Proceedings of the 3rd International Conference on Signal Processing and Machine Learning
DOI: 10.54254/2755-2721/4/2023323

The object tracking and movement prediction based on the Gaussian mixture model

Dongsheng Lyu

Rensselaer Polytechnic Institute, Troy, New York, United States

Dongshenglyu.thomas@gmail.com

Abstract. In modern computer vision study field, it has never been stopped for giving computer human ability and expecting for even more powerful functions. Most research is still in theoretical level and is hard to be applied in individual daily life. Hence it is needed to bring the advantage of new technologies in computer vision field to more people in society. This paper introduces an application in computer vision field, namely the use of Gaussian Mixture Model (GMM) for object track and movement prediction. By using GMM, the computer will have the ability to split an object from the background. Although due to sudden changes in lighting and other external factors, the general modeling of the background is not very clean and clear, the Gaussian Mixture model is one of the most successful modeling methods.

Keywords: Artificial Intelligence, Computer Vision, Object Track, Gaussian Mixture Model.

1. Introduction

It has been a while since people got interested in artificial intelligence. There are many branches in this study field, such as machine learning, robotics, or neural network. A branch of artificial intelligence known as computer vision enables computers and systems to extract useful information from digital photos, videos, and other visual inputs and to act or offer suggestions in response to that information. It is not something new for researchers, but it has not been widely used in the private sector yet. One of the purposes for this research is to make it easier for developers to apply in their applications.

A Gaussian Mixture-based background segmentation approach lies at the heart of the model utilized in the study. The primary principle behind modeling is to remove the foreground from the current frame in order to bring the background closer to the current video frame's background. That is, the weighted average of the current frame and the current background frame in the video sequence is used to update the background. This paper focuses on explaining how this system works and how it could be applied to the real world. The system can not only just be an algorithm, but it can also be an application programming interface, so that other people who need this can simply add the algorithm to their applications with fewer steps of modification. It will greatly help the developers.

DOI: 10.54254/2755-2721/4/2023323

2. Theoretical background

2.1. Background and foreground

First of all, it is needed to bring up the concept of background and foreground. For a video, it includes foreground and background. Foreground will be all those moving objects, for example the cars and people. Background usually includes all environmental objects, for example the road signs and buildings. One difficulty for the computer and computer vision algorithm is that how to let the computer know whether a pixel is in a frame foreground or background. Human can do that easily because human knows the concept of objects. From years of accumulation of knowledge, people learn what each component in our sight exactly is, and then people will view everything in a more comprehensive way, so that at the end, everything can be labeled as either environmental objects or moving objects. But computer cannot do that. This will be a key part for how to teach computers to distinguish the background from other moving objects.

2.2. Gaussian Mixture Model

The Gaussian Mixture Model characterizes each pixel in the image using K (about 3 to 5) Gaussians, and it updates the mixture Gaussian model once each new image frame is generated [6]. The stability, accuracy, and convergence of the Gaussian model are primarily defined by two parameters: variance and mean, and the various learning techniques for mean and variance will have a direct impact on the model's stability, accuracy, and convergence. Because of modeling the motion target's background extraction, the Gaussian model's variance and mean parameters must be changed in real time. The improved method uses different learning rates for updating mean and variance to improve the model's learning ability; to improve the detection of large and slow-moving targets in busy scenes, the concept of weight mean is introduced to build background images and update them in real time, and then combine the weights, weight mean, and background images to classify pixel points in the foreground and background.

3. The pseudocode for Gaussian Mixture Model

Here is the pseudocode for Gaussian Mixture Model:

1) Assign an initial mean, standard deviation, and weight to each pixel of the image.

2) Using N frames to construct the background model by the Expectation Maximization (EM) algorithm [9]. The result will contain the average value, standard deviation, and the weight of the pixel.

a. Initialize average value and standard deviation for foreground and background.

b. Calculate whether each pixel is more likely to belong to the background distribution or the foreground distribution.

c. Re-estimate the parameters of the background distribution by dividing into n pixels of the background (maximum likelihood estimation).

d. Repeating the steps a to c) until the value is stable.

3) The detection begins from the N+1 frame:

a. Sorted Gaussian k from large to small.

b. If $\frac{\omega_k}{\sigma_k}$ is large, then the pixel will be labeled as background.

c. Otherwise, it will be label as foreground.

d. Updating with the EM algorithm.

4. The implement of EM algorithm

4.1. Parameters

EM algorithm is one way in computer vision to detect the moving objects and provides the opportunity to track them. It will read the pixels of the image and apply Gaussian filter on each of them. There are three key features for this algorithm, which are the 'history', 'varThreshold' and the 'detectShadows'. The history relates with detecting the background. The value of this parameter will decide how many

DOI: 10.54254/2755-2721/4/2023323

frames will be used to build the background model [1]. If the background is fixed, then it could be a large value; otherwise, it could have noisy that can cause trouble to the final prediction. The varThreshold is the parameter to tell the algorithm how to detect whether a pixel is an object. It is the threshold on the squared Mahalanobis distance between the pixel and the model to decide whether a pixel is well described by the background model, said by the reference [2]. The algorithm will loop over all the pixels of current frame of the given video or the image from the camera. During the loop, the pixel will be counted as the object once it matches the threshold value set previously. If the pixel does not match, then it will be labeled as a part of the background components. The background component will be removed from the image of the coming frame. The detectShadows only has 2 values, 0 and 1. If it is 1, then it influences the result of object detection and returns to scalar result. If it is 0, which means disable, then the result will be binary. The reason of having two different types of result is that shadow will be grey in computer vision and it will be a number between 0 and 255, and the value will be dependent on how strong the shadow is. Although shutting the detect function down will increase the calculation speed for the program, it will decrease the accuracy of object tracking.

4.2. Implementing EM algorithm through python

It is easy to implement EM algorithm through the computer language Python. In Python, it can import the package called OpenCV, and it contains the algorithm for computer vision. First, it is needed to use the video capture function in the OpenCV package to read either the pre-recorded video or visit the camera on the equipment. The BackgroundSubtractorMOG2 is an open-source class in the OpenCV, and it supports to construct the algorithm mentioned above. In this algorithm, first step will be computing the background model and second step will be undated changing in the scene [3]. The parameter of history is set as 100 which means that it will take 100 frame to build the background model by using the Gaussian Mixture Model. The parameter of the varThreshold is set to 40 to detect the foreground object. After that the function can be run continuously by simply using a while loop, and it will stop only if the video ends or a person forces to close the program. The demo of everything now is completed and ready to work. The last step is using the apply function, which inputs the video frame and outputs the foreground mask [5]. At the end of the loop, the program will call the 'imshow' function to show both the video frame and the foreground frame.

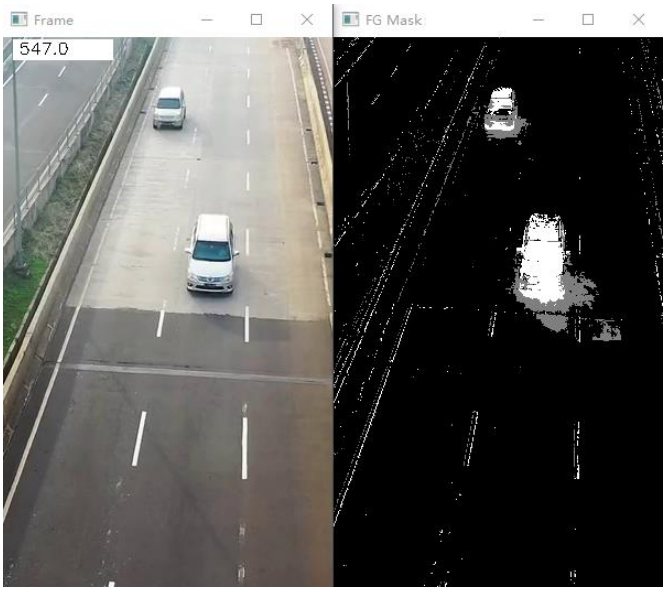

Figure 1. The screen shot of the program.

As the return result, it will split the background and other objects by given parameters. Figure 1 is the example of the program in Gaussian Mixture Model, and the example video is downloaded from the

DOI: 10.54254/2755-2721/4/2023323

website. On the left-hand side, there is a part of one frame of the original video. It shows two moving cars on the highway. In the video it is Sunday and there is no strong wind or other environment influence, which means that the camera is relatively stable, and the background data will have less noise points. On the right-hand side, there is a part of the processed graph by computer programming. Two white parts are corresponding to the two cars in the original graph. And all grey parts are the shadows belong to cars. Besides the two areas of the car, there are still white pixels, and it seems that some of them are parts of the routes and others are the edges of the guardrails. The reason that some of the background objects are counted as foreground objects is the vibration of the camera. Even though the weather is good, the wind still influences the stability of the camera. The tiny movements mislead the computer and make it believe that those areas are moving since the computer cannot differentiate the movements of the camera from the movements of an object so far.

5. The optimization through the "threshold" function

5.1. Step 1

To realize the tracking function, there is a need to add bonus functions and optimize the algorithm first, for the noise is possible to be solved if adding the threshold as the filter. When the movement of the object is not clear enough, it can be manually classified as the unimportant target which means it will not be labeled as the foreground in the result. And another function will be added is the object tracking. Now the program can tell what the foreground is and what the background is.

```
cv2.threshold(mask, 254, 255, cv2.THRESH_BINARY)
```

Figure 2. The threshold function.

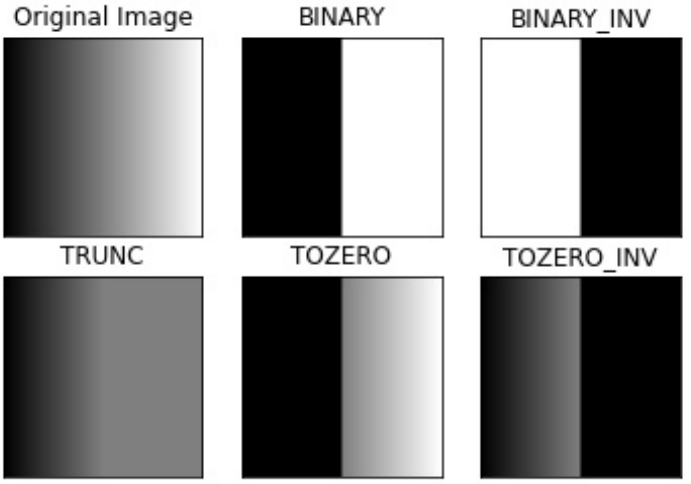

Figure 3. Multiple types of image processing.

In python, there is a function called 'threshold' [4]. A multiple-channel array is subjected to fixed-level thresholding using this function. The function is commonly used to convert a grayscale image to a bi-level (binary) image (comparison might also be used for this) or to remove noise, that is, filtering out pixels with too tiny or too big values. The function supports a variety of thresholding techniques. The type of the parameter determines what they are. It has four parameters [7]. The first one is the input image. The second one is the output array of the same size and type and the same number of channels as the first input. The third is the threshold value. The last one tells the program how to clarify the black and white area. As shown in Figure 2 and 3, the technique calculates a pixel's threshold depending on a small region around it. The Binary one satisfied the requirement for the program.

DOI: 10.54254/2755-2721/4/2023323

```
cv2.findContours(mask, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

Figure 4. FindContours () function.

5.2. Step 2

Then the next step is to find the contours of those objects in the foreground. One useful theorem is published in Satoshi Suzuki's paper "Topological structural analysis of digitized binary images by border following". In this paper, he introduces two algorithms. When a sequential digital computer is used, those two algorithms can be used for component counting, downsizing, and topological structural analysis of binary images [8]. And it has been implemented in the OpenCV package. The inside function findContours() can accept at least 3 parameters to set how this function will work for the computer.

First it needs a binary image, which means that the pixel can only be white or black. The second parameter is to decide which kind of detection method is going to be used. Hierarchy is important in objects detection. When the program finds a big contour which has a small contour inside, the outer one will be the parent contour and inner one will be the child contour. By this method, the computer can distinguish whether the two contours belong to the same object or not. There are four methods for this parameter: RETR_EXTERNAL, RETR_LIST, RETR_CCOMP, and RETR_TREE.

RETR_EXTERNAL is the most basic one. It will tell the computer detects only the outermost contours. RETR_LIST uses another logic which detects all contours and put them in a list with no hierarchical relationship. RETR_CCOMP also takes all contours. However, it will record the hierarchical relationship. This time the algorithm will build a two-level hierarchy for the contour. The top layer is the peripheral boundary of the connected domain, and the secondary layer is the inner boundary. RETR_TREE is similar with the RETR_CCOMP. It will detect all contours and build a tree to record the hierarchical relationship. To solve the complicated situation in real world, the RETR_TREE will be the best choice.

The third parameter tells the computer how to store the information of each contour. CHAIN_APPROX_NONE means that it acquires each pixel of each contour, and the pixel position difference between two adjacent points does not exceed 1. CHAIN_APPROX_SIMPLE means that it compresses the elements in the horizontal, vertical, and diagonal directions, and the values keep the focus coordinates in that direction, if a rectangular outline needs only 4 points to save the outline information.

5.3. Step 3

After implementing the findContours() function, every contour will go to a filter. If the contour is not big enough, it will be neglected. Otherwise, the location information of x-axis and y-axis, and the shape information of the width and height will be stored in a list. For each object in the information list, the computer will process the algorithm that written in another file to begin the object tracking function one by one.

The tracking function has been initiated with an empty dictionary used to store object id and a counter variable. Whenever entering the while loop mentioned before, it will call the update function in this file. After receiving the objects information as the input, for each object, the algorithm will calculate the center point first by comparing each object stored in the dictionary with the current one and using the Euclidean distance formula. If the movement or the distance is less than the threshold the program set, it will be counted as an existed object with the closest point. Then update the latest location information to the dictionary by searching with the same key id. The time of the location will also be recorded in a separate temporary list with its location and id as identity information. At the same time, the algorithm will calculate the time difference between now and the last update time, so does the displacement on x-axis and y-axis. Those data will be used to calculate the last speed vector and store in the output list with the object identity.

DOI: 10.54254/2755-2721/4/2023323

If the object does not exist in the dictionary, it will create a new one for the object. The id of the object will be current counter number plus 1, and this is also the key for the dictionary. The value of this item in the dictionary will be a tuple. The first item in the tuple is the center point location and the second will the recorded time. Both the temporary list and the return list will add this object, but both speed vectors in the return list will be set to 0. At the end, the counter will plus 1, which means there is a new object. As the last step for updating function, it will loop over the temporary list, for each object in the dictionary but not in the temporary list will be removed since there are no longer existed. Back to the main program, it will draw the box for each of the object and a line vector to predict the movement. This can be realized by the function called line() in cv2 libaray [10].

Figure 5 is an example. On the right-hand side graph, there are three people riding motorcycles. Each of them is tracked by the contour algorithm and can be seen by the computer on the left-hand side graph.

Figure 5. The example of the final program frame.

6. Conclusion

This is how to use the Gaussian Mixture Model to do the object detection and use the Euclidean distance formula to do the object. This technique can be used in any fixed background camera device or any moved mobile device. However, there is an unavoidable shortage for the Gaussian Mixture Model, which is that the computer itself cannot tell whether the movement comes from the object itself or the camera device. Taking the tree as an example, if the camera is fixed, the tree will be labeled as the background; but if camera moves, the tree will be labeled as a moving object. The latter will lead to another problem. If two moving objects overlap, the computer will get into trouble by being unable to determinate which is which. Those problems can be solved by either using a second camera to locate the object from multiple angles or using multiple algorithms such as color identity to track different objects.

References

[1] First Principles of Computer Vision. Gaussian mixture model | object tracking. YouTube. 2021. Retrieved July 12, 2022. From https://www.youtube.com/watch?v=0nz8JMyFF14.

[2] cnblogs.com. EM algorithm. Retrieved July 12, 2022. From https://www.cnblogs.com/gswang/p/7509713.html.

[3] What does the history of this function "createBackgroundSubtractorMOG2" means? OpenCV Q&A Forum. (n.d.). Retrieved July 12, 2022. From https://answers.opencv.org/question/182167/what-does-the-history-of-this-function-createbackgroundsubtractormog2-means/.

[4] What is "varThreshold". OpenCV. (n.d.). Motion Analysis. Retrieved July 12, 2022. From https://docs.opencv.org/3.4/de/de1/group__video__motion.html.

[5] How to Use Background Subtraction Methods. OpenCV. (n.d.). Retrieved July 12, 2022. From

DOI: 10.54254/2755-2721/4/2023323

https://docs.opencv.org/3.4/d1/dc5/tutorial_background_subtraction.html.
[6] OpenCV background substration: Learn OPENCV background substration. EDUCBA. 2021. Retrieved July 12, 2022. From https://www.educba.com/opencv-background-substration/.
[7] Image thresholding. OpenCV-Python Tutorials. OpenCV. (n.d.). Retrieved July 12, 2022. From https://opencv24-python-tutorials.readthedocs.io/en/latest/py_tutorials/py_imgproc/py_thresholding/py_thresholding.html.
[8] Miscellaneous image transformations. OpenCV. (n.d.). Retrieved July 12, 2022. From https://docs.opencv.org/4.x/d7/d1b/group__imgproc__misc.html#gae8a4a146d1ca78c626a53577199e9c57.
[9] Suzuki S and Abe K 1985 Topological structural analysis of digitized binary images by border following. Computer Vision. Graphics, and Image Processing. Volume 29. Issue 3. 396.
[10] Drawing Functions-OpenCV 3.0.0-dev documentation. OpenCV. (n.d.). Retrieved July 12, 2022. From https://docs.opencv.org/3.0-beta/modules/imgproc/doc/drawing_functions.html?highlight=cv2.line#cv2.line.