videofiltresampkey

March 7, 2017

1 Videofiltresampkey

Program to capture a video from a camera, filter it and display it live on the screen.

```
-Gerald Schuller, October 2014
```

- **Import the relevant modules:**

```
In [1]: import numpy as np
        import scipy.signal
        import cv2
```

- **Downsample factor:**

```
In [2]: N = 8
```

```
In [3]: cap = cv2.VideoCapture(0)
        [retval, frame] = cap.read()
        [r,c,d] = frame.shape
        print(r,c)
        Dsy = np.zeros((r,c))
```

```
(480, 640)
```

- **2D-Mask to set to zero the 7/8 highest frequencies, only kep the 1/8 lowest frequencies in each direction:**

```
In [4]: M = np.ones((r,c))
        M[(r/16.0):(r-r/16.0),(c/16.0):(c-c/16.0)] = np.zeros((7.0/8.0*r,7.0/8.0*c))
```

- **Conmpute space-domain/inverse 2D Fourier transform of Low Pass filter:**

```
In [5]: h = np.abs(np.fft.ifft2(M))
        hc = np.concatenate((h[:,(c/2):c], h[:,0:(c/2)]), axis=1)
        hc = np.concatenate((hc[(r/2):r,:], hc[0:(r/2),:]))
```

- **Only keep the part with the biggest components to reduce computations:**

```
In [6]: hc = hc[(r/2-4):(r/2+4), (c/2-4):(c/2+4)]
        #High pass filter kernel for edge detection:
        #filt=np.matrix([[-1.0,-1.0,-1],[-1,8,-1],[-1,-1,-1]])/1.0;
```

- **Low Pass Kernel:**
- ** *Rectangular filter kernel:* **

```
In [7]: filt1 = np.ones((8,8))/8;
```

- **Low Pass Kernel:**
- ** *Triangular filter kernel:* **

```
In [8]: filt2 = scipy.signal.convolve2d(filt1, filt1)/8

        filteron = False
```

- **Filter type:**

```
In [9]: rectfilt = True

In [10]: while(True):
             # Capture frame-by-frame
             [ret, frame] = cap.read()
             Y = (0.114*frame[:,:,0] + 0.587*frame[:,:,1] + 0.299*frame[:,:,2])/256;
             cv2.imshow('Original Y Signal',Y)

             if rectfilt == True:
                 filt = filt1
             else:
                 filt = filt2

             if filteron == True:
                 Y = scipy.signal.convolve2d(Y,filt,mode='same')
             #Downsample filtered frame:
             Dsy[0::N,::N] = Y[0::N,::N];
             # Display the resulting filtered frame
             #cv2.imshow('Y LP filtered, down- and upsampled',Dsy)
             #low pass filter the downsampled version to fill picture:

             if filteron == True:
                 yfilt = scipy.signal.convolve2d(Dsy,filt,mode='same')
             else:
                 yfilt = Dsy.copy()
             #print(Dsy[0:8,0:8])
             cv2.putText(yfilt,"Down - and upsampling and LP filtering Demo", (20,50), cv2.FONT_
             cv2.putText(yfilt,"Toggle LP filter on/off: key f", (20,100), cv2.FONT_HERSHEY_SIMP
             cv2.putText(yfilt,"Toggle rect and triang filt. kernel: key t", (20,150), cv2.FONT_
             cv2.putText(yfilt,"Quit: key q", (20,200), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255,128,
             cv2.imshow('(LP filtered,) down- and upsampled (, and LP filtered)',yfilt)
```

```
    key=cv2.waitKey(1) & 0xFF;
    if key == ord('f'):
        filteron = not filteron;
    if key == ord('t'):
        rectfilt = not rectfilt;
    if key == ord('q'):
        break

# When everything done, release the capture
cap.release()
cv2.destroyAllWindows()
```