EULERIAN VIDEO MAGNIFICATION USING OPENCV-PYTHON FOR MICRO MOTION ANALYSIS

A Project Submitted To

UNIVERSITY OF MADRAS

In partial fulfillment of the requirements

for the award of

MASTER OF SCIENCE

In

PHYSICS

By

BHUVANESHWARAN M

(Reg.No:2113182072009)

Under the guidance of

CAPTAIN M. ANEES AHMED

THE HEAD & ASSOCIATE PROFESSOR,

PG AND RESEARCH DEPARTMENT OF PHYSICS

THE NEW COLLEGE

CHENNAI – 600 014.

YA ALLAH
INCREASE MY KNOWLEDGE

PG AND RESEARCH DEPARTMENT OF PHYSICS

THE NEW COLLEGE CHENNAI – 600 014

APRIL 2023

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **"EULERIAN VIDEO MAGNIFICATION USING OPENCV-PYTHON FOR MICRO MOTION ANALYSIS"** submitted by **BHUVANESHWARAN M (Reg.No:2113182072009)** in partial fulfillment of the requirements for the award of the degree of **Master of Science in Physics,** for the year 2021 - 2023 is a bonafide work carried out in the **PG AND RESEARCH DEPARTMENT OF PHYSICS, THE NEW COLLEGE , Royapettah, Chennai - 600 014.**

CAPTAIN M. ANEES AHMED

The Head & Associate Professor,

PG and Research Department of Physics,

The New College,

Chennai – 600 014.

BONAFIDE CERTIFICATE

This is to certify that the project report entitled **"EULERIAN VIDEO MAGNIFICATION USING OPENCV-PYTHON FOR MICRO MOTION ANALYSIS"** submitted by **BHUVANESHWARAN M (Reg.No:2113182072009)** in partial fulfillment of the requirements for the award of the degree of **Master of Science in Physics,** for the year 2021 - 2023 is a bonafide work carried out in the **PG AND RESEARCH DEPARTMENT OF PHYSICS, THE NEW COLLEGE, Royapettah, Chennai - 600 014.**

Guide:

CAPTAIN M. ANEES AHMED

The Head & Associate Professor,

PG and Research Department of Physics,

The New College,

Chennai – 600 014.

DECLARATION

I do hereby declare that the project work entitled **"EULERIAN VIDEO MAGNIFICATION USING OPENCV-PYTHON FOR MICRO MOTION ANALYSIS"** submitted for the degree of **Master of Science in Physics**, is a record of original work done by me during 2021-2023 under the supervision and guidance of **CAPTAIN M. ANEES AHMED, The Head & Associate Professor,** PG & Research Department of Physics, The New College, Chennai – 600 014. No part of this has been previously formed the basis for the award of any other Degree, Diploma or other similar title to any candidate in any university.

PLACE : Chennai

DATE : **(BHUVANESHWARAN M)**

ACKNOWLEDGEMENT

First of all, thank Lord Almighty for all graces showered on me, throughout the course of my project. It is my honor to owe my gratitude to the management of **The New College** and to the **PG & Research Department of Physics** who enrolled me to its crown.

I am grateful to thank **CHAIRMAN, SECRETARY, TREASURER,** management of **MEASI, Dr. S. BASHEER AHAMED, The PRINCIPAL,** The **VICE PRINCIPALS Prof. S. A. SHEIK MOHAMED** (academic), **Dr. V. KAMAL NASIR** (admin) of **The New College,** Chennai, for always being supportive to the research team and also for giving me an opportunity to work in my project with ease.

I take this opportunity to thank **Captain M. ANEES AHMED, HEAD OF THE DEPARTMENT**, for providing me with the available facilities and for helping me to precede my work with great interest.

I would like to express my deepest appreciation towards **Prof. A. MOHAMED HUSSAIN,** Coordinator, PG & Research department of physics, The New College, Chennai, for providing me with all the facilities and enabling me to do my work with greatest interest and ease.

It is my humble gratitude to my internal guide **CAPTAIN M. ANEES AHMED, The Head & Associate Professor,** PG & Research Department of Physics, The New College, Chennai, for his wonderful guidance throughout the completion of my project and guiding me in a right path.

I express my sincere thanks to **Dr. G. FOIZE AHMED, Dr. S. G. MOHAMMED HUSSAIN, Dr. L. ALLWIN JOSEPH, Dr. J. PRINCE JOSHUA** and **Dr. I. MOHAMMED ZAHID** for their valuable instructions that helped a lot in my work and I am very much grateful in thanking all the staff members of shift - I and lab assistants for the boundless efforts they had taken to make my project successful.

Last but not least, I am extremely grateful to my parents, well-wishers and friends for their remarkable support and prayers.

(BHUVANESHWARAN M)

CONTENTS

CHAPTER 1

Introduction

1.1. What is Eulerian Video Magnification?

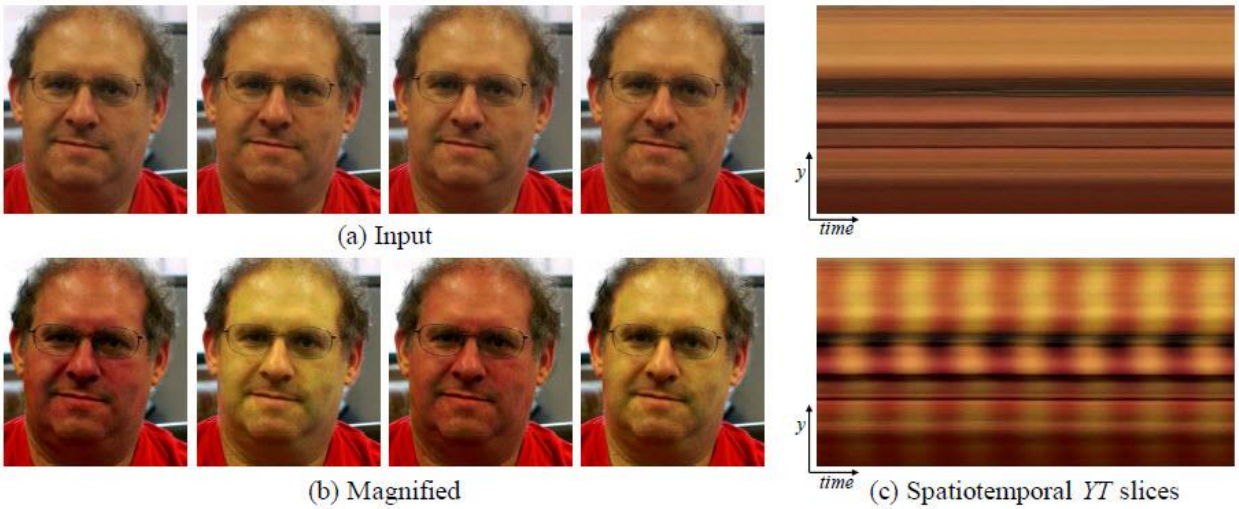
Eulerian motion amplification is a technique that enables the amplification of subtle movements in video sequences that may not be visible to the naked eye. This method has broad applications in diverse fields, including medicine, structural engineering, and other areas of scientific research. It operates by separating the temporal and spatial components of a video signal. The temporal component represents changes in pixel values over time, while the spatial component represents the image's pixel values at a specific point in time. By filtering these components, Eulerian motion amplification enables the enhancement of movements within specific frequency ranges.

1.2 Brief History of Eulerian Video Magnification

Eulerian Video Magnification is a computational technique that was first introduced by MIT researchers in 2012. The technique is based on the principles of motion magnification, which has been a topic of research in computer vision and signal processing for several decades. However, the Eulerian approach offered a novel solution to the problem of amplifying subtle motions by analyzing the spatial and temporal information present in video sequences.

The original research paper presented several applications of Eulerian Video Magnification, including the ability to detect subtle variations in physiological signals such as pulse and breathing rate from videos of human subjects. Since then, the technique has been used in a wide range of fields, including medical imaging, robotics, and remote sensing.

Over the years, researchers have continued to refine and extend the original approach, developing variants such as phase-based amplification and multiscale Eulerian video magnification. These advancements have expanded the scope of the technique and enabled its use in a broader range of applications.



An example of using Eulerian Video Magnification framework for visualizing the human pulse. (a) Four frames from the original video sequence. (b) The same four frames with the subject's pulse signal amplified. (c) A vertical scan line from the input (top) and output (bottom) videos plotted over time shows how our method amplifies the periodic color variation. In the input sequence the signal is imperceptible, but in the magnified sequence the variation is clear.

CHAPTER 2

A Technical Overview of Eulerian Video Magnification

2.1 How Eulerian Video Magnification works

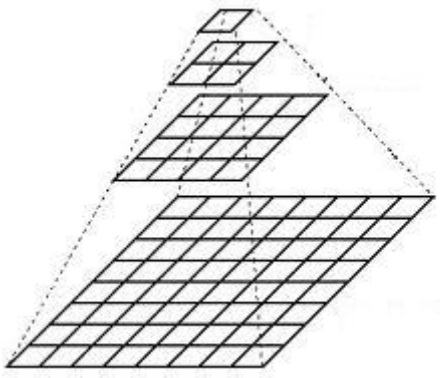
Eulerian video magnification works by analyzing the temporal variations in pixel intensity in a video sequence. This is achieved through the use of a Laplacian pyramid decomposition technique, where a video sequence is decomposed into a series of low-pass filtered and down sampled images, known as the Gaussian pyramid, and a series of high-pass filtered images, known as the Laplacian pyramid.

The Laplacian pyramid consists of images that represent the difference between two adjacent levels of the Gaussian pyramid. This allows for the extraction of high-frequency information in the video sequence, which corresponds to motion and other dynamic changes in the scene.

To magnify these subtle changes, a bandpass filter is applied to the Laplacian pyramid images, which selectively amplifies the desired frequency range of motion. This filtered signal is then multiplied by a user-defined amplification factor to increase the magnitude of the motion. The resulting amplified signal is added back to the original video sequence to create the final magnified output.

2.2 Building Gaussian Pyramid

A Gaussian pyramid is a multi-scale image representation in which an image is recursively filtered and subsampled to generate a sequence of progressively smaller images. The filter used is typically a Gaussian filter, which is a low-pass filter that attenuates high-frequency components in the image.

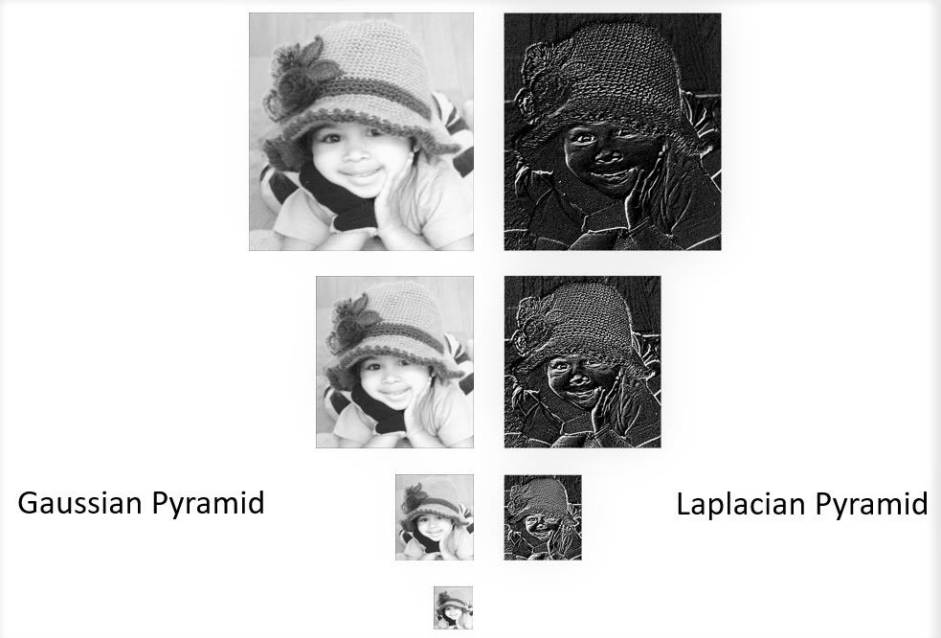


Each level of the pyramid represents an image that has been smoothed and subsampled by a factor of two compared to the previous level. The resulting image at each level is therefore a blurred and down sampled version of the original image, where the number of blurring increases with each level. The pyramid is often visualized as a stack of

images, where the top level is the original image and the bottom level is the smallest image in the pyramid.

2.3 Building Laplacian Pyramid

A Laplacian pyramid is a multi-scale image representation that decomposes an image into a set of bandpass images that capture different levels of detail. The pyramid is named after the Laplacian operator, which is a second-order differential operator that is often used in image processing to detect edges and other features.



The Laplacian pyramid is constructed by first generating a Gaussian pyramid of the original image, as described in the previous answer. Each level of the Gaussian pyramid is then expanded back to its original size and subtracted from the next higher level of the pyramid, resulting in a set of bandpass images that represent the difference in detail between adjacent levels of the Gaussian pyramid.

The resulting Laplacian pyramid is a set of images that represent the details of the original image at different scales. Each level of the pyramid contains the high-frequency components that are not present in the lower levels of the pyramid, while the lowest level of the pyramid contains the low-frequency components of the original image.

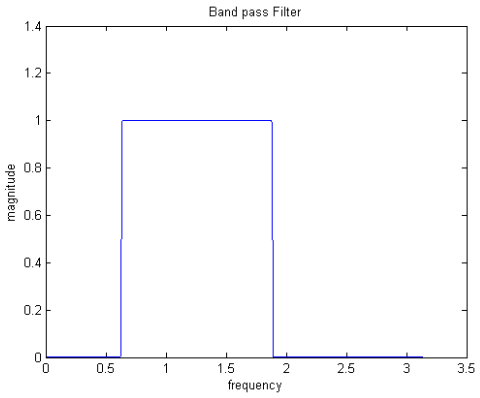
2.4 Temporal Band Pass Filters

A bandpass filter is a type of filter that allows signals within a specific frequency range to pass through while attenuating (weakening) signals outside of that range.

In particular, the bandpass filter is applied to each level of the Laplacian Pyramid that has been computed for the video frames. This filtering helps to remove low-frequency content, such as background motion, and high-frequency content, such as noise, while keeping the relevant motion information in the mid-frequency range. By amplifying the filtered signal and then recombining it with the original video frames, the motion information is magnified, making it more apparent for analysis.

2.4.1 Ideal Band Pass filter

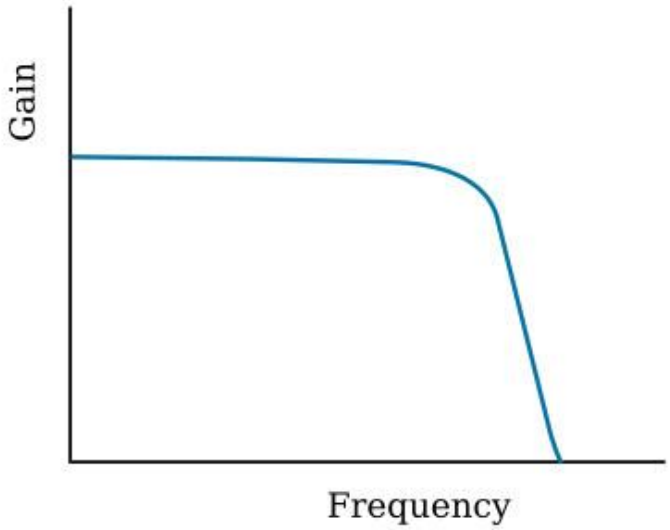
A temporal ideal bandpass filter is a digital signal processing technique used to selectively filter out unwanted frequencies from a time series signal. It is designed to allow a specific frequency range to pass through while attenuating all other frequencies outside that range. The filter is called "ideal" because it has a sharp cut-off and perfectly preserves the frequency components within the specified band.



The ideal bandpass filter is characterized by two parameters, the lower and upper cut-off frequencies, which define the frequency range of interest. Using that we can limit the range of frequency which undergoes motion or colour magnification.

2.4.2 Butterworth band pass filter

The Butterworth filter is designed to have a maximally flat response in the passband, which means that it attenuates frequencies outside the passband while introducing minimal distortion to the frequencies within the passband. The filter is characterized by two parameters, the lower and upper cutoff frequencies, which define the range of frequencies to be passed.



The order of the Butterworth filter determines the steepness of the transition from the passband to the stopband. Higher-order filters have a steeper transition but require more computational resources.

2.5 Spatio-Temporal Slicing

Spatio-temporal time slicing is a technique used in video analysis and processing to extract specific slices or subsets of data from a video stream. It involves selecting a specific region of interest in a video frame (spatial slice) and extracting the data for that region across all frames of the video (temporal slice).

This technique can be used to analyze and process videos in a more efficient way, by focusing only on the areas of interest and extracting data for those regions over time. For example, in a surveillance video, you might be interested in extracting the motion of a specific object or person over time, which can be achieved by selecting a spatial slice around that object or person and extracting the data for that region over time.

Spatio-temporal time slicing can be performed using various software tools and libraries, such as OpenCV, MATLAB, and Python, among others. It is an important technique in many fields, including computer vision, machine learning, and video processing.

Spatio-temporal time slicing can be used for a variety of applications, including:

- Object tracking: By selecting a spatial slice around an object of interest and extracting its data over time, spatio-temporal time slicing can be used to track the movement of the object.
- Activity recognition: Spatio-temporal time slicing can be used to extract data for a specific region of interest in a video and analyze it over time to recognize and classify different activities
- Video summarization: By selecting specific spatial and temporal slices of a video, spatio-temporal time slicing can be used to create a summary of the most important events and activities in the video.
- Motion analysis: By selecting a spatial slice around a moving object or region and analyzing its data over time, spatio-temporal time slicing can be used to analyze and understand the motion patterns in a video.
- Video compression: Spatio-temporal time slicing can be used to extract the most important parts of a video and compress them into a smaller, more efficient format.

Spatio-temporal time slicing is a useful technique in video analysis and processing, but it can also be computationally intensive, especially for large videos. Therefore, it is important to carefully select the regions of interest and optimize the processing algorithms for efficiency.

CHAPTER 3

Implementation

3.1 Software specifications

For building EVM we primarily used Python as the programming language, the reason behind that is it's open-source computer vision library "cv2" which has all the necessary tools for image processing technique that we are about to use. However, you can achieve the similar results using other programming languages also.

3.2 Tools and Library

3.2.1 OpenCV-Python

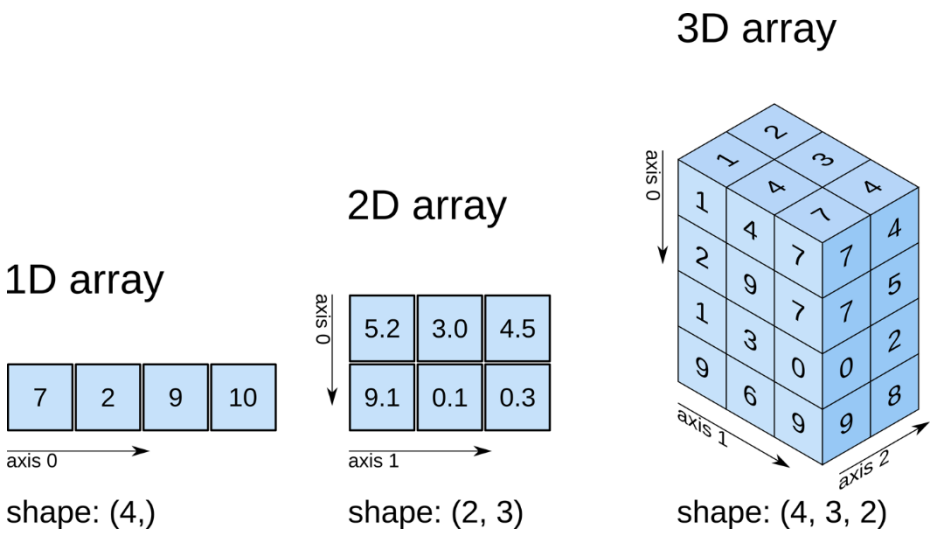
In Python, OpenCV is available as the "cv2" module. It provides a wide range of image and video processing functions, such as loading images and videos, manipulating images, applying various filters, and detecting features like edges and corners.

OpenCV also includes algorithms for object detection and recognition, face detection, motion tracking, and camera calibration. It is widely used in computer vision applications, robotics, augmented reality, and more.

In this project we'll be using OpenCV for video processing and applying various filters and algorithms on frames. Such as Gaussian pyramid, Laplacian pyramid and so on.

3.2.2 NumPy

NumPy (Numerical Python) is a popular open-source Python library used for scientific computing and data analysis. It provides a high-performance multidimensional array object, tools for working with these arrays, and a large collection of mathematical functions to operate on these arrays.



The main feature of NumPy is its powerful array object, called ndarray (short for N-dimensional array), which allows you to perform mathematical operations on entire arrays of data, instead of looping over individual elements. NumPy arrays are much faster and more efficient than Python lists, especially for large datasets.

NumPy provides a wide range of mathematical functions, including linear algebra, Fourier transform, random number generation, and more. It also provides tools for integrating with other Python libraries, such as SciPy for scientific computing.

Its efficient implementation of multidimensional arrays and mathematical functions makes it a popular choice for data analysis, scientific computing, and machine learning applications.

In this project we'll be using NumPy for constructing multidimensional array for storing our source video as tensor list.

3.2.3 Sci-py

Scipy is a powerful open-source library for scientific computing in Python that provides a wide range of modules for mathematical algorithms and scientific applications.

The signal processing module in Scipy provides functions to process and analyze signals, such as filtering, spectral analysis, and waveform generation. This module includes several functions for digital signal processing, such as finite impulse response (FIR) and infinite impulse response (IIR) filters, as well as signal processing utilities like resampling and spectrogram computation.

In this project we'll be using Scipy for signal processing and to be specific its signal and fast fourier transform methods and so on.

3.3 EVM for Motion Amplification

To achieve motion amplification, the same basic steps of Eulerian Video Magnification are followed, which involves decomposing a video into its different frequency bands and amplifying the desired frequency band to enhance the desired motion.

However, in this case, the goal is to amplify the low-frequency motion components instead of the high-frequency details. This is because low-frequency motions, such as those caused by breathing, heartbeat, or vibrations, are typically too subtle to be seen by the human eye but can be detected using Eulerian Video Magnification.

Once the low-frequency components have been extracted and amplified, they can be added back to the original video to produce a motion-amplified video.

3.3.1 Construct Tensor from Video Frames

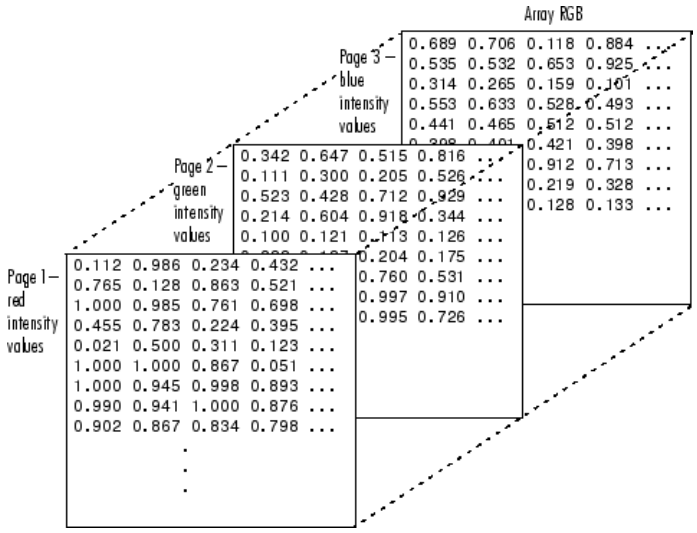
To construct a NumPy array tensor from video frames, we can use the OpenCV library in Python. OpenCV provides a function to read a video file and extract each frame as a NumPy array. The frames can then be stacked together to form a tensor.

First, we need to open the video file using cv2.VideoCapture(). This function returns a VideoCapture object that we can use to read frames from the video.

Next, we can loop through the frames of the video using the read() method of the Video Capture object. This method returns a Boolean value indicating whether a frame was successfully read, and the frame itself as a NumPy array.

Then we can append each frame to a list using the append() method. Once we have looped through all the frames, we can convert the list of frames to a NumPy array using the np.stack() function. This function takes a list of NumPy arrays and stacks them together along a new dimension.

The resulting NumPy array tensor will have dimensions (num_frames, height, width, num_channels), where num_frames is the number of frames in the video, height and width are the dimensions of each frame, and num_channels is the number of color channels in each frame (usually 3 for RGB images).



3.3.2 Building Laplacian and Gaussian pyramid

The Gaussian pyramid is a sequence of downsampled images where each level in the pyramid is produced by smoothing and subsampling the previous level of the pyramid. The purpose of the Gaussian pyramid is to create a scale-space representation of an image.

The Laplacian pyramid, on the other hand, is a sequence of images that represent the difference between two levels of the Gaussian pyramid. It is used to decompose an image into multiple levels of detail, where each level represents a different level of abstraction of the original image. The Laplacian pyramid is useful for image compression and coding, as well as for image enhancement and manipulation.

By using **pyrDown()** method from cv2 we can build a Gaussian s blur version of the frame. If we continued this for a multiple level we'll get a Gaussian Pyramid. And we can use the difference between those levels to build a Laplacian Pyramid.

3.3.3 Applying Butterworth filter

The purpose of Butterworth filter is to remove the unwanted frequency components from a signal while preserving a specific band of frequencies.

By using the **butter() and lfilter** method from **signal** library we can cut off the low and higher frequency up to certain level to reduce unwanted artifacts and noise.

3.3.4 Amplification and Reconstruction

Once we finished applying the band pass filter, we can directly amplify the resulting tensor by multiplying with amplification factor. And finally, by reconstructing the video we'll get the final tensor for each frame of the video, the function iteratively reconstructs the image by upsampling and adding the corresponding tensor from each level of the pyramid, starting from the highest level.

The reconstructed video is returned as a NumPy array final. Which can be directly converted back to video format.

3.3.5 Source Code

```
import cv2
import numpy as np
import scipy.signal as signal

#load video from file
def load_video(video_filename):
    cap=cv2.VideoCapture(video_filename)
    frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    width, height =
int(cap.get(cv2.CAP_PROP_FRAME_WIDTH)),int(cap.get(cv2.CAP_PROP_FRAM
E_HEIGHT))
    fps = int(cap.get(cv2.CAP_PROP_FPS))

video_tensor=np.zeros((frame_count,height,width,3),dtype='float')
    x=0
    while cap.isOpened():
        ret,frame=cap.read()
        if ret is True:
            video_tensor[x]=frame
            x+=1
        else:
            break
    return video_tensor,fps

#save video to files
def save_video(video_tensor):
    fourcc = cv2.VideoWriter_fourcc('M','J','P','G')
    [height,width]=video_tensor[0].shape[0:2]
    writer = cv2.VideoWriter("samples\\out.avi", fourcc, 30, (width,
height), 1)
    for i in range(0,video_tensor.shape[0]):
        writer.write(cv2.convertScaleAbs(video_tensor[i]))
    writer.release()

#Build Gaussian Pyramid
def build_gaussian_pyramid(src,level=3):
    s=src.copy()
    pyramid=[s]
    for i in range(level):
        s=cv2.pyrDown(s)
        pyramid.append(s)
    return pyramid

#Build Laplacian Pyramid
```

```
def build_laplacian_pyramid(src,levels=3):
    gaussianPyramid = build_gaussian_pyramid(src, levels)
    pyramid=[]
    for i in range(levels,0,-1):
        GE=cv2.pyrUp(gaussianPyramid[i])
        L=cv2.subtract(gaussianPyramid[i-1],GE)
        pyramid.append(L)
    return pyramid

#build laplacian pyramid for video
def laplacian_video(video_tensor,levels=3):
    tensor_list=[]
    for i in range(0,video_tensor.shape[0]):
        frame=video_tensor[i]
        pyr=build_laplacian_pyramid(frame,levels=levels)
        if i==0:
            for k in range(levels):

tensor_list.append(np.zeros((video_tensor.shape[0],pyr[k].shape[0],p
yr[k].shape[1],3)))
        for n in range(levels):
            tensor_list[n][i] = pyr[n]
    return tensor_list

#reconstract video from laplacian pyramid
def reconstract_from_tensorlist(filter_tensor_list,levels=3):
    final=np.zeros(filter_tensor_list[-1].shape)
    for i in range(filter_tensor_list[0].shape[0]):
        up = filter_tensor_list[0][i]
        for n in range(levels-1):
            up=cv2.pyrUp(up)+filter_tensor_list[n + 1][i]
up=cv2.pyrUp(up)
        final[i]=up
    return final

#butterworth bandpass filter
def butter_bandpass_filter(data, lowcut, highcut, fs, order=5):
    omega = 0.5 * fs
    low = lowcut / omega
    high = highcut / omega
    b, a = signal.butter(order, [low, high], btype='band')
    y = signal.lfilter(b, a, data, axis=0)
    return y

#magnify motion
def magnify_motion(video_name,low,high,levels=3,amplification=20):
    t,f=load_video(video_name)
    lap_video_list=laplacian_video(t,levels=levels)
    filter_tensor_list=[]
```

```
    for i in range(levels):

filter_tensor=butter_bandpass_filter(lap_video_list[i],low,high,f)
        filter_tensor*=amplification
        filter_tensor_list.append(filter_tensor)
    recon=reconstract_from_tensorlist(filter_tensor_list)
    final=t+recon
    save_video(final)

if __name__=="__main__":
     magnify_motion("samples\\antenna.mp4", 1, 6, 3, 5)
```

3.4 EVM for Color Amplification

Eulerian Video Magnification can also be used for color amplification in videos. Color amplification involves magnifying or enhancing color changes in a video that are not easily visible to the naked eye. This can be useful for analyzing color-based information in videos, such as changes in skin tone or the presence of certain objects.

To apply Eulerian Video Magnification for color amplification, the technique is first used to amplify motion or other variations in the video. Once these variations have been amplified, the color channels of the video can be processed separately to enhance their visibility.

For example, to amplify color changes in a video, we can take the Eulerian magnified video and transform it into a color space that separates the color channels, such as the Lab color space. The chromaticity channels (a* and b*) can be amplified while the luminance channel (L*) is left unchanged. The amplified chromaticity channels can then be transformed back into the original color space, resulting in a video with enhanced color changes.

3.4.1 Construct Tensor from Video Frames

As previous we are using cv2.VideoCapture() function to create a VideoCapture object for the video file. It then extracts various properties of the video using the get() method of the VideoCapture object, such as the frame count, width, height, and frames per second (FPS).

A NumPy array tensor is initialized with dimensions (frame_count, height, width, 3) using the np.zeros() function. A while loop is used to read each frame of the video using the read() method of the VideoCapture object. Each frame is assigned to a corresponding index in the tensor using the x variable as the index. The x variable is incremented after each frame is read to ensure that each frame is assigned to a unique index in the tensor.

The function returns the resulting tensor and the FPS of the video. The tensor has dimensions (frame_count, height, width, 3) and contains all the frames of the video in the order they appear in the video file.

3.4.2 Building Gaussian Pyramid

Unlike Motion Amplification we don't need to build lablacian Pyramid for colour Amplifacation. To build a Gaussian Pyramid we need to a sequence of downsampled images where each level in the pyramid is produced by smoothing and subsampling the previous level of the pyramid. The purpose of the Gaussian pyramid is to create a scale-space representation of an image.

3.4.3 Temporal Ideal Filter

The purpose of Temporal Ideal filter is to remove the unwanted frequency components from getting color amplified to prevent noice and distorsion.

By using the **FourierFieldTransform(FFT)** method from **signal** library we can cut off the low and higher frequency upto certain level by using Numpy **abs()** method to limit the range.

3.4.4 Amplification and Reconstruction

Once we finished applying the Ideal band pass filter we can directly amplify the colors of the resulting tensor by multiplying with amplification factor. And finally by reconstructing the video we'll get the final tensor for each frame of the video, the function iteratively reconstructs the image by upsampling and adding the corresponding tensor from each level of the pyramid, starting from the highest level.

The reconstructed video is returned as a NumPy array final. Which can be directly converted back to video format.

3.4.5 Source Code

```
import cv2
import numpy as np
import scipy.signal as signal
import scipy.fftpack as fftpack

#load video from file
def load_video(video_filename):
    cap=cv2.VideoCapture(video_filename)
```

```
    frame_count = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
    width, height =
int(cap.get(cv2.CAP_PROP_FRAME_WIDTH)),int(cap.get(cv2.CAP_PROP_FRAM
E_HEIGHT))
    fps = int(cap.get(cv2.CAP_PROP_FPS))

video_tensor=np.zeros((frame_count,height,width,3),dtype='float')
    x=0
    while cap.isOpened():
        ret,frame=cap.read()
        if ret is True:
            video_tensor[x]=frame
            x+=1
        else:
            break
    return video_tensor,fps

#save video to files
def save_video(video_tensor):
    fourcc = cv2.VideoWriter_fourcc('M','J','P','G')
    [height,width]=video_tensor[0].shape[0:2]
    writer = cv2.VideoWriter("samples\\out.avi", fourcc, 30, (width,
height), 1)
    for i in range(0,video_tensor.shape[0]):
        writer.write(cv2.convertScaleAbs(video_tensor[i]))
    writer.release()


#Build Gaussian Pyramid
def build_gaussian_pyramid(src,level=3):
    s=src.copy()
    pyramid=[s]
    for i in range(level):
        s=cv2.pyrDown(s)
        pyramid.append(s)
    return pyramid

# build gaussian pyramid for video
def gaussian_video(video_tensor,levels=3):
    for i in range(0,video_tensor.shape[0]):
        frame=video_tensor[i]
        pyr=build_gaussian_pyramid(frame,level=levels)
        gaussian_frame=pyr[-1]
        if i==0:

vid_data=np.zeros((video_tensor.shape[0],gaussian_frame.shape[0],gau
ssian_frame.shape[1],3))
        vid_data[i]=gaussian_frame
    return vid_data
```

```
#amplify the video
def amplify_video(gaussian_vid,amplification=50):
    return gaussian_vid*amplification

#reconstract video from original video and gaussian video
def reconstract_video(amp_video,origin_video,levels=3):
    final_video=np.zeros(origin_video.shape)
    for i in range(0,amp_video.shape[0]):
        img = amp_video[i]
        for x in range(levels):
            img=cv2.pyrUp(img)
        img=img+origin_video[i]
        final_video[i]=img
    return final_video

# apply temporal ideal bandpass filter to gaussian video
def temporal_ideal_filter(tensor,low,high,fps,axis=0):
    fft=fftpack.fft(tensor,axis=axis)
    frequencies = fftpack.fftfreq(tensor.shape[0], d=1.0 / fps)
    bound_low = (np.abs(frequencies - low)).argmin()
    bound_high = (np.abs(frequencies - high)).argmin()
    fft[:bound_low] = 0
    fft[bound_high:-bound_high] = 0
    fft[-bound_low:] = 0
    iff=fftpack.ifft(fft, axis=axis)
    return np.abs(iff)

#magnify color
def magnify_color(video_name,low,high,levels=3,amplification=20):
    t,f=load_video(video_name)
    gau_video=gaussian_video(t,levels=levels)
    filtered_tensor=temporal_ideal_filter(gau_video,low,high,f)

amplified_video=amplify_video(filtered_tensor,amplification=amplific
ation)
    final=reconstract_video(amplified_video,t,levels=3)
    save_video(final)

if __name__=="__main__":
     magnify_color("samples\\Keys.mp4", 0.4, 3)
```

CHAPTER 4

Applications of Eulerian Video Magnification

4.1 Medical Applications

4.1.1 Non-Invasive Vital Monitoring

EVM can be used to amplify small and subtle motions in a video sequence. The technique is based on the fact that small motions in an image sequence can be amplified by processing the sequence in the spatial frequency domain, where the high-frequency components correspond to small motions.

The technique has been applied to non-invasive pulse and heart rate monitoring, where it can be used to amplify the subtle changes in skin color and texture that are associated with the pulsatile blood flow. The method involves analyzing the color and texture changes in a video sequence of the subject's face, and then amplifying the frequency components that correspond to the pulse and heart rate.

This technique can be used to measure the pulse and heart rate of a subject in a non-invasive manner, without the need for any physical sensors or contact with the subject's body. The technique has the potential to be used in a variety of applications, including remote monitoring of patients, monitoring the health and well-being of athletes, and detecting physiological changes associated with stress or other conditions.

4.1.2 Early Detection of Parkinson’s disease

Eulerian video magnification (EVM) has shown promise in detecting early signs of Parkinson's disease by analyzing subtle and imperceptible changes in facial expressions and other body movements that are associated with the disease. Parkinson's disease is a neurodegenerative disorder that affects the movement and causes tremors, rigidity, and difficulty in walking.

Studies have shown that people with Parkinson's disease have certain facial expressions and body movements that are different from those without the disease. These changes can be subtle and difficult to detect with the naked eye, but can be amplified and analyzed using EVM.

In one study, researchers used EVM to analyze videos of individuals with Parkinson's disease and compared them with videos of healthy individuals. They found that the individuals with Parkinson's disease had different facial expressions and body movements, which were amplified and analyzed using EVM. The researchers were able to detect these changes with high accuracy and at an early stage of the disease, before other symptoms such as tremors became apparent.

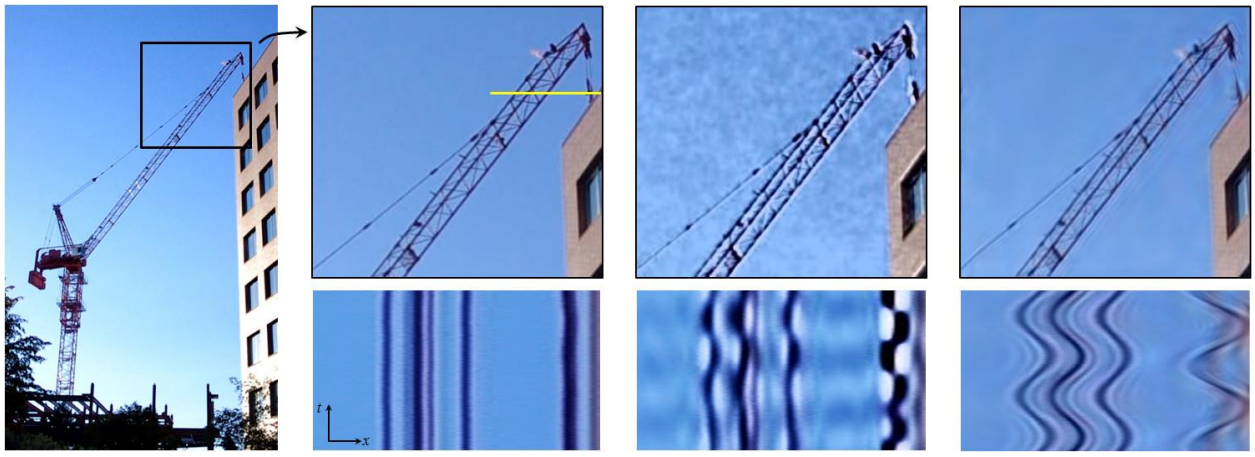
The early detection of Parkinson's disease using EVM has the potential to lead to earlier treatment and better outcomes for individuals with the disease. However, more research is needed to further validate the use of EVM for early detection of Parkinson's disease and to develop more accurate and reliable methods for analysis and interpretation of the EVM data.

4.2 Engineering Applications

Eulerian Motion Amplification (EMA) has a wide range of engineering applications, particularly in the fields of mechanical and civil engineering. One application of EMA is in the detection and analysis of structural vibrations. By amplifying the small vibrations in a structure, EMA can provide valuable insights into its mechanical behavior and help identify potential issues before they become critical.

EMA is also useful for monitoring the performance of rotating machinery, such as turbines and engines. By amplifying the subtle movements in the machinery, EMA can detect potential problems such as misalignments or worn bearings. This can help prevent costly downtime and equipment failures.

Another application of EMA is in the analysis of fluid dynamics. By magnifying the movements in a fluid, EMA can provide insights into the behavior of the flow, including turbulence, vortices, and other phenomena. This is particularly useful in the design of aerodynamic structures such as aircraft wings and wind turbines.



In addition to these applications, EMA can also be used in the design and testing of materials. By amplifying the small deformations in a material, EMA can provide information about its mechanical properties and behavior. This can be useful in the development of new materials and in ensuring the quality of existing materials.

4.3 Miscellaneous Applications

4.3.1 Surveillance:

EVM can be used to enhance video feeds from drones, surveillance cameras, and other sources to detect and track objects or people that may be difficult to see with the naked eye. This could be useful for border patrol, reconnaissance missions, and other mission critical applications.

4.3.2 Search and Rescue:

Eulerian Video Magnification (EVM) has potential applications in search and rescue operations, particularly in detecting and locating survivors in disaster-stricken areas. The technique can be used to amplify and detect very small movements, such as those made by a person trapped under rubble or debris.

EVM can be used in conjunction with thermal imaging to create a more comprehensive search system. By analyzing the thermal images of a disaster-stricken area, EVM can highlight the areas with the most movement and activity. This can help search and rescue teams locate survivors more quickly and efficiently.

In addition, EVM can also be used to detect breathing and heartbeat patterns of individuals trapped under rubble or debris. By amplifying and analyzing the subtle movements of the chest, EVM can detect even small movements associated with breathing and heartbeat.

Overall, the use of Eulerian Video Magnification in search and rescue operations has the potential to significantly improve the effectiveness of these operations. By amplifying and detecting small movements, EVM can help search and rescue teams locate survivors more quickly and with greater accuracy.

4.3.3 Extracting Audio from a Video Footage:

Eulerian Video Magnification is a technique that enables the amplification of subtle motions of a subject. Leveraging this methodology, it is possible to recover audio from a distant source by observing the surface of the object, amplifying its vibrations, and subsequently converting them back into audible sound.

CHAPTER 5

Discussion

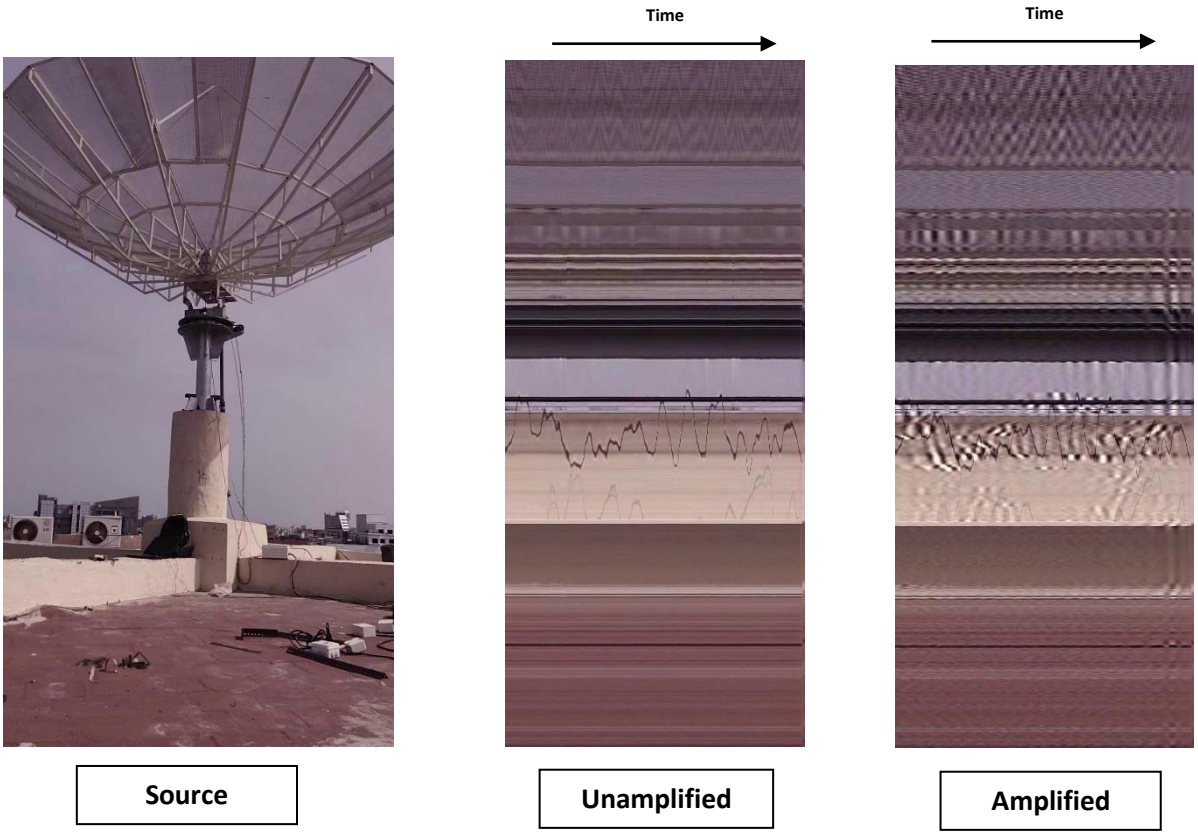
5.1 Samples and Results Breakdown

5.1.1 Structural analysis of antenna at High turbulent area

In this experiment, we aimed to study the structural behavior of an antenna located in a high turbulent area using Eulerian Video Magnification (EVM). We set up a high-speed camera to record the antenna's movements and vibrations during operation. The recorded video was then processed using the EVM algorithm to amplify small movements and vibrations.

The results of the experiment showed that EVM was an effective technique for studying the structural behavior of antennas in high turbulent areas. The amplified video provided valuable insights into the antenna's mechanical dynamics and identified potential areas of weakness that could be addressed to improve its performance. This technique has potential applications in other fields such as aerospace engineering and wind turbine design where similar structural analysis is necessary.

Spatio-Temporal Slices



Spatio-temporal slices were generated to visualize and analyze the movement of specific points on the antenna structure in a high turbulent area. These points were chosen based on their potential vulnerability to high winds and turbulence.

The time dimension of the spatio-temporal slice represents the duration of the movement. It indicates how long the selected point was in motion and how fast the movement was. By visualizing the time dimension, the researchers were able to observe the movement patterns of the selected point and identify any potential areas of stress or weakness on the antenna structure.

Overall, spatio-temporal slices in the X and time dimensions provided valuable insights into the structural dynamics of the antenna in a high turbulent area. They helped the researchers to better understand the movement patterns of specific points on the antenna structure and identify areas that needed improvement.

Results breakdown:

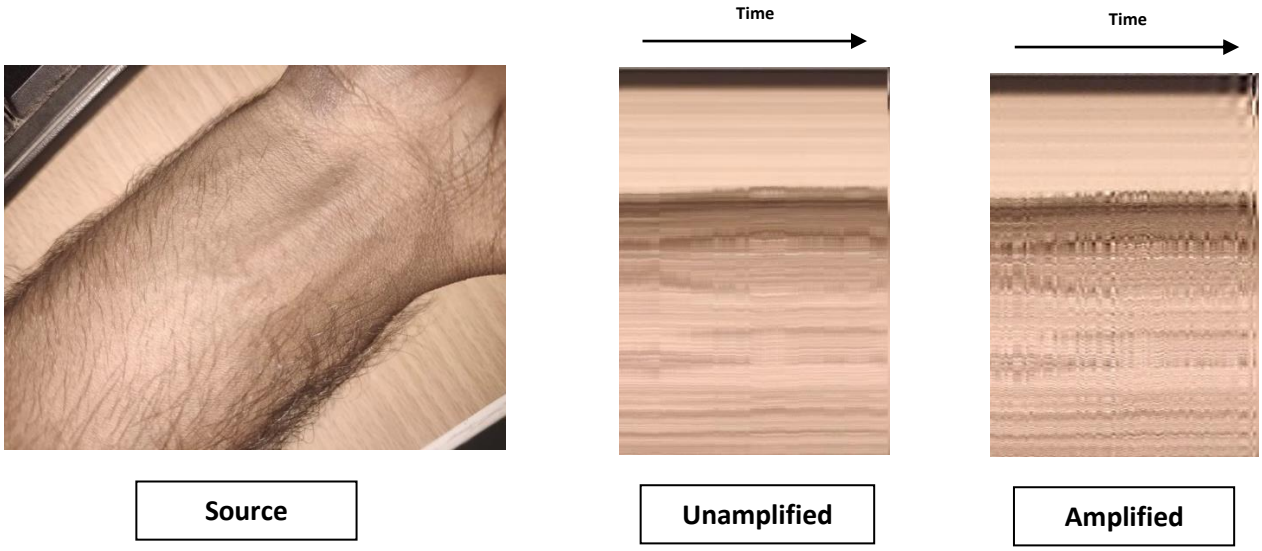
- EVM algorithm successfully amplified small vibrations and subtle movements of the antenna.
- Valuable insights into the antenna's structural dynamics were obtained.
- Potential areas of weakness were identified that could be addressed to improve its performance.

5.1.2 Motion amplification of a human hand

We used Eulerian Video Magnification (EVM) to amplify the motion of a human hand in a video recording. The EVM algorithm allowed us to visualize and analyze small, imperceptible movements of the hand that are normally difficult to detect. This technique can be used to better understand the mechanics of human movement and could have potential applications in fields such as sports science, physical therapy, and ergonomics.

The objective was to amplify the motion of a human hand in a video recording using Eulerian Video Magnification (EVM) and analyze the amplified video to gain insights into the mechanics of hand movement. Spatio-temporal slices were used as a visualization technique to analyze the amplified video.

Spatio-Temporal Slices



The time dimension of the spatio-temporal slice represents the duration of the movement. It indicates how long the selected point on the hand was in motion and how fast the movement was. By visualizing the time dimension, we were able to observe the movement patterns of the selected point over time and identify any potential areas of stress or weakness in the hand movement.

The spatio-temporal slices in the X dimension represent the position of a specific point on the hand during movement. This dimension indicates the location of the point in a particular direction, such as left or right. By visualizing the X dimension, we were able to observe the movement of the selected point and identify any patterns or areas of stress.

Overall, spatio-temporal slices in the X and time dimensions were valuable in gaining insights into the mechanics of hand movement and identifying potential areas of stress or injury. They helped us to better understand the movement patterns of specific points on the hand and analyze the motion in a more detailed manner.

Results breakdown:

- EVM algorithm successfully amplified the motion of a human hand in a video recording.
- Small, imperceptible movements of the hand were visualized and analyzed.
- Potential applications in fields such as sports science, physical therapy, and ergonomics were identified.

5.2 Limitations of Eulerian Video Magnification

Eulerian video magnification has some limitations and challenges, including:

Motion artifacts: EVM amplifies motion in the video, which can lead to motion artifacts, such as blurring, ghosting, and jittering.

Limited frequency range: EVM can only amplify frequencies within a specific frequency range. If the frequency of the desired motion is outside this range, it may not be effectively amplified.

Sensitivity to lighting and color changes: EVM may be sensitive to changes in lighting and color in the video, which can affect the accuracy and quality of the amplification. Any changes in lighting conditions or color can introduce errors in the analysis and affect the quality of the magnified video. In addition, EVM may not be suitable for all types of videos, such as those with low contrast or low frame rates, since the changes in pixel intensity may not be significant enough to detect and amplify. EVM may also introduce artifacts or distortions in the magnified video, especially when amplifying high-frequency components, which can affect the overall quality of the output.

Difficulty with complex scenes: EVM is primarily designed to amplify small motions, and its effectiveness may be limited when dealing with complex scenes with multiple moving objects or complex backgrounds. In such cases, the motion of interest may be difficult to isolate and amplify, leading to noisy or incorrect results. Additionally, EVM may not be effective in amplifying non-periodic or irregular motion, such as sudden movements or random fluctuations. Therefore, it is important to carefully consider the suitability of EVM for a particular application and to use it in combination with other techniques or approaches where necessary.

Computational complexity: The Eulerian Video Magnification (EVM) algorithm involves complex computations that require significant processing power and memory. As a result, the algorithm can be computationally intensive and may require specialized hardware or parallel processing to run in real-time. This can limit the practicality of EVM for real-time applications, such as medical imaging or video surveillance. However, recent advancements in computer hardware and software have made it possible to implement EVM in real-time on standard desktop computers, albeit with certain limitations.

5.3 Conclusion

Eulerian Video Magnification (EVM) is a powerful technique that enables the amplification of subtle changes in a video that are difficult to observe with the naked eye. By analyzing the video at different temporal scales and applying spatial filtering, EVM enhances color and motion variations that are not visible to the human eye.

EVM has a wide range of applications in fields such as medicine, biology, and engineering. For instance, it has been used for non-invasive monitoring of physiological signals such as heart rate and breathing rate, which is crucial in medical diagnosis and treatment. It has also been used to detect motion in low-light conditions, visualize air flow, and monitor brain activity.

Furthermore, EVM has the potential to advance research in several areas by providing insights into the underlying mechanisms and dynamics of complex systems. For example, EVM has been used to study the development of embryos, to analyze the behavior of insects, and to understand the interaction between humans and their environment.

Moreover, EVM can be used in combination with other techniques such as computer vision and machine learning to further enhance its capabilities. By integrating EVM with deep learning models, researchers have been able to detect and track specific features in the magnified video, such as blood vessels, and to classify abnormal patterns in medical images.

In conclusion, Eulerian Video Magnification is a powerful and versatile technique that has the potential to transform various fields of research. While it has limitations and ethical considerations, it remains a promising tool for uncovering hidden information in videos and analyzing complex systems.

REFERENCES:

1. Lauridsen, H., Gonzales, S., Hedwig, D., Perrin, K. L., Williams, C. J., Wrege, P. H., ... & Butcher, J. T. (2019). Extracting physiological information in experimental biology via Eulerian video magnification. *BMC biology*, *17*(1), 1-26.
2. Shahadi, H. I., Albattat, H. J., Al-allaq, Z. J., & Thahab, A. T. (2020). Eulerian video magnification: A review. *Indones. J. Electr. Eng. Comput. Sci*, *18*(2), 799.
3. Williams S, Fang H, Relton SD, Graham CD, Alty JE. Seeing the unseen: Could Eulerian video magnification aid clinician detection of subclinical Parkinson's tremor? J Clin Neurosci. 2020 Nov;81:101-104. doi: 10.1016/j.jocn.2020.09.046. Epub 2020 Oct 2. PMID: 33222895.
4. Wu, H. Y., Rubinstein, M., Shih, E., Guttag, J., Durand, F., & Freeman, W. (2012). Eulerian video magnification for revealing subtle changes in the world. *ACM transactions on graphics (TOG)*, *31*(4), 1-8.