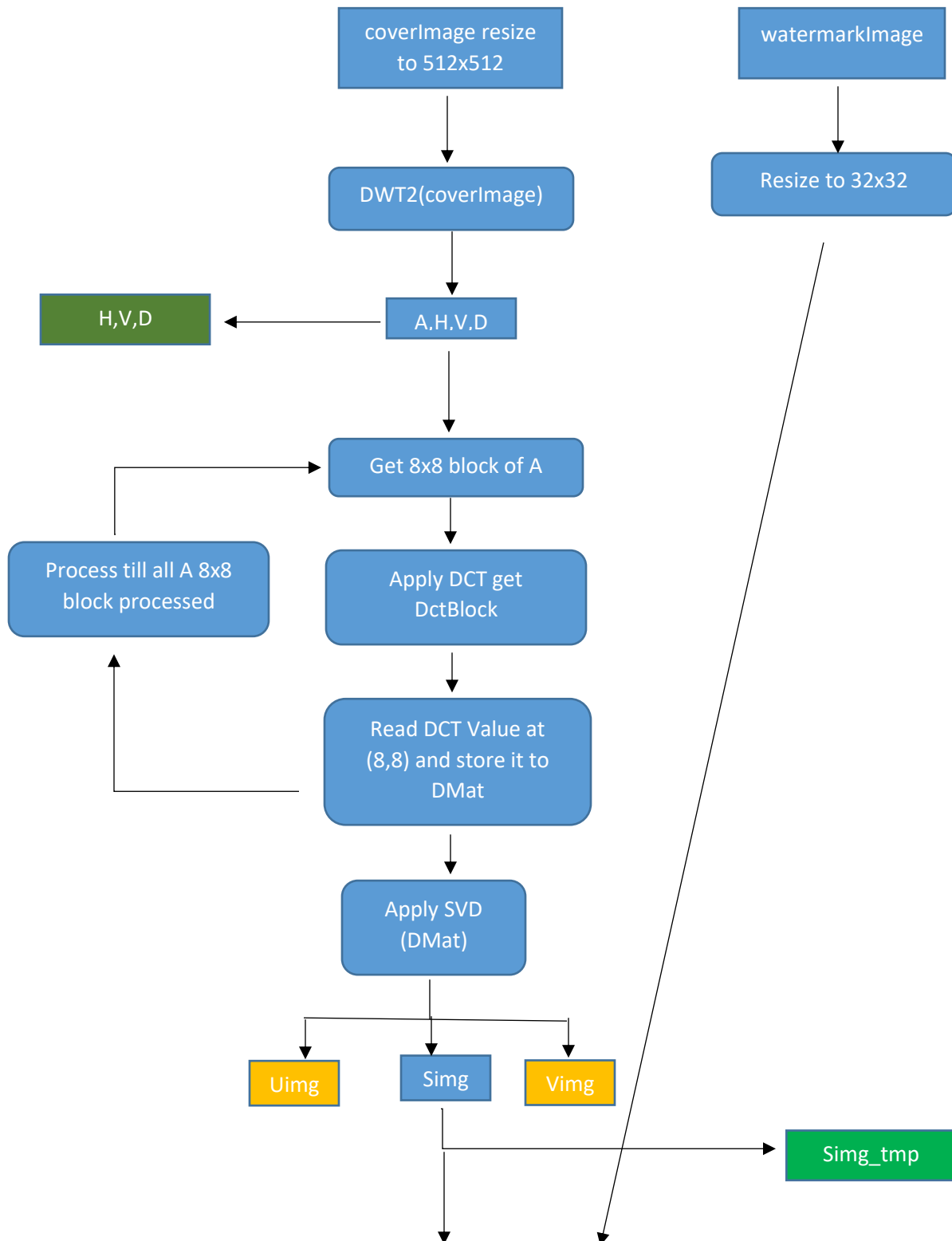
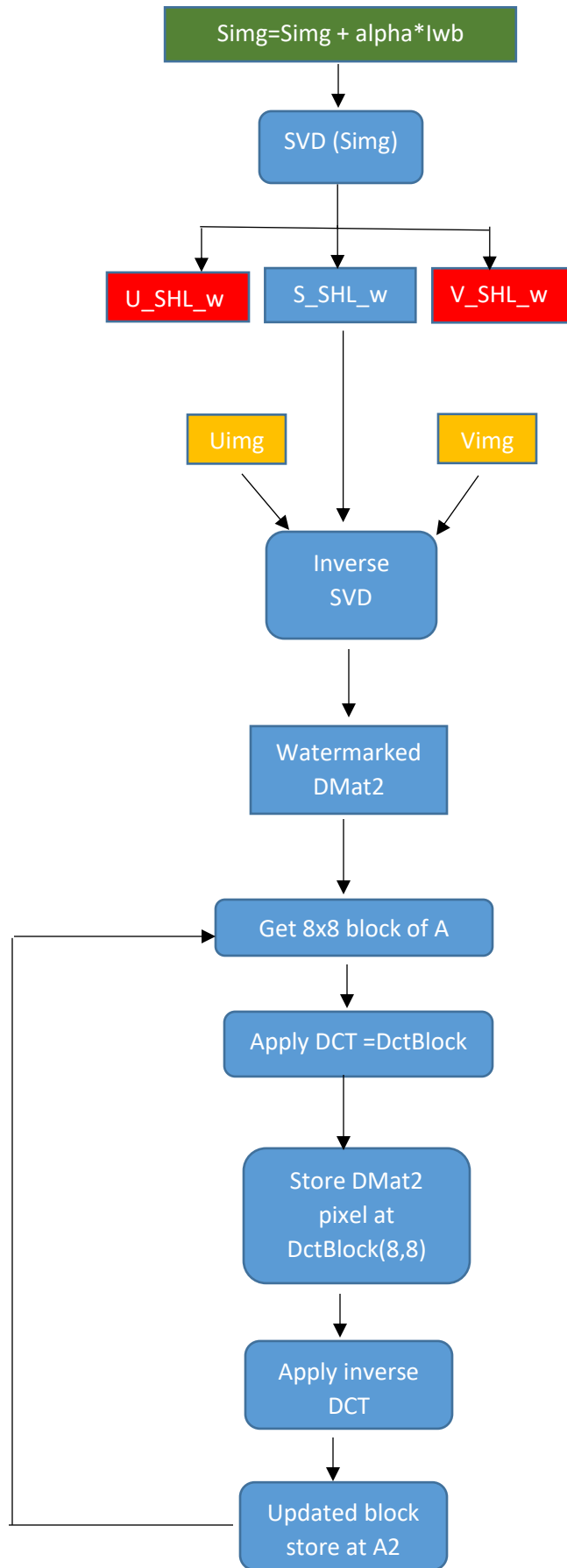
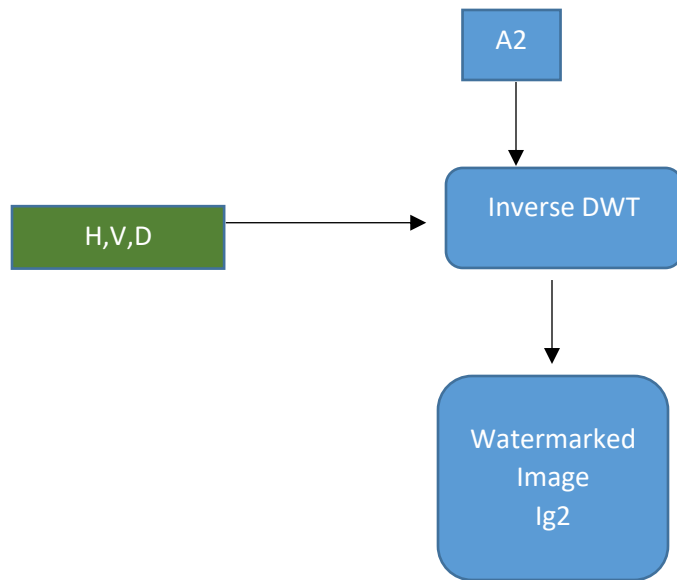


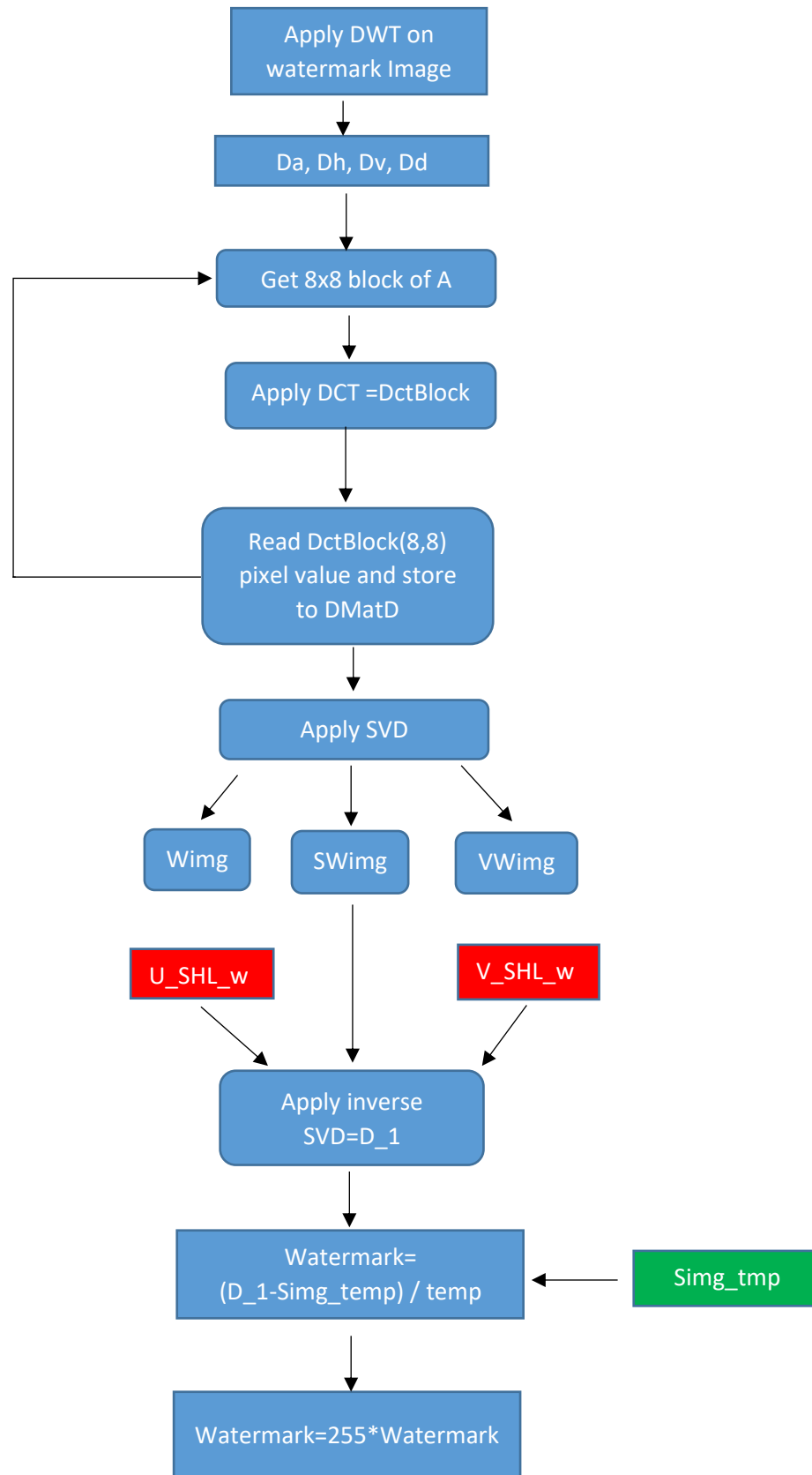
## DWT-DCT-SVD Watermarking Algorithm





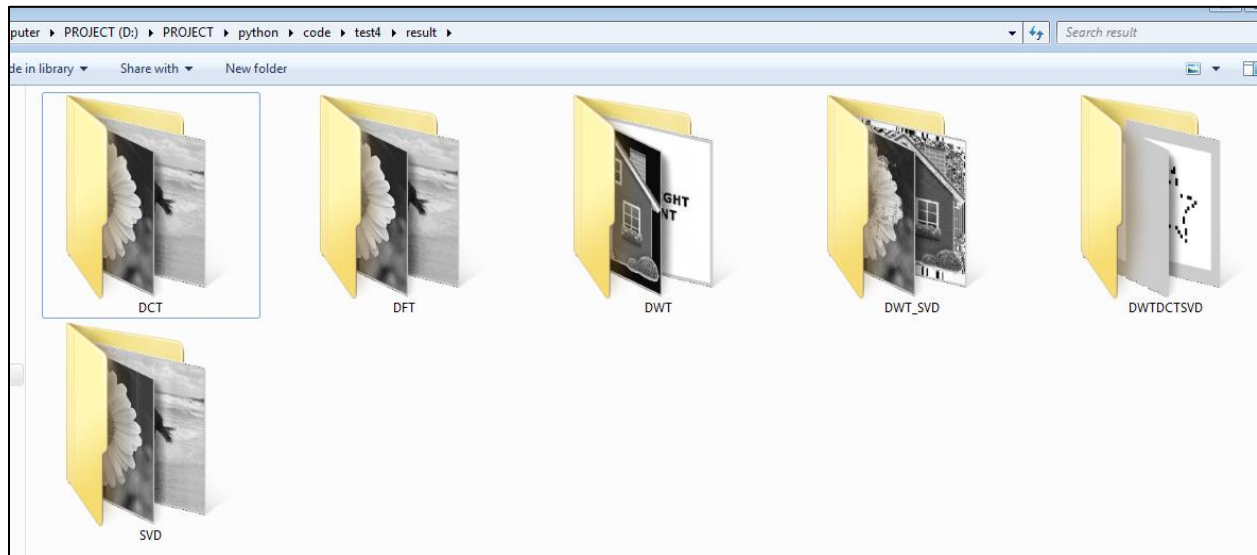


## Extraction Process:

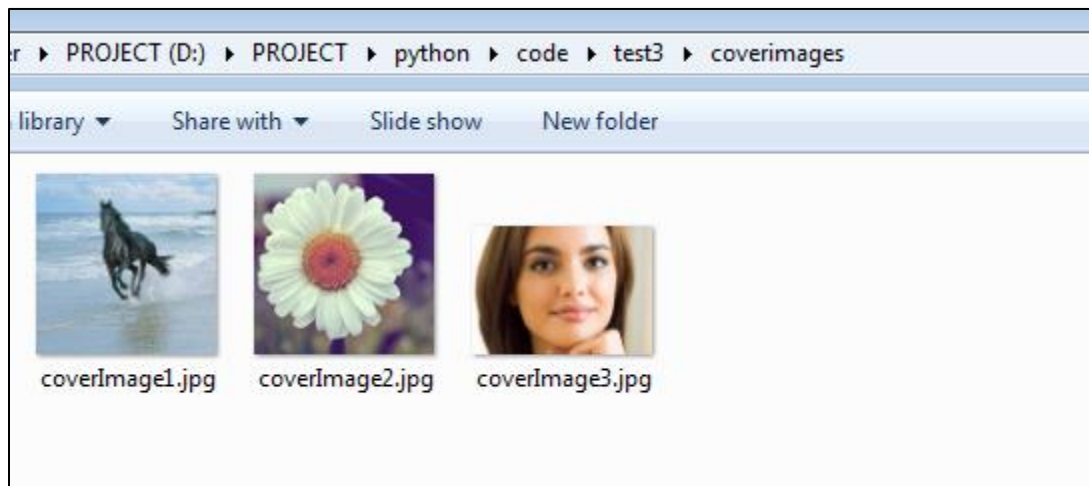


## Algorithm Detail:

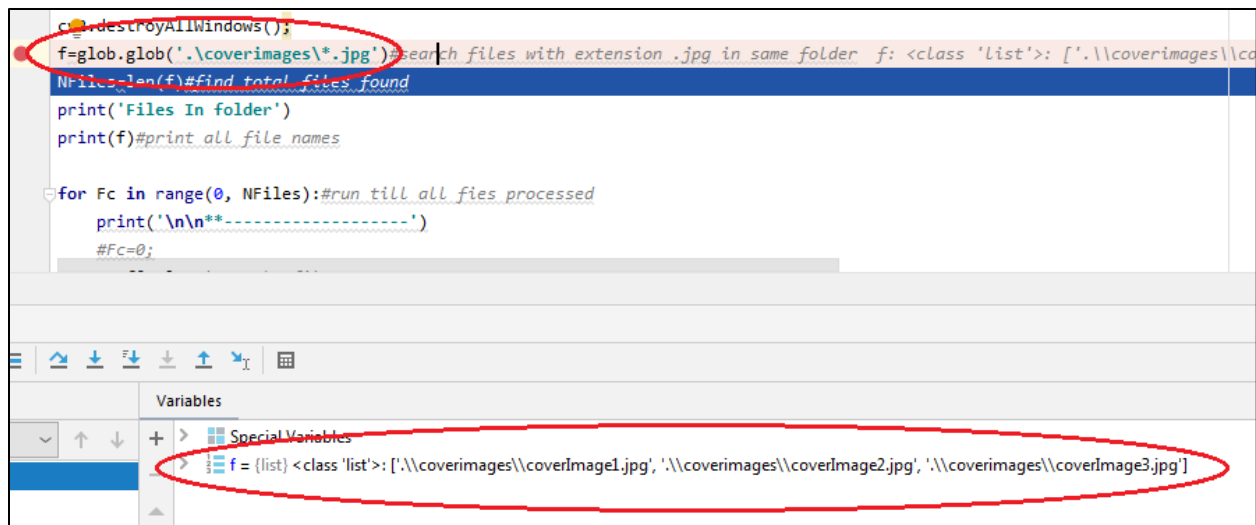
Initially individual folder created to store the output and results of each simulation methods



1. All images to be processed are stored at folder '/coverimages/' folder



2. The python script first finds all .jpg extension images of '/coverimage/' folder.



```
import sys
import os
import glob

def destroyAllWindows():
    pass

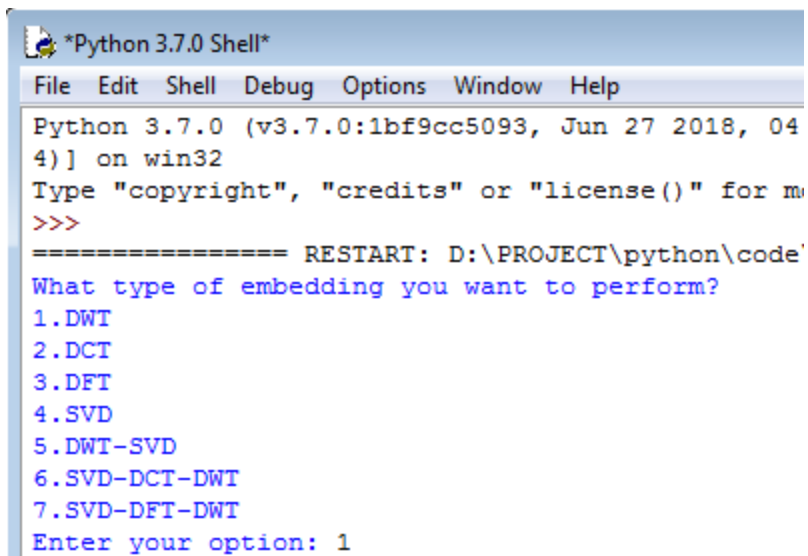
f=glob.glob('..\coverimages\*.jpg')#search files with extension .jpg in same folder f: <class 'list': ['..\coverimages\coverImage1.jpg', '..\coverimages\coverImage2.jpg', '..\coverimages\coverImage3.jpg']
NFiles=len(f)#find total files found
print('Files In folder')
print(f)#print all file names

for Fc in range(0, NFiles):#run till all files processed
    print('\n\n**-----')
    #Fc=0;
```

Variables

- Special Variables
- f = (list) <class 'list': ['..\coverimages\coverImage1.jpg', '..\coverimages\coverImage2.jpg', '..\coverimages\coverImage3.jpg']

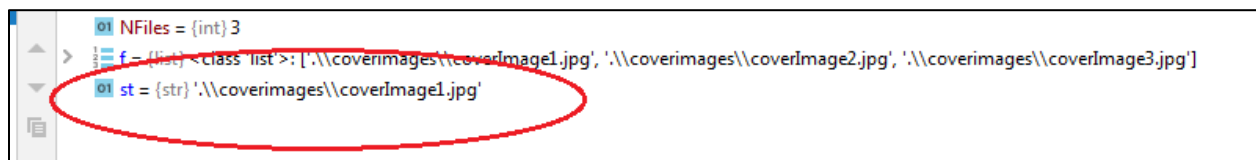
3. Algorithm ask for the option to be performed



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:44) on win32
Type "copyright", "credits" or "license()" for more
>>>
===== RESTART: D:\PROJECT\python\code'
What type of embedding you want to perform?
1.DWT
2.DCT
3.DFT
4.SVD
5.DWT-SVD
6.SVD-DCT-DWT
7.SVD-DFT-DWT
Enter your option: 1
```

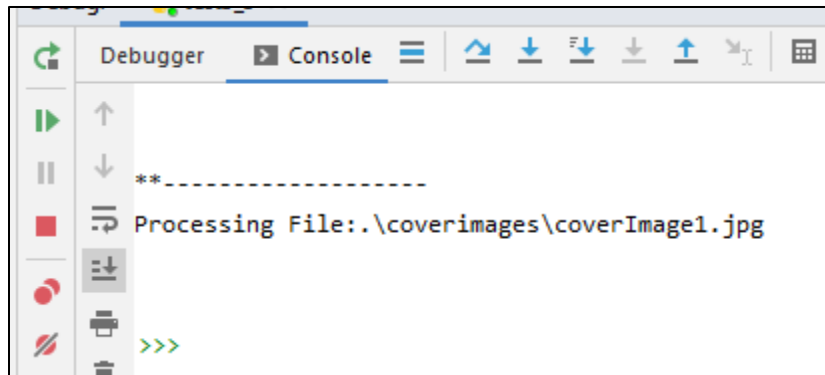
4. According to the option selected the algorithm will be applied on cover image and watermark image

5. Initially first image file coverImage1.jpg is read, files name to be read are stored at st variable



```
NFiles = (int) 3
f = (list) <class 'list': ['..\coverimages\coverImage1.jpg', '..\coverimages\coverImage2.jpg', '..\coverimages\coverImage3.jpg']
st = (str) '..\coverimages\coverImage1.jpg'
```

6. File name is displayed on Console Output



7. the algorithm used is followed as per paper

‘DWT, DCT and SVD based Digital Image, Watermarking,

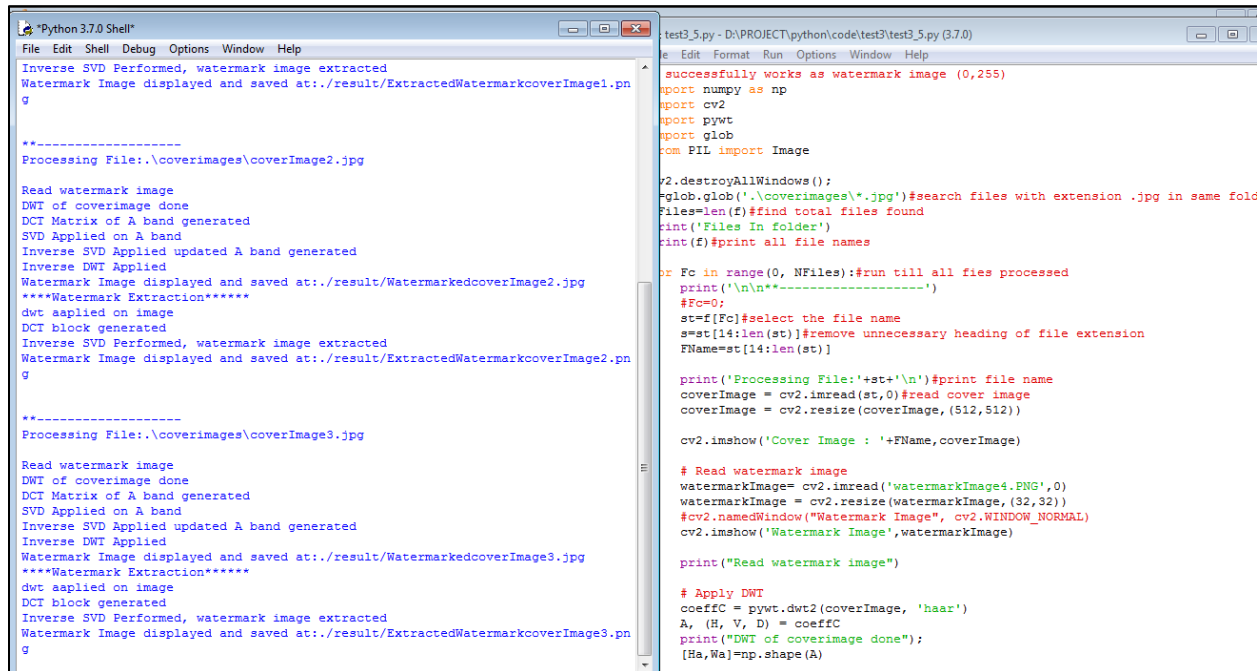
Manie Kansal, Gursharanjeet Singh, B V Kranthi

, International Conference on Computing Sciences, 2012’

As per reference research paper image is resized to 512x512 and watermark image resized to 32x32

# 1. DWT Watermarking

Console output option 1 : DWT



The image shows two side-by-side windows from a Python 3.7.0 Shell environment. The left window displays the output of a script, showing the successful extraction and watermarking of two images (coverImage2.jpg and coverImage3.jpg) using DWT. The right window shows the source code of the script, which includes file discovery, image loading, DWT application, and watermarking steps.

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Inverse SVD Performed, watermark image extracted
Watermark Image displayed and saved at:./result/ExtractedWatermarkcoverImage1.png

*****
Processing File:.\coverimages\coverImage2.jpg

Read watermark image
DWT of coverimage done
DCT Matrix of A band generated
SVD Applied on A band
Inverse SVD Applied updated A band generated
Inverse DWT Applied
Watermark Image displayed and saved at:./result/WatermarkedcoverImage2.jpg
****Watermark Extraction*****
dwt saplied on image
DCT block generated
Inverse SVD Performed, watermark image extracted
Watermark Image displayed and saved at:./result/ExtractedWatermarkcoverImage2.png

*****
Processing File:.\coverimages\coverImage3.jpg

Read watermark image
DWT of coverimage done
DCT Matrix of A band generated
SVD Applied on A band
Inverse SVD Applied updated A band generated
Inverse DWT Applied
Watermark Image displayed and saved at:./result/WatermarkedcoverImage3.jpg
****Watermark Extraction*****
dwt saplied on image
DCT block generated
Inverse SVD Performed, watermark image extracted
Watermark Image displayed and saved at:./result/ExtractedWatermarkcoverImage3.png

Python 3.7.0 Shell
test3_5.py - D:\PROJECT\python\code\test3\test3_5.py (3.7.0)
File Edit Format Run Options Window Help
successfully works as watermark image (0,255)
import numpy as np
import cv2
import pywt
import glob
from PIL import Image

cv2.destroyAllWindows();
fglob=glob('..\coverimages\*.jpg')#search files with extension .jpg in same fold
Files=len(f)#find total files found
print('Files In folder')
print(f)#print all file names

for Fc in range(0, NFiles):#run till all files processed
    print('\n\n-----')
    #Fc=0:
    st=f[Fc]#select the file name
    s=st[14:len(st)]#remove unnecessary heading of file extension
    FName=st[14:len(st)]

    print('Processing File:'+st+'\n')#print file name
    coverImage = cv2.imread(st,0)#read cover image
    coverImage = cv2.resize(coverImage, (512,512))

    cv2.imshow('Cover Image : '+FName,coverImage)

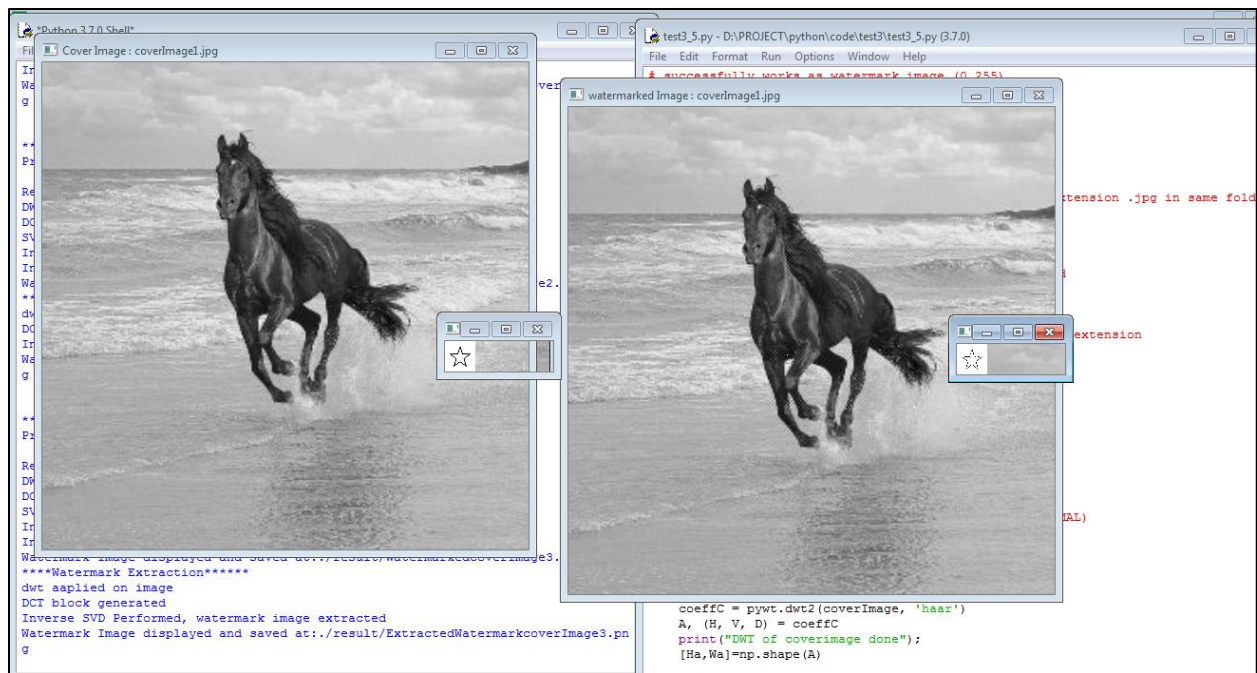
    # Read watermark image
    watermarkImage= cv2.imread('watermarkImage4.PNG',0)
    watermarkImage = cv2.resize(watermarkImage, (32,32))
    #cv2.namedWindow("Watermark Image", cv2.WINDOW_NORMAL)
    cv2.imshow('Watermark Image',watermarkImage)

    print("Read watermark image")

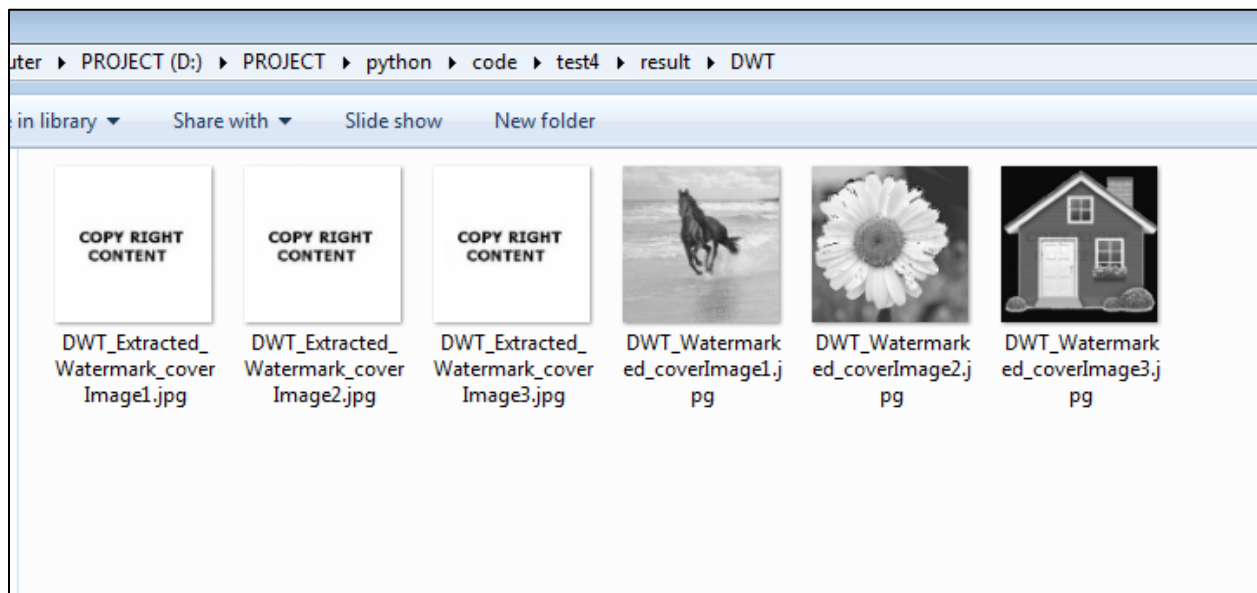
    # Apply DWT
    coeffC = pywt.dwt2(coverImage, 'haar')
    A, (H, V, D) = coeffC
    print("DWT of coverimage done");
    [Ha,Wa]=np.shape(A)
```



## Simulation screenshot cover image 1

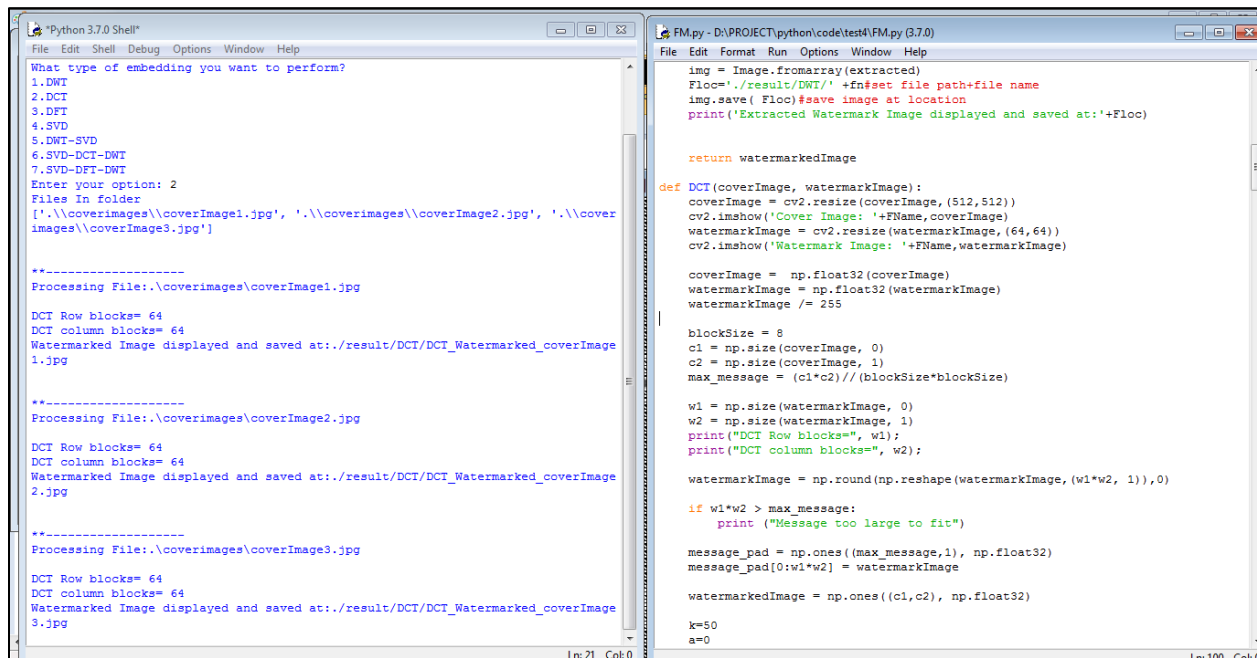


like wise above process all cover image will be processed and will be stored at 'result/DWT/' folder location



## 2. DCT Watermarking

Console output option 2 : DCT



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
What type of embedding you want to perform?
1.DWT
2.DCT
3.DFT
4.SVD
5.DWT-SVD
6.SVD-DCT-DWT
7.SVD-DFT-DWT
Enter your option: 2
Files In folder
['.\\coverimages\\coverImage1.jpg', '.\\coverimages\\coverImage2.jpg', '.\\cover
images\\coverImage3.jpg']

*****
Processing File:..\\coverimages\\coverImage1.jpg

DCT Row blocks= 64
DCT column blocks= 64
Watermarked Image displayed and saved at:./result/DCT/DCT_Watermarked_coverImage
1.jpg

*****
Processing File:..\\coverimages\\coverImage2.jpg

DCT Row blocks= 64
DCT column blocks= 64
Watermarked Image displayed and saved at:./result/DCT/DCT_Watermarked_coverImage
2.jpg

*****
Processing File:..\\coverimages\\coverImage3.jpg

DCT Row blocks= 64
DCT column blocks= 64
Watermarked Image displayed and saved at:./result/DCT/DCT_Watermarked_coverImage
3.jpg

FM.py - D:\PROJECT\python\code\test4\FM.py (3.7.0)
File Edit Format Run Options Window Help
img = Image.fromarray(extracted)
Floc='./result/DWT/' +fn#set file path+file name
img.save( Floc)#save image at location
print('Extracted Watermark Image displayed and saved at:'+Floc)

return watermarkedImage

def DCT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (512,512))
    cv2.imshow('Cover Image: '+FName,coverImage)
    watermarkImage = cv2.resize(watermarkImage, (64,64))
    cv2.imshow('Watermark Image: '+FName,watermarkImage)

    coverImage = np.float32(coverImage)
    watermarkImage = np.float32(watermarkImage)
    watermarkImage /= 255

    blockSize = 8
    c1 = np.size(coverImage, 0)
    c2 = np.size(coverImage, 1)
    max_message = (c1*c2)//(blockSize*blockSize)

    w1 = np.size(watermarkImage, 0)
    w2 = np.size(watermarkImage, 1)
    print("DCT Row blocks=", w1);
    print("DCT column blocks=", w2);

    watermarkImage = np.round(np.reshape(watermarkImage, (w1*w2, 1)),0)

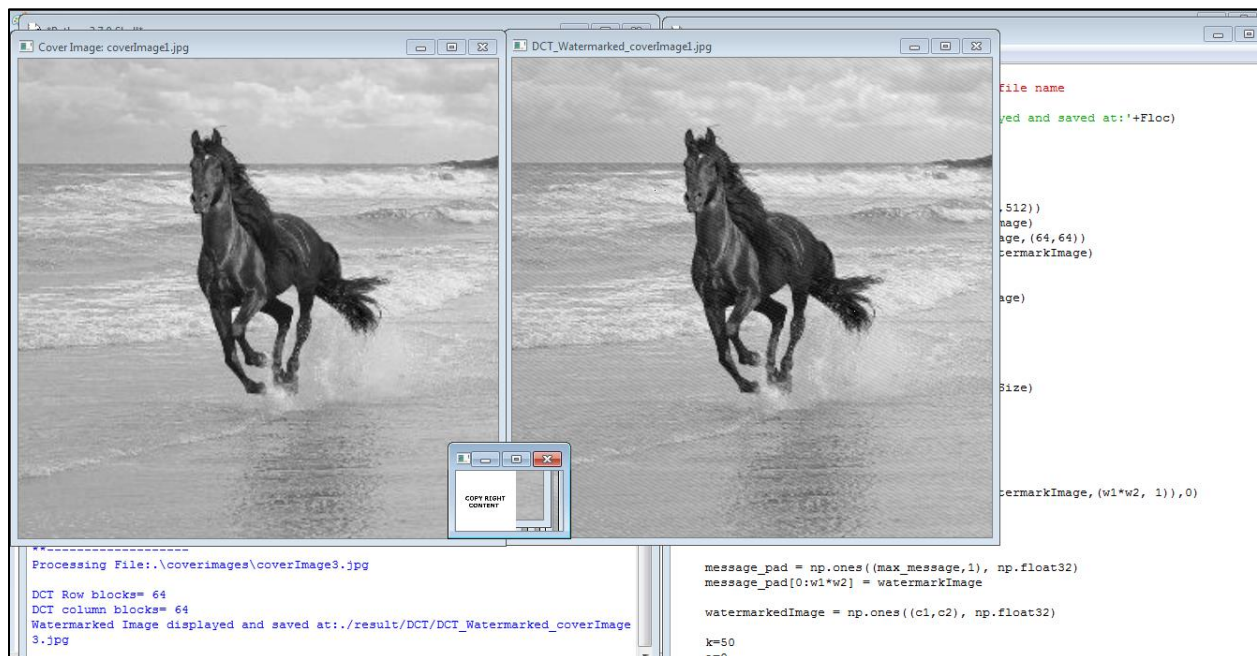
    if w1*w2 > max_message:
        print ("Message too large to fit")

    message_pad = np.ones((max_message,1), np.float32)
    message_pad[0:w1*w2] = watermarkImage

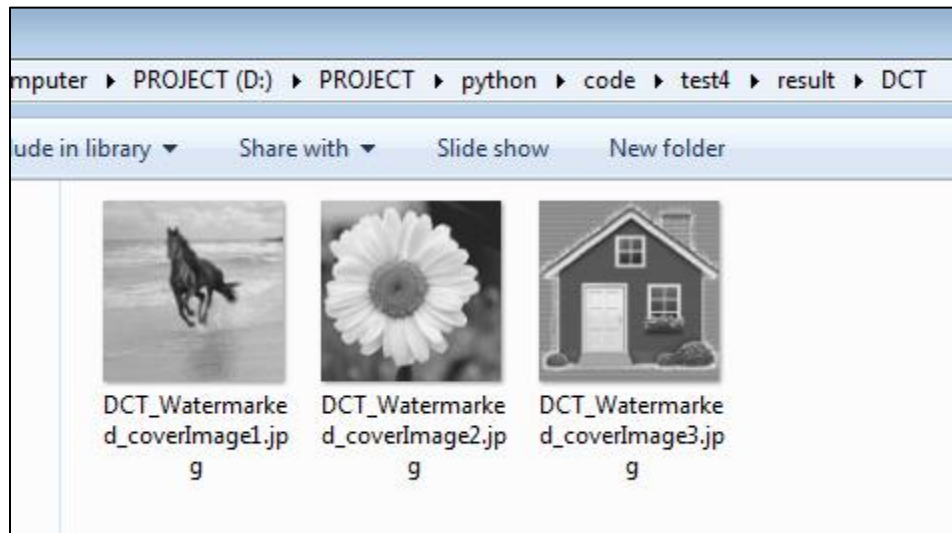
    watermarkedImage = np.ones((c1,c2), np.float32)

    k=50
    a=0
```

Simulation screenshot cover image 1



like wise above process all cover image will be processed and will be stored at 'result/DCT/' folder location



### 3. DFT Watermarking

Console output option 3 : DWT

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
2.DCT
3.DFT
4.SVD
5.DWT-SVD
6.SVD-DCT-DWT
7.SVD-DFT-DWT
Enter your option: 3
Files In folder
['.\\coverimages\\coverImage1.jpg', '.\\coverimages\\coverImage2.jpg', '.\\coverimages\\coverImage3.jpg']

*****
Processing File:..\\coverimages\\coverImage1.jpg
DFT Watermarking done
Warning (from warnings module):
  File "D:\\PROJECT\\python\\code\\test4\\FM.py", line 21
    watermarkedImage = np.uint8(watermarkedImage)
ComplexWarning: Casting complex values to real discards the imaginary part
Watermark Image displayed and saved at:./result/DFT/DFT_Watermarked_coverImage1.jpg

*****
Processing File:..\\coverimages\\coverImage2.jpg
DFT Watermarking done
Watermark Image displayed and saved at:./result/DFT/DFT_Watermarked_coverImage2.jpg

*****
Processing File:..\\coverimages\\coverImage3.jpg
DFT Watermarking done
Watermark Image displayed and saved at:./result/DFT/DFT_Watermarked_coverImage3.jpg

Ln:21 Col:0

FM.py - D:\\PROJECT\\python\\code\\test4\\FM.py (3.7.0)
File Edit Format Run Options Window Help
img = Image.fromarray(extracted)
Floc='./result/DWT/' +fn+set file path+file name
img.save(Floc)#save image at location
print('Extracted Watermark Image displayed and saved at:'+Floc)

return watermarkedImage

def DCT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (512,512))
    cv2.imshow('Cover Image: '+FName,coverImage)
    watermarkImage = cv2.resize(watermarkImage, (64,64))
    cv2.imshow('Watermark Image: '+FName,watermarkImage)

    coverImage = np.float32(coverImage)
    watermarkImage = np.float32(watermarkImage)
    watermarkImage /= 255

    blockSize = 8
    c1 = np.size(coverImage, 0)
    c2 = np.size(coverImage, 1)
    max_message = (c1*c2)/(blockSize*blockSize)

    w1 = np.size(watermarkImage, 0)
    w2 = np.size(watermarkImage, 1)
    print("DCT Row blocks=", w1);
    print("DCT column blocks=", w2);

    watermarkImage = np.round(np.reshape(watermarkImage, (w1*w2, 1)),0)

    if w1*w2 > max_message:
        print ("Message too large to fit")

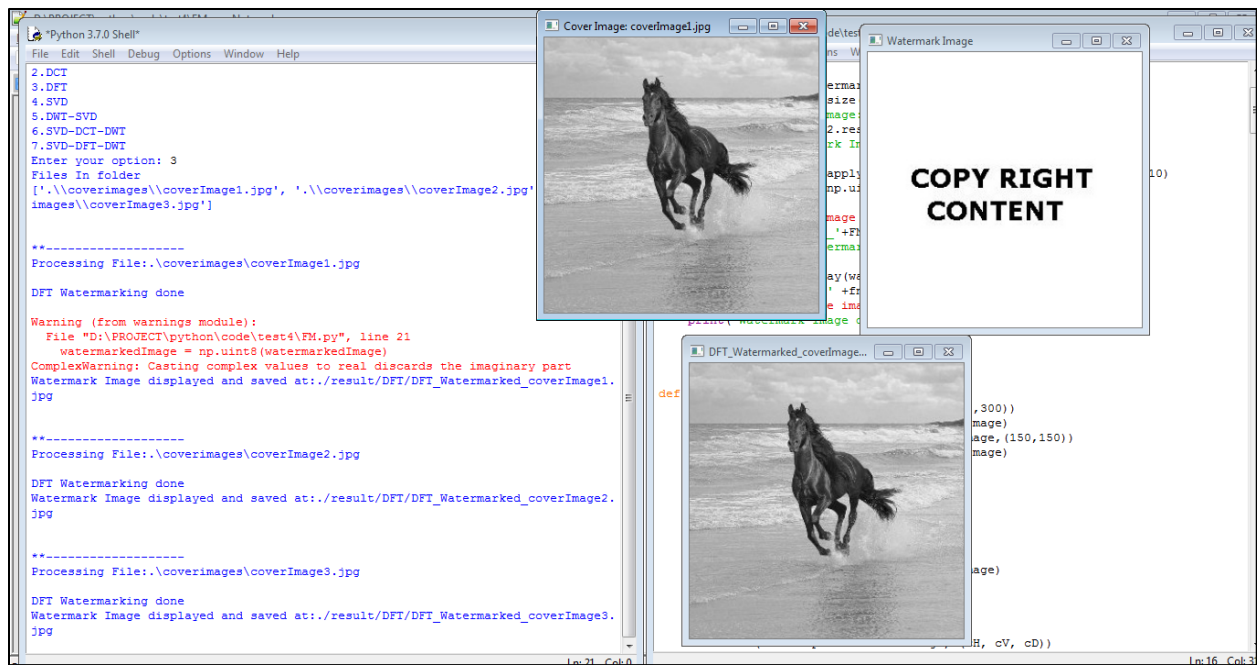
    message_pad = np.ones((max_message,1), np.float32)
    message_pad[0:w1*w2] = watermarkImage

    watermarkedImage = np.ones((c1,c2), np.float32)

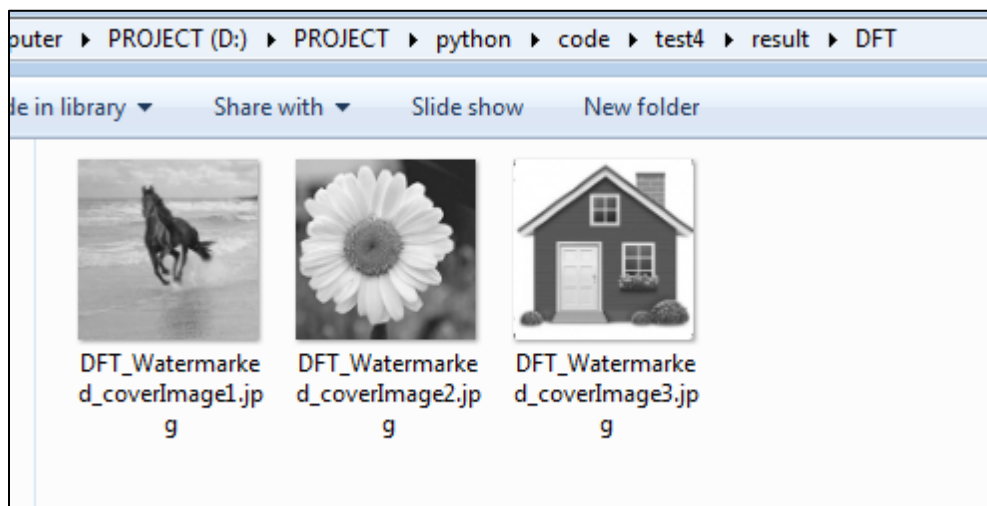
    k=50
    a=0

Ln:100 Co
```

## Simulation screenshot cover image 1



like wise above process all cover image will be processed and will be stored at 'result/DFT/' folder location



## 4. SVD Watermarking

Console output option 4 : SVD

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:\PROJECT\python\code\test4\FM.py =====
What type of embedding you want to perform?
1.DWT
2.DCT
3.DFT
4.SVD
5.DWT-SVD
6.SVD-DCT-DWT
7.SVD-DFT-DWT
Enter your option: 4
Files in folder
['..\\coverimages\\coverImage1.jpg', '..\\coverimages\\coverImage2.jpg', '..\\coverimages\\coverImage3.jpg']

*****
Processing File:..\\coverimages\\coverImage1.jpg

SVD Started
SVD Elements Generation..
Updating U matrix...
Applying Inverse SVD...
Preprocessing watermarked Image
Watermarked Image generated...
Watermark Image displayed and saved at:..\\result/SVD/SVD_Watermarked_coverImage1.jpg

*****
Processing File:..\\coverimages\\coverImage2.jpg

SVD Started
SVD Elements Generation..
Updating U matrix...
Applying Inverse SVD...
Preprocessing watermarked Image
```

```
FM.py - D:\PROJECT\python\code\test4\FM.py (3.7.0)
File Edit Format Run Options Window Help

def DFT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (300,300))
    cv2.imshow('Cover Image: ' + FName, coverImage)
    watermarkImage = cv2.resize(watermarkImage, (300,300))
    cv2.imshow('Watermark Image', watermarkImage)

    watermarkedImage = applyWatermarkDFT(coverImage, watermarkImage, 10)
    watermarkedImage = np.uint8(watermarkedImage)

    #save watermarked image
    fn='DFT_Watermarked_' + FName + set file name
    cv2.imshow('DFT_Watermarked_' + FName, watermarkedImage)

    img = Image.fromarray(watermarkedImage)
    Floc='./result/DFT/' + fn + set file path + file name
    img.save(Floc) #save image at location
    print('Watermark Image displayed and saved at: ' + Floc)

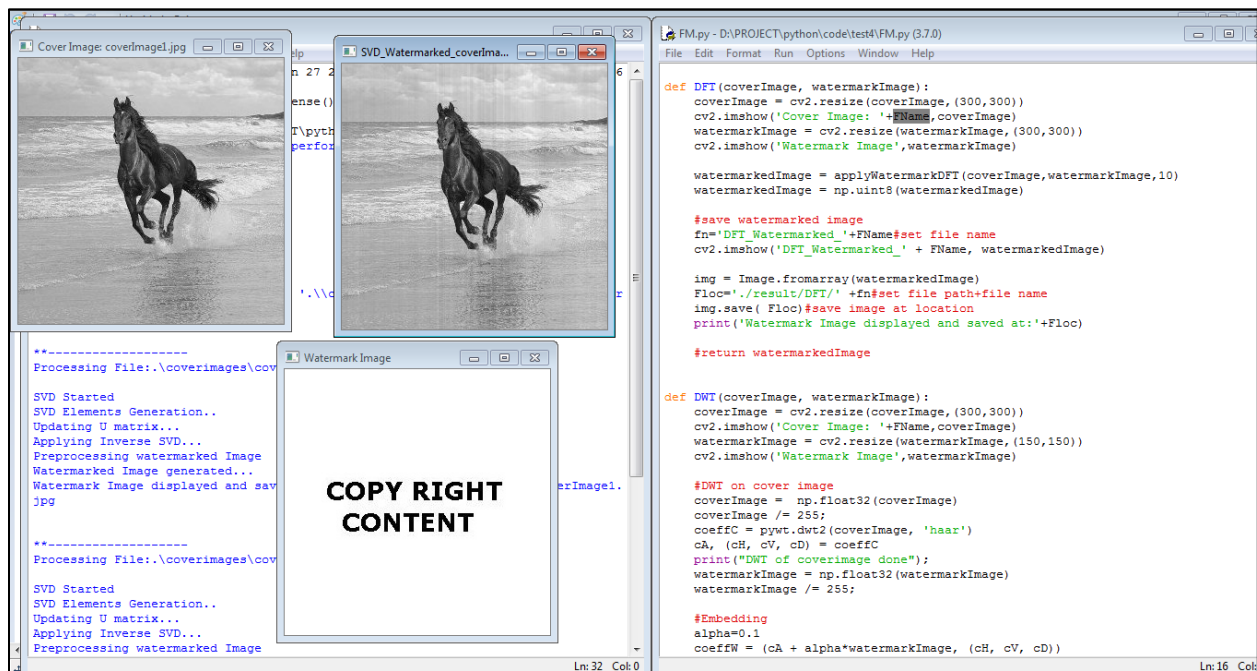
    #return watermarkedImage

def DWT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (300,300))
    cv2.imshow('Cover Image: ' + FName, coverImage)
    watermarkImage = cv2.resize(watermarkImage, (150,150))
    cv2.imshow('Watermark Image', watermarkImage)

    #DWT on cover image
    coverImage = np.float32(coverImage)
    coverImage /= 255;
    coeffC = pywt.dwt2(coverImage, 'haar')
    cA, (cH, cV, cD) = coeffC
    print("DWT of coverimage done");
    watermarkImage = np.float32(watermarkImage)
    watermarkImage /= 255;

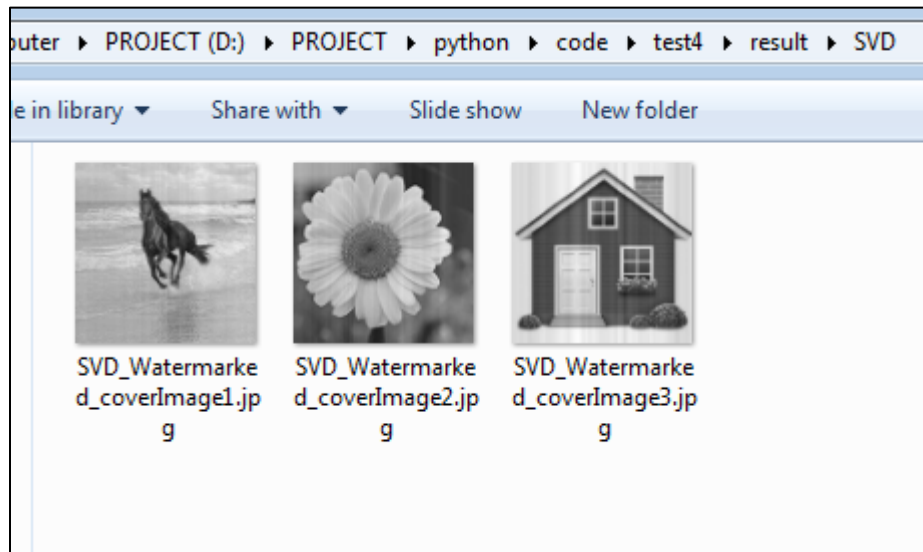
    #Embedding
    alpha=0.1
    coeffW = (cA + alpha*watermarkImage, (cH, cV, cD))
```

Simulation screenshot cover image 1





like wise above process all cover image will be processed and will be stored at 'result/SVD/' folder location



## 5. DWT-SVD Watermarking

Console output option 5 : DWT-SVD

```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD64)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:\PROJECT\python\code\test4\FM.py =====
What type of embedding you want to perform?
1.DWT
2.DCT
3.DFT
4.SVD
5.DWT-SVD
6.SVD-DCT-DWT
7.SVD-DFT-DWT
Enter your option: 5
Files In folder
['.\\coverimages\\coverImage1.jpg', '.\\coverimages\\coverImage2.jpg', '.\\coverimages\\coverImage3.jpg']

*****
Processing File:..\\coverimages\\coverImage1.jpg

DWT applied on CoverImage
SVD applied on cA...
SVD applied on cH...
SVD applied on cV...
SVD applied on cD...
SVD applied on watermark image...
Watermark image embedded in SVD of each band..
Applying inverse SVD of each band and dwt components generated...
Inverse DWT Applied to get Watermarked Image...
Watermark Image displayed and saved at:..\\result\\DWT_SVD\\DWT_SVD_Watermarked_coverImage1.jpg

*****
Processing File:..\\coverimages\\coverImage2.jpg

DWT applied on CoverImage
SVD applied on cA...
```

```
FM.py - D:\PROJECT\python\code\test4\FM.py (3.7.0)
File Edit Format Run Options Window Help

def DFT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (300,300))
    cv2.imshow('Cover Image: '+FName,coverImage)
    watermarkImage = cv2.resize(watermarkImage, (300,300))
    cv2.imshow('Watermark Image',watermarkImage)

    watermarkedImage = applyWatermarkDFT(coverImage,watermarkImage,10)
    watermarkedImage = np.uint8(watermarkedImage)

    #save watermarked image
    fn='DFT_Watermarked_'+FName#set file name
    cv2.imshow('DFT_Watermarked_' + FName, watermarkedImage)

    img = Image.fromarray(watermarkedImage)
    Floc='../result/DFT/' +fn#set file path+file name
    img.save( Floc)#save image at location
    print('Watermark Image displayed and saved at:'+Floc)

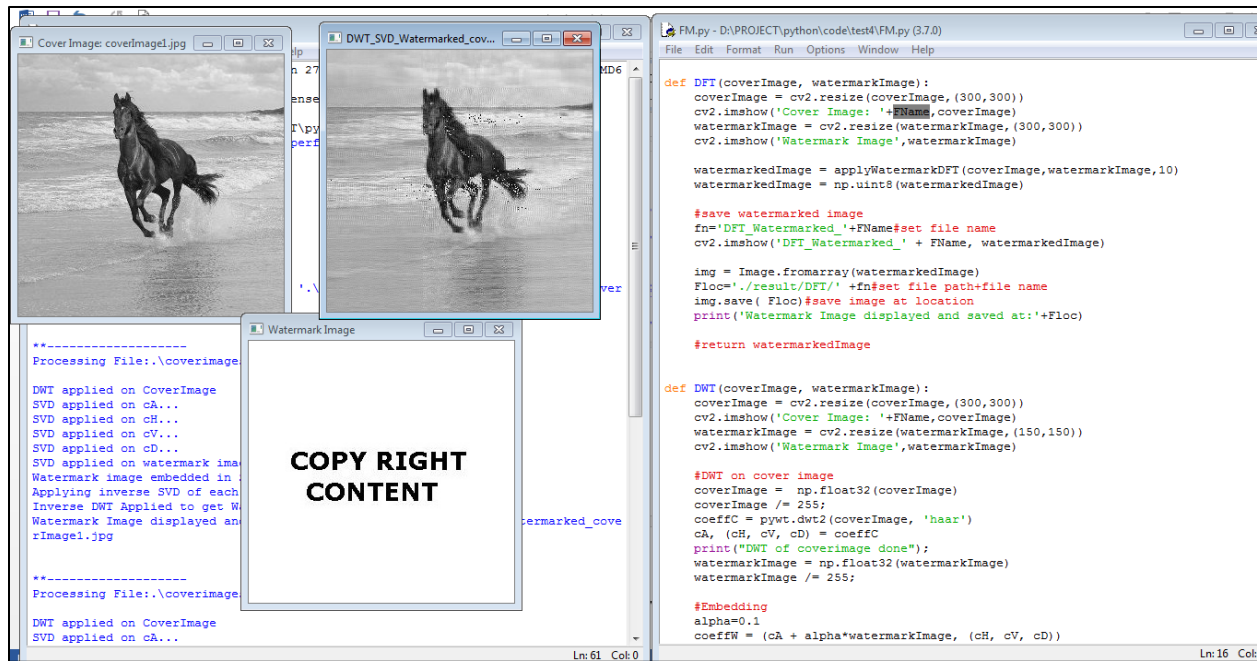
    #return watermarkedImage

def DWT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (300,300))
    cv2.imshow('Cover Image: '+FName,coverImage)
    watermarkImage = cv2.resize(watermarkImage, (150,150))
    cv2.imshow('Watermark Image',watermarkImage)

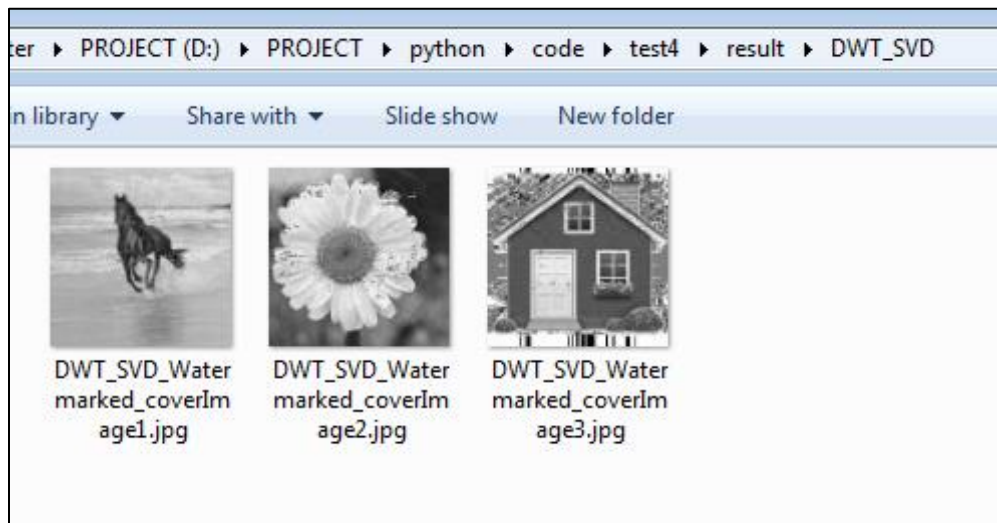
    #DWT on cover image
    coverImage = np.float32(coverImage)
    coverImage /= 255;
    coeffC = pywt.dwt2(coverImage, 'haar')
    cA, (cH, cV, cD) = coeffC
    print("DWT of coverimage done");
    watermarkImage = np.float32(watermarkImage)
    watermarkImage /= 255;

    #Embedding
    alpha=0.1
    coeffW = (cA + alpha*watermarkImage, (cH, cV, cD))
```

## Simulation screenshot cover image 1

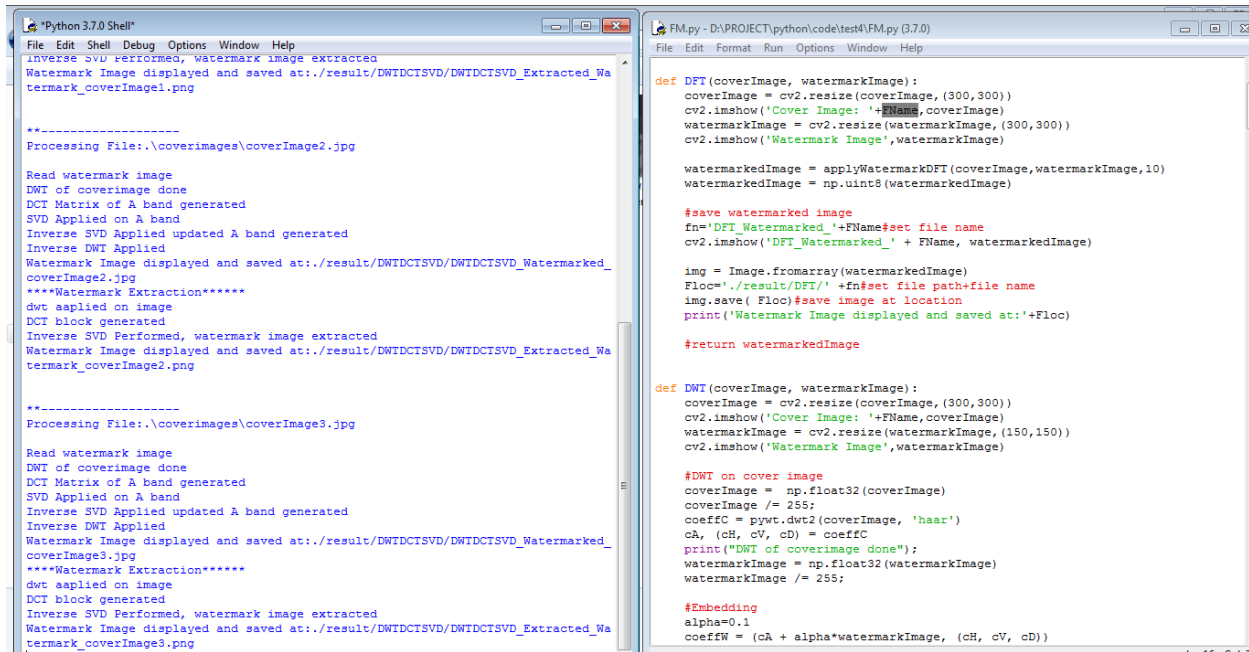


like wise above process all cover image will be processed and will be stored at 'result/DWT\_SVD/' folder location



## 6. DWT-DCT-SVD Watermarking

### Console output option 6 : DWT-SVD



```
Python 3.7.0 Shell
File Edit Shell Debug Options Window Help
Inverse SVD Performed, watermark image extracted
Watermark Image displayed and saved at:./result/DWTDCTSVSD/DWTDCTSVSD_Extracted_Wa
termark_coverImage1.png

*****
Processing File:.\coverimages\coverImage2.jpg

Read watermark image
DWT of coverimage done
DCT Matrix of A band generated
SVD Applied on A band
Inverse SVD Applied updated A band generated
Inverse DWT Applied
Watermark Image displayed and saved at:./result/DWTDCTSVSD/DWTDCTSVSD_Watermarked_
coverImage2.jpg
***Watermark Extraction*****
dwt applied on image
DCT block generated
Inverse SVD Performed, watermark image extracted
Watermark Image displayed and saved at:./result/DWTDCTSVSD/DWTDCTSVSD_Extracted_Wa
termark_coverImage2.png

*****
Processing File:.\coverimages\coverImage3.jpg

Read watermark image
DWT of coverimage done
DCT Matrix of A band generated
SVD Applied on A band
Inverse SVD Applied updated A band generated
Inverse DWT Applied
Watermark Image displayed and saved at:./result/DWTDCTSVSD/DWTDCTSVSD_Watermarked_
coverImage3.jpg
***Watermark Extraction*****
dwt applied on image
DCT block generated
Inverse SVD Performed, watermark image extracted
Watermark Image displayed and saved at:./result/DWTDCTSVSD/DWTDCTSVSD_Extracted_Wa
termark_coverImage3.png

Python 3.7.0 Shell
File Edit Format Run Options Window Help
def DFT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (300,300))
    cv2.imshow('Cover Image: '+FName,coverImage)
    watermarkImage = cv2.resize(watermarkImage, (300,300))
    cv2.imshow('Watermark Image',watermarkImage)

    watermarkedImage = applyWatermarkDFT(coverImage,watermarkImage,10)
    watermarkedImage = np.uint8(watermarkedImage)

    #save watermarked image
    fn='DFT_Watermarked_'+FName#set file name
    cv2.imshow('DFT_Watermarked_'+FName, watermarkedImage)

    img = Image.fromarray(watermarkedImage)
    Floc='./result/DFT/'+fn#set file path+file name
    img.save( Floc)#save image at location
    print('Watermark Image displayed and saved at:'+Floc)

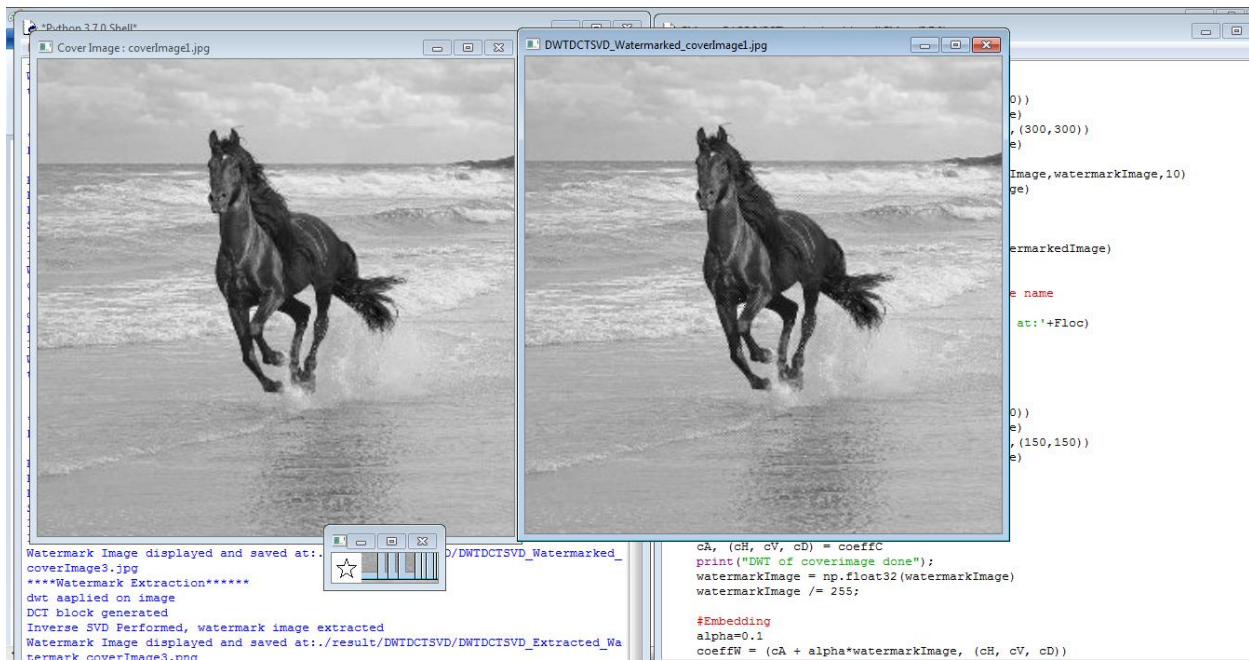
    #return watermarkedImage

def DWT(coverImage, watermarkImage):
    coverImage = cv2.resize(coverImage, (300,300))
    cv2.imshow('Cover Image: '+FName,coverImage)
    watermarkImage = cv2.resize(watermarkImage, (150,150))
    cv2.imshow('Watermark Image',watermarkImage)

    #DWT on cover image
    coverImage = np.float32(coverImage)
    coverImage /= 255;
    coeffC = pywt.dwt2(coverImage, 'haar')
    cA, (cH, cV, cD) = coeffC
    print("DWT of coverimage done");
    watermarkImage = np.float32(watermarkImage)
    watermarkImage /= 255;

    #Embedding
    alpha=0.1
    coeffW = (cA + alpha*watermarkImage, (cH, cV, cD))
```

### Simulation screenshot cover image 1





like wise above process all cover image will be processed and will be stored at 'result/DWT\_SVD/' folder location

