T.C.
ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
MATEMATİK ANABİLİM DALI
2017-YL-021

SES STEGANOGRAFİ VE METOTLARININ KARŞILAŞTIRILMASI

Kadir TEKELİ

Tez Danışmanı:
Yrd. Doç. Dr. Rıfat AŞLIYAN

AYDIN

T.C.
ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE
AYDIN

Matematik Anabilim Dalı Yüksek Lisans Programı öğrencisi Kadir TEKELİ tarafından hazırlanan "Ses Steganografi ve Metotlarının Karşılaştırılması" başlıklı tez, 23/06/2017 tarihinde yapılan savunma sonucunda aşağıda isimleri bulunan jüri üyelerince kabul edilmiştir.

		Unvanı, Adı Soyadı	Kurumu	İmzası
Başkan	:	Yrd. Doç. Dr. Rıfat AŞLIYAN	ADÜ	……
Üye	:	Yrd. Doç. Dr. Korhan GÜNEL	ADÜ	……
Üye	:	Yrd. Doç. Dr. Mehmet Ali BALCI	MSKÜ	……

Jüri üyeleri tarafından kabul edilen bu Yüksek Lisans tezi, Enstitü Yönetim Kurulunun …… Sayılı kararıyla …… tarihinde onaylanmıştır.

Prof. Dr. Aydın ÜNAY
Enstitü Müdürü

T.C.
ADNAN MENDERES ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ MÜDÜRLÜĞÜNE
AYDIN

Bu tezde sunulan tüm bilgi ve sonuçların, bilimsel yöntemlerle yürütülen gerçek deney ve gözlemler çerçevesinde tarafımdan elde edildiğini, çalışmada bana ait olmayan tüm veri, düşünce, sonuç ve bilgilere bilimsel etik kuralların gereği olarak eksiksiz şekilde uygun atıf yaptığımı ve kaynak göstererek belirttiğimi beyan ederim.

23/06/2017

Kadir TEKELİ

ÖZET

SES STEGANOGRAFİ VE METOTLARININ KARŞILAŞTIRILMASI

Kadir TEKELİ

Yüksek Lisans Tezi, Matematik Anabilim Dalı
Tez Danışmanı: Yrd. Doç. Dr. Rıfat AŞLIYAN
2017, 89 sayfa

Steganografi, bir verinin herhangi bir taşıyıcı içerisine gizlenerek güvenli bir şekilde iletilmesini sağlayan yöntemleri araştıran bilim dalıdır. Steganografi yöntemleri genel olarak resim, ses ve video gibi dijital verilerin kopya koruma amaçlı damgalamasında kullanılmaktadır. İnsanın işitme duyusu görme duyusuna kıyasla çok daha hassas olduğu için ses steganografi oldukça zorlu bir alandır. Dolayısıyla, ses steganografi ile ilgili yayınların sayısı resimlere kıyasla daha azdır ve nitelikli çalışmalar oldukça karmaşık algoritmalar üzerine kuruludur.

Bu tez çalışmasında yaygın olarak kullanılan seçilmiş ses steganografi yöntemleri ve karşılaştırmaları, matematiksel olarak ve dijital sinyal işleme temelleri ile birlikte yalın bir şekilde açıklanmaya çalışılmıştır. Çalışmanın asıl hedefi mevcut temel yöntemlerden oluşan açık kaynak kodlu bir kütüphane tasarlayıp, paylaşabilmektir. Bu kütüphane tasarımı için, bilimsel araştırmalarda yaygın olarak kullanımı göz önünde bulundurularak MATLAB platformu tercih edilmiştir. Bu çalışmada özellikle Tayf Yayılımı, Yankı Veri Gizleme ve Önemsiz Bit Kodlaması yöntemleri üzerine odaklanılmıştır. İncelenen yöntemler için karşılaştırmalar sağlamlık, kapasite ve fark edilmezlik üçgeni içerisinde değerlendirilmiştir. Yöntemlerin sağlamlıkları sıkıştırmaya karşı başarı oranları ile tartışılıp, sonuç olarak Tayf Yayılımı yöntemi üzerinde bir takım performans iyileştirmeleri önerilmiştir.

Anahtar Sözcükler: Ses Steganografi, Ses Damgalama, Tayf Yayılımı, Yankı Veri Gizleme, Önemsiz Bit Kodlaması

ABSTRACT

AUDIO STEGANOGRAPHY AND COMPARISON OF METHODS

Kadir TEKELİ

M.Sc. Thesis, Department of Mathematics
Supervisor: Assist. Prof. Dr. Rıfat AŞLIYAN
2017, 89 pages

Steganography is the science of hiding information within cover objects in order to transfer data securely. Besides, steganography methods are used to watermark digital files such as images, audios and videos for copyright protection. Audio steganography is one of the most challenging subjects since Human Auditory System (HAS) is more sensitive than Human Visual System (HVS). Therefore, there are fewer publications for audio steganography comparing to images, moreover qualified publications include complex algorithms.

In this study, commonly used existing audio steganography methods with their comparisons have been explained briefly with its mathematics and basics of digital signal processing. The main goal was to design and publish an open source library with basic methods. MATLAB was chosen as programming platform for the library design considering it is commonly used for scientific researches. In this study, we have focused on Spread Spectrum, Echo Hiding and LSB Coding. Studied methods have been discussed in a triangle of robustness, capacity and imperceptibility. Robustness against compression has been compared and a couple of performance improvements for Spread Spectrum technique have been proposed as a result.

Keywords: Audio Steganography, Audio Watermarking, Spread Spectrum, Echo Hiding, Least Significant Bit Coding

ÖNSÖZ

Bu tez çalışmasında dijital ses içerisine veri gizleme yöntemleri steganografi ve damgalama arası bir seviyede incelenmiştir. Dijital sesin yapısal karmaşıklığı ile birlikte mevcut yöntemlere ilişkin çalışmaları anlayabilmek için yeterli seviyede matematik, dijital sinyal işleme ve bilgisayar programlama bilgisine ihtiyaç vardır. Mevcut çalışmalara ilişkin kaynak kodlarına erişimde yaşanan imkansızlıklar nedeniyle yeni geliştirilen yöntemlerin başarı oranlarının mevcut yöntemler ile karşılaştırılması maalesef önceki çalışmalardaki farklı ses verileri üzerinden elde edilmiş sayısal değerler kullanılarak yapılmaktadır. Ancak ses steganografide bir yöntemin başarı oranının ölçülmesinde gizlemede kullanılan örtü verinin karakteristik özelliklerinin de büyük bir etkiye sahip olduğu bilinmektedir. Bu yüzden, karşılaştırılan yöntemler için ortak bir ses veri tabanı kullanılması oldukça önemlidir. Dolayısıyla bu çalışmada kaynak erişimlerinde yaşanan sıkıntılar göz önüne alınarak öncelikli hedef olarak mevcut temel yöntemler kaynak kodları ile birlikte verilmiştir.

Tez boyunca yaşadığım tüm sıkıntı ve zorluklara rağmen beni desteklemekten ve cesaretlendirmekten hiç vazgeçmeyen, bilgi ve birikimiyle sürekli yanımda olan saygı değer hocam ve danışmanım Yrd. Doç. Dr. Rıfat AŞLIYAN ’a, aynı şekilde bilgisi, tecrübesi ve güler yüzü ile örnek aldığım diğer saygı değer hocam Yrd. Doç. Dr. Korhan GÜNEL ’e ve hayatım boyunca desteğini benden hiç esirgemeyen, iyi ve kötü her durumda arkamda duran, benim için dünyadaki her şeyden daha değerli olan aileme teşekkürü bir borç bilirim.

İÇİNDEKİLER

KISALTMALAR DİZİNİ

BER	Bit Error Rate
CD	Compact Disc
dB	Decibel
DCT	Discrete Cosine Transform
DFT	Discrete Fourier Transform
DSSS	Direct Sequence Spread Spectrum
DST	Discrete Sine Transform
DSTFT	Discrete Short Time Fourier Transform
DWT	Discrete Wavelet Transform
EMD	Empirical Mode Decomposition
FFT	Fast Fourier Transform
FSSS	Frequency Hopping Spread Spectrum
GTY	Geleneksel Tayf Yayılımı
Hz	Hertz
ISS	Improved Spread Spectrum
İGPNY	İleri Geri Pozitif Negatif Yankı
İGY	İleri Geri Yankı
İTY	İyileştirilmiş Tayf Yayılımı
KB	Kilobyte
LPC	Linear Predictive Coding
LSB	Least Significant Bit
LWT	Lifting Wavelet Transform
MB	Megabyte
MDCT	Modified Discrete Cosine Transform
MSB	Most Significant Bit
NC	Normalized Correlation
NPY	Negatif Pozitif Yankı
ÖDT	Öznel Dinleme Testleri
ÖTY	Önerilen Tayf Yayılımı
QIM	Quantization Index Modulation
SNR	Signal-to-Noise Ratio
SVD	Singular Value Decomposition
TY	Tekli Yankı
WHT	Walsh-Hadamard Transform

ŞEKİLLER DİZİNİ

ÇİZELGELER DİZİNİ

EKLER DİZİNİ

1. GİRİŞ

Veri gizleme bilimi; özellikle matematik, fizik, bilgisayar bilimleri ve bu bilimlerin alt dalları ile birebir etkileşimde olan yığılmalı bir bilim dalıdır. Dijital veri gizleme yöntemlerini anlayıp yorumlayabilmek ve yeni yöntem geliştirebilmek için bu bilim dallarını ortak bir çatıda etkili bir şekilde kullanabilmek oldukça önemlidir. Bu tez çalışmasında özel olarak veri gizleme biliminin bir alt başlığı olan ses örtüsü içerisine veri gizleme yöntemleri karşılaştırmalı olarak incelenmektedir.

İnsanın işitme duyusu, görme duyusuna kıyasla çok daha hassas olduğu için ses üzerindeki ufak değişikleri daha kolay fark edebilmektedir. Dolayısıyla fark edilmezlik sınırları içerisinde ses örtüsü içerisine veri gizleme işlemleri daha zordur. Bu sebeple, ses örtüsü içerisine veri gizleme üzerine yapılmış nitelikli çalışmaların sayısı oldukça azdır. Mevcut nitelikli çalışmalar çoğunlukla ileri sinyal işleme kuramları ve karmaşık matematiksel modellemeler üzerine kuruludur. Bu tez çalışmasının öncelikli amacı mevcut temel veri gizleme yöntemlerini karşılaştırmalı bir şekilde ve mümkün olduğunca yalın bir anlatımla işleyerek temel yöntemlerin algoritmalarını içeren açık kaynak kodlu bir kütüphane oluşturmaktır. İncelenen yöntemlerin algoritmaları MATLAB ortamında yazılarak performans iyileştirmeleri deneyler eşliğinde araştırılıp tartışılmaktadır.

Şekil 1.1. Veri gizleme üçgeni

Yöntemlerinin karşılaştırılması Şekil 1.1'de gösterilen kapasite, veri sağlamlığı ve fark edilmezlik üçgeni içerisinde incelenmektedir. Çalışmada kullanılan karşılaştırma ölçütleri, sağlamlık için bit hata oranı (BER) ve normalleştirilmiş ilinti (NC), kapasite için 1 saniye içerisine gizlenebilen bit sayısı (bit/sn) ve fark edilmezlik için gürültü oranı (SNR) ve öznel dinleme testlerinden oluşmaktadır. Veri sağlamlığında öncelikli olarak yöntemlerin sıkıştırma algoritmalarına karşı başarı oranları araştırılmıştır.

Veri gizleme yöntemleri gizlenen verinin geri çekilmesi aşamasında ilk örtüye ihtiyaç duyulup duyulmamasına bağlı olarak kaynak-bağımlı (non-blind) ve kaynak-bağımsız (blind) olmak üzere iki grupta incelenebilir. Kaynak-bağımlı yöntemlerde alıcı tarafında her iki dosyanın da mevcut olduğu varsayıldığından istatistiksel olarak karşılaştırma çok kolay yapılabildiği için yöntem geliştirmek oldukça kolay iken güvenliğin ise oldukça zayıf olduğu söylenebilir. Kaynak-bağımsız yöntemlerde ise alıcı tarafta sadece içerisine veri gizlemiş dosya olacağından, bu yöntemlerin geliştirilmesi ve uygulanması nispeten daha karmaşık olmasına rağmen çok daha kullanışlıdır.

Tez sürecinde kaynak-bağımsız yöntemler incelenmiş, özellikle tayf yayılımı ile veri gizleme ve yankı veri gizleme yöntemleri üzerine yoğunlaşılmıştır. Yankı veri gizleme yöntemlerinden tekli yankı veri gizleme, negatif-pozitif yankı veri gizleme, ileri-geri simetrik yankı veri gizleme, negatif-pozitif ve ileri-geri simetrik yankı veri gizleme, zaman yayılımlı yankı veri gizleme yöntemleri incelenerek algoritmaları yazılmıştır. Tayf yayılımı yönteminin veri saklama kapasitesini ve sıkıştırmaya karşı sağlamlık seviyesini paralel şekilde arttırma üzerine çalışmalar yapılmıştır.

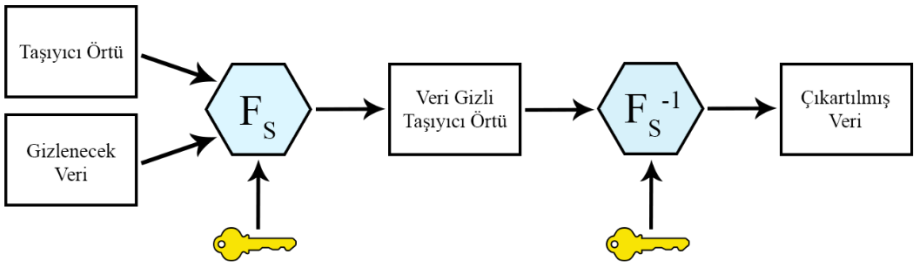
Steganografi nedir?

Veri gizleme bilimi, diğer adıyla steganografi, Yunanca steganos (gizli) ve graphein (yazı) kelimelerinden türetilmiş, iletilmek istenen bir mesaj ya da bilgiyi diğer bir taşıyıcı ortam içerisine gizleyerek aktarılabilme yöntemlerini araştıran bilim dalıdır.

Steganografi, kriptografi ile sıkça karıştırılmaktadır. Ancak, steganografide esas amaç iletilmek istenen veriyi herhangi bir taşıyıcı içerisine dışarıdan görülmeyecek bir şekilde gizleyerek iletmek iken, kriptografide amaç veriyi dışarıdan görünür fakat okunamaz hale getirerek iletmektedir.

Binlerce yıllık köklü bir tarihe sahip olan steganografi bilimi, teknoloji çağı öncesi dönemlerde iletilmek istenen mesajların fiziksel taşıyıcılar içerisine gizlenildiği uygulamalar ile doğmuştur. Tahta tabletler üzerine ince bir yazı ile yazılan mesaj balmumu ile kaplanarak gizli bir şekilde iletildikten sonra alıcı kişi tarafından balmumu tekrar eriterek gizli mesajı okunurmuş. Özellikle ikinci dünya savaşı sırasında gizli bir şekilde haberleşebilmek için mesajlar gazeteler içerisine belirli kuralara göre dağıtılarak, küçültülmüş mikroskobik resimler halinde ulakların kulak ve burun içleri, tırnak altları gibi yerlere gizlenerek iletilirmiş.

Teknolojinin gelişimi ile birlikte fiziksel veri gizleme uğraşları yerini dijital veri gizlemeye bırakmıştır. Dijital ortama aktarılmış görüntü, ses, video gibi veriler örtü veri olarak kullanılarak iletilmek istenilen mesajlar Şekil 1.3'teki gibi bu dijital taşıyıcılar aracılığıyla aktarılmaya çalışılmıştır.

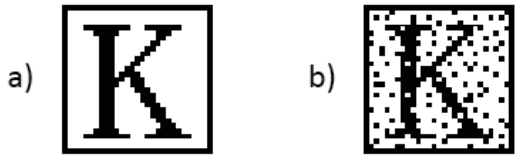


Şekil 1.2. Steganografide veri gizleme ve çekme işlemleri

Ses örtüsü içerisine veri gizleme uygulamaları ilk olarak 1970'li yıllarda karşımıza çıkmaktadır. The Beatles ve Pink Floyd başta olmak üzere birçok ünlü müzik grupları geri maskeleme (backmasking) olarak adlandırılan yöntem ile eserlerinin içerisine akış yönüne ters şekilde gizli mesajlar eklemişlerdir. Bu gizli mesajlar ancak kayıtlar tersten çalındığında anlaşılır hale gelmektedir. Örneğin, Linkin Park adlı müzik grubunun Announcement Service Public adlı parçası tersten dinlenince "dişlerini fırçalamalısın ve ellerini yıkamalısın" mesajı ortaya çıkmaktadır. (Anonim, 2015)

Steganografinin en yaygın kullanım alanı damgalama uygulamalarıdır. Damgalama, kopya koruma amacıyla eser sahibine ait bilgileri veri gizleme yöntemleri ile birlikte eserin içerisine gizleme işidir. Damgalamada önemli olan gizli verinin sağlamlığı ve fark edilmezliğidir.

Damgalama için gizlenen verinin hassasiyetini azaltmak için genel olarak eser sahibine ait bilgi Şekil 1.4'teki örnekteki gibi tek bitlik görüntü ile işlenerek gizlenir. Bu şekilde gizlenen bilgide yapılan saldırılar sonucu bozunmalar dahi olsa, geri çekilebilen kısım ile telif bilgilerinin kanıtlanabilmesi mümkün olacaktır.



Şekil 1.3. a) Gizlenmiş damga b) Saldırı altında geri çekilmiş damga

Damgalanmış ses dosyalarında, damganın bir takım saldırılara karşı dayanıklı olması beklenir. Bu saldırılar başlıca sıkıştırma işlemleri, dijital-analog-dijital dönüşümü, gürültü ekleme, yeniden örneklendirme, yankı ekleme, kırpma, frekans süzgeçleri gibi saldırı çeşitleridir. Damgalama algoritmaları kaldırabildiği saldırı sayısı kadar güçlü, fark edilmez olduğu kadar da etkilidir.

Buna karşın, fark edilebilir damgalama işlemleri de yaygın olarak uygulanmaktadır. Buna göre, herhangi bir ses dosyasına başka bir ses dosyası dışarıdan rahatça duyulabilecek şekilde damga olarak eklenebilmektedir.

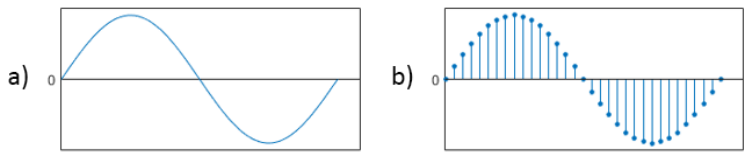
2. KURAMSAL TEMELLER

2.1. Temel Dijital Ses İşleme Terimleri

2.1.1. Sesin Yapısı

Analog ses sinyali, ses dalgalarının ortam içerisinde oluşturduğu basınç kuvvetini elektrik ortamında gerilim ile eşleştirilerek temsil edilen sürekli ölçekli ses sinyalidir. Doğadaki sesler analog ses sinyalleri şeklindedir ve insan kulağı tüm sesleri analog olarak algılamaktadır.

Dijital ses sinyali, sesin dijital ortama aktarılabilmesi amacıyla analog ses sinyalinin basınç değerlerinin Şekil 2.1'deki gibi belirli zaman aralıkları ile örneklenmesi sonucu elde edilen sayı değerlerinin zaman ekseninde sıralanmasıyla oluşturulan yapay ses sinyaldir.



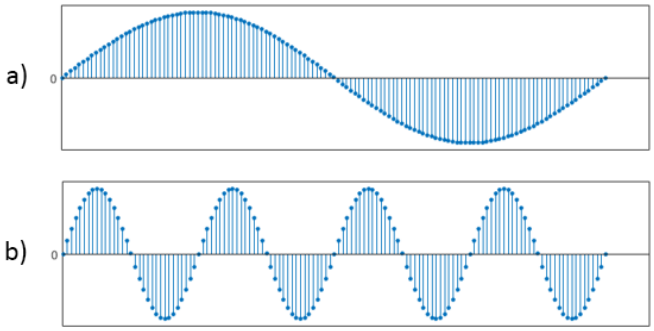
Şekil 2.1. a) Temsili analog sinyal b) Temsili dijital sinyal

Dijital ses oluşturulurken bir birim zaman içerisinde kaç adet örnekleme yapılacağı belirlenir. Örnekleme sayısı ne kadar çok olursa analog ses sinyaline o kadar yakın bir dijital ses sinyali elde edilir.

Genlik: Bir ses örneğinin zaman ekseninde ortalama basınca göre anlık basınç değişiminin sayısal temsilidir.

Periyot: İki yineleme arasında geçen süreye periyot denir, T ile gösterilir.

Frekans: Bir ses sinyalinin birim zaman içerisinde ürettiği dalga sayısıdır. Periyodun çarpmaya göre tersidir, $f = T^{-1}$. Ses kaynağının birim zamanda ürettiği titreşim sayısı arttıkça insan kulağı tarafından daha ince, azaldıkça ise daha kalın olarak algılanır. Bir saniye başına düşen titreşim sayısı Hertz (Hz) ile ifade edilir. 1 Hz saniyede 1 titreşimi, 1 kHz ise saniyede 1000 titreşimi temsil etmektedir.



Şekil 2.2. a) Düşük frekanslı ayrık sinyal b) Yüksek frekanslı ayrık sinyal

Desibel: Sesin şiddetini tanımlamak için kullanılan, iki ses değerinin güçlerinin oranına dayalı logaritmik ve boyutsuz bir birimdir, dB ile gösterilir. P ve P_0 sesin iki farklı zamandaki watt birimindeki güçleri olmak üzere desibel değeri Denklem 2.1'deki gibi hesaplanır.

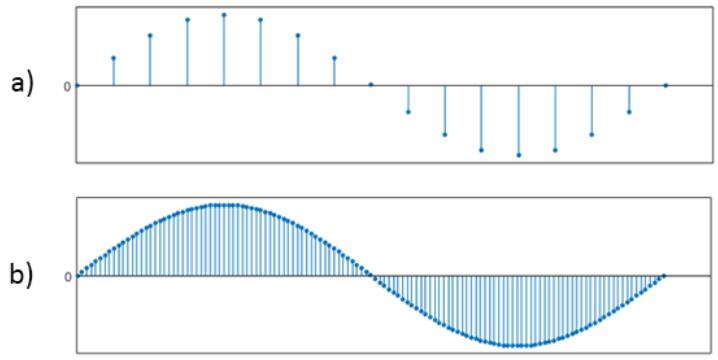
$$L_p = 10\log_{10}\left(\frac{P}{P_0}\right) \text{ dB.} \qquad (2.1)$$

Örneğin; belirli şiddetteki bir sesin güç değeri 2 katına çıkartılır ise sesin desibeli, $10\log_{10}\left(\frac{2P_0}{P_0}\right) = 10\log_{10}(2) \approx 3$ dB şeklinde hesaplanır.

2.1.1.1. Analog sesin dijitalleştirilmesi

Örnekleme aşaması

Analog ses sinyalinin basınç değerlerine sayısal değerler atamak için belirli zaman aralıklarında küçük parçalara ayrılır. Parçalama sayısına örnekleme oranı denir. Şekil 2.3'te gösterildiği gibi örnekleme oranı arttıkça analog sese daha yakın bir dijital ses elde edilir. Nyquist kriterine göre ideal bir örneklemeye örnekleme sayısı bant genişliğinin 2 katından daha yüksek seçildiğinde ulaşılabilir. İnsan kulağı 18 kHz yukarısındaki ses sinyalleri genel olarak ayırt edemediği için Nyquist kriterine göre yaklaşık olarak 36 kHz yukarısında örneklenmiş bir dijital ses insan kulağı için yüksek ve yeterli kalitededir. Telefon kalitesindeki ses 7.2 kHz ile 8 kHz arasında, CD kalitesindeki ses ise genellikle 44.1 kHz ile örneklenir. (McLoughlin, 2009:4)



Şekil 2.3. a) Düşük örnekli ayrık sinyal b) Yüksek örnekli ayrık sinyal

Nicemleme ve kodlama aşaması

Örnekleme oranı belirlendikten sonra analog ses sinyalinin sayısalaştırılabilmesi için örneklere yaklaşık değer ataması yapmak gerekir. Basınç değerlerine göre belirli oranlarda dizinleme yapılarak sürekli sinyalin gerçek değeri en yakın dizine atanır. Bu işleme nicemleme (quantization) denir. Nicemlemede kullanılan bit sayısına bit derinliği denir. 8 bit ile nicemlenen ses örneği $2^8 = 256$ farklı değerden birine atanırken, 16 bit ile nicemlenen ses örneği $2^{16} = 65536$ farklı değer alabilir. Nicemleme aşamasının ardından dijital ses her hangi bir dijital ses formatı altında kodlanır.

2.1.1.2. Dijital sesin sıkıştırılması

Analog bir sesin dijitalleştirmesinin ardından sıkıştırılmamış bir şekilde bilgisayar ortamında saklanır ise disk üzerinde çok fazla yer kaplayacaktır. Sıkıştırılmamış bir ses dosyasının diskte kapladığı alan aşağıda belirtilen değişkenler ile Denklem 2.2'deki gibi hesaplanır.

SR : Saniye başına örnek sayısı
BD : Bit derinliği
L : Süre (sn)
K : Kanal sayısı

$$\text{Dosya boyutu} = SR \times BD \times L \times K \text{ bit} \qquad (2.2)$$

Örneğin 10 saniyelik, 44100 örnekleme oranına sahip ve örnek başına 16 bit ile nicemlenmiş 2 kanallı sıkıştırılmamış bir ses dosyasının boyutu,

$$\begin{aligned}\text{Dosya boyutu} &= 44100 \times 16 \times 10 \times 2 \\ &= 14{,}120{,}000 \text{ bit} \\ &= 1{,}764{,}000 \text{ bayt} \\ &\approx 1.68 \text{ MB}\end{aligned}$$

Olarak hesaplanır. Buradan hareketle 3 dakikalık bir müzik dosyası diskte yaklaşık olarak $30,3\,\text{MB}$ gibi büyük bir yer kaplayacaktır. Bu yüksek boyut ise internet üzerinden dosya aktarımında güçlük yaratacaktır. Dijital sinyal işleme ve veri sıkıştırma yöntemleri kullanılarak bu soruna çözüm aranmıştır.

Günümüzde en çok kullanılan ses sıkıştırma formatlarından biri olan MP3 ile dijital ses içerisinden insan kulağının duyamayacağı frekanslar ayıklanarak 1/10 oranında sıkıştırma imkânı sağlanmıştır. MP3'e alternatif olarak WMA, AAC ve OGG ise diğer sıkıştırılmış ses formatlarındandır.

2.1.2. Temel Dönüşümler

2.1.2.1. Ayrık Fourier dönüşümü

DFT, zaman uzayındaki bir ayrık ses sinyalini frekans uzayında ifade etmek için en sık kullanılan dönüşümlerin başında gelir. Gerçel ya da karmaşık bir sinyal frekans uzayında karmaşık formda ifade edilmektedir.

Tanım: $x \in \mathbb{C}^N$ olmak üzere, ayrık Fourier dönüşümü $F\{\cdot\}$ Denklem 2.3'te ve ayrık ters Fourier dönüşümü $F^{-1}\{\cdot\}$ Denklem 2.4'teki gibi tanımlanır.

$$F\{x[n]\} = X[k] = \sum_{n=0}^{N-1} x[n] e^{-2\pi i n k / N},\ k = 0,1,2,...,N-1 \tag{2.3}$$

$$F^{-1}\{X[k]\} = x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{2\pi i n k / N},\ n = 0,1,2,...,N-1 \tag{2.4}$$

Burada, $i = \sqrt{-1}$ ve $e = \lim_{n \to \infty} \left(1 + \frac{1}{n}\right)^n$ dir.

Fourier dönüşümünün özellikleri

Doğrusallık özelliği

Teorem: $a,b \in \mathbb{C}$ sabit sayılar ve $x, y \in \mathbb{C}^N$ olmak üzere ayrık Fourier dönüşümü doğrusal bir dönüşümdür, Denklem 2.5 sağlanır.

$$F\{ax + by\} = aF\{x\} + bF\{y\} \tag{2.5}$$

İspat: Ayrık Fourier dönüşümünün tanımı ve integral işleminin doğrusallığından aşikârdır (Smith, 2007, Fourier Theorems, Linearity).

Eşlenik simetri özelliği

Teorem: $x \in \mathbb{R}^N$ ise, $X = F\{x\}$ eşlenik (hermitian) simetriktir.

$$x \in \mathbb{R}^N \Rightarrow X[k] = \overline{X[N-k]} \tag{2.6}$$

İspat: $X[k]$ ve $X[N-k]$ ayrık Fourier tanımı ile yazılırsa,

$$FFT\{x_k\} = X_k = \sum_{n=0}^{N-1} x_n e^{-i\frac{2\pi nk}{N}}$$

$$FFT\{x_{N-k}\} = X_{N-k} = \sum_{n=0}^{N-1} x_n e^{-i\frac{2\pi n(N-k)}{N}}$$

$$= \sum_{n=0}^{N-1} x_n e^{-i2\pi k} e^{i\frac{2\pi nk}{N}}$$

$\forall n$ için Euler eşitliğinden $e^{-i2\pi k} = \cos(2\pi k) - i\sin(2\pi k) = 1$ olacağından, ikinci eşitlikte $e^{-i2\pi k} = 1$ yazılırsa $\overline{X_{N-k}} = X_k$ eşitliği elde edilir.

$$X_{N-k} = \sum_{n=0}^{N-1} x_n e^{i\frac{2\pi nk}{N}} = \overline{\sum_{n=0}^{N-1} x_n e^{-i\frac{2\pi nk}{N}}} = \overline{X_N}$$

$$\overline{X_N} = X_{N-k} \Rightarrow \left(\overline{X_N}\right) = X_N = \overline{X_{N-k}}$$

Buradan, $x \in \mathbb{R}^N \Rightarrow X[k] = \overline{X[N-k]}$ koşullu önermesi sağlanmış olur.

Gerçel bir ses sinyali için, ayrık Fourier dönüşümü üzerinde yapılan düzenleme işlemlerinde eşlenik simetri özelliğinin dikkate alınması oldukça önemlidir. Bu kısım, faz kodlaması ile veri gizleme yöntemi anlatılırken kullanılacaktır.

2.1.2.2. Hızlı Fourier dönüşümü

Fourier dönüşümü hesaplanırken temel tanımı kullanarak dönüşümü yaklaşık olarak daha hızlı hesaplayan Hızlı Fourier Dönüşümleri (FFT) kullanılmaktadır. DFT ve FFT için zaman karmaşıklıkları analiz edildiğinde standart DFT algoritmasının karmaşıklığı $O\left(N^2\right)$ iken FFT 'nin $O\left(N\log N\right)$ olarak hesaplanmaktadır.

Fourier dönüşümünü daha hızlı hesaplamak için birçok FFT algoritması vardır. Good–Thomas, Bruun, Rader, Bluestein, Altıgensel FFT ve Cooley-Tukey bu algoritmalardan bazılarıdır. Bunlar arasında Cooley-Tukey algoritması en yaygın olarak kullanılanıdır (Cooley ve Tukey, 1965).

2.1.2.3. Ayrık kosinüs dönüşümü

Ayrık kosinüs dönüşümü, herhangi bir sonlu ayrık veri noktaları dizisinin kosinüs fonksiyonları cinsinden ifade edilmesidir. Bu işlem birçok şekilde yapılabileceği için, birçok farklı kosinüs dönüşümü tanımlanmıştır.

$\mathbf{x}=\left\{x_1,\, x_2,\, ...,\, x_N\right\}$ için 2. tip ayrık kosinüs dönüşümü Denklem 2.7 ve tersi Denklem 2.8'deki gibi tanımlanır (Ahmed vd., 1974).

$$X_k=\sum_{n=0}^{N-1}x_n\cos\left(\frac{\pi\left(2n+1\right)k}{2N}\right),\quad k=0,1,...,N-1 \tag{2.7}$$

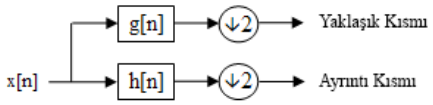
$$x_n=\frac{1}{N}X_0+\frac{2}{N}\sum_{k=1}^{N-1}X_k\cos\left(\frac{\pi\left(2k+1\right)n}{2N}\right),\ n=0,1,...,N-1 \tag{2.8}$$

Ayrık kosinüs dönüşümü ayrık Fourier dönüşümü ile benzer özellikler taşımaktadır. Nitekim, ayrık Fourier dönüşümündeki karmaşık üstel fonksiyon ile veri girdisi kosinüs ve sinüs fonksiyonları cinsinden yazılırken, ayrık kosinüs dönüşümünde veri girdileri gerçel kosinüs fonksiyonları cinsinden yazılır.

2.1.2.4. Ayrık dalgacık dönüşümü

Dalgacık dönüşümü dijital sinyal işlemede en sık kullanılan bir diğer dönüşümdür. Dönüşüm için kullanılan dalgacık fonksiyonu ile birlikte ses sinyali alçak geçiren (low-pass) ve yüksek geçiren (high-pass) süzgeçlerden geçirilerek yaklaşık ve ayrıntı olmak üzere iki parçaya ayrılır. Bu işlem, $x[t]$ sinyali ile süzgeç çekirdekleri evrişim işlemine sokularak yapılır.

$$y[n] = (x * g)_n = \sum_{m=-\infty}^{\infty} x[m]\,g[n-m]$$

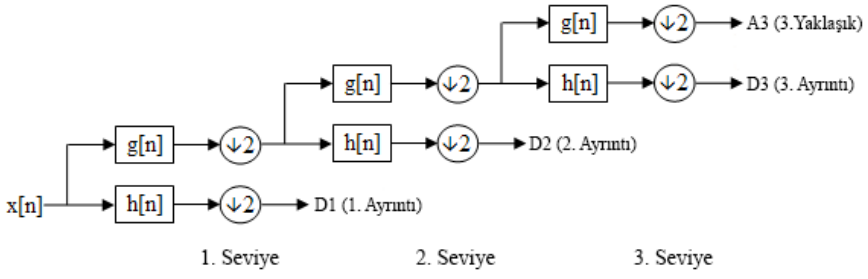


Şekil 2.4. Ayrık dalgacık dönüşümü

Yüksek geçiren süzgeç ile alçak geçiren süzgeç aynı anda uygulanarak Şekil 2.4 ve Şekil 2.5'teki gibi düşük frekanslar ile yaklaşık kısmı, yüksek frekanslar ile de ayrıntı kısmı oluşturulur. Frekanslar yarıya bölündüğü için Denklem 2.9 ve 2.10'daki gibi yarı oranda örnek-düşürme işlemi uygulanır. Örnek-düşürme işlemi $\downarrow$ ile, örnek-yükseltme işlemi ise $\uparrow$ ile gösterilir.

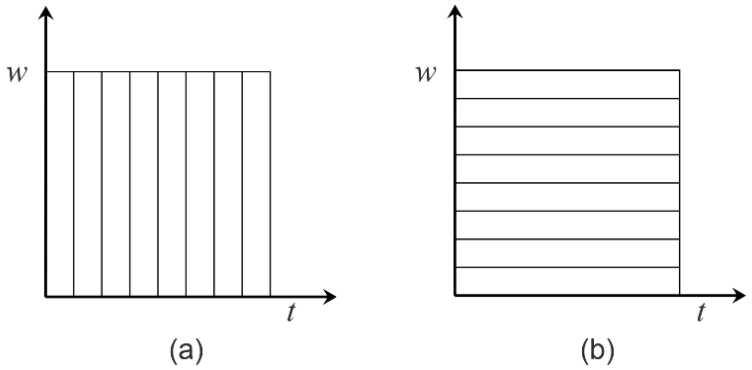
$$y_{düş}[n] = (x * g)_n = \sum_{m=-\infty}^{\infty} x[m]\,g[2n-m] \tag{2.9}$$

$$y_{yük}[n] = (x * h)_n = \sum_{m=-\infty}^{\infty} x[m]\,h[2n-m] \tag{2.10}$$



Şekil 2.5. 3 seviyeli ayrık dalgacık dönüşümü

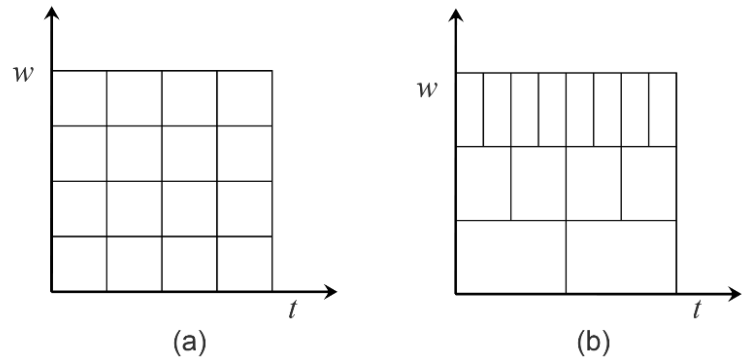
Ayrık Fourier dönüşümünde zaman kavramının yerini Şekil 2.6'da gösterildiği şekilde frekans kavramı almaktadır. Diğer bir ifadeyle, hangi frekans değerinin hangi zaman aralığından geldiği belirsizdir.



Şekil 2.6. a) Zaman uzayında zaman-frekans çözünürlüğü b) DFT altında zaman-frekans çözünürlüğü

Kısa zamanlı ayrık Fourier dönüşümü (DSTFT) ile kullanılan pencerenin genişliğine bağlı olarak zaman-frekans çözünürlükleri Şekil 2.7'deki gibi farklı oranlarda elde edilebilir. Ancak frekans çözünürlüğü ile zaman çözünürlüğü arasında ters orantı vardır. Biri artarken diğeri azalacaktır.

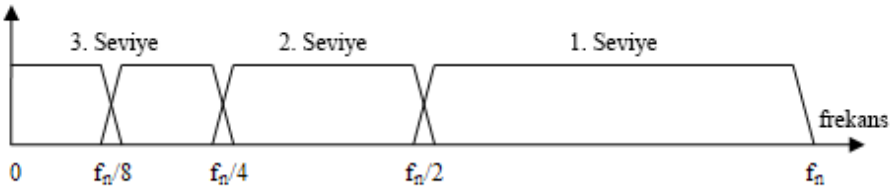
Ayrık dalgacık dönüşümünde (DWT) ise çoklu pencereler kullanılarak aşamalı olarak Şekil 2.7'deki gibi çoklu çözünürlük elde edilir. Zaman kavramının önemli olduğu durumlarda yüksek seviyeli dönüşüm elemanları, frekans kavramının önemli olduğu yerlerde ise düşük seviye dönüşüm elemanları tercih edilir.



Şekil 2.7. a) DSTFT altında zaman-frekans çözünürlüğü b) 3 seviyeli DWT altında zaman-frekans çözünürlüğü

Ayrık dalgacık dönüşümünün veri gizlemedeki en büyük işlevi, verinin hangi frekans aralığına gizlenebileceğinin dönüşüm yardımıyla seçilebilmesidir. Yaklaşık kısmı düşük frekanslardan oluştuğu için, sinyali oluşturan baskın frekansları ihtiva etmektedir. Dolayısıyla, yaklaşık kısım üzerine yapılacak gizleme işlemi birçok saldırıya karşı daha dayanıklı olacaktır. Tersine, ayrıntı kısmına yapılacak gizleme işlemi ile daha kırılgan ancak daha zor algılanabilir bir sonuç elde edilecektir. Gizleme işlemleri genellikle 3. seviye ya da 4. seviyenin yaklaşık kısmı üzerine yapılmaktadır.

MP3 gibi birçok sıkıştırma standardı sıkıştırma için örnek sayısını azaltırken ilk önce MDCT gibi dönüşümler ile duyma eşiğinin dışarısında kalan yüksek ve düşük frekansları kırpmaktadır. Dolayısıyla, uygun bir dönüşüm fonksiyonu ile gizleme işlemini duyma eşiği içerisinde kalan frekans bölgesine yapmak özellikle damgalama için nispeten daha dayanıklı sonuçlar verecektir.



Şekil 2.8. 3 seviyeli DWT'nin frekans karşılığı

Dalgacık fonksiyonu olarak gizleme uygulamalarında en sık Haar dalgacığı kullanılmakla beraber; Daubechies, Symlet, Coiflet, BiorSplines, ReverseBior, Meyer, Morlet, Mexican Hat, Gaussian ve Shannon gibi birçok dalgacık aileleri vardır (Daubechies, 1997; Misiti vd., 2007).

2.1.2.5. Kepstrum dönüşümü

Tanım: $F\{\cdot\}$ ayrık Fourier dönüşümü, $F^{-1}\{\cdot\}$ ayrık ters Fourier dönümü ve $\ln\{\cdot\}$ karmaşık sayılarda tanımlı doğal logaritma olmak üzere, $x[t]$ ayrık sinyalinin karmaşık kepstrumu Denklem 2.11'deki gibi tanımlanır.

$$\hat{x}_c[t] = F^{-1}\left\{\ln\left\{F\left\{x[t]\right\}\right\}\right\} \tag{2.11}$$

Tanım: $\hat{x}_c[t]$, $x[t]$ ayrık sinyalinin karmaşık kepstrumu olmak üzere ters kepstrumu Denklem 2.12'deki gibi tanımlanır.

$$x[t] = F^{-1}\left\{\exp\left\{F\left\{\hat{x}_c[t]\right\}\right\}\right\} \tag{2.12}$$

Tanım: $z = a + ib$ karmaşık sayısı için $\mathrm{Re}(z) = a$ gerçel kısım ve $|z| = \sqrt{a^2 + b^2}$ olmak üzere, $x[t]$ sinyalinin gerçel kepstrumu Denklem 2.13'teki gibi tanımlanır.

$$\hat{x}_r[t] = \mathrm{Re}\left(F^{-1}\left\{\ln\left\{\left|F\left\{x[t]\right\}\right|\right\}\right\}\right) \tag{2.13}$$

2.1.3. Temel İşlemler

2.1.3.1. Evrişim (convolution) işlemi

Tanım: $x, y \in \mathbb{C}^N$ için x ve y'nin evrişimi Denklem 2.14'teki gibi tanımlanır.

$$(x * y)_n = \sum_{m=0}^{N-1} x[m]y[n-m] \tag{2.14}$$

Teorem: Evrişim işlemi değişmelidir. $(x * y)_n = (y * x)_n$ eşitliği sağlanır.

İspat: $(x * y)_n$ evrişim işleminde $n - m = l$ dönüşümü yapılırsa:

$$(x * y)_n = \sum_{m=0}^{N-1} x[m](m)y[n-m] = \sum_{l=n}^{n-N+1} x[n-l]y[l] = \sum_{l=0}^{N-1} y[l]x[n-l] = (y * x)_n$$

Birim dürtü: Ayrık sinyaller için $\delta[n]$ Kronecker-delta fonksiyonu (sürekli sinyaller için $\delta(n)$ Dirac-delta fonksiyonu) birim dürtü (unit impulse) olarak adlandırılır ve Denklem 2.15'teki gibi tanımlanır.

$$\delta[n] = \begin{cases} 1, & n = 0 \text{ iken} \\ 0, & n \neq 0 \text{ iken} \end{cases} \tag{2.15}$$

Dijital sinyal işlemede sinyaller birim dürtüler ile ifade edilmektedir. Her bir sinyal birim zamana denk gelen enerji genliği ile birim dürtülerin çarpımlarının toplamları cinsinden yazılabilir.

$x[t]$ herhangi bir ayrık ses sinyali olmak üzere;

$$(x * \delta)[t] = x[t] \quad \text{ve} \quad x[t] * \delta[t-k] = x[t-k]$$

ifadeleri özellikle yankı veri gizleme yönteminde sıkça kullanılacaktır.

Evrişim Teoremi: $x, y \in \mathbb{C}^N$ olmak üzere Denklem 2.16'daki gibi iki fonksiyonun evrişimleri Fourier dönüşümü altında noktasal çarpıma dönüşür.

$$F\{x * y\} = F\{x\} \cdot F\{y\} \tag{2.16}$$

Dolayısıyla, zaman uzayında evrişim işleminin frekans uzayında karşılığı noktasal çarpımdır. Bu teorem, büyük boyutlu diziler ile işlem yaparken standart evrişim işleminin hızını arttırmak için kullanılabilir. Nitekim, algoritma analizi ile zaman karmaşıklıkları hesaplandığında, standart evrişim $O(N^2)$ iken FFT kullanılarak alınan evrişim $O(N \log N)$'dir.

$x, y \in \mathbb{C}^N$ ve $L = boy(x) + boy(y) - 1$ olmak üzere L-noktalı FFT ve IFFT kullanılarak evrişim Denklem 2.17'deki gibi hesaplanabilir.

$$x * y = IFFT\left(FFT(x, L) \cdot FFT(y, L)\right) \tag{2.17}$$

Burada FFT (hızlı Fourier dönüşümü) uygulayabilmek için x ve y'nin boyları L'ye eşit oluncaya kadar sağ taraftan sıfır eklemesi (zero padding) yapılır.

2.1.3.2. İlinti (correlation) işlemleri

İlinti işlemleri herhangi iki veri kümesi arasındaki benzerlikleri ölçmek için kullanılırlar. Nesne ve desen tanıma, ses tanıma, ses sinyallerindeki gürültü ve yankıların tespiti ve giderilmesi gibi birçok alanda kullanılmaktadır.

Doğrusal ilinti: $x, y \in \mathbb{C}^N$ olmak üzere ilinti işlemi Denklem 2.18'de tanımlanır.

$$(x \star y)_k = \sum_{n=0}^{N-1} \overline{x[n]}\, y[n+k], \; k = 0,1,2,\ldots,N-1 \tag{2.18}$$

Burada k değeri y sinyalini x üzerinde kaydırmak için kullanılan gecikme (time-lag) miktarıdır.

Çapraz ilinti: $x, y \in \mathbb{C}^N$ olmak üzere, çapraz ilinti Denklem 2.19'daki gibi tanımlanır.

$$r_{xy}[k] = \frac{1}{N}(x \star y)_k = \frac{1}{N}\sum_{n=0}^{N-1}\overline{x[n]}\,y[n+k],\ k = 0,1,2,...,N-1 \tag{2.19}$$

Öz-ilinti: $x \in \mathbb{C}^N$ olmak üzere, x 'in kendisi ile çapraz ilintisine öz-ilinti denir ve Denklem 2.20'deki gibi tanımlanır.

$$r_x[k] = \frac{1}{N}(x \star x)_k = \frac{1}{N}\sum_{n=0}^{N-1}\overline{x[n]}\,x[n+k] \tag{2.20}$$

Öz-ilinti işlemi hermitian özelliğini sağlar, yani $r_x[-k] = \overline{r_x[k]}$ eşitliği sağlanır. Ayrıca $x \in \mathbb{R}^N$ ise r_x simetrik ve gerçel tanımlı tek fonksiyondur.

Evrişim teoremi ilinti işlemleri için de geçerlidir. İki fonksiyonun ilintisinin Fourier dönüşümü Fourier dönüşümlerinin noktasal çarpımları olarak Denklem 2.21'deki gibi yazılabilir.

$$F\{x \star y\} = \overline{F\{x\}} \cdot F\{y\} \tag{2.21}$$

Bu teorem evrişim işleminde olduğu gibi büyük boyutlu dizilerin ilintileri alınırken algoritmayı hızlandırmak için FFT ile birlikte uygulanabilmektedir.

2.2. Cebirsel Ayrışımlar

2.2.1. Tekil Değer Ayrışımı

Üniter matris: Q, gerçel veya karmaşık sayılarda tanımlı kare matris, Q^* onun eşlenik devriği (transpose) ve I birim matris olmak üzere,

$$QQ^* = Q^*Q = I \tag{2.22}$$

Denklem 2.22 sağlanıyor ise, Q matrisine üniter matris denir. Buradan $Q^{-1} = Q^*$ eşitliği de elde edilir.

Köşegen matris: *S*, gerçel veya karmaşık sayılarda tanımlı herhangi bir matris olmak üzere, köşegen dışındaki tüm elemanları sıfıra eşit ise *S* matrisine köşegen matris denir.

Tekil değer ayrışımı: $A_{m\times n}$ gerçel veya karmaşık sayılarda tanımlı herhangi bir matris olmak üzere, $U_{m\times m}$ ve $V_{n\times n}$ gerçel veya karmaşık sayılarda tanımlı üniter matrisler, $S_{m\times n}$ negatif olmayan gerçel sayılarda tanımlı köşegen matris için,

$$A = USV^{*} \tag{2.23}$$

Denklem 2.23'e *A* matrisinin tekil değer ayrışımı denir. Eğer *A* bir kare matris ise köşegen elemanları azalmayan şekilde sıralı tek türlü bir *S* matrisi vardır.

$$\begin{pmatrix} a_{11} & \cdots & a_{1m} \\ \vdots & \ddots & \vdots \\ a_{mm} & \cdots & a_{mm} \end{pmatrix}_{m\times m} = \begin{pmatrix} u_{11} & \cdots & u_{1m} \\ \vdots & \ddots & \vdots \\ u_{m1} & \cdots & u_{mm} \end{pmatrix}_{m\times m} \begin{pmatrix} s_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & s_m \end{pmatrix}_{m\times m} \begin{pmatrix} v_{11} & \cdots & v_{1m} \\ \vdots & \ddots & \vdots \\ v_{m1} & \cdots & v_{mm} \end{pmatrix}^{*}_{m\times m}$$

Tekil değer ayrışımı büyük boyuttaki karmaşık verilerin analizinde, ses ve görüntü işleme, veri madenciliği ve veri sıkıştırma uygulamaları gibi birçok alanda kullanılmaktadır.

2.2.2. QR ve QL Ayrışımları

Üst üçgensel matris: *R*, gerçel veya karmaşık sayılarda tanımlı herhangi bir matris olmak üzere, köşegenin altında kalan tüm elemanları sıfır ise *R* matrisine üst üçgensel matris denir.

Alt üçgensel matris: *L*, gerçel veya karmaşık sayılarda tanımlı herhangi bir matris olmak üzere, köşegenin üstünde kalan tüm elemanları sıfır ise *L* matrisine üst üçgensel matris denir.

Ortogonal matris: Denklem 2.24'teki gibi devriği ile çarpımı birim matrise eşit olan, bir başka deyişle tersi devriğine eşit olan gerçel tanımlı karesel matrise ortogonal matris denir.

$$QQ^{T} = Q^{T}Q = I \quad \text{ve} \quad Q^{-1} = Q^{T} \tag{2.24}$$

QR ayrışımı: A, gerçel sayılarda tanımlı herhangi bir kare matrisi olmak üzere, Q ortogonal matrisi ve R üst üçgensel matrisleri için,

$$A = QR \tag{2.25}$$

Denklem 2.25'e A matrisinin QR ayrışımı denir. Eğer A tersinir bir matris ve R matrisinin köşegen elemanları pozitif ise, bu ayrışım tek türlüdür.

QR ayrışımını hesaplamak için Gram–Schmidt yöntemi başta olmak üzere birçok algoritma vardır. Benzer şekilde, L üst üçgensel matris olmak üzere QL ayrışımı ve RQ ile LQ ayrışımları da tanımlanıp hesaplanabilir.

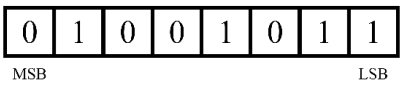
3. MATERYAL VE YÖNTEM

Bu bölümde ses örtüsü içerisine veri gizleme üzerine yapılmış çalışmaları içeren bazı temel kaynak-bağımsız (blind) yöntemlerin ve algoritmalarının olabildiği ölçüde yalın bir şekilde işlenmesi amaçlanmıştır. İncelenen yöntemlerin kaynak kodları ekte verilmektedir.

3.1. Temel Veri Gizleme Yöntemleri

3.1.1. Önemsiz Bit Kodlaması

En önemsiz bit (LSB) kodlaması veri gizleme uygulamalarında kullanılan en eski yöntemlerden biridir. Zaman ekseninde her örneğin kendi başına bir değeri temsil ettiği ses formatlarında en önemsiz bit üzerinde yapılacak değişiklikler sesin genel yapısı üzerinde fark edilemeyecek kadar küçük etkilere sebep olacağından, gizleme işlemi Şekil 3.2'deki gibi önemsiz bitler değiştirilerek yapılabilmektedir.



Şekil 3.1. 8 bit üzerinde en önemli ve en önemsiz bitler

Şekil 3.1'de 8 bitlik örnek üzerinden gösterilen ikilik tabandaki 1 baytlık verinin onluk tabandaki karşılığı basit bir işlemle aşağıdaki gibi hesaplanır.

$$0\times 2^7 + 1\times 2^6 + 0\times 2^5 + 0\times 2^4 + 1\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 = 113$$

Yukarıdaki örnekteki en önemsiz ilk bit 0 biti ile değiştirilirse onluk tabandaki 113 değeri 112 olarak değişecektir. Tersine, en önemli bit (MSB) kısmındaki ilk bit 1 biti ile değiştirilirse, verinin onluk tabandaki karşılığı 241 olarak değişecektir.

Şekil 3.2. 16 bitlik örtü ses üzerinde en önemsiz bit değişim işlemi

WAV gibi kayıplı sıkıştırma işlemi uygulanmayan dosya türleri üzerinde gizleme işlemi oldukça basit bir şekilde her bir örneğin son bitleri gizlenecek dosyanın veri bitleri ile değiştirilerek yapılabilmektedir.

3.1.1.1. Temel algoritma

Bu bölümde WAV formatı içerisine önemsiz bit kodlaması ile veri gizleme ve veri çekme işlemleri incelenmiştir. Çalışmanın MATLAB kodları Ek 1.2'de mevcuttur.

Değişkenler ve gösterimler:

I → Ses örtüsünün örnek sayısı
L → Her bir örneğin bit uzunluğu
w → Gizlenecek bit dizisi
m → Gizlenecek bit dizisinin uzunluğu

Veri gizleme işlemi

i. Taşıyıcı ses dosyasının örnekleri içeren veri kısmı ikilik veri olarak açılıp, $\{s[i,j]: 1 \le i \le I, 1 \le j \le L\}$ matrisine atanır.

Ses sinyalleri çoğunlukla 16 bit düzeyinde nicemlendiği için genellikle $L = 16$ olacaktır.

ii. Ses örtüsü içerisine gizlenecek olan w_i veri bitlerinin uzunluğu ile ses örtüsünün örnek sayısı karşılaştırılarak gizleme işi için her bir örnek üzerinde kaçar tane bitin kullanılacağı belirlenir.

Örneğin, $m > I$ ise her bir örneğin son 2 biti gizleme için kullanılabilir.

iii. $1 \le k \le m$ ve $m \le I$ için gizleme işlemi örneklerin son bitleri veri bitleri ile değiştirilerek yapılır.

$$s[k, L] = w[k]$$

iv. Taşıyıcı ses dosyasının düzenlenmiş hali ile birlikte WAV formatında kaydedilir.

Veri çekme işlemi

i. İçerisine veri gizlenmiş olan ses dosyasının örnekleri içeren veri kısmı ikilik veri olarak açılıp, $\{x[i, j]: 1 \le i \le I, 1 \le j \le L\}$ matrisine atanır.

ii. $1 \le k \le m$ için m sayısı geri okunacak $w[k]$ bit dizisinin uzunluğu ve L sayısı verinin gizlenmiş olduğu bit olmak üzere gizlenmiş veri aşağıdaki gibi geri okunur.

$$w[k] = \begin{cases} 0, & x[k, L] = 0 \text{ ise} \\ 1, & x[k, L] = 1 \text{ ise} \end{cases}$$

LSB kodlaması ile veri gizlenmiş ses örtüsü içerisinden veriyi geri çekme adımı oldukça kolaydır. Ancak, veri çekme aşamasında gizlenmiş verinin uzunluğunun ve yerinin alıcı tarafta da bilindiği varsayılmaktadır.

Alıcı taraftaki geri dönüş algoritmasının girdi değerlerini en az indirgemek için gizleme işleminin önemli parametreleri gizlenen veri ile birlikte, taşıyıcı ses içerisine bilinen yer ve uzunlukta gömülebilir. Böylelikle gizlenmiş veriyi çıkartmak için gerekli olan veri uzunluğu, gizleme yeri ve şifre karşılaştırma gibi parametreler karşı tarafa taşıyıcı ses ile birlikte iletilebilir.

3.1.1.2. Diğer çalışmalar

LSB kodlaması ile veri gizleme işlemleri kullanılacak bit sayısı üzerinden sabit sayılı ya da değişken sayılı olarak yapılabilmektedir. Sabit sayılı yapılan gizlemede her örnek üzerinde sabit ve eşit sayıdaki önemsiz bitler değiştirilirken, değişken sayılı olarak yapılan gizlemede kullanılacak bit sayısı ses örneğinin karakteristik özelliklerine göre değişken olarak seçilir.

16 bit ile nicemlenmiş ses örneği üzerinde en önemsiz 3-4 bitten daha fazla sayıdaki bitin değişimi ses örtüsünde eklenmiş gürültü etkisi yaratacaktır. Ses örneğinin diğer bitleri de sondaki değişimleri en aza indirgeyecek şekilde değiştirilirse, daha az gürültü ile daha fazla veri saklama imkânı sağlanabilir (Cvejic ve Seppänen, 2002). Ayrıca en önemsiz bit kodlamasının veri saklama kapasitesi dalgacık dönüşümü altında ayrıntı kısımlarına yapılarak da arttırılabilir (Cvejic ve Seppänen, 2002).

3.1.1.3. Sonuç

Ses sinyalleri genellikle 16 bit ile sayısallaştırılmaktadır. Yani zaman ekseninde her bir örnek genellikle 16 bit ile ifade edilmektedir. Ancak ses sinyalleri sayısallaştırıldıktan sonra WAV yerine MP3, AAC ya da WMA gibi ses sıkıştırma standartları ile sıkıştırılarak kaydedilirse, her bir örneğin temsili ve en önemsiz bit kavramı sıkıştırma standardının türüne göre değişecektir.

Örnek olarak MP3 standardı ile kaydedilmiş bir ses dosyası üzerinde en son bit en önemsiz bit değildir. Bu format üzerinde Huffman kodlaması ile bağlantılı olarak ardışık çerçeveler birbirleri ile ilintilidir. Bir çerçevenin son biti üzerinde yapılacak değişiklik diğer çerçeveyi de etkileyeceğinden veri gizleme işlemi belirli bir boyuta ulaştığında MP3 dosyası üzerinde büyük bozulmalar görülecektir. MP3 formatı üzerinde yapılacak bit düzeyinde gizlemelerde veri bitleri dönüşüm işleminde iç döngüler içerisinde değiştirilerek yapılabilir.

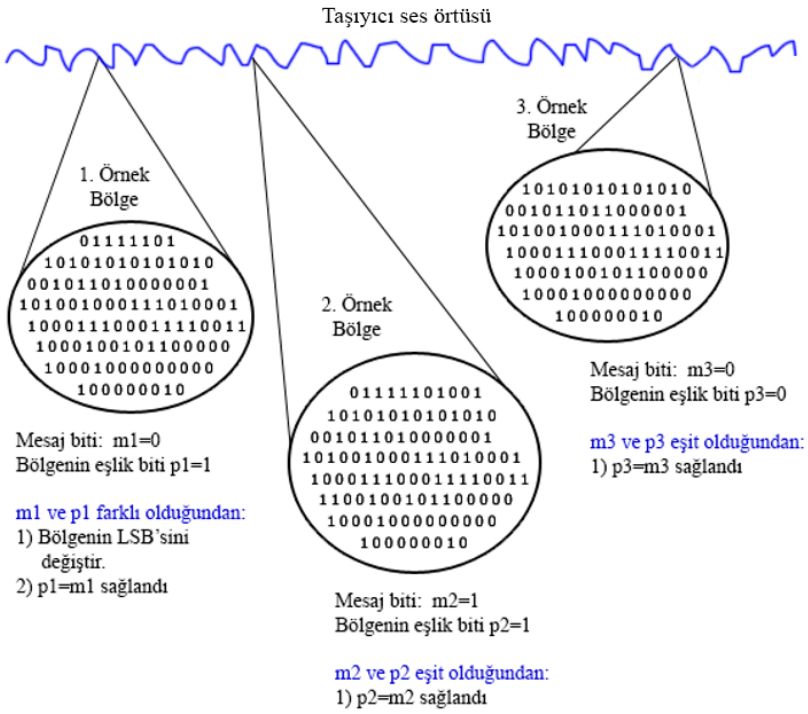
LSB kodlaması sağlamlık konusunda oldukça zayıf olmasına karşın çok yüksek düzeylerde veri saklama kapasitesi sunmaktadır. 44.1 kHz ile örneklenmiş bir ses örtüsü için fark edilmezlik çerçevesi içerisinde 44.1 kbit/sn ile 176.4 kbit/sn arasında oldukça yüksek bir gizleme kapasitesine ulaşılmaktadır.

3.1.2. Eşlik Kodlaması

Eşlik, diğer adıyla parite, herhangi bir ikilik veri dizisindeki 1 bitlerinin sayısıdır. Eğer dizideki 1 bitlerinin sayısı tek ise dizinin eşliği tek, diğer durumda eşliği çifttir denir. Karşı tarafa iletilen bit dizilerinin hata kontrollerinde ve dolayısıyla hata düzeltme kodlamalarında kullanılır.

Eşlik kodlaması ile veri gizleme işlemi esasen önemsiz bit kodlamasının veri gizlenecek bölgenin eşliğinin tek veya çift oluşuna göre yapılan gizlemedir. Seçilen bölgenin eşliği LSB kodlaması ile gizlenecek bite göre düzenlenir.

Gizleme işleminde genellikle gizlenecek 1 biti ile eşliğin tekliği 0 biti ile eşliğin çiftliği eşleştirilir. Örneğin gizlenecek veri biti 1 ise ve gizleme için seçilen bölgenin eşlik biti çift ise en önemsiz bit ters çevrilerek eşleme yapılır, 1 biti için eşlik biti tek ise hiçbir değişiklik yapılmaz.



Şekil 3.3. Eşlik kodlama ile veri gizleme işlemi

Temsili olarak Şekil 3.3'te gösterilen bölgelerin eşlik bitleri seçilen bölgelerdeki 1 bitlerinin sayısının tek ise 1 çift ise 0 olarak alınmıştır. Eşlik bitlerine göre LSB değişimi işlemi temel algoritma başlığı altında detaylı incelenmektedir.

3.1.2.1. Temel algoritma

Bu bölümde WAV formatı içerisine eşlik kodlaması ile veri gizleme ve veri çekme işlemleri incelenmiştir.

Değişkenler ve gösterimler

I → Ses örtüsünün örnek sayısı
M → Her bir örneğin bit uzunluğu
N → Gizlemede kullanılan örnek sayısı
w → Gizlenecek bit dizisi
K → Gizlenecek bit dizisinin uzunluğu

Veri gizleme işlemi

i. Taşıyıcı ses dosyasının örnekleri içeren veri kısmı ikilik veri olarak açılıp, $\{s[i,j]: 1 \le i \le I, 1 \le j \le M\}$ matrisine atanır.

ii. Gizlenecek veri uzunluğuna bağlı olarak gizlemenin kaçar örnek üzerine yapılacağı belirlenir, *N* sayısına atanır. Daha sonra, *N* sayısına bağlı olarak s matrisi *K* parçaya ayrılarak $s_k[n,m]$ matris aileleri oluşturulur.

iii. $1 \le k \le K$ için her bir bölgenin eşliğini hesaplamak için 1 bitlerinin sayısı aşağıdaki gibi hesaplanır.

$$p_k = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} s_k[n,m]$$

iv. Gizleme işlemi için p_k toplamı kullanılarak,

a. p_k çift sayı ve $w_k = 1$ ya da p_k tek sayı ve $w_k = 0$ ise s_k bölgesi üzerinde en önemsiz bit ters çevrilir. LSB 0 ise 1 olarak, 1 ise 0 olarak değiştirilir.

b. p_k tek sayı ve $w_k = 1$ ya da p_k çift sayı ve $w_k = 0$ ise s_k bölgesi üzerinde hiçbir değişiklik yapılmaz.

v. Veri bitleri ile düzenlenmiş s_k ikilik veri matrisleri tekrar onluk tabana çevrilerek kaydedilir.

Veri çekme işlemi

i. Veri gizlenmiş ses örtüsünün örneklerini içeren veri kısmı ikilik veri olarak açılıp, $\{s[i,j]: 1 \le i \le I, 1 \le j \le M\}$ matrisine atanır.

ii. $s[i,j]$ gizlemede olduğu gibi tekrar *K* parçaya ayrılarak $s_k[n,m]$ matris aileleri oluşturulur.

iii. $s_k[n,m]$ bölgesinin eşliği çift ise $w_k = 0$, tek ise $w_k = 1$ olacak şekilde veri bitleri geri çekilir.

3.1.2.2. Sonuç

Eşlik kodlaması ile veri gizleme yöntemi geleneksel önemsiz bit kodlamasına göre daha etkili olarak nitelendirilmektedir (Anderson ve Petitcolas, 1998). MP3 steganografinin ilk örneklerinden olan ve birçok stegoanaliz çalışmasında referans kaynağı olarak gösterilmiş MP3Stego (Petitcolas, 2006) programında gizleme işlemi sıkıştırma esnasında en iç döngüde eşlik kodlaması kullanılarak yapılmaktadır. Geleneksel önemsiz bit kodlamasına kıyasla daha etkili olmasına karşın yine de başta sıkıştırma algoritmaları olmak üzere çeşitli saldırılar karşısında oldukça zayıftır.

3.1.3. Faz Kodlaması

Faz kodlaması ile ses örtüsü içerisine veri gizleme yöntemi ilk olarak 1996 yılında önerilmiştir (Bender vd., 1996). Yöntemin incelendiği çalışmalarda insanın işitme duyusunun ses üzerindeki faz değişimlere karşı daha az hassas olduğu iddia edilmektedir. Geleneksel yönteme göre ses örtüsü parçalara ayrılıp, ayrık Fourier dönüşümü kullanılarak her bir bölütün örneklerinin faz açıları hesaplanarak gizleme işlemi ilk bölütün faz açıları üzerinde yapılmaktadır.

3.1.3.1. Temel algoritma

Bu bölümde FFT ile elde edilen faz açıları üzerine veri gizleme ve veri çekme işlemleri incelenmiştir. Çalışmanın MATLAB kodları Ek 1.3'te mevcuttur.

Değişkenler ve gösterimler:

I → Ses örtü dosyasının veri kısmının uzunluğu
L → Her bir bölütün uzunluğu
N → Bölüt sayısı
w → Gizlenecek veri bitleri
m → Gizlenecek bit dizinin uzunluğu

Veri gizleme işlemi

i. Taşıyıcı ses dosyası açılıp veri kısmı, $\{s[i]: 1 \le i \le I\}$ dizisine atandıktan sonra, $N = I / L$ parçaya bölünüp $1 \le n \le N$ ve $1 \le k \le L$ olacak şekilde $\{s_n[k]\}$ dizileri oluşturulur.

ii. $1 \le n \le N$ ve $F\{\cdot\}$ ayrık Fourier dönüşümü olmak üzere her bölütün faz dizi ailesi ϕ_n ve büyüklük dizi ailesi A_n aşağıdaki gibi hesaplanır.

$$A_n = \left|F\{s_n\}\right| = \sqrt{\mathrm{Re}\left(F\{s_n\}\right)^2 + \mathrm{Im}\left(F\{s_n\}\right)^2}$$

$$\phi_n = \angle\left(F\{s_n\}\right) = \tan^{-1}\left(\frac{\mathrm{Im}\left(F\{s_n\}\right)}{\mathrm{Re}\left(F\{s_n\}\right)}\right)$$

iii. $2 \le n \le N$ için ardışık bölütlerin faz dizileri arasındaki farklar aşağıdaki gibi hesaplanır.

$$\Delta\phi_n = \phi_n - \phi_{n-1}$$

iv. $1 \le i \le m$ olmak üzere ses dosyası içerine gizlenecek olan $w[i]$ veri bitleri ile $\varphi_w[i]$ faz dizisi aşağıdaki gibi oluşturulur.

$$\varphi_w[i] = \begin{cases} \dfrac{\pi}{2}, & w[i] = 0 \text{ ise} \\ -\dfrac{\pi}{2}, & w[i] = 1 \text{ ise} \end{cases}$$

v. $1 \le i \le m$ ve $m < \dfrac{L}{2}$ için ayrık Fourier dönüşümün eşlenik simetri özelliği korunarak φ_1 aşağıdaki gibi değiştirilerek φ_1' oluşturulur.

$$\varphi_1'[L/2 - m + i] = \varphi_w[i]$$

$$\varphi_1'[L/2 + 1 + i] = -\varphi_w[m + 1 - i]$$

vi. Yeni faz dizileri, bölütler arasındaki faz farkları, $\Delta\varphi_n$, kullanılarak aşağıdaki gibi tekrar oluşturulur.

$$\begin{bmatrix} \varphi_2' \\ \cdots \\ \varphi_n' \\ \cdots \\ \varphi_N' \end{bmatrix} = \begin{bmatrix} \varphi_1' + \Delta\varphi_2 \\ \cdots \\ \varphi_{n-1}' + \Delta\varphi_n \\ \cdots \\ \varphi_{N-1}' + \Delta\varphi_N \end{bmatrix}$$

vii. φ'_n faz dizileri ve A_n büyüklük dizileri ile $j=\sqrt{-1}$ için Euler özdeşliği kullanılarak elde edilen $z_n = A_n e^{j\varphi'_n}$ karmaşık sayı dizilerine ters Fourier dönüşümü uygulanarak bölütler zaman uzayında tekrar oluşturulur.

$$s'_n = F^{-1}\left\{A_n \exp\left(j\varphi'_n\right)\right\}$$

viii. s'_n bölütleri birleştirilerek içerisine veri gizlenmiş s' ses sinyali elde edilir.

Burada dikkat edilmesi gereken en önemli nokta bölüt uzunluğunun çift sayı olacak ve $2m < L$ şartını sağlayacak şeklinde seçilmesidir. Örnek olarak bölüt uzunluğu 2048 ya da 4096 olarak seçilebilir.

Veri çekme işlemi

i. İçerisine veri gizlenmiş ses dosyası açılarak veri kısmı $\left\{x[i]: 1 \le i \le I\right\}$ dizisine atanır.

ii. $x[i]$ dizisinin $1 \le k \le L$ olacak şekilde ilk bölütü, $x_1[k]$, alınır.

iii. $F\{\cdot\}$ ayrık Fourier dönüşümü olmak üzere veri gizlenmiş ilk bölütün faz dizisi ϕ_1 aşağıdaki gibi tekrar hesaplanır.

$$\phi_1 = \angle\left(F\{x_1\}\right) = \tan^{-1}\left(\frac{\mathrm{Im}\left(F\{x_1\}\right)}{\mathrm{Re}\left(F\{x_1\}\right)}\right)$$

iv. $1 \le i \le m$ olmak üzere eşlenik simetri özelliği korunarak gizlenmiş olan veri bitleri aşağıdaki şekilde geri çekilir.

$$w[i] = \begin{cases} 0, & \varphi_1[L/2 - m + i] > 0 \quad \text{ise} \\ 1, & \varphi_1[L/2 - m + i] < 0 \quad \text{ise} \end{cases}$$

3.1.3.2. Diğer çalışmalar

Faz açısı kullanılarak veri gizleme üzerine yapılan diğer çalışmaların odak noktası geleneksel yöntemin sağlamlığını arttırmak üzerinedir. Faz açısı üzerine veri gizleme yöntemleri Faz Kodlaması (Bender, 1996; Ruiz, 2000) ve Faz Düzenleme (Gang, 2001; Kuo, 2002) yöntemleri olmak üzere iki grupta incelenmektedir.

3.1.3.3. Sonuç

Geleneksel faz kodlaması yöntemi kaynak-bağımsız (blind) olarak incelendiğinde dayanıklılık açısından oldukça zayıftır. Gizleme işlemi yalnızca tek bölüt üzerinden yapıldığı için veri gizleme kapasitesi de oldukça düşüktür. Bölüt uzunlukları daha fazla veri gizleyecek şekilde daha uzun olarak seçilebilir. Ancak büyük boyutlu veri gizleme işlemlerinde ses üzerinde bozulmalar yaşanacaktır.

Algoritma kaynak-bağımsız olarak kullanıldığında ses formatı üzerinde yapılan değişiklere karşı oldukça hassastır. MP3 gibi ses sıkıştırma standartları ile yapılan format dönüşümlerde çok büyük veri kayıpları yaşanmaktadır.

3.1.4. Yankı Veri Gizleme

Yankı veri gizleme yöntemi ilk olarak 1996 yılında önerilmiştir (Gruhl vd., 1996). Yöntem genel anlamda, gizlenecek veri bitlerini farklı gecikme miktarları ile temsil ederek ses sinyali içerisine yayma mantığına dayanmaktadır. Buna göre öncelikle ses içerisine gizlenecek 0 bitini temsilen d_0 ve 1 bitini temsilen d_1 adet örnek kadar iki farklı gecikme miktarı belirlenir. Belirlenen bu gecikme miktarları ile birlikte iki yeni yankı sinyali oluşturulur. Ardından ses sinyali veri bitlerinin sayısı kadar eş parçaya bölünerek veri bitlerine karşılık her bir parçaya yankılar eklenir.

Herhangi bir *s* ses sinyaline *d* örnek miktarı kadar gecikmeye sahip bir yankı ekleme işlemi farklı şekillerde ifade edilebilir. Bu gösterimler aynı işlemi temsil etmekle birlikte, algoritma açıklanırken ağırlıklı olarak **b.** ve **c.** gösterimleri kullanılmıştır.

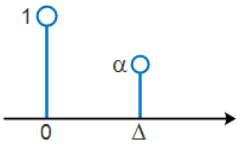
a. $x[n] = s[n] + \alpha \cdot s[n-d]$

b. $x[n] = s[n] * (\delta[n] + \alpha \cdot \delta[n-d])$

c. $x[n] = s[n] + s[n] * (\alpha \cdot \delta[n-d])$

Burada α sayısı yankının genliği, * işlemi doğrusal evrişim, $\delta[n]$ ise birim dürtüyü temsilen kronecker-delta fonksiyonudur.

Tekli yankı veri gizlemede kullanılan yankı çekirdeği görsel olarak Şekil 3.4'te ve denklem olarak Denklem 3.1'de verilmektedir.



Şekil 3.4. Tekli yankı gizlemede kullanılan yankı çekirdeği

$$h[n] = \delta[n] + \alpha \cdot \delta[n - \Delta] \tag{3.1}$$

3.1.4.1. Temel algoritma

Bu bölümde temel olarak birer yankı ile veri gizleme ve veri çekme işlemleri incelenmiştir. Çalışmanın MATLAB kodları Ek 1.4'te mevcuttur.

Değişkenler ve gösterimler:

I → Ses örtü dosyasının veri kısmının uzunluğu
N → Gizlenmek istenen bit dizisinin uzunluğu
L → Her bir bölütün uzunluğu
α → Yankının genliği $(0 < \alpha < 1)$
d_0 → 0 biti için gecikme miktarı (örnek sayısı)
d_1 → 1 biti için gecikme miktarı (örnek sayısı)

Veri gizleme işlemi

i. Taşıyıcı ses dosyası açılıp veri kısmı, $\{s[i]: 0 \le i \le I-1\}$ dizisine atanır.

ii. $\delta[n]$ kronecker-delta fonksiyonu olmak üzere gecikme miktarlarına göre k_0 ve k_1 yankı çekirdekleri oluşturulur:

$$k_0[n] = \alpha \cdot \delta[n - d_0]$$

$$k_1[n] = \alpha \cdot \delta[n - d_1]$$

Bu işlem ile d_0 adet sıfır içeren $k_0 = [0,0,\ldots,0,\alpha]$ dizisi ile d_1 adet sıfır içeren $k_1 = [0,0,\ldots,0,\alpha]$ dizisi elde edilecektir. Yankı ile veri gizleme yöntemleri yankı çekirdekleri ile ifade edildiği için çekirdek kavramının anlaşılması oldukça önemlidir.

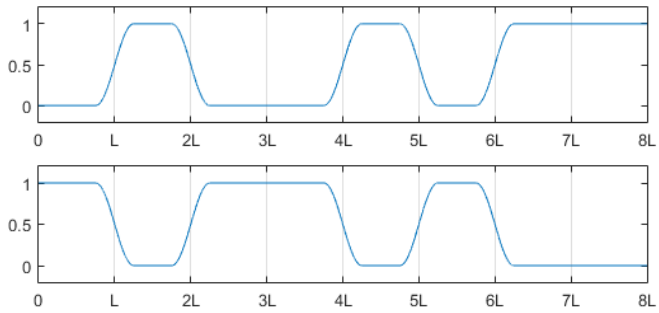
iii. Taşıyıcı ses sinyali yankı çekirdekleri ile aşağıdaki gibi evrişim işlemine sokularak d_0 ve d_1 örnek sayıları kadar gecikmeye sahip $\mathbf{e}_0$ ve $\mathbf{e}_1$ yankı sinyalleri oluşturulur.

$$\mathbf{e}_i = s * k_i \ , \ i \in \{0,1\}$$

Bu işlem sonucunda s sinyali α sabiti ile çarpıldıktan sonra başına d_0 adet 0 eklenerek $\mathbf{e}_0$ sinyali, d_1 adet 0 eklenerek ise $\mathbf{e}_1$ sinyali elde edilmiş olur.

$$\mathbf{e}_i = \left[0,0,...,0,\ \alpha \cdot s[1],\ \alpha \cdot s[2],\ ...,\ \alpha \cdot s[I-1]\right]$$

iv. Oluşturulan yankı sinyalleri ile taşıyıcı ses dosyasını birleştirme aşamasında kullanılmak üzere Şekil 3.5'teki örnekteki gibi $\mathbf{u}_0$ ve $\mathbf{u}_1$ karıştırıcı sinyalleri oluşturulur.



Şekil 3.5. 01001011 bit dizisine karşılık oluşturulan karıştırıcı sinyaller

v. Son olarak oluşturulan $\mathbf{e}_0$ ve $\mathbf{e}_1$ yankı sinyalleri ile $\mathbf{u}_0$ ve $\mathbf{u}_1$ karıştırıcı sinyallerinin noktasal çarpımları taşıyıcı ses dosyasına eklenerek s' sinyali elde edilir.

$$s' = s + \mathbf{e}_0 \cdot \mathbf{u}_0 + \mathbf{e}_1 \cdot \mathbf{u}_1$$

Veri çekme işlemi

i. Taşıyıcı ses dosyası açılıp veri kısmı alınarak, $\{s[i]: 0 \le i < I\}$ dizisine atandıktan sonra, her biri L uzunluğunda olacak kadar N adet bölüte ayrılarak $\{s_n[i]: 0 \le n < N, 0 \le i < L\}$ diziler ailesi oluşturulur. Buradaki N sayısı gizlenen bit sayısına eşittir.

ii. $\{s_n\}$ dizilerinin Denklem 3.2'deki şekilde gerçel kepstrumu alınır.

$$c_n[i] = F^{-1}\left\{\ln\left|F\{s_n[i]\}\right|\right\} \tag{3.2}$$

iii. d_0 ve d_1 gizleme aşamasında kullanılan gecikme miktarları olmak üzere, $0 \le n < N$ için bir önceki aşamada elde edilen $c_n[i]$ gerçel kepstrumunun d_0 ve d_1 değerlerinde yaptığı sıralamalar aşağıdaki gibi karşılaştırılır

$$\text{data}_n = \begin{cases} 1, c_n[d_1] \ge c_n[d_2] \text{ ise} \\ 0, c_n[d_1] < c_n[d_2] \text{ ise} \end{cases}$$

Gizlenmiş veriyi geri çekmek için kepstrum analizi kullanılmaktadır. s örtü sinyalinin üzerine d gecikme miktarıyla $h[n] = \delta[n] + \alpha \cdot \delta[n-d]$ çekirdeği kullanılarak elde edilmiş yankılı x sinyali aşağıdaki gibidir.

$$x[n] = s[n] * h[n]$$

$F\{.\}$ ayrık Fourier dönüşümü, $F^{-1}\{.\}$ ayrık ters Fourier dönüşümü ve $\ln(.)$ karmaşık sayılarda tanımlanmış doğal logaritma olmak üzere $x[n]$'nin karmaşık kepstrumu olmak üzere aşağıdaki şekilde ifade edilebilir.

$$\begin{aligned} \hat{x}[n] &= F^{-1}\left\{\log\left(F\{s[n] * h[n]\}\right)\right\} \\ &= F^{-1}\left\{\log\left(F\{s[n]\}\right)\right\} + F^{-1}\left\{\log\left(F\{h[n]\}\right)\right\} \\ &= F^{-1}\left\{\log\left(S\left[e^{jw}\right]\right)\right\} + F^{-1}\left\{\log\left(H\left[e^{jw}\right]\right)\right\} \\ &= \hat{s}[n] + \hat{h}[n] \end{aligned}$$

$\delta[n]$ fonksiyonunun tanımını kullanılarak $h[n]$'in Fourier dönüşümü hesaplanır.

$$\begin{aligned} F\{h[n]\} &= F\{\delta[n] + \alpha \cdot \delta[n-d]\} \\ &= 1 + \alpha \cdot e^{-jwd} \end{aligned}$$

$|x| < 1$ için $\log(1+x)$ fonksiyonun Taylor açılımı Denklem 3.3'teki verilmektedir.

$$\log(1+x) = \sum_{n=0}^{\infty} (-1)^{n+1} \frac{x^n}{n} \tag{3.3}$$

$\left|\alpha \cdot e^{-jwd}\right| < 1$ olduğundan yankı çekirdeğinin karmaşık kepstrumundaki logaritmik ifade Taylor serisine açılır.

$$\begin{aligned}\hat{h}[n] &= F^{-1}\left\{\log\left(1+\alpha e^{-jwd}\right)\right\} \\ &= F^{-1}\left\{\alpha e^{-jwd} - \frac{\alpha^2}{2}e^{-2jwd} + \frac{\alpha^3}{3}e^{-3jwd} - \ldots\right\} \\ &= \alpha\delta[n-d] - \frac{\alpha^2}{2}\delta[n-2d] + \frac{\alpha^3}{3}\delta[n-3d] - \ldots\end{aligned}$$

Dolayısıyla buradan $\hat{x}[n] = \hat{s}[n] + \hat{h}[n]$ olduğundan $\hat{h}[n]$'ın değeri yerine yazılır.

$$\hat{x}[n] = \hat{s}[n] + \alpha\delta[n-d] - \frac{\alpha^2}{2}\delta[n-2d] + \frac{\alpha^3}{3}\delta[n-3d] - \ldots \quad (3.4)$$

$\delta[n]$ fonksiyonunun tanımını kullanarak Denklem 3.4'ü yorumlarsak, $\hat{x}[n]$ içerisinde her d örnekte bir sıçrama olduğu görülmektedir. $\alpha < 1$ olduğundan bu sıçramalar üstel olarak azalacağından dolayı, sıçrama en net $n = d$ noktasında görülecektir. Dolayısıyla veri çekme işlemi yankılı sinyallerin kepstrum dönüşümlerinde d_0 ve d_1 noktalarında yaptığı sıçramaların kıyaslanması ile yapılır.

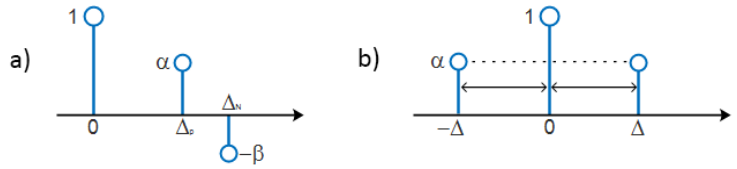
3.1.4.2. Diğer çalışmalar

Verilerin ses örtüsü içerisine yankılar ile temsil edilerek gizlenme fikri (Gruhl vd., 1996) ortaya atıldıktan sonra, yöntem üzerine performans iyileştirme odaklı birçok çalışma yapılmıştır.

Sağlamlığı arttırmak için bitleri temsil eden yankılar aynı bölüte pozitif ve negatif dürtülerle (Oh vd., 2001) Denklem 3.5'deki yankı çekirdeği ile Şekil 3.6 a'daki gibi gizlenmiştir. Diğer bir yaklaşımda ise aynı yankı bölüt içerisine ileri ve geri dürtüler ile simetrik olarak (Kim ve Choi, 2003) Denklem 3.6'daki yankı çekirdeği ile Şekil 3.6 b'deki gibi gizlenmiştir.

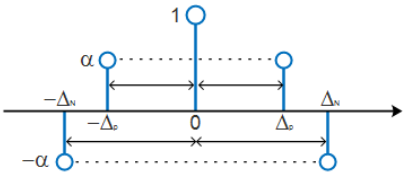
$$h[n] = \delta[n] + \frac{\alpha}{2}\cdot\left(\delta[n-\Delta_P] - \delta[n-\Delta_N]\right) \quad (3.5)$$

$$h[n] = \delta[n] + \frac{\alpha}{2}\cdot\left(\delta[n-\Delta] + \delta[n+\Delta]\right) \quad (3.6)$$



Şekil 3.6. a) Negatif-Pozitif yankı çekirdeği b) İleri-Geri yankı çekirdeği

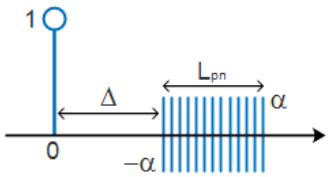
Kepstrum analizi bu iki yaklaşımın Denklem 3.7 ve Şekil 3.7'deki gibi tek çekirdekte birleştirilmesiyle yapılan gizleme işleminin nispeten daha dayanıklı olduğunu göstermektedir.



Şekil 3.7. Negatif-Pozitif ve İleri-Geri simetrik yankı çekirdeği

$$h[n]=\delta[n]+\frac{\alpha}{4}\cdot\left(\delta[n-\Delta_P]+\delta[n+\Delta_P]-\delta[n-\Delta_N]-\delta[n+\Delta_N]\right) \qquad (3.7)$$

Yankı veri gizlemenin güvenliği arttırmak için Denklem 3.8'deki yankı çekirdeği ile Şekil 3.8'de betimlenen zamana yayımlı yankı gizleme yöntemi önerilmiştir (Ko vd., 2002a, 2005). Bu yönteme göre yankı çekirdeği bir anahtar ile üretilmiş sözde-rastgele bir dizi ile oluşturulur ve yankılar bu çekirdek ile birlikte zamana yayılır. Yankı genliği sözde-rastgele dizinin uzunluğuna bağlı olarak oldukça küçük bir değer olarak seçilmektedir.



Şekil 3.8. Zaman yayılımlı yankı çekirdeği

$$h[n] = \delta[n] + \alpha \cdot \mathrm{PN}[n - \Delta] \tag{3.8}$$

Zaman yayımlı yankı gizlemede veri çekme işlemi diğer yöntemlerden farklı olarak gizlemede kullanılan sözde-rastgele dizi ve bölütün gerçel kepstrumunun çapraz ilintisi alınarak bulunmaktadır.

Gizlenen yankı sayısı oldukça fazla olduğundan, gizlemede kullanılan sözde-rastgele dizi olmaksızın gizlenen verinin varlığını tespit etmek oldukça güçtür. Bu sebeple bu yöntem diğer yöntemlere kıyasla daha güvenlidir. Ayrıca yankı sayısının fazla olması işitsel olarak da daha başarılı sonuçlar vermektedir.

Veri çekmedeki zorluk göz önünde bulundurularak, başarı oranını arttırmak için kepstrum analizi ile birlikte çoklu çerçeve ortalaması ve log-ölçeklendirmesi gibi yöntemler de önerilmektedir (Ko vd., 2004a, b).

3.1.4.3. Sonuç

Geleneksel yankı veri gizleme yönteminde veri çekme aşamasının kolaylığı gizlenmiş veriye erişilebilirlik açısından yarar sağlamasına karşın, bu kolaylık güvenlik açısından aynı ölçüde tehlike arz etmektedir. Gizleme aşamasında verinin örtü içerisine şifrelenerek gömülmesi veya bölütlere dağınık şekilde dağıtılması güvenlik açısından bir çözüm olabilir. Ancak örtü içerisine gizlenmiş verinin varlığı aynı yöntemi bilen herhangi bir üçüncü kişi tarafından yine de kolaylıkla fark edilebilir olacaktır.

Ses dalgaları ortamda bulanan yüzeylere çarparak, yüzeyin cinsine ve kulağa olan uzaklığına bağlı olarak değişen enerji seviyeleri ve gecikmeler ile çoklu olarak algılanır. Bu sebeple, ses içerisinde bölütlere birer yankı eklenerek yapılan gizleme doğada karşılaşılan yankı hissine kıyasla yapay kalmaktadır. Nitekim çoklu yankı gizlemeye dayalı birçok yöntem bu soruna çözüm üretmek amacıyla önerilmiştir.

Yapılan deneyler sonucu örtü ses dosyasının kendi içindeki mevcut yankıların, sessiz kısımlara yapılan gizlemedeki hataların, yankı genliğinin ve bölüt uzunluğunun uygun seçiminin yalancı pozitiflik oranlarında oldukça etkili olduğu görülmüştür. Genel olarak bölüt uzunluklarının özellikle 200 milisaniyenin altına düşürüldüğü deneylerde veri kayıpları yaşanmaya başlandığı gözlemlenmiştir. Yankı çekirdeğinin uygun seçimi de hata oranını düşürmede oldukça etkilidir. Ayrıntılı karşılaştırmalar 4. bölüm altında incelenmektedir.

3.1.5. Tayf Yayılımı Yöntemi

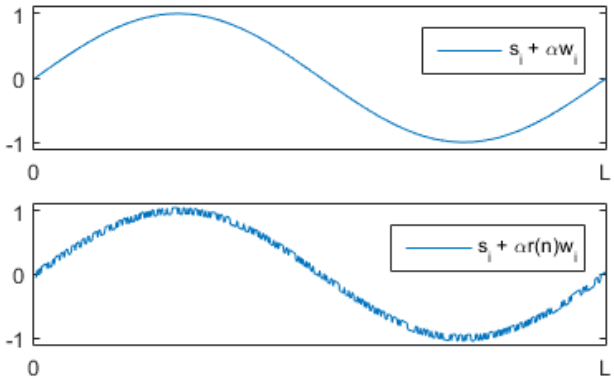
Tayf yayılımı kablosuz haberleşmede kullanılan en temel yöntemlerden biridir (Pickholtz vd., 1987). Tayf yayılımı ile veri gizleme kopya koruma amacıyla resim dosyalarının damgalanması için önerilmiştir (Cox vd., 1996). Daha sonraki süreçte yöntem ses dosyaları dahil diğer veri tipleri üzerine de uygulanmıştır. Tayf yayılımı yöntemine göre gizlenecek veri gürültü şeklinde taşıyıcı dosya üzerine yayılmaktadır. Bunun için taşıyıcı, gizlenecek veri kadar eş parçaya bölünerek her bir parçasına birer bit veri gizlenmektedir. 1996'da yayınlanan ilk çalışmada 3 farklı gizleme yöntemi önerilmiştir.

$$
\begin{aligned}
s_i' = s_i + \alpha w_i \quad &\longrightarrow \quad \text{Toplamsal} \\
s_i' = s_i \cdot (1 + \alpha w_i) \quad &\longrightarrow \quad \text{Çarpımsal} \\
s_i' = s_i \cdot \left(e^{\alpha w_i}\right) \quad &\longrightarrow \quad \text{Üstel}
\end{aligned}
$$

Burada α, $0 < \alpha \ll 1$ dayanıklılık ve fark edilmezliği kontrol eden gizleme parametresidir. N sayısı gizlenecek verinin uzunluğu olmak üzere, $0 \le i \le N-1$ için w_i gizlenecek verilerin i 'incisi, s_i taşıyıcı dosyanın i 'inci bölütü, s_i' ise veri gizlenmiş dosyanın i 'inci bölütüdür. Bu aşamadan sonra bölütler tekrar birleştirilerek veri gizlenmiş ses dosyası elde edilmektedir.

Birçok kaynakta gizleme aşamasında tayf yayılımın haberleşmede kullanıldığı şekli ile, w_i verileri kullanıcı tarafından girilen bir anahtar ile üretilen ve $\{-1,+1\}$ elemanlarından oluşan sözde rastgele bir dizi ile çarpılmaktadır. Gizlenen verinin geri çekilmesinde ilinti işleminde bu sözde rastgele dizi de gerekeceğinden bu aşama ek bir güvenlik sağlamaktadır. Ancak sözde rastgele bir dizi Şekil 3.9'da gösterildiği şekilde taşıyıcı ses üzerinde ekstra bir gürültü oluşturacağından, α parametresini bu gürültüyü azaltacak kadar küçülttüğümüzde gizlenen verilerin geri çekilmesinde kayıplar yaşanabilmektedir.

Sözde rastgele dizi üretmek için, M-sequence, Mersenne Twister, Xoroshiro gibi birçok algoritma kullanılabilir. Bu algoritmalar genel olarak girilen anahtarı kullanarak doğrusal kaydırma, mantıksal simetrik fark (XOR), döndürme gibi işlemler ile rastgele üretilmiş gibi gözüken birbirleri ile bağlantılı yeni dizi elemanları üretirler. Girilen anahtar ile aynı işlemler tekrarlanarak her seferinde aynı dizi üretileceğinden birçok veri gizleme yönteminde kullanılmışlardır.



Şekil 3.9. Ses bölütü ve sözde rastgele dizi $r[n]$ ile çarpılmış ses bölütü

3.1.5.1. Temel algoritma

Bu bölümde temel olarak sıralı dizi tayf yayılımı (DSSS) ile veri gizleme ve veri çekme işlemleri incelenmiştir. Çalışmanın MATLAB kodları Ek 1.5'te mevcuttur.

Değişkenler ve gösterimler:

I → Ses örtü dosyasının veri kısmının uzunluğu
N → Gizlenmek istenen bit dizisinin uzunluğu
L → Her bir bölütün uzunluğu
α → Gizleme parametresi $(0 < \alpha \ll 1)$
w → Gizlenecek beri bitleri

Veri gizleme işlemi

i. Taşıyıcı ses dosyası açılıp veri kısmı, $\{s[i]: 0 \le i \le I-1\}$ dizisine atanır.

ii. $s[i]$ dizisi N parçaya bölünüp $0 \le n \le N-1$ ve $0 \le k \le L-1$ olacak şekilde $\{s_n[k]\}$ dizileri oluşturulur.

iii. İsteğe bağlı olarak herhangi bir sözde-rastgele dizi üretici kullanılarak L uzunluğunda sözde-rastgele $r[k]$ dizisi oluşturulur.

iv. $0 \le n \le N-1$ için w_n gizlenecek n 'inci veri biti olmak üzere, ikilik tabandaki $\{0,1\}$ bitleri zıt kutuplu $\{-1,1\}$ şekline dönüştürülür.

v. Gizleme işlemi için, oluşturulan her parçaya gizleme aşağıdaki gibi yapılır.

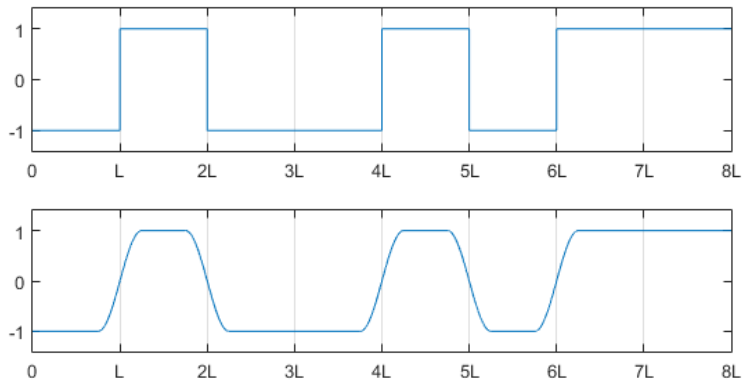
$$s'_n[k] = s_n[k] + \alpha r[k] w_n$$

Burada $0 \le k \le L-1$ olmak üzere her bir bölüte bir bit gizlenmektedir.

vi. Son aşamada $s'_n[k]$ dizileri tekrar birleştirilerek $s'[i]$ dizisi elde edilir.

L değeri veri uzunluğu ile uyumlu olacak şekilde $L = \left\lfloor \frac{I}{N} \right\rfloor$ olarak seçilebilir. Ancak gizlenecek veri miktarı arttığında L değeri ters orantılı şekilde azalacağından α sabitine de bağlı olmakla birlikte veri kayıpları yaşanabilir.

Gizleme işlemi veri bitlerine karşılık $\{-1,1\}$ değerleri alan ve her bir bölütü L uzunluğunda olacak biçimde Şekil 3.10'da gösterildiği şekilde veri geçiş sinyali oluşturularak da yapılabilir. Veri geçiş sinyali kullanıldığında geçiş sinyal üzerinde bit geçiş noktalarındaki kırılmalar pencereleme fonksiyonları ile rahatlıkla yumuşatılabileceğinden geçiş noktalarındaki ses bozulmalarının da önüne geçilebilecektir.



Şekil 3.10. 8 bit için oluşturulmuş örnek veri geçiş sinyali ve pencerelenmiş hali

Şekil 3.10'da örnek olarak sözde-rastgele dizi $r[k]$ kullanılmadan oluşturulan veri geçiş sinyali T değişkeni ile temsil edilmek üzere $s'[i]$ sinyali, $s'[i] = s[i] + \alpha T$ eşitliği ile elde edilir.

Veri çekme işlemi

i. İçerisine veri gizlenmiş ses dosyası açılarak veri kısmı $\{x[i]: 1 \le i \le I\}$ dizisine atanır.

ii. $x[i]$ dizisi N parçaya bölünüp $0 \le n \le N-1$ ve $0 \le k \le L-1$ olacak şekilde $\{x_n[k]\}$ dizileri oluşturulur.

iii. Eğer gizleme aşamasında sözde-rastgele dizi kullanıldıysa aynı anahtar ile aynı $r[k]$ dizisi tekrar oluşturulur.

iv. $0 \le n \le N-1$ olmak üzere için her bölütün sözde-rastgele dizi ile çapraz ilintisinin sıfır gecikme noktasındaki değeri Denklem 3.9 ile hesaplanır.

$$c_n = \frac{1}{L}(x_n \star r)_0 = \frac{1}{L}\sum_{k=0}^{L-1} x_n[k] \cdot r[k] \qquad (3.9)$$

v. τ sayısı önceden tanımlanmış eşik değeri olmak üzere veri bitleri aşağıdaki gibi geri çekilir.

$$w_n = \begin{cases} 0, & c_n \le \tau \text{ ise} \\ 1, & c_n > \tau \text{ ise} \end{cases}$$

Veri çekme işlemini analiz etmek için, $x_n[k] = s_n[k] + \alpha r(k) w_n$ eşitliğinin değeri Denklem 3.9'da yerine yazılırsa,

$$\begin{aligned} c_n &= \frac{1}{L} \cdot \sum_{k=0}^{L-1} \left(s_n[k] + \alpha r[k] w_n \right) \cdot r[k] \\ &= \frac{1}{L} \cdot \sum_{k=0}^{L-1} s_n[k] \cdot r[k] + \frac{1}{L} \cdot \sum_{k=0}^{L-1} \alpha r^2[k] w_n \end{aligned}$$

Yukarıdaki ikinci eşitlikte ilk terim genellikle çok küçük değerlere sahiptir. Eğer $x_n[k]$ ve $r[k]$ birbirinden bağımsız ise bu ilk terim tamamen kaybolacaktır. Ancak elbette ki bu durum her zaman sağlanmak zorunda değildir. Bu aşamayı iyileştirmek için örtü ses üzerinde yüksek geçiren (high-pass) süzgeç, beyazlatıcı (whitening) süzgeç, Savitzky-Golay yumuşatma süzgeci, LPC veya kepstrum süzgeci gibi öncül işlemlerin uygulandığı çalışmalar mevcuttur (Kim, 2003; Cvejic vd., 2008:25-28).

Karar verme aşamasındaki τ sayısı önceden belirlenmiş eşik değeridir. Genellikle $\tau = 0$ olarak seçilmektedir. Bölüt uzunlukları yeterince büyük olduğunda bu seçim dengeli sonuçlar vermektedir, ancak bölüt uzunluğu küçültüldükçe yalancı pozitiflik oranları da artmaktadır. Bir başka deyişle, veri gizli bazı bölütler ters davranış göstererek hatalı bit değerleri gösterebilmektedir.

3.1.5.2. Diğer çalışmalar

Geleneksel tayf yayılımı yönteminin başarı oranlarını arttırmak amacıyla birçok çalışma yapılmış, yöntem birçok farklı dönüşüm uzayı altında denenmiştir.

Geleneksel yönteminin hata olasılığını azaltarak dayanıklılığını arttırmak için İyileştirilmiş Tayf Yayılımı (ISS) yöntemi önerilmiştir (Malvar vd., 2003). İyileştirilmiş tayf yayılımı yönteminde taşıyıcı sese veri gizlenirken veri bitleri ile birlikte taşıyıcı sesin kendisi de kullanılmaktadır. Bu yöntem geleneksel yöntemde kullanılan değişkenler ile kısaca Denklem 3.10 ve 3.11'deki gibi ifade edilebilir.

$\mathbf{u} = u[k] = \alpha r[k]$ ve $\bar{s}_n = \dfrac{\langle s_n, \mathbf{u} \rangle}{\|\mathbf{u}\|} = \dfrac{1}{\alpha L} \sum_{k=0}^{L-1} s_n[k] \cdot r[k]$ olmak üzere,

$$s'_n[k] = s_n[k] + \mu(\bar{s}_n, w_n) \cdot \mathbf{u} \tag{3.10}$$

$\mu(\bar{s}_n, w_n)$ için doğrusal bir yaklaşım olarak $\beta w_n - \gamma \bar{s}_n$ alınabilir.

$$s'_n = s_n + (\beta w_n - \gamma \bar{s}_n) \cdot \mathbf{u} \tag{3.11}$$

Burada $\beta = 1$ ve $\gamma = 0$ alınırsa geleneksel tayf yayılımı yöntemi elde edilecektir. Varyans hesaplamaları ile birlikte γ parametresi için optimal bir seçim yöntemin önerildiği makalede ayrıntılı olarak incelenmektedir.

Diğer bir çalışmada mevcut yöntemin fark edilmezliğini arttırmak için her bölüt alt bölütlere parçalanarak gizleme işlemi enerjisi düşük parçalar atlanarak yapılmıştır (Kirovski vd, 2001, 2002, 2003).

Tayf yayılımı ile veri gizleme için sıralı dizi tayf yayılımın (DSSS) yanı sıra, frekans atlatmalı tayf yayılımı (FSSS) yöntemi de önerilmiştir (Cvejic vd., 2004).

3.1.5.3. Sonuç

Mevcut kaynaklar incelediğinde diğer yöntemler ile karşılaştırıldığında tayf yayılımı üzerine daha fazla çalışma yapıldığı görülmüştür. Damgalama başlığı altında bahsedilen saldırı yöntemlerine karşı tayf yayılımı yöntemi daha dayanıklı olduğu için, üzerinde çokça durulmuştur. Ancak yöntem dayanıklı olmasına karşın, veri saklama kapasitesi oldukça düşüktür. Dayanıklılık ve veri saklama kapasitesi aynı anda arttırıldığında gizlenen verinin fark edilmezliği de aynı oranda ortadan kalkmaktadır.

Klasik tayf yayılımı yöntemleri ile 44.1 kHz ile örneklenmiş, farklı karakterdeki ses dosyaları üzerinde yapılan çalışmalarda dayanıklılık açısından en ideal denge bölüt uzunluğunu yaklaşık 16,000 örnek seçildiğinde ölçülmüştür. Bu seçilen uzunluğa göre veri saklama kapasitesi yaklaşık 2.7 bit/sn civarında iken bit hata oranı %1-3.5 arasındadır.

3.1.6. Diğer Veri Gizleme Yöntemleri

Ses üzerine veri gizleme işlemleri enerji değerleri, frekans seviyeleri, faz açıları, ritim aralıkları, histogram ve spectrogram gibi birçok bileşen üzerinde birçok farklı dönüşüm uzayı ve ayrışım kullanılarak direkt düzenleme, kaydırma, yer değiştirme gibi sayısız şekilde yapılabilmektedir. Gizlenen verilerin istatistiksel olarak geri çekilebildiği her türlü algoritma gizleme algoritması olarak nitelendirilebilir. Ses örtüsünün farklı frekans seviyelerini hedef alan farklı gizleme algoritmaları birbirlerine karşı sağlamlık, kapasite ve fark edilmezlik üçgeninde avantajları ve dezavantajları ile birlikte değerlendirilirler.

Ses örtüsüne veri gizlemede sık kullanılan diğer yöntemlerden bazıları Nicemleme Dizin Düzenlemesi (QIM), Yineleme Yöntemi (Replica Method), Yama Yöntemi (Patchwork) olarak sayılabilir.

QIM (Chen ve Wornell, 2001), tek başına bir yöntemden ziyade yöntemler ailesi olarak nitelendirilmektedir. İsmini analog sinyallerin dijitalleştirilmesi sırasında yapılan nicemleme işleminden almaktadır. Buna göre, ses örtüsünün önemsiz bitleri gizlenecek veri bitlerine göre belirli nicemleme aralıkları ile kaydırılabilir, enerji değerleri yine belirli nicemleme aralıkları ile tam değer fonksiyonu kullanılarak küçük oranlarda arttırılıp azaltılabilir veya dayanıklılığı arttırmak için bu işlem çoklu örnekler üzerinde uygulanabilir.

Yineleme yöntemi (Petrovic, 2001), gizleme işleminin ses örtüsünün kendi verileri ile birlikte çeşitli aritmetik işlemler kullanılarak yapıldığı yöntemlere verilen genel addır. Yankı veri gizleme yöntemi de yineleme yöntemi olarak kabul edilmektedir. Nitekim sesin kendisi kendi üzerine belirli gecikme oranları ile toplama işlemi altında yankı olarak eklenmektedir.

Yama yöntemi (Yeo ve Kim, 2003), ses örtüsünün karşılaştırılabilir ikili parçaları üzerinde bu karşılaştırmalar kullanılarak yapılan gizleme yöntemleridir. En basit anlamda iki grup verinin sayısal değerleri gizlenecek bitin değerine göre arttırılarak ya da azaltılarak yapılabilir. Bir parçadan çıkarma ve bir parçaya ekleme yapıldığı için yamama yöntemi olarak isimlendirilmiştir.

3.2. Dönüşüm Uzaylarında Veri Gizleme

Dönüşüm uzayları özellikle sağlamlık ve görünmezliğin saklama kapasitesinden daha önemli olduğu damgalama uygulamalarında oldukça sık tercih edilmektedir. Zaman uzayı altında yapılan gizlemede veri zaman ekseninde ses örnekleri üzerine genellikle direk gizlenirken, dönüşüm uzayları altında gizleme işlemi gizleyicinin belirleyeceği frekans seviyeleri arasına gizlenebilmektedir.

Farklı amaçlar doğrultusunda ayrık Fourier dönüşümü (DFT) (Megías vd., 2010), kosinüs ve sinüs dönüşümleri (DCT ve DST) (Fan ve Wang, 2009; Lei vd., 2011), dalgacık dönüşümleri (DWT ve LWT), kepstrum dönüşümü (Lee ve Ho, 2000; Chowdhury ve Khan, 2013), Hilbert-Huang dönüşümü (HHT altında EMD) (Huang, 1998; Wang vd, 2010), Walsh-Hadamard dönüşümü (WHT) gibi birçok dönüşüm altında gizleme işlemleri yapılabilmektedir.

Gizleme uygulamalarında gizlenecek veri bitleri dönüşümler altında hedeflenen bölümler üzerine tayf yayılımı, nicemleme dizin düzenlemesi ve önemsiz bit kodlaması gibi temel yöntemler kullanılarak gizlenmektedir.

Dalgacık dönüşümü çoklu zaman-frekans çözünürlüğüne sahip olduğu için veri gizleme uygulamalarında oldukça tercih edilmekle birlikte yapılan çalışmalar genellikle bu alanda yoğunlaşmaktadır. Veri sağlamlığının oldukça önemli olduğu damgalama uygulamalarında yüksek seviyeli yaklaşık kısımları kullanılırken, sağlamlık ve görünmezlik arası dengeyi korumak için yüksek seviyeli ayrıntı kısımları kullanılmaktadır.

3.3. Ayrışımlar Üzerinde Veri Gizleme

Veri gizleme işlemlerinin sağlamlığını arttırmak için dönüşüm uzayları ile birlikte cebirsel ayrışımlar da oldukça sık kullanılmaktadır. Veri gizleme uygulamalarında, QR ayrışımı ve tekil değer ayrışımı (Özer vd, 2005) en çok kullanılan ayrışımlardır. Literatür incelendiğinde ayrışımlar üzerine yapılan veri gizleme çalışmalarının genellikle tekil değer ayrışımı üzerinde yoğunlaştığı görülmektedir. Tekil değer ayrışımı çoğunlukla referanslı ya da yarı referanslı çalışmalarda kullanılmıştır. Mevcut çalışmalara göre, veri gizle işleminde herhangi bir dönüşüm altında ses örtüsünün verileri kare matris olarak yazıldıktan sonra, bu matris tekil değer ayrışımına sokularak gizleme işlemi tekil değerler üzerinde yapılmaktadır. Gizleme işlemi için genellikle en büyük tekil değer kullanılmaktadır. Ayrışımlar çoğunlukla Ayrık Dalgacık Dönüşümü ve Ayrık Kosinüs Dönüşümü altındaki katsayıları ile birlikte kullanılmıştır. Bu çalışmalara örnek olarak; DWT-SVD (Bhat vd., 2010), DCT-SVD (Lei vd, 2011), DWT-DCT-SVD (Lei vd., 2013) sayılabilir.

4. BULGULAR VE TARTIŞMA

4.1. Tayf Yayılımı Üzerinde İyileştirme Çalışmaları

Bölüm 2.1.5'te temel algoritması verilen tayf yayılımı üzerinde yapılan çalışmalarda α sabiti ile birlikte taşıyıcı sesin kendi karakteristik özelliklerinin büyük etkisi olduğunu gözlemlenmiştir. Veri sağlamlığını arttırmak için α sabiti daha büyük seçildiğinde örtü ses üzerindeki bozulmalar düşük enerjiye sahip sessiz kısımlarda daha net fark edilmektedir. Dolayısıyla α parametresinin ses örtüsünün karakteristik özellikleri kullanılarak her bölüte göre değişken olarak seçildiği çalışmalarda daha başarılı sonuçlar alınmıştır. Bunun için α parametresi veri gizlenecek her bir bölütün karesel ortalamaları ile güç seviyelerinin desibel cinsinden değerleri kullanılarak adım adım Denklem 4.1, 4.2 ve 4.3'teki gibi hesaplanmıştır.

$$p_n = 10\log_{10}\left(\frac{1}{L}\sum_{k=0}^{L-1}\left(s_n[k]\right)^2\right) \tag{4.1}$$

$$q_n = \frac{\max\{p_n + \lambda_1,\ \lambda_2\}}{20} \tag{4.2}$$

$$\alpha_n = (10)^{q_n} \tag{4.3}$$

λ_1 sabiti enerji seviyesinin çok yüksek olduğu bölütlerde α_n değişkeninin çok büyük olmaması için, λ_2 sabiti ise enerji seviyesinin çok düşük olduğu bölütlerde α_n parametresin çok küçük olmaması için kontrol sabitleri olarak eklenmiştir. Uygulamalarımızda $\lambda_1 \in [-20, -10]$ ve $\lambda_2 = -60$ olarak seçilmiştir. λ_1 sabiti küçüldükçe α_n parametresi de küçüleceğinden gizlenmiş verinin fark edilmezlik oranı artarken sağlamlığı azalacaktır.

Yapılan deneyler sonucunda göze çarpan diğer bir önemli nokta ise veri çekme aşamasında kullanılan eşik değerinin nasıl seçileceği olmuştur. Bölüt uzunluğu küçüldükçe $\tau = 0$ seçimi oldukça başarısız sonuçlar vermektedir. İlinti işleminin tanımı gereği, her bir bölütün aritmetik ortalaması eşik değerinin seçiminde açıkça büyük etki göstermektedir. Dolayısıyla veri gizlenme işleminden önce taşıyıcı ses üzerinde bölütlerin ortalamaları üzerinden bir takım öncül işlemler denenmiştir.

Yapılan deneyler aşağıda adım adım irdelenmektedir.

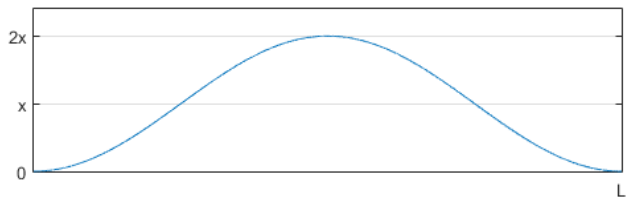
İlk adımda, yerel ortalamaların etkisini daha açık bir şekilde görebilmek için, her bir bölütün yerel ortalamasını sıfır yapmak üzere ortalamalar hesaplanıp bölütlerden çıkartılmıştır.

$$s_n'[k] = s_n[k] - \frac{1}{L}\sum_{k=0}^{L-1} s_n[k]$$

Bu işlem beklendiği gibi geleneksel yönteme göre daha başarılı sonuçlar vermiştir. Buna karşın bölüt uzunlukları azaldıkça birleşim noktalarında süreksizlik noktaları arttığı için dinleme testlerinde ses üzerinde büyük bozulmalar gözlemlenmiştir.

İkinci adımda, her bölütün yerel ortalaması kendisinden çıkartılmadan önce Şekil 4.1'deki gibi Hann pencereleme ile yumuşatılmıştır. Yeterli uzunluktaki bir Hann pencerenin ortalaması yaklaşık olarak 0.5 olduğu için ortalamayı korumak amacıyla Hann penceresi 2 ile genişletilmiştir.

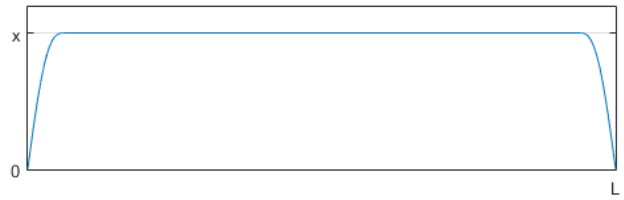
$$s_n'[k] = s_n[k] - 2\cdot\left(\frac{1}{L}\sum_{k=0}^{L-1} s_n[k]\right)\cdot \text{hann}(L)$$



Şekil 4.1. Hann pencereleme ile yumuşatılmış bölüt ortalaması

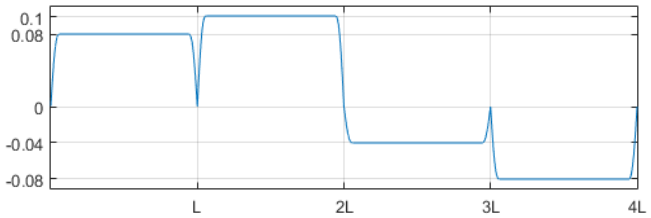
Bu işlem bölütler arası geçişleri yumuşatmasına karşın, 2 ile genişletme işleminden dolayı orta noktaların komşuluğunda sıçramalara sebep olmuştur. Dolayısıyla bu adımda da ses üzerinde bozulmalar gözlemlenmiştir.

Üçüncü adımda, orta noktalardaki sıçramaları yok etmek için bölüt ortalamalarını bölüt uzunluğuna eşit pencere ile doğrudan çarpmak yerine, ortalamalar bölüt uzunluğundan daha küçük Hann pencere ile evrişim işlemine sokulup Şekil 4.2'de elde edilen pencereler bölütlerden çıkarılmıştır.



Şekil 4.2. Hann pencere ve evrişim işlemi ile yumuşatılmış ortalama sinyali

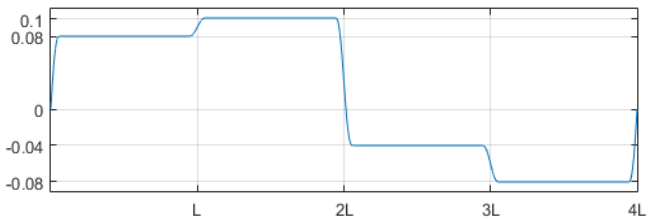
Bu işlem hem bölütlerin birleşim noktalarında oluşan süreksizliğe hem de bölütlerin orta noktasında oluşan sıçramalarda daha iyi bir çözüm sağlamıştır. İlinti işleminde başarılı sonuçlar elde edilmiştir. Ancak her bir bölütün ortalamasını kendinden bu şekilde çıkarıldığında, aynı işarete sahip ardışık bölütlerde de geçiş noktalarında Şekil 4.3'teki gibi gereksiz yere sıfır değerine inilecektir.



Şekil 4.3. Yumuşatılmış ortalama sinyalinin kırılma noktaları

Dördüncü adımda, bir önceki adımda elde edilen sonuçlar üzerinden evrişim işlemini bölüt bölüt yapmak yerine bölütlerin ortalama değerleri ile oluşturulan bir geçiş sinyali boyunca Hann pencereleme uygulanmıştır. Bu işlem ile aynı işarete sahip bölütler arası geçişlerdeki sıçramalar en aza indirgenmiştir.

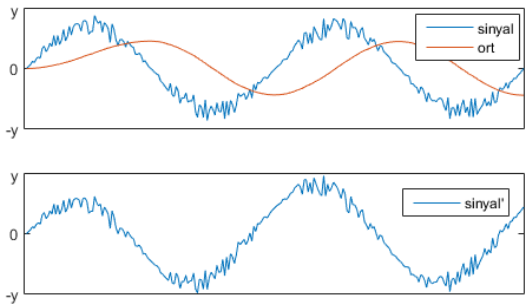
Üçüncü ve dördüncü adım, örnek olarak ortalamaları 0.08, 0.1, -0.04 ve -0.08 olan dört ardışık bölüt için Şekil 4.3 ve Şekil 4.4'te gösterilmektedir.



Şekil 4.4. Yumuşatılmış ve düzgünleştirilmiş ortalama sinyali

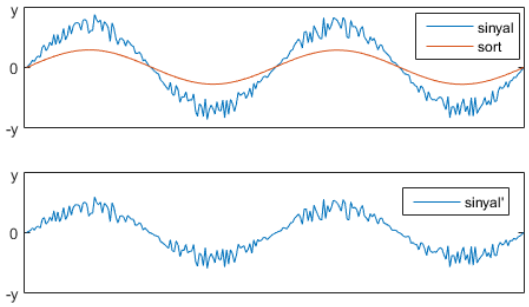
Beşinci adımda, bir önceki adımda elde edilen sonuçları iyileştirmek adına ses örtüsüne öncül işlem olarak bölüt uzunluğunca *L*-noktalı hareketli ortalamalar (moving average) filtresi uygulanmıştır.

Hareketli ortalama süzgecinin kullanılış amacı bölüt ortalamaları ile oluşturulan geçiş sinyali ile birlikte örtü sinyal üzerinde oluşabilecek bozulmaları öncül işlem ile birlikte en aza indirgemektir. Bu doğrultuda hareketli ortalamalar örtü sinyalden çıkartılarak sinyal doğrusal bir şekle sokulmaktadır. Klasik hareketli ortalamalar ve simetrik hareketli ortalamalar üzerinden yapılan çalışmalarda simetrik hareketli ortalamaların örtü sinyali doğrusallaştırmada daha başarılı olduğu görülmüştür.



Şekil 4.5. Klasik hareketli ortalamalar ile filtrelenmiş sinyal

Şekil 4.5'te gösterildiği şekilde klasik hareketli ortalamalar uygulandığında sinyalin bitiş noktası kaymakta ve doğrusal eksende genlik değerleri yükselmektedir. Şekil 4.6'da gösterilen simetrik hareketli ortalamalar süzgeci ise başlangıç ve bitiş noktalarını sabitleyip sinyali seçilen nokta sayısıyla bağlantılı olarak doğrusal bir forma sokmaktadır.



Şekil 4.6. Simetrik Hareketli ortalamalar ile filtrelenmiş sinyal

4.1.1. Önerilen Tayf Yayılımı Algoritması

Değişkenler ve gösterimler:

I → Ses örtünün veri kısmının uzunluğu
N → Gizlenmek istenen bit dizisinin uzunluğu
L → Her bir bölütün uzunluğu
M → Ortalama geçiş sinyali
T → Bit geçiş sinyali
α → Güç parametre sinyali
w → Gizlenecek beri bitleri

Veri gizleme işlemi

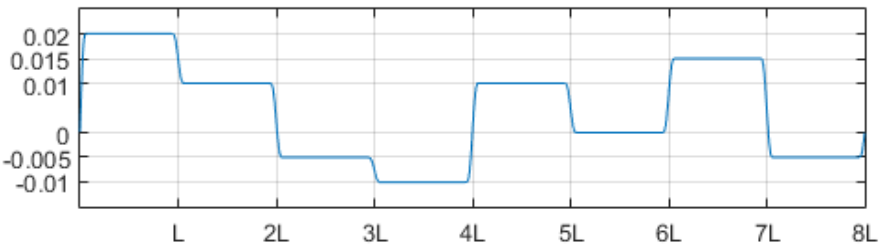
i. Taşıyıcı ses dosyası açılıp veri kısmı, $\{s[i]: 0 \le i \le I-1\}$ dizisine atanır.

ii. $s[i]$ dizisine bölüt uzunluğu kadar noktalı simetrik hareketli ortalamalar süzgeci uygulanır.

iii. $s[i]$ dizisi N parçaya bölünüp $0 \le n \le N-1$ ve $0 \le k \le L-1$ olacak şekilde $\{s_n[k]\}$ diziler ailesi oluşturulur.

iv. Her bir bölütün ortalaması $m_n = \frac{1}{L}\sum_{k=0}^{L-1} s_n[k]$ ile hesaplanarak m_n dizisinin elemanlarından oluşan ve Hann pencere ile örnek olarak Şekil 4.7'deki gibi yumuşatılmış M geçiş sinyali oluşturularak $0 \le i \le NL-1$ için ses örtüsünden çıkartılır.

$$s'[i] = s[i] - M$$

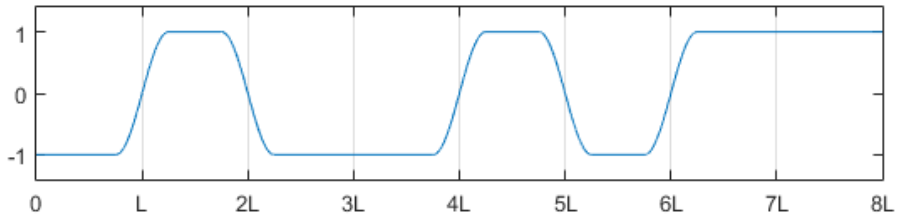


Şekil 4.7. Temsili yumuşatılmış ortalama geçiş sinyali

v. $0 \le n \le N-1$ için Denklem 4.3 kullanılarak güç parametresi her bölüt için değişken olacak şekilde α_n parametreleri hesaplanır. Ardından α_n parametreleri her bir bölütü L uzunluğunda olacak şekilde NL uzunluğuna genişletilerek α sinyali elde edilir.

vi. İsteğe bağlı olarak bir anahtar ile veri bit dizinin uzunluğunca $\{0,1\}$ elemanlarından oluşan sözde-rastgele bir dizi üretilerek w_n veri bit dizisi ile XOR işlemine sokularak şifrelenmiş w_n' dizisi elde edilir.

vii. Şifrelenmiş w_n' bit dizisi, ikilik tabandan $\{0,1\}$, zıt kutuplara $\{-1,+1\}$ dönüştürülerek her bölütü L uzunluğunda olacak şekilde Hann pencereleme ile yumuşatılmış Şekil 4.8'deki gibi T veri geçiş sinyali oluşturulur.



Şekil 4.8. Temsili yumuşatılmış veri geçiş sinyali

viii. $0 \le i \le NL-1$ için T geçiş sinyali ve α sinyallerinin noktasal çarpımları $s'[i]$ sinyaline eklenerek giz sinyali elde edilir.

$$x[i] = s'[i] + \alpha \cdot T$$

ix. Taşıyıcı ses sinyalinin NL'den arda kalan elemanları varsa onlar da eklenerek $s'[i]$ dizisinin son hali elde edilir.

Veri çekme işlemi

Veri gizleme işlemi temel yöntemin veri çekme aşamasını güçlendirmek üzerine kurgulandığı için çekme işlemi Bölüm 3.1.5.1'de verilmiş olan temel algoritmadaki gibi yapılır. Gizlemenin vi. adımında şifreleme kullanılmış ise aynı sözde-rastgele dizi aynı anahtar ile tekrar üretilerek veri çekme aşamasında ilinti işleminde çekilen şifrelenmiş veri ile XOR işlemine sokulur.

4.2. Yöntemlerin Karşılaştırılması

İncelenen yöntemlerin karşılaştırılması farklı tarzlardan seçilmiş otuzar saniyelik başlangıç ve bitiş kısımları pencerelenmiş, 44.1 kHz ile örneklenmiş tek kanallı ses veri kümesi üzerinde yapılmıştır. Veri gizleme algoritmalarının birçoğu gizlendikleri ses örtüsünün karakteristik özelliğine bağlantılı performans sergilediği için örtü ses için veri tabanı olabildiğince geniş bir yelpazede seçilmiştir. Karşılaştırmalar Çizelge 4.1'de belirtilen tarzlardan oluşan 76 adet farklı ses örneği üzerinde her tarz için ayrı ayrı incelendikten sonra başarı oranlarının ortalamaları alınarak yapılmıştır.

Çizelge 4.1. Testlerde kullanılan ses örneklerinin tarzları ve kısaltmaları

Açıklama	**Kod**
Klasik Müzik	C
Caz Müzik	J
Pop Müzik	P
Rock Müzik	R
Konuşma Örnekleri	K
Türk Sanat Müziği	TH
Türk Halk Müziği	TS

Yöntemlerin veri saklama kapasiteleri karşılaştırılırken saniye başına gizlenen veri sayısı baz alınmıştır. Tüm yapılan karşılaştırmalar 44.1 kHz ile örneklenmiş test sinyalleri üzerinden yapılmıştır. Dolayısıyla birim olarak bit/sn kullanılmaktadır.

Yöntemlerin sağlamlık derecelerini ölçmek için gizlenen $w(n)$ bit dizisi ile çıkartılan $w'(n)$ bit dizisi karşılaştırılarak yapılmıştır. Ölçüm için Denklem 4.4 ve 4.5'te tanımlanan Bit Hata Oranı (BER) ve Normalleştirilmiş İlinti (NC) kullanılmıştır.

$$BER(w,w') = \frac{100}{N} \cdot \sum_{n=0}^{N-1} XOR\left(w(n), w'(n)\right) \tag{4.4}$$

$$NC(w,w') = \frac{\sum_{n=0}^{N-1} w(n) \cdot w'(n)}{\sqrt{\sum_{n=0}^{N-1} \left(w(n)\right)^2 \cdot \sum_{n=0}^{N-1} \left(w'(n)\right)^2}} \tag{4.5}$$

BER hesaplaması hatalı bitlerin yüzdelik dilime düşen sayısını verdiği için 0 ile 100 arası değer alacaktır. NC hesaplaması gizlenen ve geri çekilen bit dizilerinin ilinti işlemi ile benzerliklerini ölçmektedir. Değer aralığı 0 ile 1 arasındadır. NC değeri 1 sayısına ne kadar yakınsa iki dizinin benzerlik oranı o kadar yüksektir.

Gizlenen verinin fark edilmezliği için ölçümler nesnel olarak Denklem 4.6'da tanımlanan Gürültü Oranı (SNR) hesaplaması ve öznel dinleme testleri ile giz sinyal Çizelge 4.2'deki şekilde 1-5 arası puanlanarak yapılmıştır.

$$SNR(s,s') = 10\log_{10}\left(\frac{\sum_{k=0}^{K-1}\left(s[k]\right)^2}{\sum_{k=0}^{K-1}\left(s'[k]-s[k]\right)^2}\right) \tag{4.6}$$

Çizelge 4.2. Öznel dinleme testleri için puanlamalar

Açıklama	**Puan**
Fark edilmez	5
Rahatsız etmeyen	4
Az oranda rahatsız edici	3
Rahatsız edici	2
Çok rahatsız edici	1

Veri gizleme uygulamalarında SNR değeri 20 dB üzerinde iken teorik olarak gizlenmiş verinin dinleyici tarafından fark edilemez olduğu kabul edilir. Ancak öznel dinleme testleri de gizlenen verinin fark edilmezliğini ölçmede oldukça önemli bir yere sahiptir.

Örnek olarak, $s[k]$ herhangi bir ayrık sinyal olmak üzere $s'[k] = -s[k]$ olarak tanımlansın. Bu durumda $SNR(s,s')$ değeri hesaplandığında logaritma ifadesinin içerisi 0.25 olacağından,

$$SNR(s,s') = 10\log_{10}\left(\frac{1}{4}\right) = -6.0206$$

Ancak bu sonuca rağmen $s[k]$ ve $-s[k]$ ses sinyalleri üzerinde yapılan öznel dinleme testlerinde iki sesin tamamen ayırt edilemez oldukları gözlemlenmiştir.

4.2.1. Yankı Veri Gizleme Yöntemleri

Yankı veri gizleme yöntemlerinin sağlamlık karşılaştırılmaları 5 farklı çekirdek üzerinde MP3 sıkıştırmasına karşı dayanıklılıkları BER ve NC hesaplamaları ile incelenerek gerçekleştirilmiştir. Fark edilmezlik karşılaştırmasında her çekirdek için SNR hesaplanmış ve öznel dinleme testleri (ÖDT) gerçekleştirilmiştir.

Çizelge 4.3. Yankı veri gizleme yöntemleri ve kısaltmaları

Yankı çekirdeği	**Kısaltma**
Tekli yankı	TY
Negatif-Pozitif yankı	NPY
İleri-Geri yankı	İGY
İleri-Geri ve Negatif-Pozitif yankı	İGNPY
Zaman Yayımlı yankı	ZYY

Yöntemlerin gizlemede kullanılan parametreler ile bağlantılı olarak performans değişiklikleri ve karşılaştırmaları her bir adımda parametreler değiştirilerek ayrı ayrı incelenmiştir. Bu doğrultuda Çizelge 4.3'te belirtilen yankı veri gizleme yöntemleri için kullanılan parametreler Çizelge 4.4'te verilmektedir.

Çizelge 4.4. Yankı veri gizleme yöntemlerinde kullanılan parametreler

Parametre	**Açıklama**
L	Bölüt uzunluğu
d_0	Bit 0 için gecikme miktarı
d_1	Bit 1 için gecikme miktarı
α_1	TY, NPY, İGY ve İGNPY için yankı genliği
α_2	ZYY için yankı genliği
L_p	ZYY için sözde-rastgele dizi uzunluğu

Zaman yayımlı yankı gizleme yönteminde L_p uzunluğu arttıkça yankı sayısı fazlalaştığından sağlamlık artarken fark edilmezlik azalacaktır. Zaman yayımlı yankı gizleme için tüm deneylerde $L_p = 512$ olarak alınmıştır. Her bir deney için sabit yankı genlikleri kullanılmıştır. Karşılaştırmalarda kolaylık açısından her bir deney için değişkenler $\{Y_1, Y_2, Y_3, Y_4,\}$ parametreleri içerisinde incelenmiştir.

$Y_1 : \{L = 1024,\ d_0 = 50,\ d_1 = 75,\ \alpha_1 = 0.4,\ \alpha_2 = 0.02\}$

Çizelge 4.5. Yankı veri gizleme yöntemlerinin Y_1 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

	MP3	**TY**		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	(kbit)	BER	NC	BER	NC	BER	NC	BER	NC	BER	NC
C	128	12.62	0.875	12.04	0.883	11.53	0.887	11.66	0.886	14.71	0.844
	96	12.59	0.876	12.07	0.882	11.75	0.884	11.65	0.886	14.67	0.845
	64	13.34	0.868	12.90	0.874	12.10	0.881	12.24	0.880	15.89	0.832
P	128	4.150	0.958	3.802	0.961	7.732	0.962	3.439	0.965	6.118	0.938
	96	4.123	0.958	3.885	0.961	3.962	0.960	3.655	0.963	6.243	0.937
	64	4.722	0.952	4.143	0.958	4.262	0.957	4.025	0.959	7.415	0.924
J	128	8.078	0.916	7.497	0.923	7.543	0.922	7.218	0.926	10.87	0.890
	96	8.264	0.914	7.660	0.921	7.613	0.921	7.296	0.925	10.95	0.889
	64	8.504	0.912	7.969	0.918	7.962	0.918	7.621	0.922	12.18	0.876
R	128	2.200	0.978	1.774	0.982	1.851	0.981	1.518	0.985	3.601	0.963
	96	2.130	0.978	1.851	0.981	1.836	0.981	1.526	0.985	3.872	0.961
	64	2.440	0.975	2.091	0.979	2.184	0.978	1.712	0.983	4.422	0.955
TH	128	4.531	0.954	4.972	0.950	3.903	0.960	4.407	0.956	8.914	0.901
	96	4.655	0.953	4.779	0.952	4.113	0.958	4.507	0.955	9.116	0.907
	64	5.266	0.947	5.344	0.946	4.616	0.953	5.026	0.950	9.913	0.899
TS	128	4.469	0.955	4.190	0.958	3.919	0.961	4.151	0.959	6.397	0.934
	96	4.732	0.953	4.399	0.956	4.066	0.959	4.205	0.958	6.761	0.931
	64	5.050	0.950	4.841	0.952	4.546	0.954	4.608	0.954	8.001	0.918
K	128	3.408	0.966	3.466	0.965	2.895	0.971	3.302	0.967	5.296	0.946
	96	3.253	0.967	3.602	0.964	3.060	0.969	3.273	0.967	5.209	0.947
	64	3.805	0.962	3.902	0.961	3.437	0.965	3.602	0.964	6.148	0.937
Ortalama		5.825	0.941	5.580	0.944	5.471	0.947	**5.269**	**0.947**	8.414	0.913

Çizelge 4.6. Yankı veri gizleme yöntemlerinin Y_1 parametreleri ile fark edilmezlik karşılaştırmaları

	TY		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
K	8.106	5	10.87	5	11.04	5	13.56	5	7.228	5
P	8.023	5	13.62	5	10.91	5	16.74	5	8.825	5
J	8.043	5	13.38	5	11.11	5	16.86	5	8.852	5
R	8.040	4	13.57	5	10.98	5	16.56	5	8.561	5
TH	8.063	5	12.30	5	11.20	5	15.36	5	8.645	5
TS	8.090	5	11.51	5	11.13	5	14.26	5	8.367	5
K	8.068	5	12.22	5	11.52	5	15.05	5	8.881	5
Ort.	8.062	4.86	12.50	5	11.13	5	15.48	5	8.480	5

$Y_2 = \{L = 4096,\ d_0 = 50,\ d_1 = 75,\ \alpha_1 = 0.4,\ \alpha_2 = 0.02\}$

Çizelge 4.7. Yankı veri gizleme yöntemlerinin Y_2 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

	MP3	**TY**		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	(kbit)	BER	NC	BER	NC	BER	NC	BER	NC	BER	NC
C	128	5.707	0.943	5.676	0.944	4.994	0.950	5.242	0.948	3.009	0.969
	96	5.707	0.943	5.428	0.946	5.025	0.950	5.273	0.950	3.071	0.948
	64	5.738	0.943	5.986	0.941	5.552	0.941	5.593	0.945	3.442	0.965
P	128	1.005	0.990	0.807	0.992	0.787	0.992	0.852	0.991	0.634	0.994
	96	0.940	0.990	0.830	0.992	0.852	0.991	0.830	0.992	0.656	0.993
	64	0.983	0.990	0.918	0.991	0.896	0.991	0.830	0.992	0.699	0.993
J	128	2.388	0.976	1.768	0.982	1.892	0.981	1.675	0.983	1.768	0.982
	96	2.264	0.977	1.675	0.983	2.047	0.979	1.520	0.985	1.768	0.982
	64	2.264	0.977	1.861	0.981	1.737	0.982	1.675	0.983	1.520	0.985
R	128	0.465	0.995	0.279	0.997	0.248	0.998	0.310	0.997	0.372	0.996
	96	0.310	0.997	0.310	0.997	0.279	0.997	0.372	0.996	0.310	0.997
	64	0.403	0.996	0.372	0.996	0.279	0.997	0.310	0.997	0.434	0.996
TH	128	1.179	0.988	1.117	0.989	1.024	0.990	0.838	0.991	0.682	0.993
	96	1.334	0.987	1.148	0.988	0.900	0.991	0.900	0.991	0.713	0.993
	64	1.210	0.988	1.148	0.989	0.931	0.991	0.962	0.990	0.807	0.992
TS	128	1.551	0.984	1.520	0.985	1.489	0.985	1.396	0.986	0.279	0.997
	96	1.830	0.982	1.706	0.983	1.643	0.983	1.582	0.984	0.248	0.998
	64	1.675	0.983	1.582	0.984	1.613	0.984	1.334	0.987	0.248	0.998
K	128	0.544	0.995	0.544	0.995	0.388	0.996	0.544	0.995	0.472	0.996
	96	0.621	0.994	0.582	0.994	0.272	0.997	0.388	0.996	0.466	0.995
	64	0.815	0.992	0.621	0.994	0.388	0.996	0.621	0.994	0.427	0.996
Ortalama		1.854	0.981	1.709	0.983	1.583	0.984	1.574	0.984	**1.048**	**0.988**

Çizelge 4.8. Yankı veri gizleme yöntemlerinin Y_2 parametreleri ile fark edilmezlik karşılaştırmaları

	TY		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
C	8.001	4	10.67	5	10.90	5	13.36	5	7.104	5
P	7.980	5	13.46	5	10.93	5	16.58	5	8.757	5
J	7.987	5	13.19	5	11.03	5	16.66	5	8.767	5
R	8.003	5	13.39	5	10.94	5	16.38	5	8.493	5
TH	7.988	5	12.09	5	11.16	5	15.16	5	8.555	5
TS	8.009	5	11.31	5	11.06	5	14.06	5	8.262	5
K	7.987	5	12.02	5	11.43	5	14.84	5	8.790	5
Ort.	7.994	4.86	12.30	5	11.06	5	15.29	5	8.390	5

$Y_3 = \{L = 4096,\ d_0 = 250,\ d_1 = 350,\ \alpha_1 = 0.4,\ \alpha_2 = 0.02\}$

Çizelge 4.9. Yankı veri gizleme yöntemlerinin Y_3 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

	MP3	**TY**		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	(kbit)	BER	NC	BER	NC	BER	NC	BER	NC	BER	NC
C	128	2.730	0.972	2.419	0.976	2.668	0.973	2.357	0.976	2.419	0.975
	96	2.978	0.970	2.357	0.976	2.543	0.974	2.202	0.978	2.326	0.976
	64	3.474	0.965	2.574	0.974	2.885	0.971	2.543	0.974	2.854	0.971
P	128	0.852	0.991	0.874	0.991	0.809	0.992	0.896	0.991	0.983	0.990
	96	0.830	0.992	0.830	0.992	0.787	0.992	0.787	0.992	0.962	0.990
	64	0.896	0.991	0.962	0.990	0.787	0.992	0.896	0.991	1.093	0.989
J	128	1.582	0.984	1.551	0.984	1.303	0.987	1.396	0.986	1.303	0.987
	96	1.737	0.982	1.551	0.984	1.179	0.988	1.489	0.985	1.241	0.987
	64	1.799	0.982	1.458	0.985	1.458	0.985	1.458	0.985	1.241	0.987
R	128	0.372	0.996	0.248	0.998	0.155	0.998	0.155	0.998	0.186	0.998
	96	0.341	0.997	0.217	0.998	0.155	0.998	0.186	0.998	0.186	0.998
	64	0.279	0.997	0.403	0.996	0.279	0.997	0.217	0.998	0.248	0.998
TH	128	0.962	0.990	0.775	0.992	0.838	0.992	0.744	0.992	0.651	0.993
	96	0.869	0.991	0.838	0.992	0.931	0.991	0.900	0.991	0.682	0.993
	64	1.148	0.988	0.962	0.990	0.867	0.991	0.620	0.994	0.589	0.994
TS	128	0.310	0.997	0.217	0.998	0.217	0.998	0.155	0.998	0.124	0.999
	96	0.310	0.997	0.248	0.998	0.310	0.997	0.217	0.998	0.124	0.999
	64	0.496	0.995	0.279	0.997	0.279	0.997	0.217	0.998	0.279	0.997
K	128	0.505	0.995	0.427	0.996	0.349	0.997	0.388	0.996	0.466	0.995
	96	0.427	0.996	0.466	0.995	0.349	0.997	0.388	0.996	0.621	0.994
	64	0.621	0.994	0.505	0.995	0.311	0.997	0.466	0.995	0.505	0.995
Ortalama		1.120	0.989	0.960	0.990	0.927	0.991	**0.889**	**0.991**	0.909	0.991

Çizelge 4.10. Yankı veri gizleme yöntemlerinin Y_3 parametreleri ile fark edilmezlik karşılaştırmaları

	TY		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
C	7.999	4	11.30	5	11.00	5	14.53	5	7.097	5
P	7.992	5	11.33	5	10.89	5	14.41	5	8.763	5
J	8.001	5	11.07	5	10.68	5	13.99	5	8.773	5
R	8.013	5	11.58	5	11.12	5	14.60	5	8.496	5
TH	7.997	5	10.83	5	11.00	5	13.82	5	8.560	5
TS	8.012	5	11.14	5	10.77	5	14.37	5	8.278	5
K	7.991	5	10.63	5	10.97	5	13.68	5	8.798	5
Ort.	8.001	4.86	11.13	5	10.92	5	14.20	5	8.395	5

$Y_4 = \{L = 8192,\ d_0 = 250,\ d_1 = 350,\ \alpha_1 = 0.4,\ \alpha_2 = 0.02\}$

Çizelge 4.11. Yankı veri gizleme yöntemlerinin Y_4 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

	MP3	**TY**		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	(kbit)	BER	NC	BER	NC	BER	NC	BER	NC	BER	NC
C	128	0.995	0.990	0.809	0.992	0.746	0.992	0.622	0.994	0.746	0.993
	96	1.119	0.989	1.057	0.989	0.871	0.991	0.871	0.991	0.871	0.991
	64	1.306	0.987	0.871	0.991	0.995	0.990	0.933	0.991	0.746	0.992
P	128	0.350	0.996	0.393	0.996	0.262	0.997	0.350	0.996	0.306	0.997
	96	0.393	0.996	0.393	0.996	0.350	0.996	0.350	0.996	0.262	0.997
	64	0.306	0.997	0.350	0.996	0.437	0.996	0.481	0.995	0.219	0.998
J	128	0.373	0.996	0.249	0.998	0.187	0.999	0.311	0.997	0.187	0.998
	96	0.435	0.996	0.311	0.997	0.249	0.998	0.311	0.997	0.187	0.998
	64	0.311	0.997	0.249	0.998	0.311	0.997	0.373	0.996	0.249	0.998
R	128	0.124	0.999	0	1	0	1	0	1	0.062	0.999
	96	0.062	0.999	0	1	0	1	0	1	0.062	0.999
	64	0.062	0.999	0	1	0	1	0	1	0	1
TH	128	0.435	0.996	0.311	0.997	0.373	0.996	0.249	0.998	0.062	0.999
	96	0.498	0.995	0.311	0.997	0.373	0.996	0.311	0.997	0.062	0.999
	64	0.435	0.996	0.373	0.996	0.498	0.995	0.435	0.996	0.062	0.999
TS	128	0.187	0.998	0.124	0.999	0.124	0.999	0.062	0.999	0.062	0.999
	96	0.124	0.999	0.062	0.999	0.124	0.999	0.062	0.999	0	1
	64	0.124	0.999	0.062	0.999	0.124	0.999	0.124	0.999	0.124	0.999
K	128	0.078	0.999	0.078	0.999	0.078	0.999	0.078	0.999	0.155	0.998
	96	0.078	0.999	0.078	0.999	0.155	0.998	0.078	0.999	0.155	0.998
	64	0.078	0.999	0.155	0.998	0.078	0.999	0.078	0.999	0.155	0.998
Ortalama		0.375	0.996	0.297	0.997	0.302	0.997	0.290	0.997	**0.225**	**0.998**

Çizelge 4.12. Yankı veri gizleme yöntemlerinin Y_4 parametreleri ile fark edilmezlik karşılaştırmaları

	TY		**NPY**		**İGY**		**İGNPY**		**ZYY**	
	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
C	7.999	4	11.28	5	11.02	5	14.51	5	7.096	5
P	7.977	5	11.30	5	10.90	5	14.38	5	8.746	5
J	7.995	5	11.05	5	10.74	5	13.97	5	8.766	5
R	8.039	5	11.58	5	11.12	5	14.60	5	8.514	5
TH	7.988	5	10.81	5	11.02	5	13.80	5	8.547	5
TS	8.019	5	11.13	5	10.81	5	14.35	5	8.262	5
K	7.978	5	10.60	5	10.10	5	13.65	5	8.778	5
Ort.	7.999	4.86	11.11	5	10.82	5	14.18	5	8.387	5

Yankı ile veri gizleme yöntemlerinin karşılaştırmalarında bölüt uzunlukları ve gecikme süreleri kademeli olarak değiştirilerek 4 aşamada incelenmiştir. Sağlamlık değerlerinde kullanılan mp3 sıkıştırması için LAME (Anonim, 2011) tabanlı harici MATLAB fonksiyonu (Ellis, 2009) kullanılmıştır.

Yankı ile temsil edilerek gizlenen verinin sağlıklı bir şekilde geri okunabilmesinde bölüt uzunluğunun oldukça önemli bir rol oynadığı görülmüştür. Gizlenen verinin sıkıştırma karşısında sağlamlığı seçilen bölüt uzunluğu ile doğru orantılı olarak yükselmektedir.

Sıkıştırma oranının veri sağlamlığına çok fazla etki etmediği gözlemlenmiştir. Hatta, bazı istisnai durumlarda sıkıştırma oranı yükseldiğinde ters davranış ile veri sağlamlığında iyileşmeler olduğu görülmüştür.

Yankı gecikme süreleri genel itibariyle bakıldığında veri sağlamlığı ile paralel davranış göstermektedir. Gecikme süreleri arttırıldığında veri çekmedeki hata oranlarında düşüşler gözlenmiştir. Ancak gecikme süreleri arttıkça öznel dinleme testlerinde verinin fark edilebilirlik oranı da paralel olarak artacağından gecikme süreleri 0.01 saniyeden daha büyük seçilmemelidir.

Özellikle piyano gibi ritim aralıkları sabit ses örneklerinde eklenen harici tekli yankılar öznel olarak daha keskin bir şekilde hissedilirken, çoklu yankılarda bu fark edilebilirlik oranı çok daha düşüktür. Ayrıca daha gürültülü ve homojen ses örnekleri için yankının fark edebilirlik oranının oldukça düşük olduğu görülmüştür.

Beş farklı yankı veri gizleme yöntemi üzerinde yapılan deneyler ile nesnel değerlendirmelerde SNR hesaplamasıyla fark edilebilirlik oranı en düşük olan yöntem İGNP yankı çekirdeği olmuştur. Sağlamlık konusunda da genel ortalamaya bakıldığında İGNP yankı çekirdeğinin başarı oranı daha yüksek ölçülmüştür.

ZYY çekirdeği için BER ve SNR dengesini diğer çekirdekler ile yakın tutabilmek için yankı genliği yöntemin önerildiği ve incelendiği çalışmalarda verilen değerlerden daha yüksek seçilmiştir. Bu koşul altında ZYY çekirdeği belirli şartlar altında İGYP çekirdeğinden daha yüksek sağlamlık performansı göstermiştir.

Genel bir bakış ile yankı veri gizleme yöntemlerinin dayanıklılığı ve fark edilmezliği kabul edilebilir seviyede olmasına karşın düşük kapasiteye sahiptirler.

4.2.2. Tayf Yayılımı Yöntemleri

Tayf yayılımı ile veri gizleme yöntemlerinin karşılaştırılmaları 3 farklı yöntem üzerinde 76 adet ses örneği ile gerçekleştirilen deneylerin ortalamaları alınarak yapılmıştır. Yöntemlerin sağlamlık karşılaştırmalarında MP3, FLAC ve OGG sıkıştırmalarına karşı dayanıklılıkları BER ve NC hesaplamaları ile incelenerek gerçekleştirilmiştir. Fark edilmezlik karşılaştırmasında her yöntem için 76 adet ses örneğinin SNR ortalamaları hesaplanmış ve öznel dinleme testleri (ÖDT) gerçekleştirilmiştir.

Çizelge 4.13. Tayf yayılımı yöntemleri ve kısaltmaları

Yöntem	**Kısaltma**
Geleneksel tayf yayılımı	GTY
İyileştirilmiş tayf yayılımı	İTY
Önerilen tayf yayılımı	ÖTY

Yöntemlerin gizlemede kullanılan parametreler ile bağlantılı olarak performans değişiklikleri ve karşılaştırmaları her bir adımda parametreler değiştirilerek ayrı ayrı incelenmiştir. Bu doğrultuda tayf yayılımı ile veri gizleme yöntemlerinde kullanılan parametreler Çizelge 4.14'te belirtilmektedir.

Çizelge 4.14. Tayf yayılımı yöntemlerinde kullanılan parametreler

Parametre	**Açıklama**
L	Bölüt uzunluğu
α	GTY ve İTY için güç parametresi
β	İTY için ikinci güç parametresi
γ	İTY için üçüncü güç parametresi

Önerilen tayf yayılımı yöntemi için güç parametresi Denklem 4.1, 4.2 ve 4.3 eşitliklerinde verildiği üzere her bölüt için farklı şekilde değişken olarak gizleme esnasında hesaplanmıştır. Denklem 4.2'de parametre değişkeninin büyüklüğünü kontrol etme amacıyla kullanılan λ_1 ve λ_2 değişkenleri $\lambda_1 = -15$ ve $\lambda_2 = -60$ olarak seçilmiştir. Sözde-rastgele dizi her bölütte her bir örnek üzerine gürültü olarak değil her bölüte tek sözde-rastgele dizi elemanı düşecek şekilde vızıltı olarak eklenmiştir. Karşılaştırmalarda kolaylık açısından her bir deney için değişkenler $\{T_1, T_2, T_3, T_4, T_5, T_6\}$ parametreleri içerisine atanarak incelenmiştir.

$T_1 = \{L = 256,\ \alpha = 0.005,\ \beta = 1,\ \gamma = 0.2\}$

Çizelge 4.15. Tayf yayılımı yöntemlerinin T_1 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

		GTY		**İTY**		**ÖTY**	
	Sıkıştırma	BER	NC	BER	NC	BER	NC
WAV	-	16.649	0.831	13.295	0.865	0	1
FLAC (bit/örnek)	16	16.591	0.831	13.305	0.865	0	1
	8	16.647	0.831	13.308	0.865	0.532	0.995
MP3 (kbit/sn)	128	18.287	0.815	15.246	0.845	0	1
	96	19.211	0.805	16.390	0.834	0	1
	64	23.609	0.761	21.677	0.780	0.005	1.000
OGG (%)	%80	18.017	0.817	14.960	0.848	0	1
	%60	18.695	0.810	15.859	0.839	0.008	1.000
	%40	20.117	0.796	17.572	0.823	0.028	0.999
Ortalama		18.647	0.811	15.735	0.840	**0.064**	**0.999**

Çizelge 4.16. Tayf yayılımı yöntemlerinin T_1 parametreleri ile fark edilmezlik karşılaştırmaları

GTY		**İTY**		**ÖTY**	
SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
21.73	2	20.36	2	9.79	**4**

$T_2 = \{L = 512,\ \alpha = 0.005,\ \beta = 1,\ \gamma = 0.2\}$

Çizelge 4.17. Tayf yayılımı yöntemlerinin T_2 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

		GTY		**İTY**		**ÖTY**	
	Sıkıştırma	BER	NC	BER	NC	BER	NC
WAV	-	12.697	0.871	9.051	0.908	0	1
FLAC (bit/örnek)	16	12.659	0.872	9.056	0.908	0	1
	8	12.704	0.871	9.066	0.908	0.427	0.996
MP3 (kbit/sn)	128	14.239	0.856	10.723	0.891	0	1
	96	15.277	0.845	11.987	0.879	0	1
	64	19.314	0.805	16.730	0.831	0	1
OGG (%)	%80	14.145	0.857	10.776	0.891	0	1
	%60	14.754	0.851	11.536	0.883	0	1
	%40	16.174	0.836	13.104	0.867	0.003	1.000
Ortalama		14.663	0.852	11.337	0.885	**0.047**	**1.000**

Çizelge 4.18. Tayf yayılımı yöntemlerinin T_2 parametreleri ile fark edilmezlik karşılaştırmaları

GTY		**İTY**		**ÖTY**	
SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
22.06	2	21.52	2	13.50	**4**

$T_3 = \{L = 512,\ \alpha = 0.05,\ \beta = 1,\ \gamma = 0.5\}$

Çizelge 4.19. Tayf yayılımı yöntemlerinin T_3 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

		GTY		**İTY**		**ÖTY**	
	Sıkıştırma	BER	NC	BER	NC	BER	NC
WAV	-	12.050	0.878	4.081	0.959	0	1
FLAC (bit/örnek)	16	12.011	0.878	4.082	0.959	0	1
	8	12.054	0.878	4.088	0.959	0.427	0.996
MP3 (kbit/sn)	128	13.575	0.862	5.917	0.940	0	1
	96	14.607	0.852	7.291	0.926	0	1
	64	18.688	0.812	13.175	0.867	0	1
OGG (%)	%80	13.471	0.864	5.632	0.943	0	1
	%60	14.110	0.857	6.737	0.931	0	1
	%40	15.467	0.843	8.780	0.911	0.003	1.000
Ortalama		14.004	0.858	6.643	0.933	**0.047**	**1.000**

Çizelge 4.20. Tayf yayılımı yöntemlerinin T_3 parametreleri ile fark edilmezlik karşılaştırmaları

GTY		**İTY**		**ÖTY**	
SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
11.73	2	10.46	1	**13.50**	**4**

$T_4 = \{L = 1024,\ \alpha = 0.05,\ \beta = 1,\ \gamma = 0.5\}$

Çizelge 4.21. Tayf yayılımı yöntemlerinin T_4 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

		GTY		**İTY**		**ÖTY**	
	Sıkıştırma	BER	NC	BER	NC	BER	NC
WAV	-	0	1	0	1	0	1
FLAC (bit/örnek)	16	0	1	0	1	0	1
	8	0	1	0	1	0.390	0.996
MP3 (kbit/sn)	128	0	1	0	1	0	1
	96	0	1	0	1	0	1
	64	0.018	1.000	0	1	0	1
OGG (%)	%80	0	1	0	1	0	1
	%60	0	1	0	1	0	1
	%40	0.002	1.000	0	1	0	1
Ortalama		0.002	1.000	**0**	**1**	0.043	1.000

Çizelge 4.22. Tayf yayılımı yöntemlerinin T_4 parametreleri ile fark edilmezlik karşılaştırmaları

GTY		**İTY**		**ÖTY**	
SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
13.94	2	12.71	1	**15.26**	**5**

$T_5 = \{L = 8192,\ \alpha = 0.003,\ \beta = 1,\ \gamma = 0.5\}$

Çizelge 4.23. Tayf yayılımı yöntemlerinin T_5 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

		GTY		**İTY**		**ÖTY**	
	Sıkıştırma	BER	NC	BER	NC	BER	NC
WAV	-	4.779	0.952	0.712	0.993	0	1
FLAC (bit/örnek)	16	4.864	0.951	0.695	0.993	0	1
	8	4.916	0.950	0.867	0.991	0.275	0.997
MP3 (kbit/sn)	128	7.018	0.929	2.033	0.979	0	1
	96	8.562	0.913	3.294	0.967	0	1
	64	12.286	0.875	7.773	0.921	0	1
OGG (%)	%80	6.881	0.930	1.930	0.980	0	1
	%60	7.747	0.921	2.574	0.974	0	1
	%40	9.971	0.907	4.084	0.959	0	1
Ortalama		7.447	0.925	2.662	0.973	**0.031**	**1.000**

Çizelge 4.24. Tayf yayılımı yöntemlerinin T_5 parametreleri ile fark edilmezlik karşılaştırmaları

GTY		**İTY**		**ÖTY**	
SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
24.78	3	22.86	2	16.57	**5**

$T_6 = \{L = 16384,\ \alpha = 0.001,\ \beta = 1,\ \gamma = 0.5\}$

Çizelge 4.25. Tayf yayılımı yöntemlerinin T_6 parametreleri ile sıkıştırmaya karşı sağlamlık karşılaştırmaları

		GTY		**İTY**		**ÖTY**	
	Sıkıştırma	BER	NC	BER	NC	BER	NC
WAV	-	1.734	0.982	0.137	0.999	0	1
FLAC (bit/örnek)	16	1.769	0.982	0.137	0.999	0	1
	8	1.889	0.981	0.258	0.997	0.291	0.997
MP3 (kbit/sn)	128	3.262	0.967	0.550	0.994	0	1
	96	4.361	0.956	1.202	0.988	0	1
	64	7.297	0.926	3.709	0.962	0	1
OGG (%)	%80	3.108	0.968	0.498	0.995	0	1
	%60	3.743	0.962	0.807	0.992	0	1
	%40	5.031	0.949	1.545	0.984	0	1
Ortalama		3.577	0.964	0.983	0.99	**0.032**	**1.000**

Çizelge 4.26. Tayf yayılımı yöntemlerinin T_6 parametreleri ile fark edilmezlik karşılaştırmaları

GTY		**İTY**		**ÖTY**	
SNR	ÖDT	SNR	ÖDT	SNR	ÖDT
25.67	4	24.92	3	17.03	**5**

Tayf yayılımı ile veri gizleme yöntemlerinin karşılaştırmalarında bölüt uzunlukları ve gizleme parametreleri kademeli olarak değiştirilerek 6 aşamada incelenmiştir. Sağlamlık değerlerinde kullanılan mp3 sıkıştırması için LAME (Anonim, 2011) tabanlı harici MATLAB fonksiyonu (Ellis, 2009) kullanılmıştır. FLAC ve OGG formatları için MATLAB programının kendi hazır ses okuma ve yazma fonksiyonları kullanılmıştır.

Her bir yöntemde gizlenen verinin fark edilmezliğini eşit seviyede korumak için geniş sinyalleri kullanılmış ve bölüt geçişleri Hann pencere ile yumuşatılmıştır.

GTY ve İTY için performansların eşit karşılaştırılabilmesi için tüm yapılan deneylerde $\beta = 1$ olarak seçilmiştir. GTY ve İTY yöntemlerinin SNR değerleri genel olarak 20 dB üzerinde hesaplanmış olmasına rağmen gizlemede kullanılan sözde-rastgele dizinin kullanılma şekline bağlı olarak oluşturduğu gürültü tüm deneylerde net olarak hissedilmektedir. α gizleme parametresi 0.001 olarak seçildiğinde sözde-rastgele dizinin oluşturduğu gürültü hissedilmeyecek seviyelere kadar inmesine karşın, küçük α değerlerinde büyük veri kayıpları gözlenmiştir.

ÖTY için gizleme parametreleri her bir bölütün enerji seviyesine göre hesaplandığı için diğer yöntemlerde özellikle enerjisi yüksek kısımlarda görülen veri kayıpları asgari seviyelere indirgenmiştir. SNR hesaplamaları 20 dB altında hesaplanmasına karşın özellikle bölüt uzunluğunun 512 örnekten fazla olduğu deneylerde gizlenmiş veri öznel dinleme testleri ile fark edilememiştir. ÖTY yönteminde MP3 ve OGG için iyi bir performans sergilenmesine karşın nicemleme örneklemesinin 8 bit olarak kodlanarak sıkıştırıldığı FLAC dönüşümlerinde düşük oranlarda veri kayıpları yaşanmıştır. Gürültü eklenerek yapılan GTY ve İTY yöntemlerinde ise FLAC dönüşümlerinde veri kayıplarından yaşanan değişim daha düşük olarak ölçülmüştür.

GTY ve İTY yöntemleri için fark edilmezlik ve sağlamlık arasındaki ideal denge bölüt uzunlukları genel ve yaklaşık olarak 16000 örnekten daha fazla seçildiğinde sağlanmaktadır. GTY ve İTY yöntemleri için sözde-rastgele dizi kullanılmaz ya da kullanım şekli değiştirilir ise öznel dinleme testlerinde fark edilmezlik oranları da yükselecektir, ancak ÖTY yönteminde olduğu gibi FLAC gibi nicemleme bit seviyesi üzerinden yapılan dönüşümlere karşı hassaslaşma görülecektir.

Kapasite karşılaştırmalarında ise $L = 256$ için kapasite $44100 \div 256 \cong 172.27$ bit/sn iken $L = 8192$ için kapasite $44100 \div 8192 \cong 5.38$ bit/sn olarak hesaplanır.

4.2.3. Önemsiz Bit Kodlaması Yöntemi

Yapılan deneyler sonucu standart hali ile en önemsiz bit kodlamasının sıkıştırmaya ve diğer saldırılara karşı tamamen dayanıksız olduğu görülmüştür. Dolayısıyla LSB yöntemi karşılaştırma için yalnızca kapasite ve fark edilmez çerçevesinde değerlendirilmiş ve sonuçlar Çizelge 4.7'de karşılaştırılmıştır. Kullanılan önemsiz bit sayısına (KÖBS) göre 10 bite kadar gizleme yapılmış ve SNR değerleri hesaplanarak öznel dinleme testleri (ÖDT) ile seslerde oluşan bozulmalar gözlemlenmiştir. Ayrıca 44.1 kHz ile örneklenmiş ses örnekleri için kullanılan her bite karşılık kapasite değerleri kbit/sn olarak karşılaştırılmıştır.

Çizelge 4.27. LSB kodlaması için fark edilmezlik ve kapasite karşılaştırmaları

KÖBS	SNR	ÖDT	Kapasite
1 bit	58.11	5	44.1 kbit/sn
2 bit	49.67	5	88.2 kbit/sn
3 bit	42.66	5	132.3 kbit/sn
4 bit	36.20	4	176.4 kbit/sn
5 bit	29.96	4	220.5 kbit/sn
6 bit	23.84	3	264.6 kbit/sn
7 bit	17.78	2	308.7 kbit/sn
8 bit	11.74	2	352.8 kbit/sn
9 bit	5.66	1	396.9 kbit/sn
10 bit	-0.59	1	441 kbit/sn

LSB kodlaması sağlamlık hususunda çok zayıf bir yöntem olmasına karşın, kapasite olarak oldukça yüksek değerlere ulaşmaktadır. Gönderilen gizli bilgi için kullanılan ses örtüsünün sıkıştırma ya da tür dönüşümleri gibi saldırılara uğramadan orijinal hali ile iletilebileceği varsayılır ise, LSB kodlaması kapasite olarak en kullanışlı yöntemlerin başında gelir.

Farklı örnekleme oranları ile örneklenmiş ses örtüleri için kullanılan bit sayısına göre kapasite değerleri birbirinden farklı olacaktır. Kapasite değeri genel olarak Denklem 4.7'deki gibi hesaplanabilir.

$$\text{Kapasite} = \text{Kullanılan Bit Sayısı} \times \text{Örnekleme Oranı} \tag{4.7}$$

4.3. Hibrit Sistemler

Veri gizleme yöntemleri kullanılan veri gizleme algoritması ve verinin gizlendiği dönüşüm uzayına bağlı olarak farklı saldırılara karşı farklı oranlarda dayanıklılık gösterir. Bir yöntem belli bir grup saldırılara karşı oldukça başarılı iken diğer bir grup saldırılara karşı zayıf kalabilir. Örneğin, Bölüm 4.2 altında detaylı incelenen yankı veri gizleme ve tayf yayılımı algoritmaları karşılaştırıldığında tayf yayılımı yönteminin sıkıştırma algoritmaları karşısında dayanıklılık ve veri kapasitesi hususunda daha başarılıdır, ancak tayf yayılımının gizleme algoritmasına bağlı olarak diğer saldırılar altında yankı veri gizlemeye kıyasla zayıf kalacağı açıktır.

Veri gizleme uygulamalarında, gizlenen verinin diğer saldırı çeşitlerine karşı dayanıklılığının ön planda olduğu durumlar için çeşitli yöntemler aynı anda kullanarak dayanıklılık arttırılabilir. Aynı veri tek bir dönüşüm uzayında tekrarlı bir biçimde gizlenebileceği gibi, DWT gibi dönüşüm uzayları da kullanılarak farklı frekans seviyelerine tekrarlı gizleme yapılarak sağlamlık artışları araştırılabilir.

Bu çalışmada, yankı veri gizleme yöntemi ve performans iyileştirilmesi ile önerilen tayf yayılımı yöntemi birlikte kullanılarak deneyler yapılmıştır. Gizlenecek veri örtü ses sinyali üzerine öncelikle ileri-geri ve negatif-pozitif yankı çekirdeği ile sinyal boyunca gizlenmiş, ardından tayf yayılımı yöntemi kullanılarak aynı (ya da farklı) veri giz sinyal üzerine tekrar gizlenmiştir. Her iki yöntem için gizleme aralıkları gizlenecek verilerin rastgele çakışmaları için farklı seçilmiştir.

Çizelge 4.28. Hibrit sistemde kullanılan parametreler

Parametre	**Açıklama**
L_1	ÖTY için bölüt uzunluğu
L_2	İGNPY için bölüt uzunluğu
α	İGNPY için yankı genliği
d_0	İGNPY için bit 0 gecikme miktarı
d_1	İGNPY için bit 1 gecikme miktarı

Sağlamlık ve fark edilmezlik karşılaştırması için 76 adet ses örneği üzerinde farklı değişkenler ile iki farklı deney yapılmıştır. Karşılaştırmalarda kolaylık açısından deneylerde kullanılan değişkenler $\{H_1, H_2\}$ parametreleri içerisinde incelenmiştir.

$H_1 = \{L_1 = 512,\ L_2 = 4096,\ \alpha = 0.4,\ d_0 = 250,\ d_1 = 350\}$

Çizelge 4.29. İGNPY ve ÖTY ile oluşturulan hibrit sistemin H_1 parametreleri ile sıkıştırmaya karşı sağlamlık ve fark edilmezlik hesaplamaları

		İGNPY		**ÖTY**			
	Sıkıştırma	BER	NC	BER	NC	SNR	ÖDT
WAV	-	0.897	0.991	0	1	9.643	4
MP3 (kbit/sn)	128	0.867	0.991	0	1	9.696	4
	96	0.879	0.991	0	1	9.642	4
	64	0.995	0.990	0	1	9.320	4
Ortalama		0.910	0.991	0	1	9.575	4

$H_2 = \{L_1 = 1024,\ L_2 = 8192,\ \alpha = 0.4,\ d_0 = 250,\ d_1 = 350\}$

Çizelge 4.30. İGNPY ve ÖTY ile oluşturulan hibrit sistemin H_2 parametreleri ile sıkıştırmaya karşı sağlamlık ve fark edilmezlik hesaplamaları

		İGNPY		**ÖTY**			
	Sıkıştırma	BER	NC	BER	NC	SNR	ÖDT
WAV	-	0.343	0.997	0	1	11.22	5
MP3 (kbit/sn)	128	0.343	0.997	0	1	11.40	5
	96	0.352	0.996	0	1	11.31	5
	64	0.370	0.996	0	1	10.85	5
Ortalama		0.352	0.997	0	1	11.20	5

Karşılaştırmalarda BER ve NC değerleri iki farklı algoritma ile aynı örtü üzerine üst üste yazılarak gizlenen veriler giz dosyası içerinden her bir algoritmadan geri çekilebilen veri miktarları baz alınarak ayrı ayrı hesaplanmıştır.

İGNPY ve ÖTY yöntemleri ile verilerin aynı bölgelerde üst üste yazılmasına rağmen aynı parametreler ile hibrit sistemdeki başarı oranlarının Çizelge 4.29 için Çizelge 4.9 ve Çizelge 4.30 için Çizelge 4.11 ile paralellik gösterdiğinden yankı veri gizleme yönteminin üzerine tayf yayılımı ile veri yazmaya karşı dayanıklı olduğu, dolayısıyla bu iki sistemin aynı anda hibrit şekilde kullanılabileceği sonucu çıkmaktadır.

5. SONUÇ

Ses içerisine veri gizleme yöntemlerinin eksiksiz anlaşılabilmesi; işitme duyusu ve sesin yapısının, temel dijital sinyal işleme kuramlarının, sıkıştırma algoritmaları ve cebir başta olmak üzere matematiksel modellemelerin iyi anlaşılıp yorumlamasına büyük oranda bağlıdır. Taşıyıcı ses aracılığıyla gizli ve güvenli veri aktarımı için steganografi ya da steganografi yöntemleri kullanılarak kopya koruma amaçlı damgalama yöntemleri ve uygulamaları üzerine literatürde birçok çalışma yapılmıştır. Ancak mevcut nitelikli çalışmaların büyük bir kısmında karmaşık modellemeler kullanıldığı için gizleme ve geri çekme adımları yeterince açık değildir.

İnternet ortamında günümüz teknolojisinin en uç noktası olan yapay zekâ algoritmalarının bile açık kaynak kodlarına kolayca ulaşmak mümkün iken, özellikle ses içerisine veri gizleme üzerine yapılmış mevcut çalışmaları temel alan açık kaynak kodlarına ulaşmak neredeyse imkansızdır. Bu sebeple, bu tez çalışmasının en temel amacı mevcut temel veri gizleme yöntemlerini içeren geniş kapsamlı açık kaynak kodlu bir kütüphane oluşturabilmektir. Kütüphane tasarımı için mevcut programlama dilleri arasından, yazılan kodların kolay okunup uygulanabilirliği ile paralel olarak bilimsel araştırmalarda kullanılma oranı göz önünde bulundurularak MATLAB platformu seçilmiştir.

Bu tez çalışmasında yöntemlerin sağlamlık karşılaştırmaları MP3 gibi kayıplı sıkıştırma formatlarına karşı dayanıklılıkları üzerinden yapılmıştır. Yöntemler içerisinden özellikle tayf yayılımı, yankı veri gizleme ve önemsiz bit kodlaması yöntemleri üzerinde durulmuştur.

Veri saklama kapasitesi ve fark edilmezlik açısından en etkili yaklaşımın önemsiz bit kodlamasını temel alan yöntemler olduğu görülmüştür, ancak yüksek kapasite ile ters orantılı olarak veri sağlamlığının oldukça düşük olduğu gözlemlenmiştir.

Yankı veri gizleme yöntemleri kapasite olarak incelenen diğer yöntemlere kısayla daha zayıf kalmasına rağmen, kaldırabildiği saldırı sayısı açısından daha üst düzey bir performans sergileyecektir. Her ne kadar bu çalışmada veri sağlamlığı yalnızca sıkıştırmaya karşı incelenmiş olsa da yankı veri gizleme yöntemleri beyaz gürültü ekleme, frekans süzgeçleri gibi saldırılara karşı da oldukça dayanıklıdır.

Yankı veri gizleme yöntemleri içerisinde yapılan karşılaştırma çalışmalarına göre negatif, pozitif, ileri ve geri simetrik yankılar ile oluşturulan yankı çekirdeğinin başarı oranının hem sağlamlık (BER ve NC) hem fark edilmezlik (SNR) açısından daha yüksek olduğu sonucuna varılmıştır.

Tayf yayılımı yöntemleri incelendiğinde, mevcut birçok çalışmanın kaynak-bağımlı (non-blind) olduğu görülmüştür. Kaynak-bağımsız (blind) olarak incelendiğinde ise tayf yayılımının başarı oranı kullanılacak örtü sesin karakteristik özellikleri ile doğrudan ilintilidir. Bu doğrultuda, sıralı dizi tayf yayılımı yönteminin kayıplı sıkıştırma karşısında başarı oranını arttırma amacıyla bir dizi çalışma yapılıp paylaşılmıştır. Örtü ses üzerine veri gizleme aşamasından önce uygulanan simetrik hareketli ortalamalar süzgeci ve aritmetik ortalamalar ile oluşturulan ikinci bir geçiş sinyali ile mevcut yöntemin MP3 sıkıştırma standardı karşısında başarı oranı 64 kbit/sn'ye kadar %100'e yakın olarak ölçülmüştür. Her bölütün kendi enerji seviyeleri kullanılarak değişken şekilde oluşturulan gizleme parametreleri ile öznel dinleme testlerindeki fark edilmezlik oranı asgari seviyeye indirgenmiştir. Nitekim yaklaşık 172 bit/sn kapasiteye kadar öznel dinleme testlerindeki fark edilmezlik büyük ölçüde korunmuştur.

Herhangi bir yöntem ile ses ya da diğer bir dijital veri içerisine hiçbir saldırı altında bozulmayacak sağlamlık seviyesi ile görünmez bir şekilde veri gizleyebilmek kopya koruma adına oldukça önemli bir iştir. Ancak örtü verinin yapısını bozmadan bu seviyede bir damgalama yöntemi geliştirmek neredeyse imkânsızdır.

Bu çalışma, mevcut algoritma kütüphanesi diğer mevcut yöntemler eklenerek ve verilen mevcut algoritmalar üzerinden diğer saldırı yöntemleri altında sağlamlık karşılaştırmaları ile birlikte genişletilebilir. Makine öğrenmesi ile eğitim ve test kümeleri oluşturularak örtü sese bağlı en ideal veri gizleme algoritması yapay öğrenme sonucu bilgisayar tarafından seçilerek veri gizleme işlemi otomatik şekilde yapılabilir.

KAYNAKLAR

Ahmed, N., Natarajan, T., Rao, K.R. 1974. Discrete Cosine Transform. **IEEE Transactions on Computers,** 23(1): 90-93.

Anderson, R., Petitcolas, F. 1998. On the limits of steganography. **IEEE Journal on Selected Areas in Communications,** 16(4): 474-481.

Anonim, 2011. LAME Project [http://lame.sourceforge.net/] Erişim Tarihi: 01.06.2017.

Anonim, 2015. Geri maskeleme örnekleri [http://www.backwardmasking.com/] Erişim Tarihi: 01.06.2017.

Bender, W., Gruhl, D., Morimoto, N. 1996. Techniques for data hiding. **IBM Systems Journal,** 35(3): 313-336.

Bhat, V., Sengupta, K. I., Das, A. 2010. An adaptive audio watermarking based on the singular value decomposition in the wavelet domain. **Digital Signal Processing,** 2010(20): 1547-1558.

Chen, B., Wornell, G.W. 2001. Quantization index modulation: a class of provably good methods for digital watermarking and information embedding. **IEEE Transactions on Information Theory,** 47(4): 1423-1443.

Chowdhury, A.K., Khan, I. 2013. A tutorial for audio watermarking in the cepstrum domain. **Smart Computing Review** [Electronic Journal], 3(5): 323-335. [http://www.smartcr.org/view/download.php?filename=smartcr_vol3no5p3.pdf], Erişim Tarihi: 01.06.2017.

Cooley, J.W., Tukey, J.W. 1965. An algorithm for the machine calculation of complex Fourier series. **Mathematics of Computation,** 19(90): 297-301.

Cox, I.J., Kilian, J., Leigton, F.T., Shamoon, T., 1996. Secure Spread Spectrum Watermarking for Multimedia. **IEEE Transactions on Image Processing,** 6: 1673-1687.

Cox, I.J., Miller, M.L. 2002. The first 50 years of electronic watermarking. **Applied Signal Processing,** 56(2): 225-230.

Cvejic, N., Seppänen, T. 2002. A wavelet domain LSB insertion algorithm for high capacity audio steganography. In **Proceedings of IEEE Digital Signal Processing Workshop,** pp. 53-55. Callaway Gardens, GA.

Cvejic, N., Seppänen, T. 2002. Increasing the capacity of LSB-based audio steganography. In **Proceedings of IEEE International Workshop on Multimedia Signal Processing,** pp. 336-338.

Cvejic, N., Seppänen, T. 2004. Spread spectrum audio watermarking using frequency hopping and attack characterization. **Signal Processing,** 84: 207-213.

Cvejic N., Seppänen T. 2008. Spread Spectrum for Digital Audio Watermarking. In: **Digital Audio Watermarking Techniques and Technologies,** (Klinger, K., vd. Eds.), IGI Global, pp. 11-49.

Daubechies, I. 1997. Ten Lectures on Wavelets (CBMS-NSF Regional Conference Series in Applied Mathematics). ISBN: 978-0-898712-74-2.

Ellis, D., 08.12.2009. Matlab için mp3 okuma ve yazma fonksiyonları [http://www.ee.columbia.edu/~dpwe/resources/matlab/mp3read.html] Erişim Tarihi: 01.06.2017.

Fan, M., Wang, H. 2009. Chaos-based discrete fractional sine transform domain audio watermarking scheme. **Comp. Elect. Eng.,** 35(3): 506-516.

Gang, L., Akansu, A., Ramkumar, M. 2001. Mp3 resistant oblivious steganography. In **Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing,** pp. 1365-1368. Salt Lake City, UT.

Gruhl, D., Lu, A., Bender, W. 1996. Echo hiding. In **Proceedings of the Information Hiding Workshop,** pp. 295-315.

Huang, N.E. 1998. The empirical mode decomposition and Hilbert spectrum for nonlinear and non-stationary time series analysis. In **Proceeding of R. Soc.,** 454(1971): pp. 903-995.

Kim, H.J. 2003. Audio watermarking techniques. In **Proceeding of Pacific Rim Workshop on Digital Steganography,** pp. 1-17. Kyushu Institute of Technology, Kitakyushu, Japan, Jul. 3-4.

Kim, H.J., Choi, Y.H. 2003. A novel echo-hiding scheme with backward and forward kernels. **IEEE Transactions on Circuits and Systems for Video Technology,** 13(8): 885-889.

Kirovski, D., Malvar, H.S. 2001. Robust spread spectrum audio watermarking. In **Proceeding of the IEEE International Conference on Acoustics, Speech, and Signal Processing,** 3: pp. 1345-1348.

Kirovski, D., Attias, H. 2002. Audio watermark robustness to de-synchronization via beat detection. In **Proceeding of Information Hiding Lecture Notes in Computer Science,** 2578: 160-175.

Kirovski, D., Malvar, H.S. 2003. Spread spectrum watermarking of audio signals. **IEEE Transaction on Signal Processing Special Issue on Data Hiding**, 51(4): 1020-1033.

Ko, B., Nishimura, R., Suzuki, Y. 2002a. Proposal of an echo-spread watermarking method using PN sequence. In **Proceedings of Spring Meeting of The Acoustical Society of Japan,** pp. 535-536.

Ko, B., Nishimura, R., Suzuki, Y. 2002b. Time-spread echo method for digital audio watermarking using PN sequences. In **Proceedings of ICASSP,** 2: pp. 2001-2004.

Ko, B., Nishimura, R., Suzuki, Y. 2004a. Log-Scaling watermark detection in digital audio watermarking. In **Proceedings of ICASSP,** 3: pp. 81-84.

Ko, B., Nishimura, R., Suzuki, Y. 2004b. Robust watermarking based on time-spread echo method with subband decomposition. **IEICE Transactions on Fundamentals,** 87(6): 1647-1650.

Ko, B., Nishimura, R., Suzuki, Y. 2005. Time-spread echo method for digital audio watermarking. **IEEE Transactions on Multimedia,** 7(2): 212-221.

Kuo, S., Johnston, J., Turin, W., Quackenbush, S. 2002. Covert audio watermarking using perceptually tuned signal independent multiband phase modulation. In **Proceedings IEEE International Conference on Acoustics, Speech, and Signal Processing,** pp. 1753-1756. Orlando, FL, USA

Lee, S.K., Ho, Y.S. 2000. Digital audio watermarking in the cepstrum domain. **IEEE Transactions on Consumer Electronics,** 46(3): 744-750.

Lei, B.Y., Soon, I.Y., Li, Z. 2011. Blind and robust audio watermarking scheme based on SVD-DCT. **Signal Processing,** 91(8): 1973-1984.

Lei, B.Y., Soon, I.Y., Tan, E.L. 2013. Robust svd-based audio watermarking scheme with differential evolution optimization. **IEEE Transactions on Audio, Speech, and Language Processing,** 21(11): 2368-2377.

Malvar, H.S., Florencio, D.A. 2003. Improved spread spectrum: A new modulation technique for robust watermarking. **IEEE Transactions on Signal Processing,** 51(4): 898-905.

McLoughlin, I. 2009. **Applied Speech and Audio Processing With Matlab Examples**. Cambridge University Press, Nanyang Technological University, Singapore.

Megías, D., Serra-Ruiz, J., Fallahpour, M. 2010. Efficient self–synchronized blind audio watermarking system based on time domain and FFT amplitude modification. **Signal Processing,** 90(12): 3078-3092.

Misiti, M., Misiti Y., Oppenheim, G., Poggi, J. 2007. Wavelets and their Applications. ISBN: 978-1-905209-31-6.

Mohsenfar, S.M., Mosleh, M., Bharti, A., 2013. Audio Watermarking method using QR decomposition and Genetic Algorithm. **Multimedia Tools and Applications,** 74: 759-779.

Oh, O.H., Seok, W.J., Hong, W.J., Youn, H.D. 2001. New echo embedding technique for robust and imperceptible audio watermarking. In **Proceedings of ICASSP,** 3: pp. 1341-1344.

Özer, H., Sankur, B., Memon, N. 2005. An SVD-based audio watermarking technique. In **Proceedings of 7th Workshop Multimedia Security,** pp. 51–56.

Petitcolas, F. 13.06.2006. Eşlik kodlamasına örnek olarak MP3Stego uygulaması [http://www.petitcolas.net/steganography/mp3stego] Erişim Tarihi: 01.06.2017.

Petrovic, R. 2001. Audio signal watermarking based on replica modulation. In **Proceedings of International Conference on Telecommunications in Modern Satellite, Cable, and Broadcasting Service,** 1: pp. 227-234.

Pickholtz R. L., Schilling D. L., Millstein L.B., 1982. Theory of spread spectrum communications - A tutorial. **IEEE Transactions on Communications,** 30: pp. 855-884.

Ruiz, F., Deller, J. 2000. Digital watermarking of speech signals for the national gallery of the spoken word. In **Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing,** pp. 1499-1502. Istanbul, Turkey.

Smith, J.O. 2007. **Mathematics of the Discrete Fourier Transform (DFT),** [https://ccrma.stanford.edu/~jos/mdft/] Erişim Tarihi: 01.06.2017.

Sweldens, W. 1996. The lifting scheme: a custom-design construction of biorthogonal wavelets. **Appl. Computational Harmonic Analysis,** 3(2): 186-200.

Wang, L., Emmanuel, S., Kankanhalli, M.S. 2010. EMD and psychoacoustic model based watermarking for audio. In **Proceedings IEEE ICME,** pp. 1427-1432.

Xiang, S.J., Huang, J.W. 2007. Histogram based audio watermarking against time scale modification and cropping attacks. **IEEE Transactions on Multimedia,** 9(7): 1357-1372.

Yeo, I.K., Kim, H.J. 2003. Modified Patchwork Algorithm: A novel audio watermarking scheme. **IEEE Transactions on Speech and Audio Processing,** 11: 381-386.

EKLER

Ek 1. MATLAB Kaynak Kodları

Ek 1.1. Ortak Fonksiyonlar

prng.m

```
 1 - function out = prng(key, L)
 2 - sifre = sum(double(key).*(1:length(key)));
 3 - rand('seed', sifre);
 4 - out = 2*(rand(L,1)>0.5)-1;
 5 - end
```

Sözde rastgele dizi üretici

hanning.m

```
 1 - function out = hanning(L)
 2 - if isnumeric(L)
 3 -     L = round(L);
 4 -     if L == 1
 5 -         out = 1;
 6 -     elseif L>1 || L==0
 7 -         n   = (0:L-1)';
 8 -         out = .5*(1-cos((2*pi*n)/(L-1)));
 9 -     else
10 -         error('Girdi sifirdan buyuk olmali!');
11 -     end
12 - else
13 -     error('Girdi sayisal deger olmali!');
14 - end
15 - end
```

Hann pencere fonksiyonu

mixer.m

```
 1 - function [w_sig, m_sig] = mixer(L,bit,alt,ust,K)
 2
 3 - if (nargin < 4)
 4 -     alt=0; ust=1;
 5 - end
 6
 7 - if (nargin<5) || (2*K>L)
 8 -     K=floor(L/4)-mod(floor(L/4),4);
 9 - else
10 -     K=K-mod(K,4);
11 - end
12 - N    = length(bits);
13 - nbit = str2num(reshape(bits,N,1))';
14 - msig = reshape(ones(L,1)*nbit,N*L,1);
15 - c    = conv(msig,hanning(K));
16 - wnor = c(K/2+1:end-K/2+1)/max(abs(c));
17 - wsig = wnor*(ust-alt)+alt;
18 - end
```

Yumuşatılmış karıştırıcı sinyal üretici

Ek 1.2. LSB Kodlaması Yöntemi

Veri gizleme fonksiyonu

lsb_enc.m

```
 1 - function lsb_enc(wavin, text, wavout)
 2
 3 - if (nargin<3)
 4 -     wavout='test_stego.wav';
 5 - end
 6
 7   %baslik = 1:40, veri boyutu = 41:43, veri = 44:end
 8 - fid = fopen(wavin,'r');
 9 - header = fread(fid,40,'uint8=>char');
10 - dsize = fread(fid,1,'uint32');
11 - [data,count] = fread(fid,inf,'uint16');
12 - fclose(fid);
13
14   %Mesaj ikilik veriye donusturulur
15 - bin = d2b(double(text),8);
16 - [m,n] = size(bin);
17 - bits = reshape(bin,m*n,1);
18 - m = d2b(m,30)';
19 - n = d2b(n,10)';
20 - len = length(bits);
21
22 - if (count>len+48)
23       %kontrol degiskeni kodlanir (1:8)
24 -     kontrol = [1 1 1 1 1 1 1 1]';
25 -     data(1:8) = bitset(data(1:8),1,kontrol(1:8));
26       % Mesaj uzunlugu kodlanir (9:48)
27 -     data(9:38) = bitset(data(9:38),1,m(1:30));
28 -     data(39:48) = bitset(data(39:48),1,n(1:10));
29       %Mesaj kodlanir (49:48+len)
30 -     data(49:48+len)=bitset(data(49:48+len),1,bin(1:len)');
31       %Wav dosyasi kaydedilir
32 -     out = fopen(wavout,'w');
33 -     fwrite(out,header,'uint8');
34 -     fwrite(out,dsize,'uint32');
35 -     fwrite(out,data,'uint16');
36 -     fclose(out);
37 - else
38 -     error('Mesaj boyutu cok uzun!');
39 - end
40 - end
41
42 - function b = d2b(d, n)
43   %DE2BI fonksiyonunun minimal uygulamasi
44 - d = d(:);
45 - power = ones(length (d), 1) * (2.^(0:n-1));
46 - b = floor(rem(d*ones(1, n), 2*power)./power);
47 - end
48
```

Veri Çekme Fonksiyonu

lsb_dec.m

```
 1 - function out = lsb_dec(wavin)
 2
 3 - if (nargin<1)
 4 -     wavin = 'test_stego.wav';
 5 - end
 6
 7   %baslik = 1:40, veri boyutu = 41:43, veri = 44:end
 8 - fid = fopen(wavin,'r');
 9 - header = fread(fid,40,'uint8=>char');
10 - dsize = fread(fid,1,'uint32');
11 - data = fread(fid,inf,'uint16');
12 - fclose(fid);
13   %Kontrol degiskeni okunur (1:8)
14 - kontrol = bitget(data(1:8),1)';
15 - if (kontrol == [1 1 1 1 1 1 1 1])
16       %Baslangic bilgileri okunur (9:48)
17 -     m = bitget(data(9:38),1);
18 -     n = bitget(data(39:48),1);
19 -     len = b2d(m')*b2d(n');
20       %Gizli mesaj okunur (49:len)
21 -     bin = reshape(bitget(data(49:48+len),1),len/8,8);
22 -     out = char(bi2de(bin))';
23 - else
24 -     warning('Gizli mesaj tespit edilemedi!');
25 - end
26 - end
27
29 - function d = b2d(b)
30   %BI2DE fonksiyonunun minimal uygulamasi
31 - d = b * (2.^(0:length(b(1, :)) - 1)');
32 - end
33
```

LSB Kodlama Demo Programı

lsb_demo.m

```
 1 - close all; clear all; clc
 2
 3 - girdi = 'test.wav';
 4 - cikti = 'stego.wav';
 5 - mesaj = 'bu bir deneme mesajidir';
 6
 7   %Veri gizleme islemi
 8 - lsb_enc(girdi, mesaj, cikti);
 9
10   %Veri cekme islemi
11 - msg = lsb_dec(cikti)
12
13
```

Ek 1.3. Faz Kodlaması Yöntemi

Veri gizleme fonksiyonu

phase_enc.m

```
function out = phase_enc(signal,text,L)

if nargin < 3
    L = 1024;          %Bolut uzunlugu
end

bin = dec2bin(uint8(text),8);
bit = reshape(bin',1,8*length(text));
I = length(signal(:,1));
m = length(bit);
N = floor(I/L);        %Bolut sayisi
s = reshape(signal(1:N*L,1),L,N);

w = fft(s);            %Her bir parcaya fft uygulanir
Phi = angle(w);        %Faz matrisi
A = abs(w);            %Genlik matrisi

for k=2:N
    DeltaPhi(:,k)=Phi(:,k)-Phi(:,k-1); %Faz farklari
end

for k=1:m
    if (bit(k)=='0')
        PhiData(k) = pi/2;
    else
        PhiData(k) = -pi/2;
    end
end

%%%%%%%%%%%%%%%%%%%%% VERI GIZLENIYOR... %%%%%%%%%%%%%%%%%%%
Phi_new(:,1)=Phi(:,1);
Phi_new(L/2-m+1:L/2,1)=PhiData;
Phi_new(L/2+1+1:L/2+1+m,1)=-flip(PhiData);  %Eslenik simetri
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

for k=2:N
    Phi_new(:,k) = Phi_new(:,k-1)+DeltaPhi(:,k);
end

out = signal;
z   = real(ifft(A.*exp(1i*Phi_new)));  %Euler ozdesligi
out(:,1) = [reshape(z,N*L,1); signal(N*L+1:I,1)];
end
```

Veri çekme fonksiyonu

phase_dec.m

```
 1 - function out=phase_dec(signal,len_msg,L)
 2
 3 - if nargin < 3
 4 -     L = 1024;           %Bolut uzunlugu
 5 - end
 6
 7 - x   = signal(1:L,1);    %Ilk bolut
 8 - Phi = angle(fft(x));    %Faz matrisi
 9 - for k=1:len_msg
10 -     if Phi(L/2-len_msg+k)>0
11 -         data(k)='0';
12 -     else
13 -         data(k)='1';
14 -     end
15 - end
16 - bin = reshape(data(1:len_msg),8,len_msg/8)';
17 - out = char(bin2dec(bin))';
18 - end
19
```

Faz kodlaması demo programı

lsb_demo.m

```
 1 - close all; clear all; clc
 2
 3 - girdi = 'test.wav';
 4 - cikti = 'stego.wav';
 5 - mesaj = 'bu bir deneme mesajidir';
 6 - uzunluk = 8*length(mesaj);
 7
 8   %Ses ortusu acilir
 9 - [data, fs] = audioread(girdi);
10
11   %Veri gizleme islemi
12 - sifreli = phase_enc(data, mesaj);
13
14   %Veri gizlenmis ses dosyasi kaydedilir
15 - audiowrite(cikti, sifreli, fs);
16
17   %Veri gizlenmis ses dosyasi tekrar acilir
18 - deşifre = audioread(cikti);
19
20   %Veri cekme islemi
21 - msg = phase_dec(cikti, uzunluk)
22
23
24
25
```

Ek 1.4. Yankı Veri Gizleme Yöntemleri

Tekli yankı veri gizleme fonksiyonu

single_echo_enc.m

```
function out = echo_enc(signal,text,d1,d2,alpha,L)

if (nargin<4)
    d1 = 50;        %bit0 icin gecikme
    d2 = 75;        %bit1 icin gecikme
end

if (nargin<5)
    alpha = 0.5;    %Yanki genligi
end

if (nargin<6)
    L = 4096;       %Bolut uzunlugu
end

[s.len, s.ch] = size(signal);

bin = dec2bin(uint8(text),8);
bit = reshape(bin',1,8*length(text));

n = floor(s.len/L);
N = n-mod(n,8);     %Bolut sayisi

if (length(bit) > N)
    warning('Mesaj cok uzun, kirpilarak gizleniyor!');
    bits = bit(1:N);
else
    bits = [bit, num2str(zeros(N-length(bit),1))'];
end

%Gecikme sureleri ile yanki sinyalleri oluşturulur.
echo_zro = [zeros(d1,s.ch);signal(1:end-d1,:)]*alpha;
echo_one = [zeros(d2,s.ch);signal(1:end-d2,:)]*alpha;

mix = mixer(L,bits)*ones(1,s.ch);     %Karistirici sinyal

%%%%%%%%%%%%%%%%%%% YANKILAR GIZLENIYOR... %%%%%%%%%%%%%%%%%%
out = signal(1:N*L,:) + echo_zro(1:N*L,:).*abs(mix-1) ...
                      + echo_one(1:N*L,:).*mix;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

out = [out; signal(N*L+1:s.len, :)];  %Sesin kalan kismi
end
```

Negatif ve pozitif yankı veri gizleme fonksiyonu

np_echo_enc.m

```
function out = echo_enc(signal,text,d1,d2,alpha,L)

if (nargin<4)
    d1 = 50;         %bit0 icin gecikme
    d2 = 75;         %bit1 icin gecikme
end

if (nargin<5)
    alpha = 0.5;     %Yanki genligi
end

if (nargin<6)
    L = 4096;        %Bolut uzunlugu
end

[s.len,s.ch] = size(signal);
bin = dec2bin(uint8(text),8);
bit = reshape(bin',1,8*length(text));
n = floor(s.len/L);
N = n-mod(n,8);      %Bolut sayisi

if (length(bit) > N)
    warning('Mesaj cok uzun, kirpilarak gizleniyor!');
    bits = bit(1:N);
else
    bits = [bit, num2str(zeros(N-length(bit),1))'];
end

%Gecikme sureleri ile yanki sinyalleri oluşturulur.
b_echo_zro = [zeros(d1,s.ch);signal(1:end-d1,:)];
b_echo_one = [zeros(d2,s.ch);signal(1:end-d2,:)];

echo_zro = alpha*(b_echo_zro - b_echo_one)/2;
echo_one = alpha*(b_echo_one - b_echo_zro)/2;
clear b_echo_zro b_echo_one;

mix = mixer(L,bits)*ones(1,s.ch);     %Karistirici sinyal

%%%%%%%%%%%%%%%%%% YANKILAR GIZLENIYOR... %%%%%%%%%%%%%%%%%
out = signal(1:N*L,:) + echo_zro(1:N*L,:).*abs(mix-1) ...
                      + echo_one(1:N*L,:).*mix;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
out = [out; signal(N*L+1:s.len, :)];  %Sesin kalan kismi
end
```

İleri ve geri yankı veri gizleme fonksiyonu

bf_echo_enc.m

```
function out = echo_enc(signal,text,d1,d2,alpha,L)

if nargin < 4
    d1 = 50;        %bit0 icin gecikme
    d2 = 75;        %bit1 icin gecikme
end

if nargin < 5
    alpha = 0.5;    %Yanki genligi
end

if nargin < 6
    L = 4096;       %Bolut uzunlugu
end

[s.len,s.ch] = size(signal);
bin = dec2bin(uint8(text),8);
bit = reshape(bin',1,8*length(text));
n = floor(s.len/L);
N = n - mod(n,8);   %Bolut sayisi

if (length(bit) > N)
    warning('Mesaj cok uzun, kirpilarak gizleniyor!');
    bits = bit(1:N);
else
    bits = [bit, num2str(zeros(N-length(bit),1))'];
end

%Gecikme sureleri ile yanki sinyalleri oluşturulur.
b_echo_zro = [zeros(d1,s.ch);signal(1:end-d1,:)]; %Bit0:geri
f_echo_zro = [signal(d1+1:end,:);zeros(d1,s.ch)]; %Bit0:ileri
b_echo_one = [zeros(d2,s.ch);signal(1:end-d2,:)]; %Bit1:geri
f_echo_one = [signal(d2+1:end,:);zeros(d2,s.ch)]; %Bit1:ileri

echo_zro = alpha*(b_echo_zro + f_echo_zro)/2;     %Bit0
echo_one = alpha*(b_echo_one + f_echo_one)/2;     %Bit1
clear b_echo_zro f_echo_zro b_echo_one f_echo_one;

mix = mixer(L,bits)*ones(1,s.ch);     %Karistirici sinyal

%%%%%%%%%%%%%%%%%%% YANKILAR GIZLENIYOR... %%%%%%%%%%%%%%%%%%
out = signal(1:N*L,:) + echo_zro(1:N*L,:).*abs(mix-1) ...
                      + echo_one(1:N*L,:).*mix;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
out = [out; signal(N*L+1:s.len, :)];  %Sesin kalan kismi
end


```

İG ve NP yankı veri gizleme fonksiyonu

bfnp_echo_enc.m

```
 1 - function out = echo_enc(signal,text,d1,d2,alpha,L)
 2
 3 - if (nargin<4)
 4 -     d1 = 50;         %bit0 icin gecikme
 5 -     d2 = 75;         %bit1 icin gecikme
 6 - end
 7
 8 - if (nargin<5)
 9 -     alpha = 0.5;     %Yanki genligi
10 - end
11
12 - if (nargin<6)
13 -     L = 4096;        %Bolut uzunlugu
14 - end
15
16 - [s.len,s.ch] = size(signal);
17 - bin = dec2bin(uint8(text),8);
18 - bit = reshape(bin',1,8*length(text));
19 - n = floor(s.len/L);
20 - N = n-mod(n,8);      %Bolut sayisi
21
22 - if (length(bit) > N)
23 -     warning('Mesaj cok uzun, kirpilarak gizleniyor!');
24 -     bits = bit(1:N);
25 - else
26 -     bits = [bit, num2str(zeros(N-length(bit),1))'];
27 - end
28
29   %Gecikme sureleri ile yanki sinyalleri oluşturulur.
30 - b_echo_zro = [zeros(d1,s.ch);signal(1:end-d1,:)]; %Bit0:geri
31 - f_echo_zro = [signal(d1+1:end,:);zeros(d1,s.ch)]; %Bit0:ileri
32 - b_echo_one = [zeros(d2,s.ch);signal(1:end-d2,:)]; %Bit1:geri
33 - f_echo_one = [signal(d2+1:end,:);zeros(d2,s.ch)]; %Bit1:ileri
34
35 - echo_zro = alpha*(b_echo_zro + f_echo_zro ...
36                       - b_echo_one - f_echo_one)/4; %Bit0
37 - echo_one = alpha*(b_echo_one + f_echo_one ...
38                       - b_echo_zro - f_echo_zro)/4; %Bit1
39 - clear b_echo_zro f_echo_zro b_echo_one f_echo_one;
40
41 - mix = mixer(L,bits)*ones(1,s.ch);     %Karistirici sinyal
42   %%%%%%%%%%%%%%%%%% YANKILAR GIZLENIYOR... %%%%%%%%%%%%%%%%%
43 - out = signal(1:N*L,:) + echo_zro(1:N*L,:).*abs(mix-1) ...
44                         + echo_one(1:N*L,:).*mix;
45   %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
46 - out = [out; signal(N*L+1:s.len, :)];  %Sesin kalan kismi
47 - end
48
```

TY, NPY, İGY ve İGNP için veri çekme fonksiyonu

echo_dec.m

```
function out=echo_dec(signal,L,d1,d2,len_msg)
%len_msg degiskeni mesaj uzunulugunun bilindigi durumlarda
kullanilir.

if nargin < 2
    L = 4096;   %Bolut uzunlugu
end

if nargin < 4
    d1 = 50;    %bit0 icin gecikme
    d2 = 75;    %bit1 icin gecikme
end

if nargin < 5
    len_msg = 0;
end

N = floor(length(signal)/L);      %Bolut sayisi
x = reshape(signal(1:N*L,1),L,N); %Ses N parcaya bolunur

for k=1:N
    ccorr=ifft(log(abs(fft(x(:,k))).^2)); %Kepstrum ilintisi
    %rceps=ifft(log(abs(fft(x(:,k)))));    %Gercel kepstrum
    if (ccorr(d1+1) >= ccorr(d2+1))
        data(k) = '0';
    else
        data(k) = '1';
    end
end

m = floor(N/8);
bin = reshape(data(1:8*m),8,m)';
out = char(bin2dec(bin))';

if (len_msg~=0)
    out = out(1:len_msg);
end
end
```

Zaman yayılımlı yankı veri gizleme fonksiyonu

ts_echo_enc.m

```
function out = echo_enc(signal,text,d1,d2,alpha,L)

if (nargin<4)
    d1 = 50;         %bit0 icin gecikme
    d2 = 75;         %bit1 icin gecikme
end

if (nargin<5)
    alpha = 0.02;    %Yanki genligi
end

if (nargin<6)
    L = 4096;        %Bolut uzunlugu
end

[s.len,s.ch] = size(signal);
bin = dec2bin(uint8(text),8);
bit = reshape(bin',1,8*length(text));
n = floor(s.len/L);
N = n - mod(n,8);    %Bolut sayisi

if (length(bit) > N)
    warning('Mesaj cok uzun, kirpilarak gizleniyor!');
    bits = bit(1:N);
else
    bits = [bit, num2str(zeros(N-length(bit),1))'];
end

pr = prng('sifre', 256);   %Sozde-rastgele dizi üretilir

%Gecikme sureleri ile yanki cekirdekleri oluşturulur:
ker_zro = [zeros(d1, 1); pr]*alpha;     %bit0
ker_one = [zeros(d2, 1); pr]*alpha;     %bit1

%Yanki cekirdekleri ile yanki sinyalleri oluşturulur:
echo_zro = filter(ker_zro,1,signal);    %bit0
echo_one = filter(ker_one,1,signal);    %bit1

mix = mixer(L,bits)*ones(1,s.ch);       %Karistirici sinyal

%%%%%%%%%%%%%%%%%% YANKILAR GIZLENIYOR... %%%%%%%%%%%%%%%%%%
out = signal(1:N*L,:) + echo_zro(1:N*L,:).*abs(mix-1) ...
                      + echo_one(1:N*L,:).*mix;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
out = [out; signal(N*L+1:s.len, :)];    %Sesin kalan kismi
end


```

ZYY için veri çekme fonksiyonu

ts_echo_dec.m

```
function out=echo_dec(signal,L,d1,d2,len_msg)
%len_msg degiskeni mesaj uzunulugunun bilindigi durumlarda
kullanilir.

if nargin < 2
    L = 4096;   %Bolut uzunlugu
end

if nargin < 4
    d1 = 50;    %bit0 icin gecikme
    d2 = 75;    %bit1 icin gecikme
end

if nargin < 5
    len_msg = 0;
end

N = floor(length(signal)/L);      %Bolut sayisi
x = reshape(signal(1:N*L,1),L,N); %Ses N parcaya bolunur

pr = prng('sifre',256);   %Sozde-rastgele dizi üretilir

for k=1:N
    cr = ifft(log(abs(fft(x(:,k)))+eps));   %Gercel kepstrum
    xc = xcorr(cr, pr);                     %Capraz ilinti

    if max(xc(L+d1-1:L+d1+1))>=max(xc(L+d2-1:L+d2+1))
        data(k) = '0';
    else
        data(k) = '1';
    end
end

m = floor(N/8);
bin = reshape(data(1:8*m),8,m)';
out = char(bin2dec(bin))';

if (len_msg~=0)
    out = out(1:len_msg);
end
end
```

Ek 1.5. Tayf Yayılımı Yöntemleri

Geleneksel tayf yayılımı veri gizleme fonksiyonu

dsss_enc.m

```
function out = dsss_enc(signal,text,key,L_min)

if (nargin<3)
    key = 'sifre';  %Sozde-rastgele dizi anahtari
end

if (nargin<4)
    L_min = 1024;   %Minimum bolut uzunlugu
end

[s.len,s.ch] = size(signal);

bin = dec2bin(uint8(text), 8);
bit = reshape(bin',1,8*length(text));

L2 = floor(s.len/length(bit));
L = max(L_min, L2);     %Bolut uzunluğu

n = floor(s.len/L);
N = n - mod(n, 8);      %Bolut sayisi

if (length(bit) > N)
    warning('Mesaj cok uzun, kirpilarak gizleniyor!');
    bits=bit(1:N);
else
    bits=[bit, num2str(zeros(N-length(bit),1))'];
end

alpha = 0.005;          %Guc parametresi

%r = ones(L,1);
r = prng(key,L);        %Sozde-rastgele dizi
pr = reshape(r*ones(1,N),N*L,1);

out = signal;
mix = mixer(L,bits,-1,1);

%%%%%%%%%%%%%%%%%%% VERI GIZLENIYOR... %%%%%%%%%%%%%%%%%%%%
stego = signal(1:N*L,1) + alpha*mix.*pr;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
out(:,1) = [stego; signal(N*L+1:s.len,1)];

end


```

İyileştirilmiş tayf yayılımı veri gizleme fonksiyonu

iss_enc.m

```
function out = dsss_enc(signal,text,key,L_min)

if (nargin<3)
    key = 'sifre';  %Sozde-rastgele dizi anahtari
end

if (nargin<4)
    L_min = 1024;   %Minimum bolut uzunlugu
end

[s.len,s.ch] = size(signal);
bin = dec2bin(uint8(text), 8);
bit = reshape(bin',1,8*length(text));
L2  = floor(s.len/length(bit));
L   = max(L_min, L2);     %Bolut uzunluğu
n   = floor(s.len/L);
N   = n - mod(n, 8);      %Bolut sayisi

if (length(bit) > N)
    warning('Mesaj cok uzun, kirpilarak gizleniyor!');
    bits=bit(1:N);
else
    bits=[bit, num2str(zeros(N-length(bit),1))'];
end

alpha = 0.05; beta = 0.1; gama = 0.5;
r  = prng(key,L);
pr = reshape(r*ones(1,N),N*L,1);
u  = alpha * pr;
x  = reshape(signal(1:N*L,1), L, N);

for k=1:N
    sx(:,k) = (sum(x(:,k).*r)/(L*alpha));
end
su = reshape(ones(L,1)*sx, N*L, 1);

out = signal;
mix = mixer(L,bits,-1,1);

%%%%%%%%%%%%%%%%%% VERI GIZLENIYOR... %%%%%%%%%%%%%%%%%%%%
stego = signal(1:N*L,1) + (beta*mix - gama*su).*u;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
out(:,1) = [stego; signal(N*L+1:s.len,1)];
end


```

GTY ve İTY için veri çekme fonksiyonu

dsss_dec.m

```
function out = dsss_dec(signal,L_msg,key,L_min)

if (nargin<3)
    key = 'sifre';
end

if (nargin<4)
    L_min = 1024;
end

[s.len,s.ch] = size(signal);

L2 = floor(s.len/L_msg);
L = max(L_min,L2);        %Bolut uzunlugu

n = floor(s.len/L);
N = n-mod(n,8);           %Bolut sayisi

x = reshape(signal(1:N*L,1),L,N); %Ses N parcaya bolunur

%r = ones(L,1);
r = prng(key,L);          %Sozde-rastgele dizi uretilir

for k=1:N
    if (sum(x(:,k).*r)/L)
        data(k)='0';
    else
        data(k)='1';
    end
end

bin = reshape(data(1:N),8,N/8)';
out = char(bin2dec(bin))';
end
```

ÖZGEÇMİŞ

KİŞİSEL BİLGİLER

Adı Soyadı : Kadir Tekeli
Doğum Yeri ve Tarihi : 18.08.1990

EĞİTİM DURUMU

Lisans Öğrenimi : Akdeniz Üniversitesi Fen Fak. Matematik Bölümü
Yüksek Lisans Öğrenimi : Adnan Menderes Üniversitesi Matematik A.B.D.
Bildiği Yabancı Diller : İngilizce, Rusça

BİLİMSEL FAALİYETLERİ

a) Bildiriler: : Çok Katmanlı Algılayıcı, K-NN ve C4.5 Metotlarıyla İstenmeyen E-postaların Tespiti, AB2016, 2016, ADÜ, Aydın
b) Katıldığı Projeler : Eco-Ambassadors, Youth Exchanges, 2011, Tbilisi, Georgia

İŞ DENEYİMİ

Çalıştığı Kurumlar ve Yıl : Söke Kavram Dershanesi, 2013-2014
Antalya Kampüs Dershanesi, 2012-2013

İLETİŞİM

E-posta : kadir.tekeli@gmail.com
Tarih : 01.06.2017