Computer Vision I [EECE]
Project 1
Sharma, Shivam [sharma.s@husky.neu.edu]
Shah, Romil [shah.romil@husky.neu.edu]

Project 1

Computer Vision I [EECE]
Project 1
Sharma, Shivam [sharma.s@husky.neu.edu]
Shah, Romil [shah.romil@husky.neu.edu]

TABLE OF CONTENTS:

Computer Vision I [EECE]
Project 1
Sharma, Shivam [sharma.s@husky.neu.edu]
Shah, Romil [shah.romil@husky.neu.edu]

1. Abstract

A technique for motion detection using simple Image filtering is proposed in this project. The devised algorithm uses the Derivative filters, Smoothing filters and Gaussian filters and other methods such as thresholding. The proposed algorithm is tested by variations in the critical parameters such as threshold values, box filter window size and Standard deviation values for Gaussian filter. Corresponding results are analysed and compared upon.

2. Introduction

The proposed methodology is implemented by applying the images with a temporal derivative filter (TDF) - (0.5*[-1, 0, 1]) for all the 3 given set of images. In the next step it is complemented first with 3x3 Box filter, then 5x5 Box filter and finally 2-d Gaussian filter (GF) with our chosen starting Standard Deviation value as 1, therefore the size of the 2-D Gaussian mask is 5x5. We gather the result with all the images and compare between them.

Then next we implement 1-D Gaussian filter with using two standard deviation (tSigma) values as 1 & 2. In the next step it is complemented first with 3x3 Box filter, then 5x5 Box filter and finally 2-d Gaussian filter with our chosen starting Standard Deviation value as 1, therefore the size of the 2-D Gaussian mask is 5x5. We gather the result with all the images and compare between them.

The major component is thresholding which is the measure to convert the images to Binary form. It is computed using Standard Deviation through noise estimation temporally by treating the background values as Gaussian Zero Mean Noise. The correspondingly scaled threshold values are then used to convert the gray scale images into Binary Images which we analyse to detect the motion.

3. Description of algorithm

The proposed algorithm is first implemented by applying the images with a temporal derivative filter (TDF) (0.5*[-1, 0, 1]) or a 1-D Gaussian filter (GF) with Standard Deviation as 1. First we exclusively test for only TDF or 1-D GF, followed by applying other processes explained in the next paragraph. We test the images again by complementing the previously applied filters with Box filters of size 3x3 , 5x5 and 2-D Gaussian filter.

After the application of smoothening techniques, we compute the threshold values based on the noise estimation model by "Trucco-Verri". We compute the Standard deviation values temporally and take the mean of the values to get one threshold value. We then have to scale the threshold value according to 3 different set of given *imag*es (Scaling Factor – 1/5 for "Office" Image set, ½ for "RedChair" & ½ for "EnterExitCrossingPaths2cor"). Now after getting the scaled threshold values we convert the images to Binary images which displays movement.

But we get a lot of noise through the edge detections of stationary objects which we eliminate by subtracting the binary mask of current image with previous image's binary mask. This method eliminates the stationary noise values.

Then for further noise removal and displaying proper detection we apply median filter to the previously obtained difference. This gives us enhanced display of moving objects. For better visibility we convert the white value to green color by applying changes in RGB image planes. Then finally the obtained images are overlapped with the original given RGB images though which we can track the motion.

4. Experiments

The devised algorithm gives very good results with just depicting the movement as the Green pixels over the color image. This was tested with various combinations of the parameters described above – Threshold values, Box filter size and standard deviation of the Gaussian Filters.

For thresholding values we tried various combinations manually which gave best result, and they were close to the automated estimated value. So as discussed above we consider subtraction of current black and white image with the previous image. For converting both the images to Black and White we used different threshold scaling for all the three set of images. We noted that the threshold value scaling vary with the different images, it came out to be different for us for the 3 given images folders.

Analysis on changing the box size: Gaussian Filter of size 3x3 and 5x5 implemented.

Analysis on changing the Standard deviation: Gaussian 2D =1 and Gaussian 1D = 1&2

Sharma, Shivam [sharma.s@husky.neu.edu]
Shah, Romil [shah.romil@husky.neu.edu]

5. Values of Parameters

Tsigma: 1 and higher in later runs

Scaling Factor – 1/5 for "Office" Image set, ½ for "RedChair" & ½ for "EnterExitCrossingPaths2cor"

Box filter: 3x3, 5x5

Temporal derivative filter: 0.5*[-1,0,1]

We use the Standard Deviations value for a 1-D Gaussian Mask as 1 and 2 and for 2-D Gaussian filter as 1.

For displaying purpose in the "Office" images we use images numbered 250 & 600, for "Red Chair" images we use images numbered 55 & 345 and in the "EnterExitCrossingPaths2cor" images we use images numbered 111 & 200 respectively.

- Thresholding Comparison:

1. RedChair

Differentiator	Filtering	Frame Number	Manual Threshold	Automatic Threshold	Performance (error %)
Simple Derivative	Box 3x3	55	0.0090	0.0667	86.5
			0.0660		1.0
			0.0700		4.9
	Box 5x5	55	0.0500	0.0510	2.0
			0.0600		17.6
			0.0550		7.8
	Gauss 2D	55	0.0650	0.0627	3.7
			0.0600		4.3
			0.0700		11.6
Gauss 1D	Box 3x3	55	0.0300	0.0275	9.1
			0.0200		27.3
			0.0250		9.1
	Box 5x5	55	0.0250	0.0310	19.4
			0.0200		35.5
			0.0150		51.6
	Gauss 2D	55	0.0700	0.0275	154.5
			0.0300		9.1
			0.0200		27.3

2. Office

Differentiator	Filtering	Frame Number	Manual Threshold	Automatic Threshold	Performance (error %)
Simple Derivative	Box 3x3	250	0.0100	0.0353	71.7
			0.0300		15.0
			0.0350		0.8
	Box 5x5	250	0.0500	0.0235	112.8
			0.0300		27.7
			0.0250		6.4
	Gauss 2D	250	0.0650	0.0275	136.4
			0.0100		63.6
			0.0250		9.1
Gauss 1D	Box 3x3	250	0.0010	0.0078	87.2
			0.0100		28.2
			0.0090		15.4
	Box 5x5	250	0.0100	0.0081	23.5
			0.0070		13.6
			0.0080		1.2
	Gauss 2D	250	0.0700	0.0078	797.4
			0.0050		35.9
			0.0080		2.6

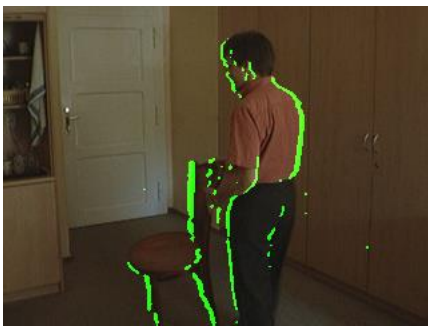
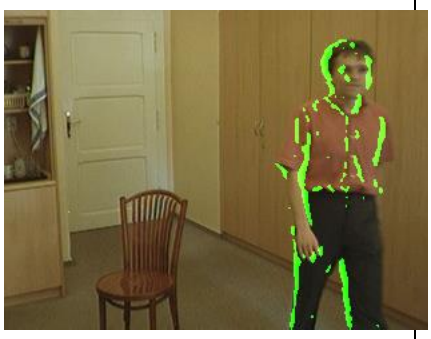
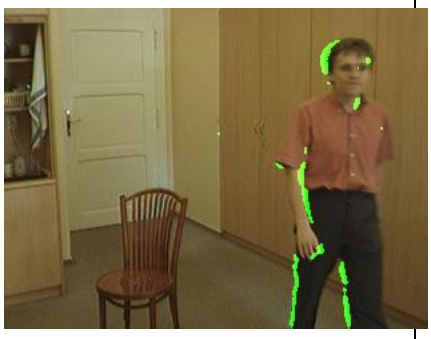
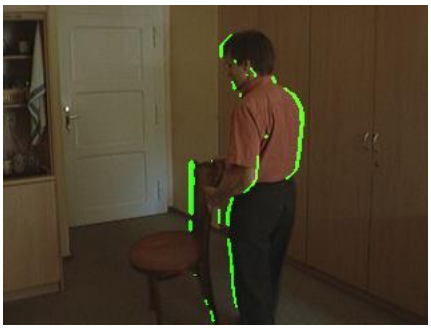
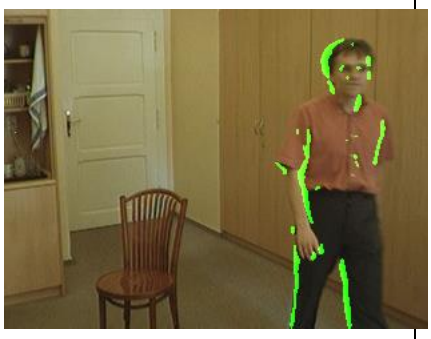
3. EnterExitCrossingPaths2cor

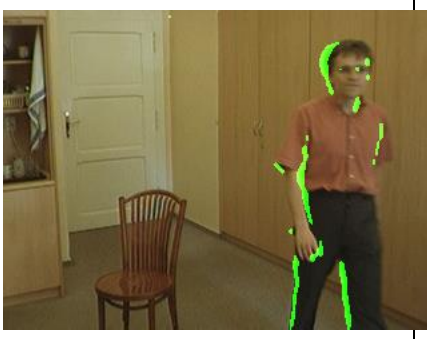
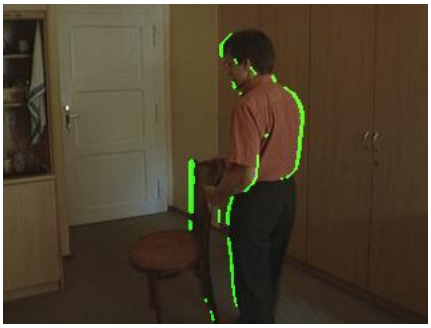
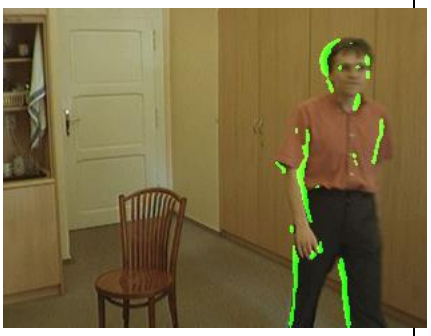
Differentiator	Filtering	Frame Number	Manual Threshold	Automatic Threshold	Performance (error %)
Simple Derivative	Box 3x3	55	0.0100	0.0392	74.5
			0.0200		49.0
			0.0250		36.2
	Box 5x5	55	0.0200	0.0253	20.9
			0.0300		18.6
			0.0250		1.2
	Gauss 2D	55	0.0300	0.0353	15.0
			0.0350		0.8
			0.0400		13.3
Gauss 1D	Box 3x3	55	0.0200	0.0196	2.0
			0.0300		53.1
			0.0150		23.5
	Box 5x5	55	0.0100	0.0215	53.5
			0.0200		7.0
			0.0250		16.3
	Gauss 2D	55	0.0100	0.018	44.4
			0.0200		11.1
			0.0150		16.7

6. Results:

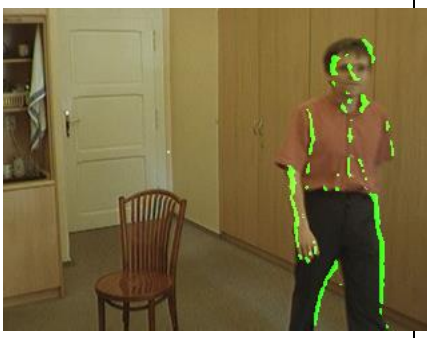
6.1. Office:

6.1.1. With 1D Gaussian Derivative:

	Frame: 250	Frame: 600
Only Derivative (tsigma = 1)		
Only Derivative (tsigma = 2)		
With Box 3x3		

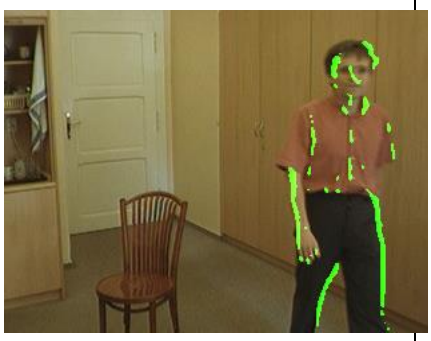
With Box 3x3		
With Gauss 2D		

6.1.2. With Simple Derivative:

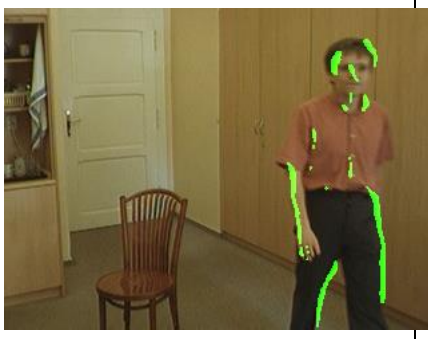
	Frame: 250	Frame: 600
Only Derivative		

Sharma, Shivam [sharma.s@husky.neu.edu]
Shah, Romil [shah.romil@husky.neu.edu]

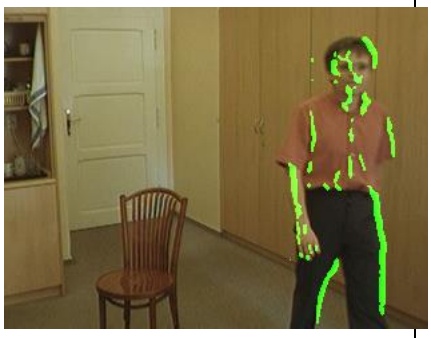
With
Box 3x3

With
Box 5x5

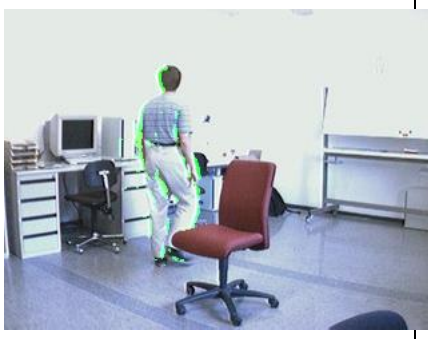
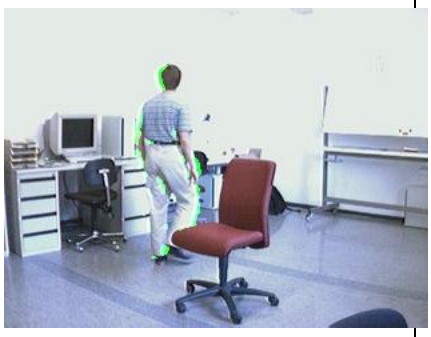
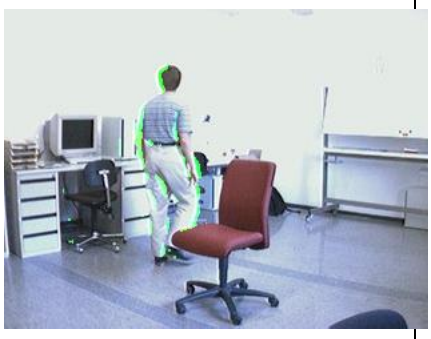



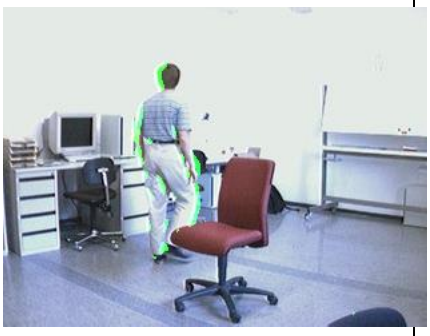
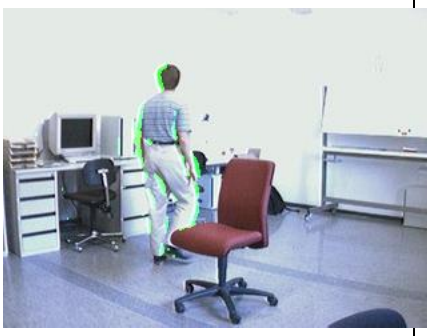
With
Gauss
2D

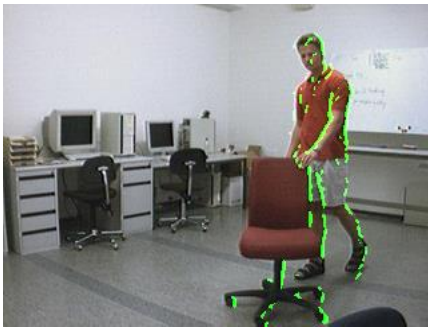
6.2. RedChair

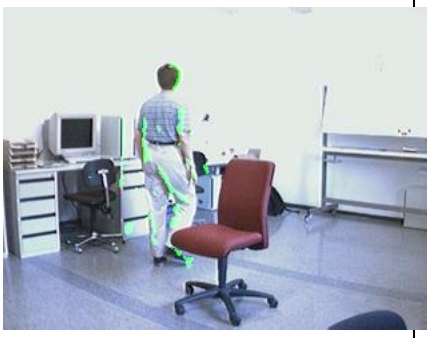
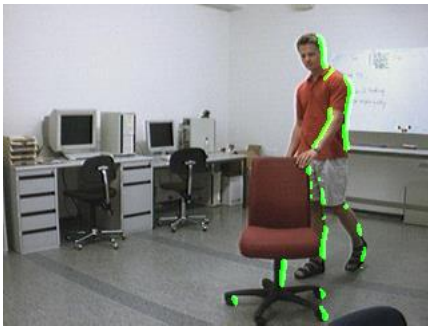
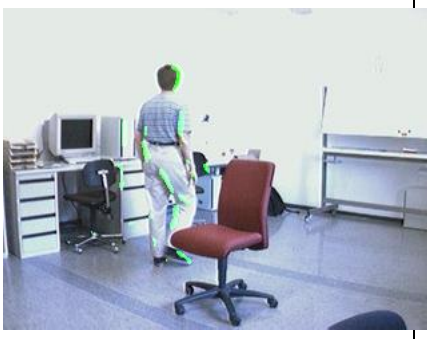
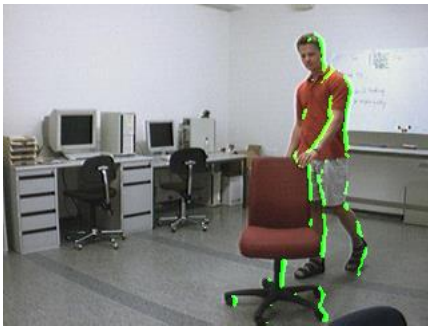
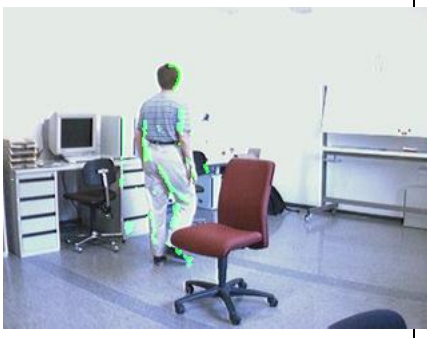
6.2.1. With 1D Gaussian Derivative:

	Frame: 55	Frame: 345
Only Derivative (tsigma = 1)		
Only Derivative (tsigma = 2)		
With Box 3x3		

With Box 5x5		
With Gauss 2D		

6.2.2. With Simple Derivative

	Frame: 55	Frame: 345
Only Derivative		

With Box 3x3		
With Box 5x5		
With Gauss 2D		

6.3. EnterExitCrossingPaths2cor:

6.3.1. With 1D Gauss Derivative:

	Frame: 90	Frame: 200
Only Derivative (tsigma = 1)		
Only Derivative (tsigma = 2)		
With Box 3x3		

With Box 5x5

With Gauss 2D

6.3.2. With Simple Derivative:

Frame: 90 Frame: 200

Only Derivative

With Box 3x3


With Box 5x5


With Gauss 2D


7. Conclusion

The images were analysed and tested for Motion Detection and seen in continuation form as a video for analysis of motion. The devised algorithm showed promising results. The areas under motion were effectively tracked and displayed. Even the motions in the shadows and the reflection were accounted for. The algorithm treated the motion critically and small movements were explicitly shown. The thresholding values vary for each set of 3 image folders. On comparison of various parameters it was found that the 3x3 Box filtered images showed better results than 5x5 Box filtered images.

8. Future Scope

- Integration of rectangular detector with the motion
- Improvement in the adaptive median filtering technique

9. Appendix (MATLAB Code)

9.1. Main Code:

```
clear all

clc


imgList = dir('C:\Users\rams1\Desktop\Spring2016\CV\Projects\Office\Office\*.jpg');

imgNos = length(imgList);


for i=1:imgNos

   cd('C:\Users\rams1\Desktop\Spring2016\CV\Projects\Office\Office\')

   filename = imgList(i).name;

   %OrigI{i} = rgb2gray(imread(filename));

   ipre{i}=imread(filename);

   OrigI{i} = rgb2gray(ipre{i});

end

cd('C:\Users\rams1\Desktop\Spring2016\CV\Projects\');


tempoDer = 0.5*[-1,0,1];

tsigma = 1;%input('Enter tsigma: ');

ssigma = 1;%input('Enter ssigma: ');

g1D = fspecial('gaussian',[5*tsigma,1],tsigma);

gau1D = gradient(g1D)';

box3 = 1/9*ones(3,3);

box5 = 1/25*ones(5,5);

gau2D = fspecial('gaussian',[5*ssigma,5*ssigma],ssigma);


%%

% Temporal Derivative and then 1D gaussian

for i=1:imgNos
```

```
    filtI{i} = imfilter(imfilter(OrigI{i},gau2D),tempoDer);
end

[~,sig] = EST_NOISE(filtI,imgNos);
sig1 = mean2(sig/5);

for i=1:imgNos
    i
    filtIa{i} = im2bw(filtI{i},sig1);
    if i<imgNos
        filtIb{i} = im2bw(filtI{i+1},sig1);
        filtIfin{i}=filtIb{i}-filtIa{i};
    else
        filtIfin{i}=filtI{i-1};
    end

end

figure(1);
for i=1:1:imgNos-1
    i
    filtIfin{i} = medfilt2(uint8(filtIfin{i})*255);
    img=filtIfin{i};
    filtIfin{i}=zeros(size(ipre{i}));
    filtIfin{i}(:,:,2) = img;
    filtIall{i}=(ipre{i+1}+uint8(filtIfin{i}));
    %filtI{i}=adapth(filtI{i},3,0.04,0);
    %pause(0.02);
    imshow(filtIall{i});
```

```
end
```

9.2. EST_NOISE function file:

```
function [E,sig] = EST_NOISE(J,imgNos)


for i=1:imgNos
   J{i} = double(J{i});
end
[n1,n2] = size(J{1});
E = zeros(n1,n2);
sig = zeros(n1,n2);

% EST_NOISE
for i=1:imgNos
   E = E + J{i};
end
E = (E/imgNos);
for i=1:imgNos
   sig = sig + (E - J{i}).^2;
end

sig = sqrt(sig/imgNos);
sig = sig/max(max(sig));
end
```

Sharma, Shivam [sharma.s@husky.neu.edu]
Shah, Romil [shah.romil@husky.neu.edu]