Glacier velocity estimation using Adaptive Search Window and Patch size

A dissertation submitted in
partial fulfilment for the degree of

Master of Technology

in

Computer Science

by

Pratik Patil
Roll no. - **CS2313**

under the supervision of

Dr. Sarbani Palit
Compute Vision and Pattern Recognition Unit (CVPRU)



Indian Statistical Institute, Kolkata
June, 2024

CERTIFICATE

This is to certify that the dissertation titled "Glacier Velocity Estimation Using Adaptive Search Window and Patch Size" submitted by Pratik Patil to the Indian Statistical Institute, Kolkata, fulfills the requirements for the degree of Master of Technology in Computer Science. The work presented in this dissertation is an authentic and original contribution conducted under my supervision and guidance. I confirm that this dissertation adheres to all the necessary academic standards and regulations of the institute.

Sarbani Palit 11.6.2025

Dr. Sarbani Palit
CVPR Unit
Indian Statistical Institute
Kolkata - 700108
India

Acknowledgement

I would like to express my gratitude to Dr. Sarbani Palit, my advisor at the Computer Vision and Pattern Recognition Unit of the Indian Statistical Institute in Kolkata. Her expert guidance, unwavering support, and inspiring insights have been invaluable throughout my academic journey. Dr. Palit's extensive knowledge and innovative ideas have significantly contributed to my understanding of various topics and have been crucial in shaping my research skills and research aptitude.

I am very thankful to Subhranil Mustafi, Senior Research Fellow at the Indian Statistical Institute, for his essential assistance in acquiring the datasets, notes and necessary materials for this study. His continuous input of ideas and continuous support have played an important role in the successful completion of this project. After various discussion related to topic we came with some improvement in our work.

I also extend my gratitude to all the faculty members at the Indian Statistical Institute for their invaluable advice, insights, and teachings, which have provided a vital perspective to my research work.

I am thankful to all my friends for their ongoing assistance and motivation. My appreciation goes out to everyone who has contributed to my personal and academic growth, even if I have not explicitly mentioned them here.

Declaration

I, **Pratik Patil**, with Roll No. **CS2313**, hereby declare that the material presented in the dissertation titled **Glacier velocity estimation using Adaptive Window and Patch size** represents original work carried out by me for the degree of **Master of Technology in Computer Science** at the **Indian Statistical Institute, Kolkata**.

Furthermore, I affirm that no sections of this report have been sourced or copied from external references without proper attribution. I am aware that any instances of plagiarism or the use of unacknowledged materials from third parties will be treated with the utmost seriousness and consequences.



Pratik Patil
M.Tech (CS), CS2313
Indian Statistical Institute

Abstract

Synthetic Aperture Radar (SAR) technology offers a robust solution for monitoring glacier surface motion, particularly in regions with challenging environmental conditions, since it do not dependence on time of day and weather. This paper presents an enhanced glacier motion monitoring approach based on a Deep Matching Network (DMN), which learns patch-pair correspondences in an end-to-end manner. Unlike traditional shallow feature tracking methods, DMN utilize deep feature similarity through a Siamese network architecture with dense connection blocks to maximize feature reuse and improve training efficiency. To further improve precision and reduce computational cost, the proposed method uses a variable search window and adaptive patch sizing, enabling efficient and accurate motion estimation across diverse glacier terrains. Experimental results demonstrate the effectiveness of the proposed approach in achieving high accuracy and efficiency in glacier motion tracking on SAR data.

Keywords: Deep Matching Network (DMN), Gaussian Pyramid, Digital elevation model (DEM), Template Matching, glacier surface motion, dense connection, CBAM

Contents

Certificate **i**

Acknowledgement **ii**

Abstract **iv**

1 Introduction **1**

1.1 Literature Survey 1

1.2 Our Proposed Work 1

2 Related Theory **2**

2.1 Gaussian Pyramid 2

2.2 Template Matching 3

2.3 GAN-Based SAR Image Reconstruction with CBAM 4

3 Dataset **6**

3.1 Study Area 6

3.2 SAR Imagery 7

3.3 Preparing Dataset for Siamese Model Deep Matching Network (DMN) 7

3.4 Preparing Dataset for CBAM GAN 8

4 Methodology **10**

4.1 Preprocessing SAR intensity images 11

4.2 Deep Learning Models 12

4.2.1 Deep Matching Network for SAR Image Matching 12

4.2.2 Methodology for Simulated SAR Image Generation 13

4.3 Post Processing 14

4.4 Proposed Methodology for Velocity Estimation in SAR Images 15

4.5 Proposed algorithm for pixel level displacement 16

5 Experiments and Results **18**

5.1 Model Selection and Trade-off Analysis 18

5.2 Analysis of Patch Size and Threshold Selection in Template Matching 19

5.3 Root Mean Square Error (RMSE) 20
5.4 Analysis of time 21
5.5 Results 22

6 Conclusion and Future Work **23**

Bibliography **24**

List of Figures

2.1 Gaussian Pyramid of greyscale image of size 64 x 64 3

3.1 Marked box is our study area (Yangong Glacier) 6
3.2 (a) Input Image, (b) Binary Mask (c) Masked Input 9

4.1 Framework for proposed method 10
4.2 Preprocessed SAR intensity images of the Yanong Glacier. (a) was captured on 23 November 2016. (b) was captured on 3 February 2017 11
4.3 Proposed framework of a deep matching network 12
4.4 (a) Original SAR Image (b) Masked Image (c) Binary Mask (d) Simulated SAR image 13
4.5 Template matching at different levels I_0, I_1 and I_2 16

5.1 Sub-Patch Matching for $\text{Threshold}_{64} = 0.8$ with varying Threshold_{32} . 19
5.2 RMSE for different Multiplicative Variance Noise for different image matching methods 20
5.3 Time taken vs Multiplicative Noise Variance for different image matching methods. 21
5.4 Velocity magnitude maps using different methods 22

Chapter 1

Introduction

1.1 Literature Survey

On 7 February 2021, a catastrophic event of mass flow descended the Ronti Gad, Rishiganga, and Dhauliganga valleys in Chamoli, Uttarakhand, India, causing widespread devastation and severely damaging two hydropower projects. As per report, more than 200 people were killed or are missing.[1] Monitoring glacier surface motion is essential for assessing and forecasting glacier-related hazards.Centroid Normalized Cross-Correlation (Centroid NCC)[2] widely used method for glacier motion tracking, operating in the spatial and frequency domains, respectively. Despite their popularity, the effectiveness of these methods can be compromised by factors such as the repetitive texture patterns present on glacier surfaces and the inherent speckle noise in SAR imagery. To address these issues, Deep Matching Network (DMN)[3] proposed for accurate motion estimation from SAR images. The DMN framework is trained end-to-end to learn patch correspondences directly, eliminating the need for manually engineered features. It uses Siamese architecture for extracting meaningful deep representations and utilizes a metric module for precise similarity assessment. Dense connectivity within the feature extractor ensures efficient feature propagation and reuse. DMN handles speckle noise and intensity variation effectively.

1.2 Our Proposed Work

During the implementation of the DMN, we observed that adapting the search window size significantly reduced computational load while maintaining accuracy. By introducing adaptive search window we reduced the search window size and by halving the patch we achieved finer sub-patch correspondence. Additionally, we introduced a simulated SAR method which uses CBAM GAN that reconstruct the SAR imagery. During reconstruction process it maintains spatial properties of image.

Chapter 2

Related Theory

2.1 Gaussian Pyramid

The Gaussian pyramid is a widely used method in image processing and computer vision for creating multi-scale representations of images [4]. It works by systematically applying smoothing and downsampling operations to an image, resulting in a hierarchy of images with progressively lower resolutions. This multi-scale representation is particularly valuable in tasks such as image blending, compression, and object detection, where analyzing features at different scales enhances performance.

In each step of the pyramid construction, the image is first smoothed using a Gaussian kernel, effectively applying a low-pass filter that suppresses high-frequency components like edges and noise. The smoothed image is then downsampled by reducing its dimensions by half in both width and height. Repeating this process generates a sequence of increasingly coarser and blurrier images.

The pyramid begins with the original image at level 0, and with each higher level, the image resolution decreases further. This progressive reduction captures image structure at various scales, enabling robust multi-resolution analysis in a wide range of computer vision applications.

Let $I_0(x, y)$ denote the original image at level 0 of the pyramid. The Gaussian pyramid $\{I_k(x, y)\}_{k=0}^{N}$ is constructed using the following recursive relation:

$$I_k(x, y) = \sum_{m=-2}^{2} \sum_{n=-2}^{2} G(m, n) \cdot I_{k-1}(2x + m, 2y + n), \quad \text{for } k = 1, 2, \ldots, N \tag{2.1}$$

where $I_k(x, y)$ is the image at level k of the pyramid. $G(m, n)$ is the Gaussian kernel, typically a 5×5 or 3×3 matrix that defines the weights for smoothing. The downsampling is performed after convolution, effectively halving the image size at each level.

The Gaussian kernel is usually separable and can be written as the outer product of two 1D Gaussians. A common choice for a 5×5 Gaussian kernel with standard deviation σ is:

$$G(x, y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{2.2}$$

By capturing information at multiple scales, the Gaussian pyramid allows for efficient and robust processing in scenarios where scale variability is a concern. Example of gaussian pyramid shown in diagram 2.1

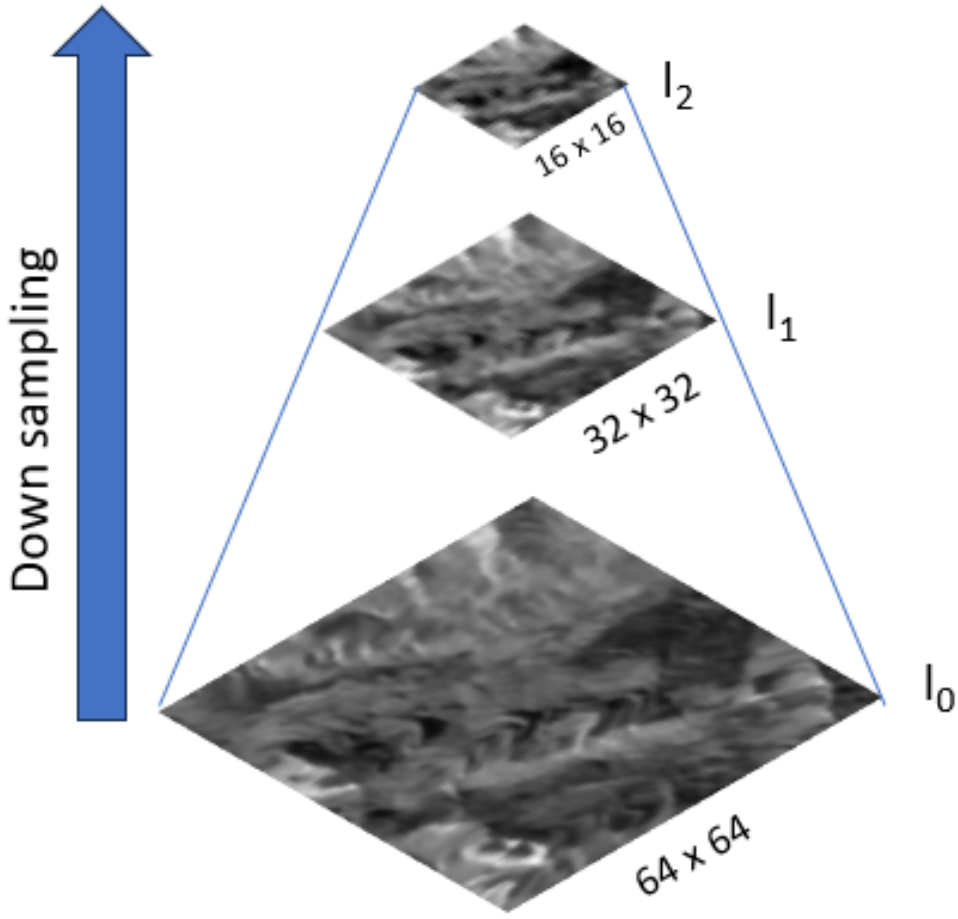


Figure 2.1: Gaussian Pyramid of greyscale image of size 64 x 64

2.2 Template Matching

Template matching [5] is a fundamental technique in image processing and computer vision used to locate a specific pattern or object (called the template) within a

larger image. This method involves comparing portions of the input image with a predefined template to find regions with high similarity.

The basic idea of template matching is to slide the template image over the input image and compute a similarity measure at each position. The location where the similarity score is maximized (or minimized, depending on the metric) indicates the best match. This method assumes that the template and the object to be detected are of the same scale and orientation.

Let $I(x, y)$ represent the input image and $T(x', y')$ be the template image of size $w \times h$. The template is compared to each sub-region of the input image, and a similarity metric $R(u, v)$ is computed for each position (u, v). A commonly used metric is the normalized cross-correlation (NCC), defined as:

$$R(u,v) = \frac{\sum_{x'=0}^{w-1}\sum_{y'=0}^{h-1}\left[T(x',y') - \bar{T}\right]\left[I(x'+u, y'+v) - \bar{I}_{u,v}\right]}{\sqrt{\sum_{x'=0}^{w-1}\sum_{y'=0}^{h-1}\left[T(x',y') - \bar{T}\right]^2} \cdot \sqrt{\sum_{x'=0}^{w-1}\sum_{y'=0}^{h-1}\left[I(x'+u, y'+v) - \bar{I}_{u,v}\right]^2}} \tag{2.3}$$

where (u, v) denotes the top-left corner of the region in the input image where the template is being compared. $\bar{T}$ is the mean of the template pixel values. $\bar{I}_{u,v}$ is the mean of the image region starting at (u, v) with the same size as the template.

The result of template matching is a similarity map, where high values of $R(u, v)$ indicate better matches. The coordinates corresponding to the maximum response in this map identify the most likely position of the template in the input image.

2.3 GAN-Based SAR Image Reconstruction with CBAM

1. Generator Architecture The generator utilizes a two-stage coarse-to-fine architecture. The coarse network (often an encoder–decoder such as a U-Net) first provides a rough estimate of missing SAR data by integrating global context. The refinement network then takes the coarse output along with the surrounding known pixels to produce a high-quality image, sharpening details and ensuring consistency. To enhance feature learning, Convolutional Block Attention Modules (CBAM) are inserted within the generator. Each CBAM module applies channel-wise and spatial attention sequentially, acting as a dynamic filter that highlights important features while suppressing irrelevant ones [6]. In practice, the encoder features are passed through a CBAM block before decoding; this adaptively rescales channel and spatial feature responses so that salient regions are emphasized, leading to more detailed reconstructions [6].

2. Discriminator (PatchGAN) The discriminator is based on the PatchGAN paradigm. Instead of producing a single real/fake score for the entire image, it classifies each $N \times N$ patch as real or fake. In implementation, the discriminator is a CNN whose final feature map is treated as a grid of patch-wise authenticity outputs [7]. This local discriminator encourages the generator to produce realistic texture and fine detail in every local region, helping to sharpen high-frequency content. During training, the discriminator takes either a real full SAR image or a generated image (with missing regions filled) and outputs a matrix of patch labels, which are compared against the real or fake labels using an adversarial loss.

3. Perceptual Loss (Pretrained VGG16) To encourage semantic consistency between the reconstructed and ground-truth SAR images, a perceptual loss is employed. This loss compares high-level feature representations of the two images, extracted from a deep network pretrained on natural images (typically VGG16) [8]. Formally, if $\phi_l(\cdot)$ denotes the activation of layer l in VGG16, the perceptual loss is

$$\mathcal{L}_{\text{perc}} = \left\| \phi_l(\hat{I}) - \phi_l(I) \right\|_2^2,$$

where $\hat{I}$ is the reconstructed image and I is the original. By measuring differences in feature space rather than raw pixels, the perceptual loss encourages the generator to produce outputs that are close to the ground truth in content and style.

4. Training Objectives and Optimization The model is trained end-to-end with a weighted combination of three losses:

1. Pixel reconstruction loss (typically L1) to enforce global fidelity,
2. Perceptual loss improves visual feature similarity,
3. Adversarial loss from the PatchGAN discriminator.

The total generator loss is

$$\mathcal{L}_G = \lambda_1 \mathcal{L}_{L1} + \lambda_{\text{perc}} \mathcal{L}_{\text{perc}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}},$$

where $\mathcal{L}_{L1} = \|I - \hat{I}\|_1$ where original SAR image represented by I, reconstructed image represented by $\hat{I}$ and $\mathcal{L}_{\text{adv}}$ is the adversarial GAN loss. A typical choice is to set $\lambda_1 \gg \lambda_{\text{perc}} \geq \lambda_{\text{adv}}$, prioritizing low-frequency accuracy while still benefiting from perceptual and adversarial terms [9, 8].

Training proceeds by alternating updates of the generator and discriminator. In each iteration, the discriminator is updated to better distinguish real SAR patches from generated ones, and the generator is then updated to minimize $\mathcal{L}_G$. An optimizer such as Adam is typically used, with a learning rate of 10^{-4} and momentum $\beta_1 \approx 0.5$. This joint optimization framework ensures a balance between structural accuracy and visual realism.

Chapter 3

Dataset

3.1 Study Area

The Yanong Glacier, situated on Gangrigabu Mountain in the southeastern region of the Qinghai–Tibet Plateau, represents the most highly developed glacier in this part of the plateau. Its size and well-defined structure make it a key feature in the regional glaciological landscape. As a eminent glacier in the southeastern sector, the Yanong Glacier plays an critical role in influencing local hydrological processes, climate interactions, and ecological systems. Its location on the Qinghai–Tibet Plateau referred as the "Third Pole" due to its vast ice reserves further underscores its importance in studies related to climate change, glacier dynamics, and water resource management in the region. We took Sentinel-1 SAR data capture on 23 November 2016 and 3 February 2017 for region of interest as shown in Figure 3.1 by red box.

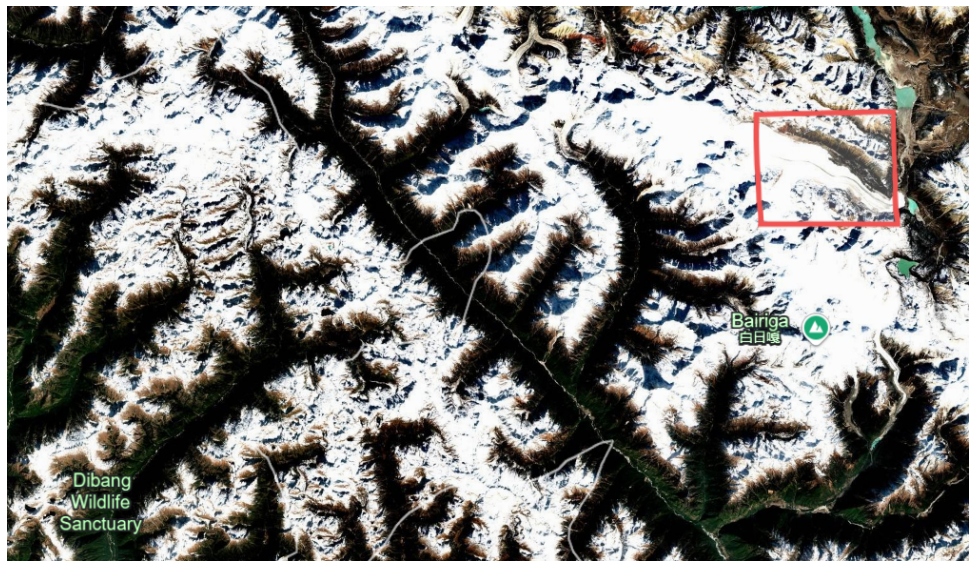


Figure 3.1: Marked box is our study area (Yangong Glacier)

3.2 SAR Imagery

Synthetic Aperture Radar (SAR) intensity images are vital for glacier motion monitoring but present challenges due to variations in intensity, texture, and speckle noise. While Sentinel-1's C-band radar is effective for surface observations and glacier dynamics, L-band radar, such as that used in ALOS missions, penetrates deeper into vegetation and snow. Despite of the general effectiveness of C-band, the resolution of SAR images plays a more crucial role in motion estimation than the frequency band, particularly for glaciers with varying surface characteristics. VV polarization [10] is often preferred for its ability to provide detailed surface scattering characteristics, which helps in detecting surface motions and roughness in glacier areas.

3.3 Preparing Dataset for Siamese Model Deep Matching Network (DMN)

Deep Matching Network (DMN) is based on a Siamese architecture, which processes a pair of image patches to learn feature similarity for glacier motion estimation. Each input sample consists of two grayscale image patches derived from co-registered Synthetic Aperture Radar (SAR) intensity images.

Two separate DMN models are trained using different spatial resolutions to explore the impact of input scale on motion estimation performance. The input configuration for each model is as follows:

- **Model 1:** Input shape $(64, 64, 1) \times 2$
- **Model 2:** Input shape $(32, 32, 1) \times 2$

Each branch of the Siamese network receives one patch from the input pair, and both branches share identical architecture and weights. The patches represent SAR backscatter intensity values and preserves the natural dynamic range and speckle properties inherent to SAR data, which can be important for detection of texture and find motion patterns.

To train the Siamese deep matching network, a high-resolution SAR intensity image of size 512×512 pixels was selected. We use Range Doppler Terrain Correction, so that pixels match ground reality. A sliding window approach was used to extract overlapping patches of size 64×64 pixels with a stride of 8 pixels in both directions, yielding a total of 3,249 patches.

Each patch was considered a base sample and augmented using Gaussian blur (coefficient in $[0, 0.5]$), small rotations (within $\pm 5°$), contrast variations (factor be-

tween 0.5 and 1.5), and multiplicative noise with fixed variance of 0.1. Three augmentations were generated for every base patch to improve data diversity.

To construct training pairs, each original patch was paired with its three augmentations to form matched pairs. Additionally, it was paired with three different patches or their augmentations to form non-matched pairs. This resulted in a total of 19,494 patch pairs, equally balanced between matched and non-matched classes. The same procedure was repeated for smaller patch sizes (32×32) extracted from images of reduced dimensions, enabling analysis of the effect of spatial resolution on network performance.

3.4 Preparing Dataset for CBAM GAN

The CBAM-GAN model used in this study is designed for SAR image inpainting, where the objective is to reconstruct missing or occluded regions in remote sensing imagery. The network is trained using grayscale SAR intensity patches of size 256×256 pixels, and leverages the Convolutional Block Attention Module (CBAM) to enhance attention to important spatial and channel-wise features.

Each training sample consists of the following components:

- **Input Image:** $I \in \mathbb{R}^{1\times256\times256}$ — a single-channel SAR patch with pixel values normalized to the range $[0, 1]$.
- **Binary Mask:** $M \in \{0, 1\}^{1\times256\times256}$ — randomly generated during training to simulate occluded regions.
- **Masked Input:** $I_{\text{masked}} = I \cdot (1 - M)$ — the final input to the generator, with missing areas there value will be zero.

The generator receives both the masked image I_{masked} and the mask M as inputs. It attempts to reconstruct the original image content within the masked regions. The discriminator evaluates the realness of the generated images, distinguishing between real SAR patches and the generator's output.

For training the CBAM GAN model,we used a high-resolution SAR intensity image of size 13000×13000 pixels. To make the data manageable and suitable for deep learning, the image is divided into non-overlapping patches of size 256×256 pixels. This will give total of $50 \times 50 = 2500$ patches, each of which is treated as a separate training sample. Each sample consists of three components: the original SAR image patch, a corresponding binary mask indicating occluded regions, and the masked image obtained by applying the mask to the original. All patches are normalized to the $[0, 1]$ range to facilitate stable training.

The input to the model includes both the masked image and the binary mask, each having a shape of $(1, 256, 256)$, while the output (target) is the original unmasked patch of the same shape. To simulate occlusions in the training data, a custom function was implemented to generate irregular binary masks with random coverage levels. [11] The function supports six predefined coverage classes, ranging from 1% to 60%, and randomly selects one class for each mask. For each training sample, the mask is initialized as a zero-valued binary image. Random line segments, simulating brush strokes with varying thicknesses, are drawn to mimic occlusion. Additionally, elliptical shapes are added with a 30% probability to further diversify occlusion patterns. This process continues until the mask's mean pixel value, corresponding to the occlusion coverage, falls within the selected coverage class. If the generated mask exceeds the maximum coverage, excess non-zero pixels are randomly removed to ensure compliance with the upper bound. The final binary mask, where a pixel value of one indicates pixels that need to be reconstruct, is returned in a PyTorch-compatible tensor format. This strategy ensures diverse and realistic occlusion patterns, enhancing the robustness of the CBAM GAN model during training.



Figure 3.2: (a) Input Image, (b) Binary Mask (c) Masked Input

Chapter 4

Methodology

To support long-term and large-scale monitoring of glacier dynamics, this study presents a deep learning-based glacier motion estimation method using SAR intensity images. A Deep Matching Network (DMN) is employed to automatically and accurately match image pairs. To improve efficiency and accuracy, the method integrates adaptive patch sizes and variable search windows.

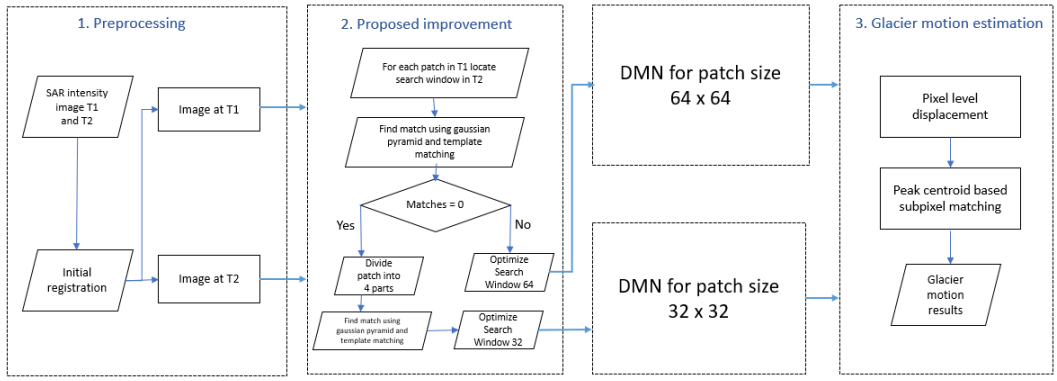


Figure 4.1: Framework for proposed method

In Figure 4.1, a three-step process is shown for glacier motion estimation using SAR intensity images. Initially, preprocessing involves registering SAR intensity images from two different time points, T1 and T2. The proposed improvement finds an optimize search window using gaussian pyramid and template matching. If no matches are found, the patch is divided into smaller parts, and the search window is optimized. Finally, the glacier motion estimation step uses a Deep Matching Network (DMN) for different patch sizes to determine pixel-level displacement and peak centroid-based subpixel matching, resulting in glacier motion estimation outputs.

4.1 Preprocessing SAR intensity images

Before glacier motion estimation, Sentinel-1 SAR intensity images must undergo key preprocessing steps to ensure geometric and radiometric integrity. The process begins by applying precise orbit files to correct satellite position inaccuracies using metadata. Radiometric calibration, though optional, can be performed to convert DN values to backscatter coefficients like Sigma0 [12], helps us in quantitative analysis of surface scattering.

Geometric distortions caused by terrain and radar imaging geometry are corrected using Range-Doppler Terrain Correction [13]. This step uses a Digital Elevation Model (e.g., SRTM 3Sec) [14] and resamples the data to a specified pixel spacing (e.g., 10 m), producing orthorectified outputs in a consistent map projection (such as UTM). Speckle filtering can optionally be applied at this stage to reduce noise, typically using the Lee or Refined Lee filter [15].

For glacier motion estimation, multiple SAR images are co-registered by creating a stack and selecting a master image for alignment. Initial coregistration [16] uses orbit metadata, followed by fine registration through cross-correlation. We take subset of this coregistered image. If using IW mode, the debursting step ensures continuity across bursts. Then we create a subset from that coregistered image shown in Figure 4.2.

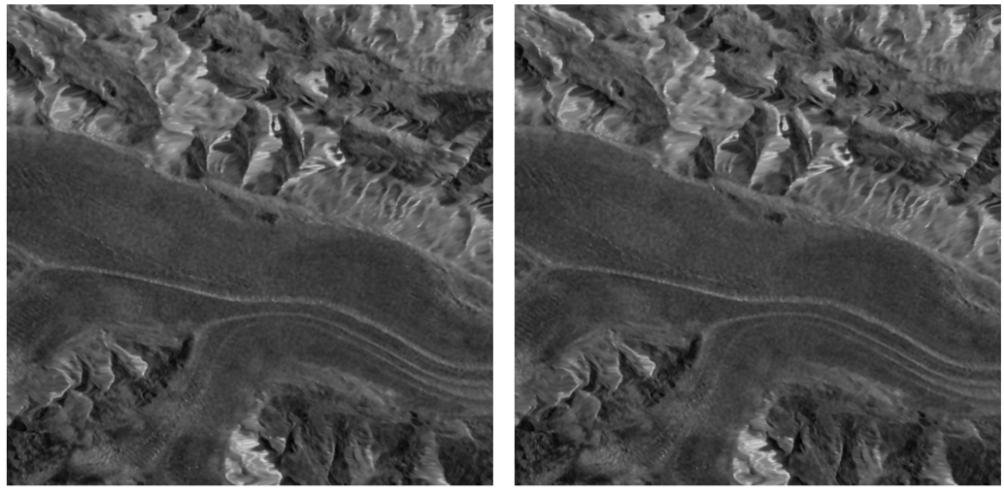


Figure 4.2: Preprocessed SAR intensity images of the Yanong Glacier. (a) was captured on 23 November 2016. (b) was captured on 3 February 2017

4.2 Deep Learning Models

4.2.1 Deep Matching Network for SAR Image Matching

The DMN combines a feature extraction network with a metric network use to handle patch-based inputs from large SAR images. For feature extraction, we use Deep Convolutional Neural Networks with Dense Connection Blocks (DCNN-DCB) [17] to extract hierarchical features. Dense blocks enhance gradient flow and feature reuse by concatenating all previous feature maps:

$$x_k = \mathcal{S}([x_0, x_1, ..., x_{k-1}])$$

where $\mathcal{S}(\cdot)$ includes convolution, and ReLU. Each output is:

$$f_j^l = \sigma\left(\sum_i f_i^{l-1} * \omega_{ij}^l + b_j^l\right)$$

with $\sigma(x) = \max(0, x)$ as the ReLU function. The DMN uses a Siamese structure to process input patch pairs (x_i, y_i) from SAR images X and Y. Both branches share weights and output deep features $\hat{x}, \hat{y}$ that are compared by the metric network. The similarity between features is computed using the Euclidean distance (D_W): The network is trained using a contrastive loss [18]:

$$\mathcal{L} = \frac{1}{2N}\sum_{n=1}^{N} \text{label} \cdot D_W^2 + (1 - \text{label}) \cdot \max(0, m - D_W)^2$$

Here, label = 1 for matched pairs and 0 otherwise, m is a margin, and N is the number of training pairs. The loss minimizes distances for matched patches and separates non-matched ones beyond a threshold. Figure 4.3 shows the network structure.

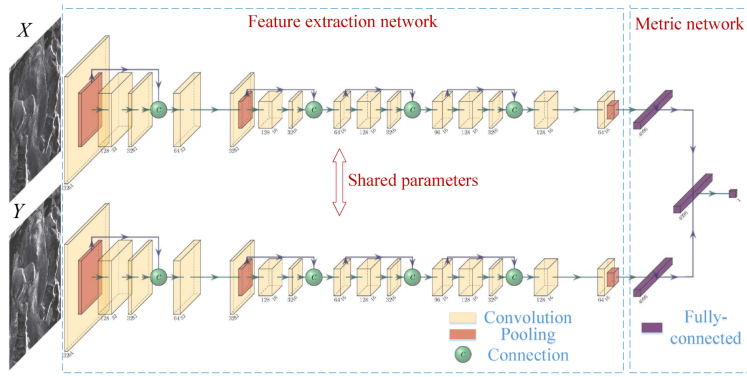


Figure 4.3: Proposed framework of a deep matching network

4.2.2 Methodology for Simulated SAR Image Generation

For DMN evaluation, simulated SAR is generated using affine transformation of whole SAR image at time $T1$ with some small constant shift in y direction but variable shift within a small range in x direction. But in reality different parts of SAR image can move in different direction. To mitigate this problem we introduce a different method.

The generation of simulated Synthetic Aperture Radar (SAR) images involves a sophisticated process that includes affine transformations, masking, and Generative Adversarial Networks (GANs). Initially, the SAR image at time $T1$ is divided into smaller patches, each subjected to random affine shifts. These shifts, which involve translations in both the x and y directions, simulate the natural displacement of patches over time. The magnitude of these translations is randomly determined within a specified range to introduce variability.

Following the affine transformations, the displaced patches are masked to create regions of missing data. This masking process involves blacking out areas opposite to the direction of the shift, effectively creating voids that simulate gaps due to temporal changes or occlusions. The displacement vectors and masked regions are recorded for further processing.

The final step employs a GAN to fill in the masked regions and generate a complete SAR image at time $T2$. The GAN's Generator component takes the masked image as input and produces a complete image by filling in the missing regions, while the Discriminator ensures the realism of the generated image. This methodology uses affine transformations, masking, and GAN-based inpainting to produce high-quality synthetic SAR images, useful for applications such as environmental monitoring and disaster response.

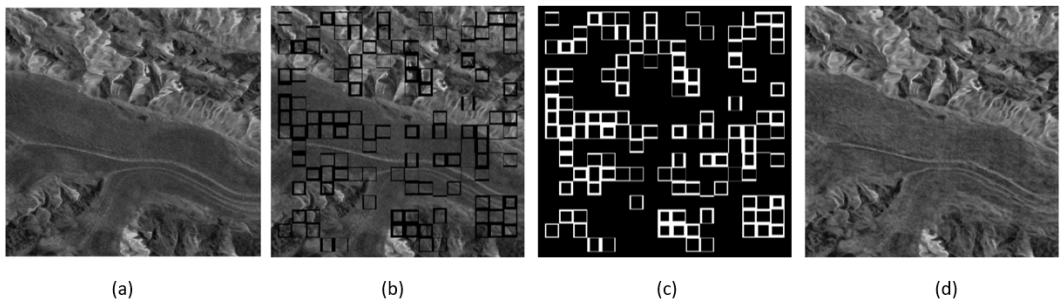


Figure 4.4: (a) Original SAR Image (b) Masked Image (c) Binary Mask (d) Simulated SAR image

4.3 Post Processing

Subpixel Displacement Estimation Using Centroid Method

Deep learning-based approaches such as Deep Matching Networks (DMNs) are effective for pixel-level displacement estimation. However, achieving precise subpixel-level displacement is essential in applications involving glacier motion monitoring and is often constrained by the resolution and quality of the SAR images.

To enhance the accuracy beyond integer-pixel alignment, subpixel displacement estimation techniques are employed. Among them, the peak centroid method is widely used due to its simplicity and robustness. This method determines the subpixel location of the correlation peak by computing the weighted centroid of correlation values in a local neighborhood. [19]

Let Ω denote a square neighborhood centered at the peak of the correlation surface, typically of size 5×5 pixels. The subpixel offsets $(\Delta a, \Delta b)$ from the integer-valued peak location are calculated as:

$$\Delta a = \frac{\sum_{a,b\in\Omega} a \cdot C(a,b)}{\sum_{a,b\in\Omega} C(a,b)}, \qquad \Delta b = \frac{\sum_{a,b\in\Omega} b \cdot C(a,b)}{\sum_{a,b\in\Omega} C(a,b)} \tag{4.1}$$

Here, a and b represent the horizontal (column) and vertical (row) coordinates within the window Ω, and $C(a,b)$ denotes the correlation value at position (a,b). Only those correlation values that are greater than the local mean within Ω are considered to enhance robustness against noise.

Displacement Estimation

To begin with, SAR intensity images of size 1152×1152 pixels are preprocessed using Range Doppler terrain correction, assisted by an external digital elevation model (DEM) to correct for topographic effects. Following this, the images are coregistered to align features across scenes.

The displacement estimation process uses a trained Deep Matching Network (DMN). The input image is divided into 64×64 pixel patches. For each reference patch, a search is performed in a 96×96 pixel window in the target image with a 1-pixel stride. The network selects the best-matching patch and calculate pixel displacement, and subpixel displacement uses centroid-based method, which estimates the peak offset from the correlation surface. Then combine both pixel and subpixel displacement.

To address this, a 3×3 mean filter is applied, improving spatial consistency by removing abrupt changes and non-glacier motions. This post-processing step results in a cleaner and more reliable glacier displacement map.

4.4 Proposed Methodology for Velocity Estimation in SAR Images

The proposed methodology for velocity estimation in Synthetic Aperture Radar (SAR) images utilize a multi-faceted approach combining Gaussian pyramid-based template matching and dividing patch size, and deep learning models. This methodology is designed to estimate the displacement and velocity of surface features between two temporal SAR images, addressing the existing complexities and variations in SAR imagery.

Initially, the preprocessed SAR images at times $T1$ and $T2$ are divided into smaller patches, typically of size 64×64 pixels in both the x and y directions, to simulate potential displacements. Gaussian pyramids constructed for both the reference patch from the $T1$ image and the search window from the $T2$ image, enabling multi-scale template matching. This multi-scale approach improves the robustness of the matching process by capturing spatial features at different levels of detail.

In cases where no matches are found using the 64×64 patch size, we switch to a smaller patch size of 32×32. This reduction in patch size increases the likelihood of finding a match by provides us more detailed and localized search. The decision to reduce the patch size is based on the observation that smaller patches can capture finer details and variations, which we might lose by larger patches. This adaptive approach ensures that the matching process remains robust even in the presence of significant variations in the SAR images.

The search window size is also made variable to further enhance the matching process. The size of the search window is adjusted based on the detected matches at different levels of the Gaussian pyramid. If matches are found did not deviate much from central locations, the search window is reduced to focus on the regions with potential matches, thereby improving the efficiency and accuracy of the template matching. This variable window size approach allows for a more flexible and adaptive search process, which is essential for handling the unnecessary convolutions of SAR imagery.

Deep learning models finds the pixel level displacement estimation. These networks measure the dissimilarity between the reference patch and the candidate patches, providing a similarity metric tailored to the characteristics of SAR images. This approach improves the accuracy of the displacement estimation by helping to learned features and similarities.

Once the best matching patch is identified, the displacement vector is calculated based on the relative location of the matching patch within the search window. This displacement vector is used to estimate the velocity of the surface features by con-

sidering the time interval between the two SAR images. The velocity estimation is performed using a subpixel-level displacement estimation technique, which involves computing the cross-power spectrum of the reference patch and the matching patch. The peak location in the cross-power spectrum is used to estimate the subpixel displacement, which is then combined with the pixel-level displacement to obtain the total displacement vector.

The proposed methodology has been implemented and evaluated using a dataset of SAR images, illustrate its effectiveness in accurately estimating the displacement and velocity of surface features. The use of Gaussian pyramid-based template matching, variable patch sizes, and deep learning models significantly enhances the robustness and accuracy of the velocity estimation, making it a valuable tool for various applications in remote sensing and environmental monitoring.

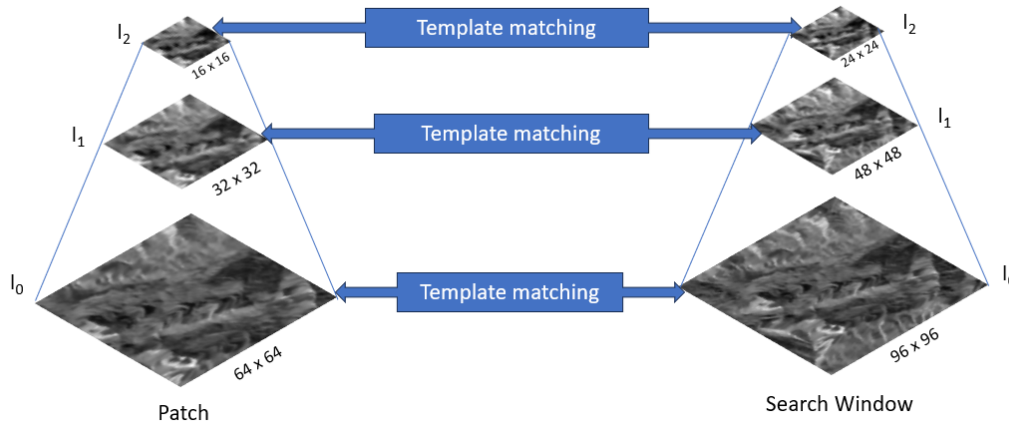


Figure 4.5: Template matching at different levels I_0, I_1 and I_2

4.5 Proposed algorithm for pixel level displacement

P_{ref}: Reference patch, W_{search}: Search window, $\mathcal{R}$: List of match locations with scale. W_{opt}: Optimized search window, $flag$: Matching strategy flag, P_{match}: Best-matched patch. δ: Dissimilarity score, $\vec{d}$: Displacement vector, C_k: Candidate patches, M_{64}, M_{32}: Pre-trained DMN models.

Algorithm 1: Deep Patch-Based Displacement Estimation

$\mathcal{X}, \mathcal{Y}, \mathcal{D}, \mathcal{M} \leftarrow \emptyset$;
for $i = 32$ *to* $H - P - 32$ *step* P **do**
  for $j = 32$ *to* $W - P - 32$ *step* P **do**
    $P_{ref} \leftarrow I_1[i : i + P, j : j + P]$;
    $W_{search} \leftarrow I_2[i - 16 : i + P + 16, j - 16 : j + P + 16]$;
    $\mathcal{R} \leftarrow$ TemplateMatchGaussian(P_{ref}, W_{search});
    if $\mathcal{R} = \emptyset$ **then**
      $flag \leftarrow 32$;
      $W_{opt} \leftarrow W_{search}$;
    else
      Extract (x, y, h, w) from $\mathcal{R}$;
      $W_{opt} \leftarrow$ Adjust window size from all x, y;
      $flag \leftarrow 0$;
    $(P_{match}, \vec{d}) \leftarrow$ FindBestMatch$(P_{ref}, W_{opt}, M_{64}, M_{32}, flag)$;
    Append P_{ref} to $\mathcal{X}$, P_{match} to $\mathcal{Y}$, δ to $\mathcal{M}$, $\vec{d}$ to $\mathcal{D}$;
return $\mathcal{X}, \mathcal{Y}, \mathcal{D}$;

Algorithm 2: TemplateMatchGaussian

Build Gaussian pyramids G_T, G_I up to level L;
$\mathcal{R} \leftarrow \emptyset$;
for $l = 0$ *to* $L - 1$ **do**
  $R \leftarrow$ MatchTemplate$(G_I[l], G_T[l])$;
  Find max_value, $max_location$ in R;
  if $max_val > threshold$ **then**
    $loc \leftarrow 2^l \cdot max_loc$;
    Append (loc, l) to $\mathcal{R}$;
return $\mathcal{R}$;

Algorithm 3: FindBestMatch

if $flag = 0$ **then**
  Extract all 64×64 candidates C_k from W;
  Predict dissimilarity scores $\delta_k \leftarrow M_{64}(P_{ref}, C_k)$;
  $C^* \leftarrow \arg\min_k \delta_k$;
  return $(C^*,$ location of $C^*)$;
else
  Divide P_{ref} into four 32×32 subpatches;
  for *each subpatch* **do**
    $R \leftarrow$ MatchTemplate(P_{ref}, W_{search});
    Extract local 64×64 search window from R;
    Predict best match using M_{32};
  return (list of best matches, refined locations);

Chapter 5

Experiments and Results

5.1 Model Selection and Trade-off Analysis

To conduct our experiments, we selected a device equipped with an Intel(R) Xeon(R) Gold 6254 CPU. Aiming to balance model accuracy and computational efficiency for patch-based motion estimation in SAR imagery, we explored multiple lightweight network architectures by varying the number of convolutional filters and the repetition count of Dense Connection Blocks (DCB). The DCB structure—designed to concatenate outputs across layers—facilitates feature reuse and enhances gradient flow, helping the network maintain representational capacity even with a reduced parameter count.

Our experiment focused on analyzing the influence of filter count, DCB repetitions, total parameter size, and resulting test losses. Architectures using 32 filters consistently outperformed those with 16 filters in terms of test loss. Notably, the configuration with 32 filters and three DCB repetitions achieved the best performance, recording the lowest test loss of 0.2115 with just 13,536 parameters. In contrast, the same number of DCB repeats with only 16 filters yielded the highest test loss (0.2224), reflecting reduced effectiveness.

This optimal model—with a moderate filter count and efficient DCB structure—demonstrated an effective trade-off between accuracy and model complexity. While more compressed models exhibited slightly higher test losses, they risked underfitting. The selected configuration, therefore, stands out as both computationally lightweight and accurate, making it well-suited for high-resolution SAR imagery applications where both precision and efficiency are essential.

5.2 Analysis of Patch Size and Threshold Selection in Template Matching

In template matching for remote sensing imagery, patch size and similarity threshold significantly influence accuracy and robustness. In this study, we used a hierarchical approach based on Gaussian pyramids (levels = 4) with 64×64 patches extracted from band 1 of Sentinel-1 SAR imagery and matched them against 96×96 search windows in band 2 of size 13000 x 13000 pixels.

At a strict threshold of 0.8, the algorithm recorded 1297 matches and 5,746 non-matches, indicating high precision but moderate rejection. To recover additional matches, each unmatched 64×64 patch was subdivided into four 32×32 sub-patches and re-evaluated with varying threshold_{32} values from 0.4 to 0.9. As shown in Figure 5.1, match count decreases and non-match count increases with higher thresholds.

This behavior highlights a trade-off between sensitivity and reliability. Lower thresholds recover more matches but risk false positives, while higher thresholds are overly strict. The results suggest that a threshold_{32} between **0.55 and 0.60** best balances recovery and accuracy, making it the preferred range for sub-patch matching when primary patch matching fails.

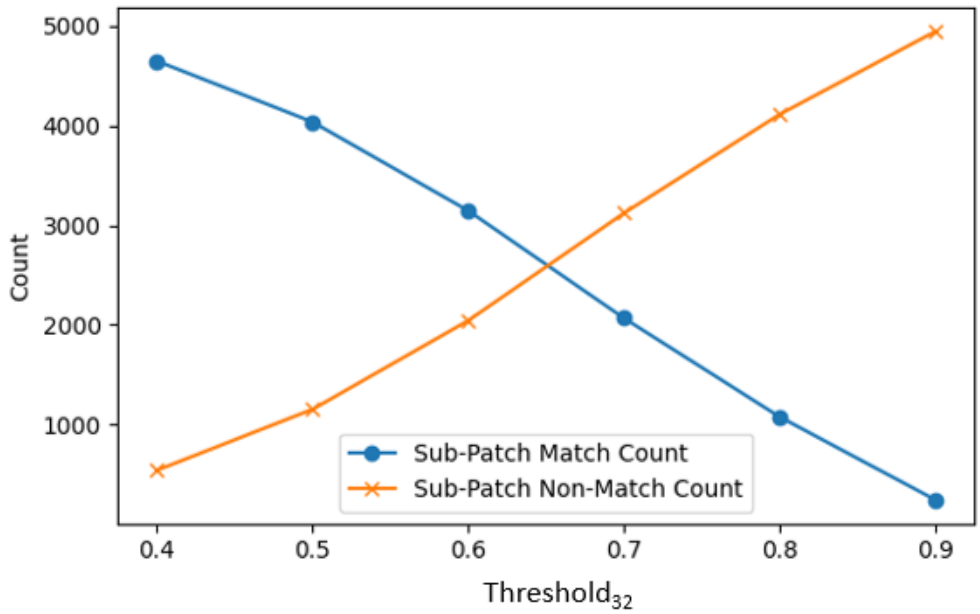


Figure 5.1: Sub-Patch Matching for $\text{Threshold}_{64} = 0.8$ with varying Threshold_{32}

5.3 Root Mean Square Error (RMSE)

The Root Mean Square Error (RMSE) is a widely used metric for quantifying the difference between predicted values and actual observations. It is calculated as the square root of the average of the squared differences between the predicted and observed values. The formula is given by:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

where $\hat{y}_i$ represents the predicted value, y_i the true value, and n the total number of observations. RMSE is especially effective in highlighting larger errors, as it penalizes them more heavily, making it a reliable indicator of significant prediction deviations.

Figure 5.2 presents the RMSE values across varying noise variances for three image matching techniques: Centroid Normalized Cross-Correlation (Centroid NCC), Deep Matching Network (DMN), and the Proposed Method, which incorporates adaptive search window and patch sizing within the DMN architecture.

Among the three, Centroid NCC consistently yields the highest RMSE values, reflecting lower accuracy in matching performance. The DMN method shows improved results with generally reduced RMSE, indicating greater robustness to data variation. The Proposed Method achieves the lowest RMSE across all variance levels, demonstrating that dynamic adjustment of search window and patch size within the DMN framework leads to more accurate and reliable image matching outcomes.

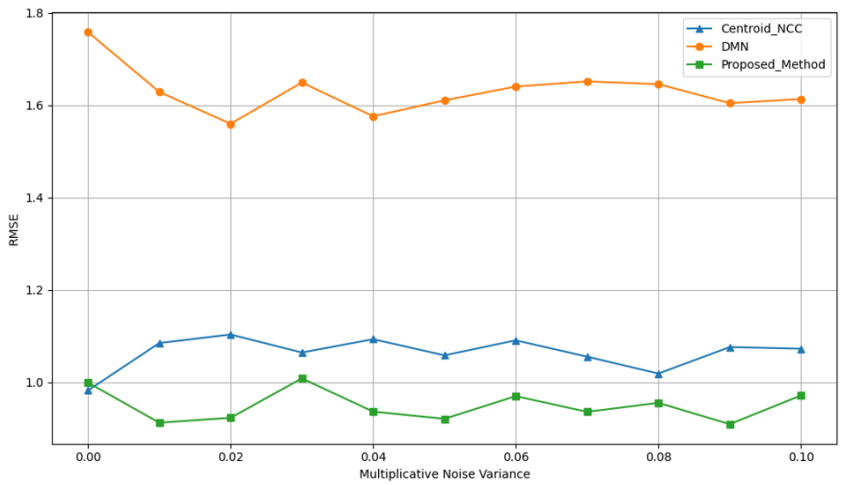


Figure 5.2: RMSE for different Multiplicative Variance Noise for different image matching methods

5.4 Analysis of time

This section evaluates the computational efficiency of three image matching techniques applied to SAR imagery of size 1152×1152 pixels: Centroid Normalized Cross-Correlation (Centroid NCC), Deep Matching Network (DMN), and a Proposed Method that integrates adaptive search window and patch size selection within the DMN framework. The relationship between noise variance and computation time is illustrated in Figure 5.3.

The DMN approach exhibits a consistently high computation time across all noise levels, indicating that its processing load is largely unaffected by input variability. This stability suggests that the method's complexity stems from its internal architecture, rather than the characteristics of the data.

In contrast, the Proposed Method shows a more adaptive response to noise. Although it experiences a slight rise in computation time at lower variance levels, the increase plateaus with higher variance, reflecting its robustness and efficiency in dealing with noisy inputs.

The Centroid NCC technique stands out for its consistently low computation time, highlighting its computational simplicity. However, its noticeable sensitivity to minor noise changes at lower variance levels suggests that it may lack scalability and resilience in more complex or degraded conditions.

Overall, the Proposed Method provides a balanced trade-off—delivering improved efficiency under noise while preserving performance—whereas DMN ensures consistent but costly processing, and Centroid NCC prioritizes speed over robustness.

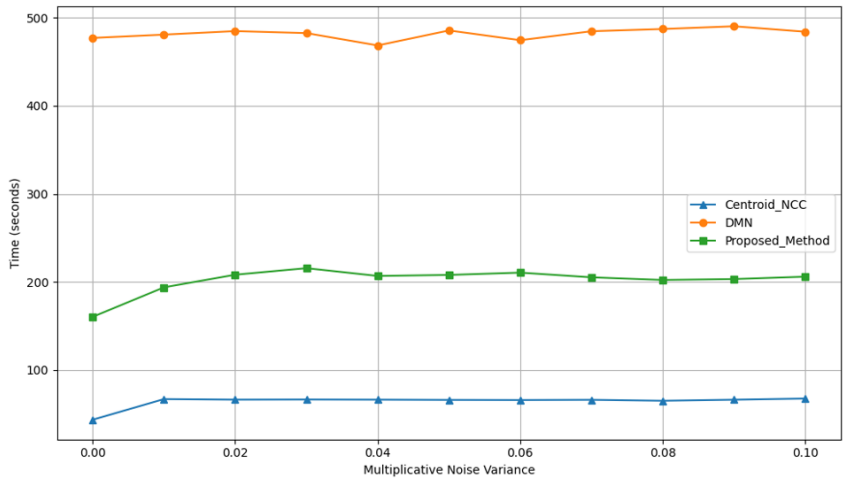
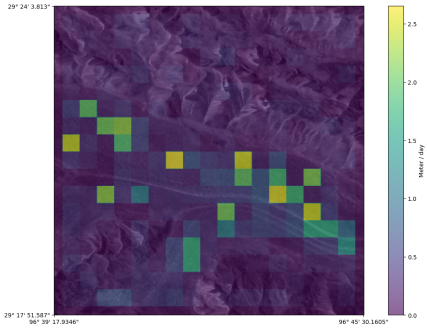


Figure 5.3: Time taken vs Multiplicative Noise Variance for different image matching methods.

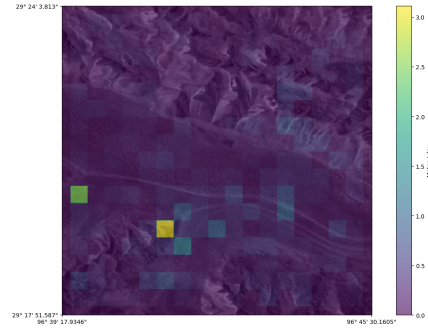
5.5 Results

The velocity magnitude maps generated by various methods provide crucial insights into the dynamics of glacier movement. The Centroid Normalized Cross-Correlation (NCC) method tends to produce broader, more generalized motion patterns. Due to its dependence on basic correlation operations, it often overlooks finer motion details and is more susceptible to noise and surface texture variations, resulting in lower estimation accuracy. In comparison, DMN uses the capabilities of deep learning to identify spatial patterns and variations in SAR imagery. This leads to significantly more detailed and accurate velocity estimations, as the model learns to interpret complex visual within the data.

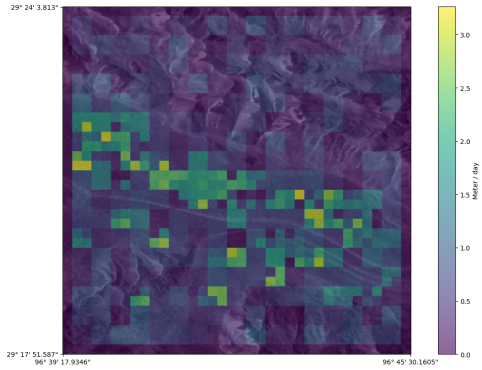
The Proposed Method enhances this further by incorporating adaptive search window and patch sizing within the DMN framework. By tailoring its parameters to the local characteristics of the image, it achieves a more precise and nuanced depiction of glacier motion. This flexibility proves particularly advantageous under varying noise conditions and heterogeneous surface structures, making the method both robust and well-suited for reliable glacier monitoring. Figure 5.4 illustrates the velocity magnitude maps generated by each approach.



(a) Velocity magnitude map using Centroid NCC



(b) Velocity magnitude map using DMN



(c) Velocity magnitude map using Proposed Method

Figure 5.4: Velocity magnitude maps using different methods

Chapter 6

Conclusion and Future Work

In this study, we introduced a Proposed Method that adapts the search window and patch size within the Deep Matching Network (DMN) framework for SAR image matching. This method outperforms traditional approaches like Centroid Normalized Cross-Correlation and standard DMN in computational efficiency and robustness to noise. The adaptability of the Proposed Method enhances accuracy and ensures efficient resource utilization, making it promising for large-scale SAR image processing. When we processed with GPU, DMN and Proposed Method consumes similar time.

Future research can focus on several areas to build on the strengths of the Proposed Method. Enhancing the adaptive mechanisms using advanced machine learning techniques could further improve performance. We can apply attention based architecture which can predict patch matching that should be light weight which can removes unnecessary computations. Integrating the method with other imaging modalities and sensor data could expand its applicability. Developing real-time implementations and optimizing the method for speed and scalability are crucial next steps. Additionally, extensive validation and testing across diverse datasets and scenarios will establish the method's reliability. Exploring parallel processing techniques, such as GPU acceleration, could enhance computational efficiency, enabling faster processing of large SAR images. To reduce computation we can use distributed system, which also enables parallelization.

Bibliography

[1] D. H. Shugar and M. Jacquemart, "A massive rock and ice avalanche caused the 2021 disaster at chamoli, indian himalaya," *Science*, vol. 373, no. 6552, pp. 300–306, 2021.

[2] J. Lewis, "Fast normalized cross-correlation," *Ind. Light Magic*, vol. 10, 10 2001.

[3] S. Huifang, Z. Shudong, F. Li, and Y. Jian, "Glacier motion monitoring using a novel deep matching network with sar intensity images," *Remote Sens*, 2022.

[4] S. Konlambigue, J.-B. Pothin, P. Honeine, and A. Bensrhair, "Fast and accurate gaussian pyramid construction by extended box filtering," in *2018 26th European Signal Processing Conference (EUSIPCO)*, pp. 400–404, 2018.

[5] N. S. Hashemi, R. B. Aghdam, A. S. B. Ghiasi, and P. Fatemi, "Template matching advances and applications in image analysis," *CoRR*, vol. abs/1610.07231, 2016.

[6] Y. ZHANG, Q. ZHANG, J. GU, H. YU, and D. XU, "U-net model based on cbam attention mechanism for coronary angiography segmentation," *Journal of Mechanics in Medicine and Biology*, vol. 24, no. 09, p. 2440062, 2024.

[7] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5967–5976, 2017.

[8] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *Computer Vision – ECCV 2016* (B. Leibe, J. Matas, N. Sebe, and M. Welling, eds.), (Cham), pp. 694–711, Springer International Publishing, 2016.

[9] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros, "Context encoders: Feature learning by inpainting," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2536–2544, 2016.

[10] D. S. Vaka, S. Sharma, and Y. Rao, "Comparison of hh and vv polarizations for deformation estimation using persistent scatterer interferometry," 10 2017.

[11] G. Liu, F. A. Reda, K. J. Shih, T.-C. Wang, A. Tao, and B. Catanzaro, "Image inpainting for irregular holes using partial convolutions," in *Proceedings of the European Conference on Computer Vision (ECCV)*, September 2018.

[12] P. Kaushik and S. Jabin, "A comparative study of pre-processing techniques of sar images," in *2018 4th International Conference on Computing Communication and Automation (ICCCA)*, pp. 1–4, 2018.

[13] G. Wang, W. Zi, C. Xie, and F. Zhang, "Dual-aspect geometric and radiometric terrain correction method for high-resolution sar data," in *2011 IEEE International Geoscience and Remote Sensing Symposium*, pp. 1894–1897, 2011.

[14] L. Yang, X. Meng, and X. Zhang, "Srtm dem and its application advances," *International Journal of Remote Sensing - INT J REMOTE SENS*, vol. 32, pp. 3875–3896, 07 2011.

[15] A. S. Yommy, R. Liu, Wu, and Shuang, "Sar image despeckling using refined lee filter," in *2015 7th International Conference on Intelligent Human-Machine Systems and Cybernetics*, vol. 2, pp. 260–265, 2015.

[16] G. Tang, Z. Wei, and L. Zhuang, "Sar image registration with sift features and edge points," in *IGARSS 2024 - 2024 IEEE International Geoscience and Remote Sensing Symposium*, pp. 9711–9715, 2024.

[17] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269, 2017.

[18] F. Wang and H. Liu, "Understanding the behaviour of contrastive loss," *CoRR*, vol. abs/2012.09740, 2020.

[19] Z. Ye, Y. Xu, H. Chen, J. Zhu, X. Tong, and U. Stilla, "Area-based dense image matching with subpixel accuracy for remote sensing applications: Practical analysis and comparative study," *Remote Sensing*, vol. 12, no. 4, 2020.