Report on Motion Detection: My Solution to Motion Detection

LaTeX template adapted from:
European Conference on Artificial Intelligence

Ozan Gokberk[1]

Other group members:
Arda Sahan[2], **Aysegul Kayikci**[3], **Asli Kanturk**[4]

Abstract. In this report I will be discussing Motion Detection. I am using Background Subtraction (BGS) to solve the problem that is motion detection. The other members of the group are using other solutions for the problem such as Frame Difference, Optical Flow and Noise Toleration. I use a piece of code that was made by someone else to achieve my desired solution using BGS. I then compare all the algorithms and what the similarities and difference of them are. What the pros and cons of each algorithm are also, how and when to use them. I also implement background subtraction by using Absolute Subtraction, Mixture of Gaussian and K - Nearest Neighbours. I go in depth on these different methods and how they differentiate from each other and when they should be used for motion detection.

1 Introduction

Artificial Intelligence is one of the most powerful tools made by humans. It is designed to help humans automate mundane tasks and assist them with efficiently completing different operations that cannot be done by humans or would take too long for it to be done. A few examples of general Artificial Intelligence applications are smartassistants, self-driving cars, manufacturing robots, spam and fraud detection. Artificial Intelligence can be trained with a specific train-ing model that is trained on peculiar data sets created for the problemthat it faces. An example would be a Chat bot. Chat bots are trainedwith data sets so that it responds appropriately to the asked question.A real-world example would be the very recent launch of Open AI'sChatGPT. Alternatively, developers can use Environments to developArtificial Intelligence. An example would be a game of chess. The AI learns where the chessboards pieces are by observing everythingaround the agent. A real-world example of this is Motion Detection which is what this report will be about. [3]

[1] School of Computing and Mathematical Sciences, University of Greenwich, London SE10 9LS, UK, email: og6773z@gre.ac.uk
[2] School of Computing and Mathematical Sciences, University of Greenwich, London SE10 9LS, UK, email: as1067x@gre.ac.uk
[3] School of Computing and Mathematical Sciences, University of Greenwich, London SE10 9LS, UK, email: ak7323k@gre.ac.uk
[4] School of Computing and Mathematical Sciences, University of Greenwich, London SE10 9LS, UK, email: ak7763t@gre.ac.uk

2 Background

Background Subtraction is a very widely used approach for static cameras to detect moving objects in the cameras vision. The basics of this approach is to detect a moving object by taking the reference frame and comparing it to the current frame. This is also known as" background model". How it works is that the background model/image must be a static plain image. It should not contain any abnormalities that the scene its taking doesn't normally have and should be kept updated regularly to that there won't be any errors. [7]

Background Subtraction is used in image processing and movement detection. The algorithm understands that there is motion by taking the foreground image and comparing it to the reference image. After doing this and finding out what the difference is when the foreground is subtracted from the background model, it calculates the threshold. The threshold is defined by using the first few frames of a video. Therefore, if the difference of the two frames is more or less than the threshold value, it will be able to determine what the difference is and known that there is movement. It knows there is movement because the new frame has a different threshold value from the original threshold value. This way, the algorithm finds post-comparison differences between the background model and foreground frame and build up a distance matrix.

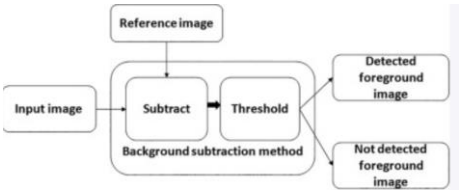
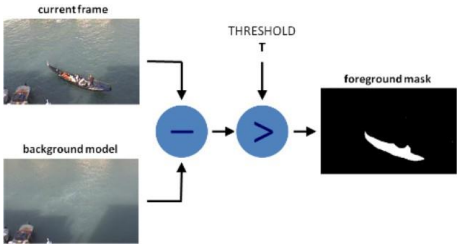


Figure 1.

There are different ways to implement background subtraction into code. One way is Absolute Subtraction. Absolute Subtraction technique takes the background image and sets that as the static background, then once its set it takes the foreground video and applies it on top of the background model. By doing this we are able to see a direct comparison.



A study was conducted on Absolute Subtraction (also known as Absolute Difference) by Rupali Nikhare. In this report Rupali created motion detection to prevent crime. She wasn't aiming to prevent a specific crime but was looking to improve motion detection in cameras to try notice crime. She used OpenCVTM for displaying an output, Global System for Mobile Communication (GSM) and Short Messaging Service (SMS) notifications. She then took all the detected info and stored it into a hard disk drive. [5]

Another method to implement background subtraction is Running Gaussian Average. This method is based upon Gaussian's probability density function which is designed to fit the last pixels values ideally. The big drawback of Gaussian's Average is that the lower the background models update rate, the fewer are the chances that the system will respond quickly in real-time when dealing with actual background dynamics.

3 Experiments and results

For the implementation and testing of the code I got a working motion detection Python code from here [9]. This code has 3 different techniques to solve the problem of motion detection. They are Absolute Subtraction, Mixture of Gaussian and K-Nearest Neighbour.

First, I used the Absolute Subtraction (Figure 2) method to solve the motion detection problem. As mentioned above Absolute Subtraction takes a static background image and uses that as the base of the frame. Then whilst using that as the background it applies the foreground frame. The foreground frame is simply the updated frames that are inputted through the camera lens. What this does is subtract the first frame from the current frame. However, there is a big downfall to Absolute Subtraction which is that if there is a object that isn't intended to be in the frame is there without the user knowing, then it will save that object in the initial frame as part of the background. Therefore, this can seriously mess up any motion

Figure 2. Absolute Subtraction

detection since it won't be able to differentiate the foreground frame to the background model with an object. For example, if there was motion detection for a security camera of a jewelry shop and the background image that was taken had a lot of customers (objects), once the store closed and there was a intruder in the store, the camera wouldn't be able to pick up on the movement since the current frame of the intruders and background frame of customers would sync together and the subtraction of the current and foreground frame will not be optimal [10]. Using Absolute Subtraction wasn't my favorite because of the disadvantages detailed above.

Figure 3. Mixture of Gaussian

After this I used Mixture of Gaussian (MOG) (Figure 3). In this model of motion detection, a mixture of k Gaussian's distributions models each individual background modelling pixel, with the values for k being 3 to 5. This method goes on the assumption that each distribution represents every background and foreground colour. Then the weight of each individual distribution on the MOG model is directly proportionate to the amount of time that each colour will stay on the single pixel. So therefore, meaning that when the weight of said distribution is considered as low, that would mean that the pixel considered low is now a part of the foreground. [8]. Using this method in the code did work better than the Absolute Subtraction method which had a major drawback. The advantage of using Mixture of Gaussian is that the background image slowly adapted in real time to the actual background of the camera. Another advantage of using Mixture of Gaussian for motion detection is that it doesn't need any Graphics Processing Unit support in order to run efficiently. This means that this method can run on small memory and low power consumption devices. Thus, making Mixture of Gaussian a great pick for surveillance devices like security cameras. [11] Mixture of Gaussian was better than Absolute Subtraction but still was lacking certain features that I envisioned for this solution.

The last method I used was K - Nearest Neighbours. K - Nearest Neighbours is a supervised machine learning algorithm meaning it relies on labelled input data to be able to learn a function that can produce relevant data when given not revised unlabeled data. KNN algorithm assumes that all similar entities are close in proximity. KNN method computes the Euclidean distance from each individual segment in the image to every training region that is defined in the initial setup. The distance between segments is measured in n-dimensional space, where n is the number of attributes for the training region. Using KNN in this situation means that recursion methods are able to constantly update the parameters of the Gaussian Mixture model and is able to simultaneously select each corresponding number to be assigned to each pixel. There are various advantages to use K - Nearest Neighbours [4]. The algorithm is simple to understand and easy to implement to any existing project. This is because on face value the KNN algorithm isn't difficult to understand and the implementation of it does not need several parameters, building of a model and additional assumptions. However, a drawback of K - Nearest Neighbours is that the more volume of data is inputted into the algorithm, the slower the algorithm will run. This is because this algorithm isn't known for its efficiency for large scale systems, it is much more useful for smaller systems like surveillance cameras since the algorithm can't identify objects rapidly. [6]

Figure 4. K - Nearest Neighbour

Overall, in my code I decided to implement the K - Nearest Neighbours algorithm because to was the most advanced out of the three options and it was the best solution for motion detection since it wasn't a large scale system and KNN is the best option for smaller applications because of the fact that it would outperform the other algorithms in speed and accuracy.

4 Discussion

My research problem was to find a solution for motion detection using Python and use the necessary algorithms to do so. From my study I have found that K - Nearest Neighbours is the smartest choice of implementation for the code. I decided on this because KNN is simple to understand for even a non technical person, it has great support online and is widely used in many systems across the globe, it also is the fastest out of Absolute Subtraction and Mixture of Gaussian.

I was not expecting to use K - Nearest Neighbours algorithm as the final method however, after researching into it deeper and implementing it into the code I was surprised at how easy and useful it was.

I compared how my algorithm worked to my other members. Arda Sahan used Frame Difference to solve the solution. Frame Difference algorithm works by taking the reference frame, taking the input frame and comparing the two and works out the difference [1]. Asli Kanturk used Optical Flow for her solution. Optical Flow is the motion of objects between consecutive frames of a sequence, which is caused by relative movement within the camera. [2] Aysegul Kayikci decided to go with Noise Tolerant. The approach for Noise Tolerant is to compare the last two frames and take the difference between them. When comparing all their algorithms with mine I came to the conclusion that Background Subtraction using K - Nearest Neighbour was the fastest algorithm to run and was also the most efficient in terms of code lines and application startup.

5 Conclusion and future work

Overall, I am happy with the outcome of this research report. I have a solution for motion detection. I used Background Subtraction with K - Nearest Neighbour as this turned out to be the most efficient algorithm for the problem from the tests I ran.

For future work I would like to improve upon the algorithm and try to attempt at writing my own Python motion detection code from scratch. I would also like to see if I could implement other algorithms to work with Background Subtraction.

ACKNOWLEDGEMENTS

I would like to thank Arda Sahan, Asli Kanturk and Aysegul Kayikci for being supportive partners and working diligently to finish this coursework to the best of their abilities thus, allowing me to work hard too.

REFERENCES

[1]
[2] Chuan en Lin, 'Introduction to motion estimation with optical flow', (2018).
[3] Pavel Hamet and Johanne Tremblay, 'Artificial intelligence in medicine', *Metabolism*, **69**, S36–S40, (2017).
[4] Onel Harrison, 'Machine learning basics with the k-nearest neighbors algorithm', (09 2018).
[5] Shraddha Mhatre, Satish Varma, and Rupali Nikhare, 'Visual surveillance using absolute difference motion detection', *Proceedings - International Conference on Technologies for Sustainable Development, ICTSD 2015*, (04 2015).
[6] Amit Pandey and Achin Jain, 'Comparative analysis of knn algorithm using various normalization techniques', *International Journal of Computer Network and Information Security*, **9**(11), 36, (2017).
[7] Massimo Piccardi, 'Background subtraction techniques: a review', in *2004 IEEE international conference on systems, man and cybernetics (IEEE Cat. No. 04CH37583)*, volume 4, pp. 3099–3104. IEEE, (2004).
[8] Bertrand Vachon Thierry Bouwmans, Fida El Baf, 'Background modeling using mixture of gaussians for foreground detection', 219–237, (2008).
[9] Anil Yadav, 'How to detect motion using background subtraction algorithms', (2021).
[10] Khalid Youssef and Peng-Yung Woo, 'Difference of the absolute differences – a new method for motion detection', *International Journal of Intelligent Systems and Applications*, **4**, (08 2012).
[11] Wu X. Gao X Zhang, C., 'An improved gaussian mixture modeling algorithm combining foreground matching and short-term stability measure for motion detection', (2019).