Synthetic Flow Field Generation for Myocardium Deformation Analysis

Shahar Zuler
Supervisors: Gal Lifshitz and Dan Raviv

ABSTRACT

This project presents a method for generating synthetic 3D flow fields to warp real cardiac Computed tomography (CT) frames, creating pairs of images with synthetic deformations that serve as ground truth annotations. A conditional Variational Autoencoder (CVAE) is employed to generate flow fields conditioned on a single CT frame, enabling the synthesis of realistic deformations for training and validating deep learning models in myocardium motion analysis.

Our proposed method significantly reduces the dependency on manual annotations, which are practically infeasible for dense voxel-level deformation tasks. By providing synthetic annotations, this project lays the groundwork for the development of more complex myocardium motion models.

Key findings include: Reconstruction Accuracy: The CVAE model achieved an average Mean Endpoint Error (mEPE) of 0.56 voxels. Realistic Deformations: The model was able to generate realistic and anatomically plausible 3D flow fields for myocardium deformation, conditioned on single CT frames. Qualitative Results: The generated flow fields effectively transformed systole frames into diastole frames, closely mirroring real-world cardiac deformations, with qualitative examples provided in Figure 1 and 5.

This method is able to create realistically looking pairs of cardiac CT frames enriched with dense 3D flow field annotations. Visit the project page for animated generated samples and other complementary material: https://shaharzuler.github.io/CVAEPage.

KEYWORDS

synthetic data generation, myocardium deformation, 3D optical flow, conditional VAE, deep learning

ACM Reference Format:
Shahar Zuler and Supervisors: Gal Lifshitz and Dan Raviv. 2024. Synthetic Flow Field Generation for Myocardium Deformation Analysis. In *Proceedings of Deep Learning Course Final Report*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Cardiac imaging plays a critical role in diagnosing cardiovascular diseases, which are a leading cause of morbidity worldwide. Techniques such as cardiac CT allow for the non-invasive assessment of heart structure and function. However, the accurate interpretation of these scans requires detailed knowledge of myocardial motion, which involves the complex deformation of the left ventricle (LV) across the cardiac cycle.

Manual annotation of dense 3D displacement fields between different timesteps of a cardiac CT scan is virtually unfeasible. The primary challenge lies in visualizing and interpreting the complex movement of 3D voxels using inherently limited 2D representations. Even with the most advanced tools, producing accurate voxel-level displacement annotations is overwhelmingly difficult. The need to track thousands of individual voxel movements in three dimensions makes the process not only exceedingly time-consuming but also highly prone to human error, rendering manual annotation impractical for real-world applications.

The scarcity of annotated data poses a significant obstacle to developing data-driven models for myocardium motion analysis. Despite the availability of advanced machine learning models, their success relies heavily on the quality and quantity of training data. In the absence of annotated 3D deformation data, models often fail to generalize well, especially when dealing with complex physiological movements like cardiac contractions. Even in unsupervised settings, ground truth (GT) annotations are still crucial for validating and benchmarking these models, as robust and accurate validation requires a reliable reference to assess the quality of the predictions.

1.1 Theoretical Background on Variational Autoencoders (VAEs)

Variational Autoencoders are generative models that encode input data into a latent representation and decode this representation to generate new data. Unlike traditional autoencoders, VAEs introduce probabilistic modeling, where the encoder maps the input to a distribution (typically Gaussian) rather than a fixed point. The decoder then samples from this distribution to reconstruct the input.

A VAE minimizes two key objectives:

- **Reconstruction Loss**: Ensures that the generated output resembles the original input by measuring pixel-wise differences (e.g., mean squared error).
- **KL-Divergence Loss**: Regularizes the latent space by encouraging it to follow a standard normal distribution, allowing new samples to be generated by sampling from this space.

In our work, we extend this concept by employing a CVAE, where the generation is conditioned on external inputs, specifically single cardiac CT frames. This enables the model to generate 3D flow fields that are anatomically realistic and tailored to the cardiac state represented by the conditioned frame.

Deep Learning Course Final Report, September 2024, TAU

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

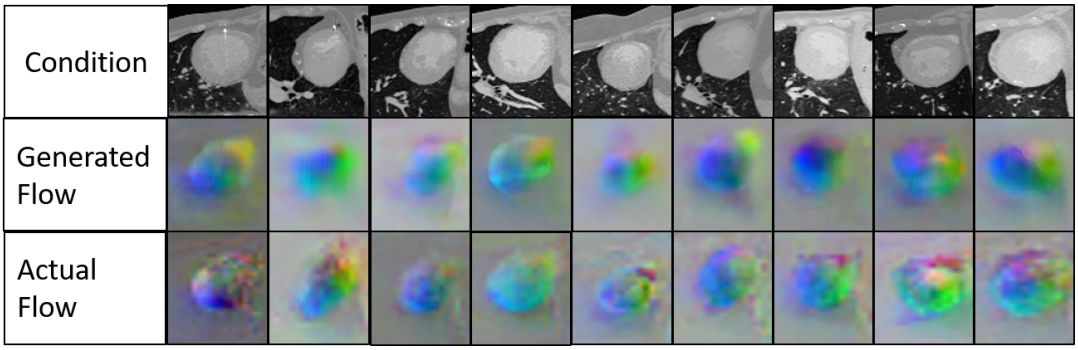

Figure 1: Flow Field Comparison for Cardiac CT Conditions: The top row ("Condition") shows 2D sections from validation CT scans (systole). The middle row ("Generated Flow") displays corresponding flow fields generated by the CVAE for each condition. The bottom row ("Actual Flow") presents the ground truth flow fields obtained from the original CT data. This comparison demonstrates the model's ability to generate plausible deformation patterns from real cardiac CT conditions. Full animated transformations from systole to diastole are available on our project page.

1.2 Our Approach

In this project, we propose a method to overcome the annotation bottleneck by using a CVAE to generate synthetic flow fields conditioned on real cardiac CT frames. These flow fields warp a single CT frame to create pairs or sequences of images with synthetic deformations. These pairs serve as ground truth annotations for training myocardium motion analysis models.

1.3 Contributions

The primary contribution of this work is:

- The development of a novel approach using a CVAE to generate synthetic 3D flow fields conditioned on single CT frames.

2 RELATED WORK

2.1 Generative Models for Image Synthesis

Conditional generative models have been extensively studied in recent years, especially in the field of image generation. The three primary families of models explored in this space include CVAE, Conditional Generative Adversarial Networks (GANs), and Conditional Diffusion Models. Each of these models has unique strengths and weaknesses, depending on the specific application domain.

2.1.1 CVAEs. VAEs were first introduced by Kingma and Welling [5], providing a probabilistic framework for modeling complex data distributions. Sohn et al. [11] extended this by introducing conditions to both the encoder and decoder, allowing the latent space and the generated output to be conditioned on external input data, forming CVAEs.

In general, VAEs, and specifically CVAEs, tend to produce blurrier results compared to GANs. However, in our application, the goal is not to generate high-resolution images but rather 3D flow fields. Flow fields are inherently smoother and less sharp than images, which makes CVAEs an ideal choice. Since flow fields encode motion information, the slight blur introduced by CVAEs does not significantly impact the quality of the generated annotations.

2.1.2 GANs. GANs were first introduced by Goodfellow et al. [2], demonstrating the ability to generate highly realistic images by having a generator network and a discriminator network compete against each other in a minimax game. Mirza and Osindero [6] extended this concept by introducing conditions to both the generator and the discriminator, creating Conditional GANs. This allows conditional GANs to generate images conditioned on specific inputs, making them useful for tasks such as image-to-image translation.

However, GANs, particularly conditional GANs, face two major challenges: unstable training dynamics and a high demand for large annotated datasets. The limited amount of annotated cardiac data available in our project makes conditional GANs impractical for our application.

2.1.3 Conditional Diffusion Models. Diffusion-based models have recently gained significant attention, starting with Ho et al. [3] and further popularized by works like Stable Diffusion by Rombach et al. [8], for their ability to generate highly realistic images from complex distributions. These models can incorporate various types of conditions and rely on an iterative denoising process to produce samples. However, they require a vast amount of data and considerable training time.

While diffusion models show great potential for tasks such as text-to-image or image-to-image generation, their complexity, large data requirements, and extensive training times make them impractical for this project.

2.1.4 Rule-based Methods. Another approach is rule-based synthetic data generation, such as the method introduced by Zuler and Raviv [12]. This approach simplifies the data generation process by applying predefined rules to deform voxel locations in ways that statistically match known cardiac movements. While rule-based methods can produce plausible synthetic data, they tend to appear less realistic and more structured compared to data generated by deep learning models. These methods also lack the complexity needed to capture subtle cardiac deformations. However, their simplicity can still make them useful for augmenting datasets in less demanding applications.

2.2 Myocardium Motion Analysis Across Non-CT Imaging Modalities

Several imaging techniques have been applied to study myocardial motion. In this section, we review some of the most prominent methodologies in the literature.

2.2.1 Speckle Tracking in Ultrasound. Speckle tracking in ultrasound relies on the unique speckle patterns created when ultrasound waves interact with myocardial tissue. By tracking these patterns over time, it is possible to estimate the movement of the myocardium [1]. While effective in analyzing 2D strain, speckle tracking is limited by the operator's experience and the quality of the ultrasound images. Additionally, it does not provide the same level of detail as CT when analyzing deeper cardiac structures.

2.2.2 Magnetic Resonance (MR) Tagging. MR tagging involves introducing a grid or pattern into an MRI scan, which serves as a reference system for analyzing myocardial strain and deformation [4]. MR tagging is a highly accurate technique, but it is resource-intensive and difficult to implement in clinical settings, especially for large datasets. Additionally, the limited through-plane resolution in MRI scans makes CT a more suitable modality for studying 3D myocardial motion.

3 DATA

The data for this project consists of 44 real cardiac 4DCT scans. These scans were collected from two primary sources:

- Two scans were sourced from publicly available repositories (one from the Magix sample via OsiriX [9] and one from the 3D Slicer sample via Slicer [10]).
- Forty-two scans were from our lab's private collection.

3.1 Annotation Process

To annotate the dataset with 3D optical flow:

(1) Four scans were manually segmented using 3DSlicer software to generate left ventricle segmentation masks, which are required for optical flow estimation by CardioSpectrum. CardioSpectrum leverages these segmentation masks to accurately address the aperture problem and capture tangential LV movement.
(2) The remaining forty scans were processed using 3D-PWC-Net, which does not require segmentation masks. 3D-PWC-Net directly estimates optical flow from the raw CT scans. Although 3D-PWC-Net is less precise in capturing tangential flow components compared to CardioSpectrum, the generated flow fields still serve as reasonable ground truth for this project.

A Note About Future Efforts: Future work can improve upon this annotation process by gaining segmentation masks for the entire dataset and using CardioSpectrum for all scans. This would enhance the accuracy of the optical flow annotations, particularly for tangential movements. However, this approach was not pursued in the current project due to the labor-intensive nature of manual segmentation. Expanding the dataset with more scans and leveraging automated or semi-automated segmentation tools could further improve the robustness of the model and the quality of the generated flow fields.

3.2 Preprocessing Pipeline

Preprocessing was applied to ensure consistency across the dataset and minimize noise:

- **Intensity Clamping**: The voxel intensity values were clamped at a maximum absolute value of 2000 to mitigate the effect of artifacts, such as pacemakers present in some scans.
- **Normalization**: The intensity values of the CT scans were normalized between 0 and 1.
- **Resizing**: All images and flow fields were resized to a fixed size of 86,86,76.
- **Augmentation**: To increase the variability in the dataset, the following augmentations were applied:
  - *Intensity Augmentation*: White noise was added to the conditions (CT images and extracted features).
  - *Shift Augmentation*: Small random shifts were applied to both the flow field and the input image.
  - *Zoom Augmentation*: Random zoom-in and zoom-out transformations were applied, with appropriate scaling of the flow field values.

4 METHODS

In this section, we describe the architecture of the CVAE, the conditioning mechanism used to introduce 3D CT features, and the post-processing methods for flow field generation. The training pipeline is illustrated in Figure 2 and the generation pipeline is illustrated in Figure 3.

4.1 CVAE Architecture

The core of our method is a CVAE designed to generate 3D flow fields conditioned on single real CT frames. We chose the end-systole frame (when the LV is most contracted) as the conditioning frame because deformations that enlarge the LV appear more realistic and contain fewer artifacts compared to those that shrink it. The model architecture is depicted in Figure 4.

Encoder-Decoder: The encoder extracts features from the input flow field, compressing the data into a latent space representation. The decoder then reconstructs the flow field from this latent space during training. To ensure that the encoder encodes meaningful latent variables, we employ a KL-divergence loss that regularizes the latent space to follow a normal distribution. During inference, the decoder generates new flow fields from a random latent vector sampled from a standard normal distribution.

- **Encoder Architecture** The encoder processes the input flow field and extracts features. Key components include:
  - **Convolutional Layers:** The encoder consists of 12 3D convolutional layers, with kernel sizes of 3, and strides alternating between 1 and 2. Each convolution is followed by a Leaky ReLU activation, progressively reducing the spatial dimensions while increasing feature depth.
  - **Latent Space Computation:** After feature extraction, the encoder computes the mean and standard deviation for the latent representation using additional convolutional layers.

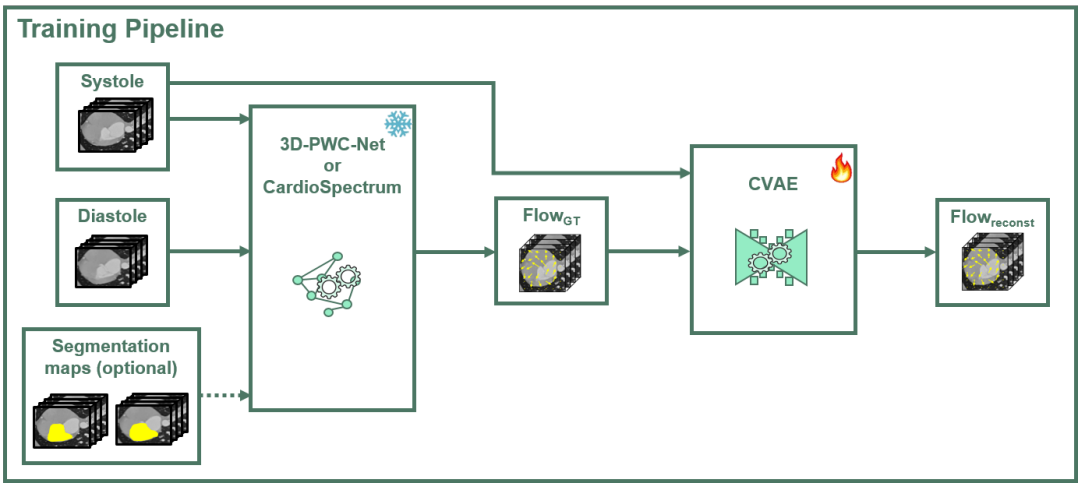

Figure 2: Overview of the training pipeline: Systole and Diastole frames, along with optional segmentation maps, are used to compute ground truth flow fields via 3D-PWC-Net or CardioSpectrum. The CVAE then learns to reconstruct these flow fields, producing $\text{Flow}_{\text{reconst}}$.

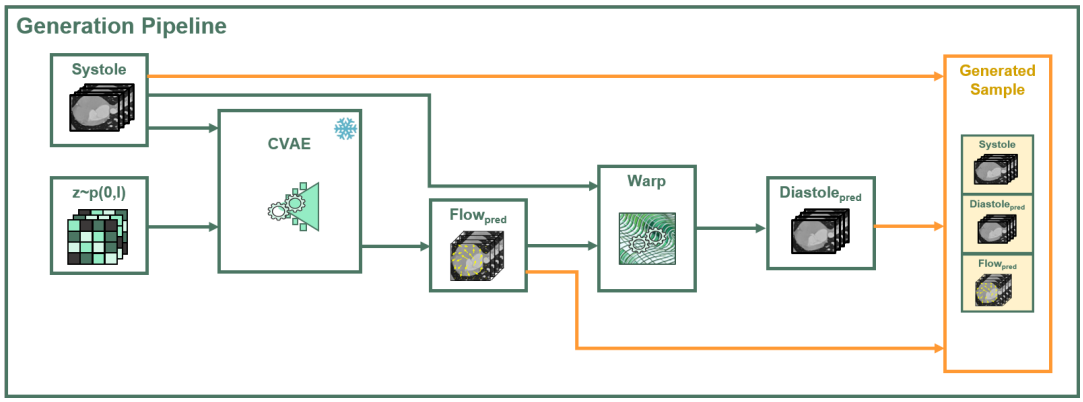

Figure 3: Overview of the generation process: The trained CVAE model generates a predicted flow field ($\text{Flow}_{\text{pred}}$) from a conditioned Systole frame and a random array (z). This flow field is then used to warp the Systole frame, producing a Diastole frame ($\text{Diastole}_{\text{pred}}$), which, along with the flow, forms the final generated sample.

These parameters define the distribution from which the latent variables are sampled.

The full encoder architecture is structured as follows:

- The input flow field is concatenated with the corresponding conditioned features from the CT frame (see Section 4.2).
- It passes through 12 3D convolutional layers, progressively extracting more abstract features while reducing spatial resolution. Every second convolutional layer is followed by concatenation with the corresponding conditioned feature maps.
- The mean and variance of the latent distribution are computed from the deepest feature maps.
- The latent variable z is sampled using the reparameterization trick [5], where $z = \mu + \sigma * \epsilon$, and ϵ being random noise sampled from a standard normal distribution. This trick allows gradients to propagate through the random sampling process, enabling the model to be trained using standard backpropagation.

- **Decoder Architecture:** The decoder mirrors the encoder and is responsible for reconstructing the flow field from the latent variable z and the conditioned pyramid of features. It progressively upsamples the latent space to the original resolution using transposed convolutional layers (also known as deconvolutions).

  The decoder architecture follows this process:

  - The latent variable z is passed through transposed convolutions to progressively increase the spatial resolution while reducing the depth of the channels.
  - At every second layer, the output is concatenated with the corresponding conditioned features from the encoder.

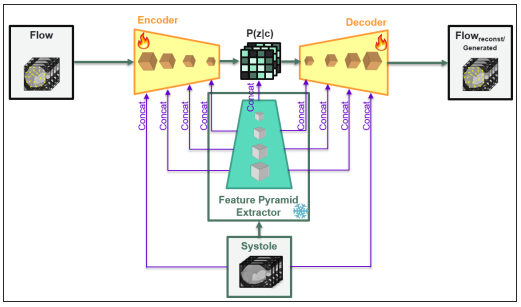

Figure 4: Architecture of the CVAE model used for flow field generation. The encoder processes input features from a pretrained feature pyramid, and the decoder generates the flow field, which is then used to warp the CT frame.

4.2 Conditioning Mechanism

To condition the CVAE, we extract features from the input CT frame using a feature pyramid network. This approach allows us to capture and introduce multi-resolution features at various stages of both the encoder and decoder. Unlike typical image generation tasks, which often condition directly on the image, or condition on a single embedding representation, we condition the model on learned multi-scale features in addition to the original image.

The feature pyramid extractor was originally trained as part of an optical flow task specifically on cardiac CT data. This prior training ensures that the features it learned to extract are highly relevant for capturing the nuances of cardiac motion and deformation. By leveraging these learned features, the CVAE gains a deeper understanding of the cardiac CT optical flow task, which helps in generating anatomically realistic and contextually coherent flow fields across different resolutions.

The feature pyramid extracts features at 5 levels of resolution. These features, as well as the original frame, are concatenated at the corresponding layers of the encoder and decoder (see Figure 4). This allows the network to incorporate global and local features into the generation process, improving the accuracy of the generated flow fields. A detailed table presenting the architecture and network dimensions can be found in the appendix (see Table 1).

4.3 Training, Inference and Loss Functions

During training, the CVAE learns to generate realistic 3D flow fields by minimizing two key losses: the **reconstruction loss** and the **Kullback-Leibler (KL) divergence loss**. The goal of the training process is to encourage the model to generate flow fields that are both faithful to the input and consistent with the learned latent space distribution.

4.3.1 Reconstruction Loss. The reconstruction loss measures the difference between the generated flow field and the ground truth flow field, ensuring that the output from the decoder closely matches the true deformation field. In this project, we use the L_2 (mean squared error) loss function for this purpose. Mathematically, the reconstruction loss is given by:

$$\mathcal{L}_{\text{recon}} = \frac{1}{N} \sum_{i=1}^{N} \|\hat{f}_i - f_i\|^2 \tag{1}$$

where:

- f_i is the ground truth flow field,
- $\hat{f}_i$ is the predicted flow field generated by the CVAE,
- N is the total number of voxels.

Minimizing this loss encourages the decoder to reconstruct flow fields that closely match the true deformations of the myocardium.

4.3.2 KL-Divergence Loss. The KL-divergence loss ensures that the latent space learned by the encoder follows a standard normal distribution. This regularization is crucial for allowing the model to generate new flow fields during inference by sampling from the latent space. The KL-divergence between the approximate posterior and the prior (in our case a standard normal distribution) is computed as:

$$\mathcal{L}_{\text{KL}} = \frac{1}{2} \sum_{j=1}^{J} \left(\sigma_j^2 + \mu_j^2 - \log(\sigma_j^2 + \epsilon) - 1 \right) \tag{2}$$

where:

- μ_j and σ_j are the mean and standard deviation of the latent variable z for each latent dimension j,
- ϵ is a small constant added for numerical stability and equals 10^{-8}.

This loss penalizes deviations from the normal distribution, ensuring that the encoder learns a smooth latent space that can be sampled from during inference.

4.3.3 Total Loss. The total loss function for the CVAE is a weighted sum of the reconstruction loss and the KL-divergence loss:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{recon}} + \beta \cdot \mathcal{L}_{\text{KL}} \tag{3}$$

where β is a weighting factor that controls the trade-off between the reconstruction accuracy and the smoothness of the latent space. β, along with other hyperparameters, is outlined in Section 5.1.

4.3.4 Inference. During inference, instead of encoding the input flow field, a latent variable z is sampled from the prior normal distribution $p(z) = N(0, I)$. This random sample is passed through the decoder, which, conditioned on the input CT frame features, generates a new flow field:

$$z \sim \mathcal{N}(0, I), \quad \hat{f} = \text{Decoder}(z, \text{Conditioned Features}) \tag{4}$$

This allows the model to generate diverse flow fields for the same input CT frame by sampling different latent variables z, providing variability in the synthetic data.

4.4 Post-processing and Visualization

Post-processing: After generating the flow field, we apply clamping to a maximum absolute predefined value to ensure that the predicted values remain within a physically meaningful range.

Visualization Methods:

- *Conditional Deformation Visualization*: While it is common to visualize a 2D projection of the encoder's latent space distribution, our leave-one-out validation approach prevents this,

as each encoder is trained to encode only a single validation sample. Instead, we opted to visualize the generated flow fields based on variations in the conditional latent variable z. This approach enables us to observe how the CVAE captures different deformation patterns for the same input condition. See Figures 6, 7 and 8 in the appendix.

- *Animated Deformation Visualization*: The final output is a couple of frames (systole and diastole) with corresponding synthetic deformations. To make the synthetic deformation more interpretable, we interpolate between the two frames using the flow field, creating an animated sequence of 10 frames (see project page). Some specific generated pair of frames are shown in Fig. 1.

5 EXPERIMENTS

We conducted several experiments to evaluate the accuracy and realism of the generated flow fields. In this section, we describe the experimental setup, evaluation metrics, and results.

5.1 Experimental Setup

To evaluate the performance of the CVAE, we split the dataset into training and validation sets using a leave-one-out cross-validation strategy. Given the small size of the dataset, this approach allowed us to train the model on almost the entire dataset while testing its generalizability on unseen samples.

Training Procedure:

- Batch size: 16
- Number of training epochs: 2000
- Learning rate: 10^{-3}
- Optimizer: Adam
- Loss weights:
  - KL Divergence weight: 0.05
  - Reconstruction Loss weight: 0.95
- Augmentation parameters:
  - Intensity noise variance: 0.015
  - Maximum absolute shift: 10 pixels
  - Zoom range: [0.9, 1.1]
- Clip gradient value (max grad norm): 0.75

5.2 Evaluation Metrics

We evaluated the performance of our model during training using the following metric:

- **Mean Squared Error (MSE)**: During training, we calculate the MSE between the reconstructed flow fields and the ground truth flow fields. Since the generated output is a 3D flow field, this MSE can be interpreted as the **mEPE**, which measures the squared Euclidean distance between the predicted and true displacement vectors. However, it is important to note that this evaluation is only performed on the reconstructed flow fields during training, as we do not have ground truth flow fields for newly generated data during inference.

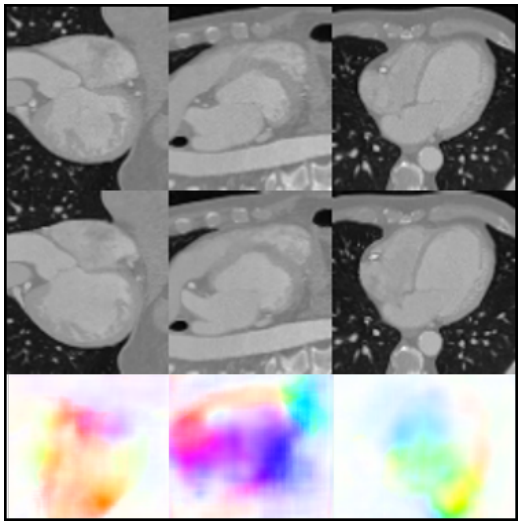

Figure 5: Qualitative Results: The top row shows three sections of the systole frame (condition). The middle row displays the generated diastole frame, created by warping the systole frame using the generated flow field. The bottom row presents the corresponding generated flow field. An animation demonstrating the gradual transformation from systole to diastole is available on our project page.

5.3 Results

Quantitative Results: The average mEPE (interpreted as the MSE) over the reconstructed flow fields in the leave-one-out cross-validation was **0.56** voxels, with a standard deviation of **0.58**. During the analysis, we observed that one sample with a relatively high mEPE had been mistakenly inputted with a different orientation than the rest. After excluding this sample from the validation, the average mEPE decreased to **0.48** voxels, with a standard deviation of **0.20**.

Qualitative Results: Figure 5 Overview of the generation process: The trained CVAE model generates a predicted flow field ($\text{Flow}_{\text{pred}}$) from a conditioned Systole frame and a random array z. This flow field is then used to warp the Systole frame, producing a Diastole frame ($\text{Diastole}_{\text{pred}}$), which, along with the flow, forms the final generated sample. For a full animated sequence, refer to the project page.

6 CONCLUSION

In this work, we proposed a method for generating synthetic 3D flow fields conditioned on single cardiac CT frames using a Conditional Variational Autoencoder. Our approach significantly reduces the need for large training dataset and enables the large-scale generation of annotated datasets for myocardium motion analysis.

Future Directions: Future work could focus on extending this approach to generate entire annotated cardiac cycles, rather than single pairs of frames. This could be achieved either through sequence-based generative models such as LSTMs, GRUs, or Transformers, which are well-suited for modeling temporal dynamics, or through a more simplified approach. In the simplified approach, the flow field generated between diastole and systole (as in this work) could

be manipulated over time in a non-linear fashion as in [7]. This time-based manipulation would adjust the rate of flow for each voxel, allowing the generated flow field to evolve dynamically. The pace of increase and decrease in flow could be determined either through statistical analysis of the dataset or via learned approaches.

Additionally, improving the augmentation pipeline and expanding the dataset size would likely enhance the generalizability and robustness of the generated flow fields.

A CODE

The code for this project is available at: https://github.com/shaharzuler/CVAE.

B ADDITIONAL SECTIONS FOR GRID SAMPLING IN LATENT SPACE

C NETWORK DETAILED ARCHITECTURE AND DIMENSIONS

REFERENCES

[1] Maria Cristina Donadio Abduch, Adriano Mesquita Alencar, Wilson Mathias Jr, and Marcelo Luiz de Campos Vieira. 2014. Cardiac mechanics evaluated by speckle tracking echocardiography. *Arquivos brasileiros de cardiologia* 102 (2014), 403–412.

[2] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Nets. In *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (Eds.), Vol. 27. Curran Associates, Inc. https://proceedings.neurips.cc/paper_files/paper/2014/file/5ca3e9b122f61f8f06494c97b1afccf3-Paper.pdf

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising Diffusion Probabilistic Models. In *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (Eds.), Vol. 33. Curran Associates, Inc., 6840–6851. https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf

[4] Mi-Young Jeung, Philippe Germain, Pierre Croisille, Soraya El ghannudi, Catherine Roy, and Afshin Gangi. 2012. Myocardial tagging with MR imaging: overview of normal and pathologic findings. *Radiographics* 32, 5 (2012), 1381–1398.

[5] Diederik P Kingma. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* (2013).

[6] Mehdi Mirza. 2014. Conditional generative adversarial nets. *arXiv preprint arXiv:1411.1784* (2014).

[7] Alex Rav-Acha, Yael Pritch, Dani Lischinski, and Shmuel Peleg. 2005. Evolving time fronts: Spatio-temporal video warping. In *Proc. 32nd Int. Conf. Comput. Graph. Interactive Tech.* 1–8.

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10684–10695.

[9] Antoine Rosset, Luca Spadola, and Osman Ratib. 2004. OsiriX: An Open-Source Software for Navigating in Multidimensional DICOM Images. *Journal of digital imaging : the official journal of the Society for Computer Applications in Radiology* 17 (10 2004), 205–16. https://doi.org/10.1007/s10278-004-1014-6

[10] Slicer Community. 2023. 3D Slicer. https://www.slicer.org/. Accessed: 2024-09-22.

[11] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems* 28 (2015).

[12] Shahar Zuler and Dan Raviv. 2024. Synthetic Data Generation for 3D Myocardium Deformation Analysis. *arXiv preprint arXiv:2406.01040* (2024).

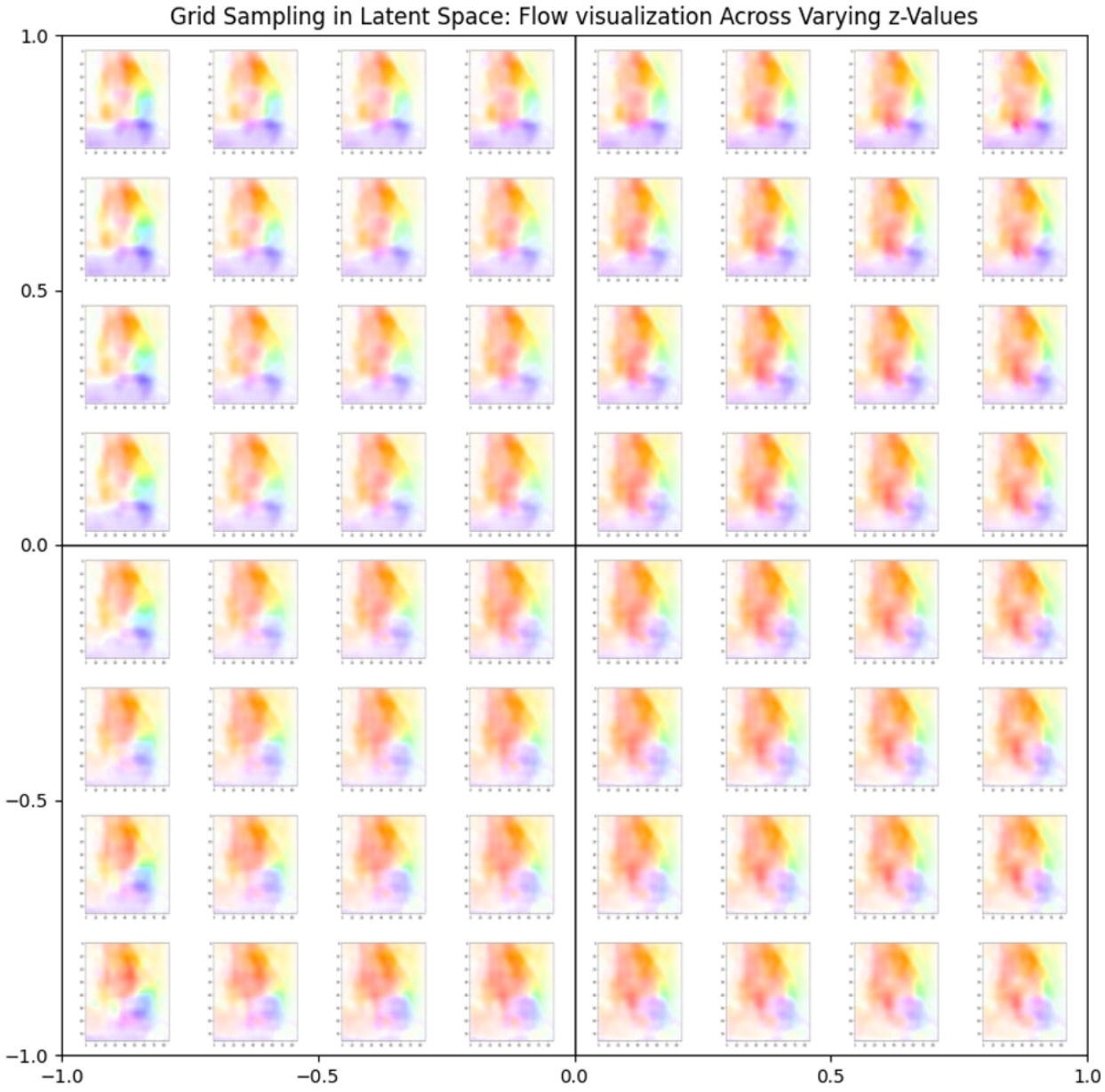

Figure 6: Grid sampling in latent space, section in the $\hat{x}$ direction: Visualizing flow fields across varying latent z values. The displayed sections are slices in the $\hat{x}$ direction. The latent variable z was varied, with latent vectors grid-sampled from the range [-1, 1] along the z latent space. This visualization allows us to inspect how the CVAE captures different deformation patterns for the same cardiac condition based on variations in the latent z direction. Animated sequences that showcase the effect of these variations on the CT frame are available in the project page.

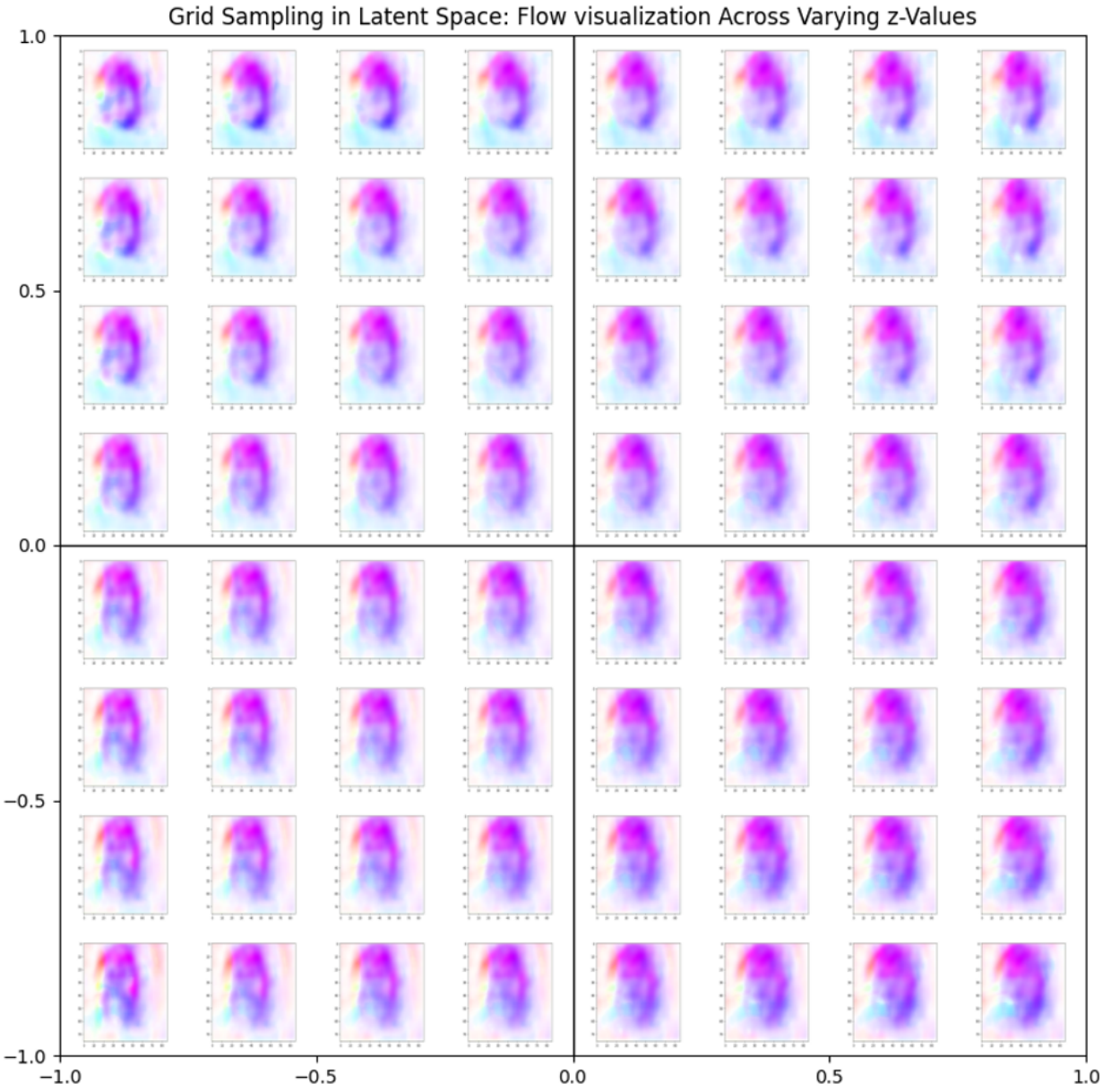

Figure 7: Grid sampling in latent space, section in the $\hat{y}$ direction: Visualizing flow fields across varying latent z values. The displayed sections are slices in the $\hat{y}$ direction. The latent variable z was varied.

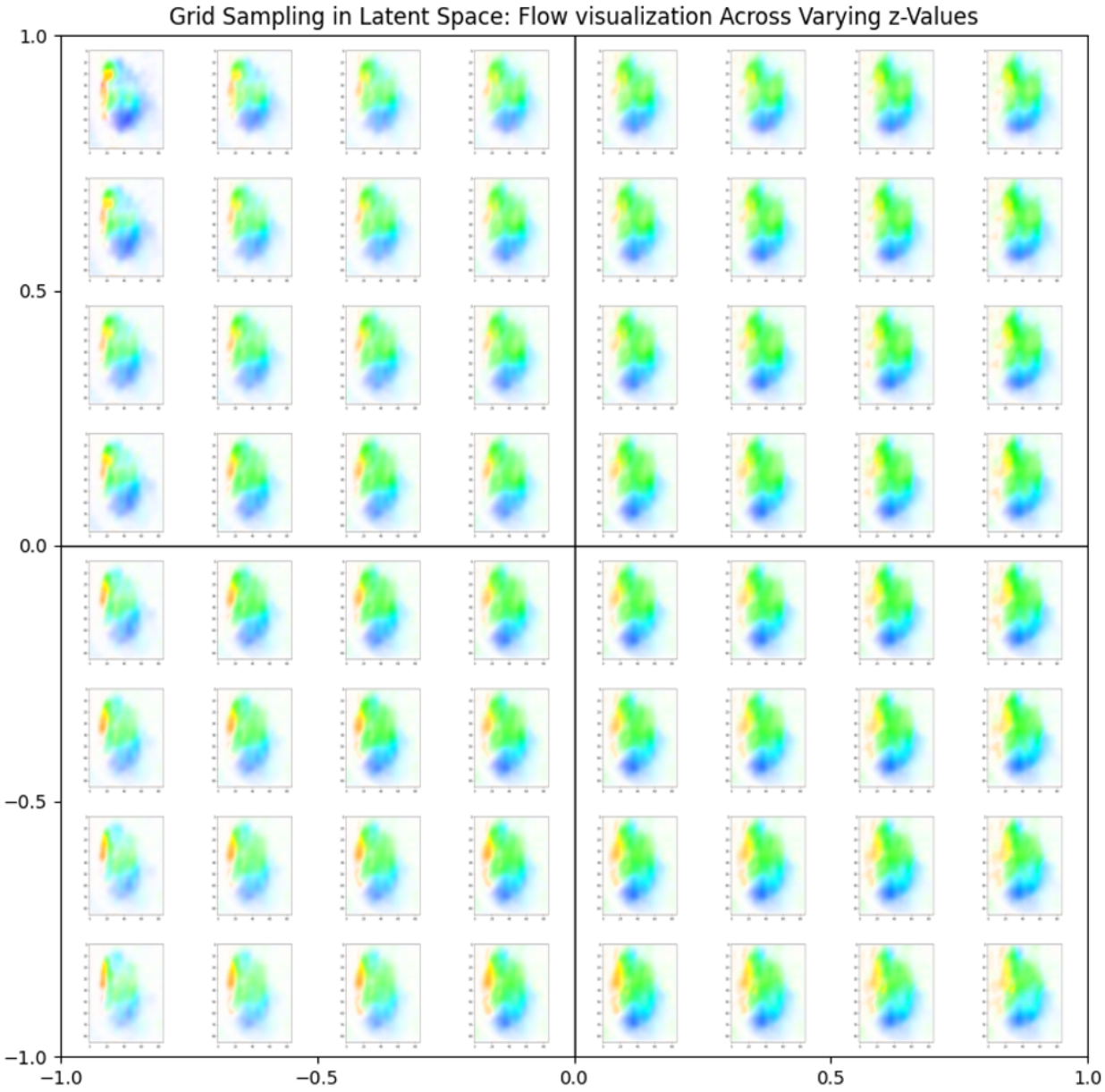

Figure 8: Grid sampling in latent space, section in the $\hat{z}$ direction: Visualizing flow fields across varying latent z values. The displayed sections are slices in the $\hat{z}$ direction. The latent variable z was varied.

Table 1: Architecture Dimensions of the CVAE Network

Layer Type	Channels In	Channels Out	Kernel Size	Stride	Activation	Input Dimensions	Output Dimensions
Encoder							
Concat	-	-	-	-	-	3,86,86,76 + 1,86,86,76	4,86,86,76
Conv3D	4	16	3	2	LeakyRelu(0.1)	4,86,86,76	16,43,43,38
Conv3D	16	16	3	1	LeakyRelu(0.1)	16,43,43,38	16,43,43,38
Concat	-	-	-	-	-	16,43,43,38+16,43,43,38	32,43,43,38
Conv3D	32	32	3	2	LeakyRelu(0.1)	32,43,43,38	32,22,22,19
Conv3D	32	32	3	1	LeakyRelu(0.1)	32,22,22,19	32,22,22,19
Concat	-	-	-	-	-	32,22,22,19+32,22,22,19	64,22,22,19
Conv3D	64	64	3	2	LeakyRelu(0.1)	64,22,22,19	64,11,11,10
Conv3D	64	64	3	1	LeakyRelu(0.1)	64,11,11,10	64,11,11,10
Concat	-	-	-	-	-	64,11,11,10+64,11,11,10	128,11,11,10
Conv3D	128	96	3	2	LeakyRelu(0.1)	128,11,11,10	96,6,6,5
Conv3D	96	96	3	1	LeakyRelu(0.1)	96,6,6,5	96,6,6,5
Concat	-	-	-	-	-	96,6,6,5+96,6,6,5	192,6,6,5
Conv3D	192	128	3	2	LeakyRelu(0.1)	192,6,6,5	128,3,3,3
Conv3D	128	128	3	1	LeakyRelu(0.1)	128,3,3,3	128,3,3,3
Concat	-	-	-	-	-	128,3,3,3+128,3,3,3	256,3,3,3
Conv3D	256	128	1	1	LeakyRelu(0.1)	256,3,3,3	128,3,3,3
Conv3D	128	128	1	1	LeakyRelu(0.1)	128,3,3,3	128,3,3,3
Conv3D - "Mean"	128	128	1	1	-	128,3,3,3	128,3,3,3
Conv3D - "Var"	128	128	1	1	-	128,3,3,3	128,3,3,3
Decoder							
Concat	-	-	-	-	-	128,3,3,3+128,3,3,3	256,3,3,3
Conv3D^{T}	256	96	3	2	LeakyRelu(0.1)	256,3,3,3	96,6,6,5
Conv3D^{T}	96	96	3	1	-	96,6,6,5	96,6,6,5
Concat	-	-	-	-	-	96,6,6,5+96,6,6,5	192,6,6,5
Conv3D^{T}	192	64	3	2	LeakyRelu(0.1)	192,6,6,5	64,11,11,10
Conv3D^{T}	64	64	3	1	-	64,11,11,10	64,11,11,10
Concat	-	-	-	-	-	64,11,11,10+64,11,11,10	128,11,11,10
Conv3D^{T}	128	32	3	2	LeakyRelu(0.1)	128,11,11,10	32,22,22,19
Conv3D^{T}	32	32	3	1	-	32,22,22,19	32,22,22,19
Concat	-	-	-	-	-	32,22,22,19+32,22,22,19	64,22,22,19
Conv3D^{T}	64	16	3	2	LeakyRelu(0.1)	64,22,22,19	16,43,43,38
Conv3D^{T}	16	16	3	1	-	16,43,43,38	16,43,43,38
Concat	-	-	-	-	-	16,43,43,38+16,43,43,38	32,43,43,38
Conv3D^{T}	32	3	3	2	LeakyRelu(0.1)	32,43,43,38	3,86,86,76
Conv3D^{T}	3	3	3	1	-	3,86,86,76	3,86,86,76
Concat	-	-	-	-	-	3,86,86,76 + 1,86,86,76	4,86,86,76
Conv3D^{T}	4	3	1	1	LeakyRelu(0.1)	4,86,86,76	3,86,86,76
Conv3D^{T}	3	3	1	1	-	3,86,86,76	3,86,86,76