



# Multimedia

Modul Praktek

# PERCOBAAN I

## PENGENALAN CITRA

---

### Tujuan Percobaan

- 1) Memahami definisi dari citra
- 2) memahami Struktur dari citra RGB
- 3) Menggunakan Matlab untuk memisahkan layer Red, Green, Blue pada Citra Image

### Alat yang digunakan

- Perangkat Komputer dengan OS Windows
- Aplikasi Matlab

## DASAR TEORI :

---

### APA ITU RGB

RGB adalah singkatan dari Red - Green - Blue adalah model warna pencahayaan (additive color mode) dipakai untuk "input devices" seperti scanner maupun "output devices" seperti display monitor, warna-warna primernya (Red, Blue, Green) tergantung pada teknologi alat yang dipakai seperti CCD atau PMT pada scanner atau digital camera, CRT atau LCD pada display monitor.

Apabila (Red - Blue - Green) ketiga warna tersebut dikombinasikan maka terciptalah warna putih inilah mengapa RGB disebut 'additive color' atau bahasa kerennya 'warna pencahayaan'. Warna RGB merupakan prinsip warna yang digunakan oleh media elektronik seperti televisi, monitor komputer, dan juga scanner. Oleh karena itu, warna yang ditampilkan RGB selalu terang dan menyenangkan, karena memang di setting untuk display monitor, bukan untuk cetak, sehingga lebih leluasa dalam bermain warna. Tapi bukan berarti RGB bebas masalah karena tampilan warna RGB akan selalu terikat dengan kapasitas/kemampuan grafis computer yang menyandangnya. Jadi apabila komputer yang kita

pakai mempunyai graphic card yang bagus serta monitor LCD, maka tampilan warna RGBnya akan jauh lebih bagus dibanding monitor tabung dengan graphic card yang biasa-biasa saja.

---

#### APA ITU CMYK

CMYK adalah singkatan dari Cyan-Magenta-Yellow-black dan biasanya juga sering disebut sebagai warna proses atau empat warna. CMYK adalah sebuah model warna berbasis pengurangan sebagian gelombang cahaya (subtractive color model) dan yang umum dipergunakan dalam pencetakan berwarna. Jadi untuk mereproduksi gambar sehingga dapat dicapai hasil yang (relative) sempurna dibutuhkan sedikitnya 4 Tinta yaitu: Cyan, Magenta, Yellow dan Black. Keempat tinta tersebut disebut Tinta / Warna Proses. Tinta Proses adalah tinta yang dipergunakan untuk mereproduksi warna dengan proses teknik cetak tertentu, seperti offset lithography, rotogravure, letterpress atau sablon. Berbeda dengan Tinta yang hanya digunakan satu lapisan (single layer), karena tinta yang digunakan dapat ditumpuk-tumpuk, maka sifat tinta proses harus memenuhi standard tertentu, seperti spesifikasi warna (dalam model warna CIE Lab) dan nilai Opacity/Transparency.

Kesalahan warna dalam penumpukan 2 macam tinta tersebut disebut: Ink Trapping Error (berbeda dengan Layout Trapping Error). (ISO 2846-1 hingga ISO 2846-5 adalah standar yang ditetapkan oleh badan standarisasi internasional terhadap warna dan nilai transparency dari tinta proses 4 warna CMYK masing-masing untuk proses pencetakan: Sheet-fed and heat-set web offset lithographic printing, Coldset offset lithographic printing, Publication gravure printing, Screen printing dan Flexographic printing.)

Teknik separasi saat ini sudah berkembang; Penggunaan 4 tinta proses masih dominan, tapi metode menambah warna tinta cetak berkembang pesat. Teknologi HiFi Color dikembangkan beberapa pihak antara lain Pantone mengembangkan Proses Hexachrome dan Opaltone. Pada teknik Digital Inkjet Printing, perkembangan Warna Proses sedemikian pesatnya, hal ini didorong lantaran karena masalah teknis (kecilnya nozzle dalam printing head), maupun persaingan untuk menghadirkan reproduksi warna yang sempurna (sesuai dengan target pasar yang dituju), ada tinta-tinta seperti: Light Magenta, Light Cyan, Grey, Matt Black, Orange dan Green dll. Jadi Empat Warna adalah spesifik untuk penyebutan proses pewarnaan dengan menggunakan CMYK.

---

## PERBEDAAN RGB DAN CMYK

### **Pada warna RGB**

- Red Green Blue (merah, hijau, biru)
- RGB merupakan warna-warna primer yang digunakan pada monitor
- RGB lebih digunakan untuk desain yang ditampilkan ke media layar monitor
- Jika warna RGB di campur semua, akan menghasilkan warna putih

### **Pada warna CMYK**

- Cyan Magenta Yellow Black (orang awam bilang biru, merah, kuning dan hitam )
- CMYK merupakan warna-warna primer yang paling banyak digunakan pada printer
- CMYK lebih digunakan untuk desain yang nantinya ditampilkan ke media cetak
- Jika warna CMY di campur semua, akan menghasilkan warna hitam

## Kesimpulan

Warna RGB biasanya lebih terang dan jelas maka dari itu kalau kita mendesain untuk kepentingan digital, misal Desain website, Iklan produk di TV dll, maka kita harus pakai warna RGB. Tapi kalau mendesain untuk tujuan percetakan, seperti desain Brosur, Undangan, Kalender, dll. maka CMYK adalah pilihan warnanya. Hal ini karena printer dan mesin percetakan hanya mengenal warna CMYK saja, karena mesin cetak menggunakan medium film yang sangat terbatas dalam mencerna warna. Namun juga memiliki keunggulan karena mesin cetak berhubungan dengan mass production (produksi dalam jumlah yang besar) jadi lebih efisien dan hemat uang & waktu.

Oleh karena itu pasti ada penurunan kualitas / grade apabila warna RGB dipaksakan masuk film / Percetakan (perubahan warnanya bisa besar). maka bisa konversi dulu saja ke CMYK, tapi jangan kaget bila nantinya warna akan berubah menjadi lebih redup dan tidak secerah pada warna RGB. maka dari itu desainer harus pintar pintar memastikan terlebih dahulu desainnya akan tampil dengan warna RGB atau CMYK





```
% Read in original RGB image.
rgbImage = imread('flower.png');
% Extract color channels.
redChannel = rgbImage(:,:,1); % Red channel
greenChannel = rgbImage(:,:,2); % Green channel
blueChannel = rgbImage(:,:,3); % Blue channel
% Create an all black channel.
allBlack = zeros(size(rgbImage, 1), size(rgbImage, 2),
'uint8');
% Create color versions of the individual color channels.
just_red = cat(3, redChannel, allBlack, allBlack);
just_green = cat(3, allBlack, greenChannel, allBlack);
just_blue = cat(3, allBlack, allBlack, blueChannel);
```

```

% Recombine the individual color channels to create the
original RGB image again.
recombinedRGBImage = cat(3, redChannel, greenChannel,
blueChannel);
% Display them all.
subplot(3, 3, 2);
imshow(rgbImage);
fontSize = 20;
title('Original RGB Image', 'FontSize', fontSize)
subplot(3, 3, 4);
imshow(just_red);
title('Red Channel in Red', 'FontSize', fontSize)
subplot(3, 3, 5);
imshow(just_green)
title('Green Channel in Green', 'FontSize', fontSize)
subplot(3, 3, 6);
imshow(just_blue);
title('Blue Channel in Blue', 'FontSize', fontSize)
subplot(3, 3, 8);
imshow(recombinedRGBImage);
title('Recombined to Form Original RGB Image Again',
'FontSize', fontSize)
% Set up figure properties:
% Enlarge figure to full screen.
set(gcf, 'Units', 'Normalized', 'OuterPosition', [0, 0, 1,
1]);
% Get rid of tool bar and pulldown menus that are along top
of figure.
% set(gcf, 'Toolbar', 'none', 'Menu', 'none');
% Give a name to the title bar.
set(gcf, 'Name', 'Demo by ImageAnalyst', 'NumberTitle',
'Off')

```

## PROSEDUR PERCOBAAN

1. Download Image ,
2. Jelaskan fungsi dari Split code rgb diatas
3. Buat GUI untuk mengubah gambar RGB ke hingga menghasilkan output Red, Blue, Green, Grayscale, citra biner

## PERCOBAAN II

### PENGENALAN AUDIO

---

#### Tujuan Percobaan

- 1) Memahami defenisi dari Audio
- 2) memahami STruktur dari citra RGB
- 3) Menggunakan Matlab untuk memisahkan layer Red, Green, Blue pada Citra Image

#### Alat yang digunakan

- Perangkat Komputer dengan OS Windows
- Aplikasi Matlab

#### **Write to Audio File**

Load sample data from the file, handel.mat

```
load handel.mat
```

The workspace now contains a matrix of audio data, *y*, and a sample rate, *Fs*.

Use the `audiowrite` function to write the data to a WAVE file named `handel.wav` in the current folder.

```
audiowrite('handel.wav',y,Fs)
```

```
clear y Fs
```

The `audiowrite` function also can write to other audio file formats such as OGG, FLAC, and MPEG-4 AAC.

#### Get Information About Audio File

Use the `audioinfo` function to get information about the WAVE file, `handel.wav`.

```
info = audioinfo('handel.wav')
```

```
INFO =  
  
      FILENAME: 'PWD\HANDEL.WAV'  
  COMPRESSIONMETHOD: 'UNCOMPRESSED'  
    NUMCHANNELS: 1  
    SAMPLERATE: 8192  
  TOTALSAMPLES: 73113  
    DURATION: 8.9249  
      TITLE: []  
    COMMENT: []  
     ARTIST: []  
  BITSPERSAMPLE: 16
```

audioinfo returns a 1-by-1 structure array. The SampleRate field indicates the sample rate of the audio data, in hertz. The Duration field indicates the duration of the file, in seconds.

### Read Audio File


Use the audioread function to read the file, handel.wav. The audioread function can support WAVE, OGG, FLAC, AU, MP3, and MPEG-4 AAC files.

```
[y,Fs] = audioread('handel.wav');
```

Play the audio.

```
sound(y,Fs)
```

You also can read WAV, AU, or SND files interactively.

Select  Import Data or double-click the file name in the Current Folder browser.

### Plot Audio Data

Create a vector t the same length as y, that represents elapsed time.

```
t = 0:seconds(1/Fs):seconds(info.Duration);
```

```
t = t(1:end-1);
```

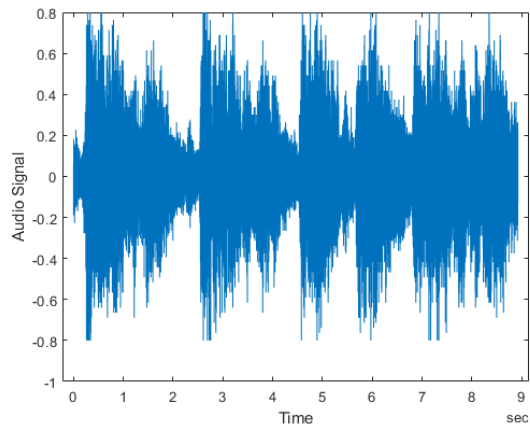


Plot the audio data as a function of time.

```
plot(t,y)
```

```
xlabel('Time')
```

```
ylabel('Audio Signal')
```



## PROSEDUR PERCOBAAN

1. BUAT GUI DENGAN MATLAB YANG MENAMPILKAN :
  - a. INFO
  - b. TYPE AUDIO
  - c. MEMUTAR FILE AUDIO /PLAY
  - d. MENAMPILKAN AUDIO SIGNAL

# PERCOBAAN III

## MEMBUAT VIDEO DARI GAMBAR

---

### CONVERT BETWEEN IMAGE SEQUENCES AND VIDEO

Try This Example

This example shows how to convert between video files and sequences of image files using VideoReader and VideoWriter.

The sample file named shuttle.avi contains 121 frames. Convert the frames to image files using VideoReader and the imwrite function. Then, convert the image files to an AVI file using VideoWriter.

---

#### SETUP

Create a temporary working folder to store the image sequence.

```
workingDir = tempname;  
mkdir(workingDir)  
mkdir(workingDir, 'images')
```

---

#### CREATE VIDEOREADER

Create a VideoReader to use for reading frames from the file.

```
shuttleVideo = VideoReader('shuttle.avi');
```

---

#### CREATE THE IMAGE SEQUENCE

Loop through the video, reading each frame into a width-by-height-by-3 array named img. Write out each image to a JPEG file with a name in the form imgN.jpg, where N is the frame number.

```
| img001.jpg|  
| img002.jpg|  
| ...|  
| img121.jpg|  
ii = 1;
```

```
while hasFrame(shuttleVideo)  
    img = readFrame(shuttleVideo);
```

```
filename = [sprintf('%03d',ii) '.jpg'];
fullname = fullfile(workingDir,'images',filename);
imwrite(img,fullname)      % Write out to a JPEG file
(img1.jpg, img2.jpg, etc.)
ii = ii+1;
```

end

---

#### FIND IMAGE FILE NAMES

Find all the JPEG file names in the images folder. Convert the set of image names to a cell array.

```
imageNames = dir(fullfile(workingDir,'images','*.jpg'));
imageNames = {imageNames.name};
```

---

#### CREATE NEW VIDEO WITH THE IMAGE SEQUENCE

Construct a VideoWriter object, which creates a Motion-JPEG AVI file by default.

```
outputVideo =
VideoWriter(fullfile(workingDir,'shuttle_out.avi'));
outputVideo.FrameRate = shuttleVideo.FrameRate;
open(outputVideo)
Loop through the image sequence, load each image, and then write
it to the video.
```

```
for ii = 1:length(imageNames)
    img = imread(fullfile(workingDir,'images',imageNames{ii}));
    writeVideo(outputVideo,img)
```

end

Finalize the video file.

```
close(outputVideo)
```

---

#### VIEW THE FINAL VIDEO

Construct a reader object.

```
shuttleAvi =
VideoReader(fullfile(workingDir,'shuttle_out.avi'));
Create a MATLAB movie struct from the video frames.
```

```
ii = 1;
while hasFrame(shuttleAvi)
    mov(ii) = im2frame(readFrame(shuttleAvi));
    ii = ii+1;
```

end

Resize the current figure and axes based on the video's width and height, and view the first frame of the movie.

```
figure
```

```
imshow(mov(1).cdata, 'Border', 'tight')
```



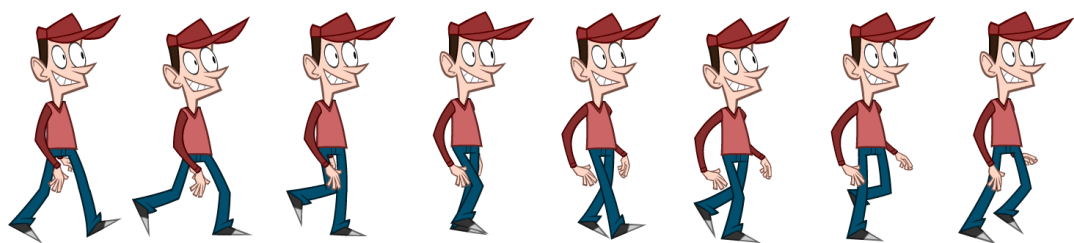
Play back the movie once at the video's frame rate.

```
movie(mov,1,shuttleAvi.FrameRate)
```



## PROSEDUR PERCOBAAN

- BUATLAH VIDEO DARI GAMBAR BERIKUT INI  
<https://bit.ly/2xYFybF>



# PERCOBAAN IV

## MENGINSTAL DAN MELAKUKAN KONVERSI VIDEO DAN AUDIO DENGAN FFMPEG

### Teori dasar

Fmpeg merupakan salah satu framework yang bisa digunakan untuk decode, encode, transcode, mux, demux, stream, filter untuk berbagai format multimedia. Menurut situsnya, ffmpeg.org, FFmpeg diklaim sebagai yang terdepan dalam fungsi-fungsi tersebut. Menurut referensi yang penulis baca namun lupa, beberapa media player dan media converter seperti VLC dan Total Media Converter/Player juga berbasis pada FFmpeg.

Berdasarkan pengalaman penulis, keuntungan menggunakan FFmpeg adalah kesederhanaannya. FFmpeg hanya melakukan proses decode dan encode berdasarkan standar dasar dari proses decode dan encode itu sendiri. FFmpeg yang memungkinkan proses decode dan encode terjadi tanpa ada tambahan "pesan-pesan sponsor" dari software encoding. Sehingga ketika terjadi penurunan kualitas akibat proses tersebut, pengguna tidak perlu khawatir terjadi penurunan kualitas dari hal-hal yang diluar kontrol seperti sisipan copyright software converter. Hal ini sangat menguntungkan untuk pemrosesan video untuk penelitian.

Akan tetapi, FFmpeg yang berbasis pada Command Line Interface (CLI) memerlukan usaha lebih dari pengguna untuk mempelajarinya dan menggunakannya. Bagi penulis, kebiasaan menggunakan Graphic User Interface (GUI) perlu waktu untuk beradaptasi dari proses "klik dan klik" menjadi "ketik dan enter".

Maka dari itu, tulisan ini dibuat sebagai catatan bagi penulis sekaligus bermanfaat bagi para pembaca dalam langkah awal menggunakan FFmpeg.

Sintaks tipikal dari perintah FFmpeg adalah:

```
ffmpeg [global_options] {[input_file_options] -i input_url} ...
```



```
{[output_file_options] output_url} ...
```

Kami sekarang akan melihat beberapa perintah FFmpeg yang penting dan berguna.

### **1. Mendapatkan informasi file audio / video**

Untuk menampilkan detail file media Anda, jalankan:

```
$ ffmpeg -i video.mp4
```

#### **Output sampel:**

```
ffmpeg versi 3.3 Hak cipta (c) 2000-2017 pengembang FFmpeg
```

```
dibangun dengan gcc 6.3.1 (GCC) 20170306
```

```
konfigurasi: --prefix = / usr --disable-debug --disable-static
--disable-stripping --enable-avisynth --enable-avresample --
enable-fontconfig --enable-gmp --enable-gnutls - enable-gpl --
enable-ladspa --enable-libass --enable-libbluray --enable-
libfreetype --enable-libfribidi --enable-libgsm --enable-
libiec61883 --enable-libmodplug --enable-libmp3lame - enable-
libopencore_amrnb --enable-libopencore_amrwb --enable-
libopenjpeg --enable-libopus --enable-libpulse --enable-
libschrödinger --enable-libsoxr --enable-libspeex --enable-
libssh --enable-libtheora - enable-libv4l2 --enable-libvidstab -
-enable-libvorbis --enable-libvpx --enable-libwebp --enable-
libx264 --enable-libx265 --enable-libxcb --enable-libxvid --
enable-netcdf - enable-shared --enable-version3
```

```
libavutil 55. 58.100 / 55. 58.100
```

```
libavcodec 57. 89.100 / 57. 89.100
```

libavformat 57. 71.100 / 57. 71.100

libavdevice 57. 6.100 / 57. 6.100

libavfilter 6. 82.100 / 6. 82.100

libavresample 3. 5. 0 / 3. 5. 0

libswscale 4. 6.100 / 4. 6.100

libswresample 2. 7.100 / 2. 7.100

libpostproc 54. 5.100 / 54. 5.100

Masukan # 0, mov, mp4, m4a, 3gp, 3g2, mj2, dari 'video.mp4':

Metadata:

major\_brand: isom

minor\_version: 512

compatible\_brands: isomiso2avc1mp41

pembuat encode: Lavf57.22.100

Durasi: 00: 43: 18,69, mulai: 0,000000, bitrate: 1039 kb / dtk

Aliran # 0: 0 (und): Video: h264 (Tinggi) (avc1 / 0x31637661), yuv420p, 1280x714 [SAR 1071: 1072 DAR 120: 67], 899 kb / s, 23.98 fps, 23.98 tbr, 24k tbn, 47.95 tbc (default)

Metadata:

handler\_name: VideoHandler

```
Aliran # 0: 1 (und): Audio: aac (LC) (mp4a / 0x6134706D), 48000
Hz, stereo, fltp, 132 kb / s (default)
```

Metadata:

```
handler_name: SoundHandler
```

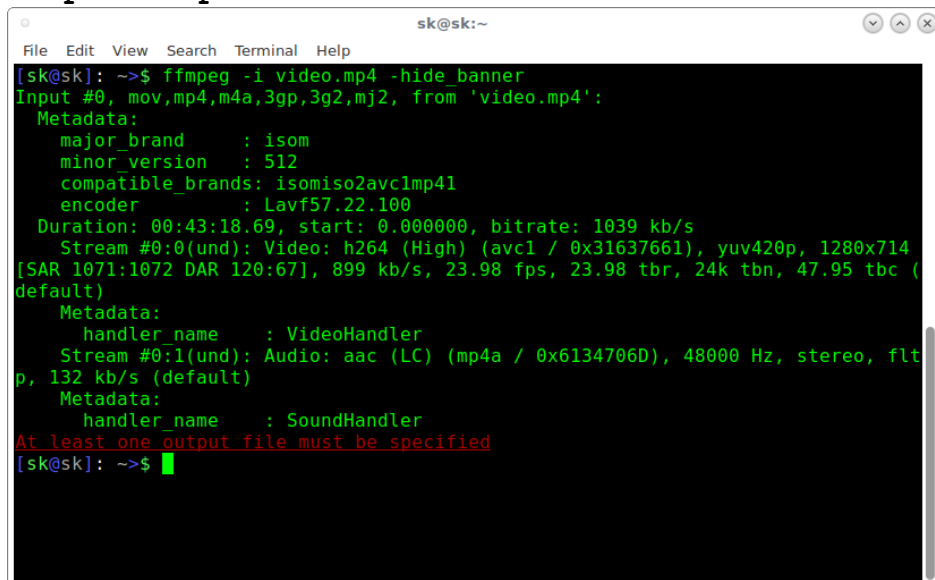
Setidaknya satu file output harus ditentukan

Seperti yang Anda lihat pada output di atas, FFmpeg menampilkan informasi file media bersama dengan rincian FFmpeg seperti versi, rincian konfigurasi, pemberitahuan hak cipta, opsi membangun dan perpustakaan, dll.

Jika Anda tidak ingin melihat banner FFmpeg dan detail lainnya, tetapi hanya informasi file media, gunakan flag -**hide\_banner** seperti di bawah ini.

```
$ ffmpeg -i video.mp4 -hide_banner
```

**Output sampel:**



```
sk@sk:~  
File Edit View Search Terminal Help  
[sk@sk]: ~->$ ffmpeg -i video.mp4 -hide_banner  
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'video.mp4':  
  Metadata:  
    major_brand      : isom  
    minor_version    : 512  
    compatible_brands: isomiso2avc1mp41  
    encoder          : Lavf57.22.100  
  Duration: 00:43:18.69, start: 0.000000, bitrate: 1039 kb/s  
  Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p, 1280x714  
[SAR 1071:1072 DAR 120:67], 899 kb/s, 23.98 fps, 23.98 tbr, 24k tbn, 47.95 tbc (default)  
  Metadata:  
    handler name      : VideoHandler  
  Stream #0:1(und): Audio: aac (LC) (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 132 kb/s (default)  
  Metadata:  
    handler_name      : SoundHandler  
At least one output file must be specified  
[sk@sk]: ~->$
```

Lihat? Sekarang, ini hanya menampilkan detail file media.

## 2. Konversi file video ke format yang berbeda

FFmpeg adalah konverter audio dan video yang kuat, jadi Anda dapat mengkonversi file media di antara berbagai format. Katakanlah misalnya, untuk mengkonversi file mp4 ke file avi, jalankan:

```
$ ffmpeg -i video.mp4 video.avi
```

Demikian pula, Anda dapat mengkonversi file media ke format pilihan Anda.

Misalnya, untuk mengonversi youtube format video flv ke format mpeg, jalankan:

```
$ ffmpeg -i video.flv video.mpeg
```

Jika Anda ingin mempertahankan kualitas file video sumber Anda, gunakan parameter '-qscale 0':

```
$ ffmpeg -i input.webm -qscale 0 output.mp4
```

Untuk memeriksa daftar format yang didukung oleh FFmpeg, jalankan:

```
$ ffmpeg -formats
```

## 3. Mengkonversi file video ke file audio

Untuk mengkonversi file video ke file audio, cukup tentukan format output sebagai .mp3, or .ogg, atau format audio lainnya.

Perintah di atas akan mengkonversi file video input.mp4 ke output.mp3 file audio.

```
$ ffmpeg -i input.mp4 -vn -ab 320 output.mp3
```

Juga, Anda dapat menggunakan berbagai opsi transcoding audio ke file output seperti yang ditunjukkan di bawah ini.

```
$ ffmpeg -i input.mp4 -vn -ar 44100 -ac 2 -ab 320 -f output  
mp3.mp3
```

Sini,

- **-vn** - Menunjukkan bahwa kami telah menonaktifkan perekaman video dalam file output.
- **-ar** - Atur frekuensi audio dari file output. Nilai-nilai umum yang digunakan adalah 22050, 44100, 48000 Hz.
- **-ac** - Mengatur jumlah saluran audio.
- **-ab** - Menunjukkan bitrate audio.
- **-f** - Format file keluaran. Dalam kasus kami, itu adalah format mp3.

#### 4. Ubah resolusi file video

Jika Anda ingin menetapkan resolusi tertentu ke file video, Anda dapat menggunakan perintah berikut:

```
$ ffmpeg -i input.mp4 -filter: skala v = 1280: 720 -c: salinan  
output.mp4
```

Atau,

```
$ ffmpeg -i input.mp4 -s 1280x720 -c: hasil salin.mp4
```

Perintah di atas akan mengatur resolusi file video yang diberikan ke 1280 × 720.

Demikian pula, untuk mengonversi file di atas menjadi ukuran 640 × 480, jalankan:

```
$ ffmpeg -i input.mp4 -filter: skala v = 640: 480 -c: salinan  
output.mp4
```

Atau,

```
$ ffmpeg -i input.mp4 -s 640x480 -c: hasil salin.mp4
```

Trik ini akan membantu Anda untuk menskalakan file video Anda ke perangkat layar yang lebih kecil seperti tablet dan ponsel.

### **5. Mengompresi file video**

Itu selalu lebih baik untuk mengurangi ukuran file media ke ukuran yang lebih rendah untuk menghemat ruang harddisk.

Perintah berikut ini akan memampatkan dan mengurangi ukuran file output.

```
$ ffmpeg -i input.mp4 -vf scale = 1280: -1 -c: v libx264 -preset  
veryslow -crf 24 output.mp4
```

Harap dicatat bahwa Anda akan kehilangan kualitas jika Anda mencoba untuk mengurangi ukuran file video. Anda dapat menurunkan nilai crf ke 23 atau lebih rendah jika 24 terlalu agresif.

Anda juga dapat mentranskode audio turun sedikit dan membuatnya stereo untuk mengurangi ukuran dengan menyertakan opsi berikut.

```
-ac 2 -c: aac -strict -2 -b: 128k
```

### **6. Mengompresi file Audio**

Cukup kompres file video, Anda dapat memampatkan file audio menggunakan **-ab** flag untuk menghemat ruang disk.



Katakanlah Anda memiliki file audio bitrate 320 kbps. Anda ingin mengompresnya dengan mengubah bitrate ke nilai yang lebih rendah seperti di bawah ini.

```
$ ffmpeg -i input.mp3 -ab 128 output.mp3
```

Daftar berbagai bitrate audio yang tersedia adalah:

1. 96kbps
2. 112 kbps
3. 128 kbps
4. 160 kbps
5. 192 kbps
6. 256 kbps
7. 320 kbps

#### **7. Menghapus aliran audio dari file media**

Jika Anda tidak ingin audio dari file video, gunakan **-an** flag.

```
$ ffmpeg -i input.mp4 -an output.mp4
```

Di sini, 'an' menunjukkan tidak ada rekaman audio.

Perintah di atas akan membatalkan semua flag yang terkait dengan audio, karena kita tidak melakukan audio dari input.mp4.

#### **8. Menghapus aliran video dari file media**

Demikian pula, jika Anda tidak ingin aliran video, Anda dapat dengan mudah menghapusnya dari file media menggunakan flag 'vn'. vn berarti tidak ada rekaman video. Dengan kata lain, perintah ini mengubah file media yang diberikan menjadi file audio.

Perintah berikut akan menghapus video dari file media yang diberikan.

```
$ ffmpeg -i input.mp4 -vn output.mp3
```

Anda juga dapat menyebutkan bitrate file output menggunakan flag '-ab' seperti yang ditunjukkan pada contoh berikut.

```
$ ffmpeg -i input.mp4 -vn -ab 320 output.mp3
```

## 9. Mengekstrak gambar dari video

Fitur lain yang bermanfaat dari FFmpeg adalah kita dapat dengan mudah mengekstrak gambar dari file video. Ini bisa sangat berguna, jika Anda ingin membuat album foto dari file video.

Untuk mengekstrak gambar dari file video, gunakan perintah berikut:

```
$ ffmpeg -i input.mp4 -r 1 -f image2 image-% 2d.png
```

Sini,

- **-r** - Mengatur frekuensi gambar. Yaitu jumlah frame yang akan diekstraksi menjadi gambar per detik. Nilai standarnya adalah 25.
- **-f** - Menunjukkan format output yaitu format gambar dalam kasus kami.
- **image-% 2d.png** - Mengindikasikan bagaimana kami ingin menamai gambar yang diekstraksi. Dalam hal ini, nama-nama harus dimulai seperti gambar-01.png, gambar-02.png, gambar-03.png dan sebagainya. Jika Anda menggunakan % 3d, maka nama gambar akan mulai seperti gambar-001.png, gambar-002.png dan seterusnya.

## 10. Memotong video

Ini agak mirip dengan mengubah resolusi file video. mari kita katakan Anda ingin video dengan ukuran 300x300. Anda bisa melakukannya dengan menggunakan perintah:

```
$ ffmpeg -i input.mp4 -croptop 100 -cropbottom 100 -cropleft 300 -cropright 300 output.mp4
```

Harap dicatat bahwa memotong video akan mempengaruhi kualitas. Jangan lakukan ini kecuali itu perlu.

### **11. Konversi bagian tertentu dari suatu video**

Terkadang, Anda mungkin ingin mengonversi hanya sebagian tertentu dari file video ke format yang berbeda. Katakanlah misalnya, perintah berikut akan mengubah 50 detik pertama dari file video.mp4 yang diberikan ke format video.avi.

```
$ ffmpeg -i input.mp4 -t 50 output.avi
```

Di sini, kita tentukan waktunya dalam hitungan detik. Juga, dimungkinkan untuk menentukan waktu dalam format **hh.mm.ss**.

### **12. Atur rasio aspek ke video**

Anda dapat mengatur rasio aspek ke file video menggunakan- **pindahkan** bendera seperti di bawah ini.

```
$ ffmpeg -i input.mp4 -mengambil 16: 9 output.mp4
```

Rasio aspek yang umum digunakan adalah:

- 16: 9
- 4: 3
- 16:10
- 5: 4
- 2: 21: 1
- 2: 35: 1
- 2: 39: 1

### **13. Menambahkan gambar poster ke file audio**

Anda dapat menambahkan gambar poster ke file Anda, sehingga gambar akan ditampilkan saat memutar file audio. Ini dapat berguna untuk meng-host file audio di Video hosting atau situs web berbagi.

```
$ ffmpeg -loop 1 -i inputimage.jpg -i inputaudio.mp3 -c: v  
libx264 -c: aac -strict experimental -b: 192k -shortest  
output.mp4
```

#### 14. Potong file media menggunakan waktu mulai dan berhenti

Untuk memangkas video ke klip yang lebih kecil menggunakan waktu mulai dan berhenti, kita dapat menggunakan perintah berikut.

```
$ ffmpeg -i input.mp4 -ss 00:00:50 -codec copy -t 50 output.mp4
```

Sini,

- -S - Menunjukkan waktu mulai klip video. Dalam contoh kami, waktu mulai adalah detik ke-50.
- -t - Menunjukkan total durasi waktu.

Ini sangat membantu ketika Anda ingin memotong bagian dari file audio atau video menggunakan waktu mulai dan akhir.

Demikian pula, kita dapat memangkas file audio seperti di bawah ini.

```
$ ffmpeg -i audio.mp3 -ss 00:01:54 -to 00:06:53 -c salin  
output.mp3
```

#### 15. Pisahkan file video menjadi beberapa bagian

Beberapa situs web akan memungkinkan Anda mengunggah hanya ukuran video tertentu. Dalam kasus seperti itu, Anda dapat membagi file video besar menjadi beberapa bagian yang lebih kecil seperti di bawah ini.

```
$ ffmpeg -i input.mp4 -t 00:00:30 -c salin part1.mp4 -ss  
00:00:30 -codec salin part2.mp4
```

Di sini, **-t 00:00:30** menunjukkan bagian yang dibuat dari awal video ke detik ke-30 video. **-ss 00:00:30** menunjukkan stempel waktu mulai untuk video. Itu berarti bahwa bagian ke-2 akan mulai dari detik ke-30 dan akan terus berlanjut hingga akhir file video asli.

**Fitur Unduhan - [Panduan Gratis: "Cara Memulai Podcast Sukses Anda Sendiri"](#)**

## 16. Bergabung dengan beberapa bagian video menjadi satu

FFmpeg juga akan bergabung dengan beberapa bagian video dan membuat satu file video.

Buat file **join.txt** yang berisi jalur yang tepat dari file yang ingin Anda gabungkan. Semua file harus memiliki format yang sama (codec yang sama). Nama jalan dari semua file harus disebutkan satu per satu seperti di bawah ini.

```
/home/sk/myvideos/part1.mp4
```

```
/home/sk/myvideos/part2.mp4
```

```
/home/sk/myvideos/part3.mp4
```

```
/home/sk/myvideos/part4.mp4
```

Sekarang, gabung semua file menggunakan perintah:

```
$ ffmpeg -f concat -i join.txt -c salin output.mp4
```

Perintah di atas akan menggabungkan part1.mp4, part2.mp4, part3.mp4, dan part4, mp4 file ke dalam satu file bernama "output.mp4".

## 17. Tambahkan sub judul ke file video

Kami juga dapat menambahkan subtitle ke file video menggunakan FFmpeg. Unduh subtitle yang benar untuk video Anda dan tambahkan video Anda seperti yang ditunjukkan di bawah ini.

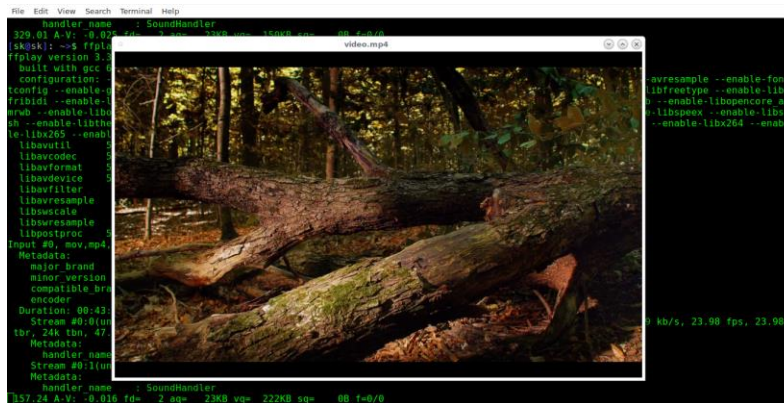
```
$ ffmpeg -i input.mp4 -i subtitle.srt -map 0 -map 1 -c copy -c:v libx264 -crf 23 -preset veryfast output.mp4
```

## 18. Pratinjau atau uji file video atau audio

Anda mungkin ingin melihat untuk memverifikasi atau menguji apakah file output telah ditranskode dengan benar atau

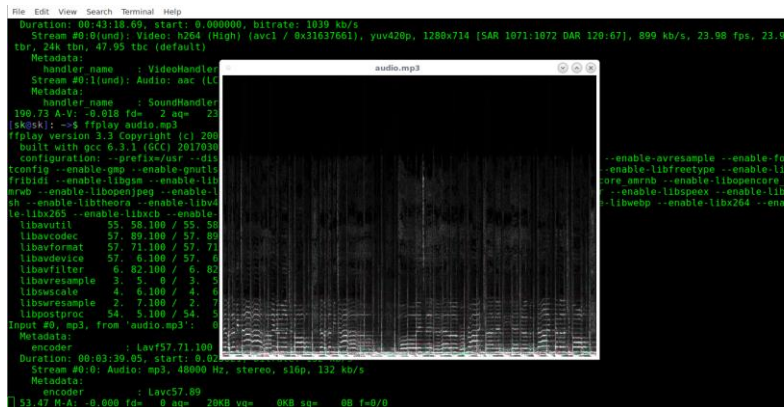
tidak. Untuk melakukannya, Anda dapat memutarnya dari Terminal dengan perintah:

```
$ ffplay video.mp4
```



Demikian pula, Anda dapat menguji file audio seperti yang ditunjukkan di bawah ini.

```
$ ffplay audio.mp3
```



## 19. Meningkatkan / menurunkan kecepatan pemutaran video

FFmpeg memungkinkan Anda untuk mengatur kecepatan pemutaran video. Untuk meningkatkan kecepatan pemutaran video, jalankan:



```
$ ffmpeg -i inputvideo.mp4 -vf "setpts = 0,5 * PTS"
outputvideo.mp4
```

Untuk mengurangi kecepatan pemutaran, jalankan:

```
$ ffmpeg -i inputvideo.mp4 -vf "setpts = 4.0 * PTS"
outputvideo.mp4
```

## 20. Mendapatkan bantuan

Dalam panduan ini, saya hanya membahas perintah FFmpeg yang paling umum digunakan. FFmpeg memiliki lebih banyak pilihan berbeda untuk melakukan berbagai fungsi lanjutan. Untuk mempelajari lebih lanjut tentang itu, lihat halaman manual.

```
$ man ffmpeg
```

## PROSEDUR PERCOBAAN

1. Download dan install aplikasi FFMPEG
2. Download 1 video dengan format mp4 dgn durasi max 1 menit  
Tampilkan meta data masing2 media tersebut
3. Catat metadata sebelum file di konversi
4. Untuk Video
  - a. 800 x 600
  - b. Aspect rasio 4:3
  - c. Fps 20
  - d. Ubah ke type MKV
  - e. Tampilkan metadata hasil konversi dan catat infonya
  - f. Ambil file AUdionya dengan format MP3
5. Untuk Audionya
  - a. Tampilkan metadatanya
  - b. Ubah bitrate ke 96 kbps
  - c. Ubah ke OGG
  - d. Tampilkan metadata hasil konversi dan catat infonya
  - Tuliskan code yang digunakan

# PERCOBAAN V

## STREAMING DENGAN LOCALHOST

Video streaming adalah istilah yang sering kita gunakan saat melihat video diinternet melalui browser dimana kita tidak perlu men-download file video tersebut untuk dapat memutaranya. Istilah ini tersebut terdiri dari dua suku kata yaitu video dan streaming, secara istilah video berarti teknologi untuk menangkap, merekam, memproses, mentransmisikan dan menata ulang gambar bergerak, sedangkan streaming berarti proses penghantaran data dalam aliran berkelanjutan dan tetap yang memungkinkan pengguna mengakses dan menggunakan file sebelum data dihantar sepenuhnya.

Jadi video streaming dapat diartikan transmisi file video secara berkelanjutan yang memungkinkan video tersebut diputar

tanpa menunggu file video tersebut tersampaikan secara keseluruhan.

Video streaming banyak diimplementasikan pada dunia pertelevisian untuk melakukan siaran dari website atau mengirimkan gambar siaran langsung melalui website atau disebut juga live streaming. Jadi gambar yang didapatkan dari siaran langsung, sesegera mungkin ditransmisikan dan dapat diputar melalui internet.

## PROSEDUR PERCOBAAN

Alat yang digunakan

- Perangkat Komputer dengan OS Windows
- Aplikasi XAMPP
- Notepad ++

Install Aplikasi XAMPP dan download 3 file Video

- Buat 1 halaman untuk menampilkan video
- Buat index video
- Catat code yang digunakan dan penjelasannya

# PERCOBAAN VI

## LIVE STREAMING DENGAN RTMP

Server nginx memiliki banyak opsi yang berguna untuk ditawarkan dan rtmp-module memperluas ini lebih jauh lagi. Hari ini saya akan segera menjelaskan cara mengonfigurasi nginx Anda untuk memasok pemirsa Anda dengan sistem video on demand. Kami akan menggunakan file video flv (saya punya masalah membuat mp4 untuk bekerja dengan benar, tetapi Anda mungkin lebih beruntung dengan itu) dan saya akan menunjukkan cara men-setup situs web sederhana yang menunjukkan kepada Anda webplayer dengan daftar putar.

Pertama-tama kita perlu mengkonfigurasi lokasi tempat video kita disimpan, jadi nginx dapat menemukannya, mari kita lihat contoh file konfigurasi:

```

rtmp {
    server {
        listen 1935;
        chunk_size 8192;

        application vod {
            play D:\Capture;
        }
    }
}

```

Dalam contoh ini, vod saya akan disimpan di drive "D:" saya (Windows nginx) di folder "Capture", untuk pengguna linux Anda mungkin ingin menggunakan sesuatu seperti "/ var / users / vod", atau folder apa pun yang Anda gunakan. ingin digunakan. Itu semua yang kita butuhkan dari bagian server nginx, file flv / mp4 apa pun dalam folder sekarang dapat dilihat menggunakan alamat server rtmp Anda: rtmp:

```
//yourserverip/application/filename.flv
```

Konfigurasi menetapkan nama aplikasi ke vod dan yourserverip tergantung jika Anda ingin mengakses server di dalam jaringan lokal Anda atau dari internet. Jadi alamat contoh untuk jaringan lokal akan terlihat seperti ini: rtmp:

```
//192.168.100.75/vod/video.flv
```

Sekali lagi, saya mengalami masalah menggunakan file mp4 dengan JWPlayer dan semua pemutar media lokal saya mencoba, jadi saya akan merekomendasikan untuk menggunakan file flv. Arut juga merekomendasikan bahwa mereka harus diindeks dengan program eksternal, untuk mencari kemampuan. Satu hal lagi yang perlu diperhatikan, jika Anda ingin memuat file video yang terletak di sub-folder Anda harus mengatur alamat yang sedikit berbeda:

```
rtmp: //yourserverip/application/subfolder/video.flv
```

TIDAK bekerja, tapi

```
rtmp://yourserverip/application//subfolder/video.flv4
```

tidak! Dan untuk memutar ulang vod menggunakan FFplay misalnya, Anda harus menentukan app dan playpath independen:

```
ffplay "rtmp://yourserverip app=vod  
playpath=subfolder/video.flv"
```

Sub-Sub-Folder tentu saja juga mungkin:

```
rtmp://yourserverip/application//subfolder/subfolder2/video.flv
```

atau

```
ffplay "rtmp://yourserverip app=vod  
playpath=subfolder/subfolder2/video.flv"
```

## PROSEDUR PERCOBAAN

Alat yang digunakan

- Perangkat Komputer dengan OS Linux
- Aplikasi Nginx

---

## INSTALL NGINX

- Konfigurasi server nginx untuk RTMP
- Letakkan file video sesuai konfigurasi nginx
- Uji coba dengan vlc apakah RTMP telah berjalan

# PERCOBAAN VII

## DISTRIBUSI MULTIMEDIA MELALUI WEB

---

### PROSEDUR PERCOBAAN

Alat yang digunakan

- Perangkat Komputer dengan OS Linux / Windows
- Aplikasi FFmpeg
- Aplikasi Nginx
- Media gambar, audio, dan video

### PRAKTEK

- Desain satu website local dengan beberapa halaman di dalamnya yang menambillakn text, gambar, video player, dan mp3 player



