# Deep Learning Based Approaches For Video Motion Magnification

Thesis Report Submitted to
Indian Institute of Technology Ropar
in partial fulfilment of the requirements for the
Degree of

**DOCTOR OF PHILOSOPHY**

By

**Jasdeep Singh**
(Reg.No. 2019eez0006)

Under the guidance of

**Dr. Subrahmanyam Murala**
**Dr. G Sankara Raju Kosuru**


**Department of Electrical Engineering,**

**Indian Institute of Technology Ropar**

**Rupnagar-140001, Punjab, India**

**May-2024**

# Dedicated to My Beloved Family

- Who are the inspiration and power behind success of this work

# Declaration of Originality

I hereby declare that the work which is being presented in the thesis entitled **MODERN DEEP LEARNING BASED APPROACHES FOR VIDEO MOTION MAGNIFICATION** has been solely authored by me. It presents the result of my own independent research conducted during the time period from AUGUST-2019 to APRIL-2024 under the supervision of Dr. Subrahmanyam Murala, Associate Professor, School of Computer Science and Statistics, Trinity College Dublin, and Dr. G. Sankara Raju Kosuru, Associate Professor, Department of Mathematics, Indian Institute of Technology Ropar. To the best of my knowledge, it is an original work, both in terms of research content and narrative, and has not been submitted or accepted elsewhere, in part or in full, for the award of any degree, diploma, fellowship, associateship, or similar title of any university or institution. Further, due credit has been attributed with appropriate citations and acknowledgments, in line with established ethical norms and practices. I also declare that any idea/data/fact/source stated in my thesis has not been fabricated/ falsified/ misrepresented. All the principles of academic honesty and integrity have been followed. I fully understand that if the thesis is found to be unoriginal, fabricated, or plagiarized, the Institute reserves the right to withdraw the thesis from its archive and revoke the associated Degree conferred. Additionally, the Institute also reserves the right to appraise all concerned sections of society of the matter for their information and necessary action (if any). If accepted, I hereby consent for my thesis to be available online in the Institute's Open Access repository, inter-library loan, and the title & abstract to be made available to outside organizations.

Signature

Name: Jasdeep Singh
Entry Number: 2019eez0006
Program: Doctor of Philosophy (Ph.D.)
Department: Electrical Engineering
Indian Institute of Technology Ropar
Rupnagar, Punjab 140001

Date: 21 May 2024

# Acknowledgement

No words are sufficient to express my sincere gratitude to my guides, **Dr. Subrahmanyam Murala**, and **Dr. G. Sankara Raju Kosuru**, for their continuous encouragement, technical discussions, and invaluable guidance during my Ph.D. It has been a great honour and privilege to work under his supervision and to be his student.

I take this opportunity to convey my sincere thanks to **Prof. Rajeev Ahuja**, *Director of Indian Institute of Technology Ropar*, for his all round support, whole-hearted cooperation during the Ph.D journey. I am grateful to the **Head of the Electrical Engineering Department**, for his zealous and encouraging attitude.

I also extend my genuine thanks to doctoral committee members **Prof. Jyotindra Singh Sahambi**, **Dr. Sam Darshi**, and **Dr. Arun Kumar** for their insightful comments and encouragement and also for the hard questions which incented me to widen my research from various perspectives.

My special thanks and appreciation goes to **Dr. Santosh Kumar Vipparthi**, for providing valuable guidance and support.

I also want to specially thank my seniors **Dr. Sachin Chaudhary**, NTU Singapore, **Dr. Prashant Patil**, IIT Guwahati, **Dr. Akshay Dudhane** , MBZUAI Dubai, **Dr. Shruti Phutke Patil**, Griffith University, **Dr. Praful Hambarde**, TECH5, and **Dr. Nancy Mehta**, University of Wuerzburg, for their valuable suggestions during my entire Ph.D. journey.

I wholeheartedly thank my cooperating juniors and colleagues **Mr. Ashutosh Kulkarni**, **Mr. Katta Ranjith**, **Mr. Amitabh Tripathi**, **Mr. Chetan Gupta**, **Mr. Omkar Thawakar**, **Mr. Kelam Sudheer Babu**, **Mr. Mohd. Ubaid Wani**, **Mr. Md Raqib Khan**, **Mr. Siddharth Kumar Singh**, **Mr. Garvit Gopalani**, **Mr. Kushal Chaudhary**, **Mr. Kuldeep Biradar**, lab interns and part-time PhDs for constant encouragement.

I want to sincerely thank my close friends **Mr. Mohit Kamboj**, **Mr. Saminderpreet Singh**, and **Dr. Iyyappan C** for their motivational and emotional support from the beginning of my journey.

Finally, my deep and sincere gratitude to my family for their continuous love, prayers, encouragement, blind trust and for understanding and cooperating me throughout my journey. Above all, I praise and thank God for blessing me with success in this research work.

May 2024.

Place : IIT Ropar, Punjab, India. Jasdeep Singh

**DEPARTMENT OF ELECTRICAL ENGINEERING**
**INDIAN INSTITUTE OF TECHNOLOGY ROPAR**
RUPNAGAR-140001, INDIA

# Certificate

This is to certify that the thesis entitled **MODERN DEEP LEARNING BASED APPROACHES FOR VIDEO MOTION MAGNIFICATION**, submitted by **Jasdeep Singh (2019eez0006)** for the award of the degree of **Doctor of Philosophy** of Indian Institute of Technology Ropar, Punjab, INDIA, is a record of bonafide research work carried out under my guidance and supervision during 2019-24. To the best of my knowledge and belief, the work presented in this thesis is original and has not been submitted, either in part or full, for the award of any other degree, diploma, fellowship, associateship or similar title of any university or institution.

In my opinion, the thesis has reached the standard fulfilling the requirements of the regulations relating to the degree.

Signature

Dr. G Sankara Raju Kosuru
Associate Professor
Department of Mathematics
Indian Institute of Technology Ropar
Rupnagar, Punjab 140001
Date: 21 May 2024

Signature

Dr. Subrahmanyam Murala
Associate Professor
Department of Electrical Engineering
Indian Institute of Technology Ropar
Rupnagar, Punjab 140001
Date: 21 May 2024

# Lay Summary

Video motion magnification is a powerful technique used to amplify subtle movements captured in video footage, making them more prominent and discernible to the human eye. It enhances imperceptible motions, allowing for better analysis and interpretation of dynamic phenomena in various applications. This process involves extracting and amplifying motion signals from video frames, enabling the visualization of movements that would otherwise go unnoticed. Video motion magnification finds wide-ranging applications in fields such as healthcare, where it can aid in diagnosing medical conditions based on subtle physiological cues, as well as in industrial settings for monitoring machinery and detecting anomalies.

However, existing methods encounter significant challenges, including artifacts from manual filters and computational complexities from deep learning approaches. These obstacles hinder the accurate extraction of subtle movements from videos.

In this work, our aim is to address these challenges and develop effective motion magnification techniques that enable more reliable detection of subtle movements while reducing artifacts and computational burden. All proposed approaches incorporate novel state-of-the-art concepts, and our solution is qualitatively and quantitatively demonstrated to achieve superior results compared to previous state-of-the-art methods.

# Abstract

Unveiling imperceptible motions in videos is a critical task with applications ranging from industrial monitoring to healthcare diagnostics. However, State-Of-The-Art (SOTA) methods in video motion magnification face challenges that impact their effectiveness. The primary identified problem lies in the trade-offs of existing techniques. Hand-crafted bandpass filter-based approaches suffer from the need for prior information, ringing artifacts, and limited magnification. Conversely, deep learning-based methods, while achieving higher magnification, introduce issues like artificially induced motion, distortions, and computational complexity, making them unsuitable for real-time applications.

To overcome these challenges, we propose a comprehensive solution. Our first approach involves a novel deep learning-based lightweight model for motion magnification. Leveraging feature sharing and an appearance encoder, our method enhances motion magnification while minimizing artifacts.

Addressing the broader computational complexity challenge associated with SOTA methods, we introduce a Knowledge Distillation-Based Latency-Aware Differentiable Architecture Search method (KL-DNAS). Instead of designing architecture by hand, we let the network decide the best possible architecture under the given constraints. We use a teacher network to search the network by parts using knowledge distillation. Further, search among different receptive fields and multi-feature connections, are applied for individual layers. Also, we use a novel latency loss is proposed to jointly optimize the target latency constraint and output quality.

In the realm of magnifying small motions prone to noise and disturbances, we identify a need for a balanced solution, which can exploit both deep learning and hand-crafted based approaches. Introducing a phase-based deep network operating in both frequency and spatial domains, we generate motion magnification from frequency domain phase fluctuations and refine it spatially. With lightweight models a balance between magnification and computational efficiency is achieved, as evidenced by comparative evaluations against SOTA methods.

However, this integration doesn't fully utilize the steerable pyramid architecture of hand-crafted based methods as it manipulate motion features in single scale. To further enhance it, we integrate traditional techniques with deep learning. The proposed $\sigma - \theta Net$ model effectively combines handcrafted intuition of complex steerable pyramid with deep learning mechanisms, significantly improving motion magnification performance.

Additionally, in response to the sensitivity of video motion magnification to noise-related distortions, we propose a hierarchical magnification network. It produces a more robust performance with a multi-scale manipulator and a novel contrastive learning-based loss. This approach effectively mitigates distortions caused by noise and illumination changes

while enhancing texture quality. It maintains a lightweight design, while maintaining its effectiveness.

***Keywords:*** Video Motion Magnification, Knowledge Distillation (KD), Latency Aware, Motion Manipulation, Neural Architecture Search (NAS), Contrastive Learning, Lightweight networks.

# Abbreviations

| | |
|---|---|
| CNN | : Convolutional Neural Network |
| GFLOPs | : Giga Floating Point Operations |
| MAE | : Mean Absolute Error |
| MSE | : Mean Square Error |
| SSIM | : Structural Similarity Index |
| SOTA | : State-Of-The-Art |
| FA | : Fractional Anisotropy |
| INR | : Implicit Neural Representation |
| STBVMM | : Swin Transformer Based Video Motion Magnification |
| KD | : Knowledge Distillation |
| NAS | : Neural Architecture Search |
| LWVMM | : LightWeight Network for Video Motion Magnification |

# Contents

# List of Figures

# Chapter 1

# Introduction

Within this chapter, we introduce the topic of video motion magnification. The introduction to video motion magnification is given in Section 1.1 and the motivation for this work is discussed in Section 1.2. Section 1.3 introduces industrial and healthcare applications of video motion magnification. The research challenges in the field of video motion magnification are discussed in Section 1.4. In Section 1.5, the main aim and objectives of our research work are discussed. Section 1.6 defines the major contributions of our work. Finally, Section 1.7 provides the overall structure and outline of the thesis.

## 1.1 Introduction

Real-life scenarios often entail subtle changes that remain imperceptible to the naked eye. These minute variations encompass a wide spectrum, ranging from the barely detectable muscle movements of athletes to the delicate expansion and contraction of the chest during breathing, or even the subtle deformation of objects. However, the challenge lies in discerning these subtle motions from the inherent noise levels present in the environment. Despite their proximity to noise, these subtle vibration signals hold immense potential for various applications across diverse domains.

The potential applications stemming from these subtle changes are extensive and diverse, spanning various fields and industries. In industries such as rotating machinery, robotics, and structural engineering, the ability to detect and analyze minute vibrations is crucial for ensuring optimal performance, safety, and reliability. For instance, researchers have utilized vibration analysis techniques in rotating machinery [1, 2, 3] and robotic devices [4] to enhance performance and identify potential faults. Moreover, in fields like civil engineering, monitoring traffic-excited bridges [5] and structures or vehicles with variable mass [6] necessitates precise detection and analysis of subtle structural changes. Similarly, in aerospace engineering, the ability to monitor geometry-variable aerospace structures is essential for ensuring structural integrity and aerodynamic efficiency [7]. Additionally, in healthcare, the measurement of vital signs through subtle body motions holds promise for non-invasive monitoring and early detection of health conditions [8, 9].

To address the challenge of distinguishing these subtle motions from noise and enhancing their visibility, video motion magnification-based algorithms have emerged as powerful

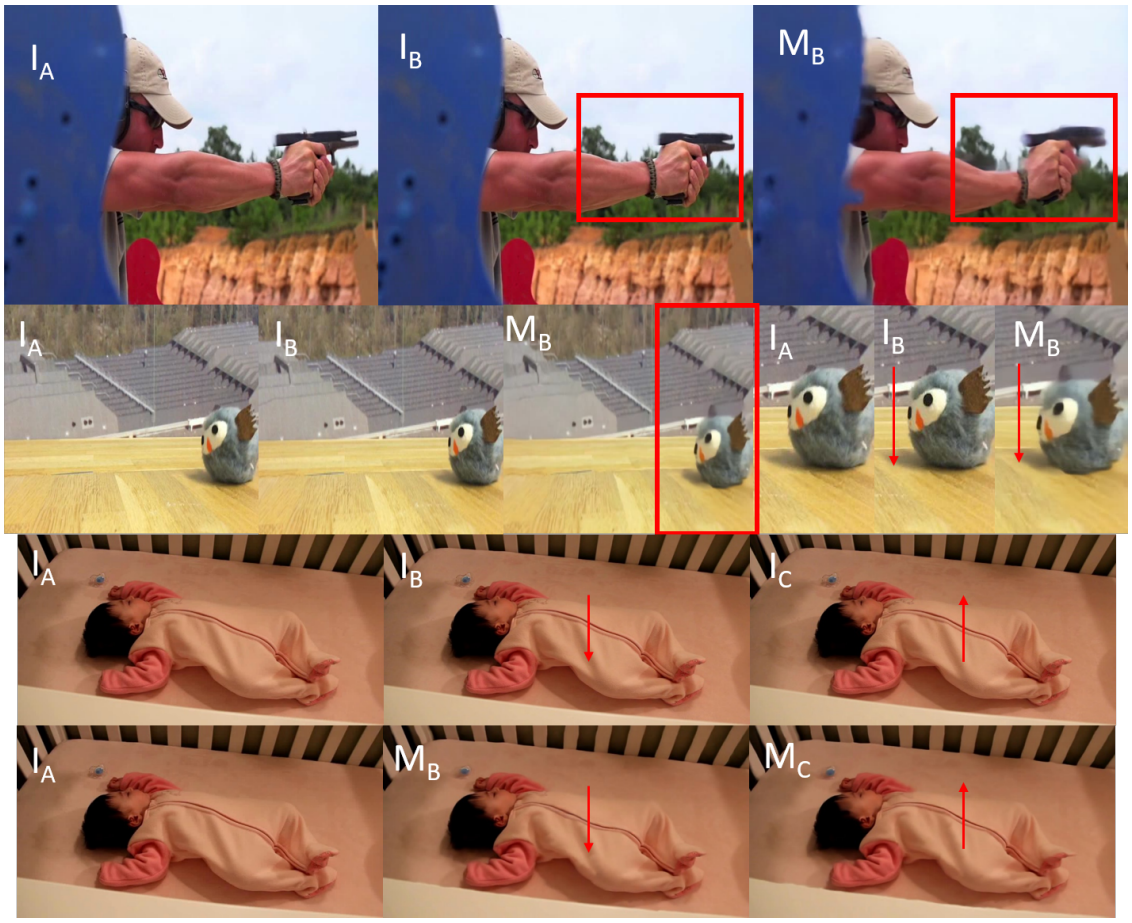


Figure 1.1: (1) The first row displays a video of a gun, while (2) the second and (3) the third row depict videos of a toy and a baby, respectively. The gun and toy videos exhibit dynamic motion, whereas the baby video depicts static motion. $I_A$, $I_B$, and $I_C$ represent the input frames, while $M_B$ and $M_C$ illustrate the magnified motion. In (1), the gun video highlights the magnified complex motion of gun recoil, emphasized within the red bounding box. In (2), the motion of the toy video is indicated by a red arrow, and a zoomed-in version of the bounding box is provided. In (3), the baby video [27]showcases subtle motion around the chest, with red arrows indicating the direction of the motion.

tools. These algorithms leverage advanced signal processing techniques to amplify subtle changes captured in video footage, thereby making them clearly visible to the human eye. By enhancing the visibility of these subtle dynamics, motion magnification techniques offer invaluable insights and enable more accurate analysis and interpretation of real-world phenomena.

## 1.2 Motivation and Applications

Magnification of subtle motions has many different computer vision applications [10, 11, 8, 12]. Motion magnification is used in micro-expression recognition [13, 14, 15, 16, 17]. These micro-expressions are generally present in $1/30^{th}$ of video frames [13]. Analysis of psychological disorders or inherent feelings of a person can be recognized with

micro-expressions. Hence, motion magnification can facilitate effective recognition of micro-expressions and identification of inherent feelings. Fan *et. al.* [18] proposed the use of motion magnification to separate pulsatile motion from non-pulsatile motion, which helps them to locate vessels and neurovascular bundles in endoscopic prostatectomy videos. This will help in endoscopic and robotic surgery.

Davis *et al.* [19] used video motion magnification to extract speech by measuring micro-displacements in the objects (which are generated due to sound vibrations). Dorkenwald *et al.* [20] utilized video motion magnification to measure posture deviation between a healthy and impaired patient. Popek *et al.* [21, 22] employed video motion magnification to extract mechanically induced micro-displacements. Kyrollos *et al.* [23] measured respiratory rate from video motion magnification. Kim *et al.* [11] applied motion magnification for the measurement of vibrations in the rails. Zhu *et al.* [24] used motion magnification in measuring the location of the partial discharge in power cables. Hence video motion magnification has significant importance in measuring subtle displacement signals related to computer vision and biomedical signal/image processing problems.

Traditionally, measuring vibrations has relied on contact-based sensors like accelerometers or non-contact laser vibrometers. However, attaching these sensors to small or lightweight objects can introduce mass-loading effects and alter the object's dynamics. Additionally, for large structures, the requirement of multiple sensors can escalate costs and labor. Motion magnification offers an alternative by amplifying vibration signals captured in videos, thereby circumventing the limitations associated with traditional sensor-based approaches [21, 22, 25]. This makes it a versatile and cost-effective solution for measuring subtle displacement signals in both computer vision and biomedical signal/image processing domains.

## 1.3 Problem Formulation

Wu *et al.* and Wadhwa *et al.* [26, 27] define the problem of motion magnification as follows. For an image $I(x,t)$, it can be expressed as a function of motion such that $I(x,t) = f(x + \delta(x,t))$, where $f(x) = I(x,0)$ and $\delta(x,t)$ is a function of motion field with respect to spatial co-ordinates $x$ at a time $t$. If video frames follow this assumption, then the magnified frame can be expressed as,

$$I(x,t) = f(x + (1+\alpha)\delta(x,t)) \tag{1.1}$$

Magnification factor $\alpha$ decides the amount of magnification. We avoid magnification of all the motion signals in $\delta(x,t)$ as it also contains non-meaningful motion signals like subtle camera shake, photographic noise *etc.* Let's take $B(.)$ as an ideal signal selector that can differentiate between meaningful and non-meaningful motions. Then the motion magnified

frame $\tilde{I}(x,t)$ is defined as

$$\tilde{I}(x,t) = f(x + (1+\alpha)B(\delta(x,t))) \tag{1.2}$$

Many different proposed methods try to approximate $B(.)$ *e.g.* [26, 27, 28] by using temporal filters as a signal selector. We are using deep learning based techniques to overcome the limitations of hand crafted methods for solving this problem. Our proposed method learns the function $\tilde{B}(.)$ as the network tries to approximate $\tilde{I}(x,t)$ directly. It can be expressed as the following

$$\tilde{I}(x,t) \approx f_a(x + M_f * \tilde{B}(\delta(x,t))) \tag{1.3}$$

In our approach, we simplify the architecture by providing two input frames $f_a$ and $f_b$, along with a magnification factor $M_f$. This simplification not only streamlines the network complexity but also reduces the number of parameters involved.

## 1.4 Research Challenges

Motion magnification poses several significant research challenges, which include addressing the following issues:

1. Ringing Artifacts in Output: Ringing artifacts, are unwanted oscillations that occur in the output of motion magnification algorithms, particularly Eulerian-based methods. These artifacts can distort the magnified motion, leading to inaccuracies in the representation of subtle movements. They often appear as spurious oscillations or halos around edges or regions of motion, in the magnified video [29, 27, 26, 28, 30, 31, 32]. To address the issue of ringing artifacts, researchers explore deep-learning based methods [29], and proposed a supervised learning based training mechanism.

2. Noise Amplification: Noise amplification poses a significant challenge in motion magnification, particularly in deep learning-based methods. Noise, whether originating from sensor noise, compression artifacts, or environmental factors, can be inadvertently magnified along with the desired motion signal during the amplification process. This results in the presence of spurious artifacts and false motion signals, compromising the accuracy and reliability of the magnified output. Effectively addressing noise amplification requires the development of noise reduction techniques [32] or the integration of noise-aware processing mechanisms into motion magnification algorithms such as being done in deep-learning-based methods [29].

3. Dependency on Training Data: Deep learning-based unsupervised methods for

motion magnification often exhibit a high degree of dependency on training data. These methods rely on large datasets of annotated video sequences to learn the complex relationships between input frames and their corresponding magnified outputs. However, the effectiveness of these models is contingent upon the similarity between the training data and the testing scenarios [20]. In cases where the testing scenarios deviate significantly from the training distribution, the performance of the model may degrade, leading to suboptimal results. Addressing the dependency on training data requires the development of robust and generalizable models that can effectively adapt to diverse real-world conditions without requiring extensive retraining or fine-tuning.

4. Computational Complexity: Computational complexity presents a significant challenge in motion magnification, particularly in methods that involve dense optical flow estimation, spatial-temporal filtering, and deep learning-based approaches. Dense optical flow estimation calculates motion vectors for every pixel in consecutive frames, requiring intensive computation. Spatial-temporal filtering involves analyzing both spatial and temporal dimensions of the video data, further adding to the computational load. Deep learning methods, while offering good capabilities for motion magnification, often consist of a large number of parameters. Additionally, different applications of motion magnification may have varying latency requirements. For example, in medical imaging or surveillance applications, where real-time analysis and response are critical, low latency is essential. On the other hand, in post-processing applications such as video enhancement for film production, latency may be less of a concern.

5. Small Magnification in Dynamic Scenarios: Handcrafted methods in motion magnification often face limitations in dynamic scenarios, resulting in small magnification of subtle motions [27, 26, 28, 30, 31, 32]. These methods typically rely on predefined filters and processing techniques, which may struggle to effectively amplify motion in environments with rapid changes or complex dynamics. Consequently, the magnification achieved in dynamic scenarios using handcrafted methods may be insufficient for capturing and highlighting subtle motions of interest. Addressing this challenge requires the development of more adaptive and responsive techniques that can accurately amplify motion across a wide range of dynamic scenarios.

6. Distortions Generated by Deep Learning-Based Supervised Method: Deep learning-based supervised training method, as demonstrated by the work of Oh et al. [29], gives better magnification but may introduce various distortions in the magnified output. While these methods offer significant potential for enhancing

motion visibility, they are not without limitations.

- They extract motion information from shape information to enhance the network's robustness to intensity changes. However, their approach to separating shape information from texture may not always be efficient, leading to distorted intermediate features that can result in unwanted flickering or spurious motion.
- Texture features extracted by this method may sometimes deviate significantly from the input textures, potentially causing blurry distortions in certain frames of the magnified video.

## 1.5 Problem Statement

From the above observations, we have identified the following problems for video motion magnification:

1. Texture distortions during magnification degrade the magnified output quality and distort the motion.
2. Limited adaptability in terms of computational complexity across various applications hampers the usability of video motion magnification techniques.
3. Real-time processing constraints impede practical deployment, urging the development of efficient algorithms and frameworks.
4. Noise interferes with the signal of interest and affects the motion magnified output.

## 1.6 Aims and Objectives

From the identified problems in existing video motion magnification methods, we define the aim and objective of our work as:
**Aim:** *To propose novel solutions for resolving different problems in video motion magnification.*
**Objectives:**

1. To design a novel approach for video motion magnification aimed at reducing texture distortions.
2. To introduce a novel approach for generating application-specific computational complex tailored models for video motion magnification.
3. To present a novel lightweight network for video motion magnification.

4. To propose a novel network to enhance noise robustness in video motion magnification.

## 1.7 Main Contributions

This study is focused on deep learning based architectures for video motion magnification task. The major contributions of this work are listed below:

- A proxy model based regularization loss with feature sharing encoder and appearance encoder is proposed to reduce the magnification of noise and other unwanted changes in motion features.
- A novel neural architecture search based approach for generating application-specific computational models for video motion magnification is proposed.
- A Multi Domain learning based efficient light-weight network for video motion magnification is proposed.
- A Learnable Directional Scale Space Filters based network aimed at enabling steerable direction space for video motion magnification.
- A novel Hierarchical network with multi-scale manipulator and novel contrastive loss is proposed for enhancing robustness against the noise-generated subtle motion for video motion magnification architectures.

## 1.8 Thesis Structure

- **Chapter 1: Introduction and Motivation**
  - This chapter introduces video motion magnification and provides the motivation behind the present work.
  - It serves as the preface of the entire thesis, setting the stage for the subsequent chapters.
- **Chapter 2: Literature Review**
  - This chapter discusses existing methods for video motion magnification and their applications.
- **Chapter 3: Proxy Model Based Training for Texture Distortion Reduction**
  - This chapter presents the proposed proxy model-based training approach aimed at reducing texture distortions in motion magnification.

- **Chapter 4: Neural Architecture Search for Application-Specific Models**
  - This chapter introduces a solution based on a neural architecture search for generating application-specific computational models tailored for video motion magnification.
- **Chapter 5: Multi-Domain Lightweight Network for Real-Time Motion Magnification**
  - This chapter discusses a Multi-Domain based lightweight network aimed at achieving real-time motion magnification.
- **Chapter 6: Learnable Directional Scale Space Filters for Motion Magnification**
  - This chapter discusses a Learnable Directional Scale Space Filters based lightweight network aimed at directional scale-space for video motion magnification.
- **Chapter 7: Hierarchical Magnification Network for Noise Robustness**
  - This chapter delves into the proposed Hierarchical Magnification Network designed to enhance the robustness of motion magnification networks to noise.
- **Chapter 8: Conclusion and Future Scope**
  - This chapter summarizes the findings and conclusions drawn from the thesis work.
  - It also explores potential future directions to further improve the performance of video motion magnification and its applicability in various high-level computer vision tasks.

# Chapter 2

# Literature Survey

In this chapter, existing hand-crafted and deep learning based approaches in the field of video motion magnification are discussed.

## 2.1 Existing Approaches for Motion Magnification

The field of motion magnification has witnessed significant advancements with the introduction of multiple methods aimed at highlighting small changes within videos. These advancements reflect the evolving landscape of motion magnification techniques, showcasing diverse approaches to address the complexities inherent in this area of research [33, 31, 32, 26, 28, 27, 30, 34, 29, 20].

Broadly categorized into Eulerian and Lagrangian methods, these approaches offer distinct strategies for motion magnification. Eulerian methods predominantly focus on the utilization of hand-designed spatiotemporal filters, leveraging the inherent properties of pixel-level changes across frames [32, 31, 26, 28, 27, 30]. This approach involves intricate processing of spatial and temporal information to identify and amplify specific motion characteristics within the video sequence. In contrast, Lagrangian methods rely on optical flow techniques to track the movement of objects or features across frames [33]. By computing the displacement of pixels between consecutive frames, Lagrangian approaches enable the isolation and magnification of motion-related features.

Recent advancements in Eulerian-based methods have led to the development of techniques capable of producing promising results across various scenarios, even in the presence of significant motion [31, 30, 32]. These methods have demonstrated effectiveness in enhancing subtle motion cues while maintaining robustness in challenging environments.

## 2.2 Lagrangian approach

Liu *et al.* [33] proposed a Lagrangian based method, for motion magnification. It matches the feature point across different frames to compute optical flow. The optical flow was used to segment the background and motion of interest. Background motions were first spatially registered and then the motion of interest was re-estimated and magnified. But computing optical flow in this method is costly [35].

Pan *et al.* [36] proposed a model that takes a pair of video frames and a magnification factor as input and generates a new pair of frames with the predicted optical flow magnified

according to the specified factor. The model is trained using self-supervised learning, meaning it can be trained solely on real unlabeled videos without requiring ground truth annotations. During training, the optical flow estimator is used within the loss function, but it is not required during testing. The simplicity and self-supervised nature of this model offer several advantages over traditional approaches. The model can be easily adapted at test time for improved performance on specific input videos and can also be extended to magnify the motion of individual objects specified by user-provided segmentation maps. But, it also has limitations. Firstly, the effectiveness of the model heavily relies on the quality and accuracy of the off-the-shelf optical flow networks used for supervision. If the optical flow estimation is inaccurate or noisy, it may result in suboptimal magnification results. Additionally, while the model can be fine-tuned for improved performance on specific input videos, this process may require additional manual intervention and the model is computationally complex, limiting its applicability in real-world scenarios.

## 2.3 Eulerian approach

### 2.3.1 Hand-Crafted methods

Eulerian-based methods offer a distinctive approach to motion magnification by eschewing explicit object tracking in favor of filter-based techniques. Initially, Wu et al. [27], demonstrate that Eulerian spatio-temporal processing of standard monocular video sequences can reveal nearly imperceptible changes in dynamic environments without explicit motion estimation. The study provides an analysis of the relationship between temporal filtering and spatial motion. Additionally, it presents a unified framework capable of amplifying both spatial motion and purely temporal changes (e.g., heart pulse) and allows adjustment for specific temporal frequencies, a capability not supported by Lagrangian methods. In their proposed method, first, they extract input frames from a video and decompose them spatially using a multi-scale Laplacian pyramid. Subsequently, a temporal filter is applied to capture changes across pixel values over different frames. By defining a specific frequency range, these methods selectively amplify the desired motion. However their output is prone to large distortions due to the magnification of noise.

To solve this issue, Wadhwa et al. [26] proposed a phase variations based motion magnification using complex steerable pyramids. It maintains an intricate balance between the compactness of the transform representation and the magnitude of magnification achievable in octave and sub-octave bandwidth pyramids. Moreover, it showcases the capability to enhance extracted low-amplitude motion signals through spatial denoising of phase signals within individual image subbands. This refinement process leads to superior results in motion processing. However, their method cannot magnify color changes, works

poorly in the presence of large motions, and is computationally complex [26].

To solve computational complexity later Wadhwa et al. [28] proposed the Riesz pyramid for Eulerian phase-based video magnification, offering comparable quality to complex steerable pyramids but with reduced overcompleteness, enabling faster processing. Constructed using an efficient Laplacian replacement and approximate Riesz transform, it avoids spatial artifacts by operating entirely in the spatial domain. Efficient implementation is achieved through shared computation, filter symmetry, and minimal real multiplies. The Riesz transform's property as a steerable Hilbert transformer allows precise phase-shifting, enhancing motion magnification. This novel representation supports real-time processing and offers advantages over existing methods, making it promising for video magnification applications. However, it does not handle dynamic scenarios and produces blurry distortions in the presence of dynamic motion.

Addressing the challenge of large motion, Zhang et al. [30] proposed a solution that magnifies only non-linear motion under the assumption that large motion alters linearly while small changes vary non-linearly. By disregarding linear variations, Zhang et al. [30] successfully magnified subtle changes even in the presence of significant motion. However, in scenarios involving fast-moving backgrounds or large non-linear motions, their approach tends to blur the output.

Takeda et al. [31] enhance video magnification by isolating subtle changes amidst slow and quick large motions. Leveraging jerk, which evaluates the smoothness of time series data, the method distinguishes between subtle changes and quick large motions based on their differing smoothness characteristics. A jerk-aware filter is developed to selectively pass subtle changes while suppressing artifacts caused by large motions. The method effectively integrates this filter into the acceleration method, yielding good magnification results. Nonetheless, a limitation of their approach is the inability to distinguish between quick large motions and subtle quick motions.

Despite the advancements achieved by these methods, they face challenges in distinguishing between non-meaningful subtle motions, such as photographic noise, from meaningful ones [32]. This limitation underscores the need for further refinement in motion magnification techniques to enhance their ability to discern and amplify only the desired motions accurately while suppressing irrelevant disturbances.

Different methods were suggested to improve the selectivity of the motion of interest. Elgharib *et. al.* [37] introduced DVMAG (Dynamic Video Motion Magnification), aimed at amplifying small motions within larger ones in video sequences. DVMAG employs a layer-based approach, decomposing images into foreground, background, and an opacity matte, facilitating the magnification of subtle motions while preserving surrounding areas. The technique handles large motions by stabilizing the region of interest (ROI) before magnification, reducing common artifacts observed in other methods. Additionally,

DVMAG utilizes matting to focus on the ROI and texture synthesis to fill in any resulting holes, ensuring high-quality magnified video output.

Whereas Verma *et. al.* [38] proposed amplifying subtle motions within videos, with a specific focus on objects of interest. It utilizes a combination of techniques such as object extraction, alpha matting, and Eulerian motion magnification to achieve its objectives. Object extraction involves employing a kernel K-means approach to segregate the object from the initial frame. Alpha matting utilizes automatic scribble drawing with superpixels and Bezier curves [39] to generate a mask with a fuzzy boundary for the object, facilitating motion magnification. Eulerian Motion Magnification is then applied to enhance imperceptible motions within the video, particularly within the area of interest. Together, these steps enable the magnification of motions that may not be readily discernible to the human eye, rendering the method valuable for analyzing videos with subtle movements.

Kooij et al. (2016) integrate depth information into their process, enabling selective amplification of movements within specific depth ranges and enhancing resilience against occlusions and large motions across different depths. Additionally, they propose an extension to the bilateral filter, crucial for effectively handling non-Gaussian filters and treating pixels at various depth layers, minimizing potential inaccuracies. Integrating depth information within the bilateral filter helps magnify tiny changes in the same depth layer while ignoring motion in other areas. However, it's essential to note that depth sensors require specific environments to operate optimally, and without user input, these methods may struggle to differentiate between meaningful and non-meaningful subtle motion

Other proposed solutions try to avoid human intervention. Verma *et al.* [40] suggested the use of Fast Local Laplacian filter to generate edge-aware motion magnification while reducing the amount of noise added to the output. In another method proposed by Wu *et al.* [41], principal component analysis (PCA) was used to decompose the input frames. After PCA decomposition, the component that has the most resemblance with the small changes was taken. It was able to remove noise in the output but requires useful motion to be more as compared to other miniature changes.

Takeda *et al.* [32] method incorporates Fractional Anisotropy (FA), drawn from neuroscience, to differentiate between meaningful subtle changes and non-meaningful photographic noise. FA measures anisotropic diffusion, emphasizing meaningful changes over noise. Utilizing FA, [32] designs a Fractional Anisotropic Filter, selectively passing meaningful subtle changes while disregarding noise, a significant improvement over previous methods. Additionally, it introduces edge-ware regularization to refine motion information, preserving edge details and mitigating noise amplification. However, their assumption of the isotropic nature of noise does hold in different videos and produces

magnification of noise.

Later, Takeda *et al.* [42] proposed Bilateral Video Magnification Filter (BVMF) method which enhances frequency selectivity by utilizing new formulations to ensure that the Laplacian of Gaussian (LoG) parameters are linked to the passband characteristics, resulting in a peak gain unity at the target frequency, thus improving upon existing methods. Additionally, BVMF achieves the exclusion of large motions by introducing a Gaussian kernel that filters the intensity of the phase signal, excluding motions outside the desired magnitude without making assumptions about motion dynamics. This approach is simpler and more resilient compared to previous methods. Lastly, BVMF implements the bilateral principle by applying two kernels simultaneously in the temporal and intensity domains, akin to bilateral filters in spatial and intensity domains. However, these methods use hand-crafted filters which are not optimum [29]. They are prone to ringing artifacts, small magnification in the presence of large motion, and also magnify subtle camera shakes. They require fine tuning of hyperparameters from video to video basis.

### 2.3.2 Deep Learning based approaches

Deep learning-based proposed solutions generate large magnifications. Oh *et al.* [29] proposed a method with the aim to magnify the motion signal $\delta(x,t)$ in input frames $I(x,t)$ by a factor $\alpha$ to obtain magnified frames $\hat{I}(x,t)$. To simplify training, they focus on a two-frame input scenario, estimating the magnified frame from two input frames with slight motion displacement. The network architecture consists of three main components: encoder, manipulator, and decoder. The encoder extracts spatial representations, including a shape representation capturing motion ($M$) and a texture representation capturing intensity information ($V$). The manipulator magnifies motion by scaling the difference between the shape representations of the two input frames by $\alpha$. The decoder reconstructs the magnified output frame using the manipulated shape representation and the original texture representation.

Due to the challenge of obtaining real motion-magnified video pairs, synthetic data is used. Synthetic data offers the advantage of scalability, but generating realistic small motions requires careful consideration. The dataset is designed to ensure generalizability to real-world scenarios. It combines foreground objects from the PASCAL VOC [43] dataset with background images from the MS COCO dataset [44], simulating occlusion effects by pasting objects onto backgrounds. Each training sample contains 7 to 15 randomly scaled foreground objects overlaid onto backgrounds to simulate occlusion effects. Scaling factors are limited to 2 to prevent blurry textures and motion direction and amount are randomized for effective learning of local motions. The magnification factor ($\alpha$) is capped at 100 and input motions are sampled within a range of up to 10 pixels to prevent magnified motion exceeding 30 pixels. To address issues of poor generalization

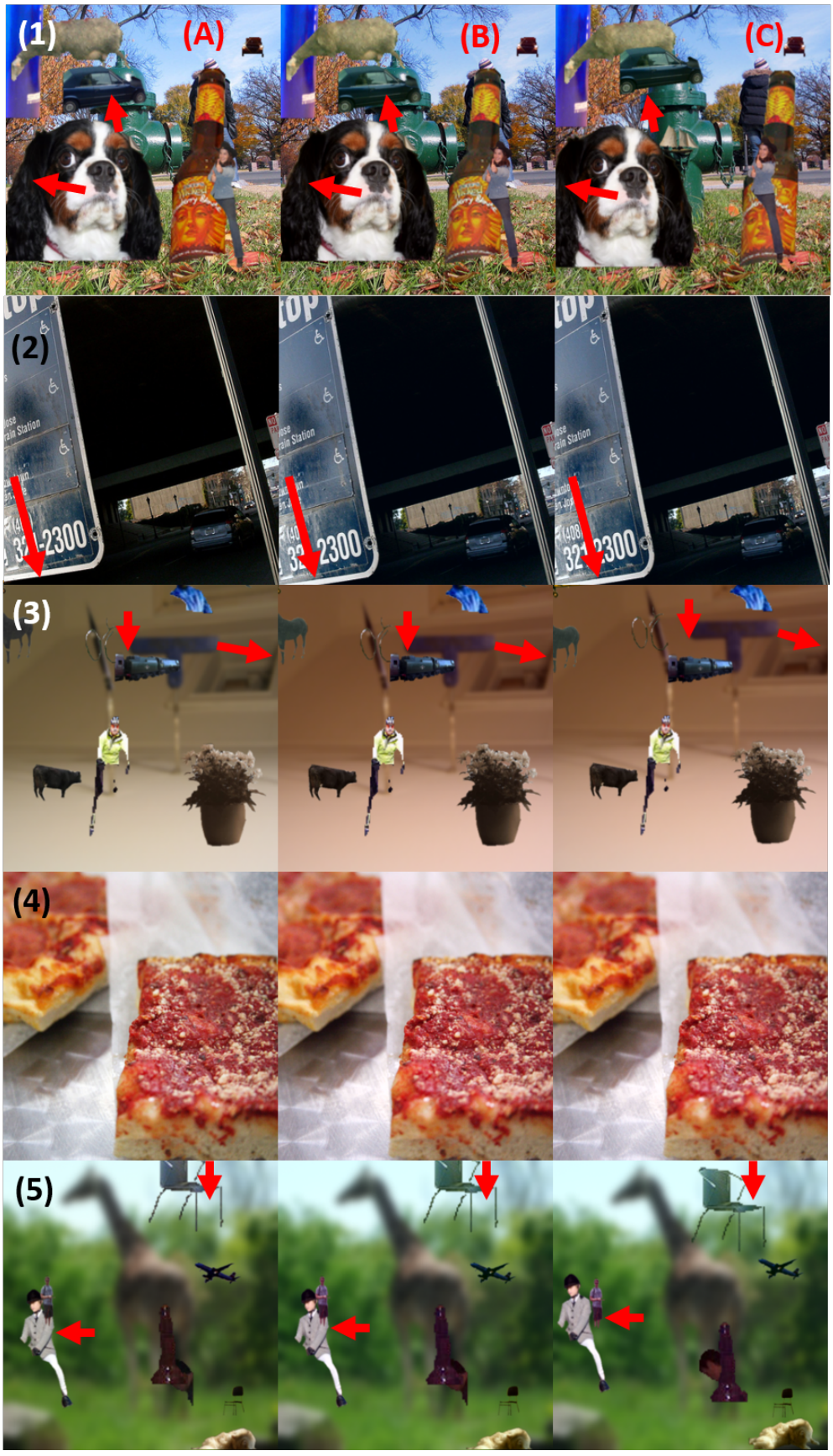


Figure 2.1: Column A and B represent the input frames $t-1$ and $t$, respectively, while column C represents the magnified frame. Each row corresponds to a different scenario: (1) Static scene with object motion, (2) only moving background, (3) Foreground and background motion, (4) without any motion, and (5) Blurred background with foreground motion.

on low-contrast textures, examples with blurred backgrounds or only moving backgrounds are included. Additionally, subsets with static scenes are incorporated to help the network learn changes due to noise. The dataset comprises five parts, each with 20,000 samples of 384 $\times$ 384 images. Parameters such as magnification factor and input motion are carefully controlled to ensure learnability, with limited magnification factors to avoid excessive motion. Subpixel motion is also incorporated into the process. This includes reconstructing motion in frames prior to downsampling, and bicubic interpolation is used for downsampling to minimize errors. To address rounding errors during quantization, uniform quantization noise is introduced to each pixel, ensuring proportional rounding based on its residual value.

The network is trained in an end-to-end manner on synthetic two-frame data using $L_1$ losses on the output, along with regularizers to separate shape and texture, enabling effective motion magnification while maintaining realistic visual quality, even in the presence of large motion. However, it exhibits artifacts and relies on temporal filters for quality enhancement, which, when applied, result in patchy magnification and ignore very small motion. Additionally, the method's approach to extracting motion information from shape information lacks efficiency, occasionally leading to distorted intermediate features and unwanted flickering or superfluous motion. Texture features may also deviate significantly from input textures, potentially causing blurry distortions. Moreover, computational complexity considerations are lacking, which is crucial for real-time applications requiring low latency.

Dorkenwald *et al.* [20] proposes an unsupervised method to magnify subtle posture variations across subjects in videos. The approach, embedded in an autoencoder framework, separates posture from appearance, enabling transfer across subjects while maintaining appearance fidelity. A novel loss function enforces effective disentanglement of posture and appearance despite appearance variations. The method integrates magnification directly into the autoencoder training, utilizing real and synthesized data without ground truth supervision. It employs two separate encoders to encode posture and appearance, disentangles them through color transformations, and introduces a disentanglement loss for meaningful representations. Magnification is achieved by extrapolating posture encodings and combining them with appearance encodings to generate realistic magnified images. The model is trained end-to-end, leveraging synthesized magnifications, showcasing effectiveness in magnifying subtle posture deviations while preserving appearance across various datasets. However, it requires a model to train on similar scenarios as a test video for motion magnification.

Chen *et al.* [34] introduces an "DeepMag" end-to-end deep learning framework for video magnification, focusing on amplifying subtle color and motion signals from specific sources, even amidst significant motions. [34] developed a novel deep neural network

architecture based on gradient ascent and tailored it for automated magnification of target signals like pulse and respiration in videos. Further, they adapt gradient ascent for video magnification, leveraging a convolutional neural network (CNN) as a motion discriminator, and integrating $L_1$ normalization and sign correction for consistent and accurate magnification based on the input magnification signal. DeepMag derives the magnification signal by utilizing the $L_2$ norm of the target signal's first-order derivative and subsequently denoising the resulting motion representation. But the magnification produced by this method is small and it is specific to spacial videos [34].

The MagFormer method [45] presents an end-to-end video motion magnification network featuring a two-branch module combining Eulerian and Lagrangian perspectives. The Eulerian branch employs a convolutional neural network (CNN) to capture global motion features from a camera-centered viewpoint, while the Lagrangian branch utilizes a Transformer to capture local motion trajectories from an object-centered perspective. Fusion blocks facilitate inter-branch interaction in a layer-by-layer fashion. Additionally, a motion-guided attention module highlights motion areas and reduces artifacts, with motion features separated from texture features by a dedicated feature separator. Optical flow extraction is conducted using GMFlow [46], and a reconstruction module combines texture and magnified motion features to generate the final output frames. This approach integrates both global and local motion information, employs attention mechanisms, and enables interaction between branches. But, it has a computationally complex pipeline.

The STB-VMM (Swin Transformer Based Video Motion Magnification) [47] method proposes a novel architecture for video motion magnification, leveraging the Swin Transformer. Its components include a shallow feature extractor for early local feature extraction, a deep feature extractor comprising N Residual Swin Transformer Blocks (RSTB), a manipulator that magnifies motion between frames by multiplying the difference in feature spaces by a user-defined factor$\alpha$, a Mixed Magnified Transformer Block (MMTB) for coherent output processing, and a reconstructor to generate the magnified frame from the processed feature space. The pipeline involves passing two input frames through feature extractors, manipulating the feature space, processing it through MMTB, and reconstructing the frame. The model is trained end-to-end on synthetic data using $L_1$ loss and regularization loss. It results in better performance but the method is computationally heavy.

Feng *et al.* [48] introduces a method for amplifying nuanced 3D movements in scenes filmed with mobile cameras, while enabling innovative view rendering. Notably, previous motion magnification techniques were confined to 2D videos captured by stationary cameras. This approach extends the capability to dynamic 3D environments by utilizing time-varying neural radiance fields (NeRF) to represent scenes. The method applies the Eulerian principle to enhance motion by magnifying variations in point embeddings over time.

Also, it employs a small MLP to induce phase shifts in positional encoding functions, and utilizes tri-plane learnable embedding functions for NeRF.

Meyer *et al.* [49] introduces Phase-NIVR, a novel neural implicit representation method designed for videos, allowing manipulation of temporal dynamics. It extends the concept of image-based implicit neural representation (INR) to videos by incorporating phase-shift information into the Fourier-based positional encoding, facilitating the modeling of video sequences. The model architecture comprises two main components: a Frame Generation Module responsible for generating video frames from phase-shifted positional encodings, and a Phase-Shift Generation Module that predicts phase-shift vectors for each time step. Through end-to-end training on video data, the model learns to associate phase-shift values with the mapping to video frames. Notably, the learned phase-shift vectors capture meaningful temporal dynamics, enabling various motion manipulation effects such as temporal interpolation, motion filtering, magnitude magnification, and video loop detection. But Nerf is data-specific and requires fine-tuning for each video, which makes it less suitable for real-time application.

# Chapter 3

# Proxy Model Based Training for Texture Distortion Reduction

To address the problem of motion magnification initially, hand-design based approaches were introduced. Many SOTA hand-crafted methods were based on temporal filters which gave good results [27], [26], [28] on static scenarios but they cannot work in dynamic scenarios. To mitigate this, later [30], [31] methods were proposed which can work in both static and dynamic scenarios. But their outputs were prone to ringing artifacts or small magnification *etc.* Also, their filters were not optimal [29]. To solve these issues of hand-crafted filters, the deep learning-based method [29] was proposed. Even without temporal filters, it shows some robustness to noise and produces higher magnification without ringing artifacts. But it has some limitations.

- They extract motion information from shape information to make the network robust to intensity changes. But, their separation of shape information from texture, is not efficient. Sometimes it results in distorted intermediate features which produce unwanted flickering or superious motion.
- Their texture features sometimes deviate much from input textures and this might be responsible for blurry distortions in some frames.
- They did not take computational complexity into account. As real-time applications like respiration rate monitoring, or in industries where time-constrained output is needed, require low latency.

Currently deep learning based approaches in different tasks like deraining, deblurring, object detection [50],[51], [52] *etc* show promise for real-time applications. Inspired by this we propose a lightweight network for video motion magnification. Our proposed lightweight method does not produce unwanted distortions like [29] and is sensitive toward subtle motions. It produces more magnification than SOTA methods in both static and dynamic scenarios. It has a simple yet efficient architecture. Further, different experiments are done to show the qualitative, and quantitative analysis, and physical accuracy of the proposed method in comparison to SOTA methods. The main contributions of the proposed work are as follows:

- A lightweight deep learning model is proposed for video motion magnification.

Table 3.1: Comparison with prior methods

| **Methods** | Hand-crafted Methods | Oh *et al.* [29] | Proposed Model |
|---|---|---|---|
| Magnification in Static Scenarios | High | High | High |
| Magnification in Dynamic Scenarios or in presence of large motion | Low | High | High |
| Noise Reduction mechanism to reduce distortions related to magnification | Narrow bandpass filters | Separate shape and Texture information | Feature sharing encoder, Common appearance encoder and Proxy model based training |

- A feature sharing encoder module is proposed for motion magnification. This module is responsible for appropriate feature map generations for motion extraction and for reducing the effect of the noise before magnification.
- An appearance encoder is proposed to extract common appearance across the frames with its output being restricted by input frames. This module is responsible for the appropriate texture synthesis of the output.
- A proxy model based regularization loss is proposed to reduce the magnification of noise and other unwanted changes in motion features.

These solutions are explained in detail in the following sections.

## 3.1 Network Architecture

We propose a lightweight deep learning based network to magnify the subtle motions in the videos. It consists of encoder-decoder based architecture. It uses two feature sharing based encoders, to translate input frames from image space to feature space where motion information can be extracted. Handcrafted methods [26], [28], [30], [31] use complex steerable pyramids for the same task. Oh *et al.* [29] uses simple encoders and gives its features to shape encoders to extract shape features. It extracts motion information from the shape features. Separating shape information from image features is done using regularization across the encoders to constrain the feature space. Instead of that, we let the network decide the encoding feature space for motion extraction.

A major issue with motion magnification is to reduce the effects of changes due to noise, illumination *etc* while magnifying meaningful changes. This is a hard problem. Hand-crafted methods [26], [28], [30], [31] depend on narrow band pass filter (which require prior information about the frequency of interest). Oh *et al.* [29] method presumes that noise, unwanted illumination *etc* changes are part of intensity changes

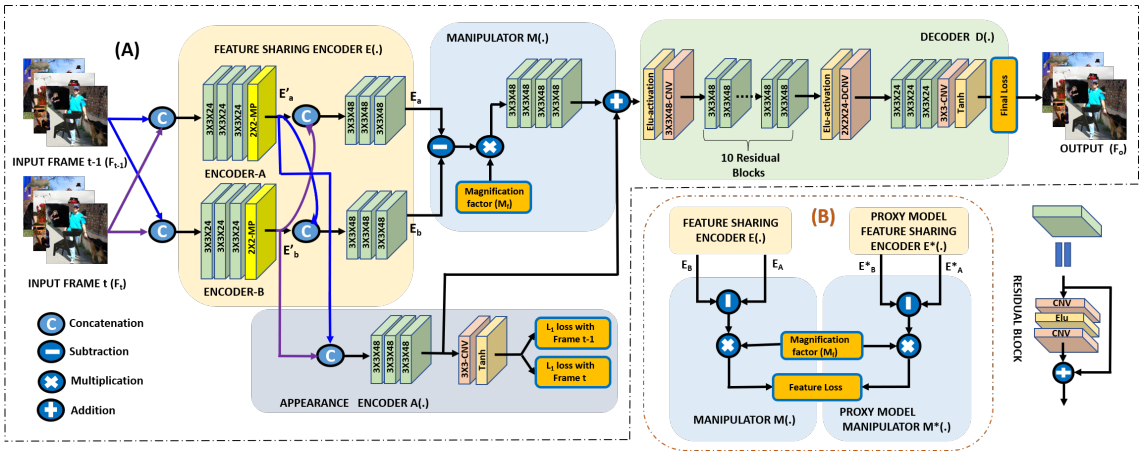


Figure 3.1: **(A)** Proposed deep learning model for motion magnification. It consists of a feature sharing encoder, appearance encoder, manipulator, and decoder. $F_t$ and $F_{t-1}$ the two consecutive frames, with $M_f$ as the magnification factor, are given as input to the network. $F_o$ is the magnified output frame. Residual Blocks with 3×3×48 show that there is a 3 × 3 convolution filter with 48 channels, similarly for Residual Blocks with 3×3×24. $E_a$ and $E_b$ are the output features and $E'_a$ and $E'_b$ are the intermediate features of the feature sharing encoder. **(B)** Proxy model feature loss across the manipulator block. *Please zoom in for a clearer view.*

and motion information is present in shape changes. So, they try to separate shape from texture representation (intensity information). For this, while training the network they provide intensity perturbed frames that have the same shape information as un-perturbed frames. Then they take $L_1$ loss across perturbed and un-perturbed frames features. They assume that shape information across intensity change should remain the same. They take the difference between these shape features, magnify it, and add it to the texture encoder features, but their method is not efficient. It sometimes results in distorted intermediate features which produce flickering or superious motion. The proposed method uses feature sharing encoder for the motion extraction and proxy model based feature loss with appearance encoder loss to reduce the effects of noise before magnification. The denoising signal in network training comes from three different places 1) from the final predicted output, 2) common appearance based regularization loss 3) proxy model based feature loss. Jointly optimizing across these losses helps to reduce the effects of noise in motion magnification (a detailed discussion is given in the section below). The manipulator multiplies the motion features to the magnification factor (which decides the amount of magnification), and applies non-linear transforms using residual blocks. The manipulator output is added to the common appearance encoder output and given to the decoder. The decoder converts intermediate features to image space and generates the final magnified output. Figure 3.1 (A) describes the proposed model.

**Feature Sharing Encoder ($E(.)$):** Feature Sharing Encoder is used to reduce the effect of noise before magnification (decoder is used to reduce the effect of noise after magnification). We assume different frames will have distinct noise. With concatenation operation across features, each encoder will have information about the input frames and improved features of the other encoder. The network can compute weighted averages to decrease the effects of illumination, noise *etc.* It's also used to convert the input from image space to feature space for motion extraction. Unlike [29], its output features ($E_a, E_b$) are not restricted by regularization. Residual blocks [53] are used to map input frames to a feature space where motion information is extracted by taking the feature differences as shown in Figure 3.2. Max-pooling is used to down-sample the features to reduce the computation and increase the receptive field. The feature sharing encoder is illustrated in Figure 3.1 (A).

**Appearance Encoder ($A(.)$):** Relevant texture content is required to combine with motion information to generate the magnified frame. For generating texture content, [29] proposes a regularization term to minimize the difference in texture feature representation between the frames. To satisfy this regularization term both texture encoders with different inputs try to generate a common representation, but this representation can deviate from the actual texture representation. We assume this can be the probable reason for producing texture distortion (blurry distortions) sometimes. To solve this, we propose an Appearance Encoder ($A(.)$). Generally, the magnified frame has a high correlation with the input frames as most of the objects are still. In $A(.)$ we exploited this fact for appropriate texture generation. Loss between appearance encoder $A(.)$ features and input frames are used to extract common appearance features. This also prevents the learnable parameters from generating features that deviate from $F_t$ and $F_{t-1}$. For calculating this loss, no noise is added to the ground truth (input frames). So, it will also force denoising characteristics in common texture features. This will help in the better generation of the output. Both encoder intermediate features $E'_a$ and $E'_b$ (as shown in Figure 3.1 (A), as the output of both encoders) are concatenated ($\zeta$ represents the concatenation operation) and is given as input to the appearance encoder. Then residual blocks are applied to them for feature transformation to produce output $A(\zeta(E'_a, E'_b))$. The regularization loss $L_A$ between input frames $F_t$, $F_{t-1}$ and appearance encoder output $A(\zeta(E'_a, E'_b))$ is defined in Eq. (3.1)

$$L_A = |\phi(A(\zeta(E'_a, E'_b))) - F_t)|_1 + |\phi(A(\zeta(E'_a, E'_b))) - F_{t-1})|_1 \tag{3.1}$$

where $\phi$ represents the convolution operation with $3 \times 3 \times 3$ filters and tanh activation.

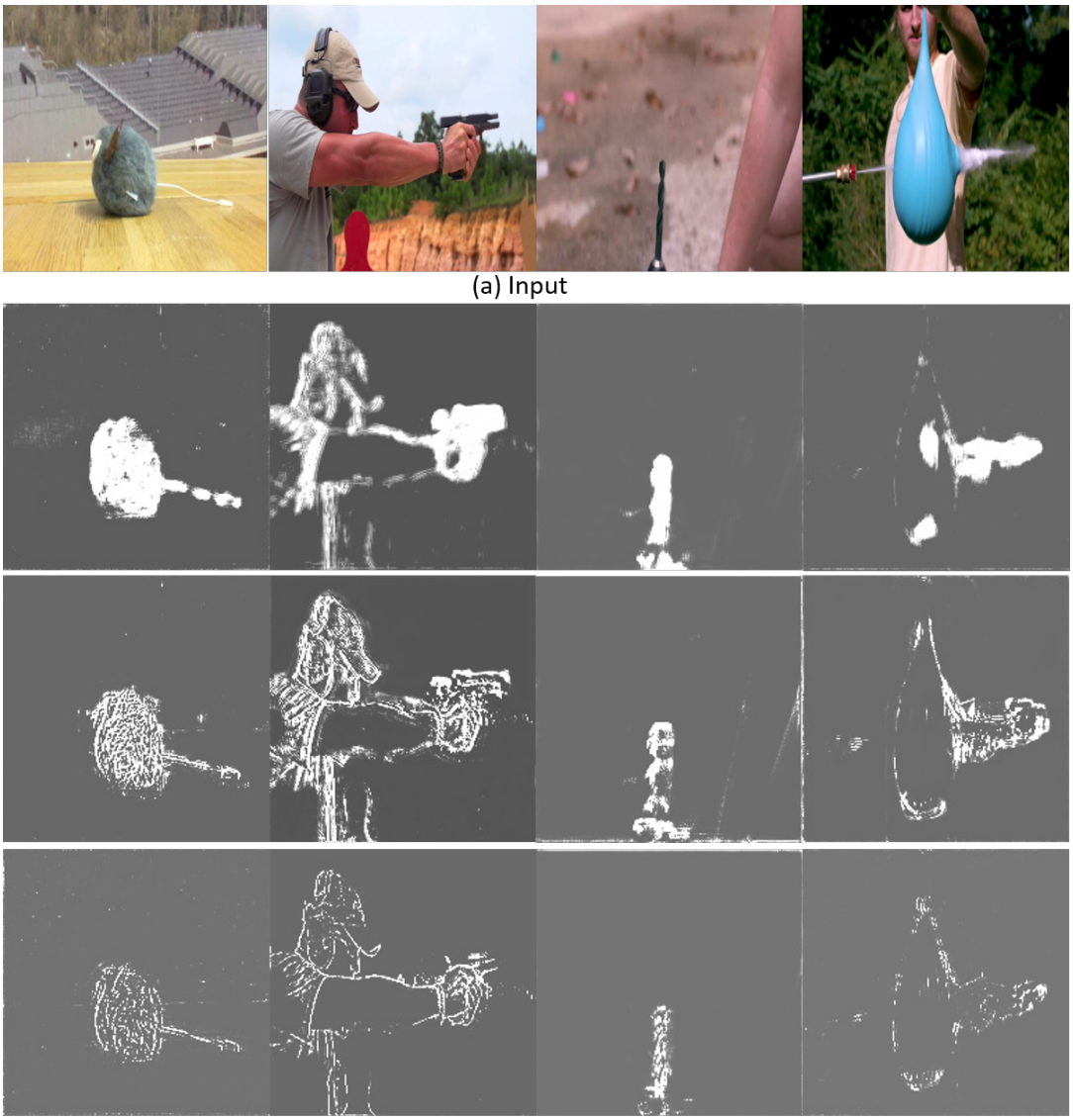


Figure 3.2: (a) depicts the input frames, (b) shows the motion features (after subtraction of encoder features). These features highlight the object of motion.

**Manipulator ($M(.)$):** We assume motion information can be extracted from the difference in encoder features. This is somewhat different from [29] assumption, where they presume motion information can be extracted from the difference of encoder shape features. The manipulator ($M$) gets the non-linear transformed encoder shared features of $E_a$ and $E_b$ as input. It takes their difference and multiplies them with the magnification factor $M_f$. Then these features are given to residual blocks for non-linear transformations to generate output $M((E_a - E_b) \times M_f)$ (the structure of manipulator is similar to [29]). Figure 3.2 shows the different features of the feature sharing encoder block that highlight the motion information.

**Decoder:** The combined output of the appearance encoder and manipulator is given to the decoder as shown in Figure 3.2. In the decoder, ten residual blocks before up-sampling are used, as they decrease the computation requirements and increase the receptive fields. The up-sampled features are passed through three residual blocks. In the end, a convolution layer with $3 \times 3$ filter size and tanh activation is used to generate the magnified output $F_o$ (the structure of the decoder is similar to [29]).

**Proxy Model Based Feature Loss:** The proxy model has the same architecture as the proposed model but it is trained without adding noise. The proxy model features (of the noiseless image) are taken as the ideal features and the main model features (with noisy input) should try to get close to it (It can also be viewed as teacher-student training paradigm [54], [55], [56], where the teacher has the same network as a student, but teacher network is used to pass denoising information to the student network in the feature space). At the time of training of the main model, proxy model weights are fixed. For calculating the distance between proxy model noiseless features and main model noisy features $L_1$ loss is used. This feature space loss is only sensitive to noise present in the motion information. The appearance encoder loss term is sensitive to noise present in texture, and predicted output loss terms are sensitive to the magnified noise (particularly which can cause large variations after magnification). So, proxy model based feature loss will help to make motion information more robust. Loss is taken in between the manipulator features after subtraction and multiplication with magnification factor $((E_a - E_b) \times M_f)$ as shown in Figure 3.1 (B). We assume that this will help to prevent any distortions that can be generated due to magnification of noise, illumination changes *etc.* Proxy model based feature loss can be defined as follows:

$$L_M = |((E_a^* - E_b^*) \times M_f) - ((E_a - E_b) \times M_f)|_1 \tag{3.2}$$

where superscript notation $*$, indicates the proxy model.

**Final Loss Function:** We consider the $L_1$ loss, loss between edges ($L_{edge}$) and Perceptual Loss ($L_p$) for bettering of output quality. The $L_1$ loss computes the pixel level difference of predicted label $\hat{y}$ and ground truth $y$. $L_1$ loss is illustrated as

$$L_1 = \sum |\hat{y} - y|_1 \tag{3.3}$$

In the motion magnification problem, the $L_1$ loss is less sensitive to object motion because most of the region in output frames does not have motion. Further, there may exist many minima in $L_1$ which produce blur output [57] around the motion parts (near the edges). So, to put more focus on the edges of the output, we take the loss between the edges of

Table 3.2: Comparison of the SOTA learning method [29] with the proposed base network ($M_1$) and the lightweight network ($M_2$) in terms of number of parameters, FLOPS, and run time. (Run time values are calculated at 720X720 resolution on NVIDIA 2080 RTX for higher quality output).

| **Model** | **Parameters** | **GFLOPs** | **Run Time** |
|---|---|---|---|
| Oh *et al.* [29] | 0.98M | 268.6 | 95 ms |
| $M_1$ | 1.10M | 375.5 | 142 ms |
| $M_2$ | 0.12 M | 42.4 | 38 ms |

the predicted and ground truth frames ($L_{edge}$), (as defined in [58] ). $L_{edge}$, helps to make the model more sensitive towards the edges [58] of the reconstructed motion magnified frames. $L_{edge}$ is given as

$$L_{edge} = \sum |\nabla \widehat{y} - \nabla y|_1 \tag{3.4}$$

$\nabla$ shows the finite differences in a horizontal and vertical direction [58] for computing edges. Another issue with the texture of the moving object is that there still exist many minima which can give low loss but with bad perceptual quality. For this, a loss in a higher dimension is needed. Hence, to increase the perceptual quality of the motion magnified frames, we use the perceptual loss ($L_p$) [59] along with the $L_1$ and $L_{edge}$. The $L_p$ is given as

$$L_p = \sum |\phi_i(\widehat{y}) - \phi_i(y)|_1 \tag{3.5}$$

Where, $\phi_i$ represents the VGG-16 [60] feature space activations. The final loss of the proposed network ($L_{total}$) is given in Eq. (7.4)

$$L_{total} = \lambda_1 L_1 + \lambda_2 L_p + L_{edge} + L_A + L_M \tag{3.6}$$

Where $\lambda_1$ and $\lambda_2$ are the weights for $L_1$ loss and Perceptual Loss ($L_p$) [59] respectively. $\lambda_1 = 10.0$, and $\lambda_2 = 0.1$ values are considered for the network training and they are determined experimentally.

**Dataset and Training:** The proposed models, base model, and lightweight model are trained on the training dataset provided by [29]. In the network, $C$ channels are used in primary layers, and after down-sampling $C \times 2$ channels. For the base model, $C = 24$, and for the lightweight model $C = 8$ is considered. For training, the learning rate is set to .0001, and an ADAM optimizer is used. Models are trained for 47 epochs. The proposed lightweight model has 7.6 $\times$ lesser parameters and 6.3 $\times$ lesser flops as compared to [29] as shown in Table 7.1.

Table 3.3: Parameters used for result generation. All the results are generated with variables and steps given by the respective authors. The source code and pre-trained model are downloaded from their official page.

| Methods | Video | $M_f$ | Frequency |
|---|---|---|---|
| Ours $(M_1, M_2)$ | Cat toy | 15 | N/A |
| Ours $(M_1, M_2)$ | Gun | 15 | N/A |
| Ours $(M_1, M_2)$ | Drill | 10 | N/A |
| Ours $(M_1, M_2)$ | Balloon | 10 | N/A |
| Ours $(M_1, M_2)$ | baby | 20 | N/A |
| Ours $(M_1, M_2)$ | guitar | 4 | N/A |
| Ours $(M_1, M_2)$ | Physical Accuracy | 10 | N/A |
| Ours$(M_1, M_2)$ | synthetic videos | 50 | N/A |
| Oh et al [29] | Cat toy | 10 | N/A |
| Oh et al [29] | Gun | 10 | N/A |
| Oh et al [29] | Drill | 10 | N/A |
| Oh et al [29] | Balloon | 10 | N/A |
| Oh et al [29] | baby | 20 | 2.5 Hz |
| Oh et al [29] | synthetic videos | 60 | N/A |
| Jerk-Aware [31] | Cat toy | 10 | 3 |
| Jerk-Aware [31] | Gun | 10 | 20 |
| Jerk-Aware [31] | Drill | 25 | 3 |
| Jerk-Aware [31] | Balloon | 25 | 3 |
| Jerk-Aware [31] | baby | 50 | 2.5 |
| Jerk-Aware [31] | synthetic videos | 200 | 15 |
| Anisotropy [32] | Cat toy | 100 | 3 |
| Anisotropy [32] | Gun | 100 | 20 |
| Anisotropy [32] | Drill | 100 | 3 |
| Anisotropy [32] | Balloon | 100 | 3 |
| Anisotropy [32] | Physical Accuracy | 200 | 3 |
| Anisotropy [32] | synthetic videos | 400 | 15 |
| Acceleration [30] | Cat toy | 4 | 3 |
| Acceleration [30] | Gun | 10 | 20 |
| Acceleration [30] | Drill | 4 | 3 |
| Acceleration [30] | Balloon | 4 | 3 |
| Acceleration [30] | Physical Accuracy | 20 | 15 |
| Acceleration [30] | synthetic videos | 200 | 15 |

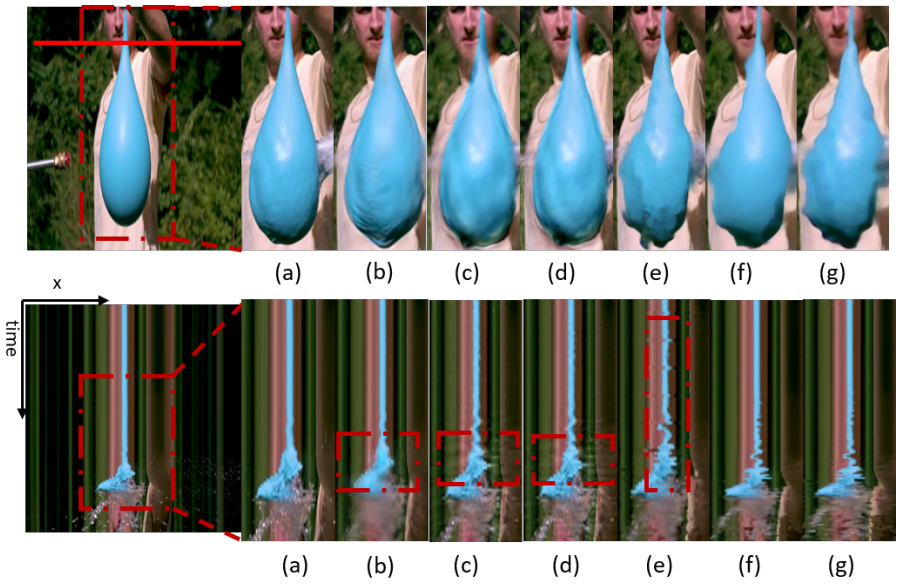


Figure 3.3: Balloon video: First frames from the video are shown and next to them temporal slices taken from the red strip are illustrated for visualization of balloon burst motion. Motion magnification can be perceived as more motion in the balloon (also visible in the temporal slice) as compared to the input. While the other methods produce distortions such as ringing artifacts, spurious motion *etc* (highlighted in the red box). The proposed method produces better magnification with fewer distortions. (a) Input video, (b) Acceleration based method [30], (c) Jerk-Aware method [31], (d) Anisotropy method [32], (e) Oh *et al.* [29], (f) Ours Base model, and (g) Our lightweight model.

## 3.2 Experimental Results

The proposed model is evaluated qualitatively and quantitatively on real-life and synthetic videos and is compared with the SOTA methods [31], [29], [30], [32] for motion magnification (linear filter based method [26] is not considered for comparisons as they produce distortions in dynamic scenarios). The details of parameters used for result generation are shown in Table 6.3. Also, an ablation study is conducted to show different aspects of the proposed method. With the least computational complexity, the proposed lightweight model provides better results than SOTA methods. The detailed discussion is given in the following subsections.

### 3.2.1 Analysis on Real Videos

**Analysis on Balloon video:** In the balloon video, a water cannon is fired on a balloon to rupture it, as shown in Figure 3.3. Due to this, small motions are developed in the balloon along with its large bursting motion. Our aim is to magnify the minute balloon motion while producing minimum distortions due to sudden large motion. Figure 3.3 shows the motion of the balloon at the red strip over time. Hand-crafted methods [31, 32, 30] create ringing artifacts along the balloon (*visible as white edges near the balloon and white*

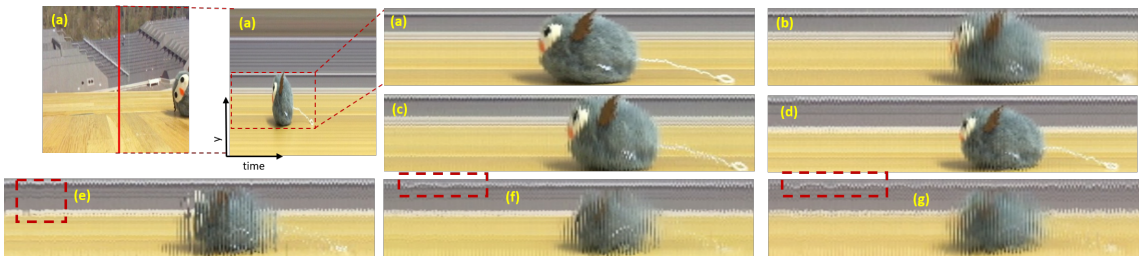


Figure 3.4: A toy is vibrating and moving along the table from right to left. The spatial-temporal slices from the respective methods are taken from the red strip. The proposed method shows more magnification (also higher motion of the background is highlighted in the red bounding boxes). (a) Input video, (b) Acceleration based method [30], (c) Jerk-Aware method [31], (d) Anisotropy method [32], (e) Oh *et al.* [29], (f) Ours Base model, and (g) Ours lightweight model.

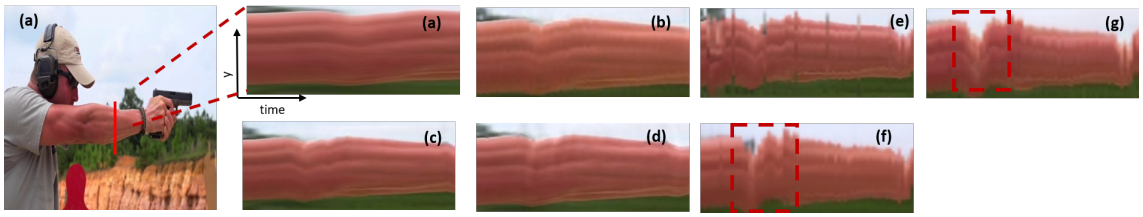


Figure 3.5: Gun-shooting video: Visualizing the impact of gun recoil through the arm. We take temporal slices at red strip to show the effect of magnification on the forearm. The proposed method output has the highest magnification (shown as more bending of the forearm in the red box). (a) Input video, (b) Acceleration based method [30], (c) Jerk-Aware method [31], (d) Anisotropy method [32], (e) Oh *et al.* [29], (f) Ours Base model, and (g) Our lightweight model.

*spikes in the temporal slices highlighted in the red boxes, in Figure 3.3* ). Further, Oh *et al.* [29] produce blurry distortions in some frames (in the balloon and the background object), visible as spikes in the temporal slice (*illustrated in red bounding box in Figure 3.3 temporal slice).* The proposed method shows better magnification with lesser distortions around the balloon.

**Analysis on Toy Video:** The toy video is illustrated in Figure 6.5. In this video, the toy is moving on the table along with vibrations. Our goal is to produce large magnification for the toy's subtle motions in the presence of the toy linear motion (moving along the table from left to right). The Jerk-aware [31], Acceleration [30] and Anisotropy [32] methods produce less magnification. Further, the Acceleration [30] and Oh *et al.* [29] produce some blurriness in the output. Oh *et al.* [29] method produces good magnification but causes spurious motion *(visible in red box as sharp spikes in Figure 6.5 (e)).* Our proposed models produce better magnification of the vibrating toy as compared to [31], [30], [32], [29].

**Analysis on Gun-shooting Video:** Figure 7.6 shows the results of different SOTA methods on gun-shooting video. This video contains a large background movement due

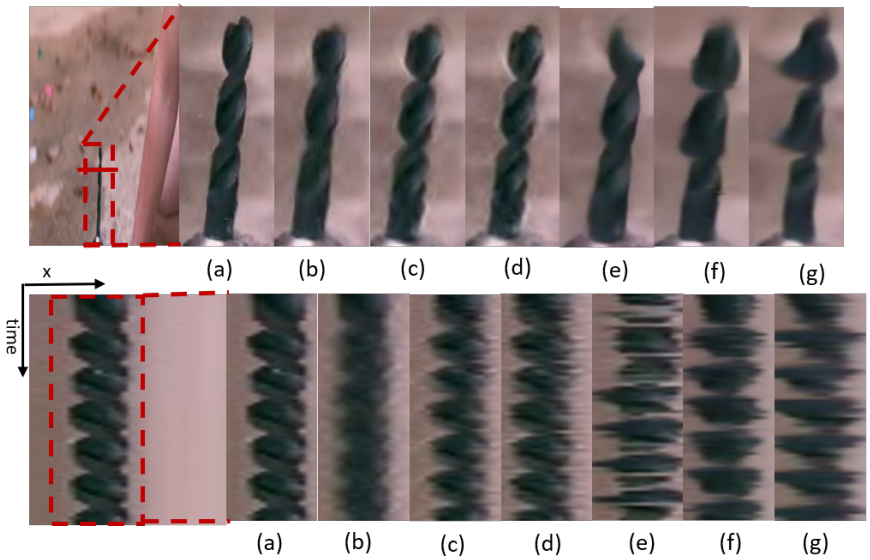


Figure 3.6: Drill Video: Comparison of proposed method with existing methods for magnification of the drill rotational motion. First, output from respective methods and then their spatio-temporal slices with respect to the red strip are shown. The proposed method produces better results with fewer artifacts. (a) Input video, (b) Acceleration based method [30], (c) Jerk-Aware method [31], (d) Anisotropy method [32], (e) Oh *et al.* [29], (f) Ours Base model, and (g) Ours lightweight model.

to camera motion and quick gun recoil produces the foreground motion. Our aim is to magnify the minute forearm motion in the presence of a large camera motion. Figure 7.6 shows the motion of the forearm using spatio-temporal slices at a red strip. Higher forearm motion can be perceived as more bending in the temporal slice (shown in the red box of Figure 7.6). Jerk-aware method [31], Anisotropy [32], Acceleration [30] methods produce lower magnification as compared to the proposed method. Oh *et al.* method [29] induce spurious motion in some frames and generate blurry distortions (*visible as large spikes in Oh et al. [29] temporal slice).* The proposed method generates higher magnification of subtle forearm movements with fewer distortions, even in the presence of large camera motion as compared to SOTA methods.

**Analysis on rotational motion:** Figure 3.6 illustrates a hand drill producing rotational motion along its axis. To analyze the effects of magnification on rotational motion a still video is taken. In 2D, hand drill rotational motion can be perceived as spiral motion. Our aim is to increase the spiral motion (higher spiral motion is displayed as more outwards extension of rod radius). The rotational motion of the hand drill is depicted in spatial temporal slice of Figure 3.6. Hand design filter-based methods [31, 32, 30] generate ringing artifacts around the rod *(seen as white edges near the rod and white spikes in the temporal slices in Figure 3.6 (b),(c),(d)).* Oh *et al.* method [29] magnifies the motion but delivers some distortions in the magnified frames (observable as white spikes in Figure 3.6(e) temporal slice). Our proposed models have better magnification and fewer artifacts

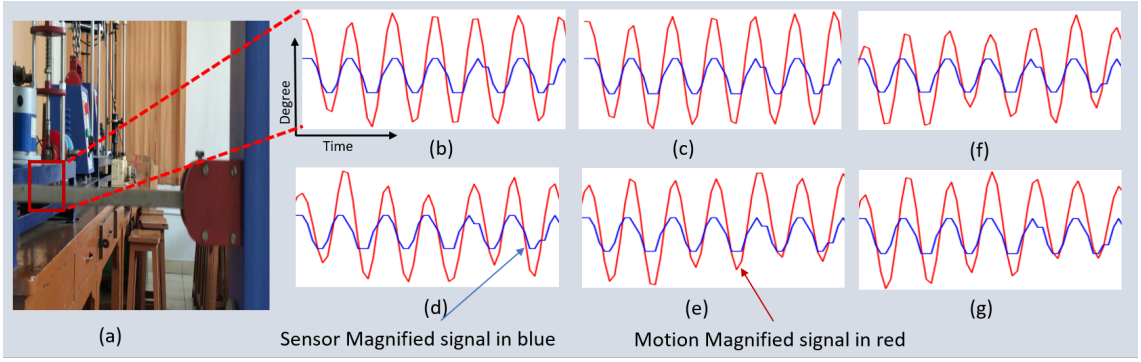


Figure 3.7: Physical Accuracy: Comparison between our method and other SOTA methods output (in red) with the sensor signal (in blue) respectively. The direction of optical flow in the patch region is computed to extract the magnified signal (in blue) from the video. (a) Input, (b) Our base model (c) Our lightweight model, (d) Oh *et al.* method [29] (e) Jerk-aware method [31], (f) Acceleration method [30] and (g) Anisotropy [32] method respectively.

Table 3.4: Mean Absolute Error (MAE) on SOTA methods of Anisotropy [32], Jerk-aware method [31], Acceleration method [30], Oh *et al.* method [29], Ours base method ($M_1$), and Ours lightweight model ($M_2$). MAE is computed between the extracted signal from the magnified video and sensor measured signal. The proposed method has the minimum MAE values. (First best shown in bold and second best shown in italics.)

| **Methods** | [32] | [31] | [30] | [29] | $M_1$ | $M_2$ |
|---|---|---|---|---|---|---|
| **MAE** | 0.146 | 0.149 | 0.146 | 0.144 | **0.121** | *0.131* |

in motion as compared to SOTA methods.

### 3.2.2 Physical Accuracy

**Whether our magnified output is physically accurate?** To verify the accuracy of our approach, we conducted an experiment with the setup depicted in Figure 3.7. We utilized a universal vibration apparatus to induce subtle up-and-down motions in a mechanical rod, while simultaneously measuring the resulting signal using an ultrasonic sensor. The rod vibrated at approximately 0.11 Hz with dimensions of 25.4 (W) x 12.7 (H) x 840 (L) mm. The video was captured using an iPhone X mounted on a tripod at a resolution of 1080p, positioned 1 meter high and 2 meters away from the beam. An HC-SR04 ultrasonic sensor connected to an Arduino captured motion signals at a sampling rate of 30 Hz while recording the video. To extract motion signals from the video, we computed optical flow across the magnified video, using frame $t-1$ as the reference frame and frame $t$ as the current frame within the region marked by the red bounding box in Figure 3.7. We then calculated the average direction of motion within the image patch. Both optical flow and sensor signals were rescaled from 0 to 1, and using these rescaled signals, we calculated the mean absolute error (MAE) for different state-of-the-art (SOTA) methods. Our proposed method exhibited the lowest MAE,


Figure 3.8: Different backgrounds used for generation of different synthetic videos for quantitative analysis. Video 1-10


Figure 3.9: Different backgrounds used for generation of different synthetic videos for quantitative analysis. Video 11-20.

confirming its superior accuracy.

### 3.2.3 Analysis on Synthetic Videos

To examine the proposed method's effectiveness in various scenarios, synthetic videos are used. 25 synthetic videos with distinct backgrounds are created as shown in Figure 3.8, 3.9, 3.10. Circles of radius 40 pixels are utilized to simulate subtle movements. When generating a sequence of input and ground truth frame pairs for motion magnification, it is generally simpler to work with single motions (such as up-down or left-right movement of a circle) compared to more complex motions (like random movement of the circle). Therefore, in the setup, three circles with movement in horizontal, vertical, and diagonal directions are used. The generated subtle movements are described as $A_{subtle} * (-1)^j$, such


Figure 3.10: Different backgrounds used for generation of different synthetic videos for quantitative analysis. Video 20-25.

Table 3.5: Aggregate Mean Square Error (MSE) of synthetic videos with different backgrounds on Acceleration method [30], Jerk-aware method [31], Anisotropy [32], Oh *et al.* method [29] Ours base method ($M_1$), and our lightweight model ($M_2$) respectively. The proposed method has the minimum error. (First best shown in bold and second best shown in italics.)

| **Methods** | [32] | [31] | [30] | [29] | $M_1$ | $M_2$ |
|---|---|---|---|---|---|---|
| **MSE** | 36.4 | 55.3 | 68.0 | 38.8 | **23.07** | *27.8* |

that $A_{subtle} = 0.1$ pixel and $A_{subtle} = 10$ pixel values are used for input and ground truth movement generation, and $j$ represents the frame number. To produce sub-pixel motion, the input frame is up-sampled by a scaling factor $S_f$ and the object is moved 1 pixel. So, when the image is down-sampled, the motion becomes $1/S_f$ pixels. Gaussian noise is used to mimic photographic noise. For different methods, we use different magnification factors to ensure they produce the same motion as the ground truth. This allowed us to examine and compare how different methods [31], [29], [32] and [30] magnify various motions in distinct environments and their robustness. Each method needs to deliver 100× magnification analogized to the input video to approximate the ground truth. With such a large magnification, the effects of distortions become more apparent and lead to degraded output. For different methods, to generate the same amount of output motion, their magnification factor is changed. Table 3.5 depicts the average MSE of 25 different synthetic videos, on different SOTA methods [31], [29], [32], [30] and ours. The individual values across each video is shown in Figure 3.12. Our method produces better results with minimum aggregate MSE.

**Noise Test, Sub-pixel Motion Test and effects of change in magnification factor:** The tests evaluate the impact of noise variation and sub-pixel motion reduction on video analysis. In Sub-pixel motion tests, the magnification factor is adjusted to maintain consistent magnification levels across methods. The noise test assesses the impact of increasing noise levels (sigma) in the input. Mean square error (MSE) is then computed for predicted output against ground truth across 25 videos as shown in Figure 5.11. Additionally, the effects of changing the magnification factor on various methods are examined, with each method interpreting the magnification factor differently. The chosen

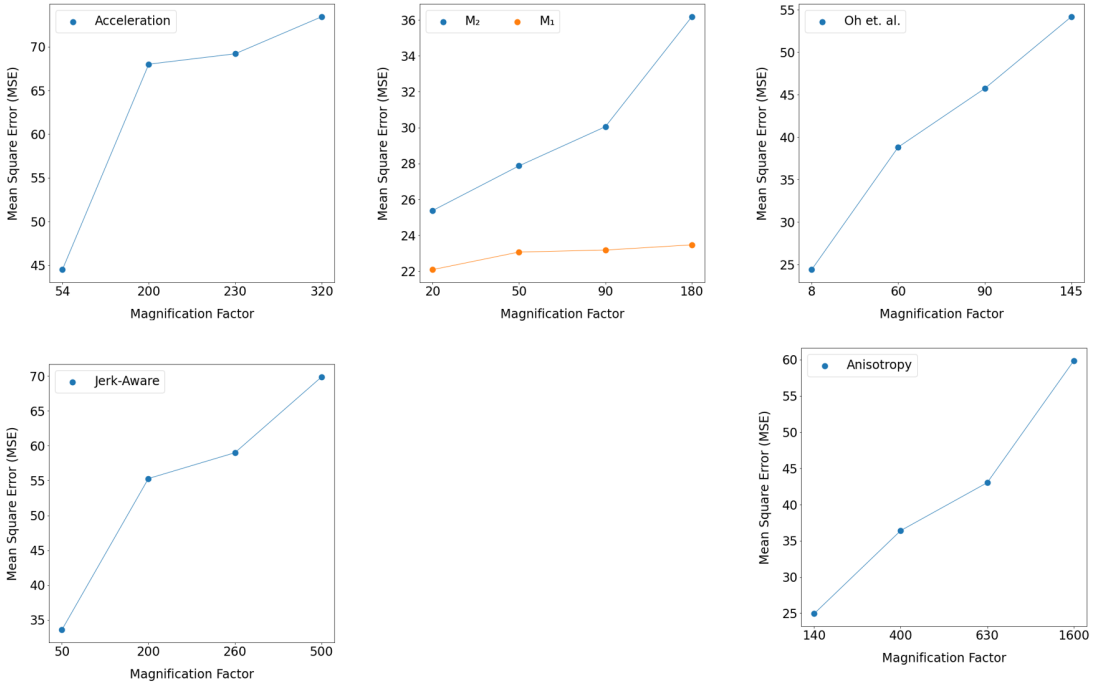


Figure 3.11: Effects of change in Magnification Factor on Acceleration based method [30], Jerk-Aware method [31], Anisotropy method [32], Oh *et al.* [29], Ours Base model ($M_1$), and Ours lightweight model ($M_2$). The magnification factor has a different meaning in each respective method [29]. The values of the magnification factor are chosen such that, they produce the same amount of output motion with different input motion (the respective input motion values with output MSE are shown in Figure 5.11 (B) ). Average mean square error (MSE) is computed across the predicted output and ground truth, over 25 different videos.

magnification factor values ensure consistent output motion levels despite varying input motion, and the resulting MSE is averaged across the videos as shown in Figure 3.11.

### 3.2.4 Additional Experiments

**Frequency selectivity:** When the deep learning method is not trained directly with temporal filters, using temporal filters on intermediate features can produce incorrect results [29]. So to avoid that, video is first pre-processed with a temporal filter to suppress unwanted motion. For this, [26] method's output at a small magnification factor ( magnification factor=4) is given as an input to our method. Figure 3.14, shows the intermediate features which highlight the motion parts. As different temporal filter inputs feature, highlight different motion parts.

**Visual effects on change in magnification factor:** Visual effects of increasing magnification factor vary across video analysis methods. The Acceleration method exhibits distortions, particularly in dynamic scenarios (Figure 3.15 [30]). The Anisotropy method shows subtle magnification changes with distortions, especially in dynamic settings (Figure

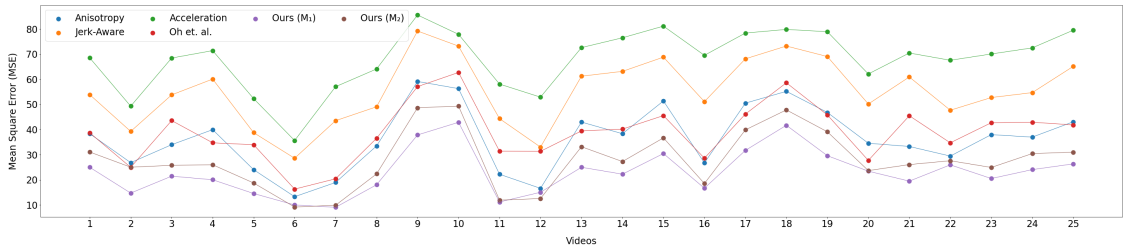


Figure 3.12: Mean Square Error (MSE) of Anisotropy method, Jerk-aware method, Acceleration method, Oh *et al.* method, and the proposed methods on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

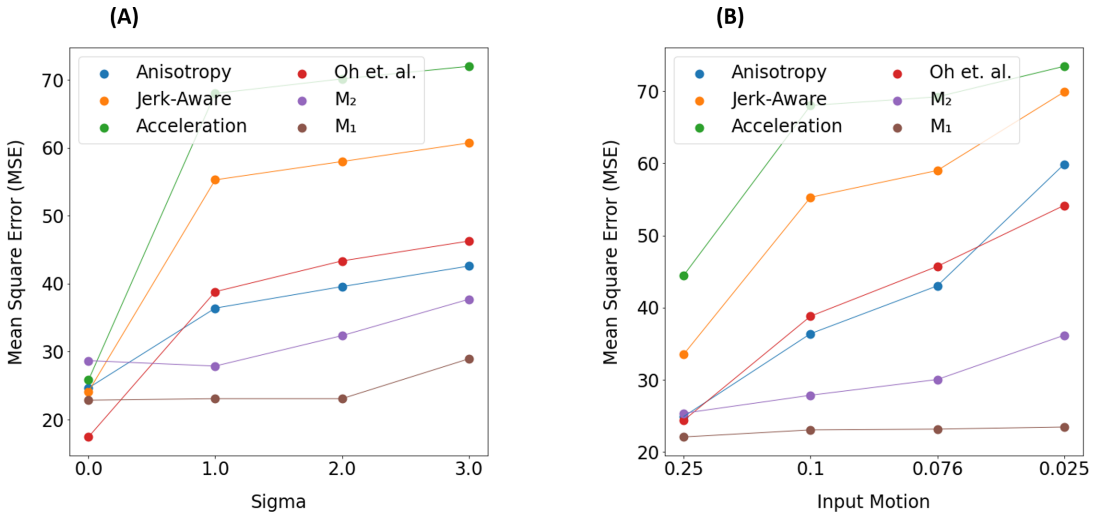


Figure 3.13: Noise Test and Sub-pixel Motion Test: (A) shows the variation across the increase of noise value (sigma) in input (noise test). (B) shows the effects of a decrease in input motion (sub-pixel motion test). The magnification factor is changed such that it produces the same amount of magnification. In both cases, the average mean square error (MSE) is computed across the predicted output and ground truth, over 25 different videos. Comparison is done with the Anisotropy method, Jerk-aware method, Acceleration method, Oh *et al.* method, the proposed base model $M_1$, and lightweight model $M_2$.

3.16 [32]). The Jerk-Aware method demonstrates minimal magnification alterations with significant distortions, particularly in dynamic scenes (Figure 3.17 [31]). The Phase-based method reveals substantial distortions in dynamic scenarios and ringing artifacts in static scenes (Figure 4.16 [26]). Oh *et al.*'s method provides enhanced magnification but introduces unwanted motion and blurry distortions, worsening with higher magnification (Figure 4.18 [29]). Conversely, the Base model ($M_1$) shows fewer distortions with increasing magnification, while the lightweight model ($M_2$) suffers performance degradation, particularly at extreme magnification levels due to reduced parameterization (Figure 7.12, Figure 7.13).

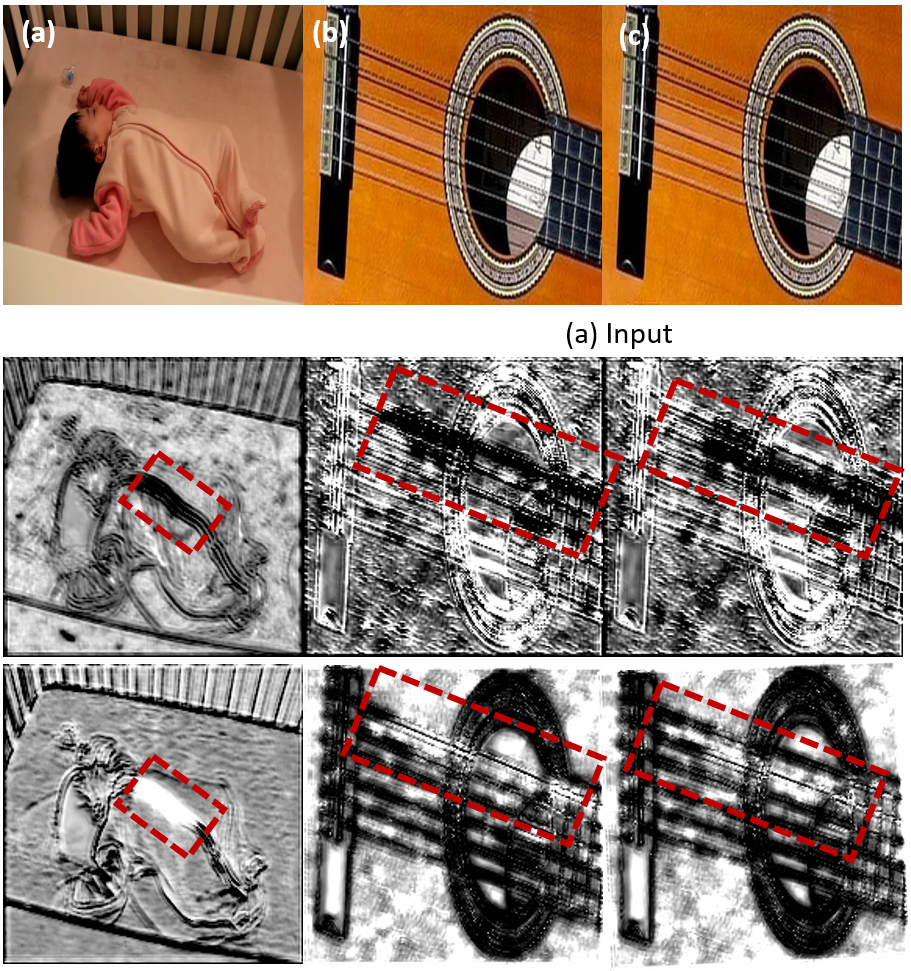


Figure 3.14: Intermediate features (*row-2,3*) of our proposed lightweight network, highlighting the motion in the red bounding box. (a) shows the baby video motion features. Baby has a minute chest motion while breathing. (b )and (c) highlight the motion parts E-string (82Hz), and A-string(110Hz) of the guitar. Hence, the proposed method is able to capture the minute motions of the baby video and temporal filter guitar video.

### 3.2.5 Ablation Study

An ablation study is conducted on the proposed architecture to see the importance of different modules. For this, five different models are trained (a) Without proxy model based feature loss training (b) Without feature Sharing Encoder, (c) Without appearance Encoder, (d) Without $L_{edge}$ loss, and (e) Without $L_p$ loss. We test them on synthetic videos and give their aggregate MSE in Table 3.6. The proposed method shows the minimum MSE value.

The proposed feature based proxy loss is used to reduce the magnification of unwanted changes. Appearance encoder based loss helps to give a denoising signal to make the network robust to illumination changes. Further, a feature sharing encoder is used to reduce the effects of noise. Also, the appearance encoder, $L_{edge}$ and $L_p$ loss help in the generation of a magnified frame of appropriate quality. As shown in Table 3.6, after the

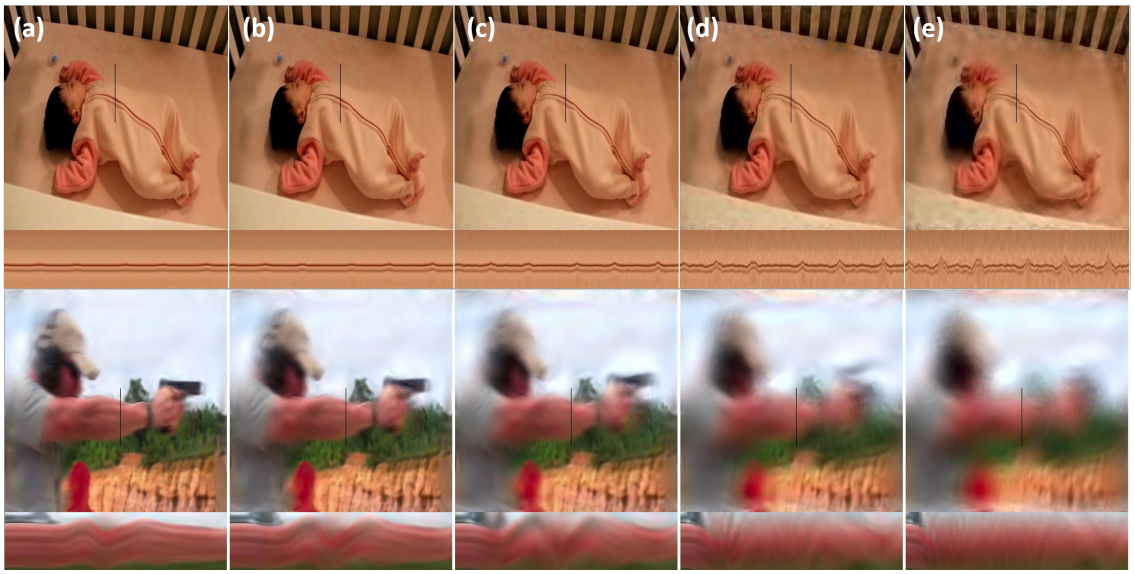


Figure 3.15: Effects of change in Magnification Factor: Figure illustrates Acceleration method [30] output. Different values of magnification factor in increasing order from (a) 20, (b) 40, (c) 60, (d) 100, and (e) 200 are used to generate the output shown in the respective column. An increase in magnification factor leads to more increase in distortions than a small increment in magnification, especially in dynamic scenarios.

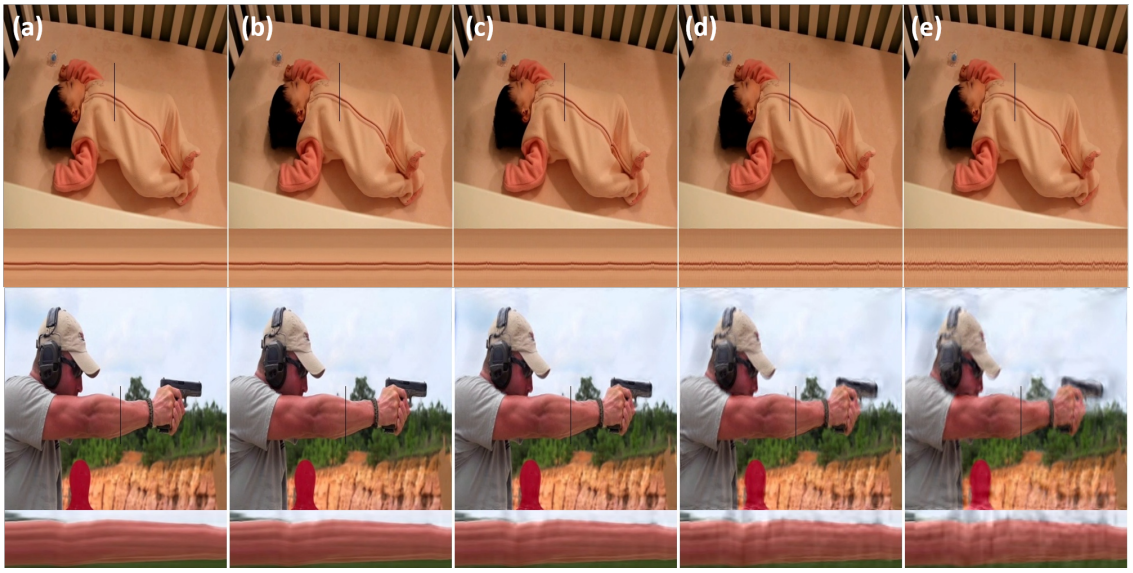


Figure 3.16: Effects of change in Magnification Factor: Figure illustrates Anisotropy method [32] output. For columns, (a) 50, (b) 100, (c) 200, (d) 500, and (e) 1000, respective magnification factor values are used to generate magnified output. As visible from the figure, with an increase in magnification factor, there are minute changes in magnification while increments in distortions (especially in dynamic scenarios).

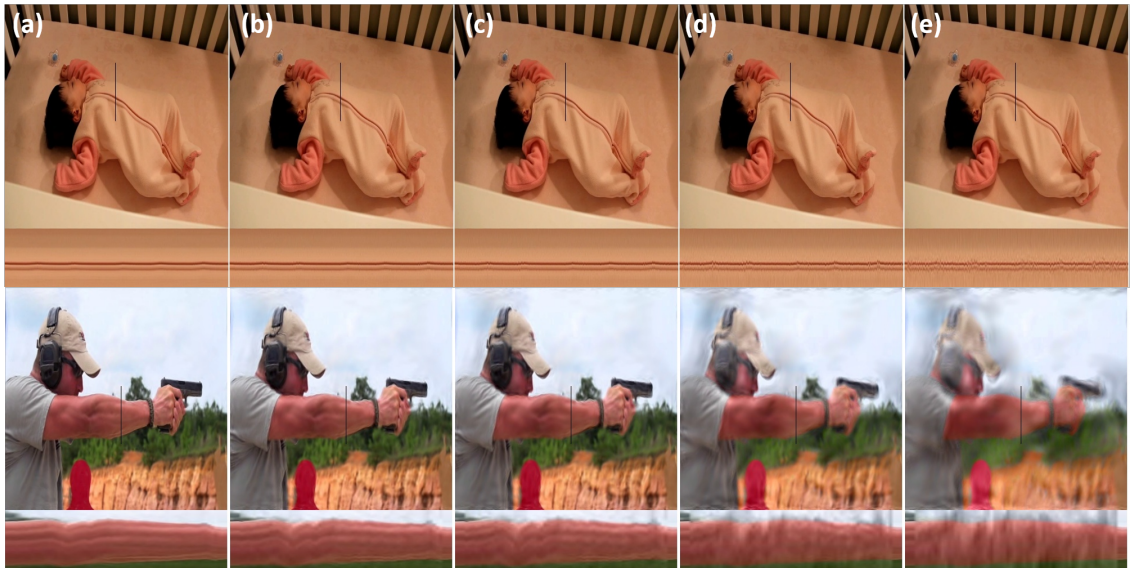


Figure 3.17: Effects of change in Magnification Factor: Figure illustrates Jerk-Aware method [31]. Different values of magnification factor in increasing order from (a) 10, (b) 30, (c) 50, (d) 150, and (e) 400 are used to generate the output shown in the respective column. As visible from the figure, with an increase in magnification factor, there are minute changes in magnification while increments in distortions (especially in dynamic scenarios).

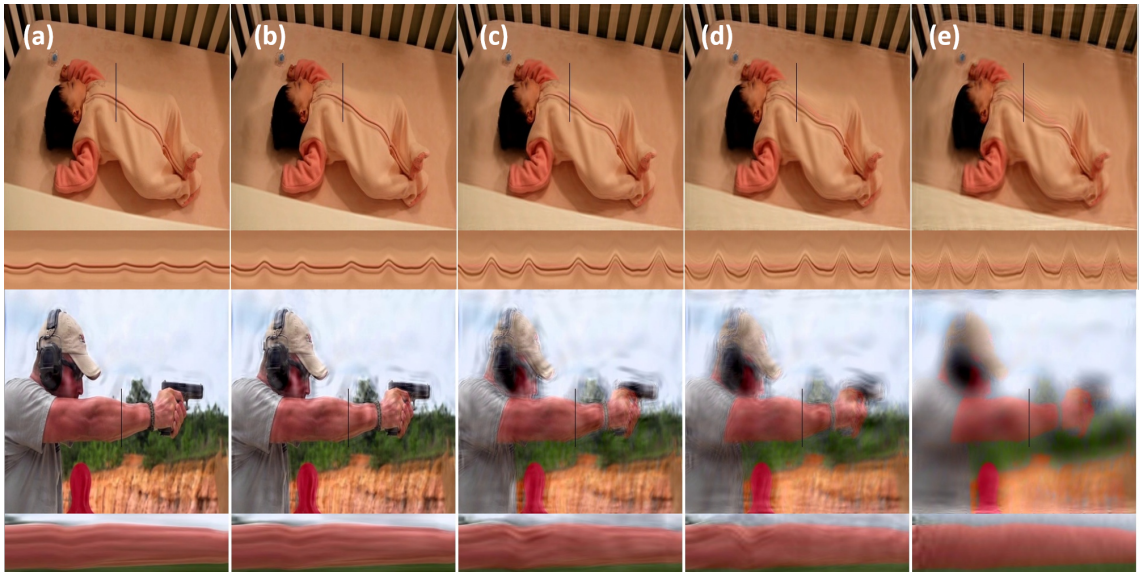


Figure 3.18: Effects of change in Magnification Factor: Figure illustrates Phase based method [26] output. Different values of magnification factor in increasing order from (a) 10, (b) 20, (c) 40, (d) 60, and (e) 100 for baby video and (a) 1, (b) 2, (c) 5, (d) 10, and (e) 100 for gun video are used to generate the output shown in respective column (note:- for other methods same values are used for both the videos). The linear methods are not suitable for dynamic scenarios, as they are unable to ignore dynamic motion. So, they produce large distortions in gun video (dynamic scenarios). Whereas in static scenario (baby videos), with an increase in magnification factor there is increment in both, the amount of magnification and ringing artifacts (visible as lines overlapping the edges of motion objects) in the static scenario.

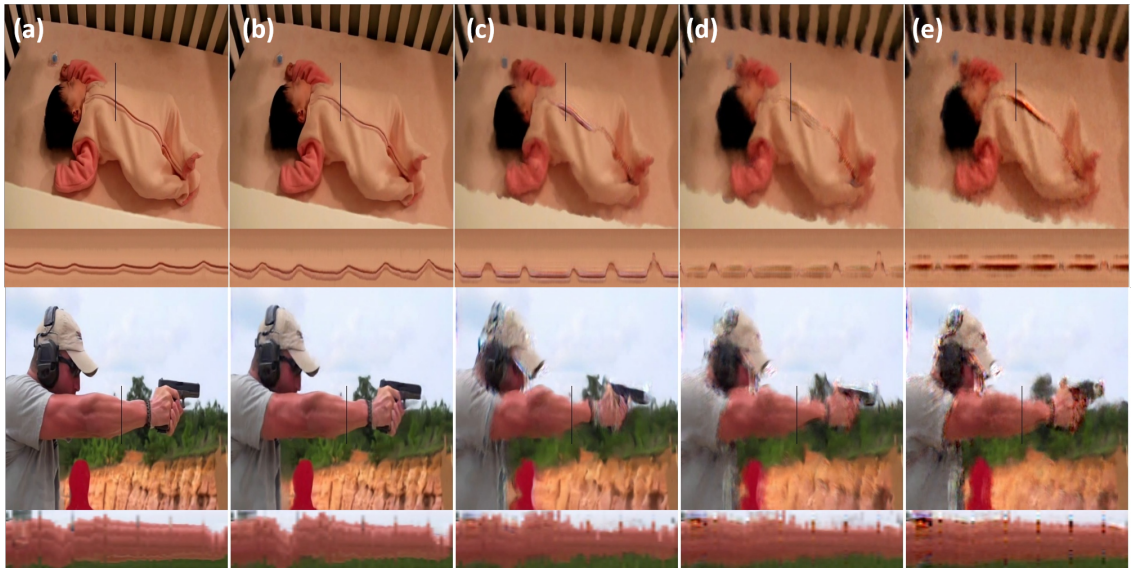


Figure 3.19: Effects of change in Magnification Factor: Figure illustrates Oh *et al.* method [29] output. Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column. It produces more magnification, (both in static and dynamic scenarios), but it also produces some unwanted motion (visible as large spikes in the temporal slice) and blurry distortions in the video. Distortions are increased with inclemently in the magnification factor.

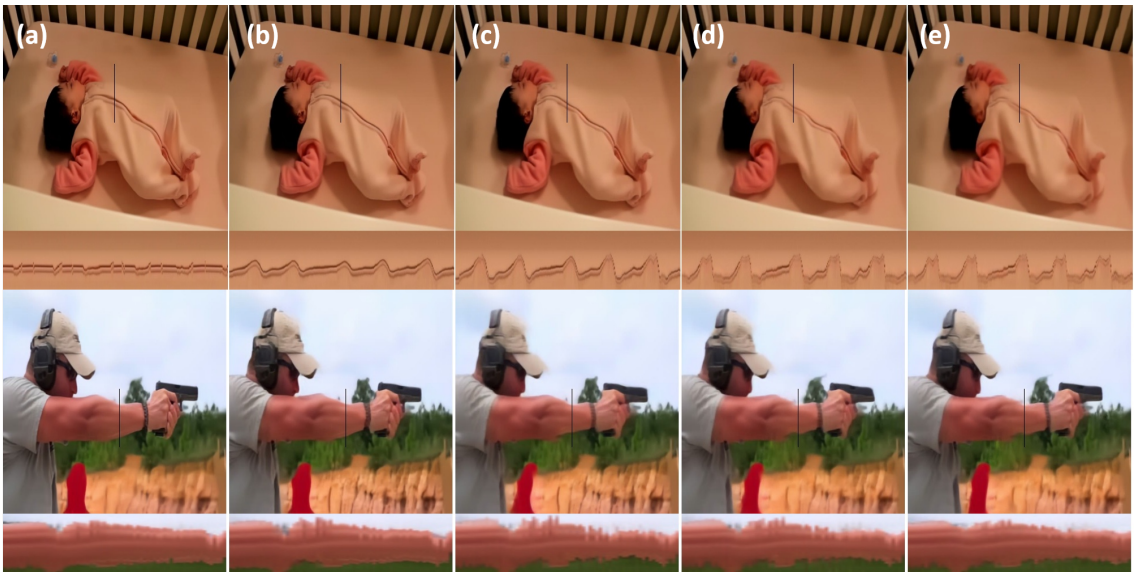


Figure 3.20: Effects of change in Magnification Factor: Figure illustrates our Base model ($M_1$) output. Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column. $M_1$ shows fewer distortions while increasing the amount of magnification as compared to other SOTA methods, both in static and dynamic scenarios.

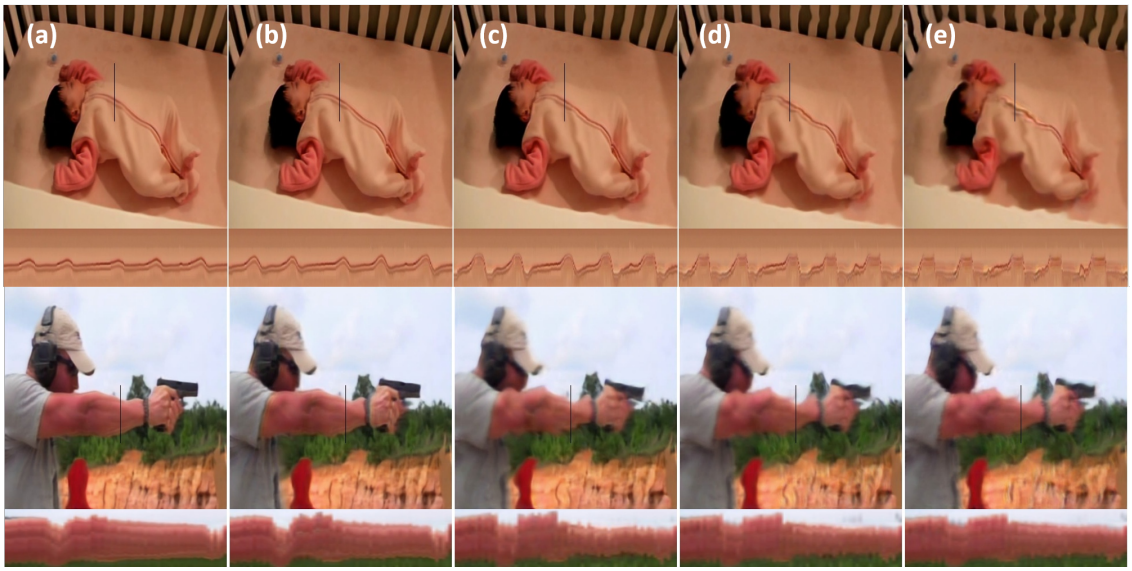


Figure 3.21: Effects of change in Magnification Factor: Figure illustrates our lightweight model ($M_2$) output. Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column. $M_2$ also shows a good amount of magnification, but with an increase in magnification factor, its performance degrades as compared to $M_1$. This is expected as $M_2$ has much fewer parameters than $M_1$, so their performance gap between becomes observable in extreme scenarios.

Table 3.6: Aggregate Mean Square Error (MSE) computed across synthetic videos on (a) Without proxy model based feature loss training, (b) Without feature sharing encoder, (c) Without appearance Encoder, (d) Without $L_{edge}$ loss, (e) Without $L_p$ loss and (e) Ours base model ($M_1$) on synthetic videos. The proposed method has the minimum error. (First best shown in bold.)

| **Methods** | $M_1$ | (a) | (b) | (c) | (d) | (e) |
|---|---|---|---|---|---|---|
| **MSE** | **23.07** | 27.85 | 30.1 | 37.7 | 31.1 | 40.2 |

inclusion of all the modules and losses in the training process, the proposed method has the minimum MSE value.

## 3.3 Summary

In this chapter, we discuss a deep learning based model for video motion magnification. It consists of proxy model based feature loss, feature sharing based encoders, and appearance encoder based regularization terms, to reduce the effects of noise, illumination *etc* and refine the motion features. The appearance encoder also helps to extract common appearances in the input frames and combine them with the manipulator output, which is given to the decoder to produce a magnified frame. Additionally, a lightweight model with reduced computational complexity is proposed along with the base model. The results of the proposed models are evaluated qualitatively and quantitatively on real

and synthetic videos with SOTA methods. Results show that the proposed models perform better than the existing methods both qualitatively and quantitatively for motion magnification. However, various applications require differing levels of computational complexity. Therefore, there is a need for a versatile approach that can accommodate application-specific computational models.

# Chapter 4

# A Knowledge Distillation Based Latency Aware – Differentiable Architecture Search for Video Motion Magnification

For healthcare applications like real-time heart rate and respiration rate monitoring, or in industries where time constrained output is needed, current methods are computationally heavy. Not much work is done to make the framework computationally cost-effective. Furthermore, applications of motion magnification are increasing day-by-day [61], [21], [62], [18], [11], [63], [64], [65] *etc.* Different applications demand different latency constraints, depending on their use, hardware *etc.* So, there is a need for a framework to generate application-specific time constrained, motion magnification models. Nowadays, hardware-aware Neural Architecture Search (NAS) based solutions are becoming popular for searching computationally constrained CNN architectures [66],[67], [68], [69], [70], [71]. But these approaches mainly focus on image classification problems and very less work has been done on image translation problems [51], [72], [73], [74]. Also, existing NAS algorithms are not directly applicable to the motion magnification problem, due to the following challenges:

- They use a single stream encoder for training a super-net, so there is a need to extend them for multiple streams for motion magnification problems [29]. But this will increase the memory consumption of the super-net drastically.
- Various applications of motion magnification require different latency constrained solutions, so joint optimization of latency and accuracy is needed. The algorithm should focus on finding the best architecture under fixed latency constrain rather than further decreasing model latency than required, at the expense of output accuracy.
- Further, finding the optimum architecture under fixed latency constraints, requires search space to be specifically tailored for video motion magnification problems.

Keeping these points into consideration, we propose a Knowledge Distillation Based Latency Aware-Differentiable Architecture Search (KL-DNAS) method to solve the latency constrained motion magnification problem. The main contributions of this work are summarized as follows:

- We propose training a student super-net architecture by parts while using knowledge distillation to transfer knowledge from the teacher model, to reduce memory requirements for the architecture search, and to improve denoising characteristics for motion magnification.
- We use layer-wise network search and propose separate search spaces for different parts of the student architecture. Further, NAS Layer-(R) and NAS Layer-(C) are proposed to search among different receptive fields and feature connections respectively.
- A novel latency loss is proposed to jointly optimize network accuracy and target latency in student architecture search.
- Physical accuracy of the searched student model is evaluated and the results are qualitatively and quantitatively analyzed. Further, the analysis shows that the student model has achieved state-of-the-art results.

Teacher and searched student networks are trained on the dataset provided in [29] and, compared qualitatively and quantitatively on different tasks with state-of-the-art methods for motion magnification. Further, an ablation study is conducted to analyze the effect of different parts of the proposed method.

### 4.0.1 Neural Architecture Search (NAS)

Primarily, computationally expensive reinforcement learning based algorithms were proposed for Neural Architecture Search (NAS) [75], [76]. Further, different methods were proposed for a reduction in their computational costs [77], [78], [76]. Evolutionary algorithms based network search is also able to give good results [79], [80],[81]. Moreover, one shot based methods were proposed, for a reduction in the search cost of the network [82], [83]. In this, the proposed super-net covers all the candidate architectures, so it is trained only once. To train a large super-net, some methods sampled a single path [84], [85] to reduce memory costs and used it as a performance estimator [66]. Whereas DARTS [86] based methods [51], [72], [73] use gradient descent to jointly optimize weights of super-net and parameters of network search. Even though, very few works are done on the extension of NAS on other computer vision problems. Due to DARTS simplicity and effectiveness, it is used in different applications [87], [88], [89], [90], [51], [72], [73], [74]. In our work, we try to extend DARTS based gradient descent network search for the video

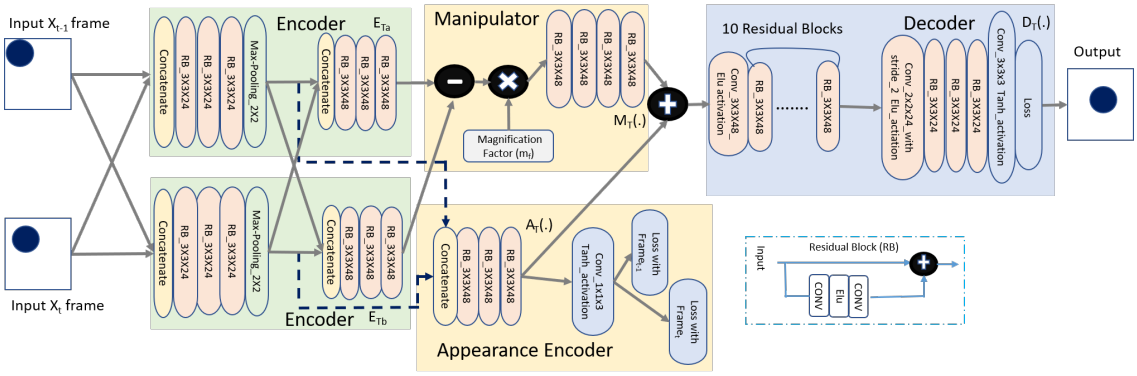


Figure 4.1: Teacher architecture for motion magnification. It consists of three main parts: 1) Encoders ($E_{Ta}$, $E_{Tb}$), 2) Manipulator $M_T$ and Appearance Encoder $A_T$, and 3) Decoder $D_T$. Frames $x_t$ and $x_{t-1}$, with $m_f$ as the magnification factor are given as input to the network.

translation problem of motion magnification. Also, we have tried a random sampling of the path, similar in [91], [92] to reduce memory cost but the model does not converge. So, such techniques are not feasible for our problem.

Knowledge Distillation (KD) is a transfer of knowledge from the teacher network to the student network [54]. KD is used to progressively train the student architecture from the teacher for efficient student training [93]. KD with NAS is preferred to increase the capacity of the student model [94]. Li *et al.* [94] suggest knowledge of a teacher model lies not only in the network parameters but also in the network architecture. Li *et al.* [94] propose KD with NAS to blockwise search the architecture, for image classification problems. It does not take into account the efficiency of previous searched blocks. Whereas, our work is different, as it searches by parts in a serial order. So, it takes into account the feature efficiency of previous search architecture blocks in the next architecture search. Further, KD is employed to enforce denoising characteristics and reduce memory requirements in network architecture search (see Section 4.1.2).

## 4.1 Proposed Framework

In the following subsections, we discuss the teacher model. Further, we explain our approach in detail. We explain the search of the student model by parts from the teacher network. Then we illustrate the layer-wise search space and define the latency loss for the student architecture search.

### 4.1.1 Teacher Model

The teacher model consists of two-stream encoders to convert input frames to feature space, in which motion information is enhanced and can be easily extracted. They are represented as $E_{Ta}$ and $E_{Tb}$ (Subscripts $T$, $S$, denote teacher and student network, and symbol $E$, $M$,

$A$, $D$ denote the encoder, manipulator, appearance encoder, and decoder respectively). It uses manipulator $M_T(.)$ to extract the motion information and appearance encoder $A_T(.)$ to get a common appearance between input frames. Decoder $D_T(.)$ combines the output of the manipulator and appearance block to generate the magnified output. Residual blocks are used to transform the features (A detailed diagram is shown in Figure 4.1).

Oh *et al.* [29] use color perturbation frames and regularization across the encoders to separate shape and texture information from features generated by each encoder. This also helps to reduce the effects of the noise [29]. But their method sometimes produces blurry distortions and artifacts. Where in the teacher model, to reduce noise, features across both the encoders are shared among each other, and appearance encoder $A_T(.)$ is used. Loss between $A_T(.)$ features and input frames is used to extract common appearance features. For calculating loss across $A_T(.)$, no noise is added to the ground truth (input frames). So, it will force denoising characteristics. The assumption is, that common appearance features will have common information across frames where motion is not present. As this information will be the same in the input and output, it will help in the generation of the output.

$L_1$ loss with edge loss and perceptual loss is used for teacher model training. The loss function for teacher training is defined as $L_{mag}(\widehat{Y}_p, Y_{gt})$ in Eq. (6.9).

$$\begin{aligned} L_{mag}(\widehat{Y}_p, Y_{gt}) = \lambda_1(||\widehat{Y}_p - Y_{gt}||_1) + \lambda_2(||\nabla\widehat{Y}_p - \nabla Y_{gt}||_1) + \lambda_3(||\phi_i(\widehat{Y}_p) - \phi_i(Y_{gt})||_1) \\ +||A_T(E_{Ta}(x_{t-1}), E_{Tb}(x_t)) - x_{t-1}||_1 + ||A_T(E_{Ta}(x_{t-1}), E_{Tb}(x_t)) - x_t||_1 \end{aligned} \tag{4.1}$$

where, $\widehat{Y}_p$ is predicted frame, $Y_{gt}$ is the ground-truth. $\nabla$ represents the finite differences in a horizontal and vertical direction for edge loss. $\phi_i$ illustrates the VGG-16 feature space and used as perceptual loss [59]. $\lambda_1 = 10.0$, $\lambda_2 = 1.0$ and $\lambda_3 = 0.1$ values are used for network training.

### 4.1.2 Proposed Method

For finding the motion magnification architecture, a super-net model is trained by parts and knowledge distillation. For this, first, a teacher model is trained without adding noise and is used to supervise the training of different parts of the student super-net. Training by parts helps to reduce memory consumption and transfer knowledge from teacher model to student [94]. So, the overall structure of the student network is similar to that of the teacher network but each part of the student network has a different search space. Also, the problem of video motion magnification is sensitive to noise, as both motion and noise are minute in nature. We assume denoising characteristics can be forced in network architecture search using KD by parts.

Instead of finding cells like [86], we search individual layers for each part of the student

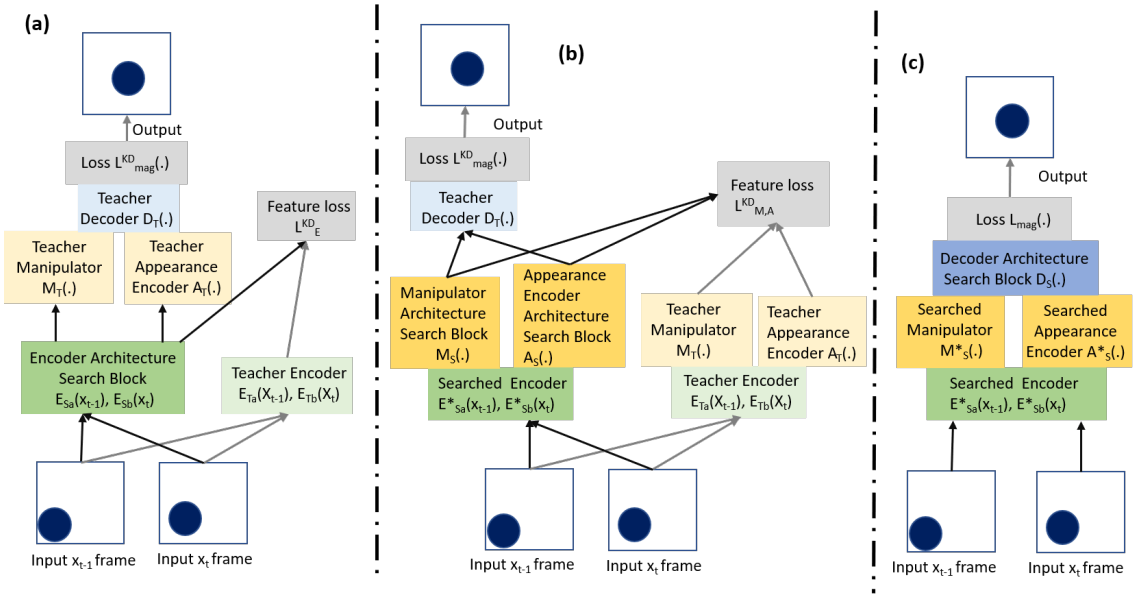


Figure 4.2: Training of student super-net by parts. Student: **(a)** Encoder architecture search, **(b)** Manipulator and Appearance encoder architecture search, and **(c)** Decoder architecture search training processes are shown respectively.

network. Layer-wise search space consists of a search among different receptive fields and connections among various layer features. In the video motion magnification problem, selecting optimum receptive fields for different parts of the network will help us to reduce the network complexity, size *etc.* Layer-wise search gives the network the ability to change the receptive field of each layer individually for better output. To facilitate the network in finding relations among different features, the network has the option to decide whether to `add`, `concate(.)`, or have `no connection` with previous layer features. Moreover, as network layers can also have a relation with more than one previous layer feature or features from different frames, so here, we used serial connections (see Section 4.1.2). The number of possible combinations of networks in the student supernet goes to $9^{33} \approx 10^{31}$. With such a large search space, a network can find residual blocks (used in [29]), or even more optimum blocks for motion magnification.

Different applications of motion magnification require different computational complexity. Instead of using the hit and trial method to achieve the desired latency, we propose latency based loss. We specify the desired latency and jointly optimize it with accuracy. It prevents any further decrease in latency at the expense of accuracy. Hence, the network can focus on searching for an efficient model under fixed target latency.

### Knowledge Distillation Based Training by Parts

Searching a whole architecture directly for video motion magnification requires a lot of memory. To overcome the memory issue, we propose an architecture training by parts. Searching by part enables us to train the network until it converges. This helps to reduce

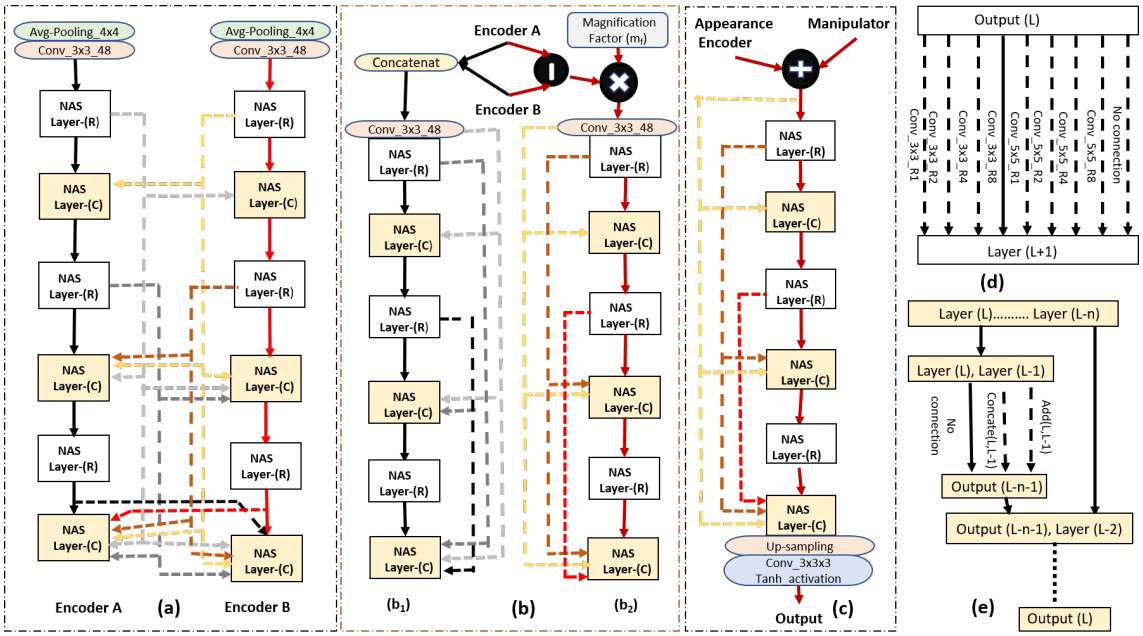


Figure 4.3: Search space of **(a)** Student Encoder architecture block, **(b)**: ($b_1$) Student Appearance encoder architecture block, ($b_2$) Student Manipulator architecture block, **(c)** Student Decoder architecture block, **(d)** NAS layers-(R), **(e)** NAS layers-(C) shows layers search spaces.

the representation shift caused by shared parameters [94] and helps in the correct selection of the candidate model. Student network is searched by taking latency constrain into account. We use the teacher network to guide the search for the student model. The teacher network is trained without adding noise, whereas the student architecture is being searched with noisy input. KD further helps to enforce de-noising characteristics in student architecture search. The student model search is divided into three parts to reduce the memory requirements: 1) Encoder Architecture Search, 2) Manipulator and Appearance Block Architecture Search, and 3) Decoder Architecture Search.

First, to search the encoder architecture, $E_{Ta}$ and $E_{Tb}$, the output features of two stream encoders of the teacher network are used to transfer knowledge ($L_E$) as shown in Figure 4.2(a) and in Eq. (5.2).

$$L_E = ||E_{Ta}(X_{t-1}) - E_{Sa}(x_{t-1})||_1 + ||E_{Tb}(X_t) - E_{Sb}(x_t)||_1 \tag{4.2}$$

Where, $x_{t-1}$, $x_t$ and $X_{t-1}$, $X_t$ are the input frames with noise and without noise respectively. Also, $E_{Sa}$ and $E_{Sb}$ are the two stream encoder features of the student super-net. Super-net encoder features are given to the teacher network during training while freezing the teacher network weights, (also helps in transfer knowledge) as shown in Figure 4.2(a) and in Eq. (5.8) to generate predictions ($\widehat{Y}_p^E$).

$$\widehat{Y}_p^E = D_T(M_T(E_{Sa}(x_{t-1}), E_{Sb}(x_t), m_f), A_T(E_{Sa}(x_{t-1}), E_{Sb}(x_t))) \tag{4.3}$$

The final loss function used for searching the encoder architecture block is defined as $L_E$ in the following:

$$L_E = L_{mag}^{KD}(\widehat{Y}_p^E, Y_{gt}) + \lambda_4 L_E^{KD} + L_{lat}(\beta) \tag{4.4}$$

$\lambda_4$= 0.1 value is used for training and $L_{lat}(\beta)$ is the proposed latency loss (Please see section 4.1.2 for more detail) . Similarly, Knowledge distillation based loss ($L_{M,A}$ as shown in Figure 4.2(b)) for a search of manipulator and appearance encoder block is defined as:

$$\begin{aligned} L_{M,A} = ||A_T(E_{Ta}(X_{t-1}), E_{Tb}(X_t)) - A_S(E_{Sa}^*(x_{t-1}), E_{Sb}^*(x_t))||_1 \\ +||M_T(E_{Ta}(X_{t-1}), E_{Tb}(X_t), m_f) - M_S(E_{Sa}^*(x_{t-1}), E_{Sb}^*(x_t), m_f)||_1 \end{aligned} \tag{4.5}$$

Where $m_f$ is the magnification factor that decides the amount of magnification. The searched encoder for the student network is used to generate features for the super-net manipulator and appearance encoder as shown in Figure 4.2(b) ($*$ indicates the searched architectures in the equations). Super-net appearance encoder $A(.)$ and manipulator $M(.)$ search block features are given to the teacher decoder to generate predictions $\widehat{Y}_p^{M,A}$, (also helps in Knowledge distillation) as shown in Figure 4.2(b) and in the following:

$$\widehat{Y}_p^{M,A} = D_T(M_S(E_{Sa}^*(x_{t-1}), E_{Sb}^*(x_t), m_f), A_S(E_{Sa}^*(x_{t-1}), E_{Sb}^*(x_t))) \tag{4.6}$$

Similarly, Eq. (4.7) defines the final loss $L_{M,A}$ for the student super-net appearance encoder $A(.)$ and manipulator $M(.)$ search block.

$$L_{M,A} = L_{mag}^{KD}(\widehat{Y}_p^{M,A}, Y_{gt}) + \lambda_4 L_{M,A}^{KD} + L_{lat}(\beta) \tag{4.7}$$

For Decoder $D(.)$ search, architectures found for encoder, manipulator and appearance encoder is used as the backbone, shown in Figure 4.2(c) to generate prediction $\widehat{Y}_p^D$ as shown below:

$$\widehat{Y}_p^D = D_S(M_S^*(E_{Sa}^*(x_{t-1}), E_{Sb}^*(x_t), m_f), A_S^*(E_{Sa}^*(x_{t-1}), E_{Sb}^*(x_t))) \tag{4.8}$$

The final loss for the student super-net decoder search block is illustrated as:

$$L_D = L_{mag}(\widehat{Y}_p^D, Y_{gt}) + L_{lat}(\beta) \tag{4.9}$$

**Gradient decent based search**

Let $O$ be a set of candidate operations (e.g., `3X3 Conv with rate, r` $\in (1, 2, 4, 8)$ `5X5 Conv with rate, r` $\in (1, 2, 4, 8)$ `and no connection`)) where each operation represents some function $o(.)$ to be applied to $i^{th}$- layer output feature, $x^{(i)}$. To make the search

space continuous, the categorical choice of a particular operation is relaxed to a softmax over all possible operations:

$$\bar{o}^{(i,j)}(x) = \sum_{o \in O}((exp(\alpha_o^{i,j}) / \sum_{o' \in O} exp(\alpha_{o'}^{i,j}))o(x)) \tag{4.10}$$

where $\alpha_o^{i,j}$ is the weight of the $j^{th}$ layer output feature of operation $o$, applied on the input feature of $i^{th}$ layer. At the end of the search, a discrete architecture can be obtained by replacing each mixed operation $\bar{o}^{(i,j)}$ with $o^{(i,j)} = argmax_{o \in O}\ (\alpha_o^{i,j})$.

### Search Space

Different search spaces are used for 1) Encoder, 2) Manipulator and Appearance module, and 3) Decoder as shown in Figure 4.3 (a), (b), and (c) respectively. Each of the different search spaces has two basic elements: NAS Layer-(R) ($LR$) and NAS Layer-(C) ($LC$). Their structure is shown in Figure 4.3 (d), (e). In [31], [26], [28], [30] uses steerable pyramids for high receptive fields. [29] depends on different numbers of residual blocks in separate parts of the network for varying receptive fields. Selecting optimum receptive fields in different parts of the motion magnification network will help to improve the quality and reduce parameters. Layer-wise search through different receptive fields: NAS Layer-(R) ($LR$), help us to achieve this. In $LR$, a higher dilation rate can make the output blur, and expanding kernel size will increase the number of parameters. With these constraints, we experimentally narrow down the $LR$ search space to a total of 9 operations with different receptive fields (`3X3 Conv with rate, r` $\in (1, 2, 4, 8)$ `5X5 Conv with rate, r` $\in (1, 2, 4, 8)$ `and no connection`) as shown in Figure 4.3 (d). It takes one input feature and maps it to an output.

$LC$ is used to find relationships among the previous layer features $Layer(L)$ and features coming from other preceding layers $Layer(L-1)....Layer(L-n)$. It takes multiple inputs and maps them to a single output tensor as shown in Figure 4.3 (e). Its search space contains three different operations (`No connection, Add, Concate(.)`). Searching is done one at a time *e.g.* first relation between $Layer(L)$ and $Layer(L-1)$ is found, and then their output's ($Output(L-n-1)$) relation with $Layer(L-2)$, is searched in a serial order till it reaches $Layer(L-n)$ and gives $Output(L)$. This gives the network an option to select more than one incoming feature with different operations. With this, the search space increases to $\approx 10^{31}$ and covers residual blocks, dense blocks, *etc.* Also, as shown in Figure 4.3 (a) encoder architecture search block $LC$ have incoming features from both the encoders. This helps the network to find relations among input frame features. Whereas in Figure 4.3 (b) appearance encoder, manipulator ($b_1$, and $b_2$) and (c) decoder respectively, $LC$ have no incoming features from parallel blocks. We assume that, if the search space is large enough to cover the well known optimum blocks and architectures like teacher

Table 4.1: Comparison of the state-of-the-art deep learning method [29],[47], the teacher network, its variants, and our searched student networks in terms of the number of parameters, latency, and FLOPS. Figure 4.4, 4.5, 4.6, 4.7 illustrates the teacher network, its variants, and our searched student network architectures.

| **Model** | **Parameters** | **Latency** | **GFLOPs** |
|---|---|---|---|
| Oh *et al.* [29] | 0.98 M | 95 ms | 268.6 |
| STBVMM [47] | 31.2 M | 872 ms | 1376.8 |
| Ours Teacher Model($TM$) | 1.1 M | 142 ms | 375.5 |
| Ours $SM1$ | 0.35 M | 64 ms | 93.1 |
| Ours $SM2$ | 0.47 M | 69 ms | 123.9 |
| Ours $SM1-7$ | 0.8 M | 91 ms | 207.3 |
| $TM1$ | 0.33M | 73 ms | 117.1 |
| $TM2$ | 0.37M | 69 ms | 113.4 |

network, it will be able to find even more optimum networks for motion magnification

#### Latency Loss

Normally, hardware-aware NAS algorithms are used to find a model with low latency [92], [66]. But, there is a trade-off between output quality and latency. Since time constraints vary with applications, and in previous methods latency weights need to be changed manually in a hit and trial manner. Decreasing latency more than necessary can have detrimental effects on the quality of the output. To prevent this, we propose a novel $L_{lat}(\beta)$ function as given in Eq. (4.11). Where, $\beta_{target}$ is the target latency value for the model and $\alpha$ is the model latency calculated from the lookup table, as done in [92]. In $L_{lat}(\beta)$, log modulus of ratio $\alpha/\beta_{target}$ is taken. If the model latency deviates from the target in the search process, $L_{lat}(\beta)$ value increases. So, super-net edge weights are optimized to find the network producing better quality output, while maintaining the target latency.

$$L_{lat}(\beta) = |log(\alpha/\beta_{target})| \tag{4.11}$$

## 4.2 Experimental Results

Two student models $SM1$ and $SM2$, with two slightly varying latency constraints of 23ms and 25ms at `384X384` on Oh *et al.* [29] dataset are searched. Both $SM1$ and $SM2$ also demonstrate the effects of $L_{lat}(\beta)$. $SM1$ has $3.14X$, $2.8X$, and $89.1X$ fewer parameters than the teacher model, Oh *et al.* [29], and STBVMM [47], respectively. When comparing with the same models, $SM2$ has $2.34X$, $2.08X$, and $66.3X$ fewer parameters, as shown in

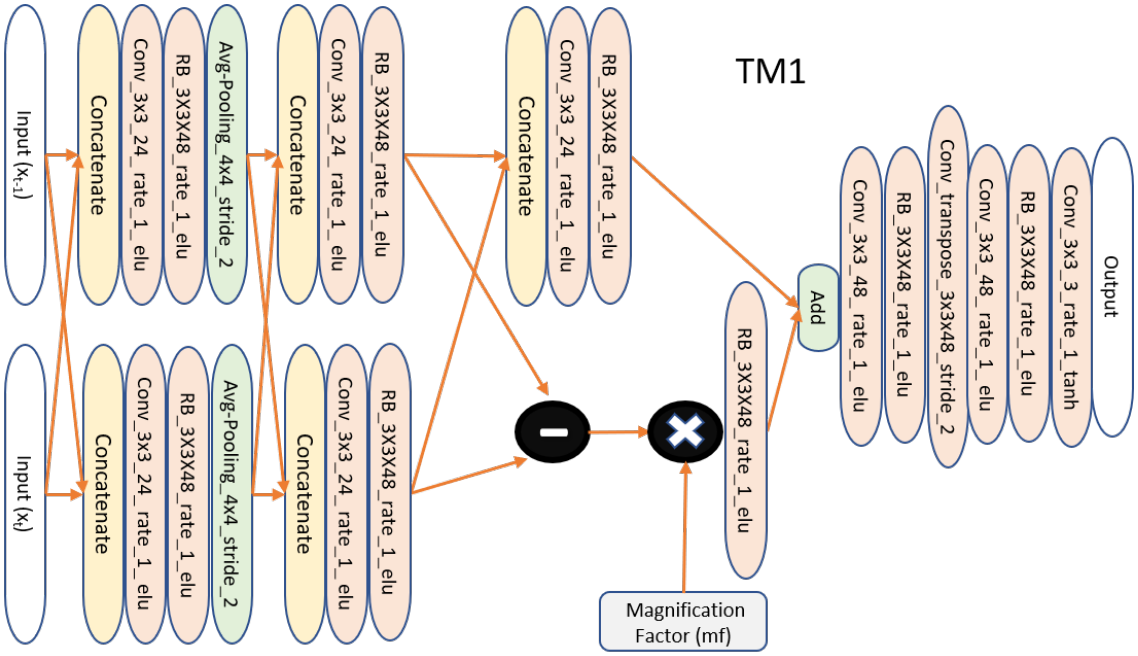


Figure 4.4: Architecture of teacher model $TM1$

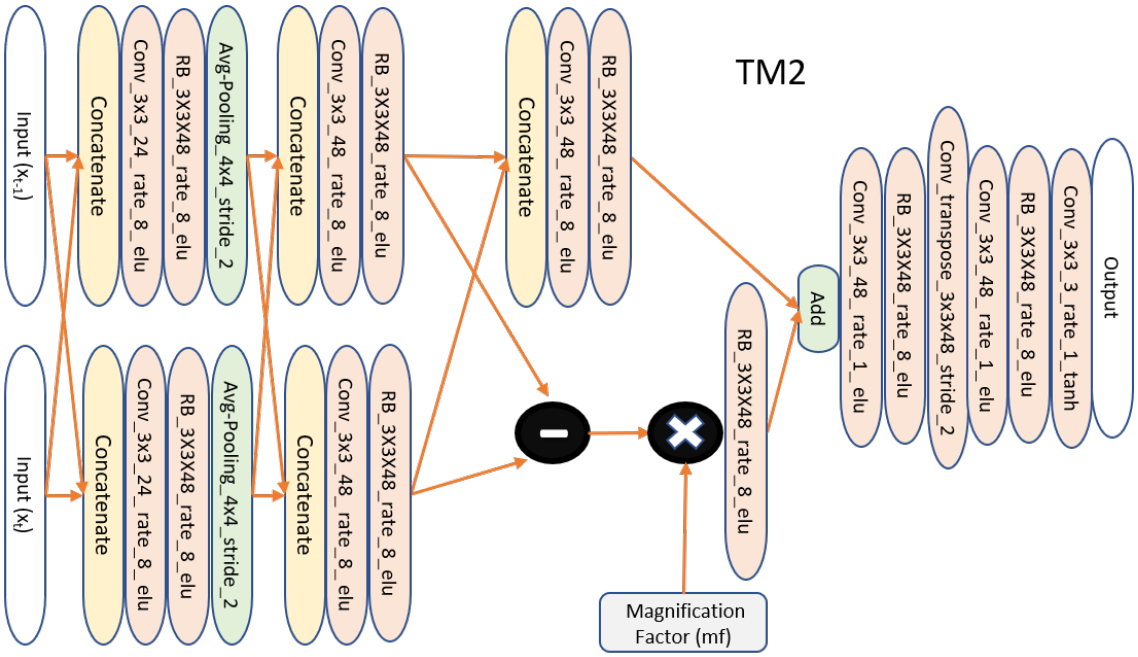


Figure 4.5: Architecture of teacher model $TM2$

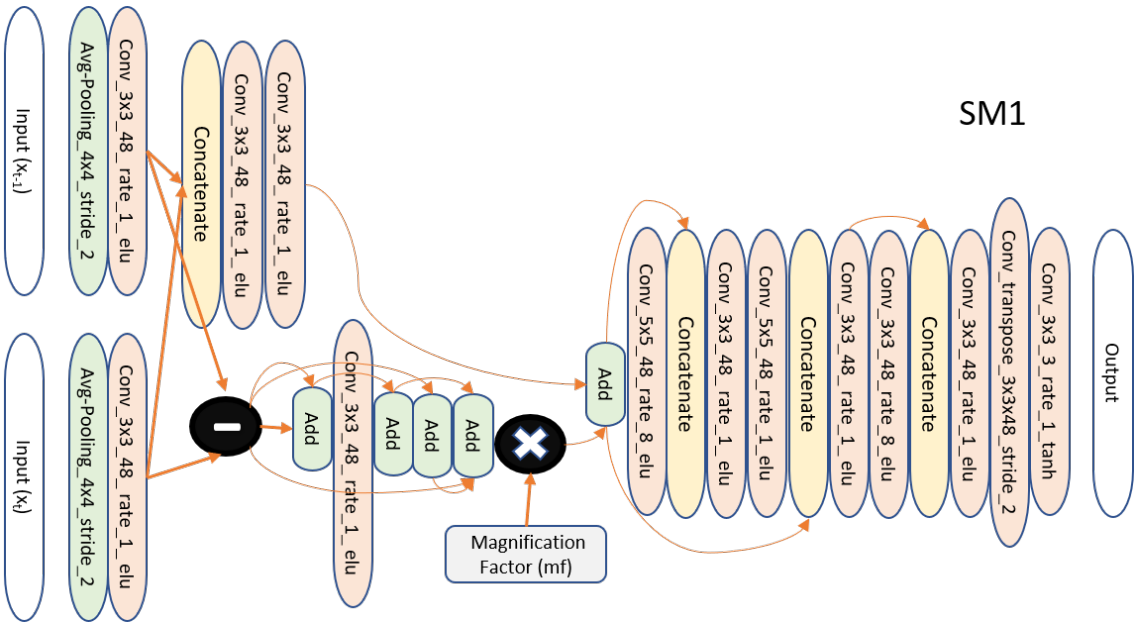


Figure 4.6: Architecture of searched student model $SM1$

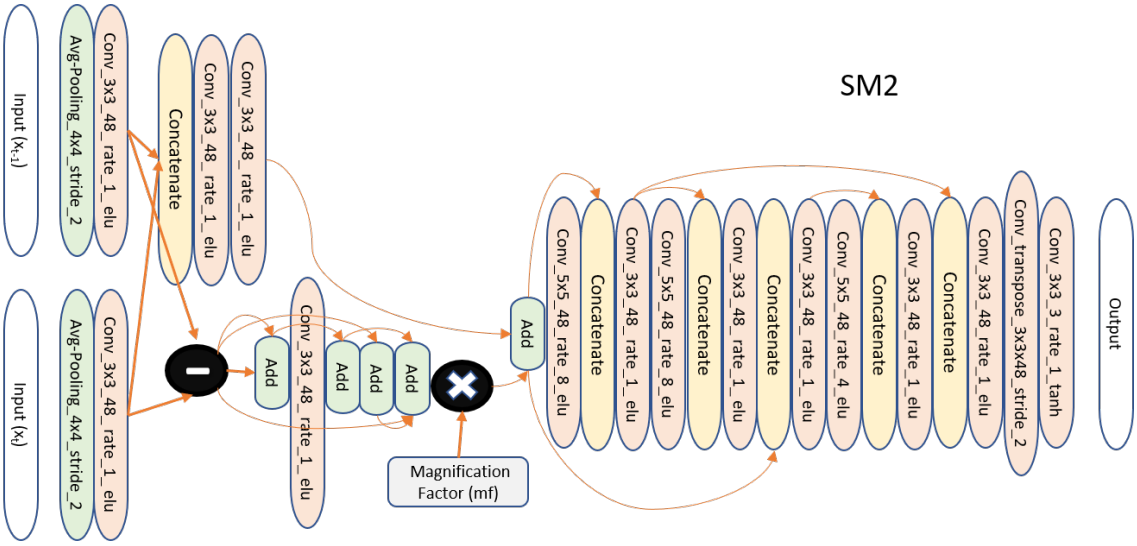


Figure 4.7: Architecture of searched student model $SM2$

Table 4.2: Parameters of different methods for motion magnified videos.

| Methods | Video | $M_f$ | Frequency |
|---|---|---|---|
| Ours ($SM1, SM2$,teacher model) | Gun | 10 | N/A |
| Ours ($SM1, SM2$,teacher model) | Drill | 10 | N/A |
| Ours ($SM1$ ) | Guitar | 9 | N/A |
| Ours ($SM1, SM2$,teacher model) | Physical accuracy(rod video) | 10 | N/A |
| Ours ($SM1, SM2$,teacher model) | synthetic videos | 50 | N/A |
| STBVMM [47] | Gun | 10 | N/A |
| STBVMM [47] | Drill | 10 | N/A |
| STBVMM [47] | Physical Accuracy | 10 | N/A |
| STBVMM [47] | synthetic videos | 50 | N/A |
| Oh *et al.* [29] | Gun | 10 | N/A |
| Oh *et al.* [29] | Drill | 10 | N/A |
| Oh *et al.* [29] | Physical Accuracy | 10 | N/A |
| Oh *et al.* [29] | synthetic videos | 60 | N/A |
| Acceleration [30] | Gun | 10 | 30 |
| Acceleration [30] | Drill | 4 | 3 |
| Acceleration [30] | Physical Accuracy | 20 | 15 |
| Acceleration [30] | synthetic videos | 100 | 15 |
| Jerk-Aware [31] | Gun | 10 | 20 |
| Jerk-Aware [31] | Drill | 25 | 3 |
| Jerk-Aware [31] | Physical Accuracy | 20 | 15 |
| Jerk-Aware [31] | synthetic videos | 100 | 15 |
| Anisotropy [32] | Gun | 100 | 20 |
| Anisotropy [32] | Drill | 100 | 3 |
| Anisotropy [32] | Physical Accuracy | 200 | 3 |
| Anisotropy [32] | synthetic videos | 200 | 15 |
| Phase Based [26] | Gun | 2 | 0.04-0.40 |
| Phase Based [26] | Drill | 5 | 1.00-14.00 |
| Phase Based [26] | Physical Accuracy | 5 | 1.00-14.00 |
| Phase Based [26] | synthetic videos | 200 | 0.00-15.00 |
| Euler [27] | Gun | 2 | 0.05-0.4 |
| Euler [27] | Drill | 5 | 0.05-0.4 |
| Euler [27] | Physical Accuracy | 5 | 0.05-0.4 |
| Euler [27] | synthetic videos | 200 | 0.00-15.00 |

Table 7.1. Furthermore, our searched student networks have lower latency compared to the teacher, Oh *et al.* [29], and STBVMM [47]. Latency values are calculated at `720X720` resolution on NVIDIA 2080 RTX for higher quality output. Searching by parts puts the target latency on each part. Sometimes, networks prefer `no connection`, resulting in lower latency than targeted. The difference between the target value and the actual search part latency is added to the target latency of the next part.

**Training:** All architectures are searched on NVIDIA 2080 RTX (8GB) in 200 hours. The whole architecture is of 6.2M parameters with 461 GFLOPS, whereas by parts, it is (a) 2.6 M parameters with 197 GFLOPS for encoder (b) 2.4 M parameters with 181 GFLOPS for appearance encoder and manipulator, (c) 1.3 M parameters with 102 GFLOPS for decoder (at 384x384 resolution for $SM1$ search). For the training of the network Gaussian

noise is added to the input.

In the following section, we compare the searched architecture to the state-of-the-art methods [27], [26], [30], [31], [32], [29],[47] qualitatively and quantitatively. The details of parameters used for result generation are shown in Table 4.2. Further, to check the physical accuracy of the output, a separate experiment is carried out. An ablation study is conducted to show the importance of various components in the architecture search.

### 4.2.1 Qualitative Analysis on Real Videos

**Analysis on Gun-shooting Video** Figure shows the results of state-of-the-art methods on gun-shooting video. The video contains large translation motion due to the movement of the camera and subtle motion in the forearm of the shooter due to gun recoil. We aim to magnify the subtle motions in the forearm in the presence of large camera motion. To show this, spatio-temporal slices at the forearm are taken as shown in Figure , at the red strip. Phase-based [26], Eulerian [27], Jerk-aware method [31], Anisotropy [32], Acceleration [30] methods have lower magnification as compared to the search student networks (more motion in the forearm results in more bending of the arm, in the temporal slice it is marked in black box). Oh *et al.* method [29] sometimes produces spurious motion and causes blurriness distortions in some frames (*they are visible as large spikes in Oh et al. temporal slice).* While STBVMM [47] shows promising results, but it comes with high computational complexity. In contrast, our proposed NAS method can search a student architecture which produces fewer distortions compared to Oh *et al.*, achieve greater magnification than [31], [32], and [30], and maintain lower computational complexity than [47], [29], even in dynamic scenarios (as seen in Figure 4.10 ).

**Analysis on rotational motion** Figure 4.9 shows the rotating hand drill, with rotational motion along its axis. Rotational motion is difficult to magnify, so a still video is taken to analyze the impact of magnification. In 2D, this rotational motion is perceived as a spiral motion. Our aim is to increase the spiral motion (increased spiral motion is shown as more outwards extension of rod radius). This motion is shown in the spatial-temporal slice in Figure 4.8. Hand design filter-based methods [26], [27], [31], [32], [30] produce ringing artifacts around the rod (*visible as white edges around the rod and white spikes in the temporal slices in Figure 4.9* ). Ringing artifacts are produced due to phase ambiguity [49]. Oh *et al.* can magnify the motion but produce some distortions. Its separation of shape information from texture is not efficient and sometimes it results in distorted intermediate features which produce unwanted flickering (visible as white spikes in Figure 4.9 temporal slice). Whereas [47] has demonstrated favorable results, but it suffers from significant computational complexity. Instead, the proposed NAS method is able to find architectures that highlight the motion of interest with less distortions and computational

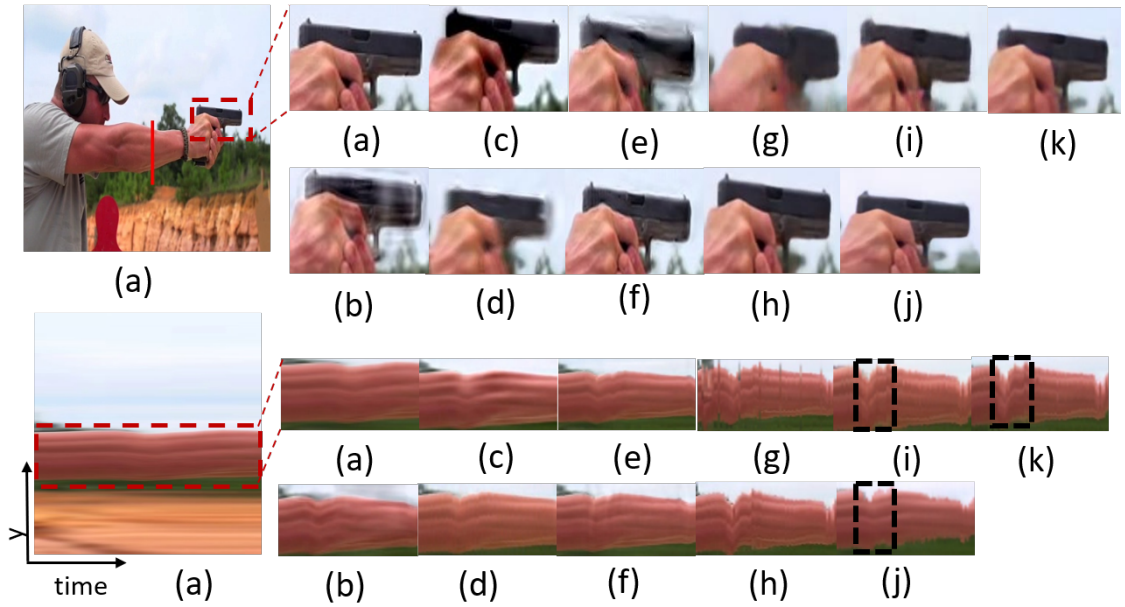


Figure 4.8: Gun-shooting video: Visualizing the impact of gun recoil through the arm. First, video frames are shown, and in the next row, the temporal slices taken from the red strip are illustrated. The visual comparison is done between (a) Input, (b) Phase-based [26] (c) Eulerian [27] (d) Acceleration based method [30], (e) Jerk-Aware method [31], (f) Anisotropy method [32], (g) Oh *et al.* [29], (h) STBVMM [47], (i) Our $SM1$, (j) Ours Teacher model and (k) Ours $SM2$ method output video frames and temporal slices are shown respectively.

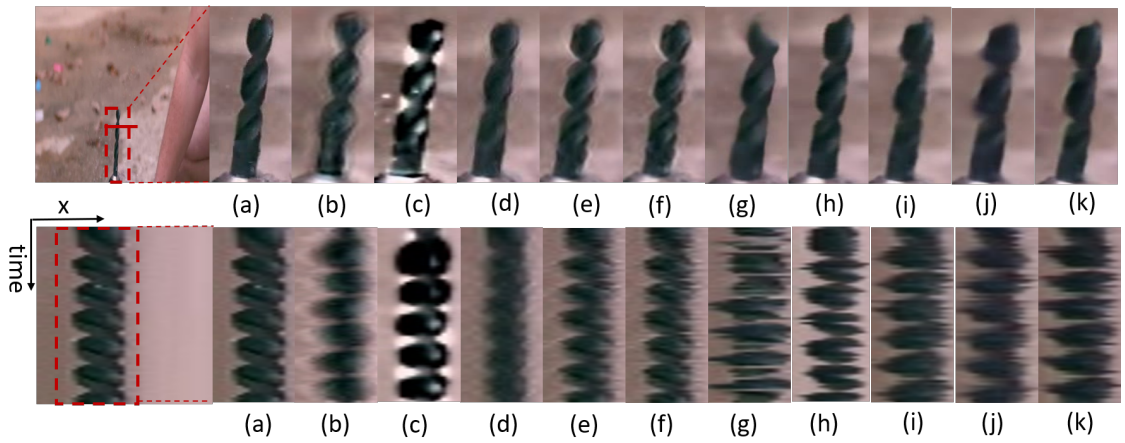


Figure 4.9: Hand Drill video: Visualizing the rotational motion of drill rod. In the first row, frames from the video depicting the drill rod are shown, and in the next row, the spatial temporal slices at the red strip are presented. The visual comparison is done between (a) Input, (b) Phase-based [26] (c) Eulerian [27] (d) Acceleration based method [30], (e) Jerk-Aware method [31], (f) Anisotropy method [32], (g) Oh *et al.* [29], (h) STBVMM [47], (i) Our $SM1$, (j) Ours Teacher model and (k) Ours $SM2$ method output video frames and temporal slices are shown respectively.

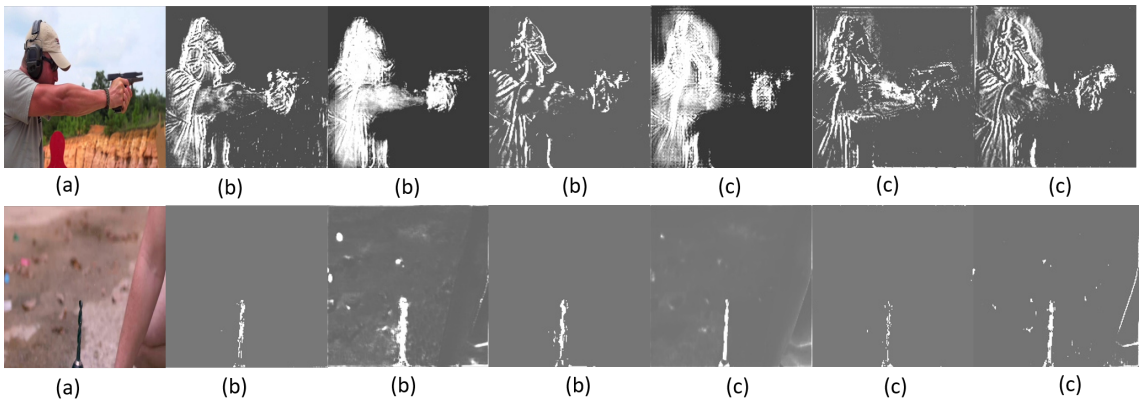


Figure 4.10: For (a) input frame, features of (b) Ours $SM1$, (c) Ours $SM2$ search student model are illustrated. These features highlight the areas which closely resemble the motion parts.

complexity as shown in Figure 4.10.

### 4.2.2 Physical Accuracy of the model:

**Is our model's output is physically accurate or not?** To answer this, we experiment with set-up as shown in Figure 4.11 and compared video magnified signal with ultrasonic sensor amplified signal. A mechanical rod is displaced from its position and an ultrasonic sensor is used to measure the signal. Video of the mechanical rod is magnified and the rod displacement signal is extracted and compared with the ultrasonic sensor signal as shown in Figure 4.11 (for more details please see section 3.3.2). Both the sensor measured and the computed magnified signal are rescaled from 0 to 1 and mean absolute error (MAE) values are calculated across SOTA methods. Distortion produced by other methods (ringing artifacts, flickering motion *etc*) leads to more error while measuring as compared to ours (illustrated in Figure 4.12).

### 4.2.3 Quantitative Analysis

**Quantitative Analysis on Synthetic Videos** To assess proposed networks, synthetic videos are used. Twenty-five videos with varied backgrounds are generated, with their average Mean Squared Error (MSE) plotted in Figure 4.13. Circles with a 40-pixel radius mimic subtle motions, including horizontal, vertical, and diagonal movements. Sub-pixel motion is simulated via up-sampling, displacement, and Gaussian noise addition. Each method's magnification aligns with ground truth, revealing noise robustness (for more details please see section 3.2.3). The magnification factor of different methods is changed to produce the same amount of motion as in ground truth. This helps in analyzing how individual methods [31], [32], [30], [29], [47] magnify different motions in separate environments and their robustness towards the noise. Each method needs to produce $100X$ magnification as compared to the input video, to match the ground truth. With

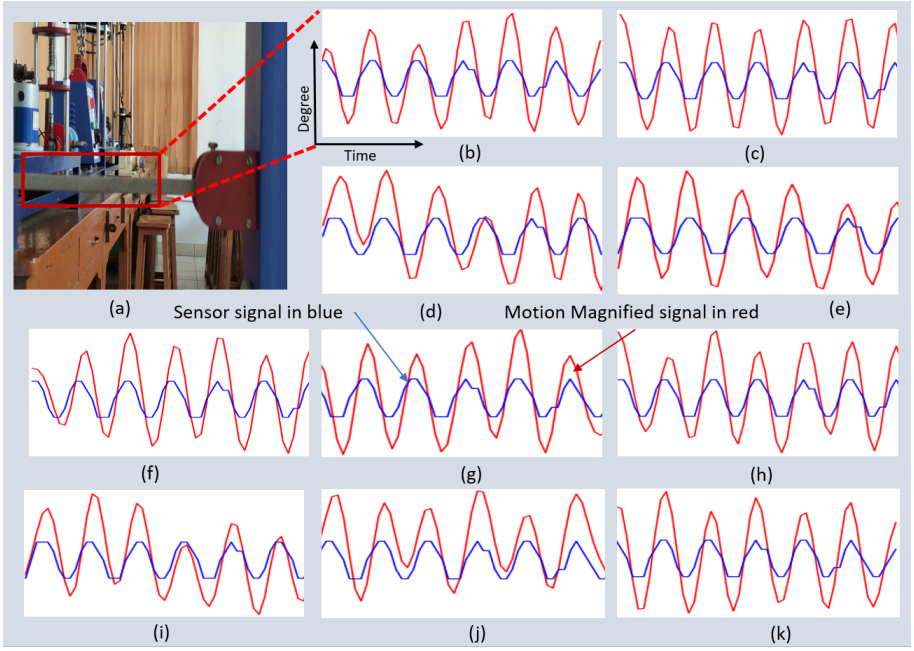


Figure 4.11: Physical Accuracy: Comparison between our method and other SOTA methods output (in red) with the sensor signal(in blue) for (a) input, (b) $SM1$, (c) $SM2$, (d) Oh *et al.* [29], (e) Jerk-aware [31], (f) STBVMM [47], (g) Acceleration [30], (h) Anisotropy [32], (i) Euler [27], (j) Phase Based [26] and (k) Teacher model ($TM$). The optical flow across the input frame and the magnified frame(of respective methods) is computed to extract the motion signal. Then the average direction along the image patch (marked in the bounding box in(a)) is calculated and shown above.

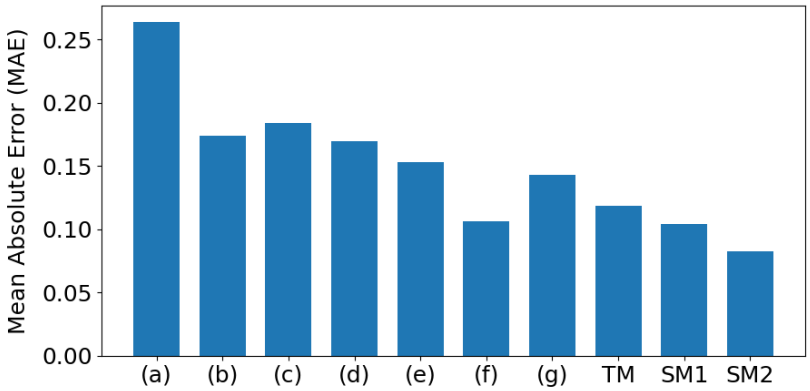


Figure 4.12: Mean Absolute Error (MAE) is computed between the extracted signal from magnified video and sensor measured signal. The error values of SOTA methods (a) Phase Based [26], (b) Euler [27], (c) Oh *et al.* [29], (d) Acceleration method [30], (e) Jerk-aware [31], (f) Anisotropy [32], (g) STBVMM [47], and the proposed method $SM1$, $SM2$ and $TM$ are shown

.

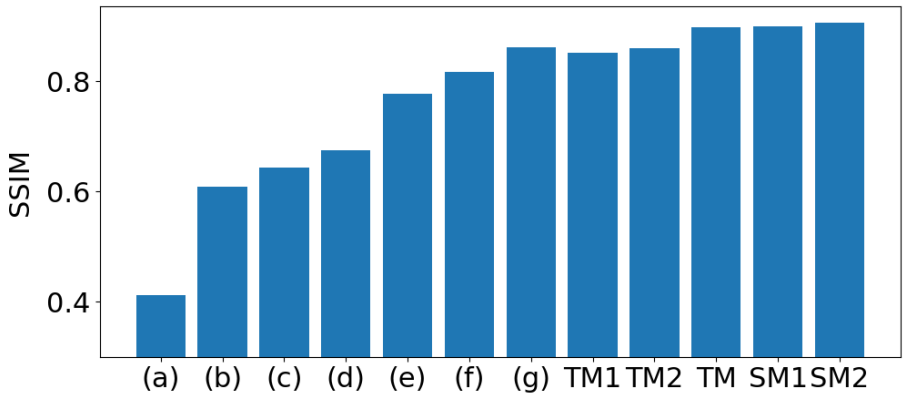


Figure 4.13: For comparision between (a) Eulerian [27], (b) Phase-based [26], (c) Acceleration based method [30], (d) Jerk-Aware method [31], (e) Oh *et al.* [29], (f) Anisotropy method [32], (g) STBVMM [47], our Teacher model $TM1$, $TM2$, $TM$, Ours searched student model $SM1$, and $SM2$, average SSIM values are computed on 25 synthetically generated videos with different backgrounds, containing subtle motion of circles.

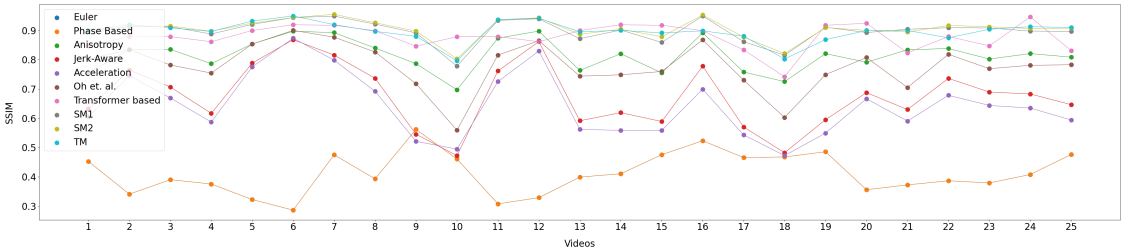


Figure 4.14: SSIM values on Euler method [27], Phase based method [26], Anisotropy method [32], Jerk-aware method [31], Acceleration method [30], Oh *et al.* method [29], Transformer based method [47], teacher model, $SM1$ and $SM2$ on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

such a large magnification, the effects of distortions become more apparent and lead to degraded output. Figure 4.13 shows that the proposed method searched networks have the highest SSIM values for the same amount of output motion as compared to SOTA methods. Further, different teacher networks ($TM1, TM2$) with lesser latency are trained for comparison and, their specification and results are shown in Table 7.1. From Figure 4.13, it can be seen that the searched networks are more efficient.

## 4.3 Additional Experiments

**Frequency selectivity:** Deep learning methods lack direct training with temporal filters, so applying temporal filters to intermediate features can result in inaccuracies [29]. To address this challenge, we advocate for an initial pre-processing step involving temporal filtering to mitigate unwanted motion artifacts. Drawing inspiration from [26], we utilize the output obtained at a conservative magnification factor (magnification factor=4) as input to our method. As depicted in Figure 4.15, our approach emphasizes relevant

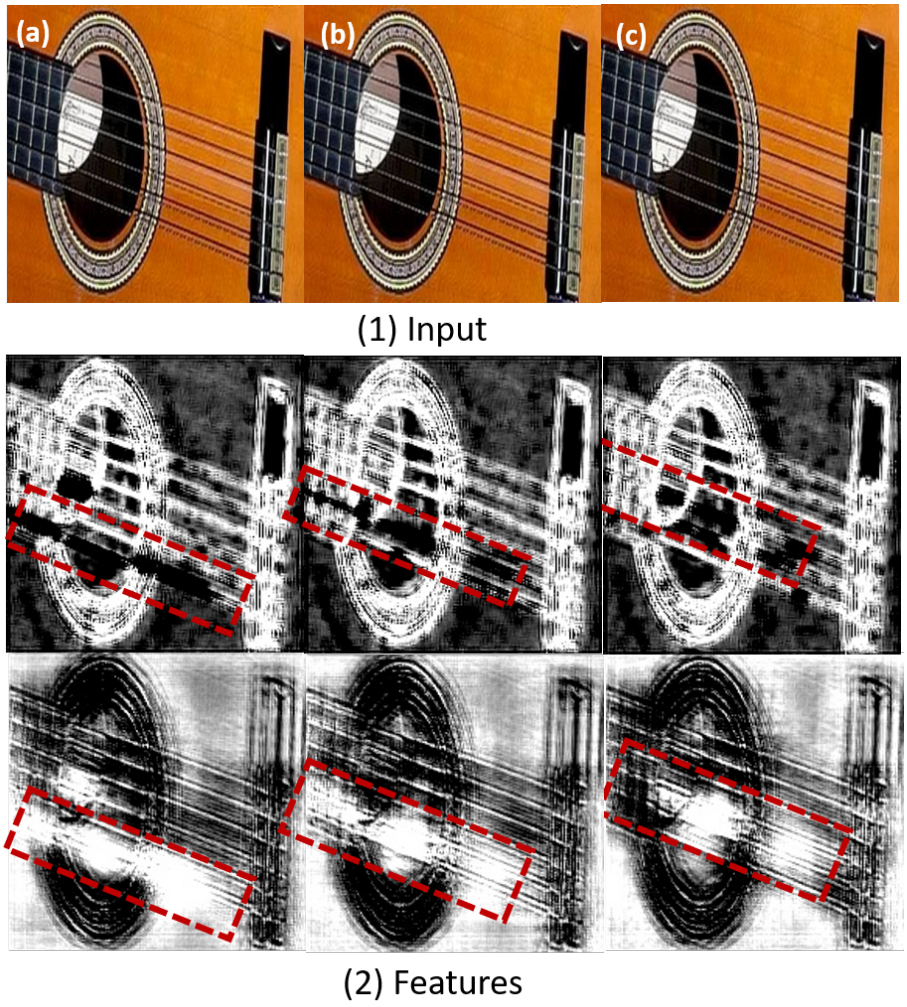


Figure 4.15: Effects of temporal filter: We first pre-processed the video with the temporal filter to suppress unwanted motion. For this, [26] method's output at a small magnification factor ( magnification factor=4) is given as an input to the searched student network $SM1$. Intermediate features (row-2,3) of the searched student model, highlighting the motion in the red bounding box. (a), (b) and (c) highlight the motion parts E-string (80Hz), A-string(108Hz), and D-string (144Hz) of the guitar.

motion characteristics within intermediate features. Notably, the variability in temporal filter inputs leads to the highlighting of distinct motion attributes.

**Visual effects on change in magnification factor:** Visual effects of increasing magnification factor vary across video analysis methods. The Acceleration method exhibits distortions, particularly in dynamic scenarios (Figure 3.15 [30]). The Anisotropy method shows subtle magnification changes with distortions, especially in dynamic settings (Figure 3.16 [32]). The Jerk-Aware method demonstrates minimal magnification alterations with significant distortions, particularly in dynamic scenes (Figure 3.17 [31]). The Phase-based method reveals substantial distortions in dynamic scenarios and ringing artifacts in static scenes (Figure 4.16 [26]). Oh *et al.*'s method provides enhanced magnification but introduces unwanted motion and blurry distortions, worsening with

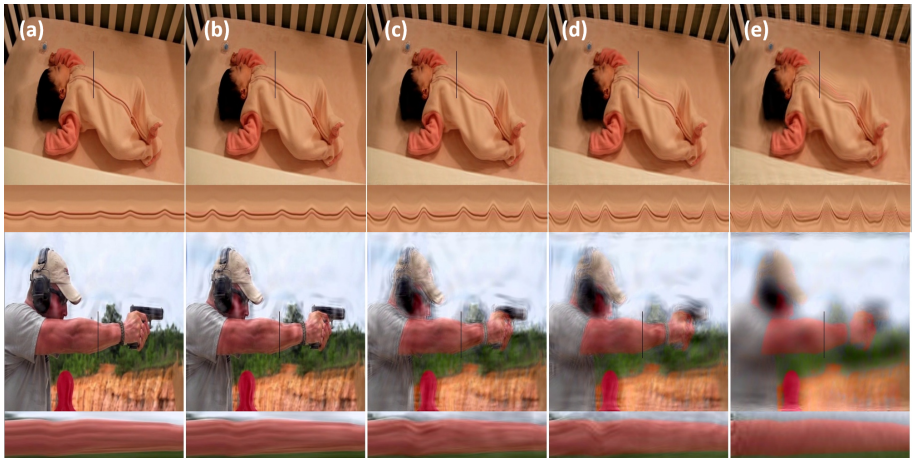


Figure 4.16: Effects of change in Magnification Factor: Figure illustrates Phase based method [26] output. Different values of magnification factor in increasing order from (a) 10, (b) 20, (c) 40, (d) 60, and (e) 100 for baby video and (a) 1, (b) 2, (c) 5, (d) 10, and (e) 100 for gun video are used to generate the output shown in the respective column. The linear methods are not suitable for dynamic scenarios, as they are unable to ignore dynamic motion. So, they produce large distortions in the gun video (dynamic scenarios). Whereas in static scenario (baby videos), with an increase in magnification factor, there is an increment in both, the amount of magnification and ringing artifacts (visible as lines overlapping the edges of moving objects) in the static scenario.

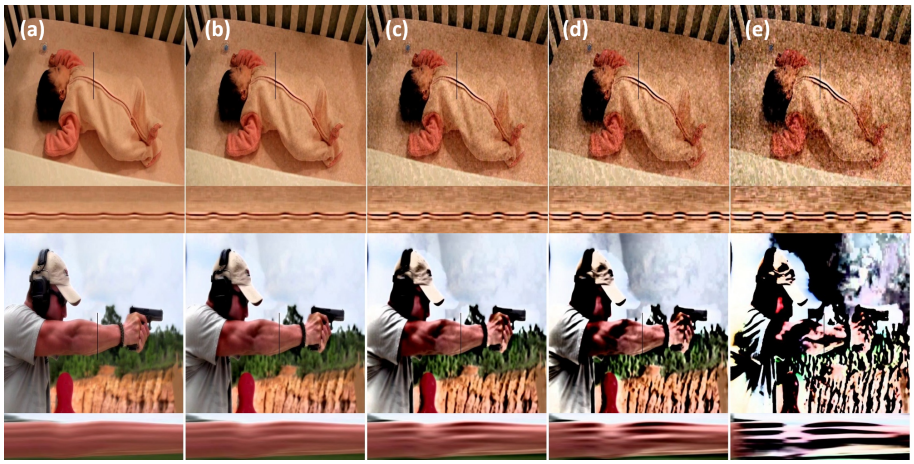


Figure 4.17: Effects of change in Magnification Factor: Figure illustrates Euler based method [27] output. Different values of magnification factor in increasing order from (a) 10, (b) 20, (c) 40, (d) 60, and (e) 100 for baby video and (a) 1, (b) 2, (c) 5, (d) 10, and (e) 100 for gun video are used to generate the output shown in the respective column. The linear methods are not suitable for dynamic scenarios, as they are unable to ignore dynamic motion. So, they produce large distortions in a gun video (dynamic scenarios).

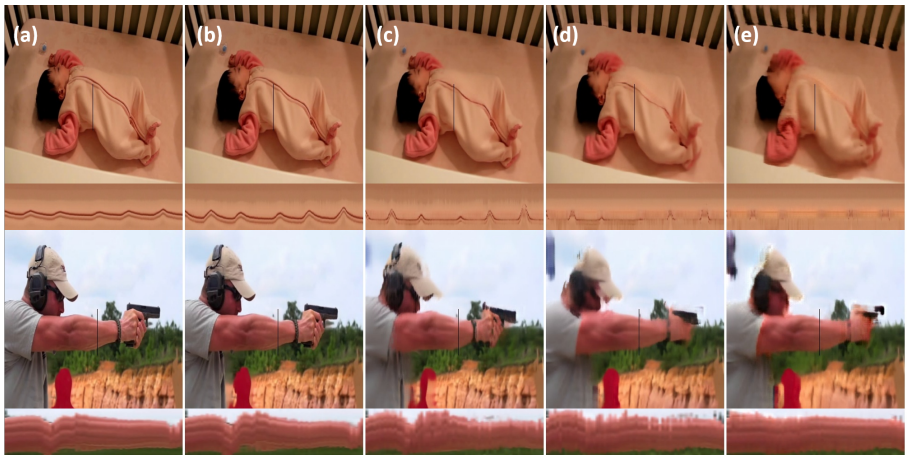


Figure 4.18: Effects of change in Magnification Factor: Figure illustrates STBVMM [47] method output. Different values of magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate output for both videos. It produces good magnification, (both in static and dynamic scenarios), but it starts to produce some blurry distortions at higher magnification factors.

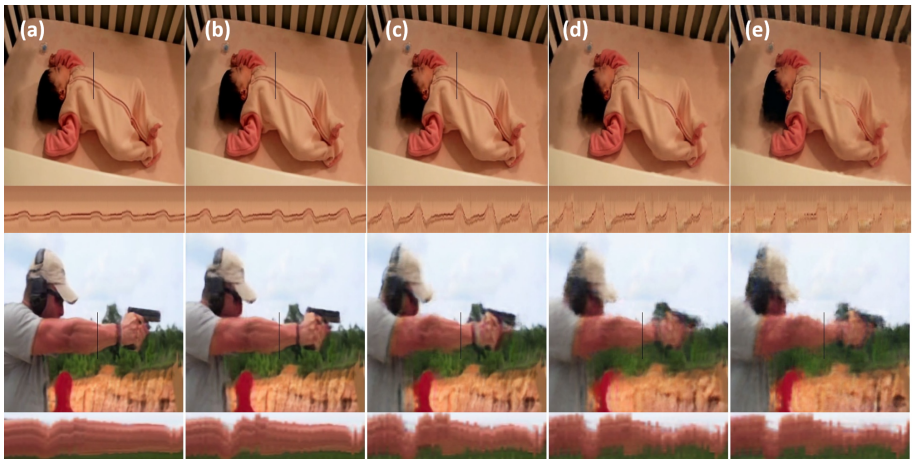


Figure 4.19: Effects of change in Magnification Factor: Figure illustrates $SM1$ model output. Different values of magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate output for both videos. $SM1$ shows fewer distortions while increasing the amount of magnification as compared to other SOTA methods, both in static and dynamic scenarios.

higher magnification (Figure 4.18 [29]). Conversely, the Base model ($M_1$) shows fewer distortions with increasing magnification, while the lightweight model ($M_2$) suffers performance degradation, particularly at extreme magnification levels due to reduced parameterization (Figure 7.12, Figure 7.13).

### 4.3.1 Ablation Study

**Effect of appearance encoder $A(.)$ and manipulator $M(.)$ in magnification:** To analyze the effects, the appearance encoder and manipulator, are removed one at a time in $SM1$ and trained. The appearance encoder helps to get a common appearance

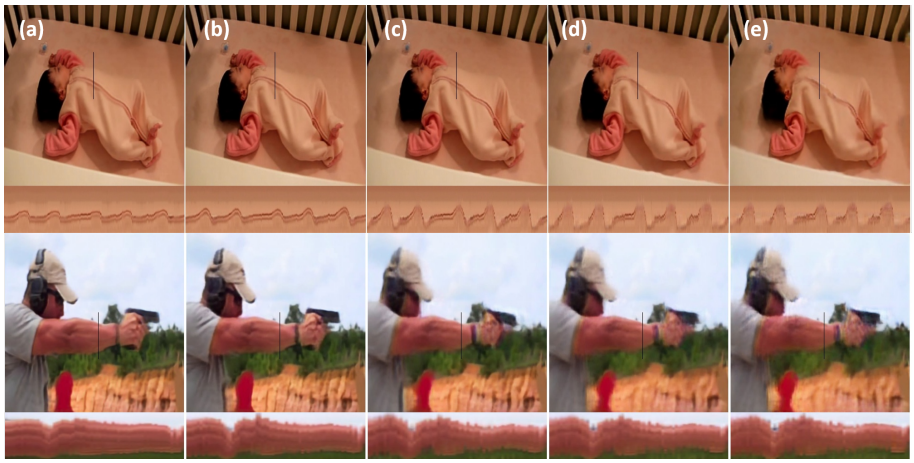


Figure 4.20: Effects of change in Magnification Factor: Figure illustrates our $SM2$ model output. Different values of magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate output for both videos.

Table 4.3: Different configurations of $SM1$ for ablation study. (Figure 4.6, S 4.7 represents the architectures of searched student model $SM1, SM2$ and Figure 4.21, 4.22, 4.23, 4.24, 4.25, 4.26, 4.27 shows different architectures of student model $SM1$ used in ablation study.)

| **Model** | $M(.)$ | $A(.)$ | $KD_{loss}$ | $L_{mag}$ | $LR$ | $LC$ | $L_{lat}(\beta)$ |
|---|---|---|---|---|---|---|---|
| $SM1-1$ | × | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |
| $SM1-2$ | ✓ | × | ✓ | ✓ | ✓ | ✓ | ✓ |
| $SM1-3$ | ✓ | ✓ | × | ✓ | ✓ | ✓ | ✓ |
| $SM1-4$ | ✓ | ✓ | ✓ | × | ✓ | ✓ | ✓ |
| $SM1-5$ | ✓ | ✓ | ✓ | ✓ | × | ✓ | ✓ |
| $SM1-6$ | ✓ | ✓ | ✓ | ✓ | ✓ | × | ✓ |
| $SM1-7$ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | × |
| $SM1$ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

and the manipulator helps in reducing the unwanted effects of noise in different feature maps. Their output is tested on the synthetic dataset and is shown in Table 4.3 and Figure 4.13. Further, Figure 4.30 showcases the SSIM values of the teacher model with and without feature sharing encoding and appearance encoder, revealing a decrease in performance without these components. Figure 4.31 compares the average SSIM values on synthetically generated videos with subtle motion circles, demonstrating the impact of color perturbation versus appearance encoder on [29]. Lastly, Figure 4.32 illustrates the effects of motion on appearance encoder features, highlighting increased blur in regions of motion as the input frame changes. Even without them deep learning can find reasonable good weights, but they are not that much efficient.

**How does $L_{mag}^{KD}$ and $L^{KD}$ effects?** To see the effects of $L_{mag}^{KD}$ and $L^{KD}$, two different architecture searches are done. One is without $L^{KD}$ and second is without $L_{mag}^{KD}$. They are

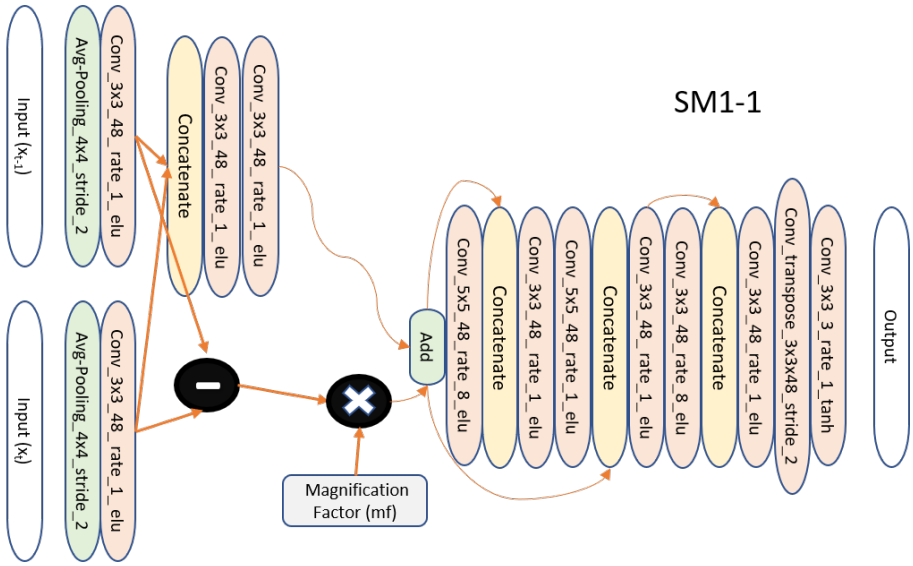


Figure 4.21: Architecture of student model $SM1-1$ (without $M(.)$).

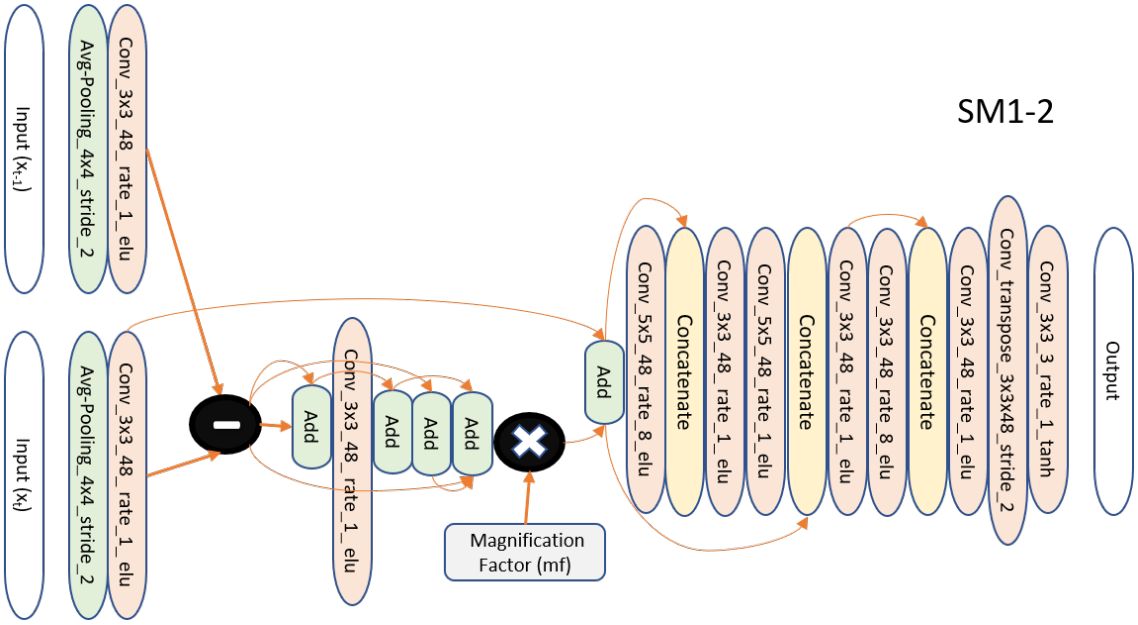


Figure 4.22: Architecture of student model $SM1-2$ (without $A(.)$).

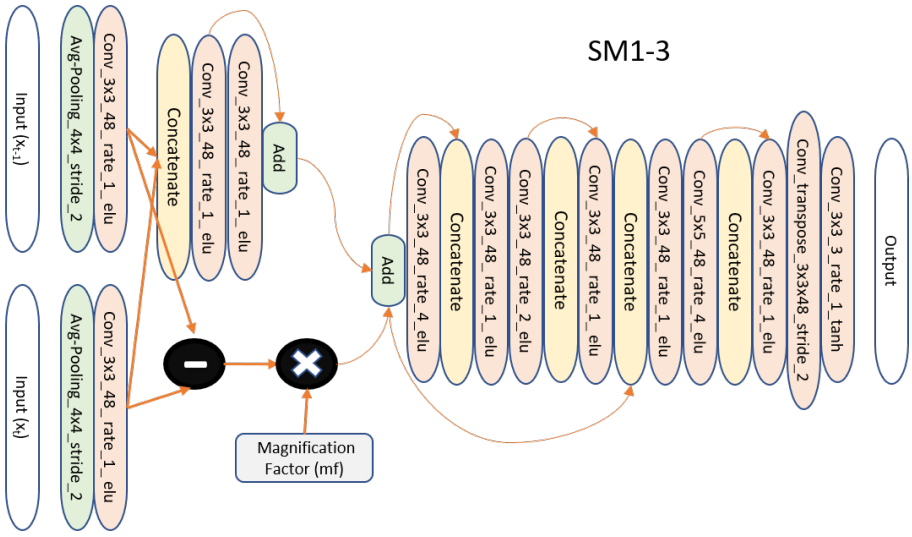


Figure 4.23: Architecture of search student model $SM1-3$ (without $KD_{loss}$).

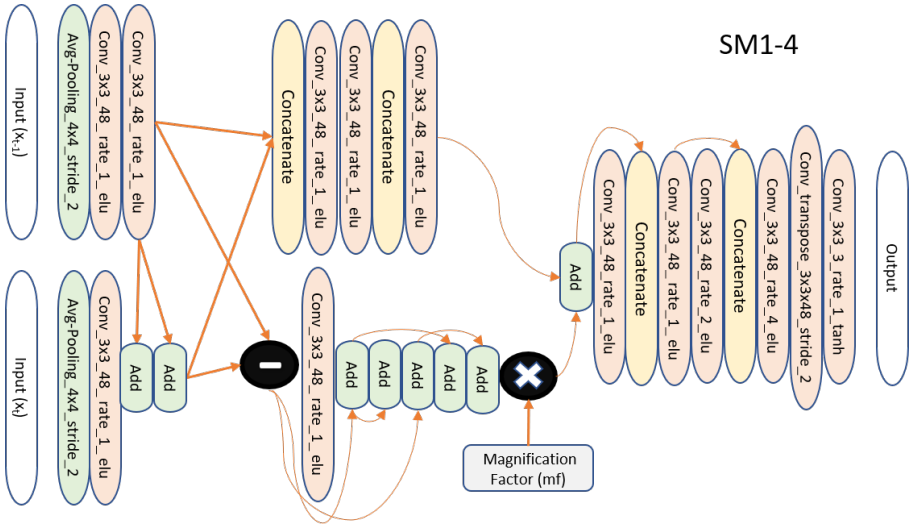


Figure 4.24: Architecture of search student model $SM1-4$ (without $L_{mag}$).

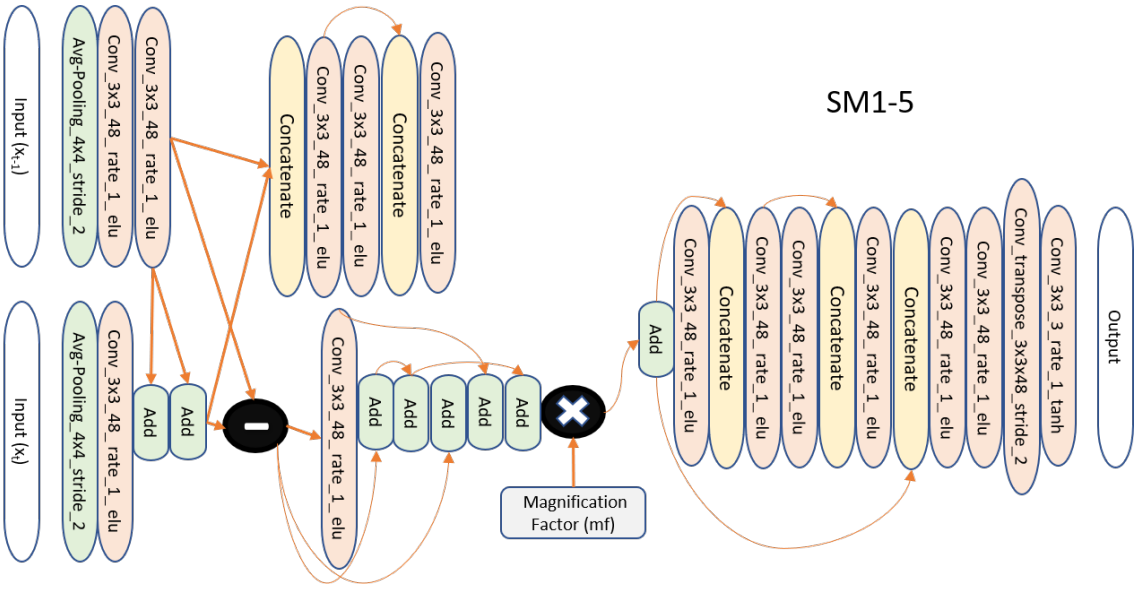


Figure 4.25: Architecture of search student model $SM1-5$ (without $LR$).

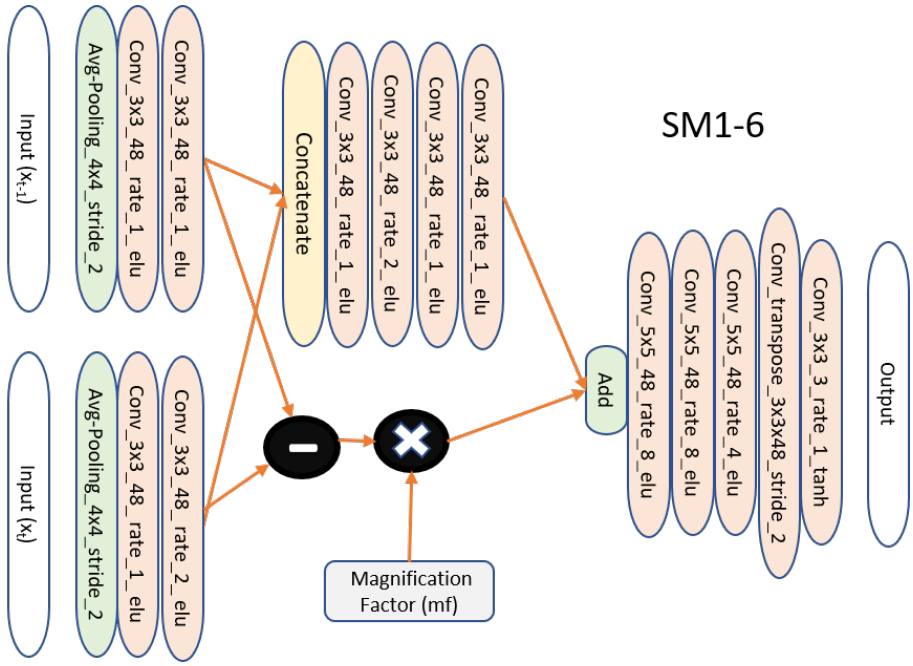


Figure 4.26: Architecture of search student model $SM1-6$ (without $LC$).

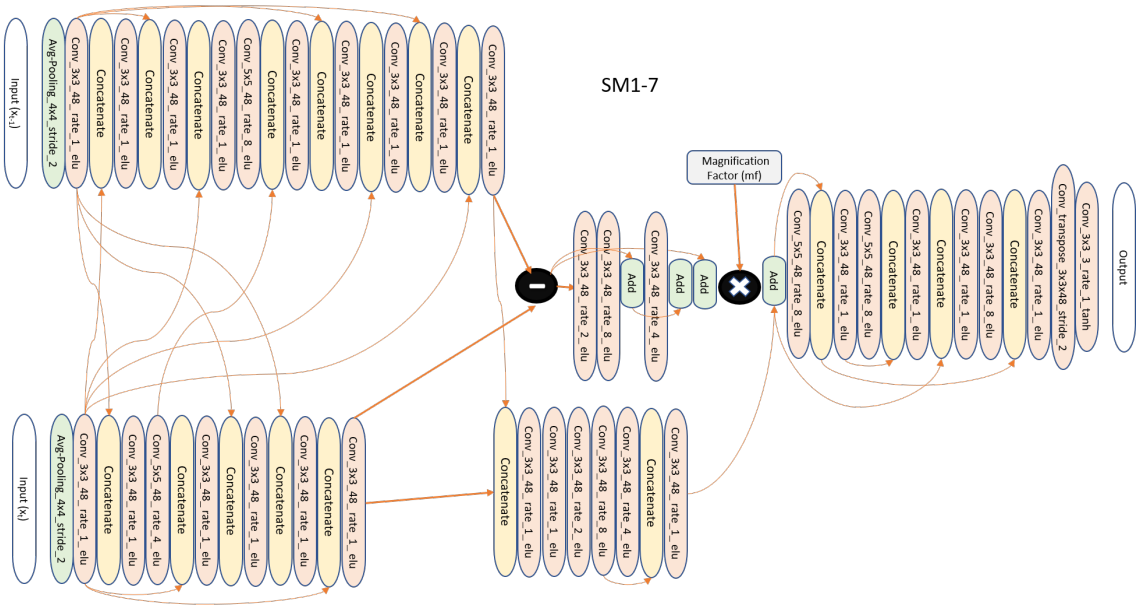


Figure 4.27: Architecture searched without $L_{lat}(\beta)$

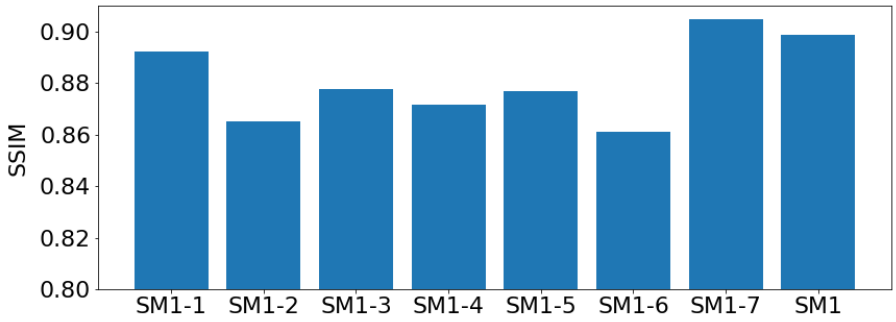


Figure 4.28: For comparison between Our searched student networks $SM1$, and $SM1-(1to7)$ (as mentioned in Table 4.3) average ssim values are computed on 25 synthetically generated videos with different backgrounds, containing subtle motion of circles.

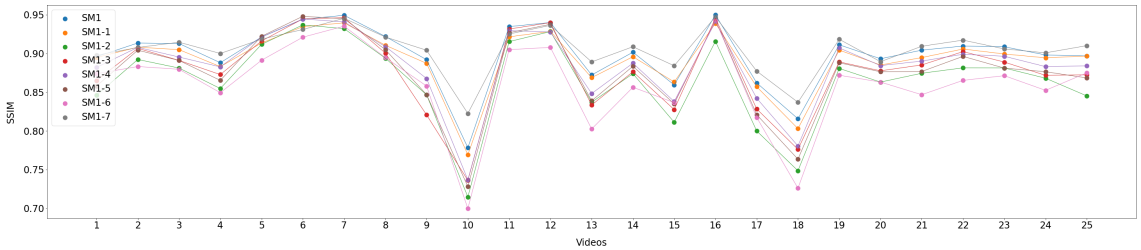


Figure 4.29: SSIM values on $SM1$, and $SM1-(1to7)$ method, on 25 synthetically generated videos containing different subtle motions of circles with various backgrounds.

used to search the $SM1$. $L^{KD}$ forces denoising characteristics and $L^{KD}_{mag}$ helps in better reconstruction. So, without either of them, there is a decrease in performance. Results of the searched model evaluated on the synthetic dataset are shown in Table 4.3 and Figure 4.28. Figure 4.33 demonstrates the impact of varying the hyperparameter $\lambda_4$ on SSIM values. Their is a trade off between $L^{KD}_{mag}$ and $L^{KD}$. $L^{KD}$ forces denoising characteristics and $L^{KD}_{mag}$ helps in better reconstruction. So, without either of them, there is a decrease in performance. We can control their effects with the help of hyperparameter $\lambda_4$. As there is an increase in value of $\lambda_4$, it puts more emphasis on noise free pseudo signal from $L^{KD}$ loss, but after a particular point denoising characteristics can interfere with good reconstruction e.g. if the model is learning to blur for denoising, then more blur will reduce the sharpness and output quality. Higher values emphasize noise-free signals from $L^{KD}$, but excessive denoising can degrade reconstruction quality.

**Effects of proposed layers in architecture search:** To verify the effects of $LR$ and $LC$ in architecture search, two different search spaces are used. Without $LR$, the receptive field is constrained to only one `3x3` convolution layer with rate 1 and `no connection`. This helps to study the effect of different receptive fields in architecture search. Secondly, $LC$ is removed from the search space to highlight the effects of different features in the search space. Both of these help in finding better layers, and while constraining them there is a decrease in the performance of the searched

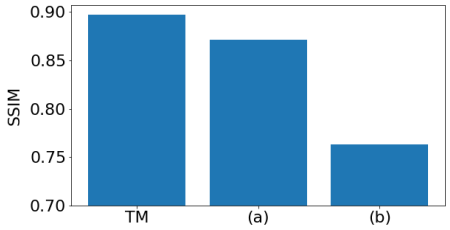


Figure 4.30: Teacher Model Ablation: SSIM values of (a) Teacher model, (b) teacher model without feature sharing encoding (c) teacher model without appearance encoder. With either of them, there is a decrease in the performance of the teacher model. SSIM values are computed across the synthetic dataset.

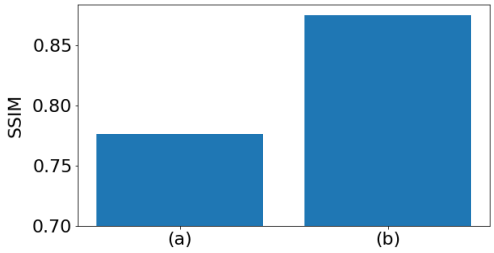


Figure 4.31: Color perturbation vs appearance encoder: Average SSIM values on 25 synthetically generated videos with different backgrounds, containing subtle motion of circles on (a) Oh *et al.* [29], (b) Oh *et al.* [29] with appearance encoder instead of color perturbation.

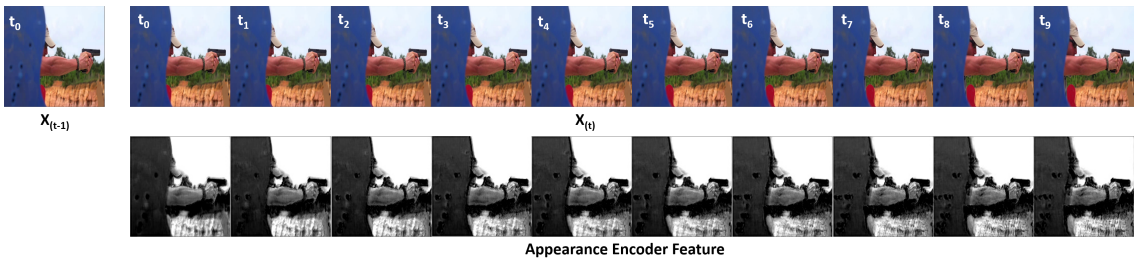


Figure 4.32: Effects of Motion on appearance encoder features: The features of the appearance encoder, as $X_{t-1}$ input is fixed to frame $t_0$ and $X_t$ input is changed from frame $t_0$ to $t_9$ lead to more blur on the regions of motion. (Please zoom in for a better view ).

model. Their evaluated results are shown in Table 4.3 and Figure 4.28. Further,, Figure 4.35 and 4.34 shows the internal weights of operations (`3X3 Conv with rate, r` $\in (1, 2, 4, 8)$ `5X5 Conv with rate, r` $\in (1, 2, 4, 8)$ `and no connection`) in the layer $LR$ (highlighted in red box in Figure 4.35) and operations ( `No connection, Add, Concate(.)`) in the layer $LC$ (highlighted in red box in Figure 4.34), respectively. In Figure 4.34, first, relation between $NASLayer(R_3)$ and $NASLayer(R_2)$ output features are found and their weights are shown in Figure 4.34 (b). It can be represented as $LC(NASLayer(R_3), NASLayer(R_2)) = Feature_1$. Then the relation between $Feature_1$ and $NASLayer(R_1)$ output features are found and their weights are shown in Figure 4.34

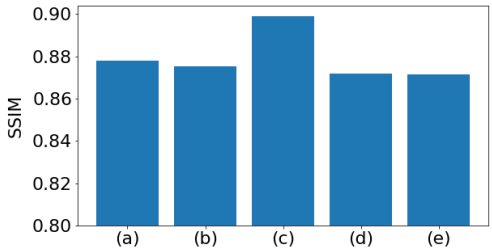


Figure 4.33: Effects of change in lambda ($\lambda_4$) with respect to SSIM values: A trade-off exists between $L^{KD}mag$ and $L^{KD}$: the former aids in reconstruction while the latter enforces denoising characteristics. Adjusting $\lambda 4$ allows control over these effects. Model are searched with different values of $\lambda$ as (a) $\lambda_4$=0.0, (without $L^{KD}$) (b) $\lambda_4$=0.01, (c) $\lambda_4$=0.1, (d) $\lambda_4$=1.0, and (e) without $L^{KD}_{mag}$ loss.

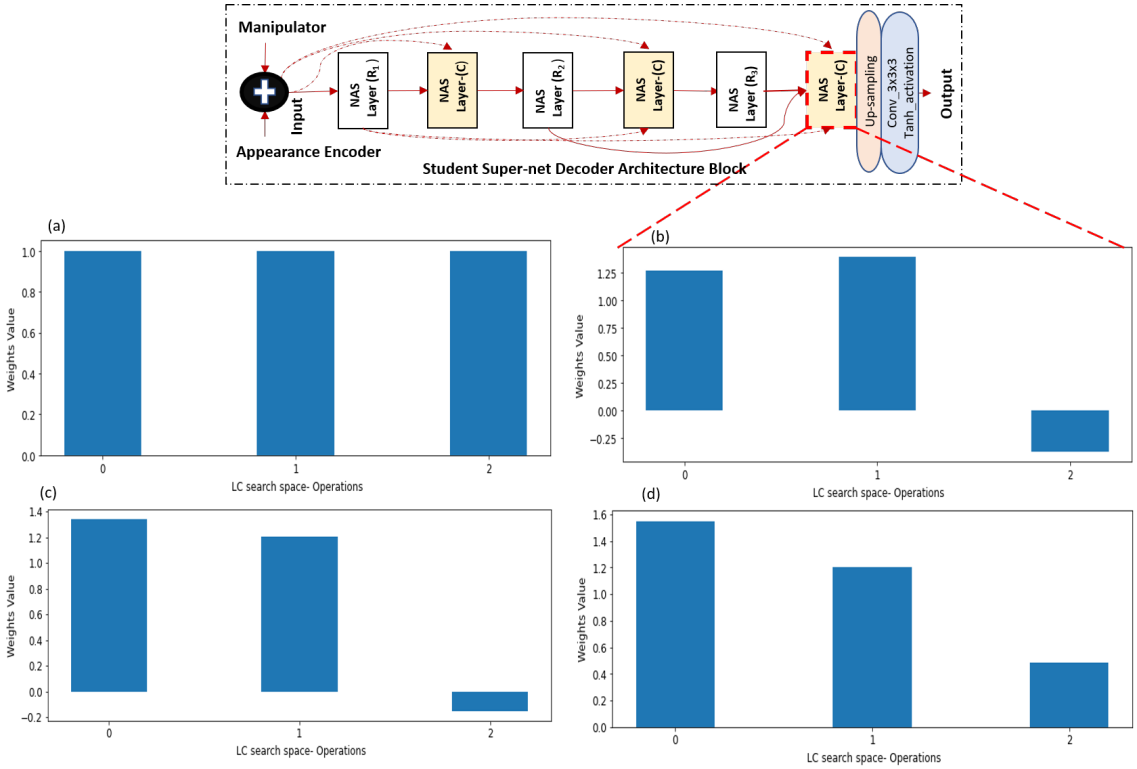


Figure 4.34: Supernet model weights for operation selection: (a) shows the initial weights and (b), (c), (d) shows the learned weights of operations after the training of $NASLayer-(C)$, highlighted in the red box. Operations `(0)` $--> $ `No Connection,` `(1)` $-->$ `Concate(.),` `(2)` $-->$ `Add` are used in $NASLayer-(C)$.

(c). It can be represented as $LC(Feature_1, NASLayer(R_1)) = Feature_2$. In last, the relation between $Feature_2$ and $Input$ features are found and their weights are shown in Figure 4.34 (d). It can be represented as $LC(Feature_2, Input) = Feature_3$.

**Effects of $L_{lat}(\beta)$:** To study the effects of latency constraints in the architecture search, a model is trained without latency loss. Its evaluated results are shown in Table 4.3 and Figure 4.28. Table 7.1 shows its memory requirement and latency. We consider without latency loss $SM1-7$ SSIM values as the highest values the searched network can achieve. As, the latency values used in $L_{lat}(\beta)$ increases, SSIM values are improved. Figure 4.36

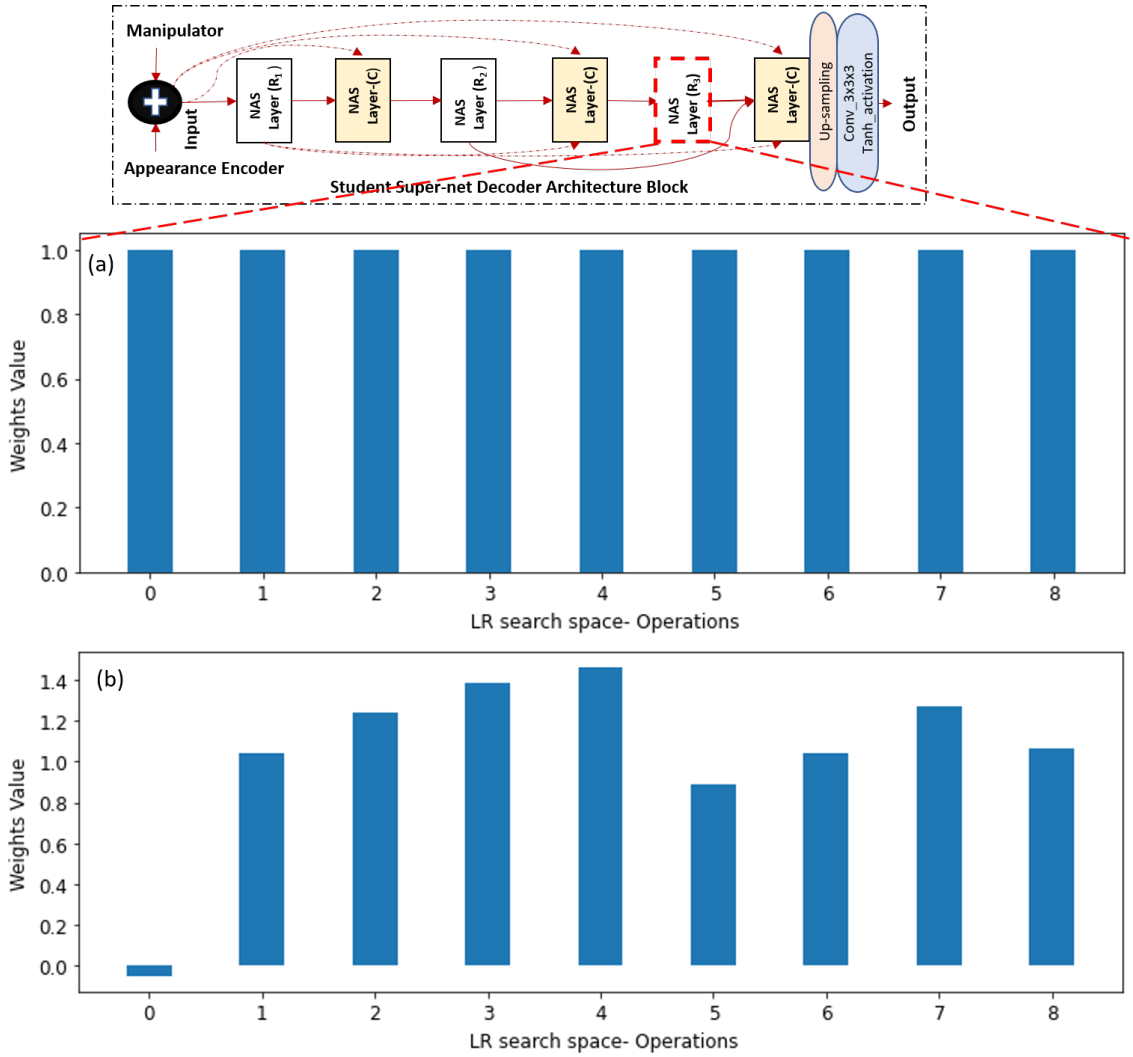


Figure 4.35: Supernet model weights for operation selection: (a) shows the initial weights and (b) shows the learned weights of operations after the training of $NASLayer - (R_3)$ highlighted in the red box. Operations `(0) -- > No Connection, (1) -- > Concate(.), (2) -- > Add (3) -- > 3X3 Conv with rate 4, (4) -- > 3X3 Conv with rate 8, (5) -- > 5X5 Conv with rate 1, (6) -- > 5X5 Conv with rate 2, (7) -- > 5X5 Conv with rate 4, (8) -- > 5X5 Conv with rate 8` are used in $NASLayer - (R)$.

illustrates the effects of change in latency with respect to SSIM values. But values between $SM2$ and $SM1-7$ are pretty comparable, and it seems beyond $SM2$ model latency, the improvement in SSIM values is stagnated. From the searched architectures ($SM1$, $SM2$, and $SM1-7$), a clear trend emerges in the reduction of layers in encoder blocks. We postulate that unlike [29], where the encoder separates shape and texture features, our encoder is utilized to reduce the effect of noise before magnification. As the output quality relies on texture synthesis, which occurs on the decoder side, the training algorithm places less emphasis on the encoder. Furthermore, we observe a general trend of selecting higher receptive fields with deep networks when the latency constraints are loosened.

**Effects of Teacher Model on student network search:** Figure 4.37 demonstrates the

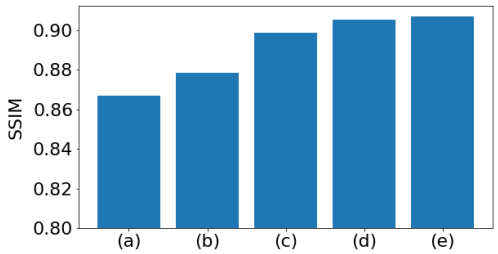


Figure 4.36: Effects of change in latency with respect to SSIM values: (a), (b), (c), (d) are models with latency constrain of 15, 18, 23, 25 respectively, and (e) is a model without latency constrain. As, the last point (e) is computed without latency loss, assuming it represents the highest SSIM value achievable by the NAS algorithm in searching the student network. Near the highest latency point, SSIM improves less with further increases in latency, indicating a saturation region. Conversely, at the lower latency end, SSIM values start to decrease non-linearly.

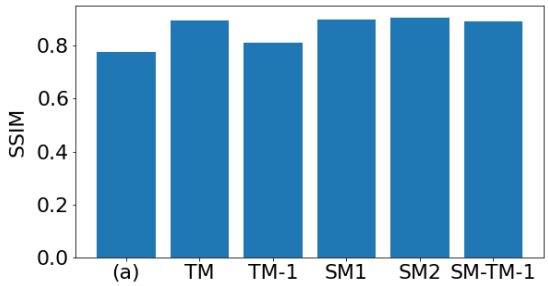


Figure 4.37: Effects of Teacher Model with lesser parameters on student network search: SSIM values of (a) Oh *et al.* [29], teacher model ($TM$), teacher model with lesser parameters ($TM-1$, searched student model ($SM1$), ($SM2$) and searched student model ($SM-TM-1$) from $TM-1$. SSIM values are computed across the synthetic dataset. The proposed NAS method searched student model works better as compared to the [29].

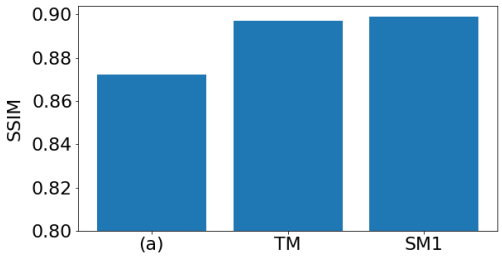


Figure 4.38: Effects of Teacher Model trained on noisy data: (a) Searched student model from teacher model trained on noisy data, $TM$ teacher model trained on noise free data, and the $SM1$ student model searched from $TM$.

impact of a teacher model with reduced parameters on the search for a student network. It compares SSIM values of various models, including the teacher model ($TM$), a teacher model with fewer parameters ($TM-1$, with 0.645 M parameters and 211.8 GFLOPs), two searched student models ($SM1$ and $SM2$), and a student model derived from the teacher model with fewer parameters ($SM-TM-1$) having 0.455M parameters and 118.2 GFLOPs from $TM-1$. The NAS method's searched student model performs better

compared to the reference model. In Figure 4.38, the effects of a teacher model trained on noisy data versus noise-free data on the performance of a searched student model are explored.

## 4.4 Summary

In this chapter, we discussed a novel Knowledge Distillation based Latency aware-Differential Architecture Search (KL-DNAS) method to search latency constrained student models under teacher network supervision for video motion magnification. Using the teacher model, the student models are searched by parts with a similar structure as the teacher model and improved denoising characteristics. Layer wise differential architecture search is used to find student networks. NAS Layer-(R) and NAS Layer-(C) layers help to find different receptive fields and connections across features. $L_{lat}(\beta)$ is used to make the architecture search process latency aware. Two different student models are searched. The performance of the proposed networks is evaluated by qualitative and quantitative analysis with state-of-the-art methods for motion magnification. Further, an experiment is done to check the physical accuracy of the magnified videos. An extensive ablation study is conducted to analyze various modules of the framework. The results demonstrate that the two searched student models having fewer parameters than the teacher model and Oh *et al.* model, give better results than state-of-the-art methods in terms of better motion magnification with fewer distortions, blurriness, and artifacts. While our approach has demonstrated promising results with the defined constrained, there are still areas for further improvement and exploration. One potential direction is to leverage intuition from hand-designed methods utilizing complex steerable pyramids, which were employed to mitigate the effect of noise during motion magnification and integrate them with deep learning-based approaches to enhance their efficiency further.

# Chapter 5

# Multi Domain Learning Based Light-Weight Network for Video Motion Magnification

## 5.1 Introduction

The traditional technique [27] comprises of handcrafted filter-based algorithms, that rely on steerable pyramids for image decomposition and filters for motion magnification. However, it produced noisy output. Through phase variations, [26] propose a complex steerable pyramid for image decomposition and magnification. This resulted in improved magnification while lowering the effects of noise in the magnification process. However, they function poorly in scenarios containing dynamic or large motion. Two different approaches are proposed to tackle these issues: hand design filtering and deep learning models. In the first approach, authors propose hand-design filters [30], [31],[32],[42] compatible with earlier methods, to work both in static and dynamic scenarios. But, they have a small amount of magnification and are prone to ringing artifacts. The second technique is deep learning, which is based on the notion that deep convolutional networks may produce a more optimal solution [29]. Oh *et. al* [29] proposed a deep network with more magnification, compared to handcrafted methods but its solution is computationally challenging and prone to distortions, artefacts, and texture generation-related problems. We propose a phase-based deep network for video motion magnification to address these concerns. It combines the handcrafted approach of phase-based motion magnification [26] with deep learning-based spatial magnification [29] to overcome each other limitations. In addition, for real-time applications, lightweight networks $D_1$ and $D_2$ are proposed. The following are the key contributions:-

- A novel multi-domain lightweight networks ($D_1$ and $D_2$) is proposed for video motion magnification.
- A frequency domain-based motion magnification block is proposed for motion synthesis. It directly estimates the phase and amplitude changes for the magnification according to the provided magnification factor. This helps to reduce

Table 5.1: Conceptual differences between the proposed approach and existing methods for motion magnification

| Methods | Motion Manipulation | Magnified Frame Texture Generation |
|---|---|---|
| Hand-crafted Methods [30, 31, 32, 42] | Phase-Variation Linear/Non-Linear filters (Frequency Domain) | Steerable Pyramid (wavelet-based reconstruction) |
| Deep-learning Method [29] | Shape Feature Differences based learnable filters (Spatial Domain operations) | Residual blocks based simple decoder (learnable filters in spatial domain) |
| Proposed Method | Phase and Spatial Variation based learnable filters (Frequency and Spatial Domain) | Multi-scale texture Correction block (learnable filters in spatial domain) |

noise effects in the magnification process and generates motion (which depends on phase variations).

- A spatial domain-based multi-scale texture correction block is proposed to improve the texture quality. It estimates the texture component at each scale using information from input frames and magnified motion features in the spatial domain.

The proposed networks ($D_1$ and $D_2$) are evaluated qualitatively and quantitatively on real-world and synthetic videos on different tasks. Additionally, an experiment is conducted to check the physical accuracy of the proposed method. An ablation study is also conducted to see the effects of different parts of the proposed network.

## 5.2 Proposed Method

The proposed methods assume that subtle variations translated through phase changes are more robust to noise[26] . So, by manipulating the phase, subtle motion can be enhanced. In the following section, first, we discuss why phase-based motion magnification has the upper hand and the challenges associated with it (for a better explanation we adapt similar example scenarios as in [26], [49]). Then, we present the proposed solutions to overcome those challenges. Later, the loss function and other implementation details are discussed.

### 5.2.1 Motivation

We consider a 1D case to give intuition on the working and challenges associated with motion magnification through phase variation. Let $f(x)$, a 1D signal at $T = 0$. The signal at time step $T = t$ is defined as the displaced version of $f(x)$, $f(x+\delta(t))$ where $\delta(t)$ represents the displacement function (not to be confused with a Dirac function). Then the motion magnification signal is defined as $f(x + (1 + \alpha)\delta(t))$, where $\alpha$ decides the amount of magnification.

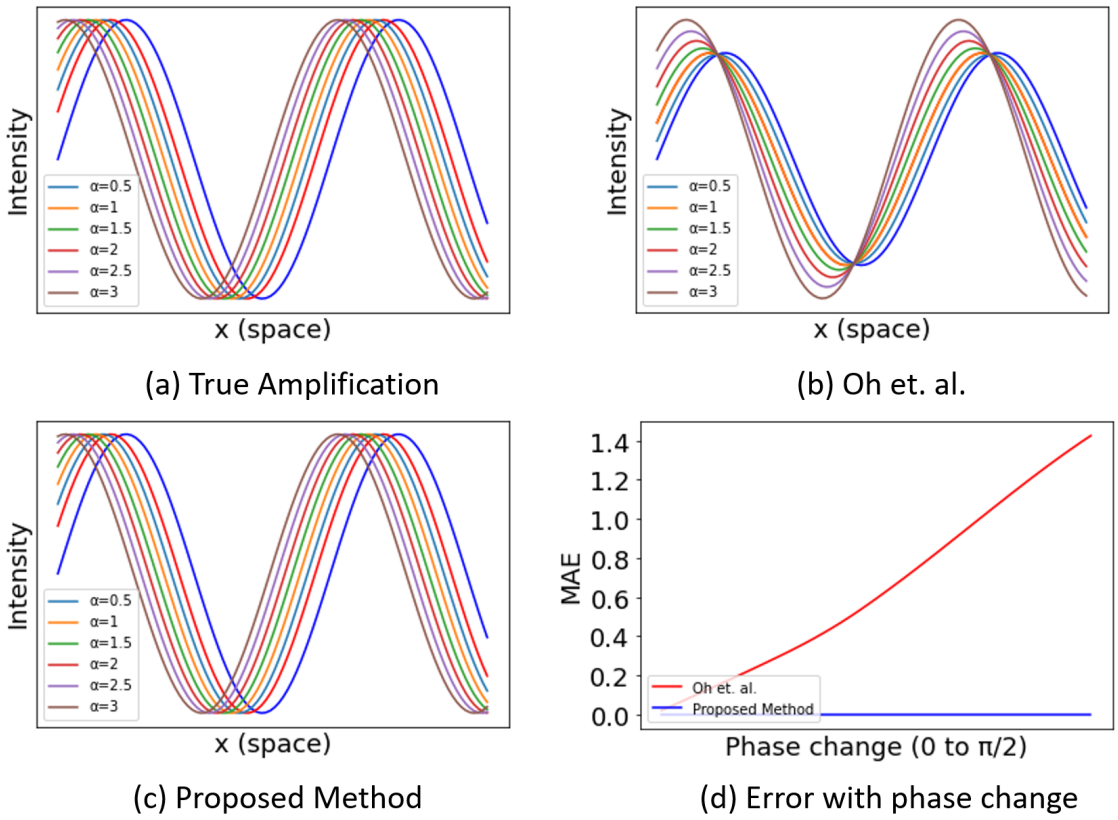


Figure 5.1: Proposed phase based motion magnification method (as illustrated in Eq (6.9) shows better output more similar to the true amplification. In comparison, the [29] 1 D simplified output (as shown in Eq (5.2), deviates more with the increase in magnification factor ($\alpha$). These results are calculated at small phase variations. The effect of change in input phase variations, for both methods, is shown at the same magnification factor in (d).

For a sinusoid wave, $y = A\sin(\omega x+\phi)$, $A$, $\omega$, $\phi$ represent its amplitude, angular frequency, and phase respectively. By setting 1 D signal as sinusoid $f(x) = A\sin(\omega x)$, we get the displaced signal $f(x+\delta(t)) = A\sin(\omega(x+\phi))\ \forall\ t \in [0,t]$ , where $\delta(t) = \omega\phi$. The magnified signal can be written in terms of phase variations of the input signal as $f(x+(1+\alpha)\delta(t)) = A\sin(\omega(x+(1+\alpha)\phi))$. For two time instances $t_1$ and $t_2$, the proposed method approximate $\delta(t_2)$ as $\omega(\phi_{t_2} - \phi_{t_1})$, and the magnified signal can be written as

$$f(x + (1 + \alpha)\delta(t_2)) \approx A\sin(\omega(x + (1 + \alpha)(\phi_{t_2} - \phi_{t_1}))) \tag{5.1}$$

Similarly, 1 D approximation of the method discussed in [29] can be written as

$$\begin{aligned} f(x + (1 + \alpha)\delta(t_2)) \approx A\sin(\omega(x + \phi_{t_1}))+ \\ (1 + \alpha)(A\sin(\omega(x + \phi_{t_2})) - A\sin(\omega(x + \phi_{t_1}))) \end{aligned} \tag{5.2}$$

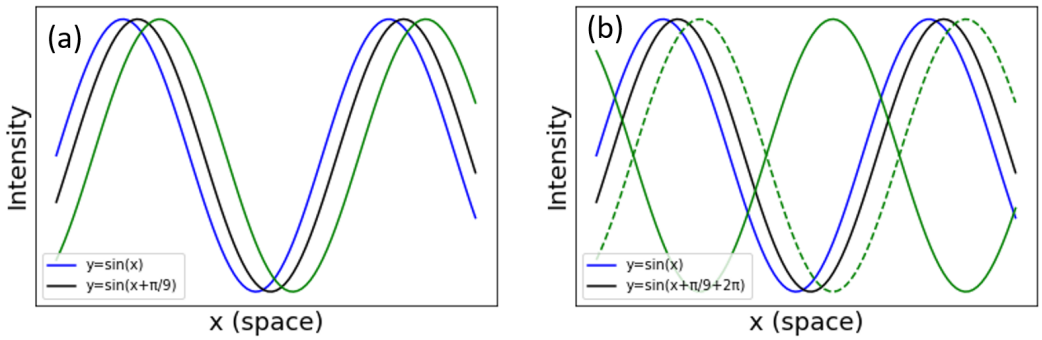


Figure 5.2: For the small phase shift between two sinusoidal waves output magnified signal is shown in green (on the left). When the shift becomes large (adding $2\pi$), similar-looking input produces a different magnified output in green (on the right). The phase shift based magnification method needs to determine the correct output (between the dotted green curve for the small phase shift and the green curve, the actual one).

Figure 5.1 illustrates the effects of change in magnification factor and error with respect to change in phase. Phase-variation based magnification has less error (from Figure 5.1). Also, in phase based magnification, the noise is translated instead of amplified, making it more robust to noise [26]. Let's extend it to a complicated function $S(x)$, by applying the Fourier series, a signal at time $t$ can be represented as:

$$S(x + \delta(t)) = \sum_{\omega=-\infty}^{\infty} A_\omega e^{j\omega(x+\phi_t)} \tag{5.3}$$

Then the proposed method magnified signal at time $t_2$ can be represented as :

$$S(x + (1+\alpha)\delta(t_2)) \approx \sum_{\omega=-\infty}^{\infty} A_\omega e^{j\omega(x+(1+\alpha)(\phi_{t_2}-\phi_{t_1}))} \tag{5.4}$$

Phase-variation based magnification has some difficulties. For instance, as the phase difference becomes large, there is phase ambiguity. As shown in Figure 5.2 for similar-looking sinusoidal waves, there are two different motion translations for the same magnification factor. This causes ringing artifacts and blurriness in the output [49]. Also, directly magnifying phase variation does not take occlusion into account. To resolve these issues, the output is first magnified in the frequency domain and then improved in the spatial domain.

### 5.2.2 Network Architecture

The proposed network takes only two frames at a time to produce a motion magnified frame according to the given magnification factor. It has two main blocks 1) Frequency domain-based motion magnification block (FDMM) and 2) Spatial domain-based

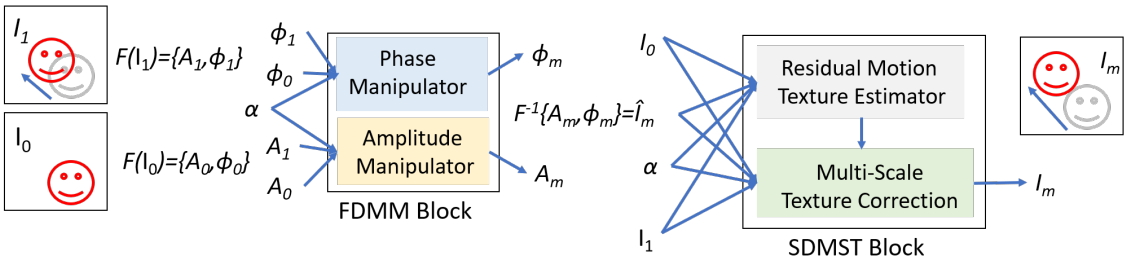


Figure 5.3: Proposed multi-domain based network for motion magnification. First, input frames $(I_1, I_0)$ Fourier transform $(\mathcal{F}(.))$ is taken and given to FDMM block. Then from the output phase, $\phi_m$ and amplitude $A_m$, intermediate magnified output $\hat{I}_m$ is generated by taking inverse Fourier transform. SDMST block process $\hat{I}_m$ in the spatial domain, to generate the final magnified output $(I_m)$.

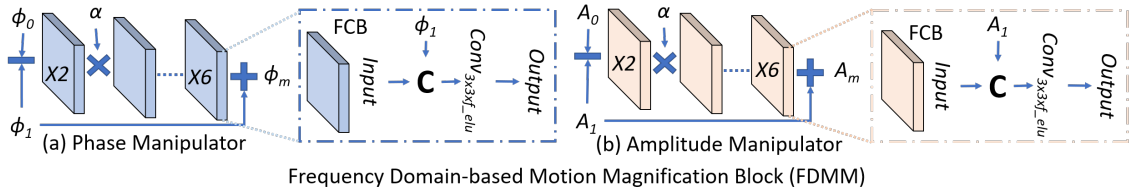


Figure 5.4: Structure of Frequency Domain-based Motion Magnification Block (FDMM). It consists of two parallel streams (a) Phase Manipulator and (b) Amplitude Manipulator. Phase Manipulator takes input frames phase $(\phi_1, \phi_0)$ and tries to estimate the magnified frame phase $(\phi_m)$. Similarly, the amplitude manipulator tries to predict output frame amplitudes $(A_m)$ from input frame amplitudes $(A_1, A_0)$.

multi-scale texture correction block (SDMST). The architecture of the proposed model is shown in Figure 5.3 and the details are discussed below

### Frequency Domain-based Motion Magnification Block (FDMM)

Let, the input frames in color space as $I_1$ and $I_2$. First, Fourier transform $(\mathcal{F})$ is applied on both frames to separate phase $(\phi)$ and amplitude $(A_t)$ as shown below

$$\mathcal{F}\{I_1, I_0\} = \{\{A_1, \phi_1\}, \{A_0, \phi_0\}\} \tag{5.5}$$

Then, the differences in phases and amplitude are processed in the FDMM block. To make the network lightweight, we did not apply convolution operations before taking the difference, as even without that the proposed network achieved good results. In dynamic scenarios, new information is getting into the image which results in a change of phase and amplitude. [26] depends on steerable pyramids to tackle this non-periodicity. But they produce distortions in dynamic scenarios. So, the FDMM block tries to estimate both, amplitude and phase changes in two parallel streams 1) Phase Manipulator and 2) Amplitude Manipulator as shown in Figure 5.3.

In the phase manipulator, first, it takes the difference of input frames phases $(\phi_1, \phi_0)$,

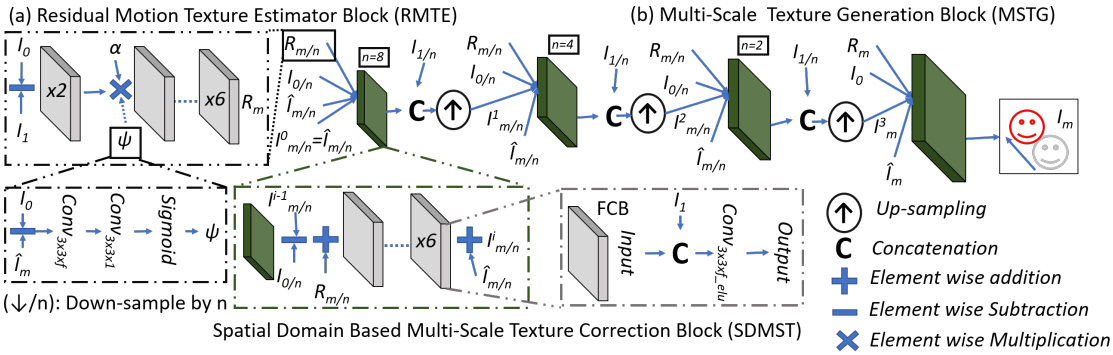


Figure 5.5: Depicts the Spatial Domain Based Multi-Scale Texture Correction Block (SDMST). It consists of two main parts (a) Residual Motion Texture Estimator Block (RMTE), and (b) Multi-Scale Texture Generation Block (MSTG). RMTE block estimated texture on magnified areas ($R_m$). While SDMST block is responsible for estimating the texture correction features, which are added to $\hat{I}_m$, to generate the magnified output ($I_m$).

and then they are passed through the fixed concatenate blocks (FCB). Similarly, this is done in an amplitude manipulator as shown in Figure 5.4. FCB takes two input features (previous output and one fixed input), and concatenates them both to give it to a $3 \times 3 \times f$ convolution layer with Elu activation, ($f$ represents the number of channels). FCB tries to predict the residual components that are added to $I_1$ features. So keeping $I_1$ features as fixed input to each layer helps in better estimation. The estimated output of the phase manipulator ($\phi_m$) and amplitude manipulator ($A_m$) are used to generate intermediate magnified output $\hat{I}_m$, where $\hat{I}_m = \mathcal{F}^{-1}\{A_m, \phi_m\}$ ( $\mathcal{F}^{-1}$ is inverse Fourier transform). $\hat{I}_m$, $\alpha$ and input frames $I_1$, $I_0$ are given as input to SDMST block to generate the final magnified input $I_m$ as shown in Figure 5.3.

### Spatial Domain Based Multi-Scale Texture Correction Block (SDMST)

SDMST block is used to improve the FDMM block output. Processing in the frequency domain leads to blur, inappropriate motion manipulation, and distortions in the output due to phase ambiguity and no-linear relation of phase changes between input and output. Spatial domain processing helps to remove them. It consists of two parts, 1) Residual Motion Texture Estimator Block (RMTE), and 2) Multi-Scale Texture Generation Block (MSTG).

**Residual Motion Texture Estimator Block (RMTE):** It assumes that a spatial manipulator can generate magnified motion features. The block structure is different from [29] manipulator. Instead of feature space difference (as in [29]), direct image space difference is taken. Also, to prevent distortions due to spatial magnification from adding, $\hat{I}_m$ based difference features are used as spatial attention features ($\hat{\Psi}$) (shown in Figure 5.5). This assumes that some distortions produced in both difference features are

orthogonal and will be canceled after multiplication. The final output of the RMTE block can be expressed as follows

$$R_m = FCB_{\times 6}\{(\alpha * \psi * (FCB_{\times 2}\{(I_1 - I_0), I_1\}), I_1\} \tag{5.6}$$

**Multi-Scale Texture Generation Block (MSTG):** Multi-Scale texture generation helps in improving the quality of the magnified frame. Features are processed from the lowest scale to the highest scale, like in U-net type architecture. However the encoder is replaced by simple average pooling to reduce the number of parameters. This structure differs from [29] as they use a simple decoder where residual blocks are stacked together to generate output. At each scale, a residual component is generated by difference between $I^{i-1}_{m/n}$ and $I_{0/n}$, such that they match the motion component with respect to $R_{m/n}$ (where $I^{i-1}_{m/n}$ is the previous scale $(i-1)_{th}$ magnified output, and $n$ subscript depicts the down-sampling rate). These features are processed with FCB and added to $\hat{I}_{m/n}$ features to create magnified features $I^{i}_{m/n}$. Then the magnified features are concatenated with $I_{1/n}$ and given to the conv-transpose layer for up-sampling. The exact process is repeated in the next scale, as shown in Figure 5.5. This assumes that the next scale blocks should work on residual input features created from previous scale-magnified features. These repeated estimations of texture components at each scale help in improving the prediction of the final texture feature map which is added to $\hat{I}_m$ (output of FDMM) for generating texture-corrected output. The magnified output at each scale can be defined as :

$$I^{i}_{m/n} = \hat{I}_{m/n} + FCB_{\times 6}\{(I^{i-1}_{m/n} - I_{0/n} + R_{m/n}), I_{1/n}\} \tag{5.7}$$

for $i = 0, I^{0}_{m/8} = \hat{I}_{m/8}$, where $i \in (0, 3)$. Texture in areas without motion is mostly similar in input and magnified frames. We assume giving $I_1$ information as fixed input in FCB helps in improving texture in areas where motion is not present.

### 5.2.3 Dataset, Loss Function, and Training

**Dataset:** For training, a synthetic dataset provided by Oh *et al.* [29] is utilized. It consists of 7,000 images of objects from the PASCAL VOC dataset [43] as foreground and 200,000 images of the MS COCO dataset [44] as background. Different foreground objects are combined with distinct backgrounds at various positions to yield random motion. It produces a total of 100,000 input pairs of 384×384 size.

**Loss Function and Training:** $L_1$ loss across the predicted and magnified is taken for training the network. To improve the edges' quality and reduce blur, edge loss $L_e$ [58] is used. These losses, penalize for each small deviation across the output, but some deviations are acceptable as long as they are not perceptible. So, a perceptual loss ($L_p$) is

Table 5.2: Parameters and GFOPs (measured at 720 X 720 resolution) of proposed lightweight networks $D_1$, $D_2$ and [29] method.

| **Methods** | Oh *et al.* [29] | $D_1$ | $D_2$ |
|---|---|---|---|
| **Parameters** | 0.98 M | 0.117 M | 0.053 M |
| **GFLOPs** | 268.6 | 65.4 | 30.4 |

also applied. Additionally, $L_1$ loss across the phase and amplitude of the predicted frames is used to train the FDMM block efficiently. The final loss function is illustrated as

$$\begin{aligned} L_f = \lambda_1 L_1(I_m, I_{gt}) + \lambda_2 L_p(I_m, I_{gt}) + \\ L_e(I_m, I_{gt}) + L_1(\phi_m, \phi_{gt}) + L_1(A_m, A_{gt}) \end{aligned} \tag{5.8}$$

where $gt$ subscript indicates the ground truth. $\lambda_1 = 10$, $\lambda_2 = 0.1$, and an ADAM optimizer with a learning rate set to 0.0001 is used for training. Gaussian noise is added to the input to mimic noise. All the models are trained on NVIDIA 2080 RTX with 8GB GPU. Different lightweight networks ($D_1$, $D_2$) are generated in the proposed pipeline by changing the number of channels $f$, as shown in Table 6.2.

## 5.3 Experimental Results

The proposed method is compared on real-world and synthetic videos with state-of-the-art methods Jerk-Aware [31], Anisotropic [32], Acceleration [30] and Oh *et al.* [29]. The details of parameters used for result generation are shown in Table 6.3. Linear filter based methods are not considered for comparisons as they produce distortions in dynamic scenarios. For comparison, results of SOTA methods are generated for different videos from their official implementation receptively (for more details please see the supplementary material). The following sub-sections include a detailed discussion of the qualitative and quantitative comparison. Also, an additional experiment on physical accuracy is provided. Further, an ablation study is performed to illustrate the significance of various parts of the network. All the results of the proposed method are generated using consecutive frames (dynamic mode in [29]) unless otherwise specified.

### 5.3.1 Qualitative Analysis on Real World Videos

We evaluate the proposed methods in a challenging set of scenarios, including rotating objects (Hand Drill in Figure 5.7), in the presence of large motion (Balloon Burst in Figure 5.8 ), and in dynamic motion (Gun Recoil in Figure 5.6). SOTA hand-crafted methods produce small magnification in challenging scenarios, as a further increase in magnification factor only leads to a rise in distortions like ringing artifacts, blurriness, *etc.* (see supplementary material for details). [29] produces high magnification with flickering

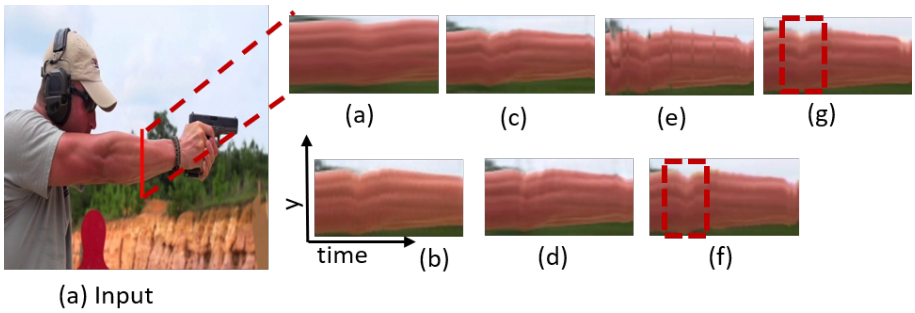


Figure 5.6: Gun Recoil: Video contains a large translation motion due to movement of the camera from left to right and the subtle motion generated in the forearm due to gun recoil. The target is to magnify the forearm motion in the dynamic scenario. Spatial-temporal slice is taken from the red strip and illustrates how SOTA methods (b) Acceleration method [30], (c) Jerk-aware [31], (d) Anisotropy [32], (e) Oh *et al.* [29], and the proposed method (f) $D_1$, (g) $D_2$ magnify subtle motion. Hand-crafted methods [32], [31], [30] have small magnification. But [29] produces more magnification but induces flickering motion (visible as spikes in temporal slice (e)). The proposed network has the highest amount of motion (highlighted in the red bounding box) with fewer distortions.

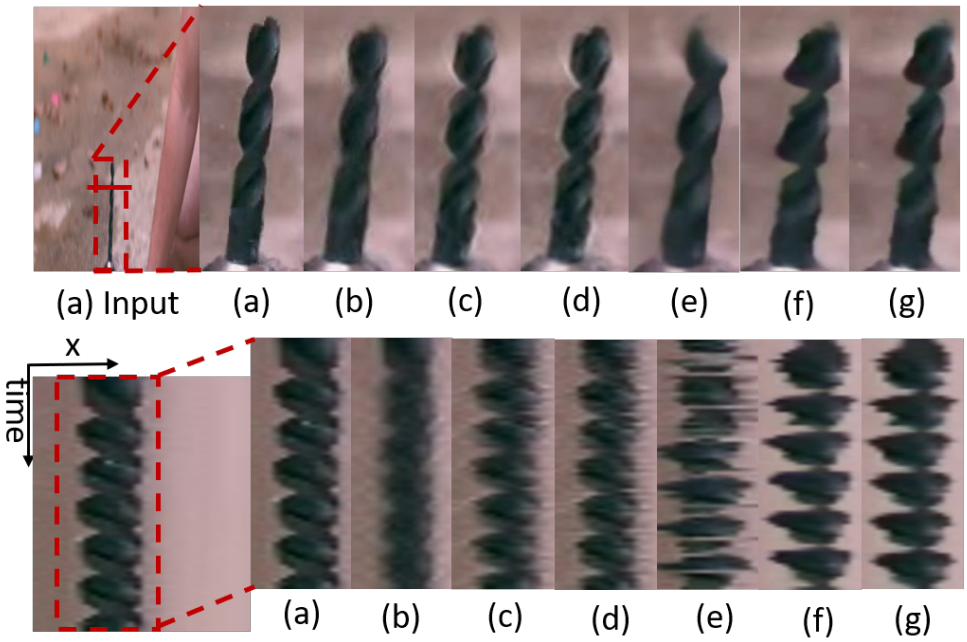


Figure 5.7: Hand Drill: Magnifying rotational motion is a difficult task. So to evaluate SOTA methods (b) Acceleration method [30], (c) Jerk-aware [31], (d) Anisotropy [32], (e) Oh *et al.* [29], and the proposed method (f) $D_1$, (g) $D_2$, a video containing a hand drill with rotational motion along its axis is used. In 2D, this motion is visible as a spiral motion. So, magnification can be perceived as an increase in spiral motion (shown in spatial-temporal slices taken from the red strip at the right part of the figure). Hand-crafted methods [32], [31], [30] have small magnification (less outward radius in temporal slices) and produce ringing artifacts (visible as white edges around the drill) and blurry spikes in the temporal slices (b), (c), (d)). Oh *et. al* [29] induce flickering motion (seen as spikes in the temporal slice (e)) and blurry distortions in some frames (visible in the frame (e)). The proposed networks ( (f) $D_1$ and (g) $D_2$) produce better magnification with fewer distortions.

Table 5.3: Parameters used for result generation. All the results are generated with variables and steps given by the respective authors.

| Methods | **Video** | $M_f$ | Frequency |
|---|---|---|---|
| Ours $(D_1, D_2)$ | Gun | 5 | N/A |
| Ours $(D_1, D_2)$ | Drill | 5 | N/A |
| Ours $(D_1, D_2)$ | Balloon | 5 | N/A |
| Ours $(D_1, D_2)$ | Physical Accuracy | 10 | N/A |
| Ours$(D_1, D_2)$ | Circle videos with different backgrounds | 60 | N/A |
| Oh*et al.* [29] | Gun | 4 | N/A |
| Oh*et al.* [29] | Drill | 10 | N/A |
| Oh*et al.* [29] | Balloon | 10 | N/A |
| Oh*et al.* [29] | Physical Accuracy | 5 | N/A |
| Oh*et al.* [29] | Circle videos with different backgrounds | 60 | N/A |
| Jerk-Aware [31] | Gun | 10 | 20 |
| Jerk-Aware [31] | Drill | 25 | 3 |
| Jerk-Aware [31] | Balloon | 25 | 3 |
| Jerk-Aware [31] | Physical Accuracy | 20 | 15 |
| Jerk-Aware [31] | Circle videos with different backgrounds | 200 | 15 |
| Anisotropy [32] | Gun | 100 | 20 |
| Anisotropy [32] | Drill | 100 | 3 |
| Anisotropy [32] | Balloon | 100 | 3 |
| Anisotropy [32] | Physical Accuracy | 200 | 3 |
| Anisotropy [32] | Circle videos with different backgrounds | 400 | 15 |
| Acceleration [30] | Gun | 10 | 20 |
| Acceleration [30] | Drill | 4 | 3 |
| Acceleration [30] | Balloon | 4 | 3 |
| Acceleration [30] | Physical Accuracy | 20 | 15 |
| Acceleration [30] | Circle videos with different backgrounds | 200 | 15 |

and superious motion in challenging scenarios. The proposed lightweight networks $D_1$ and $D_2$ generate good results. The $D_2$ model gives good results in static scenarios but its texture quality decreases compared to $D_1$ in dynamic scenarios. In the static scenario, most of the scenes of input images are same as in the output. But this changes in dynamic scenarios, where occlusion plays a much more significant role and requires good texture generation. We assume the network learns a better form of frame blending to generate motion magnified frames. However the texture generation ability decreases after a reduction in several parameters. Improving texture quality in dynamic scenarios

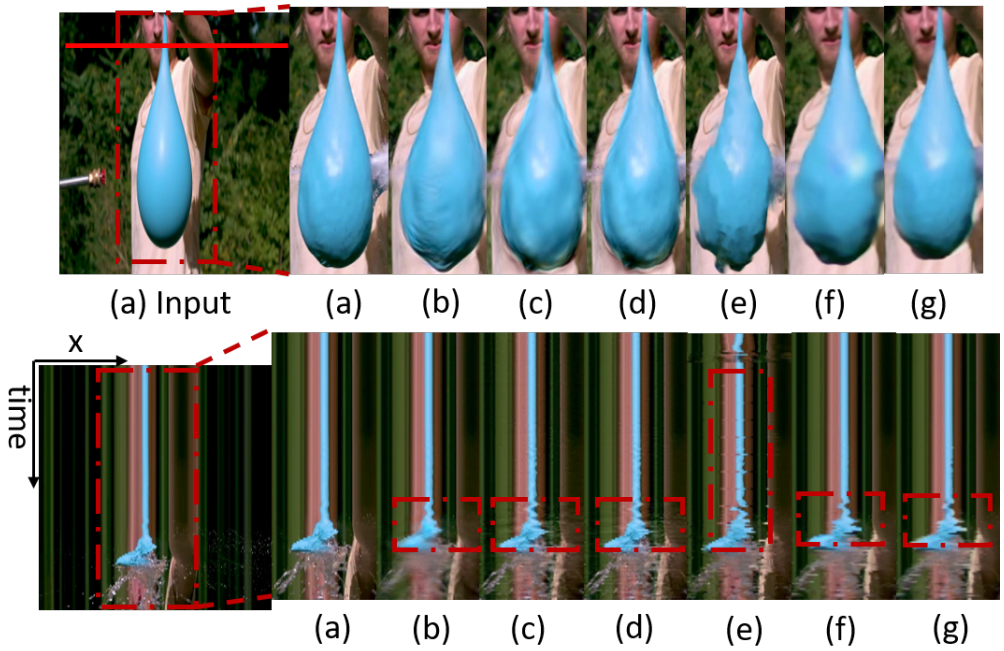


Figure 5.8: Balloon Burst: In the video, a water canon raptures the balloon. Balloon develops small and large motions as it bursts. The aim is to magnify subtle changes in the balloon while in the presence of large motion. To illustrate this intermediate frames (in the left part of the Figure) and spatial-temporal slices taken from the red strip (shown in the right part of the Figure) are shown for SOTA methods (a) Anisotropy [32], (b) Jerk-aware [31], (c) Acceleration method [30], (d) Oh *et al.* [29], and the proposed method $D_1$, $D_2$. Hand-crafted techniques [32], [31], [30] have small magnification and generate ringing artifacts around the balloon, visible as white edges around the balloon in the intermediate frames and white spikes in the temporal slice (highlighted in the bounding boxed)). They also have less magnification than the proposed method (see the bounding box). Whereas [29] produces flickering motion (seen as white spikes across the whole temporal slice (e)) and blurry distortions in some frames (see (e) frame). The proposed networks have more magnification with lesser distortions.

with less than $0.1M$ parameters is a challenging task. So, depending on the application, they give users a good trade-off between quality and the number of parameters. Despite that, the proposed networks give reasonably good magnification with fewer distortions than most SOTA methods, as shown in Figures 5.7, 5.8, 5.6.

### 5.3.2 Quantitative Analysis

Obtaining the actual ground truth of motion magnified videos in real-life scenarios are challenging. Without the ground truth, quality estimation of the magnified frame is difficult. As the amount of magnification decreases, distortions become less and the output becomes perceptible. But that will defeat the purpose of magnification. So, the analysis requires accounting for both magnification and output quality. Considering these factors synthetic videos with various backgrounds are generated. Different background videos will help to test the adaptability of the proposed method in different scenarios. Circles with

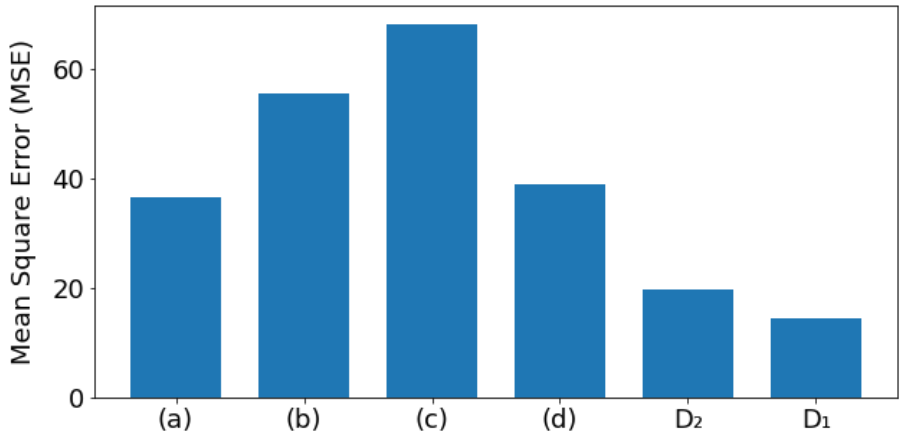


Figure 5.9: Average Mean Square Error (MSE) of 25 synthetically generated videos with different backgrounds, containing subtle motion of circles on (a) Anisotropy [32], (b) Jerk-aware [31] , (c) Acceleration method [30] , (d) Oh *et al.* [29], and the proposed method $D_1$, $D_2$. The proposed networks $(D_1, D_2)$ have the first and second best results respectively.

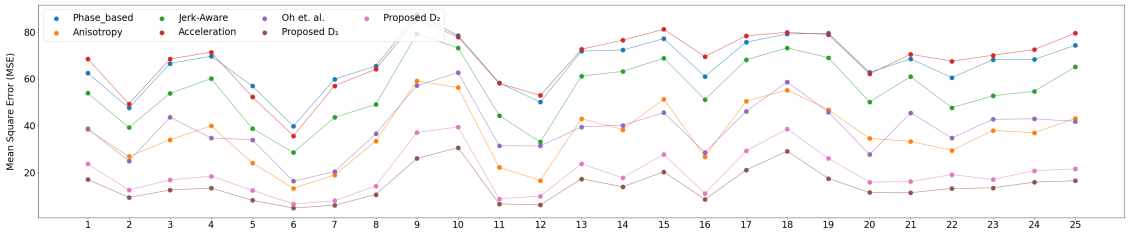


Figure 5.10: Mean Square Error (MSE) of Anisotropy method [32], Jerk-aware method [31], Acceleration method [30], Oh *et al.* method [29], Phase based method [26] and the proposed methods on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

horizontal, vertical, and diagonal directions motion are used to mimic the subtle motion. Input subtle motion is 0.1 pixels and the ground truth has 10-pixel motion (100 x more than the input). Gaussian noise is added to mimic photographic noise in the videos (Please see section 3.2.3 for more details). Different magnification factors are used to deliver the same motion as ground truth (more details are given in supplementary material). Output mean square error concerning ground truth for various SOTA methods [31], [32], [30], [29], and the proposed method are shown in Figure 5.9. First, MSE values across all the frames in a video are averaged, and then the average across 25 videos is calculated. Further, figure 5.11 illustrates the impact of increasing noise levels (sigma) on the average mean square error (MSE). From Figure 5.9 and 5.11, the proposed method has the minimum error, as it produces better magnification with lesser distortions.

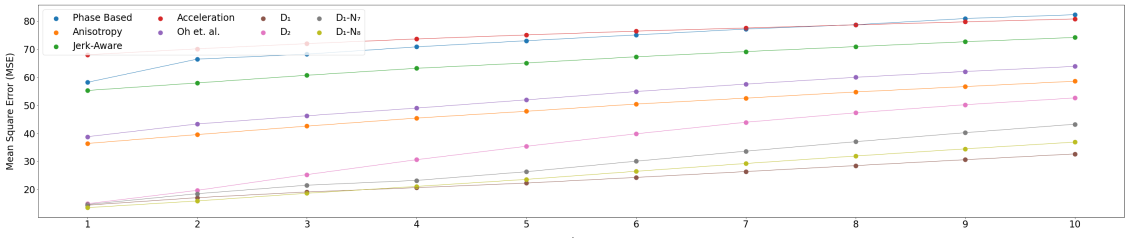


Figure 5.11: Effects of increase in noise value (sigma) in input. The average mean square error (MSE) is computed across the predicted output and ground truth, over 25 different videos. Comparison is done with the Anisotropy method [32], Jerk-aware method [31], Acceleration method [30], Oh *et al.* method [29], Phase based method [26], the proposed model $D_1$, $D_1$-$N_7$ , $D_1$-$N_8$ and $D_2$. $D_1$-$N_7$ and $D_1$-$N_8$ are the $D_1$ models trained without amplitude and phase manipulator respectively.

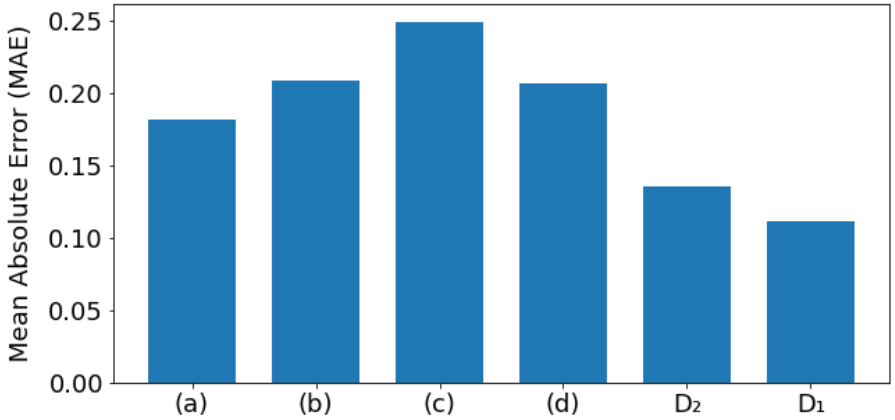


Figure 5.12: Mean Absolute Error (MAE) is computed between the extracted signal from magnified video and sensor measured signal. The error values of SOTA methods (a) Anisotropy [32], (b) Jerk-aware [31] , (c) Acceleration method [30] , (d) Oh *et al.* [29], and the proposed method $D_1$, $D_2$ are shown. The proposed networks ($D_1, D_2$) have the first and second best results respectively.

### 5.3.3 Physical Accuracy

To check the physical accuracy of the magnified output an experiment with set-up, as shown in Figure 5.13 (a), is conducted. Subtle motions (up and down) are generated in the mechanical rod of the universal vibration apparatus. These motion signals are recorded using an ultrasonic sensor and a video camera (Please see section 3.2.2 for more details). The motion signal is extracted from the magnified videos and compared with the ultrasonic sensor signal as shown in Figure 5.13. Both the sensor measured and the computed magnified signal are rescaled to 0 to 1 and mean absolute error (MAE) values are calculated across SOTA methods. As illustrated in Figure 5.12 the proposed method has the minimum MAE values.

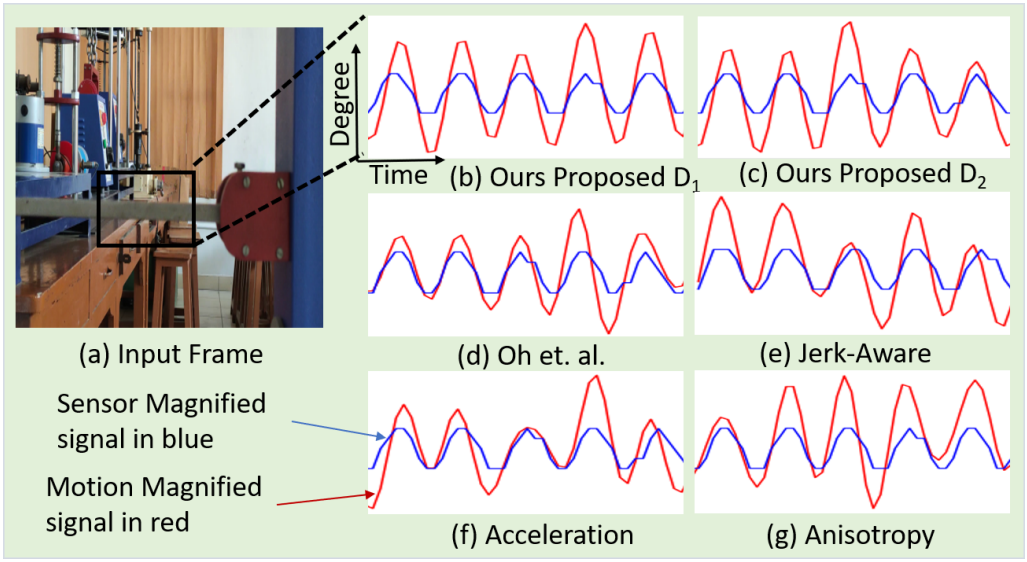


Figure 5.13: Physical Accuracy: Comparison between our method and other SOTA methods output (in red) with the sensor signal (in blue) respectively. The optical flow across the input frame and the magnified frame (of respective methods) is computed to extract the motion signal. Then the average direction along the image patch (marked in the bounding box in (a)) is calculated and shown above.

### 5.3.4 Additional Experiments

**Effects of increase in Magnification Factor:** The figures 5.14, 5.15 depict the effects of varying magnification factors on the output of two proposed models, $D_1$ and $D_2$. In Figure 5.14, $D_1$ shows less distortion as magnification increases compared to other state-of-the-art methods, both in static and dynamic scenarios. Figure 5.15 illustrates $D_2$ model output, which also exhibits good magnification, but its performance degrades with higher magnification factors compared to $D_1$, attributed to $D_2$ having fewer parameters. These visualizations offer insights into model performance under varied magnification factors, highlighting their robustness and limitations.

### 5.3.5 Ablation Study

An ablation study is conducted to verify the different aspects of the model, in contribution to motion magnification. For this different ablation models are generated with assuming $D_1$ as the base model. First, (a) $D_1$-$N_1$ model is trained without FDMM block to analyze the effects of frequency domain operation in motion magnification. Further, $D_1$-$N_2$ is trained without phase and amplitude loss to examine the influence of frequency domain regularization terms. In both cases, there is a decrease in output quality (demonstrated in Figure 5.16 in terms of an increase in error with respect to $D_1$).

To analyze the different aspects of SDMST block (a) $D_1$-$N_3$ is trained without RMTE block (b) $D_1$-$N_4$ without spatial attention $\psi$ in RMTE block. To highlight the efficiency of MSTG block in texture synthesis (c) $D_1$-$N_5$ model with a simple U-net like decoder

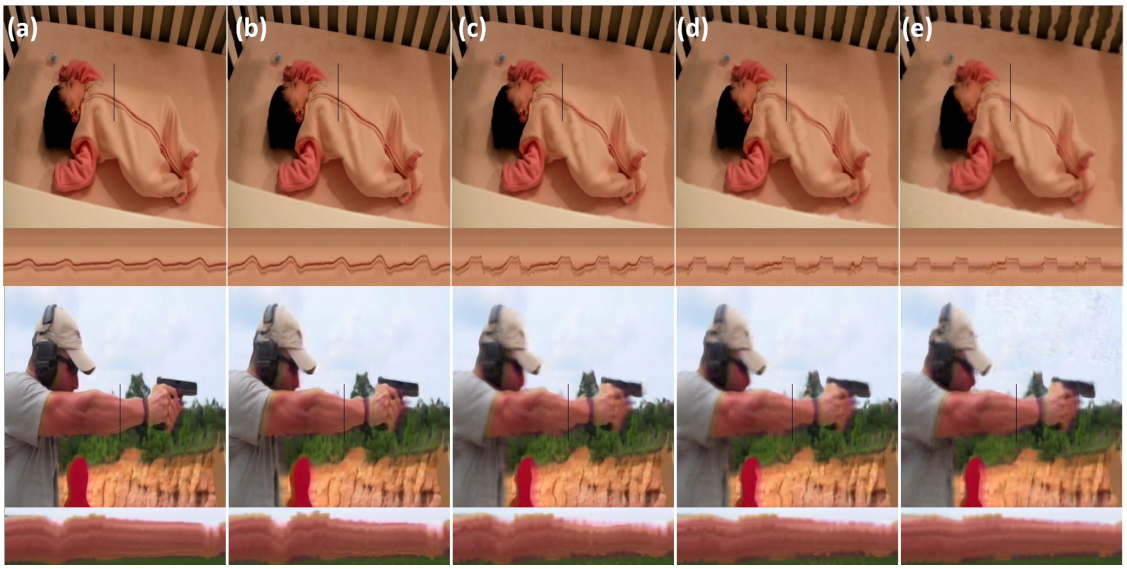


Figure 5.14: Effects of change in Magnification Factor: Figure illustrates proposed $D_1$ model output. Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column. $D_1$ shows fewer distortions while increasing the amount of magnification as compared to other SOTA methods, both in static and dynamic scenarios.

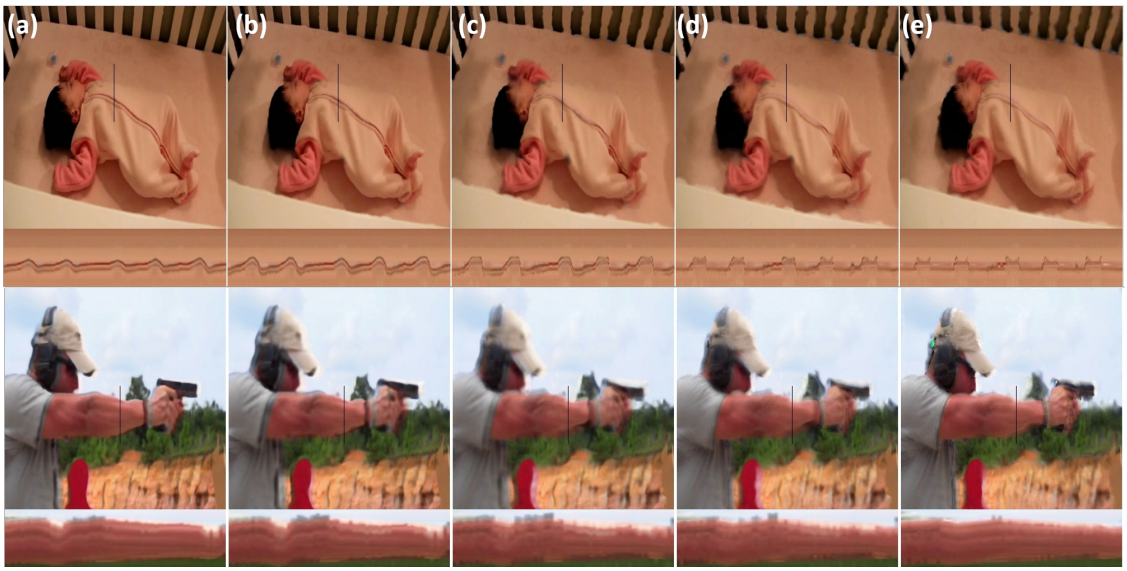


Figure 5.15: Effects of change in Magnification Factor: Figure illustrates our $D_2$ model output. Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column. $D_2$ also shows a good amount of magnification, but with an increase in magnification factor, its performance degrades as compared to $D_1$. This is expected as $D_2$ has much fewer parameters than $D_1$, so their performance gap becomes observable in extreme scenarios.

with residual blocks instead of $FCB$, is used. Further, (d) $D_1$-$N_6$ is trained at single scale texture generation block, instead of multi-scale texture generation block ($MSTG$) to emphasize the consequences of multi-scale in the SDMST block. An increase in error has been observed in the ablation models as compared to the proposed model $D_1$ (refer

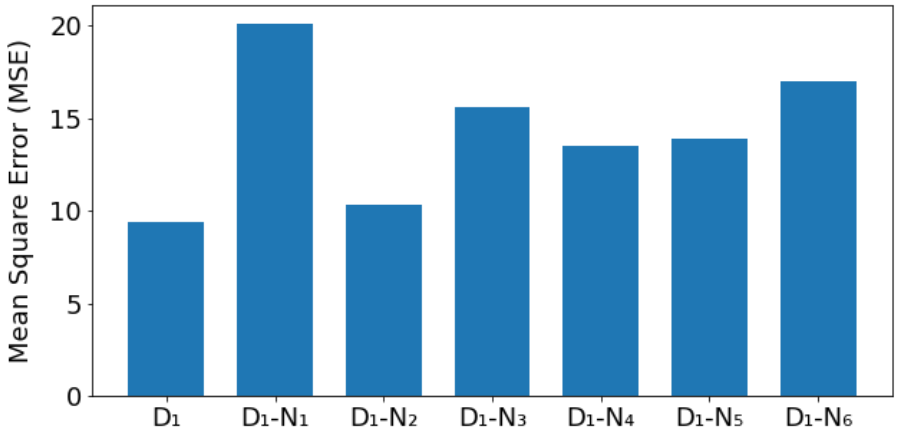


Figure 5.16: Aggregate MSE values are computed across the same synthetic videos as in 5.3.2 for $D_1$-$N_1$ to $D_1$-$N_6$ models as defined in section 6.3.4. The proposed network ($D_1$ has the minimum MSE values when compared with different ablation networks, which indicates the importance of the proposed modules.

Figure 5.16). Furthermore, to analyze the effects of different manipulators, Figure 5.11 shows the MSE of models trained without amplitude ($D_1$-$N_7$) and phase manipulators ($D_1$-$N_8$) compared to the base model $D_1$.

### 5.3.6 Summary

In this chapter, for video motion magnification, we suggest a multi-domain network that combines frequency domain and spatial domain-based operation. The proposed network first works in the Fourier domain and tries to predict phase and amplitude changes of the magnified frame, according to the magnification factor. Then, the spatial domain uses frequency domain output to generate appropriate motion magnified output. Further, lightweight models are proposed, giving comparable results with SOTA methods. Results are analyzed qualitatively and quantitatively on real-world and synthetic videos. Also, an experiment is done to highlight the physical accuracy of the proposed networks. An ablation study is conducted to show the effects of different modules in the proposed network pipeline. Results show that the proposed models ($D_1$ and $D_2$) perform better than SOTA methods in terms of more magnification with less distortions. However, the proposed work doesn't fully utilize the steerable pyramid architecture of hand-crafted-based methods, as it manipulates motion features at a single scale.

# Chapter 6

# Learnable Directional Scale Space Filters for Video Motion Magnification

Video motion magnification based techniques make subtle, minute, imperceptible changes visible to the naked eye. But this is a difficult task, as minute variations are generally affected by large motion, occlusion, illumination changes, noise, *etc.* Traditionally, this problem is solved using scale and direction space by employing complex steerable pyramids for motion representation and temporal filters for motion manipulation across different scales and orientation features. By increasing the number of filters in scale and orientation space their performance can be further improved at the expense of computation cost [26]. However, these methods [31], [32], [42], [30] do not effectively estimate the motion variations due to steerable pyramid limitations and cause ringing artifacts. Further, the presence of occlusion, dynamic scenarios, *etc.*, complicates the input scenario and results in small magnification.

Recently, deep learning based approaches have emerged to tackle these issues [20], [48], [29], [95], [96]. Some approaches are scenario specific (require separate training for each new scenario) [20], [48], while others are generalized (same weights can work for different videos) [29], [95], [96]. First, [29] (LB-MM) proposed a supervised learning based training to directly translate the pixel values. This results in higher magnification compared to hand-crafted methods but is susceptible to distortions due to noise. The task of ignoring noise while producing high magnification is quite challenging. To tackle this issue, different approaches are proposed such as knowledge distillation [95] (LW-MM), or processing in frequency and spatial domain [96] (MD-MM) for better discriminative features (which makes subtle motion standout while reducing the effect of noise). However, these solutions sometimes produce artifacts and are not efficient in ignoring noise for subtle motion magnification.

To address these challenges, we propose a framework that combines insights from both traditional and deep learning approaches. Leveraging scale and orientation space characteristics, helps hand-crafted methods to generate improved motion features [26]. Whereas, previous deep learning based methods only focused on single scale based changes

Table 6.1: Comparison between proposed and existing approaches.

| **Methods** | **Hand-crafted Methods** | **Deep-learning Methods** | **Proposed Method** |
|---|---|---|---|
| **Feature Representation for Motion Manipulation** | Multi-Scale and Directional Phase (Complex Steerable Pyramid) | Single Scale Latent Feature Representation | Multi-Scale Directional Gaussian Gradient Phase and Magnitude |
| **Motion Manipulation** | Linear/Non-Linear Filters | Consecutive Frame Feature Variation Based Learnable filters | Consecutive Frame Feature Variation Based Learnable filters |

without orientation consideration. Inspired by hand-crafted based method [26], we propose a learnable scale and direction space based motion manipulator. It computes different scale features of input frames using Gaussian functions with varying receptive fields and standard deviations. Then, these features are combined to generate an efficient scale-space ($\sigma$) feature. Further, it computes gradients of intermediate frames along different directions ($\theta$) and magnifies their magnitude and phase variations along each direction. These angular features are weighted averaged to generate an optimum direction ($\theta$) magnified gradient feature. Also, these magnified gradient features are generated at different resolutions and merged to produce the manipulator output features.

For texture synthesis, deep learning-based methods outperform traditional methods with higher magnification through the utilization of learnable convolution filters. However, deep learning-based methods tend to be computationally complex and result in texture distortion. Designing a lightweight texture generation decoder is a difficult task. To solve this, we propose a multi-scale channel compression block. It uses information from multiple scales using dilation convolution. Additionally, a channel compression block is used to reduce the number of channels by varying ratios. We assume that by decreasing the number of channels, we can minimize redundant information and achieve a denser representation. Also, this helps to reduce the number of parameters. The final output is synthesized by merging the magnified outputs at different resolutions. The main contributions of this work are summarized as follows:

- We propose a novel deep learning based lightweight ($\sim 0.084M$) architecture for video motion magnification.
- We propose a motion manipulator block with learnable scale ($\sigma$) and directional ($\theta$) spaces, to extract optimal motion features for video motion magnification.
- We also propose a multi-scale channel compression decoder block with varying compression ratios and receptive fields for generating motion magnified frames.

Our results analysis demonstrates that the proposed method outperforms the state-of-the-art (SOTA) methods both qualitatively and quantitatively, with less number of parameters. Also, we conducted additional experiments to show frequency selectivity. Additionally, an ablation study is performed to examine the effects of different components of the proposed method.

## 6.1 Related Works

Video motion magnification based algorithms have been classified into two approaches: 1) the Lagrangian approach and 2) the Eulerian approach. Lagrangian approach based techniques depend on flow vectors and compute pixel displacements directly through optical flow [33], [36], whereas Eulerian approaches estimate the intensity of the displaced pixels, independent of flow vectors [27], [26], [48], [29], [95], [96]. Eulerian-based approaches have gained popularity in the field of motion magnification. These techniques operate in three steps: firstly, they decompose input images in scale and direction space using steerable pyramids, then use temporal filters to manipulate motion, and finally reconstruct the magnified frame. Earlier work focused on feature representation space for motion manipulation. Wu *et al.* [27] employed Gaussian pyramids for image decomposition, albeit producing noisy output. To address this, Wadhwa *et al.* [26] proposed the use of complex steerable pyramids to extract local phase information for motion magnification, and showed improved performance by increasing the number of filters in scale and orientation space. Subsequent hand-designed methods [30], [31], [32], [42] are developed to handle large motions. [30] presented a filter to magnify only non-linear changes. Further, [31] proposed a jerk-aware filter to prevent the fast non-linear changes from getting magnified. Also, to reduce the effects of noise, [32] suggested the use of an anisotropic filter. They assume meaningful motion is anisotropic in nature. Moreover, [42] proposed a bilateral video magnification filter (BVMF) that offers robust and more accurate temporal filtering. However, these methods resulted in limited magnification, ringing artifacts, and failed to account for occlusions *etc.*

Recent advancements in deep learning techniques have addressed the limitations of handcrafted methods in video motion magnification. Oh *et al.* [29] (LB-MM) formulated the motion magnification problem in a supervised manner and proposed a synthetic dataset for it. In [29], encoder decoder architecture at a single resolution is used for motion magnification. However, their method sometimes distorts the output motion. Later, Singh *et al.* [95] (LW-MM) suggested the use of knowledge distillation to enhance the model's robustness but occasionally it distorts the shape and their high-performance model is computationally complex. Furthermore, [96] (MD-MM), a robust and lightweight model is introduced, leveraging frequency and spatial domain features. Multi-resolution features

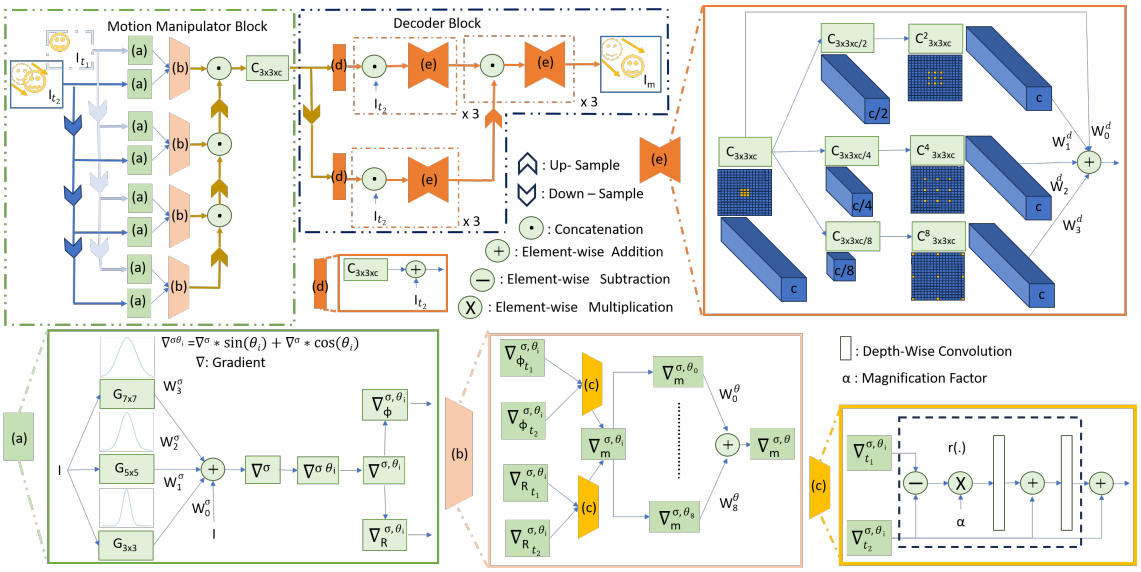


Figure 6.1: $\sigma$-$\theta$**Net** operates on two consecutive frames $(I_{t_1}, I_{t_2})$ as input and generates a magnified output $I_m$. The motion manipulator block consists of learnable direction scale space Gradients (a), (b), and (c), and the decoder block contains a Multi-Scale Channel Compression Block ($MSCC$) (e). Firstly, (a) generates different scale space features by employing a learnable Gaussian layer with varying kernel sizes. The gradients of weighted average scale-space features are computed, and the phase ($\phi$) and magnitude ($R$) feature representation across various orientations ($\theta_i$) are calculated. Next, (b) magnifies the phase and magnitude information based on the magnification factor ($\alpha$) using a function $r(\cdot)$. $r(\cdot)$ is implemented through depth-wise separable convolution blocks, (more details are visible in (c) block). Afterward, the magnified $\phi$ and $R$ are utilized to reconstruct the magnified gradient ($\nabla_m^{\sigma,\theta_i}$) along different directions. Further, the resultant features across various directions are unified, to produce feature $\nabla_m^{\sigma,\theta}$. At the decoding end, (d) Adds $I_{t_2}$ within the manipulator motion features. Moreover, in $MSCC$ (e), $I_{t_2}$ is concatenated with input features, followed by channel-wise compression using different dilation rate convolutions to generate magnified features. Furthermore, $MSCC$ block (e), is used for the feature fusion of different resolution magnified features, to construct the final magnified output.

are employed with up-samplers at the decoder, while single resolution features are used in the manipulator block (RMTE). However, [96] feature fusion method is not efficient, as it incorporates global variations and distorts fast moving small objects. To address these challenges, we integrate insights from traditional methods with a deep learning approach, as depicted in Table 6.1. Our method incorporates multi-Gaussian scale spaces in the manipulator and varied dilation rates and resolutions in decoder blocks, enhancing noise-robustness.

## 6.2 Proposed Method

Inspired by the hand-crafted mechanism of steerable pyramids [26], in this work, we propose the learnable scale ($\sigma$) and directional ($\theta$) spaces for the manipulation of motion features. Initially, we extract learnable directional Gaussian gradient phase and magnitude

of intermediate frames, then magnify changes across them. This helps to get better discriminative features to ignore noise while magnifying the subtle motion. Further, texture synthesis is done using channel compression with multi-scale features and fusing them across different resolutions. This facilitates achieving higher magnification with improved texture quality while simultaneously reducing the number of parameters. In the following section, we provided a detailed discussion on motion manipulator and decoder blocks.

### 6.2.1 Motion Manipulator

**Motivation:** Consider a video containing $N$ frames, where the frame intensity is denoted as $I(x,t)|t \in [0,N]$, with $x$ representing the spatial coordinate. Suppose, at any given time $t$, the frame intensity can be expressed as $I(x,t) = I(x+\delta(t),0)$, where, $\delta(t)$ represents the displacement function. Then the motion magnified signal is defined as [27] $I(x,t) = I(x+(1+\alpha)\delta(t),0)$, in which $\alpha$ decides the amount of magnification. Conventional methods [30], [31], [32], [42] use complex steerable pyramids ($\Psi^{\sigma,\theta}$) for local phase image decomposition across different scales. These methods magnify phase changes through a bandpass filter ($\alpha * B^{\sigma}(\phi^{\sigma,\theta}(x,t))$) and reconstruct the magnified output based on the phase changes ($\hat{\phi}^{\sigma,\theta}(x,t)$). Their relation can be defined as:

$$(I \otimes \Psi^{\sigma,\theta})(x,t) = A^{\sigma,\theta}(x,t)e^{i\phi^{\sigma,\theta}(x,t)} \tag{6.1}$$

$$\hat{\phi}^{\sigma,\theta}(x,t) = \phi^{\sigma,\theta}(x,t) + \alpha B^{\sigma}(\phi^{\sigma,\theta}(x,t)) \tag{6.2}$$

These methods perform well in static environments, but their performance degrades in dynamic scenarios. As in dynamic scenarios, $\delta(t)$ can become highly non-linear and complex as $t$ increases. In deep learning-based methods, it is assumed that as $t \to 0$, $I(x,t) = I(x+\delta(t),0)$ assumption holds in dynamic scenarios, such that $\delta(t)$ can be approximated from the change in consecutive frames [29]. So, by only considering two consecutive frames $(t_1, t_2)$ at a time, they are able to provide better results with higher magnification in dynamic videos. Initially, [29] proposed an encoder ($E$) for feature translation from the input image to the latent space, followed by the use of a manipulator ($M$) to extract the motion features (using $g(.)$, and $h(.)$ convolution layers based non-linear functions; *see [29] for more details*). A decoder is then employed to reconstruct the magnified output based on the manipulator features. Hence, in deep learning based architectures, the role of manipulator output features ($M_{t_2}$), closely resembles the $\delta(t_2)$ and can be represented as :

$$M_{t_2} = E_{t_2} + h(\alpha * g(E_{t_2} - E_{t_1})) \tag{6.3}$$

Here, we conclude two basic intuitions: 1) Improving the representation of input

features; and 2) Enhancing the manipulator's capacity to leverage input representation for extracting motion information, can help to generate robust motion features. Drawing inspiration from hand-crafted based methods, we propose a learnable directional scale space based phase and magnitude manipulator for motion manipulation (see Table 6.1). Additionally, a multi-scale channel compression decoder block is used to generate magnified output based on manipulator features.

**Motion Manipulator Block with Learnable Directional Scale Spaces :** First, let's consider a 2-D Gaussian function $G(x, y)$, such that the $n_{th}$ order Gaussian derivative can be represented as $G(x, y)^{\theta}_{n}$, where $\theta$ represents the rotation. The first order derivative of the Gaussian function at any arbitrary direction $G^{\theta}_{1}$, can be synthesized as a linear combination of $G^{0}_{1}$ and $G^{90}_{1}$ [97] as :

$$G^{\theta}_{1} = \cos(\theta)G^{0}_{1} + \sin(\theta)G^{90}_{1} \tag{6.4}$$

Here $G^{0}_{1}$, $G^{90}_{1}$ are the basis filters, which span the set of $G^{\theta}_{1}$ filters, and $\sin(\theta)$, $\cos(\theta)$ are the interpolation functions for the basis function. As the convolution function is a linear operation, we can filter any image $(I)$ to generate a response $\nabla^{\theta}$ (please note that here we represent the gradient of an image $I$ as $\nabla$)

$$\nabla^{\theta} = \nabla^{0}\cos(\theta) + \nabla^{90}\sin(\theta) \tag{6.5}$$

Different orientations can provide distinct image information. Also, given the random nature of noise, it may vary with rotations. To find an optimal set of rotations that mitigate noise while magnifying the subtle motions, we make the direction parameter $(\theta)$ learnable. Without the learnable directional gradient, noise based distortions increase as analysed in Figure 6.2.

Similarly, in the case of different scale-space, Gaussian functions with different receptive fields and learnable standard deviations are utilized to generate efficient scale features. Afterward, these features are fused together to find the optimal feature. Let a Gaussian kernel of size $(m_i \times m_i)$ be convolved with the input image $I$, where $m_i = (2 * i) + 1$. The resultant features are then multiplied with the softmax attention weight $(W^{\sigma}_{i})$, such that $i \in \{0, 1, 2, 3\}$, yielding the final scale features $(I^{\sigma})$ as:

$$I^{\sigma} = W^{\sigma}_{0} * I + \sum_{i=1}^{3} W^{\sigma}_{i} * (I \otimes G_{m_i \times m_i}) \tag{6.6}$$

Combining Eq. 6.5 and Eq. 6.6 we obtain $\nabla^{\sigma,\theta}$, which represents gradient features at scale $\sigma$ and direction $\theta$, such that $\theta \in (0, 2\pi)$. Likewise, these features can be generated for any two consecutive input frames $I_{t_1}$, $I_{t_2}$ as $\nabla^{\sigma,\theta}_{t_1}$ and $\nabla^{\sigma,\theta}_{t_2}$. The Sobel operator is utilized for

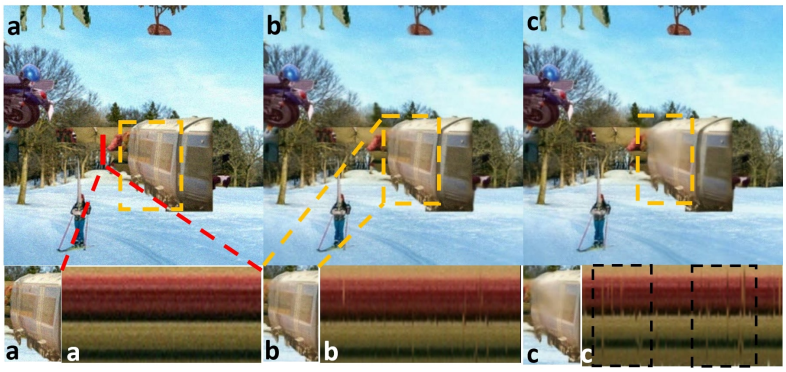


Figure 6.2: In (a), a synthetic input with noise-based variations and no input motion, is shown. The results of (b) the proposed model ($\sigma$-$\theta$Net), and (c) a model trained without directional space in the manipulator ($\sigma$-$\theta$-$M_b$ as mentioned in section 6.3.4) are showcased. The proposed model ($\sigma$-$\theta$Net) exhibits minimal artifacts, as evident in the highlighted patch (extracted from the bounding box) and the temporal slice (derived from the red strip).

gradient computation. Each directional gradient phase ($\phi$) and magnitude ($R$) features along particular orientation $\theta_i$, are computed and given to function $r(\cdot)$ (more details are present in the Figure 6.1 (c)) to generate the motion magnified features. These features are merged with softmax attention weights ($W_i^\theta$) along all the directions. The magnified gradient ($M^{\sigma,\theta}$) at a particular resolution is given as

$$\nabla_{\phi_m}^{\sigma,\theta} = \sum_{i=0}^{8} W_i^\theta * \left( \nabla_{\phi_{t_2}}^{\sigma,\theta_i} + r\left( \alpha \left( \nabla_{\phi_{t_2}}^{\sigma,\theta_i} - \nabla_{\phi_{t_1}}^{\sigma,\theta_i} \right), \nabla_{\phi_{t_2}}^{\sigma,\theta_i} \right) \right) \tag{6.7}$$

$$\nabla_{R_m}^{\sigma,\theta} = \sum_{i=0}^{8} W_i^\theta * \left( \nabla_{R_{t_2}}^{\sigma,\theta_i} + r\left( \alpha \left( \nabla_{R_{t_2}}^{\sigma,\theta_i} - \nabla_{R_{t_1}}^{\sigma,\theta_i} \right), \nabla_{R_{t_2}}^{\sigma,\theta_i} \right) \right) \tag{6.8}$$

$$M^{\sigma,\theta} = \nabla_{R_m}^{\sigma,\theta} * \cos\left( \nabla_{\phi_m}^{\sigma,\theta} \right) + \nabla_{R_m}^{\sigma,\theta} * \sin\left( \nabla_{\phi_m}^{\sigma,\theta} \right) \tag{6.9}$$

Here, the parameter $\alpha$ represents the magnification factor. The manipulator features $M^{\sigma,\theta}$ at different resolutions are fused to generate the final magnified motion features. Experimentally, nine directions are considered in the manipulator block. Additionally, detailed information about the manipulator block is depicted in Figure 6.1 (a), (b), (c). The proposed learnable direction and scale based motion manipulator assist the network in determining optimal features to minimize the effects of noise in the motion feature map.

### 6.2.2 Decoder Block

The main role of the decoder block is to synthesize the magnified output with rich texture from the manipulator motion features. To achieve this, the motion manipulator features are first added to the input image $I_{t_2}$, and then given to the decoder block to generate the motion magnified output. Texture synthesis in lightweight models is a difficult problem. To

solve this, we extract the multi-scale information using different dilation rate convolutions. Here, we assume that with an increase in scale (dilation rate), less texture information will be allowed to pass on. Hence, we can decrease the number of parameters by reducing the input information beforehand with channel compression. Further, compression of channel features encourages the network to retain only dense, meaningful representation, so redundant information in the feature space (which includes artifacts) can be reduced. Also, we assume that the output quality is dependent on how to exploit $I_{t_2}$, which is closer to the magnified output as compared to $I_{t_1}$. For this, $I_{t_2}$ is concatenated with the input features and then given to the multi-scale channel compression block ($MSCC$) (as shown in Figure 6.1 (e)). The magnified features using $MSCC$ block are generated at different resolutions. The lower-resolution features are utilized to reduce computation overhead, while the higher-resolution magnified features are exploited to further enhance the output texture quality. The detailed framework is illustrated in Figure 6.1 and is explained in the section below.

**Multi-Scale Channel Compression Block ($MSCC$):** The block takes the latent features ($f_{l-1}$), concatenates ($\bigodot$) them with $I_{t_2}$, and feeds them into a convolutional layer ($C_{3\times3\times c}(\cdot)$) with a $3\times3$ kernel size, featuring $c$ channels, such that $\tilde{f}_l = C_{3\times3\times c}(f_{l-1} \bigodot I_{t_2})$. In the block, parallel layers with distinct channel compression ratios ($c/2^i$) are utilized, with various convolutional layers $C^d_{3\times3\times c}(\cdot)$ with dilation rates $d = 2^i$, where $i \in \{1, 2, 3\}$. To determine the optimal features among different dilation rates, we employed softmax-based weights ($W^d_i$) for feature averaging to generate $MSCC$ output ($f_l$) as

$$f_l = W^d_0 * \tilde{f}_l + \sum_{i=1}^{3} W^d_i * C^{2^i}_{3\times3\times c}(C_{3\times3\times(c/2^i)}(\tilde{f}_l)) \tag{6.10}$$

### 6.2.3 Training Details

**Dataset:** Our model training relies on the synthetic dataset provided by [29], which consists of 7,000 images of objects from the PASCAL VOC dataset [43] as the foreground, and 200,000 images from the MS COCO dataset [44] as the background. Various foreground objects are combined with different backgrounds at different positions to generate random motion. The training dataset comprises a total of 100,000 input-output image pairs, each with a size of $384 \times 384$.

## 6.3 Results

**Loss Function and Training:** We employ the $L_1$ loss function as it is sensitive to individual pixels, emphasizing texture details. However, it can lead to blurring at the edges of the object. To address this, we utilize the edge loss function $L_{EL}$ to make the

Table 6.2: The parameters and GFLOPs, measured at 720 x 720 resolution, are compared with LB-MM (ECCV-2018) [29], LW-MM [95] (WACV-2023), MD-MM (CVPR-2023) [96], and the proposed $\sigma$-$\theta$Net.

| **Methods** | [29] | [95] | [96] | $\sigma$-$\theta$Net |
|---|---|---|---|---|
| **Parameters** | 0.98 M | 1.1 M | 0.117 M | 0.085 M |
| **GFLOPs** | 268.6 | 375.5 | 65.4 | 48.8 |

Table 6.3: Results were generated using parameters, and steps provided by the respective authors. The source code and pre-trained models were obtained from the official pages of Jerk-Aware [31], Anisotropy [32], MD-MM [96], LB-MM [29], and LW-MM [95].

| **Methods** | **Video** | **Magnification Factor** | **Frequency** |
|---|---|---|---|
| Ours ($\sigma$-$\theta$Net) | Gun | 10 | N/A |
| Ours ($\sigma$-$\theta$Net) | Cat toy | 10 | N/A |
| Ours ($\sigma$-$\theta$Net) | Drill | 10 | N/A |
| Ours ($\sigma$-$\theta$Net) | guitar | 10 | N/A |
| Ours ($\sigma$-$\theta$Net) | Quantitative analysis videos | 50 | N/A |
| MD-MM (CVPR-2023) | Gun | 10 | N/A |
| MD-MM (CVPR-2023) | Cat toy | 10 | N/A |
| MD-MM (CVPR-2023) | Drill | 10 | N/A |
| MD-MM (CVPR-2023) | Quantitative analysis videos | 50 | N/A |
| LW-MM (WACV-2023) | Gun | 10 | N/A |
| LW-MM (WACV-2023) | Cat toy | 10 | N/A |
| LW-MM (WACV-2023) | Drill | 10 | N/A |
| LW-MM (WACV-2023) | Quantitative analysis videos | 50 | N/A |
| LB-MM (ECCV-2018) | Gun | 10 | N/A |
| LB-MM (ECCV-2018) | Cat toy | 10 | N/A |
| LB-MM (ECCV-2018) | Drill | 10 | N/A |
| LB-MM (ECCV-2018) | Quantitative analysis videos | 60 | N/A |
| Jerk-Aware | Gun | 10 | 20 |
| Jerk-Aware | Cat toy | 10 | 3 |
| Jerk-Aware | Drill | 25 | 3 |
| Jerk-Aware | Quantitative analysis videos | 200 | 15 |
| Anisotropy | Gun | 100 | 20 |
| Anisotropy | Cat toy | 100 | 3 |
| Anisotropy | Drill | 100 | 3 |
| Anisotropy | Quantitative analysis videos | 400 | 15 |

network more responsive to edges. Additionally, to enhance the perceptual quality of the objects, we introduce the perceptual loss function $L_{PL}$. The final loss function is illustrated in Eq. 6.11

$$L_{loss} = \lambda_1 L_1 + \lambda_2 L_{EL} + \lambda_3 L_{PL} \tag{6.11}$$

$\lambda_1$=10, $\lambda_2$= 1, and $\lambda_3$=0.1 values are used for training [96]. The learning rate is set to 0.0001 and an ADAM optimizer is used. For training, the input images are normalized from -1 to 1. Noise is added to the input, while training. In the network layers, the elu non-linear activation function and in the final layer tanh activation function is used. In the current framework, the proposed lightweight model $\sigma$-$\theta$Net has $c = 12$, number of channels, and after down sampling, the number of channels of decoder $MSCC$ block is increased to $2 * c$. The specifications of existing SOTA and the proposed $\sigma$-$\theta$Net are given in Table 6.2.

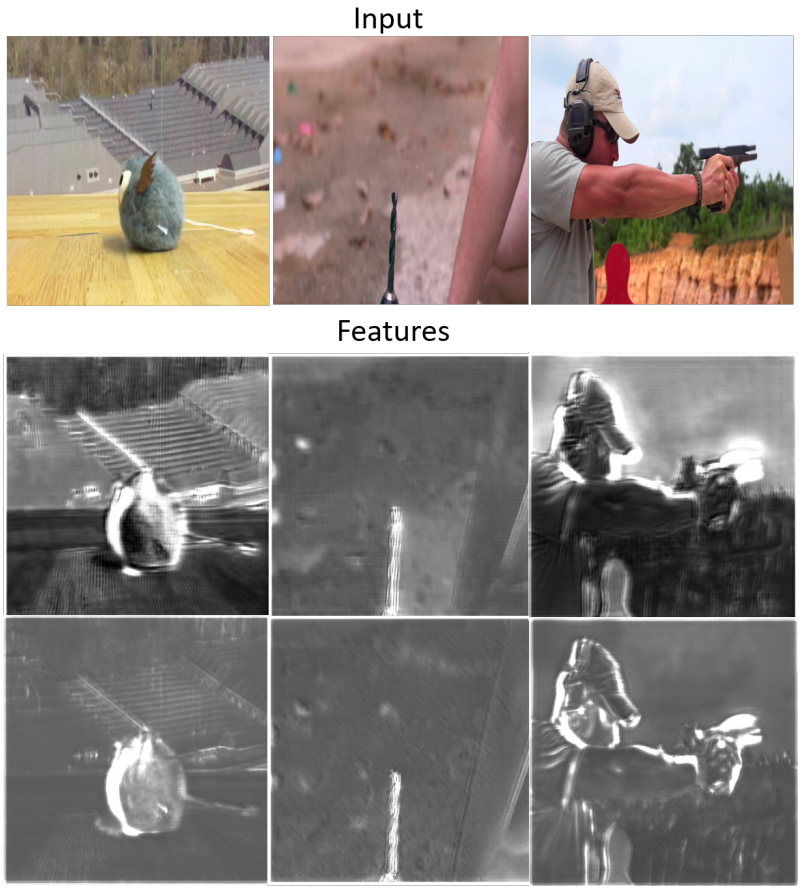


Figure 6.3: Intermediate features (row-2,3) of our proposed $\sigma$-$\theta$Net network, highlighting the motion.

The proposed framework magnifies subtle motion in different scenarios (static and dynamic). Further, it is able to magnify motion in noisy environments. To evaluate the proposed methods, qualitative and quantitative analysis is done in different scenarios and compared with SOTA methods [29], [95], [96], [32]. The details of parameters used for result generation are shown in Table 6.3. The qualitative analysis illustrates, how the proposed method is able to highlight subtle motion in different scenarios (Figure 6.3

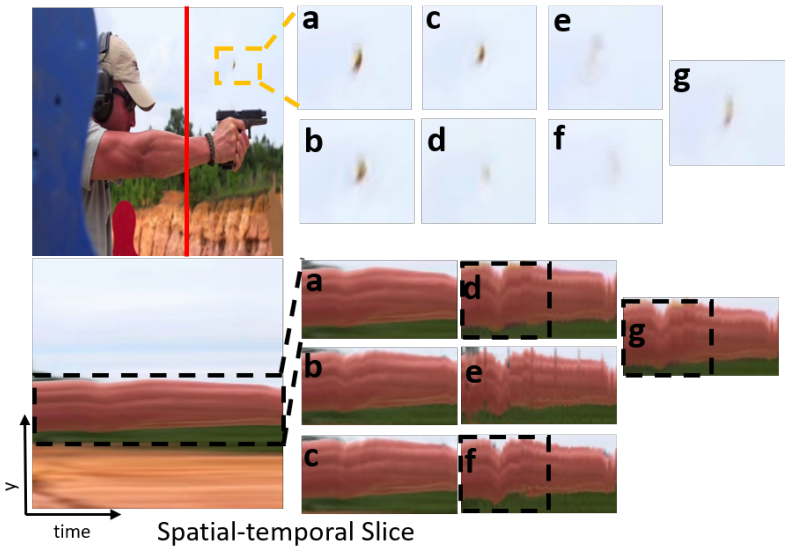


Figure 6.4: **Dynamic Scenario:** In the upper section, a bullet shell is shown to highlight challenges posed by a fast moving small object. The spatio-temporal slice helps to visualize the magnification of subtle motion due to gun recoil. (a) Input, (b) Jerk-Aware (CVPR-2018) [31], (c) Anisotropy (CVPR-2019) [32], (d) MD-MM (CVPR-2023) [96], (e) LB-MM (ECCV-2018) [29], (f) LW-MM [95] (WACV-2023), and (g) the proposed $\sigma$-$\theta$Net. The proposed method highlights the details of the gun shell while producing good magnification on subtle motion generated by the gun recoil.

shows the intermediate motion features), while the quantitative analysis measures the performance of the methods in noisy scenarios. Further, an experiment is conducted to show frequency selectivity. In the end, a detailed ablation study is provided, to illustrate the impact of different modules of the proposed method.

### 6.3.1 Qualitative Analysis

For qualitative analysis with the SOTA methods following different scenarios are considered.

**Dynamic Scenario:** A Gun shooting video as depicted in Figure 6.4 is used. The video contains a large background movement due to camera motion and a quick gun recoil motion in the foreground. The objective is to magnify subtle movements such as forearm motion generated due to the gun recoil, in the presence of large camera motion with fewer artifacts.

**Subtle Motion in Large Moving Object:** For this, we take a toy video as illustrated in Figure 6.5. In the video, the toy is moving along the table as indicated by the arrow, while the toy moves, it produces minute vibrations. Our, goal is to magnify these subtle changes even in the presence of large movements, with fewer texture artifacts.

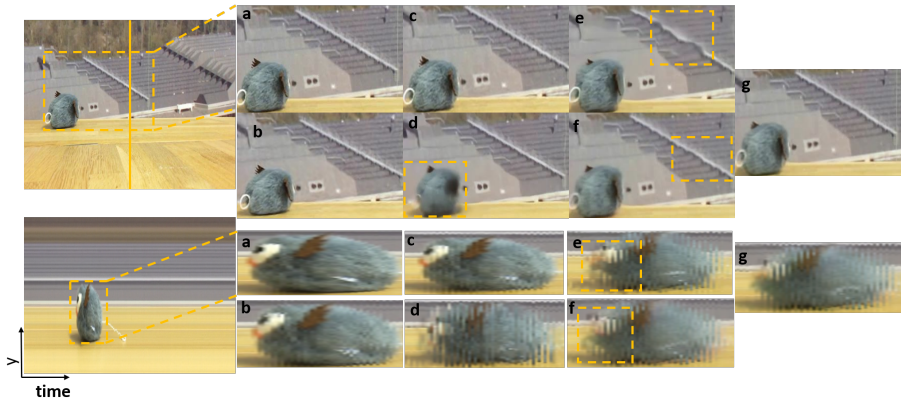


Figure 6.5: **Subtle Motion in Large Moving Object:** In the left, frames of the toy video are presented where the arrow shows the direction of the toy movement, while on the right, temporal slices extracted from the red strip are depicted. Results for (a) Input, (b) Jerk-Aware (CVPR-2018) [31], (c) Anisotropy (CVPR-2019) [32], (d) LW-MM [95] (WACV-2023), (e) LB-MM (ECCV-2018) [29], (f) MD-MM (CVPR-2023) [96], and (g) the proposed $\sigma$-$\theta$Net are shown. The proposed method demonstrates minimal distortion of shape while achieving substantial magnification, whereas other methods either produce small magnification or produce artifacts (highlighted within the bounding box).

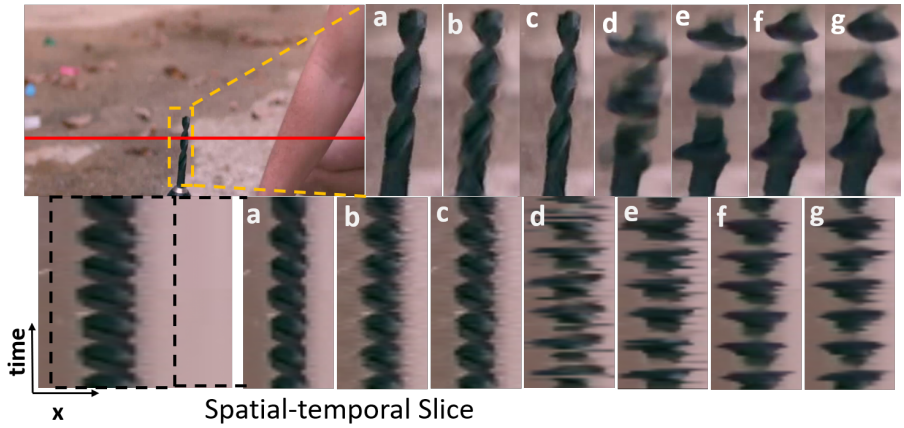


Figure 6.6: **Rotational Motion:** The top panels showcase enlarged image frames within a yellow bounding box, while the bottom panels depict spatio-temporal slices along the red line for (a) Input, (b) Jerk-Aware (CVPR-2018) [31], (c) Anisotropy (CVPR-2019) [32], (e) LW-MM [95] (WACV-2023), (f) MD-MM (CVPR-2023) [96], and (g) the proposed $\sigma$-$\theta$Net. Notably, the other methods either produce less magnification or distort the shape, whereas the proposed method achieves good magnification with minimal artifacts.

**Rotational Motion:** Figure 6.6 shows a hand drill video. The hand drill has rational motion along its axis. To analyze the rotational motion, we have taken a static video. In 2D, we perceive the drill rotational motion as a spiral motion. Our goal is to increase the spiral motion (an increase in spiral motion is shown as more outwards extension of the rod radius).

The Oh *et al.* [29] method distorts the motion, noticeable as spikes in spatio-temporal slices in Figure 6.4, 6.5, 6.6. On the other hand, [31] and [32] produce minimal magnification.

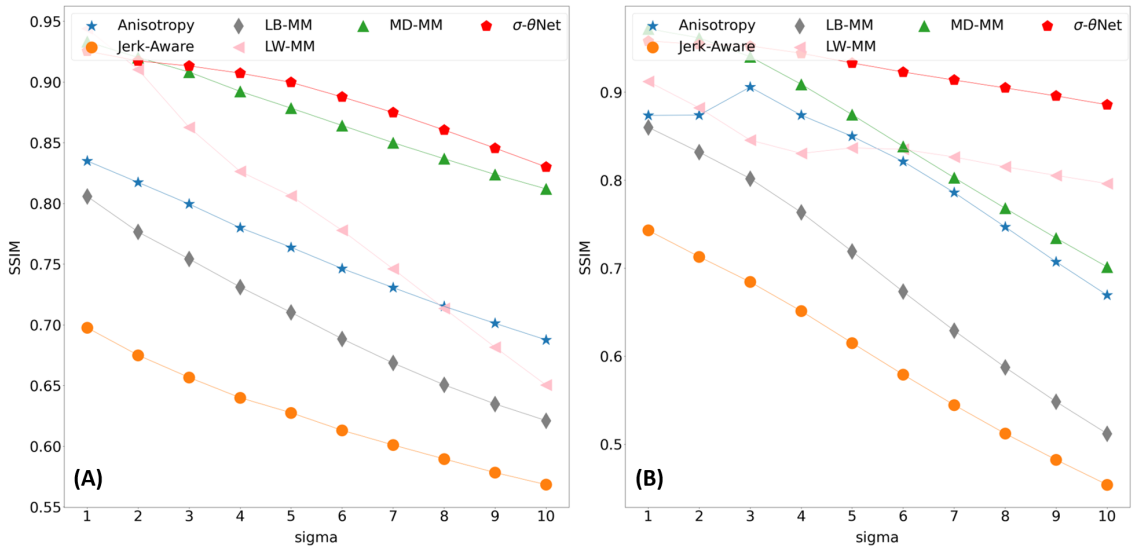


Figure 6.7: Effects of increase in noise with output SSIM is shown on Jerk-Aware (CVPR-2018) [31], Anisotropy (CVPR-2019) [32], LB-MM (ECCV-2018) [29], LW-MM [95] (WACV-2023), MD-MM (CVPR-2023) [96], and the proposed $\sigma$-$\theta$Net. (A) Shows the average SSIM values computed across 25 synthetic videos with motion and (B) without input motion magnified videos output.

Furthermore, methods like [95] and [96] distorts fast-moving small objects, as shown in Figure 6.4, and sometimes introduce shape artifacts, as illustrated in Figure 6.5 (changes in straight lines in the background) and Figure 6.6 (changes in shape as shown in the frames). In contrast, the proposed network $\sigma$-$\theta$Net, yields good magnification with minimal distortions, while being less computationally complex as compared to SOTA methods (as shown in Table 6.2).

### 6.3.2 Quantitative Analysis

To evaluate the efficacy of the proposed network across diverse scenarios in the presence of noise, we conducted an experiment using synthetic videos [96]. The purpose of motion magnification is to produce large motion with fewer artifacts. Better quality at smaller magnification ignores the purpose of motion magnification. So, the synthetic input videos have 0.1 pixels motion in the presence of noise, with respect to that the ground truth videos have 10 pixel change. Each method is required to produce 100× magnification compared to the input video to match the ground truth. Further, to analyze the network's ability to ignore noise based changes, synthetic videos are created without motion, utilizing the same scenario as described above. For this, input videos are considered as the ground truth. Given the absence of motion in these videos, we expect the output to ideally be the same as the input. The SSIM values are computed across videos both with and without motion and are shown in Figure 6.7. From Figure 6.7 the proposed network $\sigma$-$\theta$Net exhibited the highest SSIM values for the same amount of output motion with minimum parameters.

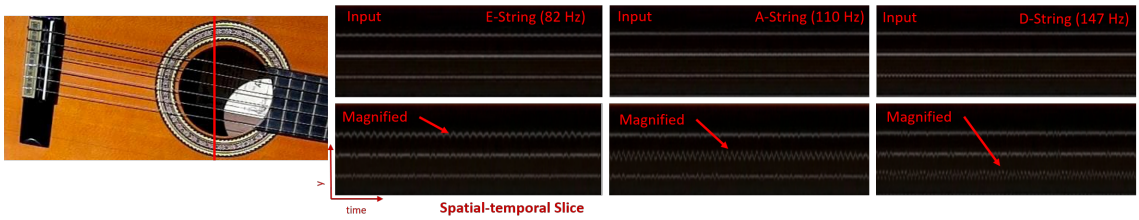


Figure 6.8: For frequency selectivity, guitar strings in motion across different frequencies are utilized. Motion is observed across the E, A, and D strings. Magnifying these motions after temporal filter pre-processing helps in accentuating the specific features of motion associated with each string, as illustrated above.

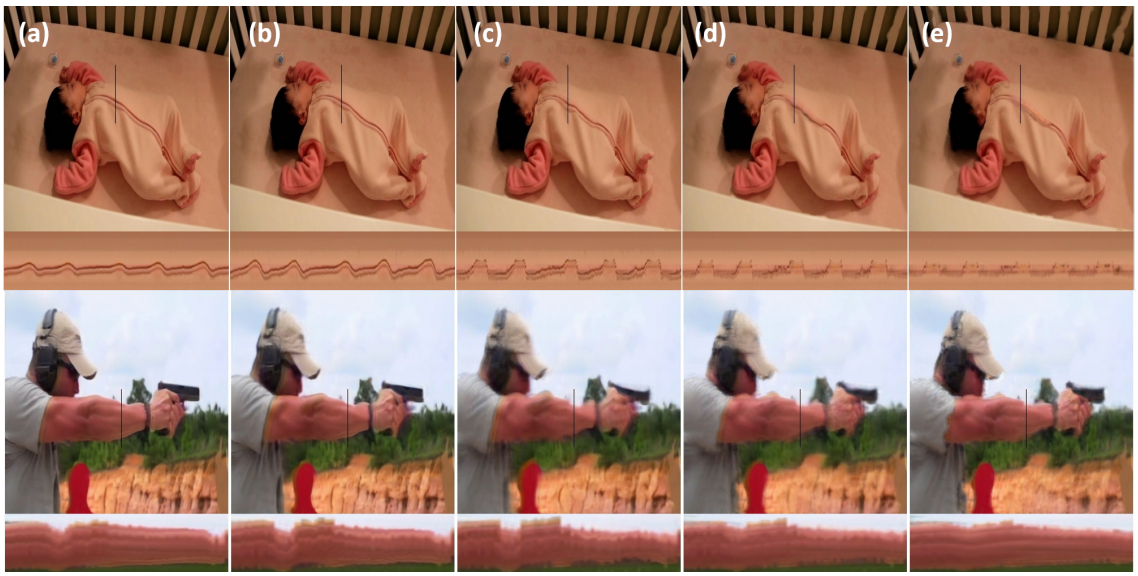


Figure 6.9: **Effects of change in Magnification Factor** Figure illustrates the proposed model output. Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output.

### 6.3.3 Additional Experiment

**Frequency selectivity:** When using a deep learning method that is not directly trained with temporal filters, applying temporal filters on intermediate features may lead to inaccurate outcomes. To mitigate this issue, we initially preprocess the video using a temporal filter to suppress undesired motion. In this context, we utilize the output of [26] method at a small magnification factor, which is then fed as input to our method. Visual results are shown in Fig 6.8.

**Effects of increase in Magnification Factor:** The figures 6.9, depict the effects of varying magnification factors on the output of the proposed models. These visualizations offer insights into model performance under varied magnification factors, highlighting their robustness and limitations.

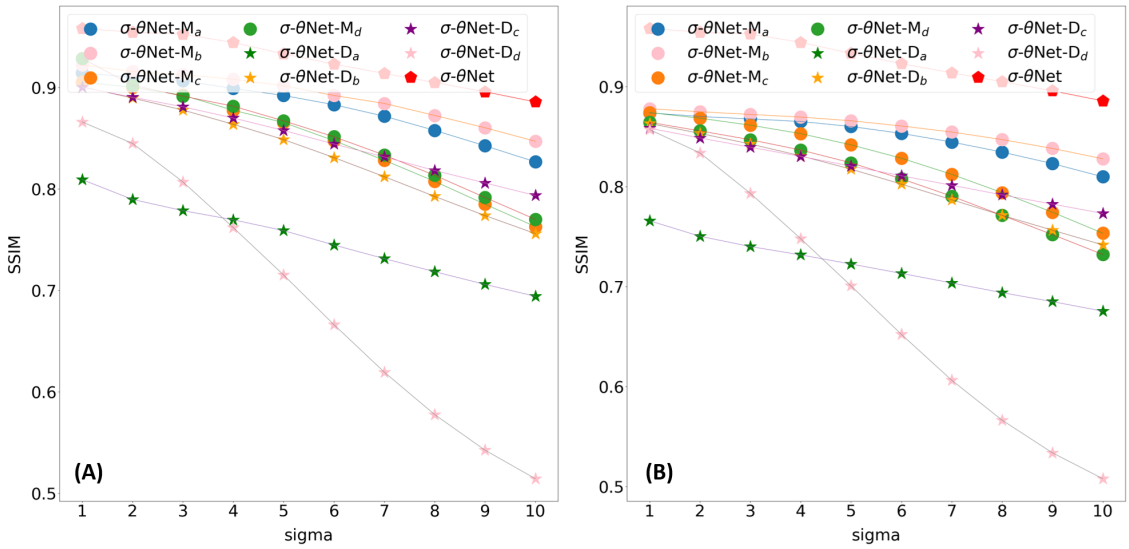


Figure 6.10: The SSIM values across a set of 25 synthetic videos, both (A) with motion and (B) without motion. The results are shown for model $\sigma$-$\theta$-$M_a$ to $\sigma$-$\theta$-$M_d$ and $\sigma$-$\theta$-$D_a$ to $\sigma$-$\theta$-$D_d$. The model details are provided in the ablation section.

### 6.3.4 Ablation

To assess the contribution of various components within the proposed framework, we conducted an ablation study. Different combinations of the model are trained and tested on the same videos as mentioned in quantitative analysis. Initially, to measure the effectiveness of the proposed manipulator, we conducted experiments by training a model $\sigma$-$\theta$-$M_a$ without incorporating the learnable scale layer in the manipulator, followed by another model $\sigma$-$\theta$-$M_b$ trained without rotation space. This allows us to analyze the individual contributions of scale and direction space in the proposed method. Additionally, we also explored various combinations, such as $\sigma$-$\theta$-$M_c$ without phase changes, and $\sigma$-$\theta$-$M_d$ without magnitude changes. The results presented in Figure 6.10 demonstrate the superior performance of the proposed method $\sigma$-$\theta$Net, compared to the other configurations.

To analyze the decoder, $\sigma$-$\theta$-$D_a$ a model is trained with the same dilation rate in the decoder block to see the effect of multi-scale, while another model $\sigma$-$\theta$-$D_b$ is trained without channel compression, to highlight the effectiveness of the dense representation. Additionally, a model $\sigma$-$\theta$-$D_c$ is trained with the residual block for establishing a standard comparison, considering the widespread use of residual blocks in various image translation tasks. Please note that the proposed channel compression block is different from the squeeze and excitation block as proposed in [98]. $\sigma$-$\theta$-$D_d$ model is trained with squeeze and excitation block, to show the differences. From the quantitative results analysis in Figure 6.10, it is observed that the proposed decoder block of $\sigma$-$\theta$Net has the highest SSIM values.

## 6.4 Summary

In this chapter, we introduce a novel learnable directional scale-space filters-based network, $\sigma$-$\theta$Net, for video motion magnification. It consists of a motion manipulator block and a decoder block. Our approach leverages learnable directional scale space phase and magnitude information for motion manipulation. The decoder block employs channel compression with varying compression ratios and incorporates different dilation rate convolutions to integrate scale space. This facilitates the generation of good quality magnified output while mitigating the impact of noise. The proposed method is evaluated qualitatively and quantitatively on both real and synthetic videos. An additional experiment is conducted to show the compatibility of the proposed method with frequency selectivity. Furthermore, an ablation study is performed to examine different components of the proposed approach. The results demonstrate that the proposed network $\sigma$-$\theta$Net achieves better results qualitatively and quantitatively compared to the SOTA methods for video motion magnification. However, in scenarios with high levels of noise, there's a demand for methods that exhibit robustness to noise while also possessing fewer parameters.

# Chapter 7

# A Hierarchical Network Based Approach for Video Motion Magnification

Earlier video motion magnification methods [33, 26, 28, 27] perform well in static scenarios, but they produce blurry results in dynamic scenarios. Later approaches [31, 32, 30] work in static as well as in dynamic scenarios. However, they produce little magnification with ringing artifacts. Oh *et. al.* [29], introduced a deep learning-based solution that can construct results without ringing artifacts and high magnification in dynamic scenarios. However, sometimes due to inadequate separation of shape and texture features, magnification of input noise, illumination changes, or other unwanted changes, cause spurious motion and blurry distortions. Later, different deep learning based approaches are being proposed to mitigate this issue using transformers [47], proxy model based feature loss [99],or processing in frequency and spatial domain [96] *etc.* However, these techniques are either computationally complex or their output quality decreases with an increase in input noise.

To address these limitations, we propose a hierarchical magnification network for video motion magnification. LWVMM [99] employs residual blocks that are not specifically tailored for reducing noise in motion features. Additionally, the proxy model solely focuses on features from which distance should decrease; incorporating negative samples could be useful. Also, it generates noise-robust motion features at a single scale and does not exploit the hierarchical structure used in handcrafted methods. Due to these design choices, [99] is sensitive to noise-induced changes. To mitigate these issues, we propose a lightweight model that is more robust to noise compared to state-of-the-art methods, producing effective magnification and better quality. Moreover, the proposed method relies on edge features instead of input features for motion extraction, eliminating the need for encoders as used in [99] for feature separation. This design choice makes our model lightweight and more memory-efficient compared to [99], rendering it an appropriate choice for various computer vision-based healthcare and industrial applications of motion magnification. The main contributions of the work are as follows:

- A hierarchical magnification network with a multi-resolution decoder is proposed for adequate magnification and reconstruction (from lower resolution to higher resolution) of the output frame.
- Multi-Scale Manipulator block is proposed with edge-based difference input for reduction of noise before magnification.
- A novel contrastive loss is proposed for further enhancing the robustness of the magnification process towards noise.
- We have also done extensive qualitative and quantitative analysis for a more adequate evaluation of the proposed method.

These solutions are explained in detail in the following sections.

## 7.1 Proposed Method

Handcrafted filter-based techniques, such as [30, 31, 32] take multiple frames as input, use complex steerable pyramids for spatial feature decomposition and temporal filters for motion manipulation. These methods rely on narrow bandpass filters to mitigate the impact of noise. [29] puts regularization on encoders to separate shape information; however, sometimes inadequate separation of shape features, along with the magnification of noise, can lead to blurry distortions and superious motion. Similarly, [99] separates appearance information from input using regularization. However, the proposed method did not employ any regularization but instead utilized differences in edge features of the input frame for motion information extraction. This helps in the reduction of complexity. We assume most of the noise is present in the texture area. So, using edge features for extracting motion information allows reduction of noise before magnification *(i.e. before multiplication with the magnification factor).* In this regard, we propose a Multi-Scale Manipulator ($MSM$) block that processes the edge differences and input features. Additionally, the proposed feature-based contrastive loss assists in reducing the impact of noise. Together, these elements help the proposed network to convey better magnification results while mitigating the effects of noise.

While [26, 27, 28] employ steerable pyramids to extract multi-scale features and utilize temporal filters at individual levels to generate multi-level magnification, they render the final output using a steerable reconstruction pyramid by merging the multi-level magnification features. In contrast, [99] utilizes only single-level information for reconstruction. Whereas, multi-level magnified features can help in the better reconstruction of the output [26, 27, 28]. Drawing inspiration from [26, 27, 28] methods, we propose the utilization of features from multi-scale manipulator ($MSM$) blocks with

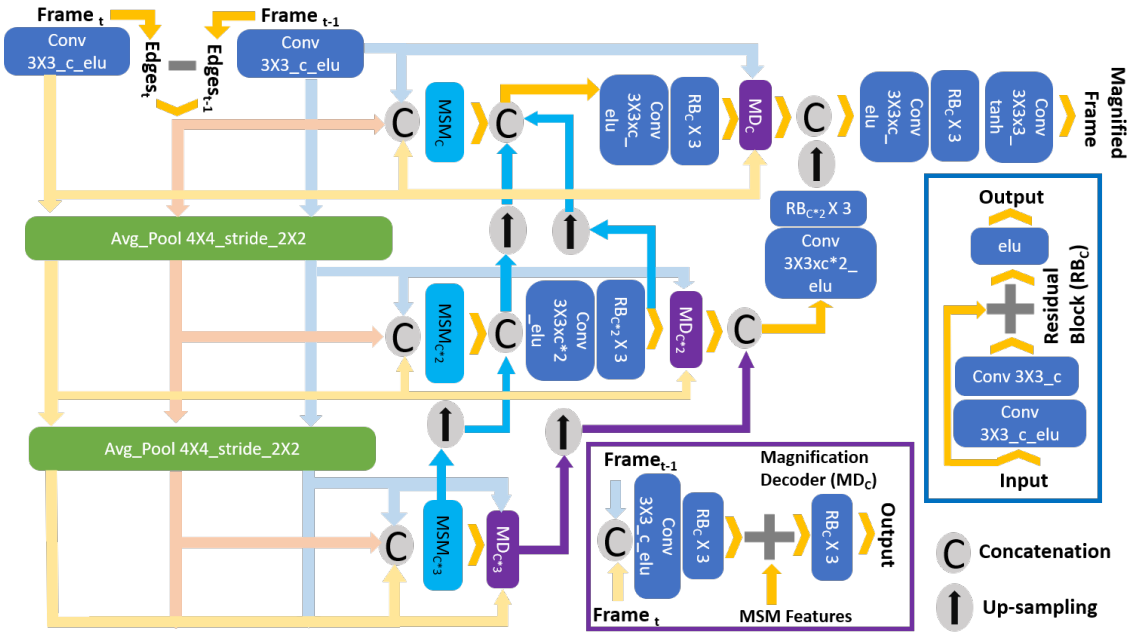


Figure 7.1: Our proposed architecture for motion magnification. It takes $Frame_t$, $Frame_{t-1}$, and the magnification factor $M_f$ as input. The $MSM$ block is used to generate motion manipulation features, and the $MD$ block is used to reconstruct a magnified frame based on these features. $c$ describes the number of channels.

multi-resolution decoder ($MD$) features for adequate reconstruction and magnification of the output frame, from lower resolution to higher resolution.

### 7.1.1 Hierarchical Magnification

The proposed network generates magnified features, progressing from lower to higher resolutions. At each level, input features and edge differences undergo processing through the $MSM$ and $MD$ blocks to generate the appropriate magnified features. These are then hierarchically merged to render the final magnified frame. Similar to the spatial pyramids, different resolution magnified features aid in better output reconstruction. In contrast to [99], where appearance features are generated, we utilize edge features, as they are more robust to illumination changes *(Sobel operator is used for edge extraction)*. At different levels of the network, the $MSM$ block uses different receptive fields to generate output features. Dissimilar receptive fields will face different noises. So, each output feature of the $MSM$ block at distinct resolutions will have different denoising effects. Merging these features further improves the network's robustness.

The $MSM$ block output holds the motion manipulation information, which is passed to the magnification decoder $MD$ block. The $MD$ block develops suitable texture features from the input features and adds them to the $MSM$ output. It uses residual blocks to render the combined features to appropriate magnified output according to MSM features. Also,

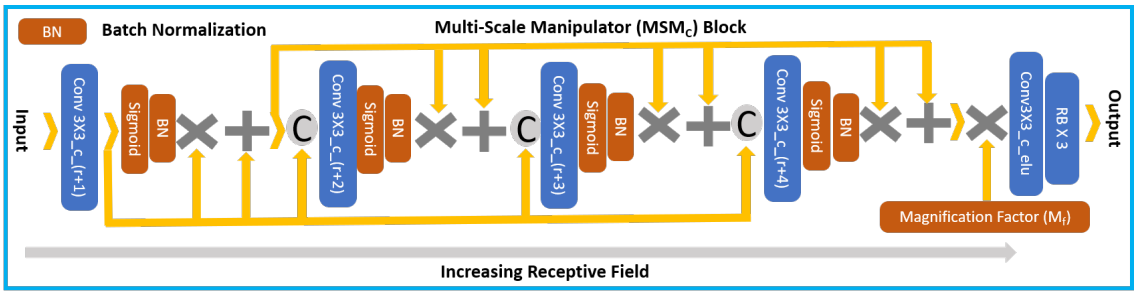


Figure 7.2: Overview of the proposed Multi-Scale Manipulator Magnification ($MSM$) Block, showing an increase in the receptive field from left to right

in the $MD$ block, taking two frames as input aids in denoising the texture area. The $MD$ block's output at each level is up-sampled and concatenated with the subsequent higher level $MD$ block's output. This combined output is then provided to residual blocks to reconstruct the final magnified output.

### 7.1.2 Multi-Scale Manipulator (MSM) Block

To lessen the effects of the magnification of noise, multi-scale convolutions are applied to different features. As noise is random in nature, it will affect various parts of the image differently. So, a dilated convolution with varying receptive fields can see distinct samples of noise for the same central element. Also, we assume that when higher receptive field convolution is applied to pre-processed features from the lower receptive field, it can see distinct values of noise for the same central element. To leverage these assumptions, the receptive field is increased in a sequential order, such that the output of the previous receptive field is set as input for the next higher receptive field. Thus, each layer can work on pre-processed features of the previous layer and can reduce noise before magnification. Whereas, in a parallel combination, each layer will see the same input, limiting its ability to distinguish noise variations.

Figure 7.2 shows the structure of $MSM$ block. Within this block, the first concatenation operation, denoted as $C(\cdot)$, is applied to input features $E_d^m, F_t^m, F_{t-1}^m$. Whereas $E_d^m \in \mathbb{R}^{h\times w\times c}$ is the difference and $F_t^m, F_{t-1}^m \in \mathbb{R}^{h\times w\times c}$ are the input features of frame $t$ and $t-1$ at level $m \in \{1,2,3\}$ such that $m=1$ is for the highest resolution of $384\times 384$, $m=2$ for $192\times 192$, and $m=3$ for the lowest resolution of $96\times 96$. After concatenation operation, convolution operation ($\otimes$) is applied using kernel $\Psi^{r+1}_{3*3*(c*m)}$, with kernel size $3\times 3\times (c*m)$ and dilation rate $r$ such that $(m,r) \in \{(1,8),(2,4),(3,1)\}$ and $(c*m)$ number of channels to generate output $F^{m,r}$ as :

$$F^{m,r+1} = \Psi^{r+1}_{3*3*(c*m)} \otimes C(E_d^m, F_t^m, F_{t-1}^m); \quad m \in \{1,2,3\}. \tag{7.1}$$

The output of the first convolution layer, denoted as $F^{m,r+1}$, is passed to other layers

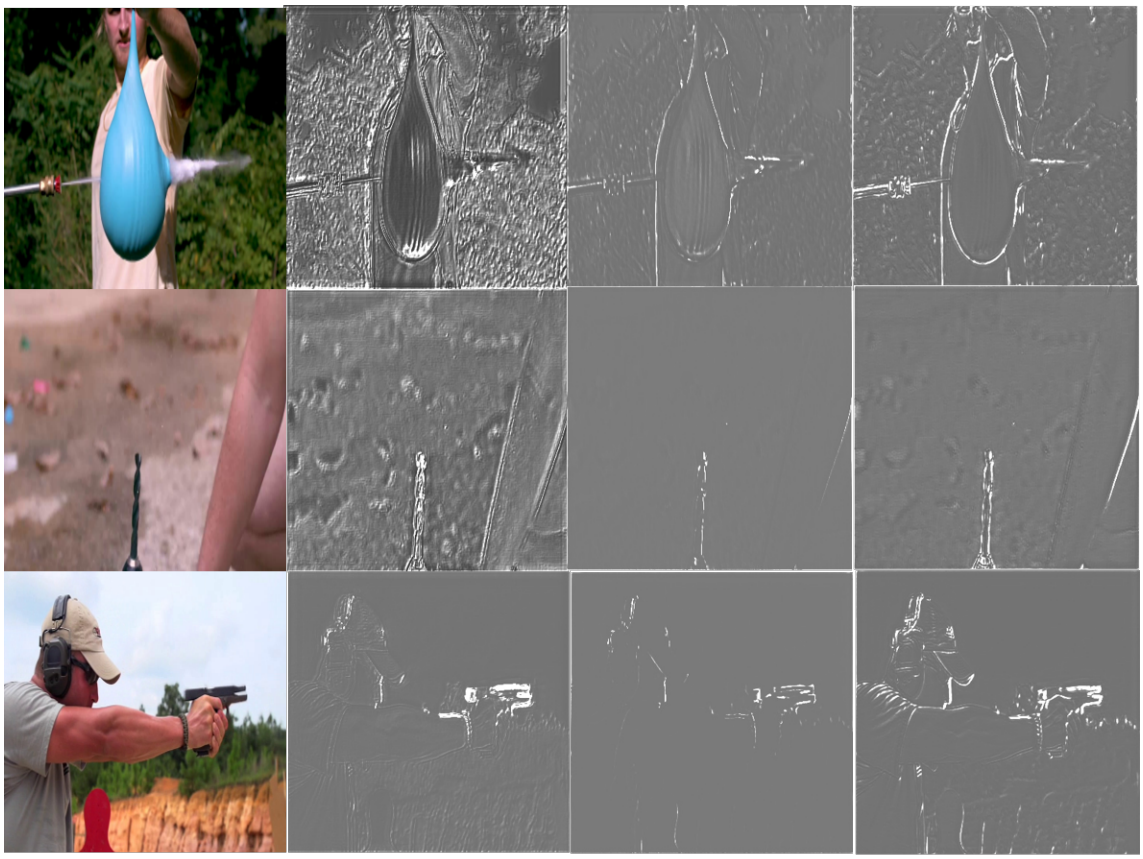


Figure 7.3: MSM Block Features after Multiplication with Magnification Factor. (a) Input frames of Balloon Video, Hand Drill Video, and Gun-shooting Video, respectively. (b), (c), and (d) are the intermediate features that highlight the motion parts.

through concatenation. This is done to provide adequate input information to all the layers. On $F^{m,r+1}$ features, $\sigma$ sigmoid activation, and $\zeta$ batchnorm are applied to highlight features of interest. Batch normalization is used only in the $MSM$ block to re-scale the features. Further, element wise multiplication ($\odot$), and addition operations ($\oplus$) are performed on normalized features with $F^{m,r+1}$ as:

$$W^{m,r+1} = F^{m,r+1} \oplus \left(F^{m,r+1} \odot \zeta\left(\sigma\left(F^{m,r+1}\right)\right)\right). \tag{7.2}$$

The output of Eq. (7.2) will act as base features, which will be further enhanced by the subsequent layers. The remaining operations are depicted in Eq. (7.3) to produce output $W^{m,r+4}$.

$$W^{m,r+n+1} = W^{m,r+1} \oplus \left(W^{m,r+1} \odot \zeta\left(\sigma\left(\Psi^{r+n+1}_{3\times3\times(c\times m)} \otimes C\right)(W^{m,r+n}, F^{m,r+1})\right)\right); \quad n \in \{1,2,3\}. \tag{7.3}$$

Where $W^{m,r+4}$ is multiplied by the magnification factor ($M_f$), which controls the amount

of magnification. Subsequently, the convolution layer and residual blocks are used to generate the final motion manipulated features. Figure 7.3 the intermediate features of MSM block.

### 7.1.3 Proposed Feature Based Contrastive Loss

Contrastive learning has been utilized to make input latent features get closer to the positive samples while increasing their distance from the negative samples. Various approaches have used positive and negative samples in the image space [100, 101, 102] or its downstream representation on a network [103, 104]. However, these techniques are hand-tailored for specific applications and cannot be applied directly to motion magnification. For instance, [104] uses encoder representation to generate positive and negative samples for image translation problems. Whereas for motion magnification, if noiseless images are passed through the same network (trained for noisy input) to develop positive samples, their representation will have some distortions due to denoising. To solve this, the same network is trained from scratch without adding noise to the input. It acts as a proxy network. So, positive samples are taken from the proxy model, and $MSM$ block features when a noiseless image is given as input. For negative samples, using noisy image features from the proxy network is not beneficial. Since noise is already added to the input while training, the network's loss function will try to move away from noisy representations. So, instead of that, previous epoch features of the $MSM$ block are taken as negative samples. The network will try to get away from its previous motion manipulation features while trying to get closer to the proxy model motion manipulation features. We assume that there are slightly different features from the previous epoch that can cause similar losses. So, the network will oscillate across different minima to find a more optimum minimum (as shown in the ablation study with the proposed contrastive loss, which leads to improvements compared to the scenario where it is not used). However training a network from scratch using this loss function can be noisy and may even prevent its convergence. Instead, the network is fine-tuned to the best-trained weights. Also, fine-tuning is done for a small number of epochs to make the process stable (as over-fitting can otherwise make it unstable). This results in further improvement in the output quality. The proposed contrastive loss ($C_{loss}(\cdot)$) is defined as:

$$C_{loss} = \frac{L\|(N(F_t^{m,r}, F_{t-1}^{m,r})_e - N^*(F_t^{m,r}, F_{t-1}^{m,r})\|_1}{L\|N(F_t^{m,r}, F_{t-1}^{m,r})_e - N(F_t^{m,r}, F_{t-1}^{m,r})_{e-1}\|_1} \tag{7.4}$$

where $N(F_t^{m,r}, F_{t-1}^{m,r})_e$ represents the multi-scale block ($MSM_e^{m,r}$) output features at $e^{th}$ epoch for the input $F_t^{m,r}$ and $F_{t-1}^{m,r}$, for $m, r \in \{(1,9),(2,5),(3,1)\}$ and $*$ indicates the proxy model for the same. $L\|\cdot\|_1$ represents the absolute difference between the two values.

Table 7.1: Comparison of the state-of-the-art deep learning method STBVMM [47], LWVMM [99], MDVMM [96] our base and lightweight model in terms of parameters and FLOPs (calculated at $384 \times 384$ resolution).

| **Model** | **Parameters** | **GFLOPs** |
|---|---|---|
| STBVMM [47] | 31.2 M | 1376.8 |
| LWVMM [99] | 1.10 M | 375.5 |
| Our Base Model ($M_1$) | 2.5 M | 602.5 |
| Our lightweight model ($M_2$) | 0.28 M | 68.3 |
| MDVMM [96] | 0.117 M | 65.4 |

### 7.1.4 Dataset

In our model training, we utilized the dataset provided by [29], maintaining consistency in the experimental setup. This dataset consists of synthetic images, featuring 7,000 foreground objects sourced from the PASCAL VOC dataset [43] and 200,000 backgrounds from the MS COCO dataset [44]. The synthesis process involves merging diverse foreground objects with various backgrounds at different positions, introducing a controlled form of random motion. This procedure results in a comprehensive dataset containing 100,000 sample pairs, each possessing a resolution of $384 \times 384$.

### 7.1.5 Loss Function and Training

We use the $L\|\cdot\|_1$, Edge loss ($L_{EL}\|\cdot\|_1$), perceptual loss ($L_{PL}\|\cdot\|_1$) with the proposed contrastive loss. $L\|\cdot\|_1$ norm loss represents the absolute difference between the predicted and ground truth values. It is pixel sensitive and puts more emphasis on texture. But this can blur the edges of the magnified object. To solve this, the Edge loss function ($L_{EL}\|\cdot\|_1$) [58] is used. But still, the perceptual quality of moving objects is not adequate. Hence, the perceptual loss function ($L_{PL}\|\cdot\|_1$) [59] is used. Thus, the final loss function is depicted as

$$L_{loss} = \lambda_1 L\|\cdot\|_1 + \lambda_2 L_{EL}\|\cdot\|_1 + \lambda_3 L_{PL}\|\cdot\|_1 + C_{loss}(\cdot) \tag{7.5}$$

where $\lambda_1$=10, $\lambda_2$= 1 and $\lambda_3$=0.1 values are utilized for training. Additionally, Gaussian noise is incorporated into the input to simulate noise. A base model with $c = 24$ and a lightweight model with $c = 8$ are proposed. The networks are implemented in the TensorFlow library and trained on an NVIDIA 2080 RTX GPU with 8GB of memory. During the training process, the learning rate is configured to be 0.0001, and the ADAM optimizer is employed. Training the base and lightweight models takes 192 and 96 GPU hours, respectively. Both models are trained for 36 epochs and further fine-tuned for 15 epochs using the proposed contrastive loss. Their characteristics are summarized in Table

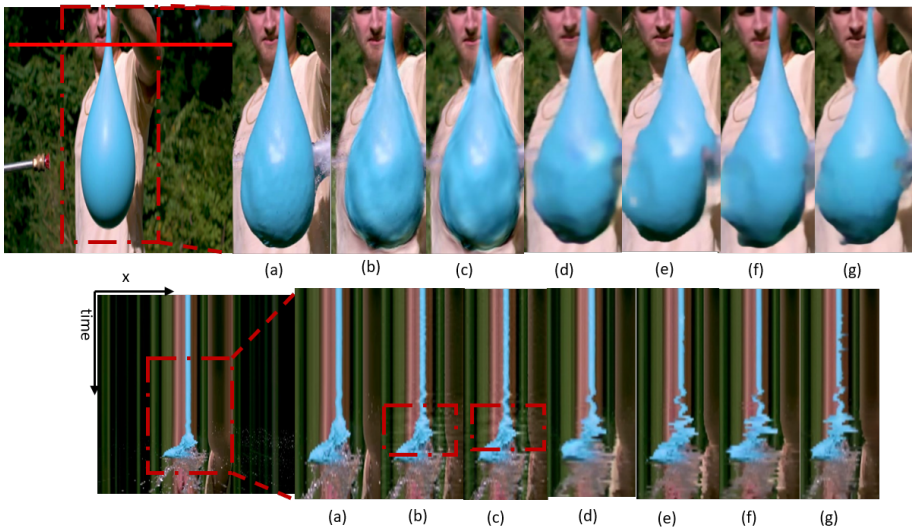


Figure 7.4: Balloon video: Initially, frames from the video are presented, followed by temporal slices extracted from the red strip to depict the motion of balloon bursts. Motion magnification is evident through the enhanced motion observed in both the balloon itself and the corresponding temporal slice, in comparison to the input. (a) Input video, (b) Anisotropy method [32], (c) Jerk-Aware method [31], (d) MDVMM [96], (e) STBVMM [47], (f) LWVMM [99], (g) Our base model, and (h) Our lightweight model, respectively.

7.1, with our lightweight model having the lowest FLOPs.

## 7.2 Experimental Results

The proposed method is evaluated both qualitatively and quantitatively, with state-of-the-art methods such as Jerk-Aware [31], Anisotropic filter [32], STBVMM [47], LWVMM [99], and MDVMM [96] using both natural and synthetic videos. The details of parameters used for result generation are shown in Table 7.2. The following sub-sections include an exhaustive discussion of the comparisons. Additionally, we conducted an experiment to assess the physical accuracy of the proposed method. Furthermore, an ablation study is performed to illustrate the significance of various parts of the network. Results in the ablation study are based on lightweight models before fine-tuning with contrastive loss unless otherwise specified.

### 7.2.1 Real world Videos

**Balloon video** In Figure 7.4, a water cannon is used to rupture the balloon, resulting in both minute and large motion in the balloon. Figure 7.4 shows these motions as spatial-temporal slices extracted from the red strip. We aim to magnify the fine-grained motion of the balloon while minimizing the distortions caused by sudden, large motion. Hand-crafted techniques [31, 32] create ringing artifacts close to the balloon. These

Table 7.2: Parameters of different methods for motion magnified videos. All the results are generated by following the steps given by the respective authors with parameters.

| Methods | Video | $M_f$ | Frequency |
|---|---|---|---|
| Ours ($M_1$,$M_2$ ) | Gun | 10, 10 | N/A |
| Ours ($M_1$,$M_2$ ) | Drill | 3,5 | N/A |
| Ours ($M_1$,$M_2$ ) | Balloon | 9, 9 | N/A |
| Ours ($M_1$,$M_2$ ) | Physical accuracy | 10, 10 | N/A |
| Ours ($M_1$,$M_2$ ) | synthetic videos | 50, 50 | N/A |
| Ours (ablation study) | synthetic videos | 50 | N/A |
| LWVMM [99] | Gun | 10 | N/A |
| LWVMM [99] | Drill | 10 | N/A |
| LWVMM [99] | Balloon | 10 | N/A |
| LWVMM [99] | Physical accuracy | 10 | N/A |
| LWVMM [99] | synthetic videos | 50 | N/A |
| MDVMM [96] | Gun | 10 | N/A |
| MDVMM [96] | Drill | 10 | N/A |
| MDVMM [96] | Balloon | 10 | N/A |
| MDVMM [96] | Physical accuracy | 5 | N/A |
| MDVMM [96] | synthetic videos | 50 | N/A |
| STBVMM [47] | Gun | 10 | N/A |
| STBVMM [47] | Drill | 10 | N/A |
| STBVMM [47] | Balloon | 10 | N/A |
| STBVMM [47] | Physical accuracy | 10 | N/A |
| STBVMM [47] | synthetic videos | 50 | N/A |
| Jerk-Aware [31] | Gun | 10 | 20 |
| Jerk-Aware [31] | Drill | 25 | 3 |
| Jerk-Aware [31] | Balloon | 25 | 3 |
| Jerk-Aware [31] | Physical accuracy | 10 | 15 |
| Jerk-Aware [31] | synthetic videos | 100 | 15 |
| Anisotropy [32] | Gun | 100 | 20 |
| Anisotropy [32] | Drill | 100 | 3 |
| Anisotropy [32] | Balloon | 100 | 3 |
| Anisotropy [32] | Physical accuracy | 200 | 3 |
| Anisotropy [32] | synthetic videos | 200 | 15 |

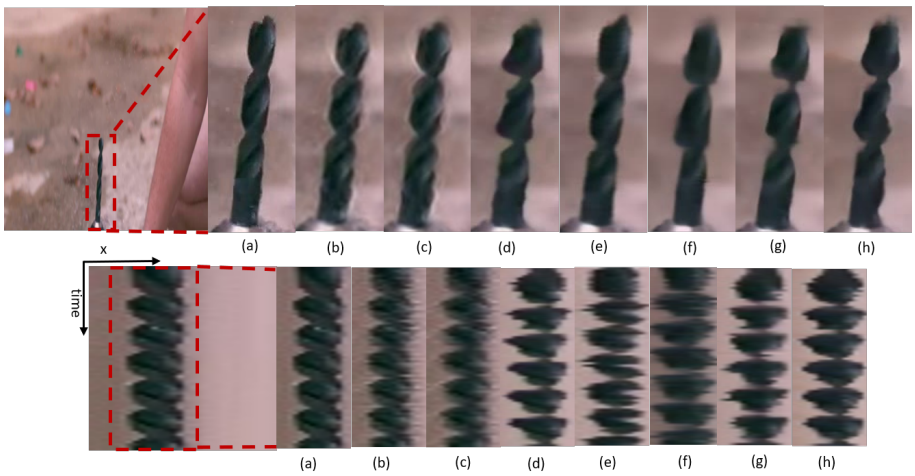


Figure 7.5: Drill Video: The proposed method is compared with state-of-the-art (SOTA) methods for magnification of drill rotational motion. Firstly, the output from each respective method is presented. Following that, spatio-temporal slices, taken from the red strip, are displayed to facilitate a detailed comparative analysis. (a) Input video, (b) Anisotropy method [32], (c) Jerk-Aware method [31], (d) MDVMM [96], (e) STBVMM [47], (f) LWVMM [99], (g) Our base model, and (h) Our lightweight model, respectively. respectively.

artifacts are highlighted in the red bounding box in Figure 7.4 (b),(c) and can be observed as white edges close to the balloon and white spikes in the temporal slices. LWVMM [99] and MDVMM [96] exhibit less magnification in comparison to the proposed method, while [47] is noted for its computational complexity. Hence, the proposed approach has better results with lesser complexity.

**Rotational motion** Magnifying rotational motion is a challenging task. To assess the impact of magnification, we employ a stationary video featuring a hand drill with rotational motion along the drill axis, as depicted in Figure 7.5. In a two-dimensional context, this motion translates into a spiral pattern. Our objective is to enhance the spiral movement, visually represented as a more outward extension of the rod radius in the spatial-temporal slices presented in Figure 7.5. Approaches based on hand-designed filters, such as [31] and [32], introduce ringing artifacts near the rod. These artifacts are discernible as white edges in proximity to the rod and white spikes in the temporal slices, as illustrated in Figure 7.5 (b) and (c). LWVMM [99], produces some blurriness in the texture (observable in Figures 7.5 (f)) and MDVMM [96] has less magnification. In contrast, STBVMM [47] has exhibited promising outcomes, however, it is hampered by notable computational complexity. The proposed method delivers more spiral motion with better quality compared to other methods.

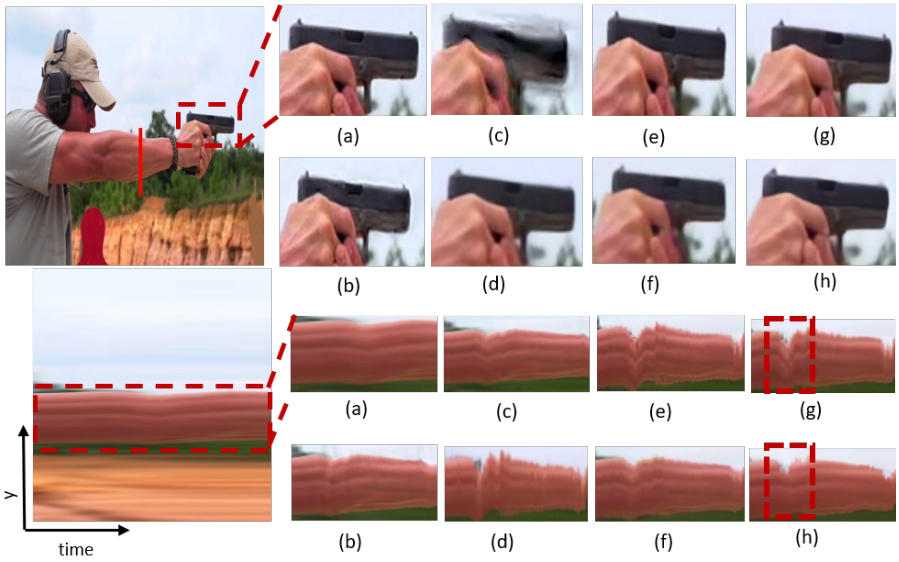


Figure 7.6: Gun-shooting video: Temporal slices taken from the red strip illustrate the impact of gun recoil on the forearm and also facilitate the observation of changes induced by motion magnification. (a) Input video, (b) Anisotropy method [32], (c) Jerk-Aware method [31], (d) LWVMM [99], (f) MDVMM [96], (e) STBVMM [47], (g) Our base model, and (h) Our lightweight model, respectively.

**Gun-shooting Video** The video captures two distinct forms of motion: the significant translation motion induced by camera movement and the subtle movement in the forearm of the shooter resulting from gun recoil. The objective is to amplify the nuanced motions in the forearm despite the presence of substantial camera movements. The forearm motion shown in the spatio-temporal slices is taken from the red strip and shown in Figure 7.6. The heightened motion in the forearm manifests as increased bending in the temporal slice, as highlighted in the red box of Figure 7.6. Comparative analyses with the Jerk-aware method [31] and Anisotropy [32] reveal lower magnification levels compared to the proposed method. LWVMM [99] exhibits less smooth forearm motion due to the magnification of unwanted distortions, visible in Figure 7.6 (d). MDVMM [96] produces modest magnification on the forearm compared to the proposed method. Although STBVMM [47] demonstrates promising results, its high computational complexity is noteworthy. The proposed method excels in generating superior magnification of subtle forearm movements even in the presence of significant camera motion when compared to state-of-the-art methods, and with less computational complexity.

**Physically Accuracy** To investigate this, an experiment was conducted using a universal vibration apparatus as shown in Figure 7.7, inducing subtle up-and-down motions in a mechanical rod. An ultrasonic sensor measured the resulting signal (for more details please see section 3.2.2). Optical flow was computed between magnified and input videos within a marked region. Optical flow used $frame_{t-1}$ as reference and $frame_t$ from the

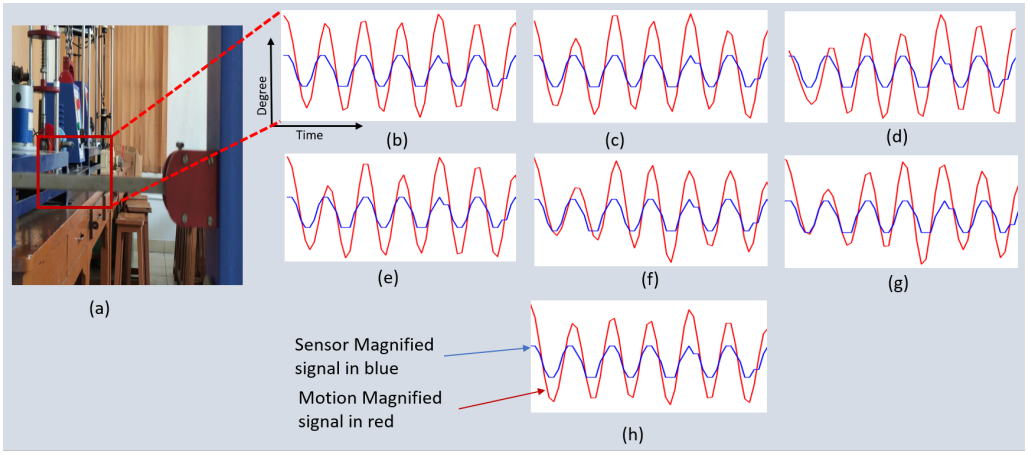


Figure 7.7: Physical Accuracy: Comparing our method with the output of other state-of-the-art (SOTA) methods, both are represented in red, alongside the sensor signal depicted in blue. The computation of optical flow direction within the patch region is performed to extract the magnified signal from the video (depicted in blue,). (a) Input, (b) Our base model, (c) Our lightweight model, (d) STBVMM [47], (e) MDVMM [96], (f) Anisotropy method [32], (g) Jerk-Aware method [31], and (h) LWVMM [99] respectively.

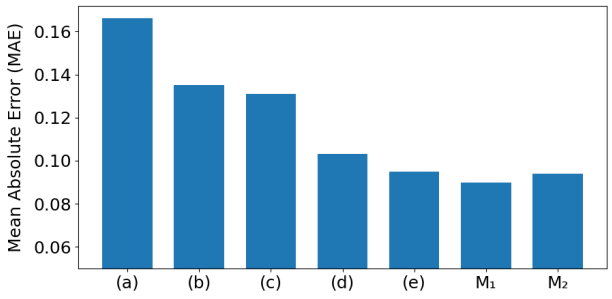


Figure 7.8: Mean Absolute Error (MAE) on SOTA methods of Our base model ($M_1$), Our lightweight model ($M_2$), (d) STBVMM [47], (e) MDVMM [96], (b) Anisotropy method [32], (a) Jerk-Aware method [31], and (h) LWVMM [99]. The MAE is calculated between the signal measured by the sensor and the extracted signal from the magnified video.

magnified video. The average motion direction within the patch was computed. Both optical flow and sensor signals were rescaled from 0 to 1 and mean absolute error (MAE) values are calculated across SOTA methods. Distortion produced by other methods (ringing artifacts, flickering motion *etc*) leads to more error while measuring as compared to ours (illustrated in Figure 7.8).

### 7.2.2 Synthetic Videos

To evaluate the method, 25 synthetic videos with diverse backgrounds are created. These videos feature circles with a 40-pixel radius, simulating subtle movements. Single-direction movements are used for simplicity, with subtle motions. Further, Gaussian noise is added to replicate photographic noise (Please see section 3.2.3 for more details). Varied

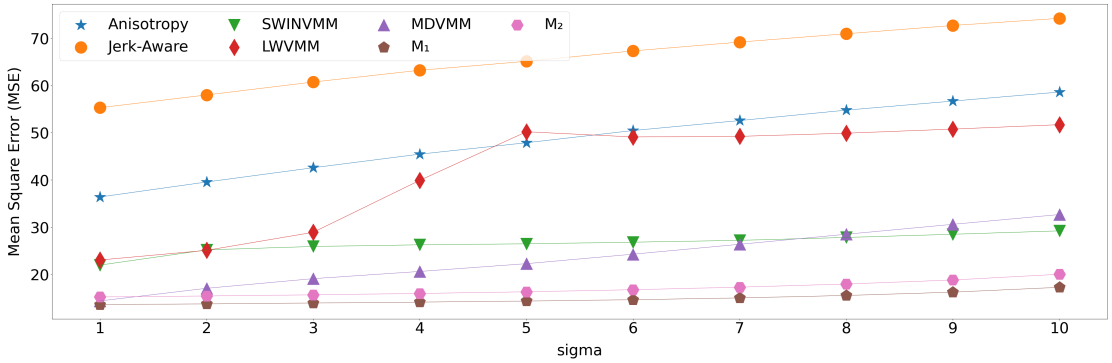


Figure 7.9: Effects of increase in sigma (standard deviation of zero mean gaussian distribution) with output MSE is shown on Anisotropy method [32], Jerk-aware method [31], STBVMM [47], MDVMM [96], LWVMM [99][99], Our base model $M_1$ and Our lightweight model $M_2$. The MSE is computed as an average across 25 different synthetic videos.

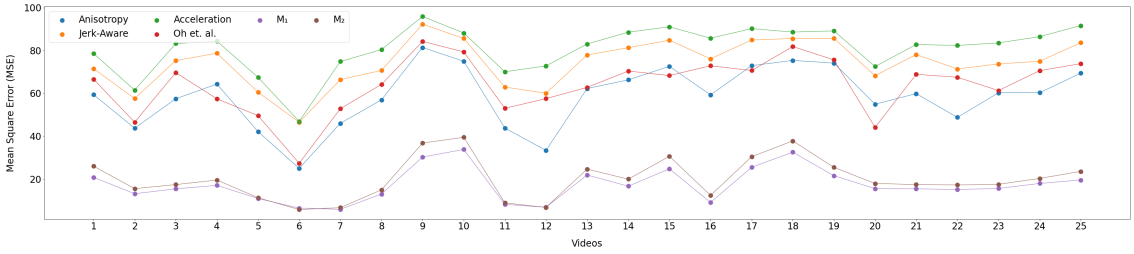


Figure 7.10: Mean Square Error (MSE) of Anisotropy method [32], Jerk-aware method [31], Acceleration method [30], Oh *et al.* method [29], Ours Base model $M_1$ and Our Lightweight model $M_2$ on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

magnification factors are applied to different methods, including [31], [32], MDVMM [96], LWVMM [99], and STBVMM [47], ensuring consistent motion generation akin to the ground truth. This approach facilitates the examination and comparison of how different methods magnify diverse motions in distinct environments, assessing their robustness. Each method is required to achieve 100× magnification, to approximate the ground truth. Figure 7.14 (1) depicts the average Mean Squared Error (MSE) across 25 videos, while Figure 7.9 illustrates the variation of output error with an increase in input noise in the synthetic videos for different methods. From Figure 7.9, it is evident that the proposed method produces the minimum error.

### 7.2.3 Additional Experiments

**Effects of increase in Magnification Factor:** The impact of magnification factor increment is depicted in Figure 7.11, showing the resulting Mean Square Error (MSE) across 25 videos in the presence of noise. Figures 7.12 and 7.13 illustrate visual changes in the proposed base model ($M_1$) and lightweight model ($M_2$) respectively. The base

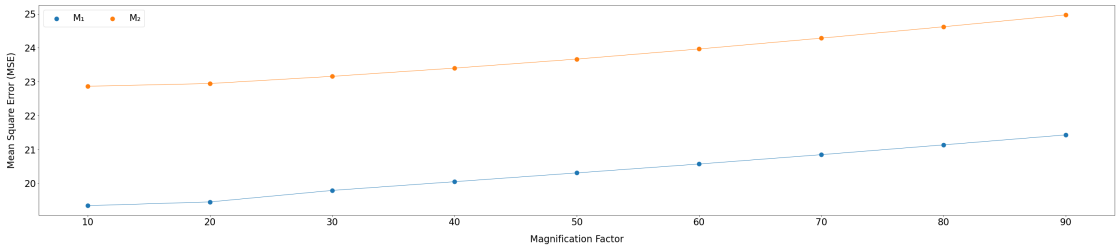


Figure 7.11: Effects of increase in Magnification factor with output MSE (computed average across 25 videos). Input videos contain different backgrounds with noise, without any motion (or circle motion). Results are shown on Our base model $M_1$, and Our lightweight model $M_2$

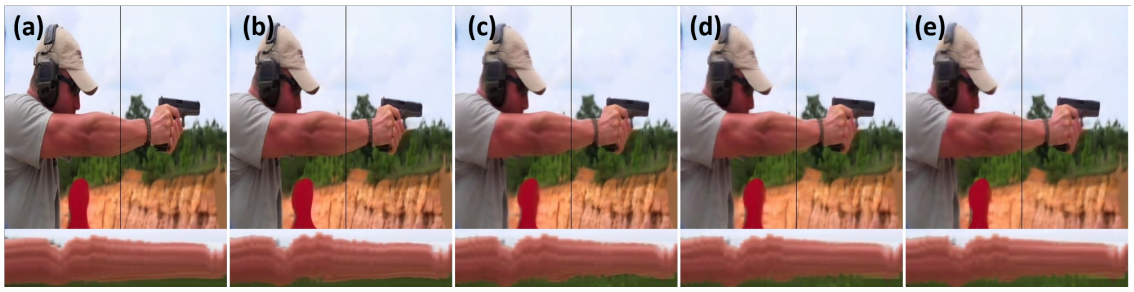


Figure 7.12: Effects of change in Magnification Factor: Figure illustrates proposed base model output ($M_1$). Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column.

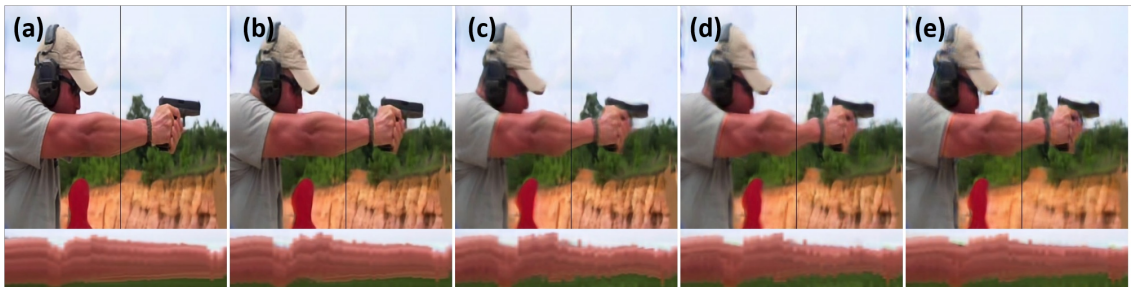


Figure 7.13: Effects of change in Magnification Factor: Figure illustrates our lightweight model output ($M_2$). Different values of the magnification factor in increasing order from (a) 10, (b) 20, (c) 50 (d) 100, and (e) 200 are used to generate the output shown in the respective column.

model exhibits fewer distortions with increasing magnification in dynamic scenarios, while the lightweight model shows decent magnification capabilities but suffers performance degradation with higher magnification factors due to its reduced parameterization. These visualizations offer insights into model performance under varied magnification factors, highlighting their robustness and limitations.

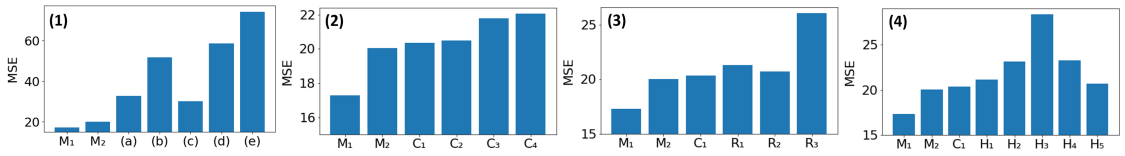


Figure 7.14: Average Mean Square Error (MSE) on 25 synthetically developed videos with various backgrounds, containing subtle motion of circles. **Quantitative Analysis: (1)** Our base model $M_1$, Our lightweight model $M_2$, (a) MDVMM [96], (b) LWVMM [99], (c) STBVMM [47], (d) Anisotropy method [32], and (e) Jerk-Aware method [31]. **Ablation Study: (2)** $M_1$, $M_2$, $C_{(1to4)}$, **(3)** $M_1$, $M_2$, $C_1, R_{(1to3)}$, and **(4)** $M_1$, $M_2$, $C_1, H_{(1to5)}$, where $R_{(1to3)}, C_{(1to4)}$ and $H_{(1to5)}$ are defined in Section 7.2.4 respectively.

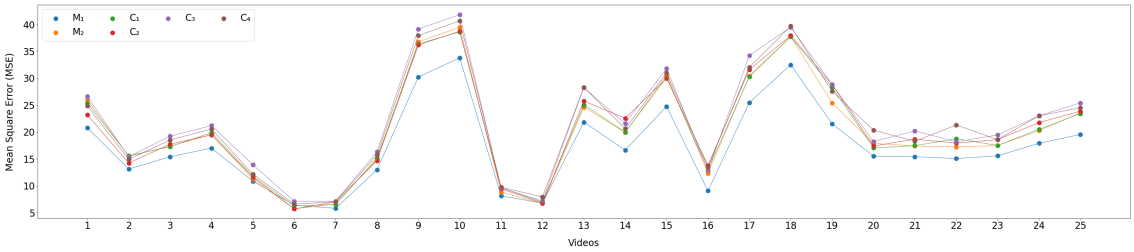


Figure 7.15: Mean Square Error (MSE) of Ours Base model $M_1$, Ours Lightweight model $M_2$ and $C_{(1to4)}$ on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

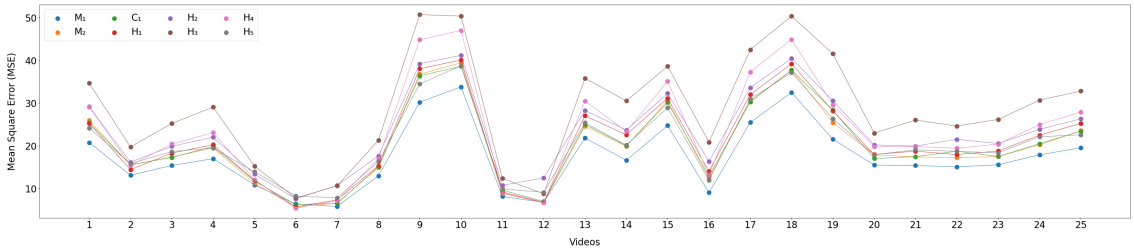


Figure 7.16: Mean Square Error (MSE) of Ours Base model $M_1$, Ours Lightweight model $M_2$ and $C_1, H_{(1to5)}$, (as defined in Section 4.4 of main manuscript respectively) on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

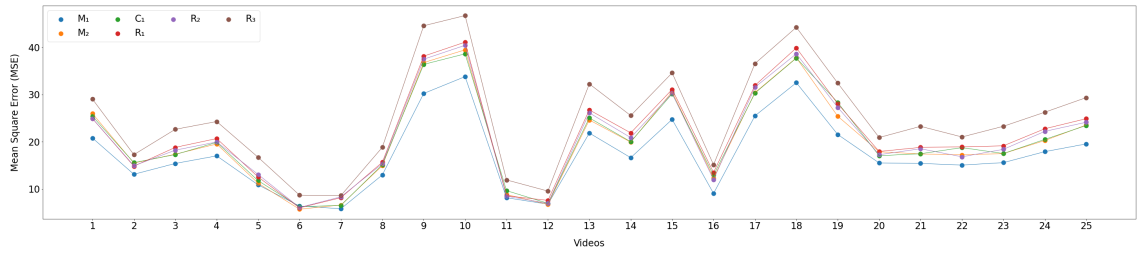


Figure 7.17: Mean Square Error (MSE) of Ours Base model $M_1$, Ours Lightweight model $M_2$ and $C_1, R_{(1to3)}$ (as defined in Section 4.4 of main manuscript respectively) on 25 synthetically generated videos containing different subtle motion of circles with various backgrounds.

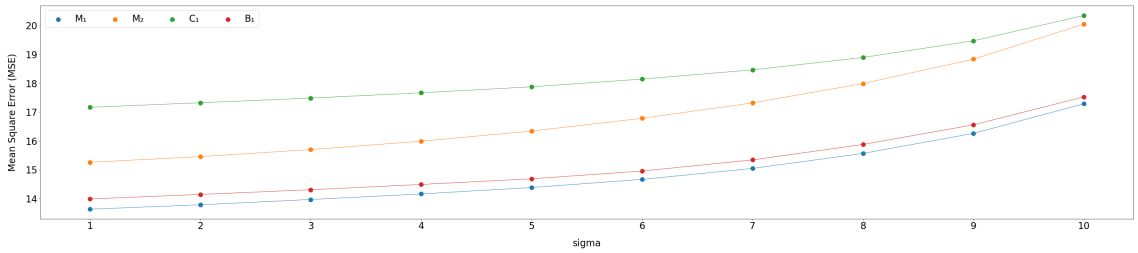


Figure 7.18: Effects of increase in $\sigma$ with output MSE (computed average across 25 different synthetic videos) is shown on Ours Base model fine tunned with a contrastive loss $M_1$, without fine tunned $B_1$ and Ours Lightweight model fine tunned with a contrastive loss $M_2$, without fine tunned $C_1$. With contrastive loss, there is an improvement in output quality, especially in lower values of noise.

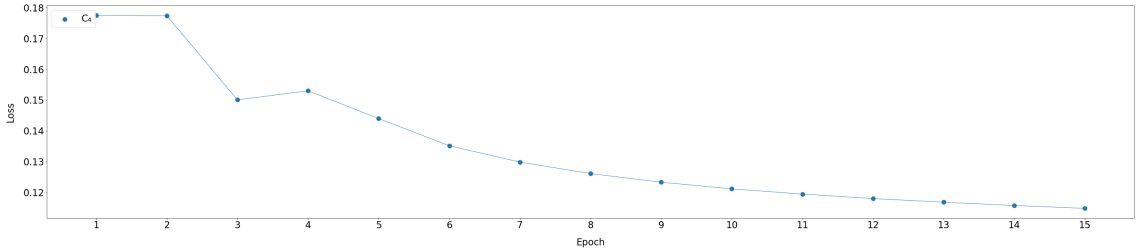


Figure 7.19: Loss curve of proxy model loss with number of epochs (training on $C_4$).

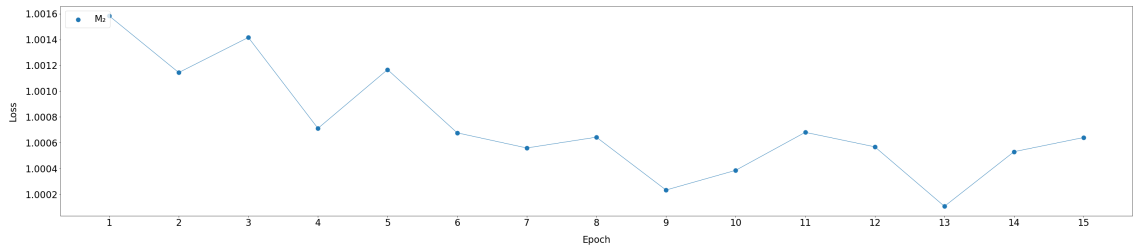


Figure 7.20: Loss curve of proposed contrastive loss with number of epochs (training on proposed lightweight network $M_2$).

### 7.2.4 Ablation Study

**How does the proposed contrastive loss affect the model?** To examine that while keeping the overall architecture first, the $C_1$ model is trained without fine-tuning the proposed contrastive loss. Also, the $C_2$ model is trained with the proposed contrastive loss right from the start. Moreover, instead of contrastive loss, $L_1$ loss with the proxy model features is used to train the $C_3$ model from scratch and fine-tune the $C_4$ model, respectively. This will show the effects of the proxy model. The individual and aggregated results over synthetic videos are shown in Figure 7.15 and 7.14 (2). Additionally, the effect of contrastive loss in noise robustness is illustrated in Figure 7.14. Further, Figure 7.19 and 7.20 shows the training data loss curve for $C_4$ and the proposed $M_2$ model.

**Effect of MSM Block:** To see the effect of different receptive fields while keeping the

same overall architecture, model $R_1$ is trained with dilation rate 1 at all levels. Further, to examine the aspect of serial processing of features in $MSM$, model $R_2$ is trained with parallel layers for $m, r \in \{(1,9),(2,5),(3,1)\}$. Also, a network $R_3$ is trained with residual blocks instead of the $MSM$ block to analyze the overall effect of the $MSM$ block. All these models are evaluated based and the individual and the aggregated $MSE$ over 25 synthetic videos, as shown in Figure 7.16 and 7.14 (3).

**Effects of Hierarchical Magnification and Other Network choices:** To verify this while keeping the same overall architecture, the $H_1$ model is trained without $MSM$ block upsampled differences features at various levels. Also, $H_2$ and $H_3$ models are trained with $m \in (1,2)$ and $m = 1$ levels, respectively, to show how magnified output generated from different levels helps. Further $MD$ blocks are trained without concatenation, with only $frame_t$ as an input in the $H_4$ model. Additionally, to see the effects of edge differences, the $H_5$ model is trained by taking differences directly from the input image. Figure 7.14 (4) and Figure 7.17 illustrate the average and individual values over synthetic videos.

## 7.3 Summary

In this chapter, we discussed a Multi-Scale Manipulator ($MSM$) block with edge features alongside a novel contrastive loss for reducing noise effects. Also, a hierarchical magnification network with a multi-resolution decoder ($MD$) is used for better texture generation. In the $MSM$ block, edge differences are preferred due to their robustness to texture variations as compared to frame differences. Furthermore, multi-receptive field convolution features in the $MSM$ block are scaled and fused with other resolutions of $MSM$ features to mitigate the impact of noise prior to magnification. $MD$ is used to generate magnified features from the fused $MSM$ block features. The magnified features are generated hierarchically, such that lower-level magnified features are combined with the next higher-level features. Additionally, contrastive loss is proposed to fine-tune the model to further improve its robustness. Within the proposed framework, both base and lightweight models are introduced, and their performance is evaluated through qualitative and quantitative analyses alongside state-of-the-art methods applied to both synthetic and real-world videos. An in-depth ablation study is conducted to scrutinize the proposed modules of the framework. The outcomes reveal that both the base and lightweight models outperform state-of-the-art methods, showcasing superior motion magnification with fewer artifacts. However, the proposed network and existing state-of-the-art methods have certain limitations, such as color magnification and motion selection based on pixel displacement, *etc.* that can be addressed in future research.

# Chapter 8

# Conclusion and Future Scope

## 8.1 Conclusion

Motion magnification, a technique that enhances subtle movements in videos, has garnered significant attention due to its wide-ranging applications across various fields. By amplifying imperceptible motions captured in video footage, motion magnification algorithms offer valuable insights and enable more accurate analysis of real-world phenomena.

The main aim of this thesis work is to design and develop novel approaches for video motion magnification. The major challenges like ringing artifacts, mitigating noise amplification, managing computational complexity, achieving adequate magnification in dynamic scenarios, and minimizing distortions introduced by deep learning-based methods need to be tackled for video motion magnification. This work mainly focuses on analyzing and designing different solutions for video motion magnification in the context of providing the solution to the above-mentioned challenges.

For video motion magnification, the deep learning-based method extracts motion information from shape information to enhance the network's robustness to intensity changes. However, its approach to separating shape information from texture may not always be efficient, leading to distorted intermediate features that can result in unwanted flickering or spurious motion. Texture features extracted by this method may sometimes deviate significantly from the input textures, potentially causing blurry distortions in certain frames of the magnified video. To overcome this, a deep learning-based model for video motion magnification is presented. It incorporates proxy model-based feature loss, feature-sharing encoders, and appearance encoder-based regularization terms to mitigate noise and illumination effects while refining motion features. Additionally, a lightweight model is proposed to balance accuracy and computational complexity. Qualitative and quantitative evaluations against state-of-the-art methods demonstrate superior performance.

However, different applications of video motion magnification require differing levels of computational complexity. Therefore, there is a need for a versatile approach that can accommodate application-specific computational models. This work introduces a novel Knowledge Distillation-based Latency-aware Differential Architecture Search (KL-DNAS)

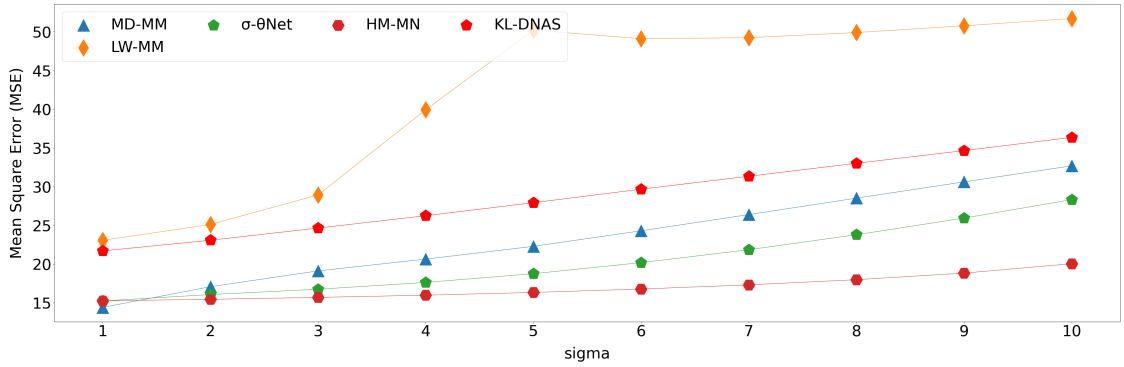


Figure 8.1: The effects of an increase in noise on output MSE are shown on the known proxy model-based training method ($LW$-$MM$), Knowledge Distillation-based Latency-aware Differential Architecture Search ($KL$-$DNAS$), Multi-Domain-based magnification ($MD$-$MM$), $\sigma$-$\theta$Net, and the hierarchical magnification network ($HM$-$MN$). Results are computed across 25 synthetic videos.

method for video motion magnification. This approach optimizes student models under teacher network supervision, leveraging differential architecture search and latency-aware mechanisms. Extensive evaluations reveal improved motion magnification quality with fewer distortions and artifacts compared to existing methods.

But, there are still areas for further improvement. One potential avenue involves leveraging insights from hand-designed methods that utilize complex steerable pyramids to address noise effects in motion magnification. Integrating these techniques with deep learning-based approaches could further enhance their efficiency and effectiveness. Additionally, we propose a multi-domain network that combines frequency and spatial domain operations. Furthermore, we introduce lightweight models that yield comparable results to state-of-the-art methods.

Whereas, the proposed work doesn't fully utilize the steerable pyramid architecture of hand-crafted-based methods, as it manipulates motion features at a single scale. We introduce $\sigma$-$\theta$Net, a novel network for video motion magnification. Leveraging learnable directional scale-space filters, it produces high-quality magnified output while reducing noise impact. Our method outperforms state-of-the-art approaches both qualitatively and quantitatively with reduced parameters, but further research is needed for robustness in noisy scenarios with fewer parameters.

To solve this, a hierarchical magnification network with Multi-Scale Manipulator ($MSM$) block with edge features and a contrastive loss is proposed to reduce noise effects in motion magnification. A hierarchical magnification network with a multi-resolution decoder ($MD$) enhances texture generation by hierarchically combining magnified features. Additionally, contrastive loss is proposed to fine-tune the model to further improve its robustness. Within the proposed framework, both base and lightweight models are introduced.

The quantitative comparison of the proposed approaches for video motion magnification is

provided in Figure 8.1. For MAE computation, the proposed synthetic dataset is used. The proposed hierarchical magnification network $HM\text{-}MN$ achieves state-of-the-art results, whereas the $\sigma$-$\theta$Net depicts the best results with the least parameters.

## 8.2 Future Scope

The aim of this thesis is to address challenges in video motion magnification, including noise amplification, computational complexity, and distortions. Novel approaches are developed to mitigate these challenges, including a deep learning-based model with feature refinement and a lightweight variant. Furthermore, a Knowledge Distillation-based method is introduced for optimizing computational models, and insights from hand-designed methods are leveraged for noise reduction. A multi-domain network and a hierarchical magnification network with Multi-Scale Manipulator block are proposed to enhance robustness and texture generation. The goal is to contribute solutions to these challenges through comprehensive quantitative and qualitative evaluations.

In the future, we aim to integrate additional hand-designed methods, like complex steerable pyramids, to enhance noise reduction and improve efficiency in motion magnification algorithms. Moreover, exploring advanced techniques for robustness to noise and dynamic scenes, such as incorporating motion estimation algorithms, could lead to more accurate and reliable magnification results. Additionally, investigating transformer-based approaches to enhance output quality while reducing computational complexity seems promising. Overall, future research endeavors should focus on refining and expanding the capabilities of video motion magnification techniques to better meet the evolving needs of various domains and applications.

# List of the Publications

## International Journals

1. Jasdeep Singh, Subrahmanyam Murala, and G. Sankara Raju Kosuru, "KL-DNAS: Knowledge Distillation Based Latency Aware - Differentiable Architecture Search for Video Motion Magnification " ***IEEE Transactions on Neural Networks and Learning Systems, pp. 1-11, doi: 10.1109/TNNLS.2023.3346169. (2024)*** (**Impact factor-10.4**).

2. Jasdeep Singh, Subrahmanyam Murala, G. Sankara Raju Kosuru, and Santosh Kumar Vipparthi,"HMN: Hierarchical Magnification Network for Video Motion Magnification ", ***IEEE Transactions on Multimedia*** **(Under review)**.

3. Jasdeep Singh, Subrahmanyam Murala, G. Sankara Raju Kosuru, and Santosh Kumar Vipparthi, " Learnable Directional Scale Space Filters for Video Motion Magnification ", ***IEEE Transactions on Image Processing*** **(Under review)**.

## International Conferences

1. Jasdeep Singh, Subrahmanyam Murala, and G. Sankara Raju Kosuru, "Multi domain learning for motion magnification", In Proceedings of the ***IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 13914–13923, June 2023.*** (**h-index- 422**).

2. Jasdeep Singh, Subrahmanyam Murala, and G. Sankara Raju Kosuru, "Lightweight Network for Video Motion Magnification ", In Proceedings of the ***IEEE/CVF WACV, Waikoloa, HI, USA, 2023, pp. 2040-2049.*** (**h-index- 95**).

# References


[1] Ryan Zachar, Peter Lindahl, John Donnal, William Cotta, Christopher Schantz, and Steven B Leeb. Utilizing spin-down transients for vibration-based diagnostics of resiliently mounted machines. *IEEE Transactions on Instrumentation and Measurement*, 65(7): 1641–1650, 2016.

[2] Shiqian Chen, Yang Yang, Xingjian Dong, Guanpei Xing, Zhike Peng, and Wenming Zhang. Warped variational mode decomposition with application to vibration signals of varying-speed rotating machineries. *IEEE Transactions on Instrumentation and Measurement*, 68(8):2755–2767, 2018.

[3] Liuyang Song, Huaqing Wang, and Peng Chen. Vibration-based intelligent fault diagnosis for roller bearings in low-speed rotating machinery. *IEEE Transactions on Instrumentation and Measurement*, 67(8):1887–1899, 2018.

[4] Wei He, Yuncheng Ouyang, and Jie Hong. Vibration control of a flexible robotic manipulator in the presence of input deadzone. *IEEE Transactions on Industrial Informatics*, 13(1):48–59, 2016.

[5] Kai Liu, Edwin Reynders, Guido De Roeck, and Geert Lombaert. Experimental and numerical analysis of a composite bridge for high-speed trains. *Journal of sound and vibration*, 320(1-2):201–220, 2009.

[6] AG Poulimenos and SD Fassois. Parametric time-domain methods for non-stationary random vibration modelling and analysis—a critical survey and comparison. *Mechanical systems and signal processing*, 20(4):763–816, 2006.

[7] Dongxu Li, Jianping Jiang, Wang Liu, and Caizhi Fan. A new mechanism for the vibration control of large flexible space structures with embedded smart devices. *IEEE/ASME Transactions on Mechatronics*, 20(4):1653–1659, 2014.

[8] Abe Davis*, Katherine L. Bouman*, Justin G. Chen, Michael Rubinstein, Oral Büyüköztürk, Frédo Durand, and William T. Freeman. Visual vibrometry: Estimating material properties from small motions in video. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(4):732–745, 2017. doi: 10.1109/TPAMI.2016.2622271.

[9] Ernesto Moya-Albor, Jorge Brieva, Hiram Ponce, and Lourdes Martínez-Villaseñor. A non-contact heart rate estimation method using video magnification and neural networks. *IEEE Instrumentation Measurement Magazine*, 23(4):56–62, 2020. doi: 10.1109/MIM.2020.9126072.

[10] Cong Peng, Cong Zeng, and Yangang Wang. Phase-based noncontact vibration measurement of high-speed magnetically suspended rotor. *IEEE Transactions on Instrumentation and Measurement*, 69(7):4807–4817, 2020. doi: 10.1109/TIM.2019.2956333.

[11] Hyungjun Kim, Youngbeen Chung, Jie Jin, and Junhong Park. Manifestation of flexural vibration modes of rails by the phase-based magnification method. *IEEE Access*, 9: 98121–98131, 2021. doi: 10.1109/ACCESS.2021.3095619.

[12] Justin G Chen, Neal Wadhwa, Young-Jin Cha, Frédo Durand, William T Freeman, and Oral Buyukozturk. Structural modal identification through high speed camera video: Motion magnification. In *Topics in Modal Analysis I, Volume 7*, pages 191–197. Springer, 2014.

[13] Mengjiong Bai, Roland Goecke, and Damith Herath. Micro-expression recognition based on video motion magnification and pre-trained neural network. In *2021 IEEE International Conference on Image Processing (ICIP)*, pages 549–553, 2021. doi: 10.1109/ICIP42928.2021.9506793.

[14] Ling Lei, Jianfeng Li, Tong Chen, and Shigang Li. A novel graph-tcn with a graph structured representation for micro-expression recognition. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2237–2245, 2020.

[15] Ankith Jain Rakesh Kumar, Rajkumar Theagarajan, Omar Peraza, and Bir Bhanu. Classification of facial micro-expressions using motion magnified emotion avatar images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2019.

[16] Yandan Wang, John See, Yee-Hui Oh, Raphael C-W Phan, Yogachandran Rahulamathavan, Huo-Chong Ling, Su-Wei Tan, and Xujie Li. Effective recognition of facial micro-expressions with video motion magnification. *Multimedia Tools and Applications*, 76(20):21665–21690, 2017.

[17] Sungsoo Park and Daijin Kim. Subtle facial expression recognition using motion magnification. *Pattern Recognition Letters*, 30(7):708–716, 2009.

[18] Wenkang Fan, Zhuohui Zheng, Wankang Zeng, Yinran Chen, Hui-Qing Zeng, Hong Shi, and Xiongbiao Luo. Robotically surgical vessel localization using robust hybrid video motion magnification. *IEEE Robotics and Automation Letters*, 6(2):1567–1573, 2021. doi: 10.1109/LRA.2021.3058906.

[19] Abe Davis, Michael Rubinstein, Neal Wadhwa, Gautham Mysore, Fredo Durand, and William T. Freeman. The visual microphone: Passive recovery of sound from video. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 33(4):79:1–79:10, 2014.

[20] Michael Dorkenwald, Uta Buchler, and Bjorn Ommer. Unsupervised magnification of posture deviations across subjects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[21] Mateusz P. Popek and D. Robert Iskander. A new approach to the phase-based video motion magnification for measuring microdisplacements. *IEEE Transactions on Instrumentation and Measurement*, 69(2):354–361, 2020. doi: 10.1109/TIM.2019.2904074.

[22] Mateusz P. Popek, Monika E. Danielewska, and D. Robert Iskander. Assessing the feasibility of the use of video motion magnification for measuring microdisplacements.

*IEEE Transactions on Instrumentation and Measurement*, 66(9):2329–2336, 2017. doi: 10.1109/TIM.2017.2700118.

[23] Daniel G. Kyrollos, Joshua B. Tanner, Kim Greenwood, JoAnn Harrold, and James R. Green. Noncontact neonatal respiration rate estimation using machine vision. In *2021 IEEE Sensors Applications Symposium (SAS)*, pages 1–6, 2021. doi: 10.1109/SAS51076.2021.9530013.

[24] Chuangxue Zhu, Wenbin Zhao, Wu Lu, Yuan Gao, Feng Li, and Min Tang. Partial discharge detection, location and continuous monitoring in power cable by using eulerian video magnification. In *2020 IEEE International Conference on High Voltage Engineering and Application (ICHVE)*, pages 1–4, 2020. doi: 10.1109/ICHVE49031.2020.9279445.

[25] Zhen Liu, Qingbo He, Shiqian Chen, Zhike Peng, and Wenming Zhang. Time-varying motion filtering for vision-based nonstationary vibration measurement. *IEEE Transactions on Instrumentation and Measurement*, 69(6):3907–3916, 2019.

[26] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T Freeman. Phase-based video motion processing. *ACM Transactions on Graphics (TOG)*, 32(4):1–10, 2013.

[27] Hao-Yu Wu, Michael Rubinstein, Eugene Shih, John Guttag, Frédo Durand, and William Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM transactions on graphics (TOG)*, 31(4):1–8, 2012.

[28] Neal Wadhwa, Michael Rubinstein, Frédo Durand, and William T Freeman. Riesz pyramids for fast phase-based video magnification. In *2014 IEEE International Conference on Computational Photography (ICCP)*, pages 1–10. IEEE, 2014.

[29] Tae-Hyun Oh, Ronnachai Jaroensri, Changil Kim, Mohamed Elgharib, Fr'edo Durand, William T Freeman, and Wojciech Matusik. Learning-based video motion magnification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 633–648, 2018.

[30] Yichao Zhang, Silvia L Pintea, and Jan C Van Gemert. Video acceleration magnification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 529–537, 2017.

[31] Shoichiro Takeda, Kazuki Okami, Dan Mikami, Megumi Isogai, and Hideaki Kimata. Jerk-aware video acceleration magnification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1769–1777, 2018.

[32] Shoichiro Takeda, Yasunori Akagi, Kazuki Okami, Megumi Isogai, and Hideaki Kimata. Video magnification in the wild using fractional anisotropy in temporal distribution. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1614–1622, 2019.

[33] Ce Liu, Antonio Torralba, William T Freeman, Frédo Durand, and Edward H Adelson. Motion magnification. *ACM transactions on graphics (TOG)*, 24(3):519–526, 2005.

[34] Weixuan Chen and Daniel McDuff. Deepmag: Source-specific change magnification using gradient ascent. *ACM Trans. Graph.*, 40(1), September 2020. ISSN 0730-0301. doi: 10.1145/3408865. URL https://doi.org/10.1145/3408865.

[35] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8934–8943, 2018.

[36] Zhaoying Pan, Daniel Geng, and Andrew Owens. Self-supervised motion magnification by backpropagating through optical flow. *arXiv preprint arXiv:2311.17056*, 2023.

[37] Mohamed Elgharib, Mohamed Hefeeda, Fredo Durand, and Bill Freeman. Video magnification in presence of large motions. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4119–4127, 2015.

[38] Manisha Verma and Shanmuganathan Raman. Interest region based motion magnification. In *International Conference on Image Analysis and Processing*, pages 27–39. Springer, 2017.

[39] Michele A. Saad, Alan C. Bovik, and Christophe Charrier. Blind prediction of natural video quality. *IEEE Transactions on Image Processing*, 23(3):1352–1365, 2014. doi: 10.1109/TIP.2014.2299154.

[40] Manisha Verma and Shanmuganathan Raman. Edge-aware spatial filtering-based motion magnification. In *Proceedings of 2nd International Conference on Computer Vision & Image Processing*, pages 117–128. Springer, 2018.

[41] Xiu Wu, Xuezhi Yang, Jing Jin, and Zhao Yang. Pca-based magnification method for revealing small signals in video. *Signal, Image and Video Processing*, 12(7):1293–1299, 2018.

[42] Shoichiro Takeda, Kenta Niwa, Mariko Isogawa, Shinya Shimizu, Kazuki Okami, and Yushi Aono. Bilateral video magnification filter. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17348–17357, 2022. doi: 10.1109/CVPR52688.2022.01685.

[43] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010.

[44] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[45] Sicheng Gao, Yutang Feng, Linlin Yang, Xuhui Liu, Zichen Zhu, David Doermann, and Baochang Zhang. Magformer: Hybrid video motion magnification transformer from eulerian and lagrangian perspectives. 2022.

[46] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8121–8130, June 2022.

[47] Ricard Lado-Roigé and Marco A. Pérez. Stb-vmm: Swin transformer based video motion magnification. *Knowledge-Based Systems*, 269:110493, June 2023. ISSN 0950-7051. doi: 10.1016/j.knosys.2023.110493. URL http://dx.doi.org/10.1016/j.knosys.2023.110493.

[48] Brandon Y. Feng, Hadi AlZayer, Michael Rubinstein, William T. Freeman, and Jia-Bin Huang. Visualizing subtle motions with time-varying neural fields. In *International Conference on Computer Vision (ICCV)*, 2023.

[49] Simone Meyer, Abdelaziz Djelouah, Brian McWilliams, Alexander Sorkine-Hornung, Markus Gross, and Christopher Schroers. Phasenet for video frame interpolation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[50] Zongsheng Yue, Jianwen Xie, Qian Zhao, and Deyu Meng. Semi-supervised video deraining with dynamical rain generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 642–652, 2021.

[51] Xiaobin Hu, Wenqi Ren, Kaicheng Yu, Kaihao Zhang, Xiaochun Cao, Wei Liu, and Bjoern Menze. Pyramid architecture search for real-time image deblurring. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4298–4307, 2021.

[52] Rachel Huang, Jonathan Pedoeem, and Cuixian Chen. Yolo-lite: a real-time object detection algorithm optimized for non-gpu computers. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 2503–2510. IEEE, 2018.

[53] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[54] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, et al. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2(7), 2015.

[55] Tambet Matiisen, Avital Oliver, Taco Cohen, and John Schulman. Teacher–student curriculum learning. *IEEE Transactions on Neural Networks and Learning Systems*, 31 (9):3732–3740, 2020. doi: 10.1109/TNNLS.2019.2934906.

[56] Guanzhe Hong, Zhiyuan Mao, Xiaojun Lin, and Stanley H Chan. Student-teacher learning from clean inputs to noisy inputs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12075–12084, 2021.

[57] Zhou Wang and Eero P Simoncelli. An adaptive linear system framework for image distortion analysis. In *IEEE International Conference on Image Processing 2005*, volume 3, pages III–1160. IEEE, 2005.

[58] Miika Aittala and Frédo Durand. Burst image deblurring using permutation invariant convolutional neural networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 731–747, 2018.

[59] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *European conference on computer vision*, pages 694–711. Springer, 2016.

[60] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[61] Henrik Lauridsen, Selina Gonzales, Daniela Hedwig, Kathryn L Perrin, Catherine JA Williams, Peter H Wrege, Mads F Bertelsen, Michael Pedersen, and Jonathan T Butcher. Extracting physiological information in experimental biology via eulerian video magnification. *BMC biology*, 17(1):1–26, 2019.

[62] Carlos I. Cardona, Hector A. Tinoco, Daniel A. Pereira, Jaime Buitrago-Osorio, Luis Perdomo-Hurtado, Mateo Hurtado-Hernandez, and Juliana Lopez-Guzman. Vibration shapes identification applying eulerian video magnification on coffee fruits to study the selective harvesting. In *2020 19th International Conference on Mechatronics - Mechatronika (ME)*, pages 1–8, 2020. doi: 10.1109/ME49197.2020.9286641.

[63] Vincent Perrot, Sébastien Salles, Didier Vray, and Hervé Liebgott. Video magnification applied in ultrasound. *IEEE Transactions on Biomedical Engineering*, 66(1):283–288, 2019. doi: 10.1109/TBME.2018.2820384.

[64] Xueyi Zhang, Changchong Sheng, and Li Liu. Lip motion magnification network for lip reading. In *2021 7th International Conference on Big Data and Information Analytics (BigDIA)*, pages 274–279, 2021. doi: 10.1109/BigDIA53151.2021.9619626.

[65] Aman Mehra, Akshay Agarwal, Mayank Vatsa, and Richa Singh. Motion magnified 3-d residual-in-dense network for deepfake detection. *IEEE Transactions on Biometrics, Behavior, and Identity Science*, 5(1):39–52, 2023. doi: 10.1109/TBIOM.2022.3201887.

[66] Han Cai, Chuang Gan, Tianzhe Wang, Zhekai Zhang, and Song Han. Once-for-all: Train one network and specialize it for efficient deployment. *arXiv preprint arXiv:1908.09791*, 2019.

[67] Li Lyna Zhang, Yuqing Yang, Yuhang Jiang, Wenwu Zhu, and Yunxin Liu. Fast hardware-aware neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 692–693, 2020.

[68] Linnan Wang, Chenhan Yu, Satish Salian, Slawomir Kierat, Szymon Migacz, and Alex Fit Florea. Searching the deployable convolution neural networks for gpus. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12217–12226, 2022. doi: 10.1109/CVPR52688.2022.01191.

[69] Xiangzhong Luo, Di Liu, Hao Kong, Shuo Huai, Hui Chen, and Weichen Liu. Lightnas: On lightweight and scalable neural architecture search for embedded platforms. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 42(6): 1784–1797, 2023. doi: 10.1109/TCAD.2022.3208187.

[70] Jaeseong Lee, Jungsub Rhim, Duseok Kang, and Soonhoi Ha. Snas: Fast hardware-aware neural architecture search methodology. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 41(11):4826–4836, 2022. doi: 10.1109/TCAD.2021.3134843.

[71] Lukasz Dudziak, Thomas Chau, Mohamed Abdelfattah, Royson Lee, Hyeji Kim, and Nicholas Lane. Brp-nas: Prediction-based nas using gcns. *Advances in Neural Information Processing Systems*, 33:10480–10490, 2020.

[72] Haokui Zhang, Ying Li, Hao Chen, and Chunhua Shen. Memory-efficient hierarchical neural architecture search for image denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3657–3666, 2020.

[73] Ruoteng Li, Robby T Tan, and Loong-Fah Cheong. All in one bad weather removal using architectural search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3175–3185, 2020.

[74] Metin Ersin Arican, Ozgur Kara, Gustav Bredell, and Ender Konukoglu. Isnas-dip: Image-specific neural architecture search for deep image prior. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1950–1958, 2022. doi: 10.1109/CVPR52688.2022.00200.

[75] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[76] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

[77] Zhao Zhong, Junjie Yan, Wei Wu, Jing Shao, and Cheng-Lin Liu. Practical block-wise neural network architecture generation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2423–2432, 2018. doi: 10.1109/CVPR.2018.00257.

[78] Hieu Pham, Melody Guan, Barret Zoph, Quoc Le, and Jeff Dean. Efficient neural architecture search via parameters sharing. In *International Conference on Machine Learning*, pages 4095–4104. PMLR, 2018.

[79] Yuqiao Liu, Yanan Sun, Bing Xue, Mengjie Zhang, Gary G. Yen, and Kay Chen Tan. A survey on evolutionary neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems*, 34(2):550–570, 2023. doi: 10.1109/TNNLS.2021.3100554.

[80] Risto Miikkulainen, Jason Liang, Elliot Meyerson, Aditya Rawal, Daniel Fink, Olivier Francon, Bala Raju, Hormoz Shahrzad, Arshak Navruzyan, Nigel Duffy, et al. Evolving deep neural networks. In *Artificial intelligence in the age of neural networks and brain computing*, pages 293–312. Elsevier, 2019.

[81] Lingxi Xie and Alan Yuille. Genetic cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1379–1388, 2017.

[82] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. Understanding and simplifying one-shot architecture search. In *International Conference on Machine Learning*, pages 550–559. PMLR, 2018.

[83] Andrew Brock, Theodore Lim, James M Ritchie, and Nick Weston. Smash: one-shot model architecture search through hypernetworks. *arXiv preprint arXiv:1708.05344*, 2017.

[84] Tao Huang, Shan You, Yibo Yang, Zhuozhuo Tu, Fei Wang, Chen Qian, and Changshui Zhang. Explicitly learning topology for differentiable neural architecture search. *arXiv preprint arXiv:2011.09300*, 2020.

[85] Liam Li and Ameet Talwalkar. Random search and reproducibility for neural architecture search. In *Uncertainty in artificial intelligence*, pages 367–377. PMLR, 2020.

[86] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[87] Monu Verma, M. Satish Kumar Reddy, Yashwanth Reddy Meedimale, Murari Mandal, and Santosh Kumar Vipparthi. Automer: Spatiotemporal neural architecture search for microexpression recognition. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11):6116–6128, 2022. doi: 10.1109/TNNLS.2021.3072290.

[88] Qinqin Zhou, Bineng Zhong, Xin Liu, and Rongrong Ji. Attention-based neural architecture search for person re-identification. *IEEE Transactions on Neural Networks and Learning Systems*, 33(11):6627–6639, 2022. doi: 10.1109/TNNLS.2021.3082701.

[89] Yijun Bian, Qingquan Song, Mengnan Du, Jun Yao, Huanhuan Chen, and Xia Hu. Subarchitecture ensemble pruning in neural architecture search. *IEEE Transactions on Neural Networks and Learning Systems*, 33(12):7928–7936, 2022. doi: 10.1109/TNNLS.2021.3085299.

[90] Junwei Dong, Boyu Hou, Liang Feng, Huajin Tang, Kay Chen Tan, and Yew-Soon Ong. A cell-based fast memetic algorithm for automated convolutional neural architecture design. *IEEE Transactions on Neural Networks and Learning Systems*, 2022.

[91] Han Cai, Ligeng Zhu, and Song Han. Proxylessnas: Direct neural architecture search on target task and hardware. *arXiv preprint arXiv:1812.00332*, 2018.

[92] Bichen Wu, Xiaoliang Dai, Peizhao Zhang, Yanghan Wang, Fei Sun, Yiming Wu, Yuandong Tian, Peter Vajda, Yangqing Jia, and Kurt Keutzer. Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[93] Hui Wang, Hanbin Zhao, Xi Li, and Xu Tan. Progressive blockwise knowledge distillation for neural network acceleration. In *IJCAI*, pages 2769–2775, 2018.

[94] Changlin Li, Jiefeng Peng, Liuchun Yuan, Guangrun Wang, Xiaodan Liang, Liang Lin, and Xiaojun Chang. Block-wisely supervised neural architecture search with knowledge distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1989–1998, 2020.

[95] Jasdeep Singh, Subrahmanyam Murala, and G Kosuru. Lightweight network for video motion magnification. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 2041–2050, 2023.

[96] Jasdeep Singh, Subrahmanyam Murala, and G. Sankara Raju Kosuru. Multi domain learning for motion magnification. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13914–13923, June 2023.

[97] William T Freeman, Edward H Adelson, et al. The design and use of steerable filters. *IEEE Transactions on Pattern analysis and machine intelligence*, 13(9):891–906, 1991.

[98] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7132–7141, 2018. doi: 10.1109/CVPR.2018.00745.

[99] Jasdeep Singh, Subrahmanyam Murala, and G. Sankara Raju Kosuru. Lightweight network for video motion magnification. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2041–2050, January 2023.

[100] Alexey Dosovitskiy, Philipp Fischer, Jost Tobias Springenberg, Martin Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with exemplar convolutional neural networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(9): 1734–1747, 2016. doi: 10.1109/TPAMI.2015.2496141.

[101] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[102] Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[103] R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. *arXiv preprint arXiv:1808.06670*, 2018.

[104] Taesung Park, Alexei A. Efros, Richard Zhang, and Jun-Yan Zhu. Contrastive learning for unpaired image-to-image translation. In *European Conference on Computer Vision*, 2020.

# Thesis Plagiarism

## Thesis

ORIGINALITY REPORT

| 9% | 5% | 7% | % |
|---|---|---|---|
| SIMILARITY INDEX | INTERNET SOURCES | PUBLICATIONS | STUDENT PAPERS |

PRIMARY SOURCES

| | | |
|---|---|---|
| 1 | arxiv.org<br>Internet Source | <1% |
| 2 | www.cse.iitk.ac.in<br>Internet Source | <1% |
| 3 | "Computer Vision – ECCV 2016", Springer Nature, 2016<br>Publication | <1% |
| 4 | Hans Fuhrmann, Anthony Boyko, Mohamed H. Abdelpakey, Mohamed S. Shehata. "DETECTren: Vehicle Object Detection Using Self-Supervised Learning based on Light-Weight Network for Low-Power Devices", 2021 IEEE 7th World Forum on Internet of Things (WF-IoT), 2021<br>Publication | <1% |
| 5 | www.iitrpr.ac.in<br>Internet Source | <1% |
| 6 | "Computer Vision – ECCV 2018", Springer Science and Business Media LLC, 2018<br>Publication | <1% |