videofft0ifftresampleykey

March 7, 2017

1 Videofft0ifftresampleykey

Program to capture a video from a camera, compute the Y-component, downsample it by a factor of N horizontally and vertically, and display it live on the screen. With keyboard switchable low pass filter and sampling key f toggles the low pass filter, key s the sampling. With explanation text and state display in the image windows.

-Gerald Schuller, April. 2015

- ** Import the relevant modules:**

```
In [ ]: import numpy as np
        import cv2

        cap = cv2.VideoCapture(0)

        import numpy as np
        import cv2
        #import scipy.signal

        cap = cv2.VideoCapture(0)
```

- **Low Pass Kernel:**
- **Downsampling factor N:**

```
In [ ]: N = 4
        [ret, frame] = cap.read()
        [rows,cols,c] = frame.shape
        r = rows
        c = cols
        Ds0 = np.zeros((rows,cols))
        Ds = Ds0
```

- **Mask to set to zero the 7/8 highest frequencies, only keep the 1/8 lowest frequencies in each direction: For rows:**

```
In [ ]: Mr = np.ones((r,1))
        Mr[(r/8.0):(r-r/8.0),0] = np.zeros((3.0/4.0*r))
```

- **For columns:**

```
In [ ]: Mc = np.ones((1,c))
        Mc[0,(c/8.0):(c-c/8)] = np.zeros((3.0/4.0*c));
```

- **Together:**

```
In [ ]: M = np.dot(Mr,Mc)

        ytext=np.zeros((rows,cols))
        cv2.putText(ytext,"Down- and upsampling and LP filtering Demo", (20,50), cv2.FONT_HERSHE
        cv2.putText(ytext,"Toggle LP filter in 2D-FFT on/off: key f", (20,100), cv2.FONT_HERSHEY
        cv2.putText(ytext,"Toggle sampling on/off: key s", (20,150), cv2.FONT_HERSHEY_SIMPLEX, 0
        cv2.putText(ytext,"Quit: key q", (20,200), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255))


        filteron=False;
        samplingon=False;

        while(True):
            #Encoding side:
            # Capture frame-by-frame
            [ret, frame] = cap.read()

            #Berechnung der Luminanz-Komponente Y:
            # Y= 0.114*B+0.587*G+0.299*R :
            # /256 because the result is float values which imshow expects in range 0...1:
            Y=(0.114*frame[:,:,0]+0.587*frame[:,:,1]+0.299*frame[:,:,2])/256;

            cv2.imshow('Encoder, Original: Luminance Y',Y+ytext)

            if filteron==True:
               #2D-FFT of Y
               X=np.fft.fft2(Y)
               #Set to zero the 7/8 highest spatial frequencies in each direction:
               X=X*M
               #inverse 2D-FFT:
               Y=np.abs(np.fft.ifft2(X))

            if samplingon==True:
                #Downgesamplets Y0, nur jedes Nte pixel horizontal und vertikal wir uebertragen:
                Y0=np.zeros((rows,cols));
                Y0[0::N,::N]=Y[0::N,::N];
                Y=Y0.copy()

            #Decoding Side
            #Make text:
            ytext2=np.zeros((rows,cols))
```

```
            if samplingon:
                cv2.putText(ytext2,"Sampling on", (20,20), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0.9))
                #print("sampling on")
            else:
                cv2.putText(ytext2,"Sampling off", (20,20), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (0.9)
                #print("sampling off")

            if filteron:
                cv2.putText(ytext2,"Filter on", (20,50), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255))
                #print("filter on")
            else:
                cv2.putText(ytext2,"Filter off", (20,50), cv2.FONT_HERSHEY_SIMPLEX, 0.8, (255))
                #print("filter off")

            #2D-DFT:
            Dsfft=np.fft.fft2(Y)

            if filteron == True:
               #Lowpass filter the sampled frame:
               #Dsfilt=N*scipy.signal.convolve2d(Ds0,filt,mode='same')
               #Set to zero the 7/8 highest spacial frequencies in each direction:
               Dsfft=Dsfft * M
            #scaling to maintain the energy after sampling and filtering
            if samplingon and filteron:
                Dsfft=Dsfft*N*N
            cv2.imshow('2D Discrete Fourier Transform of (downsampled, filtered) Luminance Y',np

            #inverse 2D-FFT:
            Y=np.abs(np.fft.ifft2(Dsfft))

            cv2.imshow('Decoder: reconstructed Luminance Y',Y+ytext2)

            #Ende durch Taste "q":
            key=cv2.waitKey(1) & 0xFF;
            if key == ord('s'):
                samplingon = not samplingon;
            if key == ord('f'):
                filteron = not filteron;
            if key == ord('q'):
                break
```

- **When everything done, release the capture**

```
In [ ]: cap.release()
        cv2.destroyAllWindows()
```