

BURSA TEKNİK ÜNİVERSİTESİ

BİLGİSAYAR MÜHENDİSLİĞİ

ALİEKBER GÖRMÜŞ

24360859061



VERİ DEPOLAMA VE SIKIŞTIRMA

- İÇİNDEKİLER

- 1) Metin, resim ve ses verilerinin bit düzeyinde temsili.**

- a) Metin verilerinin temsili

- b) Resim verilerinin temsili

- c) Ses verilerinin temsili

- 2) Veri sıkıştırma neden gereklidir?**

- a) Kayıplı (Lossy) Sıkıştırma:

- b) Kayıpsız (Lossless) Sıkıştırma

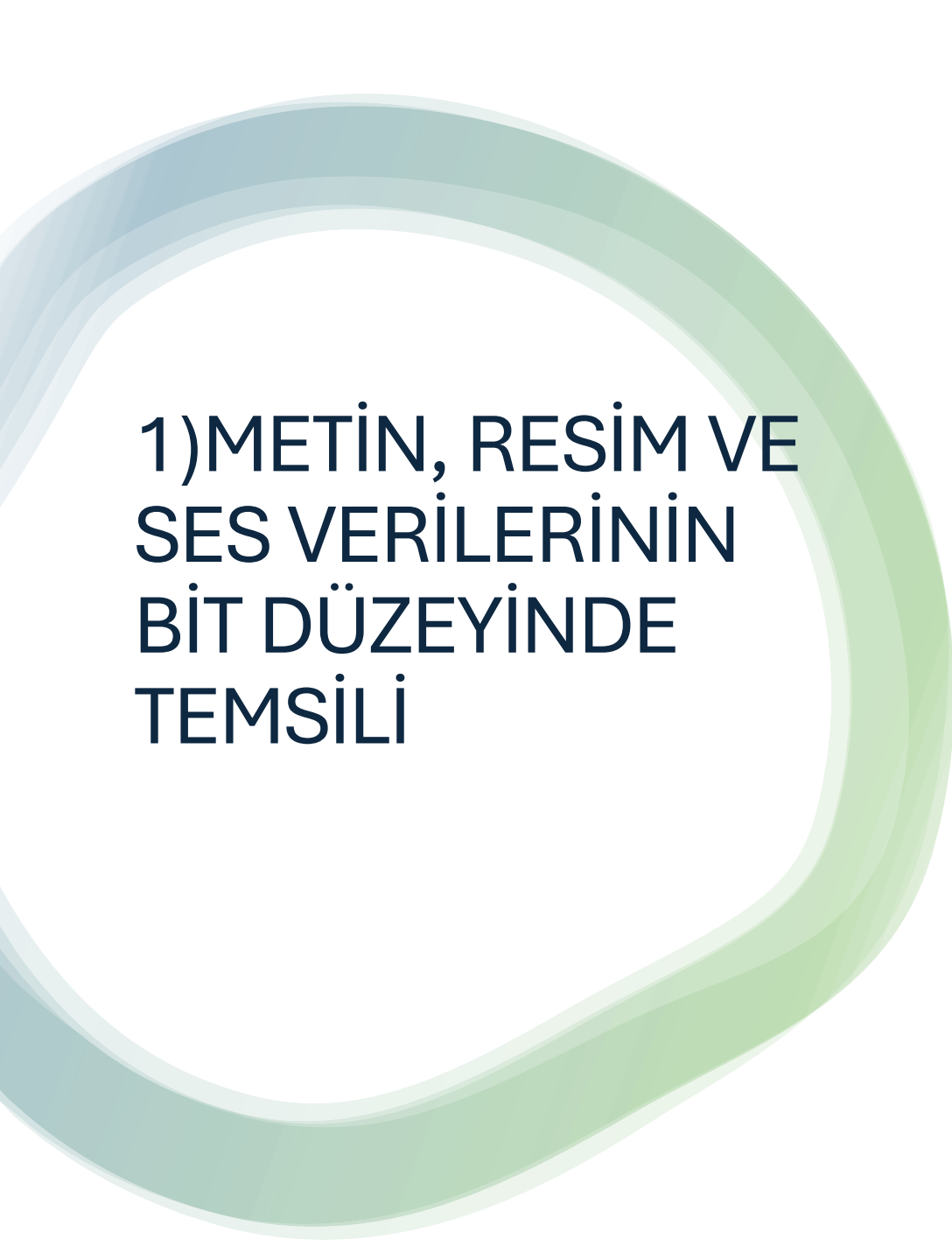
- 3) Run-Length Encoding (RLE) gibi temel sıkıştırma mantıkları.**

- a) İşlem Uzunluğu Kodlama (Run-Length Encoding)

- b) Göreceli Kodlama (Relative Encoding)

- c) Frekans Bağımlı Kodlama (Huffman Code Data Compression)

- d) Sözlük Kodlama (Dictionary-based encoding)



1) METİN, RESİM VE SES VERİLERİNİN BİT DÜZEYİNDE TEMSİLİ

- **Bit** : Bilgisayar ve dijital iletişim sistemlerindeki en küçük veri depolama birimidir.
- Sadece **0 (kapalı)** veya **1 (açık)** olmak üzere iki farklı değer alabilir.
- Tek başına anlamlı bir bilgi taşımasa da, bitler bir araya gelerek harfleri, sayıları ve görselleri meydana getirir.
- 8 bit bir araya gelerek 1 byte oluşturur.

a)Metin Verilerinin Temsili

- Metinleri Nasıl Saklarız?
- Her harfin (A, B, C...) bilgisayarda sayısal bir karşılığı vardır. Buna ASCII standardı denir. Örneğin 'A' harfinin karşılığı 65 sayısına denk gelir, bunu da ikilik sistemde 01000001 olarak gösteririz.
- ASCII sadece İngilizce karakterleri kapsar. Çince harfler, emoji ve Türkçe karakterler (Ş, Ğ) ASCII standartında yoktur.

KOD	KARAKTER	KOD	KARAKTER	KOD	KARAKTER	KOD	KARAKTER	KOD	KARAKTER
00	NULL(Null character)	51	3	103	g	156	£	208	°
01	SOH(Start of Header)	52	4	104	h	157	Ø	209	ª
02	STX(Start of Text)	53	5	105	i	158	Ş	210	Ê
03	ETX(End of Text)	54	6	106	j	159	ş	211	Ë
04	EOT(End of Trans.)	55	7	107	k	160	á	212	È
05	ENQ(Enquiry)	56	8	108	l	161	í	213	☐
06	ACK(Acknowledgement)	57	9	109	m	162	ó	214	Í
07	BEL(Bell)	58	:	110	n	163	ú	215	Î
08	BS(Backspace)	59	;	111	o	164	ñ	216	Ï
09	HT(Horizontal Tab)	60	<	112	p	165	Ñ	217	Ɔ
10	LF(Line feed)	61	=	113	q	166	Ğ	218	Ɔ
11	VT(Vertical Tab)	62	>	114	r	167	ğ	219	■
12	FF(Form feed)	63	?	115	s	168	ı	220	■
13	CR(Carriage return)	64	@	116	t	169	•	221	ı
14	SO(Shift Out)	65	A	117	u	170	~	222	ı
15	SI(Shift In)	66	B	118	v	171	½	223	■
16	DLE(Data link escape)	67	C	119	w	172	¼	224	Ó
17	DC1(Device control 1)	68	D	120	x	173	ı	225	ß

Unicode ve Genişletilmiş Karakterler

- **ASCII'nin Sınırları:**

- Standart ASCII tablosu sadece **7 bit** kullanır ve toplamda **128 karakter** barındırır.
- Bu alan yalnızca İngilizce alfabesi, rakamlar ve temel noktalama işaretleri için yeterlidir.
- **Sorun:** Türkçe (Ş, Ğ, İ), Çince, Arapça gibi dillerin harfleri bu tabloya sığmaz.

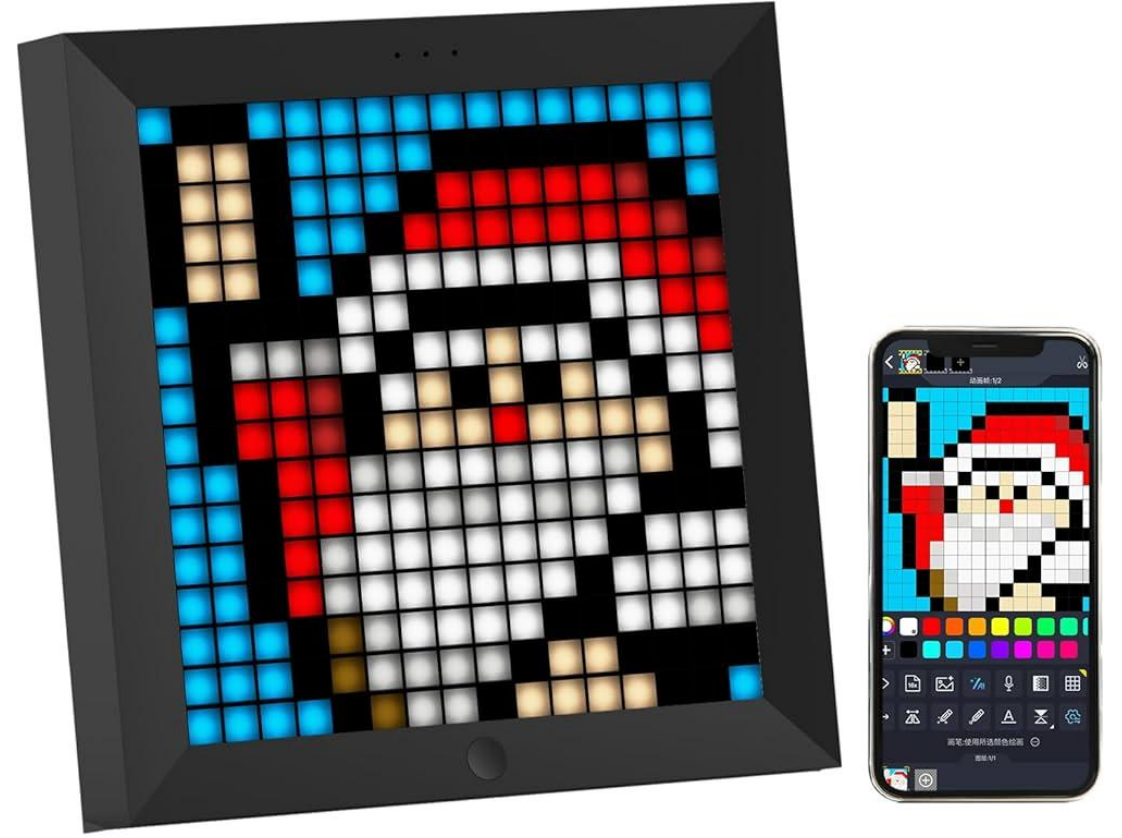
- **Çözüm: Unicode (Evrensel Kodlama):**

- Dünyadaki tüm dilleri, yazı sistemlerini ve sembolleri tek bir standartta toplamak için geliştirilmiştir.
- 8 bit yerine **16 bit veya 32 bit** kullanarak milyarlarca farklı karakteri temsil edebilir.

b)Resim Verilerinin Temsili

Resimler ve Pikseller

- Bir resim binlerce küçük noktadan (piksel) oluşur.
- Dijital görüntüler tek parça değil, **Piksel** adı verilen binlerce küçük renkli kareden oluşur.
- Her pikselin bir renk kodu vardır (Örn: Kırmızı = 255, 0, 0).
- Bilgisayar manzarayı değil, sadece bu piksellerin renk sayılarını ve sırasını saklar.



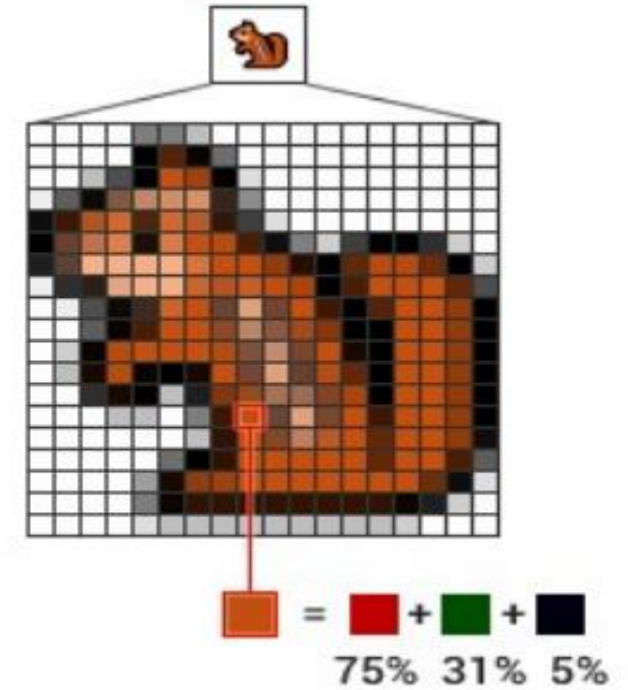
Görüntüleri Sıkıştırma

GIF= En fazla 256 renk destekler. Bu yüzden fotoğraf kalitesinde değildir ancak basit çizimler ve çizgi filmler için idealdir.

JPEG= İnsan gözünün algılayamayacağı renk detaylarını atarak (Kayıplı Sıkıştırma) dosya boyutunu küçültür. Fotoğraflar için standarttır.

PNG= Kayıpsız sıkıştırma yapar. Arka plan şeffaflığını desteklediği için logolarda ve internet sitelerinde sıkça kullanılır.

TIFF= Hiçbir veri kaybı yaşatmaz ancak dosya boyutu çok büyüktür. Matbaa ve profesyonel arşivleme için kullanılır.



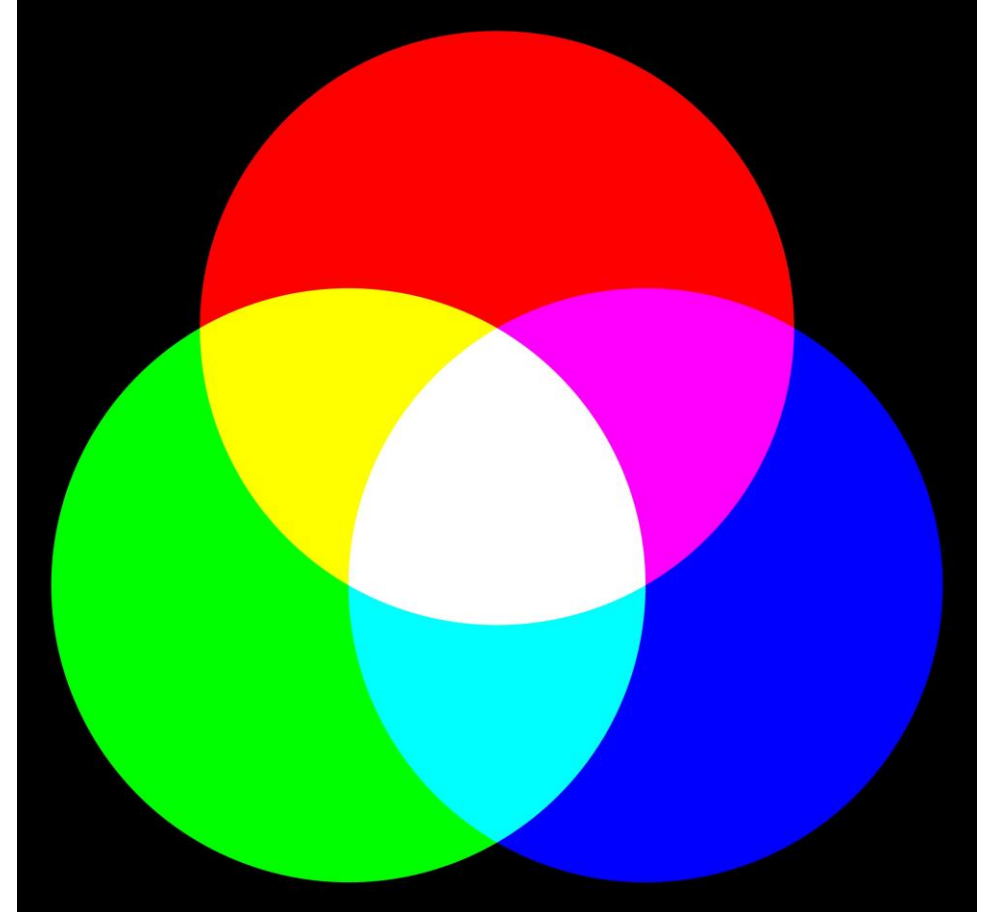
Siyah-Beyaz Resimlerin Temsili

- En basit haliyle 1 piksel siyahsa '1', beyazsa '0' olarak saklanır. Buna "Bit Map" denir.



Renkli Resimler (RGB)

- Renkli resimlerde her piksel için 3 deęer tutulur: Kırmızı, Yeşil, Mavi. Her biri 0-255 arası deęer alır.
- **Saf Kırmızı:** (255, 0, 0)
 - (Kırmızı tam güçte, dięerleri kapalı)
- **Saf Yeşil:** (0, 255, 0)
 - (Yeşil tam güçte, dięerleri kapalı)
- **Saf Mavi:** (0, 0, 255)
 - (Mavi tam güçte, dięerleri kapalı)
- **Sarı:** (255, 255, 0)
 - (Kırmızı ve Yeşil ışıık birleşince Sarı olur)
- **Beyaz:** (255, 255, 255)
 - (Tüm renkler en yüksek seviyede yanarsa Beyaz olur)
- **Siyah:** (0, 0, 0)
 - (Tüm ışıklar kapalıysa Siyah olur)



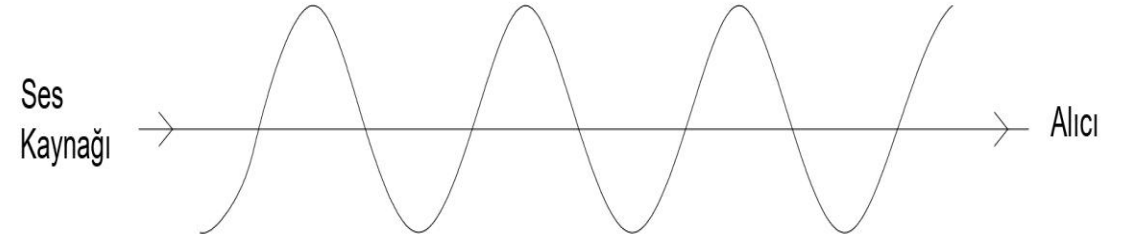
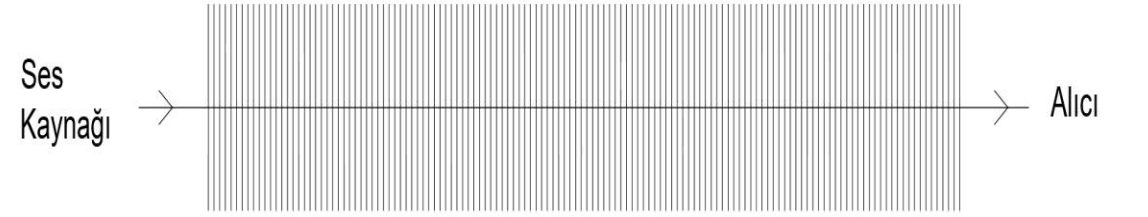
c)Ses Verilerinin Temsili

Ses Dalgaları ve Dijitalleşme

- Gerçek dünyadaki ses, kesintisiz ve sürekli bir **dalga (wave)** formundadır. Ancak bilgisayarlar "sürekli" veriyi saklayamaz, sadece kesin sayıları (0 ve 1) saklayabilir.
- Bilgisayar, bu ses dalgasını belirli aralıklarla "keser" ve o andaki ses yüksekliğini ölçer.
- Buna **Örnekleme (Sampling)** denir.
- Tıpkı bir videonun saniyede 60 kare fotoğraf çekmesi gibi, bilgisayar da saniyede binlerce kez sesin fotoğrafını çeker.
- Ölçülen her değer bir sayıya (örneğin: 128, 200, 55) dönüştürülür.
- Bu sayılar da son aşamada **Binary (0 ve 1)** formatına çevrilerek diske kaydedilir.

Örnekleme (Sampling)

- Ses dalgasının saniyede binlerce kez fotoğrafının çekilmesi gibidir.
- Bilgisayarda ses verisi şu şekilde tutulur: Kes > Ölç > 0 ve 1'e Çevir > Sakla



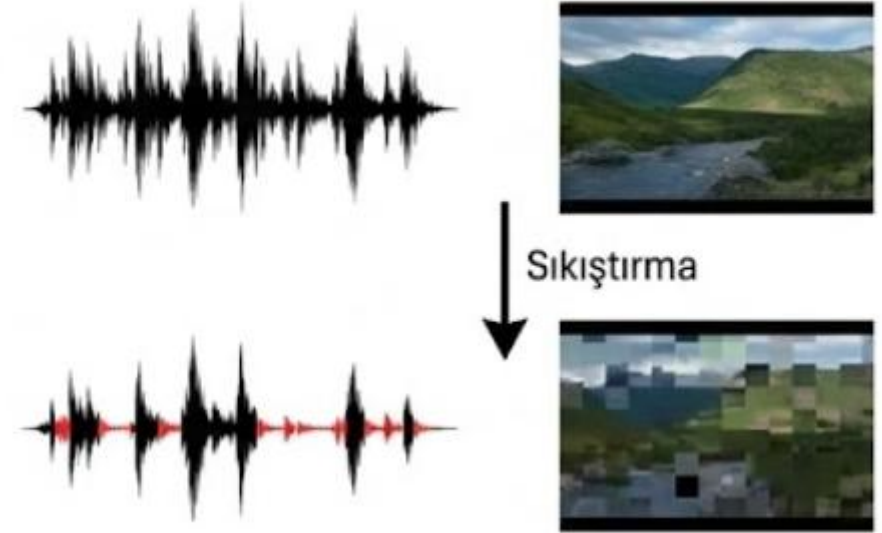
Ses ve Video Sıkıştırma

MPEG

- Hem ses hem de görüntü verilerini bir arada sıkıştırarak yüksek kalitede saklanmasını sağlar. Televizyon yayıncılığı ve film endüstrisinde standarttır. Ayrıca video konferans sistemlerinde akıcı görüntü iletimi için tercih edilir.

MP3

- Sadece ses verilerini sıkıştırmak için geliştirilmiştir. İnsan kulağının duyamayacağı veya az duyacağı ses frekanslarını atarak dosya boyutunu ciddi oranda küçültür (Kayıplı Sıkıştırma). Müzik depolama ve dijital oynatıcılar için idealdir.





2) Veri sıkıştırma neden gereklidir?

- **Veri Sıkıştırma (Data Compression):** Verinin boyutunu (bit sayısını) azaltarak, orijinal veriyi daha az yer kaplayacak şekilde kodlama işlemidir.
- Amaç: Bilgiyi kaybetmeden veya kabul edilebilir kayıplarla veriyi küçültmektir.

Neden Sıkıştırma?

Dijital Obezite (Big Data):

- Dünya çapında her gün kentilyonlarca bayt veri üretiliyor.
- Sosyal medya, sensörler ve dijital iletişim sayesinde veri üretimi her yıl katlanarak artıyor.

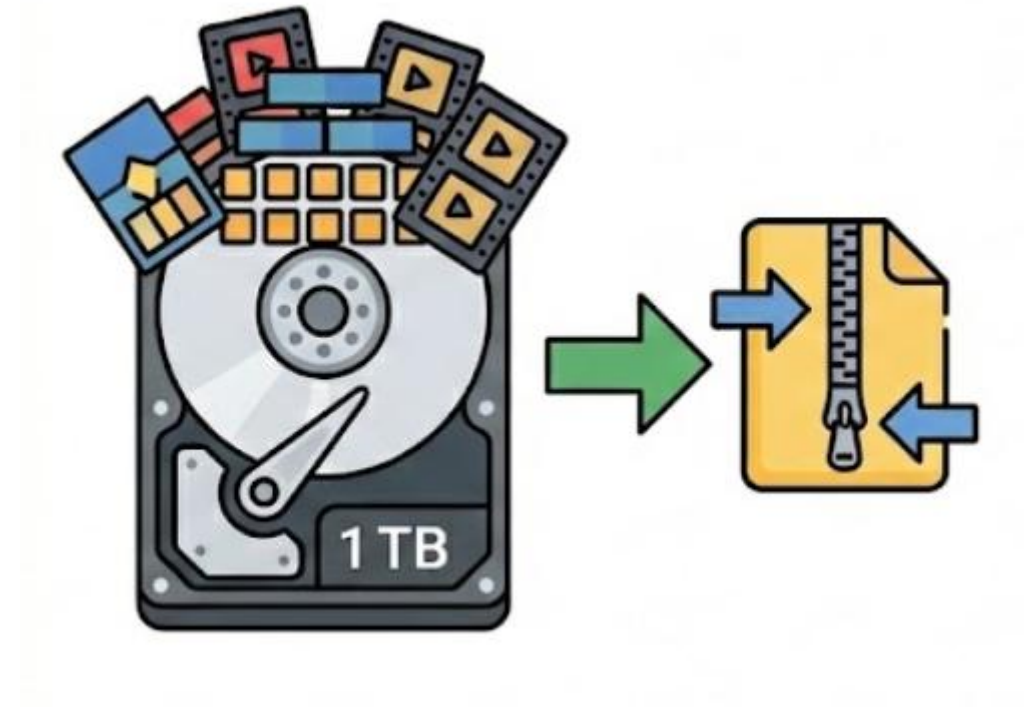
Kalite Arttıkça Boyut Artar:

- Eskiden düşük çözünürlüklü metinler varken, şimdi 4K/8K videolar, RAW formatlı fotoğraflar ve 100 GB'ı aşan oyunlar standart hale geldi.
- *Örnek:* Sıkıştırılmamış 1 dakikalık 4K video yaklaşık 40 GB yer kaplayabilir!

Neden Sıkıştırma? (devamı)

Fiziksel Sınırlar:

- Depolama teknolojileri (SSD/HDD) gelişiyor ama veri üretim hızına yetişemiyor.
- Sıkıştırma olmasaydı, 1 TB'lık bir hard diske sadece birkaç film sığdırabilirdik.



Neden Sıkıştırma? (devamı)

İletim Hızı ve Bant Genişliği

Bant Genişliği (Bandwidth) Sınırı:

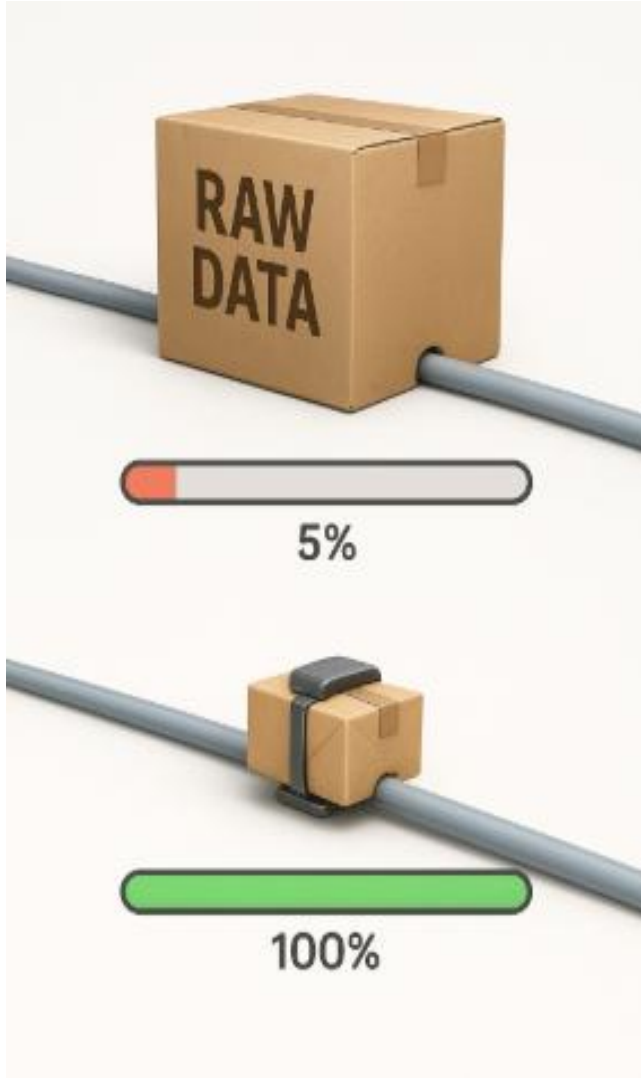
- İnternet altyapısını bir su borusu gibi düşünelim. Bu borunun genişliği sınırlıdır.
- Veriyi sıkıştırmadan göndermek, devasa bir kayayı dar bir borudan geçirmeye çalışmak gibidir; sistem tıkanır.

Streaming (Akış) Teknolojileri:

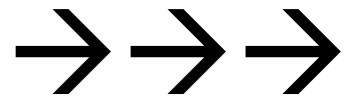
- Netflix, YouTube veya Zoom kullanırken veriler anlık olarak iletilir.
- Eğer sıkıştırma algoritmaları olmasaydı, 1 saniyelik ham videoyu indirmek için dakikalarca beklememiz gerekirdi.

Sonuç:

- Sıkıştırma, veriyi "paketleyip küçülterek" ağ trafiğini rahatlatır.
- Bu sayede filmleri donmadan izleyebilir, web sitelerini saniyeler içinde açabiliriz.



→ Veri sıkıştırma olmasaydı verilerin yüklenmesi saatlerimizi alırdı.



→ Sıkıştırılmış bir veriyi çok daha kısa bir sürede yükleyebiliriz.

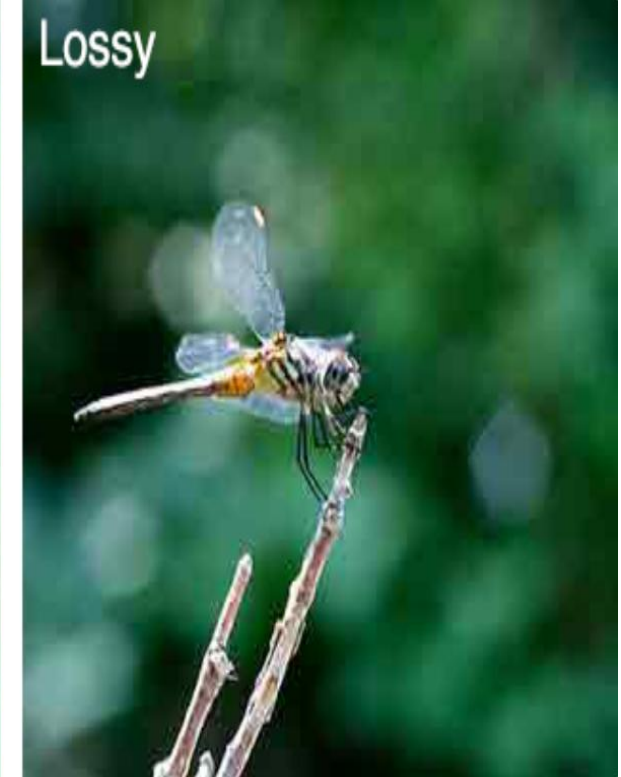
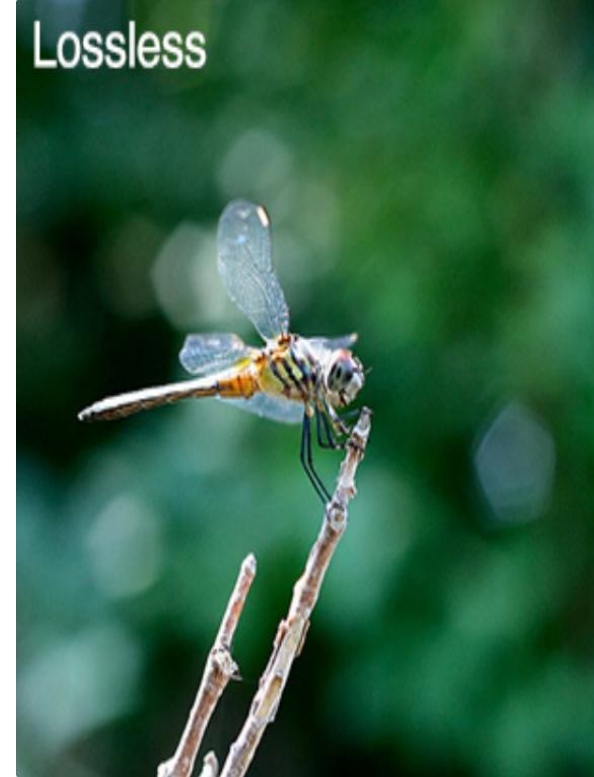
SIKIŖTIRMA YÖNTEMLERİ

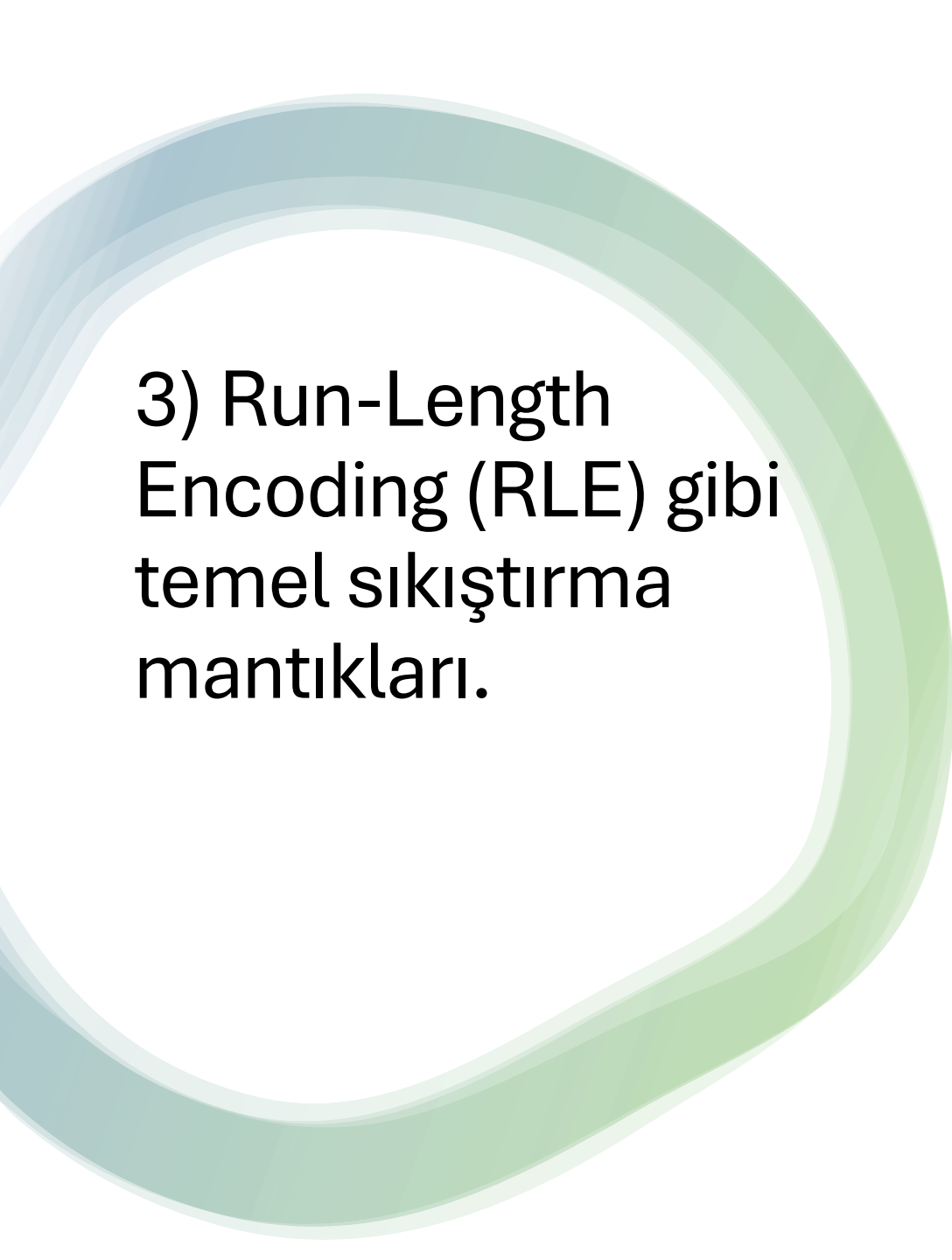
a)Kayıplı (Lossy) SıkıŖtırma:

Verinin bazı önemsiz kısımları atılır. Geri dönüşü tam olmaz ama çok yer kazanılır. (Örn: MP3, JPEG).

b)Kayıpsız (Lossless) SıkıŖtırma:

Veri bit bit, hatasız bir şekilde orijinal haline döndürülebilir. Metinler ve program kodları için zorunludur. (Örn: ZIP, PNG ve bizim projemiz RLE) .





3) Run-Length
Encoding (RLE) gibi
temel sıkıştırma
mantıkları.

a)Run-length encoding

b)Relative Encoding

c)Huffman Code Data Compression

d) Dictionary-based encoding

a) İşlem Uzunluğu Kodlama (Run-Length Encoding)

- RLE (**R**un-**L**ength **E**ncoding), kayıpsız bir veri sıkıştırma yöntemidir.
- Veri içinde arka arkaya tekrarlayan karakterleri veya pikselleri tespit eder ve bunları tek tek yazmak yerine "Sayı + Değer" formatında saklar.

Çalışma Mantığı

Ham Veri: AAAAABBBBCCDAA
(13 Karakter)

RLE Sıkıştırılmış: 5A3B2C1D2A
(10 Karakter)

Sonuç: Veri kaybı olmadan boyut küçültülmüştür.

a) İşlem Uzunluğu Kodlama (Run-Length Encoding) (devamı)

- **Nerelerde Kullanılır?**

Siyah-Beyaz Belge Taramaları:

Sayfada büyük boşluklar (beyaz) olduğu için çok etkilidir.

Basit Grafikler ve İkonlar: Çok fazla renk geçişi olmayan, düz renkli görseller (.BMP, .PCX formatları).

- **Not:** Fotoğraflar gibi çok renkli ve karmaşık görsellerde RLE verimsizdir.

a	a	a	b	c	c	c	c
---	---	---	---	---	---	---	---



run-length encoding



a	3	b	1	c	4
---	---	---	---	---	---

b) Göreceli Kodlama (Relative Encoding)

- Verinin kendisini saklamak yerine, bir önceki veri ile arasındaki farkı saklarız.
- Çoğu zaman farklar, sayıların kendisinden çok daha küçüktür (daha az bit kaplar).
- Genellikle videolarda (hareketli resimlerde) kullanılır.

Çalışma Örneği

1. Kare (Frame 1): 1 2 3 4

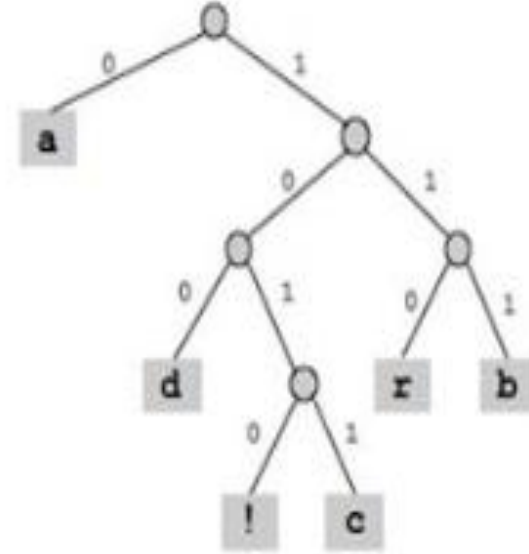
2. Kare (Frame 2): 1 3 3 4
(Sadece 2. rakam değişmiş)

Fark (Difference): 0 1 0 0

- **Açıklama:** "İkinci kareyi tamamen saklamak yerine, sadece değişen 1 değerini saklarız."

c) Frekans Bağımlı Kodlama (Huffman Coding)

- En çok kullanılan karakterlere kısa kod, az kullanılanlara uzun kod verilir.
- Her harf 8 bittir (ASCII). Huffman'da sık geçen 'a' harfi 1 bit olabilir.
- Normalde 'a' için 8 bit harcayacakken, burada sadece 1 bit harcıyoruz.
- Örneğin bir metinde a harfi çok geçeceği için bu harfe 0, c harfi daha az geçeceği için 1011 verilir.



char	encoding
a	0
b	111
c	1011
d	100
r	110
!	1010

d)Sözlük Kodlama (Dictionary-Based Encoding)

Dictionary-Based Encoding Nedir?

- Bu yöntem, veri içindeki tekrarlayan kelimeleri veya heceleri (desenleri) tespit eder.
- Tespit edilen her desen için bir "Sözlük" (Dictionary) oluşturulur ve o desene kısa bir numara (Index ID) atanır.
- Metin içinde uzun bir kelime (Örn: "Mühendislik") tekrar geçtiğinde, kelimenin tamamı tekrar yazılmaz.
- Bunun yerine, o kelimenin sözlükteki sıra numarası (Örn: #1) yazılır.
- **Sonuç:** 11 harfli "Mühendislik" kelimesi, hafızada sadece 1 karakterlik yer kaplar.

Dictionary-based encoding example

- Dictionary:

1. ASK
2. NOT
3. WHAT
4. YOUR
5. COUNTRY
6. CAN
7. DO
8. FOR
9. YOU

- Original text:

- ASK NOT WHAT YOUR COUNTRY CAN
DO FOR YOU ASK WHAT YOU CAN
DO FOR YOUR COUNTRY

- Encoded based on
dictionary :

- 1 2 3 4 5 6 7 8 9 1 3 9 6 7 8
4 5

Kaynakça

- Brookshear, J. G. ve Brylow, D. (2022). *Bilgisayar Bilimine Giriş* (12. Baskıdan Çeviri). Nobel Akademik Yayıncılık.
- Brookshear, J. G. (2019). *Computer Science: An Overview 13e*. Pearson Education.