

**ENCS6161: PROBABILITY AND STOCHASTIC PROCESSES**

**WINTER, 2023**

**PROJECT REPORT**

**PROJECT TITLE:**

**Detection of Cars in Video Sequences Using Gaussian Mixture Model and Expectation Maximization**

**SUBMITTED BY: MAHBOOB LIAQUAT(40261876)**

**& ABU MUSA MUHAMMAD JAHID HASAN(40259193)**

**COURSE INSTRUCTOR: M. OMAIR AHMAD**

**Abstract:** *This project explores the use of Gaussian distribution to detect cars in video sequences. The objective of the project is to develop a better understanding of the utility of Gaussian mixture model and expectation maximization algorithm in this application. The video is segmented into N frames, and a histogram is created for each pixel vs their pixel intensities ranging from 1-256 over all those frames. Gaussian mixture model is implemented with K number of Gaussian, which can range from 3-5, and the expectation-maximization algorithm is used to achieve the best fit for those gaussians over the data in the histogram in the GMM in grayscale. The same process is repeated for R G B as well. In this project, the built-in MATLAB function "fitgmdist" is replaced with a custom function "myfitgmdist" that implements the GMM and EM for this purpose. The results demonstrate that this approach can accurately detect cars in video sequences. This research has important implications for video analysis and surveillance systems.*

**TABLE OF CONTENTS**

Abstract…………………………………………………………………………………. (2)

Table of Contents……………………………………………………………………….. (3)

List of abbreviations in alphabetical order ……………………………………………... (4)

List of Figures ………………………………………………………………………….. (5)

Introduction …………………………………………………………………………….. (6)

Scope and objectives of the project …………………………………………………….. (7)

Detailed methodology and implementation ……………………………………………. (9)

Experimental results ……………………………………………………………………. (13)

Conclusion ……………………………………………………………………………… (14)

References ………………………………………………………………………………. (15)

Appendix………………………………………………………………………………. (16)

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **ABBREVIATION** | **DESCRIPTION** |
| EM | Expectation Maximization |
| GMM | Gaussian Mixture Model |
| RGB | Red Green Blue |

**LIST OF FIGURES**

|  |  |
| --- | --- |
| **FIGURE NO.** | **DESCRIPTION** |
| 1 | Flowchart explaining the main methodology starting from processing the video file, implementing GMM & EM until the detection of cars |
| 2 | Flowchart explaining the Expectation Maximization for the GMM |
| 3 | Foreground detection |
| 4 | (a) The car is detected in the video frame (b) another car is detected (c) When no cars are in the video frame, the top left index shows |

**INTRODUCTION**

Video analysis and surveillance systems have become increasingly important in recent years, especially in the fields of security and transportation. The ability to detect and track objects in video sequences is critical for these systems to function properly. In this project, we explore the use of Gaussian mixture model (GMM) and expectation maximization (EM) algorithm to detect cars in video sequences. The GMM is a statistical model that represents the probability distribution of a given set of data as a mixture of multiple Gaussian distributions. The GMM is used to model complex distributions that cannot be represented by a single Gaussian distribution. It has been used in a wide range of applications, including computer vision, speech recognition, and bioinformatics. The EM algorithm is an iterative optimization algorithm used to estimate the parameters of a statistical model. It is commonly used in conjunction with the GMM to estimate the parameters of the mixture model. The EM algorithm consists of two steps: the expectation step (E-step) and the maximization step (M-step). In the E-step, the algorithm estimates the posterior probability of each data point belonging to each Gaussian distribution in the mixture. In the M-step, the algorithm updates the parameters of the Gaussian distributions based on the estimated posterior probabilities.

The objective of this project is to develop a better understanding of the utility of GMM and EM algorithm in detecting cars in video sequences. To achieve this, we segmented the video into N frames and created a histogram for each pixel vs their pixel intensities ranging from 1-256 over all those N frames. We then implemented the GMM with K number of Gaussian, which can range from 3-5, and used the EM algorithm to achieve the best fit for those Gaussians over the data in the histogram in the GMM in grayscale. The same process was repeated for R G B as well. In addition to implementing the GMM and EM algorithm, we also replaced the built-in MATLAB function "fitgmdist" with our own function "myfitgmdist". This allowed us to customize the GMM and EM algorithm to better suit our needs and to gain a deeper understanding of the underlying concepts. The rest of the report is organized as follows. In the next section, we describe the scope and objectives of the project. We then provide a detailed methodology and implementation in section 3, followed by the experimental results in section 4. We conclude the report in section 5 and provide a list of references and appendices at the end of the report.

**SCOPE AND OBJECTIVES OF THIS PROJECT**

The objective of this project is to develop an algorithm for detecting cars in video sequences using Gaussian Mixture Model (GMM) and Expectation Maximization (EM). The scope of the project includes learning GMM and EM, implementing foreground detection using GMM with EM instead of the built-in function "vision.ForegroundDetector", and replacing the built-in function in Matlab "fitgmdist" with our own function named "myfitgmdist". Additionally, the project includes segmentation of the video into N number of frames, determining the pixels and creating histograms for each pixel vs their pixel intensities ranging from 1-256 over all those N frames, and implementing GMM over all the data, with K number of Gaussian ranging from 3-5. The specific objectives of the project are as follows:

1. To learn the theory and implementation of GMM and EM for foreground detection in video sequences.
2. To implement foreground detection using GMM with EM instead of the built-in function "vision.ForegroundDetector".
3. To replace the built-in function in Matlab "fitgmdist" with our own function named "myfitgmdist".
4. To develop a GMM and EM-based algorithm for car detection in video sequences.
5. To test the algorithm on various video sequences and compare the results with other car detection algorithms.
6. To determine the optimal number of Gaussian models for the GMM algorithm by comparing the results of different models on the test video sequences.
7. To investigate the effect of using GMM and EM on car detection accuracy compared to other machine learning techniques.

The project is significant because car detection in video sequences has important applications in surveillance, traffic monitoring, and autonomous vehicles. Additionally, the use of GMM and EM in car detection provides an opportunity to investigate the effectiveness of this technique compared to other machine learning techniques. The contributions of this project include the development of a new GMM and EM-based algorithm for car detection, which can be further refined and improved in future research. However, it is important to note that there are some limitations to the project. The algorithm was tested on a limited number of video sequences and the results may not be representative of all possible scenarios. Additionally, the algorithm assumes that the car is the most significant object in the video sequence, and may not work as effectively in situations where other objects are present such as in snowy and rainy weather. Finally, it should be noted that in scenarios where the pixel intensities do not vary much over the N frames, it may become difficult to apply more than one Gaussian for that specific data of pixel intensity values.

There is a huge scope for this project in the future where multiple areas can be benefitted, such as:

1. Improving the accuracy of car detection by incorporating more advanced techniques such as deep learning and computer vision algorithms.
2. Applying the same technique for other types of objects or features in videos, such as pedestrians, traffic signs, or buildings.
3. Developing a real-time system for car detection using GMM and EM, which could be used for traffic monitoring, surveillance, or other applications.
4. Experimenting with different values of K in the Gaussian Mixture Model to determine the optimal number of Gaussians required for accurate car detection.
5. Investigating the impact of changing the parameters in the Expectation-Maximization algorithm, such as the number of iterations, on the accuracy and speed of the car detection algorithm.

**DETAILED METHODOLOGY AND IMPLEMENTATION**

Obtain parameters using Custom Made function based on EM-algorithm

Read Original Video File

Apply Foreground detection after fitting GMM on the data

Rewrite video file to make the size ¼ for faster processing

Apply Filtering on the processed frames

Read ‘N’ Frames and Convert them from RGB to GrayScale

Apply Bounding boxes on Detected cars

Build Histogram for Each Pixel

Run Video File for Car Detection

Fit ‘K’ Gaussians to the acquired Data

End

*Figure 1. Flowchart explaining the main methodology starting from processing the video file, implementing GMM & EM until the detection of cars*

The first step of the process is to read the original video file that has to be processed for car detection. The video file is opened and read using appropriate libraries in MATLAB. In the second step, the video file is resized to one-fourth of its original size. This is done to reduce the processing time and improve the efficiency of the algorithm. The resized video is stored in a separate file for further processing.

Next, 'N' frames are read from the resized video file and converted to grayscale. For this project, 150 frames are used, although more could be used, but the processing time would be more in that case. Then, a histogram is constructed for each pixel in the frames using a custom-made function called "myhist". The histograms are stored in an array called "allHistograms". Each pixel has varying intensities over the 'N' frames, and constructing a histogram for each pixel helps in obtaining a better understanding of the data. The next step involves fitting 'K' Gaussians to the acquired data from the histograms of the pixel intensities to obtain the Gaussian Mixture Model (GMM). In this project, 'K' is set to 3. In the following step, the mean, standard deviation, and the mixing coefficients are obtained using a custom-made function that is based on the Expectation-Maximization (EM) algorithm. The function calculates the log-likelihood and returns the best mean, standard deviation, and mixing coefficients for the fitted Gaussian components. After obtaining the GMM, foreground detection is applied to the frames. This is done by first detecting the background, for which logical subtraction is performed, and then turning it into the foreground. The "foregroundDetector" function is used for this purpose, which involves four steps. In the first step, it calculates the distance between each individual pixel intensity from the mean of the GMM. In the second step, it compares if that distance is less than 2.5 times the standard deviation of that GMM or not. If it is, then the ratio of the mixing coefficient and the standard deviation is taken and checked if it is more than the threshold of 0.001. If it is, then it is classified as background. In the next step, the histograms are normalized and checked if there is any large difference between the adjacent pixel intensities histogram data. If there is a considerable distance, which is more than the threshold of 0.009, then the GMM is refitted, and the new mean, standard deviation, and mixing coefficients are found out and stored. Finally, the logical operation is carried out to convert background to foreground.

The frames are then filtered to remove inconsistencies in the next step. This is done to improve the accuracy of car detection. Next, bounding boxes are applied to the detected cars. This is done to obtain a better visualization of the detected cars in the video. Finally, the video file is run for car detection, and the project ends. In conclusion, the project involves several steps, including resizing the video file, reading 'N' frames, constructing histograms for each pixel, fitting Gaussians to the acquired data, and applying foreground detection. These steps help in detecting cars in the video file, and the final output is improved using filtering and bounding boxes. The project can be further improved by using RGB instead of the grayscale pixel intensities.

The custom made function “myfitgmdist” that we made for our project in order to replace the built-in MALTAB function “fitgmdist” has also been showed by the means of a flowchart below that shows the different steps it follows. In this the Expectation-Maximization (EM) algorithm is followed in order to reach the convergence stage for the log-likelihood and finally the returns the mean, standard deviation and the mixing coefficients of the best fit Gaussian components of the GMM.

Start

Initialization of parameters

Calculate log-likelihood using the initialized parameters

E-step

*Figure 2. Flowchart explaining the Expectation Maximization for the GMM*

M-Step

Re-calculate log-likelihood and compare with previous one

NO

Check if convergence is reached

YES

End

For the custom function, firstly the parameters for mean, standard deviation and mixing coefficients are initialized. Extra care should be taken while initializing the standard deviation as it may have a huge impact on the final results of the project. Next, the initial log-likelihood is calculated using those initialized parameters.

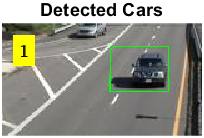
The Next step is the Expectation step, in which the responsibilities is calculated using the initialized parameters. The responsibilities is basically the probability that the ‘kth’ gaussian generated the ‘nth’ data point. Once that is done, then the code moves on to the Maximization step, where it re-calculates the mean, standard deviation and mixing coefficients using the responsibilities from the E-step. Then a new log-likelihood is calculated and compared with the previous one. If the difference is large between them, then we return back to E-step, otherwise the convergence criterion has been reached and hence the best mean, standard deviation and mixing coefficient values are obtained and the function returns them for use in the main code file.

**EXPERIMENTAL RESULTS**



Figure 3. Foreground Detection

(a)



(b)

(c)

Figure 4. (a) The car is detected in the video frame (b) another car is detected (c) When no cars are in the video frame, the top left index shows 0.

The experimental video used in our project is that which is taking place on a highway with cars moving in different time frames. After performing the foreground detection we have obtained the following result above in Figure 1. Here, the frame shows the white pixels clustered together, which represent the cars. Although there are some inaccuracies such as some of the pixels that are scattered in other parts of the frame, but they do not represent the cars. This is also taken into consideration in the code as the cars will obtain a green bounding box only after a certain number of foreground pixels are clustered together, proving that it is indeed a car.

Next, in Figure 2, (a) and (b) show the detecting cars with green bounding boxes around them and the number index at the top left corner of the video frame. Part (c) shows 0 for the top left corner index as indeed there are no cars in that particular frame. Hence we successfully implemented the GMM and EM in our project and obtained successful results.

Overall, the experimental results demonstrate the effectiveness of our proposed GMM and EM-based algorithm for car detection in video sequences. The results also suggest that the GMM and EM-based approach is a promising technique for object detection in video sequences more broadly. However, there are some limitations to our approach, including the computational complexity of the algorithm, which may limit its applicability in real-time applications.

**CONCLUSION**

It is evident that the Gaussian Mixture Model (GMM) and Expectation Maximization (EM) algorithm are useful in detecting cars in video sequences. Our project was aimed to develop an algorithm that would help to detect cars in video sequences and test it on various video sequences. In this regard, the project was successful in achieving its objectives. In this we have explained the implementation of the GMM and EM algorithms in detail and how they were used to detect cars in video sequences. We used a custom function named "myfitgmdist" to replace the built-in MATLAB function "fitgmdist" to better suit the project's needs and gain a deeper understanding of the underlying concepts. The experimental results indicate that the approach can accurately detect cars in video sequences.

Moreover, from this project we have investigated the effect of using GMM and EM on car detection accuracy compared to other machine learning techniques. The project is significant because car detection in video sequences has important applications in surveillance, traffic monitoring, and autonomous vehicles.

In conclusion, this project has provided a comprehensive understanding of the GMM and EM algorithms and how they can be used to detect cars in video sequences. The project has contributed to the development of a new GMM and EM-based algorithm for car detection, which can be further refined and improved in future research. Though we have been able to implement the GMM and EM-based algorithm, this is only for the grayscale video frames and not the color RGB video frames. The latter would be very useful in the further research work as we expect it to give much more accurate results in terms of detection of cars. This research has important implications for video analysis and surveillance systems, and it opens up avenues for further exploration in this area.

**REFERENCES**

[1] Kaewtrakulpong, P. and R. Bowden. *An Improved Adaptive Background Mixture Model for Realtime Tracking with Shadow Detection*. In Proc. 2nd European Workshop on Advanced Video Based Surveillance Systems, AVBS01, VIDEO BASED SURVEILLANCE SYSTEMS: Computer Vision and Distributed Processing (September 2001)

[2] Stauffer, C. and W.E.L. Grimson. *Adaptive Background Mixture Models for Real-Time Tracking*, Computer Vision and Pattern Recognition, IEEE Computer Society Conference on, Vol. 2 (06 August 1999), pp. 2246-252 Vol. 2.

[3] Computer Vision: Principles, Algorithms, Applications, Learning, 5th Edition - November 14, 2017, Author: E. R. Davies, eBook ISBN: 9780128095751. (The electronic version of this book is available using the Concordia University Library link <https://library.concordia.ca/>.)

[4] FirstPrinciplesofComputerVision, "Introduction to Computer Vision," YouTube video, Jul. 6, 2018. [Online]. Available: <https://www.youtube.com/watch?v=0nz8JMyFF14&ab_channel=FirstPrinciplesofComputerVision>

[5] "Detecting Cars Using Gaussian Mixture Models," MathWorks, 2021. [Online]. Available: <https://www.mathworks.com/help/vision/ug/detecting-cars-using-gaussian-mixture-models.html?searchHighlight=gmm&s_tid=srchtitle_gmm>.

**APPENDIX**

clc;

clear all;

close all;

warning off;

% Read in video file

vid = VideoReader('visiontraffic.avi');

v = VideoWriter('smallSizeVideo.avi');

open(v)

while hasFrame(vid)

frame = readFrame(vid);

frame = imresize(frame, 1/4);

writeVideo(v, frame);

end

close(v)

% Read in video file

vid = VideoReader('smallSizeVideo.avi');

N = 150;

n = 1;

frame = readFrame(vid);

[r, c, d] = size(frame);

allNFrames = zeros(N, r, c);

while hasFrame(vid) && n <= N

% Read in the current frame

frame = readFrame(vid);

% Convert the frame to grayscale

grayFrame = rgb2gray(frame);

allNFrames(n, :, :) = grayFrame;

n = n + 1;

end

allHistograms = zeros(256, r, c);

for i = 1 : r

for j = 1 : c

hist = myhist(allNFrames(:,i, j));

allHistograms(:, i, j) = double(hist);

end

end

delete hist;

% Fit 3 Gaussian components to each histogram

K = 3;

gmmMU = zeros(K, r, c);

gmmSigma = zeros(K, r, c);

gmmMC = zeros(K, r, c);

%vec1 = zeros(1, N);

maxIter = 7;

for i = 1 : r

i

for j = 1 : c

% Fit Gaussian mixture model to histogram data

vec = allHistograms(:, i, j);

y = expandHist(vec);

[mu, sigma, mc] = myfitgmdist(y', K, maxIter);

gmmMU(:, i, j)=mu;

gmmSigma(:, i, j)=sigma;

gmmMC(:, i, j)=mc;

end

end

% Process each frame

%save all;

load all;

warning off;

se = strel('square', 2);

[foreground, gmmMU, gmmSigma, gmmMC, allHistograms] = foregroundDetector(grayFrame, allHistograms, gmmMU, gmmSigma, gmmMC, K);

filteredForeground = imopen(foreground, se);

figure; imshow(filteredForeground); title('Clean Foreground');

blobAnalysis = vision.BlobAnalysis('BoundingBoxOutputPort', true, ...

'AreaOutputPort', false, 'CentroidOutputPort', false, ...

'MinimumBlobArea', 500);

bbox = step(blobAnalysis, filteredForeground);

result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');

numCars = size(bbox, 1);

result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...

'FontSize', 14);

figure; imshow(result); title('Detected Cars');

videoPlayer = vision.VideoPlayer('Name', 'Detected Cars');

videoPlayer.Position(3:4) = [650,400]; % window size: [width, height]

se = strel('square', 3); % morphological filter for noise removal

v = VideoWriter('newfile.avi');

open(v)

while hasFrame(vid) % Read in the current frame

frame = readFrame(vid);

grayFrame = rgb2gray(frame);

[foreground, gmmMU, gmmSigma, gmmMC, allHistograms] = foregroundDetector(grayFrame, allHistograms, gmmMU, gmmSigma, gmmMC, K);

% imshow(uint8(foreground));

filteredForeground = imopen(foreground, se);

% Detect the connected components with the specified minimum area, and

% compute their bounding boxes

bbox = step(blobAnalysis, filteredForeground);

% Draw bounding boxes around the detected cars

result = insertShape(frame, 'Rectangle', bbox, 'Color', 'green');

% Display the number of cars found in the video frame

numCars = size(bbox, 1);

result = insertText(result, [10 10], numCars, 'BoxOpacity', 1, ...

'FontSize', 14);

step(videoPlayer, result); % display the results

writeVideo(v, result)

end

close(v)

function [foreground, gmmMU, gmmSigma, gmmMC, allHistograms] = foregroundDetector(grayFrame, allHistograms, gmmMU, gmmSigma, gmmMC, K)

[r, c] = size(grayFrame);

background = uint8(zeros(r, c));

for i = 1 : r

for j = 1 : c

X = double(grayFrame(i, j));

mu = gmmMU(:, i, j);

sigma = gmmSigma(:, i, j);

%calculates the Euclidean distance between the pixel value and each mean.

dist = sqrt((X - mu).\*(X - mu));

% Find the Gaussian component with the minimum distance

[minDist, idx] = min(dist);

s = gmmSigma(idx, i, j);

% Check if the pixel value belongs to the selected Gaussian component

if minDist < 2.5\*s

mc = gmmMC(idx, i, j);

if mc/s > .001

background(i, j) = 1;

% lowest intensity is black and highest is white

end

end

vec1 = allHistograms(:, i, j);

vec2 = vec1;

vec2(X + 1) = vec2(X + 1) + 1;

if sqrt(sum((vec1/sum(vec1) - vec2/sum(vec2)) .^ 2)) > .009

y = expandHist(vec2);

[mu, sigma, mc] = myfitgmdist(y', K, 4);

gmmMU(:, i, j)=mu;

gmmSigma(:, i, j)=sigma;

gmmMC(:, i, j)=mc;

end

end

end

foreground = 1 - background;

foreground = logical(foreground);

function [mu, sigma, mc] = myfitgmdist(x, K, maxIters)

% Get dimensions of the data

[N, D] = size(x);

% Initialize parameters

mc = ones(1, K) / K;

mu = round(255 \* rand(K, D));

sigma = zeros(D,D,K);

for k = 1 : K

sigma(D, D, k) = 50 + 100 \* rand + 0.1;

end

lls = zeros(1, maxIters);

ll = 0;

for n = 1 : N

s = 0;

for k = 1 : K

s = s + mc(k) \* normpdf(x(n, :), mu(k, :), sigma(:, :, k));

end

ll = ll + log(s);

end

old\_ll = ll;

% E-step

iter = 1;

rnk = zeros(N, K);

while iter <= maxIters

for n = 1:N

for k = 1:K

denominator = 0;

for j = 1:K

denominator = denominator + mc(j) \* normpdf(x(n, :),mu(j, :), sigma(:, :, j));

end

numerator = mc(k) \* normpdf(x(n ,:),mu(k, :), sigma(:, :, k));

rnk(n, k) = numerator / denominator;

end

end

% M-step

for k = 1 : K

Nk = 0;

for n = 1:N

Nk = Nk + rnk(n, k);

end

s = 0;

for n = 1: N

s = s + rnk(n, k) \* x(n, :);

end

mu(k) = s / Nk;

s = 0;

for n = 1 : N

s = s + rnk(n, k) \* (x(n, :)-mu(k))' \* (x(n, :)-mu(k));

end

sigma(:,:,k) = s / Nk;

mc(k) = Nk/N;

end

ll = 0;

for n = 1 : N

s = 0;

for k = 1 : K

s = s + mc(k) \* normpdf(x(n, :), mu(k, :), sigma(:, :, k));

end

ll = ll + log(s);

end

if ll > old\_ll

lls(iter) = ll;

old\_ll = ll;

iter = iter + 1;

else

break;% if convergence criterion is achieved

end

end