**DEPARTMENT OF COMPUTER & SOFTWARE ENGINEERING**

**COLLEGE OF E&ME, NUST, RAWALPINDI**

**Subject Name**

**Digital Signal Processing**

**Lab Number**

**1**

**SUBMITTED TO:**

**LE Sundas Ashraf**

**SUBMITTED BY:**

**Student Name**

1. Wahaaj Nasir

**Reg#413238**

**DE- 44 Dept C&SE**

**Objectives:**

Processing in MATLab

**Related Topic/Chapter in theory class:**

Basics Of Digital Signal Processing

**Hardware/Software required:**

Hardware: PC

Software Tool: MATLab

**Task 1:**

**Record a voice with default sampling rate and perform following operations:**

**(a)**

**Solution:**

%Task 1%

%Sampling frequency Fs is no of samples per second

recorder = audiorecorder(8000, 8, 1);

fs = 8000 %Adding fs as variable becoz mei har jagah use kar raha

disp("Start Recording");

recordblocking(recorder, 5);

disp("Stop Recording");

play(recorder)

myVoice = getaudiodata(recorder);

reverse\_audio = flipud(myVoice);

pause(7)

sound(reverse\_audio, 8000)

pause(7);

subplot(3, 1, 1)

plot(myVoice)

xlabel("Samples");

ylabel("Amplitude");

title("Original Sound")

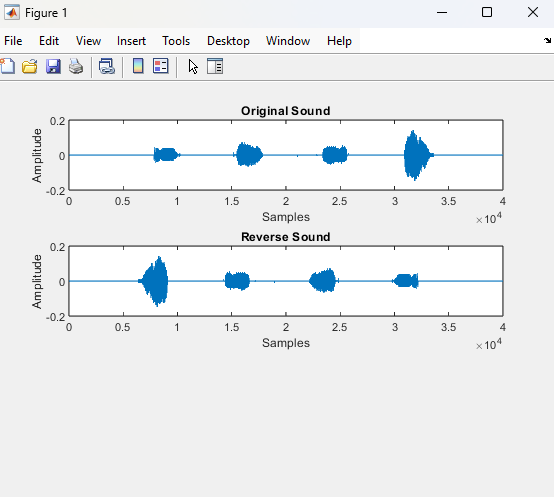
subplot(3, 1, 2)

plot(reverse\_audio)

xlabel("Samples");

ylabel("Amplitude");

title("Reverse Sound")

****

**The audiorecorder records the voice and outputs it as a column matrix. This is done flipped using the ‘flipud’ command.**

**(b)**

**Add 2 cosines into the audio with frequency ranging from 1kHz to 1.5Hz, play plot the signal using subplot.**

**Solution**

%Task2

f1 = 1000;

f2 = 1500;

%Determine length of the signal

N = length(myVoice);

t = (0:N-1)/fs; %Goes to 5 seconds

cos\_sig\_1 = cos(2 \*pi \* f1 \*t)';

cos\_sig\_2 = cos(2 \*pi \*f2 \*t)';

cos\_added\_voice = myVoice + cos\_sig\_1 + cos\_sig\_2;

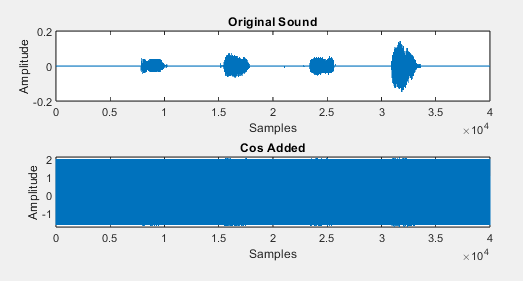
sound(cos\_added\_voice, 8000)

plot(cos\_added\_voice)

xlabel("Samples");

ylabel("Amplitude");

title("Cos Added")



**Due to the high frequency of the cosine function, the original sound is lost.**

**(c)**

**Divide the array Of your audio into two equal parts, save the divided arrays into two**

**variables (let say a and b) and add both (a+b), play the resultant voice and plot the signal**

**using subplot.**

**Solution**

%Task 3

partA = myVoice(1:20000, 1);

partB = myVoice(20001:end, 1);

sound(partA, 8000)

pause(4)

sound(partB, 8000)

pause(4)

reconstruct = partA+partB;

sound(reconstruct, fs)

subplot(4, 1, 1)

plot(partA);

title("Part A")

xlabel("Samples")

ylabel("Amplitude")

subplot(4, 1, 2)

plot(partB);

title("Part B")

xlabel("Samples")

ylabel("Amplitude")

subplot(4, 1, 3)

plot(reconstruct);

title("Reconstruct")

xlabel("Samples")

ylabel("Amplitude")

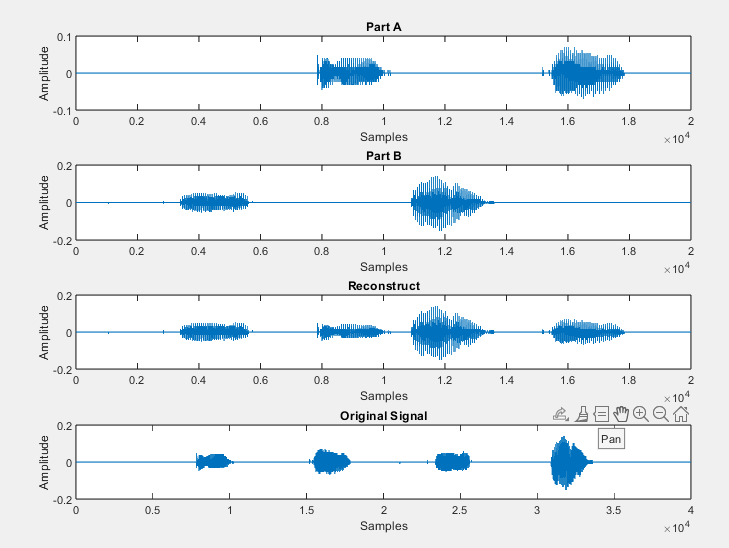
subplot(4, 1, 4)

plot(myVoice);

title("Original Signal")

xlabel("Samples")

ylabel("Amplitude")



**As you can see, we separate Part A and Part B from the original signal. Adding them increases the amplitude of the signals.**

**(d)**

**Perform [(a\*2) + (b\*0.5)] play the resultant voice and plot the signal using subplot.**

**Solution**

%Task 4

partA\_2 = partA .\* 2;

partB\_0\_5 = partB .\* 0.5;

recon = partA\_2 + partB\_0\_5;

sound(recon, fs)

subplot(4, 1, 1)

plot(partA\_2);

title("Part A\_2")

xlabel("Samples")

ylabel("Amplitude")

subplot(4, 1, 2)

plot(partB\_0\_5);

title("Part B\_0\_5")

xlabel("Samples")

ylabel("Amplitude")

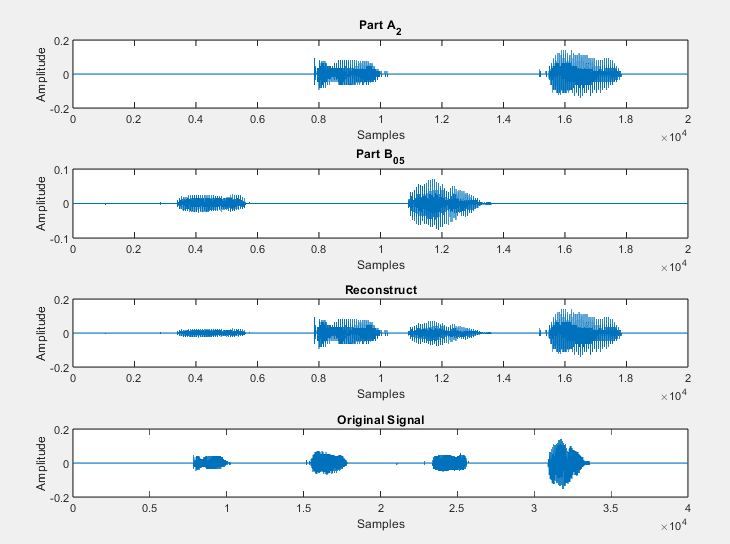
subplot(4, 1, 3)

plot(recon);

title("Reconstruct")

xlabel("Samples")

ylabel("Amplitude")

****

**From the figure and sound byte, we see that A has a louder voice while the B part is quieter.**

**(e)**

**Add 4 cosines with frequency ranging from (5Hz to 4kHz) into audio signal, play the**

**resultant voice and plot the signal using subplot.**

**Solution**

%Task 5

f1 = 5;

f2 = 200;

f3 = 1000;

f4 = 2500;

%Determine length of the signal

N = length(myVoice);

t = (0:N-1)/fs; %Goes to 5 seconds

cos\_sig\_1 = cos(2 \*pi \* f1 \*t)';

cos\_sig\_2 = cos(2 \*pi \*f2 \*t)';

cos\_sig\_3 = cos(2 \*pi \*f3 \*t)';

cos\_sig\_4 = cos(2 \*pi \*f4 \*t)';

new\_sound = myVoice + cos\_sig\_1 + cos\_sig\_2 + cos\_sig\_3 + cos\_sig\_4;

subplot(2, 1, 1)

plot(myVoice)

title("Original Sound")

xlabel("Sample")

ylabel("Amplitude")

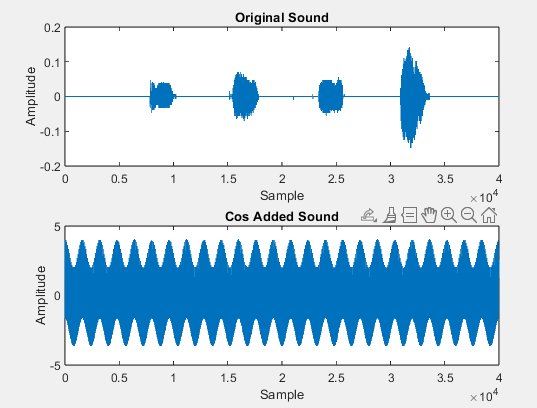
subplot(2, 1, 2)

plot(new\_sound)

title("Cos Added Sound")

xlabel("Sample")

ylabel("Amplitude")



**Using a lower frequency we can see the effect on the signal**

**(f)**

**Downsample the audio signal 5 times, dropping every other sample**

**Solution**

%Task 6

downsampleVoice\_1 = downsample(myVoice, 2); %Divide by 2

downsampleVoice\_2 = downsample(downsampleVoice\_1, 2); %Divide by 2

downsampleVoice\_3 = downsample(downsampleVoice\_2, 2); %Divide by 2

downsampleVoice\_4 = downsample(downsampleVoice\_3, 2); %Divide by 2

downsampleVoice\_5 = downsample(downsampleVoice\_4, 2); %Divide by 2

subplot(6, 1, 1)

plot(myVoice)

title("Original Sound")

xlabel("Sample")

ylabel("Amplitude")

subplot(6, 1, 2)

plot(downsampleVoice\_1)

title("Downsample Once")

xlabel("Sample")

ylabel("Amplitude")

subplot(6, 1, 3)

plot(downsampleVoice\_2)

title("Downsample Twice")

xlabel("Sample")

ylabel("Amplitude")

subplot(6, 1, 4)

plot(downsampleVoice\_3)

title("Downsample Thrice")

xlabel("Sample")

ylabel("Amplitude")

subplot(6, 1, 5)

plot(downsampleVoice\_4)

title("Downsample Four Times")

xlabel("Sample")

ylabel("Amplitude")

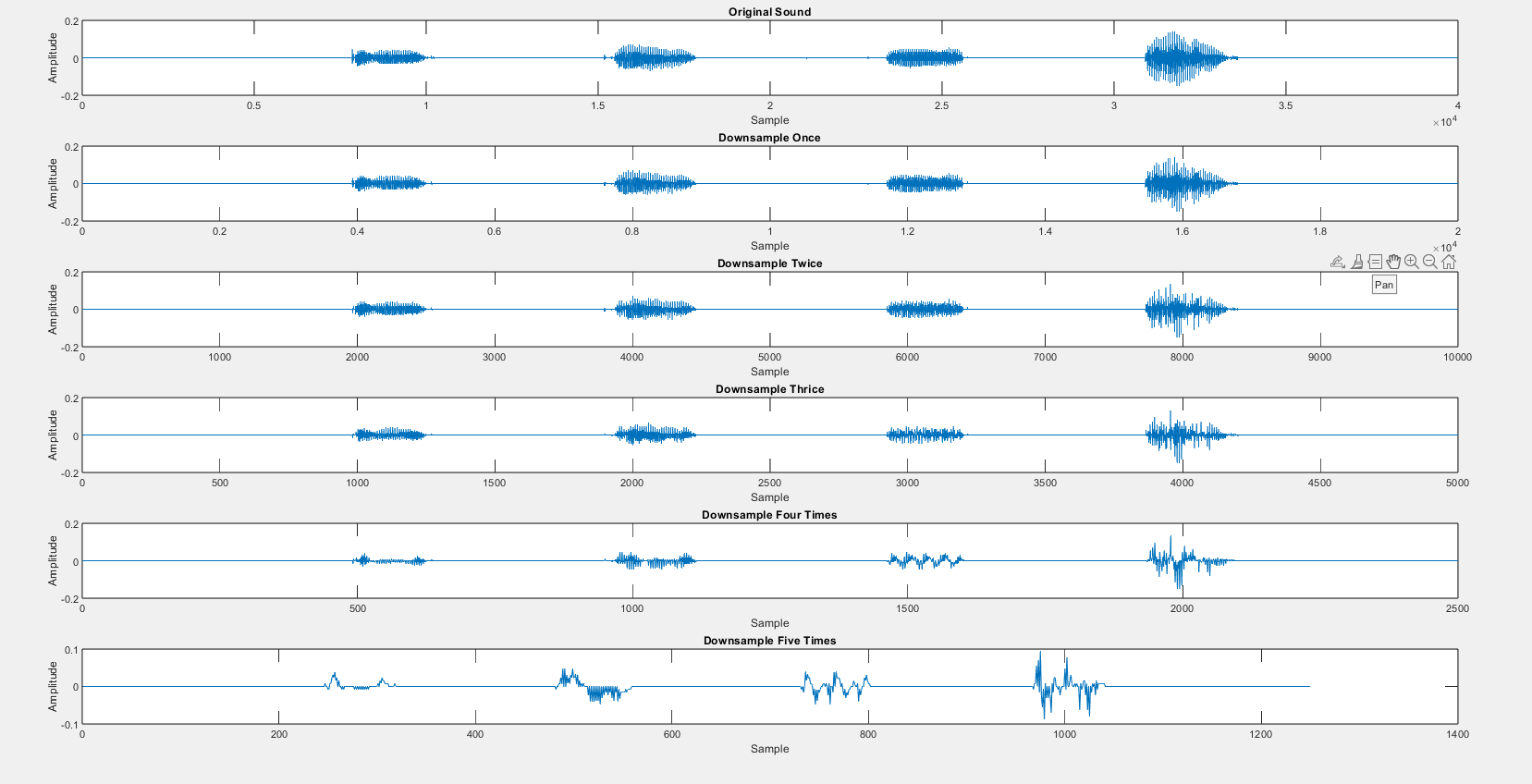
subplot(6, 1, 6)

plot(downsampleVoice\_5)

title("Downsample Five Times")

xlabel("Sample")

ylabel("Amplitude")

****

**The samples are reduced by a factor of 2 for 5 times**

**Task 2**

**Take two gray-scale images of same size and perform following operations:**

**(a)Read the images in two matrices.**

**(b)Take transpose Of the matrix and display the resultant images using subplot.**

**(c)Add two images and display the result.**

**(d)Read an image then multiply matrix of image with 0.5 and display the result.**

**Solution**

%%Task 2 - Images

Cat = imread("Cat.jpg");

Cat1\_Gray = rgb2gray(Cat);

Cat2 = imread("Cat 2.jpg");

Cat2\_Gray = rgb2gray(Cat2);

subplot(2, 2, 1)

imshow(Cat)

title("Cat 1")

subplot(2, 2, 2)

imshow(Cat2)

title("Cat 2")

subplot(2, 2, 3)

imshow(Cat1\_Gray)

title("Cat 1")

subplot(2, 2, 4)

imshow(Cat2\_Gray)

title("Cat 2")

%%

Cat1\_transpose = (Cat1\_Gray)';

Cat2\_transpose = (Cat2\_Gray)';

Cat\_Add = Cat1\_Gray + Cat2\_Gray;

Cat\_1\_Mult = Cat1\_Gray .\* 0.5;

subplot(2, 2, 1)

imshow(Cat1\_transpose)

title("Cat 1 Transpose")

subplot(2, 2, 2)

imshow(Cat2\_transpose)

title("Cat 2 Transpose")

subplot(2, 2, 3)

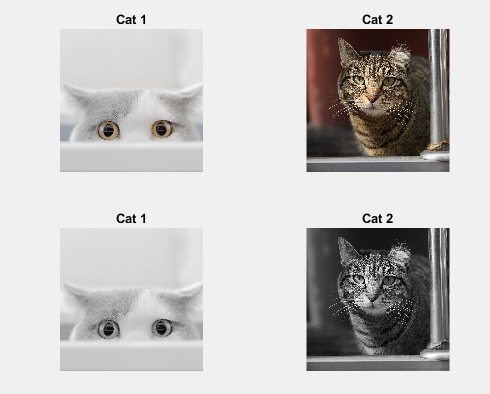
imshow(Cat\_Add)

title("Added Two Images")

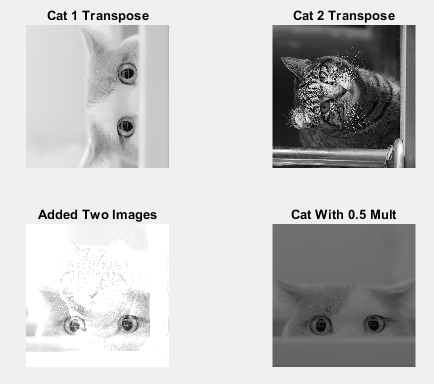
subplot(2, 2, 4)

imshow(Cat\_1\_Mult)

title("Cat 2")



**Converting to Grayscale using rgb2gray**

****

**Task 3**

**Read an video and display it using imread**

**Solution**

%Task 3

Video = VideoReader("Video.mp4")

numFrames = ceil(Video.Duration \* Video.FrameRate);

frames=cell(numFrames, 1)

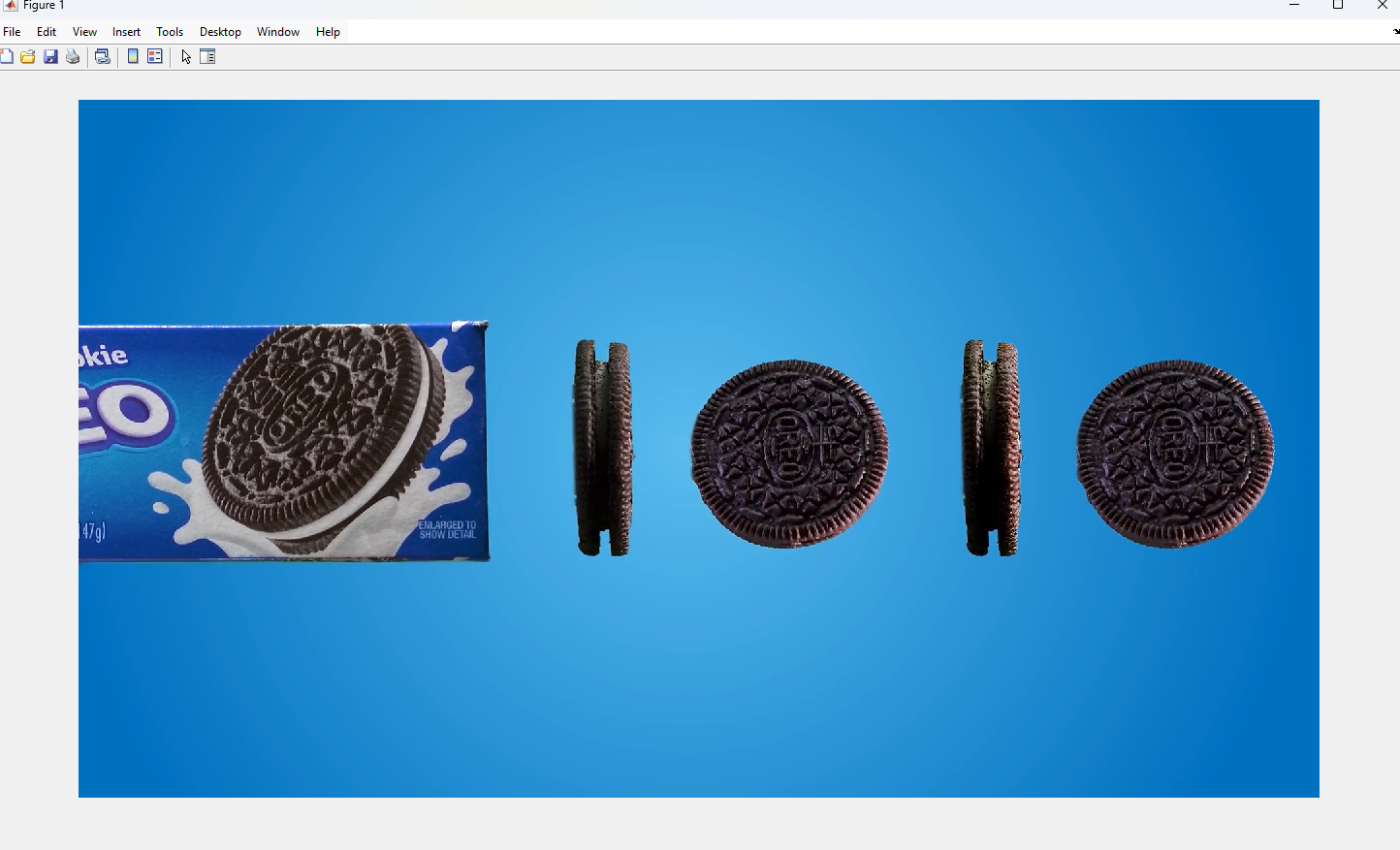
for k=1:numFrames

frame = read(Video, k);

frames{k} = frame;

imshow(frame)

end



**Using a for loop configure for the framerate of the video we can display the video**

**Home Task**

**Reverse the frames and play the video in reverse**

**Solution**

%%

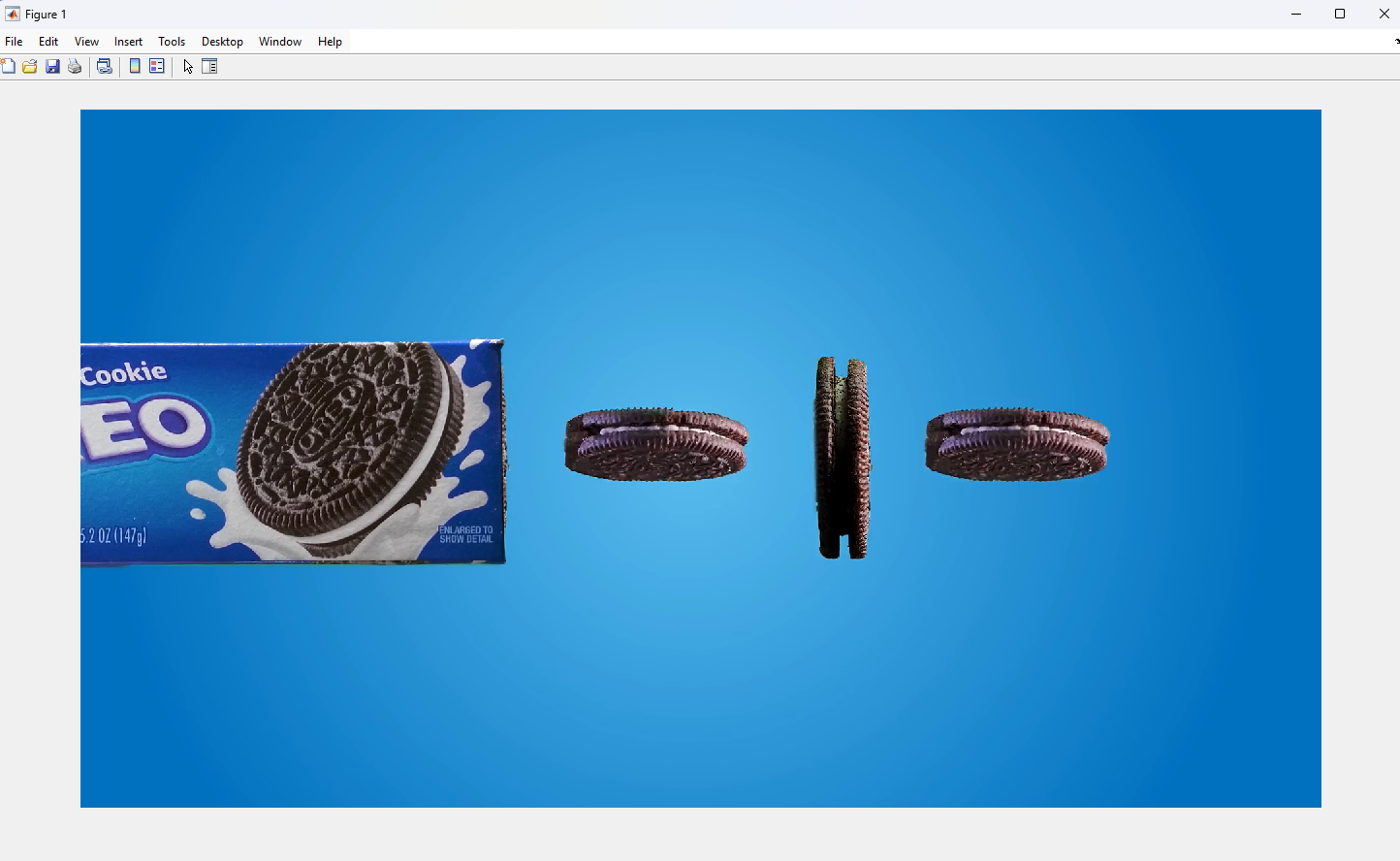
rev\_vid = frames(end:-1:1);

for k =1:numFrames

frame =rev\_vid{k};

imshow(frame)

end

****

**We use the frame array of cells and reverse the video**