Homework 03

Deepak Sharma
CSCI 731 Advance Computer Vision

July 9, 2017

Part A and B, Mosaic Creation.

Solution:
1. Empirically set frame drop ratio, for ignoring frames of the input video.
2. Used MOG2 approach, for performing background subtraction with alternative selected frames of the video.
3. By maintaining foreground generated by the previous iteration of the background subtraction method, avoided overlapping between foregrounds generated by multiple iterations of background subtraction method .
4. Performed Geometrical filtering on the connected components of the foreground by removing very small and very large connected components. Also removed connected components with unwanted aspect ratio.
5. Retain the largest connected component as foreground after performing Geometrical filtering.
6. Performed morphological closing operation, with empirically decided element size, for perfectly filling the connected components.
7.At each iteration of background subtraction, superimposed the foreground over first frame of the video. This process created the final mosaics as shown in figure .1 and .2.
Empirically Decided parameters:
Part A(DR_PEDESTRIAN_2017_06_20_MOV.MOV)

```
int const ASPACT_RATIO = 1.2;
int const MAX_AREA = 3000;
int const MIN_AREA = 500;
int const MORPH_SIZE = 20;
int const FRAME_DROP_RATIO = 60;
```

Part B(diving_video_far__board_477E38120CAF.MOV)

```
int const ASPACT_RATIO = 2.5;
int const MAX_AREA = 7000;
int const MIN_AREA = 800;
```

```
int const MORPH_SIZE = 30;
int const FRAME_DROP_RATIO = 60;
```

Figure 1: Part A

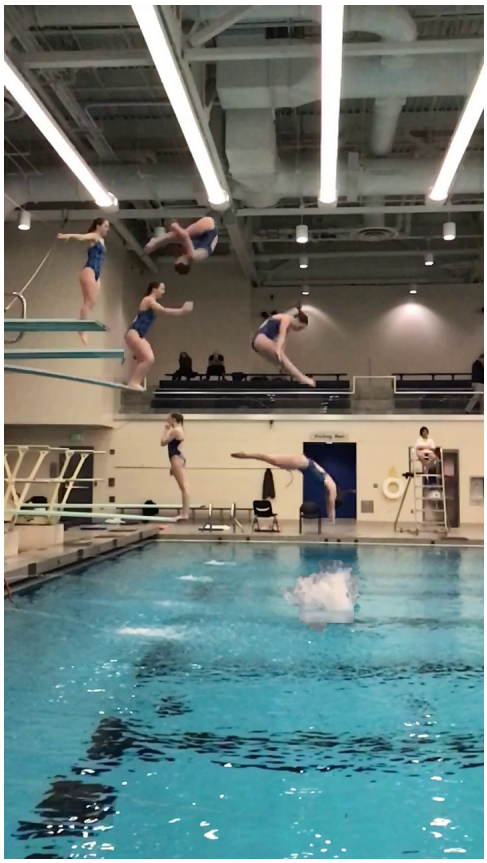

Figure 2: Part B

Implementation

Video Sampling, Background Subtraction and Foreground overlapping removal:

```
/*
* @brief Read input video, process frames of input video by
* performing background subtraction, create collage, show output.
* @param videoFileName, Relative/Absolute path to input video.
* Code help/source:
* http://docs.opencv.org/3.1.0/d1/dc5/tutorial_background_subtraction.html
*/
void processVideo(char* videoFileName) {
    //create the capture object
    char keyboard; //input from keyboard
    Size imageSize;
    Mat frame, fgMaskMOG2, fake3CfgMaskMOG2, output, combined,\
        previousMask, overlappingMask, cleanFgMask;
    Ptr<BackgroundSubtractor> pMOG2; //MOG2 Background subtractor
    //create Background Subtractor objects
    pMOG2 = createBackgroundSubtractorMOG2(); //MOG2 approach
    int iter_count = 1;
    //create video capture object for reading video.
    VideoCapture capture(videoFileName);
    if(!capture.isOpened()){
        //error in opening the video input
        cerr << "Unable to open video file: " << videoFileName << endl;
        exit(EXIT_FAILURE);
    }
    //read input data. ESC or 'q' for quitting
    keyboard = 0;
    while( keyboard != 'q' && keyboard != 27 ){
        //read the current frame
        if(!capture.read(frame)) {
            cerr << "Unable to read next frame." << endl;
            cerr << "Exiting..." << endl;
            break;
        }
        if (iter_count==1){
            //intializing variable at first itereation.
            imageSize = frame.size(); //Mesuring size of image frames.
            //Creating image for holding output progress.
            combined = Mat(imageSize.height, 2*imageSize.width, CV_8UC3);
            //Intialing output image for holding collage.
            frame.copyTo(output);
            // Maintaining Forground objects history for avoiding
            // overlapping.
            previousMask = Mat(frame.size(), CV_8UC1, Scalar(0));}
        iter_count+=1;
```

```
        //update the background model
        pMOG2->apply(frame, fgMaskMOG2);
        if (iter_count%FRAME_DROP_RATIO != 0){
            //Performing samping
            continue;}
        //Cleaning foreground Mask, removing unwanted componenets.
        processForeground(fgMaskMOG2, cleanFgMask);
        //Itersection operation between new foreground and old foreground
        bitwise_and(previousMask, cleanFgMask, overlappingMask);
        //Checking overlapping between old and new foreground
        if (countNonZero(overlappingMask) == 0){
            // If new and old foreground do not overlap then consolidate
            // both foreground for creating collage.
            frame.copyTo(output, cleanFgMask);
            // Coverting 1 channel image into 3 channel image for
            // stitching image.
            vector<Mat> fakeChannels;
            for(int channel_index=0; channel_index<3; channel_index++)
                fakeChannels.push_back(cleanFgMask);
            //Merging image channels
            merge(fakeChannels, fake3CfgMaskMOG2);
            fake3CfgMaskMOG2.copyTo(combined(Rect(0,0, \
                            imageSize.width, imageSize.height)));
            //Updating collage.
            output.copyTo(combined(Rect(imageSize.width,0, imageSize.width,
                    imageSize.height)));
            //Show collage consolidation process.
            imshow("Process", combined);
            keyboard = (char)waitKey(WAIT);
        }
        bitwise_or(previousMask, cleanFgMask, previousMask);
    }
    //Writing final collage
    imwrite("./HW03_Sharma_Deepak_PartA_OUTPUT.jpg", output);
    //delete capture object
    capture.release();
}
```

[1, 3]

Foreground Post-Processing

```
/*
* @brief clean foreground components by performing morphological and
* geometrical filtering.
* @param fgMaskMOG2, foreground mask.
* @param cleanFgMask, foreground mask containing only object of interest.
* Code Help/source: http://answers.opencv.org/\
* question/120698/drawning-labeling-components-in-a-image-opencv-c/
```

```
*/
void processForeground(Mat &fgMaskMOG2, Mat &cleanFgMask){
    Mat labelImage(fgMaskMOG2.size(), CV_32S);
    Mat stats, centroids, binaryImage;
    //Performed labeling on foreground image for generating connected
    //components.
    int nLabels = connectedComponentsWithStats(fgMaskMOG2, labelImage,
           stats, centroids, 8, CV_32S);
    std::vector<int> labels_finals;
    int largestLabel = 0;
    int maxArea = 0;
    int area = 0;
    float HWratio = 0;
    for (int label = 1; label < nLabels; ++label){
        area = stats.at<int>(label, CC_STAT_AREA) ;
        float HWratio = 1.0*stats.at<int>(label,\
               CC_STAT_HEIGHT)/stats.at<int>(label, CC_STAT_WIDTH);
        //float WHratio = 1.0/HWratio;
        //Performed geometrical filtering, removed connected components with
        //very small area, very large area, unwanted aspact ratio.
        //maintained the largest Connected components after geo metric
        //filtering.
        if (area > MIN_AREA && area < MAX_AREA && area > maxArea &&\
               HWratio > ASPACT_RATIO ){
            largestLabel = label;
            maxArea = area;}}
    for (int rowCounter=0; rowCounter<labelImage.rows; rowCounter++){
        for(int colCounter=0; colCounter<labelImage.cols; colCounter++){
            int label = labelImage.at<int>(rowCounter, colCounter);
                if(label != largestLabel)
                    labelImage.at<int>(rowCounter, colCounter) = 0;
        }
    }
    labelImage.convertTo(labelImage, CV_8UC1);
    threshold(labelImage, binaryImage, 0, 1, 0);
    Mat element = getStructuringElement( MORPH_RECT,
           Size(2*MORPH_SIZE + 1, 2*MORPH_SIZE+1),
           Point(MORPH_SIZE, MORPH_SIZE));
    //Performed morphological closing operation for filling completly the
    //traget connected component
    morphologyEx(binaryImage, cleanFgMask, MORPH_CLOSE, element);
    cleanFgMask = 255*cleanFgMask;
```

[1, 2, 3]

Source code credit

1. Background Subtraction [1].
2. Image operations (Intersection, Union, image stitching etc.) [3, 5].
3. Codding style and guidelines [4].
4. Extracting and processing connected components. [2].
6. Assertion of a file path [6].

References

[1] How to do background subtraction methods, 2013. URL `http://docs.opencv.org/3.1.0/d1/dc5/tutorial_background_subtraction.html`.

[2] Drawning labeling components in a image opencv c++, 2016. URL `http://answers.opencv.org/question/120698/drawning-labeling-components-in-a-image-opencv-c/`.

[3] Gary Bradski Adrian Kaehler. *Learning OpenCV 3: Computer Vision in C++ with the OpenCV Library 1st*. O'Reilly Media, Inc., 2 edition, 2016. ISBN 1491937998 978149193799.

[4] Errin Fulp. C++ coding guidelines, 2006. URL `http://csweb.cs.wfu.edu/~fulp/CSC112/codeStyle.html`.

[5] *The OpenCV Reference Manual*. Itseez, 3.2 edition.

[6] Pherric Oxide Vincent, Pevik. Fastest way to check if a file exist using standard c++/c++11/c?, 2012.