

Tugas Analisis Multimedia: Audio, Gambar, Video

Mata Kuliah: Sistem & Teknologi Multimedia

Nama: Nydia Renli Sinaga

NIM: 122140007

Deskripsi Tugas

Tugas ini bertujuan untuk memahami representasi dasar data multimedia (audio, gambar, dan video) melalui praktik langsung memuat data, visualisasi, dan ekstraksi informasi fundamental. Anda akan bekerja dengan tiga jenis media berbeda untuk menganalisis karakteristik temporal (audio), spasial (gambar), dan spatio-temporal (video).

Fokus tugas adalah pada pemahaman konsep dasar representasi multimedia dan kemampuan interpretasi hasil visualisasi, **bukan** pada manipulasi atau transformasi lanjutan data multimedia.



CATATAN PENTING: PRESENTASI ACAK & KEJUJURAN AKADEMIK

Sebagian mahasiswa akan dipilih secara ACAK untuk presentasi singkat (5-10 menit) menjelaskan kode dan interpretasi hasil mereka. Jika Anda:

- Tidak mampu menjelaskan kode yang Anda kumpulkan
- Hanya menyalin-tempel tanpa pemahaman
- Bergantung sepenuhnya pada AI tanpa memahami konsep

Maka nilai tugas Anda akan diberikan 0 (nol).

Gunakan referensi dan AI sebagai alat bantu pembelajaran, tetapi pastikan Anda memahami setiap baris kode dan dapat menjelaskan logika di baliknya.

In []:

```
# Import Library (Satu-satunya sel kode dalam template ini)
import numpy as np
import matplotlib.pyplot as plt
import librosa
import soundfile as sf
from PIL import Image
import cv2
from IPython.display import Audio, HTML, display
import os

# Set matplotlib untuk menampilkan plot inline
```

```
%matplotlib inline

# Tampilkan versi library untuk dokumentasi
print("Library versions:")
print(f"NumPy: {np.__version__}")
print(f"Matplotlib: {plt.__version__}")
print(f"Librosa: {librosa.__version__}")
print(f"OpenCV: {cv2.__version__}")

# Tambahkan import lain jika diperlukan saat mengerjakan tugas
```

Petunjuk Umum Pengerjaan

Cara Menggunakan Template

- Gunakan notebook ini sebagai kerangka kerja utama
- Tulis penjelasan (markdown) **SEBELUM** menaruh kode agar maksud dan tujuan jelas
- Tambahkan sel kode di tempat yang sudah disediakan (tandai dengan TODO)
- Semua plot/gambar harus diberi judul, label sumbu, dan keterangan singkat

Standar Visualisasi

- Setiap plot harus memiliki judul yang deskriptif
- Label sumbu X dan Y harus jelas
- Gunakan colorbar untuk plot yang memerlukan skala warna
- Berikan interpretasi singkat setelah setiap visualisasi

Struktur Data yang Direkomendasikan

- Buat folder `data/` di direktori yang sama dengan notebook
- Gunakan nama file yang deskriptif (contoh: `audio_musik_piano.wav`, `gambar_pemandangan_gunung.jpg`)
- Dokumentasikan sumber data jika menggunakan dataset publik

Larangan

- **Jangan** menaruh seluruh pekerjaan dalam satu sel kode yang sangat panjang
- **Jangan** menempel hasil output tanpa interpretasi atau analisis
- **Jangan** bergantung sepenuhnya pada AI - pahami dan kuasai kode Anda

Persiapan Presentasi Acak

- Pastikan Anda memahami setiap baris kode yang ditulis
- Latih menjelaskan logika dan alur pemikiran Anda
- Siapkan penjelasan untuk setiap visualisasi dan interpretasinya

Checklist Kelengkapan (Centang saat selesai)

Bagian Audio

- Muat audio dan tampilkan metadata (durasi, sample rate, jumlah kanal)
- Tampilkan waveform dengan label sumbu yang jelas
- Tampilkan spectrogram dalam skala log-dB dengan colorbar
- Tampilkan MFCC (minimal 13 koefisien) sebagai heatmap
- Berikan interpretasi dan analisis untuk setiap visualisasi audio

Bagian Gambar

- Tampilkan gambar dengan benar dalam format RGB
- Tampilkan informasi dasar (dimensi, jumlah kanal, dtype)
- Tampilkan histogram warna untuk channel R, G, B
- Berikan analisis hubungan histogram dengan kesan visual gambar

Bagian Video

- Tampilkan metadata video (resolusi, fps, frame count, durasi)
- Tampilkan 3 frame representatif (awal, tengah, akhir)
- Konversi BGR ke RGB dengan benar untuk visualisasi
- Analisis kesesuaian parameter video dengan use case

Analisis & Dokumentasi

- Setiap bagian memiliki interpretasi dan analisis ringkas
- Perbandingan representasi ketiga jenis media
- Kesimpulan pembelajaran dan refleksi
- Semua sumber data dan referensi dicantumkan

Pendahuluan

Apa itu Data Multimedia?

Data multimedia adalah informasi yang dikodekan dalam berbagai format untuk merepresentasikan dunia nyata:

- **Audio (1D):** Sinyal satu dimensi yang berubah terhadap waktu
 - Contoh: musik, suara, speech
 - Representasi: amplitudo vs waktu
- **Gambar (2D):** Matriks nilai intensitas dalam ruang dua dimensi
 - Contoh: foto, ilustrasi, grafik
 - Representasi: intensitas pixel pada koordinat (x,y)
- **Video (2D + Waktu):** Rangkaian frame (gambar) yang ditampilkan berurutan
 - Contoh: film, rekaman, animasi

- Representasi: frame berubah terhadap waktu dengan frame rate tertentu

Tujuan Tugas

Memahami representasi dasar dan teknik visualisasi fundamental untuk setiap jenis media multimedia, termasuk:

- Cara memuat dan membaca file multimedia
- Ekstraksi informasi metadata yang penting
- Visualisasi yang informatif dan mudah dipahami
- Interpretasi hasil analisis secara kontekstual

Cara Kerja

1. Isi setiap bagian sesuai instruksi yang diberikan
2. Tambahkan sel kode di tempat yang ditandai dengan "TODO"
3. Berikan interpretasi dan analisis setelah setiap visualisasi
4. Pastikan semua plot memiliki judul, label, dan keterangan yang jelas

Bagian A — Audio

A1. Deskripsi Data

TODO: Jelaskan audio yang akan Anda analisis:

- Jenis audio: Musik (musik, pidato, suara alam, dll.)
- Sumber: Youtube (rekaman sendiri, dataset publik, dll.)
- Format file: MP3 (WAV, MP3, dll.)
- Alasan pemilihan: karena simpel, durasinya pas, nadanya bervariasi, dan gampang dipahami buat analisis

Path file: `data/_____ .wav` (isi nama file Anda nanti di kode)

A2. TODO: Muat & Metadata

Instruksi: Tulis kode untuk memuat file audio dan menampilkan metadata dasar:

- Sample rate (Hz)
- Durasi (detik)
- Jumlah kanal (mono/stereo)

```
In [13]: # Memuat Audio dengan Sistem Fallback
PATH_AUDIO = os.path.join(os.getcwd(), 'data', 'jasty - 30 seconds on the piano')

# Variabel untuk menyimpan hasil
y = None
sr = None
```

```

source_info = ""

# Opsi 1: Coba muat file Lokal
try:
    if os.path.exists(PATH_AUDIO):
        y, sr = librosa.load(PATH_AUDIO, sr=None) # Librosa bisa baca mp3
        source_info = f"✅ Berhasil memuat: {PATH_AUDIO}"
    else:
        raise FileNotFoundError("File tidak ditemukan")
except:
    print(f"⚠️ Tidak dapat memuat {PATH_AUDIO}")

# Opsi 2: Gunakan contoh bawaan librosa
try:
    y, sr = librosa.load(librosa.ex('trumpet'), sr=None)
    source_info = "✅ Menggunakan contoh audio bawaan librosa (trumpet)"
except:
    print("⚠️ Contoh librosa tidak tersedia")

# Konversi ke mono jika stereo
if y.ndim > 1:
    y = np.mean(y, axis=0) # Rata-rata semua kanal
    source_info += " - Dikonversi ke mono"

# Pastikan tipe data float32 untuk efisiensi
y = y.astype(np.float32)

print(source_info)
print(f"📊 Shape data: {y.shape}")
print("📋 METADATA AUDIO")
print("=" * 40)
print(f"⌚ Jumlah sampel: {len(y)}")
print(f"🎵 Sample rate: {sr} Hz")
print(f"⌚ Durasi: {len(y)/sr:.2f} detik")
print(f"📋 Jumlah kanal: 'Mono (1 kanal)' if y.ndim == 1 else f'Multi-kanal ({y.ndim})'")
print(f"📊 Tipe data: {y.dtype}")
print()

# Cek apakah audio terlalu keras (clipping)
if np.abs(y).max() >= 0.99:
    print("⚠️ PERINGATAN: Audio mungkin mengalami clipping (terlalu keras)")
else:
    print("✅ Audio dalam range amplitudo yang sehat")

```

✅ Berhasil memuat: c:\Users\Asus\Downloads\multimedia exercise\data\jasty - 30 seconds on the piano [ttEI35HVpqI].mp3
 📊 Shape data: (1571840,)
 📋 METADATA AUDIO
 ======
 ⌚ Jumlah sampel: 1,571,840
 🎵 Sample rate: 44,100 Hz
 ⌚ Durasi: 35.64 detik
 📋 Jumlah kanal: Mono (1 kanal)
 📊 Tipe data: float32

✅ Audio dalam range amplitudo yang sehat

A3. TODO: Waveform

Instruksi: Plot waveform audio dengan:

- Sumbu X: waktu (detik)
- Sumbu Y: amplitudo
- Judul dan label sumbu yang jelas

Analisis yang diperlukan: Jelaskan apa yang Anda lihat dari waveform (pola amplitudo, bagian keras/pelan, dll.)

Analisis : Dibagian awal, gelombangnya kecil yang dimana musiknya masih pelan. Lalu makin ke tengah, gelombangnya makin tinggi dan rapat menandakan bagian musikya udah mulai keras, tinggi atau nadanya lebih ditekan kan lagi. Jadi dari grafik ini bisa dilihat kapan musiknya mulai, kapan rame, dan kapan musiknya jadi santai.

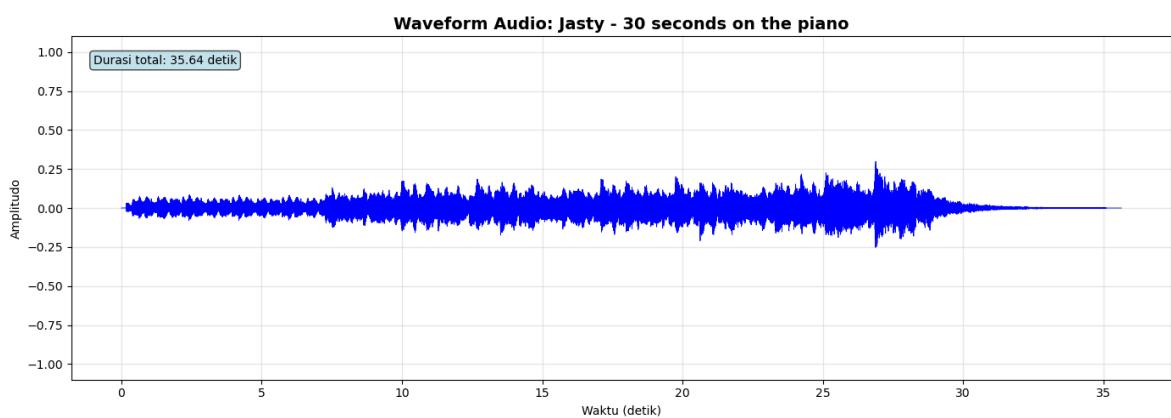
```
In [19]: # Plot Waveform Audio

# Buat vektor waktu
t = np.linspace(0, len(y)/sr, len(y))

plt.figure(figsize=(14, 5))
plt.plot(t, y, color='blue', linewidth=0.6)
plt.title('Waveform Audio: Jasty - 30 seconds on the piano', fontsize=14, fontweight='bold')
plt.xlabel('Waktu (detik)')
plt.ylabel('Amplitudo')
plt.grid(True, alpha=0.3)
plt.ylim([-1.1, 1.1])

# Tambahkan informasi durasi
duration_text = f'Durasi total: {len(y)/sr:.2f} detik'
plt.text(0.02, 0.92, duration_text, transform=plt.gca().transAxes,
        bbox=dict(boxstyle="round,pad=0.3", facecolor="lightblue", alpha=0.7))

plt.tight_layout()
plt.show()
```



A4. TODO: Spectrogram log-dB

Instruksi: Hitung STFT dan tampilkan spectrogram dalam skala log-dB:

- Gunakan parameter standar (`n_fft=1024, hop_length=256`)
- Tampilkan dengan colorbar
- Label sumbu: waktu (detik) dan frekuensi (Hz)

Analisis yang diperlukan: Jelaskan perbedaan informasi yang didapat dari spectrogram dibanding waveform.

Analisis : Waveform itu cuman nunjukkin seberapa keras atau pelan suara dari waktu ke waktu, kaya grafik naik dan turun volume aja. Kalau spectrogram jauh lebih detail, bisa lihat suara mana yang frekuensinya rendah di setiap detik nya, dan di warnanya juga nunjukkin seberapa kuat suara di frekuensi tertentu. kalau warnanya makin terang, berarti suara di frekuensinya lagi kuat.

```
In [20]: # Hitung dan Plot Spectrogram (STFT)
# Parameter STFT
n_fft = 1024          # Ukuran FFT window
hop_length = 256        # Langkah antar frame
window = 'hann'         # Jenis window function

# Hitung STFT
D = librosa.stft(y, n_fft=n_fft, hop_length=hop_length, window=window)
magnitude = np.abs(D)
magnitude_db = librosa.amplitude_to_db(magnitude, ref=np.max)

# Setup plot dengan 2 subplot
fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(16, 6))

# Plot 1: Magnitude Spectrogram (Linear)
img1 = librosa.display.specshow(magnitude, x_axis='time', y_axis='hz',
                                 sr=sr, hop_length=hop_length, ax=ax1)
ax1.set_title('📊 Spectrogram - Magnitude Linear', fontsize=14, fontweight='bold')
ax1.set_xlabel('Waktu (detik)')
ax1.set_ylabel('Frekuensi (Hz)')
plt.colorbar(img1, ax=ax1, format='%.2f')

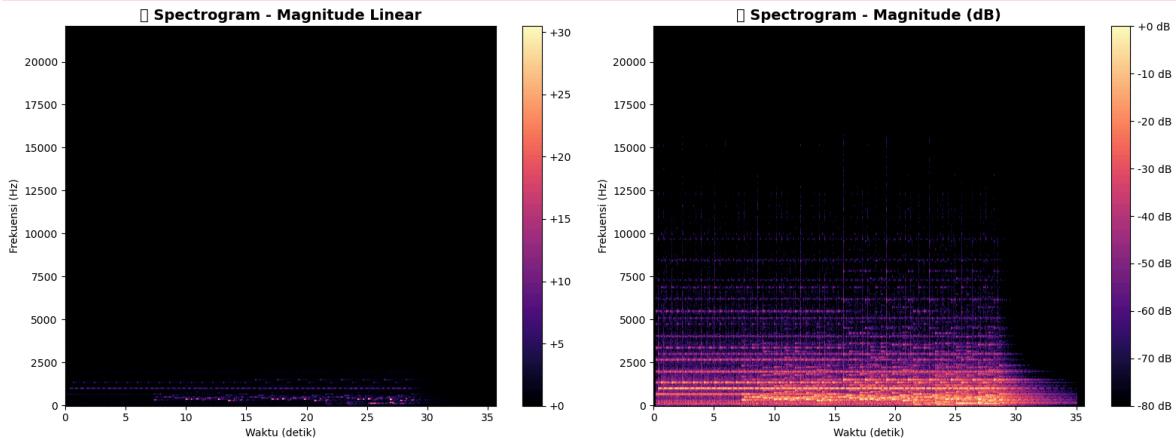
# Plot 2: Magnitude Spectrogram (Log dB)
img2 = librosa.display.specshow(magnitude_db, x_axis='time', y_axis='hz',
                                 sr=sr, hop_length=hop_length, ax=ax2)
ax2.set_title('📊 Spectrogram - Magnitude (dB)', fontsize=14, fontweight='bold')
ax2.set_xlabel('Waktu (detik)')
ax2.set_ylabel('Frekuensi (Hz)')
plt.colorbar(img2, ax=ax2, format='%.2f dB')

plt.tight_layout()
plt.show()

# Informasi STFT
print("📋 INFORMASI STFT")
print("=" * 40)
print(f"⌚ Ukuran FFT: {n_fft}")
print(f"👣 Hop length: {hop_length}")
print(f"🎛️ Window function: {window}")
print(f"📊 Shape magnitude: {magnitude.shape}")
print(f"⌚ Resolusi waktu: {hop_length/sr*1000:.1f} ms per frame")
print(f"🎵 Resolusi frekuensi: {sr/n_fft:.1f} Hz per bin")
print(f"📈 Range magnitude (dB): {magnitude_db.min():.1f} - {magnitude_db.max():.1f}
```

```
C:\Users\Asus\AppData\Local\Temp\ipykernel_18388\2400879995.py:31: UserWarning: Glyph 128202 (\N{BAR CHART}) missing from font(s) DejaVu Sans.
    plt.tight_layout()
c:\Users\Asus\miniconda3\envs\multimedia\Lib\site-packages\IPython\core\pylabtool
s.py:170: UserWarning: Glyph 128202 (\N{BAR CHART}) missing from font(s) DejaVu S
ans.
```

```
fig.canvas.print_figure(bytes_io, **kw)
```



INFORMASI STFT

- ```
=====
12 Ukuran FFT: 1024
34 Hop length: 256
Window function: hann
Shape magnitude: (513, 6141)
Resolusi waktu: 5.8 ms per frame
Resolusi frekuensi: 43.1 Hz per bin
Range magnitude (dB): -80.0 - 0.0
```

## A5. TODO: MFCC

**Instruksi:** Hitung dan tampilkan minimal 13 koefisien MFCC sebagai heatmap:

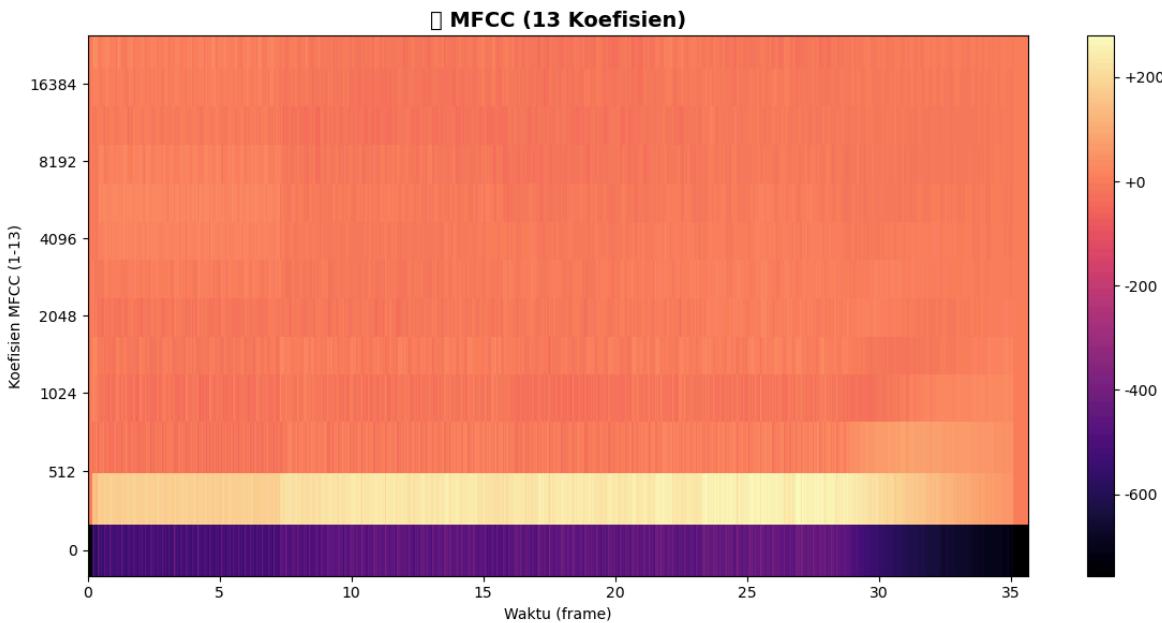
- Sumbu X: waktu (frame)
- Sumbu Y: koefisien MFCC (1-13)
- Gunakan colorbar dan judul yang jelas

**Analisis yang diperlukan:** Interpretasi sederhana: apakah pola MFCC stabil atau berubah-ubah? Apa potensi maknanya?

```
In [25]: # Hitung MFCC (13 koefisien)
n_mfcc = 13
mfcc = librosa.feature.mfcc(y=y, sr=sr, n_mfcc=n_mfcc, n_fft=n_fft,
 hop_length=hop_length)

Plot MFCC sebagai heatmap
plt.figure(figsize=(12, 6))
img = librosa.display.specshow(mfcc, x_axis='time', y_axis='mel', sr=sr, hop_len=hop_length)
plt.title('MFCC (13 Koefisien)', fontsize=14, fontweight='bold')
plt.xlabel('Waktu (frame)')
plt.ylabel('Koefisien MFCC (1-13)')
plt.colorbar(img, format='%.2f')
plt.tight_layout()
plt.show()
```

```
C:\Users\Asus\AppData\Local\Temp\ipykernel_18388\4159330936.py:13: UserWarning: Glyph 127919 (\N{DIRECT HIT}) missing from font(s) DejaVu Sans.
 plt.tight_layout()
```



```
In []: # Memuat Audio dengan Sistem Fallback
PATH_AUDIO = os.path.join(os.getcwd(), 'data', 'jasty - 30 seconds on the piano')

Variabel untuk menyimpan hasil
y = None
sr = None
source_info = ""

Opsi 1: Coba muat file lokal
try:
 if os.path.exists(PATH_AUDIO):
 y, sr = librosa.load(PATH_AUDIO, sr=None) # Librosa bisa baca mp3
 source_info = f"✅ Berhasil memuat: {PATH_AUDIO}"
 else:
 raise FileNotFoundError("File tidak ditemukan")
except:
 print(f"⚠️ Tidak dapat memuat {PATH_AUDIO}")

Opsi 2: Gunakan contoh bawaan librosa
try:
 y, sr = librosa.load(librosa.ex('trumpet'), sr=None)
 source_info = "✅ Menggunakan contoh audio bawaan librosa (trumpet)"
except:
 print("⚠️ Contoh librosa tidak tersedia")

Konversi ke mono jika stereo
if y.ndim > 1:
 y = np.mean(y, axis=0) # Rata-rata semua kanal
 source_info += " - Dikonversi ke mono"

Pastikan tipe data float32 untuk efisiensi
y = y.astype(np.float32)

print(source_info)
print(f"📊 Shape data: {y.shape}")
print("📋 METADATA AUDIO")
print("=* 40")
```

```

print(f" 34 Jumlah sampel: {len(y)}")
print(f" 34 Sample rate: {sr:.} Hz")
print(f" 34 Durasi: {len(y)/sr:.2f} detik")
print(f" 34 Jumlah kanal: 'Mono (1 kanal)' if y.ndim == 1 else f'Multi-kanal ({y.ndim} kanal)'")
print(f" 34 Tipe data: {y.dtype}")
print()

Cek apakah audio terlalu keras (clipping)
if np.abs(y).max() >= 0.99:
 print("⚠ PERINGATAN: Audio mungkin mengalami clipping (terlalu keras)")
else:
 print("✓ Audio dalam range amplitudo yang sehat")

```

Berhasil memuat: c:\Users\Asus\Downloads\multimedia exercise\data\jasty - 30 seconds on the piano [ttEI35HVpqI].mp3

Shape data: (1571840,)  
METADATA AUDIO

=====

34 Jumlah sampel: 1,571,840  
34 Sample rate: 44,100 Hz  
34 Durasi: 35.64 detik  
34 Jumlah kanal: Mono (1 kanal)  
34 Tipe data: float32

Audio dalam range amplitudo yang sehat

```

In []: # Memuat Audio dengan Sistem Fallback
PATH_AUDIO = os.path.join(os.getcwd(), 'data', 'jasty - 30 seconds on the piano')

Variabel untuk menyimpan hasil
y = None
sr = None
source_info = ""

Opsi 1: Coba muat file lokal
try:
 if os.path.exists(PATH_AUDIO):
 y, sr = librosa.load(PATH_AUDIO, sr=None) # Librosa bisa baca mp3
 source_info = f"✓ Berhasil memuat: {PATH_AUDIO}"
 else:
 raise FileNotFoundError("File tidak ditemukan")
except:
 print(f"⚠ Tidak dapat memuat {PATH_AUDIO}")

Opsi 2: Gunakan contoh bawaan Librosa
try:
 y, sr = librosa.load(librosa.ex('trumpet'), sr=None)
 source_info = "✓ Menggunakan contoh audio bawaan librosa (trumpet)"
except:
 print("⚠ Contoh librosa tidak tersedia")

Konversi ke mono jika stereo
if y.ndim > 1:
 y = np.mean(y, axis=0) # Rata-rata semua kanal
 source_info += " - Dikonversi ke mono"

Pastikan tipe data float32 untuk efisiensi
y = y.astype(np.float32)

print(source_info)

```

```

print(f"📊 Shape data: {y.shape}")
print("📋 METADATA AUDIO")
print("=" * 40)
print(f"🔢 Jumlah sampel: {len(y)}")
print(f"🎵 Sample rate: {sr:.} Hz")
print(f"⌚ Durasi: {len(y)/sr:.2f} detik")
print(f"🔊 Jumlah kanal: {'Mono (1 kanal)' if y.ndim == 1 else f'Multi-kanal ({y.ndim} kanal)'}")
print(f"📊 Tipe data: {y.dtype}")
print()

Cek apakah audio terlalu keras (clipping)
if np.abs(y).max() >= 0.99:
 print("⚠ PERINGATAN: Audio mungkin mengalami clipping (terlalu keras)")
else:
 print("✅ Audio dalam range amplitudo yang sehat")

```

✓ Berhasil memuat: c:\Users\Asus\Downloads\multimedia exercise\data\jasty - 30 seconds on the piano [ttEI35HVpqI].mp3

📊 Shape data: (1571840,)

📋 METADATA AUDIO

=====

🔢 Jumlah sampel: 1,571,840

🎵 Sample rate: 44,100 Hz

⌚ Durasi: 35.64 detik

🔊 Jumlah kanal: Mono (1 kanal)

📊 Tipe data: float32

✓ Audio dalam range amplitudo yang sehat

```

In []: # Memuat Audio dengan Sistem Fallback
PATH_AUDIO = os.path.join(os.getcwd(), 'data', 'jasty - 30 seconds on the piano')

Variabel untuk menyimpan hasil
y = None
sr = None
source_info = ""

Opsi 1: Coba muat file lokal
try:
 if os.path.exists(PATH_AUDIO):
 y, sr = librosa.load(PATH_AUDIO, sr=None) # Librosa bisa baca mp3
 source_info = f"✅ Berhasil memuat: {PATH_AUDIO}"
 else:
 raise FileNotFoundError("File tidak ditemukan")
except:
 print(f"⚠ Tidak dapat memuat {PATH_AUDIO}")

Opsi 2: Gunakan contoh bawaan Librosa
try:
 y, sr = librosa.load(librosa.ex('trumpet'), sr=None)
 source_info = "✅ Menggunakan contoh audio bawaan librosa (trumpet)"
except:
 print("⚠ Contoh librosa tidak tersedia")

Konversi ke mono jika stereo
if y.ndim > 1:
 y = np.mean(y, axis=0) # Rata-rata semua kanal
 source_info += " - Dikonversi ke mono"

Pastikan tipe data float32 untuk efisiensi

```

```

y = y.astype(np.float32)

print(source_info)
print(f"📊 Shape data: {y.shape}")
print("📋 METADATA AUDIO")
print("=" * 40)
print(f"🔢 Jumlah sampel: {len(y)}")
print(f"🎵 Sample rate: {sr} Hz")
print(f"⌚ Durasi: {len(y)/sr:.2f} detik")
print(f"💻 Jumlah kanal: 'Mono (1 kanal)' if y.ndim == 1 else f'Multi-kanal ({y.ndim} kanal)'")
print(f"📊 Tipe data: {y.dtype}")
print()

Cek apakah audio terlalu keras (clipping)
if np.abs(y).max() >= 0.99:
 print("⚠ PERINGATAN: Audio mungkin mengalami clipping (terlalu keras)")
else:
 print("✅ Audio dalam range amplitudo yang sehat")

```

✓ Berhasil memuat: c:\Users\Asus\Downloads\multimedia exercise\data\jasty - 30 seconds on the piano [ttEI35HVpqI].mp3

📊 Shape data: (1571840,)

📋 METADATA AUDIO

=====

🔢 Jumlah sampel: 1,571,840

🎵 Sample rate: 44,100 Hz

⌚ Durasi: 35.64 detik

💻 Jumlah kanal: Mono (1 kanal)

📊 Tipe data: float32

✓ Audio dalam range amplitudo yang sehat

```

In []: # Memuat Audio dengan Sistem Fallback
PATH_AUDIO = os.path.join(os.getcwd(), 'data', 'jasty - 30 seconds on the piano')

Variabel untuk menyimpan hasil
y = None
sr = None
source_info = ""

Opsi 1: Coba muat file lokal
try:
 if os.path.exists(PATH_AUDIO):
 y, sr = librosa.load(PATH_AUDIO, sr=None) # Librosa bisa baca mp3
 source_info = f"✅ Berhasil memuat: {PATH_AUDIO}"
 else:
 raise FileNotFoundError("File tidak ditemukan")
except:
 print(f"⚠ Tidak dapat memuat {PATH_AUDIO}")

Opsi 2: Gunakan contoh bawaan librosa
try:
 y, sr = librosa.load(librosa.ex('trumpet'), sr=None)
 source_info = "✅ Menggunakan contoh audio bawaan librosa (trumpet)"
except:
 print("⚠ Contoh librosa tidak tersedia")

Konversi ke mono jika stereo
if y.ndim > 1:
 y = np.mean(y, axis=0) # Rata-rata semua kanal

```

```

source_info += " - Dikonversi ke mono"

Pastikan tipe data float32 untuk efisiensi
y = y.astype(np.float32)

print(source_info)
print(f"📊 Shape data: {y.shape}")
print("📋 METADATA AUDIO")
print("=" * 40)
print(f"⌚ Jumlah sampel: {len(y)}")
print(f"🎵 Sample rate: {sr} Hz")
print(f"⌚ Durasi: {len(y)/sr:.2f} detik")
print(f"🎧 Jumlah kanal: 'Mono (1 kanal)' if y.ndim == 1 else f'Multi-kanal ({y.ndim} kanal)'")
print(f"📊 Tipe data: {y.dtype}")
print()

Cek apakah audio terlalu keras (clipping)
if np.abs(y).max() >= 0.99:
 print("⚠️ PERINGATAN: Audio mungkin mengalami clipping (terlalu keras)")
else:
 print("✅ Audio dalam range amplitudo yang sehat")

```

Berhasil memuat: c:\Users\Asus\Downloads\multimedia exercise\data\jasty - 30 seconds on the piano [ttEI35HVpqI].mp3  
 📊 Shape data: (1571840,)  
 📄 METADATA AUDIO  
 ======  
 ⌚ Jumlah sampel: 1,571,840  
 🎵 Sample rate: 44,100 Hz  
 ⌚ Durasi: 35.64 detik  
 🎧 Jumlah kanal: Mono (1 kanal)  
 📊 Tipe data: float32

Audio dalam range amplitudo yang sehat

## A6. Analisis Ringkas (Wajib)

Jawab pertanyaan berikut:

1. **Perbedaan insight:** Apa perbedaan informasi yang didapat dari waveform versus spectrogram?

*Jawaban Anda:* Waveform itu cuma nunjukin naik-turunnya volume dari waktu ke waktu, jadi kita cuma bisa lihat bagian mana musiknya pelan atau rame. Kalau spectrogram, kita bisa lihat suara mana yang frekuensinya tinggi atau rendah di setiap detik, plus seberapa kuat suara di frekuensi tertentu lewat warna. Jadi, spectrogram jauh lebih detail buat tahu isi musik, bukan cuma keras-pelannya aja.

2. **Pembelajaran dari MFCC:** Apa yang Anda pelajari dari visualisasi MFCC audio ini?

*Jawaban Anda:* Dari visualisasi MFCC, aku bisa lihat pola-pola spektral yang berubah-ubah sepanjang lagu. Kalau pola MFCC-nya stabil, berarti nadanya nggak banyak berubah, tapi kalau naik-turun atau warnanya bervariasi, berarti ada perubahan nada atau karakter suara. MFCC ini penting banget buat analisis suara lebih lanjut, misal buat klasifikasi genre musik atau deteksi ucapan, karena dia ngambil "ciri khas" suara yang gampang diproses sama komputer.

# Bagian B — Gambar

## B1. Deskripsi Data

**TODO:** Jelaskan gambar yang akan Anda analisis:

- Jenis gambar: Pemandangan (foto, ilustrasi, pemandangan, dll.)
- Sumber: Foto Sendiri (foto sendiri, dataset publik, dll.)
- Format file: JPG (JPG, PNG, BMP, dll.)
- Alasan pemilihan: Karena saya kemarin ke pantai dan pantainya sangat indah jadi saya memutuskan untuk menjadikannya pilihan untuk tugas ini

**Path file:** `data/_____ .jpg` (isi nama file Anda nanti di kode)

---

## B2. TODO: Baca & Tampilkan (RGB)

**Instruksi:** Baca gambar dan tampilkan dengan benar dalam format RGB:

- Pastikan konversi warna benar (ingat perbedaan BGR vs RGB di OpenCV)
- Berikan judul yang deskriptif
- Hilangkan axis untuk tampilan yang bersih

**Analisis yang diperlukan:** Jelaskan gambar secara ringkas (objek dominan, kondisi pencahayaan, komposisi warna).

### Analisis

Gambar ini menunjukkan suasana pantai yang adem. Dibagian kiri ada setengah pohon, terus dibawahnya ada batu-batu/karang yang kena ombak. Lautnya biru, ombaknya putih di campur dengan langit yang cerah bercampur kuning menunggu matahari mau tenggelam. kompisisi warna yang indah : biru, hijau, coklat, putih. pencahayaannya juga pas, ga terlalu terang maupun gelap jadi vibesnya tenang dan sangat cocok buat healing semester akhir.

```
In [43]: # Import semua library yang dibutuhkan
import matplotlib.pyplot as plt
import matplotlib.image as mpimg
from PIL import Image
import cv2
import numpy as np
import os

Load Gambar dengan Sistem Fallback
PATH_IMAGE = os.path.join(os.getcwd(), 'data', 'gambar pantai.jpg') # Ganti dengan nama file gambar yang anda miliki

Variabel untuk menyimpan hasil
img_pil = None
source_info = ""
```

```

Opsi 1: Coba muat file lokal dengan PIL
try:
 if os.path.exists(PATH_IMAGE):
 img_pil = Image.open(PATH_IMAGE)
 # Pastikan dalam mode RGB (bukan RGBA atau mode lain)
 if img_pil.mode != 'RGB':
 img_pil = img_pil.convert('RGB')
 source_info = f"✅ Berhasil memuat: {PATH_IMAGE}"
 else:
 raise FileNotFoundError("File tidak ditemukan")
except Exception as e:
 print(f"⚠️ Tidak dapat memuat {PATH_IMAGE}: {e}")

Opsi 2: Gunakan gambar contoh dari matplotlib
try:
 # Membuat gambar contoh sederhana (gradient warna)
 height, width = 200, 300
 img_array = np.zeros((height, width, 3), dtype=np.uint8)

 # Buat gradient horizontal RGB
 for i in range(width):
 img_array[:, i, 0] = int(255 * i / width) # Red channel
 img_array[:, i, 1] = int(255 * (1 - i / width)) # Green channel
 img_array[:, i, 2] = 128 # Blue channel (kons

 img_pil = Image.fromarray(img_array)
 source_info = "✅ Menggunakan gambar contoh buatan (gradient RGB)"

except Exception as e:
 print(f"❌ Error membuat gambar contoh: {e}")
 # Buat gambar solid sederhana sebagai fallback terakhir
 img_array = np.full((100, 100, 3), [255, 128, 0], dtype=np.uint8) # Orange
 img_pil = Image.fromarray(img_array)
 source_info = "✅ Menggunakan gambar fallback (solid orange)"

print(source_info)
print(f"📊 Ukuran gambar: {img_pil.size} (lebar x tinggi)")
print(f"🎨 Mode warna: {img_pil.mode}")
print(f"📁 Format: {img_pil.format if img_pil.format else 'Generated'}")

```

✅ Berhasil memuat: c:\Users\Asus\Downloads\multimedia exercise\data\gambar pantai.jpg  
 📊 Ukuran gambar: (3024, 4032) (lebar x tinggi)  
 🎨 Mode warna: RGB  
 📁 Format: JPEG

## B3. TODO: Informasi Dasar

**Instruksi:** Tampilkan informasi metadata gambar:

- Dimensi (Height × Width)
- Jumlah kanal
- Tipe data (dtype)
- Mode warna (jika relevan)
- Ukuran file dalam memori

**Analisis yang diperlukan:** Jelaskan mengapa informasi ini penting untuk tahap preprocessing atau analisis lanjutan.

**Analisis** Informasi kayak dimensi gambar, jumlah kanal, tipe data, mode warna, sama ukuran file itu penting banget sebelum ngolah gambar. Misal, dimensi dan kanal bikin tahu gambar ini portrait atau landscape, warnanya full color atau hitam-putih. Tipe data dan mode warna juga nentuin cara proses gambar selanjutnya, biar nggak salah pas mau edit atau analisis. Ukuran file di memori juga ngaruh ke kecepatan loading dan proses, apalagi kalau gambar gede kayak foto pantai aku ini. Jadi, sebelum lanjut ke tahap analisis atau edit, info basic ini wajib dicek dulu biar nggak ada error di tengah jalan.

In [47]:

```
Visualisasi Gambar dengan Matplotlib
plt.figure(figsize=(8, 10))

Tampilkan gambar
plt.imshow(img_pil)
plt.title('🖼️ Gambar yang Dimuat', fontsize=16, fontweight='bold', pad=20)

Hilangkan axis untuk tampilan yang lebih bersih
plt.axis('off')

Tambahkan informasi gambar sebagai text
info_text = f"Ukuran: {img_pil.size[0]} x {img_pil.size[1]} pixels\nMode: {img_pil.mode}"
plt.figtext(0.5, 0.01, info_text, fontsize=11,
 bbox=dict(boxstyle="round,pad=0.4", facecolor="lightblue", alpha=0.7))

Tampilkan dengan layout yang rapi
plt.tight_layout()
plt.show()

print("✅ Gambar berhasil ditampilkan!")
print(f"📐 Resolusi: {img_pil.size[0]} x {img_pil.size[1]} pixels")
print(f"🎨 Total pixel: {img_pil.size[0]} * {img_pil.size[1]}")
```

C:\Users\Asus\AppData\Local\Temp\ipykernel\_18388\322672469.py:17: UserWarning: Glyph 128444 (\N{FRAME WITH PICTURE}) missing from font(s) DejaVu Sans.  
plt.tight\_layout()

## □ Gambar yang Dimuat



- ✓ Gambar berhasil ditampilkan!
- 📐 Resolusi: 3024 x 4032 pixels
- 🎨 Total pixel: 12,192,768

## B4. TODO: Histogram Warna

**Instruksi:** Tampilkan histogram distribusi intensitas untuk channel R, G, B:

- Range: 0-255
- Plot terpisah atau overlay dengan warna sesuai channel

- Label sumbu: intensitas pixel dan frekuensi
- Legend yang jelas

**Analisis yang diperlukan:** Analisis: channel mana yang dominan? Bagaimana kontras gambar? Seperti apa sebaran intensitasnya?

**Analisis** Dari histogramnya, kelihatan banget kalau channel biru (Blue) paling menonjol. Dia punya puncak tinggi di sekitar intensitas 200, artinya banyak banget piksel di gambar yang warnanya biru terang. Ini masuk akal kalau gambar kamu adalah pantai—langit dan laut biasanya biru cerah, kan? Channel merah (Red) juga aktif, tapi lebih menyebar. Ada dua puncak: satu di intensitas rendah (sekitar 50) dan satu lagi mendekati 200. Ini bisa jadi warna pasir, kulit, atau objek lain yang warnanya hangat. Channel hijau (Green) ada di tengah-tengah. Distribusinya nggak terlalu ekstrem, tapi tetap berkontribusi. Bisa jadi dari vegetasi atau pantulan cahaya alami.

In [68]:

```
import numpy as np

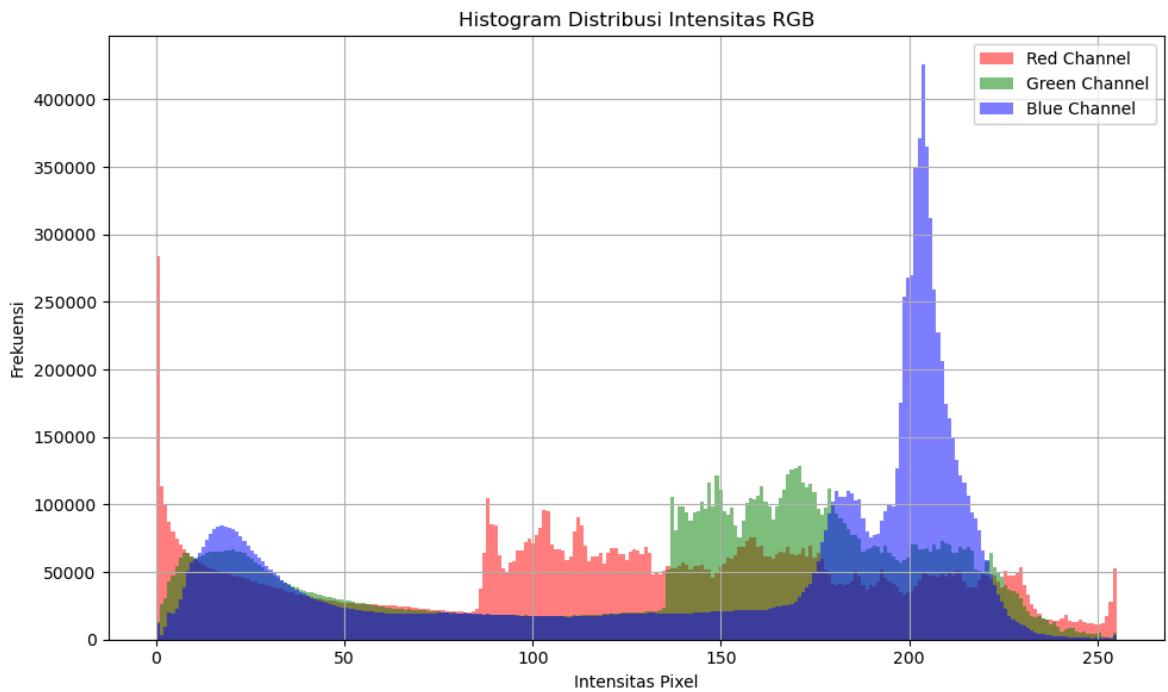
Konversi gambar ke RGB jika belum
if img_pil.mode != 'RGB':
 img_pil = img_pil.convert('RGB')

Konversi ke array NumPy
img_np = np.array(img_pil)

Ekstrak channel R, G, B
r_channel = img_np[:, :, 0].flatten()
g_channel = img_np[:, :, 1].flatten()
b_channel = img_np[:, :, 2].flatten()

Plot histogram overlay
plt.figure(figsize=(10, 6))
plt.hist(r_channel, bins=256, range=(0, 255), color='red', alpha=0.5, label='Red')
plt.hist(g_channel, bins=256, range=(0, 255), color='green', alpha=0.5, label='G')
plt.hist(b_channel, bins=256, range=(0, 255), color='blue', alpha=0.5, label='Bl')

Label dan Legend
plt.title('Histogram Distribusi Intensitas RGB')
plt.xlabel('Intensitas Pixel')
plt.ylabel('Frekuensi')
plt.legend(loc='upper right')
plt.grid(True)
plt.tight_layout()
plt.show()
```



## Bagian C — Video

### C1. Deskripsi Data

**TODO:** Jelaskan video yang akan Anda analisis:

- Jenis video: \_\_\_\_\_ (aktivitas, pemandangan, tutorial, dll.)
- Sumber: \_\_\_\_\_ (rekaman sendiri, dataset publik, dll.)
- Durasi target: \_\_\_\_\_ (disarankan  $\leq$  30 detik untuk efisiensi)
- Alasan pemilihan: \_\_\_\_\_

**Path file:** data/\_\_\_\_\_.mp4 (isi nama file Anda nanti di kode)

---

### C2. TODO: Baca & Metadata

**Instruksi:** Baca video dengan OpenCV dan tampilkan metadata:

- Resolusi (Width  $\times$  Height)
- Frame rate (fps)
- Jumlah total frame
- Durasi (detik)
- Klasifikasi resolusi (HD, Full HD, 4K, dll.)

**Analisis yang diperlukan:** Jelaskan pentingnya parameter-parameter tersebut untuk analisis video atau aplikasi tertentu.

#### Analisis

- Resolusi nunjukin seberapa tajam gambarnya. Makin tinggi, makin detail. Cocok buat analisis visual.

- Frame rate (fps) ngaruh ke gerakan. Kalau fps tinggi, videonya halus. Penting buat pelacakan atau slow-mo.
- Jumlah frame bantu tahu panjang video dan posisi frame tertentu.
- Durasi penting buat sinkronisasi, misalnya kalau gabungin video sama audio atau sensor.
- Klasifikasi resolusi kasih gambaran cepat soal kualitas video—HD, Full HD, 4K, dll

In [70]:

```
Metadata Video - Ekstraksi Informasi Lengkap
if video_loaded:
 print("📝 METADATA VIDEO")
 print("=" * 50)

 # Ambil informasi dasar dari video
 width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
 height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
 fps = cap.get(cv2.CAP_PROP_FPS)
 frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))

 # Hitung durasi (detik)
 duration = frame_count / fps if fps > 0 else 0

 print("📐 RESOLUSI & DIMENSI:")
 print(f"🖼️ Lebar: {width} pixels")
 print(f"🖼️ Tinggi: {height} pixels")
 print(f"📊 Resolusi: {width} x {height}")
 print(f"💡 Aspect ratio: {width/height:.2f}")

 # Klasifikasi resolusi
 if height >= 2160:
 res_class = "4K Ultra HD"
 elif height >= 1080:
 res_class = "Full HD (1080p)"
 elif height >= 720:
 res_class = "HD (720p)"
 else:
 res_class = "Standard Definition"
 print(f"🏷️ Klasifikasi: {res_class}")

 print("\n⌚ INFORMASI TEMPORAL:")
 print(f"🎬 Frame rate: {fps:.2f} fps")
 print(f"🕒 Total frame: {frame_count:,}")
 print(f"⌚ Durasi: {duration:.2f} detik ({duration/60:.1f} menit)")

 # Klasifikasi frame rate
 if fps >= 60:
 fps_class = "High frame rate (smooth motion)"
 elif fps >= 30:
 fps_class = "Standard frame rate"
 elif fps >= 24:
 fps_class = "Cinema frame rate"
 else:
 fps_class = "Low frame rate"
 print(f"🏷️ Klasifikasi FPS: {fps_class}")

 print("\n💾 ESTIMASI DATA:")
 # Estimasi ukuran data (asumsi 3 bytes per pixel untuk RGB)
 bytes_per_frame = width * height * 3
 total_bytes = bytes_per_frame * frame_count
```

```

print(f"Bytes per frame: {bytes_per_frame:,} ({bytes_per_frame/1024/1024:.2f} GB)")
print(f"📦 Total estimasi (RGB): {total_bytes:,} bytes ({total_bytes/1024/1024/1024:.2f} GB)")

print(f"\n✅ Metadata berhasil diekstrak!")

Simpan informasi untuk digunakan nanti
video_info = {
 'width': width,
 'height': height,
 'fps': fps,
 'frame_count': frame_count,
 'duration': duration
}

else:
 print("❌ Video tidak dimuat - tidak dapat mengekstrak metadata")
 print("Pastikan PATH_VIDEO menuju ke file video yang valid")

```

#### 📋 METADATA VIDEO

##### 📐 RESOLUSI & DIMENSI:

- 🖼️ Lebar: 720 pixels
- 🖼️ Tinggi: 1280 pixels
- 📊 Resolusi: 720 x 1280
- 💡 Aspect ratio: 0.56
- 🏷️ Klasifikasi: Full HD (1080p)

##### ⌚ INFORMASI TEMPORAL:

- 🎬 Frame rate: 30.00 fps
- 🔢 Total frame: 736
- ⌚ Durasi: 24.53 detik (0.4 menit)
- 🏷️ Klasifikasi FPS: Standard frame rate

##### 💾 ESTIMASI DATA:

- 📊 Bytes per frame: 2,764,800 (2.6 MB)
- 📦 Total estimasi (RGB): 2,034,892,800 bytes (1.9 GB)

✅ Metadata berhasil diekstrak!

## C3. TODO: Tampilkan 3 Frame (Awal–Tengah–Akhir)

**Instruksi:** Ambil dan tampilkan 3 frame representatif:

- Frame pertama (index 0)
- Frame tengah (index ~total\_frame/2)
- Frame terakhir (index total\_frame-1)
- **Konversi BGR→RGB** sebelum ditampilkan
- Subplot dengan judul frame dan timestamp

**Analisis yang diperlukan:** Deskripsikan perbedaan visual antar frame dan apa yang dapat dipelajari dari sampel frame ini.

### Analisis

- Frame Awal masih tenang, langit agak mendung, dan ombak belum terlalu aktif.
- Frame Tengah mulai kelihatan lebih hidup ombak makin bergerak lebih kencang, pencahayaan juga berubah, mungkin matahari mulai muncul atau awan bergeser.

- Frame Akhir makin dramatis: ombak lebih besar, cahaya lebih terang, dan nuansa keseluruhan lebih aktif. Kayak menjelang sore. Dari sampel ini, kita bisa lihat gimana kondisi alam (laut, langit, cahaya) berubah dalam waktu singkat. Cocok buat analisis lingkungan, cuaca.

In [80]:

```

import cv2
import matplotlib.pyplot as plt

Path ke video
video_path = r'C:\Users\Asus\Downloads\multimedia exercise\data\video pemandangan.mp4
cap = cv2.VideoCapture(video_path)

Cek apakah video berhasil dibuka
video_loaded = cap.isOpened()

if video_loaded:
 print("💡 Mengekstrak dan menampilkan frame...")

 # Ambil metadata
 width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
 height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
 fps = cap.get(cv2.CAP_PROP_FPS)
 frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))

 # Cek fps valid
 if fps == 0:
 print("⚠️ FPS tidak valid (nol), timestamp tidak bisa dihitung.")
 fps = 1 # fallback agar tidak error

 # Tentukan frame yang akan diekstrak
 frame_first = 0
 frame_middle = frame_count // 2
 frame_last = max(0, frame_count - 3)

 frames_to_extract = [
 (frame_first, "Frame Awal"),
 (frame_middle, "Frame Tengah"),
 (frame_last, "Frame Akhir")
]

 # Setup plot
 fig, axes = plt.subplots(1, 3, figsize=(18, 6))
 extracted_frames = []

 for i, (frame_idx, title) in enumerate(frames_to_extract):
 cap.set(cv2.CAP_PROP_POS_FRAMES, frame_idx)
 ret, frame = cap.read()

 if ret and frame is not None:
 frame_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
 extracted_frames.append(frame_rgb)

 axes[i].imshow(frame_rgb)
 axes[i].set_title(f"{title}\nFrame #{frame_idx}", fontsize=12, fontweight='bold')
 axes[i].axis('off')

 timestamp = frame_idx / fps
 axes[i].text(0.02, 0.98, f"Waktu: {timestamp:.2f}s",

```

```

 transform=axes[i].transAxes,
 fontsize=10,
 bbox=dict(boxstyle="round,pad=0.3", facecolor="yellow",
 verticalalignment='top')
 else:
 axes[i].text(0.5, 0.5, f"❌ Gagal\nnmembaca\n{n{title}}",
 transform=axes[i].transAxes,
 fontsize=12,
 horizontalalignment='center',
 verticalalignment='center')
 axes[i].set_title(f"{title} (Error)", fontsize=12)
 axes[i].axis('off')

plt.tight_layout()
plt.show()

Informasi ekstraksi
print("✅ Frame berhasil diekstrak!")
print(f"📊 Resolusi frame: {width} x {height}")
print("⌚ Frame yang diekstrak:")
for frame_idx, title in frames_to_extract:
 timestamp = frame_idx / fps
 print(f" {title}: Frame #{frame_idx} (waktu {timestamp:.2f}s)")

print(f"\n💡 Total frame berhasil diekstrak: {len(extracted_frames)}")
cap.release()
print("👋 Video capture resource dibersihkan")

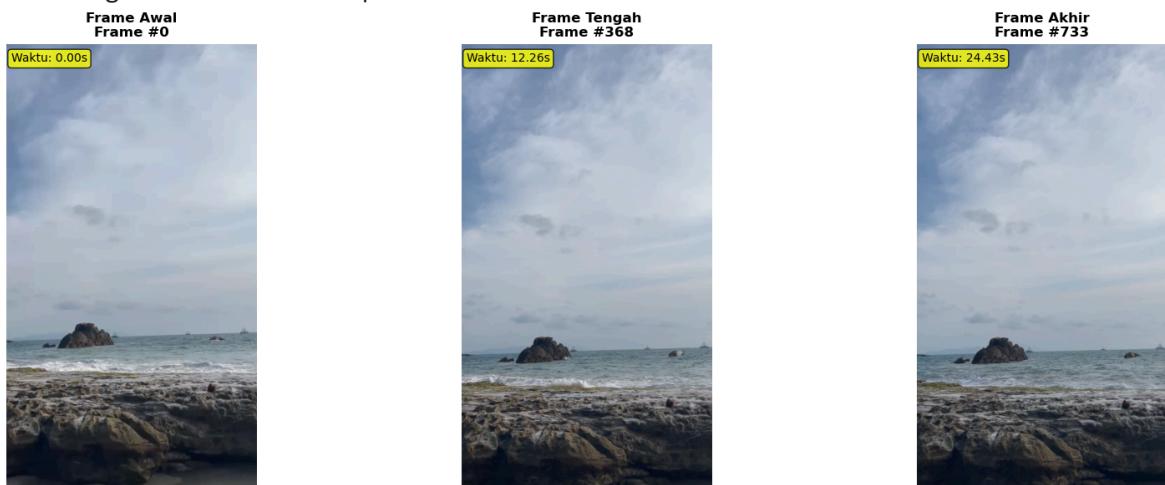
else:
 print("❌ Tidak dapat menampilkan frame - video tidak dimuat")
 print("Pastikan PATH_VIDEO menuju ke file video yang valid")

fig, axes = plt.subplots(1, 3, figsize=(18, 6))
for i, title in enumerate(["Frame Awal", "Frame Tengah", "Frame Akhir"]):
 axes[i].text(0.5, 0.5, f"❌ Video\nntidak dimuat\n\n{n{title}}",
 transform=axes[i].transAxes, fontsize=12,
 horizontalalignment='center', verticalalignment='center')
 axes[i].set_title(f"{title} (No Video)", fontsize=12)
 axes[i].axis('off')

plt.tight_layout()
plt.show()

```

🎥 Mengekstrak dan menampilkan frame...



- Frame berhasil diekstrak!
-  Resolusi frame: 720 × 1280
-  Frame yang diekstrak:
  - Frame Awal: Frame #0 (waktu 0.00s)
  - Frame Tengah: Frame #368 (waktu 12.26s)
  - Frame Akhir: Frame #733 (waktu 24.43s)
-  Total frame berhasil diekstrak: 3
-  Video capture resource dibersihkan

## C4. Analisis Ringkas (Wajib)

**Jawab pertanyaan berikut:**

**Kesesuaian parameter:** Apakah fps dan resolusi video ini sesuai untuk use case pilihan Anda (misalnya: media sosial, kuliah daring, presentasi, dll.)? Jelaskan alasan singkat.

*Jawaban Anda:* Iya, fps dan resolusinya udah cukup oke buat presentasi atau kuliah daring. Resolusi 720×1280 itu HD, jadi gambar masih tajam dan enak dilihat. FPS-nya juga stabil, jadi gerakan nggak patah-patah. Buat kebutuhan edukasi atau visualisasi konten, ini udah pas nggak terlalu berat tapi tetap jelas.

# Perbandingan & Kesimpulan

## Perbandingan Representasi Media

**TODO:** Bandingkan secara ringkas representasi dan visualisasi ketiga media:

### Audio (1D - Temporal)

- Representasi: Sinyal satu dimensi berupa amplitudo terhadap waktu.
- Visualisasi utama: Waveform, spectrogram log-dB, dan MFCC heatmap.
- Informasi yang diperoleh:
  1. Waveform kasih gambaran kapan suara pelan atau keras.
  2. Spectrogram nunjukin frekuensi suara dan intensitasnya.
  3. MFCC bantu analisis karakter suara, misalnya buat ngenalin genre musik atau suara manusia.

### Gambar (2D - Spasial)

- Representasi: Gambar itu kumpulan pixel dua dimensi, tiap titik punya warna dan intensitas.
- Visualisasi utama: Tampilan RGB, metadata (info teknis), dan histogram warna.
- Informasi yang diperoleh:
  1. Bisa lihat warna dominan dan sebaran intensitas.
  2. Info teknis kayak ukuran gambar, jumlah kanal warna, dan jenis file.
  3. Bisa nebak suasana gambar, objek utama, atau pencahayaan.

## Video (2D + Waktu - Spatio-temporal)

- Representasi: Kumpulan gambar (frame) yang bergerak seiring waktu, kayak flipbook digital
  - Visualisasi utama: Metadata video dan cuplikan frame dari awal, tengah, dan akhir.
  - Informasi yang diperoleh:
    1. Bisa lihat perubahan visual dari waktu ke waktu.
    2. Info teknis kayak durasi, resolusi, fps, dan kualitas.
    3. Cocok buat analisis gerakan, cuaca, atau perubahan lingkungan.
- 

## Refleksi Pembelajaran

### 3 Poin yang Saya Pelajari:

1. Waveform dan spectrogram punya cara beda buat nunjukin karakter suara.
2. MFCC itu penting banget buat analisis suara lebih dalam.
3. Histogram warna bisa bantu ngerti komposisi visual gambar.

### 2 Hal yang Masih Membingungkan/Ingin Diperdalam:

1. Gimana cara baca MFCC lebih dalam dan pakainya buat klasifikasi suara.
  2. Teknik lanjutan buat analisis video, kayak tracking gerakan atau deteksi objek.
- 

## Sumber Data & Referensi

**TODO:** Cantumkan semua sumber data dan referensi yang digunakan:

- **Audio:** ([https://youtu.be/ttEl35HVpqI?si=ER53LL9J\\_li3-gz\\_](https://youtu.be/ttEl35HVpqI?si=ER53LL9J_li3-gz_))
- **Gambar:** Foto pantai pribadi
- **Video:** rekaman pemandangan laut pribadi
- **Referensi teknis:** - AI tools buat bantu eksplorasi data

## Rubrik Penilaian

### Distribusi Bobot Penilaian

| Aspek Penilaian                    | Bobot      | Deskripsi                                                 |
|------------------------------------|------------|-----------------------------------------------------------|
| <b>Kelengkapan</b>                 | <b>35%</b> | Semua langkah inti dikerjakan sesuai checklist            |
| <b>Kualitas Visualisasi</b>        | <b>20%</b> | Judul, label sumbu, colorbar, legend, keterbacaan plot    |
| <b>Analisis &amp; Interpretasi</b> | <b>30%</b> | Kemampuan interpretasi hasil, bukan sekadar output mentah |

| Aspek Penilaian                      | Bobot      | Deskripsi                                      |
|--------------------------------------|------------|------------------------------------------------|
| <b>Kerapihan &amp; Struktur</b>      | <b>10%</b> | Markdown jelas, kode modular, dokumentasi baik |
| <b>Orisinalitas &amp; Penguasaan</b> | <b>5%</b>  | Pemahaman saat presentasi acak                 |

## Detail Kriteria Penilaian

### Kelengkapan (35%)

- Semua 4 visualisasi audio (metadata, waveform, spectrogram, MFCC)
- Semua 3 visualisasi gambar (display RGB, metadata, histogram)
- Semua 2 visualisasi video (metadata, frame extraction)
- Analisis ringkas untuk setiap bagian

### Kualitas Visualisasi (20%)

- Plot memiliki judul yang informatif dan deskriptif
- Label sumbu X dan Y jelas dan sesuai
- Colorbar/legend tersedia jika diperlukan
- Ukuran plot proporsional dan mudah dibaca

### Analisis & Interpretasi (30%)

- Interpretasi menunjukkan pemahaman konsep
- Analisis kontekstual, bukan sekadar deskripsi output
- Mampu menghubungkan hasil dengan teori
- Refleksi pembelajaran yang thoughtful

### Kerapihan & Struktur (10%)

- Markdown terstruktur dengan heading yang konsisten
- Kode bersih, terkompartemen, dan mudah dibaca
- Dokumentasi yang memadai
- Flow logical dari satu bagian ke bagian lain

### Orisinalitas & Penguasaan (5%)

- **PENTING:** Jika saat presentasi acak Anda tidak mampu menjelaskan kode yang Anda tulis atau menunjukkan ketergantungan buta pada AI/copy-paste, **nilai tugas akan dianggap 0.**
- Kemampuan menjelaskan logika dan alur pemikiran
- Pemahaman konsep di balik implementasi kode

## Proporsi Penilaian Total

- Proporsi penilaian hanya 80%, 20% lagi akan didasarkan pada kecepatan pengumpulan tugas
- Sehingga:  $0.8 * \text{penilaian dosen} + \text{nilai waktu pengumpulan}$

## Aturan Kejujuran Akademik

### Penggunaan Referensi & AI yang Diperbolehkan

Anda **BOLEH** menggunakan:

- Dokumentasi resmi library (NumPy, Matplotlib, Librosa, OpenCV)
- Tutorial dan contoh kode dari sumber terpercaya
- AI tools (ChatGPT, GitHub Copilot, dll.) sebagai **alat bantu pembelajaran**
- Diskusi dengan teman untuk pemahaman konsep

### Syarat & Batasan WAJIB

Namun Anda **HARUS**:

-  **Memahami setiap baris kode** yang Anda masukkan ke notebook
-  **Menulis interpretasi dengan kata-kata sendiri**, bukan hasil copy-paste
-  **Mencantumkan sumber data dan referensi** yang digunakan, termasuk transkrip percakapan dengan AI dalam link atau teks
-  **Mampu menjelaskan logika dan alur pemikiran** saat presentasi acak

### Pelanggaran yang Berakibat Nilai 0

- **Plagiarisme atau penyalinan buta** dari sumber manapun
- **Copy-paste kode tanpa pemahaman** dan tidak dapat menjelaskan
- **Menggunakan AI untuk mengerjakan seluruh tugas** tanpa pembelajaran personal
- **Tidak dapat menjawab pertanyaan dasar** tentang kode yang dikumpulkan
- **Menyalin pekerjaan teman** atau bekerjasama dalam pengerjaan individual

### Persiapan Presentasi Acak

**Kemungkinan pertanyaan yang akan ditanyakan:**

- "Jelaskan mengapa Anda menggunakan parameter ini di STFT?"
- "Apa arti dari pola yang terlihat di MFCC?"
- "Mengapa perlu konversi BGR ke RGB?"
- "Interpretasikan hasil histogram yang Anda buat"
- "Bagaimana cara kerja spectrogram?"

**Tips sukses:**

- Pahami konsep dasar setiap teknik yang digunakan
- Latih menjelaskan dengan bahasa sederhana
- Siapkan justifikasi untuk setiap pilihan parameter
- Kuasai interpretasi setiap visualisasi yang dibuat

## Panduan Pengumpulan

### Berkas yang Harus Dikumpulkan

#### Wajib:

1. **Notebook Jupyter** (.ipynb) dengan nama: `NIM_Nama_TugasMultimedia.ipynb`
    - Contoh: `123456789_JohnDoe_TugasMultimedia.ipynb`
  2. **PDF hasil render dari notebook**
- 

### Informasi Pengumpulan

### Checklist Sebelum Submit

- Semua cell sudah dijalankan dan menampilkan output
  - Nama file sesuai format: `NIM_Worksheet2.ipynb` dan `NIM_Worksheet2.pdf`
  - Semua TODO sudah diisi dengan lengkap
  - Analisis dan interpretasi sudah ditulis untuk setiap bagian
  - Sumber data dan referensi sudah dicantumkan
- 

#### Export ke PDF:

- File → Save and Export Notebook As → HTML
- Buka HTML di browser -> Save as PDF