18-798
Report-Assignment#3
Written by: mhtay

P1)
(Code in: p1.m)

The video was read into matlab frame by frame, using a video reader object and a video writer object was used to write the foreground and background frames back to output video files.

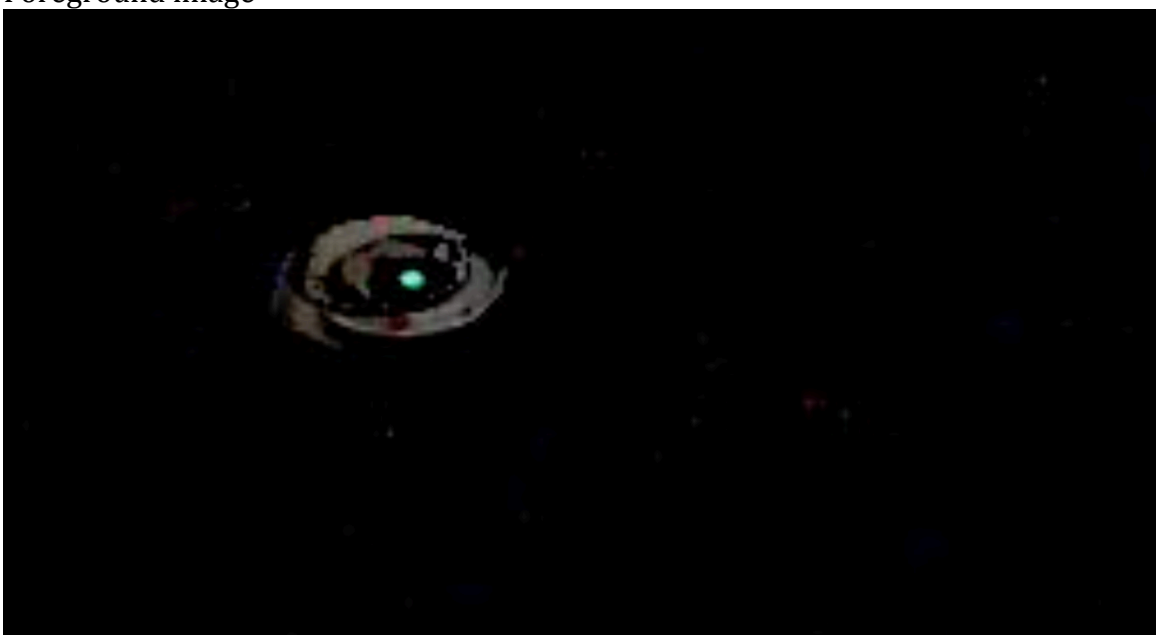
The approximate median algorithm was used to estimate the background. Because the Roomba is stationary at times, the median values of the frames contain a shadow of the Roomba in the background image. Because the initial frame was used as a background image, (ideally one where the Roomba is not present is used), there is also a faint outline of the Roomba at the start location of the image.

To increase the processing speed, each frame is downsampled to a scale of 0.2, before the pixel to pixel comparisons were done. Then the approximate median algorithm was used, modified for the RGB color case, where each test for the median and for the threshold is done for each of the layers.

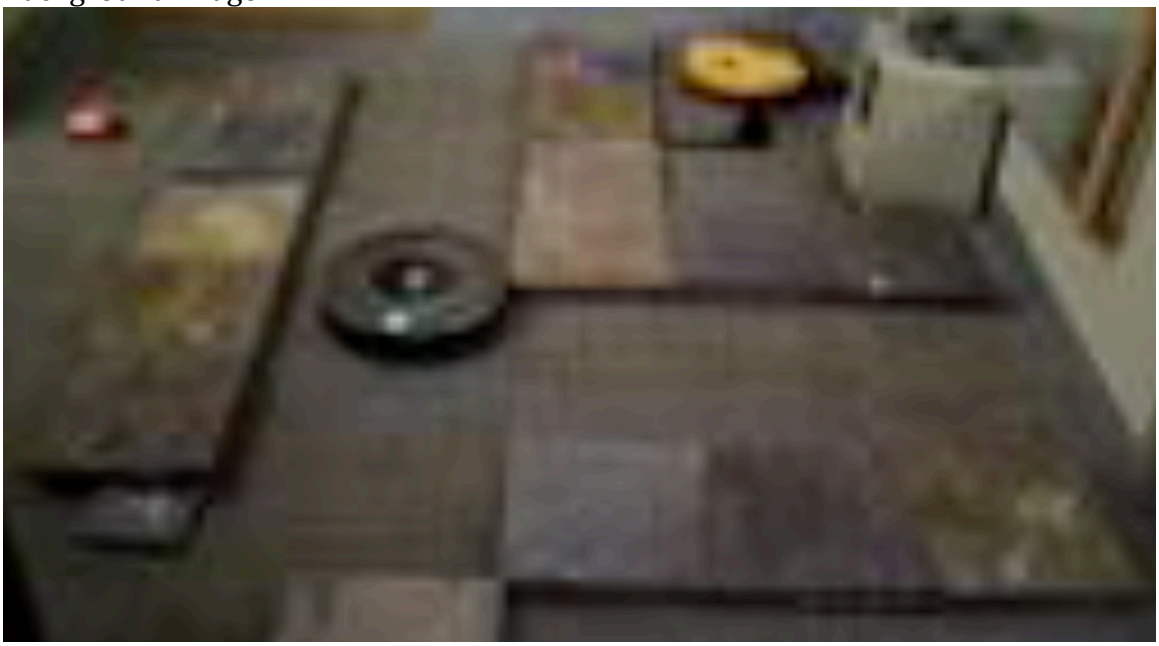
Result:

Threshold selected to: 5

Foreground image



Background image



P2)
Intensity of pixel value of Roomba: 72
Assuming this value is the same across RGB spectrum,
At each iteration, check if difference > threshold = 5, and if difference is greater than display as foreground pixel, otherwise make foreground = 0.

At each iteration per frame, background updates by maximum of 1.
If background initially started at worst case scenario of 0, then it will take 72 – threshold = 67 iterations for Roomba to disappear if it remains stationary.

Number of frames = 1688
Total time of video = 56 seconds
Frames per second = 1688/56
= 30.14
Number of seconds = Number of Frames / 30.14
= 67/30.14
= 2.22 seconds approximately. (worst case scenario)

P3)
(p3.m)
Plot of motion Energy Image was done using the base code as above, but every time a motion is detected, the foreground image is set to 255, instead of to the original frame image value. This produces a binary valued motion energy image. In addition, the incoming frames were converted from RGB to grayscale as the output is independent of color. Each frame was downsampled by a scale of 0.1 to save memory and increase processing speed.

Motion Energy Image:

P4)
(p4.m)
At the start of processing of each frame, the maximum of 0 and the foreground -1 is taken. This models the decay over time of the motion energy, thus creating the motion history image.

Motion History Image:

The brighter the intensity, the more recent in time the image is, as seen in the figure, where there is a clearly rotating Roomba toward the top, while the previous motion of the Roomba can be made out by the fainter gray hues toward the bottom.

P5)

(p5_v2.m)

The green light of the Roomba is very observable, and is approximately the brightest green light in the room, with just a few other pixels of the same magnitude in intensity. Each frame from the video was read in via mat-lab, and then the green colored layer was extracted and a threshold of 90% of the maximum green light detected was applied. The remaining indices that were above this magnitude was then captured and the corresponding indices were labelled on the foreground image as white.

Result:

Appendix

p1.m

```
% Input frames with VideoReader

inputObj = VideoReader('SAM_0562.MP4');
nFrames = inputObj.NumberOfFrames;

% Set up output video with VideoWriter
workingDir = pwd;
outputVideo = VideoWriter(fullfile(workingDir,'output_bg.avi'));
outputVideo2 = VideoWriter(fullfile(workingDir,'output_fg.avi'));
outputVideo.FrameRate = inputObj.FrameRate;
outputVideo2.FrameRate = inputObj.FrameRate;
open(outputVideo);
open(outputVideo2);

first_frame = read(inputObj,1);
first_frame = imresize(first_frame,0.1);
background = first_frame;
foreground = zeros(size(background));
% Display and write frames
for k = 2 : round(nFrames/10)
   my_rgb_frame = read(inputObj, k);
   my_rgb_frame = imresize(my_rgb_frame,0.1);

   diff = abs(my_rgb_frame - background);
   thres = 5;
   for i=1:size(my_rgb_frame,1)
      for j=1:size(my_rgb_frame,2)
         % repeat measurements for 3 layers
         if (diff(i,j,1)>0)
            background(i,j,1) = background(i,j,1) + 1;
         else
            background(i,j,1) = background(i,j,1) - 1;
         end
         if (diff(i,j,2)>0)
            background(i,j,2) = background(i,j,2) + 1;
         else
            background(i,j,2) = background(i,j,2) - 1;
         end
         if (diff(i,j,3)>0)
            background(i,j,3) = background(i,j,3) + 1;
         else
            background(i,j,3) = background(i,j,3) - 1;
         end
```

```
        if (diff(i,j,1) > thres)
           foreground(i,j,1) = my_rgb_frame(i,j,1);
        else
           foreground(i,j,1) = 0;
        end
        if (diff(i,j,2) > thres)
           foreground(i,j,2) = my_rgb_frame(i,j,2);
        else
           foreground(i,j,2) = 0;
        end
        if (diff(i,j,3) > thres)
           foreground(i,j,3) = my_rgb_frame(i,j,3);
        else
           foreground(i,j,3) = 0;
        end

      end
   end
   figure(1), imshow(uint8(foreground));
   figure(2), imshow(uint8(background));
   writeVideo(outputVideo, uint8(foreground));
   writeVideo(outputVideo2, uint8(background));

end
close(outputVideo);
close(outputVideo2);
```

p3.m

```
clear all; close all;
% Input frames with VideoReader

inputObj = VideoReader('SAM_0562.MP4');
nFrames = inputObj.NumberOfFrames;

% Set up output video with VideoWriter
workingDir = pwd;
outputVideo = VideoWriter(fullfile(workingDir,'output_mei.avi'));
outputVideo.FrameRate = inputObj.FrameRate;
open(outputVideo);

first_frame = rgb2gray(imresize(read(inputObj,1),0.1));
background = first_frame;
foreground = zeros(size(background));
% Display and write frames
```

```
for k = 2 :3:nFrames
   fprintf('frame %.f\n',k);
   my_rgb_frame = rgb2gray(imresize(read(inputObj, k),0.1));

   diff = abs(my_rgb_frame - background);
   thres = 10;
   for i=1:size(my_rgb_frame,1)
      for j=1:size(my_rgb_frame,2)
         % MEI is uses grayscale
         if (diff(i,j)>0)
            background(i,j) = background(i,j) + 1;
         else
            background(i,j) = background(i,j) - 1;
         end


         if (diff(i,j) > thres)
            % motion energy (set to 1)
            foreground(i,j) = 255;
%          else
%             foreground(i,j) = 0;
         end
      end
   end
   figure(1), imshow(uint8(foreground));
   writeVideo(outputVideo, uint8(foreground));
end
save('MEI.mat','foreground');
close(outputVideo);
```

p4.m

```
clear all; close all;
% Input frames with VideoReader

inputObj = VideoReader('SAM_0562.MP4');
nFrames = inputObj.NumberOfFrames;

% Set up output video with VideoWriter
workingDir = pwd;
outputVideo = VideoWriter(fullfile(workingDir,'output_mei.avi'));
outputVideo.FrameRate = inputObj.FrameRate;
open(outputVideo);
```

```
first_frame = rgb2gray(imresize(read(inputObj,1),0.3));
background = first_frame;
foreground = zeros(size(background));
% Display and write frames
for k = 2 :1:nFrames/10
   fprintf('frame %.f\n',k);
   my_rgb_frame = rgb2gray(imresize(read(inputObj, k),0.3));

   % motion history image: set decay term for rest of motion history
   foreground = max(foreground-1, zeros(size(foreground)));

   diff = abs(my_rgb_frame - background);
   thres = 15;
   for i=1:size(my_rgb_frame,1)
      for j=1:size(my_rgb_frame,2)
         % MEI is uses grayscale
         if (diff(i,j)>0)
            background(i,j) = background(i,j) + 1;
         else
            background(i,j) = background(i,j) - 1;
         end


         if (diff(i,j) > thres)
            % motion energy (set to 1)
            foreground(i,j) = 255;
          end
      end
   end

   figure(1), imshow(uint8(foreground));
   writeVideo(outputVideo, uint8(foreground));
end
save('MEI.mat','foreground');
close(outputVideo);
```

p5_v2.m

```
% p5.m does color intensity tracking to locate location of roomba

clear all; close all;
% Input frames with VideoReader

inputObj = VideoReader('SAM_0562.MP4');
nFrames = inputObj.NumberOfFrames;

% Set up output video with VideoWriter
workingDir = pwd;
outputVideo =
VideoWriter(fullfile(workingDir,'finding_greens.avi'));
outputVideo.FrameRate = inputObj.FrameRate;
open(outputVideo);

first_frame = imresize(read(inputObj,1),0.3);
nrows = size(first_frame,1);
ncols = size(first_frame,2);
background = first_frame;
foreground = zeros(size(background,1),size(background,2));


% Display and write frames
for k = 1 :1:nFrames
    fprintf('frame %.f\n',k);
    my_rgb_frame = imresize(read(inputObj, k),0.3);
    green_pixels = my_rgb_frame(:,:,2);

    % threshold 90% of maximum pixel brightness
    dim_pixels_idx = green_pixels <= 0.9*max(max(green_pixels));
    green_pixels(dim_pixels_idx) = 0;

    % color foreground
    foreground(green_pixels>0) = 255;
    figure(1), imshow(uint8(foreground));
    writeVideo(outputVideo, uint8(foreground));
end
close(outputVideo);
```