1. Submit the complete source code and makefile in one zip file into Dropbox by the due date and time. Don't submit any libraries, object files, input files, or images.
2. Your code should be split among three directories as done in HW08 and HW09.
3. Your code has to packaged properly as I've described though out the semester.
4. Complete the code independently. I will scrutinize the code not only in content but also in presentation. I expect you to submit code that follows the pseudo code, is modular, clear, well commented, and properly indented.

By submitting this:w homework you attest that you completed this code alone and that you did not use any source code in your project except from your own work, your previous homework assignments, and from the instructor.

Name: ____________________

Overview: In this homework assignment you'll read in a series of images and masks and

1. for each image
   (a) compute the centroid
   (b) smooth the image by the process of correlation and convolution
   (c) compute the vertical, horizontal, and total edges.
   (d) compare the centroid with the previous image's. If the same, then there's no motion.
   (e) If there's motion, identify the frame that has motion.
   (f) Use the difference between two consecutive images to identify motion.
   (g) If there's motion, identify the frame that has motion.

Algorithm 1: General Algorithm

```
Input: 4 files: images list, horizontal kernel, vertical kernel, Smoothing kernel
Output: images and centroids
begin
    Open the images list file & read the first image name
    Obtain the image size and allocate memory for several 2d arrays
    Read the first image values and assign to an array
    Assign a threshold value for motion detection for the two methods
    Read the vertical and horizontal edge kernel from the files
    Read the smoothing kernel from the given file
    Compute the vertical, horizontal, and total edges
    Trim all the edges arrays
    Save all the edges arrays to PGM files
    Smooth the image using correlation, normalize the array, and save it in an image to a pgm file
    Smooth the image using convolution, normalize the array, and save it in an image to a pgm file
    Recall that while doing the convolution you need to flip the kernel
    Compute the image centroid and write it out to a file
    comment: An example of output file names will be discussed in the video
    for the rest of the images do
        Read the next image into the array
        Do steps 8-14 above
        compare the centroid with the previous image
        If the centroid is outside the tolerance value, flag the frame as containing motion
        Take the absolute value of the difference between two consecutive images
        Compute the maximum value of the difference
        If the max is greater than the tolerance flag the frame for motion
        Swap the images
    end
end
```

Algorithm 2: Centroid

```
Input: Image name or array
Output: Position $(x, y)$ of the centroid
1 comment: The value of pixel in the image at $(row, col) = (x, y)$ is $I_{x,y}$
2 begin
3     Read the image into an array (unless an array is given)
4     Compute the sum of all the pixels in the image:$sum = \sum_{x=0}^{rows-1} \sum_{y=0}^{cols-1} I(x, y)$
5     Compute the $x$ value of the centroid: $cx = \sum_{x=0}^{rows-1} \sum_{y=0}^{cols-1} xI_{x,y}/\text{sum}$
6     Compute the $y$ value of the centroid: $cy = \sum_{x=0}^{rows-1} \sum_{y=0}^{cols-1} yI_{x,y}/\text{sum}$
7     Print the results
8 end
```

Algorithm 3: Smoothing & Edge Detection

```
Input: Original image, image size, modified image, kernel array, kernel size
begin
    Apply the kernel to the image. Don't worry about the edge pixels as discussed in class
end
```

Algorithm 4: Trim the new image

```
Input: Original image, image size, modified image
begin
    Loop through the image values (pixels)
    if $pixel > 255$ then
        $pixel = 255$
    end
    if $pixel < 0$ then
        $pixel = 0$
    end
end
```

Algorithm 5: Total edges array

```
  Input: vertical edges array ($I_{1x,y}$), horizontal edges array ($I_{2x,y}$), array dimensions
  Output: Modified array
1 begin
2     comment: The value of pixel in the image at $(row, col) = (x, y)$ is $I_{x,y}$
3     Loop through the image values (pixels)
4     For every pixel, the new value is $I_{x,y} = \sqrt{I_{1x,y}^2 + I_{2x,y}^2}$
5 end
```