Livening up and Segmentation with Blur

A Project Report

submitted by

ANSHUL BHAVESH SHAH

in partial fulfilment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY AND MASTER OF TECHNOLOGY

DEPARTMENT OF ELECTRICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY MADRAS.

May 2018

THESIS CERTIFICATE

This is to certify that the thesis titled **Livening up and Segmentation with Blur**, submitted by **Anshul Bhavesh Shah**, to the Indian Institute of Technology, Madras, for the award of the degree of **Bachelor of Technology and Master of Technology**, is a bona fide record of the research work done by him under our supervision. The contents of this thesis, in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Prof. A. N. Rajagopalan
Research Guide
Professor
Dept. of Electrical Engineering
IIT-Madras, 600 036

Place: Chennai
Date: 10 May 2018

ACKNOWLEDGEMENTS

I would like to thank Prof. A.N. Rajagopalan for being an an excellent mentor, guide and a teacher. Thank you for always being there for discussions and brain-storming sessions. I am fortunate to have been a part of the IPCV lab which has an excellent research atmosphere. Working on this project for the past year at IPCV lab gave me insight into research in academia and motivated me to pursue a PhD in the field of Computer Vision.

I am happy to have worked with Kuldeep for my projects. This research would not have been possible without your enthusiasm and dedication. I thank you for motivating me when things did not work as expected. I would also like to thank my labmates from IPCV lab Nimisha, Subeesh, Mahesh, and Srimanta for always being there whenever I needed help and for making the lab a fun place to work. Thanks to Bharat and Sunil for the interesting discussions and making the lab lively.

The thesis would not have been possible without the support of my wingmates. Thanks for bearing with me for my long hours in lab.

Last but not the least, I thank my family for being a strong pillar of support.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

CNN	Convolutional Neural Networks
RNN	Recurrent Neural Networks
ReLU	Rectified Linear Unit
Tanh	Hyperbolic Tangent
LSTM	Long Short Term Memory
BPTT	Backpropogation Through Time
ConvLSTM	Convolutional LSTM
STN	Spatial Transformer Networks
RVE	Recurrent Video Encoder
RVD	Recurrent Video Decoder
PSNR	Peak Signal to Noise Ratio
TV	Total Variation
MRF	Markov Random Fields

NOTATION

o_t	Output gate
i_t	Input gate
W_o, i, f, a	Weights for output, input, forget and state
b_o, i, f, a	bias for output, input, forget and state
x_i	Image frames
x_b	Blurred image
$\hat{x_{\frac{N}{2}}}$	Deblurred sharp frame
$\hat{x_i}$	Generated frames
h_n^{enc}	Hidden state of encoder at step n
$h_{e,N}$	Last hidden state of encoder n
$f_{n,i}$	Flow map at step n and scale i
h_n^{dec}	Hidden state of decoder at step n
$\mathcal{L}_n$	Loss at step n
$b(p)$	Blur probabilities
k_i	kernel i

CHAPTER 1

INTRODUCTION

In this thesis we explore solutions to two research problems from Computer Vision - 1) Reliving blurred moments and 2) Defocus Density estimation.

Reliving blurred moments: In this work we ask a simple question : Is it possible to relive the scene as the camera experienced during exposure? Motion blurred images are known to be a nuisance for many computer vision applications. But, in this work we show that the presence of blur in images can reveal information about how the scene evolved during capture and go on to create a video. We solve this highly ill-posed problem using Deep neural networks which have revolutionised the field of Computer Vision. While image deblurring is an active research area in computer vision, our work on extracting a video and not a single frame is the first of its kind.

Defocus density estimation: Here we explore the problem of detection and segmentation of blurred regions in images. This is useful as it can help in blur magnification to obtain the desired amount of Bokeh effect or perform image matting. We use two CNN based architectures to extract the low level and global blur information from the image and then combine these to segment the image using a MRF based framework.

1.1 Thesis outline

This thesis is organized as follows : The second chapter introduces some core ideas and concepts related to the two works. The third chapter addresses the

problem of extracting a video from a single motion blurred image and the fourth chapter gives our approach towards estimating blur estimation and segmentation.

1.2 Summary of contributions

The following is the summary of the major contributions of this thesis :

- In Chapter 3, we present a novel work on generating a video from a motion blurred image. We specifically focus on the Video autoencoder which serves as a proxy network for the final task of Blurred image to video generation. The training of this network helps simplify the non-trivial task. The architectural design allows the network to seamlessly handle spatio-temporal information. Further, it makes the learnt networks more interpretable. The Recurrent Video Decoder can be directly used in the subsequent training of the Blurred Image Encoder. The work on BIE(Blurred Image Encoder) and onwards was done by other members in our lab. However, we discuss those in the thesis for completeness.
- In Chapter 4, we propose two networks for detection of blur regions in the image and segment the image. This thesis specifically focuses on the patch level regression network which allows us to extract local blur information from the scene. This information is combined with the image level blur information coming from the Image Level Regression network to give the final blur map. The work on Image Level Regression network and onwards was done by other members in our lab but we mention it here for completeness.

CHAPTER 2

Relevant concepts

2.1 Motion Deblurring

2.1.1 Motion Blur

While imaging has improved by leaps and bounds in the recent years, captured content still faces suffers from a lot of degradations and artifacts. Motion blur is a primary problem. In Vasiljevic *et al.* (2016), it has been shown that standard network models used for vision tasks and trained only on high-quality images suffer a significant degradation in performance when applied to images degraded by blur. It arises due to the change in scene that is seen by the camera during the exposure time. It can occur due to camera motion, object motion or both. This is usually the case in low-light situations where the exposure time of each frame is high and in scenes where significant motion happens within the exposure time.

2.1.2 Deblurring using traditional approaches

Motion deblurring is a challenging problem in computer vision due to its ill-posed nature. Recent years have witnessed significant advances in deblurring Vasu and Rajagopalan (2017) Pan *et al.* (2016*c*)Pan *et al.* (2014). Numerous methods Xu and Jia (2010), Pan *et al.* (2016*b*), Krishnan *et al.* (2011) and Xu and Jia (2010) have been proposed to address this problem using hand-designed priors for representing sharp images.

2.1.3 Deblurring using Learning based approaches

Few approaches Chakrabarti (2016)Schuler *et al.* (2013)Schuler *et al.* (2016) estimate the latent sharp image using Convolutional Neural Networks (CNN) but include an additional blur kernel estimation step. Most of them account for camera motion alone thus severely limiting the possibilities. Few methods Pan *et al.* (2016*a*) Hyun Kim *et al.* (2013) segment images into regions with and without blur, and are thus restrictive in practice. A few methods Sun *et al.* (2015) Gong *et al.* (2017) have been proposed to remove heterogenous blur but are limited in their capability to handle general dynamic scenes. Since these methods strongly rely on the accuracy of the assumed image degradation model and its estimation, they do not perform well in real-world scenarios. The methods of Nah *et al.* (2016) and Nimisha *et al.* (2017) are able to overcome these limitations to some extent by learning to produce the latent sharp image, without the need for blur kernel estimation by using a CNN based architecture.

2.2 Convolutional Neural Networks

Convolutional Neural Networks (CNN) have revolutionized the field of Computer Vision. The interest in CNN started with AlexNet in 2012 and it has grown exponentially. They effectively help the computer *see* and have been used for a lot of vision tasks like classification, segmentation, enhancement, etc and they beat traditional methods by a large margin for most of image related tasks. The surge of interest in CNN has led to introduction of many different architectures, additions and modifications to the original CNNs.

Convolution Layer: Convolution layers form the major building block of a

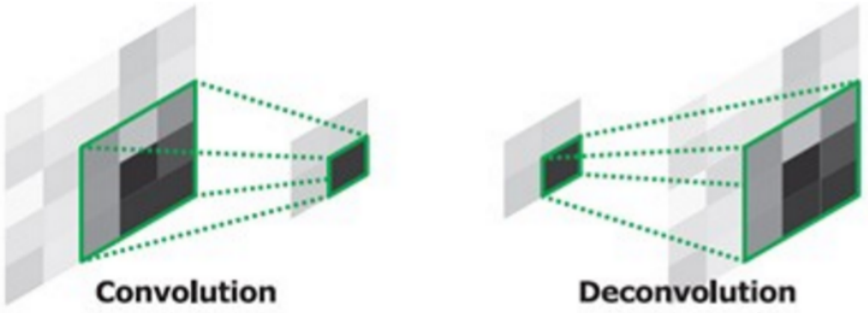

Figure 2.1: Convolution and Deconvolution layers

CNN. Here we work with 2D convolution as we are dealing with 2D feature maps. This layer is much more efficient than a fully connected layer used in other deep learning applications. Parameter sharing enforces that the feature maps are more meaningful and the number of parameters are not as high as fully connected layers. These layers are typically followed by non-linearities like ReLU, Sigmoid, TanH. Pooling is also used to reduce the dimensionality which reduces the number of parameters and combats overfitting. Commonly used types of pooling are Average pooling and max pooling.

Transposed Convolution Layers: Transposed Convolution Layers or fractionally strided convolution are used to increase the resolution of the feature maps after series of downsampling operations reduce the size of the input. Since, a naive upsampling inadvertently loses details, these layers are trainable. These are often used in Encoder Decoder networks.

2.3 Recurrent Neural Networks and ConvLSTMs

Recurrent Neural Networks (RNN) are neural network models that are often used to model sequences. They help account for dependence between inputs and can

handle variable number of inputs. The do this by sharing of parameters between the various time steps and recurring on a high dimensional vector (called the hidden state).

These networks are trained using an algorithm known as backpropogation through time (BPTT). These networks are often notoriously difficult to train due to vanishing and exploding gradients problem.

2.3.1 LSTM

It is a variant of RNN. The use of LSTM (Long Short Term Memory) cells have been instrumental in solving the above problem. The intuition is that they offer selective read, selective write and selective forget capabilities. The full set of equations for LSTMs are as follows :

Gates : The gates help to selectively pass on information.

$$o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o)$$

$$i_t = \sigma(W_i h_{t-1} + U_o x_t + b_o)$$

$$f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$$

States : The states hold the information over time steps.

$$\tilde{s_t} = \sigma(W_a h_{t-1} + U_a x_t + b_a)$$

$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s_t}$$

$$output_t = o_t \odot \sigma(s_t)$$

where o_t is the output gate, i_t is the input gate and f_t is the forget gate. The subscripts denote the step in the sequence. s_t is the hidden state and $output_t$ is the output of the cell. $W_{o,i,f,a}, U_{o,i,f,a}$ and $b_{o,i,f,a}$ are the weights that are learned.

2.3.2 ConvLSTM cells

ConvLSTM (Convolutional LSTM) is a special variant of LSTM cell which was introduced by Xingjian *et al.* (2015) in 2015 for the task of Precipitation Nowcasting. They extended the fully connected LSTMs to have convolutional structures in input-state and state-state transitions. Effectively, matrix multiplication is replaced by convolution operations. The use of ConvLSTMs, enables efficient flow of spatio temporal data in videos. Following are the equations of ConvLSTM cells.

Gates :

$$o_t = \sigma(W_o * h_{t-1} + U_o * x_t + b_o)$$
$$i_t = \sigma(W_i * h_{t-1} + U_o * x_t + b_o)$$
$$f_t = \sigma(W_f * h_{t-1} + U_f * x_t + b_f)$$

States : The states hold the information over time steps.

$$\tilde{s_t} = \sigma(W * h_{t-1} + U * x_t + b)$$
$$s_t = f_t \odot s_{t-1} + i_t \odot \tilde{s_t}$$
$$h_t = o_t \odot \sigma(s_t)$$

2.4 Spatial Transformer Networks

Spatial Transformer Networks (STN) were originally introduced by Jaderberg *et al.* (2015) for improving the classification accuracy of networks. Their motivation was that the classification models must be robust to input variations like scale, viewpoint variation and deformation. Though pooling layers impart some amount of the required invariance they do so by discarding a lot of data in the process. The idea of STN is to provide CNNs with explicit spatial transformation capabilities. STN layers are modular, differentiable and dynamic which have made them very popular.

The module itself consists of 3 components :

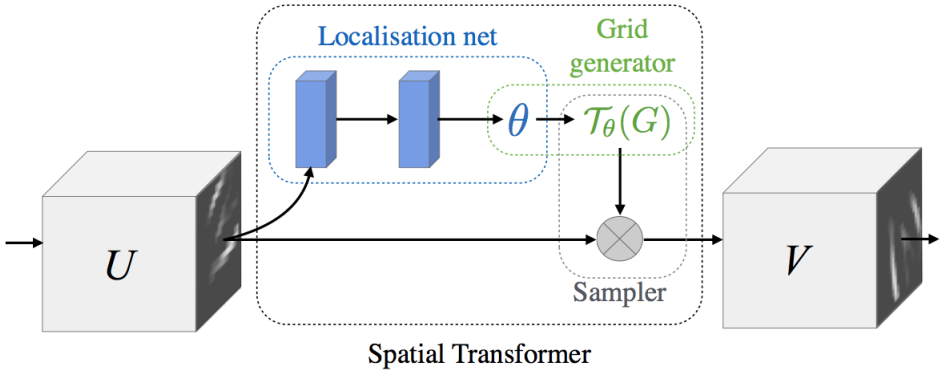

Figure 2.2: Visualization of STN architecture

Localization network : The localization network takes the input feature map through a number of hidden layers and results in the parameters of the spatial transformation that should be applied to the feature map. For example, 6 parameters are required for affine transformation. Thus this component produces a transformation that is conditioned on the input.

Grid Generator : The predicted transformation parameters are used to generate a sampling grid, which is a set of points where the input map must be sampled to produce the transformed output. This module effectively creates the inverse mapping as done in source to target mapping.

Sampler : This is the final component of the layer. Here, the feature map and the sampling grid is taken as input and the resulting output map is generated which is sampled from the inputs at the grid points. Billinear interpolation is used to sample points from the input.

CHAPTER 3

Reliving Blurred Moments

3.1 Overview

Given a blurred image, humans can mentally reconstruct (sometimes ambiguously perhaps) a temporally coherent account of the scene that represents what transpired during exposure time. However, in computer vision, natural video modelling and extraction has remained a challenging problem due to the complexity and ambiguity inherent in video data. Videos comprise a large majority of the visual data in existence, surpassing by a wide margin 'still' images, and they contain diverse spatio-temporal variations that conventional models are often incapable of synthesizing. With the success of deep neural networks in solving vision tasks, end-to-end deep networks have emerged as incredibly powerful tools. They are especially suited for video interpolation and extrapolation tasks, where training data is virtually infinite, since any video can be used to train such an unsupervised deep network.

Few works on visual prediction focus on modeling and estimation in pixel space, i.e. reconstructing images by calculating pixel values directly. In particular, Kalchbrenner *et al.* (2016) proposes a video pixel network and estimates the discrete joint distribution of the raw pixel values. Srivastava *et al.* (2015) uses a Long Short-Term Memory (LSTM) network to learn representations of video and predict future frames from it. Vondrick *et al.* (2016) employs adversarial training and generates intensities from scratch. However, since the future is similar

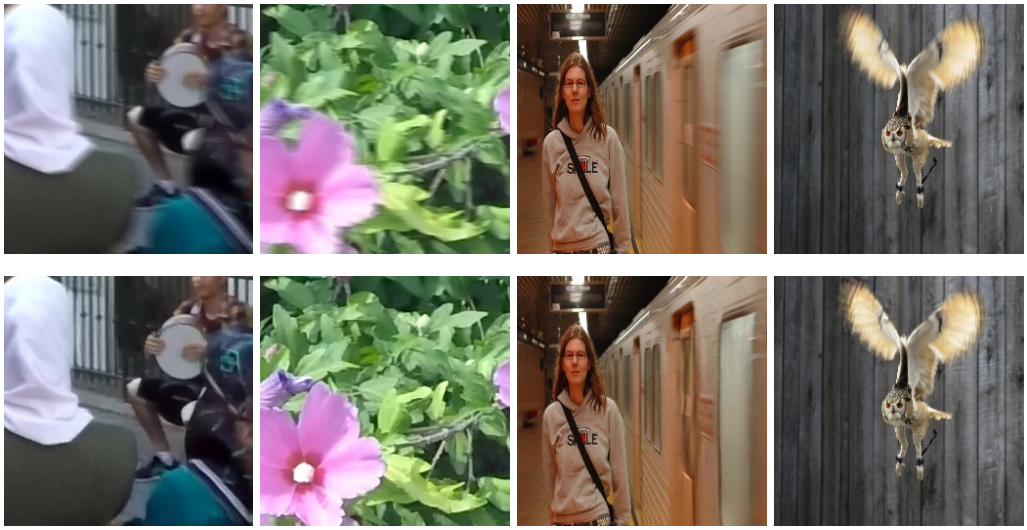

Figure 3.1: Videos generated by our method. The first row shows input blurred images while the second row contains the generated videos. *Videos can be viewed by clicking on the image, when document is opened in Adobe Reader.*

to the past, models which directly regress to pixel intensities need to inherently store low-level details (e.g., colors or edges) of the input image at every layer of representation. Not only is this inefficient use of network capacity, but it may also make it difficult for the network to learn desirable invariants that are necessary for motion prediction. Recent works on future frame prediction reveal that direct intensity estimation leads to blurred predictions but motion prediction resolves this issue. If a frame is reconstructed based on the original image and corresponding transformations, both scene dynamics and invariant appearance can be preserved well. Based on this premise, Flynn *et al.* (2016),Zhou *et al.* (2016) and Liu *et al.* (2017) model the task as a flow of image pixels.

Though there has been a lot of research in deblurring, we wish to highlight here that all existing methods limit themselves to the task of generating 'a' deblurred image. In this paper, we ask an interesting question. Given a blurred

frame, is it possible to revive and relive all the sharp views of the scene as seen by the camera during its flight within the exposure time? To the best of our knowledge, the problem of extracting an ordered motion sequence from a single blurred image has *never* been addressed in the literature. Even under the very strong assumption of static and planar scenes, state-of-the-art deblurring methods such as Vasu and Rajagopalan (2017) Pan *et al.* (2016*c*) estimate at best a group of poses which constitute the camera motion, but with total disregard to their ordering. As a post-processing step, synthesising a sequence from this group of poses is a non-trivial task. Although the camera trajectory can be tapped to an extent by gyroscope sensors attached to modern cameras, the obtained data is too sparse to completely describe trajectories within the time interval of a single lens exposure. More importantly, sensor information is seldom available for most internet images. Further, these methods can only handle blur induced by a camera imaging a static planar scene which is not representative of a typical real-world scenario and hence not very interesting.

The problem of extracting video from a single blurred frame is challenging due to the fact that a blurred image can only reveal aggregate information about the scene during exposure. The task requires recovery of frames which are temporally consistent in the sense that they emulate recording coming from a high frame-rate camera. All the frames should be sharp and the scene content must be preserved throughout. In this paper, we present a novel solution to this interesting and challenging problem using deep networks and reveal that a plausible temporal sequence can be reconstructed. The deep architecture that we present can extract latent motion representation from arbitrary motion blurred images and is applicable to general motion caused by individual or combined effects of camera motion, object motion and arbitrary depth variations in the scene. Our approach is based upon the premise that a deep neural network can learn to incorporate

visual dynamics of the world by observing a variety of videos. Our network is required to estimate spatio-temporal motion trajectories in an unsupervised manner. We demonstrate that it is possible to judiciously extract relevant information about a scene from its blurred image that suffices to solve the task of generating a plausible realistic video from a single blurred image. Fig. 3.1 shows the videos generated by our network and the corresponding input blurred images.

The main contributions of this work are:

- First ever attempt at extracting a sharp video from embedded motion information in a single blurred frame.
- We propose an end-to-end architecture which learns ordered spatio-temporal motion representation in an unsupervised manner.
- The capacity of our network is quite general and it can handle blur variations caused by camera motion, object motion, depth changes or even a combination of these.

3.2 Proposed Architecture

Convolutional neural networks (CNNs) have been successfully applied for various vision tasks on images but translating these capabilities to video is non-trivial due to their inability to exploit temporal redundancies present in videos. Recent developments in recurrent neural networks provide powerful tools for sequence modeling as demonstrated in speech recognition Graves *et al.* (2013) and caption generation for images Vinyals *et al.* (2015). Long short term memory networks (LSTMs) can be used to generate outputs that are highly correlated along the temporal dimension and hence form an integral part of our video generation framework. Considering that we are working with images, the spatial information across the image is equally important. Hence we use Convolutional LSTM

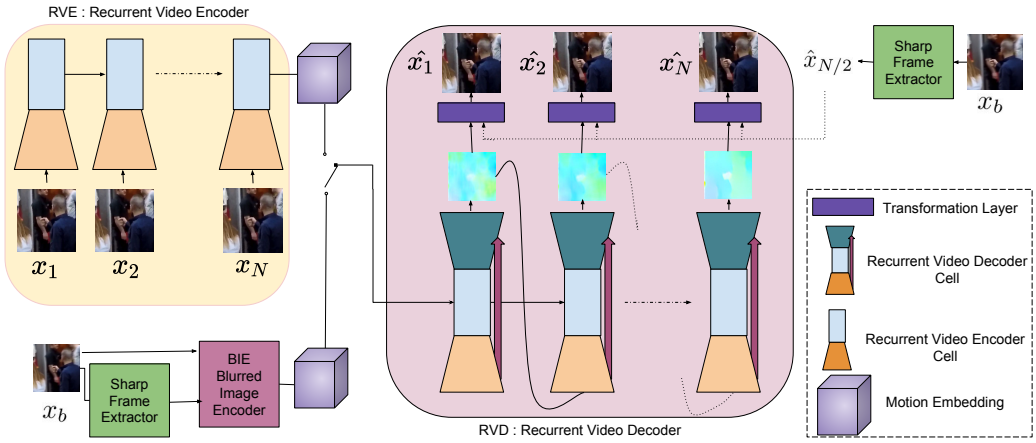

Figure 3.2: Our architecture. The first step involves training the RVE-RVD for the task of video reconstruction. This is followed by guided training of BIE-RVD.

units Xingjian *et al.* (2015) as our building blocks, which are capable of capturing both spatial and temporal dependencies since they employ convolutions for input-to-state, state-to-state and output-to-state transformations.

The task of generating an image sequence requires the network to understand and efficiently encode static as well as dynamic information for a certain period of time. Although such an encoding is not clearly defined and hence unavailable in labeled datasets, we overcome this challenge by unsupervised learning of motion representation.

We accomplish the task of extracting a video from a single blurred image by introducing a video autoencoder and a Blurred Image Encoder (BIE). The section on BIE onwards was done by another member of the lab and the same can be found in the paper Purohit *et al.* (2018) which we co-authored.

We propose to use video reconstruction as a surrogate task for training our blur-to-video generator. Our hypothesis is that a successful solution to the video

reconstruction task will allow a video autoencoder to learn a strong and meaningful motion representation which will enable it to impart spatio-temporal coherence to the generated moving scene content.

We propose a video autoencoder in which the encoder utilizes all the video frames to extract a latent representation. This is then fed to a video decoder which estimates the frame sequence in a recurrent fashion. We build the recurrent encoder-decoder using ConvLSTM modules with improved motion capture capabilities. The Recurrent Video Encoder (RVE) reads N sharp frames $x_{1..N}$, one at each time-step. It returns a tensor at the last time-step, which is utilised as the motion representation of the image sequence. This tensor is used to initialise the first hidden state of another ConvLSTM based network called Recurrent Video Decoder (RVD) whose task is to recurrently estimate N optical flows. Since the RVE-RVD pair is trained using reconstruction loss between the estimated frames $\hat{x}_{1..N}$ and ground truth frames $x_{1..N}$, the RVD must return the predicted video. To enable this, a sharp frame (typically the central frame $x_{N/2}$) is acted upon by the flows. The estimated flows are individually fed to a differentiable transformation layer to transform the sharp frame $x_{N/2}$ to obtain the frames $\hat{x}_{1..N}$. In addition to the reconstruction loss, a flow smoothness loss is also used. Once trained, we have an RVD which can estimate sequential motion flows, given a particular motion representation.

In addition, we introduce another network called Blurred Image Encoder (BIE) whose task is to accept blurred image x_B corresponding to the spatio-temporal average of the input frames $x_{1..N}$ and return a motion encoding, which too can be used to generate a sharp video. To achieve this task, we employ the already trained RVD to guide the training of BIE so as to extract the same motion information from the blurred image as the RVE would from an image sequence. In

other words, the weights are to be learnt such that $BIE(x_B) \approx RVE(x_{1..N})$. We refrain from using the encoding returned by RVE for training due to lack of ground truth for the encoded representation. Instead, the BIE is trained such that the predicted video at the output of RVD for the given x_B matches as closely as possible to the ground truth frames $x_{1..N}$. Directly training the BIE-RVD pair is cumbersome and may lead to the RVD learning to extract motion whereas this task is to be performed by RVE/BIE in the manner we have framed the problem. RVE-RVD training renders the learnt motion representation interpretable. We train the BIE by coupling it with the trained RVD to perform the video generation task. This ensures that the BIE learns to capture ordered motion information and the RVD returns a realistic video.

A Fundamental Ambiguity: It is important to realize that estimating direction of motion from a single blurred image is fundamentally ambiguous. For example, one would get the same blurred image even if the temporal order is reversed. The use of optical flow recurrence, enables the network to prefer temporally consistent sequences, which prempts it from returning arbitrarily ordered frames. However, the directional ambiguity stays. For a scene with multiple objects, the ambiguity becomes more pronounced as each object can have its own independent motion. Humans can (partly) resolve this ambiguity by seeing millions of videos during their lifetime and thus understand how things move. Gong *et al.* (2017), which estimates a single optical flow map for image deblurring dealt with this ambiguity by constraining the optical flow to be non-negative in the horizontal direction during dataset creation. Due to the unsupervised nature of our approach, we do not enforce such a constraint. Nevertheless, we mitigate this issue by feeding to the BIE a directionally-biased sharp frame in addition to the blurred frame by considering an even number of frames N and picking the $\frac{N}{2}^{th}$ frame as the sharp frame. The BIE then leverages the asymmetric blur around this sharp frame to estimate

(a) Recurrent Video Encoder architecture. (b) Blurred Image Encoder architecture.

Figure 3.3: Architectures of BIE and RVE. The RVE is trained to extract a motion representation from a sequence of frames while the BIE is trained to extract a motion representation from a blurred image and a sharp image.

the direction of motion. The choice of an even number of frames to arrive at the blurred image ensures that there is no *true middle* frame as it will be devoid of directional information.

The overall architecture of the proposed methodology is given in Fig. 3.2. The proposed architecture is fully convolutional, end-to-end differentiable and can be trained using unlabeled high frame-rate videos. Our method requires only sequences of sharp images captured at high frame rate for training, without the need for optical flow supervision, which is challenging to produce at large scale. We now describe design aspects of the different modules of our network.

3.2.1 Recurrent Video Encoder (RVE)

At each time-step, a frame is fed to a convolutional encoder, which generates a feature-map to be fed as input to the ConvLSTM cell. Interpreting ConvLSTM's hidden-states as a representation of motion, the kernel-size of a ConvLSTM is correlated with the speed of the motion which it can capture. Since we need to extract motion taking place within a single exposure at fine resolution, we choose

a kernel-size of 3×3. As can be seen in Fig. 3.3(a), the encoder block is made of 4 convolutional layers with 3×3 filters and stride of 2. The depth of the three layers are 32, 64 and 128, respectively. A ConvLSTM cell operates on this feature and augments it with memory from previous time-steps.

Overall, each module can be represented as $h_n^{enc} = enc(h_{n-1}^{enc}, x_n)$, where h_n^{enc} is encoder ConvLSTM state at time step n and x_n is the n^{th} sharp frame of the video.

3.2.2 Recurrent Video Decoder (RVD)

The task of RVD is to construct a sequence of frames using the motion representation provided by RVE and the central frame ($x_{\frac{N}{2}}$) of the video sequence. The RVD contains a flow encoder which utilizes a structure similar to the RVE. Instead of accepting images, it accepts optical flows. The flow encoding is fed to a ConvLSTM cell whose first hidden state is initialized with the last hidden state $h_{e,N}$ of the RVE. To estimate optical flows for a time-step, the output of the ConvLSTM cell is passed to a Flow decoder network (F_D). The flow estimated by F_D at each time-step is fed to a transformer module (T) which returns the estimated frame $\hat{x}_n$. The descriptions of F_D and T are provided below.

Flow Decoder (F_D): Realizing that the flow at current step is related to the previous one, we perform recurrence on optical flows for consecutive frames. The design of F_D is illustrated in Fig. 3.4. F_D accepts the output of ConvLSTM unit at any time-step and generates a flow-map. For robust estimation, we further perform estimation of flow at multiple scales using deconvolution (deconv) layers which "unpool" the feature maps and increase the spatial dimensions by a factor of 2. Inspired by Ronneberger *et al.* (2015), we make use of skip connections between the layers of flow encoder and F_D. All deconv operations use 4×4 filters

and the conv operations use 3×3 filters. The output of the ConvLSTM cell is passed through the a conv layer to estimate the flow $f_{n,1}$, and a deconv layer. The output of this deconv layer is concatenated with the upsampled $f_{n,1}$ and the feature representation coming from the encoder, to obtain a feature map at that scale. This step is repeated 3 more times to obtain the flow maps at different scales ($f_{n,2..4}$).

Transformer(T): This generates a new frame by transforming a sharp frame using the output returned by F_D. It is a modified version of the Spatial Transformer Layer Jaderberg *et al.* (2015), wherein instead of a single transformation for the entire image as originally proposed in Jaderberg *et al.* (2015) this transformer accepts one transformation per pixel. Since we focus on learning features for motion prediction, it provides immediate feedback on the flow map predicted by the optical flow generation layers. Effectively, the decoder function can be summarized as follows:

$$h_1^{dec} = h_N^{enc} \tag{3.1}$$

$$h_n^{dec}, f_{n,1..4} = G(h_{n-1}^{dec}, f_{n-1,4}) \tag{3.2}$$

$$\hat{x}_{n,1..4} = T(\hat{x}_{\frac{N}{2}}, f_{n,1..4}) \tag{3.3}$$

for $n \in$ [1,N] where h_n^{dec} is the decoder hidden state, $f_{n,1..4}$ are the flows predicted at n and $\hat{x}_{n,1..4}$ are the sharp frames predicted at different scales and G refers to a recurrent cell of RVD.

3.2.3 Blurred Image Encoder (BIE)

The RVE discussed in section 2.1 learns to extract motion information in a manner that allows the RVD to reconstruct the sequence of flows which is applied to

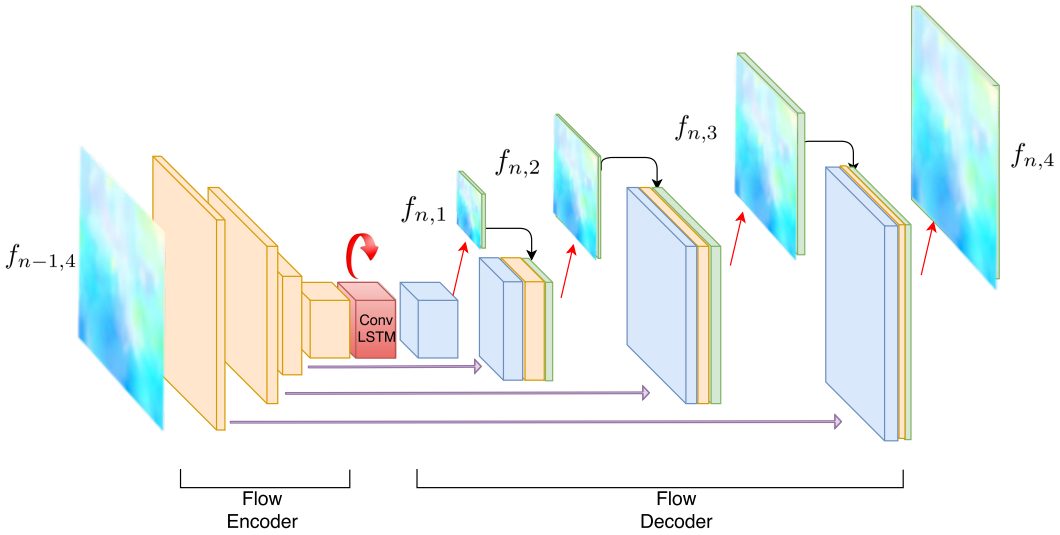

Figure 3.4: Our Recurrent Video Decoder (RVD). This module recurrently generates optical flows which are warped to transform the sharp frame. Flows are estimated at 4 different scales.

get frames in the same temporal order as the input. We make use of this trained encoder-decoder couplet to solve the task of extracting video from a blurred image. The trained decoder has learnt to generate optical flow for all time-steps from the encoder's hidden state. We now make use of this proxy network to solve the task of blurred image to video generation.

The BIE is implemented as a convolutional neural network which specializes in extracting motion features from a blurred image. The BIE is tasked to extract the sequential motion in the image by capturing local motion, e.g. at the smeared edges in the image. Moreover, the generated encoding should be such that the RVD can reconstruct motion trajectories. The BIE has 7 convolutional layers with kernel sizes as shown in Fig. 3.3(b). Each layer (except the last) is followed by batch-normalization and leaky ReLU non-linearity.

3.3 Cost Function

Both our network pairs (RVE-RVD and BIE-RVD) are end-to-end trainable and we use the same cost function for training for both the tasks. This is achieved by calculating the cost on the flows and frames estimated by the RVD. The loss function quantifies the discrepancy between the target video and the one generated by the network. Since the network is essentially estimating optical flows, we utilize a cost function motivated by learning-free variational method Brox and Malik (2011) which resembles the original formulation of Horn and Schunck (1981). The data term assumes constancy over time of some image property and a spatial (and often smooth) term models how the flow is expected to vary across the image. At each time step, the data loss measures the discrepancy between intensities of target frame and the output of transformation layer (obtained using the the predicted optical flow field). Inspired by Sun *et al.* (2014), the data term is expressed as a Charbonnier penalty Bruhn *et al.* (2005) $C(s) = \sqrt{(s^2 + 0.0012)}$: a differentiable variant of the absolute value, and a robust convex function against outliers and noise. The smoothness cost is in form of a total variation-loss-penalty on the estimated flow maps: $TV(s) = \sum_i |\nabla_x s| + |\nabla_y s|$. This constrains them to be locally smooth and preserves the original image structure.

Coarse-to-Fine Estimation: Motivated by the approach employed in FlowNet Dosovitskiy *et al.* (2015), we improve our network's accuracy by estimating the flow-maps and frames in a coarse-to-fine manner. At each time-step, four loss terms are calculated using four optical flows $f_{n,1..4}$ predicted at sizes which are $(\frac{1}{8}, \frac{1}{4}, \frac{1}{2}, 1)^{th}$ fraction of the original image resolution and applied on the corresponding downsampled central frame using the transformation layers. Reconstruction losses are enforced at each scale using suitably down-sampled ground truth. Effectively, we use a weighted sum of loss to guide the information flow

over the network, as given below.

$$\mathcal{L}_n = \sum_{j=1}^{4} \lambda_j (C(\hat{x}_{n,j} - x_{n,j}) + \mu TV(f_{n,j})) \tag{3.4}$$

Here, j represents the scale, n represents time-step, and μ is the regularization weight empirically set to 0.02. The relative weights for each scale λ_js were adopted according to the loss weight schedule suggested in Mayer *et al.* (2016) which gradually trains the network from bottom loss layers to top ones, until adding them up for further training.

3.4 Implementation Details

Preparing the training data is crucial for the proposed representation learning problem. Recent works on image Nah *et al.* (2016) and video Su *et al.* (2016) deblurring have built a training data set by recording videos captured at 240 fps with a GoPro Hero camera to minimize the blur in the ground truth. We prepare our training data from the same source, wherein 35 full videos were used for creating training sets and 5 full videos were reserved for validation and testing. Each blurred image is produced by averaging 10 successive latent frames. Such an averaging simulates a photo taken at approximately 25 fps, while the corresponding sharp image shutter speed is $1/240$. We extract 256×256 patches from these image sequences for training. Finally, our dataset is composed of 10^5 sets, each containing $N = 10$ sharp frames and the corresponding blurred image x_B. We perform data augmentation by random horizontal flipping and zooming by a factor in the range [0.2,2]. Our video-autoencoder is trained using Adam optimizer with parameters β_1 = 0.9 and the β_2 = 0.999 and the learning rate is 2×10^{-4}.

The batch size was set to 10 and the entire training took 4.5×10^5 iterations to converge. We then train the BIE-RVD pair with the same training configuration and reduce the learning rate for RVD parameters to 2×10^{-5}, which enables stable training.

Sharp Frame Extraction: Note that during testing, to obtain the sharp image to be fed to RVD and BIE, we use an off-the-shelf single image deblurring method Nah *et al.* (2016), which is a multiscale CNN with residual connections. It is known to give good results even on images with significant amounts of blur. The output of Nah *et al.* (2016) is treated as the central sharp frame for our architecture. At test time, resolution of directional ambiguity cannot be ensured as the deblurring network may not necessarily provide the $\frac{N}{2}^{th}$ frame as its output. Nevertheless, our network returns a plausible video which is temporally consistent.

3.5 Experiments

We begin with quantitative evaluation of performance including ablation analysis. This is followed by qualitative results on video generation from a single blurred image.

3.5.1 Quantitative Results

We present experiments to analyze the various design decisions for training our network. These experiments highlight the effect and advantages of these design choices. We give average PSNR values for different configurations of our network. We also provide details of these experiments and explain how the values were arrived at. The reported PSNRs are calculated over 100 image sequences

taken from 5 test videos selected at random from GoPro dataset Nah *et al.* (2016).

Direct Intensity Estimation: We experimented with training our video autoencoder network to directly estimate pixel intensities instead of optical flows but found that it tends to learn an identity mapping. Such an autoencoder does not help in our task of learning motion representations. Instead, we enforce motion learning explicitly by predicting optical flows instead of intensities. The encoding learnt by our method cannot be an identity mapping since robust optical flow prediction is not possible without capturing motion features.

Non-recurrent CNN Instead of RVE: For the task of video reconstruction, we also experimented by replacing our RVE with a CNN encoder. For this, we chose a CNN architecture similar to BIE but the number of filters was reduced to match the total number of parameters in our RVE. We fed stacked frames to this CNN-RVD and trained it for the video reconstruction task. However, it was not quite successful in reproducing local motion in dynamic videos and led to a lower PSNR of 23.9 dB against 27.4 dB achieved using RVE. This observation reaffirms the effectiveness of ConvLSTM modules to capture spatio-temporal dependencies.

Joint BIE-RVD Training: We found that joint training of BIE and RVD from scratch poses a formidable challenge and the performance is below par (PSNR 23.1 dB). Instead, our approach of training RVD for the surrogate task of extracting motion representation of short video sequences has the advantage that it renders the learned motion representation interpretable. Once RVD is trained, the BIE learns to extract the same motion information as RVE.

BIE sans Sharp Image: We tried training a network without a sharp image being fed to the BIE. This network gave higher errors than the one which uses a sharp image. We suspect that this is because, when the network is fed with a sharp frame, it has a reference of which regions and which direction the blur is in.

Effect of Different Losses: We experimented with the effect of various losses

while training the RVE-RVD pair for video reconstruction. In the first configuration, both frame reconstruction loss and TV loss were calculated at only a single scale. This model did not perform well on sequences containing complex motion, and resulted in highly fluctuating optical flows (PSNR of 22 dB). In the second configuration, we removed the single scale TV loss but found that the predicted flows lost their spatial smoothness thus resulting in unrealistic results (PSNR 20 dB). When we include multi-scale frame reconstruction and TV loss with relative weights as mentioned in section 2.4 (which is the final loss setting that we use), we got a PSNR of 27.4 dB.

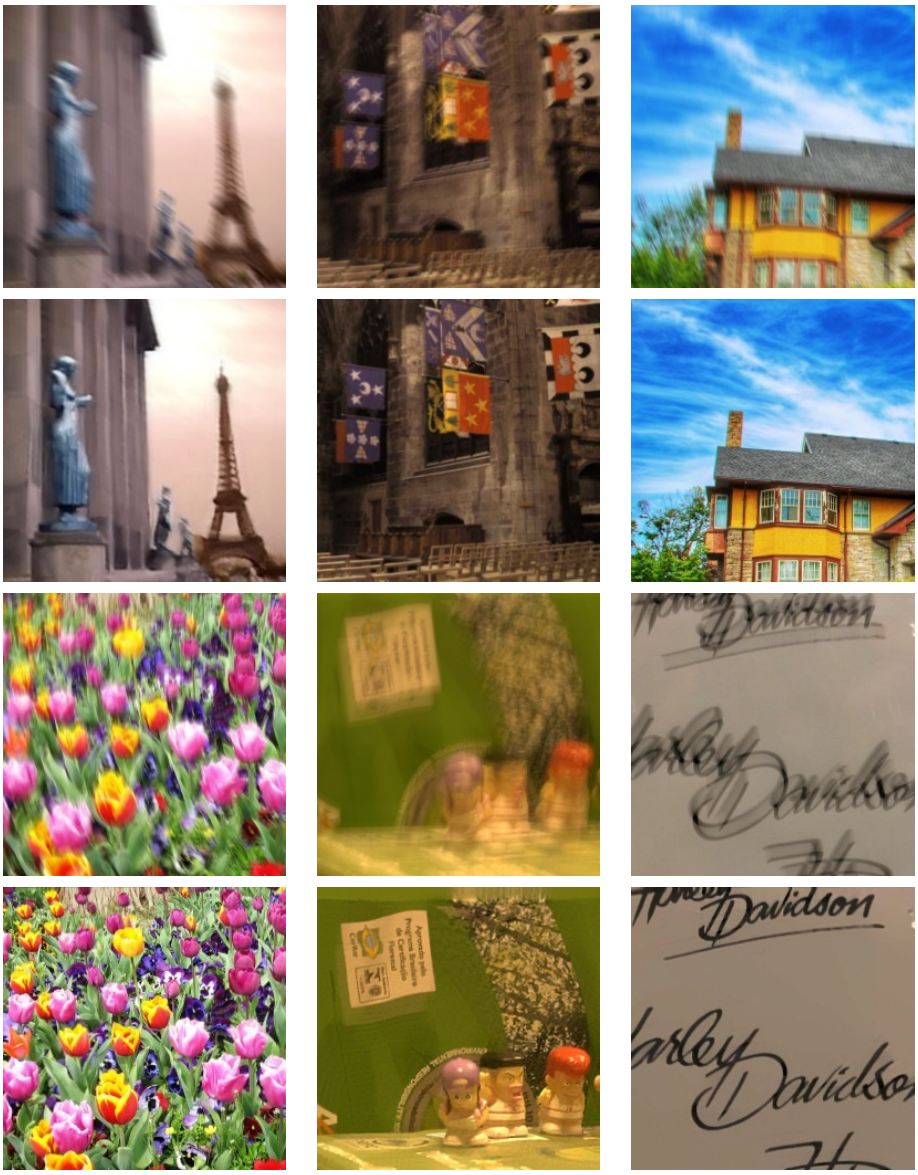

Figure 3.5: Video generation on images blurred with global camera motion from datasets of Lai *et al.* (2016) and Köhler *et al.* (2012). First row shows the blurred images and second row, the generated videos.

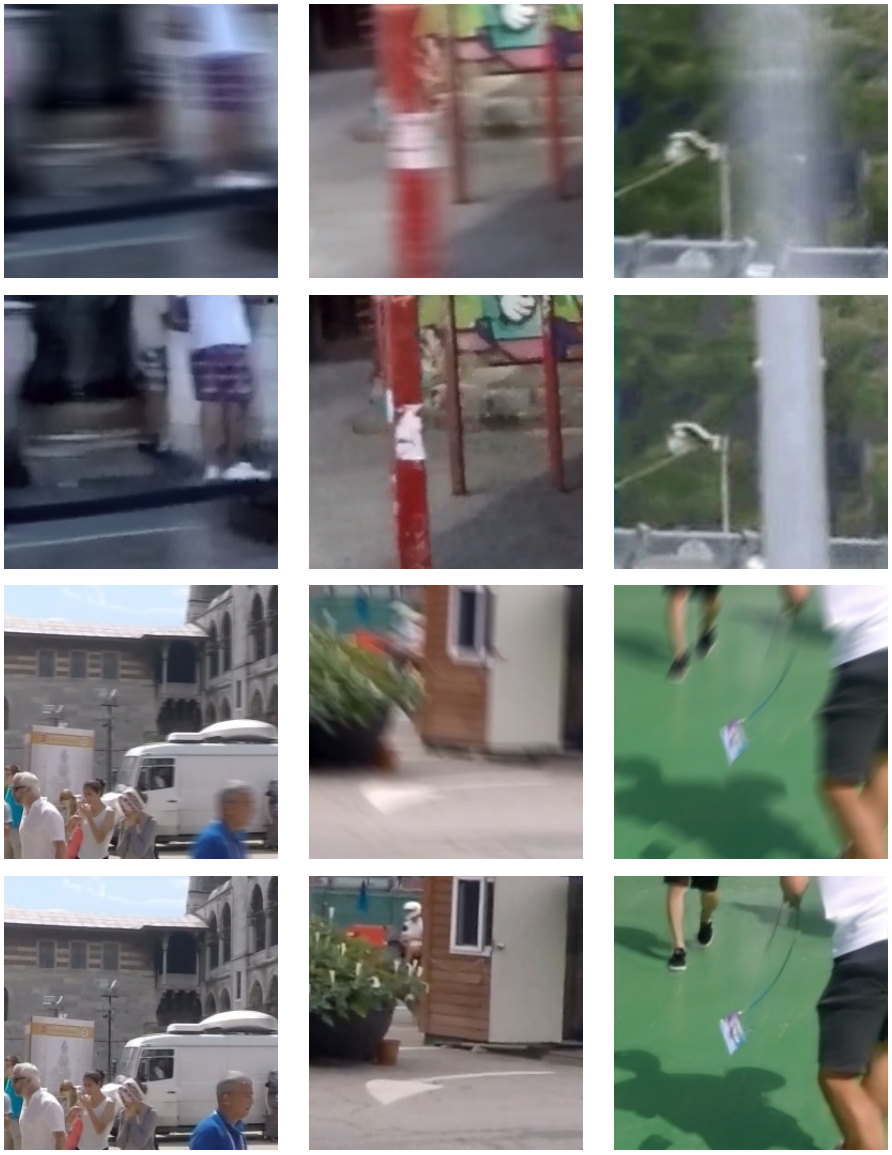

Figure 3.6: Our Results on motion blurred images obtained from dataset of Nah *et al.* (2016). First row shows the blurred images and second row, the generated videos.

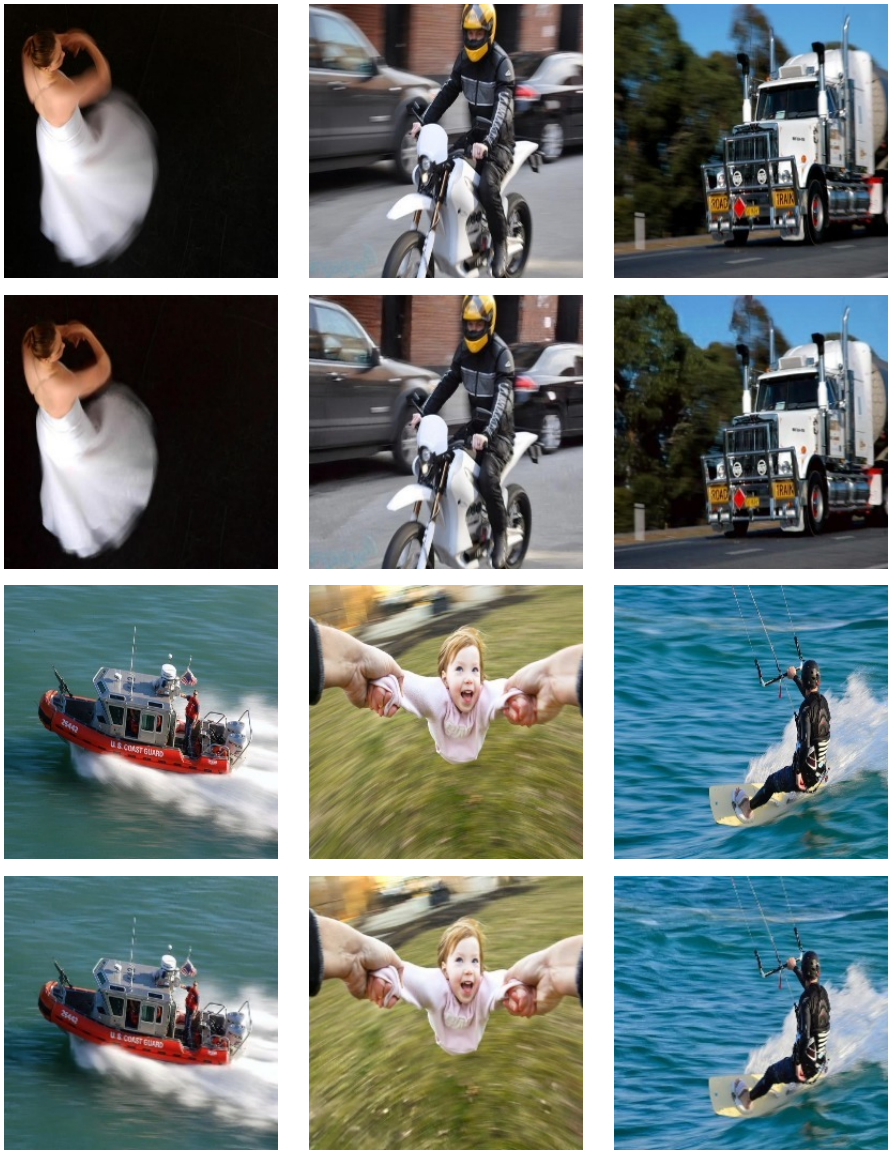

Figure 3.7: Our results on real motion blurred images obtained from dataset of Shi *et al.* (2014). The first and third rows show the blurred images and second and fourth rows show the corresponding generated videos.

3.5.2 Qualitative Results

We also qualitatively evaluated our network's ability to estimate a sharp video from a single blurred image (both synthetic and real) collected from different datasets.

Results on Camera Motion Dataset: For evaluating qualitative performance on videos with camera motion alone, we tested our network's capability to reconstruct videos from blurred images taken from datasets of Gong *et al.* (2017), Köhler *et al.* (2012) and Lai *et al.* (2016), which is commonly used for benchmarking deblurring techniques. Fig. 3.5 shows the result of our algorithm on images blurred using camera motion on Köhler *et al.* (2012) and Lai *et al.* (2016) datasets.

Results on Go-Pro Dataset: In Fig. 3.6, we show results obtained on test blurred images from dataset of Nah *et al.* (2016), which are synthetically created by averaging a number of frames. In addition to handling global camera motion, our method is able to generate realistic videos even in complicated scenarios including dynamic scenes as well which are very challenging for traditional deblurring approaches.

Results on Blur Detection Dataset: In Fig. 3.7, we show videos generated from real blurred images taken from the dataset of Shi *et al.* (2014) which contains dynamic scenes with large motion. The results establish that our network can sense direction and magnitude even in severely blurred images.

CHAPTER 4

Blur detection and segmentation

4.1 Overview

Technological advancements have revolutionized imaging industry but it is still a challenge to prevent or remove degradations that occur during image capture. To capture a well-defined frame, there is typically a compromise between aperture size and exposure time. A large aperture leads to reduced depth of field which can result in some regions being optically defocused in the image. To bring the entire 3D scene into focus, one must select a small aperture, and compensate for the reduced intensity by allowing a larger exposure time. But this increases the chances of motion blur if object and/or camera are in motion, which is often the case in cameras which are handheld or mounted over moving platform. Blur also has an aesthetic value in professional photography, as it is often used to highlight the salient regions in a static scene (defocus blur) or dynamic scene (motion blur). Although blur has traditionally been regarded as an undesirable effect, it is indirectly linked to higher level scene information. A few existing works Favaro and Soatto (2004), Deng *et al.* (2012) have used motion blur itself as a cue for segmentation, while others have exploited defocus blur as a cue to estimate depth [Chaudhuri and Rajagopalan (2012),Favaro *et al.* (2003)] or segment salient region and foreground from a scene. For object detection, the segmented sharp regions can be extracted for efficient region proposal and robust object localization [20,26]. Other applications include image quality assessment, photo editing and

image restoration. We show applications in image matting and blur magnification. Local blur detection, however, is a highly challenging task, not only because the estimated blur values vary spatially, but also because the estimated map contains ambiguities Park *et al.* (2017), where the appearances of two regions with different amounts of defocus can be very similar. Traditional approaches of determining the amount of blur only based on the strength of strong edges lead to overconfidence and errors. Moreover, proper thresholds need to be selected for such methods to perform well. These limitations call for more reliable and robust descriptors for defocus estimation. Recently, Park *et al.* (2017) experimentally determined that deep features extracted from an image patch possess much more discriminative power with regard to blurry and sharp regions as compared to other individual hand-crafted features.

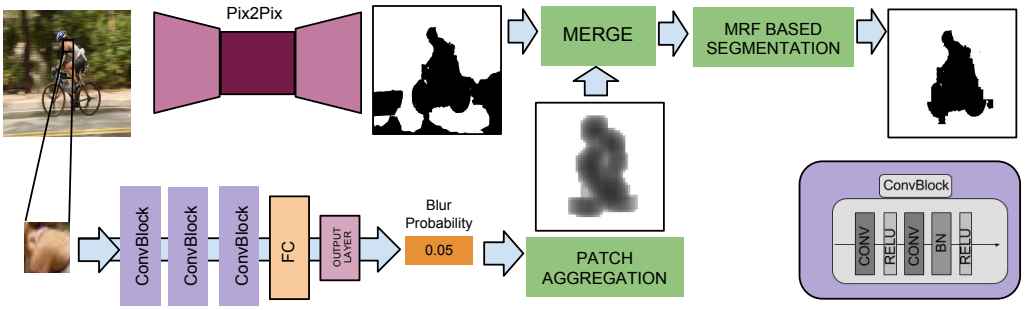

Figure 4.1: Our Blur-detection and Segmentation framework

In this paper, we propose to use end-to-end deep Convolutional Neural Networks (CNNs) to learn powerful features relevant to blur. Moreover, our method utilizes global context information to mitigate the ambiguity between blurred regions and sharp but homogenous regions in a given image. The formulation makes sure that estimates of two sub-networks do not corrupt into each-other but contribute constructively. The obtained blur-detection map is then passed through a MRF based framework to yield an accurate segmentation map.

The main contributions of this work are:

- This is the first end-to-end learning based framework for the task of blur detection from a single image, which can handle both defocus and motion blur.
- We train two CNNs to perform detection at both patch-level and image-level, and merge their estimated detection maps to obtain a robust probability of blur at each pixel.
- The method offers a fully automatic approach to perform sharp region segmentation and matting tasks using MRF-based formulation.

4.2 Related works

Classical depth from defocus (DFD) algorithms Chaudhuri and Rajagopalan (2012), Favaro *et al.* (2003) assume a stationary camera and a static scene, and use multiple images captured under different lens settings to recover the depth map using defocus metrics.

Tai and Brown (2009) estimates the defocus map by using a relationship between contrast and the image gradient. The method Zhuo and Sim (2011) extracts image gradients from both image and re-blurred images to obtain the defocus map. Key approaches which address only motion blur detection are: Chakrabarti *et al.* (2010) proposed 1-D box filters for spatially varying motion blur segmentation.Wang *et al.* (2017) proposed alternating between two tasks of blur kernel estimation and blur segmentation. However, their method is restricted to motion blur. The approach in Su *et al.* (2011) handles both types of blur by defining a singular value distribution feature and alpha-channel extraction step.

Learning based approaches like Liu *et al.* (2008) and Shi *et al.* (2014) propose to use several hand-designed local blur features dependent on local patch prop-

erties like power spectrum, histogram and gradient profiles and feed them to a classifier. Shi *et al.* (2015) proposed a defocus discriminating feature by directly finding a mapping between sparse edges and blur strength. But the method does not extend well to images with large amounts of defocus blur.

A recent method Park *et al.* (2017) proposed a shallow CNN to obtain deep feature representations of an image patch. They found that such patch level representations are useful but not sufficient in themselves for robust blur detection and hence they concatenate handcrafted features along with these representations for the task. Moreover, their method does not address motion blur.

Our method makes no assumptions about the scene and does not depend on hand-designed features, which enables it to overcome the limitations of prior works. Our method uses two sub-networks that integrate global and local estimates to enable end-to end blur detection from a single image. The deep features learnt by our method are generalized to both motion and defocus blur, as demonstrated in the experimental section.

4.3 Proposed Approach

Training Deep Neural Networks is data-intensive. However, to the best of our knowledge, the blur detection benchmark built by Shi *et al.* (2014) is the only database that is publicly available for the blur segmentation task and it contains only 1000 labeled examples. A network trained using such limited data may suffer from overfitting and not generalize to other scenes and all degrees of motion and defocus blur. To circumvent this issue, we propose to utilize two sub-networks.

The first one is a patch level classification network trained on a large dataset

of small image patches (30×30) obtained from synthetically blurred images. The second is a parametrically efficient image level regression network, which is trained using the dataset Shi *et al.* (2014) to estimate a segmentation map. The details of dataset generation are described in section 4.4.

The map obtained after the patch-aggregation step contains coarse boundaries and global inconsistencies, since the patch aggregation step predicts a label at each patch independently. Generally, the edges in this map do not align with the boundaries in the input image. Also, this process does not ensure consistency among neighboring pixels' labels, which may lead to assignment of opposite labels to neighboring patches on a sharp object if homogeneous regions are present in the vicinity. Such failures are avoided by integrating the blur probabilities obtained from the second sub-network.

The work on Image level network and segmentation was done by other members of the lab but we discuss them here for completeness.

4.3.1 Patch Level Classification Network

Training a network at patch level makes the task tractable since the number of parameters are small and the blur can safely be assumed to be constant within the patch.

The proposed network is constructed as shown in Fig. 4.1. Each ConvBlock has 2 convolutional layers containing filters of size 3×3. The first layer is a convolutional layer with 32 filters with stride of 1. This is followed by a ReLU layer. The output is passed to another convolutional layer with a stride of 2, followed by a batch-normalization layer and another ReLU layer. The two subsequent ConvBlocks carry the same structure but have 64 and 128 filters respectively. The

third ConvBlock is followed by a fully connected layer that brings down the dimensions to 1. Its output is passed through the sigmoid function to obtain the probability of patch being blurred.

Aggregating the blur probabilities for patches distributed over the input image forms a coarse estimate of segmentation map. During testing, this process involves extraction of overlapping patches from the input image and passing them through the network to get their corresponding probabilities. This estimated probability is assigned to all the pixels contained in the patch, while averaging the values in the pixels being overlapped by other patches.

Figure 4.2: Qualitative comparison of various blur detection algorithms on partially motion blurred scenes.

4.3.2 Image Level Regression Network

We propose to enforce global consistency of the labels by training an end-to-end regression network at the image level. The goal of this network is to learn the

mapping between an image and its corresponding blur segmentation-map. The targets to train this network are masks with binary values assigned to each pixel, representing blur/sharp label. In Long *et al.* (2015), it is suggested that for segmentation, image-wise and patch-wise training are equally effective and the former is faster to converge.

To this end, we train a fully convolutional encoder-decoder network with skip connections, inspired by Isola *et al.* (2017), which combines low-level feature maps with higher-level ones, enabling precise localization. The original architecture Isola *et al.* (2017) has demonstrated capability of performing a variety of image-to-image translation tasks, since it can process image of arbitrary dimensions while encoding spatial information thoroughly and efficiently. Such a network can be trained using the dataset provided in Shi *et al.* (2014). It demonstrated that when the training data is limited in size, inclusion of adversarial loss term ensures that the estimated segmentation-map is both semantically meaningful and close to the ground truth.

To train the network for the task of estimating a blur segmentation-map from a single image, we make minor modifications to the original architecture of Isola *et al.* (2017). Since our targets are binary images, we replace the original $L1$ loss with the binary cross entropy loss at each pixel, which requires changing the network's final layer activations to sigmoid. Also, their network includes an increasing number of filters with each convolution layer of stride 2. We, however, fix the number of filters to 64 throughout the network. This results in significant reduction in number of parameters. This choice was motivated by two observations. Firstly, the network can be allowed to lose some information during downsampling because the low level features can still be accessed directly through skip connections in the subsequent deconvolution layers. Secondly, the task at hand

does not require learning of very deep hierarchical features (generally needed for high-level vision tasks like depth estimation or object detection), so a large number of feature maps in higher layers are not critical for good performance. The modified network has fewer parameters and can be trained to obtain a reasonable accuracy on the test set.

4.3.3 Final Detection and Segmentation

The outputs of the two networks have complimentary properties, and hence, a meaningful way of merging the blur probabilities returned by the two networks is pixel-wise multiplication $b(p) = b_1(p) * b_2(p)$. This formulation promotes constructive combination of the two estimates, since the outliers of one result generally do not appear in the other. We find that the patch-level network is able to detect blurred regions with sufficient accuracy. However, sharp but homogeneous regions are also flagged as blurred in this map and the edges are not aligned On the other hand, the image level regression network's map contains refined edges and is able to correctly classify homogeneous regions, but sometimes misclassifies blurred regions as sharp too.

To obtain a binary segmentation-mask, we feed the probabilities $b(p)$ to an MRF formulation. The MRF cost is minimized using an off-the-shelf graphcut algorithm Boykov and Kolmogorov (2004) We define the pairwise cost (between neighbor pixels x and y) as $\beta \times (1 - \alpha^{|b(x)-b(y)|})$, where $|.|$ denotes the absolute value, and β and α are positive scalars. We consider a regular 4-connected grid while calculating cost for a particular pixel. Note that we perform an edge aware optimisation as the smoothness cost is made 0 at edge pixels.

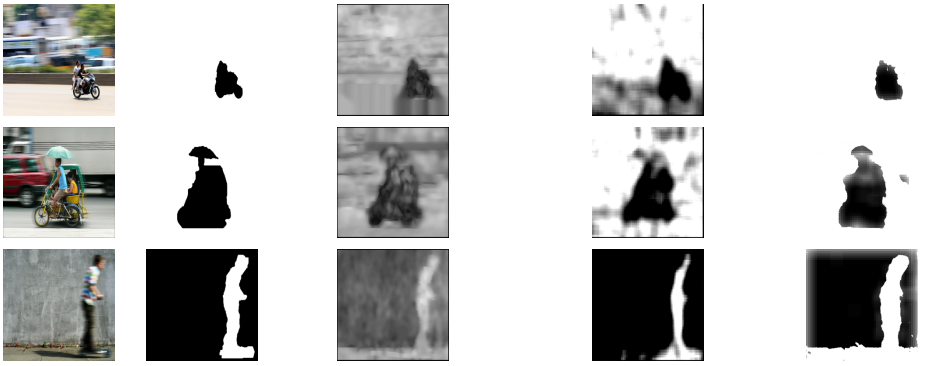

Figure 4.3: Qualitative comparison of various blur detection algorithms on partially motion blurred scenes.

4.4 Dataset and Implementation Details

As mentioned earlier, we utilize the blur segmentation dataset provided in Shi *et al.* (2014) to train the image level regression network. It contains 1000 images with human labelled blur regions, among which 296 are partially motion-blurred and 704 are defocus-blurred. Since we are limited by the number of training samples available in the existing benchmark, we only divide it into training and test sets (90%-10%). For training the regression network, each image in a batch was augmented by randomly applying horizontal and vertical flips together with random rotation and color jittering. We used the same hyper-parameters for training as described in the original paper Isola *et al.* (2017).

The dataset for training classification network is created using sharp images that are blurred using a variety of synthetically generated defocus and motion blur kernels. Our training examples come from 500 all-in-focus images containing diverse scenes and textures obtained from Flickr dataset Huiskes and Lew (2008) and 300 sharp images selected from the ILSVRC dataset Krizhevsky *et al.* (2012) . To simulate motion blur, we generate 1000 realistic motion blur kernels by follow-

ing the approach described in Schmidt *et al.* (2013). Optical blur was simulated using Gaussian blur kernels with zero mean and variance σ^2 . σ is varied from 0.5 to 4 in steps of 0.2 to simulate a wide range of blur encountered in practice. Each image I is convolved with N blur kernels randomly selected from our set to generate N blurred images as: $B_i = I * k_i$ where $i \in [1, N]$. We extract 2×10^5 image patches of size 30×30 from random locations in these images. For natural images. a large fraction of patches are likely to be textureless and hence would not contain any information to classify them as either blurred or sharp. Hence we select only those patches for training whose entropy (a measure of texture-ness) is greater than 4.5 (empirically selected threshold).

We divide this dataset into 80% train and 10% each for both test and validation purpose. The network is implemented using the Torch Library. We trained the model using SGD with a learning rate of 10^{-3} and learning rate decay of 10^{-7}. Early stopping criteria was used to avoid overfitting and the training took 6 hours on a single NVIDIA Titan X GPU.

4.5 Experiments

We verify the reliability and robustness of our algorithm by testing on 100 test images from the Shi *et al.* (2014) dataset.

In Fig.4.1 , we show the outputs of the two networks. We observe that the output of the image level network lacks local accuracy, even though it is semantically meaningful. The output of patch aggregation is locally more consistent but often assigns higher blur probabilities to homogeneous patches present in the sharp regions. Also, as expected, its result generally contains rugged edges which are misaligned with the scene boundaries. The detection-maps obtained after merging

the two outputs are devoid of these errors as can be seen in Figs. 4.2 and 4.3. We utilize this detection map and perform binary segmentation using the Graph-cut optimization. The segmentation results returned by our algorithm are very close to the ground truth.

4.5.1 Comparisons with existing methods

We compare our algorithm to the results of Shi *et al.* (2014), Shi *et al.* (2015) ,Zhuo and Sim (2011) and Park *et al.* (2017) using the implementations available on authors' webpages. Fig. 4.2 shows qualitative comparisons on three real images containing defocus blur of varying degree. Because the blur detection dataset contains only binary masks, quantitative results are obtained using binary blurry region masks from each algorithm. For binary segmentation, we apply a simple thresholding method to their defocussed maps. The best threshold for each approach is found by performing a linear search and picking the value which yields lowest error. All these examples show that our approach can effectively segment the image into defocused and focused regions.

In Fig.4.3 we demonstrate our method's capability to handle motion blur and compare with existing approaches. It can be observed that the detection maps that result from our method are significantly more accurate than prior art.

4.5.2 Applications

The estimated segmentation maps can be used for different applications. We utilize our defocus maps for blur magnification and matting as shown in Fig. 4.4. The results are quite pleasing

(a) (b) (c) (d)

Figure 4.4: Matting and blur magnification: Eagle from (a) is transferred to (b) using matting. Background blur in (c) is magnified in (d)

Blur Magnification can be used to highlight the foreground by amplifying the blurriness of the background. This is done by using our blur-based segmentation-map as a mask for the foreground layer while blurring the remaining image with a Gaussian kernel. The two layers are then merged together.

Image Matting is another application which stands to benefit from robust foreground segmentation. The resulting map M can be used to supply automatic trimaps for challenging images. Morphological erosion and dilation can be performed over the results to obtain a foreground and trimap regions in any defocused image. This technique was used to perform automated matting on real examples. We use closed-form matting algorithm Cho *et al.* (2016), which has been successfully applied on blurred images.

CHAPTER 5

Conclusion and Future Directions

In thesis we primarily addressed two problems in Computer vision namely extracting a video from a motion blurred image and segmentation and detection of blur from images. In this chapter, we summarize the contributions made in the thesis. This is followed by future directions that emerge from these works.

5.1 Summary

In Chapter 2 we discuss the relevant concepts for deblurring, CNN , RNN and STN.

In Chapter 3 we introduced a novel methodology for video generation from a single blurred image. We proposed a spatio-temporal video autoencoder based on an end-to-end differentiable architecture that learns motion representation from videos in a self-supervised manner. The network predicts a sequence of optical flows and employs them to transform a central frame and return a smooth video. Using the trained video decoder, we trained a blurred image encoder to extract a representation from a single blurred image, that mimics the representation returned by the video encoder. This when fed to the decoder returns a plausible sharp video representing the action within the blurred image. The overall setup helps to harness the motion information embedded in a motion blurred frame to generate a temporally ordered sharp video that attempts to reconstruct the scene content explored by the camera trajectory during exposure. The potential of our

work can be extended in a variety of directions including blur segmentation, video deblurring, video interpolation etc.

In Chapter 4 we proposed an automated method to obtain a blur segmentation map from a single image. The blur probability at each pixel was inferred jointly using two sub-networks and merged to obtain an approximate blur-map. An MRF based framework was introduced to obtain a dense segmentation coherent with the boundaries of blurred regions. We validated our proposed framework on images from a publicly available dataset. As future work, we will explore the scope of our framework to perform space-variant deblurring.

5.2 Future Directions

We found in practice that the our encoder-decoder modules were able to generalize well to wide variety of tasks when properly trained. In this section we discuss some directions that we are exploring.

Single Image Deblurring: In our work on extracting a video from a motion blurred image in Chapter 3, we made use of Nah *et al.* (2016) to obtain a sharp image. To make the algorithm self-reliant, we are experimenting with a modified version of our encoder-decoder modules without recurrence for the task of Single Image deblurring.

Video Deblurring: We are also exploring the use of the our recurrent encoder-decoder architecture for the task of video deblurring. Video deblurring can benefit a lot by posing the problem with recurrence between frames. This way, the network can make use of redundancy of pixel information across frames and improve the deblurring performance.

REFERENCES

1. **Boykov, Y.** and **V. Kolmogorov** (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence*, **26**(9), 1124–1137.

2. **Brox, T.** and **J. Malik** (2011). Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE transactions on pattern analysis and machine intelligence*, **33**(3), 500–513.

3. **Bruhn, A.**, **J. Weickert**, and **C. Schnörr** (2005). Lucas/kanade meets horn/schunck: Combining local and global optic flow methods. *International Journal of Computer Vision*, **61**(3), 211–231.

4. **Chakrabarti, A.**, A neural approach to blind motion deblurring. *In European Conference on Computer Vision*. Springer, 2016.

5. **Chakrabarti, A.**, **T. Zickler**, and **W. T. Freeman**, Analyzing spatially-varying blur. *In Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*. IEEE, 2010.

6. **Chaudhuri, S.** and **A. N. Rajagopalan**, *Depth from defocus: a real aperture imaging approach*. Springer Science & Business Media, 2012.

7. **Cho, D.**, **Y.-W. Tai**, and **I. Kweon**, Natural image matting using deep convolutional neural networks. *In European Conference on Computer Vision*. Springer, 2016.

8. **Deng, X.**, **Y. Shen**, **M. Song**, **D. Tao**, **J. Bu**, and **C. Chen** (2012). Video-based non-uniform object motion blur estimation and deblurring. *Neurocomputing*, **86**, 170–178.

9. **Dosovitskiy, A.**, **P. Fischer**, **E. Ilg**, **P. Hausser**, **C. Hazirbas**, **V. Golkov**, **P. van der Smagt**, **D. Cremers**, and **T. Brox**, Flownet: Learning optical flow with convolutional networks. *In Proceedings of the IEEE International Conference on Computer Vision*. 2015.

10. **Favaro, P.**, **A. Mennucci**, and **S. Soatto** (2003). Observing shape from defocused images. *International Journal of Computer Vision*, **52**(1), 25–43.

11. **Favaro, P.** and **S. Soatto**, A variational approach to scene reconstruction and image segmentation from motion-blur cues. *In Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 1. IEEE, 2004.

12. **Flynn, J.**, **I. Neulander**, **J. Philbin**, and **N. Snavely**, Deepstereo: Learning to predict new views from the world's imagery. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

13. **Gong, D.**, **J. Yang**, **L. Liu**, **Y. Zhang**, **I. Reid**, **C. Shen**, **A. Hengel**, and **Q. Shi**, From motion blur to motion flow: a deep learning solution for removing heterogeneous motion blur. *In The IEEE conference on computer vision and pattern recognition (CVPR)*. 2017.

14. **Graves, A.**, **A.-r. Mohamed**, and **G. Hinton**, Speech recognition with deep recurrent neural networks. *In Acoustics, speech and signal processing (icassp), 2013 ieee international conference on*. IEEE, 2013.

15. **Horn, B. K.** and **B. G. Schunck** (1981). Determining optical flow. *Artificial intelligence*, **17**(1-3), 185–203.

16. **Huiskes, M. J.** and **M. S. Lew**, The mir flickr retrieval evaluation. *In Proceedings of the 1st ACM international conference on Multimedia information retrieval*. ACM, 2008.

17. **Hyun Kim, T.**, **B. Ahn**, and **K. Mu Lee**, Dynamic scene deblurring. *In Proceedings of the IEEE International Conference on Computer Vision*. 2013.

18. **Isola, P.**, **J.-Y. Zhu**, **T. Zhou**, and **A. A. Efros** (2017). Image-to-image translation with conditional adversarial networks. *arXiv preprint*.

19. **Jaderberg, M.**, **K. Simonyan**, **A. Zisserman**, *et al.*, Spatial transformer networks. *In Advances in Neural Information Processing Systems*. 2015.

20. **Kalchbrenner, N.**, **A. v. d. Oord**, **K. Simonyan**, **I. Danihelka**, **O. Vinyals**, **A. Graves**, and **K. Kavukcuoglu** (2016). Video pixel networks. *arXiv preprint arXiv:1610.00527*.

21. **Köhler, R.**, **M. Hirsch**, **B. Mohler**, **B. Schölkopf**, and **S. Harmeling**, Recording and playback of camera shake: Benchmarking blind deconvolution with a real-world database. *In European Conference on Computer Vision*. Springer, 2012.

22. **Krishnan, D.**, **T. Tay**, and **R. Fergus**, Blind deconvolution using a normalized sparsity measure. *In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*. IEEE, 2011.

23. **Krizhevsky, A.**, **I. Sutskever**, and **G. E. Hinton**, Imagenet classification with deep convolutional neural networks. *In Advances in neural information processing systems*. 2012.

24. **Lai, W.-S.**, **J.-B. Huang**, **Z. Hu**, **N. Ahuja**, and **M.-H. Yang**, A comparative study for single image blind deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

25. **Liu, R.**, **Z. Li**, and **J. Jia**, Image partial blur detection and classification. *In Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*. IEEE, 2008.

26. **Liu, Z.**, **R. Yeh**, **X. Tang**, **Y. Liu**, and **A. Agarwala** (2017). Video frame synthesis using deep voxel flow. *arXiv preprint arXiv:1702.02463*.

27. **Long, J.**, **E. Shelhamer**, and **T. Darrell**, Fully convolutional networks for semantic segmentation. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

28. **Mayer, N.**, **E. Ilg**, **P. Hausser**, **P. Fischer**, **D. Cremers**, **A. Dosovitskiy**, and **T. Brox**, A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

29. **Nah, S.**, **T. H. Kim**, and **K. M. Lee** (2016). Deep multi-scale convolutional neural network for dynamic scene deblurring. *arXiv preprint arXiv:1612.02177*.

30. **Nimisha, T.**, **A. K. Singh**, and **A. Rajagopalan**, Blur-invariant deep learning for blind-deblurring. *In Proceedings of the IEEE E International Conference on Computer Vision (ICCV)*. 2017.

31. **Pan, J.**, **Z. Hu**, **Z. Su**, **H.-Y. Lee**, and **M.-H. Yang**, Soft-segmentation guided object motion deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016*a*.

32. **Pan, J.**, **Z. Hu**, **Z. Su**, and **M.-H. Yang**, Deblurring text images via l0-regularized intensity and gradient prior. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014.

33. **Pan, J.**, **Z. Lin**, **Z. Su**, and **M.-H. Yang**, Robust kernel estimation with outliers handling for image deblurring. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016*b*.

34. **Pan, J.**, **D. Sun**, **H. Pfister**, and **M.-H. Yang**, Blind image deblurring using dark channel prior. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016*c*.

35. **Park, J.**, **Y.-W. Tai**, **D. Cho**, and **I. S. Kweon**, A unified approach of multi-scale deep and hand-crafted features for defocus estimation. *In Proc. of Computer Vision and Pattern Recognition (CVPR)*, volume 1. 2017.

36. **Purohit, K.**, **A. Shah**, and **A. Rajagopalan** (2018). Bringing alive blurred moments! *arXiv preprint arXiv:1804.02913*.

37. **Ronneberger, O.**, **P. Fischer**, and **T. Brox**, U-net: Convolutional networks for biomedical image segmentation. *In International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015.

38. **Schmidt, U.**, **C. Rother**, **S. Nowozin**, **J. Jancsary**, and **S. Roth**, Discriminative non-blind deblurring. *In Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*. IEEE, 2013.

39. **Schuler, C. J.**, **H. Christopher Burger**, **S. Harmeling**, and **B. Scholkopf**, A machine learning approach for non-blind image deconvolution. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013.

40. **Schuler, C. J.**, **M. Hirsch**, **S. Harmeling**, and **B. Schölkopf** (2016). Learning to deblur. *IEEE transactions on pattern analysis and machine intelligence*, **38**(7), 1439–1451.

41. **Shi, J.**, **L. Xu**, and **J. Jia**, Discriminative blur detection features. *In Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*. IEEE, 2014.

42. **Shi, J.**, **L. Xu**, and **J. Jia**, Just noticeable defocus blur detection and estimation. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

43. **Srivastava, N.**, **E. Mansimov**, and **R. Salakhudinov**, Unsupervised learning of video representations using lstms. *In International Conference on Machine Learning*. 2015.

44. **Su, B.**, **S. Lu**, and **C. L. Tan**, Blurred image region detection and classification. *In Proceedings of the 19th ACM international conference on Multimedia*. ACM, 2011.

45. **Su, S.**, **M. Delbracio**, **J. Wang**, **G. Sapiro**, **W. Heidrich**, and **O. Wang** (2016). Deep video deblurring. *arXiv preprint arXiv:1611.08387*.

46. **Sun, D.**, **S. Roth**, and **M. J. Black** (2014). A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision*, **106**(2), 115–137.

47. **Sun, J.**, **W. Cao**, **Z. Xu**, and **J. Ponce**, Learning a convolutional neural network for non-uniform motion blur removal. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015.

48. **Tai, Y.-W.** and **M. S. Brown**, Single image defocus map estimation using local contrast prior. *In Image Processing (ICIP), 2009 16th IEEE International Conference on*. IEEE, 2009.

49. **Vasiljevic, I.**, **A. Chakrabarti**, and **G. Shakhnarovich** (2016). Examining the impact of blur on recognition by convolutional networks. *arXiv preprint arXiv:1611.05760*.

50. **Vasu, S.** and **A. Rajagopalan**, From local to global: Edge profiles to camera motion in blurred images. *In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.

51. **Vinyals, O.**, **A. Toshev**, **S. Bengio**, and **D. Erhan**, Show and tell: A neural image caption generator. *In Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015.

52. **Vondrick, C.**, **H. Pirsiavash**, and **A. Torralba**, Generating videos with scene dynamics. *In Advances In Neural Information Processing Systems*. 2016.

53. **Wang, T.-L.**, **K.-Y. Lee**, and **Y.-C. F. Wang**, Partial image blur detection and segmentation from a single snapshot. *In Acoustics, Speech and Signal Processing (ICASSP), 2017 IEEE International Conference on*. IEEE, 2017.

54. **Xingjian, S.**, **Z. Chen**, **H. Wang**, **D.-Y. Yeung**, **W.-K. Wong**, and **W.-c. Woo**, Convolutional lstm network: A machine learning approach for precipitation nowcasting. *In Advances in neural information processing systems*. 2015.

55. **Xu, L.** and **J. Jia**, Two-phase kernel estimation for robust motion deblurring. *In European Conference on Computer Vision*. Springer, 2010.

56. **Zhou, T.**, **S. Tulsiani**, **W. Sun**, **J. Malik**, and **A. A. Efros**, View synthesis by appearance flow. *In European Conference on Computer Vision*. Springer, 2016.

57. **Zhuo, S.** and **T. Sim** (2011). Defocus map estimation from a single image. *Pattern Recognition*, **44**(9), 1852–1858.