Multimedia Processing and Applications

IMAGE MORPHING

Khalid Mehtab Khan

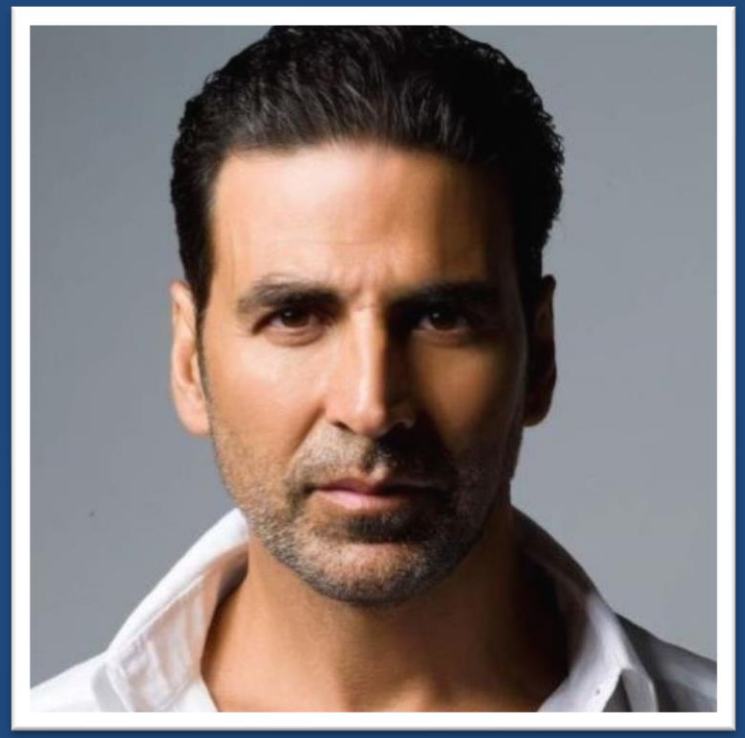
17ucs078@lnmiit.ac.in

+91-77727007427

The LNM Institute Of Information Technology, Jaipur

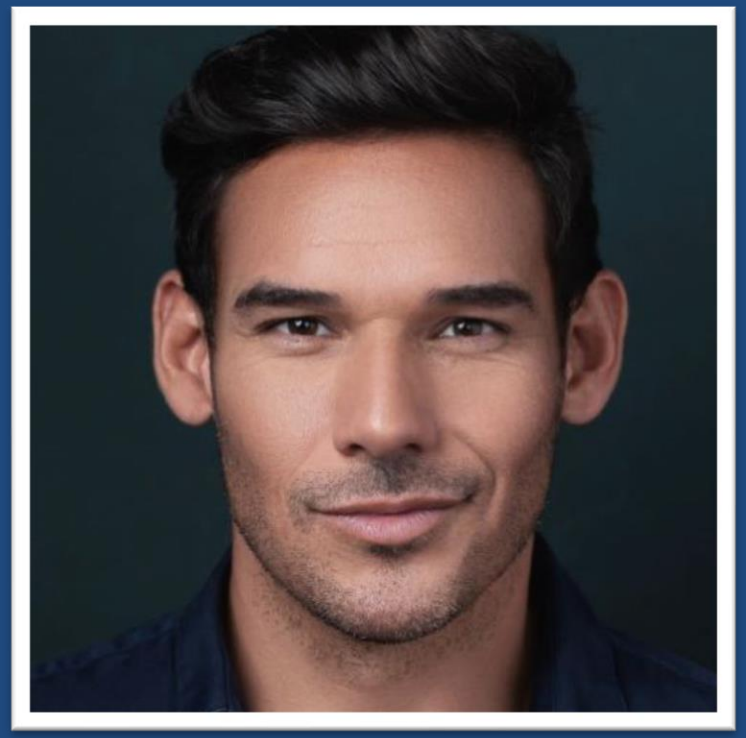
AIM: Morphing Image1 to Image2 using face morphing techniques and maintaining restrictions.

Image1



Size:(1000x1000)

Image2



Size:(1000x1000)

MORPHING:

- As the name suggests ‘MORPHING’, is the transition effect from one image to another. The speciality about morphing and why it is not called just a simple transition of images is morphing transforms one image to another in a seamless manner that is there are no awkward transformations but maintaining smooth flow.

- One more point we will be focusing in this project is the restrictions or control points in technical terms. Control points are points defined by the user in the initial and final image which remain intact. As the project is a face morphing project control points can be facial features like eyes, nose etc which will remain intact throughout the transition.

PROCEDURE:

STEP 1 - Select points in both the images and triangulate them.

STEP 2- Linearly interpolate the coordinate values of control points and find out the affine coordinates of non-control points in the intermediate frame.

STEP 3- Use these affine coordinates to find out the location of non-control points in the first and second images.

STEP 4- Use color interpolation to assign color values to pixels in the intermediate frame.

1 Morphing.py

```
import numpy as np
import cv2 as cv2
import matplotlib.pyplot as plt
from pts import myPts
from intriangle import intriangle
from alphabeta import alphabeta
from sdpoints import sdpoints

cv2.destroyAllWindows()

src =[]
des=[]
p = []

color = (0,0,255)
thickness = 3

#read images
srcimg = cv2.imread('ak1.jpg')                    #source image
desimg = cv2.imread('tw1.jpg')                    #destinaniton image

srcimg1 = cv2.resize(srcimg,(500,500))
desimg1 = cv2.resize(desimg,(500,500))
srcimg2 = cv2.resize(srcimg,(500,500))
desimg2 = cv2.resize(desimg,(500,500))


#press a after selecting the three points to form triangle
u, v = myPts(srcimg1)
u, v = myPts(desimg1)
```

Press a on your keyboard once you select points in an images to continue.

pts.py

Function

```
import numpy as np
import cv2
import matplotlib.pyplot as plt

posListx = []
posListy = []

def myPts(image):
    def onMouse(event, x, y, flags, param):
        global posListx, posListy

        if event == cv2.EVENT_LBUTTONDOWN:

            posListx.append(x)
            posListy.append(y)

    cv2.namedWindow('image')
    cv2.setMouseCallback('image',onMouse)

    while(1):
        cv2.imshow("image",image)
        k = cv2.waitKey(20) & 0xFF
        if k == 27:
            return [],[]
            break
        elif k == ord('a'):
            cv2.destroyAllWindows()
            return posListx, posListy
```

```
if __name__ == "__main__":
    srcimg = cv2.imread('dtrump.jpg')                    #source image
    desimg = cv2.imread('tcruz.jpg')                     #destinaniton image


    srcimg1 = cv2.resize(srcimg,(500,500))
    desimg2 = cv2.resize(desimg,(500,500))

    #print(myPts(srcimg1))
```

The main morphing code as been described in parts 1-12 through these slides and the functions used by there side.
We start with the input of images and converting both images to a square size of 500x500. The function ‘pts’ takes these images as inputs and allows user to select control points using mouse clicks and return array ‘u’, with x co-ordinates and array ‘v’ with y coordinates selected in both images.

2 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
#press a after selecting the three points to form triangle
u, v = myPts(srcimg1)
u, v = myPts(desimg1)

a= u[0]*u[0] + v[0]*v[0]
b= u[1]*u[1] + v[1]*v[1]
c= u[2]*u[2] + v[2]*v[2]
while(1):
    if((a<=b) and (a<=c)):
        src.append(u[0])
        src.append(v[0])
        des.append(u[3])
        des.append(v[3])

    elif((b<=a) and (b<=c)):
        src.append(u[1])
        src.append(v[1])
        des.append(u[4])
        des.append(v[4])

    elif((c<=b) and (c<=a)):
        src.append(u[2])
        src.append(v[2])
        des.append(u[5])
        des.append(v[5])
    break

d= u[0]*u[0] + ((500-v[0])*(500-v[0]))
e= u[1]*u[1] + ((500-v[1])*(500-v[1]))
f= u[2]*u[2] + ((500-v[2])*(500-v[2]))
while(1):
```

3 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
d= u[0]*u[0] + ((500-v[0])*(500-v[0]))
e= u[1]*u[1] + ((500-v[1])*(500-v[1]))
f= u[2]*u[2] + ((500-v[2])*(500-v[2]))
while(1):
    if((d<=e) and (d<=f)):
        src.append(u[0])
        src.append(v[0])
        des.append(u[3])
        des.append(v[3])
        if(src[0]==u[1]):
            src.append(u[2])
            src.append(v[2])
            des.append(u[5])
            des.append(v[5])
        else:
            src.append(u[1])
            src.append(v[1])
            des.append(u[4])
            des.append(v[4])

    elif((e<=d) and (e<=f)):
        src.append(u[1])
        src.append(v[1])
        des.append(u[4])
        des.append(v[4])
        if(src[0]==u[0]):
            src.append(u[2])
            src.append(v[2])
            des.append(u[5])
            des.append(v[5])
        else:
```

4 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
            src.append(v[2])
            des.append(u[5])
            des.append(v[5])
        else:
            src.append(u[0])
            src.append(v[0])
            des.append(u[3])
            des.append(v[3])

    elif((f<=e) and (f<=d)):
        src.append(u[2])
        src.append(v[2])
        des.append(u[5])
        des.append(v[5])
        if(src[0]==u[0]):
            src.append(u[1])
            src.append(v[1])
            des.append(u[4])
            des.append(v[4])
        else:
            src.append(u[0])
            src.append(v[0])
            des.append(u[3])
            des.append(v[3])
    break

print(src)
print(des)

#forming triangles
```

As we proceed with the code we realise that we require the points selected in specific order of ‘x’ and ‘y’ coordinates and this is the first problem we faced. To fix the problem I added this piece of code (2-3-4 Morphing.py) to re-arrange the array returned by function ‘pts’ and save the arranged ‘x’ and ‘y’ coordinates of initial image in array ‘src’ and final image in array ‘des’ respectively

5 Morphing.py

```
print(src)
print(des)

#forming triangles
cv2.line(srcimg2,(src[0],src[1]) , (0,0), color, thickness)
cv2.line(srcimg2,(src[0],src[1]) , (0,499), color, thickness)
cv2.line(srcimg2,(src[0],src[1]) , (499,0), color, thickness)
cv2.line(srcimg2,(src[0],src[1]) , (src[2],src[3]), color, thickness)
cv2.line(srcimg2,(src[0],src[1]) , (src[4],src[5]), color, thickness)
cv2.line(srcimg2,(src[2],src[3]) , (0,499), color, thickness)
cv2.line(srcimg2,(src[2],src[3]) , (499,499), color, thickness)
cv2.line(srcimg2,(src[2],src[3]) , (src[4],src[5]), color, thickness)
cv2.line(srcimg2,(src[4],src[5]) , (499,0), color, thickness)
cv2.line(srcimg2,(src[4],src[5]) , (499,499), color, thickness)
cv2.imwrite(("Lines Source.jpg"),srcimg2)

cv2.line(desimg2,(des[0],des[1]) , (0,0), color, thickness)
cv2.line(desimg2,(des[0],des[1]) , (0,499), color, thickness)
cv2.line(desimg2,(des[0],des[1]) , (499,0), color, thickness)
cv2.line(desimg2,(des[0],des[1]) , (des[2],des[3]), color, thickness)
cv2.line(desimg2,(des[0],des[1]) , (des[4],des[5]), color, thickness)
cv2.line(desimg2,(des[2],des[3]) , (0,499), color, thickness)
cv2.line(desimg2,(des[2],des[3]) , (499,499), color, thickness)
cv2.line(desimg2,(des[2],des[3]) , (des[4],des[5]), color, thickness)
cv2.line(desimg2,(des[4],des[5]) , (499,0), color, thickness)
cv2.line(desimg2,(des[4],des[5]) , (499,499), color, thickness)
cv2.imwrite(("Lines Destination.jpg"),desimg2)
```

```
while(1):
        cv2.imshow("Source Lines",srcimg2)
        k = cv2.waitKey(20) & 0xFF
        if k == ord('a'):
                cv2.destroyAllWindows()
                break


while(1):
        cv2.imshow("Destination Lines",desimg2)
        k = cv2.waitKey(20) & 0xFF
        if k == ord('a'):
                cv2.destroyAllWindows()
                break
```

src

des

x coordinate of point 1 of source image	y coordinate of point 1 of source image	X coordinate of point 2 of source image	y coordinate of point 2 of source image	x coordinate of point 3 of source image	y coordinate of point 3 of source image
x coordinate of point 1 of destination image	y coordinate of point 1 of destination image	x coordinate of point 2 of destination image	y coordinate of point 2 of destination image	x coordinate of point 3 of destination image	y coordinate of point 3 of destination image

This piece of code shows the lines for the triangles formed for triangulation once the points are selected using the selected points and the corners of the image i.e. (0,0) , (0,499) , (499,499) and (499,0). The out of this functions is stored in the directory with file name ‘source lines’ and ‘Destination Lines’ respectively and shown in the end of the slides in the OUTPUT section.

6 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
#creating new image with zero pixel values
temp = np.zeros(shape=[500,500,3],dtype=np.uint8)
#cv2.imshow("blank",temp)
N=20
K=0
f=499
i= 0
j= 0

for K in range(N+1):
        q1=((N-K)/N)*src[0]+(K/N)*des[0]
        p.append(q1)
        w1=((N-K)/N)*src[1]+(K/N)*des[1]
        p.append(w1)
        q2=((N-K)/N)*src[2]+(K/N)*des[2]
        p.append(q2)
        w2=((N-K)/N)*src[3]+(K/N)*des[3]
        p.append(w2)
        q3=((N-K)/N)*src[4]+(K/N)*des[4]
        p.append(q3)
        w3=((N-K)/N)*src[5]+(K/N)*des[5]
        p.append(w3)
#loop for every pixel
        for i in range(f+1):
                for j in range(f+1):
                        if intriangle(p[0],p[1],0,0,0,499,i,j)==1:
                                e1x=0-p[0]
                                e1y=0-p[1]
                                e2x=0-p[0]
                                e2y=499-p[1]
```

'N' can be specified to fix the number of intermediate frames required

Linear interpolation of the coordinate values of control points to find out the affine coordinates of non-control points in the intermediate frame.

intriangle.py

Function

intriangle.py - K:\LNMIIT\SEM 6\MPA\Project\intriangle.py (3.8.2)

File Edit Format Run Options Window Help

```
def area(x1, y1, x2, y2, x3, y3):
    a = abs((x1*(y2-y3) +  x2*(y3-y1) + x3*(y1-y2))/2)
    return a



def intriangle(x1, y1, x2, y2, x3, y3, x, y):
    A = area(x1, y1, x2, y2, x3, y3)
    A1 = area(x, y, x2, y2, x3, y3)
    A2 = area(x1, y1, x, y, x3, y3)
    A3 = area(x1, y1, x2, y2, x, y)

    if(A==(A1+A2+A3)):
        b= 1
    else:
        b= 0
    return b
```

The in triangle function takes the points of the current for loop and points from the corner to check which triangle the points lie in to decide the further defining of vectors.

'eix' and 'eiy' : define vectors

7 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
#loop for every pixel
        for i in range(f+1):
                for j in range(f+1):
                        if intriangle(p[0],p[1],0,0,0,499,i,j)==1:
                                e1x=0-p[0]
                                e1y=0-p[1]
                                e2x=0-p[0]
                                e2y=499-p[1]
                                A = alphabeta(e1x,e2x,e1y,e2y,p[0],p[1],i,j)
                                sx,sy =sdpoints(0-src[0],0-src[1],0-src[0],499-src[1],src[0],src[1],A)
                                dx,dy =sdpoints(0-des[0],0-des[1],0-des[0],499-des[1],des[0],des[1],A)
                                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                                temp[i,j,0]=red
                                temp[i,j,1]=green
                                temp[i,j,2]=blue

                        elif intriangle(p[0],p[1],p[2],p[3],0,499,i,j)== 1:
                                e1x=0-p[0]
                                e1y=499-p[1]
                                e2x=p[2]-p[0]
                                e2y=p[3]-p[1]

                                A=alphabeta(e1x,e2x,e1y,e2y,p[0],p[1],i,j)
                                sx,sy =sdpoints(0-src[0],499-src[1],src[2]-src[0],src[3]-src[1],src[0],src[1],A)
                                dx,dy =sdpoints(0-des[0],499-des[1],des[2]-des[0],des[3]-des[1],des[0],des[1],A)
                                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
```

Using affine coordinates to find out the location of non-control points in the first and second images.

Using color interpolation to assign color values to pixels in the intermediate frame.

alphabeta.py

Function

alphabeta.py - K:\LNMIIT\SEM 6\MPA\Project\alphabeta.py (3.8.2)

File Edit Format Run Options Window Help

```
import numpy as np


def alphabeta(e1x, e2x, e1y, e2y, x0, y0, x, y):
    C= np.array([[x-x0],[y-y0]])
    B = np.array([[e1x,e2x],[e1y,e2y]])
    B = np.linalg.inv(B)
    A = np.dot(B,C)
    return A
```

sdpoints.py

Function

sdpoints.py - K:\LNMIIT\SEM 6\MPA\Project\sdpoints.py (3.8.2)

File Edit Format Run Options Window Help

```
def sdpoints(e1x,e1y,e2x,e2y,x0,y0,m):
    alpha=m[0][0]
    beta =m[1][0]
    a=x0+(alpha*e1x)+(beta*e2x)
    b=y0+(alpha*e1y)+(beta*e2y)
    a=round(a)
    b=round(b)
    a=int(a)
    b=int(b)
    return a,b
```

After the formation of vectors the function calculates the alpha and beta values using the vectors and points and the 'sdpoints' function for further colour interpolation of pixels. The two functions and their working has been defined pointing out where they have been used.

8 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
                A=alphabeta(e1x,e2x,e1y,e2y,p[0],p[1],i,j)
                sx,sy =sdpoints(0-src[0],499-src[1],src[2]-src[0],src[3]-src[1],src[0],src[1],A)
                dx,dy =sdpoints(0-des[0],499-des[1],des[2]-des[0],des[3]-des[1],des[0],des[1],A)
                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue

        elif intriangle(0,499,p[2],p[3],499,499,i,j)==1:
                e1x=0-p[2]
                e1y=499-p[3]
                e2x=499-p[2]
                e2y=499-p[3]

                A=alphabeta( e1x,e2x,e1y,e2y,p[2],p[3],i,j)
                sx,sy=sdpoints(0-src[2],499-src[3],499-src[2],499-src[3],src[2],src[3],A)
                dx,dy=sdpoints(0-des[2],499-des[3],499-des[2],499-des[3],des[2],des[3],A)
                a1=srcimg1[sx,sy,0]
                z1=desimg1[dx,dy,0]
                red=((N-K)/N)*(a1)+(K/N)*(z1)
                #red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue
```

9 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
        elif intriangle(p[4],p[5],p[2],p[3],499,499,i,j)==1:
                e1x=p[4]-p[2]
                e1y=p[5]-p[3]
                e2x=499-p[2]
                e2y=499-p[3]


                A=alphabeta(e1x,e2x,e1y,e2y,p[2],p[3],i,j)
                sx,sy=sdpoints(src[4]-src[2],src[5]-src[3],499-src[2],499-src[3],src[2],src[3],A)
                dx,dy=sdpoints(des[4]-des[2],des[5]-des[3],499-des[2],499-des[3],des[2],des[3],A)
                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue

        elif intriangle(p[4],p[5],499,0,499,499,i,j)==1:
                e1x=499-p[4]
                e1y=0-p[5]
                e2x=499-p[4]
                e2y=499-p[5]

                A=alphabeta( e1x,e2x,e1y,e2y,p[4],p[5],i,j)
                sx,sy=sdpoints(499-src[4],0-src[5],499-src[4],499-src[5],src[4],src[5],A)
                dx,dy=sdpoints(499-des[4],0-des[5],499-des[4],499-des[5],des[4],des[5],A)
                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])
```

10 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue

        elif intriangle(p[0],p[1],p[4],p[5],499,0,i,j)==1:
                e1x=p[0]-p[4]
                e1y=p[1]-p[5]
                e2x=499-p[4]
                e2y=0-p[5]

                A=alphabeta( e1x,e2x,e1y,e2y,p[4],p[5],i,j)
                sx,sy=sdpoints(src[0]-src[4],src[1]-src[5],499-src[4],0-src[5],src[4],src[5],A)
                dx,dy=sdpoints(des[0]-des[4],des[1]-des[5],499-des[4],0-des[5],des[4],des[5],A)
                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue

        elif intriangle(p[0],p[1],0,0,499,0,i,j)==1:
                e1x=0-p[0]
                e1y=0-p[1]
                e2x=499-p[0]
                e2y=0-p[1]

                A=alphabeta( e1x,e2x,e1y,e2y,p[0],p[1],i,j)
                sx,sy=sdpoints(0-src[0],0-src[1],499-src[0],0-src[1],src[0],src[1],A)
                dx,dy=sdpoints(0-des[0],0-des[1],499-des[0],0-des[1],des[0],des[1],A)
```

11 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

File Edit Format Run Options Window Help

```
                A=alphabeta( e1x,e2x,e1y,e2y,p[0],p[1],i,j)
                sx,sy=sdpoints(0-src[0],0-src[1],499-src[0],0-src[1],src[0],src[1],A)
                dx,dy=sdpoints(0-des[0],0-des[1],499-des[0],0-des[1],des[0],des[1],A)
                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue

        elif intriangle(p[0],p[1],p[2],p[3],p[4],p[5],i,j)==1:
                e1x=p[0]-p[2]
                e1y=p[1]-p[3]
                e2x=p[4]-p[2]
                e2y=p[5]-p[3]

                A=alphabeta( e1x,e2x,e1y,e2y,p[2],p[3],i,j)
                sx,sy=sdpoints(src[0]-src[2],src[1]-src[3],src[4]-src[2],src[5]-src[3],src[2],src[3],A)
                dx,dy=sdpoints(des[0]-des[2],des[1]-des[3],des[4]-des[2],des[5]-des[3],des[2],des[3],A)
                red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                temp[i,j,0]=red
                temp[i,j,1]=green
                temp[i,j,2]=blue

#display final image
        while(1):
                cv2.imshow("Temp Lines",temp)
```

12 Morphing.py

morphing.py - K:\LNMIIT\SEM 6\MPA\Project\morphing.py (3.8.2)

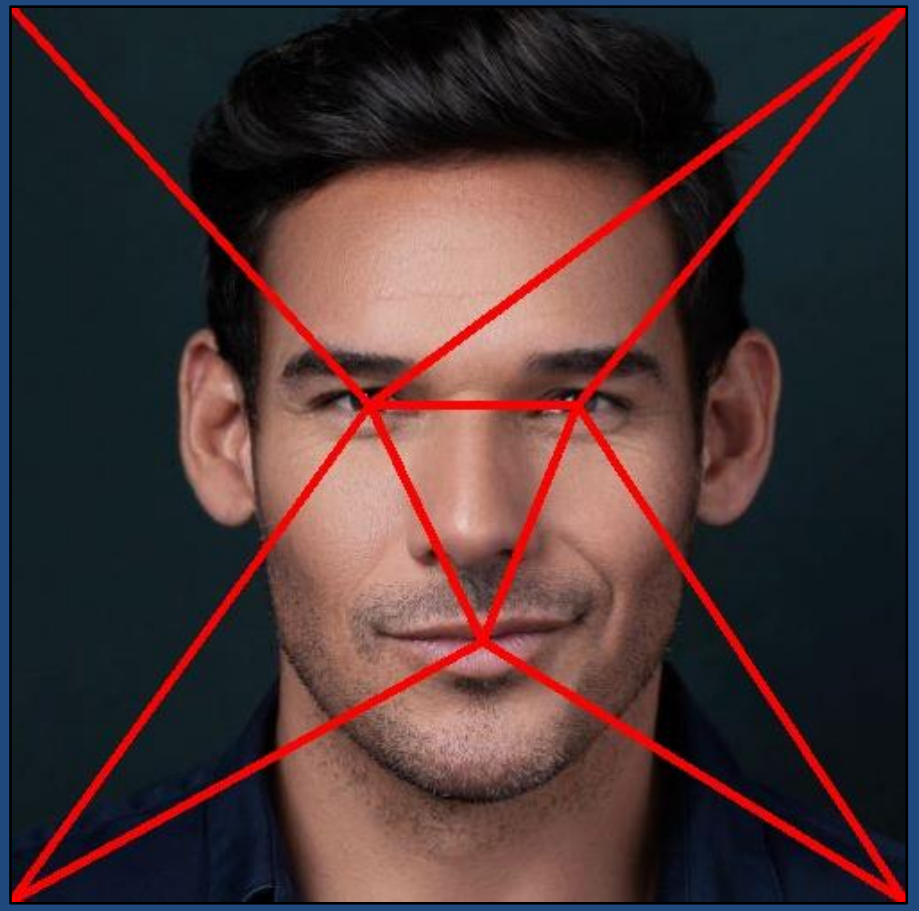
File Edit Format Run Options Window Help

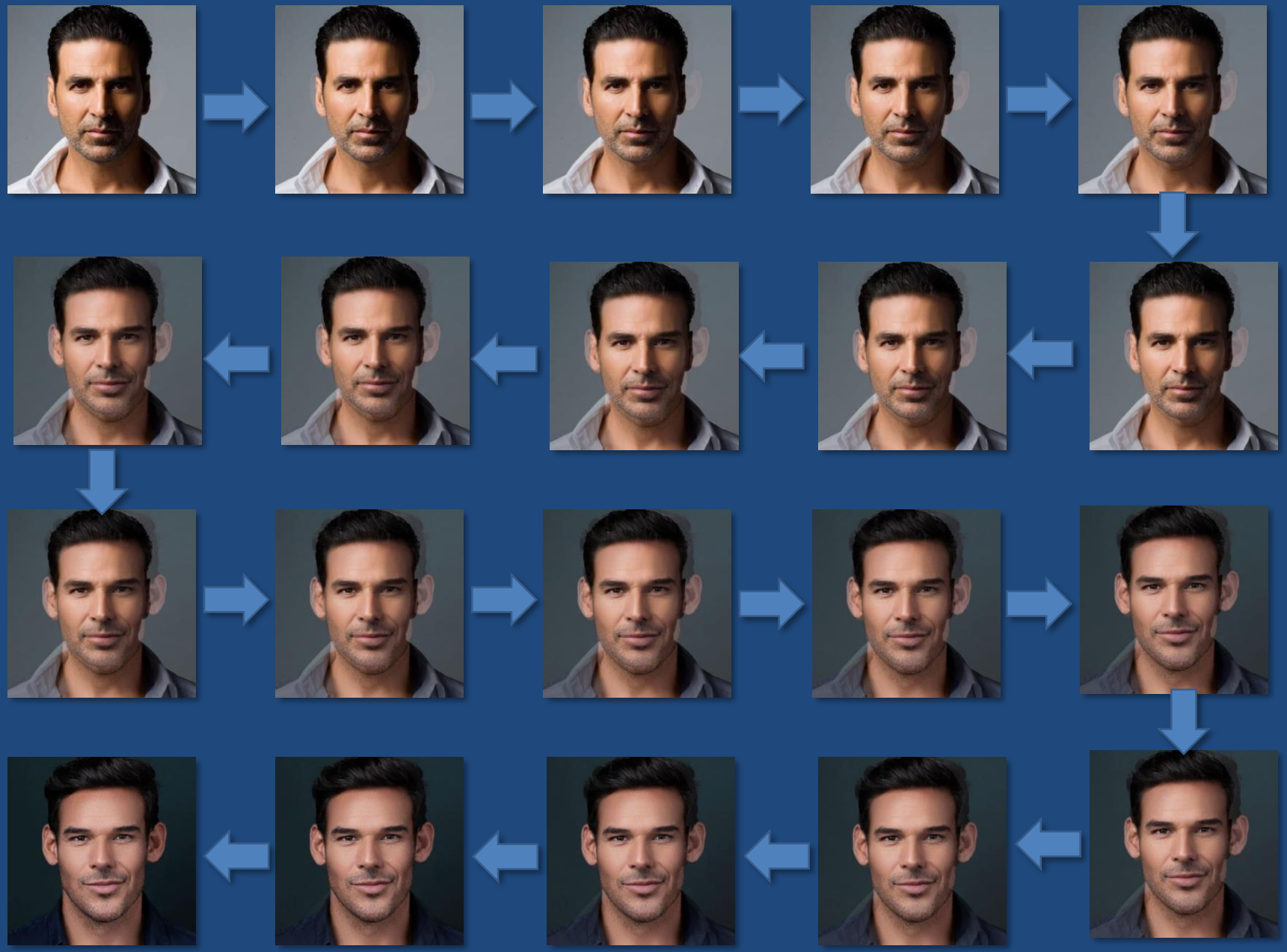
```
                    e2y=p[5]-p[3]

                    A=alphabeta( e1x,e2x,e1y,e2y,p[2],p[3],i,j)
                    sx,sy=sdpoints(src[0]-src[2],src[1]-src[3],src[4]-src[2],src[5]-src[3],src[2],src[3],A)
                    dx,dy=sdpoints(des[0]-des[2],des[1]-des[3],des[4]-des[2],des[5]-des[3],des[2],des[3],A)
                    red=((N-K)/N)*(srcimg1[sx,sy,0])+(K/N)*(desimg1[dx,dy,0])
                    green=((N-K)/N)*(srcimg1[sx,sy,1])+(K/N)*(desimg1[dx,dy,1])
                    blue=((N-K)/N)*(srcimg1[sx,sy,2])+(K/N)*(desimg1[dx,dy,2])

                    temp[i,j,0]=red
                    temp[i,j,1]=green
                    temp[i,j,2]=blue

#display final image
        while(1):
                cv2.imshow("Temp Lines",temp)

                k = cv2.waitKey(0) & 0xFF
                if k == ord('a'):
                        #filename = '"%d"morphed frame'
                        cv2.imwrite(("morphedframe%d.jpg"%K),temp)
                        cv2.destroyAllWindows()
                        break
```

In every loop an intermediate frame is generated and we save the intermediate frame to the directory along with the frame number which indicates the current loop

OUTPUT:1

These figure show the output of the lines command to show the triangles formed by the selected points and the corners of the image. We see 8 triangles are formed.





Morphed Frames 1-20

We generated 20 intermediate frames, the more the frames the more gradual and seamless the transition will look as the transition towards the final image will be divided into more stages, the transition of these frames has been shown by representation of arrows how the image is morphed into the final image

THE DIFFERENCE

Problem 2

Final Image (original)

Final image (morphed)

If we look closely we will see that the final image morphed into the destination image has some differences in it. The face marked with the red area.

The reason for these slight changes is the selection of control points, to keep the selected points intact the morphed image tries to fit the selected features in the same place giving rise to small changes

VISUALISATION:

videomaker.py - K:\LNMIIT\SEM 6\MPA\Project\Intermediate Frames\videomaker.py (3.8.2)

File Edit Format Run Options Window Help

```
import cv2
import numpy as np
import glob

img_array = []
K=1
N=20


img0 = cv2.imread('morphedframe0.jpg')
height, width, layers = img0.shape
size = (width, height)
img_array.append(img0)

for K in range(N+1):
    img = cv2.imread('morphedframe%d.jpg'%K)
    img_array.append(img)




out = cv2.VideoWriter('project.mp4',cv2.VideoWriter_fourcc(*'DIVX'),15,size)
for i in range(len(img_array)):
    out.write(img_array[i])
out.release()
```

videomaker.py

Make sure you import the 'glob' library

To have a better idea of what we did, add the file 'videomaker.py' to the folder where morphed frames are saved.
Specify N for the number of frames generated

The code will produce a video using the frames which will display the morphing i.e. the transition effect we developed

Thankyou