Report

Automatic Object Tracker Using MATLAB

Submitted By:

Vidhan Modi [SJSU\_ID: 011451362]

Jigar Agrawal [SJSU\_ID: 011469796]

Guided by:

Dr. John Kim

*In partial fulfillment of the requirements of*

*The course work for Semester II*

*Of*

MASTERS OF SCIENCE

In

ELECTRICAL ENGINEERING



Table of Contents

[Table of Figures 1](#_Toc483273216)

[Abstract 2](#_Toc483273217)

[Introduction 3](#_Toc483273218)

[Thresholding 3](#_Toc483273219)

[Adaptive Thresholding 4](#_Toc483273220)

[Global Thresholding 4](#_Toc483273221)

[Local Thresholding 5](#_Toc483273222)

[Object Detection 7](#_Toc483273223)

[Source Code 8](#_Toc483273224)

[adaptivethreshold.m 8](#_Toc483273225)

[object\_detect.m 9](#_Toc483273226)

[EE253\_Project.m 11](#_Toc483273227)

[Conclusion 13](#_Toc483273228)

[Future Expansions 13](#_Toc483273229)

[Applications 13](#_Toc483273230)

[References: 14](#_Toc483273231)

# Table of Figures

[Figure 1: Image containing Lighting Conditions. 3](#_Toc483272605)

[Figure 2: Global Thresholding 4](#_Toc483272606)

[Figure 3: Using the mean of a 7×7 neighborhood 5](#_Toc483272607)

[Figure 4:  The result for a 7×7 neighborhood and C=7 6](#_Toc483272608)

[Figure 5:  The result for a 7×7 neighborhood and C=10 6](#_Toc483272609)

[Figure 6: Flowchart for Object Detection 7](#_Toc483272610)

# Abstract

The proposed design tracks an object in videos, which plays an important role in computer vision. In this project, we performed an automatic object tracking using MATLAB. Adaptive thresholding was also used to eliminate the uneven lighting conditions. Algorithm works by detecting the edges using the High Pass Filter and apply adaptive filters. We demonstrate the effectiveness of this approach by experimenting with a simple video.

# Introduction

## Thresholding

Poor light gradients in the image makes the image detection very difficult. There are many techniques developed to threshold the image. Non-adaptive thresholding do not take the image into consideration for thresholding.

For example, the following image is taken into consideration. As it apparent from the image that there is non-uniform lighting condition across the page, which makes the thresholding difficult

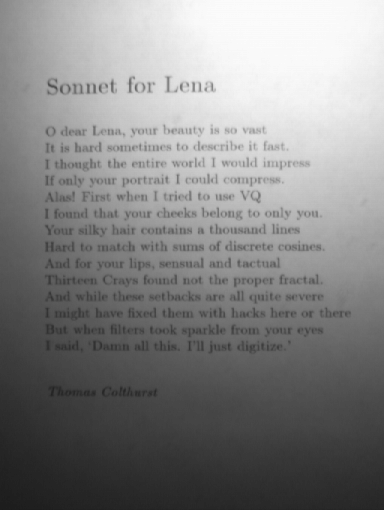


Figure 1: Image containing Lighting Conditions.

So, we use Adaptive Thresholding to mitigate this problem.

## Adaptive Thresholding

Adaptive Thresholding takes the non-uniform lighting conditions into consideration while thresholding the image. There are basically two types of adaptive thresholding:

1. Global Thresholding.
2. Local Thresholding.

### Global Thresholding

Global Thresholding takes only one threshold value for the entire image. So the thresholding for the Figure 1 using Global Thresholding gives results somewhat like following:

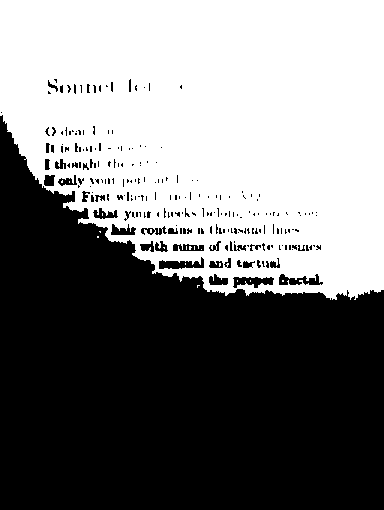


Figure 2: Global Thresholding

So Figure 2 shows that Global Thresholding do not help for strong illumination gradient.

### Local Thresholding

Local Thresholding calculates the unique threshold for each pixel. It takes neighboring pixel intensity to calculate the thresholding value for each pixel.

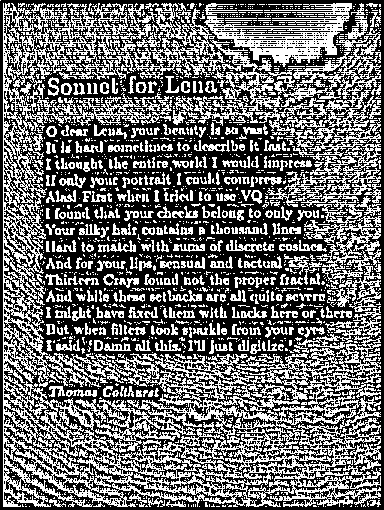


Figure 3: Using the mean of a 7×7 neighborhood

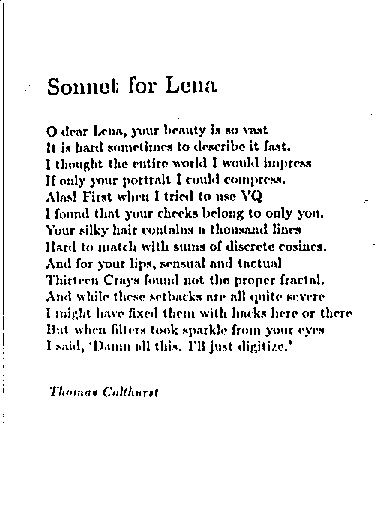


Figure 4:  The result for a 7×7 neighborhood and C=7

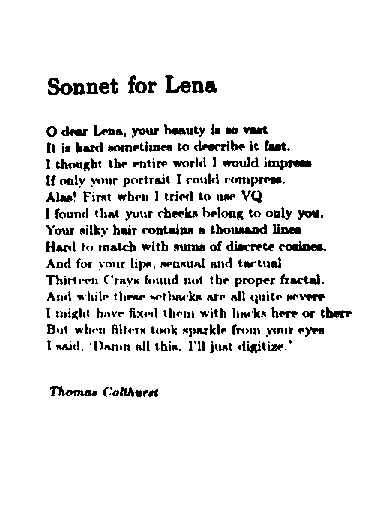


Figure 5:  The result for a 7×7 neighborhood and C=10

## Object Detection

We use neighboring pixel cluster method to identify any object. The location of the object in the image can be calculated based on the pixel values. Basic Flowchart for detecting any object is shown as follows:

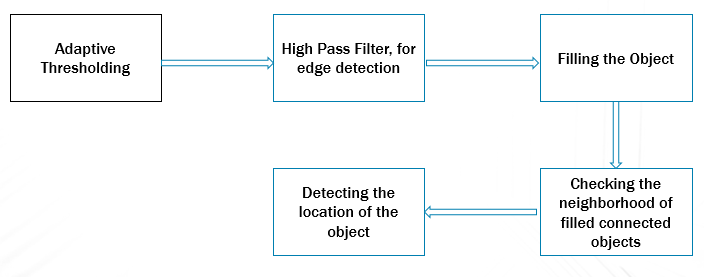


Figure 6: Flowchart for Object Detection

Any Grayscale or colored image is first converted into black and white image. To convert into black and white image we need to implement the adaptive thresholding. Here, adaptive thresholding is used for this purpose, then High Pass Filter is applied on the image to detect the object boundaries.

Once the object boundaries are detected we fill the object with all the ‘1’ so that we can detect the cluster of pixels having the same intensity. It can be said that it is obviously an object.

## Source Code

### adaptivethreshold.m

function bw = adaptivethreshold(image)

[rows, cols, dim] = size(image);

if(dim > 2)

im1 = rgb2gray(image);

else

im1 = im;

end

im2 = imfilter(im1, fspecial('average',7), 'replicate');

%Edge Detection

sim = im2- im1- 10;

bw = im2bw(sim, 0);

end

### object\_detect.m

%Fnction to detect the single object in the image

%Input Arguments : Image Matrix

%Output : Object Array, Start\_location, End Location

function [output, start, end\_a] = object\_detect(data\_input)

%Adaptive Thresholding Function

snap\_z1 = adaptivethreshold(data\_input);

%Filling the detected image

snap\_z = imfill(snap\_z1, 'holes');

[rows, cols] = size(snap\_z);

%Checking Connected pixels

for i=2:rows-1

for j=2:cols-1

if((snap\_z(i+1,j) || snap\_z(i-1,j) || snap\_z(i,j-1) || snap\_z(i,j+1)) == 1);

data(i,j) = 1;

end

end

end

[rows, cols] = size(data);

%Array Adjustment

j1 =0;

a = data;

for j=1:cols

if(data(:,j) == 0)

a(:,j-j1) = [];

j1= j1+1;

end

end

i1 =0;

d = a ;

for i=1:rows

if(d(i,:) == 0)

a(i-i1,:) = [];

i1= i1+1;

end

end

%padding zeros srounding the detected object array

d = padarray(a,[5 5]);

%Applying high pass Filter to restore the original image

filter = [1, 1, 1; 1 -8 1; 1 1 1];

output = imfilter(d, filter);

%Marker Location Caluclation

[rows\_d, cols\_d] = size(data);

[rows\_a, cols\_a] = size(a);

start = [rows\_d- rows\_a, cols\_d - cols\_a];

end\_a = [rows\_a, cols\_a];

end

### EE253\_Project.m

clear;

clc;

close all;

clear all;

% Creating The Video Object

v = VideoReader('Desktop111.wmv');

for i=1:25

%Starting the timer

tic;

%Reading the image fram to process the object detection

snap = read(v, i);

subplot(2,1,1);

imshow(snap);

title('Original Image');

%object Detection algorithm

[out, start\_add, end\_addr] = object\_detect(snap);

%Marker location calculation

marker = [(start\_add(1,2)+end\_addr(1,2)/2), start\_add(1,1) + end\_addr(1,1)/2];

a = insertMarker(snap,round(marker));

subplot(2,1,2);

imshow(a);

title('Processed Image');

%creating the bounding box around the object

rectangle('Position', [start\_add(1,2),start\_add(1,1), end\_addr(1,2),end\_addr(1,1)],'EdgeColor','r','LineWidth',2 );

fprintf('Time for one frame processing is %f Seconds\n', toc);

close all;

end

# Conclusion

Object detection was achieved using adaptive thresholding during non-uniform illuminative conditions using MATLAB.

# Future Expansions

* Multiple object detection.
* Hardware Acceleration using GPUs.

# Applications

* Face Detection
* Automatic Target Detection in Guided Missiles
* Security Systems

# References:

1. https://www.mathworks.com/discovery/object-detection.html

2. http://homepages.inf.ed.ac.uk/rbf/HIPR2/adpthrsh.htm