Digital Video Processing

Unit - I

Contents

- Fundamentals of Video Processing:
    - Digital Video Acquisition
    - Principles of Color Video
    - Video Camera
    - Video Display
    - Analog Vs Digital Video
        - Progressive Vs Interlaced scans
        - Signal
        - Bandwidth Characterization of a Digital Video Signal

Basics of Video

- Image:
  - A spatial representation of intensity values which are constant over time.
  - A time-invariant intensity value representation

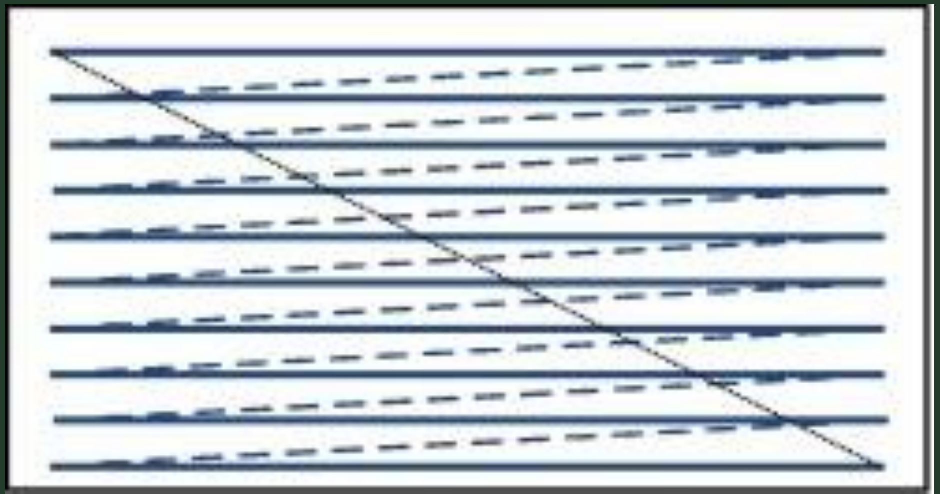
- Video
  - Time variant spatial representation of intensity values
  - Video can also be regarded as *Image Sequence*
  - *It can be regarded as a one dimensional analog or digital signal of time where the spatio-temporal information is ordered as a function of time according to a predefined scanning conversion*
  - Mathematically $I_{xy} = Sc(x, y, t)$ where I_{xy} is the intensity value at (x, y) position in frame at time t
  - Depending on the type of signal, we have two types of video signals
    - Analog Video
    - Digital Video

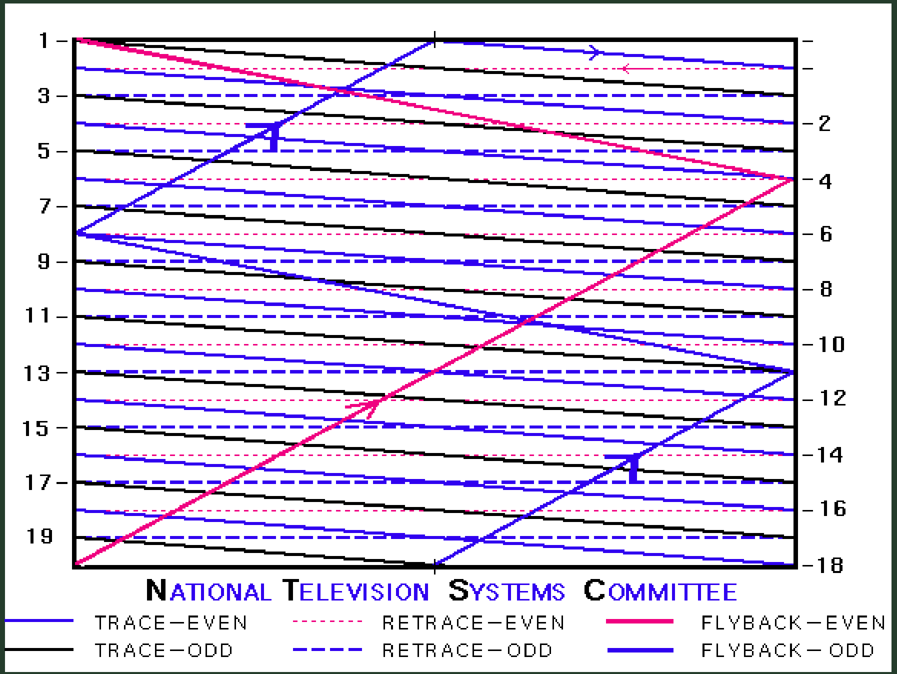
Mechanical Television

- Watching a homemade mechanical-scan television receiver in 1928.
- The "televisor" which produces the picture uses a spinning metal disk with a series of holes in it, called a Nipkow disk, in front of a neon lamp.
- Each hole in the disk passing in front of the lamp produces a scan line which makes up the image.
- The video signal from the television receiver unit (left) is applied to the neon lamp, causing its brightness to vary with the brightness of the image at each point.
- This system produced a dim orange image 1.5 inches (3.8 cm) square, with 48 scan lines, at a frame rate of 7.5 frames per second.

Analog Video

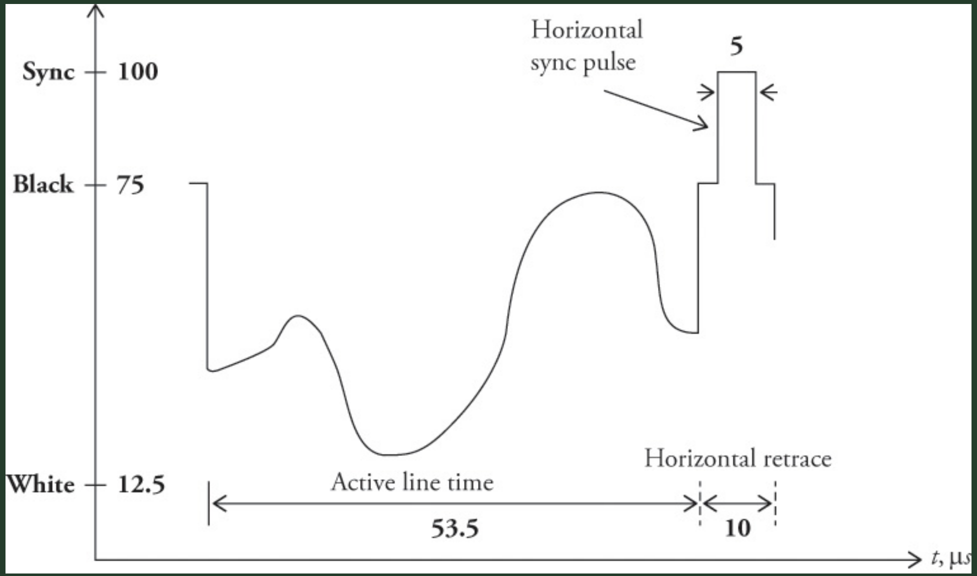
- The analog video signal refers to a 1D electrical signal $f(t)$ of time that is obtained by sampling $S_c(x, y, t)$ in the vertical y and temporal coordinates.
- This periodic sampling process is called scanning
- There are two scanning methods
  - Progressive
    - Scans complete picture frame in Δt seconds
    - Most generally $\Delta t = {}^{1}/_{72}\, sec$
  - Interlaced
    - Even lines scanned first and then the odd lines
    - NTSC is the best example





Analog Video

- An example analog-video signal s(t) is shown in Figure.
- Blanking pulses (black) are inserted during the retrace intervals to blank out retrace lines on the monitor.
- Sync pulses are added on top of the blanking pulses to synchronize the receiver's horizontal and vertical sweep circuits.
- The sync pulses ensure that the picture starts at the top-left corner of the receiving monitor.
- The timing of the sync pulses is, of course, different for progressive and interlaced video.
- Several analog-video signal standards, which are obsolete today, have different image parameters (e.g., spatial and temporal resolution) and differ in the way they handle color.
- These can be grouped as:
    - component analog video
    - composite video
    - S-video (Y/C video)



Analog Video

- Component analog video refers to individual red (R), green (G), and blue (B) video signals.
- Composite-video format encodes the chrominance components on top of the luminance signal for distribution as a single signal that has the same bandwidth as the luminance signal.
- Different composite-video formats, e.g., NTSC (National Television Systems Committee), PAL (Phase Alternation Line), and SECAM (Systeme Electronique Color Avec Memoire), have been used in different regions of the world.
- The composite signal usually results in errors in color rendition, known as hue and saturation errors, because of inaccuracies in the separation of the color signals.
- S-video is a compromise between the composite video and component video, where we represent the video with two component signals, a luminance and a composite chrominance signal.
- The chrominance signals have been based on (I,Q) or (U,V) representation for NTSC, PAL, or SECAM systems. S-video was used in consumer-quality videocasette recorders and analog camcorders to obtain image quality better than that of composite video.
- Cameras specifically designed for analog television pickup from motion picture film were called telecine cameras. They employed frame-rate conversion from 24 frames/sec to 60 fields/sec.

Progressive and Interlaced Scans

- Important aspects in Analog Video are
  - Vertical Resolution
  - Aspect Ratio
  - Frame Rate
- Vertical resolution stands for number of scan lines per frame
- Aspect ratio stands for width to height of the frame
- Frame rate stands for number of frames displayed per second
  - In general, human eye cannot perceive flicker if frame rate > 50
  - Frame rate > 50 requires large bandwidth
  - Thus TVs use interlaced way of displaying images
    - This is because, TV needs high framerate while preserving vertical resolution, meaning number of fames to be transmitted should be high and thus larger bandwidth is required.

Digital Video

- Digital video is audio/visual content in a binary format, with information is presented as a sequence of digital data rather than in a continuous signal as analog information is.
- Information in the natural world, received through the five senses, is analog. That means that it is infinitely variable. Digital A/V information, on the other hand, consists of discrete units of data that are placed so close together that the human senses perceive them as a continuous flow.
- Digital video uses component representation for representing color signal
- Most of the color cameras provide individual RGB components. This avoids artifacts (like blurring due to loss of resolutions) which are in analog video signals.
- Even if it is form a analog video source like video tape, composite video is first converted into component video and then digitized
- Direct conversion is also possible using A/D convertor
- The video in digital format is stored as a series of bitmaps. Bitmaps consist of five main features:
    - Starting address in the memory
    - Number of pixels per line
    - Pitch value (Distance between memory locations between each scan line)
    - Number of lines
    - Number of bits per pixel (for calculating bandwidth)

Analog vs Digital Video

Analog Video	Digital Video
Uses composite video signal	Uses component video signal Then converts each frame into a raster/bitmap which will be displayed
Less bandwidth	More bandwidth as each frame information is to be passed
Loss of resolution will result in bluring	Loss of resolution will result in pixelation (aliasing)
Can be used in analog devices only. Best suited for audio and video transmission.	Best suited for Computing and digital electronics.

https://www.diffen.com/difference/Analog_vs_Digital

Digital Video Standards

Parameter	CCIR601 525/60 NTSC	CCIR601 625/50 PAL/SECAM	CIF
Number of active pels/line Lum (Y) Chroma (U,V)	 720 360	 720 360	 360 180
Number of active lines/pic Lum (Y) Chroma (U,V)	 480 480	 576 576	 288 144
Interlacing	2:1	2:1	1:1
Temporal rate	60	50	30
Aspect ratio	4:3	4:3	4:3

Digital Video Compression

- Based on the bitrates of the network, video is to be compressed and transmitted.
- Some compression standards are shown in figure below

Standard	Application
CCITT G3/G4	Binary images (nonadaptive)
JBIG	Binary images
JPEG	Still-frame gray-scale and color images
H.261	p $\times$ 64 kbps
MPEG-1	1.5 Mbps
MPEG-2	10-20 Mbps
MPEG-4	4.8-32 kbps (underway)

- Interoperability between various devices required standardization of both compression and representation techniques. Some standard proprietary compression and representation systems are listed below:

Video Format	Company
DVI (Digital Video Interactive), Indeo	Intel Corporation
QuickTime	Apple Computer
CD-I (Compact Disc Interactive)	Philips Consumer Electronics
Photo CD	Eastman Kodak Company
CDTV	Commodore Electronics

Digital Video Representation Benefits

i) Open architecture video systems, meaning the existence of video at multiple spatial, temporal, and SNR resolutions within a single scalable bitstream.

ii) Interactivity, allowing interruption to take alternative paths through a video database, and retrieval of video.

iii) Variable-rate transmission on demand.

iv) Easy software conversion from one standard to another.

v) Integration of various video applications, such as TV, videophone, and so on, on a common multimedia platform.

vi) Editing capabilities, such as cutting and pasting, zooming, removal of noise and blur.

vii) Robustness to channel noise and ease of encryption.

Video Capturing Technologies

- Acquisition of image:
  - Machine vision is the creation of an image and the collection of data derived from the image, and the subsequent processing and interpretation of the data by a computer from some useful application. Machine vision is also known as computer vision, and its principal application is in industrial inspection. Machine vision is concerned with the creation, and collection of data from an image, all subsequently processed and interpreted by a computer for some useful application.

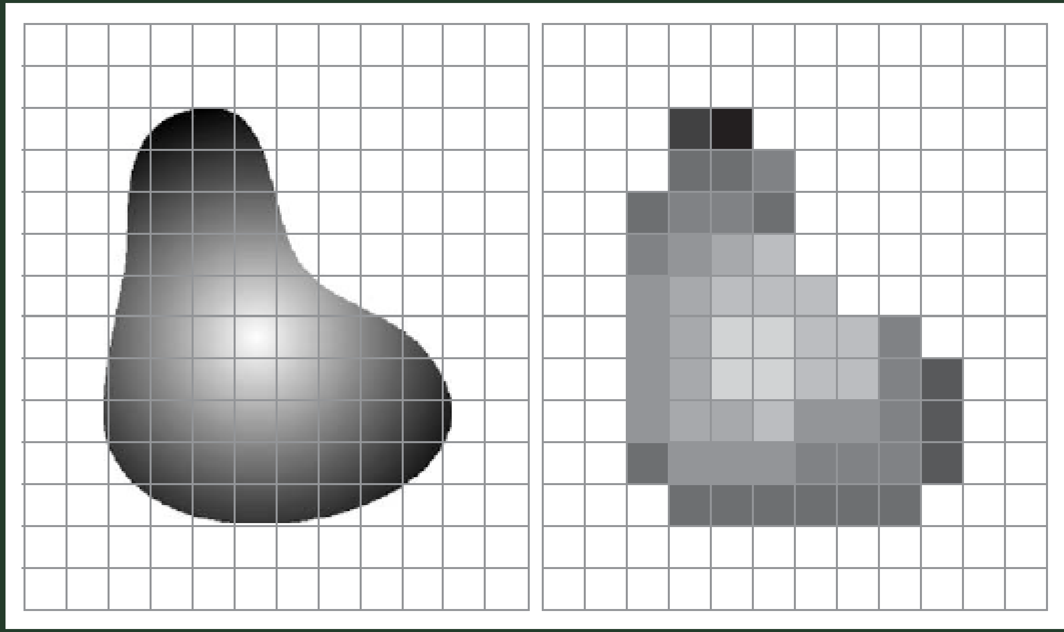
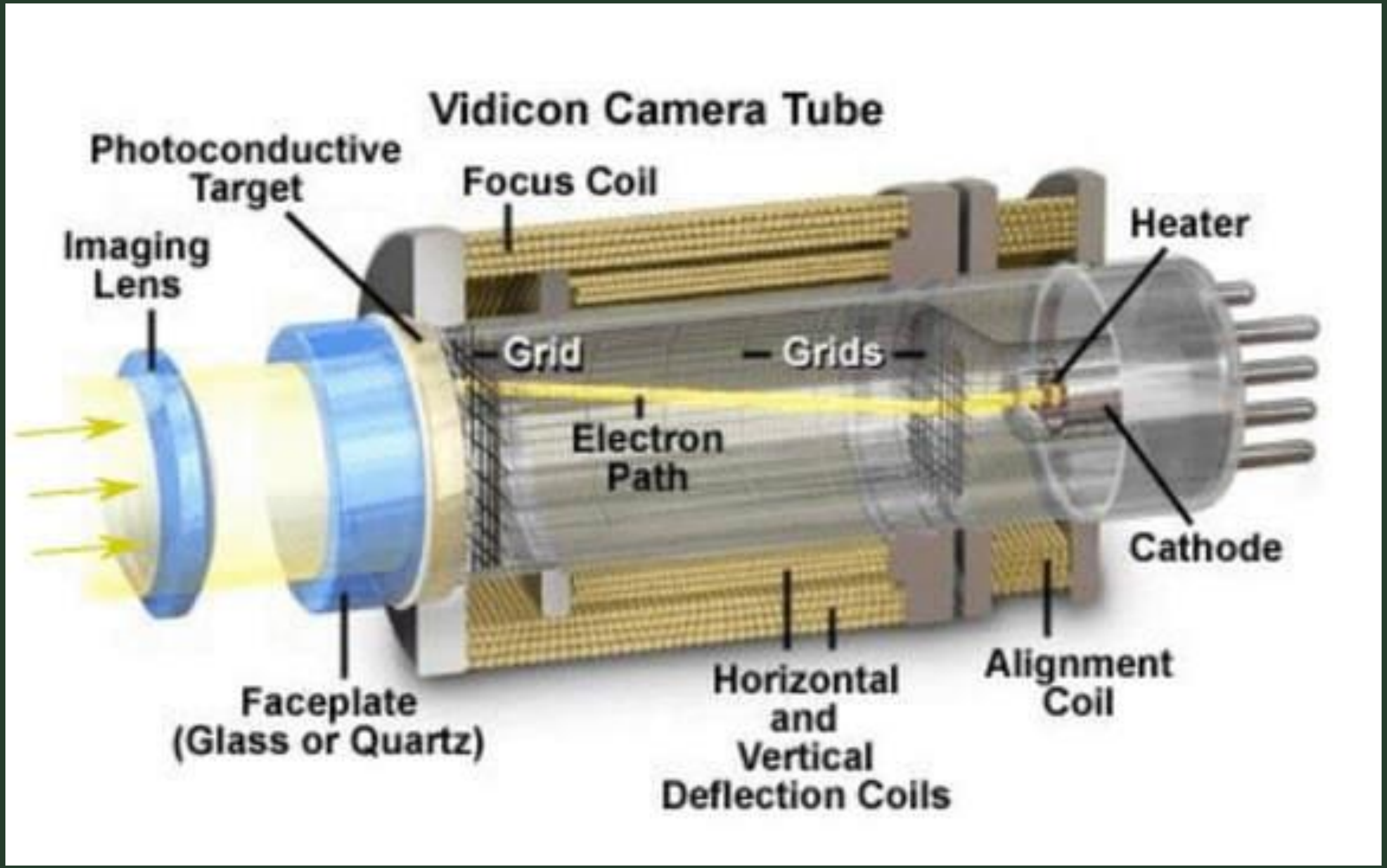
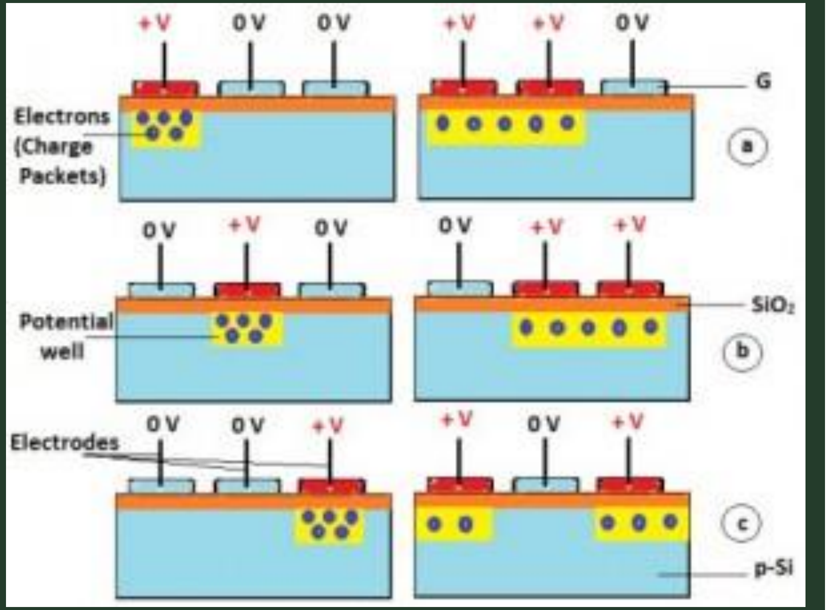
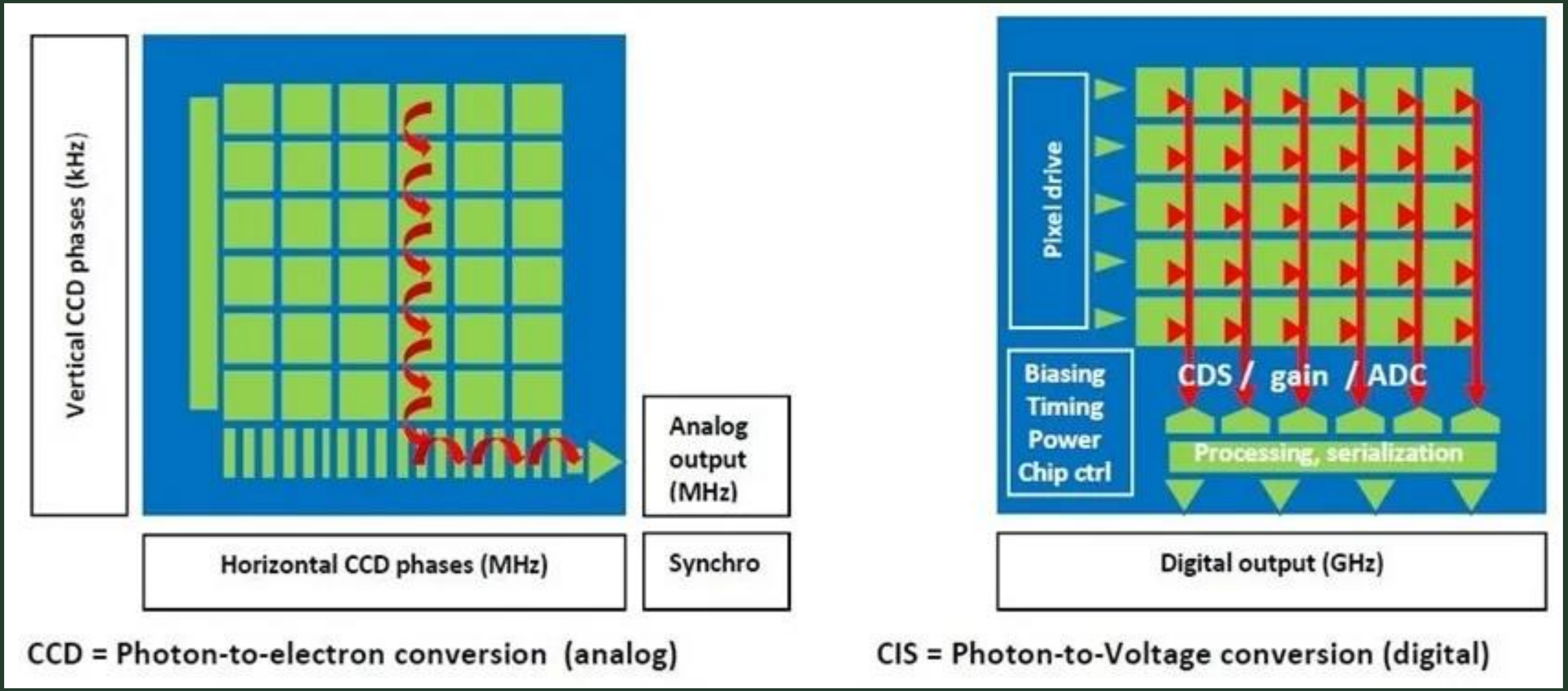


Image Sampling and Quantization



Imaging Sensors





Imaging Sensors

Characteristic	CCD	CMOS
Signal from pixel	Electron packet	Voltage
Signal from chip	Analog Voltage	Bits (digital)
Readout noise	low	Lower at equivalent frame rate
Fill factor	High	Moderate or low
Photo-Response	Moderate to high	Moderate to high
Sensitivity	High	Higher
Dynamic Range	High	Moderate to high
Uniformity	High	Slightly Lower
Power consumption	Moderate to high	Low to moderate

In [ ]:
```
!dir
```

In [ ]:
```
#IF OPENCV IS NOT INSTALLED, EXECUTE THE BELOW COMMAND
!pip install opencv-python
!pip install opencv-contrib-python
#INSTALL FFPLAYER
!pip install ffpyplayer
```

In [ ]:
```
!pip install tqdm
```

In [2]:
```
import cv2
#from ffpyplayer.player import MediaPlayer

import os
from tqdm import tqdm
```

In [ ]:
```
file = 'TestVideo.mp4'
```

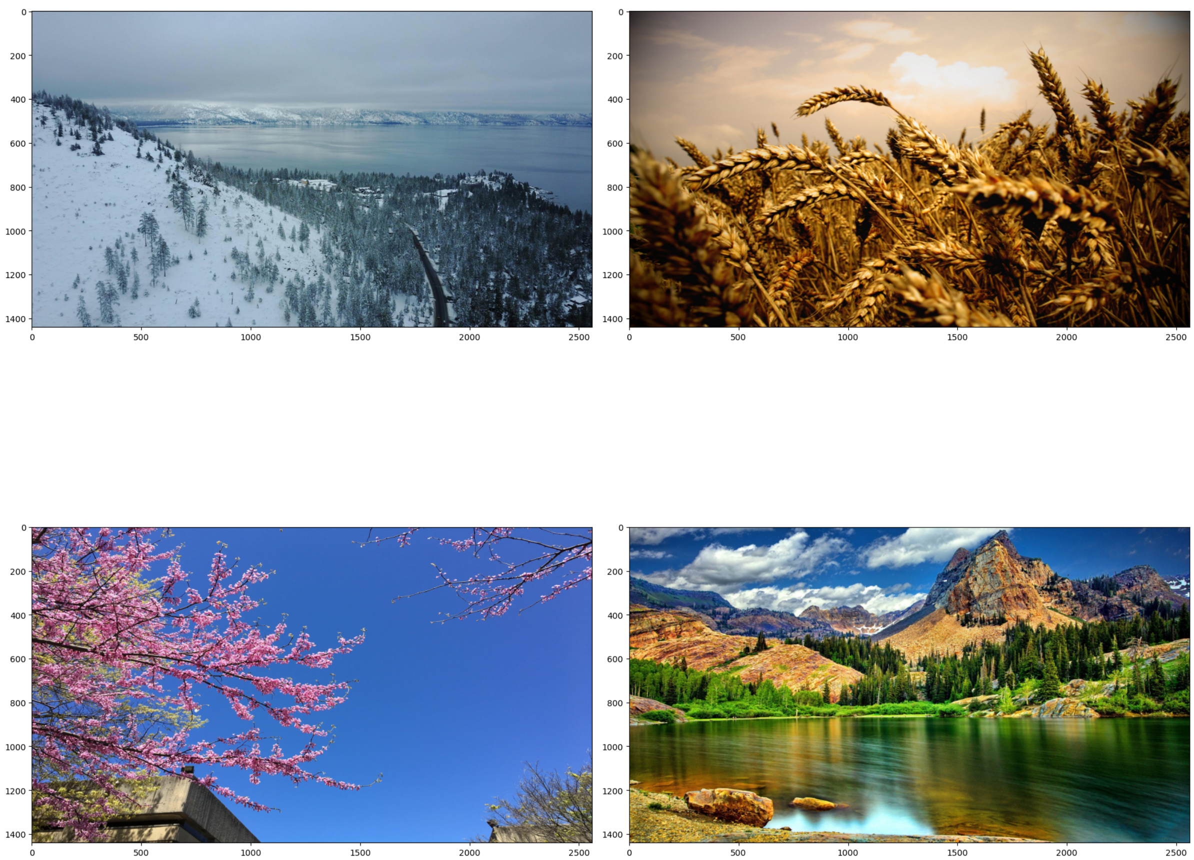
Reading, Playing and Extracting a frame from Video

In [ ]:
```
video = cv2.VideoCapture(file)

while video.isOpened():
    ret, frame = video.read()

    if not ret:
        print("End of video")
        break

    cv2.imshow("Video", frame)

    if cv2.waitKey(1) == ord("s"):
        cv2.imwrite("ExtractedFrame.jpg", frame)
        print("Frame extracted.....")

    if cv2.waitKey(1) == ord("q"):
        print("Program break...")
        break

video.release()
cv2.destroyAllWindows()
```

Creating a Video

Creating Frames for Video Creation

In [12]:
```
im1 = cv2.imread('1.jpg')
im1=cv2.cvtColor(im1, cv2.COLOR_BGR2RGB)
im2 = cv2.imread('2.jpg')
im2=cv2.cvtColor(im2, cv2.COLOR_BGR2RGB)
im3 = cv2.imread('3.jpg')
im3=cv2.cvtColor(im3, cv2.COLOR_BGR2RGB)
im4 = cv2.imread('4.jpg')
im4=cv2.cvtColor(im4, cv2.COLOR_BGR2RGB)
im2=cv2.resize(im2, (im1.shape[1], im1.shape[0]))
im3=cv2.resize(im3, (im1.shape[1], im1.shape[0]))
im4=cv2.resize(im4, (im1.shape[1], im1.shape[0]))
fig,ax = plt.subplots(nrows = 2, ncols=2, figsize=[20,20])
ax[0][0].imshow(im1)
ax[0][1].imshow(im2)
ax[1][0].imshow(im3)
ax[1][1].imshow(im4)
fig.tight_layout()
plt.show()
```



In [13]:
```
path = '.\\frames\\'
for i in tqdm(range(0,120)):
    alpha = round(i/120, 3)
    tmp = cv2.addWeighted(im1, 1-alpha, im2, alpha,0)
    cv2.imwrite(path + str(i) + '.jpg', tmp)

for i in tqdm(range(120,240)):
    alpha = round((i-120)/120, 3)
    tmp = cv2.addWeighted(im2, 1-alpha, im3, alpha,0)
    cv2.imwrite(path + str(i) + '.jpg', tmp)

for i in tqdm(range(240,360)):
    alpha = round((i-240)/120, 3)
    tmp = cv2.addWeighted(im1, 1-alpha, im2, alpha,0)
    cv2.imwrite(path + str(i) + '.jpg', tmp)
```

```
100%|██████████| 120/120 [00:08<00:00, 14.01it/s]
100%|██████████| 120/120 [00:09<00:00, 12.77it/s]
100%|██████████| 120/120 [00:09<00:00, 12.70it/s]
```

In [4]:
```
imgs = []
path = '.\\frames\\'
for fl in tqdm(os.listdir(path)):
    #tmp = cv2.imread(path + fl)
    #tmp = cv2.cvtColor(im3, cv2.COLOR_BGR2RGB)
    imgs.append(cv2.imread(path + fl))
print(str(len(imgs)) + ' images loaded')
```

```
100%|██████████| 360/360 [00:19<00:00, 18.01it/s]

360 images loaded
```

Creating a Video from images

In [5]:
```
out = cv2.VideoWriter('output_video.avi',cv2.VideoWriter_fourcc(*'DIVX'),
                      25, (imgs[0].shape[1], imgs[0].shape[0]))
for im in tqdm(imgs):
    out.write(im)

out.release()
```

```
100%|██████████| 360/360 [00:08<00:00, 43.88it/s]
```

In [1]:
```
!start output_video.avi
```