Radboud University

MASTER'S THESIS IN
ARTIFICIAL INTELLIGENCE

Towards improved video super resolution for real-world applications

Author:
Riccardo SAMPERNA
S4792475

Artificial Cognitive Systems group
Department of Artificial Intelligence
Faculty of Social Sciences

Supervisors:
Dr. Robert NIEUWENHUIZEN[a]
Prof. Marcel VAN GERVEN[b]

[a] *Intelligent Imaging Department, TNO, The Hague*
[b] *Department of Artificial Intelligence, Radboud University, Nijmegen*

December 27, 2018

Abstract

Video or, Multi Frame, Super Resolution (**VSR** or, MFSR) techniques aim to generate high-resolution frame reconstructions of corresponding low-resolution ones. These techniques differ from the Single Image or, Frame, Super Resolution (**SISR** or, SFSR) ones in that they can additionally exploit the temporal nature of the input data. The advantage of having additional temporal information does not translate directly into an easier problem to solve. The challenge lies in the optimal extraction of this additional information. Current state-of-the-art methods are still using some variants of optical flow estimation plus warping for extraction and integration of temporal information but it is well known that during the warping process a lot of the high frequency information get lost. We investigated alternative architectures to alleviate or suppress this problem but they only perform on par or slightly worst than current SOTA networks.

Contents

Chapter 1

Introduction

Nowadays numerous jobs rely on digital imaging systems to perform complex tasks. Think about doctors performing millimetric precision surgery based on computer screen videos or satellite images used for military based interventions. These are just some examples of applications that would potentially benefit from an increase in image quality and details. Generally speaking all applications where a human observer is involved in the interaction with a digital imaging system could potentially benefit from having to deal with crisper images. Furthermore, recent progresses in artificial intelligence could extend those benefits to all computer vision applications, in those cases where a software program instead of a human observer perform a task based on images or videos.

The human perceived quality of a digital image or, in other terms, the amount of details in a digital image is directly related to its resolution. The same holds true for videos, which are just a sequence of images (usually called frames) but, in this case, we talk about video resolution. The generic term *resolution* can refer to different types of resolution: pixel resolution, spatial resolution, spectral resolution, temporal resolution, radiometric resolution, therefore it is important to specify which resolution we are considering in this text. For the purpose of this thesis, we are mainly interested in *spatial resolution* and in the remaining of the text we are going to use the term resolution and spatial resolution interchangeably.

So, what is *spatial resolution*? As already said, an intuitive way to think about spatial resolution is to look at the amount of details in an image or a video but a more formal definition is the following given by Yang and Huang, 2010:

> "**Spatial resolution:** A digital image is made up of small picture elements called *pixels*. *Spatial resolution* refers to the *pixel density* in an image and measures in *pixels per unit area*. Fig. 1.1 shows a

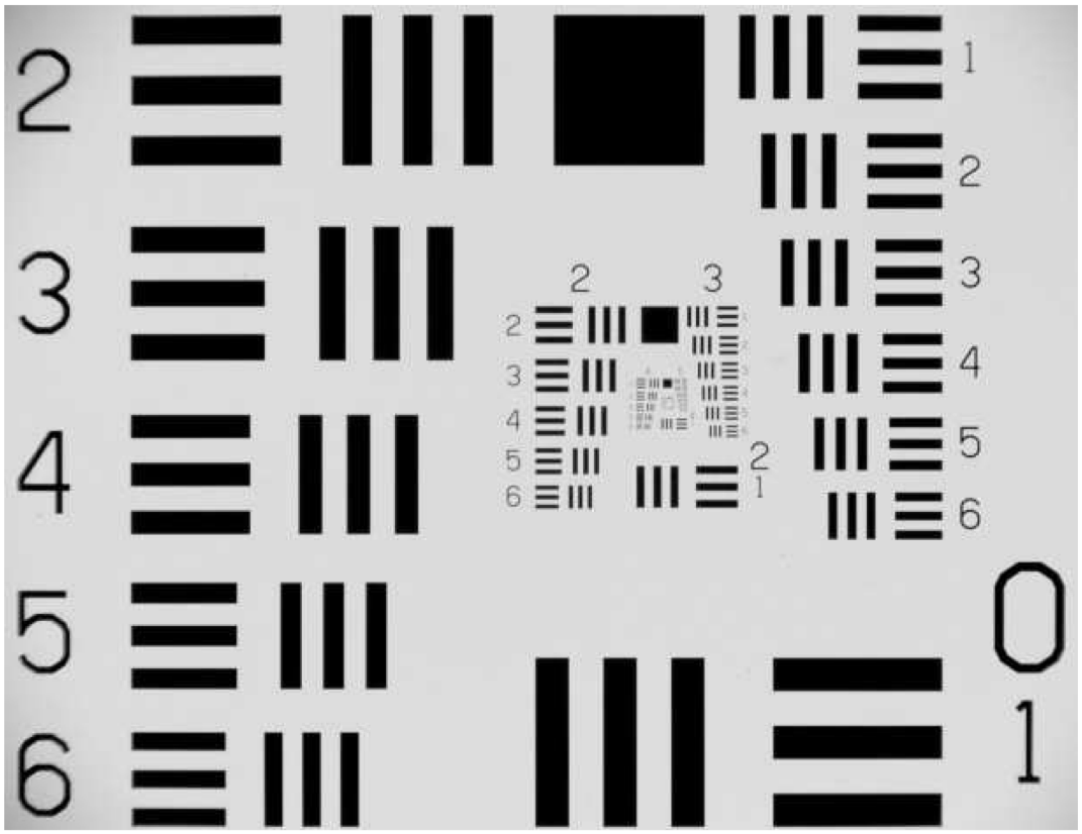


FIGURE 1.1: The 1951 USAF resolution test target, a classic test target used to determine spatial resolution of imaging sensors and imaging systems. Yang and Huang, 2010.

> classic test target to determine the spatial resolution of an imaging system".

From this definition, it becomes immediately clear that the most straightforward way to increase the spatial resolution and therefore the details in an image is to increase the pixel density. Increasing the pixel density translates, in an image sensor, to increase the photosite density. An image sensor is generally made up of a two-dimensional grid of photosites and each photosite corresponds to a pixel in the generated image. Charge-coupled device (CCD) and complementary metal oxide semiconductor (CMOS) are the most common image sensors in today's digital imaging systems. Both capture light by means of electric charge and transform it to an electrical signal in the two-dimensional grid. Semiconductor manufacturer have continuously been pushing the limits to keep the same image sensor size with an increased number of photosites therefore increasing its pixel density. Unfortunately only increasing the density of photosites by reducing their size leads also to

a decrease in incident light on each photosite which in turn causes the so-called shot noise. Conversely an image sensor with an inadequate number of photosites will generate low resolution images with a pixelation effect, due to the aliasing from low spatial sampling frequency.

The limitation in pixel density increase is not the only factor affecting an increase in resolution, a more fundamental problem is constituted by the optic used by the digital imaging systems. The optic physically limits the amount of details (high frequency bands) that is possible to capture in an image sensor due to lens blurs (associated with the sensor point spread function (PSF)), lens aberration effects and blurring due to motion (Yang and Huang, 2010). A better quality lens, which often translates into a larger lens, would in principle allow to capture more details and consequently increase the resolution of an image. Unfortunately also this solution has several drawbacks. Larger lenses have generally more imperfections, they may not fit in the digital imaging systems we want to build (e.g. smartphones) and ultimately they may reach prohibitive prices for most real-world applications.

Additionally both image sensors and lenses are physical components of a digital imaging system and in applications like remote sensing the imaging system components are usually not accessible for a direct replacement of parts.

For all these reason is really important to come up with software solutions that can increase the resolution of captured images or videos without relying on the hardware of the physical device. This is where super resolution come into play. Super resolution (SR) is a general term referring to all those techniques aiming at enhancing the spatial resolution of one or a sequence of images. SR techniques generally receive as input one or more low-resolution (LR) images and attempt to create their high-resolution (HR) counterparts.

As we will see, in general, it is hard to achieve really detailed results with SR techniques, especially for large desired increase in spatial resolution (high scale factor SR). In this thesis we will review the state-of-art of video super resolution (VSR) using deep learning techniques and we will investigate possible ways to improve the current state-of-art, tackling some of the problems known from traditional signal processing theory. Finally we will look into ways to make VSR more robust in real-world applications given the collaboration with TNO for this thesis.

TNO is the Netherlands Organization for Applied Scientific Research and

this thesis was developed in closed collaboration with the Intelligent Imaging department. The main interest of TNO and the department is on long range observation for defence/security applications with a particular focus on achieving results that benefit tasks such as text recognition, automated human action recognition and vehicle identification.

1.1 Background

This section introduces some of the concepts that are central in this thesis. We are going to characterize the type of super resolution addressed in the rest of the text and define some terms that will reoccur often in the following chapters.

1.1.1 Image observation model

When a digital imaging system captures a scene of the real world there are different factors influencing the final result that is obtained as image or sequence of images. The image observation model is generally used in SR to simulate this physical process. For the purpose of this thesis the image observation model is used to generate the LR-HR pairs which allow to train our models (example-based SR, more details to follow) therefore representing an important element in our approach.

As pointed out by Nasrollahi and Moeslund, 2014, one of the most straightforward way to model the physical process of capturing a scene of the real world is to use a linear model of the type used by Schultz and Stevenson, 1994:

$$g(m,n) = \frac{1}{q^2} \sum_{x=qm}^{(q+1)m-1} \sum_{x=qn}^{(q+1)n-1} f(x,y) \tag{1.1}$$

where g represents the "LR obtained image", f is the "HR observed real-world scene" and q is a decimation or sub-sampling factor. The LR image, using this model, is obtained averaging q^2 neighbouring pixels from the HR observed scene which implies that, if the size of the LR image is $M_1 \times M_2$, then the HR image is of size $N_1 \times N_2$ where $N_1 = qM_1$ and $N_2 = qM_2$. This linear model is too simplistic and fails to consider many other factors that contribute to the physical process of capturing a scene of the real world. In order to include some of these factors, Nasrollahi and Moeslund, 2014 extends Eq.1.1

proposing the model from Irani and Peleg, 1990:

$$g(m,n) = d(h(w(f(x,y)))) + \eta(m,n) \tag{1.2}$$

where w is a warping function, h is a blurring function, d is a down-sampling operator, and η is an additive noise. *Warping* is a geometric transformation that consists in moving around the pixels of an image based on a specified new mapping, it can be as simple as a basic rotation. In the physical process described by the model it can represent, for example, a non-centered image or a camera movement. *Blurring* refers instead to a type of filter effect which decreases the level of detail in an image. In the model the blurring function allows to mimic blurring effect due to the digital imaging system (lens and/or the sensor) or to atmospheric conditions. The operation of *down-sampling* decreases the pixel density of the image with a consequent loss of information and spatial resolution. We saw already an example of down-sampling operation in Eq.1.1 and it is generally used to define the way by which the HR scene is sub-sampled. Figure 1.2 shows an example of the application of the model described by Eq.1.2.

FIGURE 1.2: The imaging model employed in most SR algorithms. Nasrollahi and Moeslund, 2014.

In the process from HR to LR all the high-frequency details, making up the important information of an image (e.g. edges, textures), are generally lost and need to be reconstructed using SR. The inversion of this degradation process, for which SR is sometimes considered an inverse problem, does not have a unique solution making the task of SR extremely hard.

1.1.2 Classification of super resolution techniques

Super resolution (SR) techniques are generally classified based on three main factors: the domain, the number of images taken as input and the type of

reconstruction method. It follows a brief explanation of the three classification factors to make clear the type of SR techniques we are interested in this thesis. This overview does not want to be an exhaustive description of all the possible SR techniques but it just wants to give the context and enough background to understand the rest of the material. For a more exhaustive treatment of the topic please refer to Park, Park, and Kang, 2003; Farsiu et al., 2004; Nasrollahi and Moeslund, 2014.

Domain

The domain classification makes a distinction between SR techniques in the *frequency* and in the *spatial* domain. The paper by Huang and Tsay, 1984 represents the very first example of SR. The authors approached the problem in the frequency domain by exploiting the properties of the Fourier transform. Many other early papers in SR followed the same strategy tackling the problem in the frequency domain either using the Fourier or the Wavelet transform properties (for further details please refer to Nasrollahi and Moeslund, 2014 or Bevilacqua, 2014 that give a nice overview of SR techniques). However, researchers quickly figured out that treating the SR problem in the frequency domain limited the applicability of the SR techniques in real-world scenarios. Real-world turned out to be much more complicated compared to the models that could be handled by SR techniques in the frequency domain. This is the reason why most of the current SR papers address the problem in the spatial domain which allows the modelling of all kinds of image degradation and help better constraint the SR problem (Yang and Huang, 2010). Also in this thesis we are only going to consider SR techniques in the *spatial domain*.

Number of images taken as input

In the literature there is a clear distinction between two types of techniques for SR: *single frame super resolution (SFSR)* and *multi frame super resolution (MFSR)*. As both terms already explain, the distinction in this case is based on the number of images taken as input by the SR technique. SFSR techniques use only a single image to increase its resolution which can be seen as a form of interpolation because there is no additional information provided. The quality of the SFSR output is limited due to the ill-posed nature of the problem, and the lost frequency components cannot be recovered. MFSR deals

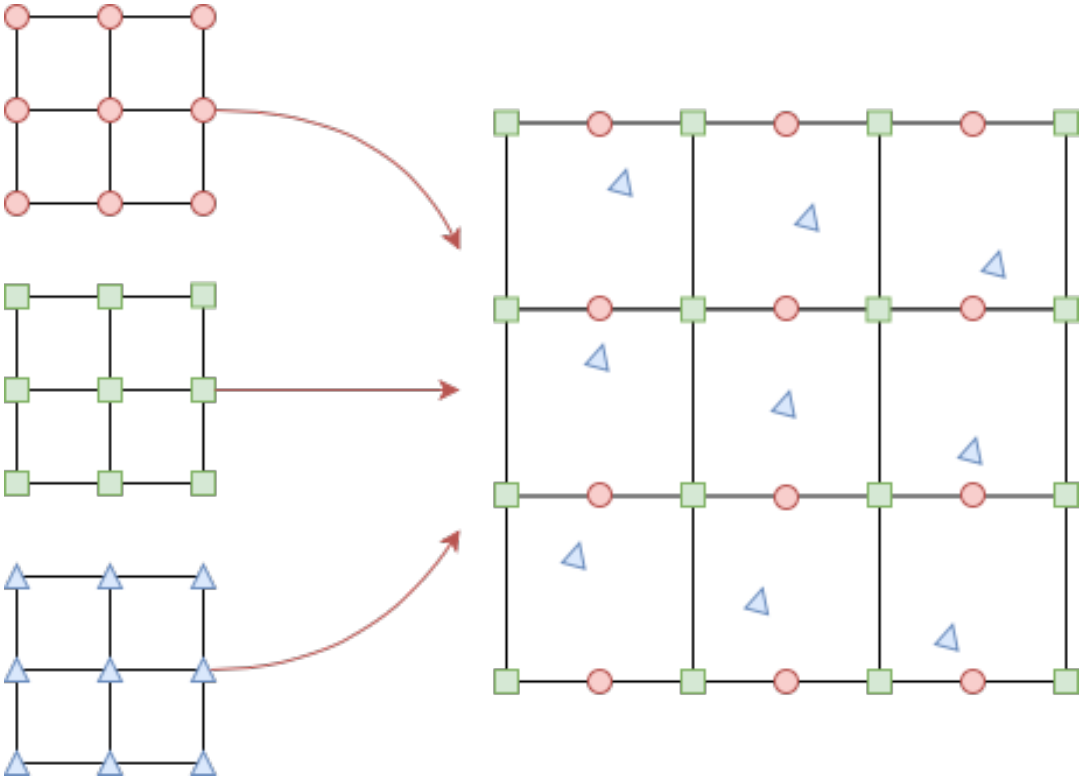


FIGURE 1.3: A schematic example of MFSR. The information contained in three LR images are merged to reconstruct an HR image.

instead with the problem of increasing the resolution of an image using multiple instances of that image, a schematic example is shown in Fig.1.3. In the case of MFSR we have additional information contained in these multiple instances which are typically introduced by subpixel shifts between them, aliasing or different noise realization, making the ill-posed problem more constrained. The challenge of MFSR lies in the registration and alignment of these multiple instances which allows the use of the additional information.

The frames of a video clip represent an example of when a subpixel shift may occur (e.g. object movements, camera movements) and in this case we talk about *video super resolution (VSR)*. VSR is a particular case of MFSR which uses the neighbouring frames of a video clip as input to the SR approach and it is going to be the main subject of this thesis.

Type of reconstruction method

In this section, we present the different approaches available in the spatial domain for the particular case of MFSR (they broadly correspond also to the distinction that is made in the SFSR case).

MFSR is usually performed in two different ways: reconstruction-based or example-based approach. The reconstruction-based approach is a model based approach, it assumes little knowledge about the content of the image and focuses instead on the process of how the image was formed. Reconstruction-based approaches use aliasing artifacts present in LR images in order to reconstruct the HR instance, they usually define a prior distribution in order to constrain the ill-posed nature of the problem. The example-based approach, instead, learns a mapping between LR and HR images using a dataset of examples which are generated using the image observation model 1.1.1. There is a first phase referred as *training* where the algorithm learns this mapping and it is successively applied to unseen data in the *test* phase. The current trend in example-based SR uses a type of machine learning techniques called *deep neural networks* which are generally referred as *deep learning (DL)*. They are particularly relevant because they proved to outperform all other types of approaches both in SFSR and MFSR. They are going to be the main topic of this thesis.

1.1.3 Deep Learning fundamentals

Before continuing in our journey it is important to introduce some of theory behind the main actor of this play, namely *Neural Networks (NNs)*, or Artificial Neural Networks.

Neural Networks

Neural Networks (NNs) is a general term to refer to a broad range of computational models vaguely inspired by what is known so far about the structure and functions of their biological counterparts. Their origins date back to Rosenblatt, 1958 at the Cornell Aeronautical Laboratory proposing one of the first examples of this learning algorithms, the Perceptron.

The Perceptron is a type of linear classifier, i.e a classification algorithm that makes its predictions based on a linear predictor function, combining a set of weights with the input vector. Back then it looked a really promising idea together with all the expectations created by the same Rosenblatt but, in a famous book entitled "Perceptrons" from Minsky and Papert, 1969, was shown soon the limitation of this learning algorithm. The single layer perceptron is able, in fact, to learn only linear separable patterns.

Some years later (after the so called "AI winter") it was recognized that a feedforward neural network (a network where the input enters only the network's first layer and propagates only to the next layer without lateral connections within a layer) with two or more layers can approximate any function (universal approximation theorem, Hornik, Stinchcombe, and White, 1989). This type of network are also called Multi Layer Perceptrons (MLPs), they can be seen, in fact, as a concatenation of multiple Perceptrons' and they introduce the concept of hidden layer (not explicitly exposed to the input) and the use of non-linear activation functions. They form the basis of the most modern NN implementations, yet many extensions to the original design and learning algorithm have been proposed. The MLP consists of an input layer, followed by at least one hidden layer and an output layer (Figure 1.4). All layers are fully connected so that each neuron performs a weighted

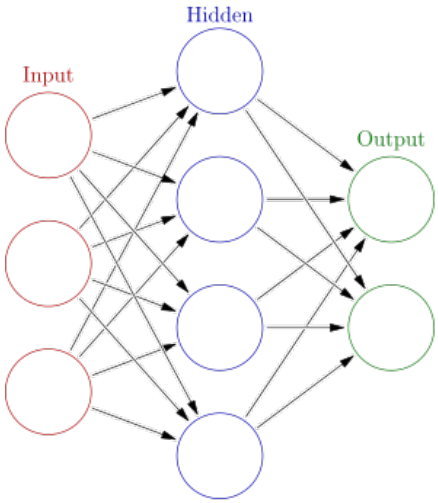


FIGURE 1.4: Artificial Neural Network example architecture.

combination of all the outputs of the previous layer. Subsequently the artificial neurons perform a non-linear mapping of the weighted sum. MLPs are trained using *backpropagation*. In this gradient descent method the activation of each neuron is calculated in a forward pass and consequently the network weights are updated in a backward pass. This latter process is termed *backpropagation*.

Convolutional Neural Networks

Krizhevsky, Sutskever, and Hinton, 2012 marked the beginning of a new era for NNs. The authors presented a Convolutional Neural Network (CNN),

that outperformed all other approaches at a famous computer vision competition called the ImageNet competition. ImageNet is a huge image dataset freely available online and the competition consist in the automatic classification of these images. This was not the first example of CNN, already Lecun et al., 1998 proposed a similar approach to tackle handwritten text recognition but the difference was made by the size of the network trained thanks to the availability of Graphic Processing Units (GPUs). CNNs are a particular type of NNs that are characterized by the presence of convolutional and pooling layers. A convolutional layer computes a convolution operation between the input and learnable filters, Figure 1.5 shows an example of this operation. This type of layer proved to be helpful in tasks concerning the

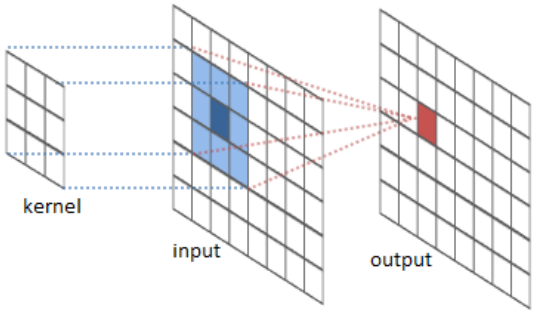


FIGURE 1.5: Convolution layer example.

processing of images because it allows the network to learn features of objects that can be used to generalize to other instances of the same objects. The other layer that characterizes a CNN architecture is the pooling layer. It can be simply described as performing a non-linear down-sampling of the input. Figure 1.6 gives a nice example of what it means in the particular case of the max-pooling, where the maximum value is selected among a group pixel values.

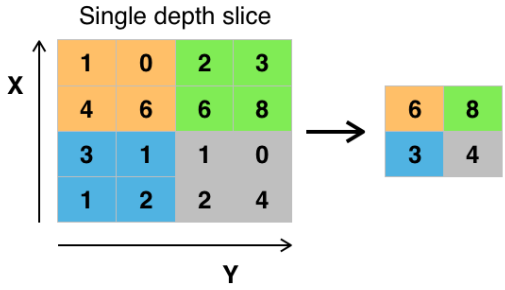


FIGURE 1.6: Max pooling layer example.

Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are a particular type of NN which contain a loop in their structure hence the name "Recurrent". The loop structure allows to work on input data in the form of a sequence. A RNN can be seen as a copy of the same network which is applied on each step to a different element of the sequence, Figure 1.7 shows an example of unfolding the structure of a RNN. The RNN has the property of propagating information from

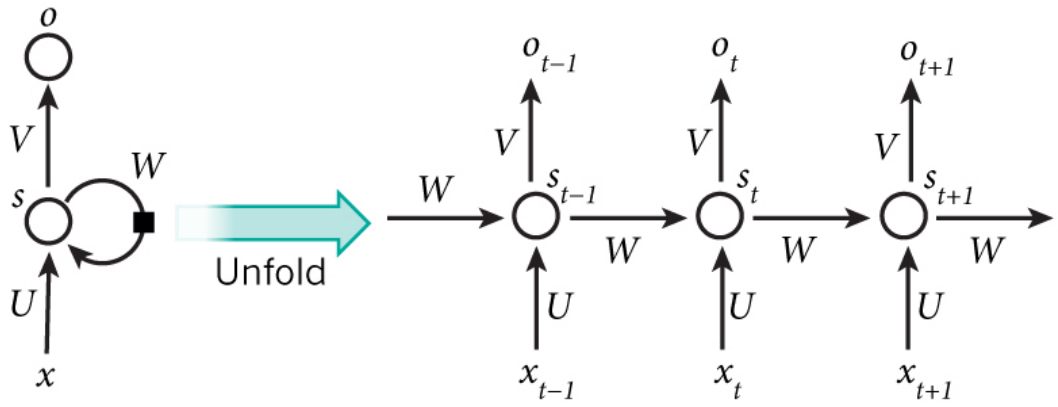


FIGURE 1.7: Example of a recurrent neural network.

the past using an internal state mechanism which makes it a perfect fit to use with sequence data. This mechanism is one of the reason why RNNs are generally preferred in the literature when the dataset comprises elements which are highly related (e.g natural language processing). Compared to a normal NN, the RNN uses another type of training algorithm called *backpropagation through time* (BPTT; Mozer, 1995) which allows to adjust the weights of the network after looping through all the elements of a sequence.

1.2 Related work

This section provides an overview of the state-of-the-art (SOTA) results in MFSR, particularly using NNs techniques.

The pioneering work of Dong et al., 2014, where they presented a shallow three layers convolutional neural network (CNN) for SFSR, marked a shift in paradigm for the whole field. Nowadays academic research in SR is largely dominated by neural networks that have proven to outperform all the previous state-of-the-art methodologies, as in many other fields.

Kappeler et al., 2016 are the first to apply a CNN to MFSR adapting the approach from Dong et al., 2014 and using a separate optical flow estimation algorithm from Drulea and Nedevschi, 2011 to exploit temporal information. The two components, the CNN and the optical flow estimation algorithm, need to be trained separately and the CNN is pre-trained on images before being trained for videos.

Caballero et al., 2016 incorporate a multi-scale spatial transformer as a motion compensation component directly into the trainable architecture using also 3D convolutions and slow fusion.

The same motion compensation is used also by Tao et al., 2017 but instead of 3D convolutions, they propose a recurrent network with an LSTM unit to process multiple frames. In addition, the "Sub-Pixel Motion Compensation (SPMC)" layer is introduced, which performs forward warping and mapping to HR space jointly.

A similar operation is also used by Makansi, Ilg, and Brox, 2017 which build upon the work of Kappeler et al., 2016 constructing an architecture that includes the optical flow estimation from Ilg et al., 2017 and it is end-to-end trainable.

Liu et al., 2017 propose instead a "temporal adaptive neural network" which uses the architecture from Shi et al., 2016 to produce several HR estimates using temporally different LR frames which are subsequently aggregated together.

Very recently, Sajjadi, Vemulapalli, and Brown, 2018 presented a fully recurrent approach that uses the previous HR frame reconstruction to produce temporally consistent frames. As pointed out from Sajjadi, Vemulapalli, and Brown, 2018 already, more than a decade ago also the paper by Farsiu et al., 2004 suggested a recurrent approach for the same motivation.

The paper from Sajjadi, Vemulapalli, and Brown, 2018 is not the first example of deep learning recurrent architecture applied to MFSR. In the work from Huang, Wang, and Wang, 2018 a bidirectional recurrent convolutional networks is proposed but no explicit motion estimation is used.

Comparing the results of the mentioned papers we identified the paper by Sajjadi, Vemulapalli, and Brown, 2018 as currently achieving the state-of-the-art (SOTA) performance in MFSR. A we will mention multiple times along the remaining of the text we are not completely sure about these SOTA results because we found inconsistency in the way results were reported and compared in different papers reviewed.

1.3 Aim of the project

The literature review presented in the previous section guided the motivation to work on this project. Deep learning, which we refer here as the field that regained interest in 2012 after the paper by Krizhevsky, Sutskever, and Hinton, 2012, is still in its infancy. There are not so many papers available on the topic of VSR using deep learning techniques compared to other fields of application. The limited number of research papers may suggest that there are still opportunities for further improvements on SOTA results.

In addition, from our literature review, we noticed that multiple papers claim SOTA results, but an accurate analysis revealed that there are often inconsistency on the way these results are compared and reported. Therefore we think it is important to investigate further these results and, as a starting point for the project, reproduce at least partially some of these approaches. This should give us a solid ground to build for further improvements.

In the introduction we talked about the collaboration with TNO, the Intelligent Imaging Department for this thesis. TNO is interested in exploring the potential of deep learning methods applied to SR to possibly integrate some of these techniques in their pipeline. Using TNO expertise on traditional SR, we identified the warping operation, which is used in most of the VSR deep learning architectures in the literature, as a potential element for improvements. The warping operation is used, in VSR, in combination with the motion estimation to be able to extract additional information from neighbouring frames. It allows to register a group of frames to the same reference, moving pixels according to the calculated motion. Unfortunately during this registration step, performed by the warping operation, is used an interpolation which causes loss of information. We decided to test architecture variations that can possibility mitigate this negative effect.

Ultimately this thesis wants to explore the possibility of applying deep learning VSR techniques on real-world scenarios, testing the robustness of the developed networks with more realistic image observation models.

1.4 Research questions

The research project, as already explained in the previous section, has first and foremost the goal of investigating how to combine TNO expertise with a SOTA deep learning solution for the problem of VSR. The prospective is to eventually substitute part of the pipeline already in place at TNO. The first

step in this process consists in the identification and analysis of SOTA deep learning solutions. In particular we are interested in understanding which structure, feed-forward or recurrent, achieves the best performance and how that is accomplished. The next step aims at solving, at least partially, one problem that is known to affect traditional SR techniques and which is still present in most of the deep learning approaches, warping. In our literature review, we noticed that, the potential negative effect the warping operation can have on the performances of VSR is generally underestimated. We decide to make it here the main focus of our investigation because we think it can help us improve on the current SOTA. Finally we look at more realistic image observation models and how they affect the performances of VSR deep learning solutions to validate the possibility of applying them outside of a controlled environment.

From what we discussed so far, we can formulate some concrete research questions that we will try to answer in the remaining of the text.

1. Which deep learning architectures are the current state-of-the-art in video super resolution?
    - What is the difference, in terms of performances, between a feed-forward and a recurrent architecture?
2. How can we address critical points of traditional video super resolution in the design of the network?
    - Can we improve on the integration of information from neighboring frames?
    - What would happen if we move away from the separation of the problem in two distinct sub-tasks: motion estimation and super resolution?
3. How does a more realistic image observation model affects the performances of these architectures?
    - Does the addition of noise in the image observation model substantially deteriorate the performances?

Chapter 2

Methods

Video super resolution (VSR) using deep learning techniques is the main subject of investigation of this thesis and to answer the research questions posed in the previous chapter we define here the tools we used to conduct our experiments. We give a detailed explanations of all their components, the dataset we used and the training procedure adopted. In this chapter we present the details that are common to most of the experiments but when necessary, in Chapter 3, we are going to point out what differs.

2.1 Architecture building blocks

In this section we introduce some of the main components of the architectures examined in the thesis. We follow a bottom-up approach to facilitate the readability of the architecture diagrams that follow. Each of the elements described in the following sections corresponds to the one used as building blocks for the baseline architectures and all further modifications, if not otherwise stated.

2.1.1 Motion estimation

Motion estimation plays a central role in VSR as we learned in Chapter 1 from the SR problem categorization. The accuracy of the motion estimation determines the success of the integration of information from neighboring frames. Motion estimation is in itself an open problem in computer vision, numerous researchers have been studying the problem in the years trying to improve the accuracy using traditional and, more recently, deep neural networks methods alike. Few years ago, Dosovitskiy et al., 2015 demonstrated the possibility to learn an accurate optical flow estimation using a CNN. In a follow-up work Ilg et al., 2017 further refined the architecture making it as accurate as traditional state-of-the-art methods but much faster. In our

experiments we employed part of the FlowNet2 architecture proposed by Ilg et al., 2017 called FlowNet-SD, the "SD" stands for "Small Displacement". Early in the project we also conducted experiments with the FlowNet2 but we did not find any performance difference compared to the FlowNet-SD. The FlowNet2, being a more complex model, should perform a more accurate motion estimation but, possibly, the lack of large motions in the dataset we used may explain the phenomenon. Ultimately we opted for the FlowNet-SD because the model is approximately eight times smaller than the FlowNet2 impacting less on the GPU memory consumption. This motion estimation architecture once integrated in the VSR model can be trained end-to-end on the particular dataset in use. We did not perfom an end-to-end training in our experiments but we employed instead a pre-trained model to save time on the overall training and because the pre-trained model should be able to generalize on our dataset. We used the Pytorch implementation provided by Reda et al., 2017 and we downloaded the pre-trained weights from the same project website*.

2.1.2 Residual block

Residual blocks made their first appearance in the paper by He et al., 2015 and since then they have been used extensively in the deep learning community in order to stabilize the learning of very deep networks. This type of networks generally suffers from the so called vanishing gradient problem (Bengio, Simard, and Frasconi, 2012) and the residual block constitutes an optimal way to solve it. It has been adopted also in SR approaches for the same motivation and Lim et al., 2017 recently introduced a variation which showed to improve performances in SFSR compared to the original design of He et al., 2015.

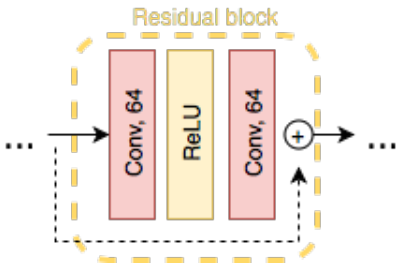


FIGURE 2.1: Residual block structure as presented by Lim et al., 2017

*https://github.com/NVIDIA/flownet2-pytorch

Figure 2.1 shows the structure of the residual block from Lim et al., 2017 which we adopted in our architectures. It consists of two convolutional layers (number of filters: 64, kernel size: 3x3, stride: 1, padding: 1), a ReLU non-linearity function and the addition operation of the residual. This version of the residual block removes the batch normalization layer which was shown to introduce instabilities and color shifts.

2.1.3 Upsampling block

Deep learning approaches to MFSR have used different up-sampling strategies to produce the HR output. Kappeler et al., 2016, for example, up-sampled the input frames directly using a bicubic interpolation (Matlab imresize). This way of up-sampling the image is both inefficient, because the rest of the operations are performed in HR, and non-trainable, because it is not the network itself that is learning how to best perform the up-sampling. Another intuitive approach that has been used to up-sample the final output is the deconvolution operation also called convolution with fractional stride. The deconvolution operation is usually performed just before the output layer therefore being both efficient, in this case all the previous operations are performed in LR, and trainable, the network can learn an optimal up-sampling strategy, but it has also been shown to produce check-board artifacts in the final reconstruction (Odena, Dumoulin, and Olah, 2016). We decided to use the layer proposed by Shi et al., 2016 called subpixel convolutional layer (*"PixelShuffle"*) which basically performs a depth to space transformation. This layer has shown to be computationally efficient and it does not create checkerboard artifacts. A scheme of our upsampling block can be seen in Figure 2.2.

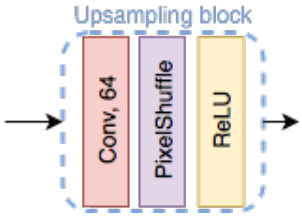


FIGURE 2.2: Upsampling block using the "PixelShuffle" layer from Shi et al., 2016

2.1.4 SRNet+

The structure of the SRNet+ is derived directly from the one used in Sajjadi, Vemulapalli, and Brown, 2018. It is an encoder-decoder like structure used frequently in SFSR and MFSR, one of the first example can be traced back to Johnson, Alahi, and Fei-Fei, 2016. Compared to Sajjadi, Vemulapalli, and Brown, 2018, we modified the part of the architecture that is responsible for the up-sampling of the final output. As we explained in section 2.1.3 the deconvolution layer tends to create check-board artifacts therefore we substituted it with a sub-pixel convolutional layer. We used three residual blocks but we acknowledge that using more residual blocks (i.e five) with an increased number of filters (i.e 128) in the convolutional layers would result in increased performance. We made this design choice to reduce the training time per experiment. Figure 2.3 shows the overall architecture of our SRNet+.

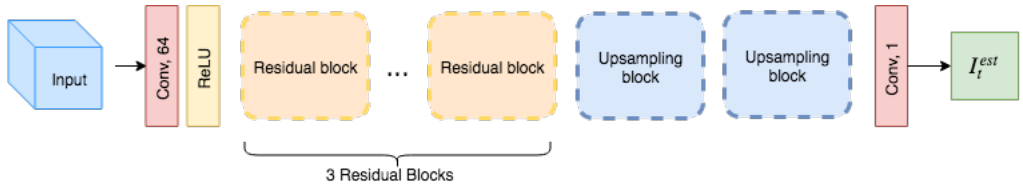


FIGURE 2.3: Architecture diagram of SRNet+ where we replaced our Upsampling block compared to the structure from-Sajjadi, Vemulapalli, and Brown, 2018.

2.2 Baseline architectures

The two baseline architectures presented in the following sections are the starting point to investigate the SOTA of VSR. The design of the two networks is inspired by the best performing architecture in the literature.

2.2.1 Feed-forward

The first architecture taken as a baseline for our experiments has a feed-forward structure, this structure is the most common in the MFSR literature. The model takes as input three LR neighboring frames f_{t-1}^{LR}, f_t^{LR}, f_{t+1}^{LR} and outputs the HR estimate of the frame at time t also referred as central frame or reference frame. First the optical flow between $\{I_{t-1}^{LR}, I_t^{LR}\}$ and between $\{I_t^{LR}, I_{t+1}^{LR}\}$ is calculated using the pre-trained FlowNetSD. The optical flow calculation allows us to register I_{t-1}^{LR} and I_{t+1}^{LR} to the central frame f_t^{LR}. The

three frames are concatenated along the channel dimension and are fed into the the rest of the network which is responsible for the super resolution. This model has a clear distinction between where the integration of the temporal information happens, first part, and where the frame gets ultimately super-resolved, SRNet+. Figure 2.4 gives an overview of the model using some of the building blocks described in Section 2.1.

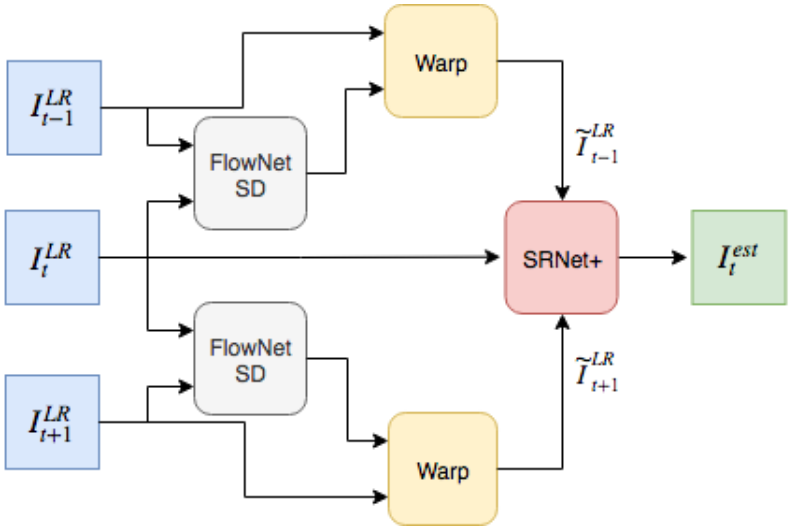


FIGURE 2.4: Feed-forward baseline architecture.

2.2.2 Recurrent

The recurrent baseline is based on the framework proposed by Sajjadi, Vemulapalli, and Brown, 2018. For the motion estimation, indicated as FNet in the original paper, we integrated the pre-trained FlowNetSD and for the super resolution part, indicated as SRNet in the original paper, we used the SRNet+ explained in Section 2.1.4. Compared to the feed-forward baseline, the recurrent network takes as input only two LR frames f_{t-1}^{LR}, f_t^{LR} but an additional input comes from the HR output at time $t-1$ indicated as I_{t-1}^{est}. I_{t-1}^{est} is initialized as a black image in the first training or testing step. Figure 2.5 shows our structure for the recurrent network. The up-scaling step uses a bilinear interpolation to up-scale the optical flow calculated between $\{I_{t-1}^{LR}, I_t^{LR}\}$ and should not be confused with the up-sampling block presented previously. The up-scaled optical flow is used to warp the I_{t-1}^{est} to the reference frame f_t^{LR}. The space-to-depth operation is used to map the HR warped $\tilde{I}_{t-1}^{est}$ to LR and once concatenated in the channel dimension with f_t^{LR}, they get fed to the super resolution part. Also the recurrent architecture separates the motion estimation task from the super resolution.

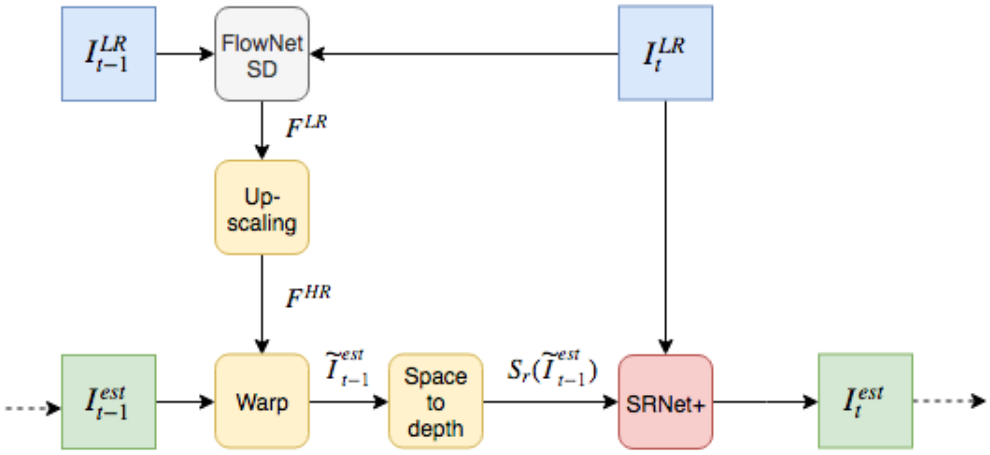


FIGURE 2.5: Recurrent baseline architecture.

2.3 Architecture variations

In this section we use the baseline feed-forward architecture as a starting point for our experimentation. All the structures of the following architectures are our design choices to improve the SOTA.

2.3.1 Warp frame features

Convolutional layers with 64 filter of size 3x3, stride 1 and a ReLU non-linearity are added to the structure of the feed-forward baseline as shown in Fig. 2.6 directly after each input frame. The idea of mapping an image to a higher-dimensional space, feature space, has already been used successfully in the deep learning literature to address other types of problems and it is a technique that goes by the name of *Embedding* (Mikolov et al., 2013). In this case the network should learn a representation of the image that once warped retains more high frequency information. The central frame gets also processed by a convolutional layer, before the concatenation, to maintain the consistency of the information that gets passed to the following layers. The convolutional layer after the frame at time t is an optional operation that should facilitate further the learning of the neural network.

2.3.2 Late features warping

The idea of warping features instead of the frame itself is further developed with the modification shown in Figure 2.7.

In this case part of the structure of SRNet+ is moved before the warping operation. This way the complexity of the model remains roughly the same

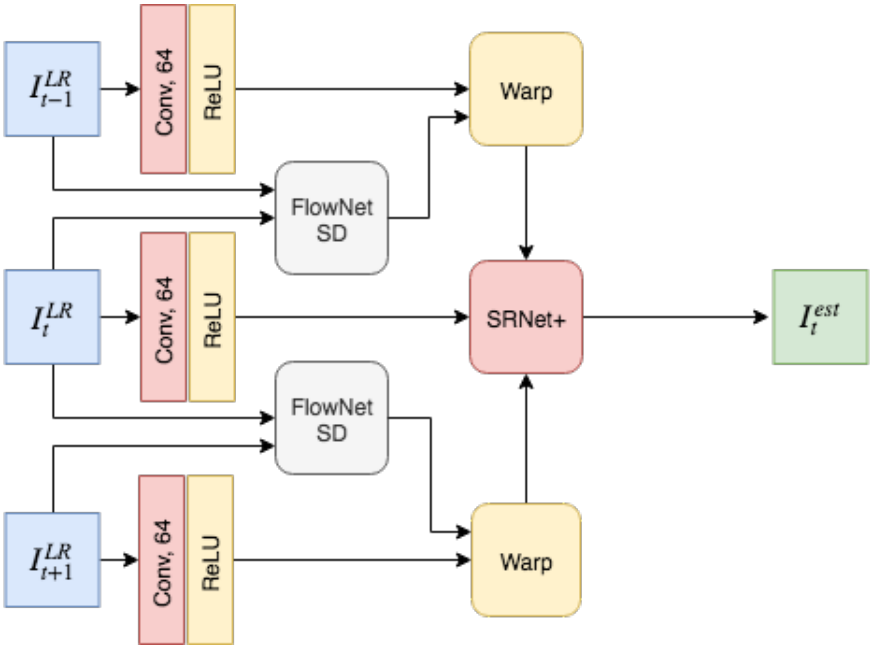


FIGURE 2.6: Schematic architecture of our frame features warping variation.

allowing a fair comparison of performance with the baseline. In pre-SRNetA and pre-SRNetB we look again for a different, more complex, representation of our frames that can possibly reduce the loss of information during the warping operation. The Pre-SRNetB part of the architecture uses shared weights for the two input frames f_{t-1}^{LR} and f_{t+1}^{LR} and it includes two residual blocks of the original SRNet+ leaving only one to the Post-SRNet part. A single convolutional layer, as in the case of the architecture presented in section 2.3.1, may not be sufficient for the network to learn a representation that is useful for the problem we are trying to solve.

2.3.3 Warping in high-resolution

The warping operation performed in high-resolution should preserve more high frequencies compared to when it is performed in low-resolution.

The architecture presented in Figure 2.8 allows first to up-sample all the input frames in HR using a shared weights SRNet+ and only at the end the integration of the information through motion estimation and warping. This approach reverse the order of the two sub-tasks, first the super resolution is performed on every frame and only after the motion estimation is calculated. As we already pointed out previously, this strategy of up-sampling first was used by most of the first deep learning approaches to MFSR (Kappeler et al., 2016) but the difference in this case is that we let the network find the best

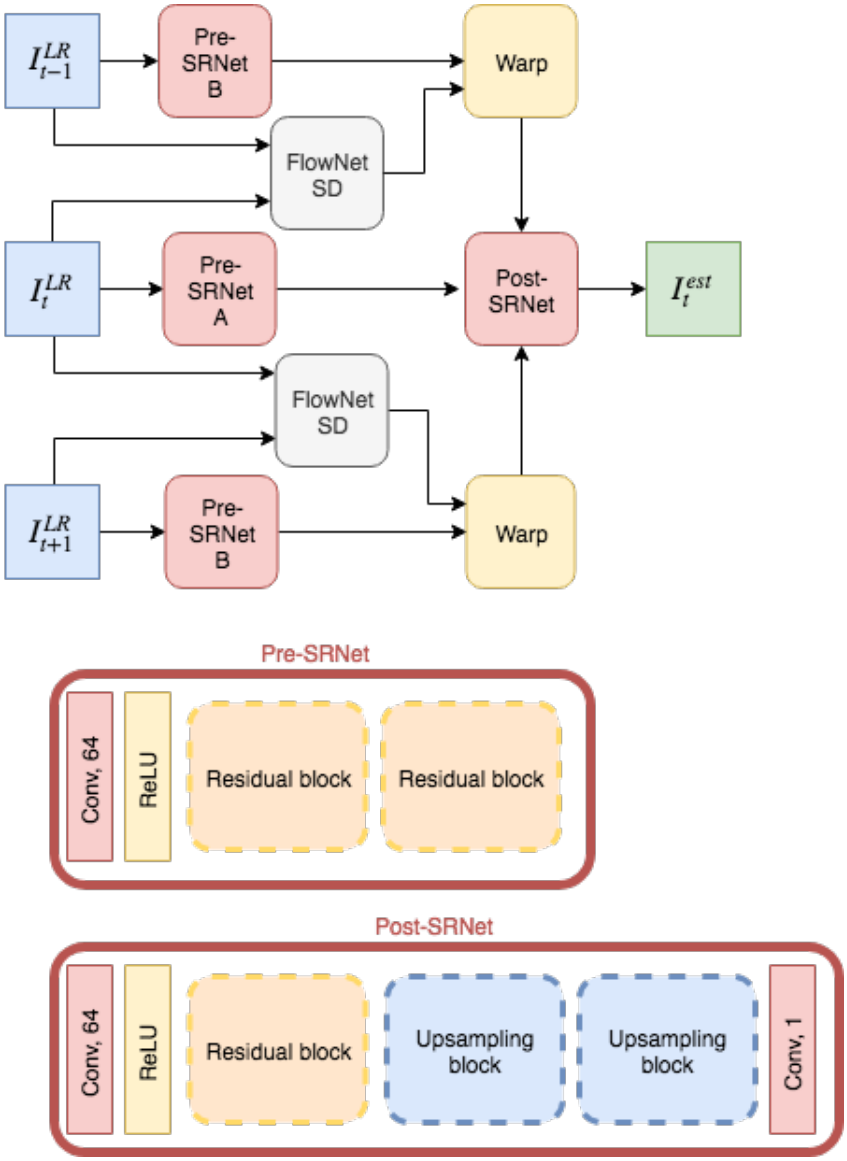


FIGURE 2.7: Schematic architecture of our late feature warping variation.

way to upsample the frames instead of using a pre-defined upsampling strategy. The drawback of up-sampling first lies on the increased computational cost of performing the warping, making the whole architecture much slower compared to all the variations seen so far. There is a trade-off between the accuracy we can achieve and the speed of the network, ideally we would like to be able to optimize both.

2.3.4 Flow augmentation

All the architecture variations seen so far introduce some additional components before the warping operation to manipulate the input frames. In this

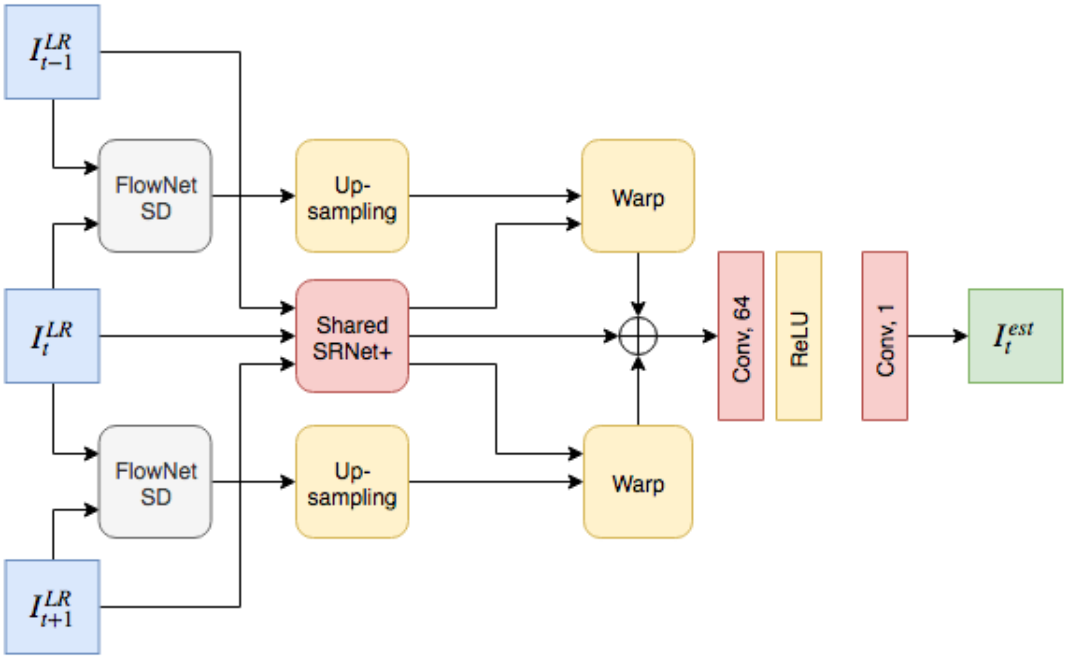


FIGURE 2.8: Schematic architecture of the warping in HR variation.

architecture variation, instead, we intervene after the warping step with the same goal of retaining more high frequency information. In this case we generate additional optical flow estimations adding small displacement to the calculated ones between $\{I_{t-1}^{LR}, I_t^{LR}\}$ and between $\{I_t^{LR}, I_{t+1}^{LR}\}$. In particular we add a 0.5 pixel displacement in all the components of the optical flow (dx, dy and (dx, dy)). The architecture is depicted in Figure 2.9. We do not target directly the warping operation as we have seen in the previous architecture variations but we indirectly pass more information to the super resolution part of the network.

2.3.5 Implicit motion estimation and warping

MFSR has originally been tackled as a two sub-tasks problem, motion estimation and super resolution, therefore also deep learning approaches have inherited this way of looking at the problem. All the models seen so far are in line with this separation of tasks but we present here one example where we move away from this scheme and we use instead what we call "implicit motion estimation and warping". The structure of the network (2.10) is derived directly from the feed-forward baseline presented earlier where we substituted the FlowNetSD and warping steps with a single convolutional layer of size 64 and filter of size 5 followed by a ReLU activation function. We still

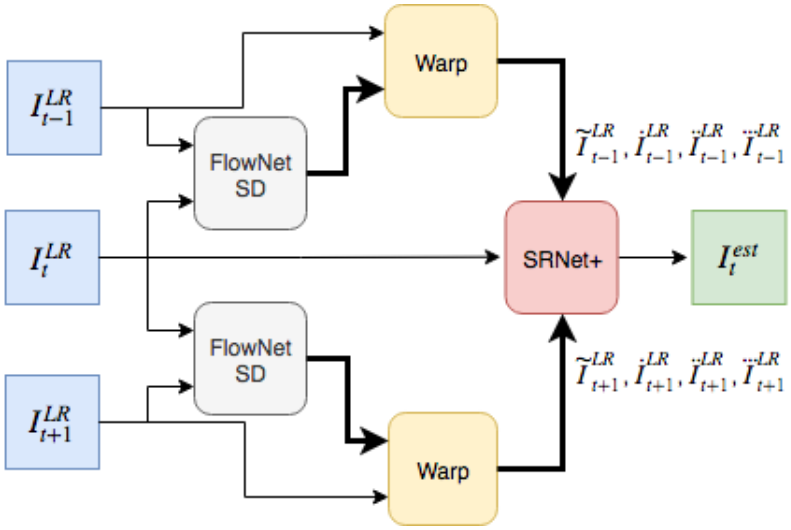


FIGURE 2.9: Schematic architecture of the flow augmentation variation.

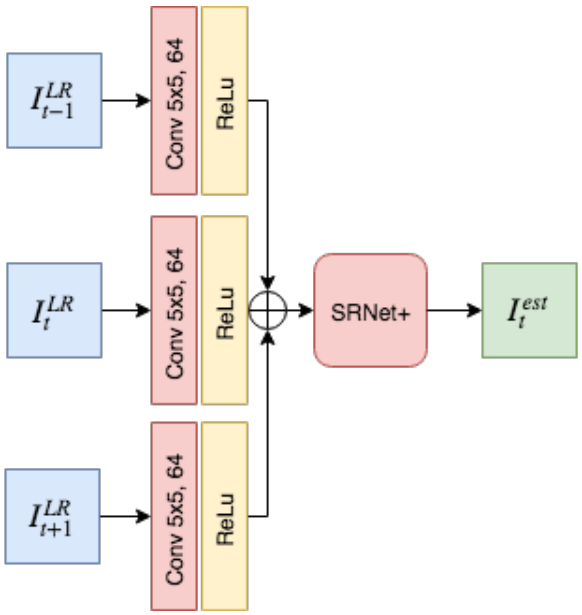


FIGURE 2.10: Schematic architecture of the implicit motion estimation and warping variation.

indicate part of the architecture as SRNet+ for consistency but we would like to stress out that there is no more a clear distinction of tasks given that both the optical flow calculation and the warping steps were removed.

2.4 Dataset

Following Kappeler et al., 2016 we used the Myanmar 4K[†] video as the source for our training and validation set. It is an ~8 minutes footage which contains 59 different scenes of approximately 8 seconds each. The original resolution of the video is 3840x2160. We pre-processed the footage, using ffmpeg[‡], to split it in scenes, to transcode it to a lower resolution (960x540) and to finally extract the frames used as the HR ground truth. We used 53 scenes as the training set and 6 scenes as validation set for all our experiments.

The original video has changed from the one used by Kappeler et al., 2016, it shows now a much bigger company logo on the bottom right which deteriorates the overall performances. In our experiments, we calculate the performances of the networks removing four pixels from each border to account for this change as already done by other authors in the literature.

2.5 Training procedure

All our experiments are performed for $x4$ scale factor SR. We investigated both full frame and patch-based training. The full frame training turned out to slow down the overall training procedure due to long loading times per frame. We finally opted for a patch-based training for all our experiments. The LR-HR pairs to train the models were generated online during training. We used patches of size 272×272 pixels. LR patches were generated applying Gaussian blur with mean $\mu = 0$ and standard deviation $\sigma = 1.5$ to the HR frames, selecting a random patch of size 272×272 from the HR blurred frame and down-scaling the patch by selecting the upper left pixel in every square of 4×4 pixels for a scale factor $s = 4$. The same patch position is used to extract the HR patch from the HR frame. For the Gaussian blur we used the implementation from OpenCV[§].

We used batches of size 64 for the training of feed-forward models and batches of size 16 for the training of recurrent models. The batch size differs between feed-forward and recurrent training because also the element of a batch is different. In the case of feed-forward training, the element of a batch consist in n LR input patches, based on the number of inputs of the model, and one HR target patch (the one we referred as central or reference

[†] https://www.harmonicinc.com/4k-demo-footage-download/
[‡] http://ffmpeg.org/
[§] https://opencv.org

frame/patch, the only one that gets super resolved). In the recurrent case, instead, the element of a batch consists in ten LR patches and nine HR patches which form a sequence.

The training procedure for both feed-forward and recurrent models initializes the network using the Xavier initialization (Glorot and Bengio, 2010), uses the Adam optimizer (Kingma and Lei Ba, 2014) with a fixed learning rate of 10^{-4} and the mean squared error (MSE) as the loss function.

2.6 Performance comparison metrics

In order to measure the performance in our experiments we used the *peak signal-to-noise ratio (PSNR)* and the *structural similarity index (SSIM)* image similarity tests which are the most widely adopted metrics in SR literature. They have been shown to compare poorly with human perceived quality of an image or a video but they allow us to benchmark our performance with the rest of the literature.

2.6.1 Peak signal-to-noise ratio

The peak signal-to-noise (PSNR) ratio is calculated as follow:

$$PSNR = 10 \log_{10} \left(\frac{R^2}{MSE} \right)$$

where MSE is the mean square error between the two images

$$MSE = \frac{\sum_{M,N} [I_1(m,n) - I_2(m,n)]^2}{M * N}$$

and R is the maximum allowed value in the images (e.g 1 or 255).

2.6.2 Structural similarity index

The structural similarity index (SSIM) is a more complex image similarity test compared to the PSNR and it was invented as a replacement for the PSNR

itself. It takes into account three different factors of the compared images: luminance, contrast and structure. The overall formula is a weighted combination of the three factors and for weights equal to 1, it looks like the following:

$$SSIM = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2 + \mu_y^2 + C_1)(\sigma_x^2 + \sigma_y^2 + C_2)}$$

where μ_x and μ_y, σ_x and σ_y, and σ_{xy} are respectively the averages, the variances and the covariance of the compared images. C_1 and C_2 are constants to stabilize the division.

Chapter 3

Results

In this chapter we present the results of the experiments carried out to answer the research questions outlined in Chapter 1. Please note that most of the details about implementation and training procedures are specified in Chapter 2.

3.1 Feed-forward vs. recurrent architecture

The first experiments serve to validate that we can replicate some of the state-of-the-art results seen in the literature in order to have a reference for the following experiments. We investigate here two different types of architecture, an encoder-decoder like feed-forward architecture described in more details in section 2.2.1 and our implementation of the recurrent framework proposed by Sajjadi, Vemulapalli, and Brown, 2018, which we identified as the best performing approach in VSR at the time of our literature review, described in section 2.2.2.

3.1.1 Comparison with literature

The nature of the dataset, a *sequence* of frames, usually implies, in the deep learning literature, the use of a recurrent model as the best performing architecture. Therefore we want to validate if this hypothesis holds also for VSR and understand eventually why one approach is superior to the other.

We mimicked, with the information available, the training procedure by Sajjadi, Vemulapalli, and Brown, 2018 but unfortunately, at the time of our experiments, the authors had not released yet their training set. We decided to use instead another standard dataset, the Myanmar4K, with the training and validation split specified in Chapter 2. We trained both architectures until convergence. Note that the training time for these experiments was

longer than in other experiments because we wanted to achieve comparable results to the one published in the literature.

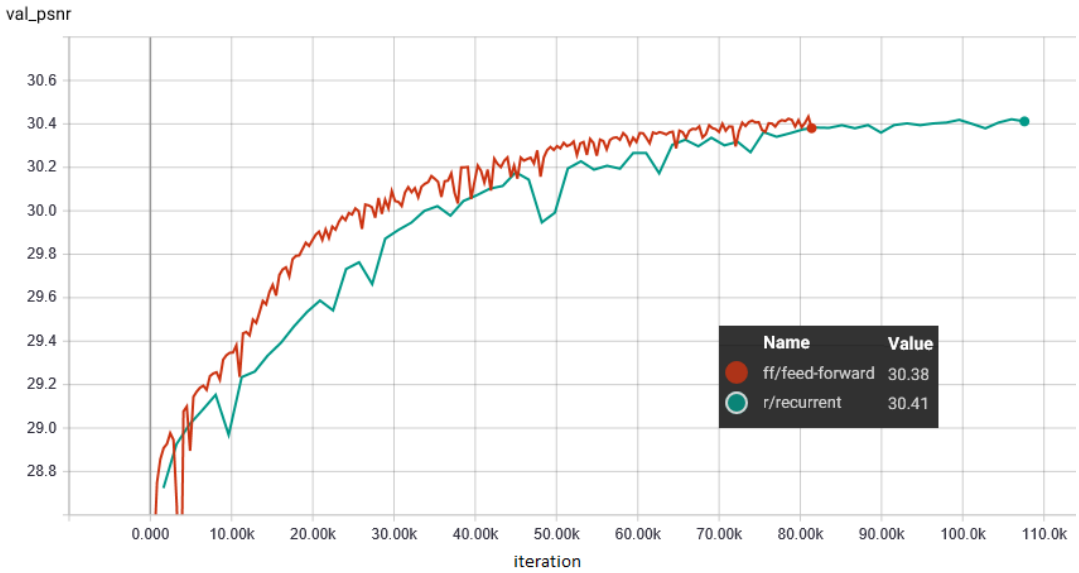


FIGURE 3.1: Evolution of the feed-forward and recurrent architectures PSNR values in decibels (dB), during training, on the Myanmar4K validation set.

Figure 3.1 illustrates how the PSNR evolves in every epoch on the Myanmar4K validation set. The feedforward architecture was trained for ~80K iterations with a batch size of 64 while the recurrent architecture was trained for ~110k iterations with a batch size of 16. The two networks have comparable performances and ultimately converge to similar PSNR values.

	City		Calendar		Walk		Foliage		**Total avg.**	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	**PSNR**	**SSIM**
Feed-forward	26.77	0.75	22.32	0.73	28.45	0.87	25.34	0.72	25.72	**0.77**
Recurrent	26.83	0.75	22.58	0.74	28.58	0.87	25.50	0.73	**25.87**	**0.77**

TABLE 3.1: Feed-forward vs. recurrent architecture PSNRs on VideoSet4 for scaling factor 4.

In Table 3.1 we report the average PSNR and SSIM for each video of the VideoSet4 test set and the total average. Again the performances of the two networks are similar, which is in line with what we observed before on the validation set.

Table 3.2 compares instead our results with what reported by Sajjadi, Vemulapalli, and Brown, 2018. We do not use any other results from the literature in our comparison because we noticed a difference in the downsampling procedure (more on the topic in the Discussion section 4). Both our feedforward and recurrent architectures achieve results well in line with the

Method	Bicubic	SISR 10-128	VSR 10-128	FRVSR 3-64	Ours feedforward	Our recurrent
PSNR	23.53	24.96	**26.25**	26.17	25.72	25.87
SSIM	0.62	0.72	**0.80**	0.79	0.77	0.77

TABLE 3.2: SISR 10-128, VSR 10-128, FRVSR 3-64 PSNR and SSIM values are copied directly from Sajjadi, Vemulapalli, and Brown, 2018. 10-128 and 3-64 refers to (# residual blocks)-(# features convolutional layer). The SISR is a single image super resolution network, the VSR is a more complex model of our feed-forward architecture and the FRVSR is an implementation of the recurrent framework we used.

FRVSR 3-64 model, which is the most fair competitor based on the complexity of our models.

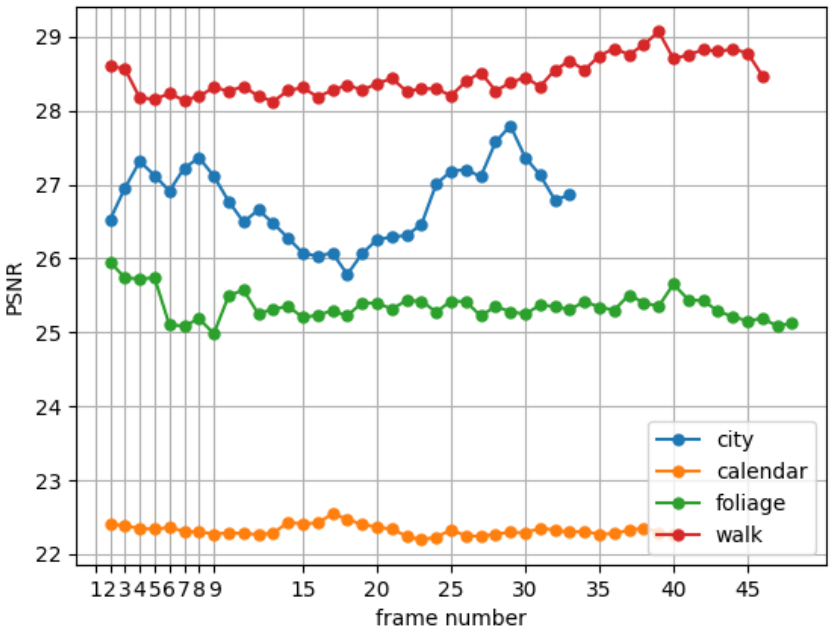
3.1.2 Temporal behaviour

The quantitative and qualitative difference between the two types of architecture does not seem to be so obvious so far. Therefore to gain a deeper understanding of the behaviour of the two networks over time, we plot in Fig. 3.2 the PSNR for each frame of every video of the VideoSet4 test set.

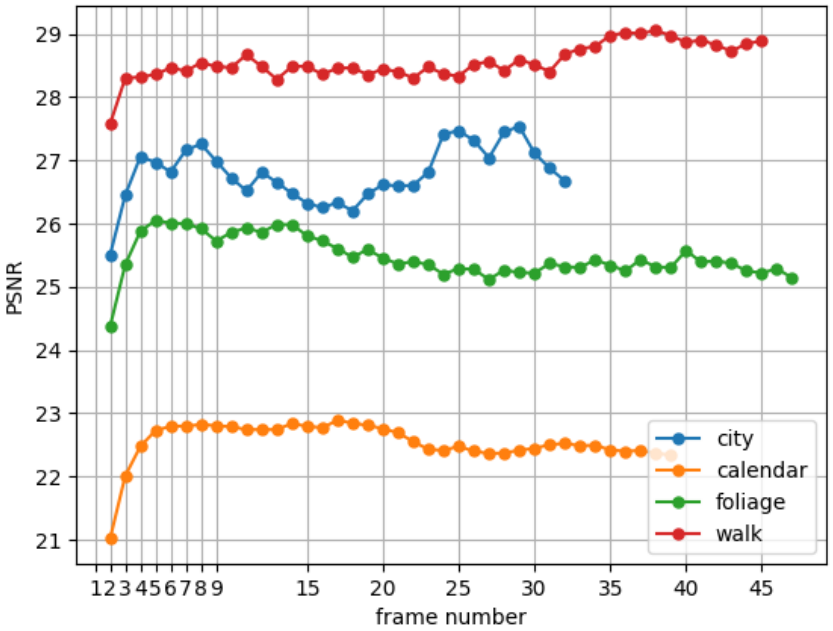
The recurrent architecture (3.2b) takes approximately three frames, on average, to match the performances of the feed-forward one (3.2a). This phenomenon is explained by the type of initialization of the recurrent architecture. The recurrent state, in this case an HR estimate of the frame at step $t-1$, which contains all the past information, gets initialized as a black image. Therefore the recurrent architecture needs some time steps to accumulate past information which explains the lower PSNR in the first frames. Another important thing to notice is that the recurrent architecture, once matched the performances of the feed-forward one, maintains a more stable PSNR over time. Particularly from the test sequence "city" in fig. 3.2 we can see this stability which suggests that the recurrent architecture is exploiting its ability to use information from more distant frames in the past.

3.1.3 The role of using multiple frames

The temporal behaviour of the recurrent and the feed-forward architectures gave us already some insights on how the structure of the network affects the way the frames are processed. The recurrent architecture takes a certain amount of time steps to match the performances of the feed-forward one, in this phase there is an increase in the PSNR value. The number of time steps



(A) Feed-forward



(B) Recurrent

FIGURE 3.2: PSNR of reconstructed images as a function of frame index for each single video frame of VideoSet4 test set.

tells us also, approximately, the number of frames the recurrent architecture is able to use for the HR frame generation.

In this section we carry out an analysis to understand the importance of using multiple frames and we look at how many input frames are most beneficial to use in the feed-forward case.

First of all, we run the feed-forward architecture on three identical input frames, which corresponds to a special case of SFSR because there is no information coming from the neighboring frames, to check the advantage of using a multi-frame approach. We refer to this feed-forward architecture with three identical input frames as "Special SFSR". In another experiment we run the feed-forward architecture using five input frames instead of three to see if we have an increase in performance. We refer to this architecture as "Feed-forward 5" opposed to the feed-forward baseline of section 3.1.1 that we refer as "Feed-forward 3". Both experiments were trained for the same number of iterations (~80K) and reached full convergence as the feed-forward baseline.

	City		Calendar		Walk		Foliage		**Total avg.**	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	**PSNR**	**SSIM**
Special SFSR	25.22	0.64	21.06	0.65	26.04	0.82	23.58	0.62	23.97	0.68
Feed-forward 3	26.77	0.75	22.32	0.73	28.45	0.87	25.34	0.72	25.72	0.77
Feed-forward 5	26.89	0.75	22.51	0.74	28.44	0.87	25.32	0.72	**25.79**	**0.77**

TABLE 3.3: Special SFSR (three identical input frames), Feeforward 3 (feedforward baseline seen previously) and Feedforward 5 (feedforward baseline with 5 input frames) architecture PSNRs on VideoSet4 for scaling factor 4.

Table 3.3 compares their PSNR and SSIM values on VideoSet4 test set. The ~2 dB difference in PSNR between the Feed-forward 3 and Special SFSR demonstrates the advantage of using a multi-frame approach. There are indeed more information in neighbouring frames that the network can exploit to make a better estimation. This comes with no surprise because other authors (e.g. Makansi, Ilg, and Brox, 2017) in the literature have already proved those benefits. The comparison between the Feed-forward 3 and Feed-forward 5 proves, instead, that the feed-forward architecture obtains a negligible improvement in using more than 3 frames as input. This result is not generalizable for every situation because it depends on the video we are using for testing but it can explain the little difference found between a recurrent and feed-forward structure. Furthermore a similar result was already reported by Kappeler et al., 2016 on the Myanmar4k dataset. From the

previous analysis we can conclude that the recurrent architecture as the Feed-forward 5 can not integrate information from more than 3 frames therefore losing its advantage.

3.2 Impact of warping

In chapter 1 we introduced already the concept of warping. This transformation performs a low-pass filtering (or smoothing) on the resulting image which suppress high-frequency information. Why is it an important aspect to consider in our case? The baseline architectures, explained in chapter 2, both make use of the warping operator to align previous and future frames to the one taken as reference (frame at time t) which gets super resolved. Motion estimation in combination with the warping operation allow the network to work with additional information. Unfortunately warping also causes most of the high frequencies to get lost and our attempt of a more detailed reconstruction vanishes together with the advantage of using a multi-frame approach.

3.2.1 Mitigate the effect of warping

Previously we investigated the difference between a recurrent and a feed-forward architecture without finding supportive arguments for one or the other. In this section we decide to use a feed-forward structure because we think that it does not affect the final outcome of the experiments. Our hypothesis is that new findings can be applied interchangeably between the two types of architecture, without loss of generality. Our intervention affects only the common operation of warping which is not specifically connected to the particular structure of the network. The advantage, in this case, lies on the shorter training time per experiment of the feed-forward architecture. To further reduce the time per experiment we trained each instance until close to convergence which demonstrated to be enough to draw our conclusions.

In this series of experiments we investigate if the presumed negative effect of the warping operation can be reduced such that it can have a positive impact on the performance of the VSR architecture. We explore some variations of the feed-forward baseline (2.2.1). The architecture variations Feature Warp (2.3.1), Late Warp (2.3.2) and Warp in HR (2.3.3) are all based on the

same idea, embedding. Embedding, in this case, can be explained as mapping the input frames to an higher dimensional feature space using, a single convolutional layer, as in the case of Feature Warp or with, more layers, as in the case of Late Warp and Warp in HR, before performing the warping operation. The network should learn ideally high-dimensional features that gets affected less by the negative effect of warping. The variation Flow augmentation (2.3.4), feeds, instead, additional warped frames to the super resolution part of the network, adding small displacement to the calculated motion between input frames. For more details about each architecture variation please refer to chapter 2 under the Architecture variations section.

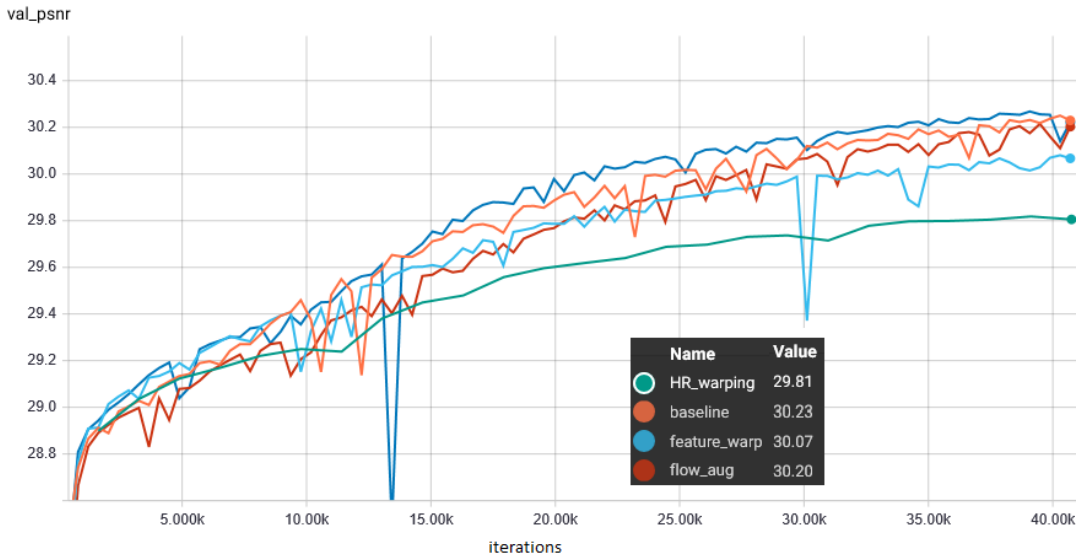


FIGURE 3.3: Architecture variations PSNR values during training on Myanmar4k validation set.

In figure 3.3 we show the evolution of the PSNR on the Myanmar4k validation set. The PSNR difference (in dB) among the different approaches looks stable at the end of the graph, suggesting that none of these variations seem to bring substantial improvements over the baseline architecture. The shorter training time does not seem to influence the previous conclusion.

	City		Calendar		Walk		Foliage		Total avg.	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Baseline	26.56	0.73	22.12	0.72	28.33	0.86	25.27	0.71	**25.57**	**0.76**
Feature warp	26.39	0.72	21.89	0.70	28.15	0.86	25.01	0.69	25.36	0.74
Late feature	26.52	0.73	22.12	0.72	28.30	0.86	25.24	0.71	25.55	**0.76**
Warp in HR	26.05	0.70	21.56	0.68	27.75	0.85	24.56	0.67	24.98	0.72
Flow augmentation	26.10	0.70	21.42	0.66	27.49	0.84	24.65	0.68	24.92	0.72

TABLE 3.4: Baseline 2.2.1, Feature warp 2.3.1, Late warp 2.3.2, Warp in HR 2.3.3 and Flow augmentation 2.3.4 PSNR and SSIM values for single VideoSet4 videos and total average.

The results on VideoSet4, which are shown in Table 3.4, further validate the point. We also run a similar analysis to the one in Figure 3.2 to observe if there is any noticeable difference in the behaviour over time of every variation but we did not find any substantial difference and therefore we decided to not report this analysis.

3.2.2 Implicit vs. explicit motion estimation

So far we learned that mapping the input frames to a new representation before the warping step or passing more warped frames to the super resolution part of the network does not help in improving the performance of the feed-forward baseline. We continue our investigation on the impact of warping considering what would happen if we stop treating the problem of VSR as a two sub-tasks problems, the motion estimation and the super resolution. This last way of looking at the problem derives from the traditional approach to VSR. The advantage of letting the neural network figure out the best way to use the information inherent in neighboring frames allows us to drop completely the need of an explicit warping step. The architecture with an implicit motion estimation is explained in details in section 2.3.5 and we refer here as "No Flow". It resembles the feed-forward baseline but where we removed the FlowNet-SD plus warping in favor of a single convolutional layer. Table 3.5 shows a PSNR comparison with the feed-forward baseline

	City		Calendar		Walk		Foliage		Total avg.	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Baseline	26.56	0.73	22.12	0.72	28.33	0.86	25.27	0.71	**25.57**	**0.76**
No Flow	26.34	0.72	21.78	0.69	28.04	0.86	25.01	0.70	25.29	0.74

TABLE 3.5: PSNR values of the feed-forward baseline 2.2.1 and the implicit motion estimation network 2.2.2 for VideoSet4 test set.

on the VideoSet4 test set. We find no substantial difference between the two architectures which suggests that the convolutional layer substituting the explicit motion estimation is a feasible alternative for the explicit motion estimation.

In order to further validate these findings we run another analysis which looks at the performances of the two architectures when neighboring frames further in time gets fed to the networks at test time. We expect the network using the implicit motion estimation to be able to cope also with this situation, confirming the possibility to safely substitute the explicit motion estimation part. In particular we feed the two networks with frames one, two

and three time steps further apart at test time. This analysis should insure that the previous results are not based only on the fact that in the test set video there is no large motion which can be handled also by simpler implicit motion estimation models.

Method	City		Calendar		Walk		Foliage		Total avg.	
	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Baseline 1	26.16	0.70	21.98	0.70	28.33	0.86	24.63	0.67	**25.16**	**0.73**
No Flow 1	26.06	0.69	21.73	0.69	27.73	0.85	24.51	0.66	25.01	0.72
Baseline 2	25.73	0.67	21.67	0.68	27.60	0.85	24.39	0.65	**24.85**	**0.71**
No Flow 2	25.72	0.67	21.53	0.67	27.55	0.85	24.27	0.65	24.77	**0.71**
Baseline 3	25.42	0.65	21.40	0.66	27.49	0.85	24.30	0.65	**24.65**	**0.70**
No Flow 3	25.47	0.65	21.32	0.66	27.47	0.84	24.23	0.64	24.62	**0.70**

TABLE 3.6: Feed-forward baseline 2.2.1 and No Flow implicit motion estimation network 2.3.5 PSNR values using VideoSet4 input frames 1 (Baseline 1, No Flow 1), 2 (Baseline 2, No Flow 2) and 3 (Baseline 3, No Flow 3) time steps apart.

The results are shown in Table 3.6. It looks that what already observed before still holds also for this new analysis confirming that an implicit motion estimation can achieve performances comparable to the explicit one.

3.3 Additional Gaussian noise variation

After a critical analysis of the state-of-the-art results with a feed-forward versus recurrent comparison, a look at potential downsides of using the warping operation with some architecture variations and, the possibility to treat the VSR problem as single task problem with implicit motion estimation we now turn our attention to the real world.

So far we have adopted the image observation model from Sajjadi, Vemulapalli, and Brown, 2018 which, as said before, closely resemble one of the simplest models used also by TNO, to be able to compare our results and findings. In this last section we go one step further and we match completely an image observation model used at TNO substituting the simple down-sampling, first pixel value in every four pixels square in the case of $x4$ scale factor, with an average pooling, the average among sixteen pixels is taken in the case of $x4$ scale factor (more detailed information can be found in Chapter 2). The average pooling down-sampling best resemble a simulation of the physical world for the type of applications TNO is interested in. We decided to change the image observation model for these last experiments

because we are interested in understanding the robustness of the two types of architecture to noise for real-world applications.

We trained the feed-forward 2.2.1 and recurrent 2.2.2 models for ~80K iterations adding random Gaussian noise with a standard deviation in the interval $\sigma = [0.00, 0.05]$ to the input frames.

	$\sigma = 0$		$\sigma = 0.01$		$\sigma = 0.03$		$\sigma = 0.05$	
Method	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM	PSNR	SSIM
Feed-forward NV	24.96	0.71	24.86	0.70	24.42	0.67	**23.94**	**0.63**
Feed-forward w\o NV	**25.57**	**0.76**	**25.36**	**0.74**	24.25	0.64	22.87	0.54
Recurrent NV	25.07	0.72	24.96	0.71	**24.44**	**0.67**	23.90	0.63
Recurrent w\o NV	25.53	0.75	25.26	0.73	24.08	0.62	22.75	0.53

TABLE 3.7: Feed-forward vs. recurrent architecture PSNRs on VideoSet4 for different noise levels.

Table 3.7 reports the PSNR values of four noise levels applied to the VideoSet4 test set for both architecture structures trained with and without the noise variation. Again there is no substantial difference between the feedforward and the recurrent architecture and, as expected, when the level of noise increases the models trained with the noise variation perform better than the models trained without.

Chapter 4

Discussion

Based on the research questions posed in Chapter 1 and with the intuitions and results built in Chapter 3 we reflect critically on our design decisions and what can be eventually improved for future research on the subject.

4.1 State of the art

The first topic we are going to cover does not only concern our results but it is a more general observation on the VSR literature which uses deep learning methodologies.

As we pointed out already in different part of the text, based on our literature study, we believe the current SOTA architecture in VSR is the one proposed by Sajjadi, Vemulapalli, and Brown, 2018, the one we based most of our research on. We chose to use this paper also for the type of image observation model selected by the authors which is the only one resembling the image observation model developed within this research project. The image observation model is a key element for the comparability of the results among different research papers and it generally determines the application(s) the VSR system is targeting. In most of the papers we reviewed, we noticed that the target application(s) of the paper is not always clear which in turn results in a heterogeneity in the image observation models used. This heterogeneity, which is proved also by the bicubic PSNR value reported by different authors that can differ by ~0.5 dB on the same test set, leads to an impossibility to compare results and determine a unique SOTA in the field. Therefore we find difficult to insert our research results in the context of the deep learning VSR literature as a whole which explains why we made comparison only with the paper from Sajjadi, Vemulapalli, and Brown, 2018. We believe that, in order to facilitate the comparison of the results from different papers, it is extremely important to create a challenge or a pipeline similar to

the one that has been created in SFSR*. In this challenge there are different tracks that define the dataset, the type of image observation model used to create the dataset and the particular implementation and parameters used for the image observation model components (i.e. Matlab imresize).

In addition, as briefly mentioned already in Chapter 2, the image similarity metrics (PSNR and SSIM), widely adopted in the literature to compare the different VSR approaches, fail to capture the human perceived quality of the images. In the particular case of VSR, these similarity metrics do not give any insight on the temporal dimension of the results treating the videos as separate frames. Furthermore, the quantitative difference between two images, calculated using the previously mentioned image similarity metrics, does not allow to make clear and meaningful conclusions. What does a difference in PSNR between two images (or frames) of 0.1 dB compared to one of 0.3 dB tell us? When is a difference in PSNR substantial or significant for a real added value in the final image reconstruction?

Too often in our literature review we noticed claims of SOTA results without any statistical significant tests but solely based on small quantitative difference in PSNR or SSIM. We believe it is important to come up with an image or video similarity metric which can address all these problems and which allows a more transparent comparison.

4.2 Feed-forward vs recurrent

The framework presented by Sajjadi, Vemulapalli, and Brown, 2018 is one of the few examples in VSR using a recurrent approach. In our investigation of Section 3.1 we used the framework as inspiration for our recurrent architecture and we compared it with a feed-forward architecture of similar complexity. We did not find any benefits in using a recurrent approach other than a slightly more stable PSNR value which, ultimately, did not translate in better performance as otherwise reported by Sajjadi, Vemulapalli, and Brown, 2018. The causes of the recurrent architecture not standing out against the feed-forward one can be related to different factors.

First, the complexity of the models we used. Sajjadi, Vemulapalli, and Brown, 2018 do not report the performance of their smaller baseline feed-forward model (3 residual blocks, convolutional layers of size 64) but they state that a more complex model is beneficial for the recurrent architecture. We decided not to use more complex models in the interest of time so, we

*NTIRE2018

can only conclude that the model complexity we used (3 residual blocks and convolutional layers of size 64) was not enough to see substantial difference between the two architecture structures.

Second, the analysis we made about the importance of using multiple frames (section 3.1.3) revealed that the use of five frames in the case of the feed-forward architecture did not show to improve substantially the performance. The maximum number of frames containing relevant information for the final HR reconstruction on the test set used, can explain the lack of difference between the two models. Unfortunately this last finding contrast slightly with some of the findings from Sajjadi, Vemulapalli, and Brown, 2018 where for more complex models they indeed see a difference in performance between the two structures. The hypothesis we made in this case is that a more complex model should be able to process information from more frames.

Finally, as we pointed out in the Chapter 3, we could not use exactly the same training set as the original paper because at the time of our experiments the author had not yet published the list of video used as training set. The difference in training sets is the least probable cause of failure but it is worth noting because it is also, probably, related to the overall lower performance achieved by both our architectures compared to the original paper.

4.3 Warping

TNO expertise in traditional super resolution guided the exploration of the potential negative effects of the warping operation. It is known from digital image processing theory that the warping operation applies a low-pass filter to the warped image which suppress some of the high frequencies in the image. In order to overcome this problem we looked at ways that could potentially alleviate this effect hoping to retain more high frequency information which, in turn, translate to a more detailed HR reconstruction. Unfortunately all the architecture variations we tried did not show to improve the overall performance of the feed-forward model both quantitatively and qualitatively which suggest that the warping operation may not be a problem. Our hypothesis is that the neural network is treating the problem of VSR in a fundamental different way compared to a traditional approach, the neural network is trying to reconstruct structures while a more traditional approach focus on reconstructing a signal. This hypothesis is also backed up

by our analysis on the implicit versus explicit motion estimation and warping. The comparison looked at the possibility of treating the problem as a single problem instead of dividing it in two different sub-tasks, motion estimation and super resolution, as done in traditional VSR. The model using the implicit motion estimation and warping proved to be a valid alternative to the one using an explicit motion estimation also in the challenging case of input frames further apart in time. These findings can explain the negative outcome of our architecture variations for the warping operation and they can go as far as telling us that we should let the neural network figure out the best way of solving the problem without forcing a task separation in its structure.

4.4 Real world applications

Our last analysis looked at how to align the image observation model to the one used for simulation of real-world application at TNO. We replaced the type of down-sampling used, we trained both the feed-forward and the recurrent model adding random noise to input frames and we compared them with models without noise variation. We found again no substantial difference between the two types of architecture. The models trained with the noise variation proved to be more robust to noise at test time, as expected. We thought the recurrent model would prove to be more robust, averaging out the noise using information from more frames but it was not the case. This finding partially confirm the fact that the recurrent architecture is not able to use information from more than three frames therefore losing its advantage against the feed-forward one and performing on par.

Chapter 5

Conclusions

The research project started with a clear and ambitious goal, improve the current state-of-the-art in video super resolution. Video super resolution has been studied since the 80s and many progresses have been achieved so far especially with the recent advent of deep learning techniques. Super resolution and in particular video super resolution still remains a very challenging problem also due to its ill-posed nature. Many applications can potentially benefit from improvements in SR techniques therefore it is important to keep on pushing the boundaries of the field.

Along the way we reviewed the literature on VSR using deep learning techniques, we identified potential elements for improvement and we tested our hypothesis after partially reproducing some of the SOTA results. These are some of our contributions:

- In the baselines analysis, we compared a feed-forward and a recurrent architecture both based on the best performing networks in the literature and we did not find any substantial difference between the two structures.

- Using the feed-forward architecture as baseline, we came up with several architecture variations to investigate the impact of warping on performance. All our architecture variations turned out to not improve the performance compared to the baseline.

- Traditional VSR techniques approached the problem by dividing it in two sub-tasks: motion estimation and super resolution. We showed that deep learning techniques perform equally well without the need of sub-tasks division.

- Finally, given the collaboration with TNO for this thesis, we looked into more realistic image observation models and we proved the better

robustness to noisy environment of networks trained with noise variation.

Appendix A

Qualitative comparison VideoSet4 test set

For this qualitative comparison on the VideoSet4 test set we take only our best performing approach which corresponds to the recurrent architecture of Section 2.2.2 referred as "Ours". The other two sample images shown correspond to a bicubic upsampling (using Matlab imresize) and the original HR frame.

(A) Bicubic

(B) Ours

(C) Original

FIGURE A.1: The VideoSet4 sequence Calendar.

(A) Bicubic

(B) Ours

(C) Original

FIGURE A.2: The VideoSet4 sequence City.

(A) Bicubic

(B) Ours

(C) original

FIGURE A.3: The VideoSet4 sequence Foliage.

(A) Bicubic

(B) Ours

(C) Original

FIGURE A.4: The VideoSet4 sequence Calendar.

Bibliography

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (2012). "Learning Long-Term Dependencies with Graident Descent is Difficult". In: *Saudi Med J* 33, pp. 3–8. ISSN: 0027-8424. DOI: 10.1073/pnas.0703993104.

Bevilacqua, Marco (2014). "Algorithms for super-resolution of images and videos based on learning methods". In: URL: https://tel.archives-ouvertes.fr/tel-01064396/document.

Caballero, Jose et al. (2016). "Real-Time Video Super-Resolution with Spatio-Temporal Networks and Motion Compensation". In: DOI: 10.1109/CVPR.2017.304. arXiv: 1611.05250. URL: http://arxiv.org/abs/1611.05250.

Dong, Chao et al. (2014). "Learning a Deep Convolutional Network for Image Super-Resolution". In: *Proc. of European Conf. on Computer Vision (ECCV)*, pp. 184–199. ISSN: 15505499. DOI: 10.1007/978-3-319-10593-2_13. arXiv: 1506.0275. URL: http://arxiv.org/abs/1506.0275.

Dosovitskiy, Alexey et al. (2015). "FlowNet: Learning optical flow with convolutional networks". In: *Proceedings of the IEEE International Conference on Computer Vision* 2015 International Conference on Computer Vision, ICCV 2015, pp. 2758–2766. ISSN: 15505499. DOI: 10.1109/ICCV.2015.316. arXiv: 1504.06852.

Drulea, M and S Nedevschi (2011). "Total variation regularization of local-global optical flow". In: *IEEE Conference on Intelligent Transportation Systems, Proceedings, ITSC*, pp. 318–323. ISSN: 2153-0009. DOI: 10.1109/ITSC.2011.6082986.

Farsiu, Sina et al. (2004). "Fast and robust multiframe super resolution". In: *IEEE Transactions on Image Processing* 13.10, pp. 1327–1344. ISSN: 10577149. DOI: 10.1109/TIP.2004.834669.

Glorot, Xavier and Yoshua Bengio (2010). *Understanding the difficulty of training deep feedforward neural networks*. Tech. rep., pp. 249–256. DOI: 10.1.1.207.2059. arXiv: arXiv:1011.1669v3. URL: http://machinelearning.wustl.edu/mlpapers/paper{_}files/AISTATS2010{_}GlorotB10.pdf.

He, Kaiming et al. (2015). "Deep Residual Learning for Image Recognition". In: *Arxiv.Org* 7.3, pp. 171–180. ISSN: 1664-1078. DOI: 10.3389/fpsyg.2013.

00124. arXiv: 1512.03385. URL: http://arxiv.org/pdf/1512.03385v1.pdf.

Hornik, K., M. Stinchcombe, and H. White (1989). "Multilayer Feedforward Networks Are Universal Approximators". In: *Neural Netw.* 2.5, pp. 359–366. ISSN: 0893-6080. DOI: 10.1016/0893-6080(89)90020-8. URL: http://dx.doi.org/10.1016/0893-6080(89)90020-8.

Huang, T. S. and R. Y. Tsay (1984). "Multiple frame image restoration and registration". In: *Advances in Computer Vision and Image Processing*. Vol. 1. Greenwich: JAI, pp. 317–339.

Huang, Yan, Wei Wang, and Liang Wang (2018). "Video Super-Resolution via Bidirectional Recurrent Convolutional Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4, pp. 1015–1028. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2017.2701380. URL: http://ieeexplore.ieee.org/document/7919264/.

Ilg, Eddy et al. (2017). "FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks". In: *Cvpr*, pp. 1–9. ISSN: 10636919. DOI: 10.1109/CVPR.2017.179. arXiv: 1612.01925. URL: http://openaccess.thecvf.com/content{_}cvpr{_}2017/papers/Ilg{_}FlowNet{_}2.0{_}Evolution{_}CVPR{_}2017{_}paper.pdf.

Irani, M. and S. Peleg (1990). "Super resolution from image sequences". In: *[1990] Proceedings. 10th International Conference on Pattern Recognition*. Vol. ii, 115–120 vol.2. DOI: 10.1109/ICPR.1990.119340.

Johnson, Justin, Alexandre Alahi, and Li Fei-Fei (2016). *Perceptual losses for real-time style transfer and super-resolution*. Tech. rep., pp. 694–711. DOI: 10.1007/978-3-319-46475-6_43. arXiv: 1603.08155. URL: https://arxiv.org/pdf/1603.08155.pdf.

Kappeler, Armin et al. (2016). "Video Super-Resolution With Convolutional Neural Networks". In: *IEEE Transactions on Computational Imaging* 2.2, pp. 109–122. ISSN: 2333-9403. DOI: 10.1109/TCI.2016.2532323. URL: http://ieeexplore.ieee.org/document/7444187/.

Kingma, Diederik P and Jimmy Lei Ba (2014). *ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION*. Tech. rep. arXiv: 1412.6980v5. URL: https://arxiv.org/pdf/1412.6980v5.pdf.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances In Neural Information Processing Systems*, pp. 1–9. ISSN: 10495258. DOI: http://dx.doi.org/10.1016/j.protcy.2014.09.007. eprint: 1102.0183. URL:

https://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Lecun, Y. et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324. ISSN: 00189219. DOI: 10.1109/5.726791. URL: http://ieeexplore.ieee.org/document/726791/.

Lim, Bee et al. (2017). "Enhanced Deep Residual Networks for Single Image Super-Resolution". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops* 2017-July, pp. 1132–1140. ISSN: 21607516. DOI: 10.1109/CVPRW.2017.151. arXiv: 1707.02921.

Liu, Ding et al. (2017). "Robust Video Super-Resolution with Learned Temporal Dynamics". In: *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2526–2534. ISSN: 15505499. DOI: 10.1109/ICCV.2017.274. URL: http://ieeexplore.ieee.org/document/8237536/.

Makansi, Osama, Eddy Ilg, and Thomas Brox (2017). "End-to-End learning of video super-resolution with motion compensation". In: *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10496 LNCS. Springer, Cham, pp. 203–214. ISBN: 9783319667089. DOI: 10.1007/978-3-319-66709-6_17. arXiv: 1707.00471. URL: http://link.springer.com/10.1007/978-3-319-66709-6{_}17.

Mikolov, Tomas et al. (2013). "Distributed Representations of Words and Phrases and their Compositionality". In: *arXiv e-prints*, arXiv:1310.4546, arXiv:1310.4546. arXiv: 1310.4546 [cs.CL].

Minsky, Marvin and Seymour Papert (1969). *Perceptrons: An Introduction to Computational Geometry*. Cambridge, MA, USA: MIT Press.

Mozer, Michael C. (1995). "Backpropagation". In: ed. by Yves Chauvin and David E. Rumelhart. Hillsdale, NJ, USA: L. Erlbaum Associates Inc. Chap. A Focused Backpropagation Algorithm for Temporal Pattern Recognition, pp. 137–169. ISBN: 0-8058-1259-8. URL: http://dl.acm.org/citation.cfm?id=201784.201791.

Nasrollahi, Kamal and Thomas B. Moeslund (2014). "Super-resolution: A comprehensive survey". In: *Machine Vision and Applications* 25.6, pp. 1423–1468. ISSN: 14321769. DOI: 10.1007/s00138-014-0623-4. arXiv: 1610.04490. URL: http://vbn.aau.dk/files/197899085/super{_}resolution{_}survey.pdf.

Odena, Augustus, Vincent Dumoulin, and Chris Olah (2016). "Deconvolution and Checkerboard Artifacts". In: *Distill*. DOI: 10.23915/distill.00003. URL: http://distill.pub/2016/deconv-checkerboard.

Park, Sung Cheol, Min Kyu Park, and Moon Gi Kang (2003). "Super-resolution image reconstruction: a technical overview". In: *IEEE Signal Processing Magazine* 20.3, pp. 21–36. ISSN: 1053-5888. DOI: 10.1109/MSP.2003.1203207.

Reda, Fitsum et al. (2017). *flownet2-pytorch: Pytorch implementation of FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks*. https://github.com/NVIDIA/flownet2-pytorch.

Rosenblatt, Frank (1958). "The Perceptron: A Probabilistic Model for Information Storage and Organization in The Brain". In: *Psychological Review*, pp. 65–386.

Sajjadi, Mehdi S. M., Raviteja Vemulapalli, and Matthew Brown (2018). "Frame-Recurrent Video Super-Resolution". In: arXiv: 1801.04590. URL: http://arxiv.org/abs/1801.04590.

Schultz, R. R. and R. L. Stevenson (1994). "A Bayesian approach to image expansion for improved definition". In: *IEEE Transactions on Image Processing* 3.3, pp. 233–242. ISSN: 1057-7149. DOI: 10.1109/83.287017.

Shi, Wenzhe et al. (2016). "Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1874–1883. ISSN: 1609.05158. DOI: 10.1109/CVPR.2016.207. arXiv: 1609.05158. URL: http://ieeexplore.ieee.org/document/7780576/.

Tao, Xin et al. (2017). "Detail-revealing Deep Video Super-resolution". In: 413113. ISSN: 15505499. DOI: 10.1109/ICCV.2017.479. arXiv: 1704.02738. URL: http://arxiv.org/abs/1704.02738.

Yang, Jianchao and Thomas Huang (2010). "Image super-resolution: Historical overview and future challenges". In: *Super-resolution imaging*, pp. 3–25. URL: http://books.google.com/books?hl=en{\&}lr={\&}id=fjTUbMnvOkgC{\&}oi=fnd{\&}pg=PA1{\&}dq=Image+super-resolution:+Historical+overview+and+future+challenges{\&}ots=53GZConOGy{\&}sig=oHgGJVo{_}57Tv19cxLV{_}XMtC1OPw.