CS7.505 - Assignment 0
OpenCV and Chroma Keying

Avneesh Mishra

January 9, 2022

Contents

1 Installing and Testing OpenCV

Main docs: `https://docs.opencv.org/4.x/`
Python docs: `https://docs.opencv.org/4.x/d6/d00/tutorial_py_root.html`

1.1 OpenCV Setup

The procedure to setup OpenCV using Anaconda on windows is listed below

1. Download and install Anaconda from here.
2. Setup the anaconda environment for the shell using `conda init`
3. Setup the environment

   Create the environment

   ```
   conda create -yn "cv-cs7-505"
   conda activate cv-cs7-505
   ```

   Install Python 3.9 into it

   ```
   conda install python=3.9
   ```

   Install the essential packages (before OpenCV)

   ```
   conda install numpy jupyterlab
   ```

4. Install OpenCV using `pip`

   Install using pip

   ```
   pip install opencv-python opencv-contrib-python
   ```

1.2 Testing Installation of OpenCV

Checking Version

Check version using the script below

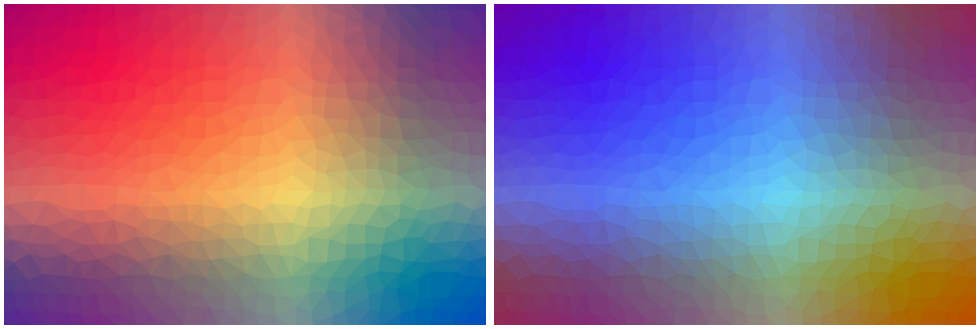
```python
# Check OpenCV version
# %% Import everything
import cv2 as cv
import numpy as np

# %% Main entrypoint
if __name__ == "__main__":
    print(f"OpenCV version: {cv.__version__}")
    print(f"Numpy version: {np.__version__}")

# %%
```

The output of the above script is

```
OpenCV version: 4.5.5
Numpy version: 1.21.2
```



(a) Input (b) Output

Figure 1: Test images
The red and blue channels are flipped (their layer intensities are swapped). Check code listing 1

Reading and changing color channels

Consider the images in figure 1. The code to get this output is given below

```
# Change the color channels of an image
"""
    Flips the Reg and Blue channel of an image
"""

# %% Import everything
import cv2 as cv
import numpy as np

# %% Functions
# Show image in a window
def show_img(img, win_name = "Image"):
    cv.namedWindow(win_name, cv.WINDOW_GUI_EXPANDED)
    while True:
        cv.imshow(win_name, img)
        if cv.waitKey(1) == ord('q'):
            break
    cv.destroyWindow(win_name)

# %% Main module
if __name__ == "__main__":
    # Read image
    img_in = cv.imread("./images/test.jpg")
    # Show image
    show_img(img_in, "Input")
    img_out = img_in[:,:,::-1]
    show_img(img_out, "Output")
    # Save output
    cv.imwrite("./images/output.jpg", img_out)

# %%
```

Listing 1: Transform images

2 Chroma Keying with OpenCV

2.1 Video and Images

Video to images

Download video from here and store as `./videos/vtest.avi`. Then run the following

```
python .\vid_to_imgs.py -n 10
```

The code is shown in listing 2. Output is in figure 2.

```python
# Convert a video to images
"""
    Given a video, generate the images. Run as main.
"""

# %% Import everything
import cv2 as cv
import numpy as np
import argparse
import sys
import os

# %% Argument parser
parser = argparse.ArgumentParser(
    formatter_class=argparse.ArgumentDefaultsHelpFormatter)
parser.add_argument('-i', '--vid-file', default="./videos/vtest.avi",
    help="Video file to read")
parser.add_argument('-n', '--num-imgs', default=0, type=int,
    help="The maximum number of images to output from video (0=all)")
parser.add_argument('-o', '--out-prefix', default="./out/img",
    type=str, help="Output prefix for images")

# %% Main entrypoint
if __name__ == "__main__":
    # Parse all (known) arguments
    args, unknown_args = parser.parse_known_args(sys.argv)
    # Check if output directory (if passed) exists
    out_dir = os.path.split(args.out_prefix)[0]
    if not os.path.isdir(out_dir):
        print(f"Folder {out_dir} is being created")
        os.makedirs(out_dir)
    # Read video from file
    cap = cv.VideoCapture(args.vid_file)
    try:
        # Read frames
        fnum = 0
        while cap.isOpened():
            ret, frame = cap.read()
            if not ret:
                print("Probably EOF reached!")
                break
            cv.imshow("Video Feed", frame)
            if cv.waitKey(100) == ord('q'):
                print(f"Break encountered")
                break
            if args.num_imgs == 0 or fnum < args.num_imgs:
                # Write to disk
                cv.imwrite(f"{args.out_prefix}{fnum+1}.jpg", frame)
                fnum += 1
            else:
                print(f"Reached {fnum} frames")
                break
        print(f"Wrote {fnum} frames under {args.out_prefix}*.jpg")
    finally:
        # Cleanup
        cap.release()
        cv.destroyAllWindows()

# %%
```

Listing 2: vid_to_imgs.py

(a) Image 1 (b) Image 5 (c) Image 10

Figure 2: Sequence of images
First, fifth, and tenth image in the 10-image sequence produced by listing 2.

The above script (listing 2) gives the following output (along with a GUI window to show the images of the video)

```
Folder ./out is being created
Reached 10 frames
Wrote 10 frames under ./out/img*.jpg
```

Learned the following things in this question

- Using argparse module to parse arguments passed to a script
- Reading a video file (reference from tutorial)