Moving Object Detection Using Unsupervised and Weakly Supervised Neural Networks in Videos with Illumination Changes and Dynamic Background

by

Fateme Bahri

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

University of Alberta

Abstract

Background subtraction is a crucial task in computer vision applications, such as video surveillance, traffic monitoring, autonomous navigation, and human-computer interaction. This approach involves acquiring a background model to separate moving objects and the background from an input image. However, challenges such as sudden and gradual illumination changes and dynamic backgrounds can make this task difficult. Among various methods proposed for background subtraction, supervised deep learning-based techniques are currently considered state-of-the-art. However, these methods require pixel-wise ground-truth labeling, which can be time-consuming and expensive. The aim of this thesis is to develop unsupervised and weakly supervised background subtraction methods that can handle illumination changes or dynamic backgrounds.

Most methods handle illumination changes and shadows in batch mode and are unsuitable for long video sequences or real-time applications. To address this, we propose an extension of a state-of-the-art batch Moving Object Detection (MOD) method, ILISD, to an online/incremental MOD method using unsupervised and generative neural networks. Our method uses illumination invariant image representations and obtains a low-dimensional representation of the background image using a neural network. It then decomposes the foreground image into illumination changes and moving objects.

Yet another challenge is dynamic background, where a background pixel can have different values due to periodical or irregular movements, negatively affecting a method's performance. For example, surging of water, water fountains and waving trees cause dynamic variations in the background. To address this, we propose a new unsupervised method, called DBSGen, which estimates a dense dynamic motion map using a generative multi-resolution convolutional network and warps the input images by the obtained motion map. Then, a generative fully connected network generates background images using the warped input images in its reconstruction loss term. Finally, a pixel-wise distance threshold that utilizes a dynamic entropy map obtains the binary segmented results.

Finally, we propose a weakly supervised background subtraction method, where the training set consists of a moving object-free sequence of images, without requiring per-pixel ground-truth annotations. Our method consists of two neural networks. The first network, an autoencoder, generates dynamic background images for training the second network. Dynamic background images are obtained by applying a threshold to background-subtracted images. The second network is a U-Net that uses as input the same moving object-free video and is trained by using the dynamic background images produced by the autoencoder for its target output. During the testing phase, the autoencoder and U-Net process input images to generate background and dynamic background images, respectively. The dynamic background image helps remove dynamic motion from the background-subtracted image, resulting in a foreground image that is free of dynamic artifacts.

Preface

This thesis is an original work by Fateme Bahri. The chapters included in this thesis are based on the following papers:

- **Chapter 3: F. Bahri**, M. Shakeri, and N. Ray, "Online illumination invariant moving object detection by generative neural network," in Proceedings of the 11th Indian Conference on Computer Vision, Graphics and Image Processing, 2018, pp. 1–8.
- **Chapter 4: F. Bahri** and N. Ray, "Dynamic background subtraction by generative neural networks," in 2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS), IEEE, 2022, pp. 1–8.
- **Chapter 5: F. Bahri** and N. Ray, "Weakly supervised realtime dynamic background subtraction," arXiv preprint arXiv:2303.02857, 2023.

I dedicate it to my wonderful parents, Shahla and Mahmood, whose unconditional love and support have been the bedrock of my life. I also dedicate my thesis to Mohammad who has a unique way of expressing his love and support. Last but not least, I dedicate my thesis to Ryan, my dear son, who is my sunshine, my only sunshine.

He who has a why to live for can bear almost any how.

– Friedrich Nietzsche

Acknowledgements

I would like to express my sincere appreciation to Professor Nilanjan Ray for his exceptional support and supervision throughout the course of this research project. Without his invaluable guidance, this research and dissertation would not have been possible.

Furthermore, I would like to extend my gratitude to Dr. Pierre Boulanger and Dr. Kumaradevan Punithakumar, who served as my supervisory committee members, as well as Dr. Irene Cheng and Dr. Vahid Partovi Nia, my examiners, for generously dedicating their time to review my thesis and provide insightful feedback that enhanced its technical quality.

Additionally, I would like to convey my thanks to the University of Alberta for awarding me the University of Alberta Doctoral Recruitment Scholarship.

Last but not least, I am deeply grateful for my parents who have been a constant source of encouragement and support in my life. I am thankful for their unwavering belief in my abilities and for inspiring me to follow my dreams.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Moving Object Detection

Moving Object Detection (MOD) is a fundamental task in computer vision that involves identifying and separating non-stationary objects from a sequence of video frames. The objective of MOD is to extract moving objects while minimizing interference from background and noise.

MOD has numerous applications, including video surveillance and security in airports, residential areas, and shopping malls [15], [88], traffic monitoring [21], [78], object tracking [38], [76], [84], human activity recognition [18], [85], gesture recognition in human-machine interaction [90], content-based video coding [106], background substitution [42] and visual observation of animals [35], [43].

There are three main approaches used for moving object detection: background subtraction, temporal differencing, and optical flow [79]. While Optical flow and temporal differencing handle videos taken by a moving camera, background subtraction is well suited for videos of a static camera. We will discuss each of these approaches in detail in the 1.1.1 section. In this thesis, we propose methods based on background subtraction, as we consider videos captured by static cameras.

Moving object detection face several challenges, including illumination changes, shadows, dynamic background, challenging weather, camera jitter, bootstrapping, camouflage, occlusion, and intermittent object motion [79]. We describe these challenges in Section 2.1. In this thesis, one of our proposed methods addresses gradual and sudden illumination changes and shadows, while the other two methods focus on handling scenes with dynamic background.

1.1.1 Main Approaches for Moving Object Detection

In the following sections, the main approaches for moving object detection are explained.

Background Subtraction

When dealing with videos captured by a static camera, a popular approach is to use the technique of background subtraction. This approach involves comparing the current image with a reference background image, pixel-by-pixel, to detect any differences. Pixels that exceed a predetermined threshold are classified as foreground. The reference background image is created by background modeling. Once a foreground pixel map is generated, morphological post-processing techniques such as erosion, dilation, and closing are applied to minimize noise and enhance the detected regions. To account for dynamic scene changes, the reference background should get updated with new images.

There are several different categories of background subtraction methods, each with varying methods of background modeling, background maintenance, foreground detection, and post-processing. These methods address the basic scheme of background subtraction in different ways. Some of the categories of background subtraction methods include statistical methods, low rank and sparse matrix decomposition methods, methods based on dynamic feedback mechanisms, and neural network methods. In Section 2.3, we will provide an overview of the various categories of background subtraction methods.

In one study [40], Heikkila and Silven used a straightforward version of background subtraction scheme. In their method, a pixel at location (x, y) in the current image, I_t, is classified as foreground if $|I(x, y) - B(x, y)| > T$ holds true, where T is a predetermined threshold.

After the creation of the foreground pixel map, morphological closing and the removal of small-sized regions are applied to refine the results. However, background subtraction techniques can be sensitive to dynamic changes in the scene, such as when stationary objects reveal the background (e.g. a parked car exiting a lot) or sudden changes in illumination occur. Nonetheless, despite these limitations, background subtraction techniques are still effective at identifying the majority of relevant pixels in moving regions.

Temporal Differencing

When videos are captured by a moving camera, background subtraction cannot be used to detect moving objects because the background is constantly changing. Instead, temporal differencing is commonly employed to identify moving regions by computing the difference between consecutive frames in a video sequence. However, since the motion of the camera and the object are mixed in a moving camera, some techniques require estimating the camera's motion before detecting the object.

Temporal differencing is advantageous as it is adaptive to dynamic changes in the scene and involves the most recent frames in the computation of the moving regions. However, it may fail to detect the entire relevant pixels of certain types of moving objects and will not detect objects that have stopped in the scene since it uses the last frame of the video sequence as a reference.

Optical Flow

Optical flow techniques are a popular method for detecting moving objects in an image by utilizing flow vectors of moving objects. This involves calculating the velocity and direction of each pixel in the frame, which can be time-consuming. However, using optical flow, a background motion model can be generated to stabilize the image of the background plane. Additionally, independent motion can be detected as residual flow or flow in the direction of the image gradient, which is not predicted by the background plane motion.

Optical flow is capable of detecting motion in video sequences even when the camera and the background are in motion. However, most optical flow methods are computationally complex and may require specialized hardware to be used in real-time applications. Therefore, these methods may not be suitable for certain applications that require real-time detection.

1.2 Contributions

Moving object detection has been tackled by various methods, including supervised deep learning-based techniques that are currently considered state-of-the-art. However, these methods require pixel-wise ground-truth labeling, which is both time-consuming and expensive. On the other hand, effective methods based on low rank and sparse matrix decomposition have been proposed for MOD, which demonstrate strong performance. However, these latter methods work in a batch mode, which is unsuitable for real-time applications, and they are also computationally inefficient.

To address these limitations, this thesis proposes two unsupervised and one weakly supervised neural network frameworks for background subtraction. The objective is to perform online (i.e., not in batch mode) background subtraction without the need for per-pixel ground-truth labels, providing an efficient and effective alternative that can significantly reduce the annotation burden while maintaining comparable performance to state-of-the-art methods. Below, we will briefly explain the contributions of our proposed methods.

Online Illumination Invariant Moving Object Detection by Generative Neural Network: One of the challenges is to separate moving objects from illumination changes and shadows that are present in most real world videos. State-of-the-art methods that can handle illumination changes and shadows work in a batch mode; thus, these methods are unsuitable for processing long video sequences or for real-time applications. In our first proposed method [6], we address this challenge by extending a state-of-the-art batch MOD method (ILISD) [82] to an online/incremental MOD approach, using unsupervised and generative neural networks that leverage illumination-invariant image representations. For each image in a video sequence, we use a neural network to generate a low-dimensional representation of the background image, and then we decompose the foreground image into its illumination change and moving object components based on the illumination-invariant representation. Optimization is performed by stochastic gradient descent in an end-to-end and unsupervised manner, and our method can work in both batch and online modes. In batch mode, like other batch methods, the optimizer uses all of the images in the sequence, while in online mode, images can be incrementally fed into the optimizer. Our experimental evaluation on benchmark image sequences demonstrates that both the online and batch modes of our algorithm achieved state-of-the-art accuracy on most datasets.

Dynamic Background Subtraction by Generative Neural Networks: One of the challenges in background subtraction methods is dealing with dynamic backgrounds, which can have stochastic movements in some parts of the scene. In our second method [4], we propose a novel background subtraction framework called DBSGen, which leverages two generative neural networks: one for dynamic motion removal and another for background generation. DBSGen utilizes a generative multi-resolution convolutional network to estimate a dense motion map that minimizes the difference between each input image and a fixed background image. The fixed background image is chosen from the video as an initial background model. Next, our method warps each input image using its pixel-wise motion, which maps most dynamic background pixels to the corresponding pixels in the fixed background image. However, some moving objects may also be warped in this process. DBSGen then leverages a generative fully connected network to create background images for the warped input images, from which foreground images are obtained by subtracting the background images from the warped images. To avoid deformed objects in the results, an inverse warping of the motion map is applied to the foreground images to warp back the moving objects. Inspired by the SuBSENSE method [19], DBSGen computes a pixel-wise dynamic entropy map that serves as an indicator of dynamic background spots. This map is used to determine a pixel-wise distance threshold, which in turn is used to obtain binary segmented images. Finally, some basic post-processing operations are applied to enhance the results. DBSGen is an end-to-end, unsupervised optimization method that achieves a near real-time frame rate. We evaluated the performance of our method over dynamic background sequences and found that it outperforms most state-of-the-art unsupervised methods.

Weakly Supervised Realtime Dynamic Background Subtraction: In our third method [5], we propose a weakly supervised framework that can perform background subtraction without requiring per-pixel ground-truth labels. Our framework is trained on a moving object-free sequence of images and comprises two networks. The first network is an autoencoder that generates static background images and prepares dynamic background images for training the second network. The dynamic background images are obtained by thresholding the background-subtracted images. The second network is a U-Net that uses the same moving object-free video for training and the dynamic background images as pixel-wise ground-truth labels. During the test phase, the input images are processed by the autoencoder and U-Net, which generate static and dynamic background images, respectively. The dynamic background image helps remove dynamic motion from the background subtracted image, enabling us to obtain a foreground image that is free of dynamic artifacts. To demonstrate the effectiveness of our method, we conducted experiments on various dataset. Our method outperformed all top-ranked unsupervised methods. It also surpassed one of the two existing weakly supervised methods, while achieving comparable results to the other method but with a shorter running time. Our proposed method is online, realtime, efficient, and requires minimal frame-level annotation, making it suitable for a wide range of real-world applications.

1.3 Thesis Outline

The thesis is organized as follows: Chapter 2 provides an overview of various background subtraction methods. Chapter 3 presents a comprehensive explanation of our framework for online illumination invariant moving object detection using generative neural networks. Chapter 4 discusses our unsupervised dynamic background subtraction method utilizing generative neural networks. Chapter 5 describes our weakly supervised dynamic background subtraction method. Finally, in chapter 6, we summarize our findings and highlight future research directions.

Chapter 2

Background and Related Works

The initial step in many computer vision applications that use videos involves detecting moving objects. Following this, background subtraction is employed to segment foreground from the background. Despite being a well-studied problem, background subtraction still needs considerable research efforts to address unresolved challenges [49].

Different stages involved in background subtraction methods can be broken down into four steps [11], [34]. The first step is background initialization, which involves calculating the initial background image. The second step is background modeling, which involves creating a model to represent the background. The third step is background maintenance, which deals with the process of updating the model to account for changes over time. Finally, the fourth step is pixel classification, which entails determining whether a pixel belongs to the background or the moving objects class. Background subtraction methods can be categorized into several groups, each with different techniques for background initialization, background modeling, background maintenance and foreground detection.

This chapter is divided into two sections. Section 2.1 discusses the challenges associated with background subtraction, while Section 2.3 provides an overview of existing methods.

2.1 Challenges

In this section, we provide an overview of some of the challenges of background subtraction methods.

2.1.1 Illumination Changes

Gradual and sudden illumination changes can have a significant impact on the background's appearance and lead to false positive detection. Ideally, the background subtraction model should be able to adapt to changes in the environment's appearance. This is particularly important when dealing with outdoor settings where lighting conditions can vary throughout the day. In addition, sudden changes in illumination, such as turning a light on or off indoors, or shifting from cloudy to sunny outdoors, can also occur.

2.1.2 Shadows

Shadows produced by objects in the foreground can have a negative impact on the performance of background subtraction methods. Overlapping shadows can make it challenging to separate and classify foreground regions accurately. As a result, researchers have proposed different techniques to identify shadows.

2.1.3 Dynamic Background

Certain elements of the scene, such as a fountain, swaying tree branches, moving clouds, or water waves, may display motion but are still considered part of the background due to their relevance. This movement can occur periodically or irregularly, such as the blinking of traffic lights or the waving trees in the wind. Managing such background dynamics poses a significant challenge in the background subtraction.

2.1.4 Challenging Weather

When videos are recorded under harsh winter weather conditions, such as snowstorms, snowy terrains, or fog, as well as in the presence of air turbulence, detecting moving objects becomes an exceedingly challenging task.

2.1.5 Camera Jitter

Sometimes, videos may be captured using unstable cameras that can vibrate or shake, leading to an unsteady image. The amount of jitter or shaking can differ from one video to another, making it challenging to standardize a solution for such an issue.

2.1.6 Bootstrapping

Bootstrapping videos are captured in cluttered environments where there is no initial training data available without foreground objects. To initialize the background model in such scenarios, a bootstrapping strategy is employed to estimate a background frame that is devoid of any foreground objects [41], [89].

2.1.7 Camouflage

Correct classification can be challenging in surveillance applications when certain objects have a poor contrast with the background. Camouflage presents a significant challenge, especially when using temporal differencing methods.

2.1.8 Occlusion

The computation of the background frame and the moving objects can be affected by partial or full occlusion. Occurring at any time when an object passes behind an obstacle with respect to the camera, occlusion can significantly impact moving object detection in real-life situations [48].

2.1.9 Intermittent Object Motion

The detection of objects with intermittent motion can give rise to 'ghosting' artifacts [79], which occur when objects start and stop moving within short intervals. Additionally, there may be instances when a video contains still objects that suddenly begin to move, such as parked cars being driven away.

2.2 Evaluation Metrics

The evaluation metrics are used to measure the performance of a background subtraction algorithm. These metrics are calculated based on the number of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN) in the output of the algorithm. The definitions of TP, TN, FP, and FN are as follows:

- TP: true positives are foreground pixels detected correctly.
- TN: true negatives are background pixels detected correctly.
- FP: false positives are background pixels misclassified as foreground.
- FN: false negatives are foreground pixels misclassified as background.

2.2.1 Precision

In the context of background subtraction, precision is a metric that measures the proportion of detected pixels that correspond to the moving object. It represents the percentage of the detected foreground pixels that are actually part of the foreground, i.e., the percentage of true positive detections among all the positive detections.

In other words, precision measures the algorithm's ability to avoid falsely detecting pixels in the background as foreground. A high precision score indicates that the algorithm is accurately detecting only the pixels that belong to the foreground and not misclassifying background pixels as foreground. On the other hand, a low precision score indicates that the algorithm is detecting a significant number of false positives (background pixels classified as foreground) along with the true positives (foreground pixels correctly classified as foreground).

To calculate precision, we use the following formula:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.1}$$

2.2.2 Recall

recall is a metric that measures the percentage of all pixels belonging to the moving object that are correctly detected. Recall represents the algorithm's ability to detect all the foreground pixels, including the true positives and the false negatives.

In other words, recall measures the completeness of the detection process, i.e., how well the algorithm can detect all the foreground pixels in the video sequence. A high recall score indicates that the algorithm is detecting a high proportion of the foreground pixels, while a low recall score indicates that the algorithm is failing to detect a significant number of foreground pixels, leading to false negatives.

To calculate recall, we use the following formula:

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.2}$$

2.2.3 F-measure

Overall, recall and precision are complementary metrics that provide a measure of the performance of a background subtraction algorithm. While precision measures the accuracy of the algorithm in detecting foreground pixels, recall measures the completeness of the detection process. A good background subtraction algorithm should have both high precision and high recall scores. The F-measure formula is a harmonic mean of precision and recall, and it combines these two metrics into a single score that ranges from 0 to 1, with higher values indicating better performance.

To calculate F-measure, the following formula is used:

$$\text{F-measure} = 2 \cdot \frac{\text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \tag{2.3}$$

By calculating the F-measure for different background subtraction algorithms and comparing the scores, we can determine which algorithm performs better in terms of accurately detecting foreground objects while minimizing false positives and false negatives. Therefore, the F-measure is widely used in the field of computer vision and image processing to assess the performance of object detection algorithms, including those used for background subtraction. Overall, the F-measure provides a comprehensive and reliable way to evaluate the effectiveness of an algorithm in detecting foreground objects in a video sequence, and thus is employed in most previous studies. We also use this metric to compare the performance of different background subtraction and moving object detection algorithms.

2.3 Related Work

2.3.1 Low Rank and Sparse Matrix Decomposition Methods

Many effective methods for MOD are based on low rank and sparse (LRS) matrix decomposition. These methods use the assumption that background pixels are linearly correlated to each other temporally in a sequence of images and usually apply a variant of Principal Component Analysis (PCA) on a sequence of images to obtain a low rank representation for the background and sparse outliers representing moving objects.

PCP method [17] solved the matrix decomposition problem by minimizing a combination of the nuclear norm of the low rank matrix and L_1-norm of the sparse matrix. Zhou *et al.* [109] accelerated the decomposition in their proposed algorithm GoDec and its faster variant Semi-Soft GoDec (SSGoDec). Wang *et al.* [96] proposed Probabilistic Robust Matrix Factorization (PRMF) that uses a Laplace error and a Gaussian prior. A group of the LRS methods made use of connectivity constraint on moving objects [59], [97], [110]. In [97] a Bayesian robust matrix factorization (BRMF) model is proposed. Its extension, Markov BRMF (MBRMF), assumes outliers in foreground form groups with spatial or temporal proximity by placing a first-order Markov random field.

Another method called Decolor [110] assumes foreground objects form small contiguous regions and incorporates prior knowledge of contiguity in detecting outliers using Markov Random Fields. Liu *et al.* [59] proposed Low rank and Structured sparse Decomposition (LSD) framework. LSD considers spatial information in sparse outliers and model the foreground by structured sparsity-inducing norms. It also uses a motion saliency map to remove background motion from foreground candidates, thus, it can tolerate some sudden background variations. In general, for scenes with moderate illumination changes, LRS methods can handle illumination changes. However, with the presence of significant illumination changes and shadows, LRS methods fail to separate moving objects from illumination changes [82].

Another method, ILISD [82], incorporated low rank and sparse decomposition along with prior knowledge about illumination. ILISD was able to distinguish between illumination changes and real foreground changes. The drawback of this method is that it can only work in a batch mode. For continuous monitoring tasks and very long sequences, an online (incremental) method is required. When number of images in the sequence increases, memory storage and computations grow significantly for ILISD.

Another group of MOD methods works in an online/incremental manner. Most of these method are based on robust PCA. GRASTA is an online incremental gradient descent algorithm which estimates robust subspace from subsampled data [39]. OPRMF is the online extension of PRMF which uses expectation-maximization algorithm which can be updated incrementally [96]. OR-PCA proposes an online robust PCA method. To process the frames incrementally, it uses multiplication of the subspace basis and coefficients instead of nuclear norm and updates the basis for the new frame [31]. COROLA [80] method uses a sequential low rank approximation on a fixed window of images. Moreover, it considers outliers as small contiguous regions like the Decolor method. Although these methods have been applied successfully in real-time moving object detection, they fail to provide satisfactory results, when significant illumination changes are present.

2.3.2 Statistical Methods

There is a category of methods that use statistical approaches based on probability density estimation of pixel values. The most basic of these methods is the single Gaussian model [101]. However, this approach has limitations as a single function cannot account for all variations in pixel values. To overcome this, the Gaussian mixture model (GMM) [83] was proposed, which uses several Gaussians. Various improved versions of this traditional and widely used method have been presented [47], [53], [111], [112] with better results. Flux Tensor with Split Gaussian models (FTSG) [98] is a state-of-the-art method that uses flux tensor-based motion segmentation and GMM-based background modeling separately and then merges the results. Finally, it enhances the results using a multi-cue appearance comparison. However, parametric methods such as GMM and its successors are unable to handle sudden changes in a scene. To address this issue, a statistical non-parametric algorithm called KDE [30] was introduced, which uses kernel density estimation to model the probability of pixel values.

2.3.3 Methods Based on Dynamic Feedback Mechanisms

One of the main categories of methods for background modeling involves using controller parameters that update the background model based on dynamic feedback mechanisms. One such method, SuBSENSE [19], incorporates color channel intensity and spatio-temporal binary features and adjusts its parameters using pixel-wise feedback loops based on segmentation noise. PAWCS [20], a newer and more advanced method, extends the capabilities of SuBSENSE by generating a strong and persistent dictionary model based on spatio-temporal features and color. Similar to SuBSENSE, PAWCS also employs automatic feedback mechanisms to adjust its parameters. Another method, SWCD [44], combines the dynamic controllers of SuBSENSE with a sliding window approach to update background frames. Finally, CVABS [45], a recent subspace-based method, utilizes dynamic self-adjustment mechanisms like SuBSENSE and PAWCS to update the background model.

2.3.4 Ensemble Methods

Ensemble methods have emerged as a new approach for change detection algorithms. The authors of [9], [10] have recently introduced a method named IUTIS (In Unity There Is Strength) that utilizes genetic programming (GP) to combine different algorithms and maximize their individual strengths. By selecting the best methods, combining them in various ways, and applying appropriate post-processing techniques, GP enables IUTIS to achieve high performance. The method shows promising performance by integrating several top-ranked methods evaluated on *CDnet 2014* ([99]).

2.3.5 Deep Learning Methods

Several deep neural network (NN) techniques have been proposed in recent years for foreground segmentation [3], [14], [24], [75], [104], [105], owing to the success of deep learning in other computer vision domains. FgSegNet and its variations [33], [57], [58] are presently considered state-of-the-art, based on their performance on *CDnet 2014*. Motion U-Net [70] is another deep NN method that requires fewer parameters than FgSegNet. BSPVGAN [108] employs Bayesian Generative Adversarial Networks (GANs) to create a background subtraction model. Another technique called Cascade CNN [100] uses a multi-resolution convolutional neural network (CNN) to segment moving objects. In DeepBS [3], a CNN is trained using patches of input images, which are then merged to reconstruct the frame. Temporal and spatial median filtering is utilized to enhance the segmentation outcomes. Another supervised approach, BSUV-Net [86], [87], is trained on some image sequences along with their spatio-temporal data augmentations, and exhibits good performance on unseen videos after training. Among the assessed methods on *CDnet 2014*, the aforementioned neural network techniques are ranked at the top. However, they require supervised training, which entails pixel-wise annotated ground-truth, a time-consuming and impractical task in many situations.

A number of recently developed techniques, including SemanticBGS [13] and its variations RT-SBS-v1 and RT-SBS-v2 [26], integrate semantic segmentation with background subtraction methods. They employ the information from a semantic segmentation algorithm to obtain a pixel-wise probability that enhances the output of any background subtraction method. However, we cannot compare them to our method because they rely on pixel-level information as input, even though they are not trained using ground-truth labels.

2.3.6 Unsupervised Neural Network Based Methods

BEN-BLN is an unsupervised neural network based method proposed in [102]. In this work, a stack of autoencoders,called Background Extraction Network (BEN), is used to estimate a background model and then a second autoencoder, called Background Learning Network (BLN), is in charge of enhancing the background. Also an online extension of this method is proposed, BEN-BLN-Online, that trains the network on a first batch of images and then finetunes it for streams of input images. Even though BEN-BLN model builds a non-linear background model, it is unable to accommodate illumination changes.

2.3.7 Weakly Supervised Methods

In recent years, some weakly supervised methods have emerged that solely rely on image-level tags, which indicate whether a foreground object is present in the image [66], [107] and avoids pixel level annotations. The method proposed in [66] generates a binary mask image by subtracting a background image from an input image to identify foreground regions. It then uses intermediate feature maps of a CNN to refine the foreground locations. However, image-level supervision presents a challenge due to the lack of location information in training the network. To address this, the method introduces some constraints that help to locate foreground pixels.

Another recent technique, LDB [107], adopts a tensor-based decomposition framework to represent the background as a low rank tensor and classify the sparse noise as foreground. Additionally, it trains a two-stream neural network using an object-free video to explicitly learn the dynamic background. The dynamic background component of LDB leads to a more precise decomposition of the background and foreground, making it the current state-of-the-art method in weakly supervised moving object detection, particularly in dynamic background scenes.

Chapter 3

Online Illumination Invariant Moving Object Detection by Generative Neural Network

3.1 Introduction

Moving object detection (MOD), which tries to separate moving objects out of a sequence of images, is one of the fundamental tasks in computer vision that has various applications, such as video surveillance in airports, residential areas, shopping malls [15], [88], traffic monitoring [21], [78], object tracking [76], [84] and gesture recognition in human machine interaction [90]. Different types of methods have been proposed for MOD. However, many of these methods are vulnerable to illumination changes.

In this chapter, we propose an end-to-end framework called Neural Unsupervised Moving Object Detection (NUMOD) [6] following the principles of the batch method, ILISD [82]. Because of the parameterization via generative neural network, NUMOD can work **both in the online and in the batch mode.** NUMOD's goal is decomposing each frame into three parts: background, illumination changes and moving objects that we are interested in. It uses a fully connected generative neural network to generate a background model by finding a low-dimensional manifold for the background of the image sequence. For each image, after subtracting the background, the sparse outliers remains. The sparse outliers include not only moving objects, but also moving shadows and illumination changes. To distinguish between

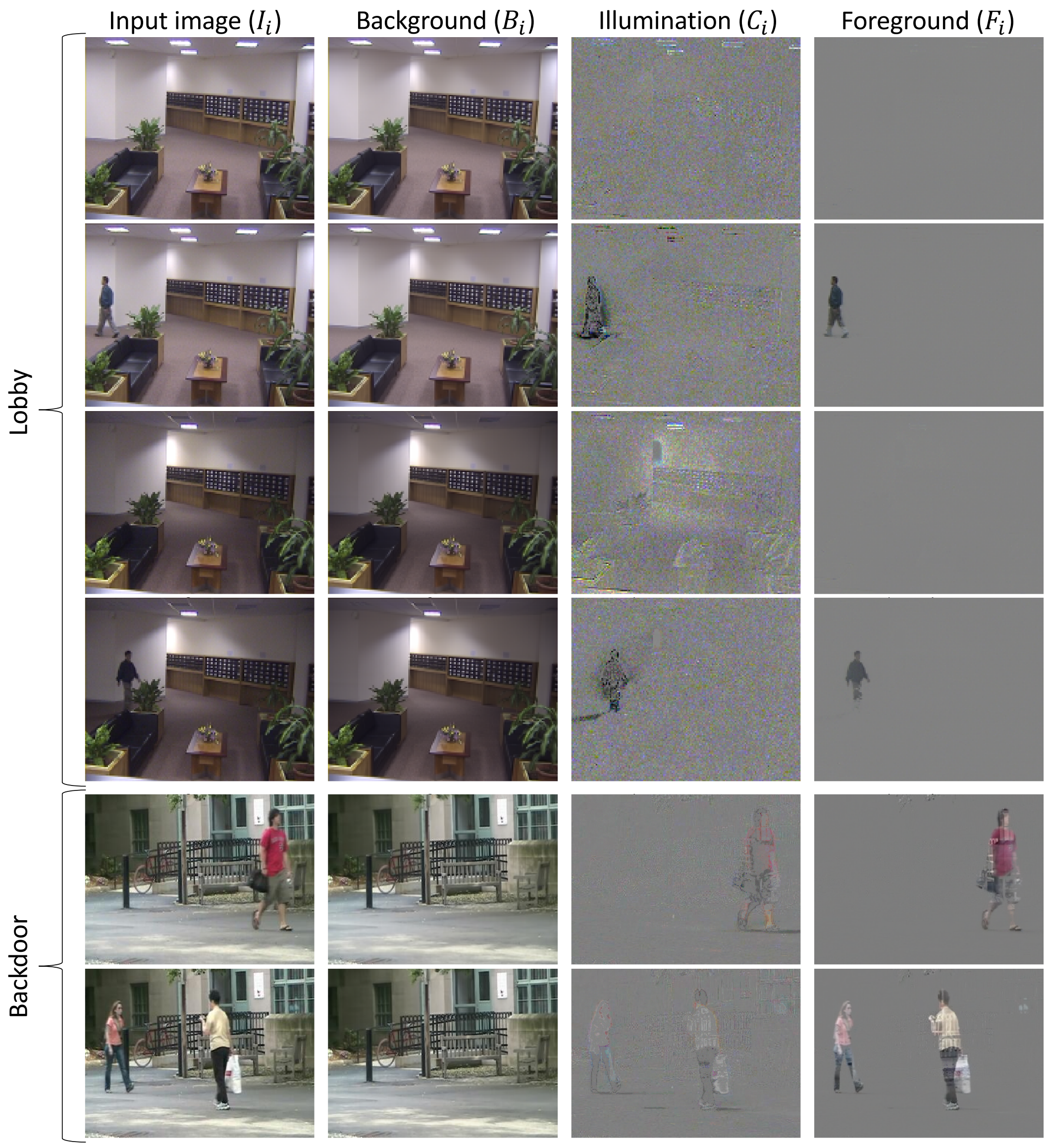


Figure 3.1: Proposed method (NUMOD) decomposes input image, I_i, into background image, B_i, illumination changes and shadows, C_i and foreground moving objects F_i. Examples are shown on two benchmark sequences “Lobby” and “Backdoor.”

them, NUMOD uses an illumination invariant representation of the images as prior knowledge that is robust against illumination changes and shadows. This representation has been successfully used in ILISD [82]. NUMOD adds some constraints to the loss function based on this invariant representations that enables it to decompose the sparse part of each image into illumination and foreground.

Fig. 3.1 shows example decomposition of input images obtained by NUMOD. This qualitative results illustrate how NUMOD decomposes an input image, I_i, to background image, B_i, illumination changes and moving shadows, C_i, and detected foreground objects, F_i. The first four rows are selected from "Lobby" sequence, which is an indoor scene with illumination changes. It can be seen that background images capture some part of the illumination changes, but most of the illumination changes and shadows are captured in C_i. The foreground image, F_i, is free of shadows and illumination changes. The last two rows are from "Backdoor" sequence, which is an outdoor scene with moving shadows. Again, we observe that moving shadows and illumination changes are separated from moving objects. These qualitative results show capability of our method in handling illumination changes and shadows. We also perform quantitative experiments.

NUMOD's advantages can be summarized as follows. First, it can work in both batch and online modes of operation. Second, it uses prior knowledge to overcome illumination challenge in an end-to-end neural network framework. Third, unlike other neural network based methods, it trains in an unsupervised way, without requiring expensive pixel-wise ground-truth masks. In a nutshell, the main contribution of NUMOD is that it is an online method, which has excellent capability of handling illumination changes.

The rest of the chapter is structured as follow: Section 3.2 explains the illumination invariant representation we use in NUMOD. In Section 3.3, NUMOD methodology and framework are described. We report our experimental results in Section 3.4. Finally, Section 3.5 provides conclusions and an outline of future work.

3.2 Illumination Invariance Prior Knowledge

Many methods have been proposed for shadow free images and illumination invariant images. One of the well known methods is proposed by Finlayson *et al.* [32], which shows that for an RGB image under illumination changes, the 2D log-chromaticity vector for a color surface moves along a straight line in the scatter-plot of $\log(R/G)$ vs. $\log(B/G)$. The direction of this line, e, is same for different surfaces. If we project all the chromaticities on a line orthogonal to e then all the points of the same surface, independent of the illumination, will be projected to the same point. This model provides a shadow free image useful for distinguishing between real moving objects and illumination changes.

However, [32] has some initial assumptions, which do not always hold in real data sets. The model assumes that the scene's illumination is Planckian, the camera sensors are narrow-band and the image surfaces have Lambertian reflectance. When these assumptions are not correct, chromaticities of an image's surface do not move along a straight line.

To overcome this problem, we use Wiener filter to get illumination invariant features of images while preserving their structural information [22]. In [22] Wiener filter is used to separate illumination and reflectance of an image across the whole frequency spectrum. The advantage of this method is that unlike other methods, it considers low frequency part of spectrum as well. Consequently, it preserves features at every frequency.

Let $I_i, i = 1, 2, ..., n$ be an input image sequence. We combine Finlayson *et al.*'s shadow free images and illumination invariant images extracted by Wiener filter by a simple averaging as proposed in [81]. The following function denotes transformation of an input image into an illumination invariant representation:

$$I_i^{\text{inv}} = \Psi(I_i). \tag{3.1}$$

Fig. 3.2 shows two input images I_i of a same scene under illumination changes and their respective invariant representations I_i^{inv}.

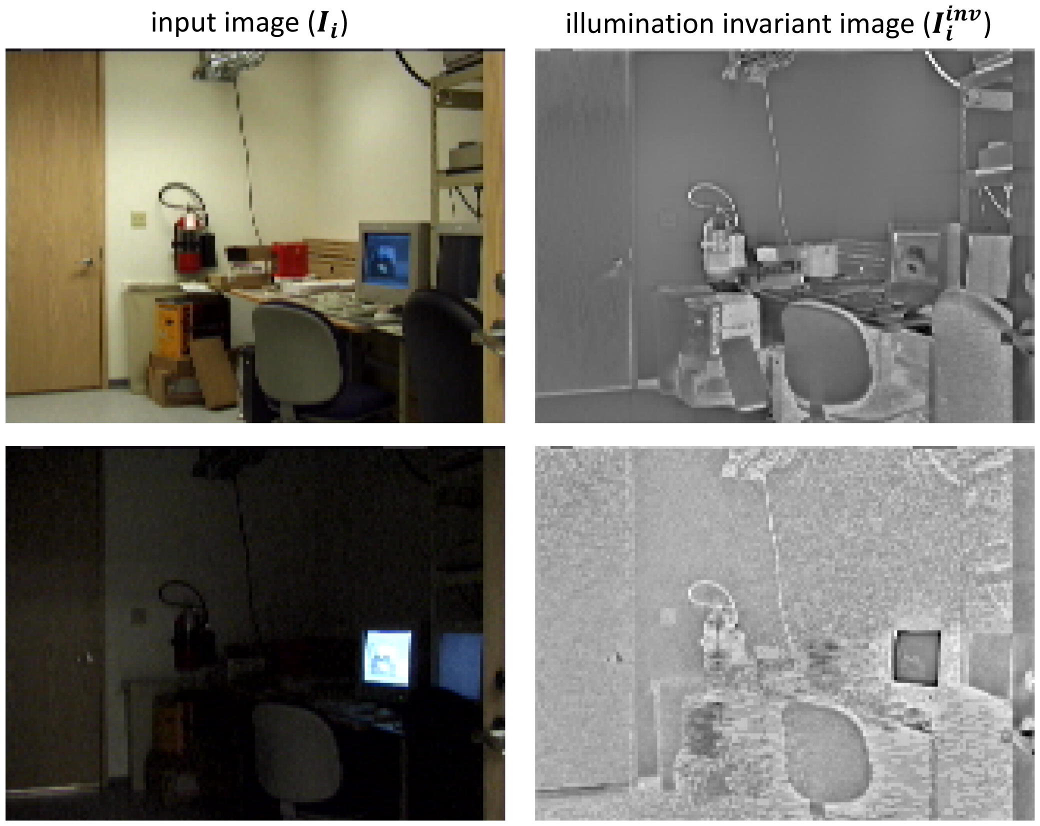


Figure 3.2: Left column shows two input images, I_i, from "LightSwitch" sequence representing illumination changes; Right column shows corresponding illumination invariant images, I_i^{inv}.

3.3 Proposed method: NUMOD

Let $I_i \in \mathbb{R}^m, i = 1, 2, ..., n$ be a sequence of vectorized RGB input images, where m is the number of pixels in each image. Our method decomposes I_i into three images: a background image B_i, an image C_i representing illumination change and another image F_i denoting moving objects as follows:

$$I_i = B_i + C_i + F_i. \tag{3.2}$$

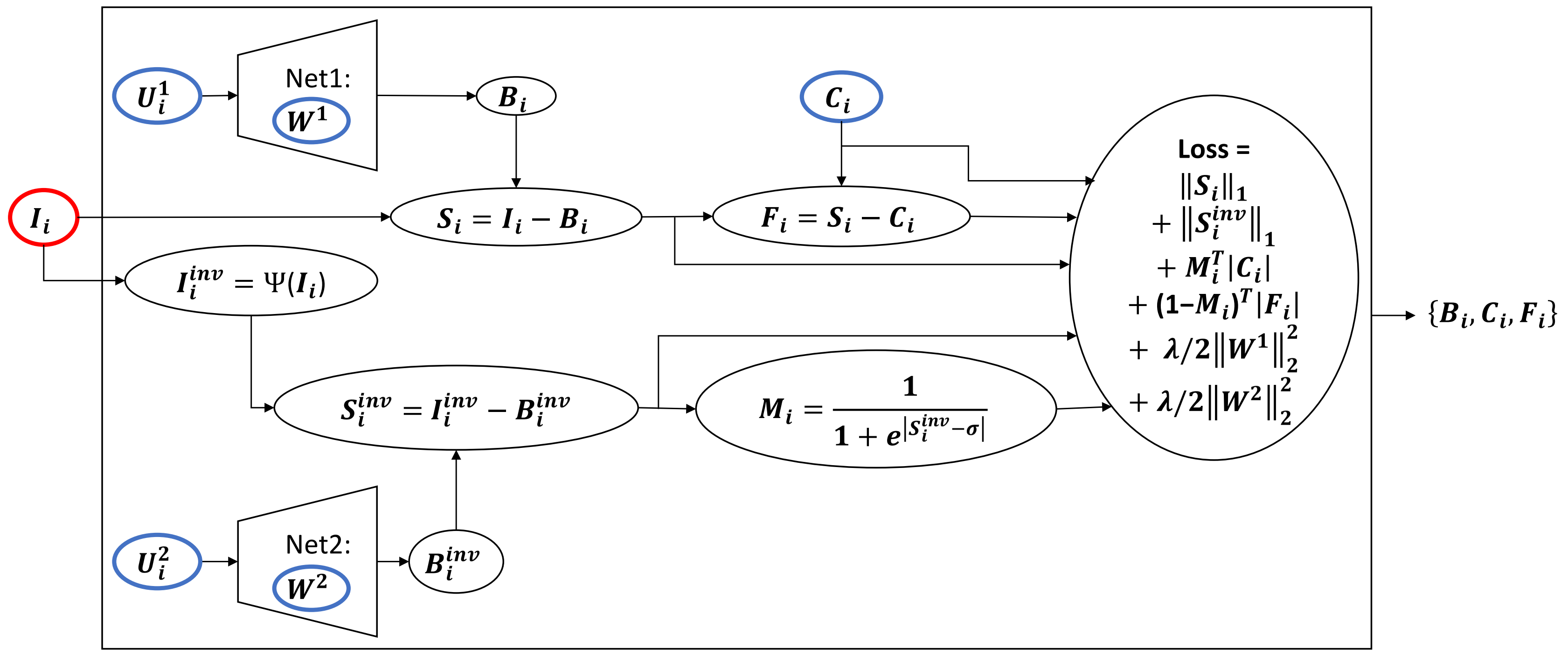
The flow of computations for an input frame is shown in Fig. 3.3. In summary, the background image B_i is generated using a fully connected generative network with a low dimensional input vector U_i^1. We also decompose the remaining sparse vector $(I_i - B_i)$ into illumination changes C_i and foreground moving objects F_i by applying illumination invariant constraints. We will explain details of our algorithm in the following sections.

3.3.1 Generative Network Architecture

There are two Generative Fully Connected Networks (GFCN) in NUMOD: Net1 and Net2 (Fig. 3.3). Net1 is in charge of estimating background image, B_i, from the input image I_i and Net2 generates background image B_i^{inv} for the illumination invariant image I_i^{inv}. These two networks have the exact same architecture which is shown in Fig. 3.4.

Input to GFCN is an optimizable low-dimensional latent vector (U_i^1 and U_i^2 in Fig. 3.3). After that there are two fully connected hidden layers each followed by ReLU non-linearity. The second hidden layer is fully connected to the output layer which is followed by the sigmoid function. The reason to use the sigmoid function at the last layer is to limit background values between zero and one.

A loss term (3.3) imposes that the output of GFCN be similar to the current input frame. GFCN is similar to the decoder part of an autoencoder. In an autoencoder, the low-dimensional latent code is learned by the encoder, whereas in GFCN, it is a free parameter that can be optimized and is the input to the network. During training, this latent vector learns a low-dimensional



U_i^1
Net1: W^1
B_i
C_i
I_i
$S_i = I_i - B_i$
$F_i = S_i - C_i$
$I_i^{inv} = \Psi(I_i)$
Loss = $\|S_i\|_1 + \|S_i^{inv}\|_1 + M_i^T|C_i| + (1-M_i)^T|F_i| + \lambda/2\|W^1\|_2^2 + \lambda/2\|W^2\|_2^2$
$\{B_i, C_i, F_i\}$
$S_i^{inv} = I_i^{inv} - B_i^{inv}$
$M_i = \frac{1}{1+e^{|S_i^{inv}-\sigma|}}$
U_i^2
Net2: W^2
B_i^{inv}

Figure 3.3: The graph shows the flow of the computations in our framework for the i^{th} image frame. Input image I_i is shown in red outline. Net1 and Net2 are the two generative neural networks with parameters W^1 and W^2, respectively. Optimizable variables $\{U_i^1, U_i^2, W^1, W^2, C_i\}$ are shown in blue outline. Output is the triplet $\{B_i, C_i, F_i\}$ with the relation: $I_i = B_i + C_i + F_i$.

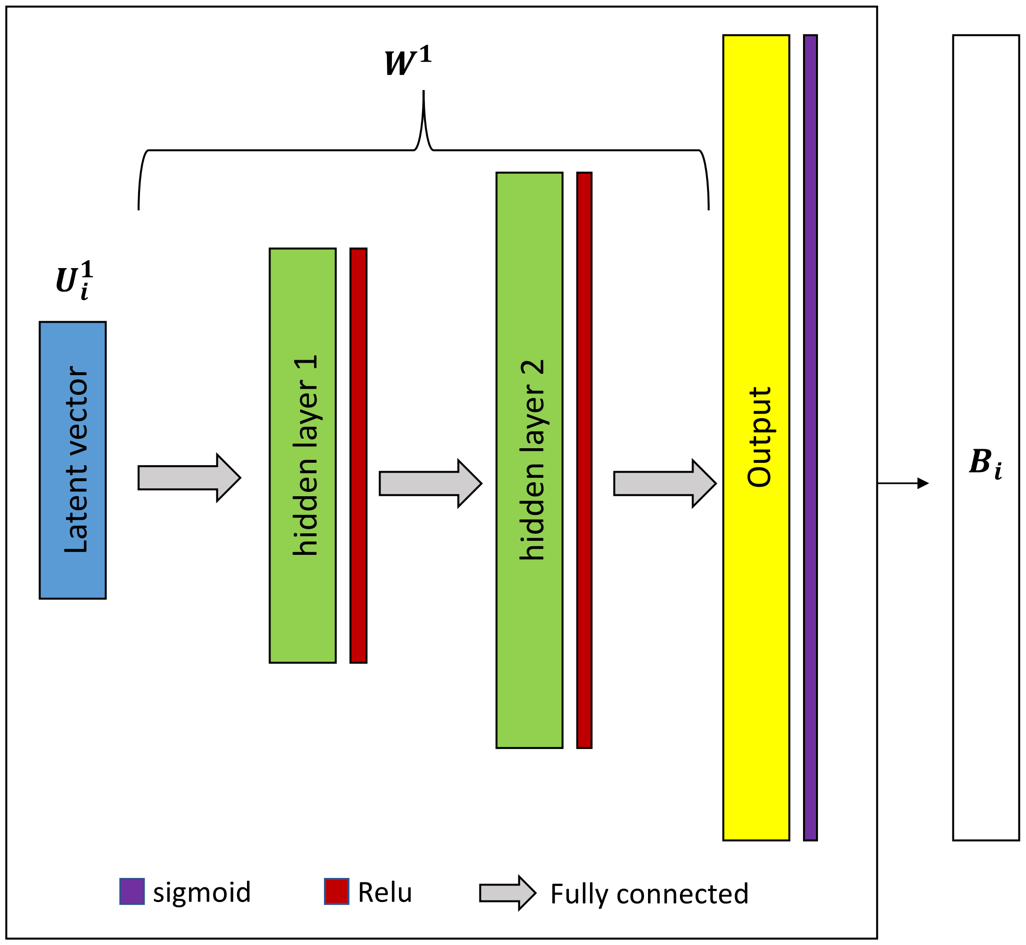


Figure 3.4: Generative fully connected network: Net1

manifold of the input distribution. If no further constraint is applied, the network will learn a naive identity function. Hence by restricting the capacity of the network, and by limiting number of hidden units in the hidden layers, the network is able to extract the most salient features of the data and gets a structure of the data distribution [36]. In our problem, since images of the sequence are temporally correlated to each other, GFCN is able to learn a background model of the images and output of the network is the background image.

The L_{reconst} loss term, responsible for constructing background images of I_i and I_i^{inv}, is as follows:

$$L_{\text{reconst}} = \sum_i \|I_i - B_i\|_1 + \sum_i \|I_i^{\text{inv}} - B_i^{\text{inv}}\|_1, \tag{3.3}$$

where B_i and B_i^{inv} are outputs of Net1 and Net2, respectively. We use L_1-norm instead of L_2-norm in L_{reconst} to encourage sparsity of the sparse remainder of images [16].

3.3.2 Illumination and Foreground Decomposition

NUMOD is a unified framework shown in Fig. 3.3. In the previous section, we explained how Net1 and Net2 generate backgrounds for an input image and an invariant image, respectively. Subtracting the background from an input image gives the sparse outliers of that frame (3.4). S_i includes illumination changes and foreground moving objects:

$$\begin{aligned} S_i &= I_i - B_i \\ S_i^{\text{inv}} &= I_i^{\text{inv}} - B_i^{\text{inv}}. \end{aligned} \tag{3.4}$$

As explained earlier, using Net2, We decompose an illumination invariant image I_i^{inv} into invariant background image B_i^{inv} and invariant sparse foreground moving objects S_i^{inv}. Since, we assume that I_i^{inv} is independent of illumination changes, we can use its sparse part S_i^{inv} as a map for actual moving objects as follows [82]:

$$M_i = \frac{1}{1 + e^{-(|S_i^{\text{inv}} - \sigma|)}}, \tag{3.5}$$

where $| \cdot |$ denotes absolute value, σ is the standard deviation of pixels in I^{inv} and Sigmoid function normalizes M_i values between zero and one.

Prior map M_i is used to apply constraints to separate real foreground changes from illumination changes and moving shadows. One of the initial assumptions is that in ideal circumstances, prior map M_i has a large value for the pixels which include real changes and has a small value for the pixels in which only illumination variations happen. Since, illumination changes should be contained in a subspace orthogonal to the real changes, each input frame should satisfy the following constraints [82]:

$$\begin{aligned} & M_i^\top \mid C_i \mid = 0, \\ & (1 - M_i)^\top \mid F_i \mid = 0 \\ & s.t. \quad S_i = F_i + C_i, \end{aligned} \tag{3.6}$$

where $\top$ is the transpose symbol, C_i is the illumination change image and F_i is the image denoting moving foreground objects. C_i is an optimizable parameter in our framework (Fig. 3.3). Based on the illumination constraints, We can write the L_{decomp} loss term that is responsible for decomposing S_i into C_i and F_i as (3.7):

$$L_{\text{decomp}} = \sum_i M_i^T \mid C_i \mid + \sum_i (1 - M_i)^T \mid F_i \mid . \tag{3.7}$$

3.3.3 End-to-end Optimization

To prevent overfitting to noise, we apply $l2$ regularization, or in other words weight decay, on the parameters of the generative networks. L_{reg} of the network is defined as follows.

$$L_{\text{reg}} = \lambda(1/2\|W^1\|_2^2 + 1/2\|W^2\|_2^2) \tag{3.8}$$

W^1 and W^2 denote parameters of Net1 and Net2, except biases. λ is a hyper-parameter. The overall loss function of the whole framework is defined in (3.9):

$$L = L_{\text{reconst}} + L_{\text{decomp}} + L_{\text{reg}} \tag{3.9}$$

For optimization, we perform no preprocessing on the input data. We optimize the network for each image sequence independently. As mentioned earlier, NUMOD does not use ground-truths in the loss function L and it is optimized in an unsupervised manner. We use Adam [50], a stochastic gradient-based optimizer, to optimize all the parameters $\{U_i^1, U_i^2, C_i\}_{i=1}^n, W^1, W^2$. Due to the end-to-end optimization, Net1 and Net2 can have a good effect on each other and lead each other for a more accurate decomposition.

Finally, we apply a threshold on F_i vector at each pixel location (x, y) to obtain foreground binary mask b_i.

$$b_i(x,y) = \begin{cases} 1 & \mid F_i(x,y) \mid \geq 2t \\ 0 & \mid F_i(x,y) \mid < 2t \end{cases} \tag{3.10}$$

where t is the standard deviation of pixels in $\{F_i\}_{i=1}^n$.

3.3.4 Online Mode

The advantage of NUMOD is that it works in both batch and online modes. In the batch mode, like other batch methods, we optimize the parameters on batches of the sequence of images. The extension to online mode is natural by the virtue of parameterized networks, Net1 and Net2.

For the online mode, first, we optimize the network on an initial batch of images. Then, parameters W^1 and W^2 of the networks are frozen and the other variables U_i^1, U_i^2 and C_i are left optimizable (Fig. 3.3). We freeze network parameters to avoid overfitting while fine-tuning NUMOD for the oncoming stream of image frames. At this point, one or a few input frames are feed into the network, incrementally, and by backpropagating the error, their corresponding low-dimensional latent variables U_i^1, U_i^2 and illumination variable C_i are optimized. The loss term dose not change during online mode except that L_{reg} is omitted:

$$L_{\text{online}} = L_{\text{reconst}} + L_{\text{decomp}}. \tag{3.11}$$

Based on our experimental results, with an adequate size of the initial batch, the network parameters are well-optimized in the initial phase, so that after freezing those and feeding new frames, and just by optimizing U_i^1, U_i^2 and C_i, NUMOD is able to learn accurate background images and separate real changes from shadows and illumination changes.

3.3.5 Proximal Gradient Descent Algorithms

Since in our loss function we are optimizing the L_1-norm and the L_2-norm simultaneously, it is difficult to optimize them using a gradient descent algorithm. The gradient descent method assumes that the gradient is smooth everywhere on the real line, but this assumption fails when the L_1 regularization term is added to the loss function. The reason for this is that the L_1-norm is non-differentiable and oscillates during minimization. As a result, proximal gradient descent algorithms were developed to overcome this issue.

Proximal gradient descent is an optimization algorithm that is used when we want to minimize a function $f(x)$ that's the sum of two parts: a smooth function $g(x)$ which is convex and differentiable and a non-smooth function $h(x)$ which is convex and not necessarily differentiable [69].

$$f(x) = g(x) + h(x) \tag{3.12}$$

To minimize this function, we can not use regular gradient descent, because the non-smooth function $h(x)$ makes the gradient non-differentiable at certain points. Instead, we use proximal gradient descent, which iteratively performs two steps: a gradient descent step on the smooth part of the function, followed by a proximal operator on the non-smooth part of the function. The algorithm iterates until convergence, typically defined as a small change in the objective function or a maximum number of iterations.

The update equation for proximal gradient descent can be expressed as:

$$x_{k+1} = \text{prox}_{t,h}(x_k - t\nabla g(x_k)) \tag{3.13}$$

Here, x_k is the current point, x_{k+1} is the updated point, t is the step size or learning rate and $\nabla g(x_k)$ is the gradient of the g function at x_k.

The proximal operator, or proximal mapping, can be defined as:

$$\text{prox}_{t,h}(x) = \underset{u}{\text{argmin}} \left[h(u) + \frac{1}{2t}\|u - x\|_2^2 \right] \tag{3.14}$$

Here, $\|\cdot\|_2$ denotes the L_2-norm, and $h(x)$ encodes the constraints or regularization on the optimization problem. The function $h(x)$ is specific to the problem being solved and typically encodes constraints or regularization [68].

When the proximal gradient descent algorithm is used to solve a convex optimization problem with an L1 regularization term, $h(x)$ corresponds to the L_1-norm of the parameter vector x:

$$h(x) = \|x\|_1 = \sum_{i=1}^{n} |x_i| \tag{3.15}$$

The L_1-norm of a vector is the sum of the absolute values of its components. This regularization term promotes sparsity in the solution, meaning that it encourages many of the components of x to be exactly zero. This is useful in settings where we believe that only a small subset of the features or variables are relevant for the problem being solved.

To apply the proximal operator to the updated point $x_k - t\nabla g(x_k)$, we need to compute:

$$\operatorname{prox}_{t,\|\cdot\|_1}(x_k - t\nabla g(x_k)) = \underset{u}{\operatorname{argmin}} \left[\|u\|_1 + \frac{1}{2t}\|u - (x_k - t\nabla g(x_k))\|_2^2 \right] \tag{3.16}$$

The solution to this optimization problem is known as the soft-thresholding operator and can be defined component-wise as:

$$\operatorname{prox}_{t,\|\cdot\|_1}(x_i) = \begin{cases} x_i - t & \text{if } x_i > t \\ 0 & \text{if } -t \leq x_i \leq t \\ x_i + t & \text{if } x_i < -t \end{cases} \tag{3.17}$$

The soft-thresholding operator shrinks the components of $x_k - t\nabla g(x_k)$ towards zero by the threshold amount t while also setting them exactly to zero if they lie within the threshold range. This has the effect of promoting sparsity in the solution while also maintaining convexity of the optimization problem [8].

3.4 Experimental Results and Discussion

3.4.1 Experimental Setup

Benchmark Data Sets

We evaluate our proposed method on some benchmark data sets that include illumination changes and moving shadows to demonstrate that NUMOD can handle illumination changes.

Image sequences “LightSwitch” and “Camouflage” from Wallflower data set [89] and “Lobby” from I2R dataset [55] are selected due to sudden and global illumination changes. We Also selected “Cubicle”, “PeopleInShade”, “CopyMachine” and “Backdoor” sequences from CDnet data set [37] that include illumination changes and moving shadows and contain both indoor and outdoor scenes.

Evaluation Metric

We use F-Measure metric, defined in 2.2.3 for comparing different algorithms, which is used generally as a overall performance indicator of the moving object detection and background subtraction methods. For each sequence, F-measure is computed for each individual frame first and then the average over all the frames are computed and reported.

Network Setup

The size of latent vectors U_i^1 and U_i^2 is 5. In Net1 and Net2, the first and second layer size are 10 and 20, respectively. The only hyperparameter of the network is λ in (3.8), which is set 0.005 in all our experiments. Since overall cost function includes L_1-norm and L_2-norm,as mentioned in section 3.3.5, it is recommended to use proximal gradient descent algorithms. However, we were able to optimize it using the Adam optimizer algorithm [50]. Adam algorithm with fixed learning rate of 0.001 is used throughout. For the online mode, the first half of the sequence is used to pre-train the network and then streams of 10 frames are used for online optimization.

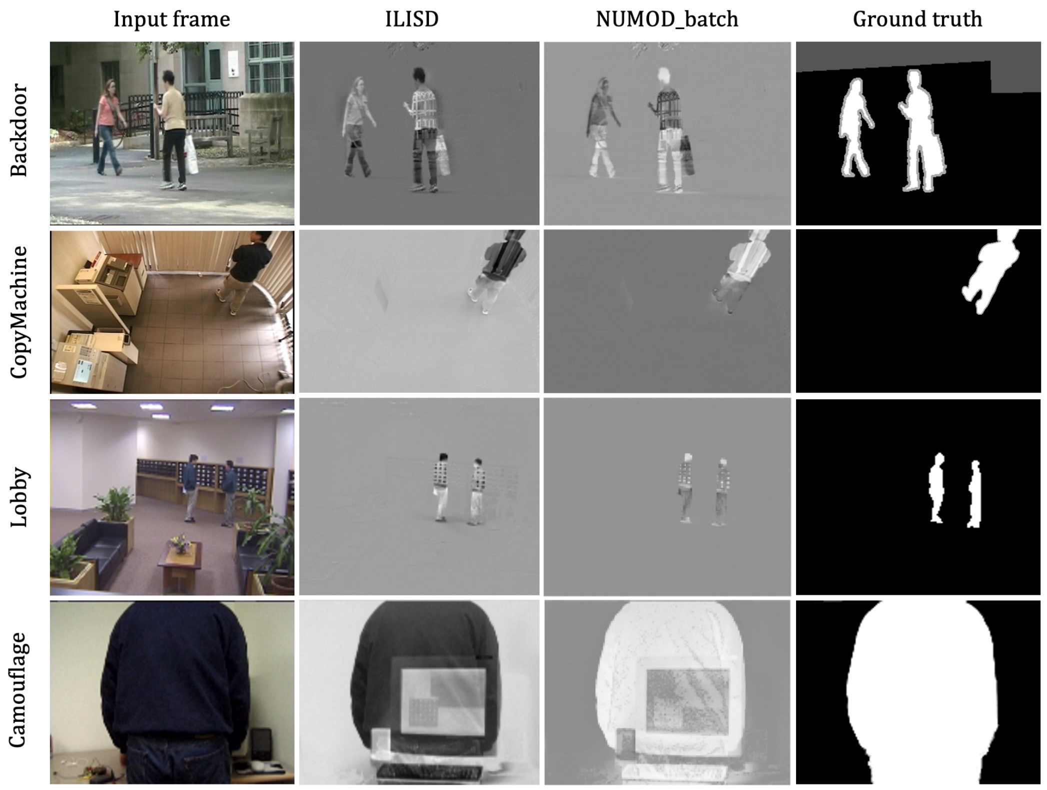


Figure 3.5: Comparison of ILISD and NUMOD-batch qualitative results on backdoor, CopyMachine, Lobby and Camouflage sequences. Columns from left to right are input frames, ILISD foreground image, NUMOD-batch foreground image, F_i, in grayscale and ground truth images, respectively

3.4.2 Evaluation of NUMOD Batch Mode

In the first set of experiments, we compare batch mode of NUMOD to competing batch methods. These methods include some of the best low rank and sparse decomposition methods SSGoDec[109], PRMF[96], Decolor[110], PCP[17], BRMF[97], LSD[59] and ILISD[82]. ILISD is the method that can handle illumination change and moving shadows and comes closest to NUMOD in terms of the objective function. We also compare our method to BEN-BLN [102], which is based on stacked autoencoders.

Performance comparison of batch methods is presented in table 3.1 in terms of F-measure. We observe that NUMOD outperforms other methods in most of the sequences. In “Camouflage” sequence, a person suddenly appears in a large portion of the scene and causes illumination change. For “LightSwitch”

Table 3.1: Performance comparison of batch methods based on F-measure score

Sequence	SSGoDec[109]	PRMF[96]	Decolor[110]	PCP[17]	BRMF[97]	LSD[59]	ILISD[82]	BEN-BLN[102]	NUMOD-Batch
Backdoor	0.6611	0.7251	0.7656	0.7594	0.6291	0.7603	0.8150	0.6212	**0.8536**
CopyMachine	0.5401	0.6834	0.7511	0.6798	0.3293	0.8174	0.8179	0.5518	**0.8519**
Cubicle	0.3035	0.3397	0.5503	0.4978	0.3746	0.4232	0.6887	0.4868	**0.7468**
PeopleInShade	0.2258	0.5163	0.5559	0.6583	0.3313	0.6168	0.8010	0.6802	**0.8661**
Camouflage	0.6452	0.6048	0.8125	0.3388	0.6048	0.9456	0.8663	**0.9552**	0.9224
LightSwitch	0.3804	0.2922	0.5782	**0.8375**	0.2872	0.6640	0.7128	0.0868	0.7761
Lobby	0.0831	0.6256	0.7983	0.6240	0.3161	0.7313	0.7849	0.4691	**0.8355**

and "Camouflage" sequences, ground-truth is available just for one frame and so the results are not based on the performance over the whole sequence. Generally, the results demonstrate capability of our method in separating illumination and real moving objects.

In Fig. 3.5 we show samples for comparing qualitative results between ILISD and NUMOD-batch. The shown results are the decomposed foreground part, called F_i in our notations, in both methods. Since ILISD result is in grayscale, we also show our result in grayscale to make comparison easier. We selected ILISD for comparison since it is the only batch method capable of handling severe illumination changes and comes closest to NUMOD. As it can be seen, these methods produce comparable results.

3.4.3 Evaluation of NUMOD Online Mode

In this set of experiments, we compare online mode of NUMOD to competing online/sequential methods including the GMM method [83] and some of the best LRS decomposition methods GRASTA[39], OR-PCA[31], COROLA[80] and online mode of BEN-BLN [102] neural network based algorithm [102].

Performance of each method based on F-measure is reported in table 3.2. Our proposed method obtains the best results in all sequences except "LightSwitch". As we mentioned earlier, results for this sequence are based on only a single ground-truth frame. Numerical results shows that our method outperforms other online methods and is able to handle illumination changes.

In Fig. 3.6 we compare qualitative results of online methods on "LightSwitch", "Lobby" and "Backdoor" sequences. The results from other methods is their sparse output and the result of our method is the the moving foreground output, F_i in grayscale. In the "LightSwitch" sequence there is a sudden and heavy illumination change. The third row shows all other methods fail to handle illumination change except NUMOD. The first and fourth rows demonstrates that NUMOD is able to obtain a illumination-free foreground when no moving object is in the scene.

The "Lobby" sequence has a moderate illumination change. in this sequence, referring to Fig. 3.6, the only method that is able to distinguish

Table 3.2: Performance comparison of online methods based on F-measure score

Sequence	GMM[83]	GRASTA[39]	OR-PCA[31]	COROLA[80]	BEN-BLN-Online[102]	NUMOD-Online
Backdoor	0.6512	0.6822	0.7360	0.6821	0.5539	**0.8394**
CopyMachine	0.5298	0.6490	0.5599	0.4155	0.5197	**0.8589**
Cubicle	0.3410	0.4113	0.5998	0.5213	0.4164	**0.7473**
PeopleInShade	0.3305	0.5288	0.6088	0.2474	0.7380	**0.8671**
Camouflage	0.8102	0.6528	0.0823	0.7138	0.8311	**0.9117**
LightSwitch	0.4946	0.5631	**0.8006**	0.2830	0.2707	0.7518
Lobby	0.3441	0.6727	0.5831	0.7641	0.3380	**0.7687**

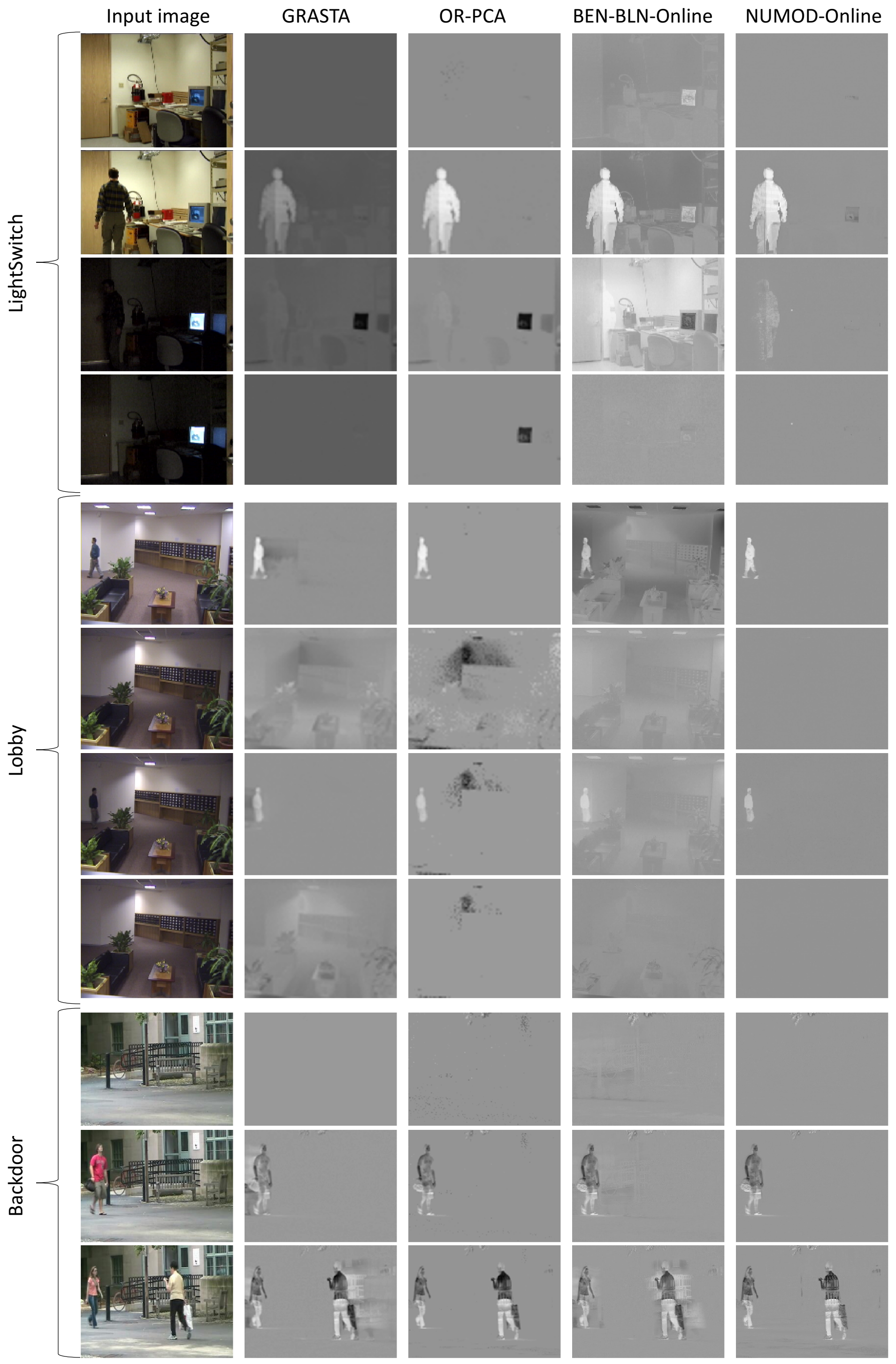


Figure 3.6: Qualitative results for online methods on “LightSwitch”, “Lobby” and “Backdoor” sequences. Columns from left to right are input image, sparse output of GRASTA, OR-PCS and BEN-BLN-Online algorithms and moving foreground output,F_i, of NuMOD-online in grayscale, respectively.

moving foreground and illumination changes is NUMOD. GRASTA and OR-PCA are doing a good job in some frames but in other frames their foreground contains illumination changes. BEN-BLN is not able to capture the illumination changes in its background and as it is shown it appears in the foreground. Once more, we see in “Lobby” sequence results that only NUMOD’s foreground does not include illumination changes in frames that do not have any moving objects.

The last three rows in Fig. 3.6 shows results for “Backdoor” sequence. In this sequence, OR-PCA and NUMOD are doing a good job. However, in the frame without moving objects, OR-PCA captures some noises in its result.

In general, the quantitative and qualitative results demonstrate that NOMOD online mode can outperforms other methods in handling illumination changes.

As an example execution time, frames per second for ’PeopleInShade’ sequence, which size of each image is (234, 370, 3), is 3.230 in batch mode and 4.939 in online mode on a computer with one GEFORCE GTX 1080 Ti GPU card.

3.5 Conclusions and Future Work

In this chapter, we proposed a MOD method called NUMOD based on a generative neural network model. In NUMOD, unsupervised generative networks learn low-dimensional latent representations of the high dimensional background images. For separating illumination changes from foreground moving objects, NUMOD uses an illumination invariant representation of images. The method works well both in the online and the batch modes. To the best of our knowledge, NUMUD is the only online method that can handle illumination changes quite successfully. Based on the quantitative and qualitative experimental results, our method outperforms batch and online methods in most of the bench mark image sequences.

Chapter 4

Dynamic Background Subtraction by Generative Neural Networks

4.1 Introduction

Background subtraction is an effective approach for change detection problem that is a fundamental task in computer vision applications, such as video surveillance, autonomous navigation, traffic monitoring and Human computer interaction [12], [34]. Different methods have been proposed for background subtraction, however many of these methods are vulnerable to image sequences with dynamic background. In a scene with dynamic background, a background pixel can have different values due to periodical or irregular movements [103]. For example, surging of water, water fountains and waving trees cause dynamic variations in the background. Segmenting such dynamic background variations from foreground is a challenging task and negatively affects the methods' performance.

In this chapter, we propose a Dynamic Background Subtraction by Generative neural networks (DBSGen) [4]. DBSGen exploits a generative multi-resolution convolutional network to estimate a dense motion map that minimizes the difference between each input image and a fixed image. The fixed image is chosen from the video as an initial background model. Next, our method warps each input image using its pixel-wise motion map. In the warped images, most of pixels due to the dynamic motions are mapped to pixels of the fixed image. However, some moving objects are also warped in the process. Subsequently, DBSGen leverages a generative fully connected network to generate background images for the warped input images. Then, foreground images are obtained by subtracting background images from warped images. Afterwards, an inverse warping of the motion map is applied on the foreground images to warp back the moving objects, otherwise, results would contain deformed objects. Then, inspired by SuBSENSE method [19], DBSGen computes a pixel-wise dynamic entropy map that is an indicator of dynamic background spots. By utilizing this map, a pixel-wise distance threshold is achieved. Next, DBSGen obtains binary segmented images using the distance threshold. Finally, some basic post-processing operations enhance the results. A block diagram of DBSGen is presented in Fig. 4.1.

DBSGen's contributions can be summarized as follows. First, it estimates a pixel-wise motion map by a generative network and exploits it for dynamic background subtraction problem. Second, unlike many other neural network based methods, it is optimized in an unsupervised way, without requiring expensive pixel-wise ground-truth masks. Third, it is an end-to-end neural network framework, which is optimized in one stage.

The rest of the chapter is organized as follows. Section 4.2 explains details of DBSGen framework and how it performs dynamic background subtraction. In section 4.3, we report our implementation details, experimental results and comparison with state-of-the-art methods. Finally, Section 4.4 provides conclusions and an outline of the future work.

4.2 Proposed Method

DBSGen is based on dynamic motion removal, background generation and pixel-wise thresholding. Optimizations of the networks are performed in an end-to-end manner by stochastic gradient descent. In the following subsections, the description of each of these steps is given.

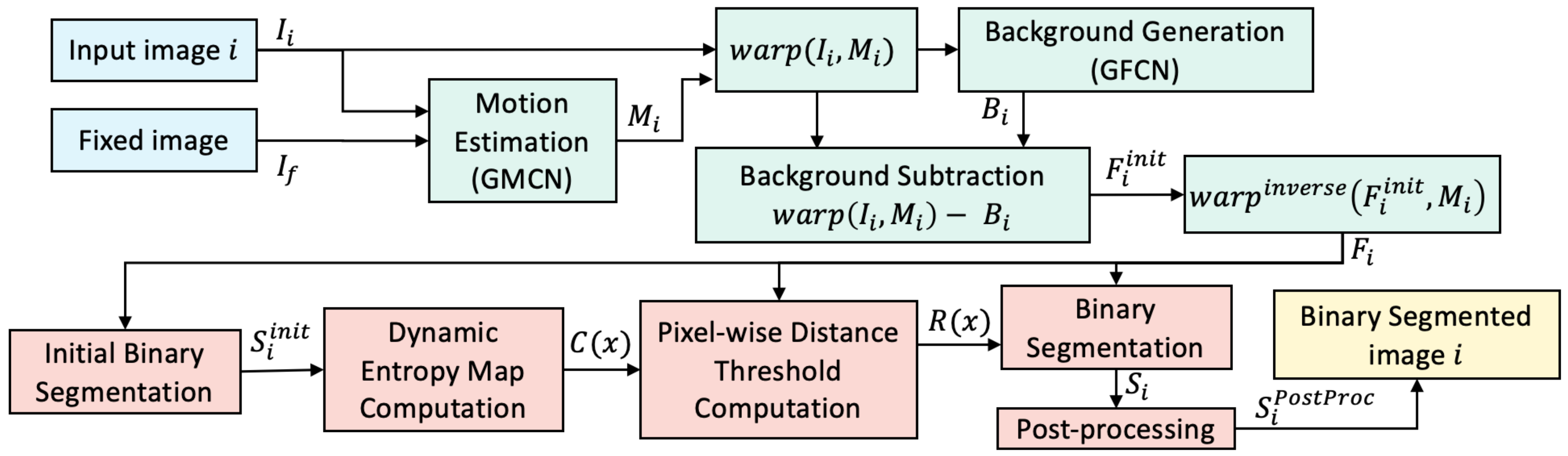

Input image i
I_i
Fixed image
I_f
Motion Estimation (GMCN)
M_i
$warp(I_i, M_i)$
Background Generation (GFCN)
B_i
Background Subtraction $warp(I_i, M_i) - B_i$
F_i^{init}
$warp^{inverse}(F_i^{init}, M_i)$
F_i
Initial Binary Segmentation
S_i^{init}
Dynamic Entropy Map Computation
$C(x)$
Pixel-wise Distance Threshold Computation
$R(x)$
Binary Segmentation
S_i
Post-processing
$S_i^{PostProc}$
Binary Segmented image i

Figure 4.1: Block diagram of DBSGen. Input images and fixed images are shown in blue. The end-to-end modules are depicted in green. Foreground segmentation modules and final binary segmented results are represented in pink and yellow, respectively.

4.2.1 Motion Estimation

By estimating a pixel-wise motion map, DBSGen aims to warp each input image such that it becomes similar to a fixed image. It helps to remove some of the dynamic background motions in the warped input images. For this purpose, we use a Generative Multi-resolution Convolutional Network (GMCN) that generates motion maps in three coarse-to-fine resolutions. We utilize it for estimating small motions including dynamic background motions in the input frames by applying a motion compensation loss. Fig. 4.2 shows the GMCN's architecture used in DBSGen.

The input to GMCN is an optimizable latent tensor with size $N \times H/8 \times W/8$, where N is the number of the frames in the sequence and H and W are the height and width of each image, respectively. GMCN computes 2D motion estimation maps in three resolutions called $M_i^{1/4}$, $M_i^{1/2}$ and M_i that are used to warp the i^{th} input frame of the sequence, I_i, and reduce dynamic background motions. The upsampled motion map of each resolution is added to the the higher resolution's motion map to refine it. In each resolution, a loss term is responsible for minimizing difference of the warped input frame and the fixed image. L_{motion} loss term, optimizes parameters of the GMCN.

$$\begin{aligned}
L^{1/4} &= \sum_{i=1}^{N} \|warp(I_i^{1/4}), M_i^{1/4}) - I_f^{1/4}\|_2, \\
L^{1/2} &= \sum_{i=1}^{N} \|warp(I_i^{1/2}, M_i^{1/2}) - I_f^{1/2}\|_2, \\
L^{1} &= \sum_{i=1}^{N} \|warp(I_i, M_i) - I_f\|_2, \\
L_{motion-reg} &= \sum_{i=1}^{N} \|M_i^{1/4}\|_2 + \|M_i^{1/2}\|_2 + \|M_i\|_2, \\
L_{motion} &= L^{1/4} + \lambda L^{1/2} + \lambda^2 L^1 + L_{motion-reg},
\end{aligned} \tag{4.1}$$

where $\|\cdot\|_2$ denotes the L_2-norm and I_f represents a background image selected from one or an average of a few frames without a moving objects from the input sequence. Function $warp(I_i, M_i)$ warps the image I_i with the pixel-wise

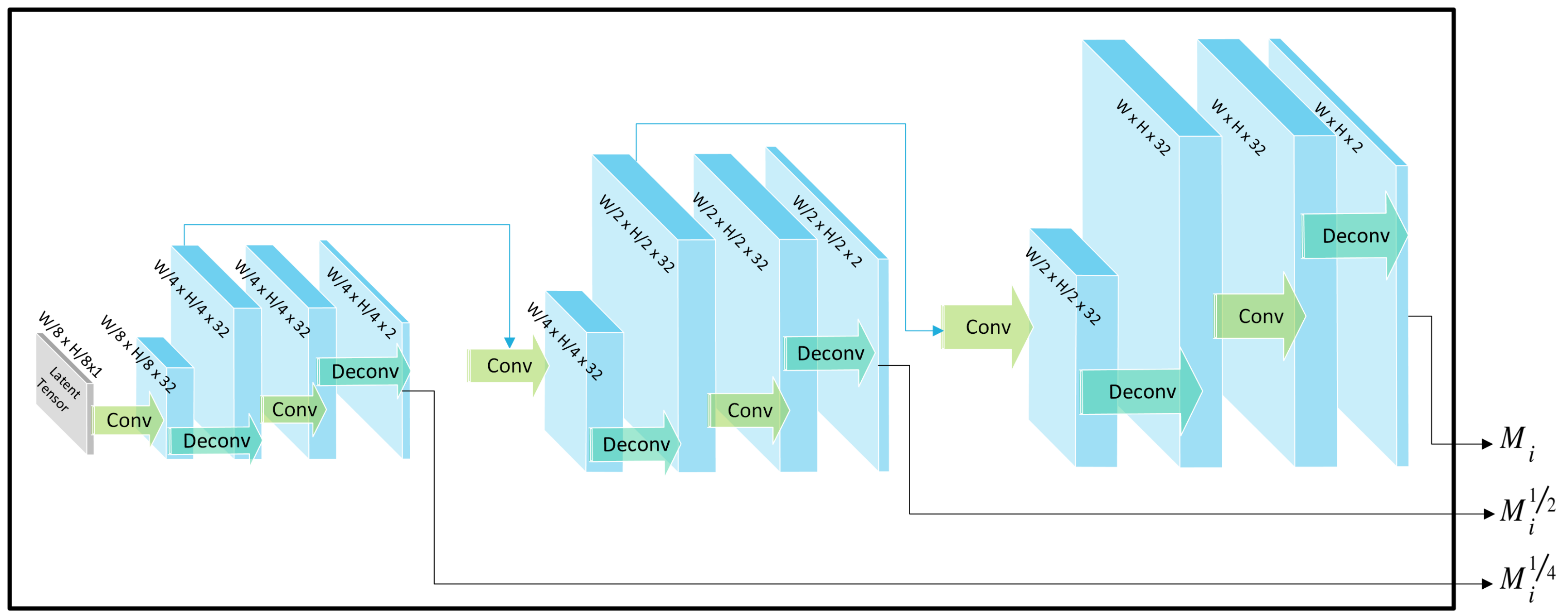


Figure 4.2: Architecture of the Generative Multi-resolution Convolutional Network (GMCN). Input to the network is an optimizable latent tensor. Outputs are dense motion maps in three resolutions.

motion map M_i. λ is a hyper-parameter to control relative importance of the terms and its value is chosen by experiments. $L_{motion-reg}$ is a regularization term for motion maps that does not allow estimated motion values grow large. Although, we do it to avoid warping of moving objects still some motions of foreground moving objects are captured in the motion map and as a result, they get warped. That is why DBSGen applies an inverse warping, based on motion maps, on foreground images, in a later step.

4.2.2 Background Generation

Background is generated by a Generative Fully Connected Network (GFCN) that has an optimizable low-dimensional latent vector as the input. The input layer is followed by two fully connected hidden layers each connected to a batch normalization layer and an ELU [27] activation function. The last layer is a fully connected one with Sigmoid activation function to limit output values between zero and one. GFCN architecture is shown in Fig. 4.3.

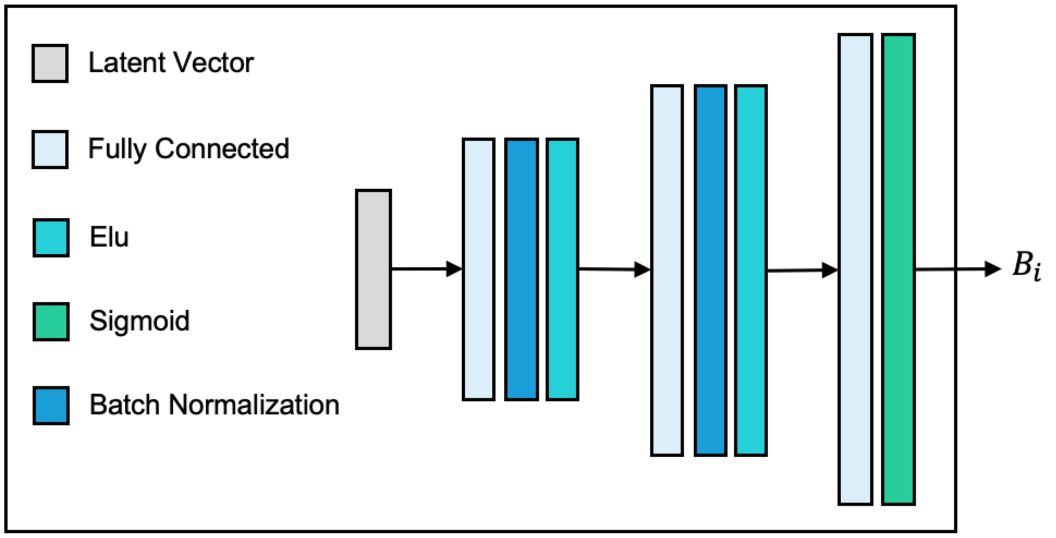


Figure 4.3: Generative Fully Connected Network (GFCN)

L_{recons} loss term that is responsible for constructing background images is as follows:

$$L_{recons} = \sum_{i=1}^{N} \|warp(I_i, M_i) - B_i\|_1, \tag{4.2}$$

where B_i is the i^{th} output of GFCN and M_i is the obtained motion map from GMCN. $\|.\|_1$ denotes the L_1-norm. We used L_1-norm instead of L_2-norm in L_{recons} because it encourages sparsity [16].

GFCN behaves like a decoder in an autoencoder network with the difference that here, the input to the decoder is an optimizable latent vector, which can learn a low-dimensional manifold of the data distribution by applying some constraints like limiting the capacity of the network and choosing a small input latent vector size [36]. Since The network is able to extract the most salient features of the data and L_{recons} loss term is imposing similarity of output and input frames, therefore, during optimization, GFCN learns a background model. This happens because the sequence of input images are temporally correlated to each other and the background part of images are common among them [6]. The overall loss function of DBSGen is defined as:

$$L = \alpha L_{recons} + L_{motion} + L_{reg}, \tag{4.3}$$

where L_{reg} is the L_2 regularization that we apply on parameters of the networks to prevent overfitting to noise. α is a hyper-parameters to take into account relative importance of L_{recons} term and is determined by conducting experiments.

4.2.3 Foreground Segmentation

For obtaining foreground part of the images, F_i^{init}, our method subtracts the obtained background image from the warped input image. Then, it applies an inverse warping on the result to warp the moving objects back to their original shape and acquires foreground, F_i as follows:

$$\begin{aligned} F_i^{init} &= warp(I_i, M_i) - B_i, \\ F_i &= warp^{inverse}(F_i^{init}, M_i). \end{aligned} \tag{4.4}$$

For obtaining the foreground mask, we use a pixel-wise thresholding method. This is adopted from SuBSENSE method [19] for detecting blinking pixels by measuring the dynamic entropy of each pixel. $C(x)$, dynamic entropy map, counts the number of times a pixel switches from being a foreground to a background or vice versa between consequent frames and is computed as follows:

$$C(x) = \frac{1}{N-1}\sum_{i=2}^{N} XOR(S_i^{init}(x), S_{i-1}^{init}(x)), \quad (4.5)$$

where x is a pixel and S_i^{init} is the binary result of the i^{th} frame in the sequence after an initial segmentation. This initial segmentation uses the standard deviation of all foreground frames, F, in each color channel as the distance threshold. Note that these three threshold values for RGB channels are same among all frames. Values of dynamic entropy map, C, are in the range $[0,1]$, where dynamic background regions would have greater values, while static background regions would have $C(x) \approx 0$. Dynamic entropy map of "fountain01" and "fall" videos can be observed in Fig. 4.4.

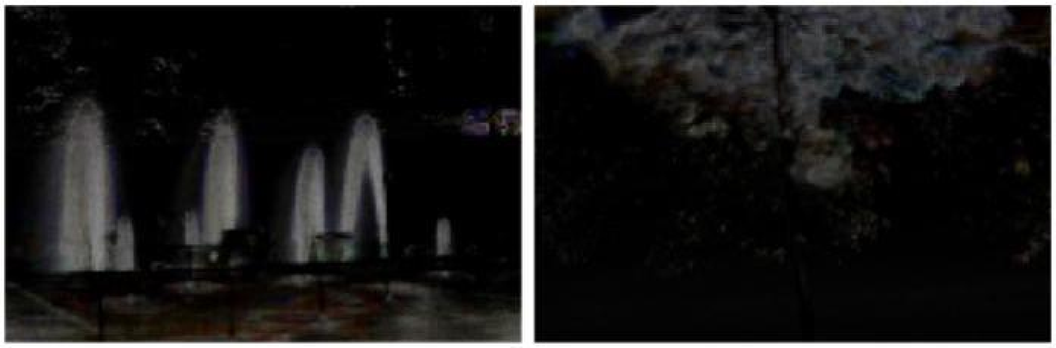


Figure 4.4: Dynamic entropy map, $C(x)$, of "fountain01" and "fall" videos

In the following step, we compute the pixel-wise distance thresholds:

$$R(x) = \mu_{ch} + \beta_1\sigma_{ch} + \beta_2\sigma_{ch}C(x) + \beta_3\sigma^2_{C_{ch}}C(x), \quad (4.6)$$

where μ_{ch} and σ_{ch} are the mean and standard deviation of the foreground frames F in each color channel, respectively, and $\sigma^2_{C_{ch}}C(x)$ is the variance of the counter C in each color channel. The binary segmented result, S_i, is obtained by applying $R(x)$ distance threshold on the foreground $F_i(x)$.

Our post-processing step is minimal like other state-of-the-art methods [19], [64]: we apply a median blur filter and binary morphological closing on S_i to eliminate salt-and-pepper noise. The final binary segmented result is called $S_i^{PostProc}$.

4.3 Experimental Results and Discussion

4.3.1 Implementation Details

DBSGen is implemented in TensorFlow platform. GFCN has an optimizable vector of size 3 as its input and three fully connected layers of sizes 12, 24, and 43, successively. Convolutional and deconvolutional layers in GMCN each have 32 filters of size 7×7. Values of hyper-parameters λ and α are set to 0.25 and 0.1, respectively, by conducting several trial and error experiments. Adam [50] with learning rate of 0.006 is used as the optimization algorithm. The whole framework is optimized in 50 epochs in an end-to-end fashion. The average speed of DBSGen on Dynamic Background category of CDnet 2014 [99] is about 33 frames per second on a GeForce GTX 1080 Ti GPU.

4.3.2 Dataset and Evaluation Metric

We evaluate DBSGen on videos of Dynamic Background category of change detection (CDnet 2014) dataset [99] to validate its effectiveness in challenging dynamic background scenarios. It includes six videos; “fountain01” and “fountain02” contain dynamic water background, also, “canoe” and “boats” videos exhibit water surface motion, while “overpass” and “fall” videos have waving trees in their background.

For evaluation, we use F-Measure (FM) metric, defined in 2.2.3, that is used generally as an overall performance indicator of the moving object detection and background subtraction methods. To ensure consistency with existing methods, all the evaluations are computed as defined in [99].

4.3.3 DBSGen Results

Qualitative results of DBSGen can be observed in Fig. 4.5. Each row shows the intermediate and final results for one frame of each video. Columns show input frames, difference of the input frames and the fixed image, the obtained foreground images, the binary segmented results, the post-processed segmented results and ground-truths, successively. Comparison between the second and third columns illustrates DBSGen was able to remove dynamic background

Table 4.1: Performance comparison of DBSGen with or without motion estimation component, as well as with or without post-processing, based on F-measure score.

Motion estimation	Post-processing	fountain01	fountain02	canoe	boats	overpass	fall	Average
No	No	0.16	0.45	0.77	0.34	0.59	0.62	0.49
Yes	No	0.66	0.76	0.86	0.76	0.78	0.86	0.78
No	Yes	0.27	0.74	0.82	0.87	0.79	0.82	0.72
Yes	Yes	0.73	0.80	0.90	0.91	0.87	0.93	0.86

noise to an acceptable level, before pixel-wise thresholding. Additionally, the pre-post-processing results, in the fourth column, demonstrate that DBSGen, even without the help of post-processing operations, is capable of handling dynamic background challenge to a good extent by its pixel-wise distance threshold, $R(x)$, based on dynamic entropy map, $C(x)$. The final results, in the fifth column, show DBSGen eliminates dynamic background noise successfully.

To evaluate effectiveness of the motion estimation component of DBSGen, we omitted GMCN and L_{motion} that are responsible for removing some dynamic background motions by warping. The obtained results, reported in Table 4.1 in terms of FM, indicate motion estimation component plays an important role in our method and positively affects the performance of DBSGen. Table 4.1 also includes results with and without post-processing as reference points. Comparison between the second and fourth rows, where motion component is not removed, proves DBSGen's performance without post-processing step dose not drop drastically .

4.3.4 Comparison

For comparison, we chose the top 30 methods which had the best performance in terms of F-measure on Dynamic Background category of CDnet 2014 challenge results [99] listed on ChangeDetection.net website. The supervised methods and ensemble method IUTIS, that combines several algorithms, [9] are not considered. In addition, CANDID algorithm [64], that was specifically proposed for dynamic background subtraction, is also considered.

The quantitative results are presented in Table 4.2, where all methods are sorted based on their average FM over all videos, listed in the last column. DBSGen results are reported in the last row. As visible through last column, DBSGen achieves an average of 0.86 in terms of FM and outperforms most of the top-ranked methods. It is only surpassed by FTSG [98] and PAWCS [20] methods. In the "fall" video, we obtain the best performance along with FTSG. Note that DBSGen does not yield low accuracy in *any* of the videos unlike GMM methods [83], [111], KDE [30] and SOBS_CF [62] that do not get satisfactory results on the "fountain01" video.

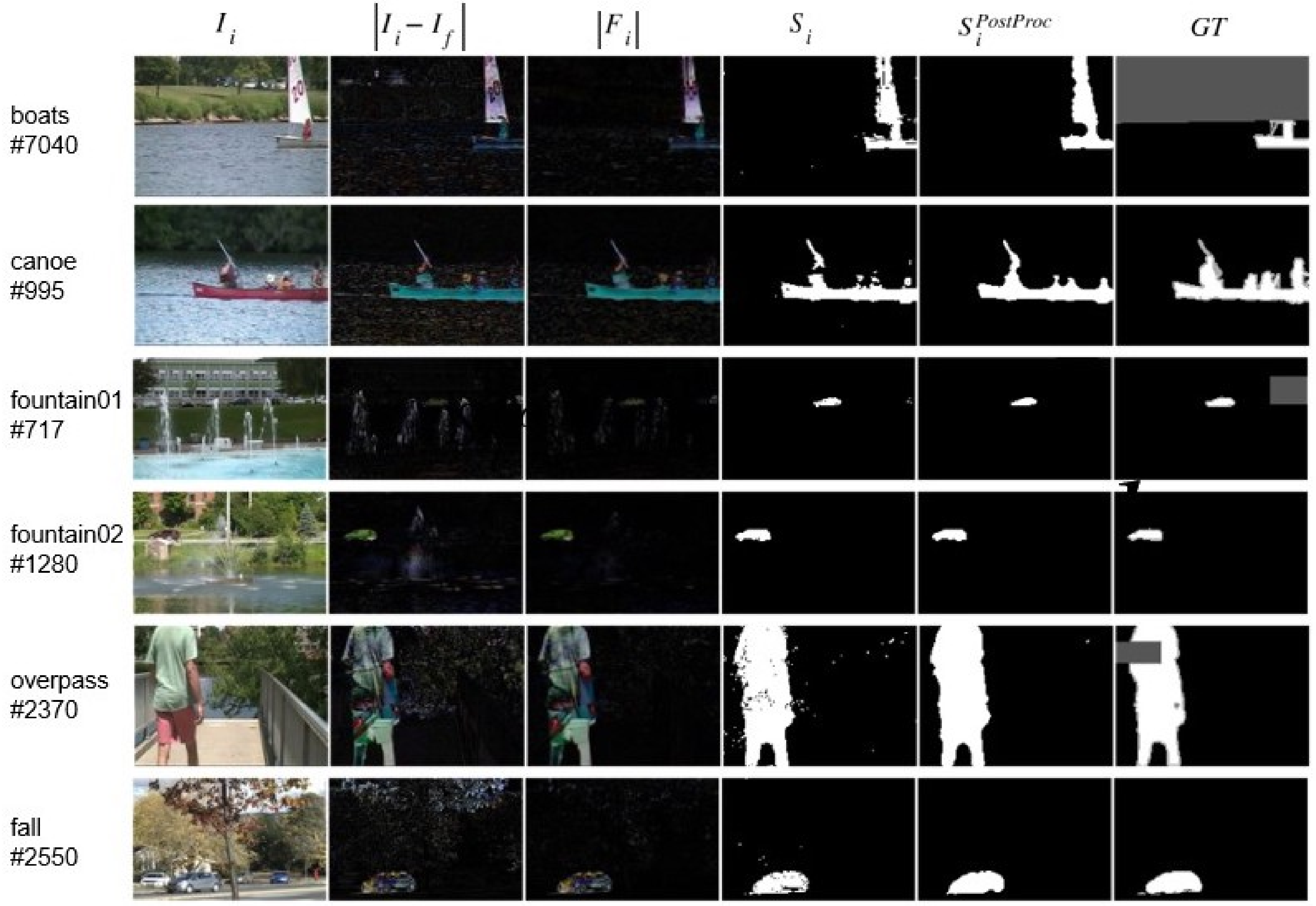


Figure 4.5: Qualitative results of consecutive steps of DBSGen. In each row, columns from left to right show an input frame of a video, difference of the input frame and the fixed image, the obtained foreground, the binary segmented result, the post-processed segmented result, and the ground-truth.

Table 4.2: Performance comparison of the top-ranked methods, evaluated on CDnet 2014 Dynamic Background category, in terms of F-measure. The best performance achieved, in each column, is shown in bold.

Methods	fount01	fount02	canoe	boats	overpass	fall	Avg
CL-VID [60]	0.05	0.45	0.93	0.81	0.85	0.23	0.55
C-EFIC [2]	0.27	0.34	0.93	0.37	0.90	0.56	0.56
EFIC [1]	0.23	0.91	0.36	0.36	0.88	0.72	0.58
Multiscale ST BG[61]	0.14	0.82	0.48	0.89	0.84	0.41	0.60
KDE [30]	0.11	0.82	0.88	0.63	0.82	0.31	0.60
CP3-online [56]	0.54	0.91	0.63	0.17	0.64	0.77	0.61
DCB [52]	0.40	0.83	0.45	0.87	0.83	0.30	0.61
GMM_Zivkovic [111]	0.08	0.79	0.89	0.75	0.87	0.42	0.63
GMM_Stauf-Grim [83]	0.08	0.80	0.88	0.73	0.87	0.44	0.63
SOBS_CF [62]	0.11	0.83	**0.95**	0.91	0.85	0.26	0.65
SC_SOBS [63]	0.12	0.89	**0.95**	0.90	0.88	0.28	0.67
AAPSA [71]	0.44	0.36	0.89	0.76	0.82	0.75	0.67
M4CD Version 2.0 [95]	0.17	0.93	0.61	**0.95**	0.95	0.50	0.69
RMoG [91]	0.20	0.87	0.94	0.83	0.90	0.67	0.74
WeSamBE [46]	0.73	0.94	0.61	0.64	0.72	0.81	0.74
Spectral-360 [77]	0.47	0.92	0.88	0.69	0.81	0.90	0.78
MBS Version 0[73]	0.52	0.92	0.93	0.90	0.90	0.57	0.79
MBS [74]	0.52	0.92	0.93	0.90	0.90	0.57	0.79
BMOG [65]	0.38	0.93	**0.95**	0.84	**0.96**	0.69	0.79
CANDID [64]	0.55	0.92	0.91	0.67	0.92	0.81	0.80
SBBS [92]	0.73	0.93	0.49	0.94	0.91	0.88	0.81
SuBSENSE [19]	0.75	0.94	0.79	0.69	0.86	0.87	0.82
SharedModel [23]	0.78	0.94	0.62	0.88	0.82	0.89	0.82
CwisarDH [28]	0.61	0.93	0.94	0.84	0.90	0.75	0.83
WisenetMD [54]	0.75	**0.95**	0.87	0.71	0.87	0.87	0.84
AMBER [94]	0.77	0.93	0.93	0.85	0.95	0.63	0.84
CwisarDRP [29]	0.69	0.92	0.91	0.84	0.92	0.82	0.85
CVABS [45]	0.77	0.94	0.88	0.81	0.86	0.91	0.86
SWCD [44]	0.76	0.93	0.92	0.85	0.85	0.88	0.86
FTSG [98]	**0.81**	**0.95**	0.69	**0.95**	0.94	**0.93**	0.88
PAWCS [20]	0.78	0.94	0.94	0.84	**0.96**	0.91	**0.89**
DBSGen	0.73	0.80	0.90	0.91	0.87	**0.93**	0.86

4.4 Conclusion

We have presented a generative neural net based background subtraction method called DBSGen to handle dynamic background challenge. DBSGen is unsupervised, so it does not need annotated ground-truth. Furthermore, it gets optimized in an end-to-end way. Besides, it has a minimal post-processing step, which can be also omitted without a significant performance drop. DBSGen estimates a dense dynamic motion map by use of a Generative Multi-resolution Convolutional Network (GMCN) and warps the input images by the obtained motion map. Then, a Generative Fully Connected Network (GFCN) generates background images by using warped input images in its reconstruction loss term. In the following step, a pixel-wise distance threshold that utilizes a dynamic entropy map obtains the binary segmented results. Finally, a basic median filter and morphological closing is applied as the post-processing step. Experiments on Dynamic Background category of CDnet 2014 demonstrates that DBSGen surpasses all previously tested methods, which are unsupervised and not ensemble of several methods, on CDnet 2014 in terms of F-measure. Only two state-of-the-art methods outperform DBSGen. Overall, quantitative and qualitative results confirm that DBSGen is capable of eliminating dynamic background motions quite effectively.

Chapter 5

Weakly Supervised Realtime Dynamic Background Subtraction

5.1 Introduction

Background subtraction is a crucial problem in computer vision that has practical applications in various domains like video surveillance, human-computer interaction, traffic monitoring, and autonomous navigation [12], [34]. Dealing with dynamic backgrounds is a significant challenge in background subtraction, where a background pixel's value can change due to periodical or irregular movements [103]. Although various methods have been proposed for background subtraction, not all of them can effectively handle sequences with dynamic backgrounds. Scenes with dynamic elements like fountains, waving trees, and water motions are prime examples of dynamic backgrounds. Detecting these dynamic variations as parts of the foreground negatively impacts the performance of the methods.

Because of the effectiveness of deep learning methods in computer vision, numerous neural network models have been developed for the purpose of foreground and background segmentation. These models have ranked highly among the evaluated methods in the *CDnet 2014* dataset. However, they require supervised training, which involves manual annotation at the pixel level. This process is time-consuming and expensive, and may not be practical for every situation.

In recent years, weakly supervised methods have gained popularity and have demonstrated impressive performance in various tasks. One of their primary advantages is that they achieve satisfactory performance without relying on costly pixel-wise ground truth annotations. In the context of background subtraction, two new methods have been proposed by Zhang et al. [107] and Minematsu et al. [66]. Both methods are trained using frame-level labels, which is a less demanding and more cost-effective labeling approach compared to pixel-level annotation.

In this chapter, we present a new approach to background subtraction [5] that learns the dynamic background component in a weakly supervised manner. It uses a fully connected autoencoder and a U-Net convolutional neural network [72]. To explain the overall working principle, let us first consider the scenario where no moving object is present in a sequence. In this case, the autoencoder takes in the sequence and produces the static background images. The difference between the input image and the output of the autoencoder contains dynamic clutter. The U-Net takes in the same sequence of images and is expected to produce only the dynamic clutter, which in this paper is referred to as the dynamic background image. So, when we subtract the autoencoder output from the input image and multiply it with the inverted output of the U-Net, we ideally obtain a zero image showing no moving objects or dynamic background. In the second scenario, when moving objects are present, the autoencoder output again contains static background and the output of the U-Net is still expected to produce only a dynamic background image. So, when we subtract the output of the autoencoder from the input image and multiply it with the inverted output of the U-Net, it will show the moving objects only.

The autoencoder is trained on a moving object-free sequence to produce static background images that capture some of the temporal and spatial variations in the scene. We obtain a binary dynamic background image by subtracting autoencoder output from the input image and applying a threshold. Then, We train the U-Net on the same object-free sequence using the binary dynamic background images from the previous step as the target output for the U-Net. Thus, the U-Net learns the temporal and spatial variations of the dynamic background in the scene.

During the testing phase, the autoencoder and U-Net generate the static and dynamic background images, respectively. Multiplying the inverted dynamic background with the static background-subtracted image produces the foreground image. Finally, we apply pixel-wise thresholding to the foreground image and use some simple post-processing techniques to enhance the final result. Fig. 5.1 illustrates an overview of our proposed method.

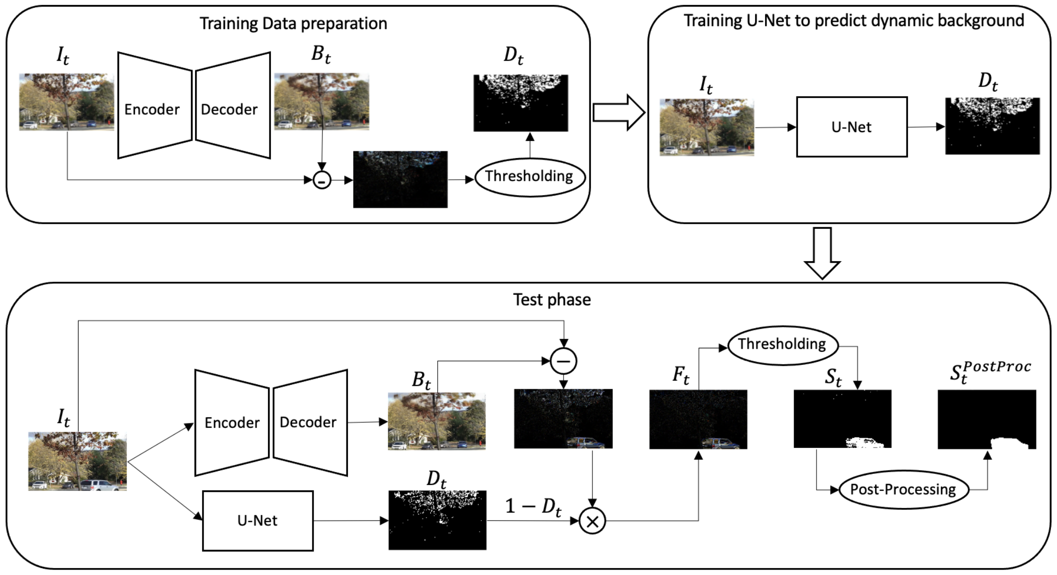


Figure 5.1: Overview of the proposed method. The top two boxes show the training phase and the below box shows the test phase. To handle dynamic background, our method combines an autoencoder and a U-Net. The autoencoder and U-Net are trained to generate the static and dynamic background images, respectively. By subtracting the autoencoder's output from the input image and multiplying it with the inverted output of the U-Net, we can obtain an image that shows only the moving objects, free from dynamic artifacts.

Our proposed approach offers several key contributions. First, it is a practical and cost-effective weakly supervised framework, which eliminates the need for costly pixel-wise annotations by using only an object-free training sequence. Second, it effectively learns and predicts the dynamic background component in each image and segments it from the foreground. Finally, our experimental results show that the proposed algorithm achieves superior performance in dynamic background scenes compared to other state-of-the-art unsupervised and weakly supervised methods, both quantitatively and qualitatively.

This chapter is structured as follows: Section 5.2 reviews other weakly supervised methods for background subtraction. Section 5.3 provides a detailed explanation of our framework, including its training and test phases. In Section 5.4, we present our implementation details, and experimental results, and compare them with state-of-the-art methods. Finally, Section 5.5 concludes the chapter with a summary of our findings and future research directions.

5.2 Related Weakly Supervised Methods

In recent years, some weakly supervised methods have emerged that solely rely on image-level tags, which indicate whether a foreground object is present in the image [66], [107]. The method proposed in [66] generates a binary mask image by subtracting a background image from an input image to identify foreground regions. It then uses intermediate feature maps of a CNN to refine the foreground locations. However, image-level supervision presents a challenge due to the lack of location information in training the network. To address this, the method introduces some constraints that help to locate foreground pixels.

Another recent technique, LDB [107], adopts a tensor-based decomposition framework to represent the background as a low rank tensor and classify the sparse noise as foreground. Additionally, it trains a two-stream neural network using an object-free video to explicitly learn the dynamic background. The dynamic background component of LDB leads to a more precise decomposition of the background and foreground, making it the current state-of-the-art method in weakly supervised moving object detection, particularly in dynamic background scenes.

Our proposed method is also based on explicit modeling of the dynamic background using a neural network. However, our framework differs from LDB in that it relies exclusively on neural networks for the segmentation of the background and foreground, rather than using a low rank-based approach.

As a result, we gain significant advantages in reducing the running time once our networks are trained. Further, LDB uses a very light CNN to model dynamic background. Instead, we use a U-Net to model the same. Because of a large number of parameters, U-Net has significant representative power, and it **overfits** the scene sequence to generate the dynamic background. We make use of this overfitting, because for a sequence containing moving objects the U-Net should ignore the moving objects and output only the dynamic background.

5.3 Proposed Method

Our proposed method, depicted in Fig. 5.1, is based on two neural networks and is trained in a weakly supervised way. The first network is an autoencoder that generates static background images and is trained in an unsupervised manner. The second network is a U-Net [72], which requires pixel-wise labels for training. Using the background images generated by the autoencoder, ground-truth labels for the U-Net are acquired. In the following sections, we explain the training and testing phases, as well as the roles of each network in our framework.

5.3.1 Background Generation

Our framework uses an autoencoder to generate static background images. Autoencoders are a type of neural network that consists of two components: an encoder and a decoder. The encoder maps the input to a compressed code, and the decoder reconstructs the input from the code, aiming to make the output as close to the input as possible [7]. Consequently, autoencoders learn a compressed and meaningful representation of the input data. This results in the removal of insignificant data and noise from the reconstructed input [93].

The autoencoder used in our method for generating the static background images is a fully connected one with dense layers, each followed by a SELU [51] activation function. The only exception is the last layer, which uses a Sigmoid activation function to limit the output values between zero and one.

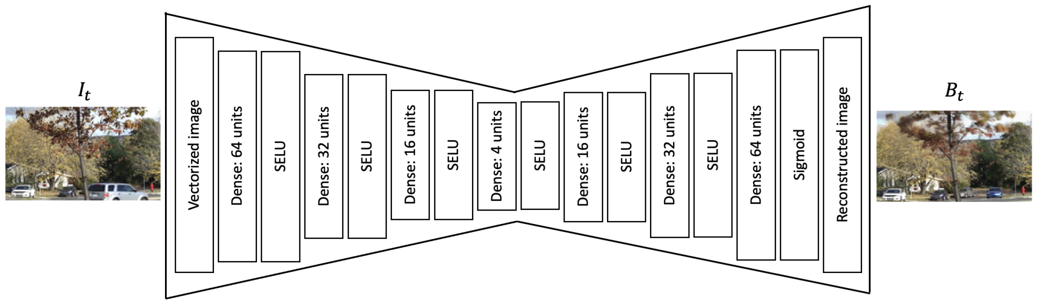


Figure 5.2: The autoencoder architecture has fully connected layers, each followed by a SELU activation function. The last layer has a Sigmoid activation function to limit the output values between zero and one.

Fig. 5.2 shows the details of the architecture of the autoencoder used in our approach.

The L_{recons} loss term, which is responsible for constructing the background images, is defined as follows (Eq. 5.1):

$$L_{recons} = \sum_{t=1}^{N} \|I_t - B_t\|_1, \tag{5.1}$$

Here, I_t and B_t are the t^{th} input and output of the autoencoder, respectively, and $\|\cdot\|_1$ denotes the L_1-norm. We used the L_1-norm instead of the L_2-norm in L_{recons} because it encourages sparsity [16].

The autoencoder can learn a low-dimensional manifold of the data distribution by applying constraints such as limiting the network's capacity and choosing a small code size [36]. The network can extract the most salient features of the data, and the L_{recons} loss term imposes similarity between the input and reconstructed frames, allowing the autoencoder to learn a background model during optimization. This is possible because the input image sequence is temporally correlated, and the background of the images is common among them [6].

5.3.2 Dynamic Background Data Preparation

Autoencoders can be optimized in an unsupervised way and do not require labeled data. However, our second network, U-Net, requires pixel-wise labeled training data to be trained. A moving object-free sequence of input frames is used as training data. During the training phase, the autoencoder generates static background images of the training frames. The static background images are then subtracted from the input images, and after applying a threshold on them, the binary images are obtained. These binary images exhibit a dynamic background since they are extracted from training images without any foreground object. The entire process is illustrated in Fig. 5.1.

5.3.3 Dynamic Background Prediction

The second network in our proposed method is a U-Net, which was originally developed for image segmentation tasks and has shown great success in medical image analysis [72]. As depicted in Fig. 5.3, its architecture resembles a U-shape and consists of two paths: a contracting path and an expansive path. The contracting path is composed of convolutional layers followed by ReLU activation functions and max-pooling layers, where the number of features gets doubled in each contracting step. The expansive path consists of up-convolutional layers for upsampling and halving the number of features, concatenations of features from the contracting path, convolutional layers, and ReLU activation functions. In our method, we used the basic U-Net architecture as described in [72].

We chose U-Net because it produces a pixel-wise binary classification output that is ideal for our task of identifying whether each pixel belongs to the dynamic background class or not. U-Net is a network that requires pixel-wise labels for supervised training. However, in our framework, it is trained using the prepared binary images that were explained in the Section 5.3.2, which makes our method weakly supervised. All we need for training is a moving object-free sequence. In other words, if we have a training sequence with frame-level tags whether the frame contains only background or not, we select only the frames with a tag value of zero as the training data. The output of U-Net is a binary image with pixel values of zero or one, where each pixel labeled as one indicates the presence of dynamic background.

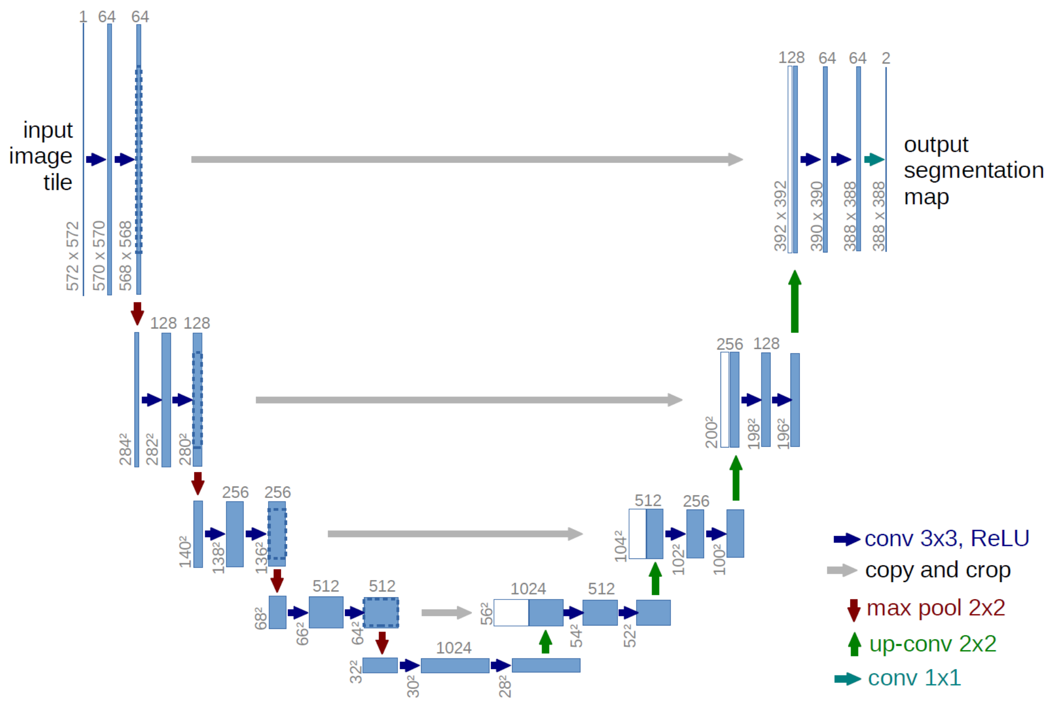


Figure 5.3: U-Net architecture [72]

5.3.4 Training and Test Phases

Training Phase

In the training phase, our proposed method first optimizes the autoencoder on an object-free training sequence to generate static background images. Then, a threshold is applied to the background-subtracted images to obtain dynamic background binary images. The U-Net network is then trained on the same object-free training sequence using the dynamic background binary images as its target output to enable it to predict dynamic background pixels. We train the U-Net model long enough until it overfits to the training sequence. We exploit this overfitting to our advantage because during testing, the U-Net should identify only the dynamic background pixels as label one, while ignoring the moving objects present in the video sequence. All the steps of the training phase are illustrated in Fig. 5.1.

Test Phase

During the test phase, an input test image I_t is fed into the autoencoder to obtain the static background image B_t. The same input image I_t is also fed into the U-Net, which produces a dynamic background image D_t. Next, the background subtracted image, $I_t - B_t$, is multiplied by the inverted dynamic background, $1 - D_t$, as shown in equation 5.2. This step generates the foreground image F_t, which excludes the dynamic background artifacts. Then, a pixel-wise thresholding technique is applied to F_t to obtain the initial segmented image, S_t, as described in the Section 5.3.5. Finally, two standard post-processing techniques, median blurring, and morphological closing are applied to S_t to improve the results, and the final segmented image, $S_t^{PostProc}$, is obtained. The entire process is illustrated in Fig. 5.1.

$$F_t = (I_t - B_t) \times (1 - D_t) \tag{5.2}$$

5.3.5 Foreground Segmentation with Pixel-wise Thresholding

Although most of the dynamic background pixels are detected by the U-Net, there is still a possibility that some of them may be missed due to the selected threshold when preparing the dynamic background ground-truth images. To address this, we use a per-pixel thresholding technique inspired by the SuBSENSE method [19] to obtain the foreground masks. This technique calculates the dynamic entropy of each pixel, represented by the dynamic entropy map $C(x)$, to detect blinking pixels. The dynamic entropy map tracks how often a pixel changes from being a foreground pixel to a background pixel, or vice versa, between consecutive frames.

The calculation of $C(x)$ is based on the XOR operator and is given by:

$$C(x) = \frac{1}{N-1} \sum_{t=2}^{N} XOR(S_t^{init}(x), S_{t-1}^{init}(x)), \tag{5.3}$$

Here, x represents a pixel, S_t^{init} is the binary result of the t^{th} frame in the sequence after an initial segmentation, and N is the total number of frames in the sequence. The initial segmentation uses $\alpha \max(F)$ as the threshold, $\max(F)$ is the maximum value of the foreground frames F, and α is a coefficient. The dynamic entropy map values, C, range from 0 to 1.

In the next step, we compute the pixel-wise distance thresholds using the following equation:

$$R(x) = \beta \max(F) + C(x), \tag{5.4}$$

Here, $\max(F)$ is the maximum value of the foreground frames F, and β is a coefficient. The binary segmented result S_t is obtained by applying the distance threshold $R(x)$ to the foreground $F_t(x)$.

5.4 Experimental Results and Discussion

5.4.1 Implementation Details

Our method was implemented using the Keras deep learning framework [25]. The autoencoder architecture shown in Figure 5.2 consists of densely connected layers with 64, 32, 16, 4, 16, 32, and 64 units, respectively. To avoid the occurrence of exploding and vanishing gradients, all layers utilize the scaled exponential linear unit (SELU) activation function, which possesses self-normalizing properties as described in [51]. The final layer which uses the sigmoid activation function to produce output values within the range of $[0, 1]$.

The U-Net's contracting path consists of four steps, each composed of two convolutional layers with a 3×3 kernel size and ReLU activation function, followed by a 2×2 max pooling operation with a stride of 2. The numbers of features are 64, 128, 256, 512, and 1024 for the top-to-bottom steps. The expansive path mirrors the contracting path but with two differences: first, features of the same contracting level are concatenated to the feature channels. Second, the max pooling operation is replaced with a transposed convolution layer with a 3×3 kernel size and stride 2. Consequently, the number of features is halved in each expansive step. The final layer of the model is a convolutional layer with a 1×1 kernel size and two features that construct the binary output image. Our U-Net architecture has the same design as the basic U-Net proposed in [72], except that we use convolutional and transposed convolution layers with the padding mode set to 'same', which eliminates the need for the cropping operation in [72].

The hyper-parameters α and β were set to 0.2 and 0.08, respectively, after conducting several trial and error experiments. We used the Adam optimization algorithm [50] with learning rates of 0.0001 and 0.005 for the autoencoder and U-Net, respectively. Both networks were trained for 50 epochs. During the testing phase, we achieved an average processing speed of 107 frames per second on the *CDnet 2014* dataset [99] using a GeForce GTX 1080 Ti GPU.

5.4.2 Datasets

We evaluated the effectiveness of our approach on various video datasets to demonstrate its suitability for real-world scenarios.

CDnet 2014

To show the effectiveness of our method in challenging dynamic background scenarios, we conducted evaluations on the *Dynamic Background* category of the *CDnet 2014* dataset [99]. This category consists of six videos with different types of dynamic background motions. The videos "fountain01" and "fountain02" feature a dynamic water background, while "canoe" and "boats" depict water surface movement. "Overpass" and "fall" exhibit waving trees in the background. Additionally, we evaluated our approach on the *Bad Weather* category of the same dataset, which features sparse dynamic variations in the background caused by snow and rain, making it a challenging category. The four videos in this category are "blizzard", "skating", "snowFall", and "wetSnow".

We manually selected the frames without objects for the training data for each sequence. These frames were chosen from the frames before the starting frame in the temporal ROI. In the case of the "WetSnow" video, no background images were available in the initial frames of the sequence. Therefore, we chose 10 object-free frames from the sequence after frame number 2000. We used a maximum of 300 frames for training, or the number of available frames, whichever was less.

I2R Dataset

The *I2R* dataset [55] is a widely recognized benchmark for background subtraction tasks, consisting of 10 real videos shot in indoor and outdoor settings. These videos contain challenging conditions like bootstrapping, shadows, camouflage, lighting changes, noise, weather, and dynamic backgrounds. To assess our approach, we chose three outdoor scenes with dynamic backgrounds: "Campus," "Fountain," and "WaterSurface." As outlined in the Section 5.4.2, we manually selected the training frames.

5.4.3 Evaluation Metric

To evaluate the performance of our method, we utilize the F-Measure (FM) metric, which is a widely used performance indicator for moving object detection and background subtraction algorithms. The F-measure, which combines the recall and precision scores, is calculated using the equation 2.3. In order to maintain consistency with existing methods, we compute all evaluations according to the definitions provided in [99].

5.4.4 Qualitative Results

In Fig. 5.4, we present the intermediate and final qualitative results of our method's steps on the videos. The first six rows depict the *Dynamic Background* category, followed by the next four rows from the *Bad Weather* category of the *CDnet 2014* dataset. The last three rows show videos from the *I2R* dataset.

The first three columns display the input frame, the autoencoder-generated background, and the background-subtracted image, respectively. The fourth column exhibits the dynamic background image predicted by the U-Net. The next column displays the foreground image obtained by multiplying the background-subtracted image with the inverted version of the dynamic background image. The sixth and seventh rows showcase the initial segmented image after thresholding and the final segmented image after post-processing, respectively. The last column shows the ground-truth images.

In Fig. 5.4, it is evident from the 4th column that the U-Net model can efficiently capture the dynamic background motion and create a precise representation of the dynamic background image, particularly for the *Dynamic Background* videos. By comparing the background-subtracted image in the third column with the obtained foreground in the fifth column, it proves our method can effectively decompose the dynamic background from the foreground objects.

For the *Bad Weather* videos, our method is able to predict some of the dynamic background pixels. However, due to the nature of our autoencoder, it tends to absorb snow noise in the generated static background image, leading to a lack of visible snow pixels in the dynamic background image. Nevertheless, our method is still able to produce high-quality results, demonstrating its effectiveness in handling challenging weather scenarios.

Regarding the *I2R* dataset, our U-Net accurately predicts the dynamic background pixels in the "Campus" sequence and some of the dynamic background pixels in the "WaterSurface" sequence. However, in the "Fountain" sequence, the U-Net is not able to predict the fountain pixels in the dynamic background image since they are already detected as part of the background generated by the autoencoder. This is because the values of the fountain pixels remain constant in consecutive frames, and therefore, they are absorbed in the static background image.

The key aspect is to effectively separate the foreground pixels from the dynamic background pixels, which our framework achieves well through the use of both the autoencoder and the U-Net models. This ultimately leads to the superior performance of our method.

5.4.5 Quantitative Results

In this section, we present the quantitative results of our method compared to the top-performing methods on the *CDnet 2014* dataset [99] listed on ChangeDetection.net website. Specifically, we chose the top 30 methods based on their average F-measure (FM) performance on the *Dynamic Background*

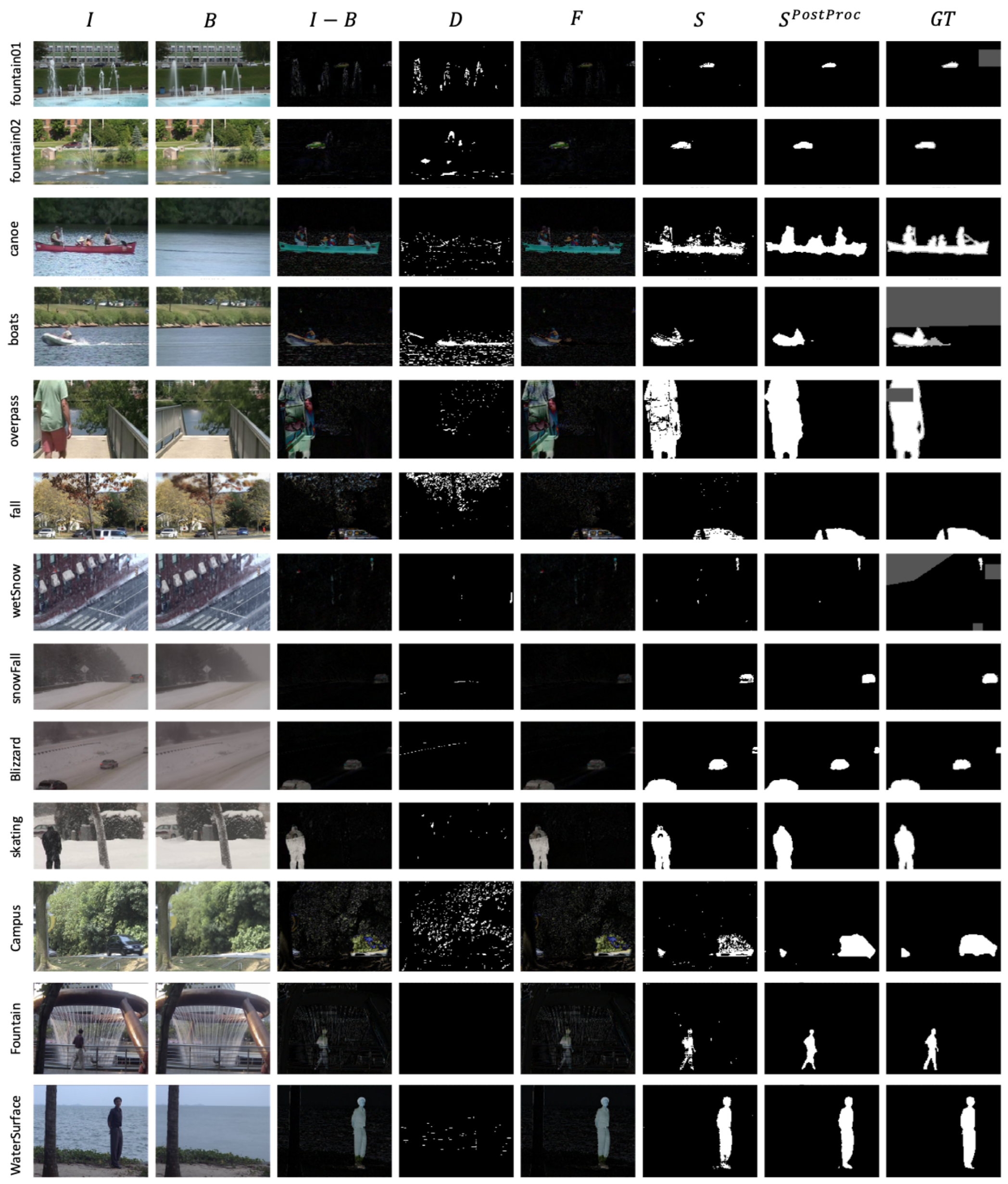


Figure 5.4: Qualitative results of the proposed method. The columns from left to right display the input images, the generated background images, the background-subtracted images, the predicted dynamic background images, the obtained foreground images, the initial segmented images after thresholding, the final segmented images after post-processing and the ground-truth images, respectively

and *Bad Weather* categories, excluding supervised methods and the ensemble method IUTIS [9]. We also included the results of the LDB weakly supervised method [107], as well as the CANDID algorithm [64], which was specifically proposed for dynamic background subtraction.

Table 5.1 displays the results on *Dynamic Background* videos, and Table 5.2 shows the results on *Bad Weather* videos. The methods are sorted based on their average FM on *Dynamic Background* videos. Our method's results are reported in the last row of the tables.

As shown in Table 5.1, our method achieves an average FM of 0.91 on *Dynamic Background*, outperforming all unsupervised methods and the LDB [107] method. Our method also achieves the highest FM on the "fall" video and performs the best on "fountain01" along with the FTSG method [98]. These results demonstrate the practicality of our method for dynamic background scenes with only the cost of frame-level tag training data.

As shown in Table 5.2, on *Bad Weather* sequences, our method achieves an average FM of 0.89, outperforming all unsupervised top-ranked methods, but is surpassed by the LDB weakly supervised method [107], which has an FM of 0.91. Our method also achieves the best FM on the "blizzard" video, while LDB obtains the best FM on the "WetSnow" and "snowFall" sequences.

To compare our method to LDB more comprehensively, we also perform experiments on the *I2R* dataset and report the results in Table 5.3. As shown, our method achieves the same average FM as LDB, but we obtain slightly better results on the "Fountain" and "WaterSurface" sequences, while LDB performs slightly better on the "Campus" sequence.

A comparison of our method and LDB on various videos shows that our method performs better in the *Dynamic Background* category, while LDB performs better in the *Bad Weather* category. For the *I2R* sequences, both methods achieve similar performance. It is worth noting that LDB uses a low rank tensor decomposition and is a batch method, whereas our method is an online method achieving realtime performance where an input image is fed through the two networks to obtain the result.

We also compared our method to another weakly supervised method [66] described in the Section 5.2. Minematsu et al. performed experiments on eight categories of the *CDnet 2014* dataset [99], but only selected some of the sequences for each category. We performed the same experiments and report the results in Table 5.4. As shown in the table, our method achieves an average FM of 0.72, while Minematsu et al. [66] obtain an average FM of 0.66. Our method outperforms theirs on the *Bad Weather*, *Dynamic Background*, *Shadow*, *Thermal*, and *Turbulence* sequences, while they achieve better performance on the *Baseline*, *Camera Jitter*, and *Night Videos* sequences.

5.4.6 Experiments on Impact of the U-Net capacity on Performance

To evaluate how changes in U-Net capacity affect the performance of our method, we conducted experiments on three videos from the *CDnet 2014* dataset: "fountain02", "canoe"and "fall". These videos exhibit different types of dynamic background motions. In this set of experiments, we kept all hyperparameters fixed except for the number of contracting/expansive steps of the U-Net and the number of features in each step. Since the contracting and expansive paths are symmetric, we report only the top-to-bottom steps of the contracting path and their corresponding number of features.

The results are presented in Table 5.5. Notably, our algorithm's performance remained consistent across different U-Net capacities, as evidenced by Table 5.5. This suggests that the U-Net can effectively classify dynamic background pixels even when its capacity is reduced.

5.5 Conclusion

In this chapter, we presented a novel weakly supervised realtime method for dynamic background subtraction, which utilizes two neural networks: an autoencoder for static background image generation and a U-Net for dynamic background image generation. While the autoencoder learns in an unsupervised manner, the U-Net requires pixel-wise ground-truth labels for supervised training. However, obtaining pixel-wise annotations can be an expensive and

Table 5.1: Performance comparison of the top-ranked methods, evaluated on *Dynamic Background* category of *CDnet 2014*, in terms of F-measure. The best performance achieved, in each column, is shown in bold. Methods are sorted based on their Average F-Measure.

Methods	fount01	fount02	canoe	boats	overpass	fall	Avg
GraphCutDiff [67]	0.08	0.91	0.57	0.12	0.84	0.72	0.54
CL-VID [60]	0.05	0.45	0.93	0.81	0.85	0.23	0.55
C-EFIC [2]	0.27	0.34	0.93	0.37	0.90	0.56	0.56
EFIC [1]	0.23	0.91	0.36	0.36	0.88	0.72	0.58
Multi_ST_BG [61]	0.14	0.82	0.48	0.89	0.84	0.41	0.60
KDE-ElGamm [30]	0.11	0.82	0.88	0.63	0.82	0.31	0.60
CP3-online [56]	0.54	0.91	0.63	0.17	0.64	0.77	0.61
DCB [52]	0.40	0.83	0.45	0.87	0.83	0.30	0.61
GMM_Zivk [111]	0.08	0.79	0.89	0.75	0.87	0.42	0.63
GMM_Grim [83]	0.08	0.80	0.88	0.73	0.87	0.44	0.63
SOBS_CF [62]	0.11	0.83	**0.95**	0.91	0.85	0.26	0.65
SC_SOBS [63]	0.12	0.89	**0.95**	0.90	0.88	0.28	0.67
AAPSA [71]	0.44	0.36	0.89	0.76	0.82	0.75	0.67
M4CD_V2 [95]	0.17	0.93	0.61	**0.95**	0.95	0.50	0.69
RMoG [91]	0.20	0.87	0.94	0.83	0.90	0.67	0.74
WeSamBE [46]	0.73	0.94	0.61	0.64	0.72	0.81	0.74
Spectral360 [77]	0.47	0.92	0.88	0.69	0.81	0.90	0.78
LDB [107]	0.14	0.93	0.92	0.92	0.95	0.79	0.78
MBS_V0 [73]	0.52	0.92	0.93	0.90	0.90	0.57	0.79
MBS [74]	0.52	0.92	0.93	0.90	0.90	0.57	0.79
BMOG [65]	0.38	0.93	**0.95**	0.84	**0.96**	0.69	0.79
CANDID [64]	0.55	0.92	0.91	0.66	0.92	0.81	0.80
SBBS [92]	0.73	0.93	0.49	0.94	0.91	0.88	0.81
SuBSENSE [19]	0.75	0.94	0.79	0.69	0.86	0.87	0.82
SharedModel [23]	0.78	0.94	0.62	0.88	0.82	0.89	0.82
CwisarDH [28]	0.61	0.93	0.94	0.84	0.90	0.75	0.83
WisenetMD [54]	0.75	**0.95**	0.87	0.71	0.87	0.87	0.84
AMBER [94]	0.77	0.93	0.93	0.85	0.95	0.63	0.84
CwisarDRP [29]	0.69	0.92	0.91	0.84	0.92	0.82	0.85
CVABS [45]	0.77	0.94	0.88	0.81	0.86	0.91	0.86
SWCD [44]	0.76	0.93	0.92	0.85	0.85	0.88	0.86
DBSGen [4]	0.73	0.80	0.90	0.91	0.87	0.93	0.86
FTSG [98]	**0.81**	**0.95**	0.69	**0.95**	0.94	0.93	0.88
PAWCS [20]	0.78	0.94	0.94	0.84	**0.96**	0.91	0.89
Our Method	**0.81**	0.94	0.92	0.94	0.91	**0.94**	**0.91**

Table 5.2: Performance comparison of the top-ranked methods, evaluated on *bad Weather* category of *CDnet 2014*, in terms of F-measure. The best performance achieved, in each column, is shown in bold. The methods are listed in the same order as the table 5.1

Methods	wetSnow	snowFall	blizzard	skating	Avg
GraphCutDiff [67]	0.83	0.90	0.86	0.92	0.88
CL-VID [60]	0.54	0.79	0.75	0.87	0.74
C-EFIC [2]	0.65	0.74	0.86	0.90	0.79
EFIC [1]	0.62	0.71	0.86	0.92	0.78
Multi_ST_BG [61]	0.53	0.71	0.71	0.59	0.64
KDE-ElGamm [30]	0.57	0.78	0.77	0.91	0.76
CP3-online [56]	0.75	0.76	0.85	0.63	0.75
DCB [52]	0.30	0.34	0.41	0.49	0.38
GMM_Zivk [111]	0.58	0.76	0.86	0.76	0.74
GMM_Grim [83]	0.61	0.73	0.88	0.74	0.74
SOBS_CF [62]	0.50	0.62	0.67	0.76	0.64
SC_SOBS [63]	0.50	0.60	0.66	0.90	0.66
AAPSA [71]	0.63	0.80	0.82	0.85	0.77
M4CD_V2 [95]	0.69	0.81	0.81	**0.94**	0.81
RMoG [91]	0.60	0.58	0.76	0.79	0.68
WeSamBE [46]	0.81	0.87	0.90	0.86	0.86
Spectral360 [77]	0.65	0.79	0.67	0.92	0.76
LDB [107]	**0.89**	**0.93**	0.90	0.90	**0.91**
MBS_V0 [73]	0.43	0.88	0.86	0.92	0.77
MBS [74]	0.53	0.88	0.86	0.92	0.80
BMOG [65]	0.69	0.73	0.79	0.92	0.78
CANDID [64]	0.83	0.78	0.87	0.92	0.85
SBBS [92]	0.45	0.79	0.81	0.90	0.74
SuBSENSE [19]	0.80	0.89	0.85	0.91	0.86
SharedModel [23]	0.73	0.89	0.91	0.86	0.85
CwisarDH [28]	0.32	0.75	0.91	0.77	0.68
WisenetMD [54]	0.80	0.89	0.85	0.91	0.86
AMBER [94]	0.65	0.72	0.79	0.91	0.77
CwisarDRP [29]	0.71	0.80	0.91	0.78	0.80
CVABS [45]	0.83	0.84	0.87	0.89	0.86
SWCD [44]	0.78	0.83	0.82	0.86	0.82
DBSGen [4]	0.82	0.76	0.80	0.86	0.81
FTSG [98]	0.71	0.82	0.85	0.91	0.82
PAWCS [20]	0.75	0.77	0.84	0.90	0.82
Our Method	0.83	0.89	**0.92**	0.93	0.89

Table 5.3: Performance comparison with LDB weakly supervised method [107], evaluated on three dynamic background videos of *I2R* dataset, in terms of F-measure. The best performance achieved, in each column, is shown in bold.

Methods	Campus	Fountain	WaterSurface	Avg
LDB [107]	**0.85**	0.85	0.94	0.88
Our Method	0.83	**0.86**	**0.95**	0.88

Table 5.4: Performance comparison with the Weakly Supervised method proposed in [66], evaluated on eight categories of *CDnet 2014*, in terms of F-measure. The best performance achieved, in each column, is shown in bold.

Methods	Method in [66]	Our Method
BadWeather	0.72	**0.89**
Baseline	**0.97**	0.92
CameraJitter	**0.61**	0.42
DynamicBackground	0.82	**0.91**
NightVideos	**0.38**	0.29
Shadow	0.56	**0.92**
Thermal	0.66	**0.78**
turbulence	0.58	**0.59**
Avg	0.66	**0.72**

Table 5.5: Effects of U-Net capacity on performance, as measured by F-measure, for varying numbers of contracting/expansive steps and features. The second row displays the steps and features used in our method's results, reported in previous tables.

Number of the Features in Steps of the U-Net	canoe	fountain02	fall	Avg.
64, 128, 256, 512, 1024, 2048	0.91	0.94	0.91	0.92
64, 128, 256, 512, 1024	0.92	0.94	0.94	0.93
32, 64, 128, 256, 512	0.94	0.94	0.94	0.94
16, 32, 64, 128, 256	0.92	0.94	0.94	0.93
64, 128, 256, 512	0.91	0.94	0.93	0.93
32, 64, 128, 256	0.9	0.94	0.93	0.92
64, 128, 256	0.88	0.94	0.93	0.92
64, 128	0.9	0.94	0.94	0.93

time-consuming task. To overcome this challenge, we prepared these labels in a weakly supervised way by selecting training frames that do not contain any moving objects. The autoencoder can generate the static background image by leveraging the temporal correlation between frames. After performing background subtraction and thresholding, the resulting image represents the dynamic background since the input image is moving object-free and only contains dynamic and static background. The U-Net then trains with the same moving object-free sequence of images and the binary dynamic background images as the ground-truth labels. During testing, we can feed an input image to the networks and obtain the static and dynamic background images in the output, resulting in a clean foreground image without dynamic background motions.

Our experiments on various sequences demonstrated that our method is effective in real-world scenarios. Our algorithm outperformed all top-ranked unsupervised methods as well as a weakly supervised method. We performed equally to another state-of-the-art weakly supervised method [107], which is specifically designed for handling dynamic background scenes.

In summary, our proposed method has a training phase followed by an online test phase, during which it can effectively detect dynamic background artifacts and separate them from the moving object foreground.

Chapter 6

Conclusion

In this thesis, we focused on moving object detection problem using a background subtraction approach, which is an essential task in computer vision with many practical applications. Although there are various techniques for background subtraction, deep learning-based methods that require supervised learning are currently considered the state-of-the-art. However, obtaining pixel-wise ground-truth labels for these approaches can be time-consuming and expensive. Hence, our goal was to propose background subtraction methods that do not need per-pixel ground-truth labeling during training.

We introduced an unsupervised background subtraction approach that can handle illumination changes and shadows effectively. In addition, we presented two more frameworks - one unsupervised and one weakly supervised - that can perform background subtraction while dealing with dynamic backgrounds in the scene.

In summary, we have made contributions to the field of background subtraction by proposing novel techniques that do not require pixel-wise ground-truth annotations, making the process more efficient and cost-effective.

6.1 Contributions

6.1.1 Online Illumination Invariant Moving Object Detection by Generative Neural Network

Although state-of-the-art techniques exist that can handle illumination changes and shadows, they work only in batch mode, making them impractical for processing long video sequences or for real-time applications. To address this challenge, we proposed an online/incremental moving object detection (MOD) method [6] that builds on a state-of-the-art batch MOD method (ILISD) [82]. Our approach uses unsupervised and generative neural networks to leverage illumination-invariant image representations. Specifically, for each image in the video sequence, we employ a neural network to generate a low-dimensional representation of the background image. We then decompose the foreground image into its illumination change and moving object components based on this illumination-invariant representation. Our method can operate in both batch and online modes. In batch mode, like other batch methods, the optimizer utilizes all the images in the sequence, while in online mode, images can be incrementally fed into the optimizer. We have evaluated our method experimentally on benchmark image sequences and demonstrated that both the online and batch modes of our algorithm achieve state-of-the-art accuracy on most datasets.

Our contribution is introducing an online/incremental MOD method that can handle illumination changes and shadows effectively.

6.1.2 Dynamic Background Subtraction by Generative Neural Networks

Dynamic backgrounds, which can have stochastic movements in some parts of the scene, pose a challenge to background subtraction methods. To address this challenge, we proposed a novel background subtraction framework called DBSGen [4] in our second method, which utilizes two generative neural networks. The first network estimates a dense motion map that minimizes the difference between each input image and a fixed background image, while the second network generates background images for the warped input images. The foreground images are then obtained by subtracting the background images from the warped images. To avoid deformed objects in the results, an inverse warping of the motion map is applied to the foreground images to warp back the moving objects. Finally, our method computes a pixel-wise distance threshold, which is used to obtain binary segmented images. DBS-

Gen is an end-to-end, unsupervised optimization method that achieves a near real-time frame rate. We evaluated the performance of our method over dynamic background dataset and found that it outperforms most state-of-the-art unsupervised methods.

To summarize the contributions of DBSGen, the first contribution is its use of a generative network to estimate a pixel-wise motion map, which is then utilized for dynamic background subtraction. Unlike many other neural network-based methods, DBSGen is optimized in an unsupervised manner, meaning it does not require expensive pixel-wise ground-truth masks. Lastly, DBSGen is an end-to-end neural network framework, meaning it is optimized in a single stage.

6.1.3 Weakly Supervised Realtime Dynamic Background Subtraction

Our third proposed method [5] introduces a weakly supervised framework for dynamic background subtraction that does not rely on per-pixel ground-truth labels. The framework is composed of two networks trained on a sequence of images without moving objects. The first network is an autoencoder that generates background images and prepares dynamic background images for training the second network. The second network is a U-Net that employs the same object-free video for training, along with dynamic background images serving as pixel-wise ground-truth labels. During test phase, the autoencoder and U-Net process the input images and generate background and dynamic background images, respectively. The dynamic background image helps eliminate dynamic motion from the background subtracted image, providing a foreground image that is free of dynamic artifacts. Our method's effectiveness was demonstrated through experiments on various datasets, including dynamic background sequences, where we outperformed all top-ranked unsupervised methods. It outperformed one of the two current weakly supervised methods, while achieving similar results to the other one, but with lower computational time. Our proposed method requires minimal frame-level annotation, is efficient, and online, making it suitable for numerous real-world applications.

Our proposed method offers significant contributions. Firstly, it is a cost-effective and practical weakly supervised framework that eliminates the need for expensive per-pixel annotations by training only on a sequence of object-free images. Secondly, it accurately learns and predicts the dynamic background component independently in each frame and effectively segments it from the foreground. Lastly, our experimental results demonstrate that our algorithm achieves superior performance in dynamic background scenes compared to other state-of-the-art unsupervised and weakly supervised methods, both quantitatively and qualitatively.

6.2 Future Work

We have several suggestions for future work:

- In the NUMOD method [6] presented in Chapter 3, we did not consider any assumptions about the structure of the sparse foreground regions. In future work, the results could be enhanced by incorporating assumptions regarding the connectivity of foreground outliers.
- In Chapter 4, we proposed the DBSGen method [4] that can handle dynamic backgrounds. Merging DBSGen with NUMOD [6] could enable coping with illumination changes, shadows, and dynamic backgrounds simultaneously, in one unsupervised framework.
- For the weakly supervised method [5] proposed in Chapter 5, currently, the training data consists of images with only dynamic background without any foreground objects. One possible extension could be to incorporate data augmentation techniques to acquire more comprehensive training data that includes pixel-wise dynamic background labels for images containing moving objects. Additionally, data augmentation with different brightness levels could help to handle illumination changes more effectively.

References

[1] G. Allebosch, F. Deboeverie, P. Veelaert, and W. Philips, “Efic: Edge based foreground background segmentation and interior classification for dynamic camera viewpoints,” in *International conference on advanced concepts for intelligent vision systems*, Springer, 2015, pp. 130–141.

[2] G. Allebosch, D. Van Hamme, F. Deboeverie, P. Veelaert, and W. Philips, “C-efic: Color and edge based foreground background segmentation with interior classification,” in *International joint conference on computer vision, imaging and computer graphics*, Springer, 2015, pp. 433–454.

[3] M. Babaee, D. T. Dinh, and G. Rigoll, “A deep convolutional neural network for video sequence background subtraction,” *Pattern Recognition*, vol. 76, pp. 635–649, 2018.

[4] F. Bahri and N. Ray, “Dynamic background subtraction by generative neural networks,” in *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, IEEE, 2022, pp. 1–8.

[5] F. Bahri and N. Ray, “Weakly supervised realtime dynamic background subtraction,” *arXiv preprint arXiv:2303.02857*, 2023.

[6] F. Bahri, M. Shakeri, and N. Ray, “Online illumination invariant moving object detection by generative neural network,” in *Proceedings of the 11th Indian Conference on Computer Vision, Graphics and Image Processing*, 2018, pp. 1–8.

[7] D. Bank, N. Koenigstein, and R. Giryes, “Autoencoders,” *arXiv preprint arXiv:2003.05991*, 2020.

[8] A. Beck and M. Teboulle, “A fast iterative shrinkage-thresholding algorithm for linear inverse problems,” *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[9] S. Bianco, G. Ciocca, and R. Schettini, “Combination of video change detection algorithms by genetic programming,” *IEEE Transactions on Evolutionary Computation*, vol. 21, no. 6, pp. 914–928, 2017.

[10] S. Bianco, G. Ciocca, and R. Schettini, "How far can you get by combining change detection algorithms?" In *International conference on image analysis and processing*, Springer, 2017, pp. 96–107.

[11] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer science review*, vol. 11, pp. 31–66, 2014.

[12] T. Bouwmans, S. Javed, M. Sultana, and S. K. Jung, "Deep neural network concepts for background subtraction: A systematic review and comparative evaluation," *Neural Networks*, vol. 117, pp. 8–66, 2019.

[13] M. Braham, S. Pierard, and M. Van Droogenbroeck, "Semantic background subtraction," in *2017 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2017, pp. 4552–4556.

[14] M. Braham and M. Van Droogenbroeck, "Deep background subtraction with scene-specific convolutional neural networks," in *IEEE International Conference on Systems, Signals and Image Processing (IWSSIP), Bratislava 23-25 May 2016*, IEEE, 2016, pp. 1–4.

[15] S. Brutzer, B. Höferlin, and G. Heidemann, "Evaluation of background subtraction techniques for video surveillance," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, IEEE, 2011, pp. 1937–1944.

[16] E. J. Candes, M. B. Wakin, and S. P. Boyd, "Enhancing sparsity by reweighted ℓ 1 minimization," *Journal of Fourier analysis and applications*, vol. 14, no. 5-6, pp. 877–905, 2008.

[17] E. J. Candès, X. Li, Y. Ma, and J. Wright, "Robust principal component analysis?" *Journal of the ACM (JACM)*, vol. 58, no. 3, p. 11, 2011.

[18] J. M. Chaquet, E. J. Carmona, and A. Fernández-Caballero, "A survey of video datasets for human action and activity recognition," *Computer Vision and Image Understanding*, vol. 117, no. 6, pp. 633–659, 2013.

[19] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin, "Subsense: A universal change detection method with local adaptive sensitivity," *IEEE Transactions on Image Processing*, vol. 24, no. 1, pp. 359–373, 2014.

[20] P.-L. St-Charles, G.-A. Bilodeau, and R. Bergevin, "A self-adjusting approach to change detection based on background word consensus," in *2015 IEEE winter conference on applications of computer vision*, IEEE, 2015, pp. 990–997.

[21] B.-H. Chen and S.-C. Huang, "An advanced moving object detection algorithm for automatic traffic monitoring in real-world limited bandwidth networks," *IEEE Transactions on Multimedia*, vol. 16, no. 3, pp. 837–847, 2014.

[22] L.-H. Chen, Y.-H. Yang, C.-S. Chen, and M.-Y. Cheng, "Illumination invariant feature extraction based on natural images statistics—taking face images as an example," in *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, IEEE, 2011, pp. 681–688.

[23] Y. Chen, J. Wang, and H. Lu, "Learning sharable models for robust background subtraction," in *2015 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, 2015, pp. 1–6.

[24] Y. Chen, J. Wang, B. Zhu, M. Tang, and H. Lu, "Pixel-wise deep sequence learning for moving object detection," *IEEE Transactions on Circuits and Systems for Video Technology*, 2017.

[25] F. Chollet *et al.*, *Keras*, `https://keras.io`, 2015.

[26] A. Cioppa, M. Van Droogenbroeck, and M. Braham, "Real-time semantic background subtraction," in *2020 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2020, pp. 3214–3218.

[27] D.-A. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *arXiv preprint arXiv:1511.07289*, 2015.

[28] M. De Gregorio and M. Giordano, "Change detection with weightless neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 403–407.

[29] M. De Gregorio and M. Giordano, "Wisardrp for change detection in video sequences.," in *ESANN*, 2017.

[30] A. Elgammal, D. Harwood, and L. Davis, "Non-parametric model for background subtraction," in *European conference on computer vision*, Springer, 2000, pp. 751–767.

[31] J. Feng, H. Xu, and S. Yan, "Online robust pca via stochastic optimization," in *Advances in Neural Information Processing Systems*, 2013, pp. 404–412.

[32] G. D. Finlayson, S. D. Hordley, C. Lu, and M. S. Drew, "On the removal of shadows from images," *IEEE transactions on pattern analysis and machine intelligence*, vol. 28, no. 1, pp. 59–68, 2006.

[33] F. Gao, Y. Li, and S. Lu, "Extracting moving objects more accurately: A cda contour optimizer," *IEEE Transactions on Circuits and Systems for Video Technology*, 2021.

[34] B. Garcia-Garcia, T. Bouwmans, and A. J. R. Silva, "Background subtraction in real applications: Challenges, current models and future directions," *Computer Science Review*, vol. 35, p. 100 204, 2020.

[35] J.-H. Giraldo-Zuluaga, A. Salazar, A. Gomez, and A. Diaz-Pulido, "Recognition of mammal genera on camera-trap images using multi-layer robust principal component analysis and mixture neural networks," in *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, IEEE, 2017, pp. 53–60.

[36] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning.* MIT press Cambridge, 2016, vol. 1.

[37] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, P. Ishwar, *et al.*, "Changedetection. net: A new change detection benchmark dataset.," in *CVPR Workshops*, 2012, pp. 1–8.

[38] R. A. Hadi, L. E. George, and M. J. Mohammed, "A computationally economic novel approach for real-time moving multi-vehicle detection and tracking toward efficient traffic surveillance," *Arabian Journal for Science and Engineering*, vol. 42, no. 2, pp. 817–831, 2017.

[39] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, IEEE, 2012, pp. 1568–1575.

[40] J. Heikkila and O. Silvén, "A real-time system for monitoring of cyclists and pedestrians," in *Proceedings Second IEEE Workshop on Visual Surveillance (VS'99)(Cat. No. 98-89223)*, IEEE, 1999, pp. 74–81.

[41] H.-H. Hsiao and J.-J. Leou, "Background initialization and foreground segmentation for bootstrapping video sequences," *EURASIP Journal on Image and Video Processing*, vol. 2013, no. 1, pp. 1–19, 2013.

[42] H. Huang, X. Fang, Y. Ye, S. Zhang, and P. L. Rosin, "Practical automatic background substitution for live video," *Computational Visual Media*, vol. 3, pp. 273–284, 2017.

[43] W. Huang, Q. Zeng, and M. Chen, "Motion characteristics estimation of animals in video surveillance," in *2017 IEEE 2nd Advanced Information Technology, Electronic and Automation Control Conference (IAEAC)*, IEEE, 2017, pp. 1098–1102.

[44] Ş. Işık, K. Özkan, S. Günal, and Ö. Nezih Gerek, "Swcd: A sliding window and self-regulated learning-based background updating method for change detection in videos," *Journal of Electronic Imaging*, vol. 27, no. 2, p. 023 002, 2018.

[45] Ş. Işık, K. Özkan, and Ö. Nezih Gerek, "Cvabs: Moving object segmentation with common vector approach for videos," *IET Computer Vision*, vol. 13, no. 8, pp. 719–729, 2019.

[46] S. Jiang and X. Lu, "Wesambe: A weight-sample-based method for background subtraction," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 28, no. 9, pp. 2105–2115, 2017.

[47] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Video-based surveillance systems*, Springer, 2002, pp. 135–144.

[48] K. Kalirajan and M. Sudha, "Moving object detection for video surveillance," *The Scientific World Journal*, vol. 2015, 2015.

[49] R. Kalsotra and S. Arora, "Background subtraction for moving object detection: Explorations of recent developments and challenges," *The Visual Computer*, vol. 38, no. 12, pp. 4151–4178, 2022.

[50] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[51] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," *Advances in neural information processing systems*, vol. 30, 2017.

[52] R. Krungkaew and W. Kusakunniran, "Foreground segmentation in a video by using a novel dynamic codebook," in *2016 13th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*, IEEE, 2016, pp. 1–6.

[53] D.-S. Lee, "Effective gaussian mixture learning for video background subtraction," *IEEE transactions on pattern analysis and machine intelligence*, vol. 27, no. 5, pp. 827–832, 2005.

[54] S.-h. Lee, G.-c. Lee, J. Yoo, and S. Kwon, "Wisenetmd: Motion detection using dynamic background region analysis," *Symmetry*, vol. 11, no. 5, p. 621, 2019.

[55] L. Li, W. Huang, I. Y.-H. Gu, and Q. Tian, "Statistical modeling of complex backgrounds for foreground object detection," *IEEE Transactions on Image Processing*, vol. 13, no. 11, pp. 1459–1472, 2004.

[56] D. Liang, M. Hashimoto, K. Iwata, X. Zhao, *et al.*, "Co-occurrence probability-based pixel pairs background model for robust object detection in dynamic scenes," *Pattern Recognition*, vol. 48, no. 4, pp. 1374–1390, 2015.

[57] L. A. Lim and H. Y. Keles, "Foreground segmentation using convolutional neural networks for multiscale feature encoding," *Pattern Recognition Letters*, vol. 112, pp. 256–262, 2018.

[58] L. A. Lim and H. Y. Keles, "Learning multi-scale features for foreground segmentation," *Pattern Analysis and Applications*, vol. 23, no. 3, pp. 1369–1380, 2020.

[59] X. Liu, G. Zhao, J. Yao, and C. Qi, "Background subtraction based on low-rank and structured sparse decomposition," *IEEE Transactions on Image Processing*, vol. 24, no. 8, pp. 2502–2514, 2015.

[60] E. López-Rubio, M. A. Molina-Cabello, R. M. Luque-Baena, and E. Domínguez, “Foreground detection by competitive learning for varying input distributions,” *International journal of neural systems*, vol. 28, no. 05, p. 1 750 056, 2018.

[61] X. Lu, “A multiscale spatio-temporal background model for motion detection,” in *2014 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2014, pp. 3268–3271.

[62] L. Maddalena and A. Petrosino, “A fuzzy spatial coherence-based approach to background/foreground separation for moving object detection,” *Neural Computing and Applications*, vol. 19, no. 2, pp. 179–186, 2010.

[63] L. Maddalena and A. Petrosino, “The sobs algorithm: What are the limits?” In *2012 IEEE computer society conference on computer vision and pattern recognition workshops*, IEEE, 2012, pp. 21–26.

[64] M. Mandal, P. Saxena, S. K. Vipparthi, and S. Murala, “Candid: Robust change dynamics and deterministic update policy for dynamic background subtraction,” in *2018 24th international conference on pattern recognition (ICPR)*, IEEE, 2018, pp. 2468–2473.

[65] I. Martins, P. Carvalho, L. Corte-Real, and J. L. Alba-Castro, “Bmog: Boosted gaussian mixture model with controlled complexity,” in *Iberian conference on pattern recognition and image analysis*, Springer, 2017, pp. 50–57.

[66] T. Minematsu, A. Shimada, and R.-i. Taniguchi, “Simple background subtraction constraint for weakly supervised background subtraction network,” in *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, IEEE, 2019, pp. 1–8.

[67] A. Miron and A. Badii, “Change detection based on graph cuts,” in *2015 International Conference on Systems, Signals and Image Processing (IWSSIP)*, IEEE, 2015, pp. 273–276.

[68] A. Nitanda, “Stochastic proximal gradient descent with acceleration techniques,” *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[69] N. Parikh, S. Boyd, *et al.*, “Proximal algorithms,” *Foundations and trends® in Optimization*, vol. 1, no. 3, pp. 127–239, 2014.

[70] G. Rahmon, F. Bunyak, G. Seetharaman, and K. Palaniappan, “Motion u-net: Multi-cue encoder-decoder network for motion segmentation,” in *2020 25th International Conference on Pattern Recognition (ICPR)*, IEEE, 2021, pp. 8125–8132.

[71] G. Ramirez-Alonso and M. I. Chacon-Murguia, “Auto-adaptive parallel som architecture with a modular analysis for dynamic object segmentation in videos,” *Neurocomputing*, vol. 175, pp. 990–1000, 2016.

[72] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, Springer, 2015, pp. 234–241.

[73] H. Sajid and S.-C. S. Cheung, “Background subtraction for static & moving camera,” in *2015 IEEE International Conference on Image Processing (ICIP)*, IEEE, 2015, pp. 4530–4534.

[74] H. Sajid and S.-C. S. Cheung, “Universal multimode background subtraction,” *IEEE Transactions on Image Processing*, vol. 26, no. 7, pp. 3249–3260, 2017.

[75] D. Sakkos, H. Liu, J. Han, and L. Shao, “End-to-end video background subtraction with 3d convolutional neural networks,” *Multimedia Tools and Applications*, pp. 1–19, 2017.

[76] S. Saravanakumar, A. Vadivel, and C. S. Ahmed, “Multiple human object tracking using background subtraction and shadow removal techniques,” in *Signal and Image Processing (ICSIP), 2010 International Conference on*, IEEE, 2010, pp. 79–84.

[77] M. Sedky, M. Moniri, and C. C. Chibelushi, “Spectral-360: A physics-based technique for change detection,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 399–402.

[78] S. C. Sen-Ching and C. Kamath, “Robust techniques for background subtraction in urban traffic video,” in *Visual Communications and Image Processing 2004*, International Society for Optics and Photonics, vol. 5308, 2004, pp. 881–893.

[79] S. H. Shaikh, K. Saeed, N. Chaki, S. H. Shaikh, K. Saeed, and N. Chaki, *Moving object detection using background subtraction*. Springer, 2014.

[80] M. Shakeri and H. Zhang, “Corola: A sequential solution to moving object detection using low-rank approximation,” *Computer Vision and Image Understanding*, vol. 146, pp. 27–39, 2016.

[81] M. Shakeri and H. Zhang, “Illumination invariant representation of natural images for visual place recognition,” in *Intelligent Robots and Systems (IROS), 2016 IEEE/RSJ International Conference on*, IEEE, 2016, pp. 466–472.

[82] M. Shakeri and H. Zhang, “Moving object detection in time-lapse or motion trigger image sequences using low-rank and invariant sparse decomposition,” in *Proc. IEEE Int. Conf. Comput. Vis.(ICCV)*, 2017, pp. 5133–5141.

[83] C. Stauffer and W. E. L. Grimson, “Adaptive background mixture models for real-time tracking,” in *Proceedings. 1999 IEEE computer society conference on computer vision and pattern recognition (Cat. No PR00149)*, IEEE, vol. 2, 1999, pp. 246–252.

[84] S. Suresh, P. Deepak, and K. Chitra, “An efficient low cost background subtraction method to extract foreground object during human tracking,” in *Circuit, Power and Computing Technologies (ICCPCT), 2014 International Conference on*, IEEE, 2014, pp. 1432–1436.

[85] B. Tamás, “Detecting and analyzing rowing motion in videos,” in *BME scientific student conference*, 2016, pp. 1–29.

[86] M. O. Tezcan, P. Ishwar, and J. Konrad, “Bsuv-net 2.0: Spatio-temporal data augmentations for video-agnostic supervised background subtraction,” *IEEE Access*, vol. 9, pp. 53 849–53 860, 2021.

[87] O. Tezcan, P. Ishwar, and J. Konrad, “Bsuv-net: A fully-convolutional neural network for background subtraction of unseen videos,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2020, pp. 2774–2783.

[88] Y. Tian, A. Senior, and M. Lu, “Robust and efficient foreground analysis in complex surveillance videos,” *Machine vision and applications*, vol. 23, no. 5, pp. 967–983, 2012.

[89] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: Principles and practice of background maintenance,” in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, IEEE, vol. 1, 1999, pp. 255–261.

[90] T. H. Tsai, C.-Y. Lin, and S.-Y. Li, “Algorithm and architecture design of human–machine interaction in foreground object detection with dynamic scene,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 23, no. 1, pp. 15–29, 2013.

[91] S. Varadarajan, P. Miller, and H. Zhou, “Spatial mixture of gaussians for dynamic background modelling,” in *2013 10th IEEE International Conference on Advanced Video and Signal Based Surveillance*, IEEE, 2013, pp. 63–68.

[92] A. Varghese and G. Sreelekha, “Sample-based integrated background subtraction and shadow detection,” *IPSJ Transactions on Computer Vision and Applications*, vol. 9, no. 1, pp. 1–12, 2017.

[93] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[94] B. Wang and P. Dudek, "A fast self-tuning background subtraction algorithm," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 395–398.

[95] K. Wang, C. Gou, and F.-Y. Wang, "M^4CD: A robust change detection method for intelligent visual surveillance," *IEEE Access*, vol. 6, pp. 15 505–15 520, 2018.

[96] N. Wang, T. Yao, J. Wang, and D.-Y. Yeung, "A probabilistic approach to robust matrix factorization," in *European Conference on Computer Vision*, Springer, 2012, pp. 126–139.

[97] N. Wang and D.-Y. Yeung, "Bayesian robust matrix factorization for image and video processing," in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, pp. 1785–1792.

[98] R. Wang, F. Bunyak, G. Seetharaman, and K. Palaniappan, "Static and moving object detection using flux tensor with split gaussian models," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014, pp. 414–418.

[99] Y. Wang, P.-M. Jodoin, F. Porikli, J. Konrad, Y. Benezeth, and P. Ishwar, "Cdnet 2014: An expanded change detection benchmark dataset," in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2014, pp. 387–394.

[100] Y. Wang, Z. Luo, and P.-M. Jodoin, "Interactive deep learning method for segmenting moving objects," *Pattern Recognition Letters*, vol. 96, pp. 66–75, 2017.

[101] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland, "Pfinder: Real-time tracking of the human body," *IEEE Transactions on pattern analysis and machine intelligence*, vol. 19, no. 7, pp. 780–785, 1997.

[102] P. Xu, M. Ye, X. Li, Q. Liu, Y. Yang, and J. Ding, "Dynamic background learning through deep auto-encoder networks," in *Proceedings of the 22nd ACM international conference on Multimedia*, ACM, 2014, pp. 107–116.

[103] Y. Xu, J. Dong, B. Zhang, and D. Xu, "Background modeling methods in video analysis: A review and comparative evaluation," *CAAI Transactions on Intelligence Technology*, vol. 1, no. 1, pp. 43–60, 2016.

[104] L. Yang, J. Li, Y. Luo, Y. Zhao, H. Cheng, and J. Li, "Deep background modeling using fully convolutional network," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 254–262, 2018.

[105] D. Zeng and M. Zhu, "Background subtraction using multiscale fully convolutional network," *IEEE Access*, vol. 6, pp. 16 010–16 021, 2018.

[106] X. Zhang, Y. Tian, T. Huang, S. Dong, and W. Gao, "Optimizing the hierarchical prediction and coding in hevc for surveillance and conference videos with background modeling," *IEEE transactions on image processing*, vol. 23, no. 10, pp. 4511–4526, 2014.

[107] Z. Zhang, Y. Chang, S. Zhong, L. Yan, and X. Zou, "Learning dynamic background for weakly supervised moving object detection," *Image and Vision Computing*, vol. 121, p. 104 425, 2022.

[108] W. Zheng, K. Wang, and F.-Y. Wang, "A novel background subtraction algorithm based on parallel vision and bayesian gans," *Neurocomputing*, vol. 394, pp. 178–200, 2020.

[109] T. Zhou and D. Tao, "Godec: Randomized low-rank & sparse matrix decomposition in noisy case," in *International conference on machine learning*, Omnipress, 2011.

[110] X. Zhou, C. Yang, and W. Yu, "Moving object detection by detecting contiguous outliers in the low-rank representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 3, pp. 597–610, 2013.

[111] Z. Zivkovic, "Improved adaptive gaussian mixture model for background subtraction," in *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, IEEE, vol. 2, 2004, pp. 28–31.

[112] Z. Zivkovic and F. Van Der Heijden, "Efficient adaptive density estimation per image pixel for the task of background subtraction," *Pattern recognition letters*, vol. 27, no. 7, pp. 773–780, 2006.