T.C.
HALİÇ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI
BİLGİSAYAR MÜHENDİSLİĞİ PROGRAMI

ÇOKLU ORTAMLARDA BİLGİ GİZLEME UYGULAMASI TASARIMI VE GELİŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Hazırlayan
Fahri ESKİÇIRAK

Danışmanı
Yrd. Doç. Dr. Oğuz KARAN

İstanbul – 2012

T.C.

HALİÇ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE

Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği Programı Tezli Yüksek Lisans öğrencisi **Fahri ESKİÇIRAK** tarafından hazırlanan **"Çoklu Ortamlarda Bilgi Gizleme Uygulaması Tasarımı ve Geliştirilmesi"** adlı bu çalışma jürimizce Yüksek Lisans Tezi olarak kabul edilmiştir.

Sınav Tarihi : 26.06.2012

(Jüri Üyesinin Ünvanı , Adı , Soyadı ve Kurumu) :	İmzası :
Jüri Üyesi: Yrd.Doç.Dr.Oğuz KARAN Dan.-HAL.Üniv.Bilgisayar Müh. ABD Öğr.Üyesi	.....
Jüri Üyesi : Yrd.Doç.Dr.Ulviye HACIYEVA HAL.Üniv. Bilgisayar Müh. ABD Öğr.Üyesi	.....
Jüri Üyesi :Yrd.Doç.Dr.Tayfun YEGÜL HAL.Üniv. Matematik ABD Öğr.Üyesi	.....
Jüri Üyesi :Prof.Dr.Mübariz EMİNLİ HAL.Üniv. Bilgisayar Müh.ABD Öğr.Üyesi (Yedek)	.....
Jüri Üyesi : Yrd.Doç.Dr. Murat BEKEN HAL.Üniv. End.Müh. ABD Öğr.Üyesi (Yedek)	.....

ÖNSÖZ

İnsanlık, var olduğundan beri bilgiyi, bulunduğu zamanın imkanlarına göre geliştirdiği yöntemler ile gizlemeye çalışmıştır. Günümüzde ise artık hayatımızın bir parçası olan bilgisayar tabanlı teknolojiler bilgiyi gizleme yolunun en önemli araçlarıdır. Bilgiyi gizlemek artık insan eliyle değil, makina diliyle olmaktadır. Bu tezde bilginin bir formu olan çoklu ortamı kullanabilen, popüler çoklu ortam elamanlarını birbirleri içinde gizleyebilecek, gizlenen bilgiyi çıkarabilecek bir uygulama tasarımı ve geliştirilmesi yapılmıştır.

Bu hedefe giderken yüksek lisans eğitimimde ve tez çalışmamın tamamlanmasında büyük bir gayret ve özveriyle çalışmamı takip eden, gösterdiği sabır ve hoşgörüyle bana destek olan tez danışmanım Sayın Yrd. Doç. Dr. Oğuz Karan'a teşekkür ederim. Bu uzun soluklu tez çalışmam boyunca yardımlarını esirgemeyen Sayın Arş.Gör. Ayşe Merve Gün'e ve değerli arkadaşlarım Ayşe Taşpınar, Ozan Selim, Selçuk Yaşa, Pınar Tekdal, Ali Ekşioğlu, Cihan Akcan ve Serkan Türkkan'a teşekkür ederim.

Son olarak eğitim hayatım boyunca bana destek olan ve verdiğim her kararın arkasında durarak beni bu günlere getiren sevgili anne ve babama sonsuz teşekkür ederim.

İstanbul, 2012 Fahri ESKİÇIRAK

İÇİNDEKİLER

Sayfa No.

KISALTMALAR

LSB : Least Significat Bit.

UDP : User Datagram Protocol

CD : Compact Disc

DVD : Digital Versatile Disc

Mp3 : Mpeg-1 Audio Layer III

Mp4 : Mpeg-4

PDA : Personal Digital Assistant

JPEG : Joint Photographic Experts Group

JFIF : JPEG File Interchange Format

JPEG-LS : Lossless JPEG

ASCII : American Standard Code for Information Interchange

UDP : User Datagram Protocol

TCP : Transmission Control Protocol

DCT : Discrete Cosine Transform

DFT : Discrete Fourier Transform

DWT : Discrete Wavelet Transform

PCM : Pulse Code Modulation

WAV : Windows Audio-Visual

TIFF : Tagged Image File Format

BMP : Windows Bitmap

RGB : Red Green Blue

RGBA : Red Green Blue Alpha

RAR : Roshal Archive

ISO : International Organization for Standardization

IEC : International Electrotechnical Commision

AVI : Audio Video Interleave

FourCC : four-character code

DivX : DivXNetworks, Video codec.

WMA : Windows Media Audio format.

RDI : Device independent bitmap file

RMI : Midi information

RIFF : Resouce Interchange File Format

Hdrl **:** Header List

Strl : Stream List

Strh : Stream Header

Strf : Stream Format

Strd : Stream Header Data

Strn : Stream Name

Strf : Stream Format

Avih : Avi Header

Db : Uncompressed video frame

Dc : Compressed video frame

Pc : Palette change

Wb : Audio data

Idx1 : Index 1

AAC : Advanced Audio Coding

VBR : Variable Bitrate

DES : Data Encryption Standard

RSA : Public-key encryption method.

LCD : Liquid Crystal Display

LED : Light Emitting Diode

API : Application Program Interface

ŞEKİLLER

Sayfa No.

TABLOLAR

Sayfa No.

GENEL BİLGİLER

Adı ve Soyadı : Fahri ESKİÇIRAK
Anabilim Dalı : Bilgisayar Mühendisliği
Programı : Bilgisayar Mühendisliği
Tez Danışmanı : Yrd. Doç. Dr. Oğuz KARAN
Tez Türü ve Tarihi : Yüksek Lisans – Haziran 2012

ÇOKLU ORTAMLARDA BİLGİ GİZLEME UYGULAMASI TASARIMI VE GELİŞTİRİLMESİ

ÖZET

Bu çalışmada yaygın olarak kullanılan son iki bite veri gizleme ve çıkarma olarak adlandırılan bir steganografik yöntem ile çoklu ortam kapsamında tanımlanan farklı dosya türlerinin birbirleri içinde gizlenebileceği bir uygulama tasarlanmış ve geliştirilmiştir. Uygulama için mümkün olduğunca kullanıcı dostu grafiksel bir arayüz tasarlanmıştır. Uygulamanın geliştirilebilmesi için platform bağımsız olan Net framework tercih edilmiştir. Böylece uygulama sadece Microsoft Windows yüklü sistemlerde değil, .Net framework'unun kurulu olduğu tüm platformlarda çalışabilmektedir. Uygulamanın çalışması süresince bellekte tutulan, içine veri gizlenecek veya çıkarılacak olan bitmap yapıdaki görüntü piksellerinin üzerindeki işlemlerin kontrolü için büyük önem taşıyan ve onların mantıksal karşılığı olan iki boyutlu, video ve ses dosyaları için tek boyutlu liste tipinde tablolar ve haritalar tasarlanmıştır. Formen gibi davranan bu dinamik yapılar, uygulama döngülerinin içinde ya da dışında iş akışına göre güncellenmektedir. Böylece üzerinde bilgi gizleyecek dosya türünün hangi kısımlarında bilgi gizlenebileceği tespit edilebilmektedir. Böylece çalışma süresince yine bellekte yaşayan, görüntü dosyalarına gizlenecek verilerin koordinatları gibi değerleri tutması için tasarlanmış tablolara, bu kısımlar sayısal olarak depolanmaktadır. Depolanmış sayısal veriler okunarak, bilgi gizleyecek tür üzerindeki bölgelere, gizlenecek tür bilgilerini yaymaktadır. Uygulanan bu yöntem, çoklu ortam türleri arasında gizleme ve çıkarmanın her safhasında başarımı test edilmiştir.

Anahtar Kelimeler : Çoklu ortam, steganografi, bilgi gizleme.

GENERAL INFORMATION

Name and Surname : Fahri ESKİÇIRAK

Field : Computer Engineering

Program : Computer Engineering

Supervisor : Assist. Prof.Dr. Oğuz KARAN

Degree Awarded and Date : Master of Science – June 2012

DESIGN AND IMPLEMENTATION OF INFORMATION HIDING APPLICATION IN MULTI-MEDIA

SUMMARY

In this study, two bits of data are used in hiding and extracting with a so-called multimedia steganographic method. In each of the different types of files an application designed and developed may be hidden. For the possibility of application, a user friendly graphical interface is designed. Net framework, which was chosen to develop thus application is platform independant. and not only installed for Microsoft Windows systems. Net framework runs on all platforms installed. Composition of image pixels into the bitmap data is kept or to be issued for the control of great importance. Their logical operations on the corresponding two dimensional, video and audio files; charts and maps, are designed for the type one dimensional list and are all held in memory during program execution. The application is updated according to the workflow inside or outside the loops thus behaving as a dynamic structure for a Foreman. Therefore, this system not only can be taught to hide information but can also determine which file to hide the information.During the study, living memory,image files and data are valued as coordinates and are all stored digitally. Such information is kept private and radiates. This method of hiding and extracting of multiple media types has been successfully tested at every stage of performance.

Keywords: Multi-media, steganography, information hiding.

1. GİRİŞ

Bilgi gizleme (information hiding), özellikle son yıllarda üzerinde çok fazla çalışılan ve oldukça ilgi gören bilimsel faaliyetlerin bir bütünüdür. 1990 yılından itibaren, çok az sayıda yayın yapılan bu konu üzerindeki çalışmaların yoğunluğunun artışına sebep olarak birçok detay verilebilir. Özellikle internet, bu yükselişteki en önemli etkendir. Bu etken internet'in çok büyük oranda, 90'lı yılların ikinci yarısında, tüm Dünya üzerinde hızla yaygınlaşması ve buna bağlı olarak iletişim teknolojilerinin hayatın her noktasında kullanılmasıdır (Uslu, 2003:6).

Bilgiyi bir yerden başka bir yere taşıyan, bilgisayar tabanlı sistemlerin teknolojilerinin giderek gelişmesi, yaşamın her alanında çok faydalı olmuştur. Bununla birlikte taşınacak bilginin güvenliğini sağlamak için, yeni yöntemlerin geliştirilmesi ve bilgisayar tabanlı sistemler ile birleştirilmesi gerekmektedir. Çünkü mevcut güvenlik yöntemlerini aşarak gizli bilgiye ulaşmaya çalışan teknolojilerde geliştirilmektedir. Böylece bilginin korunması için zamanın şartlarına bağlı olarak farklı yaklaşımları içeren yöntemler geliştirilmiştir ve geliştirilmeye de devam edilecektir (Oral ve Furat,2007).

Bu yöntemlerin bilgisayarlar tarafından kullanılabilecek şekilde tasarlanmasının ve geliştirilmesi hedefinin, bir birey tarafından sadece kişisel bilgileri koruyan basit uygulamaların oluşturulması olarak düşünülürken, ülkelerin gelecekleri için çok büyük önem arz eden savunma sistemlerinin yönetimi gibi özel görevleri yerine getiren birimler/kurumlar için çok fazla önem verilmesi gereken karmaşık teknolojiler olarak görülmektedir (Gürel, 2006:2).

1.1. Tezin Amacı

Bu çalışmanın amacı yaygın olarak kullanılan son iki bit'e veri gizleme ve çıkarma olarak adlandırılan bir steganografik yöntem ile çoklu ortam kapsamında tanımlanan farklı dosya türlerinin birbirleri içinde gizlenebileceği bir uygulama tasarlamak ve geliştirmektir.

1.2. Tezin Yapısı

Bu tez beş bölümden oluşmaktadır. İlk bölümde tezin konusuna genel bir giriş yaptıktan sonra, ikinci bölümde literatürde yapılan bazı çalışmalara değinilmiştir. Üçüncü bölümde ise uygulamada kullanılan teknolojiler çoklu ortam, bilgi gizleme, steganografi, steganografinin alt dalları ve uygulamada kullanılan görüntü, ses ve diğer dosya türleri tanımlanarak açıklanmaktadır.

Dördüncü bölümde MMHIDE uygulamasının tasarımında ve geliştirilmesinde kullanılan yöntemler detayları ile anlatılmaktadır.

Beşinci bölümde ise tasarlanmış ve geliştirilmiş uygulamadan elde edilen sonuçlar genel olarak tartışılmıştır.

2. LİTERATÜRDE YAPILAN ÇALIŞMALAR

İnternet ve sayısal bilgi devriminin hızla gelişmesi, Dünyada önemli değişikliklere sebep olmuştur. Bu gelişim, esnek ve kullanımı kolay yazılım ve sayısal aygıtların fiyatlarının düşmesine (taşınabilir CD ve mp3 oynatıcılar, DVD oynatıcılar, CD ve DVD kayıt ediciler, laptop'lar, PDA'lar) böylece çoklu ortam verilerinin oluşturulması, düzenlenmesi ve değiş tokuşunun yapılmasını kolaylaştırmıştır. Ayrıca verinin hatasız taşınmasında geniş bant internet bağlantısı kulllanımı çoklu ortam dosyalarının dağıtımında ve onların sayısal kopyalarının tamamen aynısının oluşturulmasında yardımcı olmaktadır. Güvenlik sistemleri ne kadar geliştirilse bile internet üzerinden önemli mesajlar ve dosyalar güvenli olmayan bir formda taşınmaktadır. Tüm bunlar çoklu ortam'larda bilgi gizleme konusunu son derece önemli hale getirmiştir.

2.1. Görüntü Dosyalarına Bilgi Gizleme Konusundaki Bazı Çalışmalar

Uslu (2003:5), genel veri gizleme yöntemlerinden olan Tip I ve Tip II yöntemlerini karşılaştırarak kullanım alanlarını detaylı olarak incelemiş, jpeg algoritması, ve bu algoritma ile birlikte kullanılan diğer algoritma ve yöntemleri ele almıştır .

Gürel (2006:9), veriyi gizlerken LSB, iki bit ve bir piksel içerisine bir ASCII kodun gömülmesi yöntemleri kullanarak resim üzerinde gözle görülür bir bozulma oluşmadan, UDP/TCP protokollerinden yararlanarak kablosuz ortamda istenen hedefe iletmiş ve bant genişliğinin etkin bir şekilde kullanılmasını hedeflemiştir. Şahin, Buluş ve Sakallı (2006), 24-bit renkli resim dosyaları üzerinde en önemsiz bite ekleme yöntemi geliştirerek bir örnek steganografi uygulaması gerçekleştirmiştir. Oğuz (2006:9), taşıyıcı görüntüye bilgi damgalamak için dalgacık dönüşümü yöntemi kullanmıştır. Bu yöntem ile görüntüye görünmez ve dayanıklı damga damgalamıştır. Damganın farklı gürültü türlerine karşılık ve değişik görüntü işleme saldırılarına karşılık (kesme, döndürme, ölçekleme) dayanıklılığı incelemiştir.

Furat (2006:3), taşıyıcı görüntülere bir bilgi damgalamak için yeni bir yöntem önermiştir. Bu yöntem ile görüntüye görünmez ve dayanıklı filigran damgalamıştır. Damgalama işleminde, uzay ve frekans düzlemi bileşenlerinin bir arada ve damgalama DCT (Ayrık kosinüs dönüşümü) ile elde edilen frekans bileşenlerine yaparken, referans noktası olarak uzay düzleminden elde edilen enerji kullanmıştır. Damgalama işleminden önce filigrana uygulanan permutasyon ile filigranın görüntü işleme saldırılarına karşı dayanıklılığı bir kat daha arttırmıştır. Esin ve Güvenoğlu (2006), resmin içine yazı gizlenmesi amacıyla kullanılan LSB ekleme yönteminin shuffle algoritması ile iyileştirilmesi yoluyla kullanımı kolay ve etkin bir yöntem elde etmiştir. Öztürk (2009:36), Video ile güvenli veri aktarımı hakkındaki lisans bitirme çalışmasında DES şifreleme yöntemi, RSA açık anahtarlı şifreleme yöntemi ve özel bir yöntem olan, görüntüye veriyi belirli bir tohum değerinden ortaya çıkan sıra ile gömme yöntemi kullanarak bir video dosyasının çerçevelerine veri gizleyerek iki nokta arasında güvenli bir haberleşme ve veri aktarım kanalı ve bu kanalı kullanıcısına sağlayan bir uygulama tasarımı gerçekleştirmiştir.

2.2. Ses Dosyalarına Veri Gizleme Akademik Çalışmalar

Atıcı (2007:3), steganografik yaklaşımların incelenmesi, tasarımı ve geliştirilmesi üzerine yapmış olduğu çalışmasında, ses içerisine veri saklama analizleri yapmış, kişisel bilgi güvenliğinin sağlanmasında kullanılabilicek PCM formatındaki Wav dosya seti içerisine klasör saklayan bir Türkçe steganografik yazılım geliştirmiştir.

Akbal (2008:3), ses verilerine sıkıştırılmış ve şifrelenmiş ham verilerin gömülmesi üzerine yaptığı yüksek lisans tezinde, sayısal ses içerisinde gönderilecek verinin daha güvenli ve hızlı iletimi için şifreleme ve sıkıştırma yöntemleri kullanarak gizli veri transferinin kablosuz ortamda gerçekleştirmişti.

Dutta, Bhattacharyya ve Kim (2009:1), ses sinyaline veri gizleme üzerine yapmış oldukları önbakış bildirimlerinde, sayısal ses'e gizli bilgi gizlemek için genellikle hangi süreçlerin izlenmesi gerektiğini ve bu bağlamda kullanılacak farklı tekniklere ve fonksiyonlarda incelemişlerdir.

3. KULLANILAN TEKNOLOJİLER

3.1. Çoklu Ortam

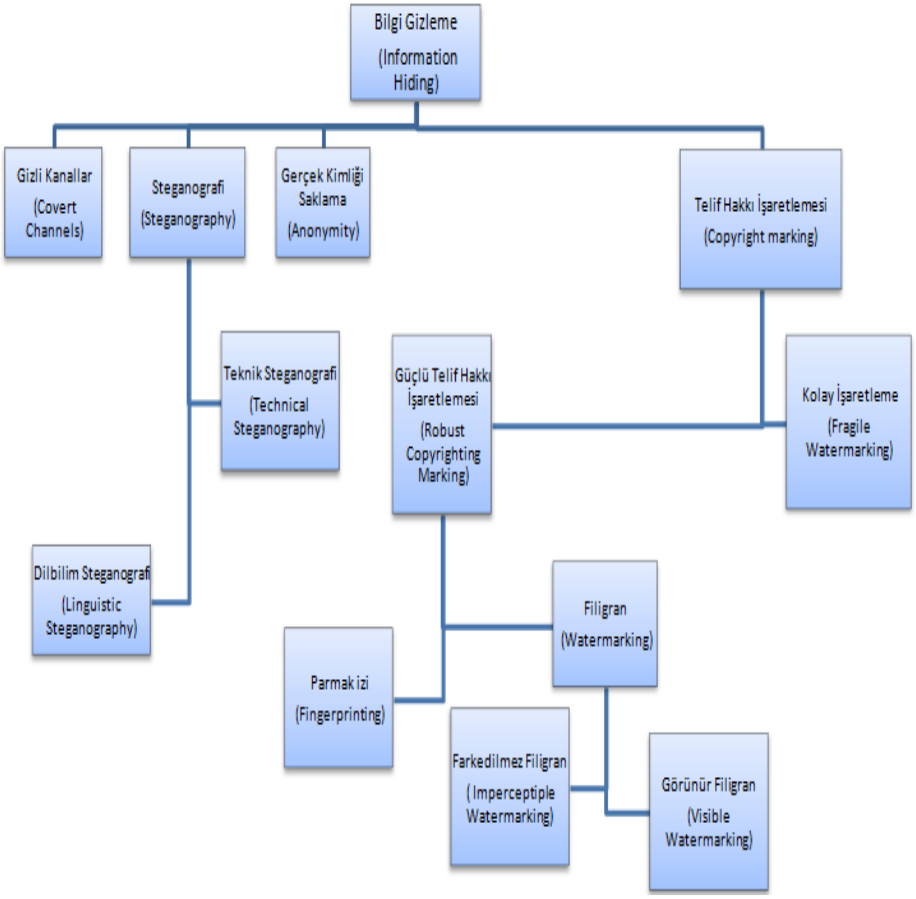
Çoklu Ortam'ın (Multi-medya) belirli bir tarifini yapmak ve sınırlarını belirlemek zor olmakla birlikte, bir dönem müze ve galerilerde kullanılan sunular, video, ses kasetleri ve günümüzde ise konuşan kitaplar, internet sayfaları özellikle de bilgisayar yazılımlarını içine alan oldukça kapsamlı ve teknolojinin gelişmesine paralel olarak sürekli bir değişim içinde olan bir iletişim şekli (Aslı, Maden ve Durukan, 2010) ya da metin, görüntü, grafik, çizim, ses, video ve animasyon türündeki bilgi kaynaklarının bir araya getirilmesi ile oluşan ortamlar olarak tanımlanabilmektedir (Wikipedia, 2012 g).

3.2. Bilgi Gizleme

Bilgi gizleme, veritabanında bulunan bilgilere yetkisiz kullanıcıların erişiminin kısıtlanması, bilginin başka bir bilginin içine gizlenmesi veya bir yerden başka bir yere taşınması gibi birçok tanımı içeren bir kavramdır (Ertürkler, 2007:12). Bu kavrama göre bilgi, bir mesaj, gönderici veya alıcı kimlikleri gibi korunmak istenen herhangi birşey olabilir. Koruma, istenmeyen kişiler tarafından bilginin okunmasının ya da yok edilmesini engelleyecek şekilde gerekli tedbirlerin alınması anlamına gelmektedir. Şekil 3.1'de görüleceği gibi bilgi gizleme için geliştirilen çeşitli yöntemler, bu konu hakkındaki ilk çalıştay ile üzerinde çalışılarak bir sınıflandırmaya tabi tutulmuştur (Şahin, 2007:3).

Gizli Kanallar, gizli bilgilerin iki kişi arasında el değiştirmesi için kullanılan bir iletişim kanalıdır. Hedefi, iletişimin amacının ve taşınan verinin saklanmasıdır. Gerçek kimliği saklama, bilginin bilinmeyen veya anlaşılmayan biri üzerinden gönderilerek, göndericinin gerçek kimliğini saklamaktır (Şahin, 2007:3).

Telif hakkı işaretlemesinde ise orijinal dosyanın çeşitli yöntemler ile telif hakkı sahibi, üretim tarihi, üretici iletişim adresi gibi bilgilerini içine gizlemeyi amaçlayan sayısal filigranların sayısal görüntülerde korunması üzerine yapılan çalışmalardır (Şahin, 2007:4).



Şekil 3.1. Bilgi gizleme yöntemlerinin sınıflandırılmasına bir örnek (Şahin,2007:3).

Steganografi, gizlenecek veriyi başka bir forma dönüştürmekten ziyade, farklı veri türlerinin içine gizler ve böylece gözlemci gizli verinin varlığını algılayamaz (Reddy, Subramanyam ve Reddy, 2011). Yani steganografinin amacı, bir bilginin gizliliğinin diğer kişiler tarafından belirlenebilmesine engel olmaktır. Bir steganografik yöntem, bilginin gizlendiği taşıyıcı ortamda şüphe yaratırsa, bu yöntemin tam olarak başarılı olduğu söylenemez (Alçı ve Çivicioğlu, 2003). Steganografik yöntemlerin ticari uygulamaları dijital damgalama olarak tanımlanmaktadır. Örneğin bilgisayar ortamındaki çalışmalara telif hakkı, lisans, logo vb. bilgiler damgalanır (Gürel, 2006:2).

Damgalama hedef, bir ses ya da görüntü dosyasının sahibini belirtmek, steganografideki gibi algılanamaz olmak ve damgalama işaretini kaldırma veya değiştirmeye yönelik girişimlere karşı güçlü olmaktır (Gürel, 2006:2). Bununla birlikte steganografi uygulamalarında, damgalamadaki gibi çeşitli saldırılara karşı daynanıklılık beklentisi mevcut değildir (Oral ve Furat, 2007).

Etiketleme olarak da adlandırılan parmak izi ekleme, benzer sayısal ürünleri birbirinden ayırmak amacıyla kullanılan yöntemlerdendir. Böylece ürün sahibini temsil eden bilgi görünmez şekilde damgalanır ve telif hakkı korunur (Oral ve Furat, 2007). Dış saldırılara karşı dayanıklığı olan bu bilgi, sayısal ürünlerin izinsiz kopyalama işlemlerinin oldukça basitleştiği günümüzde, ürünün gerçek sahibinin hukuksal takibinde oldukça önemli avantajlar sağlamaktadır. Bununla birlikte parmak izi ekleme, amaç bakımından sayısal damgalamadan farklı olmakla birlikte yöntem olarak sayısal damgalama ile aynı yapıya sahiptir (Oral ve Furat,2007).

Veri gizleme literatüründe Kriptoloji'de bulunmaktadır. Gizlenmesi gereken bilgi, bir algoritma kullanılarak şifrelenmekte ve bilginin kendisi steganografide olduğu gibi herhangi bir taşıyıcı kullanılmadan alıcıya iletilmektedir. Alıcı, aldığı bilgiyi kendisinde bulunan deşifre edecek mekanizmalar ile gönderilen gizli bilgiyi görebilecektir. Böylece, sayısal bilgiyi yetkisiz olarak elde eden kişiler deşifre algoritmasını bilmedikçe bilginin aslını elde edemeyeceklerdir (Oğuz, 2006:2).

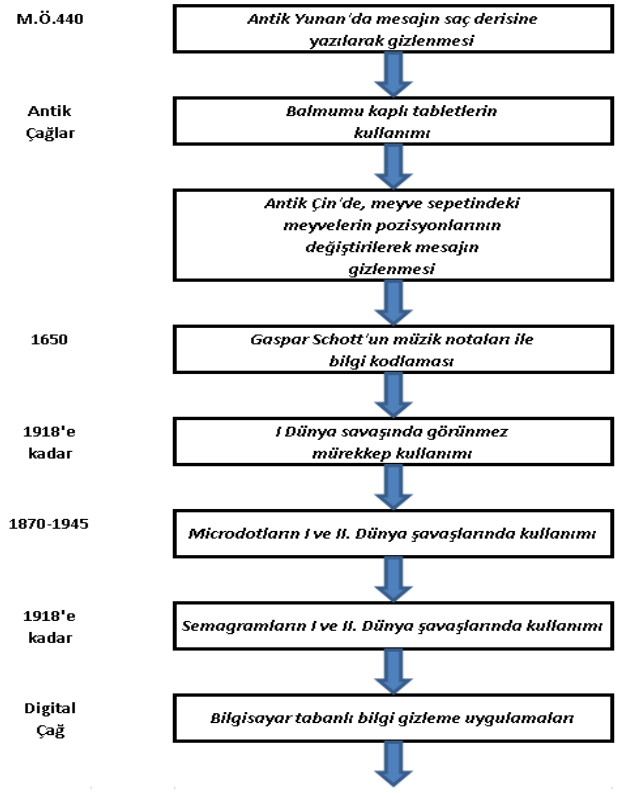
3.3. Steganografi ve Steganografinin Alt Alanları

Eski bir bilgi gizleme sanatı olan Steganografi kelimesi kökeni Yunanca da kullanılan "στεγαυος" ve "γραΦειν"'den kelimelerinden gelir. Tam olarak karşılığı "kaplanmış yazı"'dır (Şahin, 2007:7).

Yöntem olarak Steganografi, kriptografi ile benzerlikler göstermesine rağmen, önemli farklılıklarda içermektedir. Kriptografide amaç, bilginin içeriğinin saklamak iken steganografi de ise bilginin varlığını saklamaktır. Kriptografi işleminde veriyi gizlemek amacıyla farklı kriptografik yöntemler kullanır. Steganografide ise verinin saklandığı hiçbir şekilde anlaşılmamalıdır. İki yöntem, önce gizli mesajın şifrelenmesi ve sonra da steganografik yöntemlerle saklanması şeklinde birleştirilebilir. Gizlenmek istenen bilgi, şifrelenerek saklandığında gözlemcinin dikkatini çekebilirken, çoklu ortam dosyalarının içine saklandığında farkedilmez olacak ve kırılmaya çalışılmayacaktır (Gürel,2006:3).

Tarih boyunca insanlar ihtiyaçları doğrultusunda Şekil 3.2.'de görüldüğü gibi farklı metodlar kullanarak steganografi bilimini geliştirmişlerdir. Örneğin eski yunanlılar tabletlere yazdıkları yazının üstünü balmumu kapatarak gizlemişlerdir, gönderilecek gizli bilgiyi elçilerin kafasına yazmışlar, ulaşacakları yere geldiklerinde saçlarını kazıyarak gizli mesajı okumuşlardır. Yine görünmez mürekkeplerle mesaj iletimi başka bir steganografi uygulaması olarak karşımıza çıkmaktadır. Bu türden veri iletimi özellikle II. Dünya Savaşı öncesinde yaygın olarak kullanılmıştır (Öksüzoğlu, 2009).

Taşıyıcılar, bir diğer adıyla örtü veriler steganografinin temel taşlarıdır. Örtü nesnesine göre steganografi genel olarak yazı, ses, görüntü (imge) ve protokol steganografisi olarak sınıflandırılabilir. Bu sınıflandırmada, yazı steganografisi anlamsal, yazımsal, açık alan ve ses steganografisi LSB kodlama, faz kodlama ve eko kodlama yöntemleri olarak alt kategorilere ayrılabilir (Kurtuldu, 2008:4). Protokol steganografisi, bilgisayar içerisinde protokol katmanları ve boot sektörü gibi boş alanlardan, cep telefonlarında kullanılan SMS'ler ve internette kullanılan web alanlarına kadar farklı kullanım alanlarına sahiptir (Kurtuldu, 2008:4).



Şekil 3.2. Steganografinin tarihsel gelişimi (Şahin, 2007:9).

Üzerinde çok fazla çalışılan görüntü steganografisi ise LSB tabanlı, damgalama (maskeleme ve filtreleme), DCT, Ayrık Frourer Dönüşümü (DFT) ve Ayrık Dalgacık Dönüşümü (DWT) gibi farklı teknikleri içeren dönüşüm uzayı tabanlı steganografi (transform domain based steganography) kapsamındadır (Kurtuldu, 2008:4). Bu kapsam, son iki bit'e gizleme, bir piksel içerisine bir ASCII kodunu gizleme , R kodlama ağırlığının değiştirilmesi, kısmi optimizasyon ile veri gizleme, logaritma kullanılarak rastgele LSB ekleme, iz düşürme yöntemi (Mapping Metod), frekans alanına gizleme yöntemi (Frequency Domain Embedding), sıçramalı tayfa gizleme yöntemi (Spread Spectrum Embedding) gibi farklı yöntemleri içerir (Şahin, Buluş ve Sakallı, 2006).

3.4. Görüntü, Ses ve Diğer Dosya Türleri

Dosya türü, bilgisayalardaki standart depolanma yoludur ve bir program tarafından algılanabilmesi için gerekli bir yapıdır. Uygulamanın tasarımında ve geliştirilmesinde günümüzde en popüler olan çoklu ortam dosya türlerine yer verilmiştir.

3.4.1. Görüntü Dosya Türleri

En popüler görüntü dosya türleri arasında JPEG, TIFF ve BMP bulunmaktadır. Ayrıca bazı dijital fotoğraf makinalarında sıkça kullanılan RAW dosya türleri de güncelliğini korumaktadır ve çeşitli fotoğraf programlarında açılıp düzenlenebilmektedirler. Bu popüler resim dosya türleri arasından uygulamanın destekleyeceği biçim olarak JPEG seçilmiştir, bunun ana nedeni internet kullanıcıları arasında en sık kullanılan dosya biçimi olması, diğer resim dosya biçimlerinden çok daha az yer kaplaması, görüntüleme programlarının büyük çoğunluğu tarafından desteklenmesidir (Reddy, Subramanyam ve Reddy, 2011).

3.4.1.1. Jpeg

Jpeg olarak bildiğimiz dosya biçimi, Independent JPEG Group adlı başka bir grubun JFIF (JPEG Dosya Alışveriş Biçimi) adlı standardı tarafından tanımlanmıştır (Wikipedia, 2012 h). Aslında JPEG, bir dosya formatından daha öte veri üzerinde bir takım sıkıştırma yöntemleri kullanarak onun boyutunu düşürmektedir (Erişir, 2012).

Böylece Jpeg standardı zamanla geniş bir uygulama alanı bulmuştur fakat bazı sorunlarıda beraberinde getirmiştir (Göçeri ve Yaldır, 2007). Çözüm olarak jpeg farklı veri sıkıştırma yöntemlerine göre alt sınıflara ayrılmış ve bu düzende geliştirilmeye çalışılmıştır. Veri sıkıştırma yöntemlerinin kullanımı verilerin daha az yer kaplaması ve iletişim ağları üzerinden daha hızlı transferini hedefler. Bu noktada sıkıştırma yöntemleri kayıpsız ve kayıplı sıkıştırma olarak ikiye ayrılabilir. Kayıpsız sıkıştırmada orijinal veri sıkıştırma işleminden sonra istendiğinde bütünüyle elde edilebilir (Yeşilyurt ve diğ., 2012).

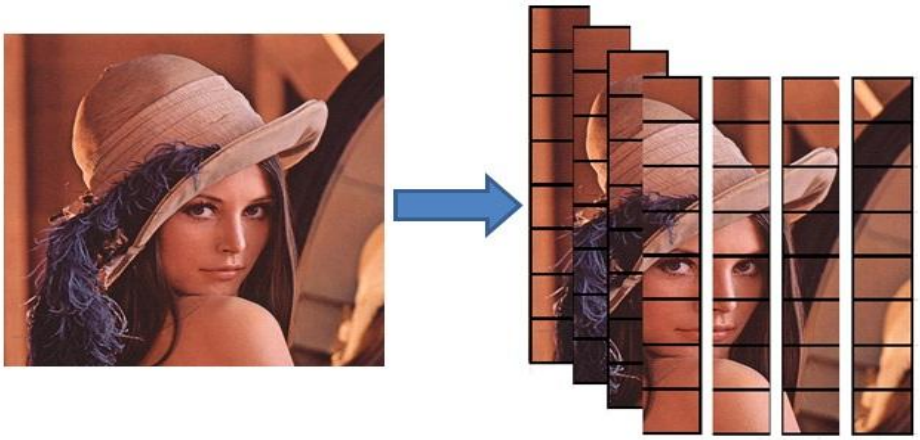
Kayıplı sıkıştırmada ise orijinal veride kayıplar meydana gelmektedir bununla birlikte verinin kapladığı alan azalmakta böylece iletimde daha az band genişliğine ihtiyaç duyulmaktadır (Yeşilyurt ve diğ., 2012).

Artan gereksinimler, Ricoh Innovations firması tarafından CREW algoritmasınının, Jpeg-Ls içinde kullanımının kabul edilmesi ve sonra JPEG2000 çalışma grubunun mevcut standardın hatalarını, eksikliklerini ortaya koyarak yeni bir standart geliştirilmeye başlamasına neden olmuştur. ISO/IEC tarafından geliştirilen, görüntüleri kayıpsız veya az kayıplı sıkıştırabilen Jpeg-Ls, 1998 yılının sonlarında tamamlanmıştır (Mesut ve Carus, 2005). Jpeg-ls, JPEG2000'in geliştirilmesine paralel olarak yaygın bir kullanıma sahip olamamıştır. JPEG2000, DWT teknolojisini temel alarak, bilinen en iyi sıkıştırma yöntemlerinin kullanılmasıyla meydana getirilmiş bir kodlama sistemidir (Hamza, 2008: 84). DWT son zamanlarda sıklıkla kullanılan bir yöntemdir. En geniş uygulama alanını resim sıkıştırmada bulmuştur (Özdemir ve Artıklar, 2009). Bu işlem sinyallerin ve görüntülerin analizi ve ayrıştırılmasında kullanılan bir tekniktir (Weeks, 2006:275). Standart Jpeg ise DCT kullanır. Görüntüde yüksek frekanslı bileşenlerinin istenilen sıkıştırma oranında silinmesine dayanan bir tekniktir (Oğuz, 2006:20). Jpeg 2000 ise DWT teknolojisinin getirileri sayesinde, DCT'de olduğu gibi sadece kayıplı sıkıştırma değil, aynı zamanda kayıpsız sıkıştırma da yapabilmektedir (Göçeri ve Yaldır, 2007).

Jpeg2000 standardı henüz yeterince yaygınlaşmadığından, kullanılan görüntü işleme araçları genelde bu veri tipini desteklememekte veya üzerinde işlemler yapıldığında görüntü kalitesinde bozulmalar gözlenmektedir (Göçeri ve Yaldır, 2007). Bu yüzden geliştirilen uygulamada jpeg2000 desteği içermemektedir. Uygulama internet üzerinden görüntü iletmek ve fotografik görüntü saklamak için en popüler dosya biçimi olan standart kayıplı Jpeg dosya biçimini desteklemektedir.

Jpeg sıkıştırma algoritmasının çalışma prensibi genel olarak şöyle tarif edilebilir. Kayıpsız ve sıkıştırılmamış Bitmap grafikleri Jpeg'e dönüştürmek için öncelikle standart renk paleti olan RGB'den YCbCr olarak adlandırılan farklı bir palet türüne dönüştürülür (Chip, 2009). Bununla birlikte, JPEG formatlarında YCbCr yanında YUV (Y: Luminance, U: Chrominance1 (blue), V: Chrominance2) renk uzayıda kullanılabilmektedir. (Singh ve diğ., 2009). YCbCr'de Cb ve Cr bileşenleri (kroma olarak da adlandırılır) kırmızı ve mavi renklerin yoğunluğunu ifade ederken, Y bileşeni renklerin parlaklık değerini taşır (Chip, 2009). Bir sonraki adımda, Şekil 3.3.'de görüldüğü gibi renk bileşenlerinden oluşan 8x8 bloklara ayrık kosinüs dönüşümü uygulanır (Wikipedia, 2012 h). Bu kısımda herhangi bir sıkıştırma yapılmaz dolayısıyla bilgi kaybı olmaz (Çıbuk, 2004:69).

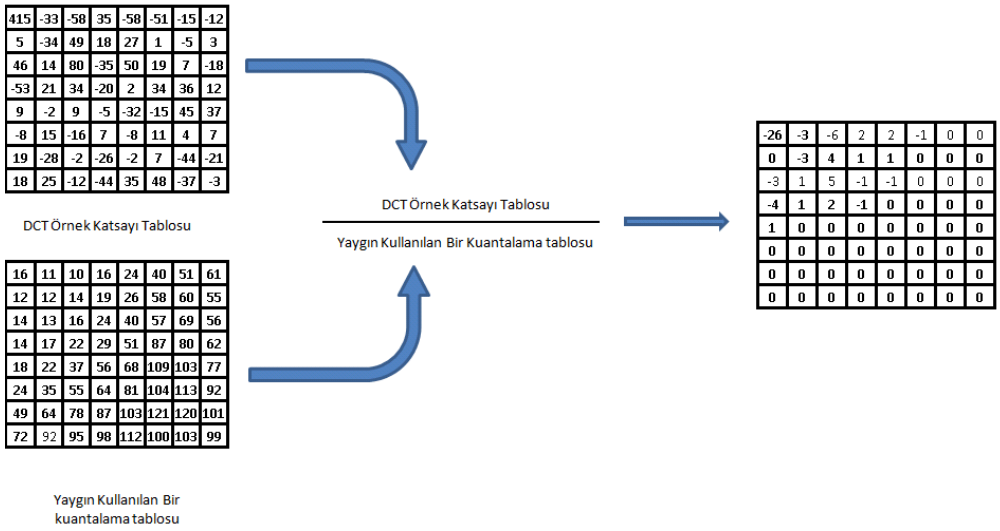
Resmin dikey ve yatay orijinal renk değerleri, özel bir formül yardımıyla hesaplanır ve frekanslarının en yüksek ve düşük olduğu noktalar saptanır (Chip, 2009). Sonuçta görüntü yalnızca verinin fazlalıklarının daha rahat atılabileceği bir biçime dönüştürülür (Çıbuk, 2004:69).



Şekil 3.3. 8x8 bloklara ayrılan Jpeg resim.

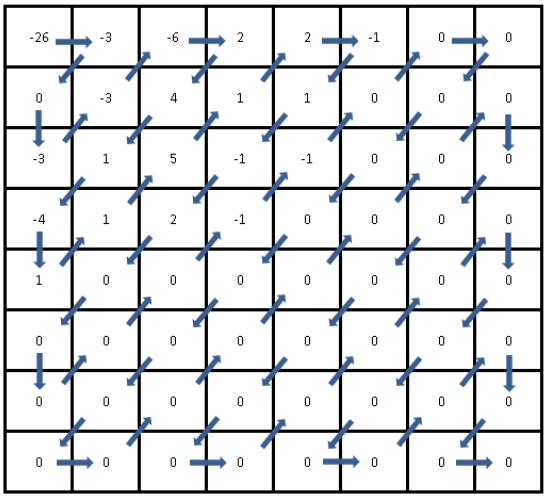
DCT işlemi ile frekans katsayıları ile ifade edilmiş verilerin gereksiz bitleri atılır (Çıbuk, 2004:69). DCT ile elde edilen 8x8 blok sonuçları, seçilen kalite oranına göre belirlenen bir kuantulama tablosuna bölünerek kuantulama işlemine tabi tutulur. Böylece çoğu yüksek frekanslı değerlerin 0 (sıfır)'a indirgenmesi sağlanır ve yeni tablodaki değerlerin tamsayı olabilmesi için bu değerler en yakın tamsayıya yuvarlanır (Uslu, 2003:22).

Şekil 3.4'te görülen kuantalama işlemi sonucunda DCT ile elde edilen 64 adet frekans katsayılarının belirli bir oranı sıfır değerini alacaktır. Bu aşama, Jpeg sıkıştırma en önemli kısımlardan biridir, çünkü orijinal resmin kalitesi azalmakta, insan gözünün algılamayacağı veriler atılmaktadır. Sonuçta, Huffman veya aritmetik kodlama şemaları kullanılarak çok etkili şekilde sıkıştırılabilecek elverişli DCT katsayıları listesi üretilir (Schroeder,2012).



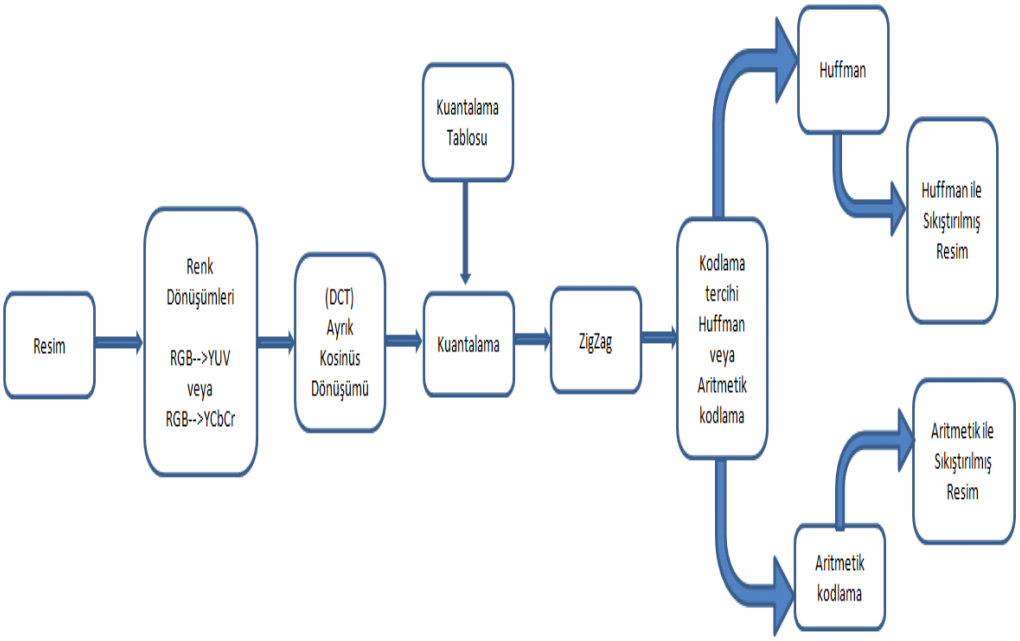
Şekil 3.4. Örnek Kuantalama İşlemi.

Şekil 3.5'te görüldüğü gibi kodlanmadan önce 8x8 blok zig-zag düzeninde okunarak sıralanır. Ayrıca düşük frekansla gösterilen elemanlar bloğun başlangıcına taşınır ve yüksek frekansla gösterilen elemanlar bloğun sonunda doğru koyulur (Schroeder, 2012). Son aşama, resmin sıkıştırılmasıdır ve Huffman ya da aritmetik kodlama şemaları göz önünde bulundurulur. Aritmetik kodlama Jpeg algoritmasının kodlama safhasında kullanılmak istendiğinde üstünde bulunan patent sorunu ile karşılaşılır. Bu standart IBM, AT&T ve Mitsubishi tarafından patentlidir. Kullanım için bu firmalardan lisans sağlanma zorunluluğu vardır (Fags, 2012). Bu yüzden aritmetik yöntem popüler değildir.



Şekil 3.5. ZigZag işlemi.

Şekil 3.6'daki görülen Huffman Kodu, bilgisayar biliminde**,** veri sıkıştırması için kullanılan, bir entropi kodlama algoritmasıdır (Wikipedia, 2012 b)**.** Entropi kodlaması, verinin içeriği ile ilgilenmektedir. Yalnızca, bir kodlama algoritması aracılığı ile veriyi oluşturan bitleri yeniden kodlar. Bu nedenle, bilgi kaybı olmayan entropi kodlamasında , sıkıştırılmış veriyi alan bilgisayar kod çözme algoritmaları ile gizli verinin ilk halini geri elde edebilir. Bu sebepten bu tip sıkıştırmaya "kayıpsız sıkıştırma" adı verilir (Çıbuk, 2004:18).



Şekil 3.6. Genel olarak Jpeg algoritması aşamaları.

3.4.1.2. Avi

Ses ve Video birleşimini temsil eden AVI, Windows çoklu ortam framework'u için Microsoft'un kullandığı bir kapsayıcı biçimdir (Afterdawn, 2012). Framework kısaca; windows uygulamalarının geliştirilmesi, uygulanması ve yönetilmesi için gerekli bir alt yapı platformu olarak tanımlanabilir. 1992 yılında Microsoft tarafından geliştirilmiştir. Bununla birlikte Avi, Windows dışında Macintosh, Linux, Unix gibi işletim sistemleri ve güncel web browserlerin çoğunluğu tarafından desteklenmektedir.

Temel olarak 3 tip AVI dosyası mevcuttur. Orijinal eski AVI dosya yapısı olan AVI 1.0, AVI'nin genişletilmiş bir sürümü olan Open-DML ve uyumluluk adına bir ek miras indeks (legacy index) içeren Open-DML Hibrid Dosya (Hybride-Files) (Techkiwhitepaper, 2012).

Bir indeks, video, ses veya diğer indekslerinin yeri hakkında bilgi içeren yapıdır. Avi dosyalar içinse, farklı indeks türleri mevcuttur. Az önce bahsi geçen ve bazen AVI 1.0 adı ile anılan Legacy indeks (Legacy index), normal AVI dosyasının bir bölümde depolanır. Bu indeks video, ses verilerinin yerini gösterir (Alexander-neo, 2012).

Orijinal eski AVI (bazen AVI 1.0 olarak gösterilir), kullanılacağı işletim sisteminin dosya sistemi daha büyük dosyaları desteklemesine rağmen, boyut sınırı 2 GB'tır. Bu limiti aşmak için, Open-DML uzantısı (ayrıca "AVI 2.0" olarak gösterilir) Matrox OpenDML grubu tarafından 1996 yılında geliştirilmiştir (Videospark, 2012). Böylece 2 GB'ı aşan AVI dosyaları oluşturulabilmiştir.

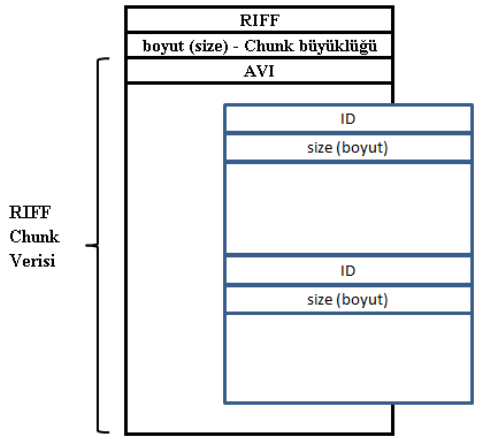
Avı dosyaları windows'ta görüntülemek için, okuyucu yazılımın bilgisayarda hangi çözücülerin (decoder) olduğunu ilgili avi dosyasını yorumlayabilmesi (render) için bilmesi gerekmektedir (Afterdawn, 2012). Bunun için bütün AVI dosyaları dört karakterlik kod yapısında olan FourCC adı verilen bir özelliğe sahiptir. FourCC ASCII karakter sıralanarak oluşturulmuş 32 bitlik işaretsiz tamsayıdır. Örneğin FourCC 'abcd' 0x64636361 olarak gösterilir. FourCC'ler boşluk karakteri de içerebilmektedir, bu durumda 'abc ' de doğru bir gösterimdir (Microsoft, 2012 a). Böylece AVI içeriğindeki çeşitli kısımlar bir etiket gibi düşünebileceğimiz FourCC tarafından tanımlanmaktadır (Microsoft,2012 b). Örneğin popüler bir kodek (codec) olan Cinepak video kodek 'CVID' olarak FOURCC yapısında gösterilir (McGowan,2012).

Sonuçta bu özellik sayesinde medya dosyaların başlangıcında bulunan, çoğunlukla AVI ile ilişkili olan, işletim sistemine dosyanın çözülmesi için hangi kodek'e gereksinim olduğu bildirilebilmektedir. Kodek, bir şarkı veya video gibi dijital medya dosyasını sıkıştırmak veya sıkıştırılmış dosyayı açmak için kullanılan yazılımdır. Bazen kodek ile AVI dosyası karıştırılmaktadır, oysa AVI ve kodek tamamen birbirinden farklıdır. AVI, kodekler ile sıkıştırılmış ya da sıkıştırılmamış ses veya video dosyalarının tutulabildiği bir kaptır. Örneğin, DivX bir video'nun nasıl kodlanacağını söylerken, AVI ise kodlanan bu bilginin ilgili diğer bilgiler ile nasıl depolanacağından sorumludur (Differencebetween, 2012). Bu bağlamda Windows Media Player ve diğer programlar dijital medya dosyalarını yürütmek ve oluşturmak için kodek bileşenlerini kullanır. Bir kodek bir kodlayıcı ve bir kod çözücü olarak iki bileşenden oluşur. Kodlayıcı sıkıştırma (encoding) işlevini gerçekleştirir ve kod çözücü açma (decoding) işlevini gerçekleştirir.

Bazı kodek bileşenleri bunların ikisini de içerirken bazıları yalnızca birini içerir. Örneğin, bir ses CD'sindeki bir şarkı bilgisayara kopyaladığında, Player şarkıyı küçük bir WMA dosyası halinde sıkıştırmak için varsayılan olarak Windows Media Audio codec bileşenini kullanır.

Bu WMA dosyası yürütüdüğünde, Player Windows Media Audio kodek bileşeniyle dosyayı genişleterek, müziğin hoparlörler yoluyla çalınabilmesini sağlar (Microsoft, 2012 b). Bunun sonucunda, dosyayı sıkıştırmak için hangi kodek bileşenlerinin kullanıldığına ve bilgisayarınızda hangi kodek bileşenlerinin yüklü olduğuna bağlı olarak, bazı AVI dosyalarını çalıştırabilirken diğerlerini çalıştıramayabilir. Aynı nedenden dolayı, bir AVI dosyasının ses kısmını yürütebiliyorken, video kısmı yürütüleyemeyebilir (Microsoft, 2012 b). AVI için FourCC Yöntemine geri dönüldüğünde, IFF dosya biçiminde 80'li yıllarda Amiga bilgisayarı için Electronic Arts tarafından uygulandığını görürüz. Daha sonra bu yöntem Apple'da AIFF'ye ve Microsoft'da RIFF'e kopyalanmıştır. Şu anda Microsoft RIFF, resim ve ses gibi farklı türdeki verileri depolamak için yaygın olarak kullanılmakta ve windows çoklu ortam uygulamalarının temel biçimini oluşturmaktadır (Yoo ve diğ., 2011). Ayrıca RIFF bir dosya biçimi (file format) değil birçok özel dosya biçimini tanımlayan bir yapı (file structure) olduğu unutulmamalıdır ve bu biçim çeşitli çoklu ortam dosyalarini kapsadığından dosya uzantısı olarak kendi ismini kullanmak yerine onları içeren veri türü uzantısı ile isimlendirilir (Digitalpreservation, 2012).

Şekil 3.7.'de görülen RIFF dosyaların içine Wav, RDI, RMI ve AVI depolanabilecek veri örnekleri olarak gösterilebilir (Fileformat, 2012). Böylece RIFF dosya Wav, RDI yada AVI ismini alabilir. Sonuç olarak Avi, bir dosya verisini parçalar(chunks) halinde bölen RIFF ın türevi (Wikipedia, 2012 c) ve onun özel bir durumudur (McGowan, 2012).



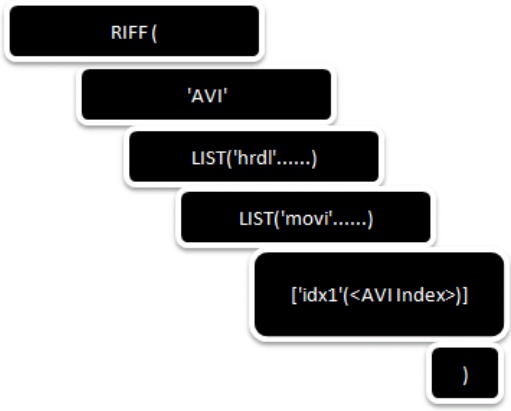
Şekil 3.7. Standart RIFF AVI formu örneği.

RIFF temel olarak 'RIFF', dosya boyutu (fileSize), dosya tipi (fileType) ve veri (data)'dan oluşmaktadır. 'RIFF' olduğu yerde literal FourCC kodu 'RIFF'' tir. Dosya boyutu dosya'nın büyüklüğünü veren 4 byte'lık değerdir. Dosya Tipi spesifik dosya tipi'ni tanımlayan bir FourCC'dir. Dosya Büyüklüğü'nün değeri FourCC dosya tipinin boyutunu ayrıca takip eden verilerin boyutunu da içermektedir, fakat RIFF FourCC'sinin veya dosya boyutunun büyüklüğünü içermez. Dosya verisi (file Data) ise herhangi bir sırada listeler (list), parçaları kapsar (Microsoft, 2012 a).

Bir RIFF dosyanın temel yapı taşı parça olarak adlandırılır (Microsoft, 2012 a) ve bir AVI içindeki verinin en küçük parçasıdır (Alexander-neo, 2012). Bir parça 'ckID', 'ckSize', 'ckData' oluşmaktadır. 'ckId' parça içindeki verinin kapsamını tanımlayan bir FourCC'dir. 'ckSize' ise 'ckData' içindeki verinin boyutunu veren 4 byte'lık değerdir ve ckData 0(sıfır) ya da daha fazla byte yer kaplar. 'ckSize' parça içindeki verinin boyutunu verir. ckSize dolgu (padding)'i, 'ckID' ve 'ckSize' boyutu içermez (Microsoft, 2012 a).

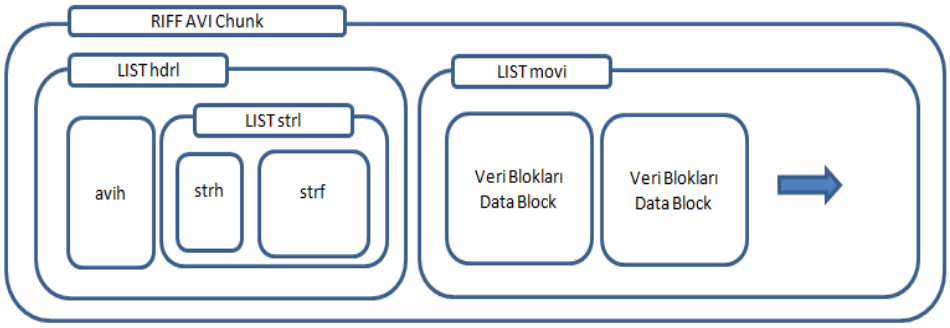
Bir list 'LIST', 'listSize', 'listType', 'listData''dan meydana gelmektedir. LIST olduğu yerde literal FourCC kodu 'LIST'dir ve listSize bu listenin boyutunu veren 4 byte'lık değerdir (Microsoft, 2012 a).

‘listType’ bir FourCC kodudur ve ‘listDATA’ herhangi bir düzende listelerden ve parçalardan meydana gelmektedir, ‘listSize’ değeri ‘listType’ boyutu ek olarak listData boyutunu içermekle birlikte ‘LIST’ FourCC veya ‘listSize’ boyutunu içermez (Microsoft, 2012 a). Avı dosyalar RIFF başlığında(header) FourCC ‘AVI’ ile tanımlanır. Tüm avi dosyalarda sırasıyla stream biçimini ve stream veriyi tanımlayan iki zorunlu LIST parçayı içerir. Bir AVI dosya ayrıca içindeki veri parçalarının yerini veren bir indeks parça içerebilir. Bir avi dosya ile bu bileşenler Şekil 3.8.’de görüldüğü gibi bir yapıya sahiptir (Microsoft, 2012 a) :



Şekil 3.8. Rıff içindeki avi bileşenlerine bir örnek.

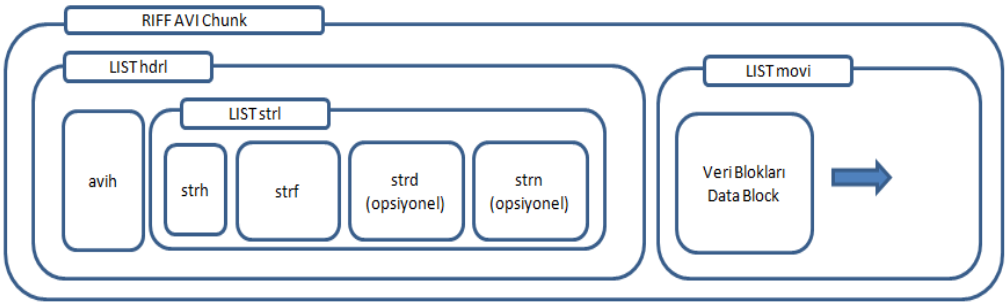
Hdrl (header list) listesi verinin biçimini tanımlar ve ilk gerekli List parçasıdır. ‘movi’ list AVI için veriyi tutar ve bu liste ikinci gerekli List parçasıdır. ‘idx1’ list bu indeksi kapsar. AVI bu üç bileşeni doğru sırada tutmak zorundadır (McGowan, 2012). ‘hdrl’ ve ‘movi’ listeleri, bu veriler için altparçaları kullanır. Şekil 3.9’de örnek bir AVI RIFF formu gösterilmektedir.



Şekil 3.9. Bir RIFF AVI formu.

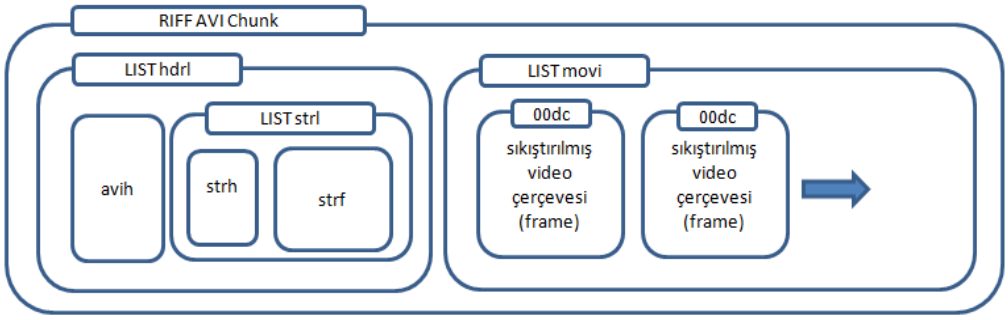
Ana AVI başlığı (Avi Main Header), ‘hdrl’ listesi içinde tutulmuş bir avih (avi header) parçası ile başlar. Ana başlık (Main header) mevcut AVI dosyanın dosya içindeki akışların (streams) sayısı ve AVI dizisinin yüksekliği ve genişliği gibi global bilgisini içerir (Microsoft, 2012 a). Böylece ‘hdrl’ başlık özelliği yanında video hakkında metadata’yı da tutmaktadır (Videospark, 2012). Bir veya daha fazla strl (stream list) listesi ana başlığı takip eder. Bir ‘strl’ her veri bilgi akışı (data stream) için gereklidir (Microsoft, 2012 a).

Her bir strl liste bu avi dosya içinde bir bilgi akışı hakkında veri içerir ve her bir strl içinde bir srth (stream header) ve strf (stream format) parçası içermek zorundadır. Ayrıca, Şekil 3.10’da olduğu gibi bir ‘strl’ listesi bir strd (stream header data) ve strn (stream name) parçası da içerebilir (Microsoft, 2012 a).



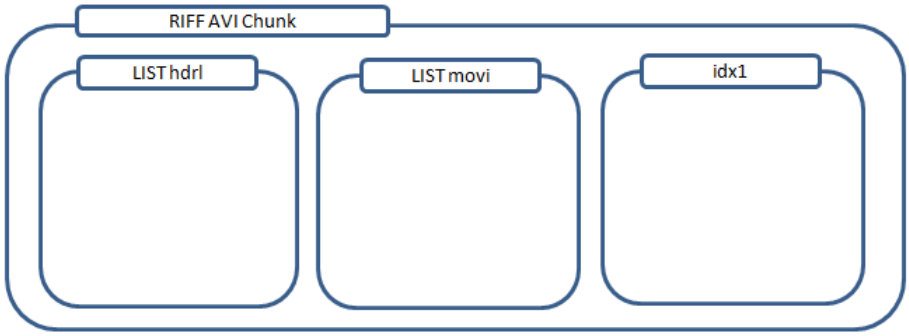
Şekil 3.10. Strd ve srtn eklenmiş AVI formu.

Bilgi strf parçası bilgi akışı içinde verinin biçimini tarif eder. Bu chunk içinde kapsanan veri bilgi akışı tipine bağlıdır (Microsoft, 2012 a). Eğer bilgi strd mevcutsa, bu strf parçasını takip eder. Bu biçim ve chunk içeriği kodek sürücüsü tarafından tanımlanır. Genellikle, sürücüler (drivers) yapılandırma (configuration) için bu bilgiyi kullanır (Microsoft, 2012 a). ‘hrdl’ list içinde stream başlıkları ‘strl’ chunklarının düzenine göre ‘movi’ list içinde stream verisi ile ilişkilidir. Başlık bilgisini takip eden ‘movi’ listesi akışlar içinde mevcut veriyi yani ses ve video çerçevelerini içeren bir yapıdır. Data parçaları ‘movi’ list içinde doğrudan tutabilmekte veya ‘rec’ listeleri içinde gruplanabilmektedir (Microsoft, 2012 a). İki-digit akış sayısından meydana gelen her bir veri parçasını tanımlayan FourCC, chunk içindeki bilgi tipini tanımlayan iki karakter kodunu takip eder. Bunlar db (uncompressed video frame), dc (compressed video frame), pc (palette change), wb (audio data)’dir. Örneğin, Şekil 3.11.’de görüldüğü gibi sıkıştırılmış video verisi içeren veri parçası FourCC’si ‘00dc’ dir (Microsoft, 2012 a).



Şekil 3.11. AVI formu içinde '00dc' sıkıştırılmış video çerçeveleri.

Bir opsiyonel idx1 (index1) parçası movi listesini takip edebilir. Şekil 3.12.'de görüldüğü bu indeks veri parçalarının bir listesini ve onların dosyadaki yerlerini tutar. Veri bir gereksinim olarak çöp (Junk) parçaları da bir AVI dosya içine koyulabilmektedir (Microsoft, 2012 a).



Şekil 3.12. AVI formu içinde 'idx1' eklentisi.

Uygulama'da ise verinin gizlendiği çıktı Avi'si, onun sıkıştırılma uygulanmamış formudur. Bu durum AVI'nin daha büyük dosya boyutu ile oluşturulmasına neden olmasına rağmen kalite kaybı söz konusu değildir ve içine gizlenen verilerde kayıplar oluşmaz. Sıkıştırılmamış AVI üzerinde bir kodek seçimi ile yapılan bir dönüşüm ise farklı oranlarda sıkıştırılmaya yol açacaktır ve gizlenmiş verilerin bulunduğu yerlerde kayıplar oluşturacaktır.

3.4.2. Kullanılan Ses Dosya Türü Mp3

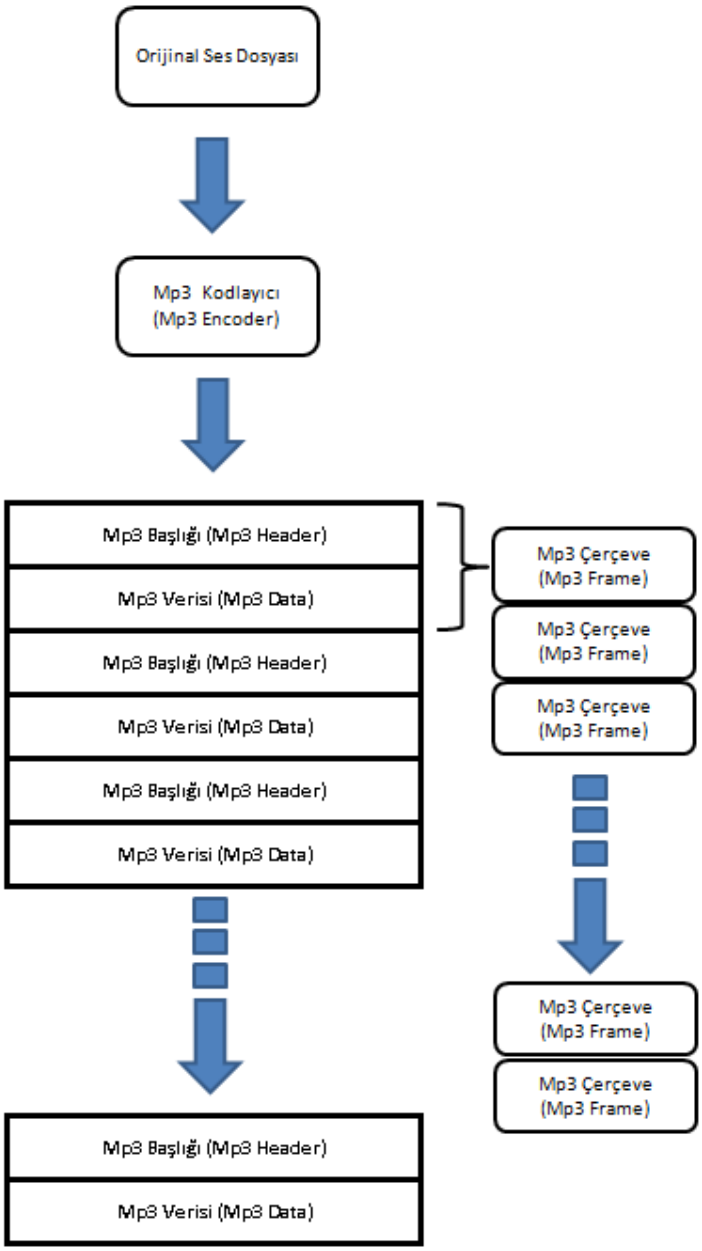
Mpeg-1 standartı ISO (International Organization of Standardization) içinde Motion Pictures Expert tarafından geliştirilmiştir. Bu standart ses verisinin kodlanması için 3 (üç) farklı katmanı (layer) tanımlar. Mp3 olarak adlandırılan üçüncü katman diğer katmanlara nazaran en verimli olandır ve cd kalitesine yakın ses kodlamasına sahiptir (Wikipedia, 2012 d). Bu standart içinde coder-decoder (kodek,kodlayıcı-çözücü) sıkıştırma algoritmaları önemli bir yer tutmaktadır. Kodek ses verilerinde herhangi bir değişiklik yaptırmayan Flac, WavPack, Shorten, Monkey's Audio, Apple Lossless gibi kayıpsız ses kodekleri "lossless sound codecs" ve Ogg Vorbis, Mp3, Mp4, WMA, QuickTime, AAC gibi ses verilerini değiştirebilen kayıplı ses kodekleri "lossy codecs" olmak üzere iki kısma ayrılmaktadır (Selasky, 2006:9). Bu sıkıştırma algoritmalarından önce ses örnekleri bilgisayarlarda sadece wav, pcm gibi farklı biçimlerde depolanırdı. Halen kullanılmakta olan bu biçimlere, ses sayısal olarak depolanırken insan kulağının duyamayacağı ses frekansları da eklendiğinden dosya boyutu aşırı büyümektedir (Wikipedia, 2012 d).

Mp3 sıkıştırma tekniği aşamalarında önemli bir yere sahip olan, basit tanımıyla ses algılaması üzerine çalışan bir bilim dalı Psychoacoustics'ten yararlanılmaktadır (Wikipedia, 2012 e). Kullanılan psychoacoustic model insan kulağının bir ses kaydındaki tüm ses frekanslarını duyamayacağını kabul etmektedir. İnsan için, duyma aralığı 20 Hz ile 20 kHz arasındadır ve bu aralığın 2-4 kHz kısmı en duyarlı kısmıdır (Sripada, 2006:9). Mp3 içinde ses sıkıştırıldığında, duyulamayan frekanslar ortadan kaldırılmaktadır. Buna yok edici (destructive) sıkıştırma adı verilmektedir çünkü bu işlemden sonra saf dışı edilen bilgiler geri döndürülememektedir (Sripada, 2006:9).

Şekil 3.13'de görüldüğü gibi Mp3 algoritmaları, ses verisini çerçeve (frame) adı verilen küçük parçalar ayırır. Çerçevenin boyutu ise ses çözünürlüğüne veya bit oranına bağlıdır. Bunu yapmak için kodlayıcı (encoder) kayıt boyunca sabit bir bit oranını varsayar. Bu noktada MP3 spesifikasyonu, ses çerçevelerinin aynı boyutta olmadığını ifade eden bir değişken bit hızı formatında (VBR), verinin saklanmasına olanak sağlamaktadır (Maciak ve diğ, 2006:5).

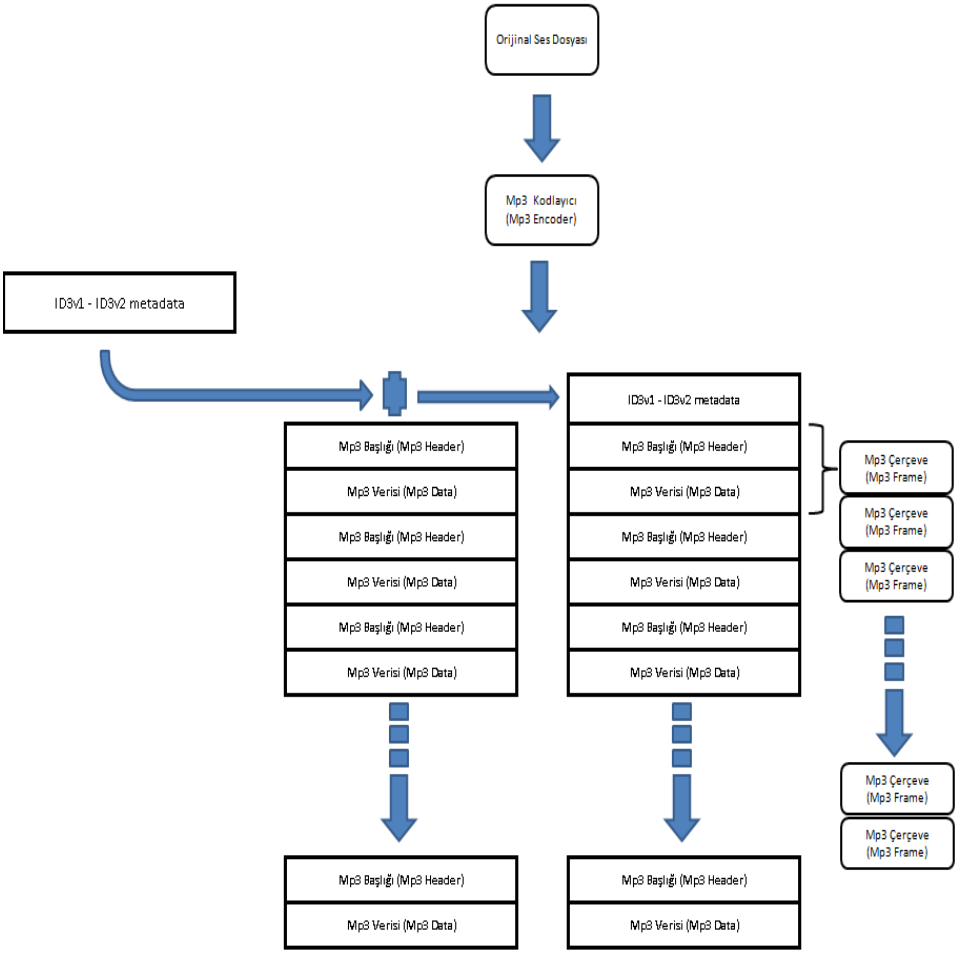
Her çerceve psychoacoustic model kullanılarak algısal olarak analiz edilmiştir. İşitilemeyen bu frekanslar atılır, veya bitlerin en düşük sayısına tahsis edilir (Maciak ve diğ, 2006:5).

Algısal optimizasyon yapılır yapılmaz veri Huffman kodlama yapılarak sıkıştırılır. Huffman kayıpsız bir algoritmadır. Böylece sesin bilgisi korunurken depolama alanı azaltılır. Bu steganografi geliştiriciler için önemli bir olaydır. Sıkıştırma algoritmalarının doğasından dolayı, Huffman ile kodlanmış veri kolayca değiştirilemez (Maciak ve diğ, 2006:5).



Şekil 3.13. Mp3 kodlayıcı'in ses verisini çerçevelere ayırması.

Şekil 3.14.'de görüldüğü gibi Mp3 dosya ayrıca bazı metadata etiketleri (tag) içerebilir. Bu etiketlerin iki türü vardır. ID3v1, daima 128 byte uzunluktadır ve sanatçı ismi, şarkı ismi, album, tarz ve gibi belirten yedi alan içerir.



Şekil 3.14. ID3v1 etiketi eklenmiş bir Mp3 dosya yapısı.

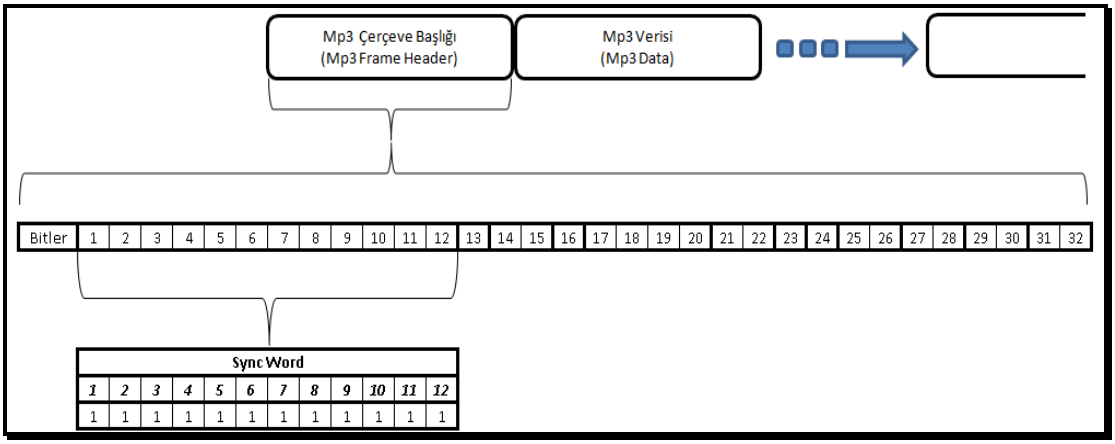
Statik boyutu ve esneklik eksikliği nedeniyle, bu etiket türü yavaş yavaş daha ileri düzey ID3v2 standartı tarafından değiştirilmiştir. Etiketlerin yapısı hemen hemen Mp3 dosyanın kendi yapısı kadar esnektir (Maciak ve diğ, 2006:5). ID3v2 etiketleri bilginin çeşitleri bitlerini depolayan kendi çerçevelerinden oluşmaktadır. Bu sanatçı ismi ve şarkı başlığı gibi standart karakter dizisi olabilir veya daha gelişmiş bilgi bu yolla dosyaya kodlanır. ID3v2 etiketleri üzerinde herhangi boyut sınırı yoktur böylece teoride etiketler sonsuza kadar büyüyebilir. Bununla birlikte Mp3 dosyalarında her iki etiketide içerebilir (Maciak ve diğ, 2006:5).

Çerçeve boyutu, bit oranı ve örnekleme frekansı'nın bir fonksiyonudur. Belirli bir çerçeve boyutu byte cinsinden aşağıdaki formül ile hesaplanır.

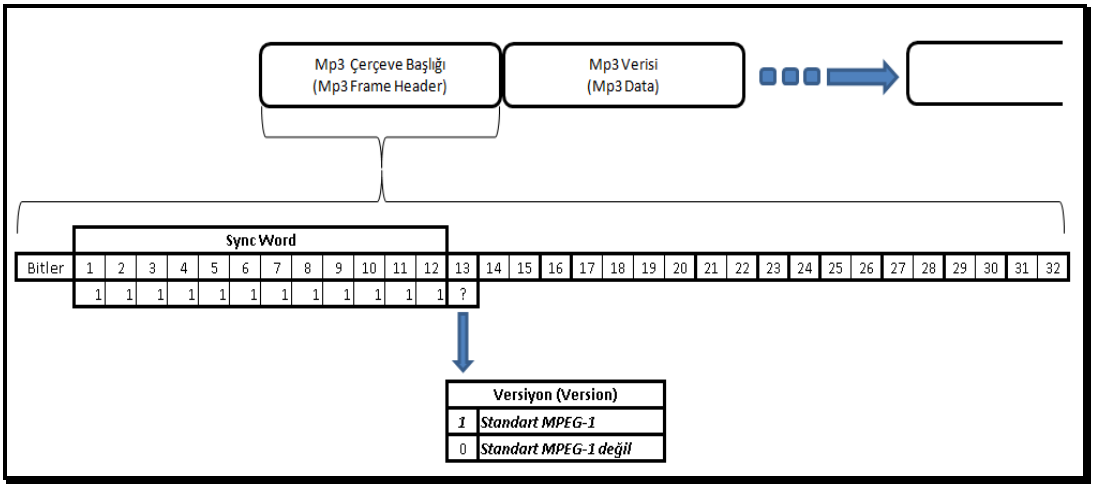
Çerçeve boyutu = (144* bitrate / sample frequency)+ Padding [byte] **(3.1)**

Boyut hesaplamaya dahil olmayan alanların çoğu önemsizdir. Aslında, bunların bazıları çok nadiren kullanılmaktadır. Bazı zamanlar çerçeve oranını eşit şekilde yaymak için çerçevelerin bazı boş byte'larının doldurulmaya ihtiyacı vardır (Maciak ve diğ, 2006:5).

Daha önce de belirttiğimiz gibi, MP3 dosyaları aslında çerçeve boyutu değişir yapan, değişken kare hızı ile kodlanmış olabilir. Çerçeve boyutları belli olmadığı için, bir çerçevenin nerede başladığını ve nerede bittiğini tanımlamak gereklidir. Tüm başlıkların yapıları, içerikleri çok benzerdir ve aslında, çoğu zaman da başlıklar aynı olacaktır. Şekil 3.15'de görüldüğü gibi her başlık 32 (otuziki) bit uzunluğundadır ve Sync word olarak adlandırılan 12 (oniki) bitlik blok ile başlar. Sync bir başlık üstüne hedeflenen bir çözücü programa yardımcı olan bir dizidir. Bu nedenle çözücü bir çerçeve bulmak için 1 (bir) olarak başlatılan 12 (oniki) ardışık bit algılamasına ihtiyaç duyar (Maciak ve diğ, 2006:5).

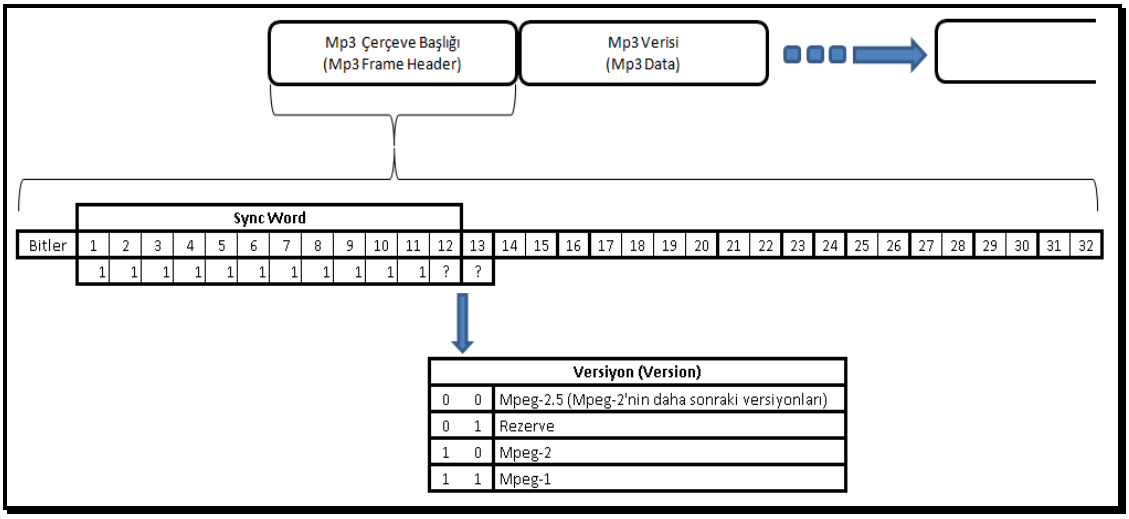


Şekil 3.15. Bir Mp3 çerçevede ilk 12 (oniki) bit Sync Word yapısı.

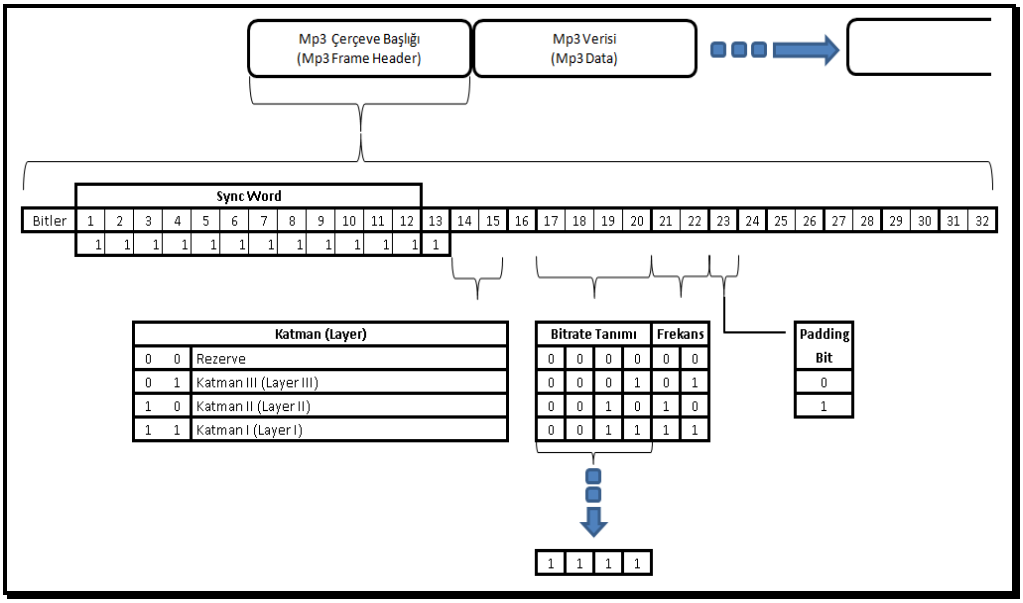


Şekil 3.16. Bir Mp3 çerçevede versiyon tespiti.

Şekil 3.16'da görüldüğü gibi Mp3 çerçevesinin 13'nci bit'i bir ise standart Mpeg-1, değilse Mpeg-2 olarak tanımlanır. Bununla birlikte bazı geliştirmeler sync word'u 11 bit olarak alabilir, bu durumda 12'nci ve 13'ncü bitler alacak değere göre Şekil 3.17'de görüldüğü gibi versiyon tanımlaması yapar (Raissi, 2002:21).



Şekil 3.17. Mp3 11 bitlik sync word ve 2 bitlik versiyon tespiti.



Şekil 3.18. Mp3 başlığındaki katman, bitrate, frekans ve dolgu bit tanımları

Şekil 3.18'de görüldüğü gibi Mp3 çerçevesinin 14'ncü ve 15'nci bitleri, katman tanımı yapmaktadır. 17-20'nci bitler kodlayacı'nın oluşturduğu bitrate oranını göstermektedir. Tablo 3.1. tanımlanmış bitrate değerlerini göstermektedir ve bitlere karşılık gelen değerler çerçeve boyutunun hesaplanmasında kullanılmaktadır. Örneğin, başlık çerçevesindeki ilgili yer "0010" olarak belirtilirse o çerçevenin bitrate oranı değeri Mpeg-1 layer III için 40'tır.

Tablo 3.1. Bitrate oranı tablosu.

bitler	MPEG-1 layer I	MPEG-1 layer II	MPEG-1 layer III	MPEG-2 layer I	MPEG-2 layer II	MPEG-2 layer III
0 0 0 0						
0 0 0 1	32	32	32	32	32	8
0 0 1 0	64	48	40	64	48	16
0 0 1 1	96	56	48	96	56	24
0 1 0 0	128	64	56	128	64	32
0 1 0 1	160	80	64	160	80	64
0 1 1 0	192	96	80	192	96	80
0 1 1 1	224	112	96	224	112	56
1 0 0 0	256	128	112	256	128	64
1 0 0 1	288	160	128	288	160	128
1 0 1 0	320	192	160	320	192	160
1 0 1 1	352	224	192	352	224	112
1 1 0 0	384	256	224	384	256	128
1 1 0 1	416	320	256	416	320	256
1 1 1 0	448	384	320	448	384	320
1 1 1 1						

21’nci ve 22’nci bitler örnekleme frekansını vermektedir. Alabilecek değerler ve karşılıkları Tablo 3.2.’de gösterilmektedir (Raissi, 2002:21). Bazı mp3 ses çerçeveleri çerçeve oranını eşit derecede yaymak için bir "boş" byte doldurur. Örneğin bitrate değeri 128 kbit/s ve örnekleme frekansı 44100 Hz olduğunda bir çerçeve boyutu 417 byte olmaktadır. Bu çerçevenin boyutunu 418 byte yapmak için padding bit olarak tanımlanan 23’ncü bit kullanılmaktadır.

Tablo 3.2. Kabul edilmiş örnekleme frekans tablosu.

bitler		MPEG-1	MPEG-2	MPEG-2.5
0	0	44100 Hz	22050 Hz	11025 Hz
0	1	48000 Hz	24000 Hz	12000 Hz
1	0	32000 Hz	16000 Hz	8000 Hz
1	1	rezerve	rezerve	rezerve

.

Çerçeve başlığı içinde tarif edilen bitler dışındaki bitler tezin kapsamı dışındadır. Bunun ile birlikte bunlar kısaca özetlenirse. 16. bit hata koruma (error protection) açıksa, daha sonra checksum olarak adlandırılan bir başlık izler**,** 24’ncu bit özel bit açıksa (private bit) uygulamaya özgü tetikleyicilere için izin verir**,** 25’nci ve 26’ncı Stereo, joint streo, çift kanal, tek kanal olmak üzere kanal modunun ne olduğunu**,** 27’nci ve 28’nci bitler eğer kanal modu joint stereo ise kullanılan ekstra ayarları , 29’ncı bit parçanın telif hakkının olup olmadığını, 30’ncu parçanın orijinal medyanın birer kopyası olup olmadığını, son olarak 31’nci ve 32’nci bitler ses vurgusu’nu **(**audio emphasis)’i tanımlar.

3.4.3. Kullanılan Diğer Dosya Türleri

3.4.3.1. Metin

1881 ile 1991 yılları arasında Unicode birliği tarafından geliştirilen 16 bit’lik karakter kodlama standartıdır. Her bir 16 bit’lik karakterde 2 byte’lik kodlama sistemi ile hemen hemen tüm lisanları gösteribilecek durumdadır. Günümüzde 21.000 adedi Çin’ce dil karakterlerini içeren kodlanmış 39.000 adet karakter mevcuttur bununla birlikte 65.536 olası Unicode karakter kodu vardır (Altan, 2003:602). Uygulama tasarımında örtü türlerine gizlecek metin türü unicode yapıya uygun olarak geliştirilmiştir.

3.4.3.2. Rar

RAR, bir dosya sıkıştırma ve arşivleme formatıdır. Eugene Roshal tarafından oluşturulmuştur. Oluşturucusunun soyadını almıştır. RAR uzantılı dosyalar .rar şeklinde gözükür. RAR'ın görevi sıkıştırma, bir verinin boyutunu küçültme ile kodlanmasını sağlamaktır ve bunun için WinZip**,** WinRAR gibi aracı yazılımlara ihtiyaç duyar. RAR ile büyük dosyaların parçalanarak sıkıştırılması itibari uzantılar .rar, .r00, .r01, .r02 şeklini alır (Wikipedia, 2012 f).

Uygulama algoritmasının, gizlenecek dosya içine çoklu ortam dosyaları dışında her türlü dosya biçimini de uygulabileneceğini göstermek amacıyla gizlenecek nesneler listesine rar'da eklenmiştir. Bununla birlikte kullanıcı bir çoklu ortam dosyasını rar biçime dönüştüren winzip yada winrar programı ile şifreyebilir, böylece herhangi bir çoklu ortam türü içinde şifrelenmiş çoklu ortam türü saklayabilir.

3.5. Geliştirme Ortamı

Yazılımların tasarımı ve geliştirilmesinde C# programlama dili ve Microsoft Visual Studio 2010 IDE'si kullanılmıştır. Bir IDE, program yazmayı kolaylaştıran bir geliştirme aracıdır. Bu IDE, geliştiricilerin Windows, Web, Bulut, Office ve SharePoint dahil farklı platformlarda yüksek kaliteli uygulamalar oluşturmasını sağlamaktadır. Çoklu monitör desteği sağlayarak çalışmaların istenilen şekilde düzenlenmesine ve yönetilmesine olanak veren bir yapıya sahiptir. IDE içinde birim testlerini derlemek için gereken yöntem saplamalarını oluşturabilen ve böylelikle her kod biriminin düzgün şekilde çalıştığından emin olmayı sağlayan birim test özellikleri içermektedir. Bununla birlikte geliştirilen yazılımlar Net Framework 4.0'a uygun olarak tasarlanmıştır ve en son işletim sistemlerinden biri olan Microsoft Windows 7.0 ailesinde çalışmaktadır.

4. MMHIDE UYGULAMASININ TASARIMI ve GELİŞTİRİLMESİ

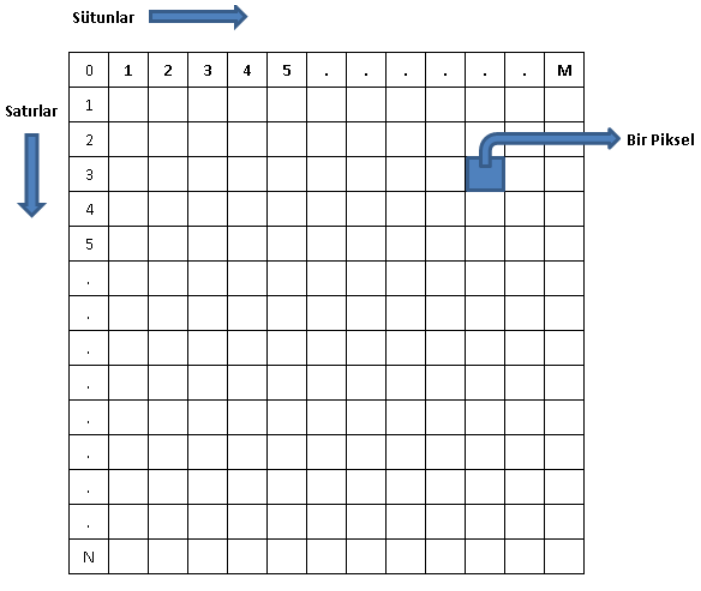
4.1. Görüntü Dosyalarına Bilgi Gizleme ve Çıkarma

Görüntü üzerinde bilgisayar tabanlı steganografi hem resim hem de video dosyalarını kapsamaktadır. Bununla birlikte uygulamalarda büyük payı resim dosyaları almaktadır. Bunun sonucu olarak bu konudaki tasarım ve geliştirmelerin büyük çoğunluğu video ve farklı bir sınıftan olan ses dosyaları üzerindeki çalışmalara göre daha fazladır. Bu yöntemlerdeki çalışma mekanizmaları farklı yaklaşımlar içermesine rağmen, sayısal dosyalar üzerinde yapılacak bilgi gizleme uygulamalarında gerçekleştirilen geleneksel yöntem Şekil 4.1.'de olduğu gibi gösterilebilir.

Şekil 4.1. Bilgi Gizleme Uygulamasında Geleneksel Yöntem.

Resim dosyalarındaki bilgi gizleme uygumaların temelini sayısal resimlerdir. Bu dosya yapıları kare ebatlarında birçok resim noktasından meydana gelmektedir. Sayısal resimler Şekil 4.2'de gösterildiği gibi N satır ve M sütunluk bir dizi ile temsil edilmektedir.

Resim dizisinin elemanlarına piksel (pixel) denir. Kelime anlamı olarak piksel ingilizce pixel (picture element) teriminin kısaltılmasından oluşan, mobil cihazların, monitorlerin, LCD ve LED televizyonların, fotoğraf ve dijital fotoğraf-kameların sensör ve ekran kısımlarının teknik olarak kapasitesini anlatan bir tanımdır ve bu kısımların en temel kısmıdır (Teknovadi, 2012).



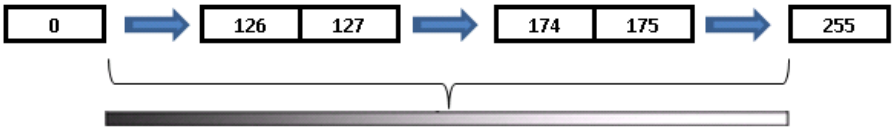
Şekil 4.2. Sayısal resmin yapısı.

En basit piksel yapısı sadece 0 veya 1 değerini almaktadır. Bu şekilde oluşturulan resimlere Şekil 4.3.'de görüldüğü gibi ikili (binary) resim denilmektedir.

Şekil 4.3. İkili resim örneği.

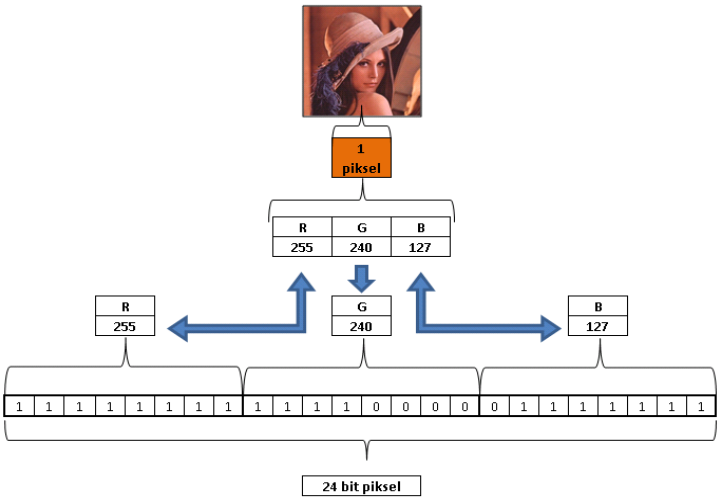
Resimlerin ışık seviyelerini belirleyebilmek için piksel başına 1 byte kullanılmaktadır. Böylece 0 (siyah) ile 255 (beyaz) aralığında tam sayılar elde edilmektedir (Gürel, 2006:13).

Şekil 4.4'de de görüldüğü gibi bu sayılar arasındaki değerlerden oluşan resim gridir ve bundan dolayı bir resme ait bu tam sayılar genel olarak "gri ton seviye" olarak adlandırılır (Gürel, 2006:13).



Şekil 4.4. Gri ton yapısında bir örnek resim yapısı.

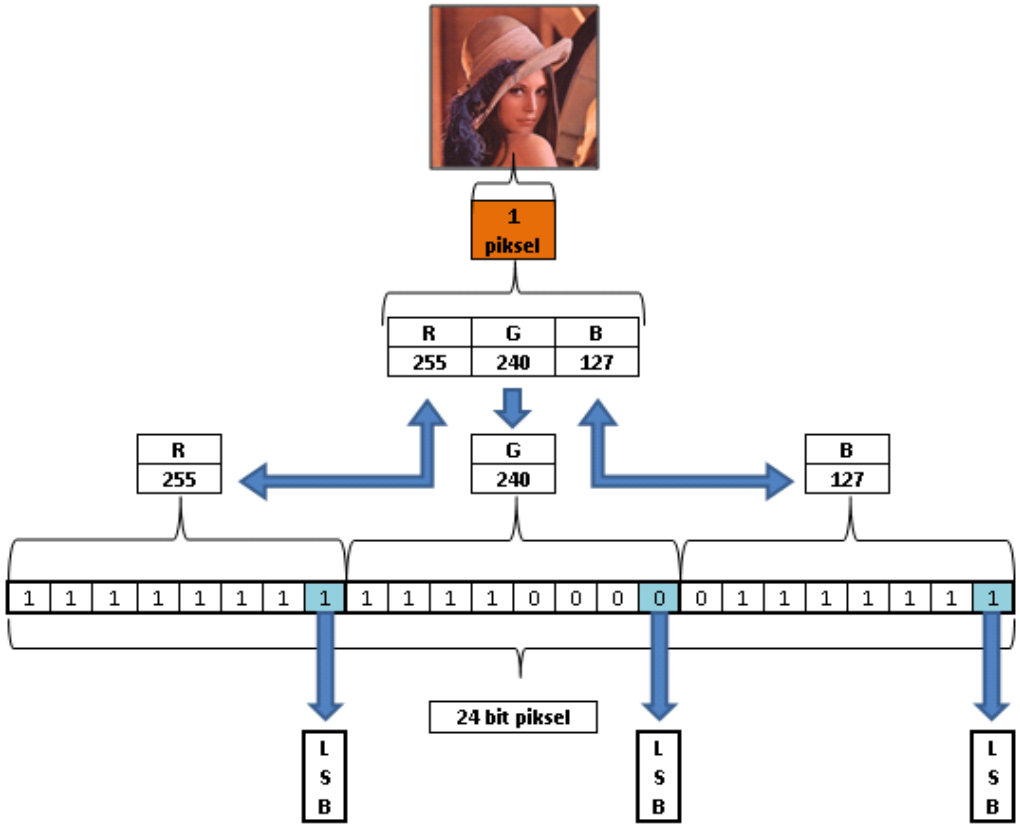
Bir pikselin birden fazla renk alabilmesi 1990-2000 yılları arasında geliştirilen bir teknolojidir (Teknovadi, 2012). Temel olarak renkli resimlerde renkler, kırmızı yeşil ve mavi gibi 3 gruptan oluşmaktadır. Bu üç temel renk karşılık gelen bit değerleri değiştirilerek farklı renk tonları oluşturulabilmektedir. Böylece RGB (Kırmızı–Yeşil–Mavi) modeli ile 256 ton kırmızı x 256 ton yeşil x 256 ton mavi = 16.777.216 adet değişik tonda rengin ortaya çıkmasını sağlamaktadır (Gürel, 2006:14). Bununla birlikte RGB'nin her biri 1 byte kapasiteye sahiptir. Böylece her piksel 3 byte'tır ve her byte 8 bitten oluştuğundan 1 piksel toplam 24 bitlik kapasiteye sahiptir. Şekil 4.5'te 24 bitlik bir örnek resmin oluşumu aşağıda gösterilmiştir.



Şekil 4.5. 24 bitlik bir resmin temel yapısı.

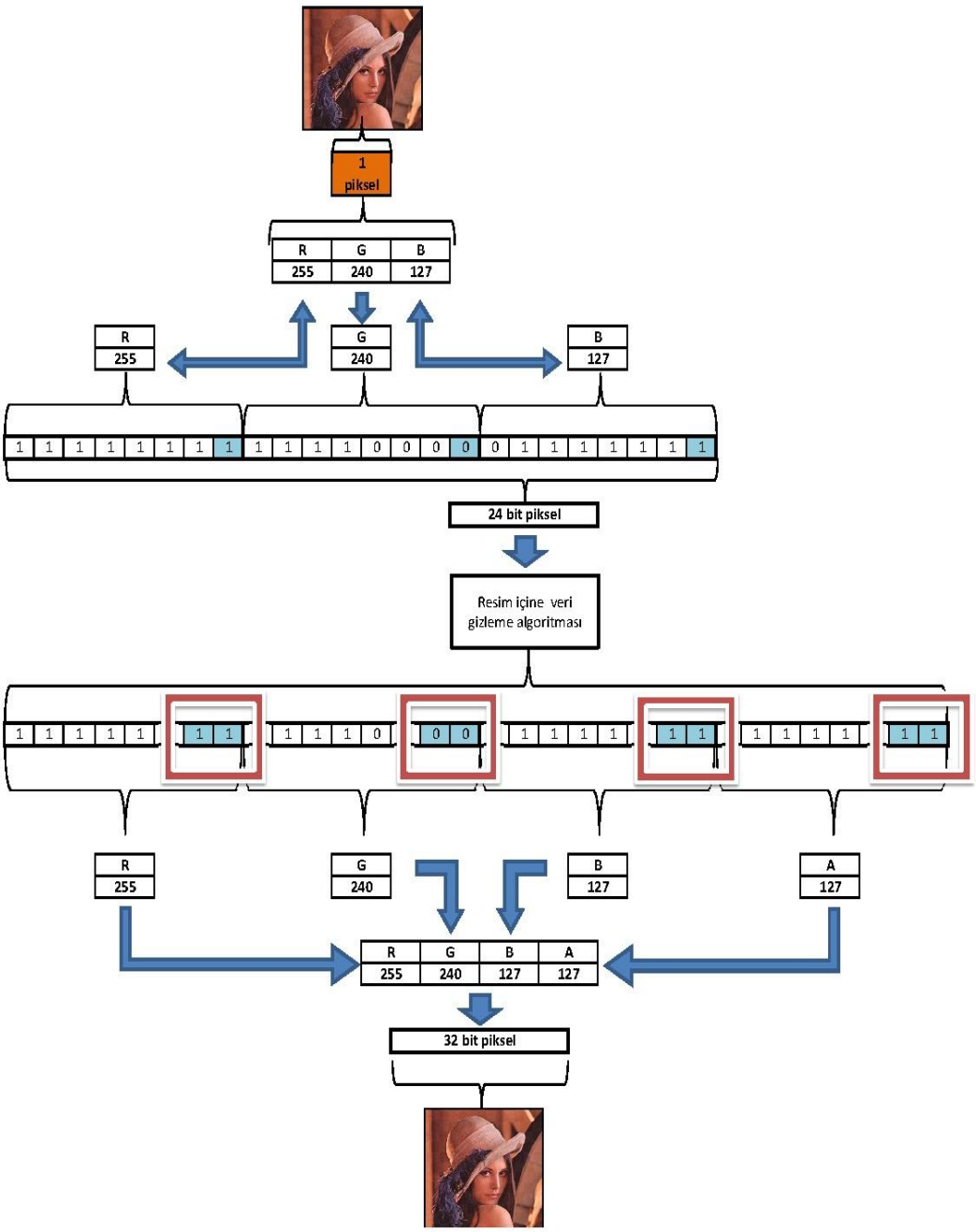
24 bit resim dosyalarının piksellerinin RGB'lerinin en az öneme sahip bitleri (LSB) üzerinde yapılacak herhangi bir değişim insan gözü tarafından ayırtedilmemektedir (Gürel, 2006:14).

Şekil 4.6'da görüldüğü gibi bu yöntemde en fazla bir bit değişimi mevcuttur ve her pikselin her byte'nın en önemsiz biti (son biti)'ne, gizlenmesini istediğimiz verinin bitleri dağıtılarak yerleştirilir.



Şekil 4.6. 24 bit resim üzerinde LSB'nin gösterimi.

Uygulamanın tasarımında resim içine bilgi gizleme için son iki bite gizleme yöntemi örtü verisi olarak da jpeg format yapısı tercih edilmiştir. Bununla birlikte uygulama, RGB'ye ilaveten alpha kanalını da kullanıma sokarak, örtü verisinin depolama kapasitesinin arttırmakta ve bir pixel'in 4 byte'lık bir yapıya sahip olmasını sağlamaktadır. Böylece örtü nesnesi olarak kullanılan 24 bitlik örtü resim 32 bit resme çevrilmektedir. Şekil 4.7'de gösterildiği gibi tüm kanalların son iki bitlerinin kullanımı halinde 8 bitlik bilgi bir pixel'e saklanabilmektedir.

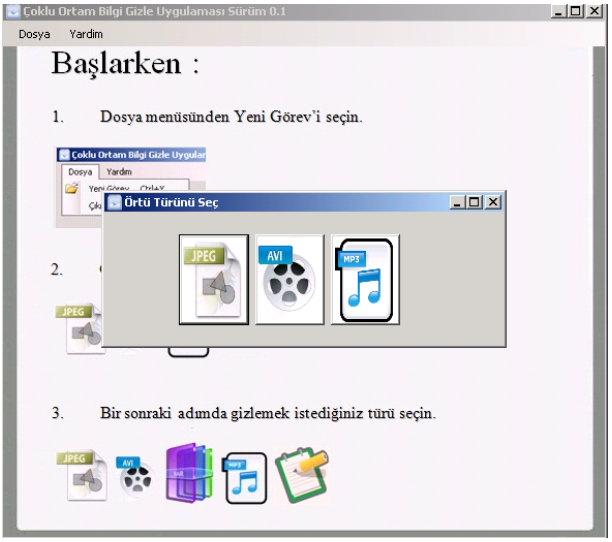


Şekil 4.7. 24 bit resim veri gizleme algoritması ile 32 bit resim dönüşümü.

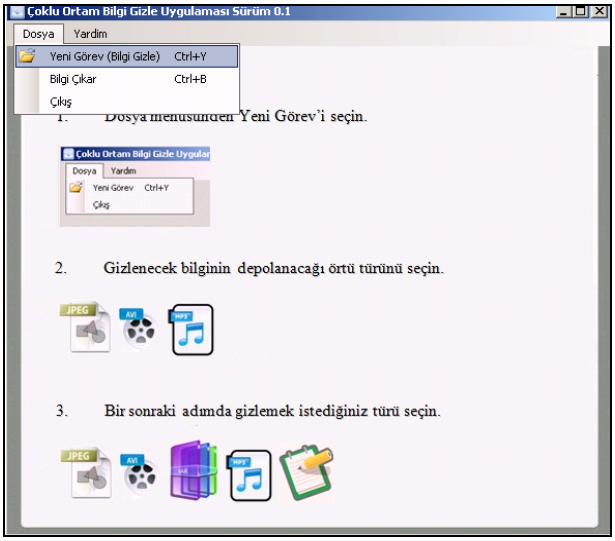
Son yöntem denendiğinde resimdeki içsel bozulma LSB değişim yöntemine göre daha fazla olmasına rağmen gözle görülür bir algısal değişikliğin oluşmadığı gözlemlenir. Saklanmak istenen her byte için gereken piksel sayısının azalması bakımından da bu yöntem LSB'ye göre daha avantajlıdır (Gürel, 2006:23).

4.1.1. Resim Dosyasına Metin Gizleme ve Çıkarma

Uygulamanın başlatılması ile birlikte içerisine veri gizlenecek olan örtü dosyası Şekil 4.8.'de verilen ekranda yeni görev seçilerek belirlenir ve ardından açılan Şekil 4.9'da görülen örtü türünü seç ekranından jpeg işaretlenir.



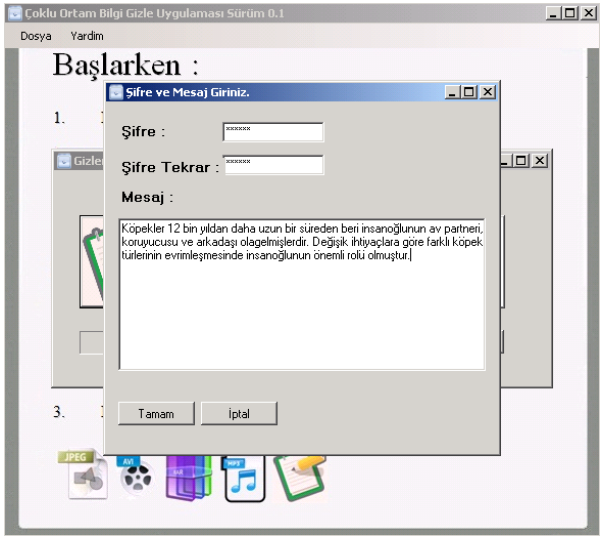
Şekil 4.8. Resim içine metin gizleme sürecinde yeni görev seçimi ekranı.



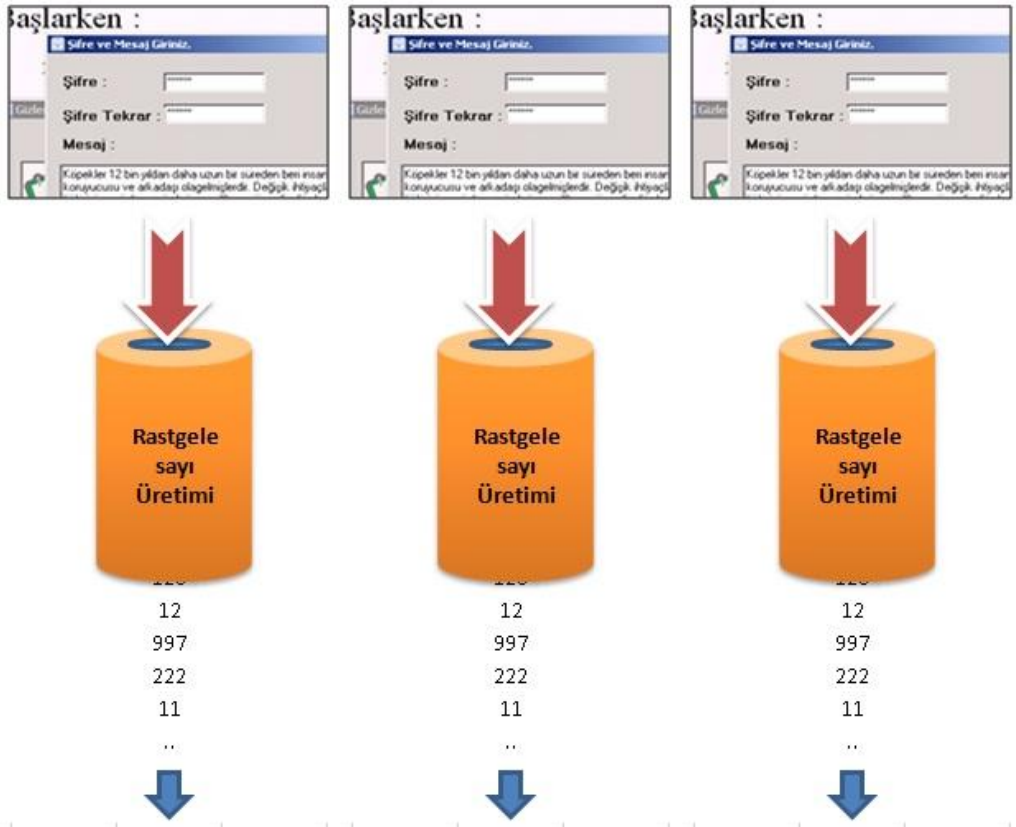
Şekil 4.9. Resim içine metin gizleme sürecinde örtü türü seçimi ekranı.

Şekil 4.10'da görülen örtü içine gizlenecek dosya türü ekranında metin ikonu seçilir. Seçimin ardından Şekil 4.11'daki şifre ve mesaj ekranı belirir. Uygulamanın bu sürümünde şifre sayı girilebilecek şekilde tasarlanmıştır. Girilen şifre tohum değerdir ve rastlantısal sayı üretimi için kullanılır. Şekil 4.12'de görüleceği gibi tekrar tekrar aynı tohum değer kullanıldığında, aynı sayı dizisini oluşturulur.

Şekil 4.10. Resim içine metin gizleme sürecinde örtü tür seçimi ekranı.

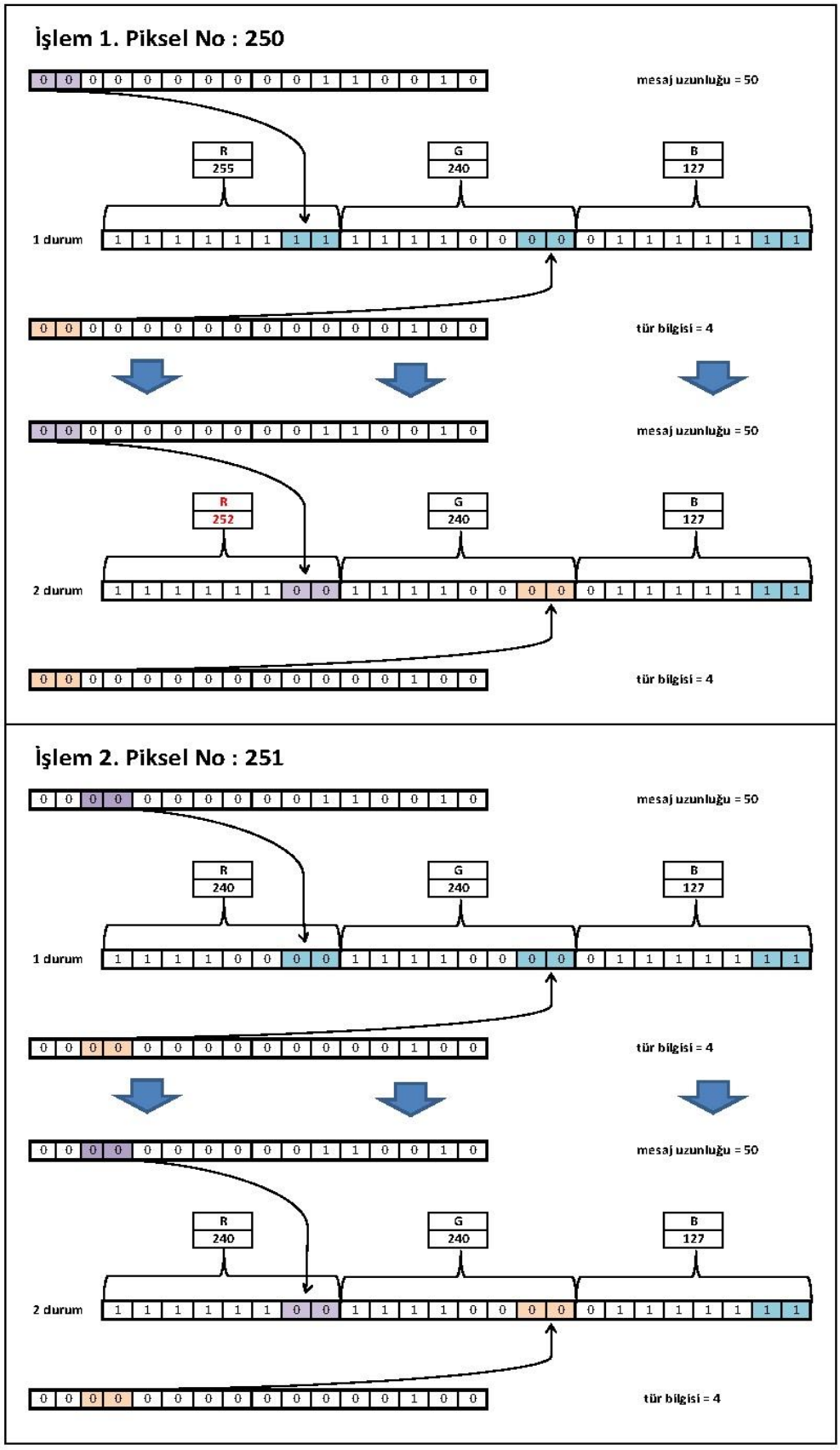


Şekil 4.11. Resim içine metin gizleme sürecinde şifre ve metin girişi ekranı.



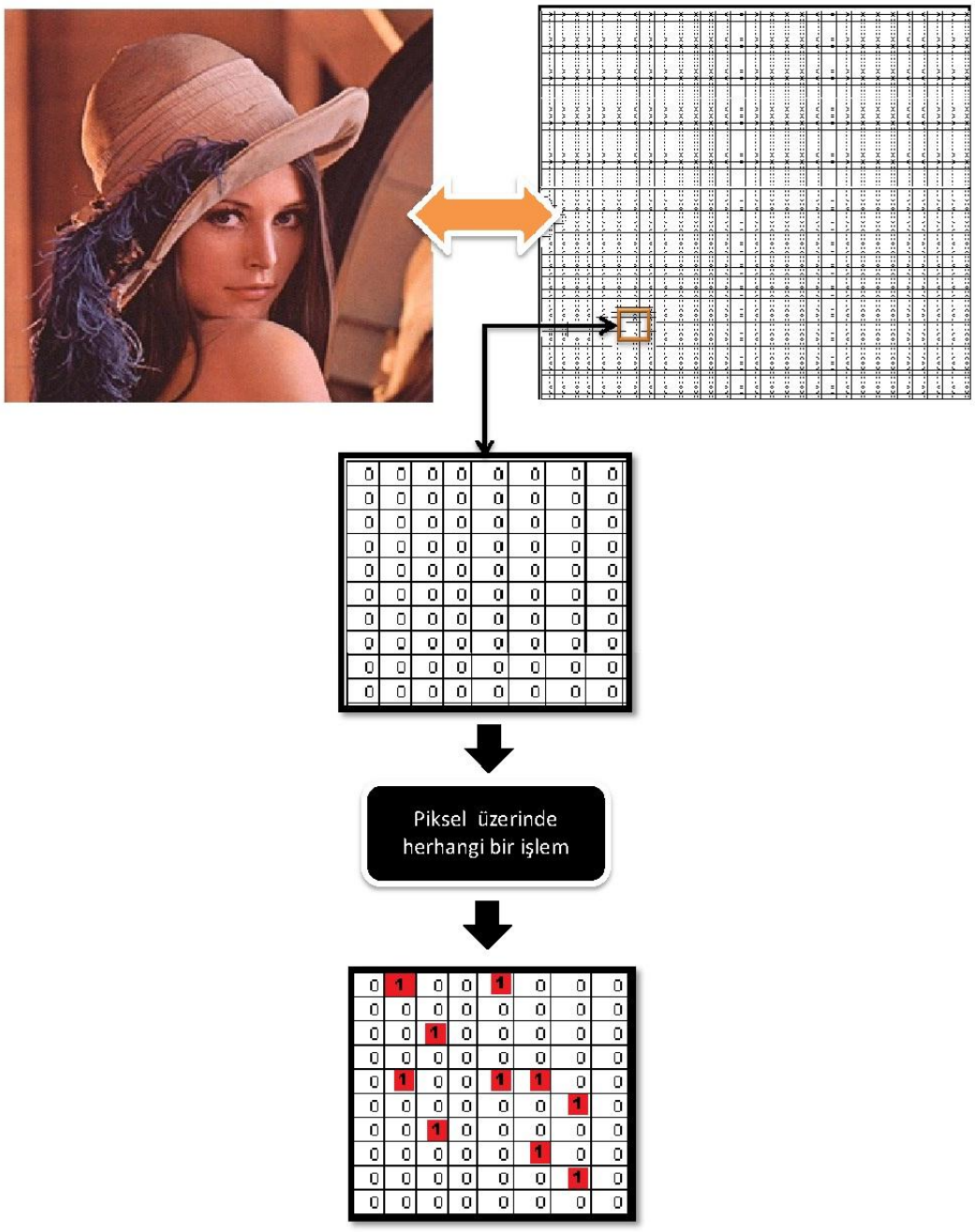
Şekil 4.12. Şifre'den rastlantısal sayı üretimi.

Şifre ve metin girişi tamamlandıktan sonra resim içine mesaj gizleme prosesi ekranı gelmektedir. Bu noktada seçilmiş olan örtü resim orijinal resim kısmında görülebilmektedir. Prosesi başlat çalıştırıldığında ilk aşamada uygulama mesaj uzunluğunu ve tür bilgisini gizleme işlemi için rastlantısal olarak bir piksel noktası belirler bunun için şifre girişi kısmında kullanılan algoritmayı kullanır. Metin uzunluğu ve metin türü belirlenen piksel noktasından başlanarak gizlenir ve uzunluk için R (kırmızı), tür için G (yeşil) kanal kullanılır ve kullanılan her kanalın son iki bitine saklanır. Örneğin Metin uzunluğunun bit olarak 50, uygulamada metin tür tanımının 4 olduğu, gizlenmek istenen bir metni ele aldığımızda Şekil 4.13'de görüldüğü gibi gizleme aşamasının ilk iki işleminde, metin uzunluğu ve tür bilgisi tanımı içeren bitlerde, işlem 1 durum 1 kısmında ilk iki bit okunur, bu bitler durum 2'de R ve G'nin son iki bitlerine yazılır. İşlem 2'de bir sonraki piksel'e geçilir, metin ve tür bitlerinin ikinci iki biti okunur, bir önceki işlemdeki gibi R ve G'nin son iki bitine yazılır. Bu süreç ilgili bitler okunana kadar devam etmektedir.



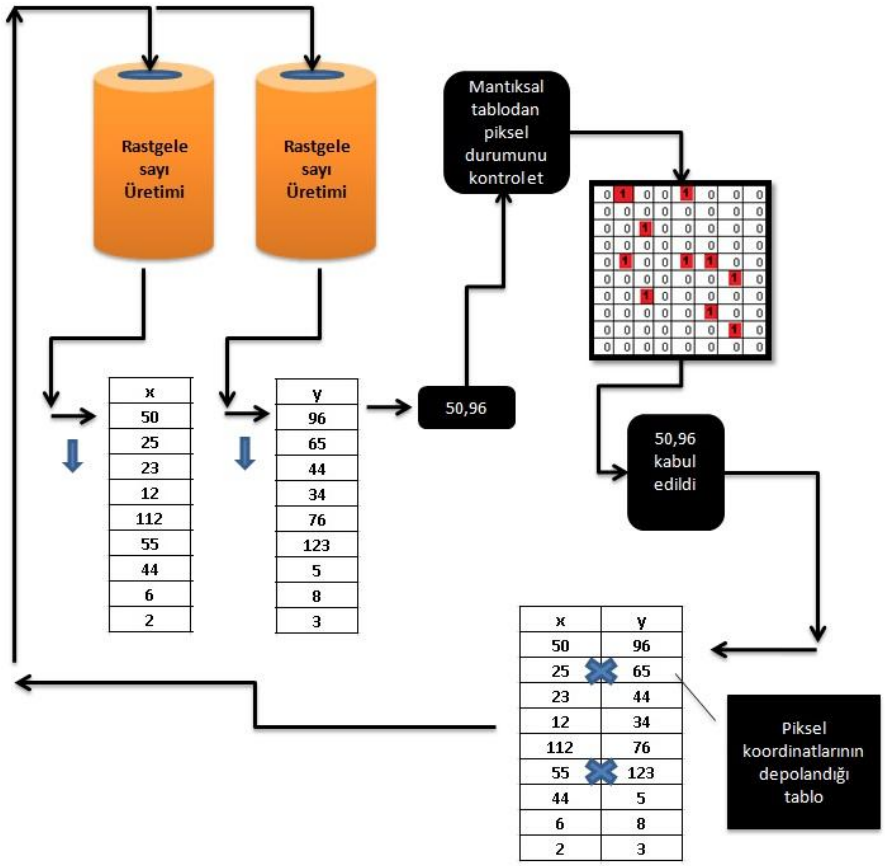
Şekil 4.13. Tür ve metin uzunluk bilgisinin R ve G kanalına gizlenmesi.

Rastlantısal sayı üreten algoritmanın tekrar aynı sayıyı üreterek aynı piksel noktası üzerinde, uygulamaya işlem yaptırmasını engellemek için resim'deki her piksel noktasına karşılık gelecek şekil 4.14'de görüldüğü gibi sadece 0 (yanlış)'lardan oluşan iki boyutlu bir mantıksal harita oluşturulur. Üzerinde işlem yapılan her piksel noktası bu mantıksal haritada 1 (doğru) olarak işaretlenir. Uygulamanın birçok safhasında bu alandan yararlanılmaktadır.



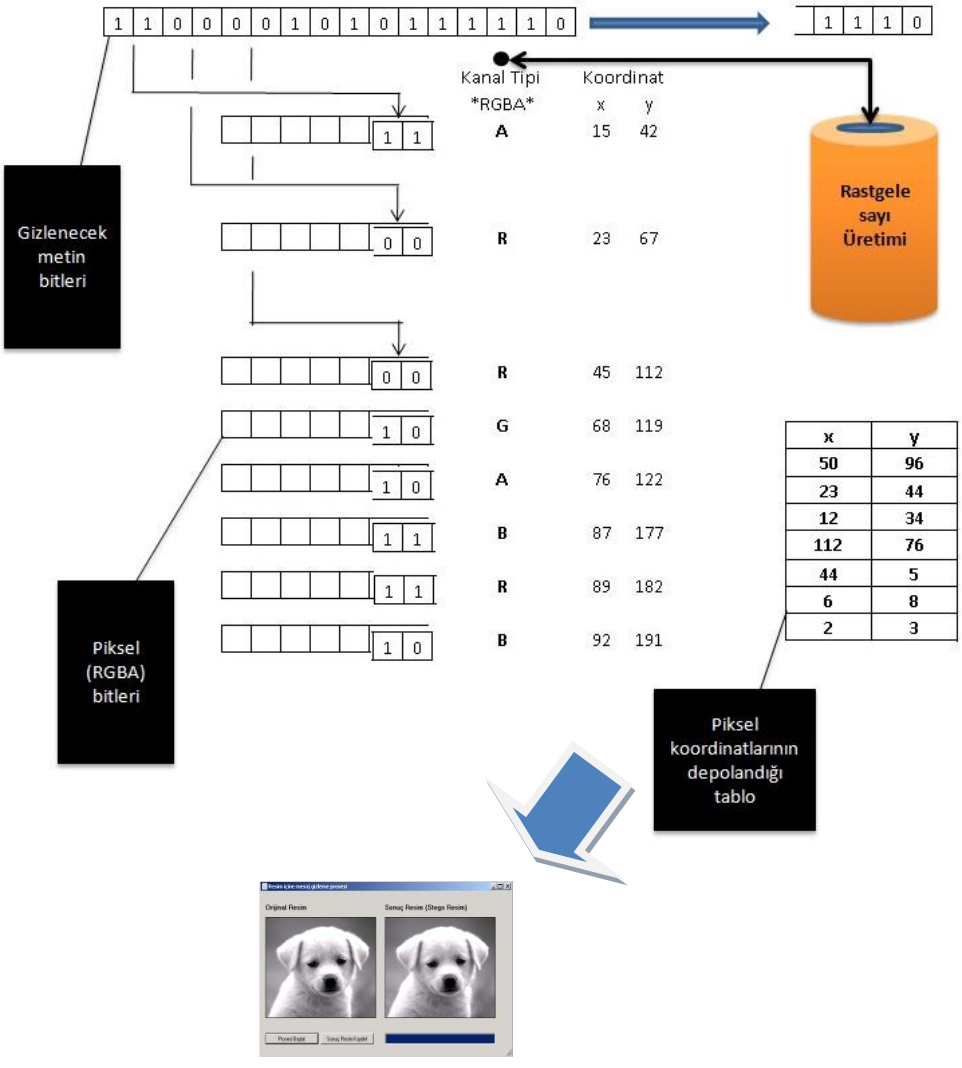
Şekil 4.14. Resim piksellerine karşılık gelen iki boyutlu mantıksal harita.

Metin uzunluğu ve tür bilgisi ile ilgili işlemler yapıldıktan sonra mesajın gizlenmesi kısmına geçilir. Unicode (Evrensel Kod) sistemini desteklemesi için girilen metindeki tek bir karakter 16 bitlik kapasiteye ihtiyaç duyar. Örneğin "¾" karekterinin unicode karşılığı binary olarak "11000010 10111110" olmaktadır. Her piksel unicode karakterin sadece 2 bit'ini gizleyecektir ve böylece metin karakterinin gizlenmesi için metin karakter sayısı 8 ile çarpılacaktır. Bu sonuç uygulama için gerekli olan piksel sayısının hesaplanmasında kullanılır. Bu sayı kadar oluşturulan döngü işlemi, metin bitlerinin gizlenebileceği koordinatların tespitine kadar rastlantısal sayı üreten algoritma'nın kullanımını sağlar. Buna göre örnek olarak verilen "¾" unicode karakteri için 8 piksel yeterli olacaktır. Böylece 16 bitlik bir karakter bilgisinin her iki biti, uygulama tarafından belirlenen pikselerin RGBA kanallarından herhangi birinin son iki bitine yazılacaktır. Gizleme için oluşturulan koordinat sayılarından uygun olanları bulunana kadar mantıksal haritadan kontrol edilir, uygun olanlar Şekil 4.15'de görüldüğü gibi piksel değerleri tablosunda depolanır.

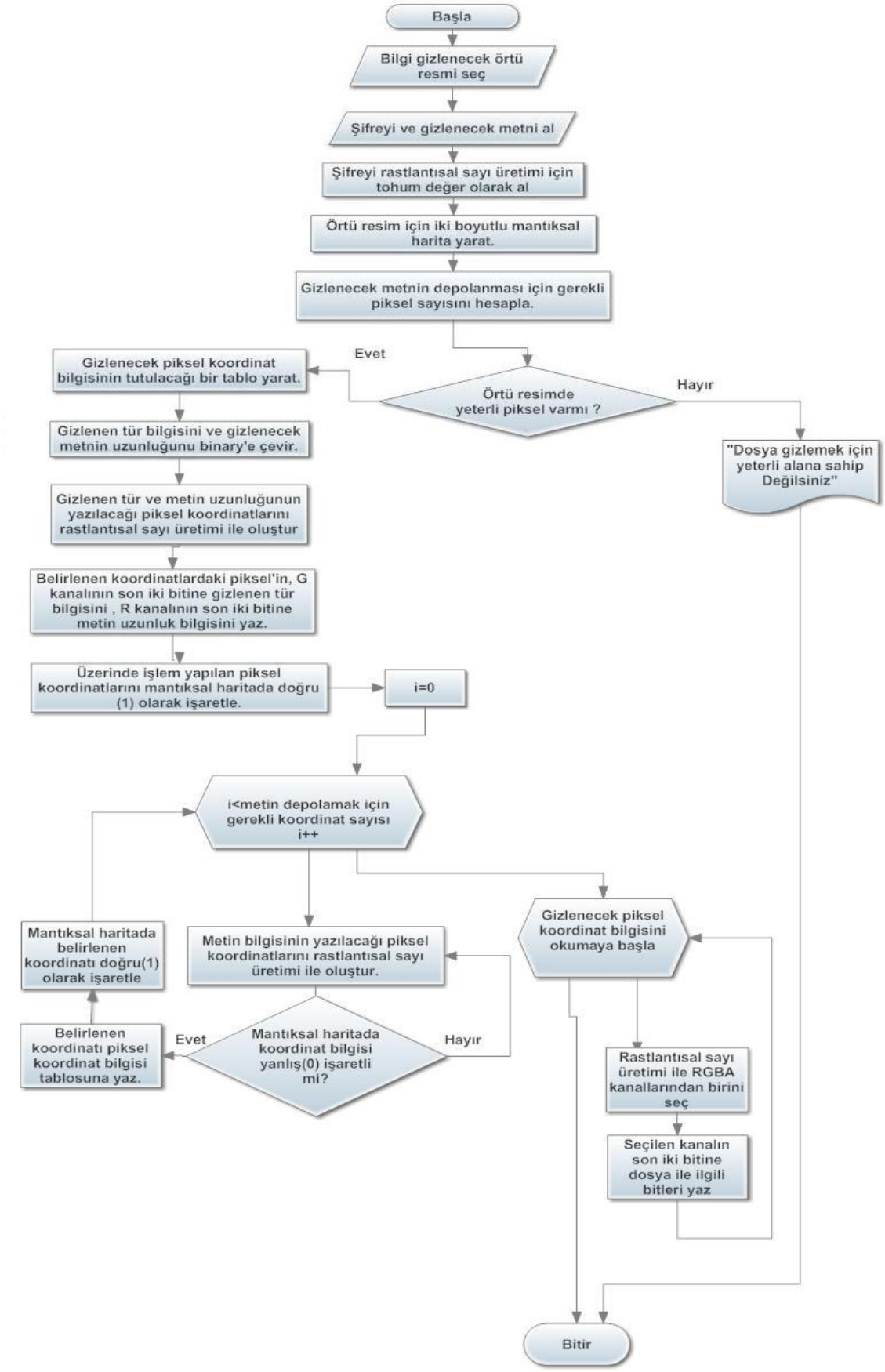


Şekil 4.15. Piksel koordinat değerlerinin bir tabloda toplanması.

Aşağıdaki Şekil 4.16'da görüldüğü gibi binary karakter bit'leri rastlantısal olarak farklı pikseller ve piksel içinde farklı kanallara dağıtılmaktadır. Hangi pikselere veri gizleneceği koordinat değerlerinin tutulduğu tablodaki elamanı sayısı kadar oluşturulan bir döngü içerisinde gerçekleştirilir. Bu tablodan çağrılan pikselin hangi kanalına (RGBA) metin bitlerinin gizleneceği bilgisi ise rastgele sayı üretimi ile tespit edilir. Döngü tamamlandığında uygulama içine metin gizlenmiş olan sonuç uygulama ekranında görüntülenmektedir. Kullanıcı isterse sonuç resmi kayıt altında tutabilir. Bu sürecin uygulamada kullanılan algoritması Şekil 4.17'da görüldüğü gibidir.

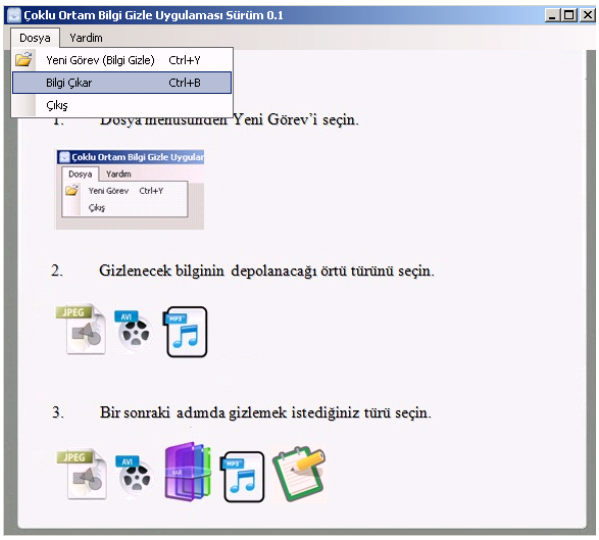


Şekil 4.16. Metin karakterinin farklı piksellere ve kanallara dağıtılması süreci.



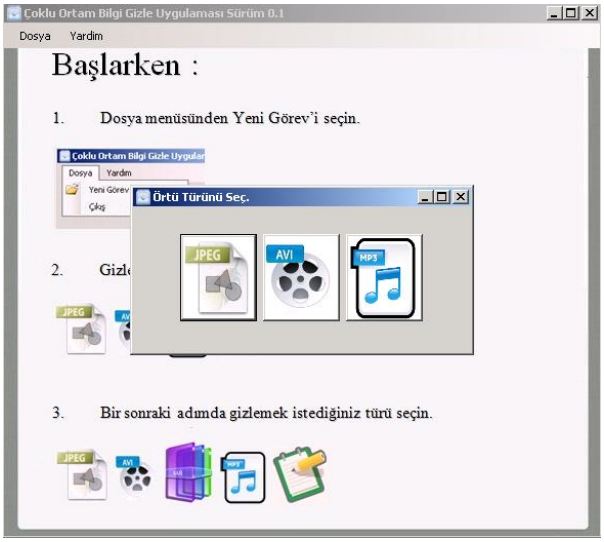
Şekil 4.17. Resim içine metin gizleme algoritmasının akış şeması.

Şekil 4.18'de görüldüğü gibi bir örtü Jpeg dosyasından gizli metni çıkarma için dosya aç menüsünden bilgi çıkar komutu seçildiğinde, Şekil 4.19'daki örtü türünü seç ekranı gelmektedir.



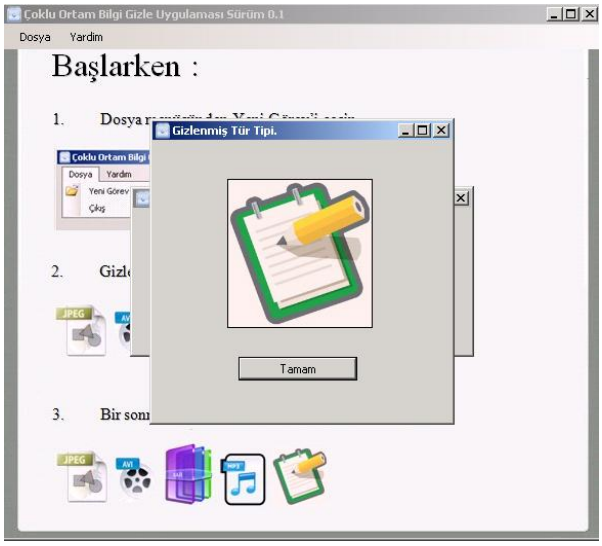
Şekil 4.18. Resim içinden gizli metin çıkarma dosya menüsü ekranı.

Örtü türü seçildikten ve şifre girildikten sonra, uygulama örtü türünün içinde gizlenmiş verinin türünü ve metin uzunluğunu tespit etmelidir. Bunun için rastegele sayı üretimi kullanılarak ilgili koordinat belirlenir. Bir döngü içerisinde tespit edilen bu koordinatın R ve G değerlerinin son iki bitleri okunur.



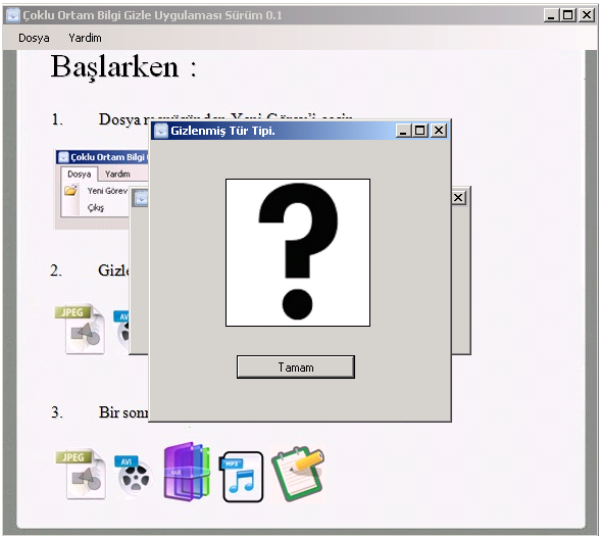
Şekil 4.19. Resim içinden metin çıkarma için örtü türünü seç ekranı.

Şekil 4.20'de görüldüğü gibi bir örtü içinde gizli nesnenin ne olduğu bilgisi gizlenmiş tür tipi ekranında kullanıcıya iletilir.



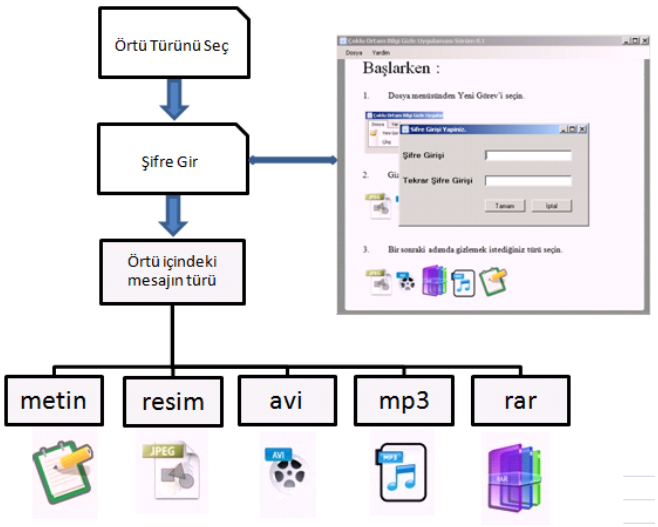
Şekil 4.20. Resim içinden metin çıkarma sürecinde gizli türün tespiti ekranı.

Eğer tür uyumsuzluğu mevcut ise bu durum kullanıcıya Şekil 4.21'de gizlenmiş tür tipi ekranında soru işaret simgesi ile bildirilir. Bununla birlikte çok az ihtimal ile ilgili yanlış şifre girişine karşın doğru bir tür belirleyebilir. Örneğin kullanıcının gizlenmiş bilgiyi çıkarmak için yaptığı yanlış şifre girişine rağmen uygulama bir tür bilgisi çıkarıp, bir sonraki aşamaya geçebilir.



Şekil 4.21. Resim içinden metin çıkarma sürecinde uygunsuz tür uyarı ekranı.

Fakat şifre yanlış olduğundan açılan bilgi anlamsız karakterlerden oluşacaktır. Şekil 4.22'de belirtildiği gibi tür tespitine göre uygulama her tür için farklı bir alt programa yönlenir. Örneğin ilgili mesajın içinde metin türü tespit edildiğinde, uygulama ilgili örtü resim içinden metin çıkarma alt programına geçiş yapar.

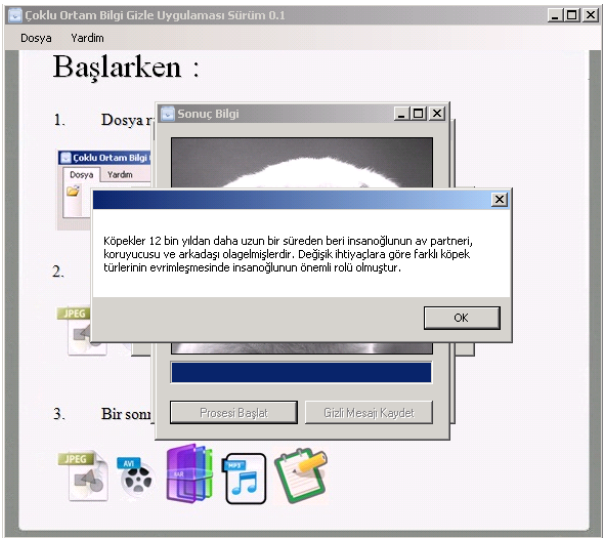


Şekil 4.22. Örtü mesaj türünün uygulamadaki alt dalları.

Girilen şifre tohum değer olarak rastlantısal sayı üretiminde kullanılır . Metin uzunluk bilgisi sekiz ile çarpılır. Çıkan sonuç metin için ne kadar koordinat kullanıldığı bilgisini verir. Şekil 4.32'deki ekranda Prosesi başlat tıklantığında metin içinden gizli metin çıkarma işlemi başlar.

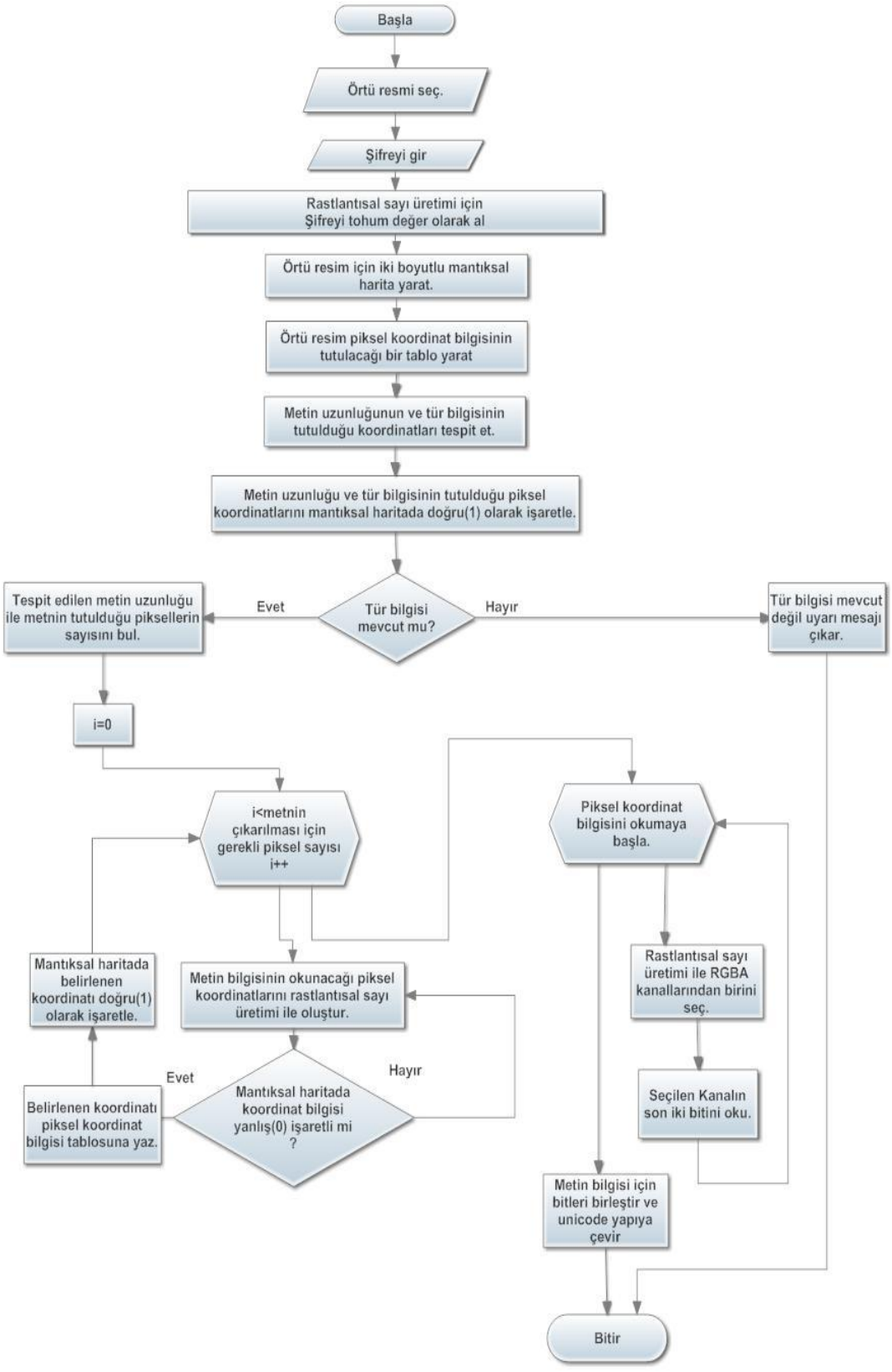
Şekil 4.23. Resim içinden metin çıkarma sürecinde gizle metin çıkarma ekranı.

Bu işlem içinde bitlerin okunması haricindeki kısımlar metin gizleme işlemi kısmında meydana gelen işlemler ile tamamen aynıdır. Okuma resim içine metin gizleme işleminin tersidir. Süreç tamamlandığında Şekil 4.24.'teki gibi gizlenmiş metin görüntülenir .



Şekil 4.24. Örtü resim'den çıkarılan metin örneği ekranı.

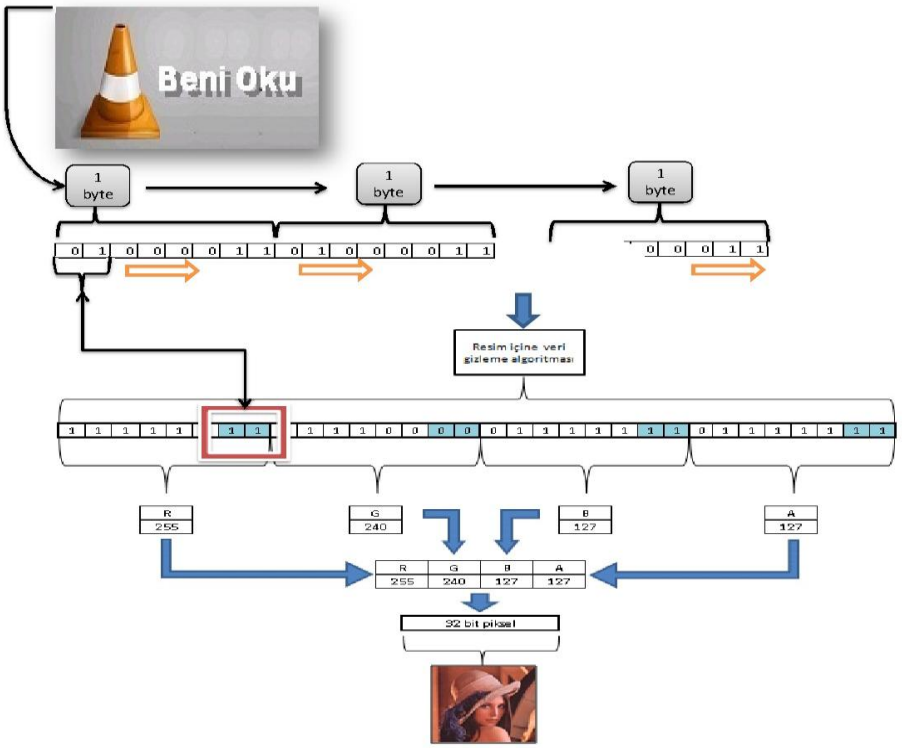
Şekil 4.25.'te resim içinden metin çıkarma'da uygulanan algoritmanın akış şeması verilmektedir.



Şekil 4.25. Resim içinden metin çıkarma algoritmasının akış şeması.

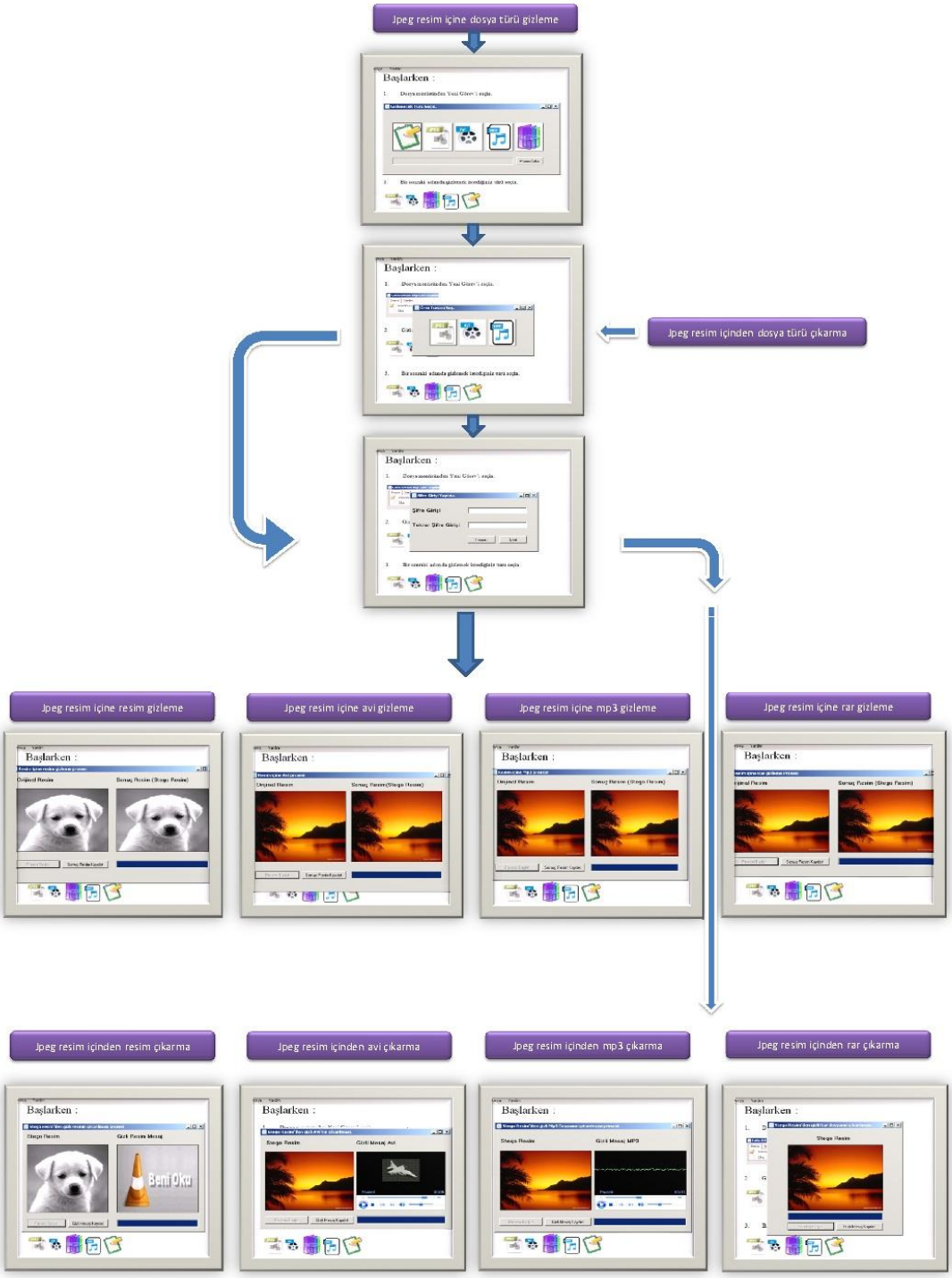
4.1.2. Resim Dosyasına Diğer Dosya Yapılarını Gizleme ve Çıkarma

Metin gizleme ve çıkarma algoritmasında üzerinde bazı küçük geliştirmeler ve değişiklikler yapılarak farklı dosya yapıları resim içine gizlenebilmektedir. Uygulama kapsamında ise bu türler jpeg, mp3 ,avi ,rar gibi en popüler farklı dosya türlerinden seçilmiştir. Bununla birlikte örtü içine metin dışındaki bilgi gizleme yada bilgi çıkarma işlemlerinin sonuçlandırılma süresi, işlemde kullanılan dosyaların boyutlarına bağlı olmakla birlikte, kullanılan bilgisayar işlem gücünede bağlıdır. Uygulamada resim içine dosya türü gizleme işlemi için, örtü türü olarak JPEG resim türü seçilir ve gizlenecek tür seçim ekranı açılır. Gizlenecek tür seçiminden sonra uygulama gizlenecek dosya yapısı için gerekli olan piksel sayısını hesaplar. Bu nokta, metin gizleme sürecindeki gerekli piksel sayısını tespit yönteminden farklı bir yaklaşım içerir. Bu bilgiyi elde etmek için dosyanın toplam bit sayısı tespit edilir. Bu sayı ikiye bölünür. Çünkü dosyanın her iki biti için bir piksel yeterli olacaktır. Örneğin Şekil 4.26'da görüldüğü gibi tür olarak bir resim dosyası seçilmiştir. uygulama dosyanın tüm bitlerini hafızaya alır, daha sonra metin içine gizleme ile aynı veri gizleme algoritmasını kullanır ve örtü bitin verilerine gizleme işlemini yapar .

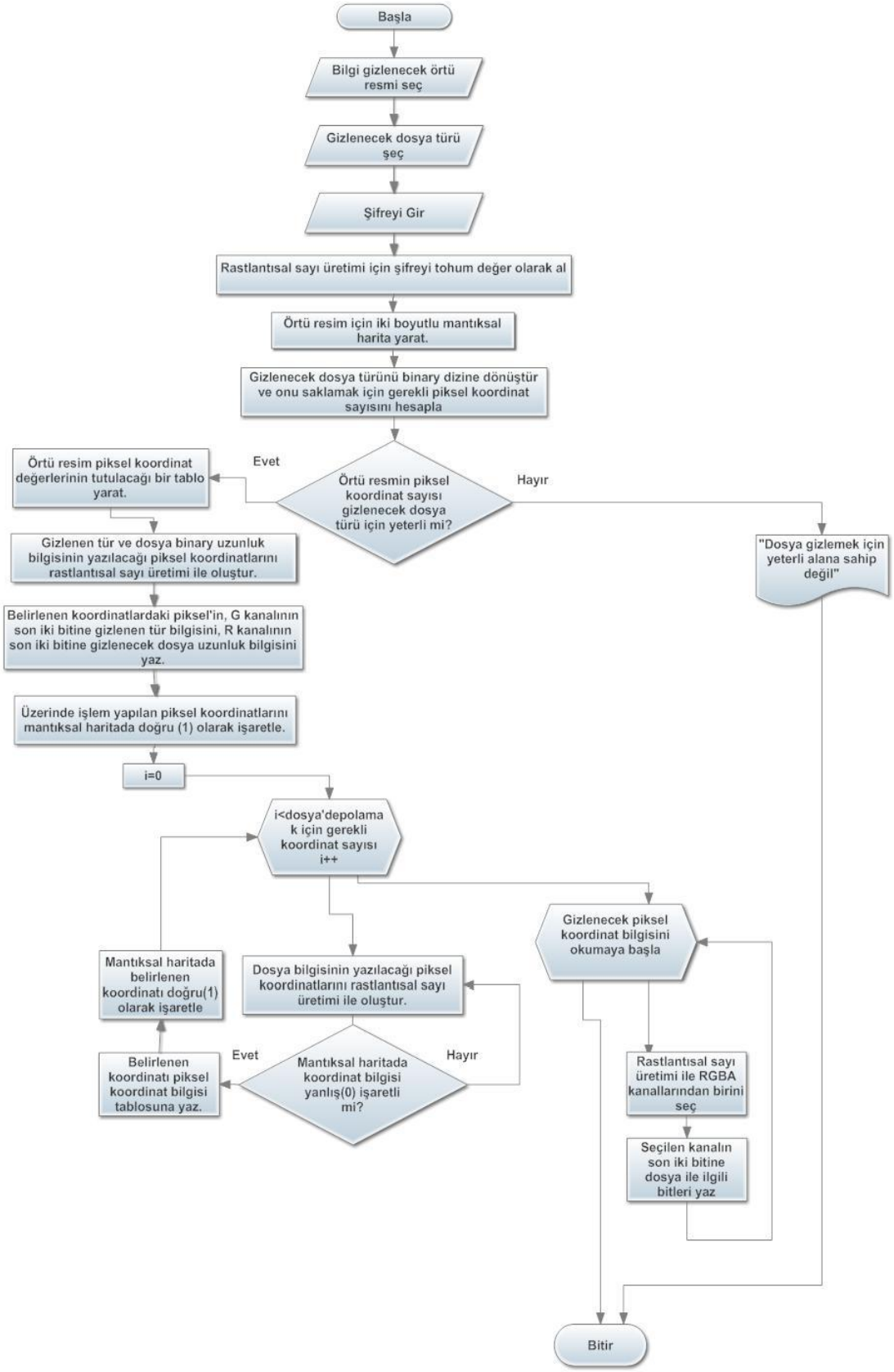


Şekil 4.26. Farklı dosya türlerini resim içine gizleme süreci.

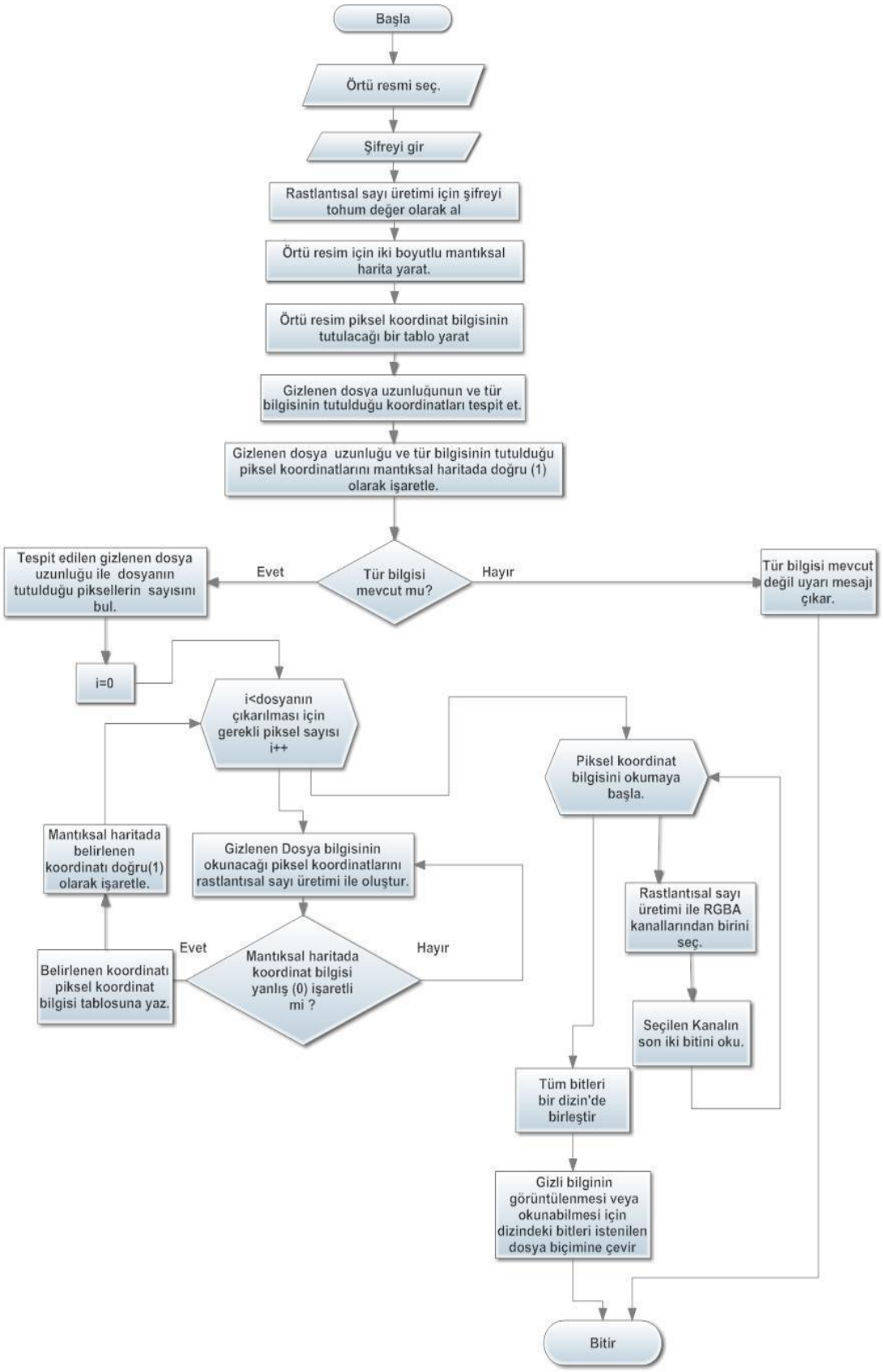
Uygulamada resim olarak Jpeg dosya türüne farklı dosya türlerini gizleme ve çıkarma süreçleri kademeli olarak Şekil 4.27'de özetlenmektedir. Şekil 4.28.'de Jpeg resmi içine dosya türü gizleme ve şekil 4.29'da ise resim içinden dosya çıkarma algoritmalarının akış şemaları gösterilmektedir.



Şekil 4.27. Resim içine farklı dosya türlerinin gizleme ve çıkarma ekranları.



Şekil 4.28. Resim içine farklı dosya türlerini gizleme algoritması akış şeması.



Şekil 4.29. Resim içinden gizli dosya türlerini çıkarma algoritması akış şeması.

4.1.3. Video Dosyasına Metin Gizleme ve Çıkarma

Avi dosyalarına erişim, okuma ve yazma işlemleri için Windows API fonksiyonlarından yararlanılmıştır (Öztürk, 2009:28). Şekil 4.30.'da görüldüğü gibi kullanıcı avi içine metin gizleme işlemini gerçekleştirmek için dosya aç menüsünden yeni görev (bilgi gizle)'i seçtikten sonra beliren örtü türünü seç ekranından avi simgesini seçer.

Şekil 4.30. Avi içine metin gizleme sürecinde örtü türü seç ekranı.

Örtü türünün içine gizlenecek tür seçimi Şekil 4.31'de görülen istenen tür ekranında yapılır.

Şekil 4.31. Avi içine metin gizleme sürecinde gizlenecek tür seçim ekranı.

Metin türü seçimi yapıldıktan sonra, şifre ve metin girişi ekranı gelir. Metin unicode yapıya uygun binary şekle çevrilir. Avi'nin tüm çerçeve özellikleri belirlenir ve avi piksel kapasitesi hesaplanır. Avi çerçeveleri bitmap türünde bir dizine depolanır. Şifreyi tohum değer olarak alan bir rastlantsısal sayı üreteci nesnesi oluşturulur. Tür ve metin uzunluk bilgisinin depolanacağı Bitmap türünde dizi değişkendeki çerçeveler üzerinde sağlıklı işlem yapabilmek için de bir mantıksal tablo oluşturulur. Gizlenecek metin için çerçeve sayısı hesaplanır. Metni gizlemek için yeterli çerçeve yoksa uygulama ilgili uyarı mesajını verir.

Yeterli çerçeve mevcutsa, gizlenecek tür ve metin uzunluğu tamsayı değerleri binary sayı sistemi şekline çevrilir. Random nesnesi ile gizlenecek tür ve metin uzunluğu bilgisinin depolanacağı piksellerin tespiti yapılır. Belirlenen piksel koordinatlarının G kanalının son iki bitine gizlenecek tür bilgisini, R kanalının son iki bitine metin uzunluk bilgisi yazılır. Bu tür bilgilerin bulunduğu çerçevelerin mantıksal tablosunda doğru(1) olarak işaretlenir. Değişiklik yapılmış çerçeve de depolanmış avi dizinindeki yerine geri yüklenir.

Metnin gizlenmesi için gerekli çerçeve sayısı kadar rastlantısal sayı üretimi sonuçları kullanılarak avi bitmap dizi değişkeni çerçeve elemanlarını çağırır. Belirlenen elemanlar mantıksal tablodan kontrol edilir. Üzerinde işlem yapılan çerçeveler doğru olarak işaretlenir, böylece aynı çerçeve üzerinde tekrar işlem yapılması engellenmiş olur. Her çerçeve için bir piksel koordinat mantıksal tablosu oluşturulur. Çünkü rastlantısal sayı üretiminin oluşturacağı sayıların tekrarlanmamasının garantisi yoktur.

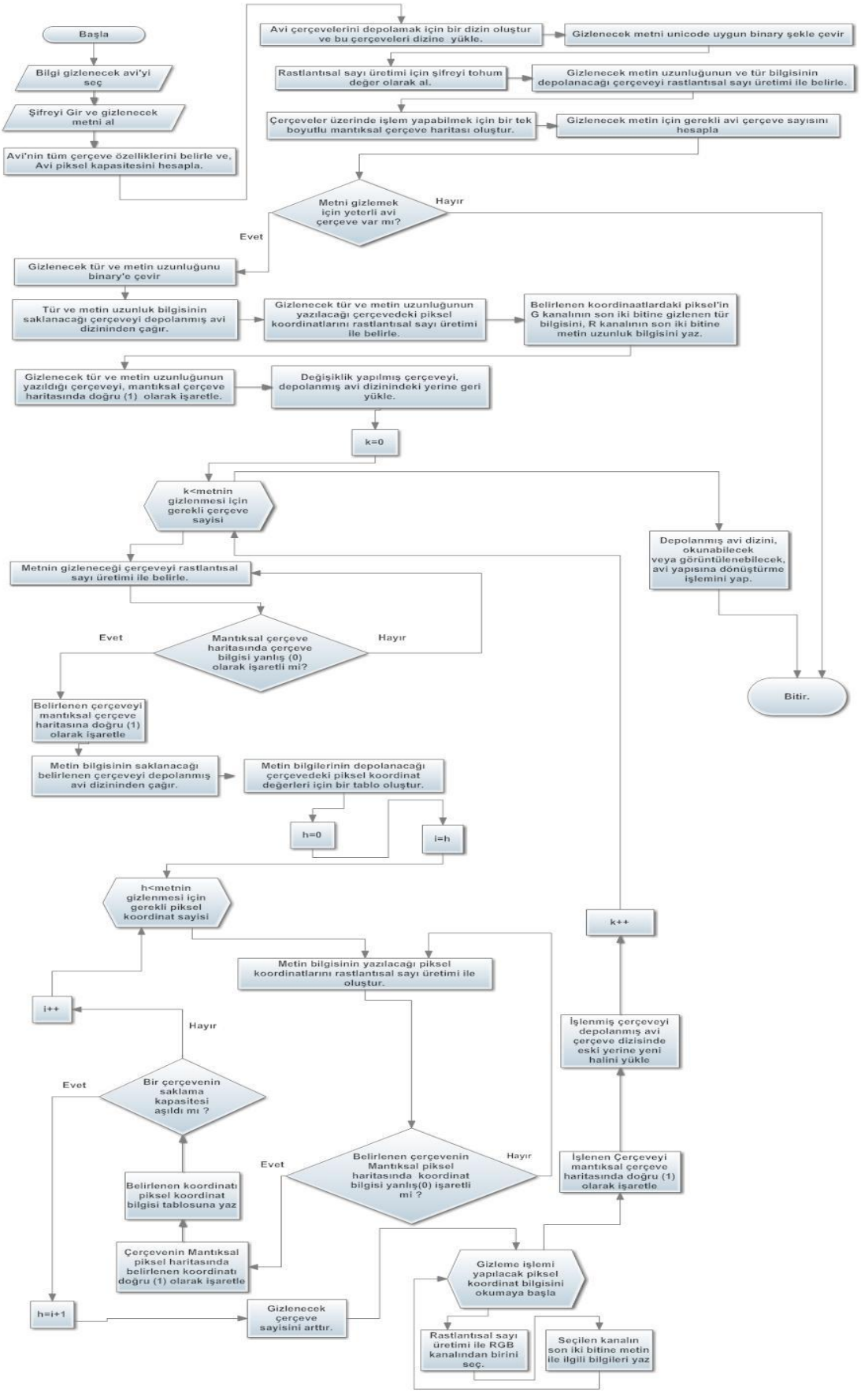
Çağrılan çerçeve içinde metin bitlerinin gizleneceği piksel koordinatlarının tutulacağı bir tablo oluşturulur. Böylece rastlantısal seçilen piksellerin koordinat değerleri bu nesneye depolanır. Depolanan noktalar mantıksal piksel çerçeve tablosunda doğru olarak işaretlenir. Bu noktalar oluşturulan Şekil 4.32'deki döngüde okunarak piksellerin son iki bitlerine işlenir.

```
foreach (Point p in birCerceninDegerleri)
{
.....c = bitmapFrameIcin.GetPixel(p.X, p.Y);
.....tohumSifreNesnesi.Next(1, 4);
switch (bandDegeri)
                                        {
                                        ............GizlenecekTur.getBits(c.R);
                                        case 1:
                                         {
                                        ....Remove(pixelValueBits.Length - 1, 1);
                                         ....Insert(pixelValueBits.Length, msgBits
                                         ....Remove(pixelValueBits.Length - 2, 1);
                                         ....Insert(pixelValueBits.Length - 1, msgE
                                         }
                                        case 2:
                                         {
                                        ..............
                                         }
                                        case 3:
                                        {
                                        ..............
                                         }
                                        default:
                                        MessageBox.Show("sonraki sürümler");
                                         break;
                                         }
}
```

Şekil 4.32. Piksellerin bir döngü içeresinde işlenmesi kod örneği.

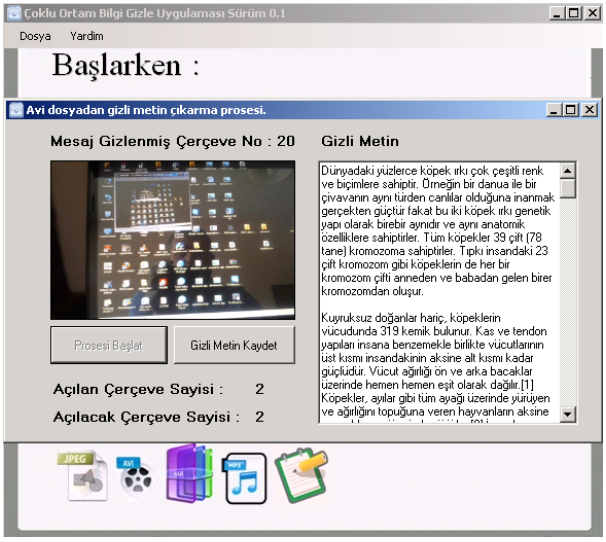
Bir çerçeve, gizleme kapasitesine ulaştığında uygulama geri kalan bilgileri işlemek için rastlantısal olarak belirlenmiş bir sonraki çerçeveye gider. Döngü metin bitlerinin belirlenen çerçevelere tamamen depolanmasıyla sonlanır. Çerçeveler, ilk başta bahsedilen API kütüphanesi yardımıyla yürütülülebilir avı biçimine döndürülür. Uygulama ayrıca bit'lerin gizlendiği çerçeveleri süreç sırasında göstermekle birlikte, işlem sonunda kullanıcı gizli bilgiyi taşıyan avi'yi Şekil 4.33.'te görüldüğü gibi oynatmakta isterse kayıt altına alabilmektedir. Şekil 4.34.'te Avi içine metin gizleme algoritma akış şeması gösterilmektedir.

Şekil 4.33. Avi'ye metin gizleme süreci ekranı.

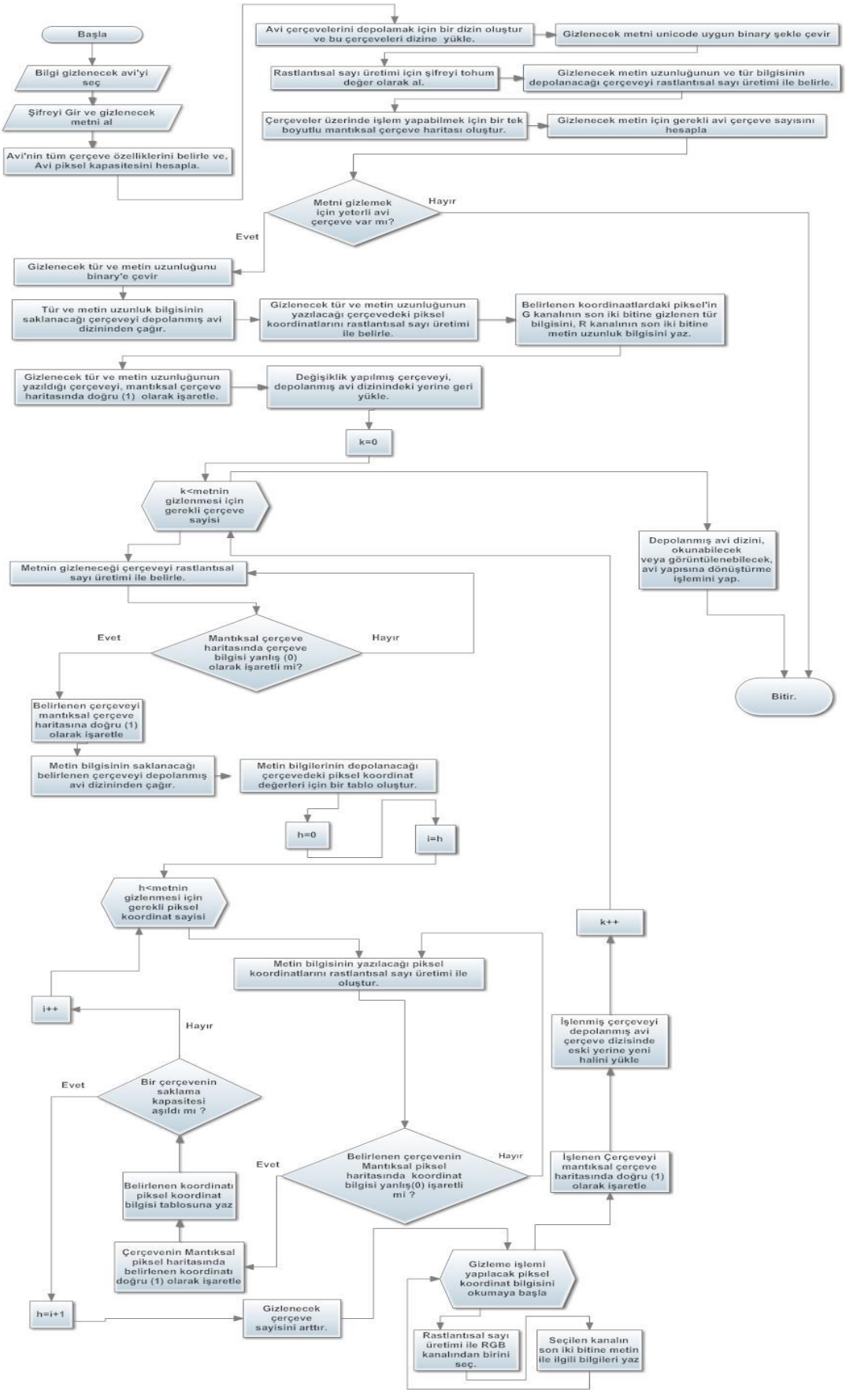


Şekil 4.34. Avi içine metin gizleme algoritması akış şeması

Avi içinde gizlenmiş metni çıkarma işleminde, içinde metin gizlenmiş avi dosyası açıldığında uygulama bir şifre sorar. Alacağı şifreyi rastlantısal sayı üretimi için tohum değer olarak verir ve ilk aşamada tür, metin uzunluk bilgisinin bulunduğu çerçeveyi tespit eder. Tür bilgisinde eğer uyumsuzluk varsa kullanıcıya bir ekran ile bildirilir. Metin uzunluk bilgisi ile metnin gizlenmiş olduğu çerçeve sayısı tespit edilir. Bundan sonraki süreçler avi içine metin gizleme aşamaları ile genel olarak aynıdır. Sadece ilgili piksellerin son iki bitlerine gizlenen veriler, bu kısımda tam tersi bir işlem ile hafızada depolanır ve Şekil 4.35'de görüldüğü gibi okunabilir bir forma dönüştürülür. Şekil 4.36'da Avi içine metin çıkarma algoritması akış şeması gösterilmektedir.



Şekil 4.35. Avi'den gizli metin çıkarma prosesi ekranı.



Şekil 4.36. Avi içinden metin çıkarma algoritması akış şeması.

4.1.4. Video Dosyasına Diğer Dosya Yapılarını Gizleme ve Çıkarma

Geliştirilen uygulama Avi içerisine farklı dosya biçimlerinin gizlenmesine olanak sağlar. Resim içine metin gizleme ve çıkarmada kullanılan algoritma mantığı bazı küçük değişiklikler ile bu kısımda da kullanılmıştır. Avi içine resim gizleme işleminde, prosesi başlat ikonu tıklandığında, gizlenecek avi çerçeveleri işlem sürecinde veri gizlenen çerçeve no penceresinde anlık olarak şekil 4.37'de görüldüğü gibi kullanıcıya belirtilmektedir.

Şekil 4.37. Avi içine resim gizleme işlemi ekranı.

Avi'den gizli resmin çıkarılması işlemi sürecinde, içinden gizli veri çıkarılan avi çerçeveleri şekil 4.38'de görüldüğü gibi yine mesaj gizlenmiş çerçeve no altındaki pencerede anlık olarak belirtilmektedir.

Şekil 4.38. Avi'den gizli resim çıkarma işlemi ekranı.

Avi içine avi gizleme işlemi ekranında, gizlenecek avi kullanıcı tarafından görüntülenebilir. Prosesi başlat ikonu tıklandığında veri gizlenen çerçeveler işlem sırasında gösterilir. İşlem bittiğinde sonuç avi kısmında içine veri gizlenmiş avi Şekil 4.39'daki görüntülenmekte yada kayıt edilebilmektedir.

Şekil 4.39. Avi içine avi gizleme işlemi ekranı.

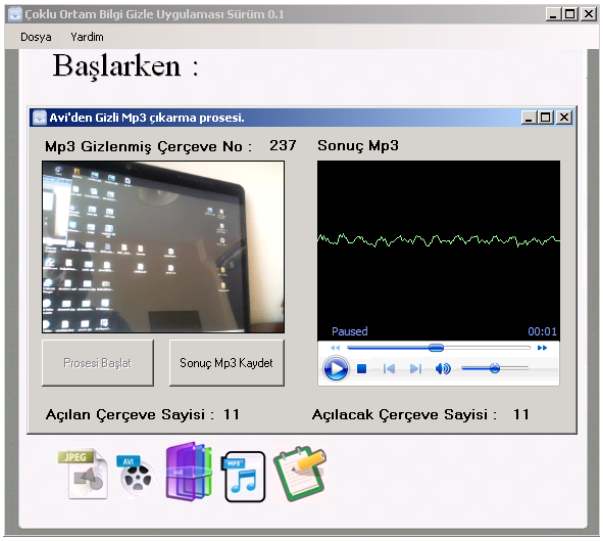
Avi'den gizli avi'nin çıkarılması işlemi ekranında, seçilen avi dosyası içindeki verilerin gizlenmiş olduğu çerçeve sayısı ve görüntüsü anlık olarak belirtilmektedir. İşlem bittiğinde, Şekil 4.40.'de görüldüğü gibi sonuç avi kısmında gizlenmiş avi oynatılabilmektedir.

Şekil 4.40. Avi'den gizli avi çıkarma işlemi ekranı.

Avi içine mp3 gizleme işlemi ekranında, prosesi başlat ikonu tıklandığında veri gizlenen çerçeveler tanımı altında çerçeve numarası ve görüntüsü kullanıcıya belirtilir. İşlem bittiğinde sonuç avi kısmında içine veri gizlenmiş avi Şekil 4.41.'deki gibi görüntülenebilir yada sonuç avi kaydet ikonu seçilerek kayıt ortamına depolanabilir.

Şekil 4.41. Avi içine mp3 gizleme işlemi ekranı.

Şekil 4.42'de görüldüğü gibi Avi'den gizli mp3'ün çıkarılması işlemi ekranında seçilen avi dosyasındaki verilerin gizlenmiş olduğu çerçeve sayısı anlık olarak gösterilmekte ve sonuçlanan mp3 depolama ortamına kayıt edilebilmektedir.

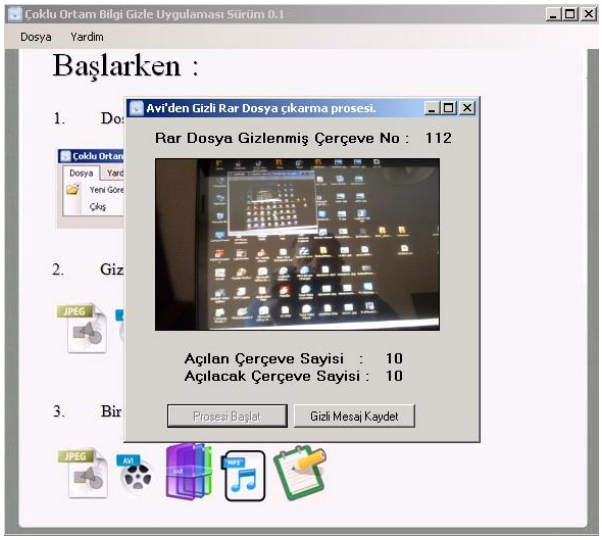


Şekil 4.42. Avi içinden mp3 çıkarma işlemi ekranı.

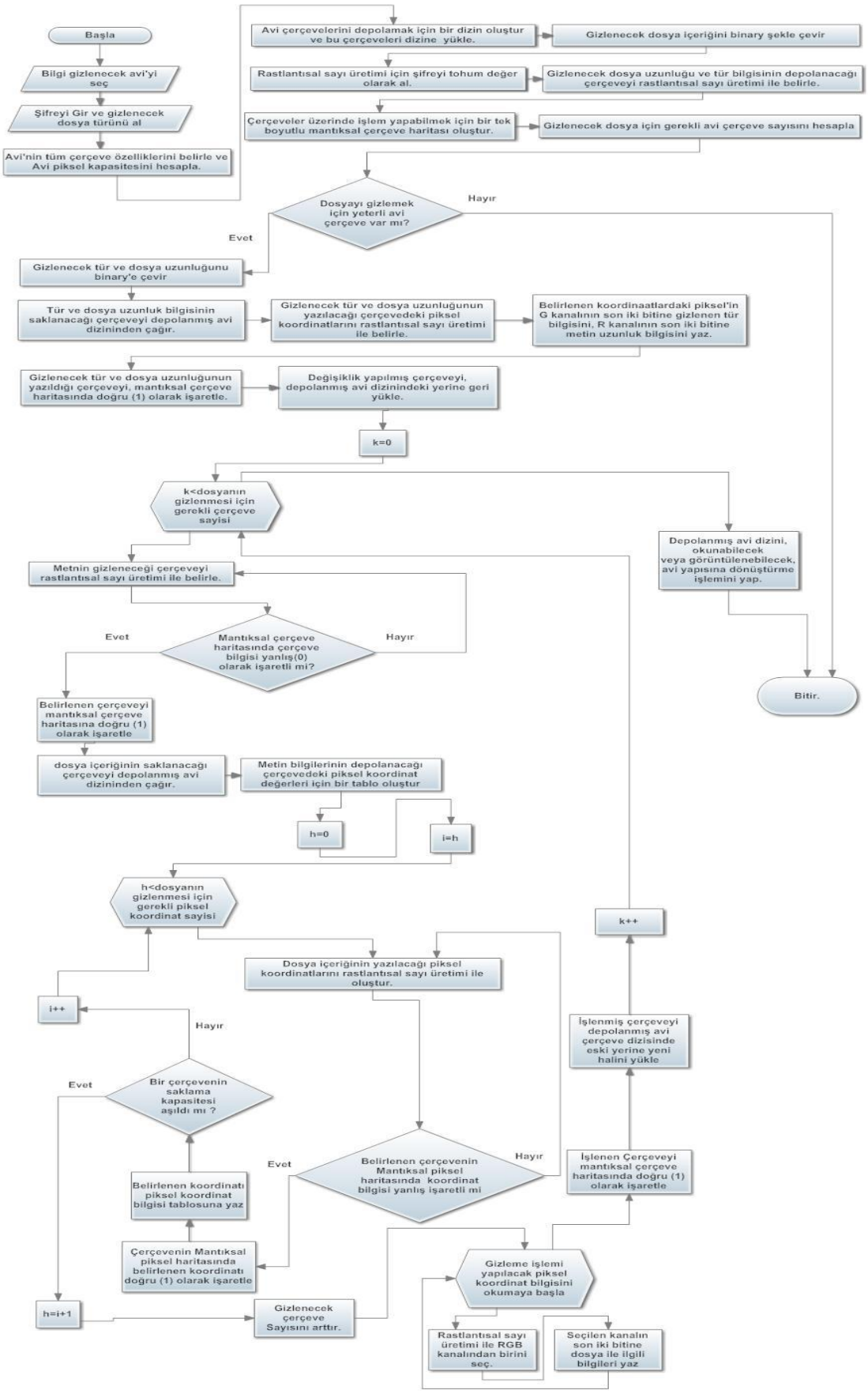
Avi içine rar gizleme işlemi ekranında, prosesi başlat ikonu tıklandığında rar verilerinin gizleneceği çerçeveler işlem sırasında kullanıcıya gösterilir. İşlem bittiğinde, sonuç avi altındaki pencerede içine rar verisi gizlenmiş avi şekil 4.43'teki gibi ekranda belirir.

Şekil 4.43. Avi içine rar gizleme işlemi ekranı.

Şekil 4.44'teki ekranda Avi'den gizli rar'ın çıkarılması işlemi ekranında seçilen avi dosyasındaki verilerin gizlenmiş olduğu çerçeve sayısı anlık olarak gösterilmektedir. İşlem bittiğinde gizli mesaj kaydet ikonu seçilerek avi'den çıkarılmış rar dosya, kayıt ortamına depolanabilir. Şekil 4.45'te Avi içine farklı dosya türlerini gizleme ve şekil 4.46'da farklı dosya türlerini çıkarma algoritmalarının akış şemaları gösterilmektedir.

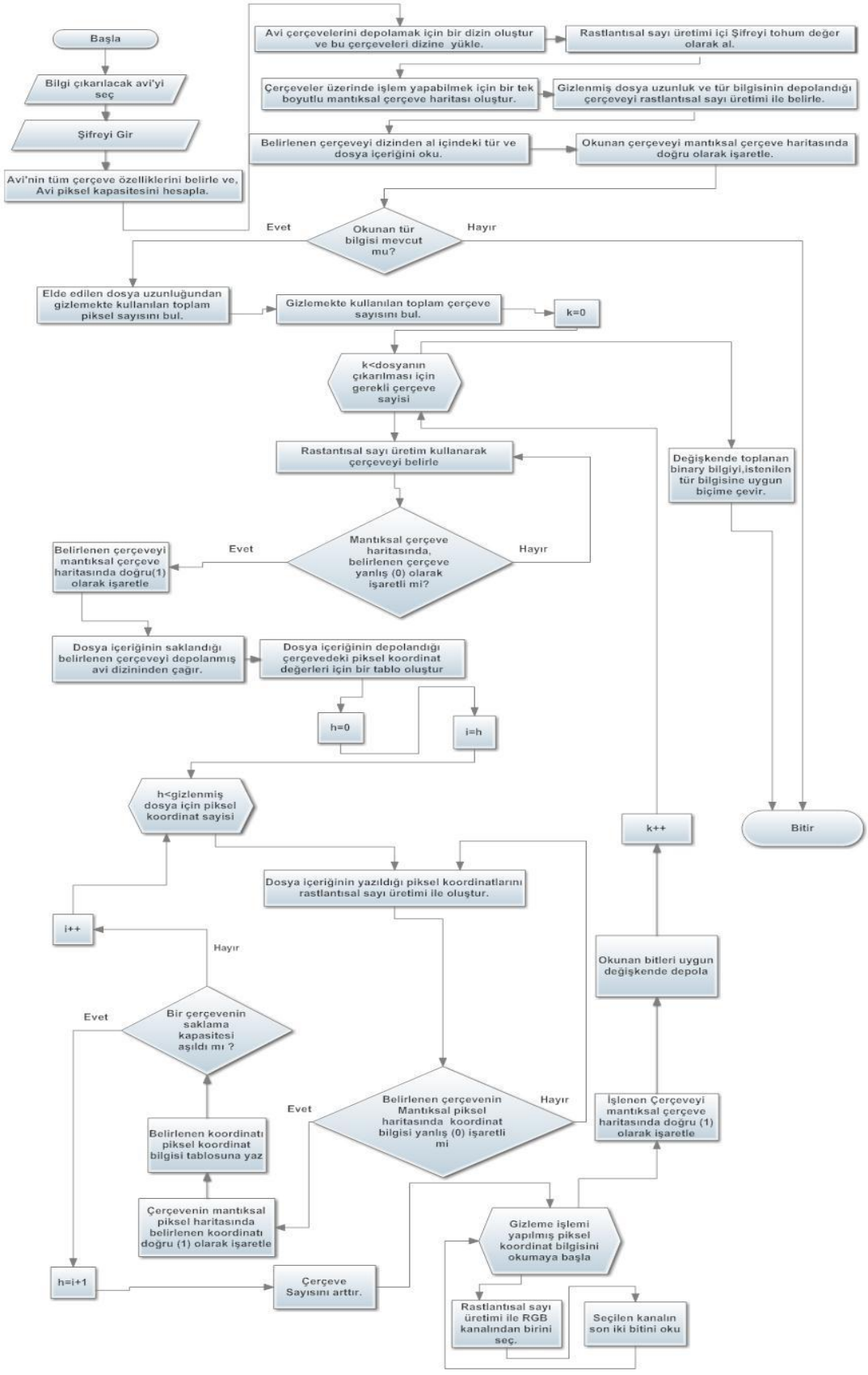


Şekil 4.44. Avi içinden rar çıkarma işlemi ekranı.



Şekil 4.45. Avi içine farklı dosya türlerini gizleme algoritmasının akış şeması.

.



Şekil 4.46. Avi içinden farklı dosya türlerini çıkarma algoritmasının akış şeması.

4.2. Ses Dosyasına Bilgi Gizleme ve Çıkarma

4.2.1. Ses Dosyalarına Bilgi Gizleme

Ses verisine bilgi gizleme, görüntü dosyalarına bilgi gizleme yöntemlerinde olduğu gibi insan algısal sisteminin zayıflıklarını kullanmaktadır. Son yıllarda ses verisine bilgi saklama oldukça popülerdir ve bu konuda çalışmalar devam etmektedir. Bu çalışmalarda mp3 tercihi, boyutunun küçük olabilmesi, bilgi gizleme uygulamaları için bir taşıyıcı olarak seçilmesinde önemli bir faktördür (Atıcı, 2007:17). İkinci önemli faktör, WMA gibi patentli bir biçim üzerinde yapılacak tasarım değişiklikleri veya geliştirmelerden doğacak yasal risk'in, Mp3 için söz konusu olmamasıdır. Çünkü mp3 geliştirilmeye açık bir standarttır (Maciak ve diğ., 2006).

Mp3 biçim, resimde saklanan sanal bilgiden farklı olarak kodlanmış ses verisinin depolanması için tasarlanmıştır. Bu nedenle görüntü steganografi yöntemleri ile ses verisi üzerinde istenilen sonuçlara ulaşmak zorlayıcı olabilir. Sonuçta mp3, stegonografik veriler için en iyi örtü dosyaları değildir. Yapılan çalışmalar genelde LSB modifikasyonu ve dönüştürme teknikleri şemsiyesi altındadır. Bununla birlikte kullanılabilen yöntemler üç kategoriye ayrılabilir: Steganografik olmayan (non-steganography), kodlama zamanı mp3 steganografisi (Encode time mp3 steganography), gönderi kodlama mp3 steganografisi (Post encoding mp3 steganography) yöntemleri (Maciak ve diğ., 2006).

Steganografik olmayan yöntemlerde, ID3v2 etiketleri ya da Lyrics3 denilen ID3'ın özel uzantısı kullanılmaktadır. ID3v2 etiketleri, metin gizlemek için sınırlı desteğe sahiptir. Bu yöntem çok verimli değildir çünkü şarkı sözleri gibi metin gizleme işleminde metin'in nasıl biçimlendirileceğine, yazı çerçeveleri boyunca onların nasıl bölüneceğine ait talimatlar yoktur. Ayrıca, bu evrensel bir yaklaşım değildir çünkü tüm mp3 dosyaları ID3v2 etiketleri kullanmamaktadır (Maciak ve diğ., 2006). Ne yazık ki, bu yöntem uygulamanın hedefi için uygun olmayıp ve steganografi ile bağdaştırılabilecek yapıya sahip değildirler. Hedefimiz, mp3 içine metin gömebilen (gizli bir mesaj, şarkı sözleri gibi tamamen kullanıcının tercihinde olan) mümkün olduğunca steganografik ilkelere yakın bir geliştirme yapmaktır.

Kodlama zamanı mp3 steganografisi, mp3 kodlama prosesi sırasında yapılarak yürütülmektedir. Bununla birlikte bazı önemli kusurlara sahiptir. Kodlama zamanı metodlarından birini uygulamak için bir araştırmacının ya var olan mp3 kodlayıcısını modifiye etmesi ya da tekrar oluşturması gerekmektedir. Fabien Petitcolas tarafından yazılan Mp3stegoc yazılımı kodlama zamanı steganografisinin bir büyük örneğidir (Terminaly-incoherent, 2012).

Kodlama zamanı mp3 steganografisi, düşük bit kodlama (Low Bit Encoding), faz kodlama (Phase Coding), yayılmış spectrum kodlama (Spread Spectrum Coding), eko veri gizleme (Echo Data Hiding) metodlarını içermektedir (Maciak ve diğ., 2006).

Düşük bit kodlama, kodlama zamanında LSB yapılması yaklaşımıyla mümkündür. Gizlenmiş bilgi daha sonra, her bir çerçevenin Huffman kodunu çözerek ve LSB'yi açarak geri döndürülebilir. Bu yöntem etkili iken, müzik verileri içine gürültüyü önemli düzeyde arttırmaktadır (Maciak ve diğ., 2006). Mp3 veri çerçeveleri Huffman kodlama verisinden oluştuğundan dolayı, gerçek LSB steganografisi başarımını elde etmek zor olmaktadır. Bununla birlikte bu yaklaşım sesin analog ortamlara girmeksizin, tamamen sayısal ortamlarda transferi durumunda kullanılabilir.

Faz kodlama yöntemi de resim dosyalarında uygulanan jpeg algoritması benzeri bir yapı taşımaktadır. Gizleme işleminde örtü ses dosyası küçük segmentlere bölünerek, bu segmentlerdeki faz değerleri hesaplanır ve gizlenecek bilgiye ait faz referansı ile değiştirilir (Kutucu ve Kaya, 2002). Böylece veriyi saklamak için segmentlere ait faz değeri, yeniden şekillendirilmektedir. Bu yöntem resim içine veri saklama yöntemlerinden, dönüştürme yöntemine benzemektedir (Atıcı, 2007:15-16).

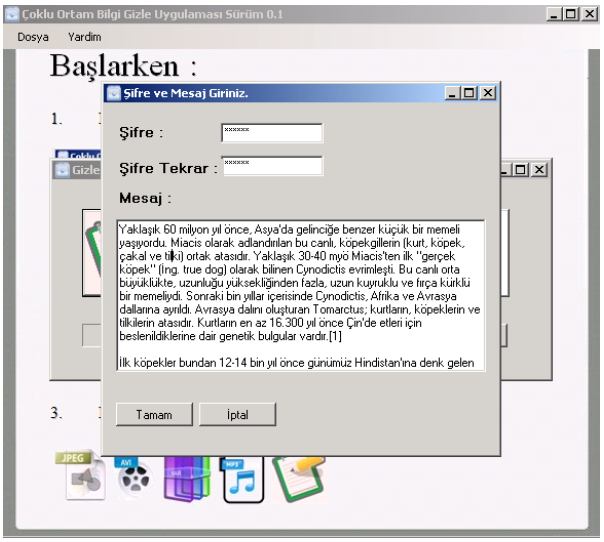
Eko veri gizleme, bilginin gizlenmesi için taşıyıcı ses sinyale bir yankı ilavesi ile sağlanmaktadır. Bilgi, yankının zayıflama oranı, gecikme veya büyüklük gibi değerler kullanılarak gizlenir. Böylece, insan kulağının algılamayacağı düzeyde 0 veya 1 kodlama için iki farklı gecikme değeri yeterli olacaktır. Bunun için sinyal segmentlere bölünerek her bit kodlanır. Sonuçta bu yöntem herhangi bir gürültüye neden olmamakta veya kayıplı bir kodlama sistemi kullanmamaktadır (Kutucu ve Kaya, 2002).

Gönderi kodlama mp3 Stegonografisi için steganografi alanında yapılan çok az araştırma vardır. Mp3 dosya biçimi çok esnek değildir ve rastgele byte üzerinde yazma yukarıda da değinildiği gibi ses verilerinin geri dönüşümsüz bozulmasına neden olabilir. Gönderi kodlama metodları da genellikle mp3 dosya yapısından yararlanır, ve mevcut ses veri dizisine ait olmayan bloklara veriyi gizler. Bu sesin bozulmasını imkansızlaştırır (Maciak ve diğ., 2006:16). Ayrıca mp3 çerçeve başlıklarındaki özel bit, telif hakkı biti, orijinal bit, vurgu biti gibi bazı bloklar nadiren kullanılmaktadır ve çoğu mp3 oynatıcı bu alanları göz ardı etmektedir. Bu noktada bu alanların değerlerinin ses çerçevesinin bütünlüğünde çok önemli olmadığını da varsaymak daha güven verici bir durumdur. Bu nedenle, her ses çerçevesi verileri gizlemek için kullanılabilecek en az dört bitlik depolama kapasitesi içermektedir (Maciak ve diğ., 2006:17).

Bir alternatif başlıklardaki dolgu byte'lardır. Yukarıda bahsedilen yöntemlerin aksine dolgu byte'lar herhangi ses bilgisi taşımadığından ve implementasyonları diğerlerine nazaran kolay olduğundan bilgi gizleme için uygun alanlar olabilirler. Dolgu byte'lar mevcut olduğu sürece her çerçeveye bir byte bilgi kodlamak mümkündür. Şaşırtıcı bir şekilde, dolgu byte'a bilgi gizleme ile ilgili çok az araştırma vardır. Ayrıca dolgu byte'ların çerçevedeki yeri, benzer bilgiler ve doğası ile ilgili çok az araştırma mevcuttur. Bu durum Steganografik araştırmalar için bir yeni bulvar olarak görülmektedir (Maciak ve diğ., 2006:17).

4.2.2. Ses Dosyasına Metin Gizleme ve Çıkarma

Kullanıcı, ilk aşamada mp3 içine metin gizlemek için dosya aç menüsünden yeni görev seçer ve böylece örtü türünü seç ekranı belirir. İlgili ekrandan mp3 ikonu seçildikten sonra mp3 dosya seçme ekranı belirir. Mp3 dosya seçiminin ardından uygulama kullanıcıya gizlenecek bilgi türünün seçileceği ekranı getiri. Bu kısımda metin ikonunun seçilmesiyle şifre ve gizlenecek metnin girileceği Şekil 4.47'deki görüldüğü gibi bir ekran gelecektir.



Şekil 4.47. Mp3 içine metin gizleme sürecindeki şifre ve metin giriş ekranı.

Mp3 dosya hafızaya depolanır. Depolanan verinin, baştan ilk 32 (otuziki) elemanı bit düzeyinde okunarak ilk başlık bilgisi elde edilir. Başlık bilgisinin ilk 12 (oniki) bitinin sadece 1 (bir)'lerden oluşup oluşmadığı kontrol edilir çünkü idv1,v2,v3 etiketleri uygulamaya dahil edilmemiştir. Bu etiket yapıları ile karşılaşıldığında, uygulama bir uyumsuzluk bilgisi verecektir. Belirlenen metin uzunluğu sekiz ile çarpılarak metin bitlerinin gizleneceği byte blok sayısı belirlenir. Metin bitleri ikişer ikişer gruplar halinde, bu dizi elemanlarının içeriklerinde tutuğu byte'ların son iki bitlerine ilerki aşamalarda aktarılacaktır.

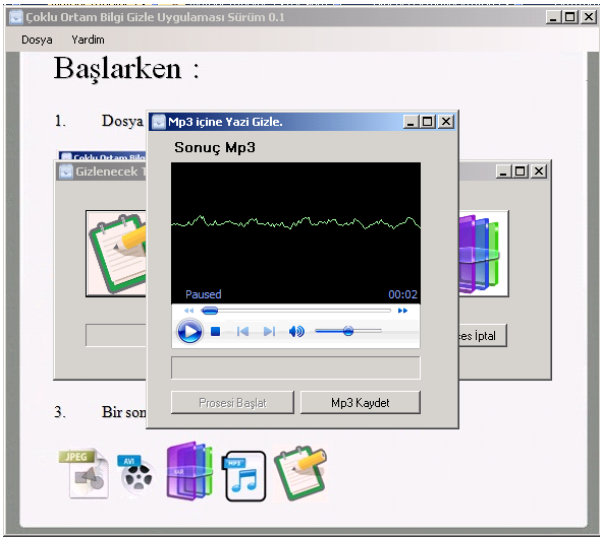
Gizlemenin ileri aşamalarında, metin bilgisinin depolanacağı mp3 dizi değişken'in elemanları üzerinde yazma işleminin uygun olup olmadığını kontrol etmek için mantıksal bir tablo yaratılır. Bu şekilde mp3 dizinin her byte'ında yapılan herhangi bir işlem tekrardan yapılmayacaktır.

Bu yapılmadığı taktirde işlem yapılmaması gereken yerler (Örneğin çerçeve başlıkları) üzerinde bir işlem gerçekleşebilir ve mp3 dosyanın bozulmasına neden olabilir. Çünkü veri gizlenecek dizi değişkenin elemanları rastlantısal sayı üretimi ile seçilmektedir. Rastlantısal seçilen ve üzerinde değişiklik yapılabilir onayından geçen dizi değişken elemanlarının aktarılması için indeks değerlerini tutan bir tablo oluşturulur.Her çerçeve boyutunun ölçüsü, çerçeve başlıklarında bulunan ve önceki kısımlarda anlatılan farklı yerlerdeki bitlerin okunması ile oluşturulan değerlerin çerçeve boyutu formülüne geçirilmesiyle elde edilir. Her çerçeve mantıksal tabloda doğru olarak işaretlenir. Üzerinde değişiklik yapılabilecek toplam dizi değişen eleman sayısı aşağıdaki formül ile hesaplanır.

Bu formülde DTE, Değiştirilebilecek toplam dizi değişken eleman sayısı, BO Mp3 dizi değişkenin boyutu'nu, TCS Toplam Çerçeve Sayısı'nı ifade etmektedir.

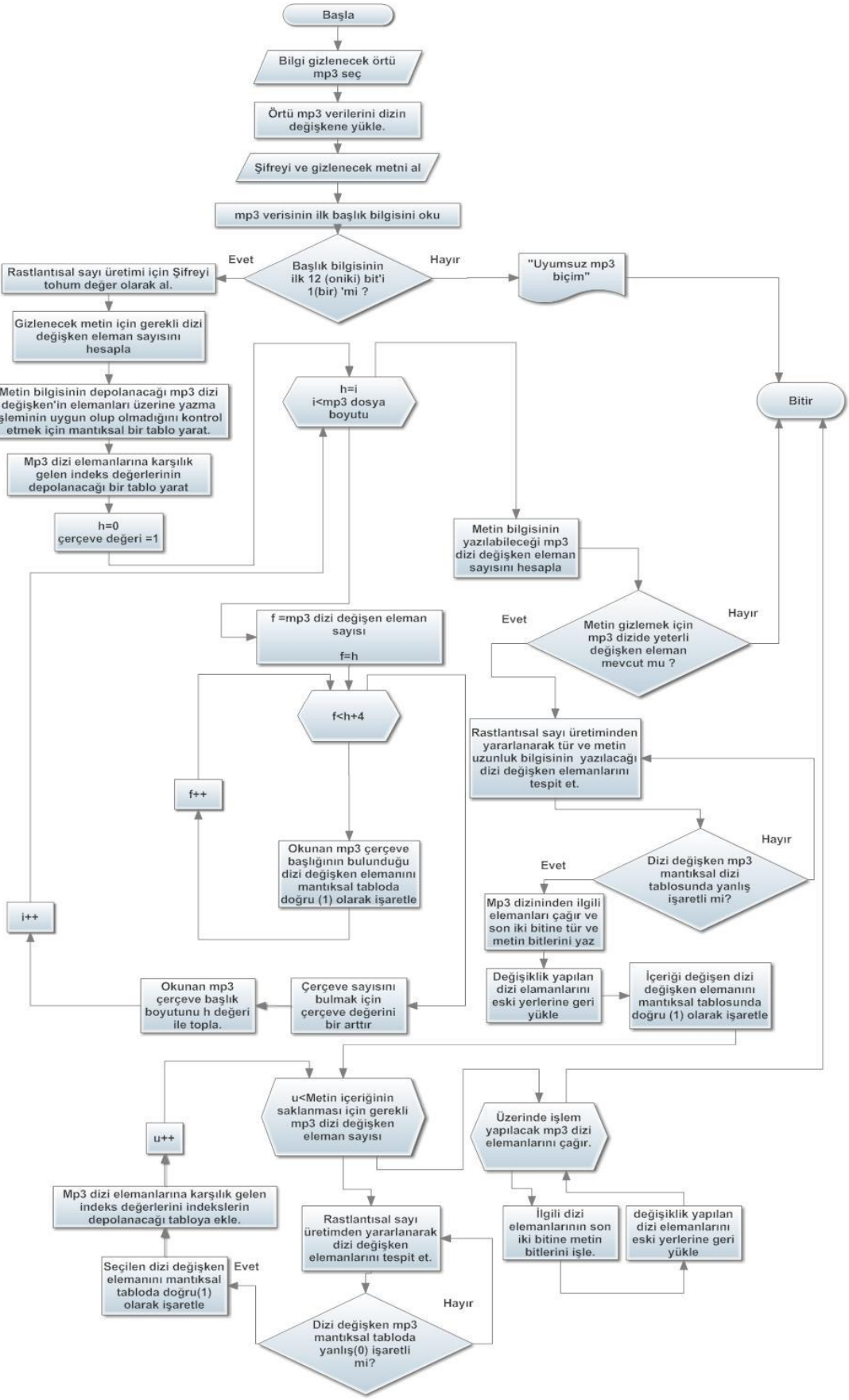
$$DTE = BO-(TCS*4) \quad \textbf{(4.1)}$$

Eğer metnin gizlenmesi için gereksinim duyulan kapasite DTE değerinden büyükse program sonlandırılır. DTE değerinden küçükse tür ve metin uzunluk bilgisinin rastlantısal bir dizi elemana gizlenmesi aşamasına geçilir. Uygun dizi elemanlarının son iki bitlerine tür ve metin bitleri ikişer olarak dağıtılar gizlenir ve bu dizinler doğru olarak mantıksal tabloya kayıt edilir.



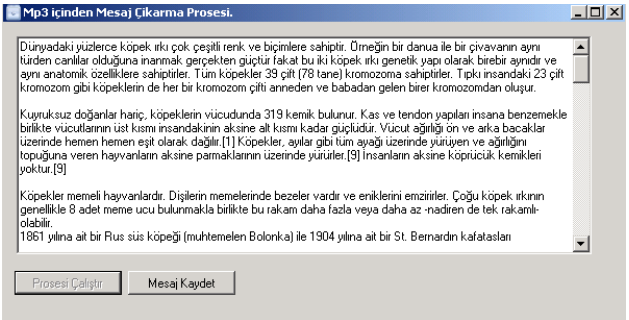
Şekil 4.48. Mp3 içine metin gizleme sürecindeki işlem sonuç ekranı.

Metin bilgisinin saklanması için rastlantısal olarak dizi değişken elemanları taranır ve uygun olanların indeks numaraları ilgili tabloya eklenir. Bu indeks tablosu bir döngüde okunarak elamanların son iki bitlerine gizlenecek metnin bitleri ikişer olarak dağıtılarak depolanır ve Şekil 4.48'de görüldüğü gibi mp3 içine yazı gizleme sonuç ekranında dinlenebilir. Şekil 4.49'da mp3 içine metin gizleme algoritmasının akış şeması verilmektedir.

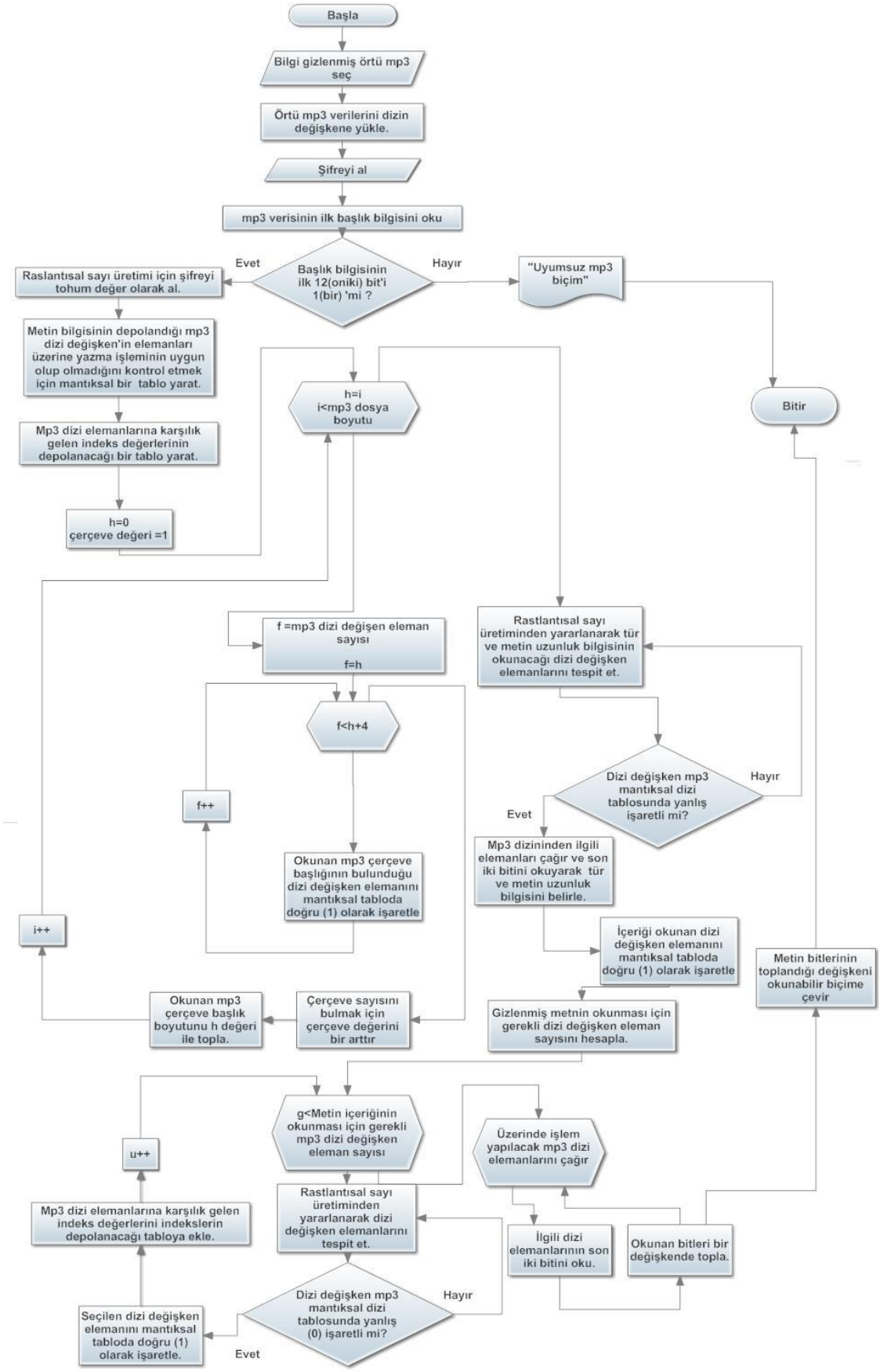


Şekil 4.49. Mp3 içine metin gizleme algoritması akış şeması.

Kullanıcı ilk aşamada mp3 içine gizlenmiş metni çıkarmak için dosya aç menüsünden bilgi çıkar'ı seçer ve böylece örtü türünü seç ekranı belirir. İlgili ekrandan Mp3 butonu tıklandıktan sonra mp3 dosya seçme ekranı çıkacaktır. Mp3 dosya seçiminin ardından uygulama kullanıcıya şifre sorar. Mp3 dosya byte türünde dizi değişkene aktarılır ve baştan 4 (dört) elemanı okunarak ilk başlık bilgisi elde edilir. Başlık bilgisinin ilk 12 (oniki) bitinin 1(bir)'lerden oluşup oluşmadığı kontrol edilir ve uyumsuzluk mevcut ise bilgi verilir. Girilen şifre rastlantısal sayı üretimi için tohum değer olarak kullanılır. Metin bilgisinin depolanacağı mp3 dizi değişken'in elemanları üzerinde yazma işleminin uygun olup olmadığını kontrol etmek için mantıksal bir tablo yaratılır. Mp3 dizi elemanlarına karşılık gelen indeks değerlerinin depolanacağı ayrıca bir tablo oluşturulur. Her çerçeve boyutu hesaplanır ve gizleme işlemindeki olduğu gibi mantıksal tabloda doğru olarak işaretlenir. Rastlantısal sayı üretimine bağlı olarak tür ve metin uzunluk bilgisinin bulunduğu bitlerin yerleri tespit edilir ve okunur. Tür ve metin uzunluk bilgisi mantıksal tabloda doğru olarak işaretlenir. Okunan tür bilgisi ekranda görüntülenir. Eğer uygulamaya dahil edilen tür bilgileri haricinde bir yapı mevcutsa, soru işaretli bir ekran belirerek kullanıcıya tür uyumsuzluğunu bildirilir. Gizlenmiş metnin okunabilmesi için gerekli dizi değişken eleman sayısı hesaplanır. Bunun için metin uzunluk bilgisi sekiz ile çarpılır. Elde edilen sonuç değer kadar rastlantısal sayı üretimi tarafından sayısal değerler oluşturur. Bu değerler mp3 dizi değişkeninde bulunan elemanların indeks numalarına karşılık gelmektedir. Oluşturulan değerler mantıksal tablodan kontrol edilir. Doğru olan değerler alınarak ilgili tabloya eklenir. Döngüde okunan değerler mp3 dizi değişkeni elemanlarıdır. Bu elemanlara karşılık gelen değerlerin son iki bitleri bir değişkende birleştirilir. Şekil 4.50.'de görüldüğü gibi birleştirilen bitler görüntülenebilecek yapıya dönüştürülür. Şekil 4.51'de Mp3 içinden metin çıkarma algoritması akış şeması gösterilmektedir.



Şekil 4.50. Mp3 içinde gizlenmiş metnin görüntülenmesi ekranı.



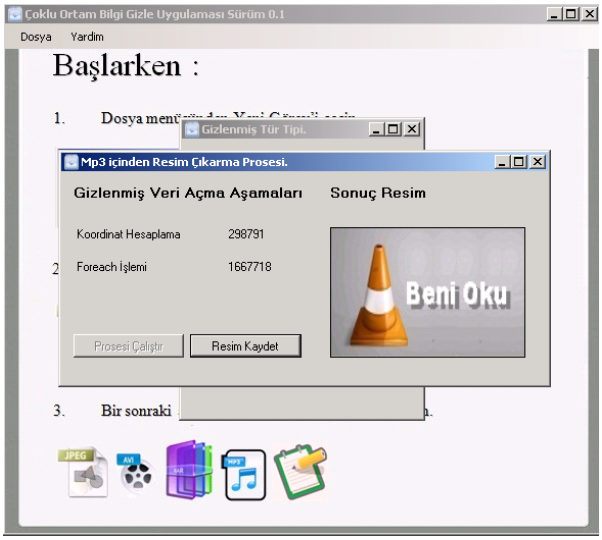
Şekil 4.51. Mp3 içinden metin çıkarma algoritması akış şeması.

4.2.3. Ses Dosyasına Diğer Dosya Yapılarını Gizleme ve Çıkarma

Uygulamanın bu kısmında mp3 içine farklı dosya biçimleri gizlenmeye çalışılmıştır. Kullanılan algoritma, metin gizleme ve çıkarmada kullanılan yapı baz alınarak geliştirilmiş ve tasarlanmıştır. Uygulama arayüzündeki resim gizleme ekranında, gizleme süreci başlatılmadan önce, içine mp3 gizlenecek resim görüntülenebilmektedir. Prosesi başlat ikonu seçildiğinde, Şekil.4.52'de görüldüğü gibi algoritmadaki bazı işlem aşamaları sayısal olarak, gizleme işlemi boyunca kullanıcıya iletilir.

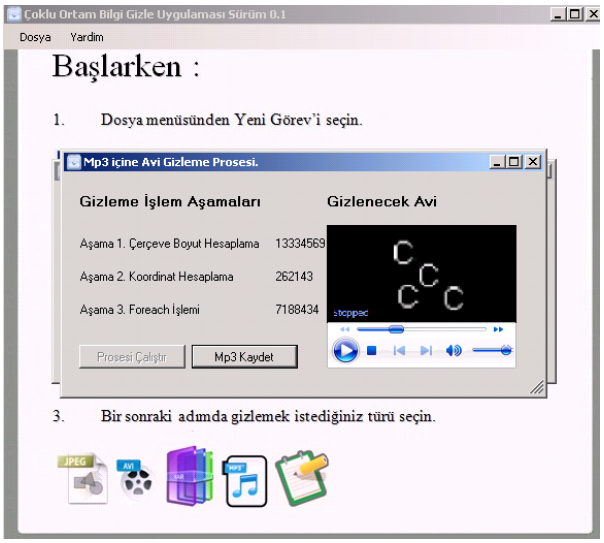
Şekil 4.52. Mp3 içine dosya türü gizleme sürecindeki aşamaları gösteren ekran.

Gizleme sürecinde, İlk aşamada mp3 dosya hafızaya depolanmaktadır. Uygulama, hafızaya depolanan mp3 çerçevelerinin başlangıç ve bitiş koordinatlarını tespit etmektedir. Bu sağlamak için uygulama her çerçevinin başlık dosyası bilgisini okumaktadır. Böylece mp3 ses verilerinin hangi koordinatlarda olduğu tespit edilmektedir. Bir sonraki aşamada rastlantısal olarak seçilen çerçevelere resim verileri bir döngü içinde yazılmaktadır. Bu durum, mp3 içinden resim çıkarma işleminde tersine bir sürec izler. Şekil 4.53'te görüldüğü gibi sürec tamamlandığından sonuç resim ekranda belirir ve kullanıcı isterse bu resmi depolama ortamına kayıt edebilir.



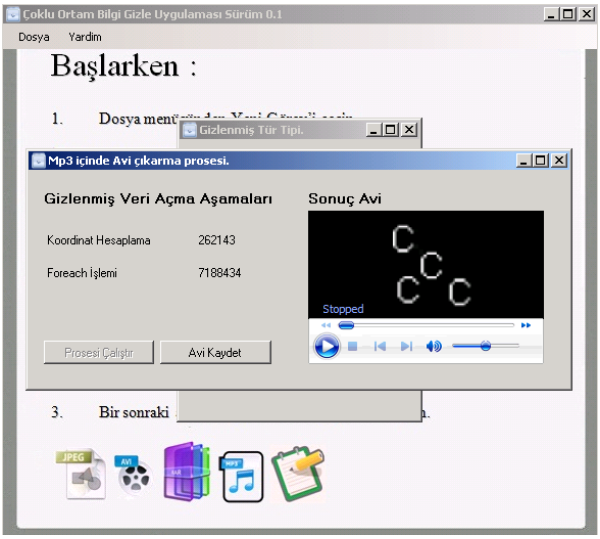
Şekil 4.53. Mp3 içinden dosya türü çıkarma ekranı.

Mp3 içine avi gizleme işlemi ekranında, gizlenecek avi görüntülenebilmektedir. Prosesi başlat ikonu tıklandığında genel olarak bazı veri gizleme aşamaları işlem sırasında gösterilir. İşlem bittiğinde veri gizlenmiş mp3 Şekil 4.54'te gösterildiği gibi Mp3 kaydet seçilerek kayıt ortamında depolanabilir.



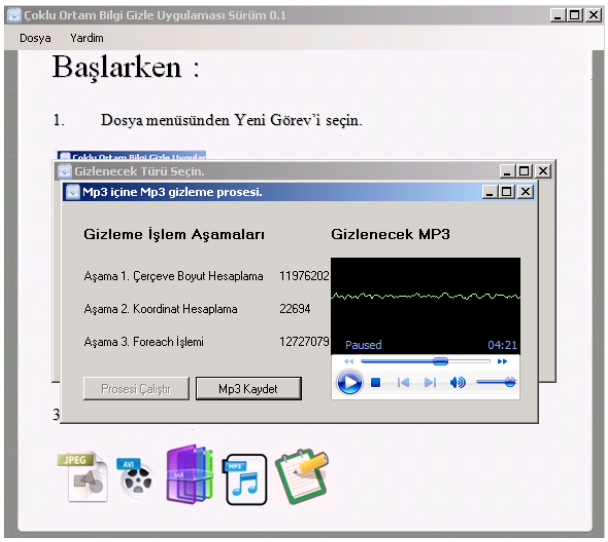
Şekil 4.54. Mp3 içine avi gizleme işlemi ekranı.

Şekil 4.55'te mp3'den gizli avi'nin çıkarılması işlemi ekranında, bazı hesaplamalar, veri çıkarma süreci boyunca sayısal olarak ifade edilmektedir. İşlem bittiğinde, sonuç avi kısmında gizlenmiş avi görüntülenebilir yada depolama ortamına kayıt edilebilir.



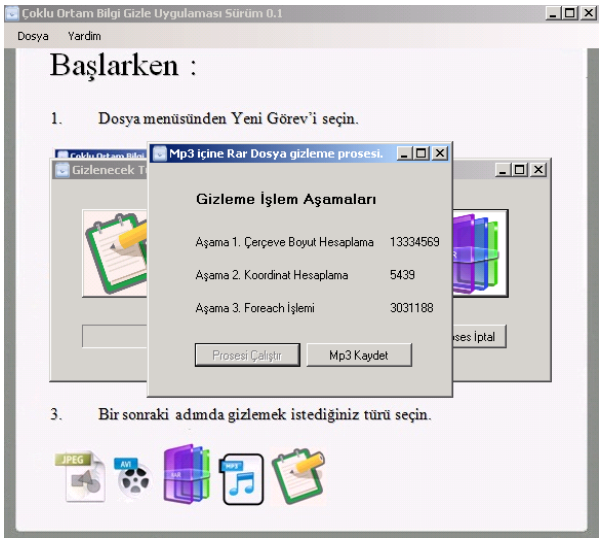
Şekil 4.55. Mp3 içinden gizli avi çıkarma işlemi ekranı.

Mp3 içine mp3 gizleme işlemi ekranında, gizlenecek mp3 dinlenebilmektedir. Prosesi başlat ikonu tıklandığında genel olarak veri gizleme adımları anlık olarak gösterilir. Şekil 4.56'da görüldüğü üzere işlem bittiğinde veri gizlenmiş mp3 depolama ortamına kayıt edilebilmektedir. Sürecin tersi olan mp3 içinden mp3 çıkarmada, mp3 içinden avi çıkarma ekranına benzerlik göstermektedir.



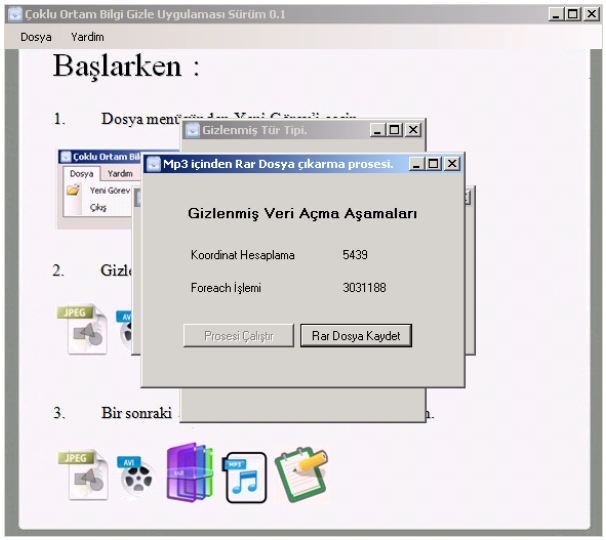
Şekil 4.56. Mp3 içine mp3 gizleme işlemi ekranı.

Mp3 içine rar gizleme işlemi ekranında, Şekil 4.57'de görüldüğü gibi prosesi çalıştır ikonu tıklandığında, bazı veri gizleme aşamaları, gizlem işlemi sırasında gösterilir. İşlem bittiğinde veri gizlenmiş mp3 kayıt ortamına depolanabilir.

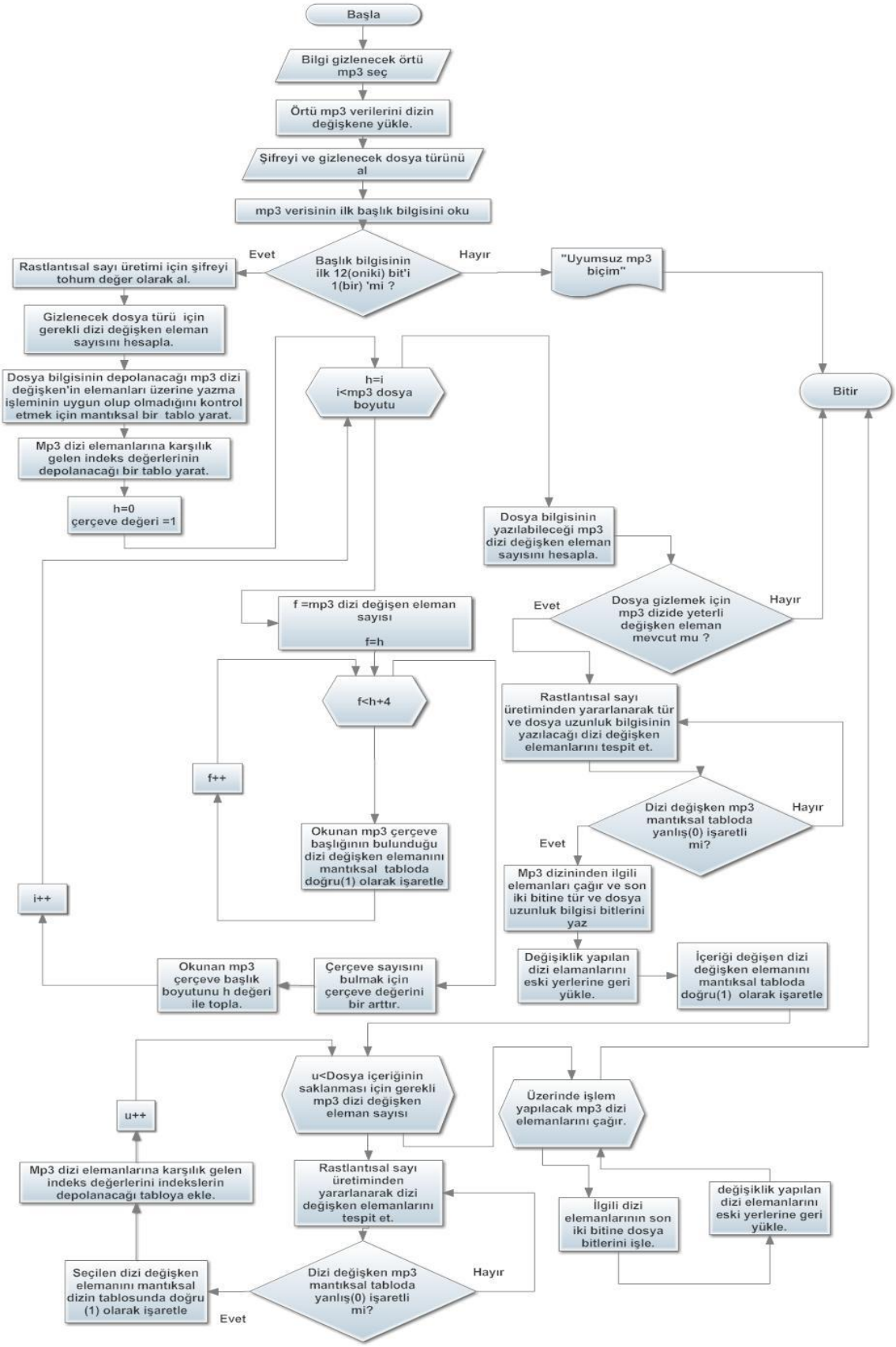


Şekil 4.57. Mp3 içine rar gizleme işlemi ekranı.

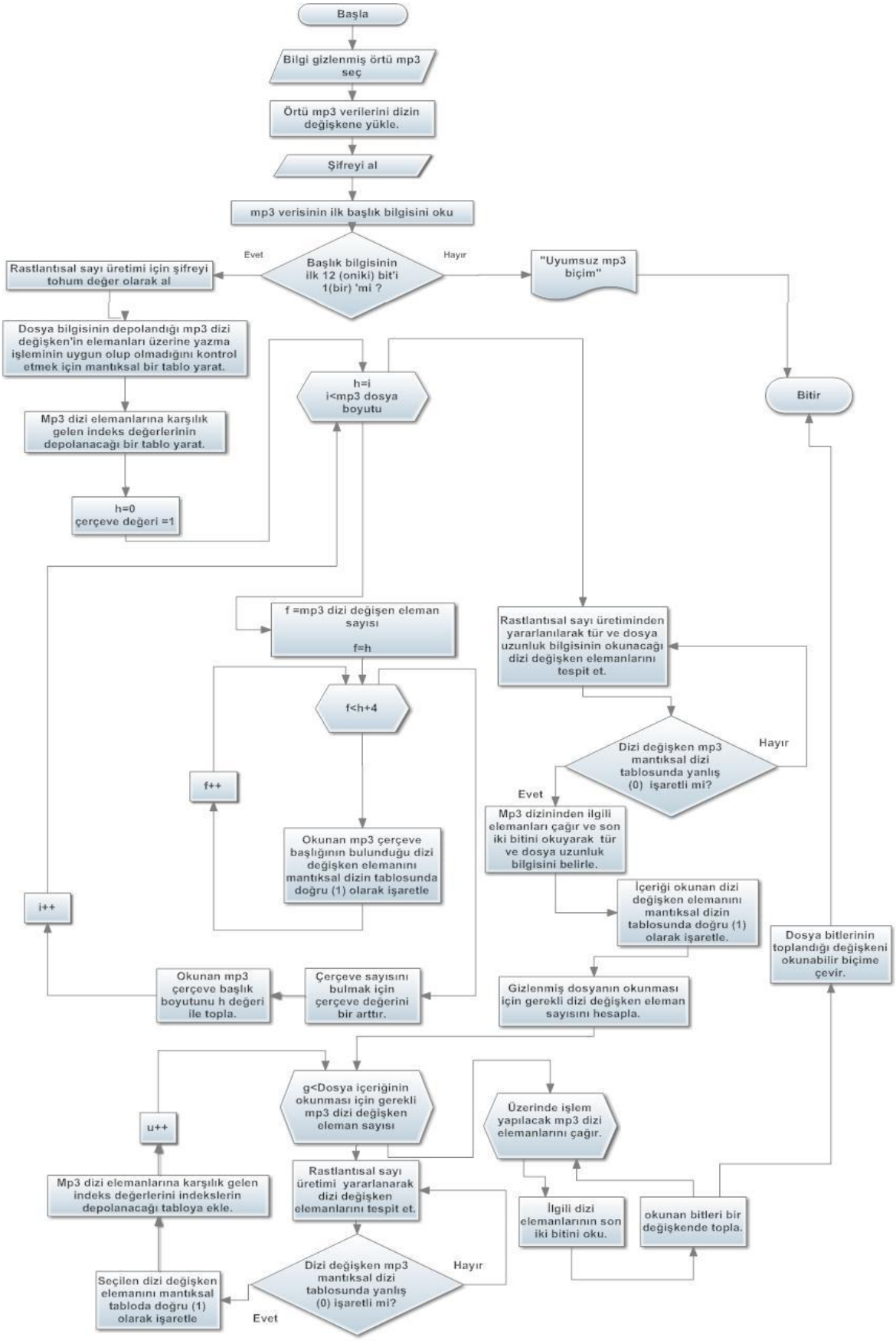
Şekil 4.58'te mp3'den gizli rar dosyanın çıkarılması işlemi, ilgili ekranda prosesi çalıştır seçildiğinde başlar ve işlem bittiğinde rar dosya kaydet ikonu seçilerek kayıt ortamına depolanabilir. Şekil 4.59.'da mp3 içine farklı dosya türlerinin gizlenmesi ve şekil 4.60'da bu dosyaların çıkarılmasında uygulanan algoritmaların akış şemaları görülmektedir.



Şekil 4.58. Mp3 içinden rar çıkarma işlemi ekranı.



Şekil 4.59. Mp3 içine farklı dosya türlerini gizleme algoritması akış şeması.



Şekil 4.60. Mp3 içinden farklı dosya türlerini çıkarma algoritması akış şeması.

5. SONUÇ

Bu çalışmada, steganografi ve bilgi gizleme algoritmaları bir araya getirilip entegre çalışan bir uygulama tasarlanmış ve geliştirilmiştir. Uygulama, çoklu ortam bileşenlerinin herbirini kendi arasında gizleyebilmekte ve gizlenen bileşenleri çıkarılabilmektedir. Uygulama, teknolojik gelişmelere bağlı olarak yeni yada güncellenecek her türlü çoklu ortam biçiminde kullanılabilecek bir algoritmaya sahiptir. Bu esneklik özellikle giderek popülerliğini arttıran mobil teknolojilerde çalışabilirliği de sağlar. Bununla birlikte mobil cihazlar veya kişisel bilgisayarlar dışında her türlü çoklu ortam kullanma yeteneğine sahip digital fotoğraf makinaları veya kameralar gibi elektronik cihazların yapısına uygun şekilde gerekli düzenlemeler yapılarak uygulamanın adaptasyonu kolaylıkla sağlanabilinir.

Geliştirilen uygulamada gizlenmek istenen bilgi için boyut sınırlaması getirmemektedir. İstenilen büyüklükte metin, video ya da ses dosyası gizlenebilir. Bu noktada içine gizlenecek çoklu ortamın saklama kapasitesi ve elektronik cihazın donanım gücü önemlidir. Elektronik cihazlarının başta cpu ve hafıza kaynakları gibi donanım özelliklerinin işlem güçleri ve yetenekleri her geçen yıl artmaktadır. Böylece teknolojik ilerlemeler, uygulamanın daha büyük boyutlarda ki çoklu ortam bileşenlerini daha hızlı olarak gizleyebilme ve çıkarabilme özelliğine sahip olmasını sağlayacaktır.

6. KAYNAKLAR

Aslı A., Maden S., Durukan E.(2010), *Çoklu Ortam Aktiviteleriyle Metin Öğretimine Bir Model(Fabl Örneği)*, Uluslarası Sosyal Araştırmalar Dergisi, cilt 3/10.

Atıcı M.(2007), *Steganografik Yaklaşımların İncelenmesi, Tasarımı ve Geliştirilmesi*, T.C. Gazi Üniversitesi Fen Bilimleri Enstitüsü, Ankara.

Akbal T.(2008), *Ses Verilerine Sıkıştırılmış ve Şifrelenmiş Ham Verilerin Gömülmesi,* Yüksek Lisans Tezi, Sakarya Üniversitesi Fen Bilimleri Enstitüsü, Haziran 2008.

Altan N. (2003), Unicode,*Bilgisayar Terimleri Ansiklopedik Sözlüğü(3.baskı)içinde p.602*, Sistem Yayıncılık, İstanbul.

Çıbuk M. (2004), Haberleşme *Sistemlerinde Kullanılan Temel Kodlama ve Sıkıştırma Teknikleri,* Doktora Semineri, T.C. Fırat Üniversitesi Fen Bilimleri Enstitüsü, Elazığ.

Dutta P., Bhattacharyya D., Kim T.(2009), *Data Hiding in Audio Signal : A*

Review International Journal of Database Theory and Application, Vol.2, No.2.

Ertürkler M.(2007), *Sayısal Filigranlar İle Kripto İmzalarının Birlikte Kullanılması ve Çoklu Ortam Verisi Üzerindeki Uygulamaları*, Doktora Tezi, T.C. Fırat Üniversitesi Fen Bilimleri Enstitüsü, Elazığ.

Furat M.(2006), Sayısal Ortamlarda Veri Damgalanması ve Geri Eldesi, Yüksek Lisan Tezi, T.C. Mustafa Kemal Üniversitesi, Fen Bilimleri Enstitüsü, Antakya, 2006.

Gürel H. (2006), *Sayısal resim içerisine veri gizleme uygulaması*, Yüksek Lisans Tezi, T.C.Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Kocaeli.

Göçeri E., Yaldır K. (2007*), JPEG2000 Standardının Yeni Özelliklerini Destekleyen Bir Görüntü İşleme Uygulaması Geliştirilmesi,* Pamukkale Üniversitesi Mühendislik Bilimleri Dergisi, sayi :3, cilt :13,p. 337-346.

Hamza Y. (2008), *Blok Kırpma Kodlamasına ve Ayrık Dalgacık Dönüşümüne Dayalı, Dayanıklı Digital Renkli Resim Damgalama Sistemi*, Yüksek Lisans Tezi, Anadolu Üniversitesi Fen Bilimleri Enstitüsü, Eskişehir.

Kurtuldu Ö. (2008), *İmge Steganografi İçin Yeni Yöntemler*, Yüksek Lisans Tezi, T.C. Deniz Harp Okulu Deniz Bilimleri ve Mühendisliği Enstitüsü, İstanbul.

Oğuz C. (2006), *Görüntü İşaretçileri İçin Yeni Bir Sayısal Damgalama Yöntemi*, Yüksek Lisans Tezi, İstanbul Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

Özdemir A., Artıklar M. (2009*), İki boyutlu Dalgacık Dönüşümü Kullanarak Ön cepheden Çekilmiş İnsan Yüzü Resimlerini Tanıma Üzerine Yaklaşımlar*, KSÜ Mühendislik Bilimleri Dergisi, 12(1):6-9.

Öztürk T.(2009), *Video ile Güvenli Veri Aktarımı*, Bitirme Projesi, T.C. Haliç Üniversitesi Fen Bilimleri Enstitüsü, İstanbul.

Raissi R. (2002), *The Theory Behind Mp3*, Erişim tarihi : 2 Şubat 2012, http://www.mp3-tech.org/programmer/docs/mp3_theory.pdf

Reddy V., Subramanyam A., Reddy C. (2011), *SteganPEG Steganography + JPEG,* International Journal of Computer Graphics , Vol. 2, No. 1, p.31-42.

Singh S., Agarwal A., Agarwal D., Gambhir A., Kumar S. (2009), *Colour Space Entropy Based Lossy and Lossless Colour Image Compression System*, International Journal of Computer Science and Network Security, Vol : 9 ,No:3, 327-336.

Sripada P. (2006), *Mp3 Decoder in Theory and Practice, Master Thesis Report, Blekinge Institute of Technology*, Department of Singal Processing and Telecommunications, Sweden.

Selasky H.(2006), *Evaluation of perceptual sound compression with regard to perceived quality and compression methods*, Master Thesis, Agder University College, Faculty of Engineering and Science, Norway.

Şahin A. (2007) , *Görüntü Steganografide Kullanılan Yeni Metodlar Ve Bu Metodların Güvenirlikleri*, Doktora Tezi, T.C. Trakya Üniversitesi Fen Bilimleri Enstitüsü, Edirne.

Uslu Y. (2003), *Çoklu Ortamda Veri Gizleme*, Yüksek Lisans Tezi, T.C. İstanbul Kültür Üniversitesi Fen Bilimleri Enstitüsü,İstanbul.

Weeks M., (2006), *Digital Signal Processing Using MATLAB and Wavelets*, Jones Bartlett Publishers, Boston.

INTERNET

Alçı M. ve Çivicioğlu P.(2003) ,*Güvenli İletişim İçin Veri Gizleme Tekniklerinin Kullanımı*, Elektrik-Elektronik – Bilgisayar Mühendisliği 10.Ulusal Kongresi. Erişim Tarihi : 25 Mayıs 2012, http://www.emo.org.tr/ekler/746dff176d822bb_ek.pdf.

Afterdawn (2012), *AVI*, Erişim Tarihi : 6 Mayıs 2012, http://www.afterdawn.com/ glossary/term.cfm/avi

Alexander-neo (2012), Erişim Tarihi : 17 Mart 2012,http://www.alexander-neo.com/

video/amg/definitions.html

Chip (2009), Erişim Tarihi : 18 mayıs 2012, http://www.chip.com.tr/makale/resim-sikistirma-yontemleri-ve-jpeg_11812.html**.**

Digitalpreservation (2012), Erişim Tarihi : 13 Mayıs 2012, http://www.digitalpreservation.gov/formats/fdd/fdd000025.shtml

Differencebetween(2012), Erişim Tarihi : 11 Mart 2012, http://www.differencebetween.net/technology/difference-between-divx-and-avi/#ixzz1xUUgN700

Esin M., Güvenoğlu E.(2006), *Resim İçine Yazı Gizlenmesi Amacıyla Kullanılan LSB Ekleme Yönteminin Shuffle Algoritmasıyla iyileştirilmesi*. Türkiye Bilişim Vakfı

Bilgisayar Bilimleri Dergisi, Sayı:2, Aralık 2006, http://www.bmbb.tc/dosya/dergi/ 2_8.pdf

Erişir E. (2012), *Teknik Fotoğrafçılık : Digital Görüntüleme*, Erişim Tarihi : 25 Mayıs 2012, http://metalurji.kocaeli.edu.tr/files/DersNotlari/mmt106-06.pdf

Fags (2012), Erişim Tarihi : 12 Mayıs 2012, http://www.faqs.org/faqs/jpeg-faq/part1/section-18.html

Fileformat(2012), Erişim Tarihi : 7 Mayıs 2012, http://www.fileformat.info/riff/

egff.htm

Kutucu H., Kaya M. (2002), *Cryptography and Network Security*, Term Project,

Ege University International Computer Institute, Erişim Tarihi : 2 Mart 2012, www.iyte.edu.tr/~hakankutucu/stegano.doc

Maciak L., Ponniah M., Sharma R (2006), *Applying Steganography to Music Captioning - Embedding Lyrics in MP3*, Montclair State University, ABD, Erişim Tarihi: 14 Nisan 2012, http://www.terminally-incoherent.com/stuff/projects/mp3stego/paper.doc

Mesut A., Carus A (2005), *Kayıpsız Görüntü Sıkıştırma Yöntemlerinin Karşılaştırılması*, II. Mühendislik Bilimleri Genç Araştırmacılar Kongresi, MBGAK2005, İstanbul, Erişim Tarihi : 11 Mart 2012, http://altanmesut.trakya.edu.tr/pubs/A1-14.pdf

Microsoft (2012 a), Erişim Tarihi : 12 Mayıs 1012, http://msdn.microsoft.com/en-us/library/ms779636%28VS.85%29.aspx

Microsoft(2012 b), Erişim Tarihi : 29 Mayıs 2012, http://windows.microsoft.com/tr-TR/windows7/Codecs-frequently-asked-questions

McGowan J.(2012), Erişim Tarihi : 7 Mayıs 2012, http://www.jmcgowan.com/avitech.html

Oral M. ve Furat M. (2007), *Veri Saklama Yöntemleri: Sayısal Görüntülerin Damgalanması, Amaçları ve Uygulamalar Alanları*, III. İLETİŞİM TEKNOLOJİLERİ ULUSAL SEMPOZYUMU, Erişim Tarihi : 25 Mayıs 2012 http://www.emo.org.tr/ekler/d6072ea730f0625_ek.pdf

Öksüzoğlu S.(2009) ,*Radyografik Görüntülere Veri Gizleme Uygulaması*, Elektrik-Eletronik-Bilgisayar ve Biyomedikal mühendisliği 13.ulusal kongresi, Ankara, Erişim Tarihi : 25 Aralık 2012, http://www.emo.org.tr/etkinlikler/ulusal/etkinlik_bildirileri_detay.php?etkinlikkod=116&bilkod=4499

Schroeder M. (2012), *Jpeg Compression Algorithm and Associated Data Structure*, Erişim Tarihi : 16 mayıs 2012, http://akbar.marlboro.edu/~mahoney/courses/Fall01/computation/compression/jpeg/jpeg.html

Şahin A., Buluş E., Sakallı M. (2006), *24-Bit Renkli Resimler Üzerinde En Önemsiz Bite Ekleme Yöntemini Kullanarak Bilgi Gizleme*, Araştırma Makalesi, Trakya

Üniversitesi, Mühendislik Mimarlık Fakültesi, Bilgisayar Mühendisliği Bölümü, EDİRNE. Erişim Tarihi : 25 mayıs 2012, http://fbe.trakya.edu.tr/tujs/arsiv/2006-1/183Andac_Sahin_renkli.pdf.

Techkiwhitepaper (2012), Erişim Tarihi : 24 Mayıs 2012, http://techkiwhitepaer. .blogspot.com/2009/03/avi-file-format-data-structures.html

Teknovadi(2012), Erişim tarihi : 10 Mart 2012, http://www.teknovadi.com/teknoloji-rehberi/piksel-nedir-gorevi-ve-calisma-mantigi/

Terminaly-incoherent (2012), Erişim Tarihi : 15 Nisan 2012, http://www.terminally-incoherent.com/stuff/projects/mp3stego/paper.htm

Videospark (2012), Erişim Tarihi : 2 Şubat 2012, http://www.videospark.com/avi/

what-is-avi.html

Wikipedia (2012 b), Erişim Tarihi : 12 Mayıs 2012, http://tr.wikipedia.org/wiki/ Huffman_kodu.

Wikipedia (2012 c), Erişim Tarihi : 11 Nisan 2012, http://en.wikipedia.org/wiki/

Audio_Video_Interleave

Wikipedia (2012 d),Erişim Tarihi : 29 Mayıs 2012, http://tr.wikipedia.org/wiki/MP3

Wikipedia (2012 e), Erişim tarihi : 2 Nisan 2012, http://en.wikipedia.org/wiki/ psychoacoustics

Wikipedia (2012 f), Erişim Tarihi : 29 Mayıs 2012, http://tr.wikipedia.org/wiki/RAR

Wikipedia (2012 g), Erişim Tarihi : 25 Mayıs 2012, http://tr.wikipedia.org/wiki/ çoklu_ortam

Wikipedia, (2012 h), Erişim Tarihi : 29 Mayıs 2012, http://tr.wikipedia.org/wiki/

JPEG

Yoo B., Park J., Lim S., Bang J., Lee S. (2011), *A study on multimedia file carving method,* Springer Science+Business Media, LLC 2011. Erişim Tarihi : 12 Nisan 2012, http://posgrado.escom.ipn.mx/biblioteca/A study on multimedia file carving method.pdf

Yeşilyurt M., Özcerit A., Yalman Y., Ertürk İ (2012), Sayısal *İmgeler İçin Ayrık Kosinüs Dönüşümü Esaslı Veri Gizlemenin Ataklara Dayanıklılığı*, Akademik Bilişim Konferansları, Erişim Tarihi : 4 Mayıs 2012, http://ab.org.tr/ab12/bildiri/53.pdf

7. ÖZGEÇMİŞ

1975 yılında doğdu. 1992 yılında Üsküdar Halide Edip Adıvar Lisesi'nden mezun oldu. 1995-1999 yılında Özkesoğlu Isı San. A.Ş.'de Pazarlama departmanında bilgi işlem sorumlusu olarak çalıştı. 1999 yılında Karadeniz Teknik Üniversitesi Sürmene Deniz Bilimleri Fakültesi Balıkçılık Teknolojisi Mühendisliğinde başlamış olduğu lisans eğitimini 2003 yılında tamamladı. 2003-2004 yıllarında CPI Bilgisayar Paz.Ltd.'de ağ-sistem uzmanı olarak çalıştı. 2005-2011 yıllarında Martı Konteyner Hizmetleri A.Ş.'de bilgi işlem müdürü yardımcısı olarak görev aldı. 2009-2012 yılları arasında Haliç Üniversitesi Fen Bilimleri Enstitüsü Bilgisayar Mühendisliği'nde yüksek lisans eğitimi aldı.