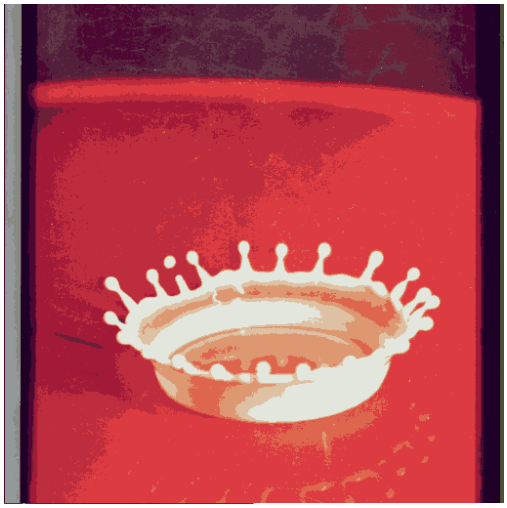


a) b)

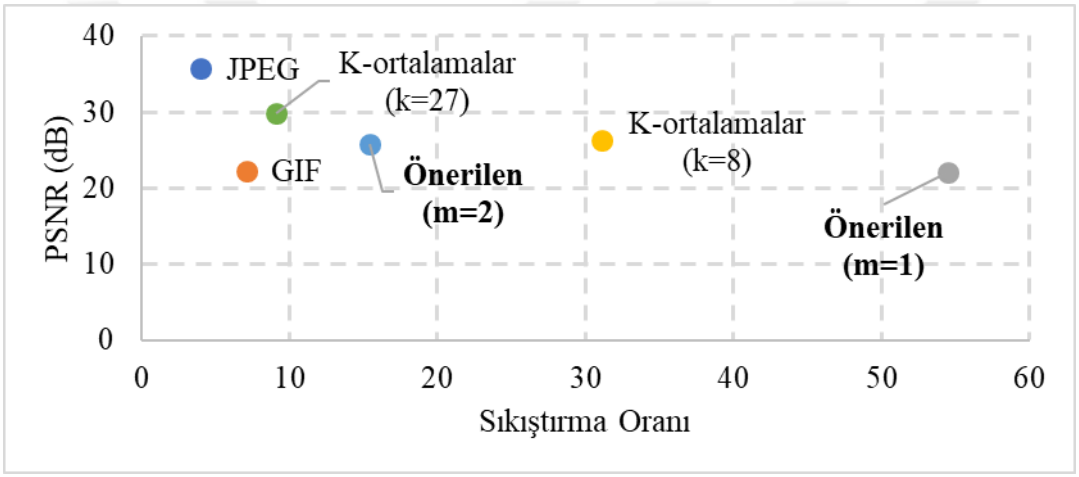
c) d)

e) f)

Şekil 4.6. Damla a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi



g)



h)

Şekil 4.6. (devam) Damla a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.5. Performans ölçütleri: Damla

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Damla, 512x512 (768,00 kB)	JPEG	3,95	35,76	194,22
	GIF	7,06	22,31	108,80
	Önerilen (m=1)	54,45	22,03	14,11
	K-ortalamalar (k=8)	31,12	26,22	24,68
	Önerilen (m=2)	15,38	25,82	49,93
	K-ortalamalar (k=27)	9,06	29,83	84,81

a)

b)

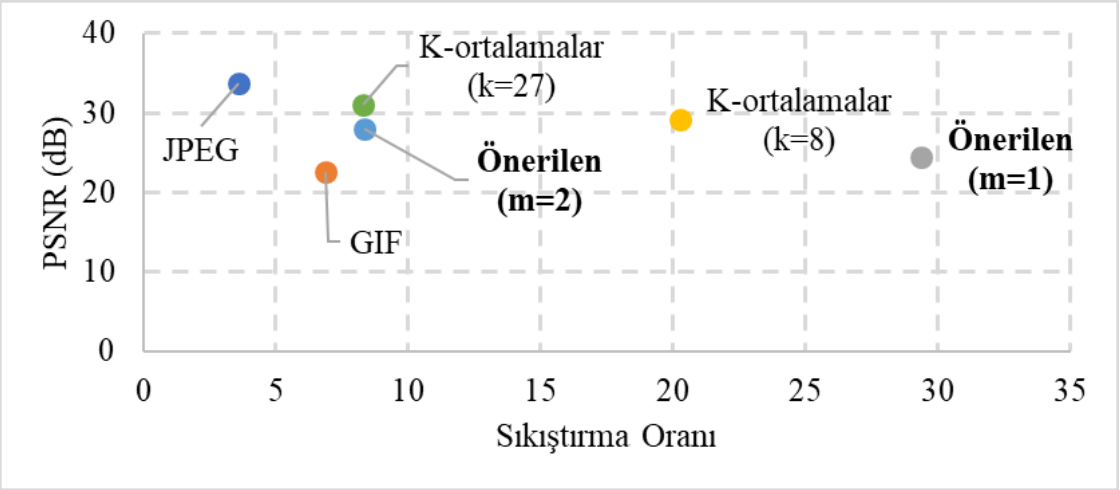
c)

d)

e)

f)

Şekil 4.7. Ev a) JPEG b) GIF c) Önerilen (m=1) d) K-ortalamalar (k=8) e) Önerilen (m=2) f) K-ortalamalar (k=27) g) Sıkıştırma oranı – PSNR değişimi



g)

Şekil 4.7. (devam) Ev a) JPEG b) GIF c) Önerilen (m=1) d) K-ortalamalar (k=8) e) Önerilen (m=2) f) K-ortalamalar (k=27) g) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.6. Performans ölçütleri: Ev

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Ev, 256x256 (192,00 kB)	JPEG	3,61	33,79	53,14
	GIF	6,87	22,60	27,96
	Önerilen (m=1)	29,37	24,36	6,54
	K-ortalamalar (k=8)	20,24	29,14	9,49
	Önerilen (m=2)	8,33	27,92	23,07
	K-ortalamalar (k=27)	8,29	30,97	23,15

Orijinali Şekil 2.8(a)'da verilmiş olan Lena görüntüsüne ait deneysel sonuçlar Çizelge 4.7'de gösterilmektedir. Şekil 4.8(a)'da JPEG biçimi düşük sıkıştırma oranının yanında yüksek PSNR elde etmiştir. Şekil 4.8(b)'deki GIF biçiminin ise PSNR değeri, çıkış görüntüsü Şekil 4.8(c)'de verilen renk uzayı yöntemine ait tek eşikli ayrıştırmasıyla yakın değerde olmuştur. Bununla beraber, tek eşikli sınıflandırmanın, Şekil 4.8(d)'de verilen K-ortalamalar 8 sınıflı durumuna göre hem PSNR hem de sıkıştırma oranı açısından yetersiz kaldığı görülmektedir. Bunun sebebi olarak K-ortalamalar algoritmasının Lena görüntüsü için başarılı bir sınıflandırma yaptığı söylenebilir. Fakat K-ortalamalar yönteminin başlangıç merkezlerini rastgele seçmesi düşünüldüğünde işlemin tekrarlanması durumunda aynı sonucun alınacağı garantisi bulunmamaktadır. Geliştirilen algoritmanın Şekil 4.8(e)'de verilmiş olan iki eşikli durumu, K-ortalamalar algoritmasının Şekil 4.8(f)'deki 27 sınıflı ayrıştırması ve GIF sıkıştırması ise birbirlerine yakın performans göstermiştir. Şekil 4.8(g)'de ise sonuçların değişim grafiği verilmiştir.

Şekil 4.9(a)'da orijinali verilmiş olan Manzara görüntüsünün test sonuçları Çizelge 4.8'de yer almaktadır. Şekil 4.9(b)'de JPEG biçimine ait, Şekil 4.9(c)'de ise GIF biçimine ait çıkış görüntüleri yer almaktadır. GIF sıkıştırması, düşük PSNR ve ortalama bir sıkıştırma oranı, JPEG ise yüksek benzerliğin yanında düşük bir sıkıştırma oranı elde etmiştir. Geliştirilen metodun tek eşikli durumda diğer yöntemlere göre yüksek sıkıştırma oranı ettiği fakat PSNR tarafında zayıf kaldığı görülmüştür. Diğer taraftan, yöntemin Şekil 4.9(d)'de verilen çıkış görüntüsü incelendiğinde bulutların, sandalın ve nehrin arkasında bulunan orman içerisindeki kırmızı bir ağacın kaybolduğu, dolayısıyla yüksek derecede bilgi kaybı meydana geldiği görülmektedir. Çıktı görüntüsü Şekil 4.9(e)'deki gibi olan K-ortalamalar algoritmasının 8 sınıflı durumu ile önerilen algoritmanın Şekil 4.9(f)'deki iki eşikli sınıflandırmasının, farklı sınıf sayılarına rağmen çok yakın sonuçlar elde ettikleri görülmektedir. Bu bağlamda, geliştirilen algoritma, her iki durumda da Manzara görüntüsünde için iyi bir sıkıştırma başarımı göstermiş olsa PSNR tarafında oldukça yetersiz kaldığı söylenebilir. Şekil 4.9(g)'deki K-ortalamalar yönteminin 27 sınıflı durumu, JPEG sıkıştırması ile yakın sonuçlar etmişlerdir. Fakat burada K-ortalamalar algoritmasında kırmızı ağacın kaybolduğu gözlenmektedir. Sonuçlara ait değişim grafiği ise Şekil 4.9(h)'de verilmiştir.

a)

b)

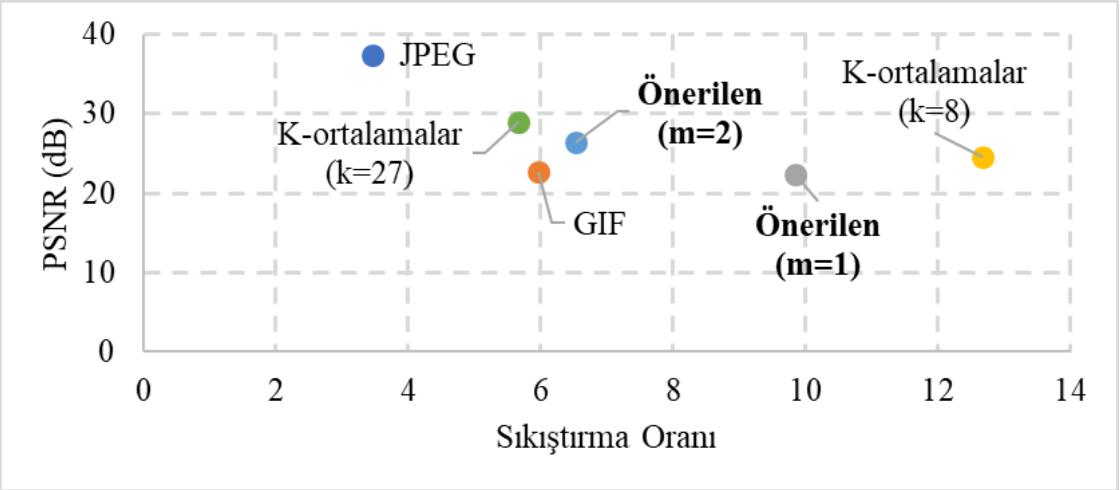
c)

d)

e)

f)

Şekil 4.8. Lena a) JPEG b) GIF c) Önerilen (m=1) d) K-ortalamalar (k=8) e) Önerilen (m=2) f) K-ortalamalar (k=27) g) Sıkıştırma oranı – PSNR değişimi



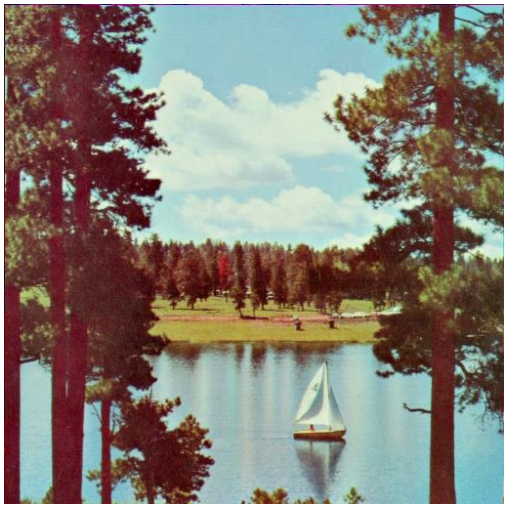
g)

Şekil 4.8. (devam) Lena a) JPEG b) GIF c) Önerilen (m=1) d) K-ortalamalar (k=8) e) Önerilen (m=2) f) K-ortalamalar (k=27) g) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.7. Performans ölçütleri: Lena

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Lena, 512x512 (768,00 kB)	JPEG	3,46	37,48	221,65
	GIF	5,96	22,62	128,76
	Önerilen (m=1)	9,85	22,38	77,98
	K-ortalamalar (k=8)	12,68	24,48	60,55
	Önerilen (m=2)	6,52	26,39	117,73
	K-ortalamalar (k=27)	5,65	29,02	135,97

a)



b)

c)

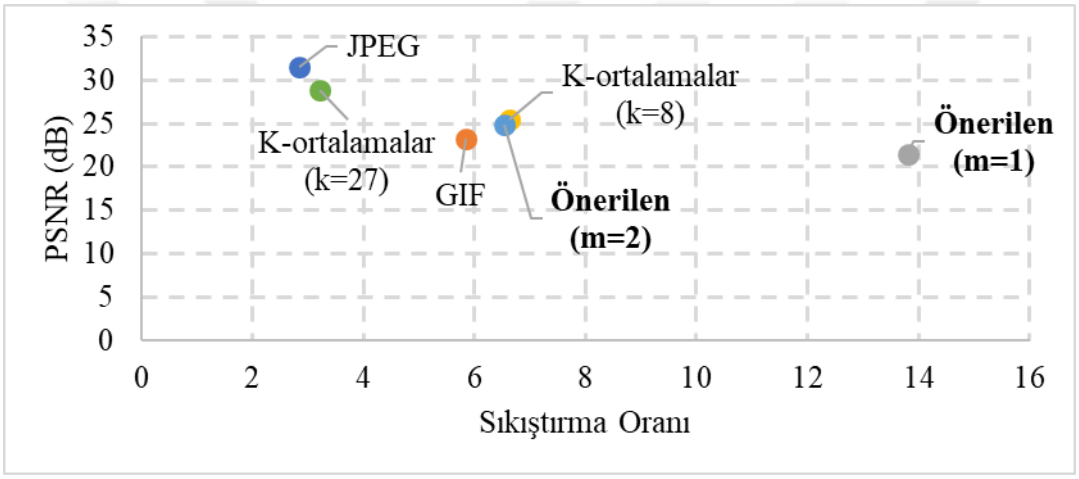
d)

e)

f)

Şekil 4.9. Manzara a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

g)



h)

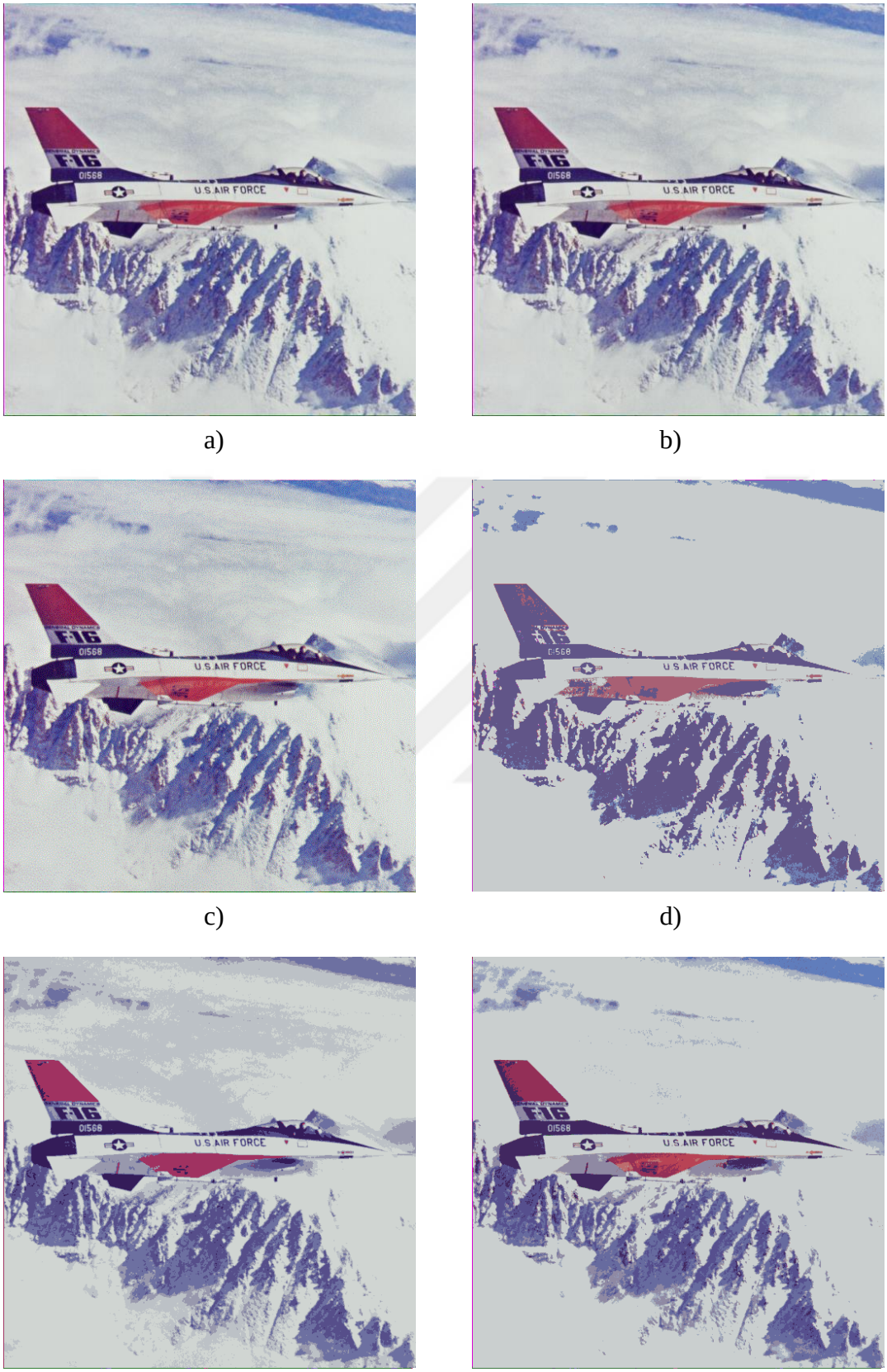
Şekil 4.9. (devam) Manzara a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.8. Performans ölçütleri: Manzara

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Manzara, 512x512 (768,00 kB)	JPEG	2,84	31,48	270,02
	GIF	5,85	23,25	131,35
	Önerilen (m=1)	13,81	21,46	55,63
	K-ortalamalar (k=8)	6,63	25,40	115,88
	Önerilen (m=2)	6,53	24,87	117,63
	K-ortalamalar (k=27)	3,22	28,80	238,80

Şekil 4.10(a)'da orijinali verilmiş Uçak görüntüsünün test sonuçları Çizelge 4.9'da gösterilmiştir. Sonuçlara bakıldığında bir önceki Manzara görüntüsüne genel olarak benzer bir performans dağılımı elde edildiği görülmektedir. Şekil 4.10(b)'de verilen JPEG sıkıştırması, düşük sıkıştırma oranı ve yüksek benzerlik değeri elde etmiştir. Şekil 4.10(c)'de yer alan GIF sıkıştırması ise hem sıkıştırma oranında hem de PSNR tarafında düşük başarım göstermiştir. Çıkış görüntüsünün Şekil 4.10(d)'de verildiği, önerilen yöntemin tek eşik durumundaki sonucu incelendiğinde yüksek sıkıştırma oranının yanında düşük PSNR seviyesi ile karşılaşılmıştır. Bunun yanında görüntüde parlaklık ve kontrast düşümünün yanı sıra arka planda bulunan kar yığınlarında kayıplar oluştuğu gözlenmektedir. Benzer şekilde, uçağın arka kanadındaki F16 yazısının da kaybolduğu görülmektedir. Fakat önerilen yöntemin kar yığınlarını aynı sınıfa dahil etmesi ve bunun sonucunda sıralı renk tekrarlarının oluşması yüksek sıkıştırma oranının elde edilmesini sağlamıştır. K-ortalamalar'ın Şekil 4.10(e)'de verilen 8 sınıflı sonucu ile geliştirilen metodun, görüntüsü Şekil 4.10(f)'de verilen iki eşikli durumundaki sonuçları birbirine yakın olmuştur. Şekil 4.10(g)'deki K-ortalamalar algoritmasının 27 sınıflı durumu, JPEG'e yakın düzeyde PSNR değerine ulaşmıştır. Şekil 4.9(h)'de ise sonuçlara ait grafiksel gösterim yer almaktadır.

Orijinali Şekil 4.11(a)'da verilmiş olan Çocuk resminin Çizelge 4.10'daki sonuçları incelendiğinde yine Manzara görüntüsüne benzer bir dağılım meydana gelmiştir. Şekil 4.11(b)'deki JPEG görüntüsünde ise yüksek PSNR ve düşük sıkıştırma oranı oluşmuştur. Şekil 4.11(c)'deki GIF sıkıştırması ise ortalama değerler elde etmiştir. Çıkış görüntüsü Şekil 4.11(d)'deki gibi olan tek eşikli bölütlemede yüksek sıkıştırma oranının yanında yüksek bilgi kaybı oluşmuştur. Arka plandaki lalelerin dallarının kaybolduğu, görüntüde ise genel olarak parlaklık ve kontrastın belirgin biçimde düşmüş olduğu görülmektedir. Benzer bilgi kayıpları Şekil 4.11(e)'deki K-ortalamalar algoritmasının 8 sınıflı durumunda da meydana gelmiş olsa da bu yöntemde yeterli sıkıştırma oranlarına ulaşılamamış, aynı zamanda Şekil 4.11(f)'deki iki eşikli bölütleme ve GIF sıkıştırması ile yakın sonuçlar elde edilmiştir. Çıkış görüntüsü Şekil 4.11(g)'deki gibi olan K-ortalamalar algoritmasının 27 sınıflı ayrıştırması ise JPEG ve GIF arasında bir performans göstermiştir. Sonuçların değişim grafiği ise Şekil 4.11(h)'de sunulmuştur.



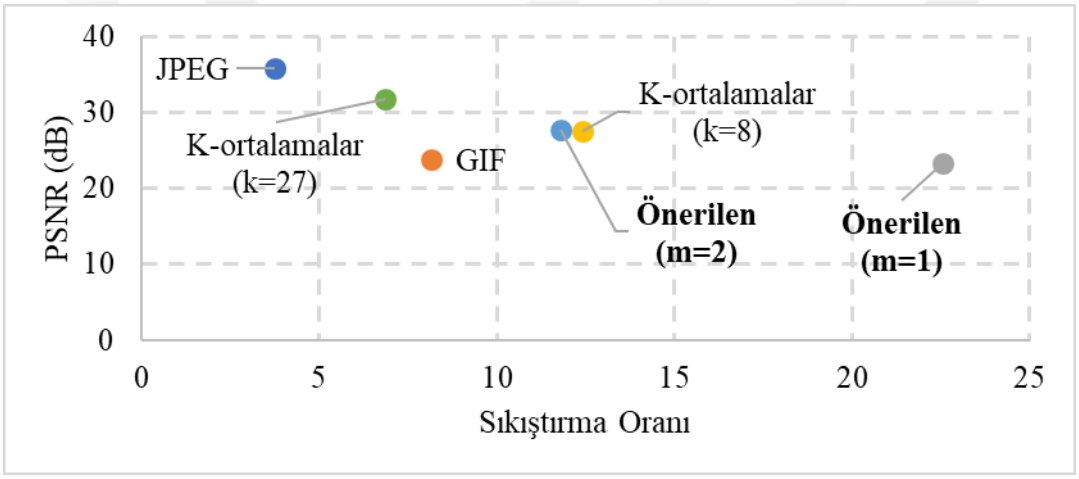
a) b)

c) d)

e) f)

Şekil 4.10. Uçak a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

g)

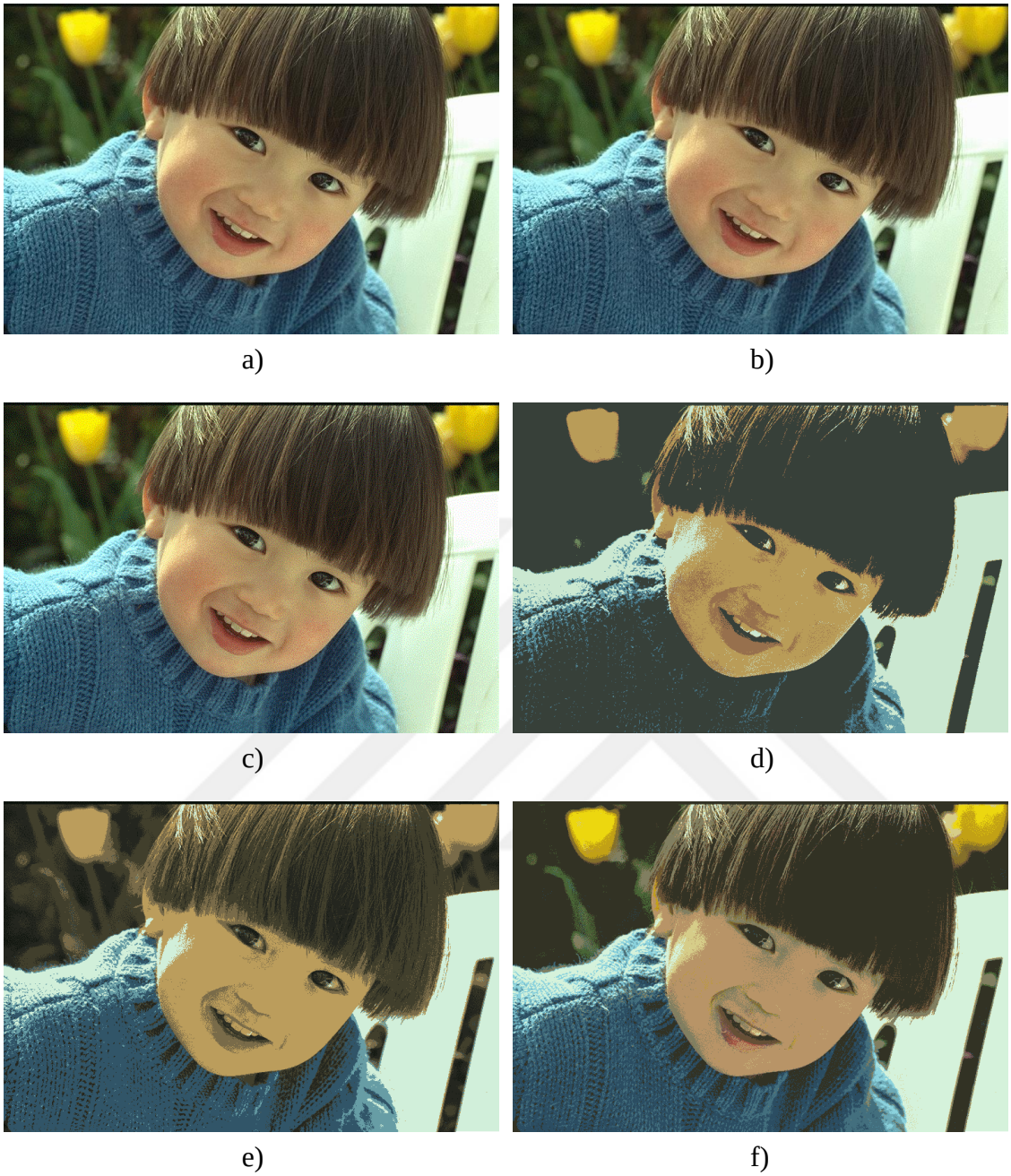


h)

Şekil 4.10. (devam) Uçak a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.9. Performans ölçütleri: Uçak

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Uçak, 512x512 (768,00 kB)	JPEG	3,74	35,83	205,32
	GIF	8,14	23,84	94,37
	Önerilen (m=1)	22,55	23,27	34,05
	K-ortalamalar (k=8)	12,42	27,59	61,84
	Önerilen (m=2)	11,79	27,70	65,14
	K-ortalamalar (k=27)	6,85	31,84	112,14



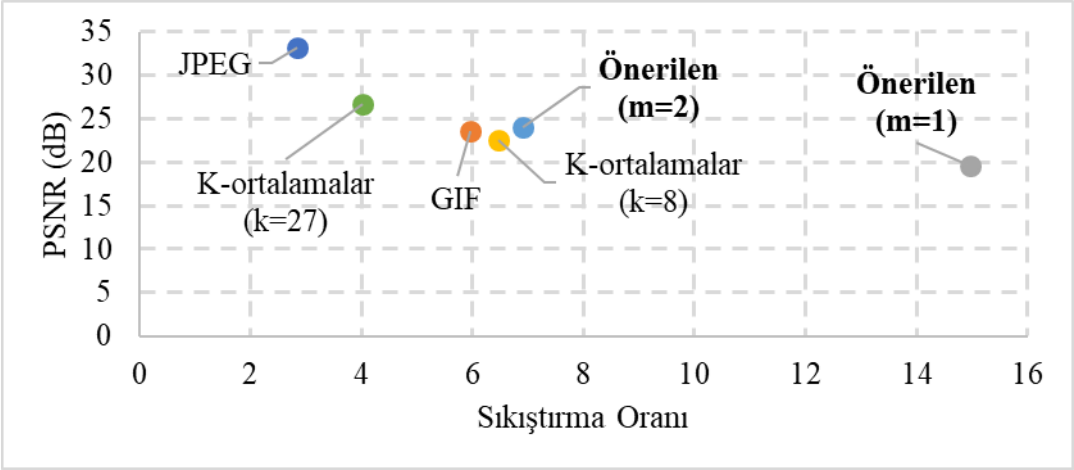
a) b)

c) d)

e) f)

Şekil 4.11. Çocuk a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

g)



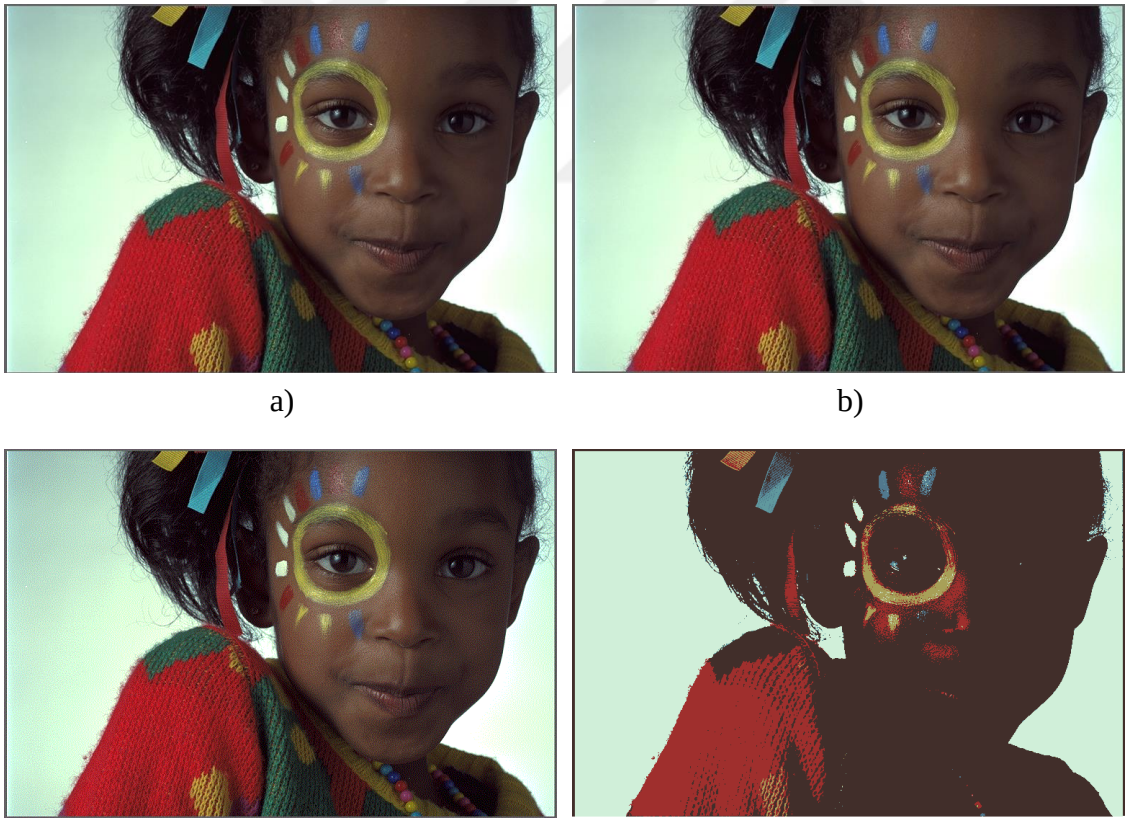
h)

Şekil 4.11. (devam) Çocuk a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.10. Performans ölçütleri: Çocuk

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Çocuk, 768x512 (1152,00 kB)	JPEG	2,84	33,27	405,63
	GIF	5,95	23,57	193,77
	Önerilen (m=1)	14,96	19,66	77,00
	K-ortalamalar (k=8)	6,47	22,61	178,04
	Önerilen (m=2)	6,91	24,04	166,72
	K-ortalamalar (k=27)	4,02	26,65	286,49

Şekil 4.12(a)'da Kız görüntüsünün orijinali, Çizelge 4.11'de ise görüntüye ait deneysel sonuçlar verilmiştir. Sonuçlar incelendiğinde, Şekil 4.12(b)'deki JPEG sıkıştırmasında yüksek benzerlik elde edilmiş fakat sıkıştırma oranı yetersiz kalmıştır. Şekil 4.12(c)'deki GIF sıkıştırması ise hem PSNR hem de sıkıştırma oranı bazında düşük performans elde etmiştir. Çıkış görüntüsü Şekil 4.12(d)'deki gibi olan tek eşikli sınıflandırmada yüksek sıkıştırma oranıyla beraber kızın yüzünün seçilemeyeceği derecede bir bilgi kaybı meydana geldiği görülmektedir. Şekil 4.12(e)'de verilmiş olan K-ortalamalar yönteminin 8 sınıflı ayrıştırması ve geliştirilen algoritmanın Şekil 4.12(f)'de verilmiş olan iki eşikli ayrıştırması yakın değerler elde etmiştir. Her iki durumda da PSNR değerleri düşük seviyelerde kalmıştır. Ayrıca Şekil 4.12(e)'de arka plandaki mavi tonu haricinde görüntüdeki diğer mavi renklerin kaybolduğu gözlenmektedir. K-ortalamalar yönteminin Şekil 4.12(g)'de verilmiş olan 27 sınıflı ayrıştırmasında ise iyi bir PSNR değerine ulaşılmış olsa da yeterli sıkıştırma oranı elde edilememiştir. Sonuçlara ait grafiksel gösteri ise Şekil 4.12(h)'de verilmiştir.

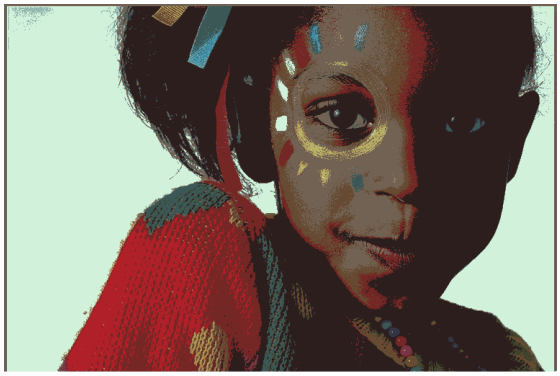


a) b)

c) d)

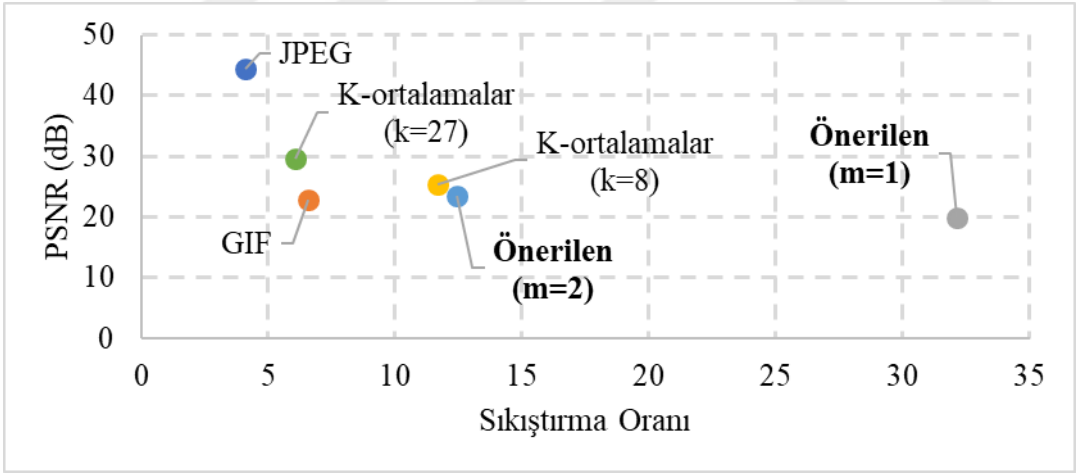
Şekil 4.12. Kız a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

e)



f)

g)



h)

Şekil 4.12. (devam) Kız a) Orijinal b) JPEG c) GIF d) Önerilen (m=1) e) K-ortalamalar (k=8) f) Önerilen (m=2) g) K-ortalamalar (k=27) h) Sıkıştırma oranı – PSNR değişimi

Çizelge 4.11. Performans ölçütleri: Kız

Orijinal Görüntü (BMP)	Sıkıştırma yöntemi	Sıkıştırma oranı: CR	Ortalama $PSNR_{rgb}$	Sıkıştırılmış boyut (kB)
Kız, 768x512 (1152,00 kB)	JPEG	4,10	44,35	280,75
	GIF	6,57	22,77	175,44
	Önerilen (m=1)	32,13	19,98	35,85
	K-ortalamalar (k=8)	11,66	25,35	98,81
	Önerilen (m=2)	12,42	23,47	92,75
	K-ortalamalar (k=27)	6,07	29,72	189,86

5. SONUÇ

Bu çalışmada sınıflandırma algoritmaları ve tekrarlı uzunluk kodlaması birleştirilerek yeni bir kayıplı sıkıştırma metodu geliştirilmiştir. Sınıflandırma sürecinde renk uzayı tabanlı renkli görüntü sınıflandırma ve K-ortalamalar algoritmaları, RLE tekniği ile entegre edilmiştir. Sınıflandırma sonucunda oluşan kümelere dahil olan piksellerin renk ortalamaları alınarak renk indirgemesi yapılmış ve kod kitabı tasarlanmıştır. Her iki ayrıştırma yönteminde de oluşan sınıf sayısı kontrol edilebilmektedir. Birinci yaklaşımda eşik sayısı, ikinci yaklaşımda ise küme sayısı parametreleri kullanıcıdan alınabilmektedir. Küme veya eşik sayısının artması sıkıştırılmış görüntüdeki bilgi kaybını azaltırken diğer taraftan sıkıştırma oranına olumsuz yönde etki etmiştir. RLE kayıpsız bir sıkıştırma algoritması olmasına rağmen, renk indirgemede bilgi kaybı kaçınılmaz olmaktadır. Dolayısıyla, bu çalışma ile iki durumun birleşimi olan hibrit bir çözüm önerisi tasarlanmıştır. Diğer yaklaşımlarda sıkıştırma işlemi renk bilgileri üzerinden yapılmakta iken, geliştirilen yöntemde sınıflandırma sonucu elde edilen etiket bilgileri sıkıştırılarak bellek alanından önemli ölçüde tasarruf sağlanmıştır.

Geliştirilen algoritmanın benzerlik ölçütü açısından eksikleri bulunsa da sıkıştırma oranı tarafındaki başarısı yüksektir. Eşik veya küme sayısı artırılarak daha az bilgi kaybı sağlanabilmekte, dolayısıyla sıkıştırma oranına müdahale edilebilmektedir. Ayrıca, geliştiren teknik farklı sınıflandırma yöntemleriyle kullanılabilme esnekliğine de sahiptir. Bu bağlamda, sınıflandırıcı olarak K-ortalamalar algoritmasının performansı incelenmiştir. K-ortalamalar kullanıldığı durumda genellikle daha yüksek PSNR değerlerinin elde edildiği fakat sıkıştırma oranlarında yeterli seviyelere çıkılamadığı görülmüştür. Bununla beraber, K-ortalamalar'ın başlangıç merkezlerini rastgele seçmesi, her seferinde farklı sonuçlar üretmesine sebep olmakta ve güvenilirliği düşürmektedir.

Geliştirilen yaklaşım, renk geçiş ve değişimlerinin az olduğu ve renk dağılımının homojenlik gösterdiği görüntülerde daha iyi performans göstermekle beraber, kayıpların fazla önemsenmediği kullanım alanları için tercih edilebilir bir alternatif olarak öne çıkmaktadır. Bununla beraber, PSNR oranlarının artırılması ve sıkıştırma algoritması tarafında farklı metot ve stratejilerin incelenmesi ise ileriki çalışma konularıdır.

KAYNAKLAR

1. Sayood, K. (2012). *Introduction to data compression (Third Edition).* San Francisco: Morgan Kaufmann, 1-2.

2. Mesut, A. (2006). *Veri Sıkıştırmada Yeni Yöntemler*, Doktora Tezi, Trakya Üniversitesi Fen Bilimleri Enstitüsü, Edirne, 1-2.

3. Moradi, B. and Demirci, R. (2015). *Fuzzy logic and graph based segmentation*. 23nd Signal Processing and Communications Applications Conference (SIU), Malatya, Turkey, 479-482.

4. Yaman, K., Sarucan, A., Atak, M. ve Aktürk, N. (2001). Dinamik çizelgeleme için görüntü işleme ve arıma modelleri yardımıyla veri hazırlama. *Gazi Üniversitesi Mühendislik-Mimarlık Fakültesi Dergisi, 16*(1), 19-40.

5. Patane, G. and Russo, M. (2001). The enhanced LBG algorithm. *Neural Networks, 14*(9), 1219-1237.

6. Horng, M-H. (2012). Vector quantization using the firefly algorithm for image compression. *Expert Systems with Applications, 39*(1), 1078-1091.

7. Ueda, N. and Nakano, R. (1994). A new competitive learning approach based on an equidistortion principle for designing optimal vector quantizers. *Neural Networks, 7*(8), 1211-1227.

8. Tsolakis, D. and Tsekouras, G. (2016). A fuzzy-soft competitive learning approach for grayscale ımage compression. *Unsupervised Learning Algorithms*, New York: Springer, 385-404.

9. Nasrabadi, N. M. and Feng, Y. (1998). *Vector quantization of images based upon the Kohonen self-organizing feature maps.* IEEE 1988 International Conference on Neural Networks, San Diego, 101-105.

10. Celebi, M. E. and Schaefer, G. (2010). *Neural gas clustering for color reduction*. International Conference on Image Processing, Computer Vision, and Pattern Recognition, Las Vegas, 429-432.

11. MacQueen, J. (1967). *Some methods for classification and analysis of multivariate observations.* Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability. California.

12. Linde, Y., Buzo, A. and Gray, R. (1980). An algorithm for vector quantizer design. *IEEE Transactions on Communications, 28*(1), 84-95.

13. Kekre, H. B., Natu, P. and Sarode, T. (2016). Color image compression using vector quantization and hybrid wavelet transform. *Twelfth International Multi-Conference on Information Processing.*

14. Chiranjeevi, K. and Jena, U. R. (2016). Image compression based on vector quantization using cuckoo search optimization technique. *Ain Shams Engineering Journal* (9), 1417-1431.

15. Celebi, M. E. (2011). Improving the performance of k-means for color quantization. *Image and Vision Computing, 29*(4), 260-271.

16. Dursunoğlu, N. (2006). *Vektör Kuantalama ile Görüntü Sıkıştırma,* Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü, İstanbul, 4-7.

17. Grgic, S., Grgic, M. and Zovko-Cihlar, B. (2001). Performance analysis of ımage compression using wavelets. *IEEE Transactions on Industrial Electronics, 48*(3), 682-695.

18. Kaushik, P. and Sharma, Y. (2012). Comparison of different image enhancement techniques based upon psnr and mse. *International Journal of Applied Engineering Research, 7*(11).

19. Gupta, P., Srivastava, P., Bhardwaj, S. and Bhateja, V. (2011). *A modified psnr metric based on hvs for quality assessment of color images*. International Conference on Communication and Industrial Application. Kolkata, West Bengal.

20. Mesut, A. ve Carus, A. (2005). *Kayıpsız görüntü sıkıştırma yöntemlerinin karşılaştırılması*. II. Mühendislik Bilimleri Genç Araştırmacılar Kongresi. İstanbul.

21. Miano, J. (1999). *Compressed image file formats: jpeg, png, gif, xbm, bmp.* Massachusetts: Addison Wesley Longman Inc, 189.

22. Tan, L.K. (2006). Image file formats. *Biomedical Imaging and Intervention Journal, 2*(1), 1-7.

23. Murray, J. D. and vanRyper, W. (1996). *Encyclopedia of graphics file formats* (2). California: O'Reilly Media, 586.

24. Boutell, T. (1997). *RFC 2083: PNG (Portable Network Graphics) Specification.* IETF.

25. Internet: GIF89a Specification, URL: http://webcitation.org/query?url=https%3A%2F%2Fwww.w3.org%2FGraphics%2FGIF%2Fspec-gif89a.txt&date=2019-05-09, Son Erişim Tarihi: 5 Mayıs 2019

26. Internet: LZW Patent Information, http://webcitation.org/query?url=http%3A%2F%2Fweb.archive.org%2Fweb%2F20080113182757%2Fhttp%3A%2F%2Fwww.unisys.com%2Fabout__unisys%2Flzw%2F&date=2019-05-09, Son Erişim Tarihi: 5 Mayıs 2019.

27. Pandit, R., Khosla, N., Singh, G. and Sharma, H. (July, 2013). Image compression and quality factor in case of jpeg image format. *International Journal of Advanced Research in Computer and Communication Engineering, 2*(7), 2578-2581.

28. Demirci, R., Güven, U. ve Kahraman, H. (2014). Görüntülerin renk uzayı yardımıyla ayrıştırılması. *İleri Teknoloji Bilimleri Dergisi, 3*(1), 1-8.

29. Güvenç, U., Elmas, Ç. ve Demirci, R. (2008). Renkli görüntülerin otomatik ayrıştırılması. *Politeknik Dergisi, 11*(1), 9-12.

30. Kumar, V., Lal, T., Dhuliya, P. and Pant, D. (2016). *A study and comparison of different ımage segmentation algorithms.* 2nd International Conference on Advances in Computing, Communication, and Automation (ICACCA), Bareilly, 1-6.

31. Kamdi, S. and Krishna, R. (2012). Image segmentation and region growing algorithm. *International Journal of Computer Technology and Electronics Engineering (IJCTEE), 2*(1), 103-107.

32. R, M. and Radha, M. (2011). Edge detection techniques for image segmentation. *International Journal of Computer Science and Information Technology (IJCSIT), 3*(6), 259-267.

33. Bezdek, J., Ehrlich, R. and Full, W. (1984). FCM: the fuzzy c-means clustering algorithm. *Computers and Geosciences, 10*(2-3), 191-203.

34. Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics, 9*(1), 62-66.

35. Kapur, J., Sahoo, P. and Wong, A. (1985). A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing, 29*(3), 273-285.

36. Gonzalez, R. C. and Woods, R. E. (2007). *Digital Image Processing* (3). New Jersey: Prentice Hall, 1-3.

37. Ibraheem, N. A., Hasan, M. M., Khan, R. Z. and Mishra, P. K. (2012). Understanding color models: a review. *ARPN Journal of Science and Technology, 2*(3), 265-275.

38. Işık, M. ve Çamurcu, Y. (2007). K-means, k-medoids ve bulanık c-means algoritmalarının uygulamalı olarak performanslarının tespiti. *İstanbul Ticaret Üniversitesi Fen Bilimleri Dergisi, 6*(11), 31-45.

39. Bozkurt, K. (2009). *Retina Görüntülerinin Ayrıştırılması,* Yüksek Lisans Tezi, Düzce Üniversitesi, Düzce, 31-33.

40. Kılıçarslan, M., Tanyeri, U. ve Demirci, R. (2018). Renkli görüntüler için tek boyutlu histogram. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 6*(4), 1094-1107.

41. Kahraman, A., Farshi, T. ve Demirci, R. (2018). Renkli görüntülerin çok seviyeli eşiklenmesi ve sınıflandırılması. *Düzce Üniversitesi Bilim ve Teknoloji Dergisi, 6*(4), 846-859.

42. Sathya, P. and Kayalvizhi, R. (2011). Amended bacterial foraging algorithm for multilevel thresholding of magnetic resonance brain images. *Measurement, 44*(10), 1828-1848.

43. Horé, A. and Ziou, D. (August, 2010). *Image quality metrics: psnr vs. ssim.* 20th International Conference on Pattern Recognition, Istanbul, 2366-2369.

44. Elmas, Ç., Demirci, R. and Güvenç, U. (2013). Fuzzy diffusion filter with extended neighborhood. *Expert Systems with Applications, 40*(3), 866-872.

45. Kalkan, M. B. ve Demirci, R. (2019). *Görüntülerin renk uzayı yardımıyla ayrıştırılması ve tekrarlı uzunluk kodlaması tabanlı görüntü sıkıştırma*. 3rd International Symposium on Innovative Approach in Scientific Studies, Ankara, 667-671.

46. Stallings, W. (2013). *Computer Organization and Architecture* (9). New Jersey: Pearson Education Inc., 447-448.

47. Internet: SIPI Image Database, University of Southern California, URL: http://webcitation.org/query?url=http%3A%2F%2Fsipi.usc.edu%2Fdatabase%2Fdatabase.php%3Fvolume%3Dmisc&date=2019-05-09, Son Erişim Tarihi: 5 Mayıs 2019.

48. Internet: Public-Domain Test Images for Homeworks and Projects, University of Wisconsin-Medison, URL: http://webcitation.org/query?url=https%3A%2F%2Fhomepages.cae.wisc.edu%2F~ece533%2Fimages%2F&date=2019-05-09, Son Erişim Tarihi: 5 Mayıs 2019.

EKLER

EK-1. Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin byte dizisi görünümü (kırmızı blok: genişlik-yükseklik bilgisi, yeşil blok: kod kitabı uzunluğu, mavi blok: kod kitabı, siyah blok: sıkıştırılmış veri)

```
00 01 00 00 00 01 00 00 08 00 00 00 55 48 66 FF 61 6A A5 FF 81 98 7B FF 8C 9A
99 FF A4 86 6A FF D2 1A 9C FF B7 A4 7C FF D9 C5 A3 FF 00 06 00 01 00 08 59 17
04 09 08 F4 04 04 02 07 04 06 08 ED 04 03 02 0F 04 05 08 E5 04 04 02 17 04 05
08 DE 04 03 02 0A 01 02 02 13 04 01 08 DA 04 03 02 0A 01 04 02 03 01 04 02 0E
08 D7 04 03 02 0A 01 04 02 0B 01 04 02 09 08 D4 04 03 02 0A 01 04 02 13 01 05
02 03 08 D1 04 03 02 0A 01 04 02 19 01 04 02 01 08 D1 04 01 02 09 01 04 02 20
06 01 08 D1 02 07 01 04 02 23 06 01 08 D1 02 04 01 05 02 25 08 D2 02 01 01 05
02 28 08 D2 01 03 02 2B 08 D2 06 01 02 2D 08 D3 02 2D 08 D3 02 2D 08 D3 02 2D
08 D3 02 2D 08 D3 02 2D 08 D3 02 2D 08 D3 02 2D 08 D3 02 2D 08 D3 02 2B 01 02
08 D2 04 01 02 1F 01 05 02 01 01 01 02 05 01 02 08 D2 02 21 01 0D 08 D2 04 01
02 1F 01 0E 08 D2 04 01 02 21 06 01 02 01 01 0A 08 D2 02 21 08 02 06 02 01 0A
03 02 07 02 08 CD 02 21 08 04 06 02 01 07 02 01 01 04 03 03 08 CA 02 21 08 06
06 02 01 0E 03 04 08 C5 02 21 08 08 06 01 02 01 01 11 03 03 08 1A 07 01 03 04
07 01 08 A1 02 21 08 0A 06 01 02 02 01 13 03 05 08 13 03 01 01 05 08 A1 02 20
04 01 08 0E 01 1C 06 02 08 0C 07 01 01 05 08 A1 02 21 08 0F 03 01 01 1F 02 01
06 01 08 07 05 01 01 05 08 A1 02 22 08 10 03 01 01 24 08 02 01 06 08 A1 02 21
01 03 08 10 01 0F 02 02 01 07 02 01 01 01 02 01 01 01 02 01 01 0E 08 A1 02 22
01 03 02 01 08 10 03 01 01 0C 02 02 01 1D 06 01 08 9D 02 22 01 05 02 01 08 10
03 01 01 04 02 01 01 05 02 01 01 23 06 01 08 98 02 22 01 07 02 01 08 10 01 20
02 03 01 0E 02 01 06 01 08 93 02 23 01 08 08 11 01 20 02 02 01 11 02 02 06 01
08 8E 02 23 01 0A 08 10 04 01 01 0C 02 01 01 12 02 03 01 16 06 01 08 89 02 23
01 0C 08 11 01 0A 02 01 01 07 02 02 01 26 06 02 08 84 02 24 01 0C 02 01 08 10
04 01 01 3E 06 01 08 7F 02 24 01 0E 02 01 08 10 04 01 01 26 02 02 01 1A 06 02
08 78 02 24 01 10 02 01 08 10 04 01 01 12 02 02 01 03 02 03 01 04 02 01 01 05
02 03 01 1E 02 01 06 01 08 73 02 24 01 12 02 01 08 10 03 01 01 1B 02 01 01 08
02 01 01 23 02 01 06 01 08 6E 02 24 01 14 06 01 08 10 04 01 01 1B 02 01 01 08
02 03 01 25 05 01 06 01 08 68 02 25 01 15 08 11 03 01 01 14 02 01 01 01 02 01
01 38 02 01 06 01 08 62 04 01 02 25 01 17 08 11 04 01 01 02 02 01 01 0B 02 03
01 01 02 01 01 01 02 02 01 3C 02 01 06 01 08 5E 02 25 01 18 02 01 08 11 04 01
02 01 01 01 02 02 01 23 02 01 01 2E 02 01 06 01 08 58 02 26 01 19 02 01 08 11
04 01 01 02 02 03 01 1F 02 01 08 01 01 34 06 01 08 52 04 01 02 26 01 1B 02 01
08 11 01 02 02 03 01 0E 02 01 01 0C 02 04 06 01 02 01 01 39 06 02 08 4C 02 26
01 1D 02 01 08 11 01 01 02 04 01 0C 02 01 01 0D 02 01 01 01 02 01 01 04 02 01
01 1C 02 01 01 01 02 02 01 1C 06 02 08 45 04 01 02 26 01 1F 02 01 08 10 04 01
01 06 02 01 01 0C 02 04 01 1B 02 04 01 06 02 01 01 29 05 01 08 41 04 01 02 27
01 20 02 01 08 11 01 04 02 03 01 0B 02 02 01 1E 02 03 01 1D 02 01 01 01 02 02
01 0F 08 41 04 01 02 27 01 22 08 11 04 01 01 03 02 04 01 15 02 01 01 06 02 01
01 02 02 03 01 29 02 02 01 0F 08 41 02 28 01 24 08 11 04 01 01 05 02 02 01 05
02 05 01 09 02 03 01 01 02 01 01 02 02 06 01 01 02 01 01 07 02 02 01 13 02 03
01 19 08 41 04 01 02 27 01 26 08 11 03 01 01 0D 02 01 01 0B 02 05 01 1D 02 01
01 01 02 02 01 20 08 41 02 29 01 26 02 01 08 11 04 01 01 17 02 01 01 22 02 03
01 1A 02 04 01 02 08 41 02 29 01 28 02 01 08 11 03 01 01 0B 02 01 01 01 02 01
01 02 02 01 01 10 02 05 01 14 02 01 01 0A 02 01 01 0B 02 01 01 06 02 01 01 02
08 41 02 29 01 2A 02 01 08 11 01 09 02 03 01 01 02 01 01 07 02 04 01 18 02 01
01 1E 02 01 01 09 08 40 01 01 02 2A 01 2B 02 01 08 11 01 13 02 03 01 01 02 01
01 0A 02 02 01 20 02 02 01 12 08 3F 01 01 02 2B 01 2D 02 01 08 10 04 01 01 20
02 02 01 34 08 3E 01 01 04 01 08 01 02 2A 01 2F 08 11 04 01 01 09 02 01 01 0A
02 01 01 0A 02 02 01 33 08 3E 04 01 08 02 02 2B 01 30 08 11 04 01 01 09 02 01
01 08 02 04 01 03 02 01 01 05 02 02 01 1D 02 01 01 13 08 3D 04 01 08 03 02 2B
01 31 02 01 08 11 04 01 01 0A 02 01 01 08 02 05 01 06 02 01 01 0F 02 01 01 1B
02 01 01 05 08 3C 01 01 08 04 02 2B 01 33 02 01 08 11 01 12 02 05 01 06 02 01
01 19 02 02 01 16 08 3B 01 01 08 05 04 01 02 2B 01 34 02 01 08 11 01 13 02 04
01 07 02 02 01 17 02 02 01 14 08 3A 01 01 08 06 02 2F 01 33 02 01 08 11 01 0A
02 01 01 08 02 03 01 1F 02 01 01 01 02 06 01 09 02 01 01 04 08 39 01 01 08 07
02 2F 01 35 06 01 08 10 04 01 01 12 02 02 01 02 02 01 01 11 02 01 01 0A 02 06
```

EK-1. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin byte dizisi görünümü

```
01 0C 02 03 01 01 08 38 01 01 08 08 02 31 01 35 08 11 04 01 01 31 02 03 01 0B
02 02 01 06 08 37 01 01 08 08 02 33 01 35 02 01 08 11 04 01 01 03 02 06 01 28
02 02 01 0D 02 03 01 02 08 36 05 01 04 01 08 07 01 02 02 35 01 34 02 01 08 11
04 01 02 02 01 03 02 05 01 25 02 05 01 0B 02 03 01 01 08 35 01 01 04 01 08 07
02 01 01 02 02 36 01 35 02 01 08 10 04 01 01 08 02 03 01 23 02 02 01 01 02 01
01 10 08 34 07 01 04 01 08 07 02 01 01 03 02 38 01 34 02 01 08 10 02 01 04 03
02 01 01 07 02 01 01 06 02 02 01 2D 08 33 05 01 02 01 08 07 04 01 01 03 02 3A
01 34 08 18 04 03 01 26 02 01 01 02 02 01 01 0D 08 32 07 01 01 01 08 07 04 01
01 04 02 3C 01 32 02 01 08 1C 04 03 02 01 01 20 02 01 01 10 08 32 01 01 08 07
04 01 01 06 02 3D 01 31 08 22 04 03 02 01 01 1B 02 01 01 01 02 02 01 0A 02 01
01 01 08 31 01 01 08 08 01 06 02 3F 01 30 08 11 04 01 08 17 04 02 02 01 01 12
02 04 01 0E 02 01 08 30 07 01 08 08 01 07 02 42 01 2D 08 0F 04 01 01 04 02 01
08 01 04 02 08 17 04 02 02 01 01 0D 02 03 01 0F 08 2F 01 01 08 08 01 08 02 43
01 2C 08 0F 04 01 01 09 02 02 08 1A 04 02 02 01 01 06 02 05 01 0E 08 2E 01 01
04 01 08 07 01 09 02 46 01 29 08 10 04 03 02 01 01 0A 04 01 08 1C 04 02 02 01
01 0A 02 02 01 07 08 2D 01 01 04 01 08 07 02 01 01 09 02 46 01 29 08 10 02 01
08 05 04 02 02 02 01 04 08 04 01 03 02 02 08 01 04 01 08 17 04 03 01 09 02 02
01 03 08 2C 01 01 04 01 08 07 02 01 01 0A 02 49 01 26 08 0D 04 01 01 07 02 01
08 06 01 01 04 01 08 04 04 01 01 07 02 03 04 02 08 17 04 03 01 08 08 2B 01 01
04 01 08 07 04 01 01 0B 02 4A 01 25 08 08 04 01 02 01 01 11 05 01 01 01 08 06
04 01 02 01 04 01 02 01 01 0A 02 01 04 01 07 01 08 01 04 02 08 15 04 02 02 01
01 02 08 2A 07 01 04 01 08 07 04 01 01 0C 02 4C 01 22 02 01 08 04 04 01 02 01
01 17 08 06 06 01 08 04 04 03 02 02 01 0A 02 01 01 01 04 01 08 01 04 02 08 3D
07 01 02 01 08 07 04 01 01 0D 02 4E 01 21 02 02 01 1B 08 05 01 05 02 02 08 05
04 03 02 01 01 0A 02 02 08 01 06 01 08 02 04 01 08 36 07 01 01 01 08 08 01 0E
02 4F 01 3C 02 01 08 05 01 0B 02 02 08 05 04 03 02 01 01 0A 02 02 06 01 08 34
01 01 08 08 01 0F 02 51 01 3A 08 05 02 01 01 10 02 01 01 01 02 02 08 04 04 04
01 0B 02 01 05 01 02 01 08 01 04 01 08 2A 01 01 08 08 01 10 02 53 01 37 04 01
08 02 04 02 02 01 01 19 02 02 06 01 08 02 04 05 01 0B 02 02 08 26 01 01 08 08
01 11 02 55 01 34 02 01 08 02 04 01 01 22 02 02 06 01 08 03 04 05 01 06 08 25
01 01 08 08 01 12 02 56 01 14 02 01 04 02 01 1B 04 02 02 02 01 29 02 03 08 04
04 04 08 24 01 01 04 01 08 07 01 13 02 58 01 0D 02 02 04 06 01 19 02 01 04 02
02 01 01 32 02 01 06 02 08 24 05 01 04 01 08 07 02 01 01 13 02 59 01 09 02 01
04 03 08 01 04 02 08 01 04 02 08 01 04 01 01 17 02 01 04 02 02 01 01 37 08 22
07 01 02 01 08 07 04 01 01 14 02 58 01 05 02 04 04 02 08 08 04 02 01 17 02 01
04 01 01 3A 08 21 07 01 01 01 08 07 04 01 01 15 02 57 01 04 02 02 04 03 08 08
04 02 01 01 02 01 01 17 02 02 01 3B 08 20 07 01 01 01 08 07 04 01 01 15 02 59
01 02 02 01 04 01 08 07 04 01 02 01 04 01 01 1D 02 02 01 3B 08 20 01 01 08 07
04 01 01 17 02 57 01 02 02 01 08 03 04 02 02 03 01 21 08 01 04 01 01 3B 08 1E
07 01 01 01 08 08 01 17 02 59 01 01 04 01 08 01 04 01 02 02 01 24 02 01 08 01
04 02 01 3A 08 1D 07 01 01 01 08 08 01 18 02 46 01 01 02 01 01 01 02 11 08 03
04 01 01 0B 05 02 01 18 04 01 08 03 01 3A 08 1C 07 01 01 01 04 01 08 07 01 1A
02 42 01 04 02 13 08 04 03 01 01 0A 05 02 01 01 05 01 01 16 08 06 04 01 01 37
08 1C 01 01 04 01 08 07 01 1A 02 3D 01 03 02 02 01 01 02 02 04 01 08 01 04 02
02 11 08 05 01 06 05 01 01 02 05 03 01 01 05 01 08 02 04 02 02 01 01 11 08 06
03 01 05 01 01 36 08 1B 05 01 04 01 08 07 02 01 01 1A 02 3D 01 02 02 03 04 01
08 02 04 01 01 03 02 11 08 05 01 05 05 08 08 05 04 02 08 01 01 0F 08 04 04 01
01 02 08 01 01 36 08 1A 05 01 04 01 08 07 04 01 01 19 02 36 01 01 02 08 04 01
08 03 01 08 02 11 08 05 01 05 05 08 06 01 08 03 02 01 04 01 08 02 01 0F 08 04
04 01 01 39 08 19 07 01 02 01 08 07 04 01 01 19 02 34 01 04 02 03 04 02 08 01
04 01 02 03 08 02 01 08 02 10 04 01 08 05 01 05 05 08 01 01 04 01 08 01 01 02
02 01 04 01 08 01 01 0F 08 04 04 01 01 39 08 18 07 01 01 01 08 07 04 01 01 1A
02 34 01 03 02 02 04 02 02 02 01 05 08 02 07 01 01 07 02 10 04 01 08 05 01 04
05 0A 01 01 02 01 01 14 08 05 01 39 08 18 01 01 08 07 04 01 01 1A 02 36 01 02
08 01 02 01 01 09 08 02 04 01 01 07 02 10 04 01 08 05 01 03 05 0C 01 15 08 04
04 01 01 39 08 16 07 01 01 01 08 07 04 01 01 1A 02 39 08 01 01 0A 08 03 01 07
```

EK-1. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin byte dizisi görünümü

```
02 10 04 01 08 05 05 02 01 01 05 0A 07 01 05 01 02 03 01 12 08 04 04 01 01 39
08 16 01 01 08 08 01 1A 02 39 04 01 08 01 03 01 01 08 02 01 08 03 01 04 05 01
01 02 02 10 04 01 08 05 07 02 01 01 05 0C 02 03 01 12 08 04 04 01 01 39 08 14
07 01 01 01 08 08 01 1A 02 3A 04 01 08 01 04 01 01 08 02 01 08 03 07 01 01 02
05 01 01 01 05 02 02 10 04 01 08 05 07 02 05 0D 02 03 01 12 08 04 04 01 01 39
08 14 01 01 08 09 01 19 02 3B 08 02 04 01 01 08 02 01 08 03 05 01 01 02 05 04
02 10 04 01 08 06 07 01 03 01 07 01 05 0B 02 03 01 12 08 04 04 01 01 39 08 13
01 01 04 01 08 08 04 01 01 18 02 3C 08 03 02 01 01 07 04 01 08 04 01 02 05 04
02 10 08 0A 07 01 05 0A 02 03 01 11 02 01 08 04 04 01 01 39 08 12 01 01 04 01
08 09 01 18 02 3D 08 03 07 01 01 07 04 01 08 04 01 02 05 04 02 10 08 0A 07 01
05 0A 02 03 01 12 08 04 04 01 01 39 08 11 07 01 04 01 08 0A 03 02 04 01 01 15
02 3D 08 04 01 07 08 05 01 02 05 04 02 10 08 08 07 03 05 02 07 01 05 07 02 03
01 11 02 01 08 04 02 01 01 39 08 11 01 01 08 0F 03 01 02 02 04 01 03 01 01 0E
02 3E 08 04 01 07 08 05 01 02 05 03 06 01 02 10 08 08 07 03 05 0A 02 02 01 12
02 01 08 04 02 01 01 39 08 22 03 01 04 02 08 01 01 0C 02 09 01 04 02 32 04 01
08 04 01 07 08 06 05 03 07 01 06 01 02 10 08 08 07 04 05 01 07 03 05 03 07 01
05 01 02 01 01 0D 02 02 01 04 02 01 08 04 02 01 01 39 08 22 04 03 08 01 01 0C
02 05 01 07 02 33 04 01 08 04 01 07 08 06 05 03 07 01 06 01 02 10 08 08 07 09
05 01 07 02 05 01 02 01 01 08 02 08 01 03 02 01 08 04 02 01 01 39 08 23 04 02
08 01 01 0B 02 07 01 05 02 34 04 01 08 04 01 07 08 06 01 01 05 02 07 01 06 01
02 10 08 08 07 09 05 01 07 02 05 01 01 06 02 0A 01 04 02 01 08 04 04 01 01 39
08 26 01 0A 02 08 01 06 02 33 04 01 08 04 04 01 01 06 08 05 07 01 05 03 07 01
06 01 02 10 08 08 07 0A 05 03 01 02 02 03 01 02 02 0A 01 03 02 01 08 04 04 01
01 39 08 26 01 09 02 09 01 02 04 01 01 04 02 32 04 01 08 04 04 01 01 06 08 06
05 03 07 01 06 01 02 10 08 08 07 0C 05 01 01 02 02 02 01 03 02 0A 01 03 08 05
04 01 01 39 08 19 04 01 01 04 02 01 04 01 02 01 01 03 04 01 01 09 02 0A 01 02
08 01 01 04 02 32 04 01 08 04 04 01 01 06 08 05 07 01 05 02 07 02 08 01 02 10
08 08 07 0C 05 01 01 02 02 02 01 02 02 0B 01 03 08 05 04 01 01 39 08 15 02 01
01 17 02 0D 08 01 01 04 02 32 08 05 04 01 01 06 08 05 07 01 05 01 07 03 02 11
08 09 07 0C 01 01 02 03 01 01 02 0D 01 02 08 05 04 01 01 39 08 16 03 02 01 15
02 0C 04 01 08 01 03 01 01 03 02 32 04 01 08 04 04 01 01 06 08 05 07 01 01 01
07 03 08 01 02 10 08 09 07 0C 01 01 02 11 01 02 08 05 04 01 01 39 08 1D 04 01
03 01 01 0D 02 0D 08 03 01 03 02 32 08 05 04 01 01 05 02 01 08 06 05 01 07 02
08 02 02 10 08 09 07 0B 08 01 02 12 01 02 08 05 04 01 01 39 08 20 01 0A 02 0F
08 03 01 03 02 32 04 01 08 04 02 01 01 05 02 01 08 05 04 01 05 01 07 02 08 01
04 01 02 10 08 09 07 07 08 01 07 02 08 02 02 14 08 05 04 01 01 39 08 20 01 0A
02 0F 08 03 02 01 01 02 02 32 08 05 04 01 01 04 05 01 02 01 08 05 03 01 05 01
08 04 02 10 08 09 07 07 08 01 07 02 08 02 02 14 08 05 02 01 01 39 08 20 03 01
01 08 02 10 08 03 02 01 01 02 02 32 08 05 04 01 01 04 05 01 02 01 08 05 03 01
05 01 08 04 02 10 08 09 07 08 08 01 07 02 08 01 02 14 08 05 04 01 02 03 01 36
08 21 01 07 02 11 08 03 04 01 01 01 02 33 08 05 04 01 01 02 07 01 01 02 08 06
01 01 05 01 08 03 04 01 02 10 08 09 07 09 08 03 02 14 08 05 02 06 01 05 02 03
01 2C 08 21 03 01 01 05 02 01 01 01 02 10 08 04 01 02 02 32 08 05 04 01 01 01
02 01 04 01 02 01 06 01 08 06 03 01 07 01 08 03 04 01 02 10 08 09 07 0A 08 02
02 14 08 05 04 01 02 06 01 04 02 0A 01 25 08 23 01 01 02 04 01 01 02 10 08 04
01 01 02 33 08 05 04 01 01 01 02 01 08 01 06 02 08 0B 04 01 02 10 08 09 07 0A
08 02 02 14 08 05 02 07 01 04 02 13 01 1C 08 25 02 01 04 01 08 01 01 02 02 0F
08 04 04 01 02 33 08 05 04 01 01 01 05 01 04 01 06 01 08 07 07 02 08 04 02 10
08 08 07 0A 08 01 07 01 08 01 02 14 08 05 02 08 01 02 02 19 01 17 08 24 02 02
08 02 01 02 02 0E 04 01 08 05 02 33 08 05 04 01 01 02 08 01 06 01 08 07 03 01
07 01 08 04 02 10 08 08 07 0C 08 01 02 14 08 05 04 01 02 2A 01 0F 08 23 01 01
02 03 01 02 02 0F 04 01 08 05 02 33 08 05 04 01 01 01 02 01 08 02 06 01 08 06
03 01 08 05 02 10 08 09 07 0A 08 02 02 14 08 05 02 30 01 0A 08 22 02 04 04 01
01 02 02 10 08 05 02 33 08 05 04 01 01 01 08 01 04 02 08 07 04 01 08 05 02 10
08 08 07 0B 08 02 02 14 08 05 02 37 01 03 08 22 04 01 02 03 04 01 02 11 04 01
08 04 04 01 02 33 08 05 04 01 01 01 04 02 08 0E 02 10 08 08 07 0B 08 02 02 14
```

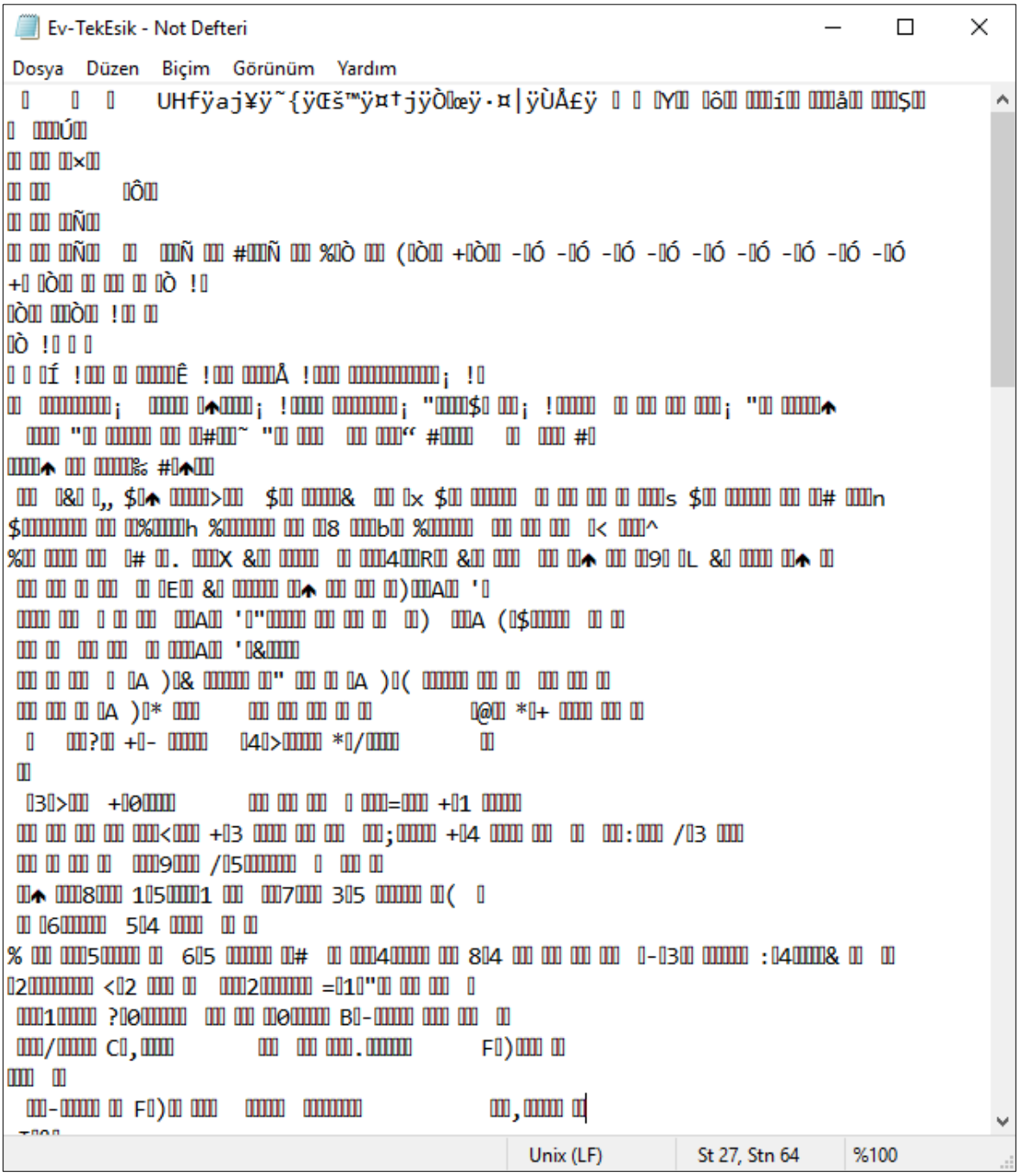
EK-1. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin byte dizisi görünümü

```
08 05 02 3A 08 23 02 03 08 01 02 11 04 01 08 04 04 01 02 33 08 05 04 02 08 10
02 10 08 08 07 0C 08 01 02 14 08 05 02 3A 08 23 02 03 01 02 02 10 04 01 08 04
04 01 02 33 08 17 02 10 08 08 07 0B 08 01 04 01 02 13 04 01 08 05 02 3A 08 23
02 04 01 01 02 10 04 01 08 04 02 34 08 17 02 10 08 08 07 0C 08 01 02 13 04 01
08 05 02 3A 08 23 02 04 01 01 02 10 04 01 08 04 02 32 04 02 08 14 04 01 02 01
04 01 02 0F 04 01 08 08 07 0B 08 01 02 15 08 05 02 3A 08 23 02 04 01 01 02 10
04 01 08 04 02 31 04 01 08 11 02 01 01 05 02 11 04 01 08 08 07 09 08 01 07 01
08 01 02 14 04 01 08 05 02 3A 08 23 02 04 01 01 02 10 04 01 08 04 02 31 08 0C
04 01 02 01 01 08 02 13 04 01 08 09 07 0A 08 01 02 15 08 05 02 3A 08 23 02 01
01 01 02 02 01 01 02 10 04 01 08 04 02 31 08 07 04 01 02 01 01 08 02 18 04 01
08 08 07 0B 08 01 02 14 04 01 08 05 02 37 01 03 08 23 02 04 01 01 02 10 04 01
08 04 02 31 04 01 08 01 02 02 01 07 02 1E 04 01 08 08 07 0B 08 01 02 14 04 01
08 05 02 31 01 03 02 03 01 03 08 23 02 03 01 02 02 10 04 01 08 03 02 32 01 06
02 23 04 01 08 08 07 0C 02 14 04 01 08 05 02 31 01 09 08 23 01 01 02 02 01 01
02 11 08 04 02 5B 04 01 08 07 07 0C 08 01 02 14 04 01 08 05 02 2A 01 05 02 01
01 0A 08 23 01 01 02 02 01 01 02 11 08 04 02 5B 04 01 08 08 07 0B 08 01 02 14
04 01 08 05 02 2A 01 10 08 23 01 01 02 02 01 01 02 11 04 01 08 02 04 01 02 5B
04 01 08 09 07 0A 08 01 02 14 04 01 08 05 02 24 01 14 02 02 08 23 01 01 02 14
08 03 04 01 02 5B 04 01 08 08 07 0B 08 01 02 14 08 06 02 24 01 15 02 01 08 23
01 01 02 14 04 01 08 02 02 5C 08 09 07 0B 08 01 02 14 08 06 02 1D 01 04 02 01
01 13 02 02 01 03 08 23 02 13 04 02 08 03 02 5C 08 0A 07 0A 08 01 02 14 08 06
02 1D 01 11 02 09 01 03 08 23 02 12 08 05 04 01 02 5C 08 0A 07 0A 08 01 02 14
08 06 02 16 01 03 02 04 01 0F 02 04 01 01 02 01 01 08 08 23 02 12 08 05 02 5D
08 0A 07 0B 02 14 08 05 04 01 02 16 01 13 02 08 01 09 08 23 02 12 08 03 02 5F
08 08 07 0C 08 01 02 14 08 05 04 01 02 10 01 03 02 03 01 12 02 04 01 0E 08 23
02 13 01 03 02 5E 08 0A 07 0A 08 01 02 14 08 06 02 10 01 12 02 04 01 14 08 22
04 01 02 74 08 09 07 0B 08 01 02 14 08 05 04 01 02 09 01 17 02 04 01 01 02 02
01 13 08 22 04 01 02 74 08 08 07 0C 08 01 02 14 08 05 04 01 02 09 01 16 02 02
01 04 02 02 01 13 08 22 04 01 02 74 08 09 07 0B 08 01 02 14 08 05 04 01 02 02
01 05 02 01 01 11 02 04 01 1D 08 22 04 01 02 74 08 08 07 0B 08 02 02 14 08 05
04 01 02 02 01 13 02 03 01 02 02 01 01 1F 08 22 04 01 02 74 08 08 07 0B 08 02
02 14 08 06 01 11 02 04 01 25 08 22 04 01 02 74 08 09 07 0B 08 01 02 14 08 06
01 10 02 01 01 01 02 01 01 27 08 22 02 75 08 09 07 0A 08 02 02 10 01 03 02 02
08 05 04 01 01 0C 02 02 01 04 02 04 01 0C 02 01 04 01 02 01 01 04 02 01 01 0F
08 22 02 75 08 09 07 0B 08 01 02 10 01 05 08 05 02 01 01 08 02 02 01 08 02 02
01 0F 02 02 01 14 08 22 02 75 08 09 07 0B 08 01 02 0F 01 06 02 01 08 03 04 01
01 03 02 01 01 36 08 22 02 75 08 08 07 0C 08 01 02 0B 01 0B 04 01 08 02 01 0D
02 01 01 2D 08 22 02 75 08 09 07 0A 08 02 02 05 01 4F 08 22 02 75 08 09 07 07
08 05 02 01 01 13 02 02 01 3E 08 22 02 75 08 0A 07 06 08 02 07 01 08 02 01 4D
02 03 01 04 08 22 02 75 08 0A 07 05 08 03 07 01 08 02 01 27 02 01 01 20 02 01
04 03 08 2A 02 74 04 01 08 0A 07 05 08 03 07 02 01 04 02 03 01 09 02 02 01 30
02 02 04 02 08 31 02 74 04 01 08 0A 07 05 08 02 06 01 01 05 02 01 01 09 02 01
01 2D 02 02 04 02 08 38 02 74 04 01 08 0B 07 01 08 01 07 01 01 02 02 04 01 02
02 01 01 0A 02 01 01 27 02 01 04 02 08 0C 04 01 08 32 02 75 08 0A 01 09 02 01
01 2D 02 02 04 02 08 0C 04 02 08 0D 04 02 03 01 08 01 04 01 02 01 08 25 02 74
04 01 08 07 04 01 02 01 01 32 02 01 04 02 08 0E 04 01 08 0D 04 01 08 01 04 01
08 02 04 01 02 01 01 06 02 01 08 23 02 75 08 05 04 01 01 2E 02 01 04 02 08 0C
04 04 08 09 04 03 02 01 01 03 04 01 08 01 04 01 03 01 01 0D 04 01 08 22 02 75
08 03 02 02 01 28 02 02 04 02 08 0B 04 05 08 08 04 07 03 01 01 06 04 01 08 03
01 0C 07 01 08 23 02 77 01 13 02 01 01 04 02 01 01 0C 04 03 08 0C 04 04 08 08
04 01 03 01 02 02 04 01 03 01 04 01 02 03 01 0B 08 04 04 01 01 03 02 01 04 01
03 01 01 07 08 22 02 74 01 10 02 01 01 0F 02 01 04 03 08 0B 04 06 08 07 04 01
02 01 01 01 02 01 03 01 01 01 04 01 02 01 01 02 02 01 01 11 08 05 01 0D 08 22
02 6E 01 1F 02 02 04 02 08 0C 04 03 08 02 04 01 08 05 04 01 03 01 01 01 03 02
01 02 02 01 01 15 03 01 01 01 04 02 08 01 06 01 02 01 08 05 01 0D 08 22 02 6A
01 1C 02 01 04 02 08 0D 04 04 08 07 04 03 01 01 02 02 01 16 03 01 04 06 02 02
```

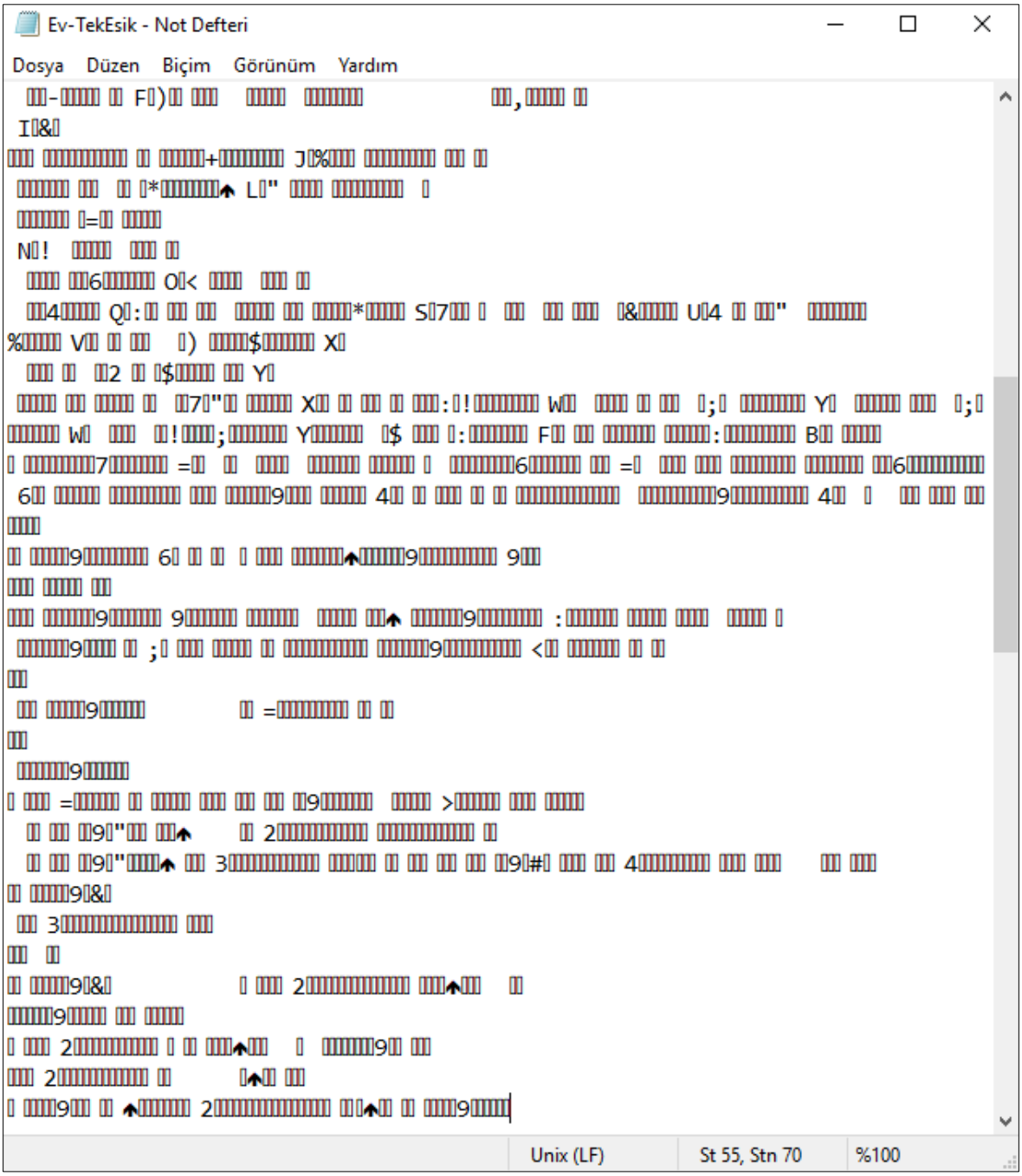
EK-1. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin byte dizisi görünümü

```
01 06 02 01 08 04 04 01 01 0C 08 21 04 01 02 68 01 17 02 01 04 02 08 0C 04 04
08 08 04 01 03 01 04 01 03 01 01 01 03 01 02 02 01 13 03 02 04 01 08 01 04 03
02 03 01 0E 08 04 04 01 01 0C 08 22 02 65 01 12 02 01 04 03 08 0B 04 06 08 08
04 02 01 01 04 01 01 02 03 01 02 01 01 11 02 01 04 04 08 02 04 01 01 18 08 05
04 01 01 0B 08 21 04 01 02 61 01 0F 02 01 04 02 08 0D 04 05 08 04 04 04 03 01
01 01 03 01 04 02 01 16 04 02 08 01 04 03 02 01 01 1F 08 01 04 01 08 02 04 02
01 0A 08 21 04 01 02 5D 01 0C 02 01 04 02 08 0C 04 05 08 06 04 02 03 01 04 01
03 04 04 01 01 15 04 02 02 01 04 01 02 02 01 28 02 01 04 01 08 02 04 02 01 09
08 21 02 5A 01 09 02 01 04 02 08 0C 04 03 08 09 04 03 02 01 03 01 01 01 02 01
01 15 02 02 04 02 02 02 01 33 04 01 08 02 02 01 04 03 01 05 08 21 02 5B 04 03
08 0D 04 03 08 01 04 01 08 05 04 06 01 01 02 01 03 01 01 14 04 01 03 01 04 01
01 42 04 06 01 02 08 21 02 55 04 02 08 0C 04 02 08 0B 04 06 02 01 03 01 01 15
04 01 02 03 01 48 02 01 04 03 01 01 02 01 08 21 02 54 04 01 08 06 04 04 08 09
04 04 02 01 01 01 02 02 01 01 02 01 01 14 02 03 01 53 02 02 04 01 08 21 02 55
04 05 08 08 04 04 03 01 01 18 02 01 01 01 02 01 01 5D 08 21 02 55 04 01 08 04
04 01 07 01 02 02 01 16 02 02 01 69 08 21 02 56 01 01 02 03 01 14 02 06 01 6B
08 20 02 5B 01 0C 02 01 01 02 02 02 01 02 02 01 01 01 02 03 01 6D 08 1C 05 01
01 03 02 5C 01 84 08 19 01 08 02 5D 01 82 08 15 06 01 01 0B 02 63 01 7C 08 12
06 01 01 0E 02 6C 01 73 08 0E 06 01 01 12 02 74 01 6B 08 0B 01 16 02 74 01 6B
08 08 01 19 02 74 01 5F 03 01 04 01 01 0A 08 05 01 1B 02 75 01 5F 08 02 01 0A
08 01 01 20 02 72 04 02 01 5F 02 01 08 01 01 28 02 01 04 01 02 73 08 02 01 86
02 01 04 03 08 01 04 01 02 72 08 02 01 83 02 01 04 01 08 06 04 01 02 72 08 02
01 7C 02 01 04 05 08 09 04 01 02 72 04 01 08 01 01 73 02 01 04 03 08 15 02 73
01 66 03 01 04 02 01 03 02 01 04 03 08 0B 04 02 02 01 01 02 08 0C 04 01 02 72
01 65 03 02 04 02 08 01 01 03 08 08 04 03 07 01 03 04 01 05 08 0B 04 01 02 73
01 63 03 06 01 03 04 03 08 01 07 01 03 03 01 02 05 01 01 0B 02 01 04 01 08 08
04 01 02 74 01 64 03 04 01 03 03 01 01 17 04 01 08 07 04 01 02 74 01 66 03 01
01 1D 08 07 04 01 02 74 01 61 03 02 01 27 02 76 01 61 03 01 01 27 04 02 02 75
01 88 02 01 08 02 02 75 01 88 04 01 08 02 02 75 01 68 03 02 01 1D 02 01 04 01
08 02 02 75 01 89 02 77 01 6A 03 01 01 20 02 75 01 6C 05 01 01 1E 02 75 01 8B
02 74 01 8C 02 75 01 69 03 02 01 20 02 74 01 8C 02 75 01 6B 08 09 04 01 03 02
01 13 02 75 01 6A 03 02 08 14 04 02 03 01 01 08 02 75 01 69 03 03 08 1F 02 75
01 68 03 01 01 02 03 01 08 1F 04 01 02 74 01 6C 08 1F 04 01 02 74 01 6B 03 01
08 1F 02 75 01 6B 04 01 08 1F 04 01 02 74 01 6B 04 01 08 1F 02 75 01 5B 03 01
01 0F 08 20 02 75 01 5A 03 03 01 0E
```

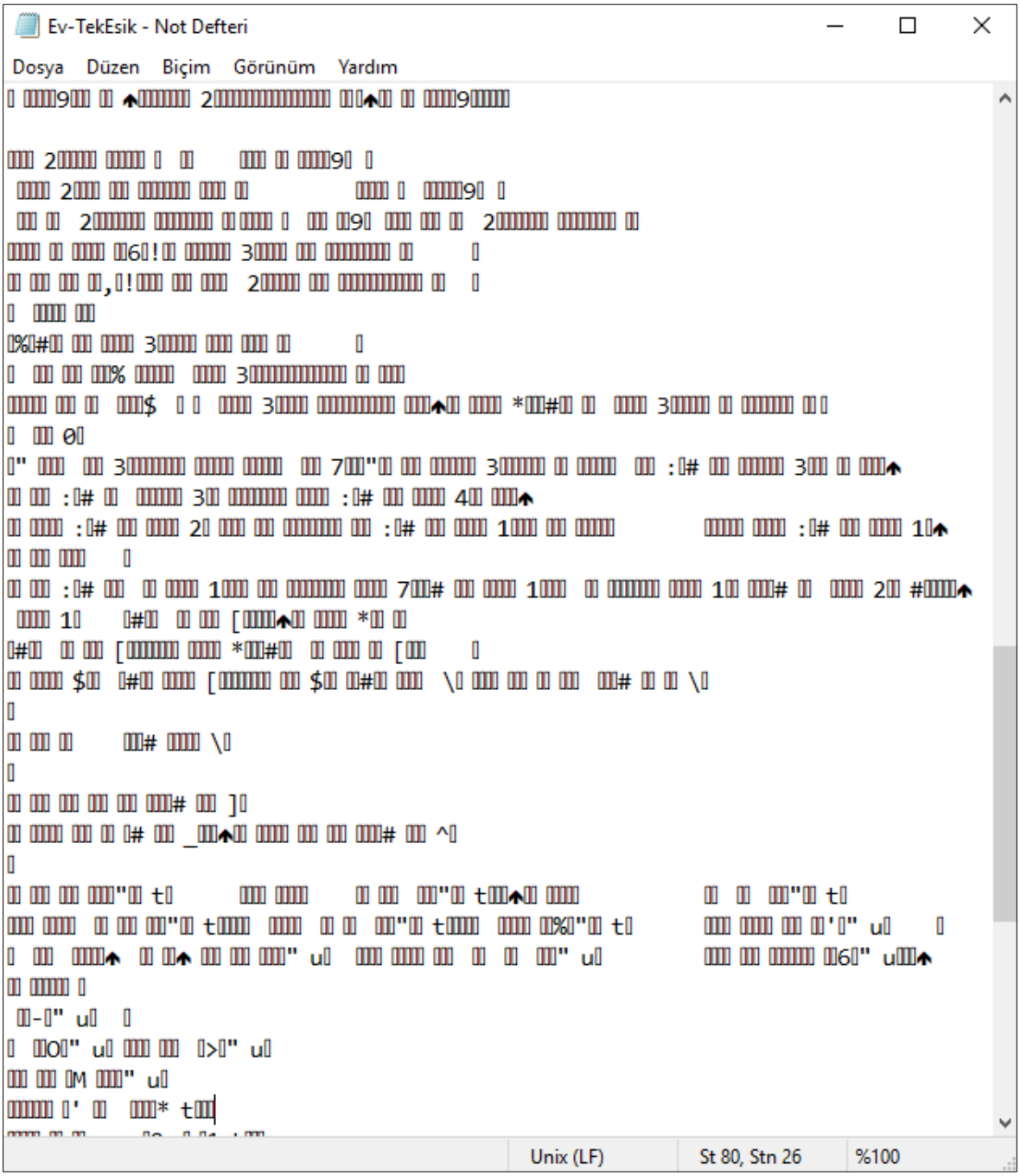
EK-2. Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin metin düzenleyicide ASCII biçiminde görünümü



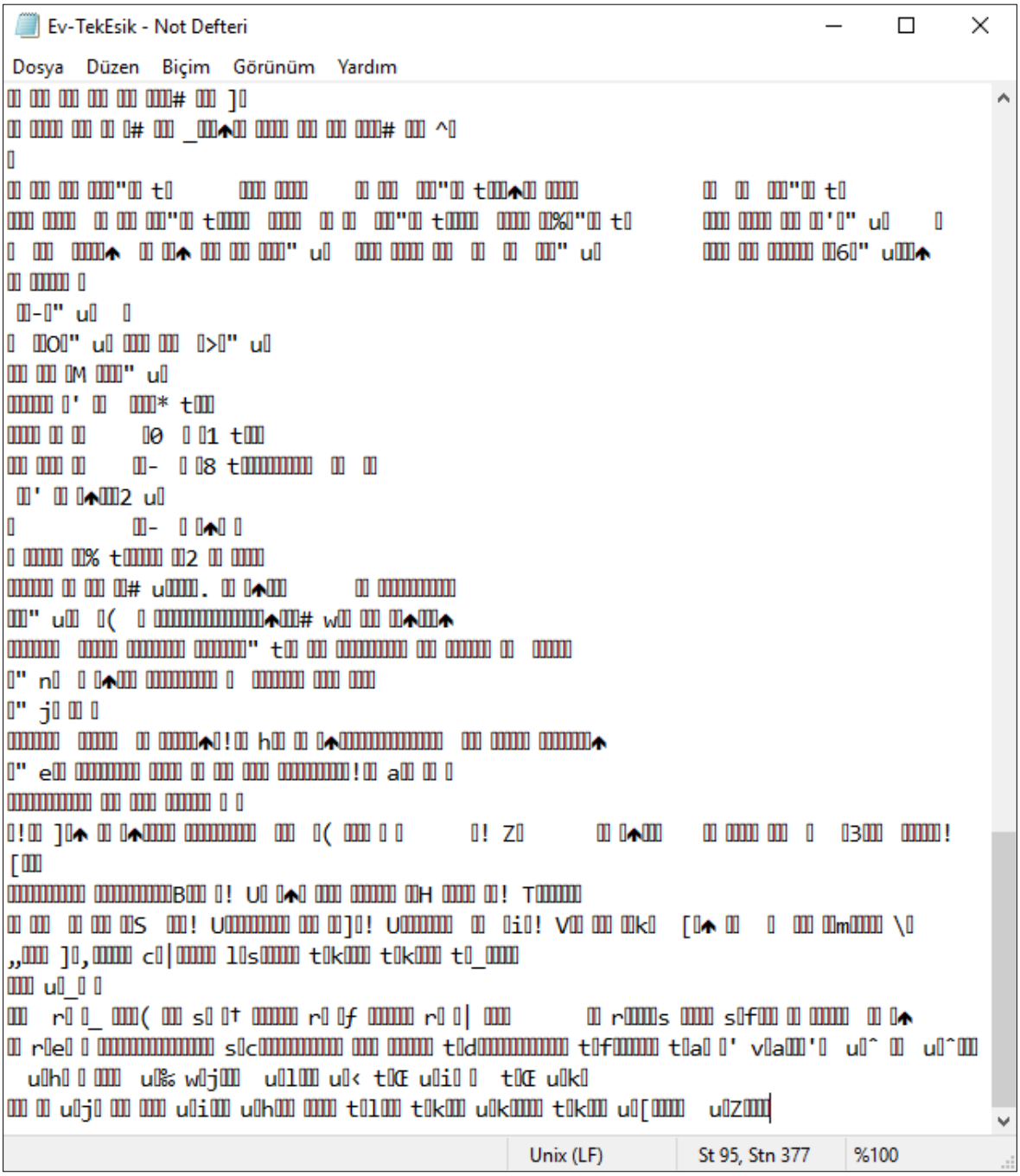
EK-2. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin metin düzenleyicide ASCII biçiminde görünümü



EK-2. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin metin düzenleyicide ASCII biçiminde görünümü



EK-2. (devam) Tek eşikli ayrıştırma ile sıkıştırılmış Ev görüntüsüne ait dosya içeriğinin metin düzenleyicide ASCII biçiminde görünümü



ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: KALKAN, Mustafa Burak
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 02.09.1987, Erzincan
Medeni hali	: Evli
Telefon	: 0 (544) 6121059
E-Posta	: mburakkalkan@gmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet Tarihi
Yüksek Lisans	Gazi Üniversitesi / Bilgisayar Mühendisliği	Devam ediyor
Lisans	Eskişehir Osmangazi Üniversitesi / Bilgisayar Mühendisliği	2015
Lisans	Marmara Üniversitesi / Bilgisayar ve Kontrol Öğretmenliği	2009
Lise	Ankara Polatlı Anadolu Lisesi	2005

İş Deneyimi

Yıl	Yer	Görev
2010 - Halen	Milli Eğitim Bakanlığı	Bilişim Teknolojileri Öğretmeni

Yabancı Dil

İngilizce

Yayınlar

1. Kalkan, M. B. ve Demirci, R. (2019). *Görüntülerin renk uzayı yardımıyla ayrıştırılması ve tekrarlı uzunluk kodlaması tabanlı görüntü sıkıştırma*. 3rd International Symposium on Innovative Approach in Scientific Studies, Ankara, 667-671.

Hobiler

Programlama, Hobi Elektroniği, Gezi, Müzik

GAZİ GELECEKTİR..