CSE 344: Computer Vision

Homework 3 ; Arka Sarkar 2018222

Question 1

Steps to implement background subtraction :

1. Calculate background frame → Mean, Median, Mode methods
2. Calculate Frame difference of all the frames.
3. Take otsu threshold of all the frames

Algorithm

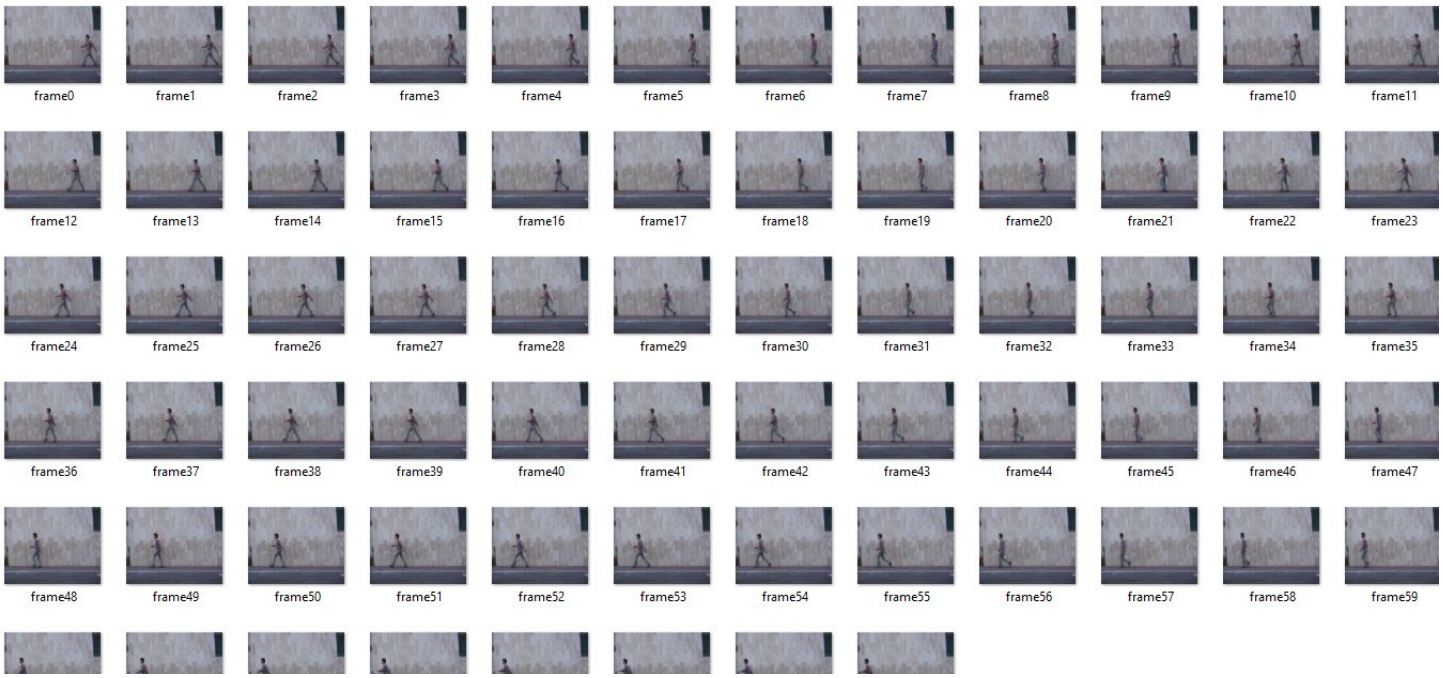
```
def calculate_background(frames, method = "mean"):
        if(method == "mean"):
                mean = np.zeros(frames[0].shape)

                for frame in frames:
                mean = mean + frame

                mean = mean//len(frames)
                return mean
        elif(method == "median"):
                median = np.stack([f for f in frames])
                median = np.median(median, axis = 0)
                return median
        elif(method == "mode"):
                mode = np.stack([f for f in frames])
                mode = stats.mode(mode, axis = 0)
                mode = np.squeeze(mode[0])
                return mode
        else:
                raise Exception("Not a valid method")


def background_subtraction( frames, method = "mean", loc = 0):

        if(method == "mean"):
                mean = calculate_background(frames, "mean")

                count = 0
                for frame in frames:
                curr = abs(frame - mean)
                thres = np.mean(curr,2)
                val = filters.threshold_otsu(curr)
                thres[thres >= val] = 255
                thres[thres <val] = 0
                B = curr[:,:,0]
                G = curr[:,:,1]
                R = curr[:,:,2]
                B[thres == 255] = 255
                G[thres == 255] = 255
                R[thres == 255] = 0
                final_image =  np.dstack((R,G,B))
```

```
        if(loc == 0):
        cv2.imwrite("bgsub/dynamic/mean/frame%d.jpg" % count, final_image)
        else:
        cv2.imwrite("bgsub/static/mean/frame%d.jpg" % count, final_image)
        count = count + 1
elif(method == "median"):
        median = calculate_background(frames, "median")

        count = 0
        for frame in frames:
        curr = abs(frame - median)
        thres = np.mean(curr,2)
        val = filters.threshold_otsu(curr)
        thres[thres >= val] = 255
        thres[thres <val] = 0
        B = curr[:,:,0]
        G = curr[:,:,1]
        R = curr[:,:,2]
        B[thres == 255] = 255
        G[thres == 255] = 255
        R[thres == 255] = 0
        final_image =  np.dstack((R,G,B))
        if(loc == 0):
        cv2.imwrite("bgsub/dynamic/median/frame%d.jpg" % count, final_image)
        else:
        cv2.imwrite("bgsub/static/median/frame%d.jpg" % count, final_image)
        count = count + 1

elif(method == "mode"):
        mode = calculate_background(frames, "mode")

        count = 0
        for frame in frames:
        curr = abs(frame - mode)
        thres = np.mean(curr,2)
        val = filters.threshold_otsu(curr)
        thres[thres >= val] = 255
        thres[thres <val] = 0
        B = curr[:,:,0]
        G = curr[:,:,1]
        R = curr[:,:,2]
        B[thres == 255] = 255
        G[thres == 255] = 255
        R[thres == 255] = 0
        final_image =  np.dstack((R,G,B))
        if(loc == 0):
        cv2.imwrite("bgsub/dynamic/mode/frame%d.jpg" % count, final_image)
        else:
        cv2.imwrite("bgsub/static/mode/frame%d.jpg" % count, final_image)
        count = count + 1
else:
        raise Exception("Not a valid method")
```

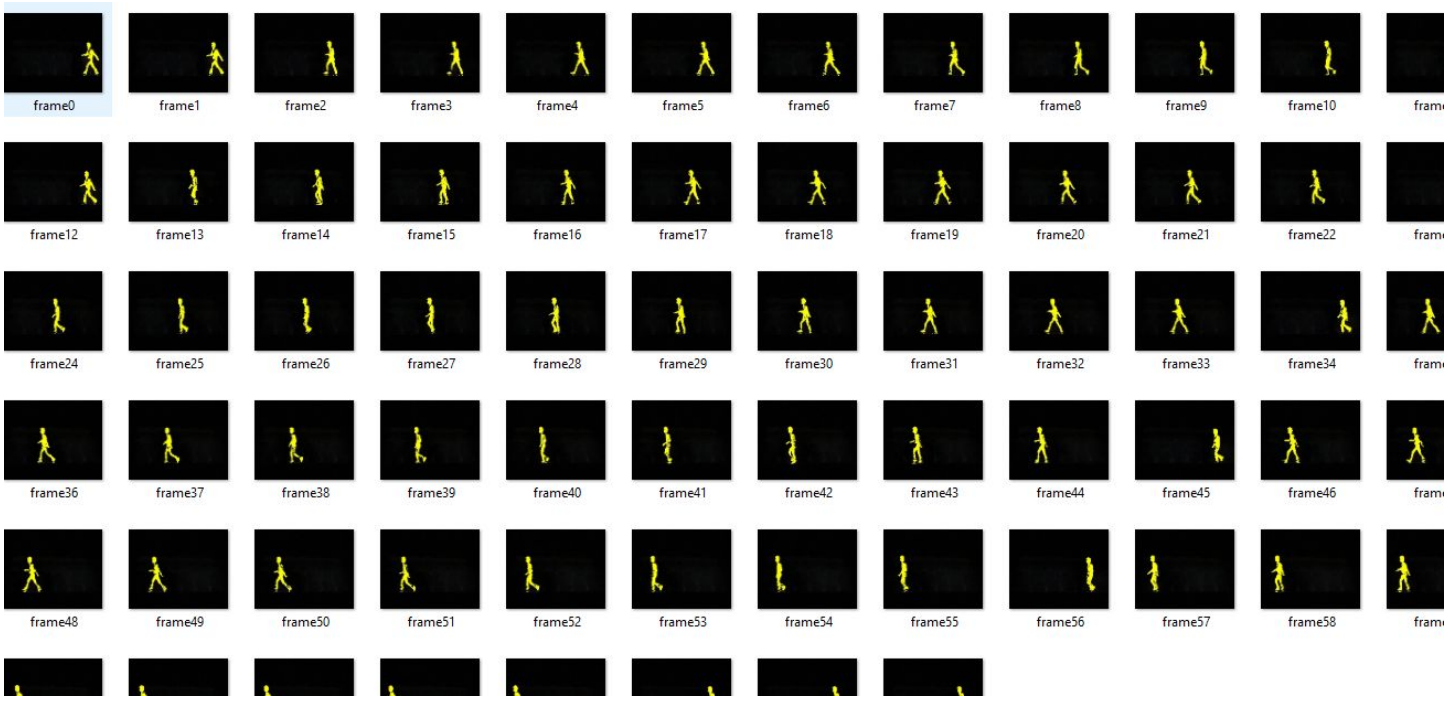
Final Input/Output

1. **Dynamic**

Original frames

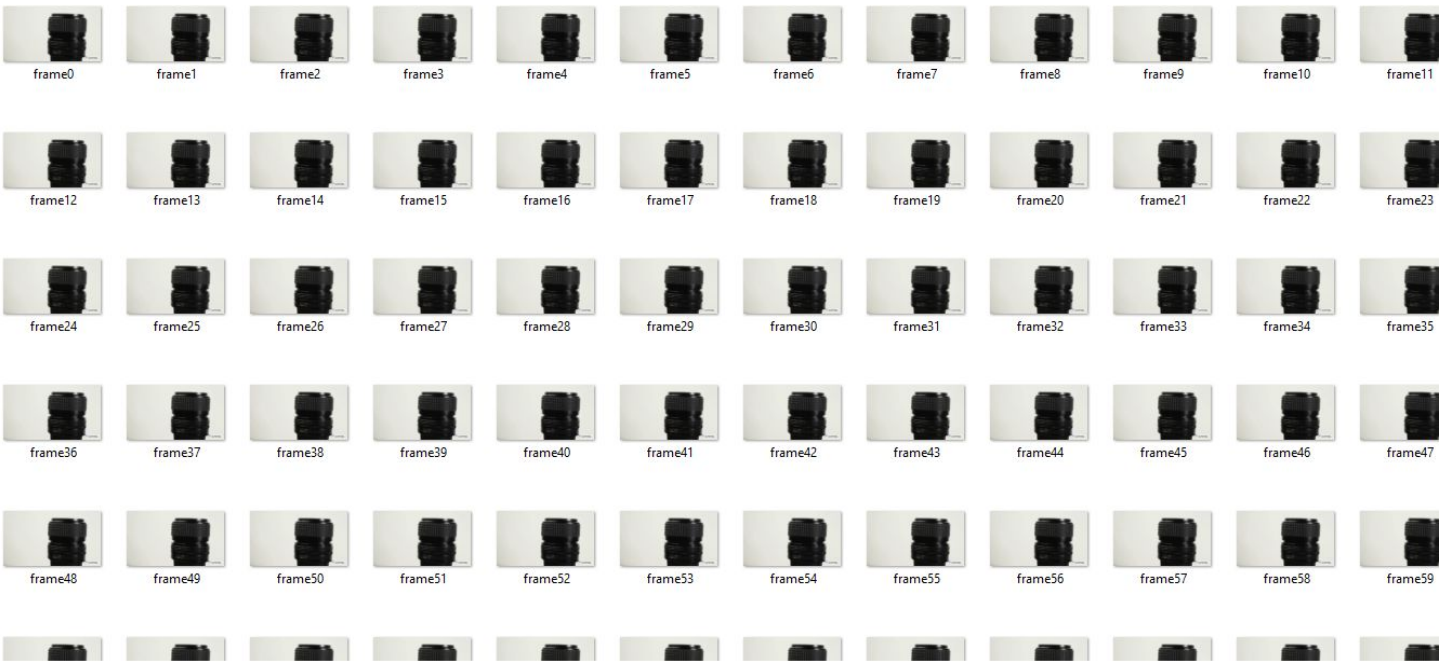


Background subtracted frames



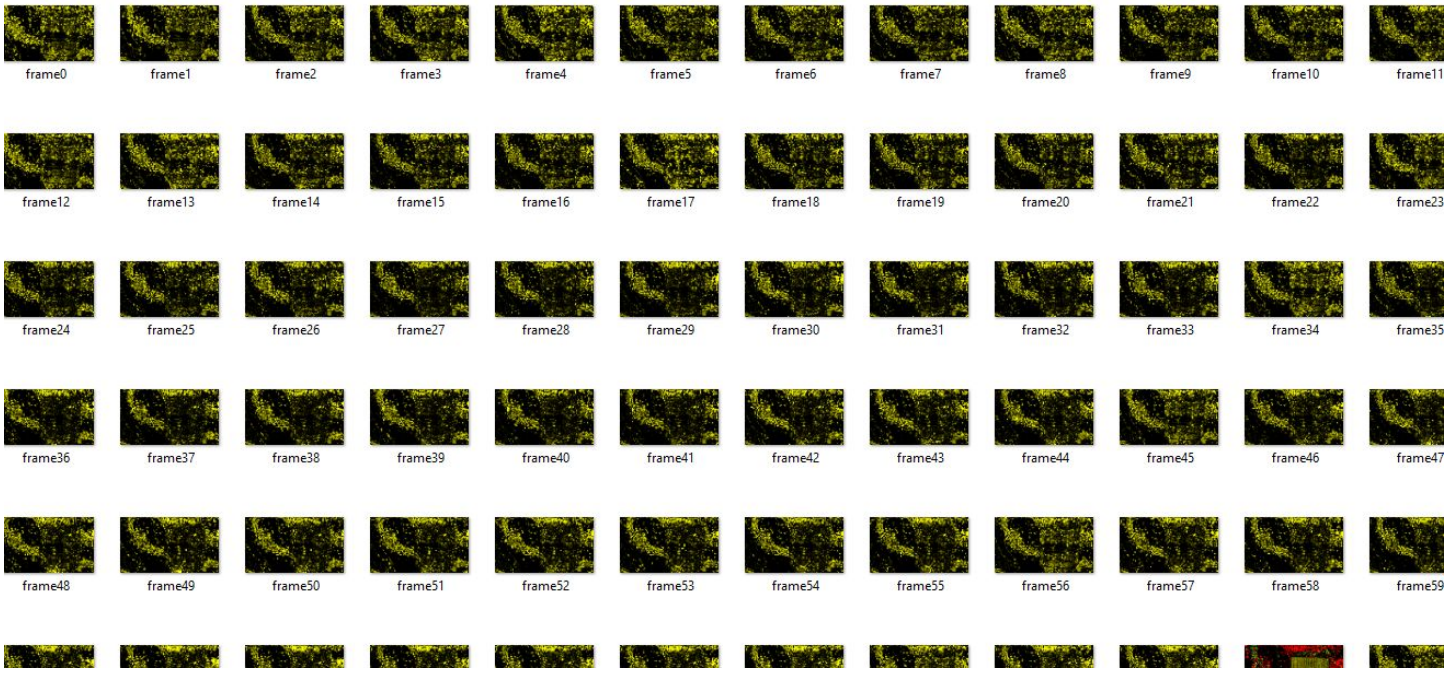
The foreground is moving much as compared to the background hence we get such a good subtraction, similar for both mean, median and mode.

2. Static



Original frames

Background subtracted Frames



We can see slight differences in the background subtraction because the camera lens is moving a bit toh that makes a small difference, also these small differences are highlighted when otsu is applied hence can be seens here.