Thesis

# Tutorial on Minimum Output Sum of Squared Error Filter

Submitted by

Rumpal Kaur Sidhu

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Master of Science

Colorado State University

Fort Collins, Colorado

Spring 2016

Master's Committee:

Advisor: Bruce A. Draper

Ross J. Beveridge
Lucy Troup

Copyright by Rumpal Kaur Sidhu 2016

All Rights Reserved

## Abstract

### Tutorial on Minimum Output Sum of Squared Error Filter

One of the major challenges in computer vision is visual tracking. Many different techniques have been developed for visual tracking over time. Visual tracking using correlation filters is faster than other techniques. A visual tracking technique using Optimized Correlation Output Filters (OCOF), Average of Synthetic Exact Filter (ASEF) and Minimum Output Sum of Squared Error (MOSSE) filters is presented in this tutorial. The MOSSE filter is a stable correlation filter which can be initialized on a single frame of a video. The MOSSE filter adapts with the changes in the appearance of the object while tracking. Tracking using the MOSSE filter is not dependent on changes in lighting, non-rigid transformations, pose, and scale. This work presents systematic steps towards building the filter and its application as a tracker in visual tracking. A number of experiments are performed to demonstrate the working of the tracker.

## Acknowledgments

I am sincerely grateful for all the patience, motivation and guidance of my advisor Dr. Bruce Draper who provided me with his knowledge and expertise in the subject. I would also like to thank Dr. Ross Beveridge for his guidance and motivation. I appreciate the assistance of David Bolme. I am thankful to all my friends and family who have helped during this project.

"If I have seen further, it is by standing on the shoulders of giants." - Isaac Newton

Contents

# List of Tables

# List of Figures

CHAPTER 1

# Introduction

Visual tracking can be defined as the process of detecting the position of an object in each frame of a video. Visual tracking has many real life applications in security, surveillance, traffic control, etc. Once an object can be tracked it can be labelled and more information can be extracted. Minimum Output Sum of Squared Error (MOSSE), a form of Optimized Correlation Output Filter, can be used for tracking. Examples of other robust tracking techniques are Fragments-based Robust Tracking [1], Incremental Visual Tracking [2], Struck: Structured output tracking with kernels [3], Locally Orderless Tracking [4], L1 Tracker [5], and Multiple Instance Learning Tracking [6].

This tutorial discusses a filter initialization technique called Minimum Output Sum of Squared Error (MOSSE) [7] and its application in visual tracking. The MOSSE filter was developed by David Bolme, a PhD student in the Computer Science department and Computer Vision group of Colorado State University. MOSSE creates correlation filters, that significantly outperform simple templates and map input images to their ideal outputs. This reduces interference with the background and achieves better performance. MOSSE filters are robust to changes in lighting, scale, pose and shape of an object.

In this tutorial, four incremental versions of a MOSSE tracker are presented, ranging from simple to complex. The tutorial presents the math and the steps for building the filter. It also presents systematic steps for implementing the MOSSE filter inside a tracker along with the math and the examples on the MOSSE tracking algorithm. Experiments are presented to illustrate the working of the tracker.

(A) Frame 10

(B) Frame 65

(C) Frame 75

(D) Frame 120

FIGURE 1.1. A series of frames from a video illustrating the tracking process of a soccer ball.

(A) Frame 100

(B) Frame 150

(C) Frame 200

(D) Frame 990

FIGURE 1.2. A series of frames from a video illustrating the tracking process of a girl playing in the playground.

### 1.1. MOSSE tracking algorithm

The MOSSE tracking algorithm is briefly presented in this section. There are two main components to the algorithm, initialization and tracking. For initialization purposes, an object is selected using either the first few frames for a simple version of the tracker or the first frame for a more complex version of the tracker. The object is cropped and centered to initialize the filter. The initialized filter is then correlated with a tracking window in the video to find the new location of the object. In a complex version, the tracker updates the filter as it tracks.

An object is selected by a user by clicking on its center for initializing the filter. A bounding box is drawn around the object after selecting an object for labeling. The bounding box represents the template during initialization and the tracking window during tracking. The template is cropped to initialize and update the filter during tracking.

A preprocessing step is performed on each frame of the video before initializing the filter and tracking the object. The template obtained from the preprocessing step is converted to the Fourier domain. A synthetic output is generated to initialize and update the filter. The synthetic output is also converted to the Fourier domain, and the filter is computed in the Fourier domain. The output of the correlation is converted back to the spatial domain to find the new position of the object in the next frames of the video during tracking. The correlation is performed in the Fourier domain to make the computations faster.

The data set used for demonstrating the MOSSE tracker is a video collected by Colorado State University (CSU) of children playing in a play ground. In this video children are swinging, playing tag, playing with a soccer ball, playing with a football, and playing on a jungle gym. Figure 1.1 shows a tracked soccer ball in the $10^{th}$, $65^{th}$, $75^{th}$, and $120^{th}$ frames of the video. Figure 1.2 illustrates the tracking process of a girl in the $100^{th}$, $150^{th}$, $200^{th}$,

and $990^{th}$ frames of the video. The black bounding box around the soccer ball and the girl illustrates the tracking window. The black point inside the bounding box illustrates the position of the highest peak when the filter and the tracking window are correlated. The filter for each frame is displayed on the top left corner of the frame. All figures presented in this tutorial were produced as a result of the work performed for this Master's thesis by Rumpal Sidhu.

### 1.2. Challenges with visual tracking

Visual tracking is difficult due to variations in the appearance of an object and its surroundings in a video. Object appearance changes due to pose, lighting, natural variations or non rigid transformation. Objects may be occluded or move out of the frame of the video. These variations make tracking difficult.

The pose of an object can change, such as the bearing of a living being that changes within an activity, or with different activities. Lighting also changes, for example when a shadow falls over an object, the light source, or the angle changes, or there is a change in the intensity or brightness of the light. Natural variation include changes in the physical appearance among the objects in a group, such as different sizes of balls, or people with different physical characteristics. In non rigid transformations the shape and/or the size of an object changes, for example, as a ball moves away from a camera, the ball appears smaller in the video. Stretching or shrinking are non rigid transformations. These changes can cause the tracker to lose the original object.

When an object being tracked is occluded or moves out of the frame, the tracker might lose the object and stop tracking, or the tracker might start tracking an object that resembles the original object. The tracker may stop updating a filter once the object is no longer in

the frame or is occluded, but it may resume tracking and updating the filter once the object reenters the field of view.

### 1.3. Definition of terms

A series of terms and their definition in context with this tutorial are presented here. These terms repeat throughout the tutorial and are important to understand. It is not necessary to go through this section right now but one may return to understand the terms as required.

*Object/Target*: An object is a person or thing which has definable characteristics. An object in this tutorial is referred to a physical entity to be tracked. Sometimes, the object is also referred to as a target.

*Template*: A template is a sub-image cropped from a larger image. The template is used as an input image to initialize the filter. An image of an object is present in the template. Also it may be referred to as correlation template.

*Correlation*: Correlation is a measure of similarity between two images. It is the sum of pairwise products of corresponding pixels of two images. Template matching uses correlation of a template and an image to find the location of an object in the image. The template is moved to different locations in the image and the position of the best match, highest correlation is found.

*Filter*: Typically a filter is just a sub-image and the act of filtering is the process of placing the filter over the image at different locations and computing the pairwise-sum of the filter and corresponding image pixels. When the filter is applied across an entire image the result is itself a new image, i.e. a filtered image.

*Tracking window*: A tracking window is a sub-image in which the tracker looks for the new location of the object. The tracking window is retrieved from a frame of the video. The tracking window correlates with the filter to give the new location of the object being tracked.

*Synthetic target*: A synthetic target is a synthetically generated image with a Gaussian peak at the location of the object to be tracked. A synthetic target is used to map the input image to its corresponding correlation output to generate a filter.

*Occlusion*: An occlusion is caused by an object which blocks the view of another object in a video.

*Tracking*: Tracking is an action where the algorithm finds the new location of an object in all the frames of a video over time.

*Initialization*: Here initialization is a process where the filter used for tracking an object in a video is generated. In this tutorial, initialization is also referred to as training.

*Updating*: In this tutorial, updating is a process where a filter is updated with the new information about the object, for example change in the pose of a person or change in the scale of an object is a new information.

### 1.4. Organization of the tutorial

Chapter 2 provides an introduction on Optimized Correlation Output Filters (OCOF). It provides a step by step incremental tutorial on the Exact filter, Average of Synthetic Exact Filter (ASEF) and MOSSE filter along with the basic math for MOSSE.

Chapter 3 provides a tutorial on four incremental versions of the MOSSE tracker, including the necessary math. The first version is the bare minimum form of the MOSSE tracker.

The second version updates the position of the tracking window over time. The third version updates the filter over time. Finally, the fourth version implements affine perturbations during training to improve the stability of the filter initialization process and increase the initialization data set size.

Chapter 4 presents the results of experiments performed on different versions of the MOSSE tracker. The chapter also discusses how MOSSE parameters values effect tracking for different objects. Finally, Chapter 5 summarizes the conclusions and discusses future work.

CHAPTER 2

# Optimized Correlation Output Filters

A detailed tutorial on the two Optimized Correlation Output Filters (OCOF), Average of Synthetic Exact filter (ASEF) and Minimizing the Output Sum of Squared Error (MOSSE) filter, is presented in this chapter. The following steps are required to initialize an OCOF: (1) obtain a template and preprocess it, (2) create a synthetic target, (3) convert both the preprocessed template and the synthetic target from the spatial to the Fourier domain. The goal of this chapter is to walk through the steps for initializing the filters.

## 2.1. Comparing an image to a template

A simple technique known as template matching can be used to find similarity between a template and an image by comparing their pixels. A dot product compares the pixels between a template and an image. The following equation is used to compute a dot product,

$$F(x, y) = x \cdot y = \sum_i x_i y_i \tag{1}$$

In Equation 1, $x$ is the template and $y$ is the image. The dot product of two similar images is high compared to that of two different images; hence a threshold value can be determined to find an object in an image.

In template matching, correlation is used to slide a template over an image to find every possible alignment of the template over an image. The pixels for every possible alignment are compared by computing a dot product to find a similarity score. Figure 2.1 shows an image of $1280 \times 720$ pixels and Figure 2.2 shows a template of $64 \times 64$ pixels. Figure 2.3 illustrates

the correlation of a template over an image for every possible combination. Starting at the top left corner, the template is moved by one pixel at a time from left to right and top to bottom. Correlation produces a new image with the dot product for every alignment. The peak in the correlation image is used to find the location of the template in the image. A threshold can be used to determine the existence of the object in the image.

During correlation, the images are projected in a correlation space. A correlation space is an N dimensional space where N is the product of the width and the height of an image. To project an image in the correlation space, first the image is unrolled into a one dimensional vector and a mean of the vector is computed. The mean value is then subtracted from the vector, after which the vector is normalized to make it a unit length vector. Therefore, correlation space is a unit length sphere and taking a dot product of two vectors in the correlation space is taking cosine of the angles between the vectors.

This technique of template matching works fairly well for matching the same type of objects with similar appearance yet it is not useful for comparing different objects with similar characteristics. Template matching does not work as well, when comparing two different images of the same object taken in completely different environments and conditions. Therefore, template matching does not work well for the variations in pose, lighting, natural variations or nonrigid transformations.

## 2.2. Preprocessing

The MOSSE filter tracking algorithm has two main components, initializing the filter and tracking the object. A preprocessing step is performed on every frame of the video before initialization or tracking. This section discusses the preprocessing steps for the MOSSE

FIGURE 2.1. A frame from a video of children playing in the play ground. The size of the frame is $1280 \times 720$ pixels.

FIGURE 2.2. A template of a soccer ball cropped from a frame of the video illustrated in Figure 2.1. The size of the template is $64 \times 64$.

filter tracking algorithm. The following preprocessing steps are performed on every frame of a video before they are used in the MOSSE filter tracking algorithm,

(1) A template centered on the object with a dimension of $2^n \times 2^n$ is cropped from the frame of a video. Figure 2.2 illustrates a template of $64 \times 64$ pixels cropped from the frame of $1280 \times 720$ pixels shown in Figure 2.1.

(2) The template obtained in the previous step is converted from its color to a gray scale image. Figure 2.4 illustrates the gray scale intensity image of the template.

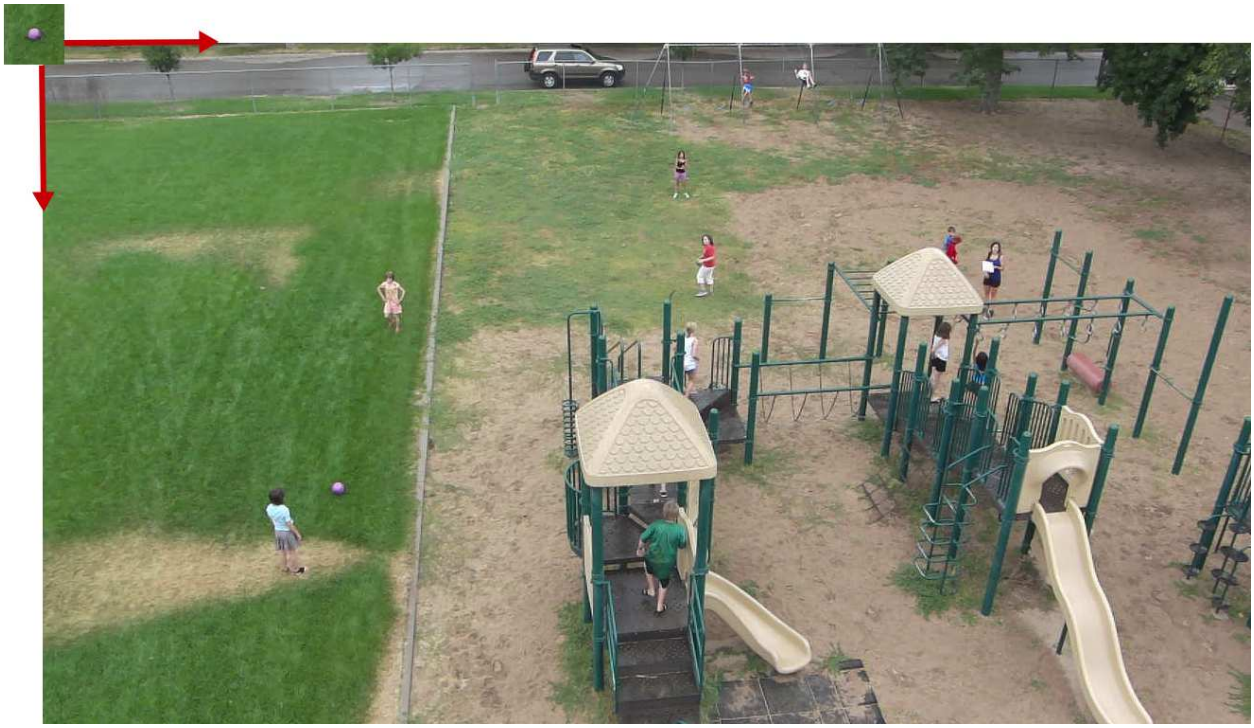


FIGURE 2.3. Illustration of correlation by sliding a template against an image.

FIGURE 2.4. A grayscale intensity image of the template of a soccer ball illustrated in Figure 2.2.

(3) A log transformation is performed on the template obtained from the previous step by using the following equation,

$$x = \ln(y + 1) \tag{2}$$

In Equation 2, x and y are the pixel values of the output and the input images respectively. The log function is applied to reduce lighting effects and enhance contrast, making high contrast features available for the filter to initialize on.

(4) The pixel values of the template from the previous step are normalized to get a mean of zero and a normal of one. The normalization helps in reducing the effects of change in illumination and maintaining a consistency in illumination between the different frames of the video.

(5) The template obtained from the previous step is converted from the spatial domain to the Fourier domain. The Fourier transform is used to decompose a signal into its sine and cosine components. An image is not an analogue signal therefore, discrete Fourier transform (DFT) is used for an image. The output of this transformation represents the image in the Fourier or frequency domain. The following equation is used for the DFT in two dimensional image,

$$(3) \qquad F(u,v) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x,y) e^{-2\pi i \left(\frac{xu}{M} + \frac{yv}{N}\right)} \qquad u = 0, \ldots, M-1; \quad v = 0, \ldots, N-1$$

In Equation 3 [8] [9], $F(u,v)$ represents the image in frequency domain and $f(x,y)$ represents the image in the spatial domain. The following equation is used to convert from the frequency domain to the spatial domain know as inverse DFT,

$$(4) \qquad f(x,y) = \frac{1}{NM} \sum_{u=0}^{M-1} \sum_{v=0}^{N-1} F(u,v) e^{2\pi i \left(\frac{xu}{M} + \frac{yv}{N}\right)} \qquad x = 0, \ldots, M-1; \quad y = 0, \ldots, N-1$$

In Equation 4 [8] [9], $F(u,v)$ represents the image in frequency domain and $f(a,b)$ represents the image in the spatial domain.

In Figure 2.5, the preprocessing step is illustrated through a flow chart. The preprocessing subroutine takes a frame as an input, and returns a preprocessed template image

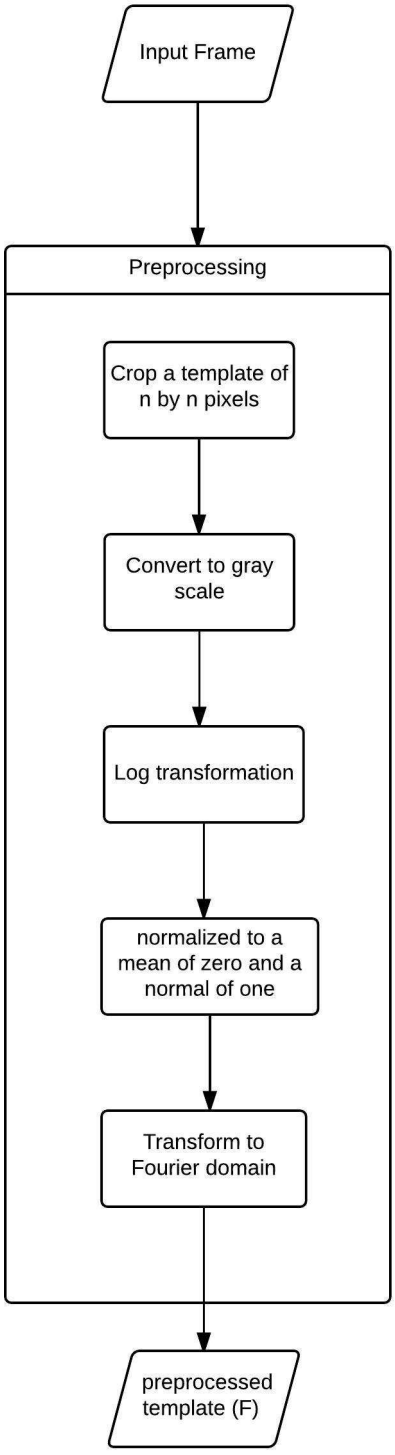



FIGURE 2.5. Flow chart for the preprocessing step.

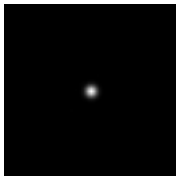


FIGURE 2.6. An image of a synthetic target for the template illustrated in Figure 2.2.

in Fourier domain. The five important steps in the processing subroutine are shown beginning with cropping a template, converting it to gray scale, performing a log transformation, normalizing the template and performing discrete Fourier transform.

### 2.3. Synthetic target

A synthetic target is a synthetically generated desired correlation output. A synthetic target image contains a Gaussian peak centered on the object. The synthetic target is used in the initialization of the filter and for updating the filter during tracking by mapping the template to its desired output. The following equation is used to synthetically generate the synthetic target image,

$$g_i = \sum e^{-\frac{(x-x_j)^2+(y-y_j)^2}{\sigma^2}} \tag{5}$$

In Equation 5 [10], $g_i$ is the synthetically generated target. x and y represent the location of pixels in the image. $x_j$ and $y_j$ represent the location of the center of the object to be initialized on. The radius of the peak is specified by $\sigma$. A value of 2 is used for $\sigma$ in this tutorial. Figure 2.6 illustrates the image of a synthetic target for the template shown in Figure 2.2.

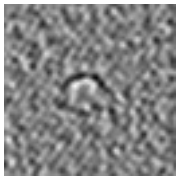


FIGURE 2.7. An Exact filter of the soccer ball illustrated in Figure 2.2.

## 2.4. Exact filter

An exact filter, when correlated with a template on which the filter was initialized, produces a strong peak at the location of the object and zero values at all the other locations. The exact filter gets its name due to the ability to exactly transform the input image to its target output image [10]. The initialization data consists of a pair of training images, a template and a corresponding synthetically desired output [11]. The following steps are used to initialize an exact filter,

(1) A template of $64 \times 64$ pixels is cropped from a frame of a video with the center of object centered on the template. Figure 2.1 illustrates a frame from a video and Figure 2.2 illustrates its corresponding cropped template.

(2) The template is preprocessed using the steps from Section 2.2.

(3) A synthetic target of $64 \times 64$ pixels is generated with the Gaussian peak centered on the center of the object using Equation 5. The steps from Section 2.3 are used to generate the synthetic target. Figure 2.6 illustrates the synthetic target image for the preprocessed template in Figure 2.2.

(4) The synthetic target image is converted to the Fourier domain using the Fast Fourier Transformation. The function *np.fft.fft2()* from the *numpy* package in the Python programming language is used to convert images to the Fourier domain in this tutorial.

(5) The following equation is used to obtain the exact filter,

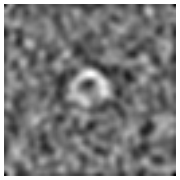


FIGURE 2.8. ASEF for the template illustrated in Figure 2.2.

$$H^* = \frac{G_i \odot F_i^*}{F_i \odot F_i^* + \varepsilon} \tag{6}$$

In Equation 6 [10], $H^*$ is a complex conjugate of the filter in the frequency domain. $F$ is the preprocessed template in the Fourier domain. $G$ is the synthetic target image in the Fourier domain and $F^*$ is complex conjugate of $F$. Element-wise multiplication is denoted by the symbol $\odot$. Figure 2.7 illustrates an exact filter of the soccer ball shown in Figure 2.2.

## 2.5. ASEF

An ASEF is an average of multiple exact filters. The exact filters do not perform well when tested on images other than the initialization images because they do not generalize. An average of exact filters is taken to create a more general filter. Taking an average cancels out the inconsistent features of the object while retaining the consistent features [10]. The following steps are used to initialize ASEF,

(1) A template of $64 \times 64$ pixels is cropped with the center of the template as the center of the object. Figure 2.2 illustrates the cropped template for a frame of a video illustrated in Figure 2.1.

(2) The steps from Section 2.2 are used to preprocess the template.

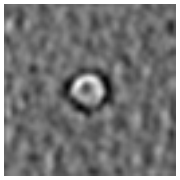


FIGURE 2.9. MOSSE filter for the template illustrated in Figure 2.2.

(3) The steps from Section 2.3 are used to generate a synthetic target of $64 \times 64$ pixels with the center of the object centered on the Gaussian peak of the synthetic target.

(4) The synthetic target image is converted to the Fourier domain using the Fast Fourier Transformation.

(5) The following equation is used to generate ASEF,

$$H^* = \frac{1}{N} \sum_{i=1}^{N} {H_i}^* \tag{7}$$

In Equation 7 [10], $H^*$ is a complex conjugate of the filter in the Fourier domain. N is the number of initialization images. ${H_i}^*$ is the $i^{th}$ exact filter for the $i^{th}$ initialization image. Figure 2.8 illustrates an ASEF for the soccer ball shown in Figure 2.2.

## 2.6. MOSSE filter

The MOSSE filter minimizes the sum of squared error between the actual output of the correlation and the desired output of the correlation [10]. The MOSSE filter uses a set of image pairs for initialization. The initialization data for the MOSSE filter is identical to ASEF however, MOSSE filter requires a smaller number of initialization images as compared to ASEF. The first three steps for the MOSSE filter are the same as the first three steps for ASEF. First a template is cropped, then it is preprocessed and converted to the Fourier

domain. After that a synthetic target image is created and converted to the Fourier domain. The MOSSE filter is initialized using the following formula,

$$H^* = \frac{\sum_i G_i \odot F_i^*}{\sum_i F_i \odot F_i^* + \epsilon} \tag{8}$$

In Equation 8 [10], $H^*$ is a complex conjugate of the filter. $F_i$ represents the preprocessed cropped template in the Fourier domain for the $i^{th}$ frame of the video. $G_i$ is the synthetic target image in the Fourier domain for the $i^{th}$ frame of the video. $F_i^*$ is the complex conjugate of $F_i$. An element-wise multiplication is denoted by the symbol $\odot$. $\epsilon$ is the regularization parameter. A value of 0.001 is used for $\epsilon$ in this tutorial. $\epsilon$ is used to avoid divide by zero errors. Figure 2.9 illustrates the MOSSE filter generated using one frame of the video for the soccer ball shown in Figure 2.2.

CHAPTER 3

# MOSSE Tracker

A detailed tutorial on the application of the MOSSE filter as a tracker is presented in this chapter. The MOSSE filter is initialized on a set of templates for tracking. The basic math for the tracker is presented in this chapter. A step wise development of the MOSSE tracker is discussed, beginning with a basic MOSSE tracker, then adding updates first to the tracking window and then to the filter. Lastly, the use of affine transformations to perturb frame sample for initializing the filter is discussed.

## 3.1. Mathematics of the MOSSE tracker

A filter initialized using one template can track a simple object for a small number of frames. The filter has to be initialized on a set of templates for efficient and continuous tracking. The basic MOSSE tracker uses the first few consecutive frames of the video to initialize a filter. The preprocessed cropped template in the Fourier domain, $F$, is obtained from Section 2.2. The synthetic target in the Fourier domain, $G$, is obtained from Section 2.3. F and G are used to initialize the filter,

$$N_i = \eta(G_i \odot F_i^*) + (1 - \eta)N_{i-1} \tag{9}$$

$$D_i = \eta(F_i \odot F_i^* + \epsilon) + (1 - \eta)D_{i-1} \tag{10}$$

In Equation 9 [10] and 10 [10], $F_i$ is the $i^{th}$ preprocessed template in the Fourier domain for the $i^{th}$ frame in the video. $G_i$ is the $i^{th}$ synthetic output image in the Fourier domain for the $i^{th}$ frame in the video. The learning rate is represented by $\eta$. The learning rate lies between 0 and 1. The symbol $^*$ represents the complex conjugate. The symbol $\odot$ represents element-wise multiplication. The regularization parameter is represented by $\epsilon$. The cumulative values of $N_i$ and $D_i$ over a set of initial frames is used in the following equation to initialize the filter:

$$H_i^* = \frac{N_i}{D_i} \tag{11}$$

In Equation 11 [10], $H_i^*$ denotes the complex conjugate of the MOSSE filter. Once the filter is initialized, the next frame of the video, becomes the current frame for the tracker. A tracking window is cropped from the current frame of the video with the tracking window centered on the position of the object in the previous frame. The tracking window is preprocessed and transformed to the Fourier domain. The tracking window is then multiplied with the filter to get the new position of the object in the current frame using the following equation:

$$G = H^* \odot F \tag{12}$$

In Equation 12 [10], $F$ is the cropped and preprocessed tracking window. The size of the template, the tracking window, and the filter is the same so that element-wise multiplication can be performed. The multiplication of $H^*$ and $F$ results in a peak which indicates the new

location of the object [10]. The math presented briefly in this section is illustrated with an example in the next section for the basic MOSSE tracker.

### 3.2. The basic MOSSE tracker

An example for the basic MOSSE tracker is presented in this section. The input video is a video of children playing in a playground. A fixed camera was used to record this video. There are many moving objects in the video such as a soccer ball, football, children, swings, etc. Two objects from this video are used as objects to demonstrate tracking in this section. The first object used for tracking is a soccer ball. A girl facing the camera in a beige color dress playing with the soccer ball is the second object. A torso is the most rigid part of a human body and does not have a wide range of movements unlike that of the limbs. Therefore, the filter for the second object is initialized on the torso of the girl.

The tracker works better when the filter is initialized on a set of templates rather than single template. Therefore, the filter is initialized on the first seven frames of the video. The object is selected by the user using a mouse to click on the center of the object. This is repeated six times. The steps performed for a basic MOSSE tracker are as follows,

(1) The user clicks on the center of the object which provides the tracker with the x and y coordinates for the center of the object. The first frame of the video is shown in Figure 2.1.

(2) A $64 \times 64$ pixel template is cropped from the first frame of the video with the center as the x and y coordinates obtained from the previous step. The cropped template is illustrated in Figure 2.2.

(3) The template obtained from previous step is preprocessed and transformed to the Fourier domain retrieving $F$ as explained in Section 2.2

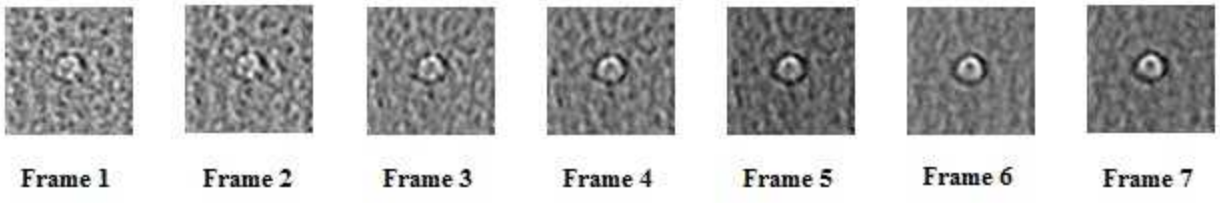



FIGURE 3.1. A basic MOSSE filter is initialized on the first seven frames of the video of the children playing in the play ground with soccer ball as the object.

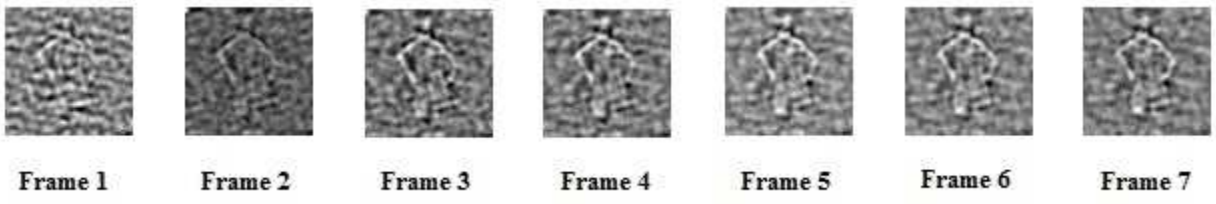



FIGURE 3.2. A basic MOSSE filter is initialized on the first seven frames of the video of the children playing in the play ground with the girl in the beige color dress as the object

(4) A synthetic target, $G$, is created as explained in Section 2.3.

(5) The values of $F$ and $G$ from the previous steps are substituted for the values of $F_1$ and $G_1$ respectively in the following equation to compute $N_1$ and $D_1$ for the first frame of the video,

$$N_1 = (G_1 \odot F_1^*) \tag{13}$$

$$D_1 = (F_1 \odot F_1^* + \epsilon) \tag{14}$$

(6) Steps 1 through 5 are repeated for the next six consecutive frames. $N_i$ and $D_i$ are computed for every frame using Equations 9 and 10. A value of 0.065 is used for the learning rate in Equation 10. A value of 0.001 is used for the regularization parameter.

(7) The final $N_7$ and $D_7$ obtained from the previous step is used to compute the filter using Equation 11. Equation 11 gives the complex conjugate of the filter, $H^*$. Figure 3.1 illustrates the basic MOSSE filters while initializing over the first seven frames of the video. The filter adapts over seven frames of the video and the boundary of the ball becomes more distinct in the seventh filter.

(8) The tracker proceeds to the eighth frame of the video and retrieves a tracking window. A tracking window of $64 \times 64$ pixels and a fixed location is used for this demonstration. The center of the tracking window is the same as the center of the object selected by the user in the last frame of filter initialization, i.e., the seventh frame of the video for this example. The tracking window is then preprocessed and converted to the Fourier domain, $F$. This preprocessed tracking window in the Fourier domain is correlated with the filter using Equation 12. The location of the peak in $G$ indicates the new location of the object.

(9) The previous step is used to track the object in the consecutive frames of the video with the location of the tracking window fixed at one position. The object can be tracked as long as the entire object remains inside the tracking window.

Sample code is provided in a package for the basic MOSSE tracker. The code can be run for illustration purposes. The user can select the center of the soccer ball by clicking with a mouse on the display window. A black bounding box will appear showing the selection of the object by the user. After the object is selected, press the space bar to proceed to the next frame of the video. Repeat the object selection process for the next six consecutive frames. The GUI starts displaying the filter on the top left corner of the display window from the second frame onward. Figure 3.3 illustrates the GUI of the second frame of the video, the filter in spatial domain is displayed on the top left corner of the GUI. The tracker starts

FIGURE 3.3. GUI displaying the second frame from the video of the children playing in the play ground during initialization.

FIGURE 3.4. GUI displaying the tenth frame from the video of the children playing in the play ground during tracking.

tracking the object after initializing the filter on the first seven frames of the video. Figure 3.4 illustrates the tenth frame of the video during the tracking process. The black dot on the white ball indicates the new position obtained from the tracker and a black bounding box shows the tracking window. Press the space bar to proceed to the next frame of the video. The same steps can be performed for the girl. Click on the center of the torso of the girl to select her as an object.

When the soccer ball is the target, the tracker tracks the ball until the ball moves out of the tracking window. On the other hand when tracking the girl playing with the soccer ball, the tracker does not perform as well when the girl's pose changes; the tracker starts tracking the head of the girl instead of the torso. Nonetheless, the tracker tracks the girl until she moves out of the tracking window.

### 3.3. Tracking window update

Tracking without updating the tracking window gives a basic tracker that is limited. If the object moves out of the tracking window, the tracker loses the object and begins tracking the most similar object inside the tracking window. This is not the desired result, therefore the position of the tracking window should be updated to continue tracking the object. When the tracker starts tracking after initializing the filter, a new position of the object is obtained for every frame. For a moving tracking window, the center of the tracking window is updated with the new position of the object obtained for every new frame. The new $x - y$ coordinates of the object becomes the new center of the tracking window.

The tracker tracks the ball for a larger number of frames with the tracking window update in comparison with the basic MOSSE tracker. The tracker keeps tracking the ball when the

ball is kicked for the first time. The tracker loses the ball when the ball is kicked for the second time by another girl.

The tracker tracks the second example object for a greater number of frames with the tracking window update rather than the basic MOSSE tracker. The tracker loses the girl for a few frames as the girl's pose changes, and then starts tracking again. The tracker loses the girl when she is kicking the ball. The tracker starts tracking the girl again when she gets back to the standing pose, which is close to the initialization pose. When there is a slight variation in the pose of the girl, the tracker starts tracking the head of the girl instead of her torso. For example, if the girl's arms are in a different position than the initialization frames, or if the girl is looking in a slightly different direction, the tracker starts tracking the head or the edge of the torso, rather than the center of the torso.

### 3.4. Tracking with filter update

Tracking without updating the filter is still limited. The tracker stops tracking the object if the pose or the size of the object changes. In this section, updating the filter is discussed as the tracker tracks. The basic MOSSE tracker can be used for tracking simple symmetrical objects such as a soccer ball. A more advanced version of the tracker is required when the appearance of an object changes continuously, for example, a human's pose can change while performing an activity. This version of the tracker updates the filter as it tracks the object so that the filter is updated with the current pose of the object.

The steps followed to create the tracker are the same as the first 9 steps of the basic MOSSE tracker. When the tracker starts tracking, the tracking window and filter are updated. Once the tracker gets the new position of the object, a new tracking window is cropped with the new position of the tracker. The the size of the tracking window remains

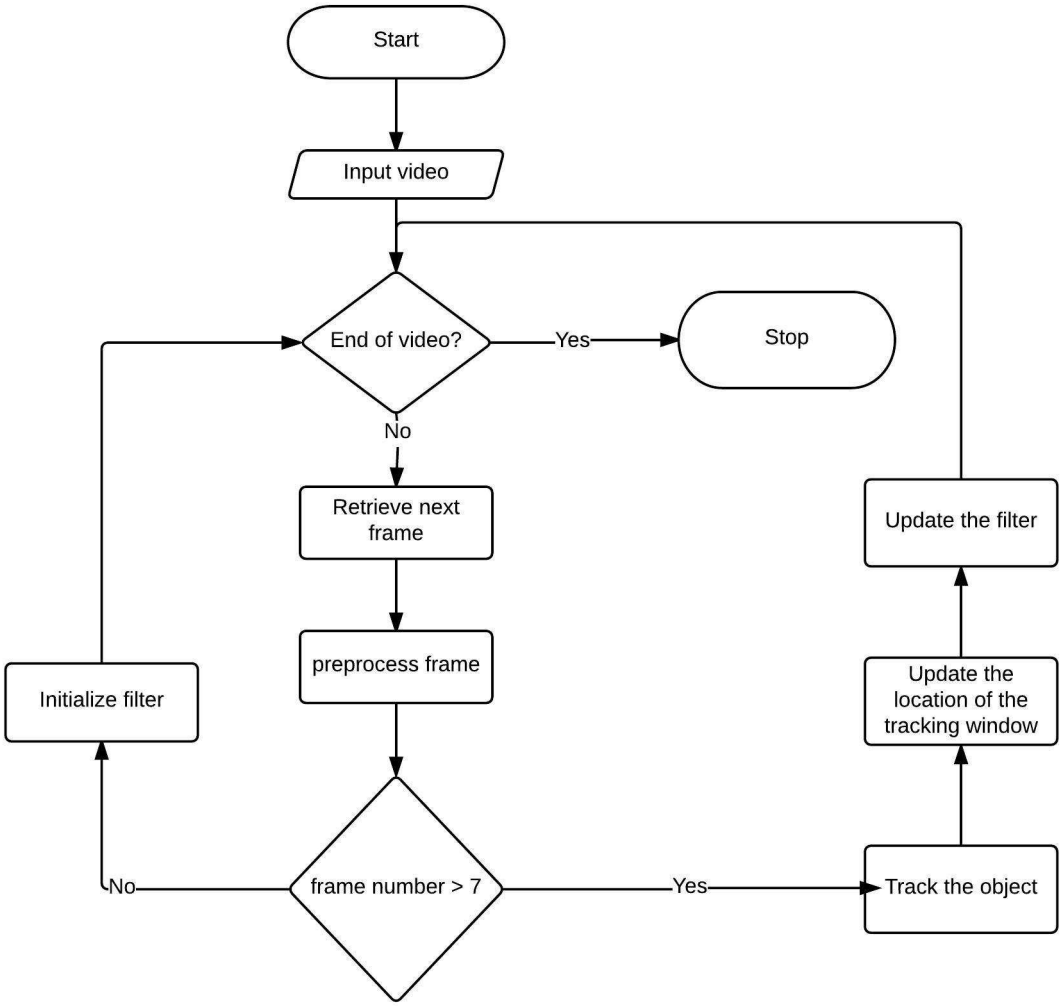



FIGURE 3.5. Flow chart for the MOSSE tracking algorithm with the filter update.

same throughout the tracking process. The new position of the object is the center of the new tracking window. The tracking window is preprocessed and converted to the Fourier domain, $F$. A new synthetic output in Fourier domain, $G$, is generated. The filter is updated by substituting the new values of $F$ and $G$ in Equations 9 and 10. The $N_{i-1}$ and $D_{i-1}$ are the weighted sums of all the previous frames. In Figure 3.5, the MOSSE tracking algorithm is illustrated.

The two examples presented for the basic MOSSE tracker are used to illustrate tracking with the filter update. A learning rate of 0.01 is used in Equations 9 and 10 for the soccer

ball and a learning rate of 0.1 is used for the girl playing soccer. With the filter update, the tracker tracks the ball for more frames as compared to the previous two examples. The tracker loses the ball only when it is occluded by another object. The tracker does not lose the girl until she is occluded by another girl.

### 3.5. MOSSE filter with affine transformations

In the MOSSE tracker, an option for initializing the filter is to use affine transformations of a single template to create more training samples. Affine transformations such as scale and rotation can be applied to the template obtained from the first frame of a video. The affine transformations are used to initialize the filter.. Due to slight variations of an object in consecutive frames, small affine transformations are applied.

#### 3.5.1. Affine transformation math

An affine transformation has 6 degrees of freedom. It includes scale, rotation, translation and shear. For the MOSSE tracker, scale, rotation and translation are used. The general affine transformations can be applied using the following equation:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{15}$$

In Equation 15, $x$ and $y$ represent the coordinates of the pixels on an image, $u$ and $v$ represent the new coordinates of the pixels after transformation. The values of $a$ through $f$ are used to perform different affine transformations.

The coordinates of an image can be translated using Equation 16. In Equation 16, $t_x$ and $t_y$ represent the number of pixels by which the x and y coordinates of an image are translated.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{16}$$

Equation 17 is used to change the scale of an image. $s_x$ and $s_y$ represent the magnitude by which the $x$ and $y$ coordinates of an image are scaled respectively. The origin for changing the scale of an image is the top left corner of the image.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{17}$$

Equation 18 is used to rotate an image around the top left corner of the image. The symbol $\theta$ represents the angle of rotation.

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{18}$$

To rotate an image around its center, rather than the top left corner, a combination of translation and rotation is performed. First, the image is translated to its center. The translated image is then rotated and then the rotated image is translated back to its original

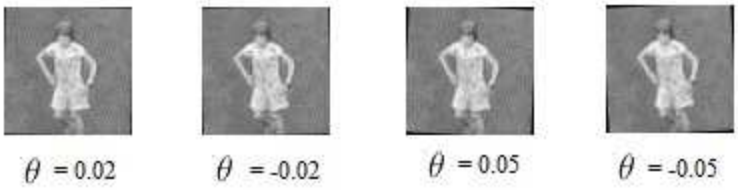



FIGURE 3.6. The input template with small degrees of rotation

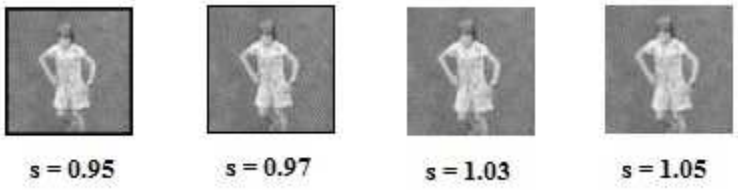



FIGURE 3.7. The input template with small changes in scale

position. The following equation is used for applying the combination:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos(\theta) & -\sin(\theta) & 0 \\ \sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{19}$$

In Equation 19, $t_x$ and $t_y$ is the coordinates of the center of an image. Similarly, to scale an image around its center, a combination of translation and scale needs to be performed. The following equation is used for a combination of translation and scale:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -t_x \\ 0 & 1 & -t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{20}$$

### 3.5.2. MOSSE tracker with affine changes

In this subsection, initializing the MOSSE filter with affine transformations is presented. The following steps are used for initializing the filter with affine transformations:

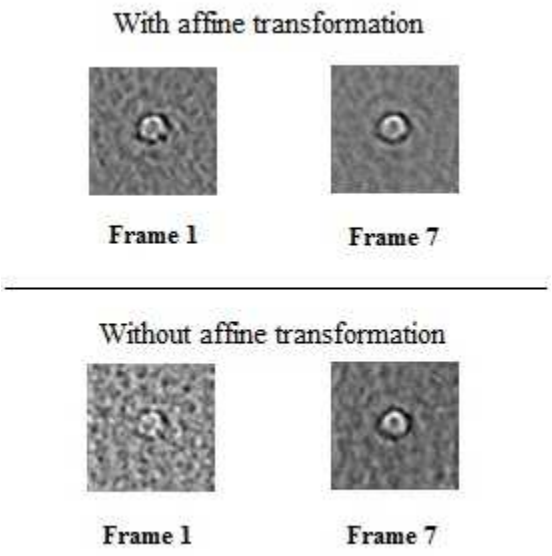



FIGURE 3.8. The MOSSE filter of the ball during initializing

(1) A template of $64 \times 64$ pixels is cropped after the user selects the center of an object to be tracked from the first frame of the video.

(2) Affine transformations are applied to the template. The template is scaled by 0.95, 0.97, 1.03, and 1.05 and rotated by 0.02, $-0.02$, 0.05, and $-0.05$ angles. All of the templates are rotated and scaled around the center of the object. Figure 3.7 and 3.6 illustrate these transformations.

(3) The transformed templates are preprocessed as described in Section 2.2.

(4) The preprocessed templates are converted to the Fourier domain.

(5) Equations 9, 10 and 11 are used to initialize the filter with the transformed and preprocessed templates.

The tracker along with affine transformations is initialized on two different objects for illustration purposes. The soccer ball is the first object. The girl playing with the soccer ball is the second object from the same video. Figure 3.8 illustrates the MOSSE filter for the soccer ball during the initializing. The top left figure illustrates the filter obtained

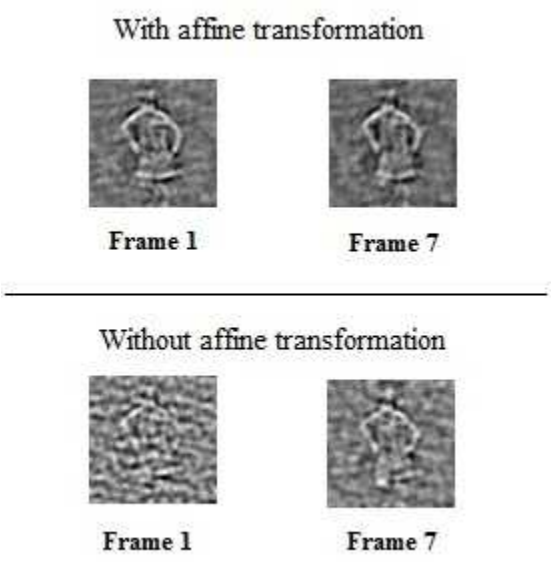



FIGURE 3.9. The MOSSE filter of the girl in beige color dress during initializing on the first frame of the video with affine transformations during initializing. The bottom left illustrates the filter without affine transformations. The top right and bottom right images illustrate the filter obtained at the seventh initializing frame with and without affine transformations respectively. Figure 3.9 illustrates the MOSSE filter obtained with and without affine transformations during the initializing for the girl playing soccer. The overall quality of the tracker improves with the inclusion of the affine transformations.

CHAPTER 4

# Experiments

Experiments are performed on five different objects to compare different versions of the MOSSE tracker presented in Chapter 3. The video of the children playing in the playground is used to perform experiments. The objects with increasing complexity are as follows:

Object 1: Soccer ball. The soccer ball is one of the simplest objects because its shape does not change even though there are changes in viewpoint of scale of the ball. Figure 4.1a illustrates the soccer ball with a template size of $64 \times 64$ pixels. Template sizes of $32 \times 32$, $64 \times 64$ and $128 \times 128$ pixels are used for different experiments.

Object 2: The girl in beige. The girl in a beige color dress playing with the soccer ball is the second object.The pose of the girl changes over a period of time in the video. She remains in clear view of the camera for more than $2,000$ frames, thus she is a good example to demonstrate the working of the tracker with changes in the human pose. Figure 4.1b illustrates the girl in the beige color dress with a template size of $64 \times 64$ pixels. Template sizes of $32 \times 32$, $64 \times 64$ and $128 \times 128$ pixels are used for different experiments.

(A) (B) (C) (D) (E)

Figure 4.1. The objects from the video with the children playing in the playground. Figure 4.1a Object 1: A $64 \times 64$ pixel template of a soccer ball. Figure 4.1b Object 2: A $64 \times 64$ pixel template of a girl playing with the soccer ball. Figure 4.1c Object 3: A $32 \times 32$ pixel template of a passing by car. Figure 4.1d Object 4: A $64 \times 64$ pixel template of a boy swinging on the swing. Figure 4.1e Object 5: A $64 \times 64$ pixel template of another boy swinging on a different swing.

Object 3: Cars. A number of cars pass by the playground in the video. One of the passing by cars is the next object. It is simple to track the car because the car's direction does not change in the video. However, a fence in front of the car increases the complexity of tracking the car. Figure 4.1c illustrates a template of $32 \times 32$ pixels, the only size used for the experiments. The original video is trimmed so that the car appears in the first frame of the video to initialize the filter.

Object 4: Boy in white. The boy in a white t-shirt on the swing is the next object. It is difficult to track him due to rapid changes in his pose, position, and scale because of the swinging action. He also gets partially occluded by the support beam of the swing. A template of $64 \times 64$ pixels is shown in Figure 4.1d and this is what was used for the experiments.

Object 5: Boy in red. The boy in the red t-shirt on the swing is the most difficult object to track because of the motion of the swing and occlusion by the person on the other swing. The example template of $64 \times 64$ pixels is shown in Figure 4.1e as used for the experiments.

The tracking of an object depends on the careful selection of the template (tracking point) during initialization of the filter. If the template is not centered properly, the tracker will not keep track of the object for as long as expected. For example, it is important to click at the center of the soccer ball while initializing the filter due the small size and swift movement of the ball in the video.

### 4.1. Experiments on Basic MOSSE Tracker

Experiments are performed on five different objects discussed above. A filter is initialized for each object to demonstrate the working of the basic MOSSE tracker. In this version of the tracker, the tracking window does not move, thereby fixing the location of the tracking

TABLE 4.1. Initialization parameters

| **Object** | **Template size (in pixels)** | **Learning rate** |
|---|---|---|
| *Soccer ball* | $64 \times 64$ | 0.0625 |
| *Girl in beige* | $64 \times 64$ | 0.0625 |
| *Car* | $32 \times 32$ | 0.0625 |
| *Boy in white* | $64 \times 64$ | 0.1 |
| *Boy in red* | $64 \times 64$ | 0.1 |

window for the rest of the video. The initial location of the tracking window is centered on the center of the object from the last frame of filter initialization. The results from the experiments are discussed below. The algorithm mentioned in Section 3.2 is used to initialize the basic MOSSE filter and track the objects for the experiments presented in this section.

The basic MOOSE tracking algorithm uses a learning rate to initialize the filter on the first seven frames of the video. Different learning rates are used for different objects. The learning rates to initialize the filter are consistent throughout the experiments performed on the different versions of the MOSSE tracker for an object. The learning rates were experimented with to find an optimum learning rate for an object. A learning rate of 0.0625 is used for the soccer ball, 0.0625 for the girl in beige, 0.0625 for the car, 0.1 for the boy in white and 0.1 for the boy in red. A template size of $64 \times 64$ pixels is used for the soccer ball, the girl in beige, the boy in white and the boy in red for the experiments in this section. A template of $32 \times 32$ pixels is used for the car in this section. Table 4.1 shows the template sizes and the learning rates used for different objects for the basic MOSSE tracking algorithm.

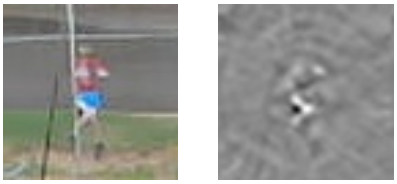
**Basic tracker Object 1:** The MOSSE filter is initialized on the first seven frames of the video and the filter is shown in Figure 4.2a. Figure 4.3 illustrates how the the basic MOSSE filter adapts during initialization. Once the filter is initialized, the tracker starts tracking the soccer ball. When the tracker starts tracking the ball, the ball is already rolling. The

(A) Basic tracker Object 1

(B) Basic tracker Object 2

(C) Basic tracker Object 3

(D) Basic tracker Object 4



(E) Basic tracker Object 5

FIGURE 4.2. The filters for the basic MOSSE algorithm initialized on the first seven frames of the video. Figure 4.2a: The MOSSE filter for the soccer ball. Figure 4.2b: The MOSSE filter for the girl in beige. Figure 4.2c: The MOSSE filter for the car. Figure 4.2d: The MOSSE filter for the boy in white. Figure 4.2e: The MOSSE filter for the boy in red.

ball moves from the center of the tracking window towards the edge of the tracking window and stops close to one of the edges of the tracking window for a moment. Then a girl comes forth and kicks the ball, rolling the ball in the opposite direction. The ball eventually moves

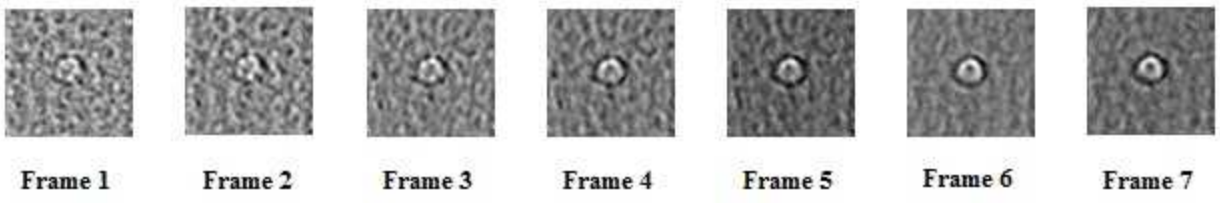



FIGURE 4.3. A basic MOSSE filter initialized on the first seven frames of the video of the children playing in the play ground is presented frame by frame as the filter adapts. The filter is initialized on the soccer ball.

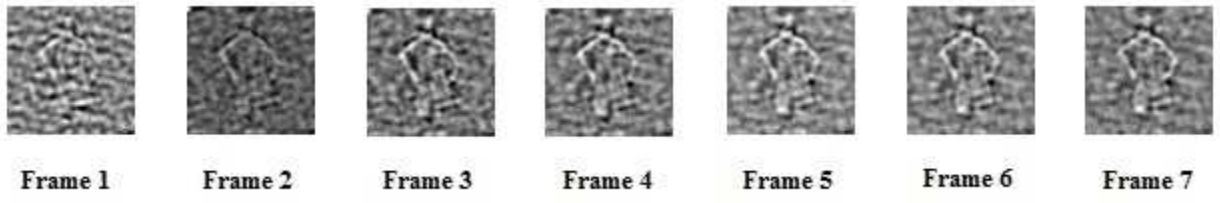



FIGURE 4.4. A basic MOSSE filter is initialized on the first seven frames of the video of the children playing in the play ground is presented frame by frame as the filter adapts. The girl playing with the soccer ball in a beige color dress is the object.

out of the tracking window around frame number 81 and the tracker stops tracking the ball in approximately 74 frames.

**Basic tracker Object 2:** The basic MOSSE filter is initialized on the girl playing with the soccer ball, wearing a beige color dress and facing the camera. The filter is initialized on the first seven frames of the video by selecting the center of the girl's torso as the center of the tracking window. Figure 4.2b illustrates the template and the corresponding initialized MOSSE filter. Figure 4.4 illustrates how the the basic MOSSE filter adapts on the first seven frames of the video. The location of the center of the tracking window is the location of the center of the torso of the girl. The tracker starts tracking the torso of the girl as soon as the filter is initialized. At the beginning of the video, the girl is waiting for the soccer ball to be kicked by another girl. While she is waiting for the soccer ball, her hands are on her waist. The tracker keeps tracking the girl while she maintains her initial pose for about

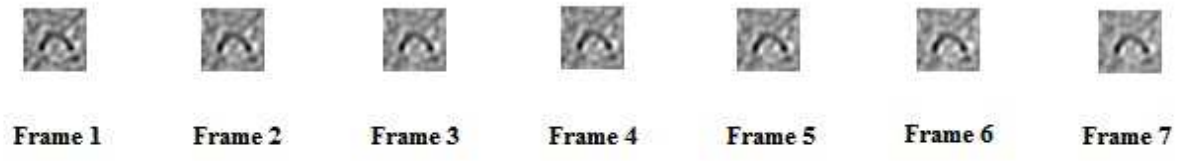



FIGURE 4.5. A basic MOSSE filter initialized on the first seven frames of a clip of the video of the children playing in the play ground is presented frame by frame as the filter adapts. A car passing by the playground is the object of interest.

115 frames. When the ball reaches the girl, her pose starts changing and she moves forward to kick the ball with her hands spread in the air and no longer on her waist. When there are significant changes in her pose, the tracker starts tracking her head instead of her torso. The tracker loses her when she moves out of the tracking window in 142 frames.

**Basic tracker Object 3:** The MOSSE filter is initialized on the front quarter panel and the front wheel of the SUV passing by on the road behind the playground. As soon as the filter is initialized in the first seven frames of the video, the tracker starts tracking the front quarter panel of the car. The tracker tracks the car as long as the car stays inside the tracking window. Due to a stationary tracking window of $32 \times 32$ pixels, the car's front panel moves out of the tracking window in 5 frames and the tracker starts tracking the side doors and then the rear wheel. The tracker stops tracking the car as soon as the car's rear wheel moves out of the tracking window. Figure 4.2c illustrates the basic MOSSE filter initialized on the first seven frames of the video. Figure 4.5 illustrates how the the basic MOSSE filter adapts when initialized on the first seven frames of the video. The tracker tracks the front wheel of the car for 5 frames. However, the tracker tracks the doors and the rear wheel of the car for another 18 frames.

**Basic tracker Object 4:** For this experiment, the boy wearing a white t-shirt and black shorts swinging on the swing is the object. The filter is initialized using the algorithm from

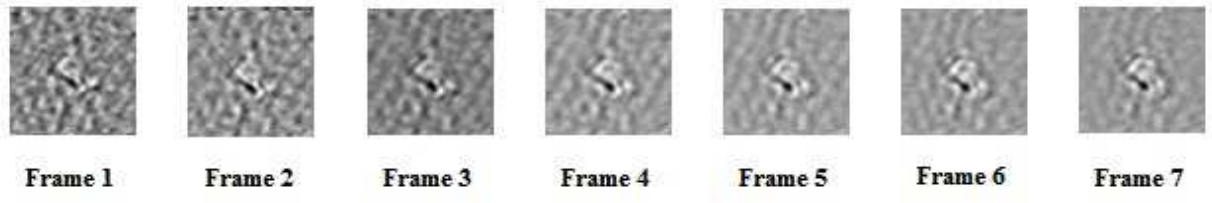



FIGURE 4.6. A basic MOSSE filter initialized on the first seven frames of the video of the children playing in the play ground is presented frame by frame as the filter adapts. A boy in white t-shirt swinging on the swing is the object.


FIGURE 4.7. A basic MOSSE filter initialized on the first seven frames of the video of the children playing in the play ground is presented frame by frame as the filter adapts. A boy in red t-shirt swinging on the swing is the object.

Section 3.2. Figure 4.2d illustrates the basic MOSSE filter initialized on the first seven frames of the video. Once the filter is initialized, the tracker tracks the boy for 10 frames and then loses him. As the swing moves away from the camera, the boy on the other swing besides him swings towards the camera and enters the stationary tracking window. The tracker starts tracking the boy on the other swing instead of the original object. As soon as the second boy moves out of the tracking window, the tracker sporadically tracks the original object. Figure 4.6 illustrates how the the basic MOSSE filter adapts when initialized on the first seven frames of the video.

**Basic tracker Object 5:** A boy wearing a red t-shirt and blue shorts swinging on the other swing is the next object for tracking. The filter is initialized on the first seven frames of the video. After initializing the filter, the tracker tracks the boy for 2 frames and then loses him. The tracker tracks the boy sporadically, and loses him more often than it tracks.

While moving towards the camera on the swing the boy mostly stays inside the tracking window, but his pose changes dramatically. Hence, the tracker only tracks him when his pose is similar to his original pose. Figure 4.2e illustrates the basic MOSSE filter initialized on the first seven frames of the video. Figure 4.7 illustrates how the the basic MOSSE filter adapts when initialized on the first seven frames of the video.

The basic MOSSE tracker tracks the symmetric objects as long as they stay inside the tracking window. The basic MOSSE tracker does not perform well for tracking the human objects as their pose changes with movement. The next section discusses how updating the location of the tracking window during tracking helps in tracking the object better.

## 4.2. Experiments on MOSSE tracker with update in tracking window

Experiments are performed on the five objects from the video of the children playing in the playground. In this section, experiments are performed to demonstrate the affects of updating the location of the tracking window as the tracker tracks. The algorithm described in Section 3.3 is used for initializing the filter and tracking the object for all of the experiments presented in this section. The algorithm uses the learning rate and template size as mentioned in Table 4.1.

**Moving window tracker Object 1:** The MOSSE filter is initialized on the first seven frames of the video with the soccer ball as the object of interest. Once the filter is initialized, the tracker starts tracking the soccer ball. This version of the tracker uses a moving tracking window. When the tracker starts tracking, the ball is moving towards a girl playing with the soccer ball and facing away from the camera. The girl comes forth and kicks the ball, the ball starts rolling in the opposite direction. The ball starts moving towards the second girl playing with the soccer ball and facing the camera. Because the tracking window is

not stationary and its location updates while tracking, the tracker tracks the ball for longer period of time than earlier. When the second girl approaches the ball to kick it, the tracker loses the ball around frame number 108 and starts tracking the girl's skirt. In this frame, there is a clear change in the ball's scale from the first seven frames because it has moved further away from the camera.

**Moving window tracker Object 2:** The girl playing with the soccer ball in a beige color dress is the object for this tracking experiment. The tracker starts tracking the girl as soon as the filter is initialized. As mentioned in the previous section, the girl's hands are on her waist during initialization of the filter. As the soccer ball approaches the girl, her pose starts changing and she moves towards the ball. Her hands start moving and are spread out in the air by the time she starts kicking the ball. The tracker tracks the girl as long as there is not much change in her pose. As soon as her pose is different from the original pose around frame number 115, the tracker starts tracking her limbs or the side of the torso instead of the center of the torso. The tracker returns back to the center of the torso when her pose is similar to the original pose. The tracker loses the girl in about 1018 frames when she moves out of the tracking window.

**Moving window tracker Object 3:** The MOSSE filter is initialized on the front quarter panel and the front wheel of the SUV passing by on the road behind the playground. The filter is initialized on first seven frames of the video on a tracking window of $32 \times 32$ pixels. As soon as the filter is initialized, the tracker starts tracking the front quarter panel of the car. Because the tracking window is not stationary, the tracker loses the front panel of the car in about $4 - 5$ frames, hence losing the car.

**Moving window tracker Object 4:** The boy wearing a white t-shirt and black shorts, swinging on the swing is the next object for tracking. The filter is initialized on the first seven

frames of the video. The tracker tracks the boy for about 10 frames after initialization and starts losing the object when there are dramatic changes in his pose. The tracker sometimes tracks the other boy on the swing when he too is in the tracking window. Since the tracking window is not stationary and its location changes with the change in the object's location, the tracker starts tracking other objects which look similar to the original pose of the boy and the tracker loses the boy.

**Moving window tracker Object 5:** The boy wearing a red t-shirt and blue shorts, swinging on the swing is the next object for the tracking experiment. The boy is initialized on the first seven frames of the video. Once the tracker starts tracking the object, the tracker tracks the boy for 2 frames and then starts losing him. Because the tracking window is not stationary and its location is updated with the new location of the object, the tracker loses the boy due to dramatic changes in his pose. Once the tracker loses the boy, the tracking window moves and does not return.

The MOSSE tracker with an update in the location of the tracking window works better for the rigid objects such as the ball. It did not show much improvement in tracking human objects and the tracker loses the human objects due to the drastic changes in the human pose.

### 4.3. Size vs Speed vs Background

The size of the template and the tracking window has to be same to perform a multiplication in the Fourier domain. Element wise multiplication is performed between the filter and the tracking window to find the new location of the object in the new frame of the video during tracking. A template image is used to initialize the filter, therefore the size of the filter is the same as the size of the template used to initialize the filter.

(A)

(B)

(C)

(D)

(E)

(F)

(G) Soccer ball

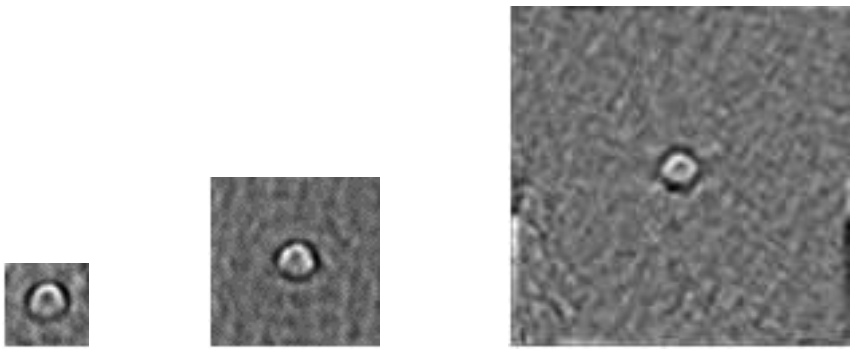
(H) The girl in beige

(I) The lady in blue

FIGURE 4.8. The templates from the video with the children playing in the playground. Figures 4.8a, 4.8b and 4.8c are the $32 \times 32$ pixel size templates of the soccer ball, the girl in beige and the lady in blue respectively. Figures 4.8d, 4.8e and 4.8f are the $64 \times 64$ pixel size templates of the soccer ball, the girl in beige and the lady in blue respectively. Figures 4.8g, 4.8h and 4.8i are the $128 \times 128$ pixel size templates of the soccer ball, the girl in beige and the lady in blue respectively.

Experiments are performed on three different objects with three different window sizes to achieve an optimum template and tracking window size. The three sizes used for the template and the tracking window are: $32 \times 32$, $64 \times 64$ and $128 \times 128$ pixels. The first object is the soccer ball, the second object is the girl playing with the soccer ball in a beige color dress and the third object is the lady in a blue color top[1] near the jungle gym. The MOSSE tracker with moving tracking window is used to perform the experiments. The

[1]Note: This person is not in the original data set of the objects.



(A) (B) (C)

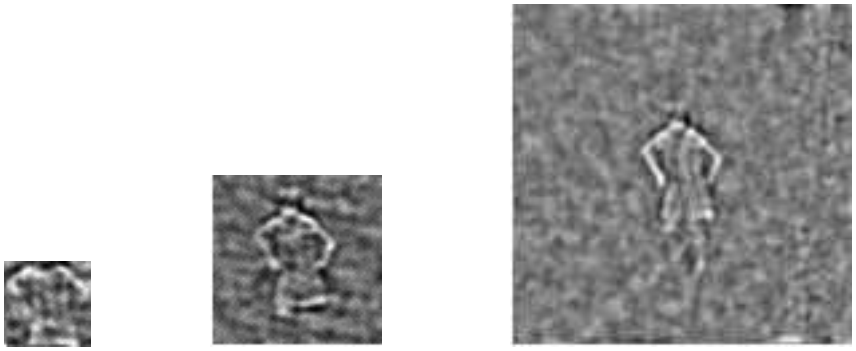
FIGURE 4.9. The filters for the soccer ball is initialized over the first seven frames of the video. Figure 4.9a Size 32: A filter of $32 \times 32$ pixel size. Figure 4.9b Size 64: A filter of $64 \times 64$ pixel size. Figure 4.9c Size 128: A filter of $128 \times 128$ pixel size.

$32 \times 32$ pixel size templates of the soccer ball, the girl in beige and the lady in blue are shown in Figures 4.8a, 4.8b and 4.8c respectively. Figures 4.8d, 4.8e and 4.8f illustrate the $64 \times 64$ pixel size templates of the soccer ball, the girl in beige and the lady in blue respectively. The $128 \times 128$ pixel size templates of the soccer ball, the girl in beige and the lady in blue are shown in Figures 4.8g, 4.8h and 4.8i respectively.

### 4.3.1. Object 1: Soccer ball

The soccer ball was initialized on the first seven frames of the video with a learning rate of 0.01. Three different template sizes are used to initialize the filter. The following are the results for the three different experiments:

**Size 32**: A template and a tracking window of $32 \times 32$ pixels is used. The tracker tracks the rolling soccer ball until the ball is kicked for the first time in the video and loses the ball in 69 frames. The tracker loses the ball when it starts moving faster after being kicked for the first time. The tracking window with a size of $32 \times 32$ pixels is not big enough to track a fast moving object. Figure 4.9a illustrates a $32 \times 32$ pixel MOSSE filter initialized over first seven frames of the video.



(A) (B) (C)

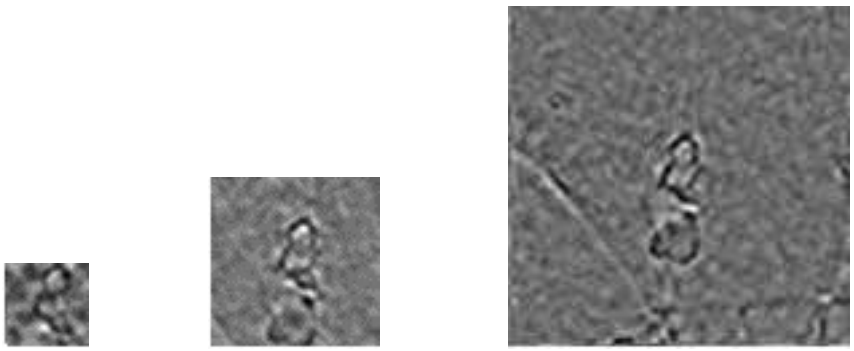
FIGURE 4.10. The filters for the girl in a beige color dress is initialized over the first seven frames of the video. Figure 4.10a Size 32: A filter of $32 \times 32$ pixel size. Figure 4.10b Size 64: A filter of $64 \times 64$ pixel size. Figure 4.10c Size 128: A filter of $128 \times 128$ pixel size.

**Size 64**: When the filter is initialized using a template window of $64 \times 64$ pixels, the tracker keeps tracking the ball till it is kicked for the second time. When the ball is kicked for the second time, the tracker loses the ball and stops tracking it. Figure 4.9b illustrates a $64 \times 64$ pixel MOSSE filter initialized over first seven frames of the video. The tracker loses the ball in 108 frames.

**Size 128**: In this experiment, the filter is initialized on a template of $128 \times 128$ pixels. The tracker does not track the ball at all because there is a lot of background information in the template as well as the filter. The filter also captured a part of one of the girls playing soccer. Figure 4.9c illustrates a $128 \times 128$ pixel MOSSE filter initialized over first seven frames of the video.

### 4.3.2. Object 2: Girl in beige

The girl in a beige color dress playing with the soccer ball is the second object. The girl was initialized on the first seven frames of the video with a learning rate of 0.08. The following three experiments were performed:



(A) (B) (C)

FIGURE 4.11. The filters for the lady in blue color top near the jungle gym is initialized over the first seven frames of the video. Figure 4.11a Size 32: A filter of $32 \times 32$ pixel size. Figure 4.11b Size 64: A filter of $64 \times 64$ pixel size. Figure 4.11c Size 128: A filter of $128 \times 128$ pixel size.

**Size 32**: In this experiment, a template and a tracking window of $32 \times 32$ pixels was used. The filter was initialized on the torso of the girl. The tracker keeps tracking the girl as long as she does not move too much from her location. The tracker loses the girl when the girl's pose starts changing in 100 frames. Figure 4.10a illustrates a $32 \times 32$ pixel MOSSE filter initialized over the first seven frames of the video.

**Size 64**: When the filter is initialized on a template of $64 \times 64$ pixels then the tracker tracks the girl until her pose starts changing, and then loses her. Figure 4.10b illustrates a $64 \times 64$ pixel MOSSE filter initialized over the first seven frames of the video. The tracker loses the girl in 1018 frames.

**Size 128**: When a template and a tracking window of $128 \times 128$ pixels is used then the tracker loses the girl in about 23 frames. There is a lot of background information in the filter because the area of the background is much larger than the area of the girl in the template. Figure 4.10c illustrates a $128 \times 128$ pixel MOSSE filter initialized over first seven frames of the video.

### 4.3.3. Object 3: Lady in blue

The third object is the lady in the blue colored top. The filter is again initialized on the first seven frames of the video with a learning rate of 0.08. The filter is initialized on three different template sizes.

**Size 32**: The lady does not move as fast as the soccer ball, therefore the tracker does not lose her until she is occluded by the jungle gym when initialized on a template of $32 \times 32$ pixels size. Figure 4.11a illustrates a $32 \times 32$ pixel MOSSE filter initialized over the first seven frames of the video.

**Size 64**: The tracker starts losing the lady before it loses her in the Size 32 experiment. The tracker loses her when she is partially occluded because of too much background information in the filter. Figure 4.11b illustrates a $64 \times 64$ pixel MOSSE filter initialized over the first seven frames of the video.

**Size 128**: The tracker loses the lady as soon because she starts moving away from her location as there is a lot of background information in filter. Figure 4.11c illustrates a $128 \times 128$ pixel MOSSE filter initialized over the first seven frames of the video.

With all three objects, the tracker keeps tracking the objects with $32 \times 32$ pixels tracking window as long as they do not move fast. If the object starts moving fast, the tracker loses the object. The tracker tracks the objects best with $64 \times 64$ pixel size tracking window. The tracker does not track most of the objects with a tracking window size of $128 \times 128$ pixels because there is more background information than foreground information.

The size of the template determines the amount of background information in the filter during initialization. The size of the tracking window determines how much area will the tracker search for the object. A very small tracking window is not good to track fast moving objects. A template much larger than the object has significant amounts of background

information. Therefore, a big tracking window will not work for objects small in size. The tracking window size is dependent on the size of the object. If an object is small in size a smaller tracking window is good for tracking. If an object is big in size a bigger tracking window is good for tracking.

### 4.4. MOSSE tracker with filter update

In this section, experiments are performed to discuss the performance of the MOSSE tracker with the filter update during tracking. The filter is updated with the new templates acquired during tracking of the object. The algorithm described in Section 3.4 is used to initialize the filter and track the object for all the experiments presented in this section. In this section, a moving tracking window is used. The learning rates and the template sizes mentioned in Table 4.1 are used for the experiments.

**Filter update Object 1**: The first experiment is performed on the soccer ball. Once the filter is initialized on the first seven frames of the video, the tracker starts tracking the ball. As the tracker tracks the ball, the filter is updated with the new template of the ball from the new location of the ball. The learning rate used to update the filter while tracking is 0.01. The tracker keeps tracking the ball even when it is kicked multiple times. The ball gets occluded by one of the girl's playing with the ball around frame number 206 and the tracker stops tracking the ball.

Another experiment is performed on the soccer ball with the same parameters as the previous experiment except that the filter is initialized only on the first frame of the video. As the ball is moving and the filter is initialized only on one frame of the video, the tracker does not track the ball at all. This experiment shows how the tracker tracks when initialized


FIGURE 4.12. Demonstrating the differences in the pose of the girl and the corresponding filters. Five different frames are used to show the changes in the girl's pose.

only on the first frame of the video as compared to the filter initialized on the first seven frames of the video.

**Filter update Object 2**: The girl playing with soccer ball in a beige color dress is the object for the second experiment with the filter update. The filter is initialized for the first seven frames of the video with the center of the torso of the girl as the center of the template window. After the filter is initialized, the tracker starts tracking and the filter is updated with the latest template. A learning rate of 0.0625 is used to update the filter during tracking. The tracker keeps tracking the girl even when there are dramatic changes in her pose. There are many variations in the girl's pose. During initialization, the girl is standing with her hands on the waist. After a while she is kicking the ball and running around to reach to the ball. The tracker tracks the girl for approximately 2289 frames! The 4.12 illustrates the object and the corresponding updated filter at that frame. The object's pose and the corresponding filter for frame 1, 50, 100, 150, 200, 250 and 300 are illustrated.

Another experiment is performed by changing the learning rate for the filter update to 0.01. All the other parameters remain the same as the previous experiment with the girl as the object. The tracker tracks the girl for 2419 frames! Around the frame number 2426, the girl gets partially occluded by another girl. At this point, the tracker starts tracking the second girl instead of the original object. Although the filter adapts slower as compared to the previous experiment, the tracker tracks the girl for a longer period of time. The two experiments presented here with different leaning rates to update the tracker illustrate how the leaning rate effects the tracking of the object. The tracker loses the girl if the learning rate is higher because the filter captures a lot of background information, however with a lower learning rate the filter does not capture as much of the background information.

**Filter update Object 3**: The MOSSE filter is initialized on the front quarter panel and the front wheel of the SUV passing by on the road behind the playground. A learning rate of 0.01 is used to update the filter during tracking. The tracker tracks the car's front quarter panel for 85 frames and loses the car when the car is occluded behind the car parked by the side of the road.

**Filter update Object 4**: The boy wearing a white t-shirt and black shorts, swinging on the swing is the next object for the experiment with filter update. The filter is initialized on the first seven frames of the video. The learning rate for the filter update after the tracker starts tracking is 0.0625. The tracker starts tracking the boy after the filter is initialized considering the center of the template as the center of the torso of the boy. The tracker keeps tracking the boy even when he is slightly hidden behind the other boy on the swing. The tracker tracks the boy for 487 frames. The tracker loses the boy when occluded partially behind the other boy on the swing.

When an experiment is performed using the same parameters as the previous experiment while using only the first frame of the video to initialize the filter, the tracker loses the boy faster than the previous experiment. When the other boy swinging on the swing comes in front of the object, the tracker starts tracking the second boy and loses the original object in about 70 frames.

**Filter update Object 5**: The boy wearing a red t-shirt and blue shorts, swinging on the swing is the next object for the experiment. The boy is initialized on the first seven frames of the video. The filter is initialized with the center of the template as the center of the torso of the boy. The learning rate for the filter update after the tracker starts tracking is 0.0625. Despite of the dramatic changes in the pose of the boy on the swing, the tracker tracks the boy for about 2 frames. The tracker loses the boy when he is partially hidden behind the leg of the swing set. When another experiment is performed using the same parameters as the previous experiment but the filter is initialized on just the first frame of the video, the tracker does not track the boy at all.

Updating the filter works better than the previous version of the tracker because the filter adapts to the changes in the object's appearance over a period of time. This version of the tracker is not too dependent on the changes in the appearance of an object. However, when the filter is initialized on just the first frame of the video then the tracker does not do a good job of tracking the object. The experiments with initializing the filter only on the first frame of the video will help in comparing the experiments from the next version of the tracker.

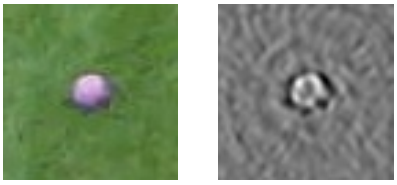
## 4.5. MOSSE tracker with affine transformations

In this section, the MOSSE tracker with the use of the affine transformations to initialize the filter is discussed. The filter is initialized only on the first frame of the video instead of the first seven frames of the video. The algorithm described in Section 3.5.2 is used to initialize the filter and track the object for all the experiments presented in this section. In this algorithm the tracking window as well as the filter is updated during tracking. The learning rates used to initialize the filters are mentioned in Table 4.1 along with the template sizes.

**Affine transformations Object 1**: The filter is initialized on the first frame of the video with the soccer ball as the object. The learning rate for the filter update while tracking is 0.01. After initializing, the tracker keeps tracking the ball as long as it does not get occluded by one of the girls playing pass with the ball. However, when the filter is initialized using just the first frame of the video for the previous version of the tracker, the tracker does not track the object at all. Figure 4.13a illustrates the filter initialization with affine transformations. The tracker tracks the soccer ball for 206 frames.

**Affine transformations Object 2**: In this experiment, the girl in a beige color dress playing soccer is the object. The learning rate for updating the filter while tracking is 0.01. The tracker keeps tracking the girl while she is playing with the soccer ball and does not lose her as long as she is not occluded by another girl. Around frame number 2426, the girl is partially occluded and that is when the tracker stops tracking the original girl. Figure 4.13b illustrates the filter initialization with affine transformations. The tracker tracks the girl in beige for 2419 frames.

**Affine transformations Object 3**: For this experiment, the MOSSE filter is initialized on first frame of the video of the SUV passing by on the road behind the playground. A



(A) Affine transformations Object 1

(B) Affine transformations Object 2

(C) Affine transformations Object 3

(D) Affine transformations Object 4

(E) Affine transformations Object 5

FIGURE 4.13. The MOSSE filters with affine transformations initialized on the first frame of the video. Figure 4.13a: The MOSSE filter for the soccer ball. Figure 4.13b: The MOSSE filter for the girl in beige. Figure 4.13c: The MOSSE filter for the car. Figure 4.13d: The MOSSE filter for the boy in white. Figure 4.13e: The MOSSE filter for the boy in red.

learning rate of 0.01 is used to update the filter during tracking. The tracker tracks the car's front quarter panel for 85 frames and loses the car when the car is occluded behind the car parked by the side of the road. Figure 4.13c shows the filter for the car with affine transformations.

TABLE 4.2. Summary: The number of tracked frames for different objects using different versions of the algorithm.

| | *Object 1: Soccer ball* | *Object 2: Girl in beige* | *Object 3: Car* | *Object 4: Boy in white* | *Object 5: Boy in red* |
|---|---|---|---|---|---|
| *Algorithm 1: Basic MOSSE* | 74 | 142 | 5 | 10 | 2 |
| *Algorithm 2: Tracking window update* | 108 | 1018 | 5 | 10 | 2 |
| *Algorithm 3: Filter update* | 206 | 2419 | 85 | 487 | 2 |
| *Algorithm 4: Affine transformations* | 206 | 2419 | 85 | 487 | 2 |

**Affine transformations Object 4**: The boy wearing a white t-shirt and black shorts, swinging on the swing is the next object for the experiment with affine transformations. The learning rate for the filter update after the tracker starts tracking is 0.0625. The tracker starts tracking the boy after the filter is initialized considering the center of the template as the center of the torso of the boy. The tracker keeps tracking the boy even when he is partially hidden behind the second boy on the swing. The tracker tracks the boy while the boy is swinging back and forth. The tracker loses the boy in 487 frame due to partial occlusion. Figure 4.13d shows the filter with affine transformations.

**Affine transformations Object 5**: The boy wearing a red t-shirt and blue shorts, swinging on the swing is the next object for the experiment with affine transformations. The filter is initialized with the center of the template as the center of the torso of the boy. The learning rate for the filter update after the tracker starts tracking is 0.0625. The tracker tracks the boy for 2 frames and then the tracker loses the boy and does not track him again. Figure 4.13e illustrates the filter initialization with affine transformations.

## 4.6. Summary

The table 4.2 shows the number of frames the tracker tracks for different objects using the different versions of the MOSSE tracking algorithm. In the basic MOSSE tracking algorithm, the tracker loses most of the objects earlier than other versions. Because the objects are in motion and they move out of the tracking window with time, the tracker cannot track an object. Therefore, it is very important to have a moving tracking window to track objects in a video. A stationary tracking window does not go far with tracking as the main goal of a tracking algorithm is to be able to track any moving object in a video.

The MOSSE tracker with an update in the location of the tracking window works better than the basic version. It tracks the object as long as there are not many changes in the object. It did not show much improvement in tracking human objects with dramatic pose changes. The tracker loses the humans with drastic changes in the pose earlier than the next version of the tracker. Therefore, the algorithm with the filter update during tracking is presented. Due to filter update, the tracker adapts to the changes in the objects over time and does not lose the objects easily. The tracker is still unable to track the boy in red due to his rapid movement and drastic changes in his pose.

Next an algorithm with affine transformations is presented. The performance of this version is very similar to the version with filter update. However, this version is very useful as the filter can be initialized on just the first frame of the video; hence making it easy to perform tracking on a real time video feed.

CHAPTER 5

# Conclusion

A tutorial on building and implementing different versions of the MOSSE tracker has been presented. A number of experiments were performed to illustrate the performance of the four versions of the tracker using templates of different objects. The first tracker was the basic MOSSE tracker. In the basic MOSSE tracker the tracking window does not move therefore once the object moves out of the tracking window, the tracker loses the object and is usually unable to retrack it. The basic MOSSE tracker is good for tracking symmetric rigid objects as long as they stay inside the tracking window. The basic tracker is not good for tracking humans and other similar objects.

Next, a MOSSE tracker which updates the tracking window as it tracks an object was presented. As the tracking window in this version of the tracker is not stationary, the tracker tracks the symmetric rigid objects for larger number of frames than the basic MOSSE tracker. Even though the position of the tracking window is updated, the tracker is not very good at tracking the humans as their pose changes based on the activity. In some activities there are dramatic changes in the pose of a human, for example, a person swinging on a swing. However, there are less dramatic changes if a person is running and kicking a ball.

In the next version of the MOSSE tracker, the filter is updated as the tracker tracks and the filter adapts to the changing appearance of the object. Therefore, the tracker performs much better on human objects as compared to previous versions of the MOSSE tracker. However, the filter is required to be initialized on more than one frame. The lesser the number of frames required to initialize the filter, the better the tracking algorithm.

The last version of the tracker uses affine transformations for initializing the filter. Only the first frame of the video is used to initialize the filter. When just the first frame of the video is used to initialize the filter for the previous version of the tracker, the tracker does not perform well. Therefore, to be able to initialize the filter on the first frame of the video, the MOSSE tracker with affine transformations works best as compared to the other versions. The tracker fails when there is an occlusion and starts tracking an object that closely resembles the original object.

The motivation for this work was to explore, understand and re-implement a state-of-the-art tracking approach. In order to understand the implementation of the MOSSE filter and the tracking algorithm, the algorithm was carefully evaluated. The algorithm was broken down into many simple parts. The first step towards re-implementing was to perform step by step preprocessing and verify the results of the code at every step.

Since the major computations in MOSSE are performed in the Fourier domain, Discrete Fourier Transformation is performed on the template as well as the synthetic output. It is also required to transform the tracking window to the Fourier domain during tracking and transform the result back to the spatial domain. Therefore, one of the major implementation difficulties was to verify the accuracy of the transformations from the spatial to the Fourier domain and vice-a-versa. Due to the visualization limitations in the Fourier domain, it is very important to ensure the correctness of the computations in the Fourier domain during programming, hence making debugging a difficult task.

After making sure that the frames were being preprocessed correctly, the exact filter was implemented and then the ASEF leading to the MOSSE filter. Once the MOSSE filter was implemented correctly, the basic version of the tracker was built and then the successive

versions of the tracker were build. The difficult parts were to implement the MOSSE filter, basic version of the tracker and then updating the filter during tracking.

## 5.1. Future work

Re-implementation of more versions of the tracker can be performed. David Bolme suggests a technique for occlusion detection and object recovery. If the value of the correlation peak is lower than a calculated threshold, the tracker suspends the filter update indicating that the original object is occluded by another object. However, the tracker keeps looking for the object in the consecutive frames. If the object reappears the tracker starts updating the filter again.

Another functionality which could be helpful is giving the user freedom to be able to select a template and tracking window size to match an object of interest. In the current implementation, the ability to change the size of the tracking window is limited. The window size can be changed, however the width and the height of the window should be same and is limited to the power of $2^n$ pixels.

# Bibliography


[1] A. Adam, E. Rivlin, and I. Shimshoni, "Robust fragments-based tracking using the integral histogram," in *Computer vision and pattern recognition, 2006 IEEE Computer Society Conference on*, vol. 1, pp. 798–805, IEEE, 2006.

[2] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008.

[3] S. Hare, A. Saffari, and P. H. Torr, "Struck: Structured output tracking with kernels," in *Computer Vision (ICCV), 2011 IEEE International Conference on*, pp. 263–270, IEEE, 2011.

[4] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," *International Journal of Computer Vision*, vol. 111, no. 2, pp. 213–228, 2015.

[5] X. Mei, H. Ling, Y. Wu, E. Blasch, and L. Bai, "Minimum error bounded efficient 1 tracker with occlusion detection," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pp. 1257–1264, IEEE, 2011.

[6] B. Babenko, M.-H. Yang, and S. Belongie, "Visual tracking with online multiple instance learning," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 983–990, IEEE, 2009.

[7] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pp. 2544–2550, IEEE, 2010.

[8] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of computation*, vol. 19, no. 90, pp. 297–301, 1965.

[9] E. W. Weisstein, "Discrete fourier transform," 2002.

[10] D. S. Bolme, "Theory and applications of optimized correlation output filters," 2007.

[11] D. S. Bolme, B. A. Draper, and J. R. Beveridge, "Average of synthetic exact filters," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pp. 2105–2112, IEEE, 2009.