University of Pannonia

Faculty of Information Technology

THESIS

Motion Detection and Aperture Problem Solution with Support of Artificial Neural Network

József Köő

Supervisor: dr. László Czúni

2018

Table of contents

1 Introduction

Through my thesis labour I worked on an alternate solution of aperture problem during motion detection. On camera records it is a problem to determine whether the individual pixel on a specific frame is part of a background or part of a moving object.

1.1 Motion detection

Motion detection is a process to determine the observed object relative movement to its motionless background. Motion can be detected by various items, such as infrared sensors, radar, lidar, electromagnetic sensor and camera. Application area is in wide range since the generality if definition. Magnetic sensors are commonly used on automotive industry to determine wheel speed based on the motion of a magnetic encoder. Optical motion detection are presented in surveillance camera and alarm systems.

1.2 Problem description

In my thesis I work with object detection on camera. The task is to determine the movement on pixel level, and make a decision on every pixel whether it is background or foreground. The main stream is to decide based on the history of pixel value through frames. There are several problems on this method. Background pixels shall be those which are not changed for a while. But there are essential changes on background such as shadows, or changes of light.

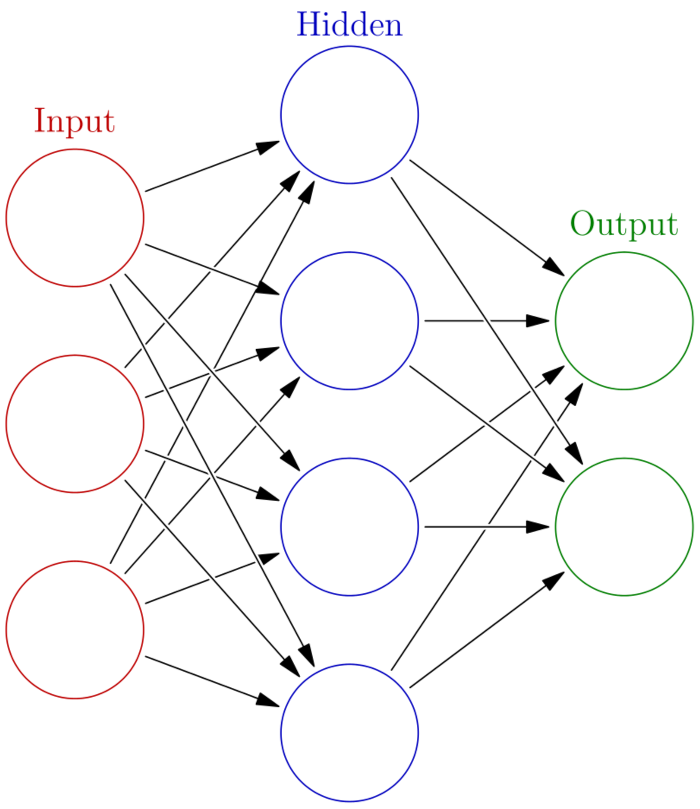
The aperture problem also makes the task challenging. Aperture problem, when a bigger object is moving on frames, or coming toward the perception system, and the contours are detected, but the pixels inside the curvature of object make no visible movement, and detected as a background faulty.

1.3 Suggested solution

On pixel level the aperture problem is hard to solve. My suggestion as a solution is to investigate the extended Mixture of Gaussians method, described in following article [Czúni, Utasi: Reducing the Foreground Aperture Problem in Mixture of Gaussians Based Motion Detection] and extend the solution with artificial neural network.

2 Artificial neural network

Artificial Neural Network (ANN) is a populate field of computer science, also it is not a new machine learning concept. The model is based on biology fundamentals. Biologists researching the neuron behaviour in living organisms and created the basic model of it. In computer science such a model is the base of the artificial neural networks.

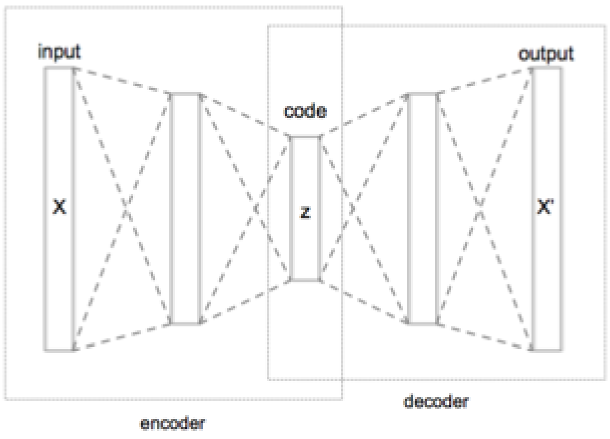


2.1 Fields of use

Artificial neural networks are a common used machine learning method. There is a wide range of usage including image processing, data mining, decision making systems, control systems, expert systems, etc… For image processing it is widely used in computer vision problems.

2.2 Autoencoder

Autoencoder is a special type of artificial neural networks. Its main purpose is to learn a way to compress the input data and restore it from a small code such a way it causes the least damage on original data.



2.3 Convolutional neural network

Convolution is often used in image processing and pre-processing. Convolution is a method where a kernel is used to iterate over the pixels and operations can applied on pixels. Convolutional neural network also uses a kernel window and these gives the information for the input nodes of neural network.

3 Development environment

For choosing the best development environment, I need to consider the usage of ANNs. Since in python these concepts are widely spread and intensively developed, it was a common sense to choose this. There are plenty of library for constructing networks and these are well optimised.

3.1 Python

Python is an interpreted high level programming language.

3.1.1 Numpy

Numpy is a library for python. It adds a support for using high dimension arrays. There are optimised array operations implemented in it. And since it is C based, it can reach such computation speed as other high level programming language.

3.1.2 Tensor flow

Tensor flow is a python library especially for machine learning solutions

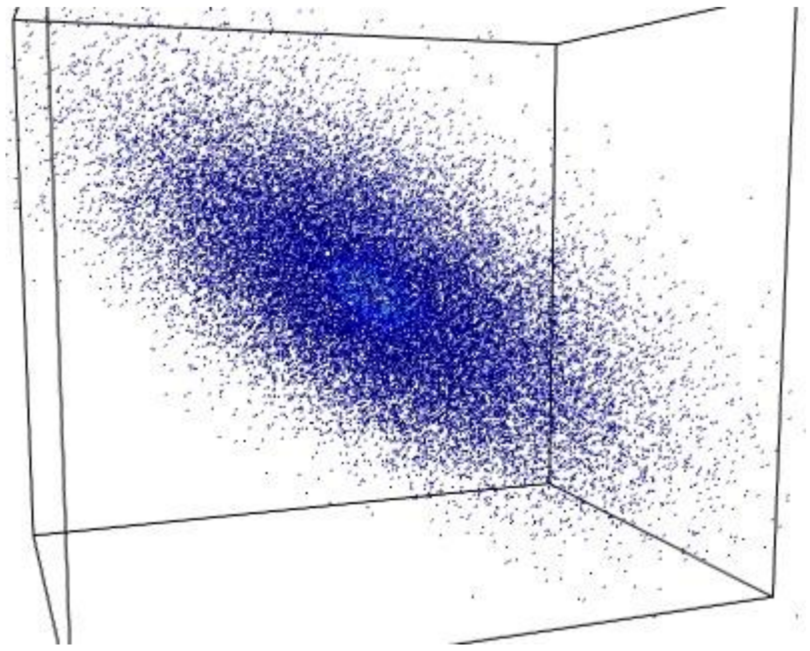
TensorFlow

4 Algorithm

The algorithm I want to extend and optimize is written in [Czúni, Utasi: Reducing the Foreground Aperture Problem in Mixture of Gaussians Based Motion Detection]

4.1 Basic mixture of Gaussian

Mixture of Gaussian method is not only used in image processing. it is a common method to describe probability variables also. The concept is, every pixel in a frame, which considered as background can be determined a K number of Gaussian distribution functions. Consider a 3 dimension Gaussian function, and the three axis are the channels of a pixel. this distribution can describe the value of a pixel even if it has small – not significant regarding motion detection – changes in time. There can be several distributions attached to a pixel, to be able to describe the natural movement of the pixel in the above mentioned colour space.



4.2 Extended mixture of Gaussians

To determine the motion, we need to check for every pixel if their current values are in the defined range. If not, we must think it is not a background pixel, so a motion detected.

$$P(x_t) = \sum_{i=1}^{K} \omega_{i,t} \eta(x_t, \mu_{i,t}, \Sigma_{i,t})$$

4.2.1 Background model

For background we calculate and refresh continuously the distributions in colour-space. So the background be adapted to still object.

$$B = \arg\min_b \left(\sum_{i=1}^{b} \omega_i > T \right)$$

4.2.2 Foreground model

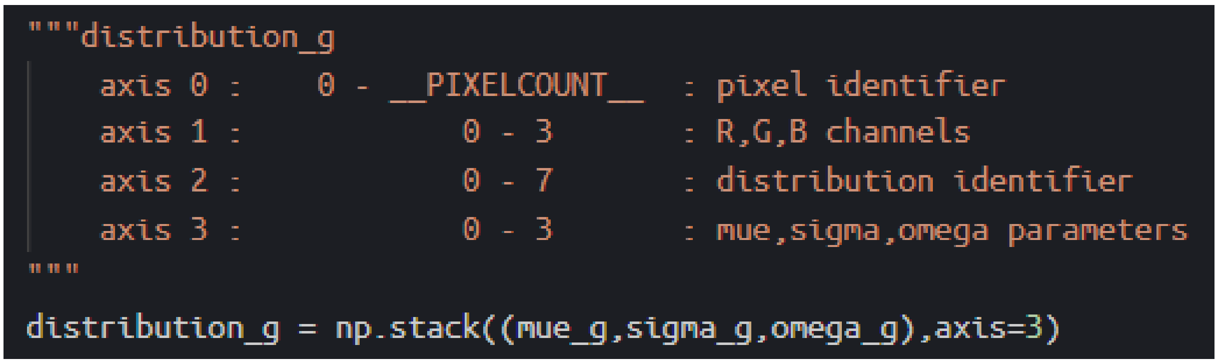
For a foreground model, if a pixel is not determined as background, we create a Gaussian for the pixel. It prevents the algorithm to absorb foreground pixel as a new value of background model.

5 Implementation

For implementation I used GIT as a version-control system. My source code can be found on [https://github.com/koojozsef/Szakdolgozat_PE]

5.1 Multidimensional array

To be able to use the advantages of Numpy library I stored the image in a multidimensional array. Pixels of a frame are flattened to a row and for each pixel I had to store the parameters of distributions.



```
"""distribution_g
    axis 0 :    0 - __PIXELCOUNT__   : pixel identifier
    axis 1 :          0 - 3          : R,G,B channels
    axis 2 :          0 - 7          : distribution identifier
    axis 3 :          0 - 3          : mue,sigma,omega parameters
"""
distribution_g = np.stack((mue_g,sigma_g,omega_g),axis=3)
```

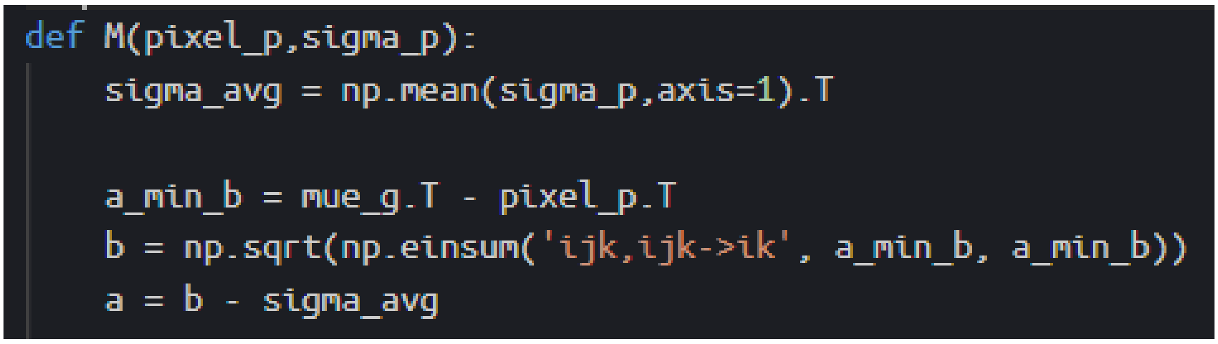
The stored parameters are: mue for expected value, sigma for covariance and omega for the weight of individual distributions. As the first two are stored per channels, and omega is stored per pixels, currently it is a redundant presence of it, since it is stored per channels.

5.2 Array operations

Array operations are useful and fast in Numpy.

5.2.1 Einsum()

Einsum is an Einstein summation. It is a well-optimised and multipurpose function. You can give two arras, and select the axis alongside you want to execute multiplication, and then you can chose the axis you want to sum.



```
def M(pixel_p,sigma_p):
    sigma_avg = np.mean(sigma_p,axis=1).T

    a_min_b = mue_g.T - pixel_p.T
    b = np.sqrt(np.einsum('ijk,ijk->ik', a_min_b, a_min_b))
    a = b - sigma_avg
```

6 Measurements

Using numpy I reached faster execution.

6.1 Runtime

During development I used the same test video. It is a 600x400 video, and I measured the computation time of my algorithm excluding the display operations.

By the first background model computation I measured the following (in seconds):

4.142091274261475
4.386620283126831
4.0766541957855225
4.1128294467926025
4.46052360534668
4.086043834686279

By using einsum() method:

3.6927168369293213
3.215099573135376
3.1615333557128906
3.1853103637695312
3.4759979248046875
3.3286538124084473

Before I iterated on pixels with a for loop, I finally used the above mentioned multidimensional array structure. It resulted the following measurements.

0.1869971752166748
0.12697458267211914
0.1254880428314209
0.12796759605407715
0.12598490715026855
0.1254885196685791

7 Accomplishment

During my Thesis labour I accomplished to implement the base of extended Mixture of Gaussians algorithm for motion detection. I also get familiar with the development environment and the concept of neural network.

8 Future improvement

My future plan is to find a solution using artificial neural network for motion detection, which can deal with aperture problem.